

# Memory Systems and Memory-Centric Computing

## Topic 2: Memory-Centric Computing

Onur Mutlu

[omutlu@gmail.com](mailto:omutlu@gmail.com)

<https://people.inf.ethz.ch/omutlu>

16-17 July 2024

HiPEAC ACACES Summer School 2024

**SAFARI**

**ETH** zürich

Computing

is Bottlenecked by Data



# Data is Key for AI, ML, Genomics, ...

---

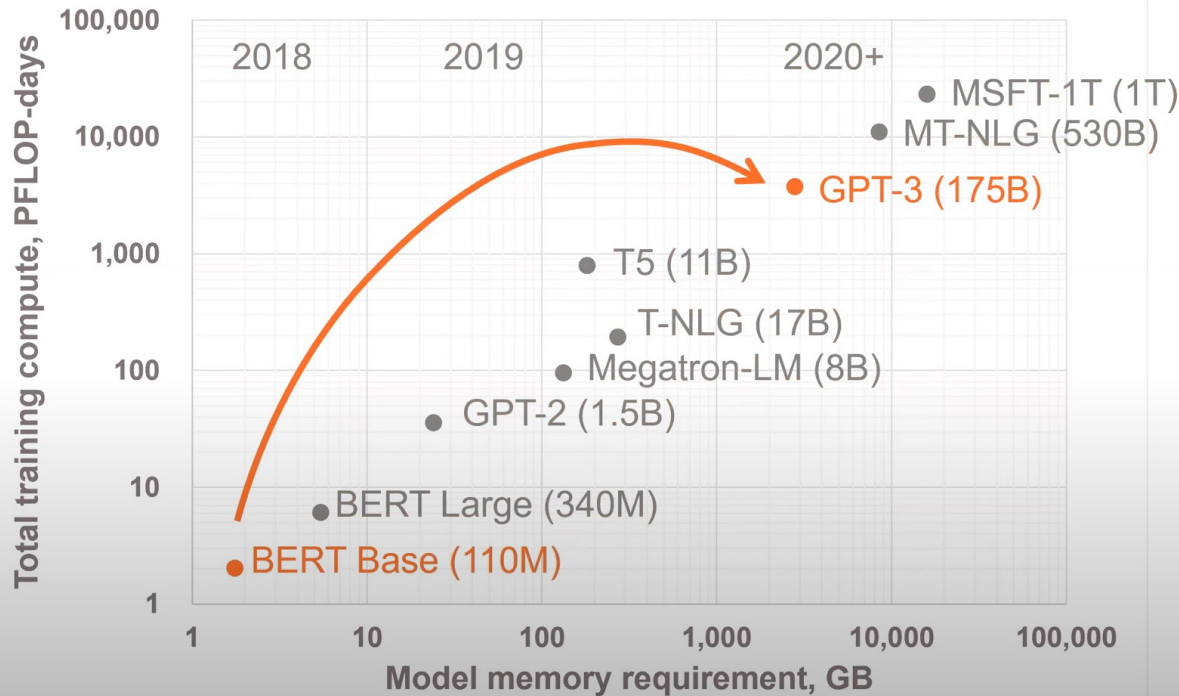
- Important workloads are all data intensive
- They require rapid and efficient processing of large amounts of data
- Data is increasing
  - We can generate more than we can process
  - We need to perform more sophisticated analyses on more data

# Huge Demand for Performance & Efficiency

## Exponential Growth of Neural Networks



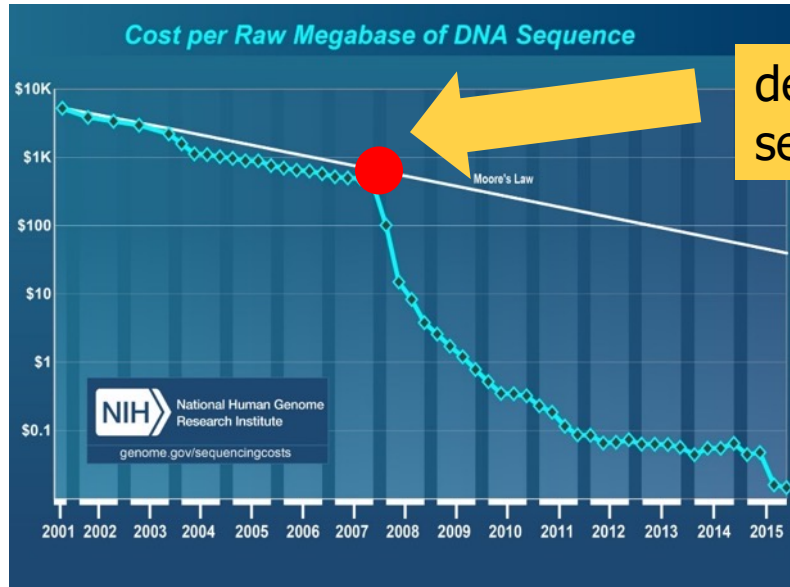
Memory and compute requirements



**1800x more compute**  
In just **2 years**

Tomorrow, **multi-trillion** parameter models

# Huge Demand for Performance & Efficiency

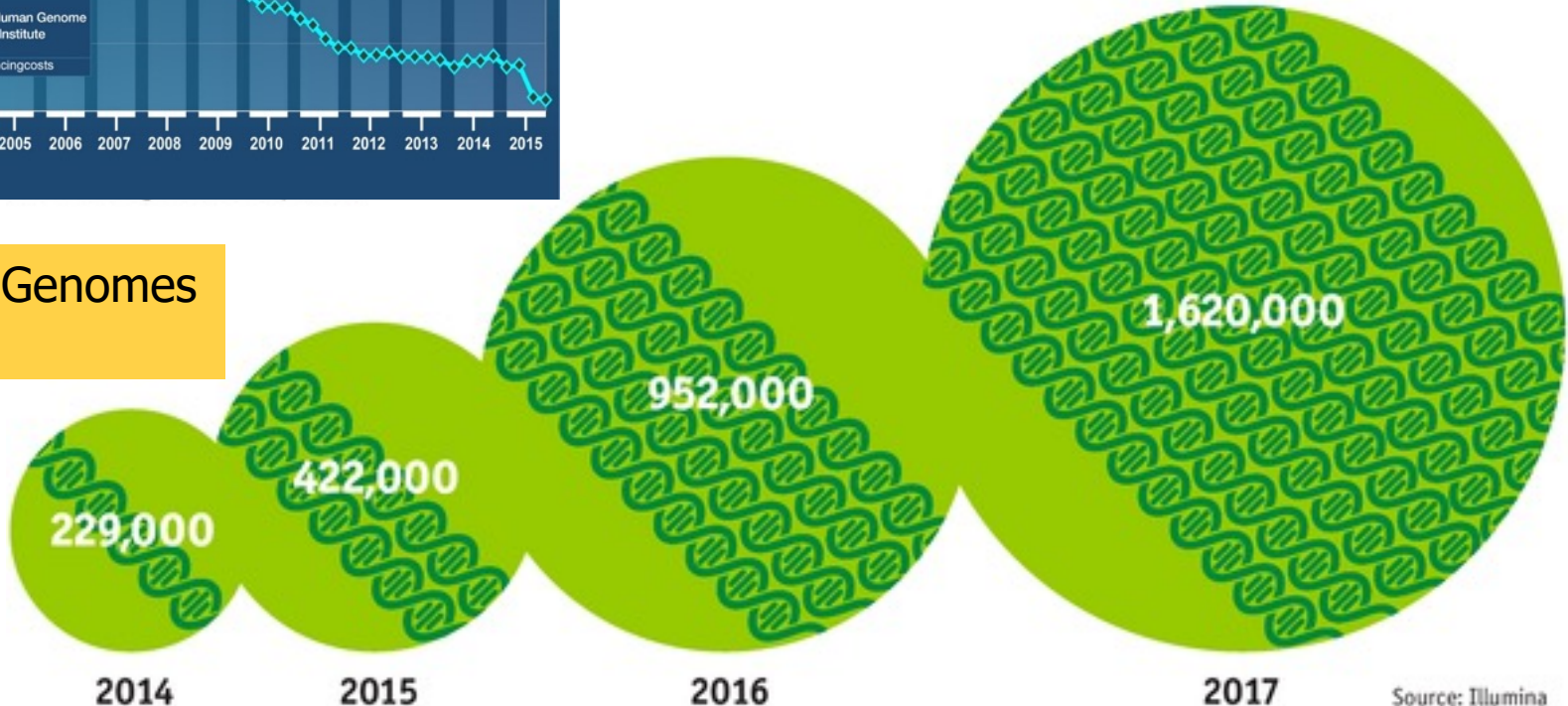


development of new sequencing technologies



Oxford Nanopore MinION

Number of Genomes Sequenced



The Economist



# Do We Want This?

---





# Or This?

---





High Performance,

Energy Efficient,

Sustainable

(All at the Same Time)

# 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

---

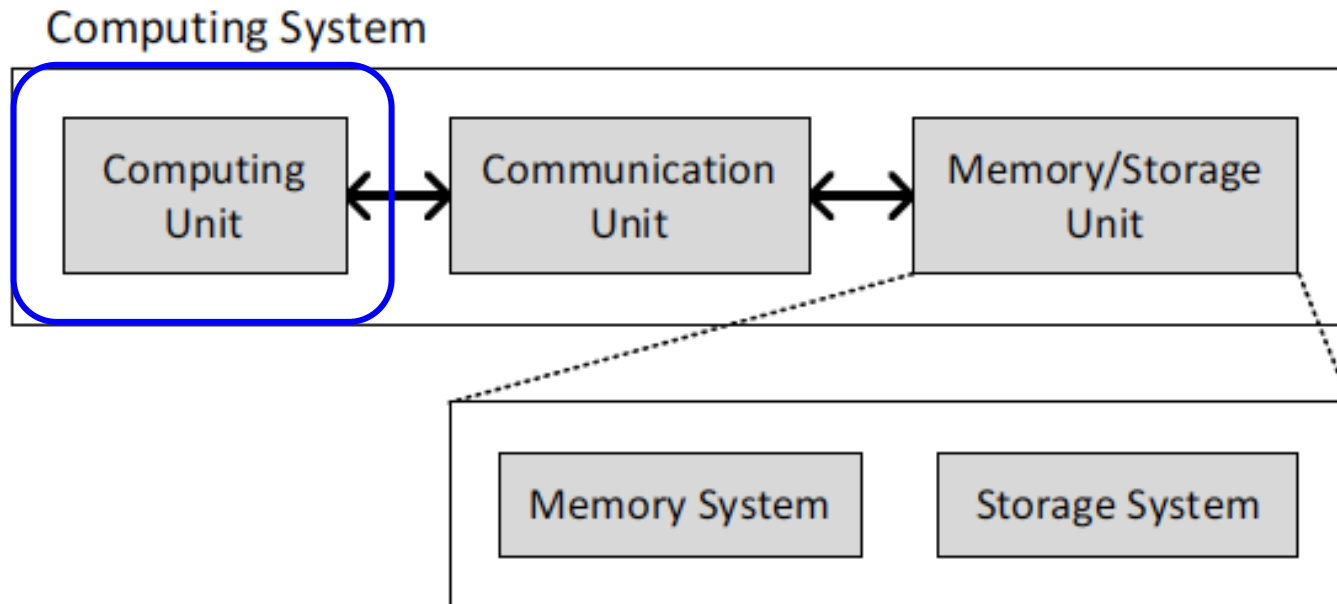
Processing of data  
is performed  
far away from the data



# Today's Computing Systems

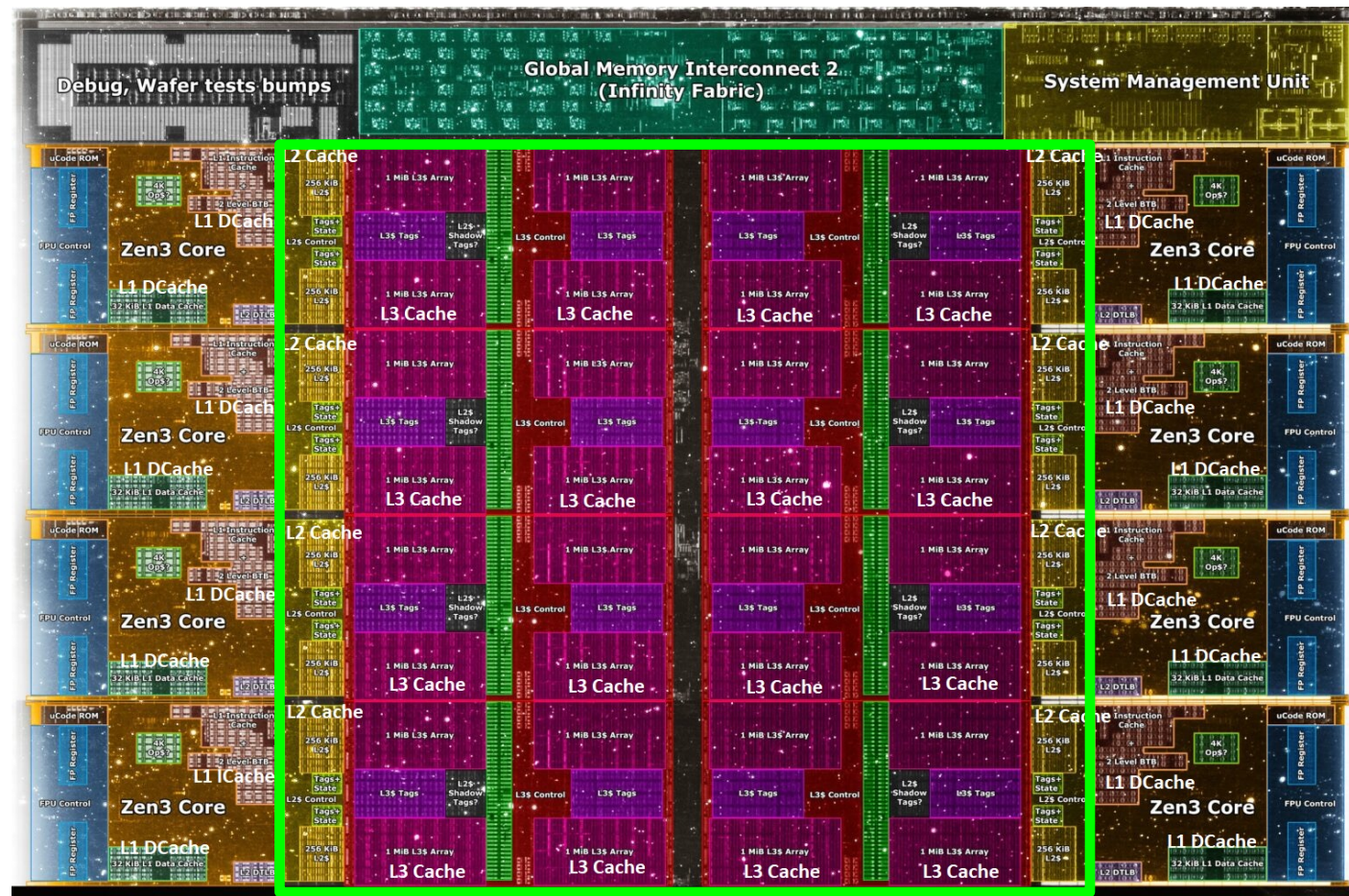
---

- Processor centric
- All data processed in the processor → at great system cost





# Deeper and Larger Memory Hierarchies



## Core Count:

8 cores/16 threads

L1 Caches:  
32 KB per core

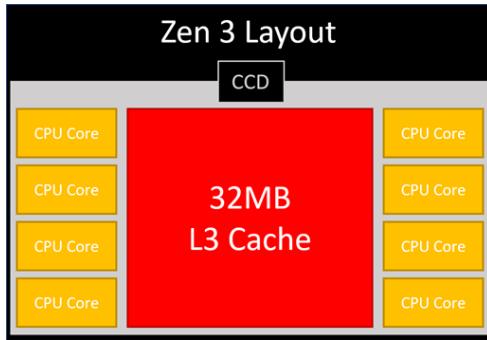
L2 Caches:  
512 KB per core

L3 Cache:  
32 MB shared

# AMD Ryzen 5000, 2020



# AMD's 3D Last Level Cache (2021)

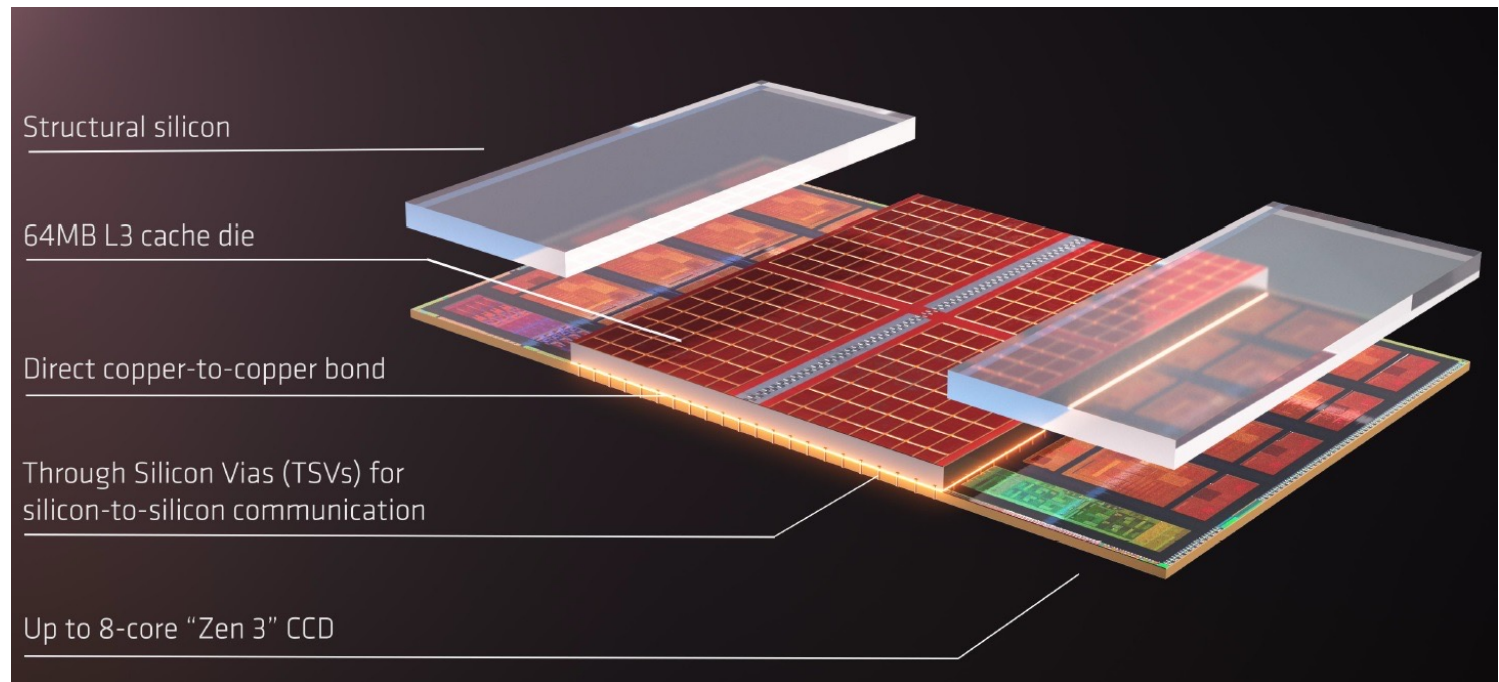


<https://community.microcenter.com/discussion/5134/comparing-zen-3-to-zen-2>

AMD increases the L3 size of their 8-core Zen 3 processors from 32 MB to 96 MB

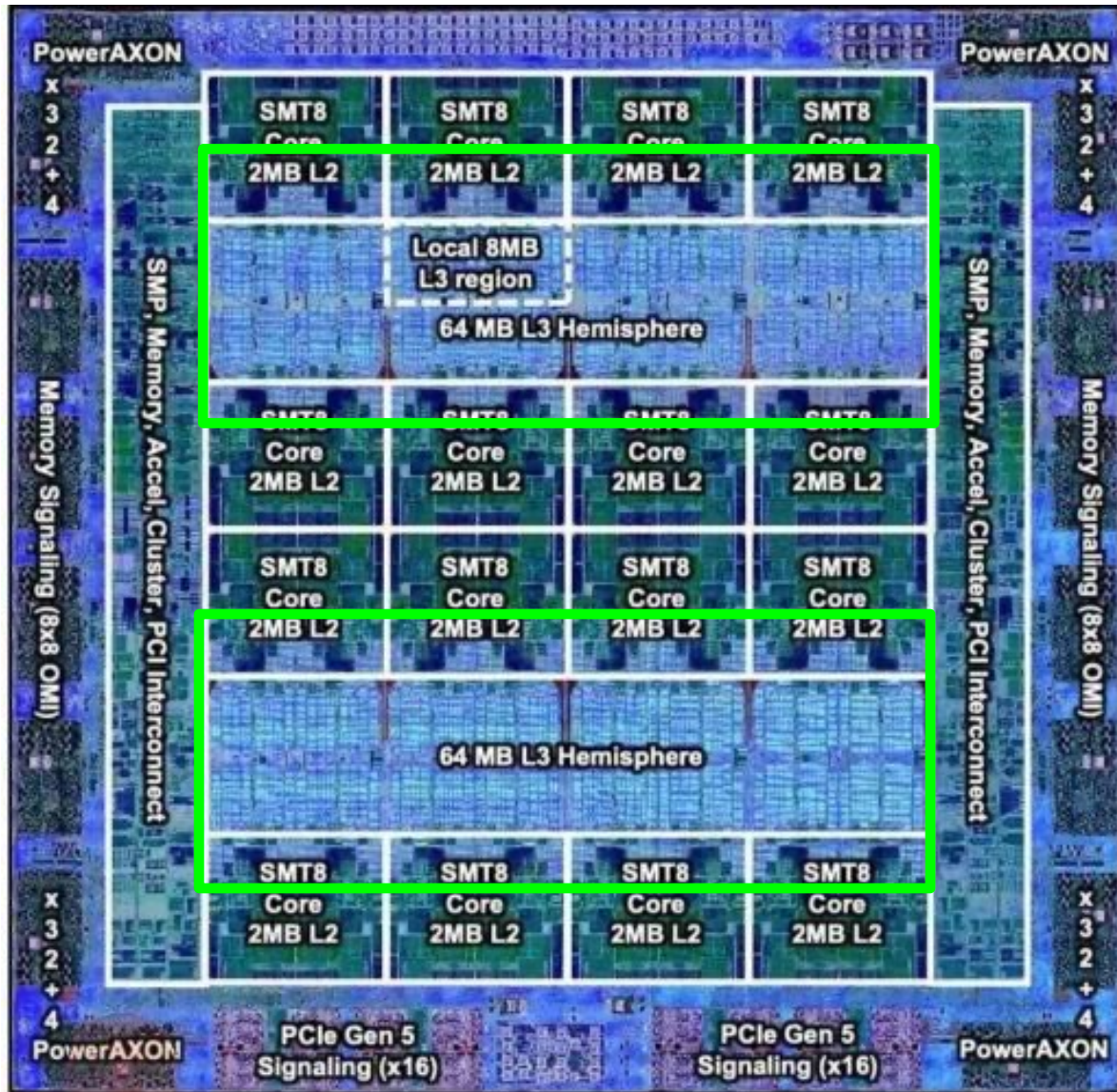
**Additional 64 MB L3 cache die**  
**stacked on top of the processor die**

- Connected using Through Silicon Vias (TSVs)
- Total of 96 MB L3 cache





# Deeper and Larger Memory Hierarchies



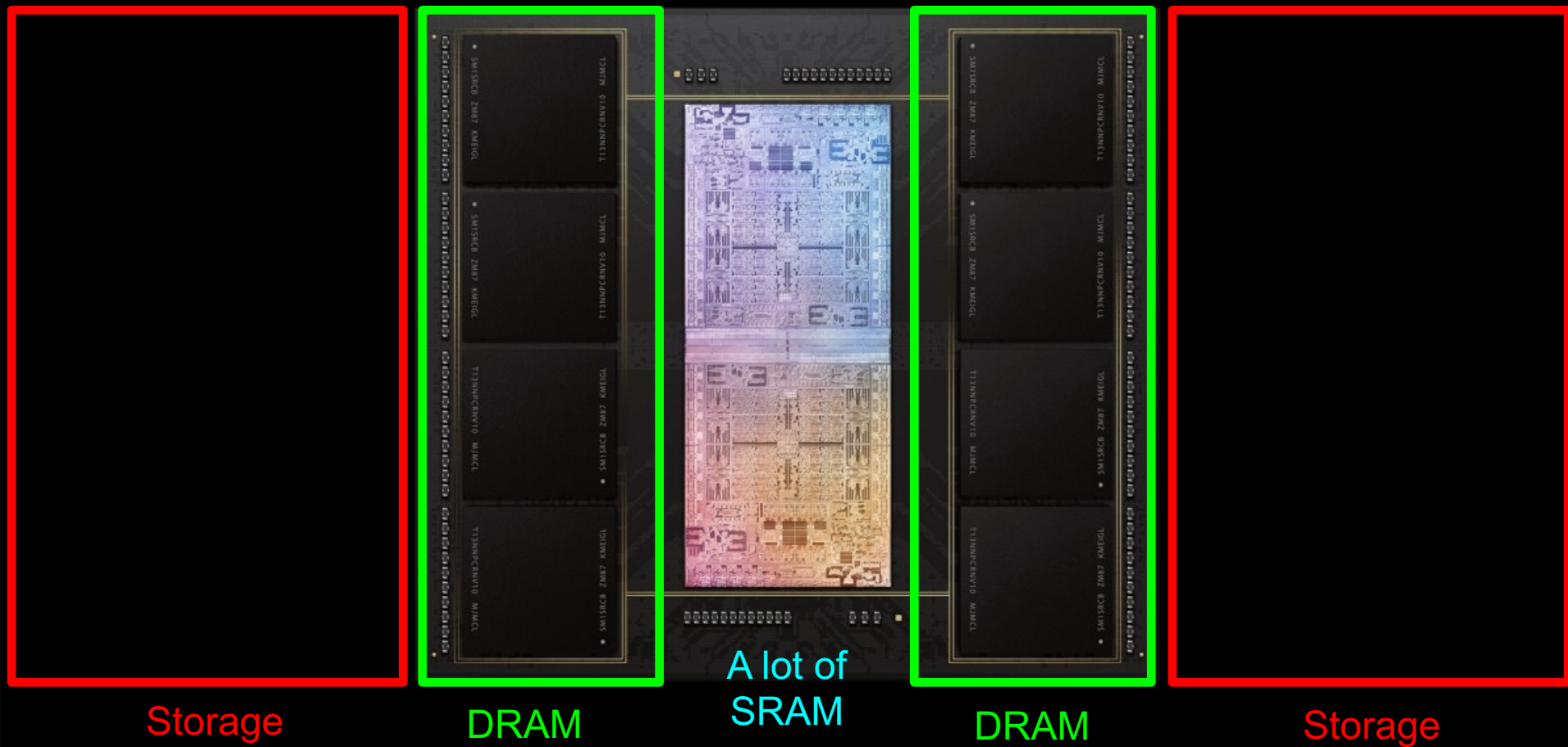
IBM POWER10,  
2020

Cores:  
15-16 cores,  
8 threads/core

L2 Caches:  
2 MB per core

L3 Cache:  
120 MB shared

# Deeper and Larger Memory Hierarchies



## Apple M1 Ultra System (2022)

# Data Overwhelms Modern Machines ...

---

- Storage/memory capability
- Communication capability
- Computation capability
- Greatly impacts robustness, energy, performance, cost

# It's the Memory, Stupid!

---

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

**RICHARD SITES**

## **It's the Memory, Stupid!**

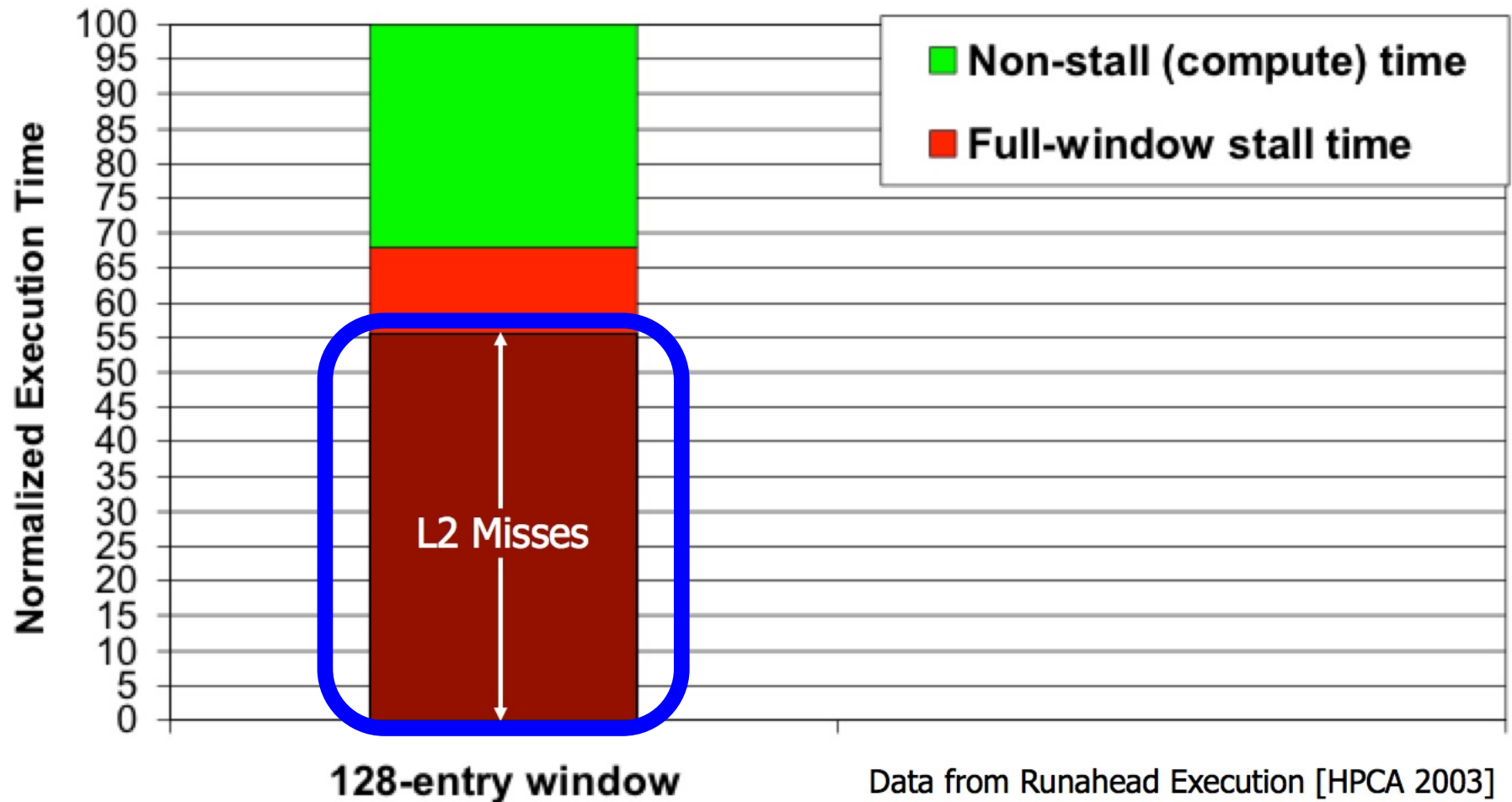
When we started the Alpha architecture design in 1988, we estimated a 25-year lifetime and a relatively modest 32% per year compounded performance improvement of implementations over that lifetime (1,000× total). We guestimated about 10× would come from CPU clock improvement, 10× from multiple instruction issue, and 10× from multiple processors.

5, 1996  MICROPROCESSOR REPORT

I expect that over the coming decade memory subsystem design will be the *only* important design issue for microprocessors.



# The Performance Perspective



# The Performance Perspective

---

- Onur Mutlu, Jared Stark, Chris Wilkerson, and Yale N. Patt,  
**"Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-order Processors"**  
*Proceedings of the 9th International Symposium on High-Performance Computer Architecture (HPCA)*, pages 129-140, Anaheim, CA, February 2003. [Slides \(pdf\)](#)  
***One of the 15 computer arch. papers of 2003 selected as Top Picks by IEEE Micro. HPCA Test of Time Award (awarded in 2021).***

## Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-order Processors

Onur Mutlu §    Jared Stark †    Chris Wilkerson ‡    Yale N. Patt §

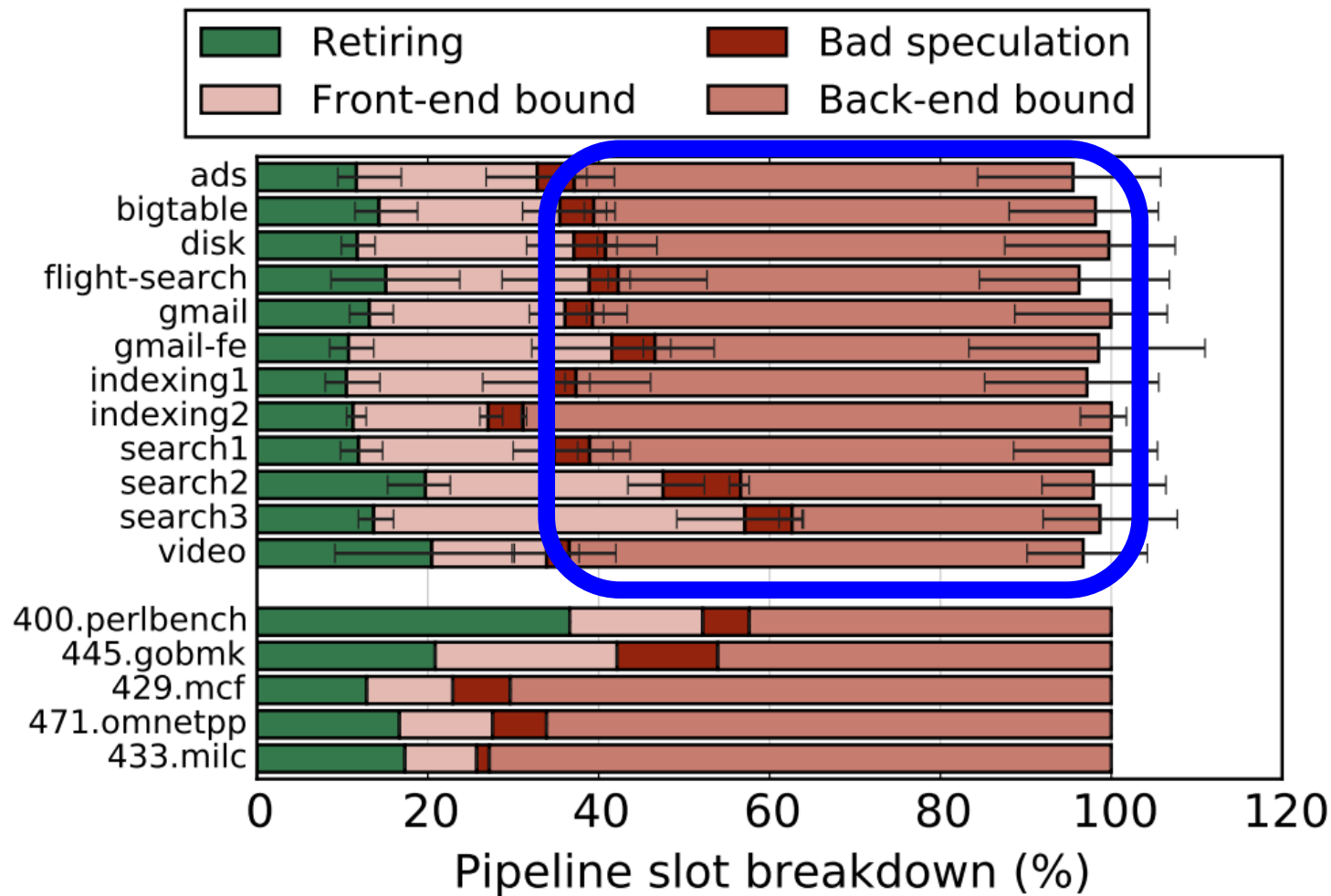
§ECE Department  
The University of Texas at Austin  
{onur,patt}@ece.utexas.edu

†Microprocessor Research  
Intel Labs  
jared.w.stark@intel.com

‡Desktop Platforms Group  
Intel Corporation  
chris.wilkerson@intel.com

# The Performance Perspective (Today)

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



# 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

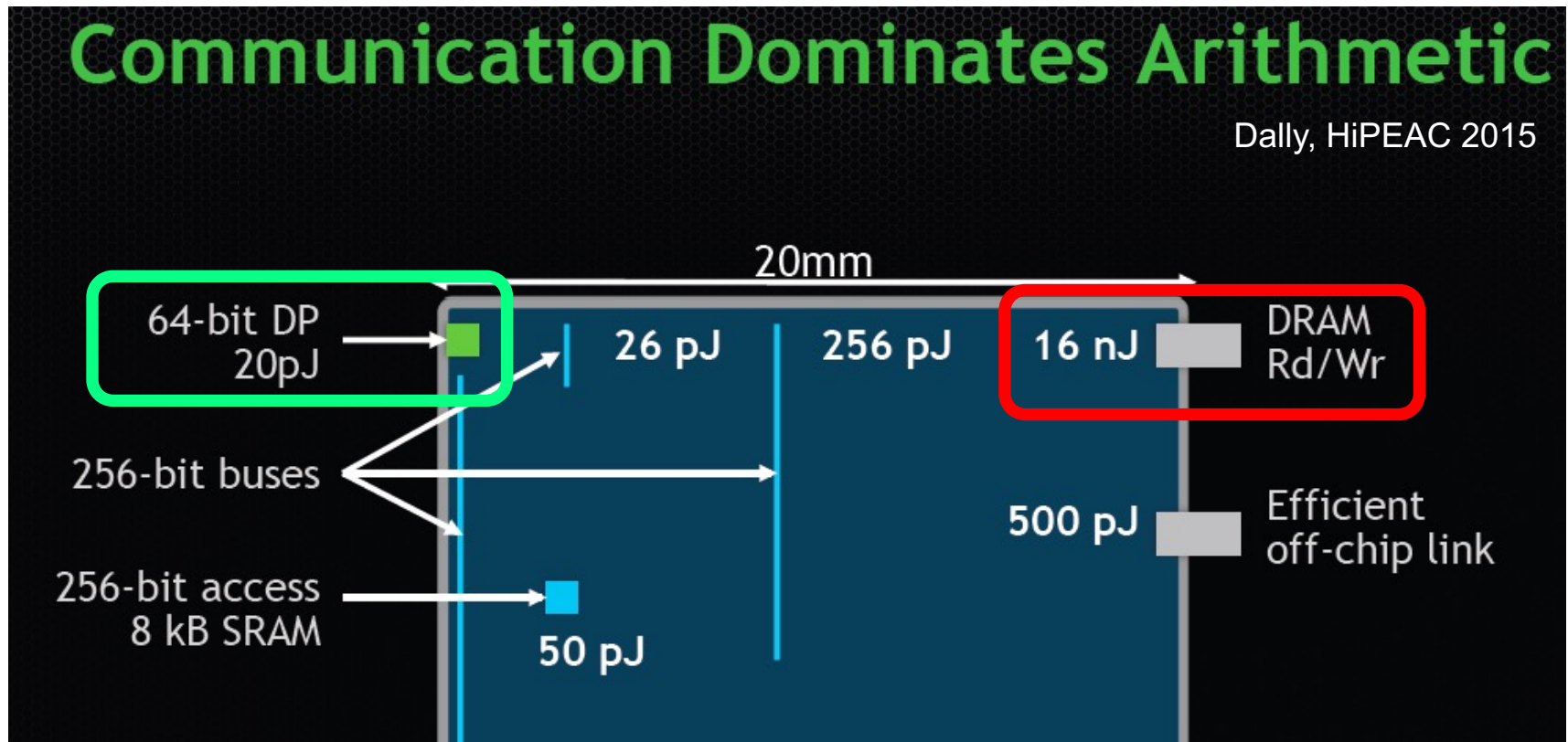
- ▣ Especially true for off-chip to on-chip movement



# Data Movement vs. Computation Energy

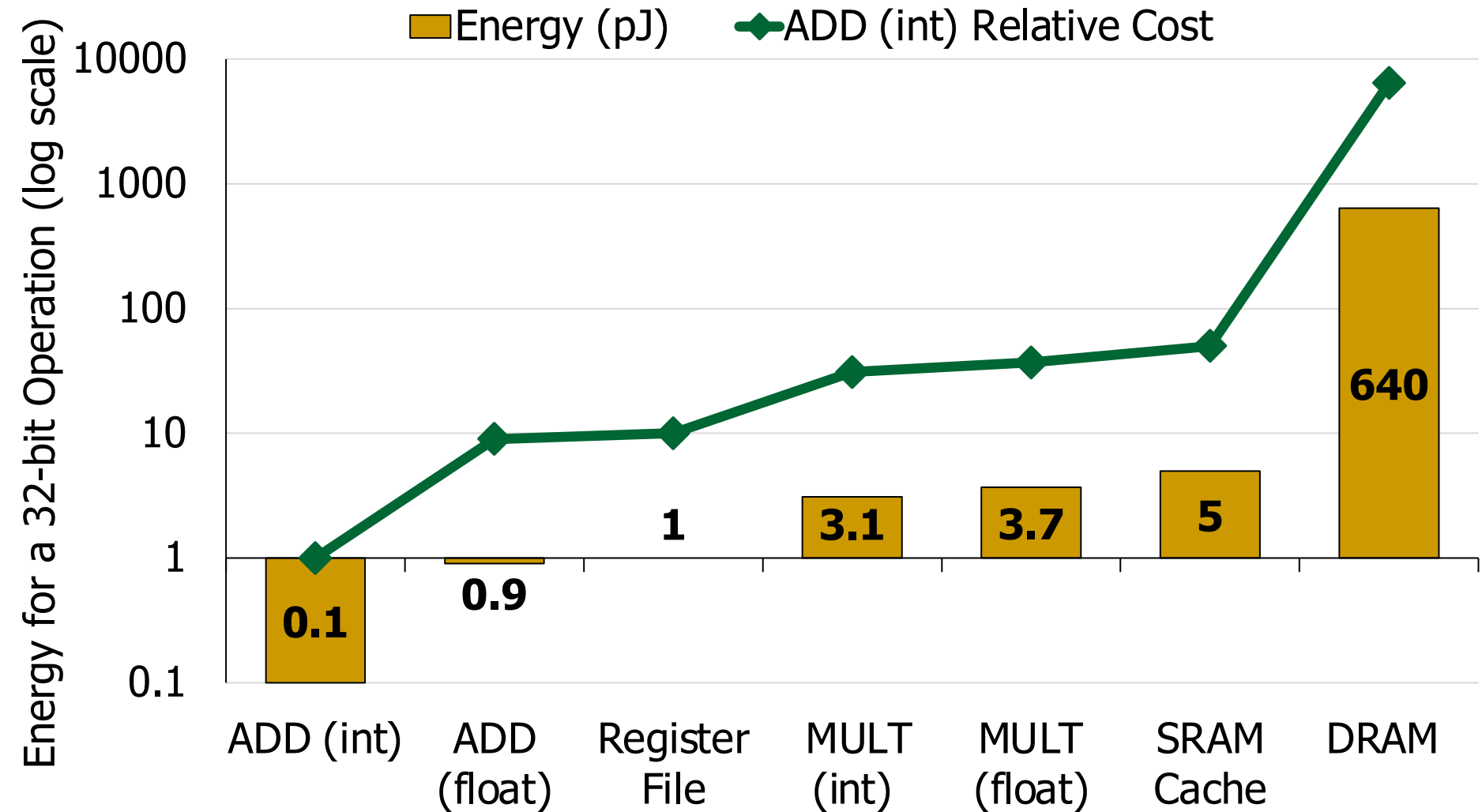
## Communication Dominates Arithmetic

Dally, HiPEAC 2015

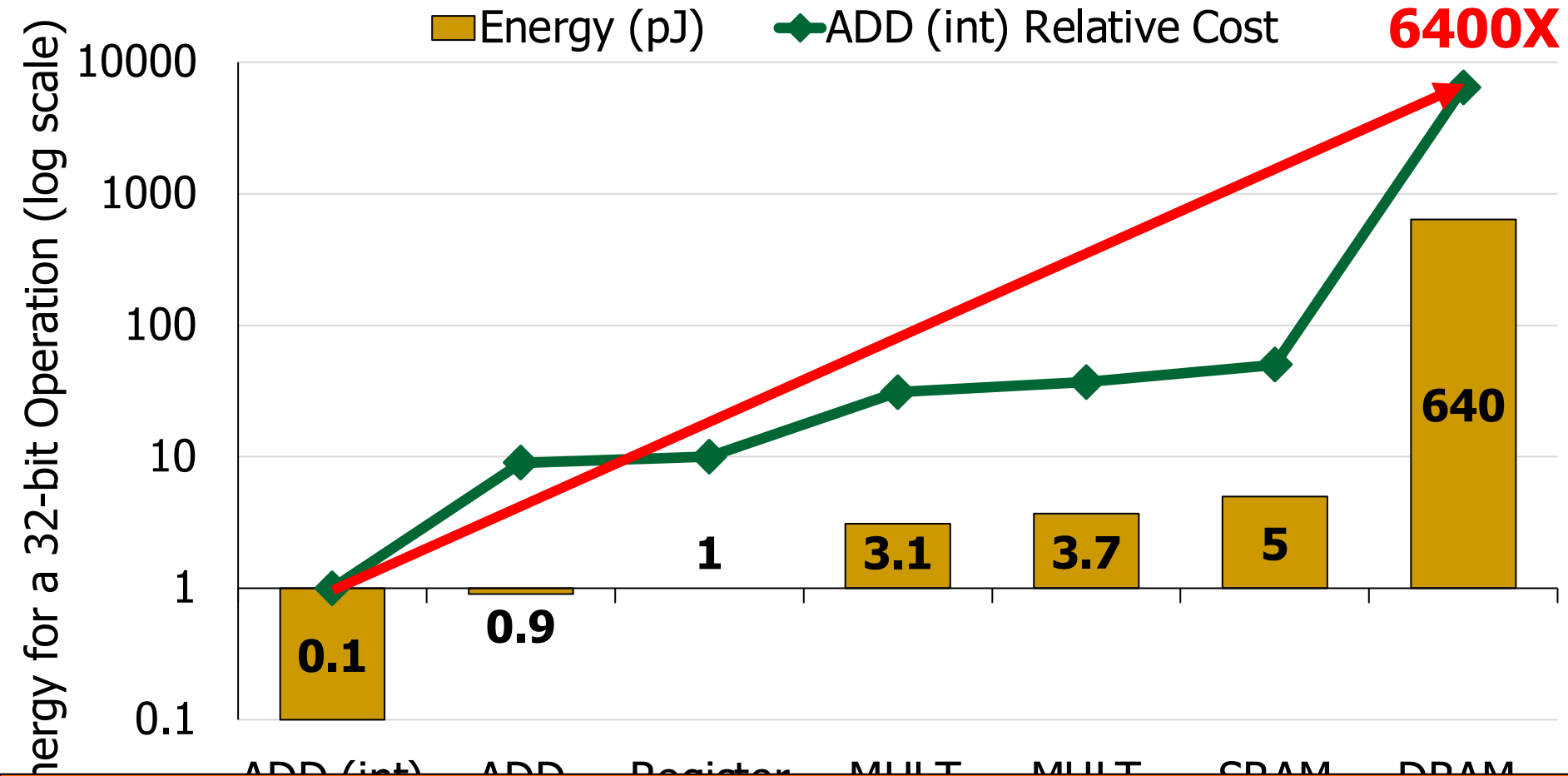


A memory access consumes  $\sim 100\text{-}1000\times$  the energy of a complex addition

# Data Movement vs. Computation Energy



# Data Movement vs. Computation Energy



A memory access consumes 6400X the energy of a simple integer addition

# 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<sup>1</sup>

Saugata Ghose<sup>1</sup>

Youngsok Kim<sup>2</sup>

Rachata Ausavarungnirun<sup>1</sup>

Eric Shiu<sup>3</sup>

Rahul Thakur<sup>3</sup>

Daehyun Kim<sup>4,3</sup>

Aki Kuusela<sup>3</sup>

Allan Knies<sup>3</sup>

Parthasarathy Ranganathan<sup>3</sup>

Onur Mutlu<sup>5,1</sup>



# Energy Waste in Accelerators

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,  
["Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"](#)  
*Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Virtual, September 2021.  
[[Slides \(pptx\)](#)] ([pdf](#))  
[[Talk Video](#) (14 minutes)]

**> 90% of the total system energy  
is spent on **memory** in large ML models**

## Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand<sup>†◇</sup>  
Geraldo F. Oliveira<sup>\*</sup>

Saugata Ghose<sup>‡</sup>  
Xiaoyu Ma<sup>§</sup>

Berkin Akin<sup>§</sup>  
Eric Shiu<sup>§</sup>

Ravi Narayanaswami<sup>§</sup>  
Onur Mutlu<sup>\*†</sup>

<sup>†</sup>Carnegie Mellon Univ.

<sup>◇</sup>Stanford Univ.

<sup>‡</sup>Univ. of Illinois Urbana-Champaign

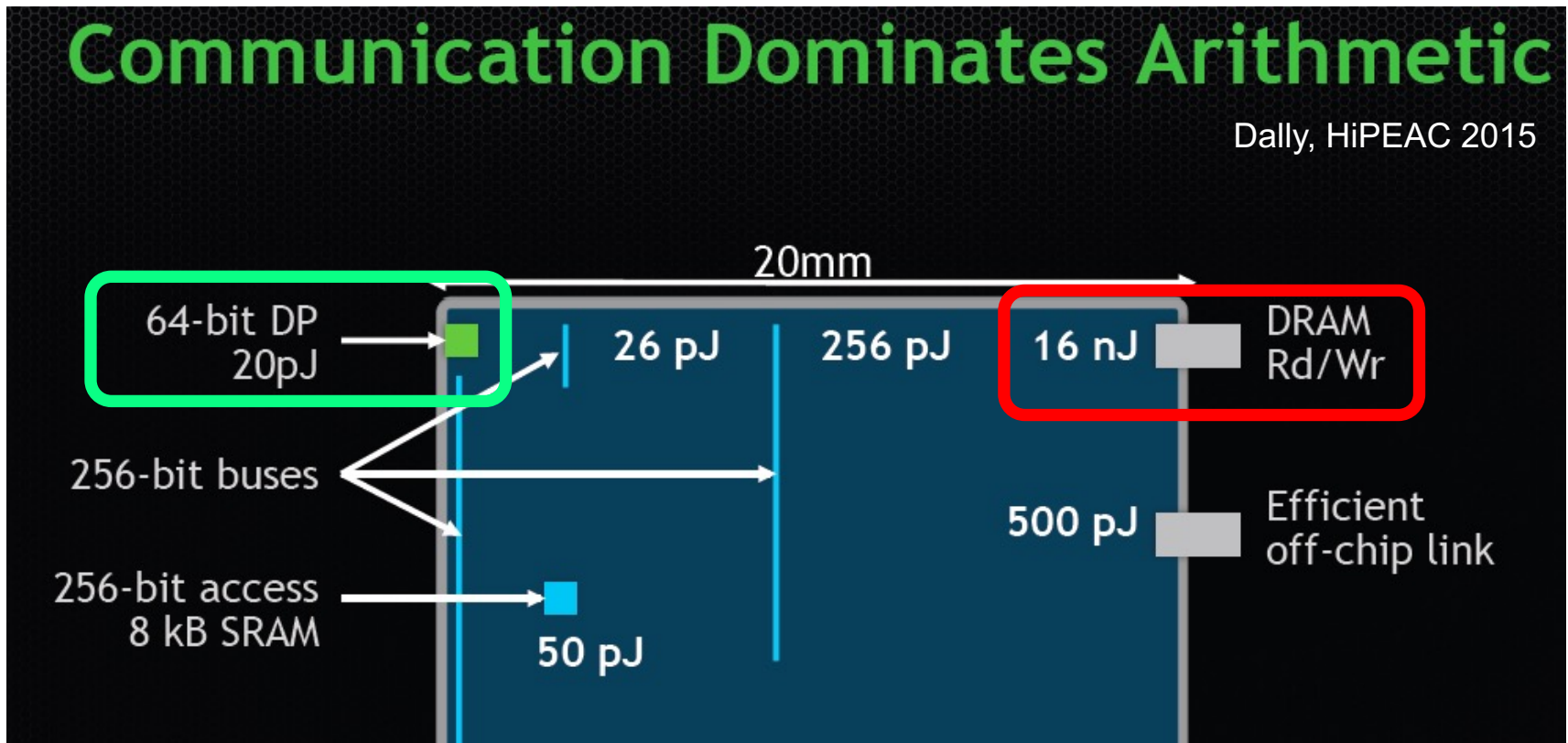
<sup>§</sup>Google

<sup>\*</sup>ETH Zürich

# We Do Not Want to Move Data!

## Communication Dominates Arithmetic

Dally, HiPEAC 2015



A memory access consumes  $\sim 100-1000X$  the energy of a complex addition

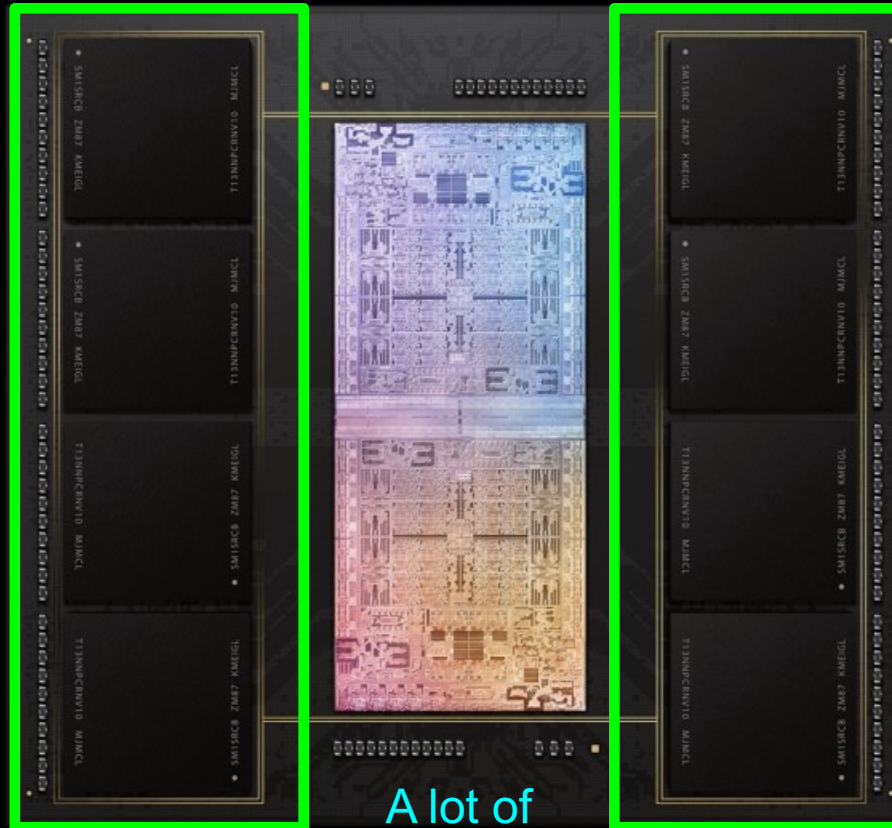
# We Need A Paradigm Shift To ...

---

- Enable computation with minimal data movement
- Compute where it makes sense (where data resides)
- Make computing architectures more data-centric

# Process Data Where It Makes Sense

Sensors



A lot of  
SRAM

DRAM

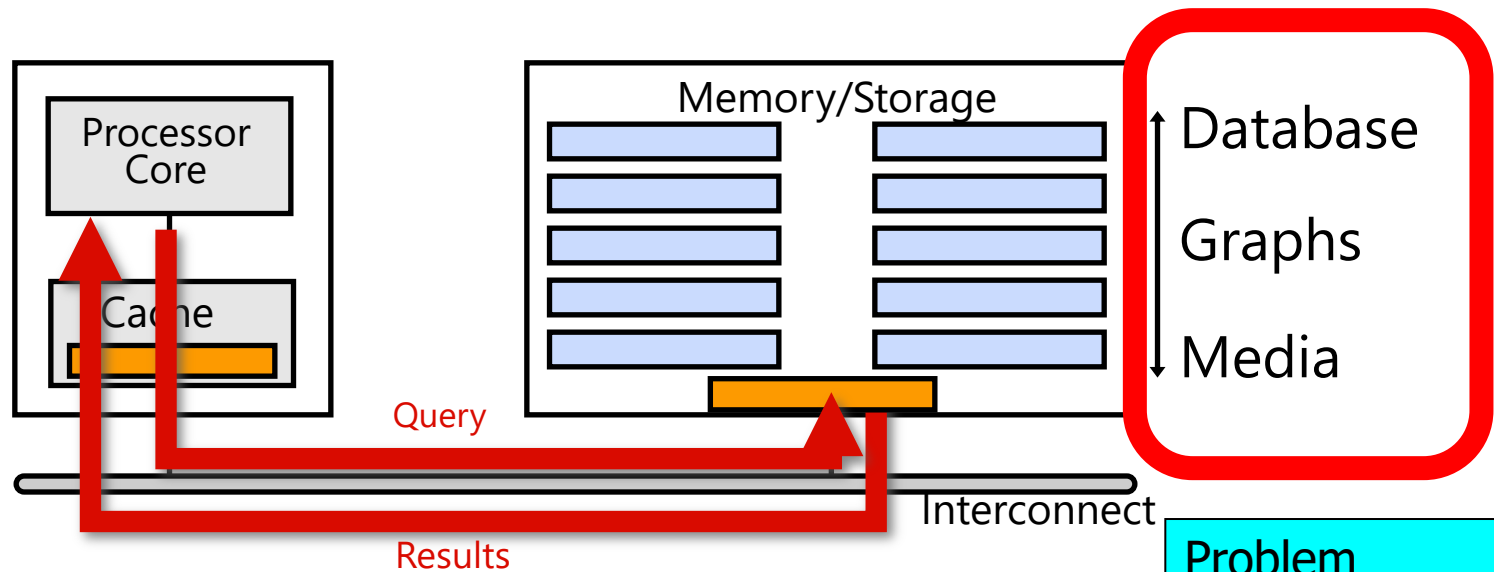
DRAM

Storage

Storage

Apple M1 Ultra System (2022)

# Goal: Processing Inside Memory/Storage



- Many questions ... How do we design the:
  - ❑ compute-capable memory & controllers?
  - ❑ processors & communication units?
  - ❑ software & hardware interfaces?
  - ❑ system software, compilers, languages?
  - ❑ algorithms & theoretical foundations?

|                    |
|--------------------|
| Problem            |
| Algorithm          |
| Program/Language   |
| System Software    |
| SW/HW Interface    |
| Micro-architecture |
| Logic              |
| Devices            |
| Electrons          |



# Processing in/near Memory: An Old Idea

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

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

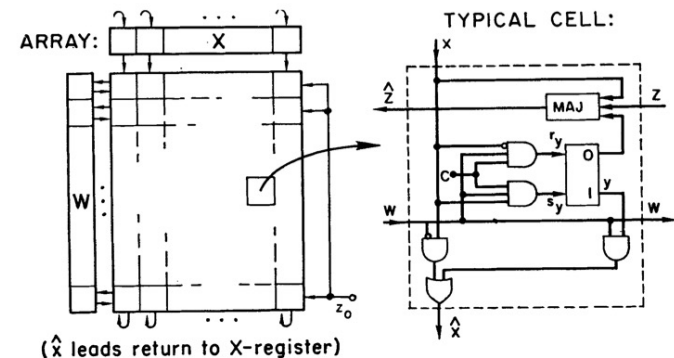
## Cellular Logic-in-Memory Arrays

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.



$$\begin{aligned}\hat{x} &= \bar{w}x + wy \\ s_y &= wcx, r_y = wc\bar{x} \\ \hat{z} &= M(x, \bar{y}, z) = x\bar{y} + z(x + \bar{y})\end{aligned}$$

Fig. 1. Cellular sorting array I.

# 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.

# Why In-Memory Computation Today?

---

- **Huge demand from Applications & Systems**
  - ❑ Data access bottleneck
  - ❑ Energy & power bottlenecks
  - ❑ Data movement energy dominates computation energy
  - ❑ Need all at the same time: performance, energy, sustainability
  - ❑ We can improve all metrics by minimizing data movement
- **Huge problems with Memory Technology**
  - ❑ Memory technology scaling is not going well (e.g., RowHammer)
  - ❑ Many scaling issues demand intelligence in memory
- **Designs are squeezed in the middle**

One can  
predictably induce errors  
in most DRAM memory chips

Kim+, "[Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors](#)," ISCA 2014.



Rowhammer

---



# RowHammer [ISCA 2014]

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu,  
**"Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"**  
*Proceedings of the 41st International Symposium on Computer Architecture (ISCA)*, Minneapolis, MN, June 2014.  
[[Slides \(pptx\) \(pdf\)](#)] [[Lightning Session Slides \(pptx\) \(pdf\)](#)] [[Source Code and Data](#)] [[Lecture Video](#) (1 hr 49 mins), 25 September 2020]  
***One of the 7 papers of 2012-2017 selected as Top Picks in Hardware and Embedded Security for IEEE TCAD ([link](#)). Selected to the ISCA-50 25-Year Retrospective Issue covering 1996-2020 in 2023 ([Retrospective \(pdf\)](#) [Full Issue](#)). Winner of the 2024 IFIP Jean-Claude Laprie Award in dependable computing ([link](#)).***

## Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim<sup>1</sup>   Ross Daly\*   Jeremie Kim<sup>1</sup>   Chris Fallin\*   Ji Hye Lee<sup>1</sup>  
Donghyuk Lee<sup>1</sup>   Chris Wilkerson<sup>2</sup>   Konrad Lai   Onur Mutlu<sup>1</sup>

<sup>1</sup>Carnegie Mellon University   <sup>2</sup>Intel Labs

## Main Memory Needs Intelligent Controllers

# Industry's Intelligent DRAM Controllers (I)

## ISSCC 2023 / SESSION 28 / HIGH-DENSITY MEMORIES /

### **28.8 A 1.1V 16Gb DDR5 DRAM with Probabilistic-Aggressor Tracking, Refresh-Management Functionality, Per-Row Hammer Tracking, a Multi-Step Precharge, and Core-Bias Modulation for Security and Reliability Enhancement**

Woongrae Kim, Chulmoon Jung, Seongnyuh Yoo, Duckhwa Hong, Jeongjin Hwang, Jungmin Yoon, Ohyong Jung, Joonwoo Choi, Sanga Hyun, Mankeun Kang, Sangho Lee, Dohong Kim, Sanghyun Ku, Donhyun Choi, Nogeun Joo, Sangwoo Yoon, Junseok Noh, Byeongyong Go, Cheolhoe Kim, Sunil Hwang, Mihyun Hwang, Seol-Min Yi, Hyungmin Kim, Sanghyuk Heo, Yeonsu Jang, Kyoungchul Jang, Shinho Chu, Yoonna Oh, Kwidong Kim, Junghyun Kim, Soohwan Kim, Jeongtae Hwang, Sangil Park, Junphyo Lee, Inchul Jeong, Joohwan Cho, Jonghwan Kim

SK hynix Semiconductor, Icheon, Korea



# Industry's Intelligent DRAM Controllers (II)

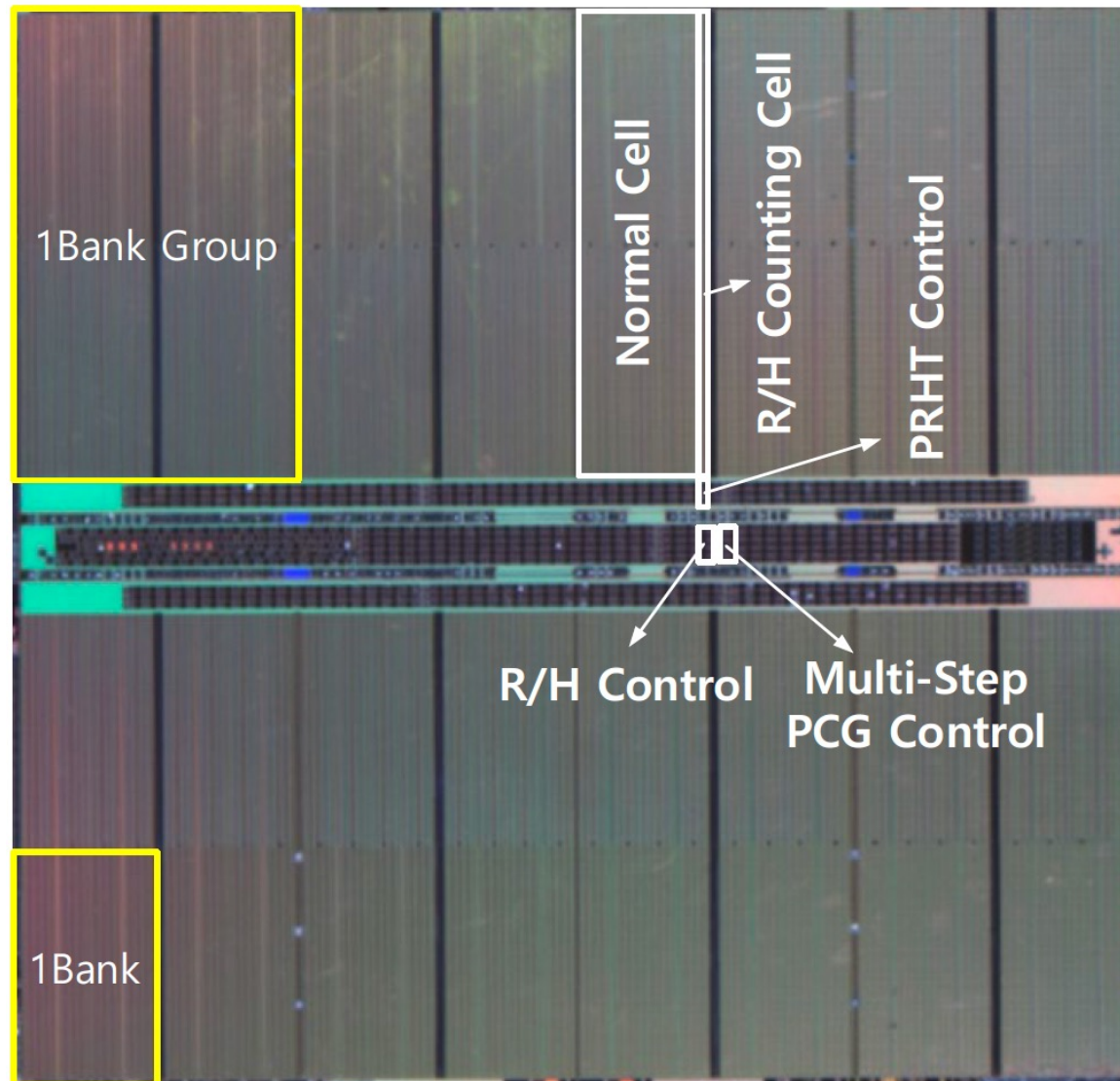
---

SK hynix Semiconductor, Icheon, Korea

DRAM products have been recently adopted in a wide range of high-performance computing applications: such as in cloud computing, in big data systems, and IoT devices. This demand creates larger memory capacity requirements, thereby requiring aggressive DRAM technology node scaling to reduce the cost per bit [1,2]. However, DRAM manufacturers are facing technology scaling challenges due to row hammer and refresh retention time beyond 1a-nm [2]. Row hammer is a failure mechanism, where repeatedly activating a DRAM row disturbs data in adjacent rows. Scaling down severely threatens reliability since a reduction of DRAM cell size leads to a reduction in the intrinsic row hammer tolerance [2,3]. To improve row hammer tolerance, there is a need to probabilistically activate adjacent rows with carefully sampled active addresses and to improve intrinsic row hammer tolerance [2]. In this paper, row-hammer-protection and refresh-management schemes are presented to guarantee DRAM security and reliability despite the aggressive scaling from 1a-nm to sub 10-nm nodes. The probabilistic-aggressor-tracking scheme with a refresh-management function (RFM) and per-row hammer tracking (PRHT) improve DRAM resilience. A multi-step precharge reinforces intrinsic row-hammer tolerance and a core-bias modulation improves retention time: even in the face of cell-transistor degradation due to technology scaling. This comprehensive scheme leads to a reduced probability of failure, due to row hammer attacks, by 93.1% and an improvement in retention time by 17%.



# Industry's Intelligent DRAM Controllers (III)



ISSCC 2023 / SESSION 28 / HIGH-DENSITY MEMORIES

**28.8 A 1.1V 16Gb DDR5 DRAM with Probabilistic-Aggressor Tracking, Refresh-Management Functionality, Per-Row Hammer Tracking, a Multi-Step Precharge, and Core-Bias Modulation for Security and Reliability Enhancement**

Woongrae Kim, Chulmoon Jung, Seongnyuh Yoo, Duckhwa Hong, Jeongjin Hwang, Jungmin Yoon, Ohyoung Jung, Joonwoo Choi, Sanga Hyun, Mankeun Kang, Sangho Lee, Dohong Kim, Sanghyun Ku, Donhyun Choi, Nogeun Joo, Sangwoo Yoon, Junseok Noh, Byeongyong Go, Cheolhoe Kim, Sunil Hwang, Mihyun Hwang, Seol-Min Yi, Hyungmin Kim, Sanghyuk Heo, Yeonsu Jang, Kyoungchul Jang, Shinho Chu, Yoonna Oh, Kwidong Kim, Junghyun Kim, Soohwan Kim, Jeongtae Hwang, Sangil Park, Junphyo Lee, Inchul Jeong, Joohwan Cho, Jonghwan Kim

SK hynix Semiconductor, Icheon, Korea



# Solutions in JEDEC (2024)



Global Standards for the Microelectronics Industry

STANDARDS & DOCUMENTS

COMMITTEES

NEWS

EVENTS & MEETINGS

JOIN

**DDR5 SDRAM**

JESD79-5C

Apr 2024

Release Number: Version 1.30

Version 1.30

This standard defines the DDR5 SDRAM specification, including features, functionalities, AC and DC characteristics, packages, and ball/signal assignments. The purpose of this Standard is to define the minimum set of requirements for JEDEC compliant 8 Gb through 32 Gb for x4, x8, and x16 DDR5 SDRAM devices. This standard was created based on the DDR4 standards (JESD79-4) and some aspects of the DDR, DDR2, DDR3, and LPDDR4 standards (JESD79, JESD79-2, JESD79-3, and JESD209-4).

Committee(s): [JC-42](#), [JC-42.3](#)

# A RowHammer Survey Across the Stack

---

- Onur Mutlu and Jeremie Kim,  
[\*\*"RowHammer: A Retrospective"\*\*](#)  
*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) Special Issue on Top Picks in Hardware and Embedded Security*, 2019.  
[[Preliminary arXiv version](#)]  
[[Slides from COSADE 2019 \(pptx\)](#)]  
[[Slides from VLSI-SOC 2020 \(pptx\) \(pdf\)](#)]  
[[Talk Video](#) (1 hr 15 minutes, with Q&A)]

## RowHammer: A Retrospective

Onur Mutlu<sup>§‡</sup>      Jeremie S. Kim<sup>‡§</sup>  
§ETH Zürich      ‡Carnegie Mellon University

# A RowHammer Survey: Recent Update

---

- Onur Mutlu, Ataberk Olgun, and A. Giray Yaglikci,  
**"Fundamentally Understanding and Solving RowHammer"**  
*Invited Special Session Paper at the 28th Asia and South Pacific Design Automation Conference (ASP-DAC), Tokyo, Japan, January 2023.*  
[[arXiv version](#)]  
[[Slides \(pptx\)](#) ([pdf](#))]  
[[Talk Video](#) (26 minutes)]

## Fundamentally Understanding and Solving RowHammer

Onur Mutlu  
onur.mutlu@safari.ethz.ch  
ETH Zürich  
Zürich, Switzerland

Ataberk Olgun  
ataberk.olgund@safari.ethz.ch  
ETH Zürich  
Zürich, Switzerland

A. Giray Yağlıkçı  
giray.yaglikci@safari.ethz.ch  
ETH Zürich  
Zürich, Switzerland

<https://arxiv.org/pdf/2211.07613.pdf>

---



- Haocong Luo, Ataberk Olgun, Giray Yaglikci, Yahya Can Tugrul, Steve Rhyner, M. Banu Cavlak, Joel Lindegger, Mohammad Sadrosadati, and Onur Mutlu, **"RowPress: Amplifying Read Disturbance in Modern DRAM Chips"**

*Proceedings of the 50th International Symposium on Computer Architecture (ISCA), Orlando, FL, USA, June 2023.*

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Video](#) (3 minutes)]

[[RowPress Source Code and Datasets \(Officially Artifact Evaluated with All Badges\)](#)]

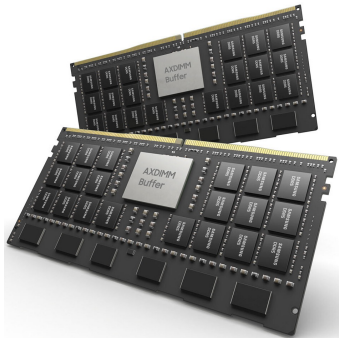
***Officially artifact evaluated as available, reusable and reproducible.  
Best artifact award at ISCA 2023. IEEE Micro Top Pick in 2024.***

## RowPress: Amplifying Read-Disturbance in Modern DRAM Chips

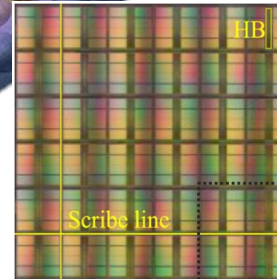
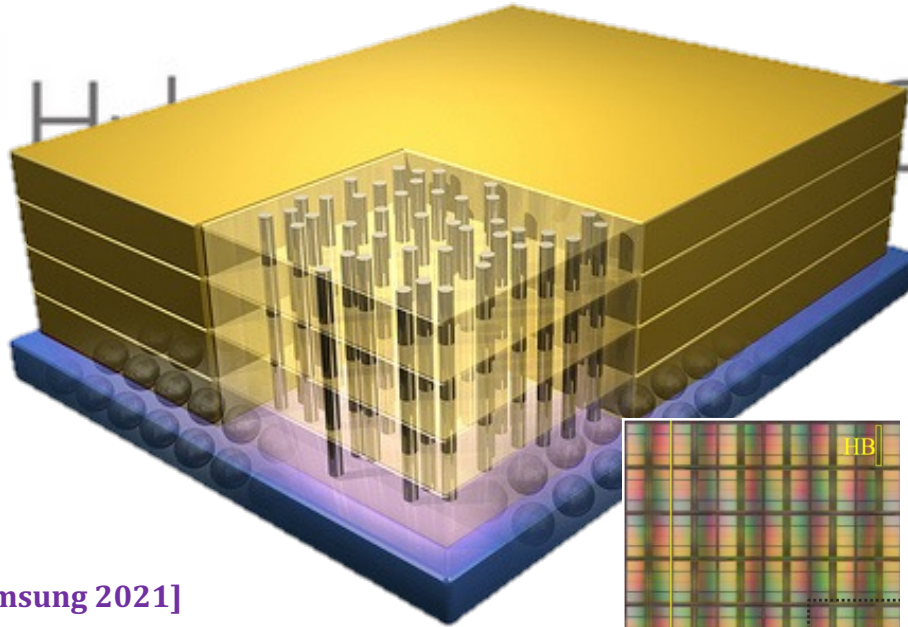
Haocong Luo   Ataberk Olgun   A. Giray Yağlıkçı   Yahya Can Tuğrul   Steve Rhyner  
Meryem Banu Cavlak   Joël Lindegger   Mohammad Sadrosadati   Onur Mutlu

ETH Zürich

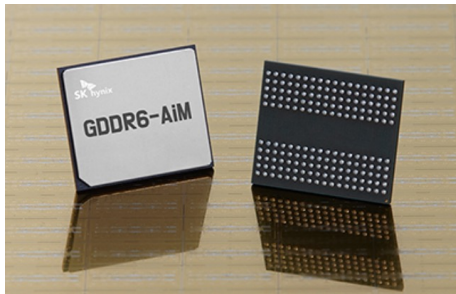
# Processing-in-Memory Landscape Today



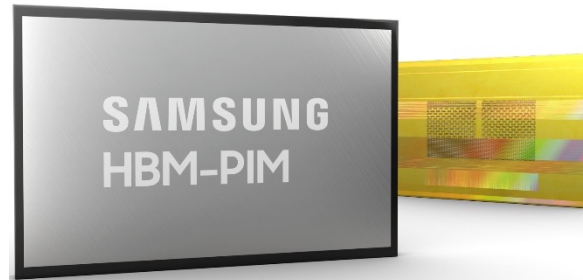
[Samsung 2021]



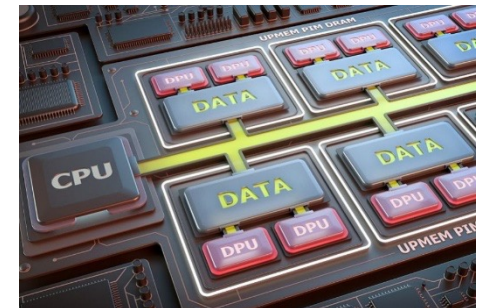
[Alibaba 2022]



[SK Hynix 2022]



[Samsung 2021]



[UPMEM 2019]



# Processing-in-Memory Landscape Today

IEEE COMPUTER ARCHITECTURE LETTERS, VOL. 22, NO. 1, JANUARY-JUNE

## Computational CXL-Memory Solution for Accelerating Memory-Intensive Applications

Joonseop Sim<sup>ID</sup>, Soohong Ahn<sup>ID</sup>, Taeyoung Ahn<sup>ID</sup>,  
Seungyong Lee<sup>ID</sup>, Myunghyun Rhee, Jooyoung Kim<sup>ID</sup>,  
Kwangsik Shin, Donguk Moon<sup>ID</sup>,  
Euseok Kim, and Kyoung Park<sup>ID</sup>

**Abstract**—CXL interface is the up-to-date technology that enables effective memory expansion by providing a memory-sharing protocol in configuring heterogeneous devices. However, its limited physical bandwidth can be a significant bottleneck for emerging data-intensive applications. In this work, we propose a novel CXL-based memory disaggregation architecture with a real-world prototype demonstration, which overcomes the bandwidth limitation of the CXL interface using near-data processing. The experimental results demonstrate that our design achieves up to  $1.9\times$  better performance/power efficiency than the existing CPU system.

**Index Terms**—Compute express link (CXL), near-data-processing (NDP)

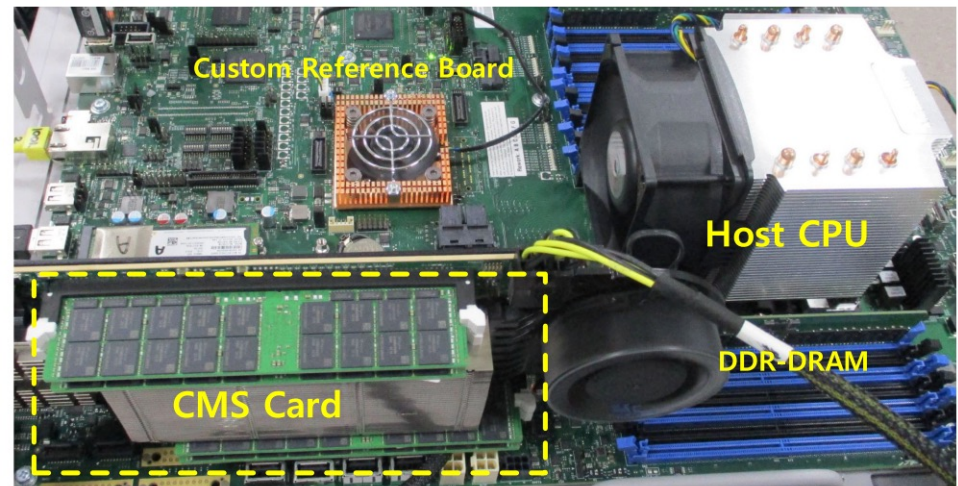


Fig. 6. FPGA prototype of proposed CMS card.

# Processing-in-Memory Landscape Today

## Samsung Processing in Memory Technology at Hot Chips 2023

By Patrick Kennedy - August 28, 2023



Samsung PIM PNM For Transformer Based AI HC35\_Page\_24

# PIM Review and Open Problems

---

## A Modern Primer on Processing in Memory

Onur Mutlu<sup>a,b</sup>, Saugata Ghose<sup>b,c</sup>, Juan Gómez-Luna<sup>a</sup>, Rachata Ausavarungnirun<sup>d</sup>

*SAFARI Research Group*

<sup>a</sup>*ETH Zürich*

<sup>b</sup>*Carnegie Mellon University*

<sup>c</sup>*University of Illinois at Urbana-Champaign*

<sup>d</sup>*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.*

# PIM Course (Fall 2022)

## ■ Fall 2022 Edition:

- [https://safari.ethz.ch/projects\\_and\\_seminars/fall2022/doku.php?id=processing\\_in\\_memory](https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=processing_in_memory)

## ■ Spring 2022 Edition:

- [https://safari.ethz.ch/projects\\_and\\_seminars/spring2022/doku.php?id=processing\\_in\\_memory](https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=processing_in_memory)

## ■ Youtube Livestream (Fall 2022):

- <https://www.youtube.com/watch?v=QLL0wQ9I4Dw&list=PL5Q2soXY2Zi8KzG2CQYRNQOVD0GOBrnKy>

## ■ Youtube Livestream (Spring 2022):

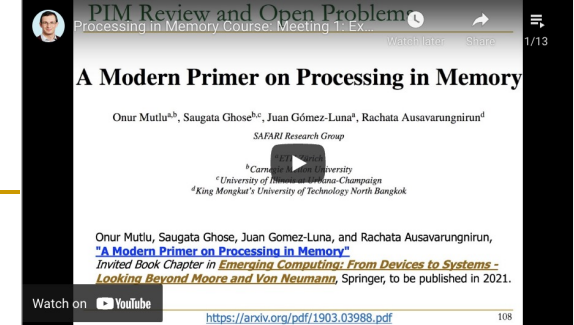
- <https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYUK9EsXKhQKRPyX>

## ■ 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>

**SAFARI**



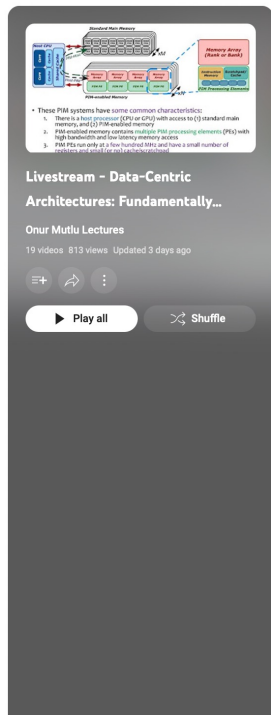
### Spring 2022 Meetings/Schedule

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



# Processing-in-Memory Course (Spring 2023)

- Short weekly lectures
- Hands-on projects



- PIM Course: Lecture 1: Data-Centric Architectures: Improving Performance & Energy (Spring 2023)**  
Onur Mutlu Lectures • 1.1K views • Streamed 3 months ago
- PIM Course: Lecture 2: How to Evaluate Data Movement Bottlenecks (Spring 2023)**  
Onur Mutlu Lectures • 332 views • 2 months ago
- ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads**  
Onur Mutlu Lectures • 1.5K views • Streamed 2 months ago
- PIM Course: Lecture 3: Real-world PIM: UPMEM PIM (Spring 2023)**  
Onur Mutlu Lectures • 411 views • 2 months ago
- PIM Course: Lecture 4: Real-world PIM: Microbenchmarking of UPMEM PIM (Spring 2023)**  
Onur Mutlu Lectures • 188 views • 2 months ago
- Análisis Experimental de una Arquitectura PIM - Juan Gómez Luna - Lecture in Spanish @ U. de Córdoba**  
Onur Mutlu Lectures • 169 views • 2 months ago
- PIM Course: Lecture 5: Real-world PIM: Samsung HBM-PIM (Spring 2023)**  
Onur Mutlu Lectures • 483 views • 2 months ago
- PIM Course: Lecture 6: Real-world PIM: SK Hynix AIM (Spring 2023)**  
Onur Mutlu Lectures • 573 views • 1 month ago
- PIM Course: Lecture 7: Real-world PIM: Samsung AxDIMM (Spring 2023)**  
Onur Mutlu Lectures • 325 views • 1 month ago

[https://www.youtube.com/playlist?list=PL5Q2soXY2zi\\_EObuoAZVSq\\_o6UySWQHvz](https://www.youtube.com/playlist?list=PL5Q2soXY2zi_EObuoAZVSq_o6UySWQHvz)

**SAFARI Project & Seminars Courses**  
(Spring 2023)

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

**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.

Table of Contents

- Data-Centric Architectures: Fundamentally Improving Performance and Energy (227-0085-37L)
- Course Description
- Mentors
- Lecture Video Playlist on YouTube
- Spring 2023 Meetings/Schedule
- Past Lecture Video Playlists on YouTube
- Learning Materials
- Assignments

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

[https://safari.ethz.ch/projects\\_and\\_seminars/spring2023/doku.php?id=processing\\_in\\_memory](https://safari.ethz.ch/projects_and_seminars/spring2023/doku.php?id=processing_in_memory)



# SSD Course (Spring 2023)

## Spring 2023 Edition:

- [https://safari.ethz.ch/projects\\_and\\_seminars/spring2023/doku.php?id=modern\\_ssd](https://safari.ethz.ch/projects_and_seminars/spring2023/doku.php?id=modern_ssd)

## Fall 2022 Edition:

- [https://safari.ethz.ch/projects\\_and\\_seminars/fall2022/doku.php?id=modern\\_ssd](https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=modern_ssd)

## Youtube Livestream (Spring 2023):

- [https://www.youtube.com/watch?v=4VTwOMmsnJY&list=PL5Q2soXY2Zi\\_8qOM5Icpp8hB2SHtm4z57&pp=iAQB](https://www.youtube.com/watch?v=4VTwOMmsnJY&list=PL5Q2soXY2Zi_8qOM5Icpp8hB2SHtm4z57&pp=iAQB)

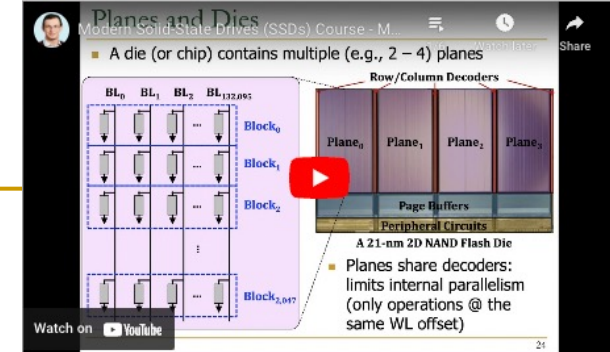
## Youtube Livestream (Fall 2022):

- <https://www.youtube.com/watch?v=hqLrd-Uj0aU&list=PL5Q2soXY2Zi9BJhenUq4JI5bwhAMpAp13&pp=iAQB>

## Project course

- Taken by Bachelor's/Master's students
- SSD Basics and Advanced Topics
- Hands-on research exploration
- Many research readings

<https://www.youtube.com/onurmutlulectures>



Fall 2022 Meetings/Schedule

| Week | Date       | Livestream       | Meeting                                                                                                                             | Learning Materials      | Assignments |
|------|------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------|-------------------------|-------------|
| W1   | 06.10      |                  | M1: P&S Course Presentation<br>PDF PPT                                                                                              | Required<br>Recommended |             |
| W2   | 12.10      | YouTube Live     | M2: Basics of NAND Flash-Based SSDs<br>PDF PPT                                                                                      | Required<br>Recommended |             |
| W3   | 19.10      | YouTube Live     | M3: NAND Flash Read/Write Operations<br>PDF PPT                                                                                     | Required<br>Recommended |             |
| W4   | 26.10      | YouTube Live     | M4: Processing inside NAND Flash<br>PDF PPT                                                                                         | Required<br>Recommended |             |
| W5   | 02.11      | YouTube Live     | M5: Advanced NAND Flash Commands & Mapping<br>PDF PPT                                                                               | Required<br>Recommended |             |
| W6   | 09.11      | YouTube Live     | M6: Processing inside Storage<br>PDF PPT                                                                                            | Required<br>Recommended |             |
| W7   | 23.11      | YouTube Live     | M7: Address Mapping & Garbage Collection<br>PDF PPT                                                                                 | Required<br>Recommended |             |
| W8   | 30.11      | YouTube Live     | M8: Introduction to MQSim<br>PDF PPT                                                                                                | Required<br>Recommended |             |
| W9   | 14.12      | YouTube Live     | M9: Fine-Grained Mapping and Multi-Plane Operation-Aware Block Management<br>PDF PPT                                                | Required<br>Recommended |             |
| W10  | 04.01.2023 | YouTube Premiere | M10a: NAND Flash Basics<br>PDF PPT                                                                                                  | Required<br>Recommended |             |
|      |            |                  | M10b: Reducing Solid-State Drive Read Latency by Optimizing Read-Retry<br>PDF PPT Paper                                             | Required<br>Recommended |             |
|      |            |                  | M10c: Evanescence: Architectural Support for Efficient Data Sanitization in Modern Flash-Based Storage Systems<br>PDF PPT Paper     | Required<br>Recommended |             |
|      |            |                  | M10d: DeepSketch: A New Machine Learning-Based Reference Search Technique for Post-Deduplication Delta Compression<br>PDF PPT Paper | Required<br>Recommended |             |
| W11  | 11.01      | YouTube Live     | M11: FLIN: Enabling Fairness and Enhancing Performance in Modern NVMe Solid State Drives<br>PDF PPT                                 | Required                |             |
| W12  | 25.01      | YouTube Premiere | M12: Flash Memory and Solid-State Drives<br>PDF PPT                                                                                 | Recommended             |             |

# Real PIM Tutorials [MICRO'23, ISCA'23, ASPLOS'23, HPCA'23]

- Jun, Mar, Feb, Oct : Lectures + Hands-on labs + Invited talks



## ISCA 2023 Real-World PIM Tutorial

Search

[Recent Changes](#) [Media Manager](#) [Sitemap](#)

Trace: • [start](#)

### Real-world Processing-in-Memory Systems for Modern Workloads

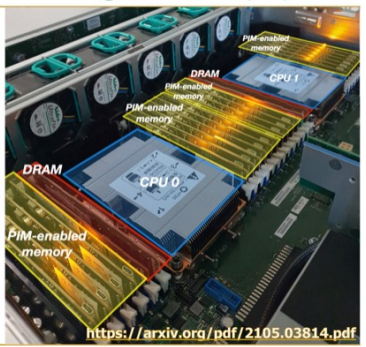
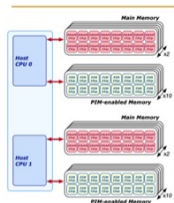
#### Tutorial Description

Processing-in-Memory (PIM) is a computing paradigm that aims at overcoming the data movement bottleneck (i.e., the waste of execution cycles and energy resulting from the back-and-forth data movement between memory units and compute units) by making memory compute-capable.

Explored over several decades since the 1960s, PIM systems are becoming a reality with the advent of the first commercial products and prototypes.

A number of startups (e.g., UPMEM, Neuroblade) are already commercializing real PIM hardware, each with its own design approach and target applications. Several major vendors (e.g., Samsung, SK Hynix, Alibaba) have presented real PIM chip prototypes in the last two years. Most of these architectures have in common that they place compute units near the memory arrays. This type of PIM is called processing near memory (PNM).

#### 2,560-DPU Processing-in-Memory System



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

#### Table of Contents

- [Real-world Processing-in-Memory Systems for Modern Workloads](#)
- [Tutorial Description](#)
- [Organizers](#)
- [Agenda \(June 18, 2023\)](#)
- [Lectures \(tentative\)](#)
- [Hands-on Labs \(tentative\)](#)
- [Learning Materials](#)

PIM can provide large improvements in both performance and energy consumption for many modern applications, thereby enabling a commercially viable way of dealing with huge amounts of data that is bottlenecking our computing systems. Yet, it is critical to (1) study and understand the characteristics that make a workload suitable for a PIM architecture, (2) propose optimization strategies for PIM kernels, and (3) develop programming frameworks and tools that can lower the learning curve and ease the adoption of PIM.

This tutorial focuses on the latest advances in PIM technology, workload characterization for PIM, and programming and optimizing PIM kernels. We will (1) provide an introduction to PIM and taxonomy of PIM systems, (2) give an overview and a rigorous analysis of existing real-world PIM hardware, (3) conduct hand-on labs about important workloads (machine learning, sparse linear algebra, bioinformatics, etc.) using real PIM systems, and (4) shed light on how to improve future PIM systems for such workloads.

<https://events.safari.ethz.ch/isca-pim-tutorial/>



# Upcoming PIM Tutorial at ISCA 2024

## ISCA 2024 Memory-Centric Computing Systems Tutorial

Saturday, June 29, Buenos Aires, Argentina

**Organizers:** Geraldo F. Oliveira, Dr. Mohammad Sadrosadati, Ataberk Olgun, Professor Onur Mutlu

**Program:** <https://events.safari.ethz.ch/isca24-memorycentric-tutorial/>

Overview of PIM | PIM taxonomy  
PIM in memory & storage  
Real-world PNM systems  
PUM for bulk bitwise operations  
Programming techniques & tools  
Infrastructures for PIM Research  
Research challenges & opportunities



We Need to Think Differently  
from the Past Approaches

# Processing in Memory: Two Approaches

1. Processing **using** Memory
2. Processing **near** Memory



# A PIM Taxonomy

---

- **Nature** (of computation)

- **Using**: Use operational properties of memory structures
- **Near**: Add logic close to memory structures

- **Technology**

- Flash, DRAM, SRAM, RRAM, MRAM, FeRAM, PCM, 3D, ...

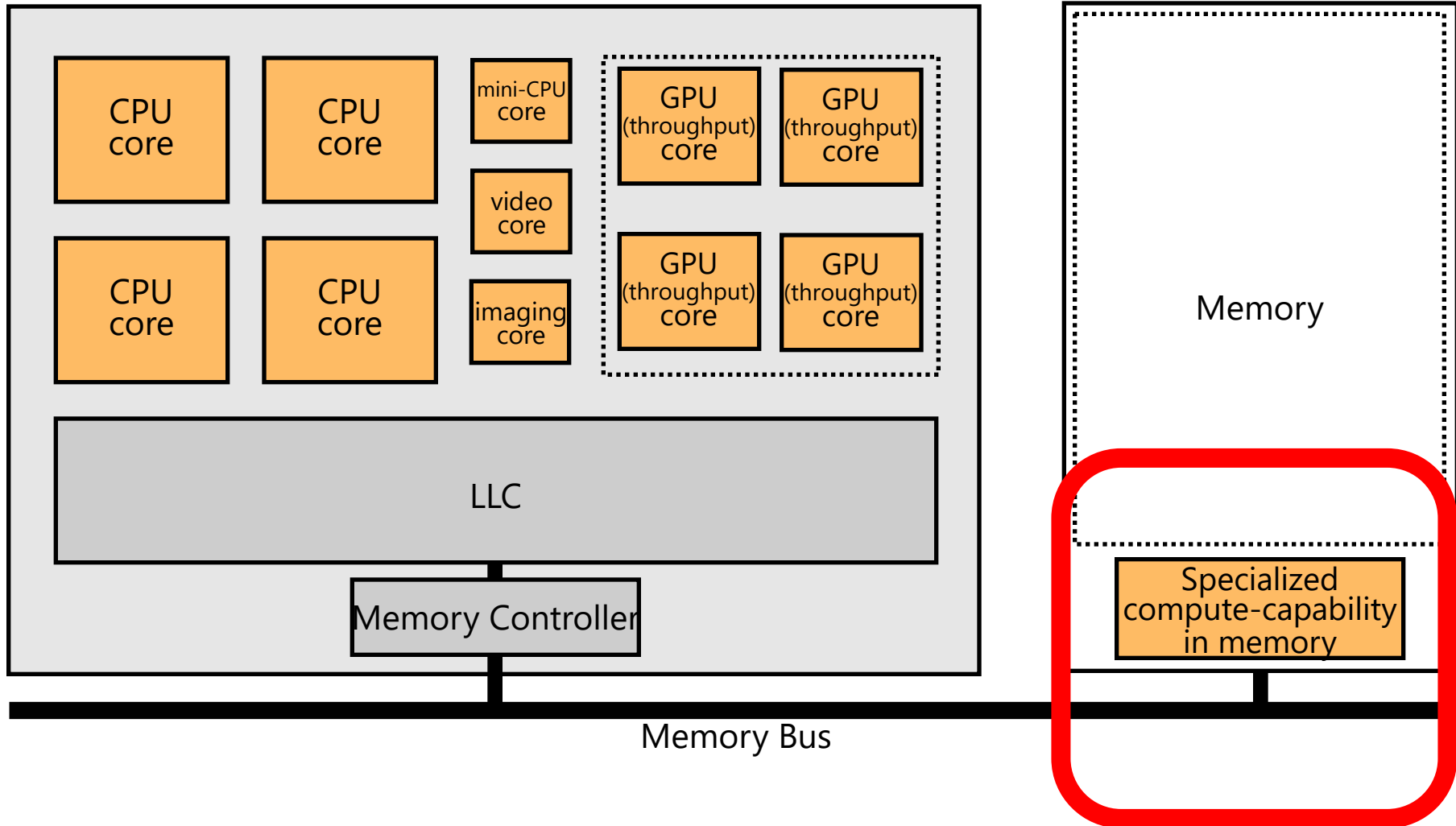
- **Location**

- Sensor, Cold Storage, Hard Disk, SSD, Main Memory, Cache, Register File, Memory Controller, Interconnect, ...

- A tuple of the three determines “PIM type”

- One can combine multiple “PIM types” in a system

# Mindset: Memory as an Accelerator



**Memory similar to a "conventional" accelerator**

# Example PIM Type: Processing using DRAM

---

- Nature: Using
- Technology: DRAM
- Location: Main Memory
  
- Processing using DRAM in Main Memory
  
- Seshadri+, "Fast Bulk Bitwise AND and OR in DRAM", IEEE CAL 2015.
- Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO 2017.
- Hajinazar+, "SIMDRAM: A Framework for Bit-Serial SIMD Processing using DRAM," ASPLOS 2021.
- Oliveira+, "MIMDRAM: An End-to-End Processing-Using-DRAM System for High-Throughput, Energy-Efficient and Programmer-Transparent Multiple-Instruction Multiple-Data Processing," HPCA 2024.

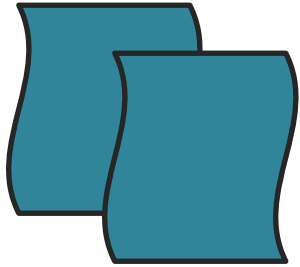
# Processing using DRAM

---

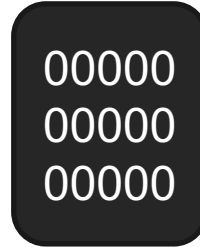
- We can support
  - Bulk bitwise AND, OR, NOT, MAJ
  - Bulk bitwise COPY and INIT/ZERO
  - True Random Number Generation; Physical Unclonable Functions
  - Lookup Table based more complex computation
- At low cost
- Using analog computation capability of DRAM
  - Idea: activating (multiple) rows performs computation
    - Even in commodity off-the-shelf DRAM chips!
- 30-77X performance and energy improvement
  - Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO 2017.
  - Seshadri+"RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data," MICRO 2013.

# Starting Simple: Data Copy and Initialization

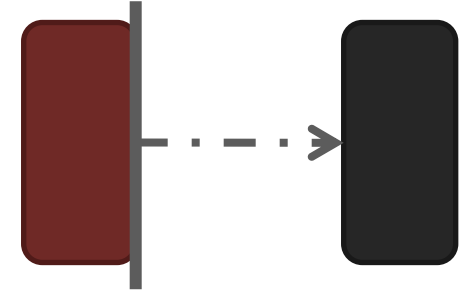
*memmove & memcpy: 5% cycles in Google's datacenter [Kanev+ ISCA'15]*



**Forking**



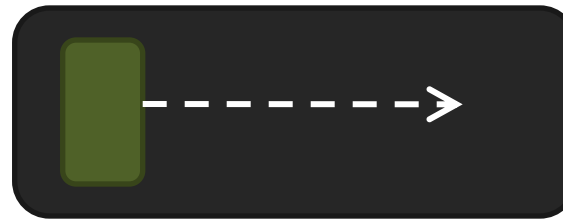
**Zero initialization  
(e.g., security)**



**Checkpointing**



**VM Cloning  
Deduplication**

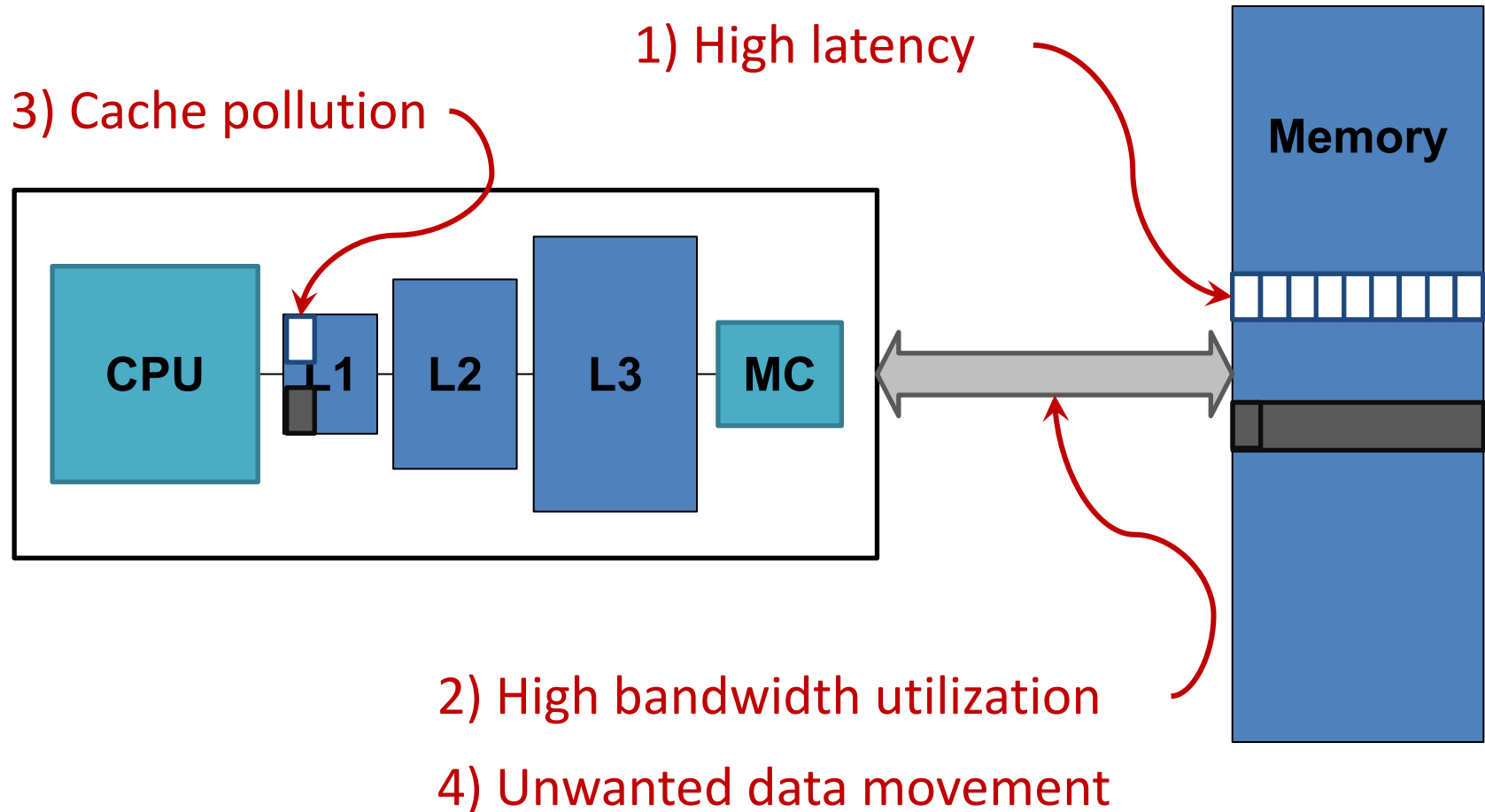


**Page Migration**

...  
**Many more**

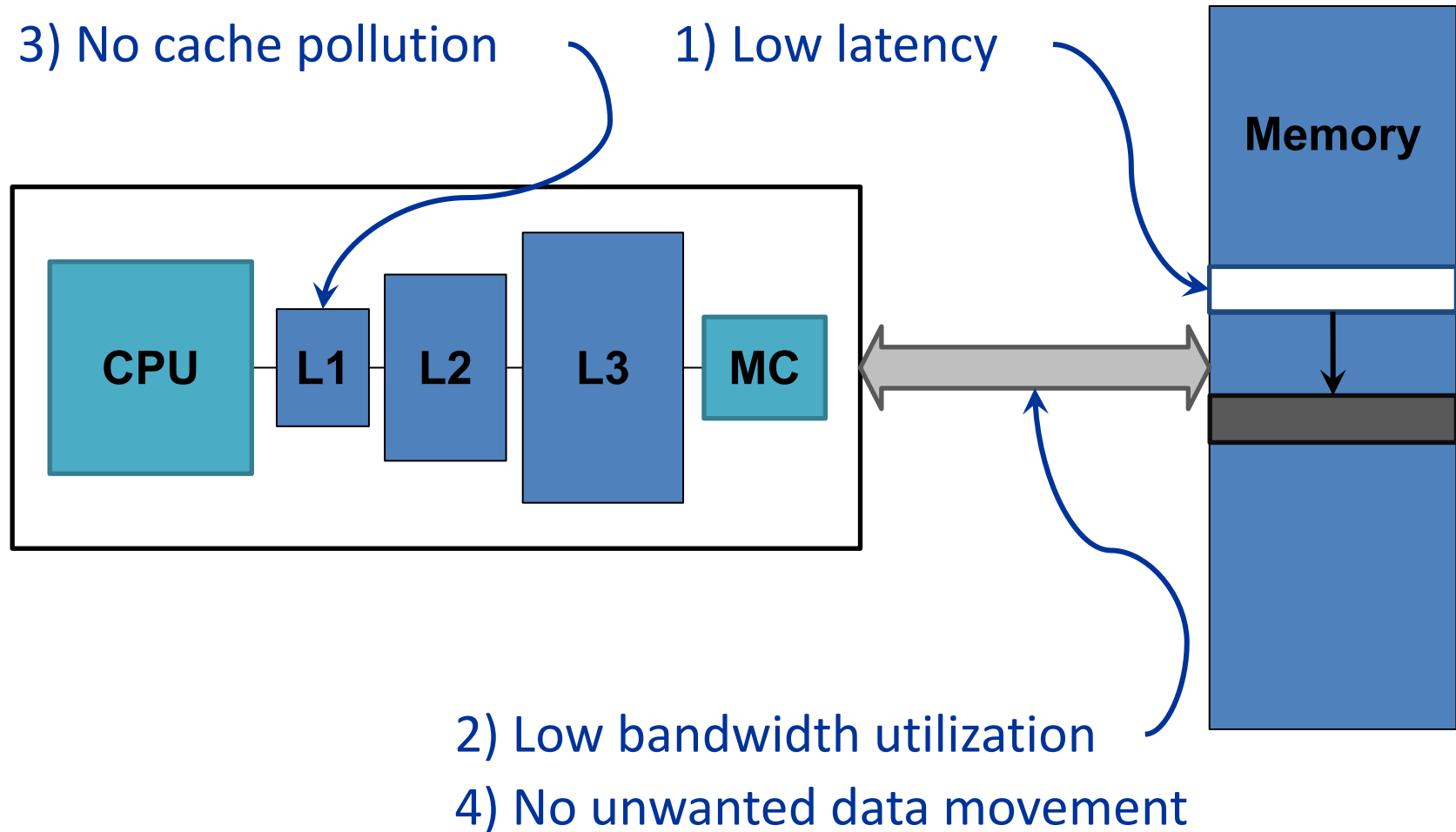


# Today's Systems: Bulk Data Copy



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

# Future Systems: In-Memory Copy

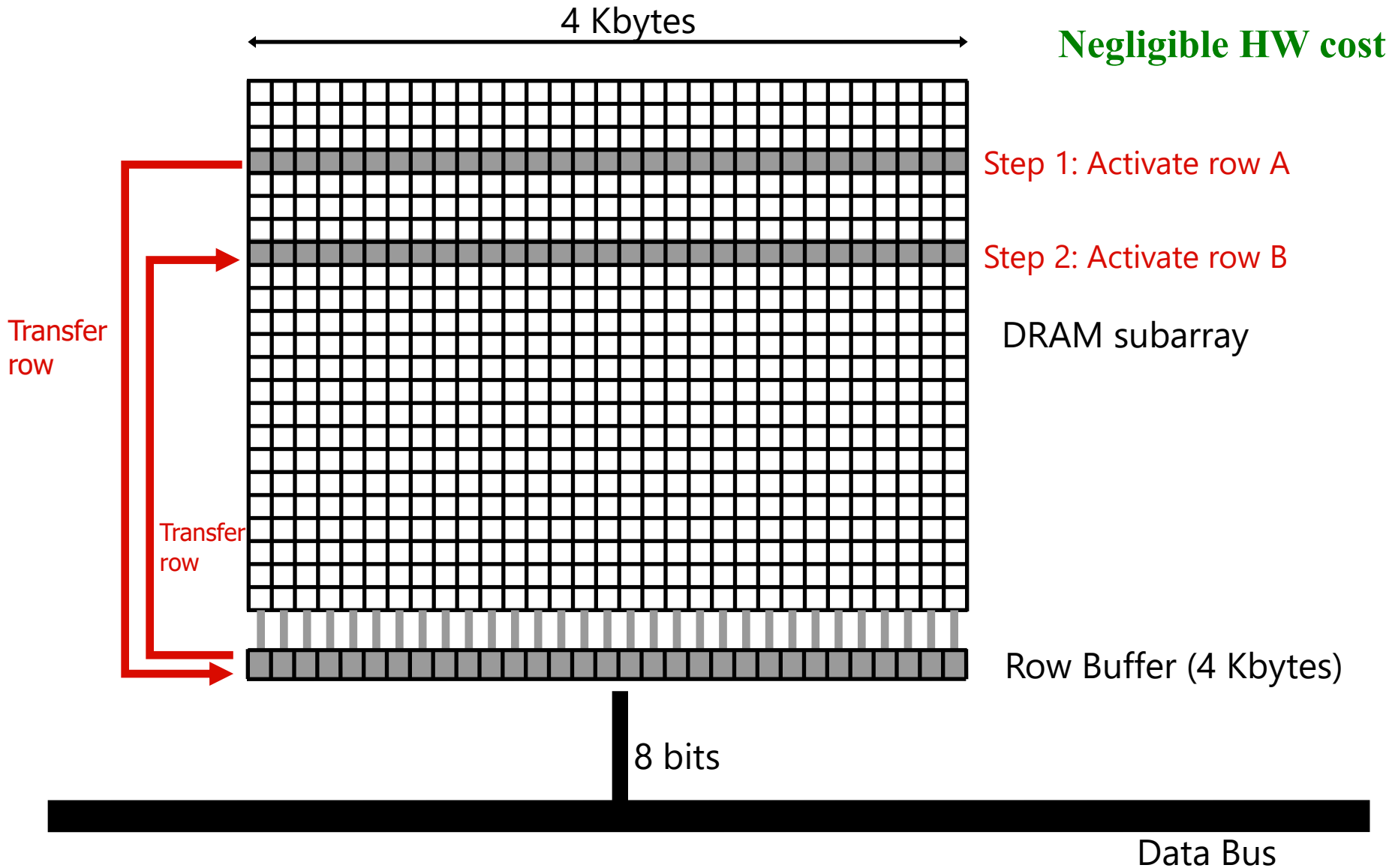


1046ns, 3.6uJ → 90ns, 0.04uJ

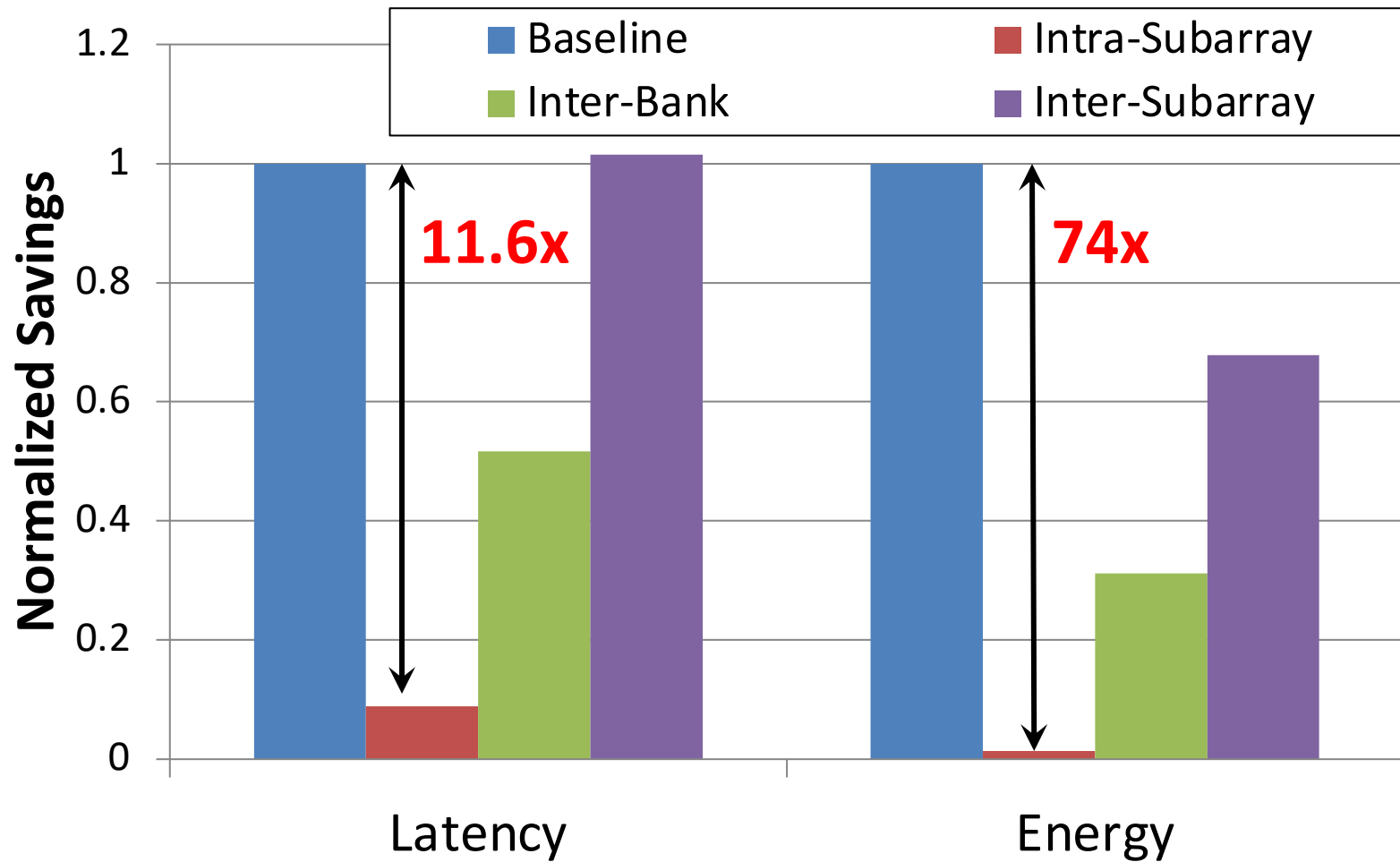
# RowClone: In-DRAM Row Copy

**Idea: Two consecutive ACTivates**

**Negligible HW cost**

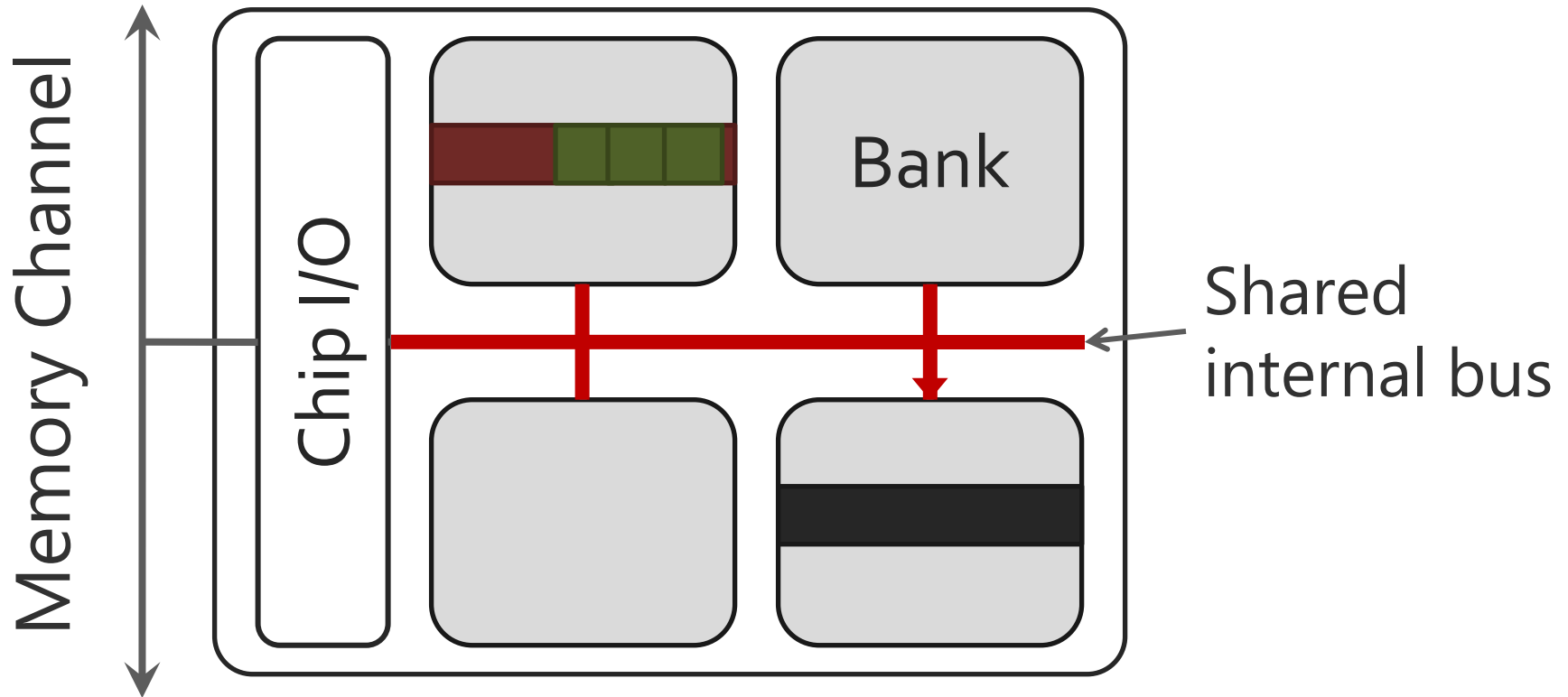


# RowClone: Latency and Energy Savings



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

# RowClone: Inter-Bank



Overlap the latency of the read and the write  
**1.9X** latency reduction, **3.2X** energy reduction



# 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      Yoongu Kim      Chris Fallin\*      Donghyuk Lee  
vseshadr@cs.cmu.edu    yoongukim@cmu.edu    cfallin@c1f.net    donghyuk1@cmu.edu

Rachata Ausavarungnirun      Gennady Pekhimenko      Yixin Luo  
rachata@cmu.edu      gpekhime@cs.cmu.edu    yixinluo@andrew.cmu.edu

Onur Mutlu      Phillip B. Gibbons†      Michael A. Kozuch†      Todd C. Mowry  
onur@cmu.edu    phillip.b.gibbons@intel.com    michael.a.kozuch@intel.com    tcm@cs.cmu.edu

Carnegie Mellon University    †Intel Pittsburgh

# RowClone in Off-the-Shelf DRAM Chips

---

- Idea: Violate DRAM timing parameters to mimic RowClone

## 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

# Real Processing Using Memory Prototype

---

- End-to-end RowClone & TRNG using off-the-shelf DRAM chips
- Idea: Violate DRAM timing parameters to mimic RowClone

## **PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM**

Ataberk Olgun<sup>§†</sup>

Juan Gómez Luna<sup>§</sup>

Konstantinos Kanellopoulos<sup>§</sup>

Behzad Salami<sup>§\*</sup>

Hasan Hassan<sup>§</sup>

Oğuz Ergin<sup>†</sup>

Onur Mutlu<sup>§</sup>

<sup>§</sup>ETH Zürich

<sup>†</sup>TOBB ETÜ

<sup>\*</sup>BSC

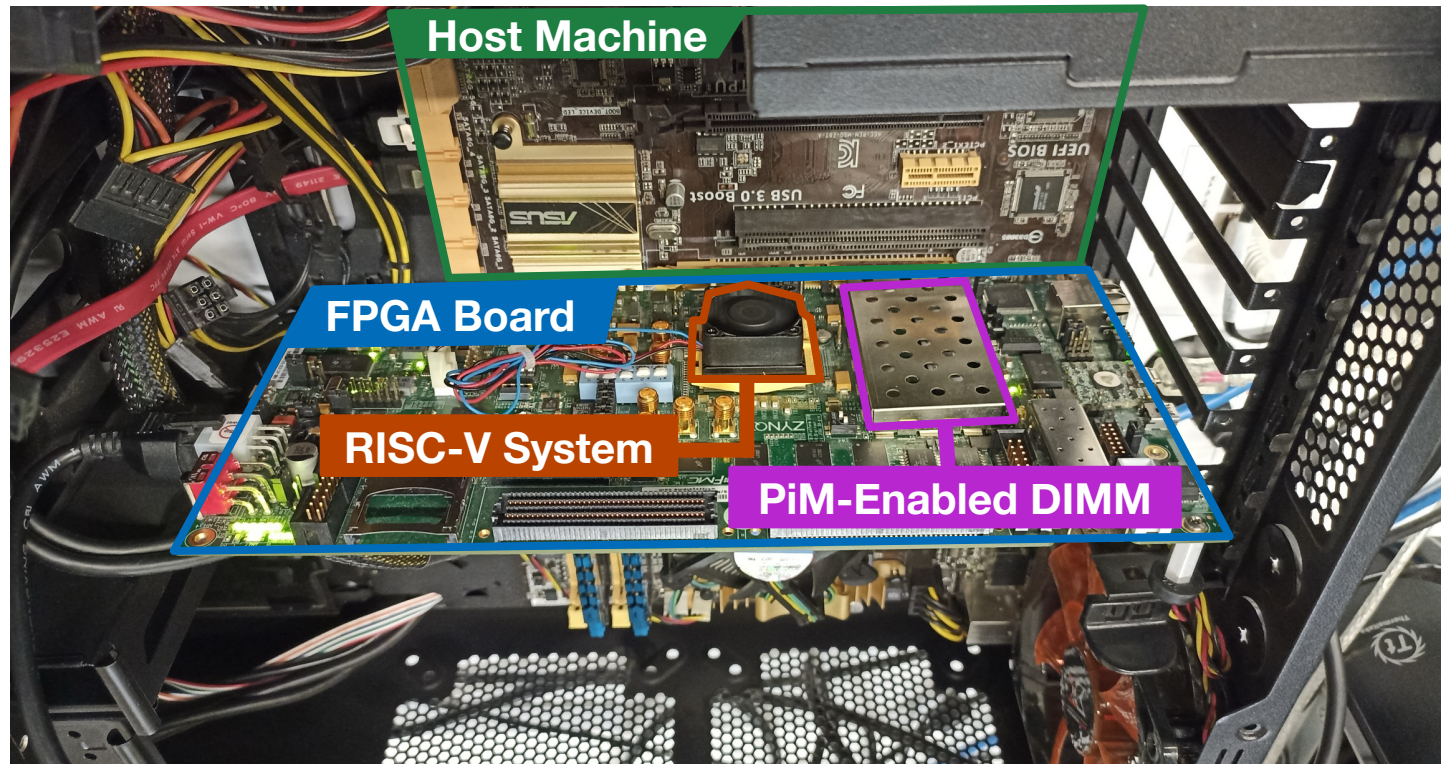
<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>

# Real Processing-using-Memory Prototype

---



<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>

# Real Processing-using-Memory Prototype

☰ README.md

## Building a PiDRAM Prototype

To build PiDRAM's prototype on Xilinx ZC706 boards, developers need to use the two sub-projects in this directory. `fpga-zynq` is a repository branched off of [UCB-BAR's fpga-zynq](#) repository. We use `fpga-zynq` to generate rocket chip designs that support end-to-end DRAM PuM execution. `controller-hardware` is where we keep the main Vivado project and Verilog sources for PiDRAM's memory controller and the top level system design.

## Rebuilding Steps

1. Navigate into `fpga-zynq` and read the README file to understand the overall workflow of the repository
  - Follow the readme in `fpga-zynq/rocket-chip/riscv-tools` to install dependencies
2. Create the Verilog source of the rocket chip design using the `ZynqCopyFPGAConfig`
  - Navigate into `zc706`, then run `make rocket CONFIG=ZynqCopyFPGAConfig -j<number of cores>`
3. Copy the generated Verilog file (should be under `zc706/src`) and overwrite the same file in `controller-hardware/source/hdl/impl/rocket-chip`
4. Open the Vivado project in `controller-hardware/Vivado_Project` using Vivado 2016.2
5. Generate a bitstream
6. Copy the bitstream (`system_top.bit`) to `fpga-zynq/zc706`
7. Use the `./build_script.sh` to generate the new `boot.bin` under `fpga-images-zc706`, you can use this file to program the FPGA using the SD-Card
  - For details, follow the relevant instructions in `fpga-zynq/README.md`

You can run programs compiled with the RISC-V Toolchain supplied within the `fpga-zynq` repository. To install the toolchain, follow the instructions under `fpga-zynq/rocket-chip/riscv-tools`.

## Generating DDR3 Controller IP sources

We cannot provide the sources for the Xilinx PHY IP we use in PiDRAM's memory controller due to licensing issues. We describe here how to regenerate them using Vivado 2016.2. First, you need to generate the IP RTL files:

- 1- Open IP Catalog
- 2- Find "Memory Interface Generator (MIG 7 Series)" IP and double click

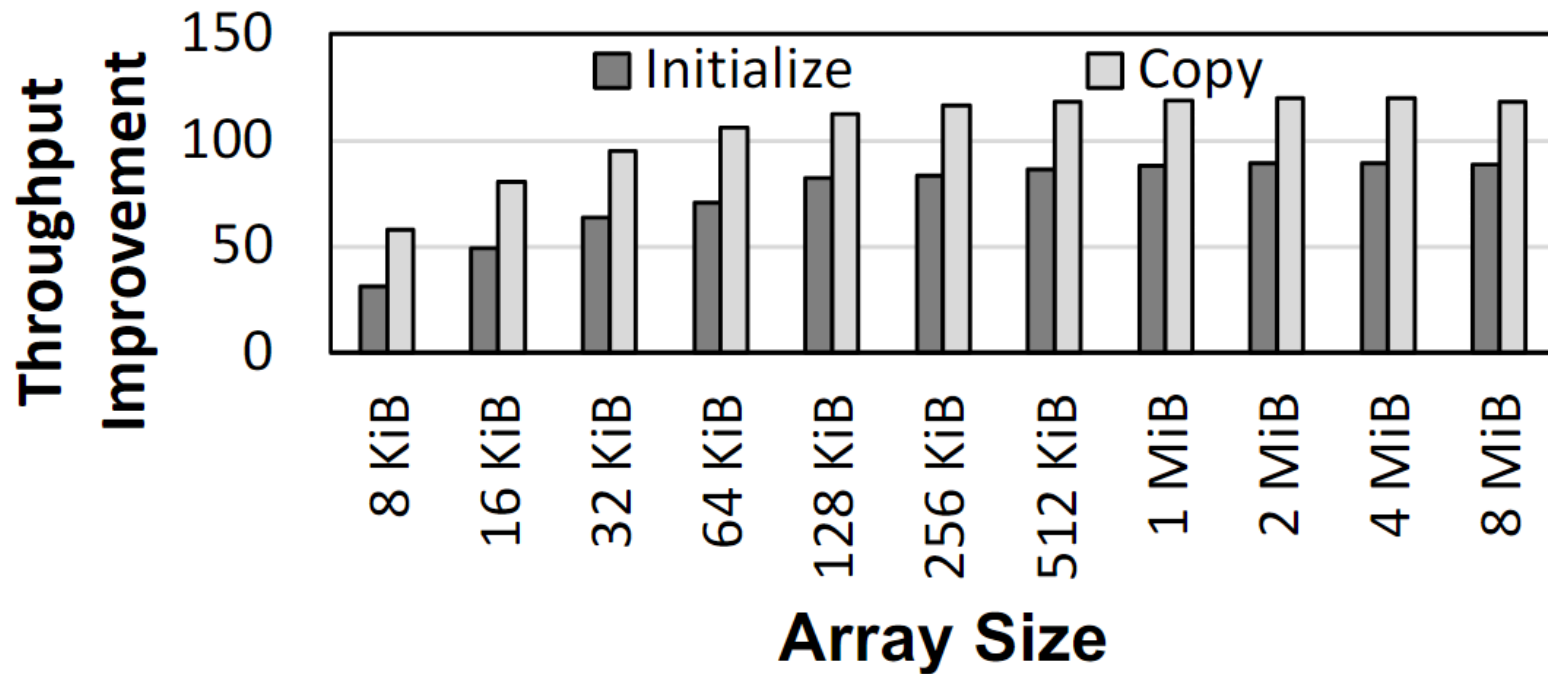
<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>



# Microbenchmark Copy/Initialization Throughput



**In-DRAM Copy and Initialization  
improve throughput by 119x and 89x**

# More on PiDRAM

---

- Ataberk Olgun, Juan Gomez Luna, Konstantinos Kanellopoulos, Behzad Salami, Hasan Hassan, Oguz Ergin, and Onur Mutlu,  
**["PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM"](#)**  
*[ACM Transactions on Architecture and Code Optimization \(TACO\)](#)*, March 2023.  
[\[arXiv version\]](#)  
Presented at the [18th HiPEAC Conference](#), Toulouse, France, January 2023.  
[\[Slides \(pptx\) \(pdf\)\]](#)  
[\[Longer Lecture Slides \(pptx\) \(pdf\)\]](#)  
[\[Lecture Video \(40 minutes\)\]](#)  
[\[PiDRAM Source Code\]](#)

## PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM

Ataberk Olgun<sup>§</sup>      Juan Gómez Luna<sup>§</sup>      Konstantinos Kanellopoulos<sup>§</sup>      Behzad Salami<sup>§</sup>  
Hasan Hassan<sup>§</sup>      Oğuz Ergin<sup>†</sup>      Onur Mutlu<sup>§</sup>

<sup>§</sup>*ETH Zürich*

<sup>†</sup>*TOBB University of Economics and Technology*

# RowClone Extensions and Follow-Up Work

---

- Can this be improved to do **faster inter-subarray copy**?
  - Yes, **see LISA [Chang et al., HPCA 2016]**
- Can we enable **data movement at smaller granularities within a bank**?
  - Yes, **see FIGARO [Wang et al., MICRO 2020]**
- Can this be improved to do **better inter-bank copy**?
  - Yes, **see Network-on-Memory [CAL 2020]**
- Can similar ideas and DRAM properties be used to perform **computation on data**?
  - Yes, **see Ambit [Seshadri et al., CAL 2015, MICRO 2017]**

# LISA: Increasing Connectivity in DRAM

---

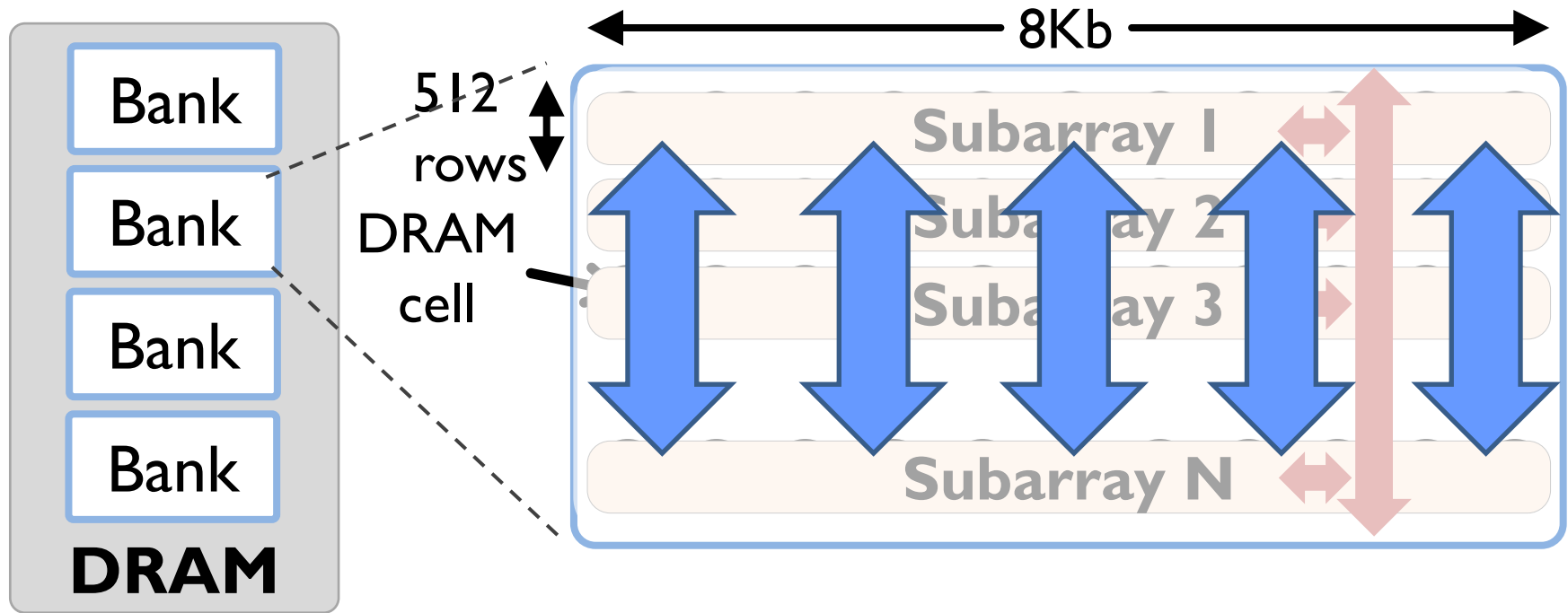
- Kevin K. Chang, Prashant J. Nair, Saugata Ghose, Donghyuk Lee, Moinuddin K. Qureshi, and Onur Mutlu,  
**"Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM"**  
*Proceedings of the 22nd International Symposium on High-Performance Computer Architecture (HPCA)*, Barcelona, Spain, March 2016.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Source Code](#)]

## Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM

Kevin K. Chang<sup>†</sup>, Prashant J. Nair<sup>\*</sup>, Donghyuk Lee<sup>†</sup>, Saugata Ghose<sup>†</sup>, Moinuddin K. Qureshi<sup>\*</sup>, and Onur Mutlu<sup>†</sup>

<sup>†</sup>Carnegie Mellon University    <sup>\*</sup>Georgia Institute of Technology

# Moving Data Inside DRAM?

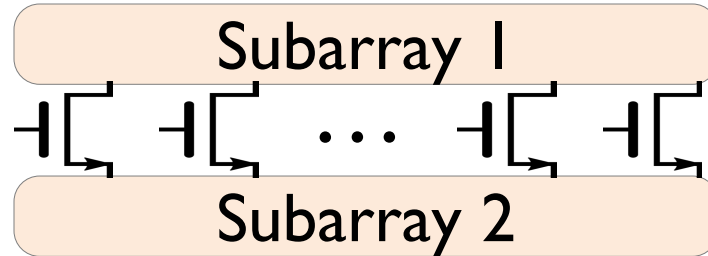


**Goal: Provide a new substrate to enable wide connectivity between subarrays**



# Key Idea and Applications

- **Low-cost Inter-linked subarrays (LISA)**
  - Fast bulk data movement between subarrays
  - **Wide datapath via isolation transistors**: 0.8% DRAM chip area



- LISA is a **versatile substrate** → new applications
  - Fast bulk data copy**: Copy latency 1.363ms→0.148ms (9.2x)  
→ 66% speedup, -55% DRAM energy
  - In-DRAM caching**: Hot data access latency 48.7ns→21.5ns (2.2x)  
→ 5% speedup
  - Fast precharge**: Precharge latency 13.1ns→5.0ns (2.6x)  
→ 8% speedup

# More on LISA

---

- Kevin K. Chang, Prashant J. Nair, Saugata Ghose, Donghyuk Lee, Moinuddin K. Qureshi, and Onur Mutlu,  
**"Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM"**  
*Proceedings of the 22nd International Symposium on High-Performance Computer Architecture (HPCA)*, Barcelona, Spain, March 2016.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Source Code](#)]

## Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM

Kevin K. Chang<sup>†</sup>, Prashant J. Nair<sup>\*</sup>, Donghyuk Lee<sup>†</sup>, Saugata Ghose<sup>†</sup>, Moinuddin K. Qureshi<sup>\*</sup>, and Onur Mutlu<sup>†</sup>

<sup>†</sup>Carnegie Mellon University    <sup>\*</sup>Georgia Institute of Technology

# FIGARO: Fine-Grained In-DRAM Copy

---

- Yaohua Wang, Lois Orosa, Xiangjun Peng, Yang Guo, Saugata Ghose, Minesh Patel, Jeremie S. Kim, Juan Gómez Luna, Mohammad Sadrosadati, Nika Mansouri Ghiasi, and Onur Mutlu,  
**"FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching"**  
*Proceedings of the 53rd International Symposium on Microarchitecture (MICRO), Virtual, October 2020.*

## FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching

Yaohua Wang<sup>\*</sup> Lois Orosa<sup>†</sup> Xiangjun Peng<sup>⊙\*</sup> Yang Guo<sup>\*</sup> Saugata Ghose<sup>◇‡</sup> Minesh Patel<sup>†</sup>  
Jeremie S. Kim<sup>†</sup> Juan Gómez Luna<sup>†</sup> Mohammad Sadrosadati<sup>§</sup> Nika Mansouri Ghiasi<sup>†</sup> Onur Mutlu<sup>†‡</sup>

<sup>\*</sup>National University of Defense Technology <sup>†</sup>ETH Zürich <sup>⊙</sup>Chinese University of Hong Kong

<sup>◇</sup>University of Illinois at Urbana–Champaign <sup>‡</sup>Carnegie Mellon University <sup>§</sup>Institute of Research in Fundamental Sciences

# Network-On-Memory: Fast Inter-Bank Copy

---

- Seyyed Hossein SeyyedAghaei Rezaei, Mehdi Modarressi, Rachata Ausavarungnirun, Mohammad Sadrosadati, Onur Mutlu, and Masoud Daneshtalab,  
["NoM: Network-on-Memory for Inter-Bank Data Transfer in Highly-Banked Memories"](#)  
[IEEE Computer Architecture Letters](#) (**CAL**), to appear in 2020.

## NOm: NETWORK-ON-MEMORY FOR INTER-BANK DATA TRANSFER IN HIGHLY-BANKED MEMORIES

Seyyed Hossein SeyyedAghaei Rezaei<sup>1</sup>  
Mohammad Sadrosadati<sup>3</sup>

Mehdi Modarressi<sup>1,3</sup>  
Onur Mutlu<sup>4</sup>

Rachata Ausavarungnirun<sup>2</sup>  
Masoud Daneshtalab<sup>5</sup>

<sup>1</sup>University of Tehran

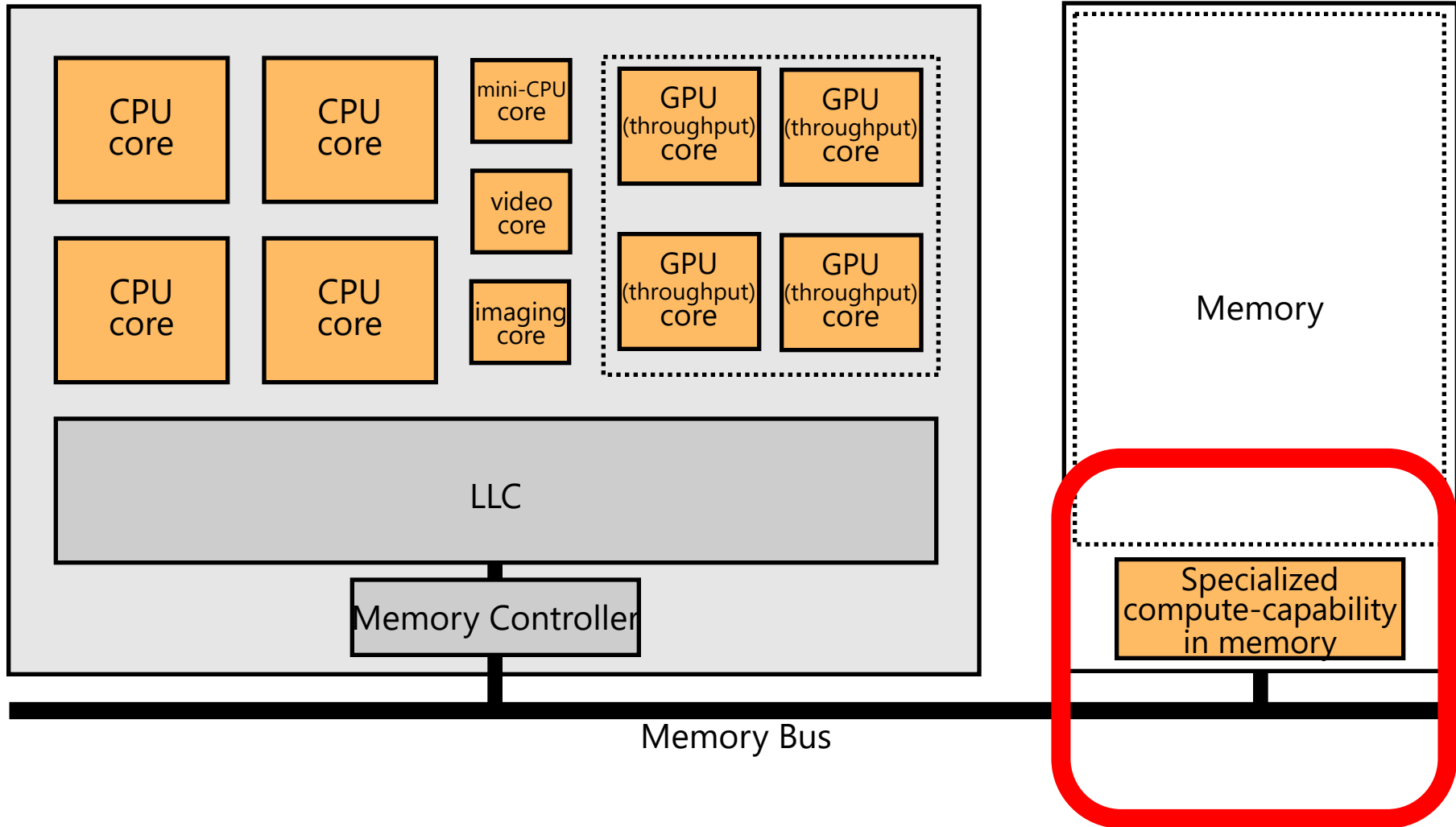
<sup>2</sup>King Mongkut's University of Technology North Bangkok

<sup>3</sup>Institute for Research in Fundamental Sciences

<sup>4</sup>ETH Zürich

<sup>5</sup>Mälardalens University

# Mindset: Memory as an Accelerator



**Memory similar to a "conventional" accelerator**

# Lecture on RowClone & Processing using DRAM

Mindset: Memory as an Accelerator

The diagram illustrates a system architecture where memory is treated as an accelerator. On the left, a large gray box represents the system, containing several components: four 'CPU core' blocks, one 'mini-CPU core', one 'video core', one 'imaging core', and four 'GPU (throughput) core' blocks. Below these is a large 'LLC' (Last Level Cache) block, which is connected to a 'Memory Controller' block. The 'Memory Controller' is connected to a 'Memory Bus'. To the right of the system box is a large 'Memory' block. Within the 'Memory' block, a red rounded rectangle highlights a 'Specialized compute-capability in memory' block, which is also connected to the 'Memory Bus'. A video player interface is overlaid on the bottom of the diagram, showing a red text overlay: 'Memory similar to a "conventional" accelerator'. The video player also shows a progress bar at 43:48 / 2:03:45 and various control icons.

Onur Mutlu

Memory

Specialized compute-capability in memory

Memory Bus

LLC

Memory Controller

CPU core

mini-CPU core

video core

imaging core

GPU (throughput) core

Memory similar to a "conventional" accelerator

DEPARTMENT OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING (D-ITET)

Seminar in Computer Arch. - Meeting 3: RowClone: In-Memory Data Copy and Initialization (Fall 2021)

292 views • Streamed live on Oct 7, 2021

21 0 SHARE SAVE ...



Onur Mutlu Lectures  
19.1K subscribers

SUBSCRIBED



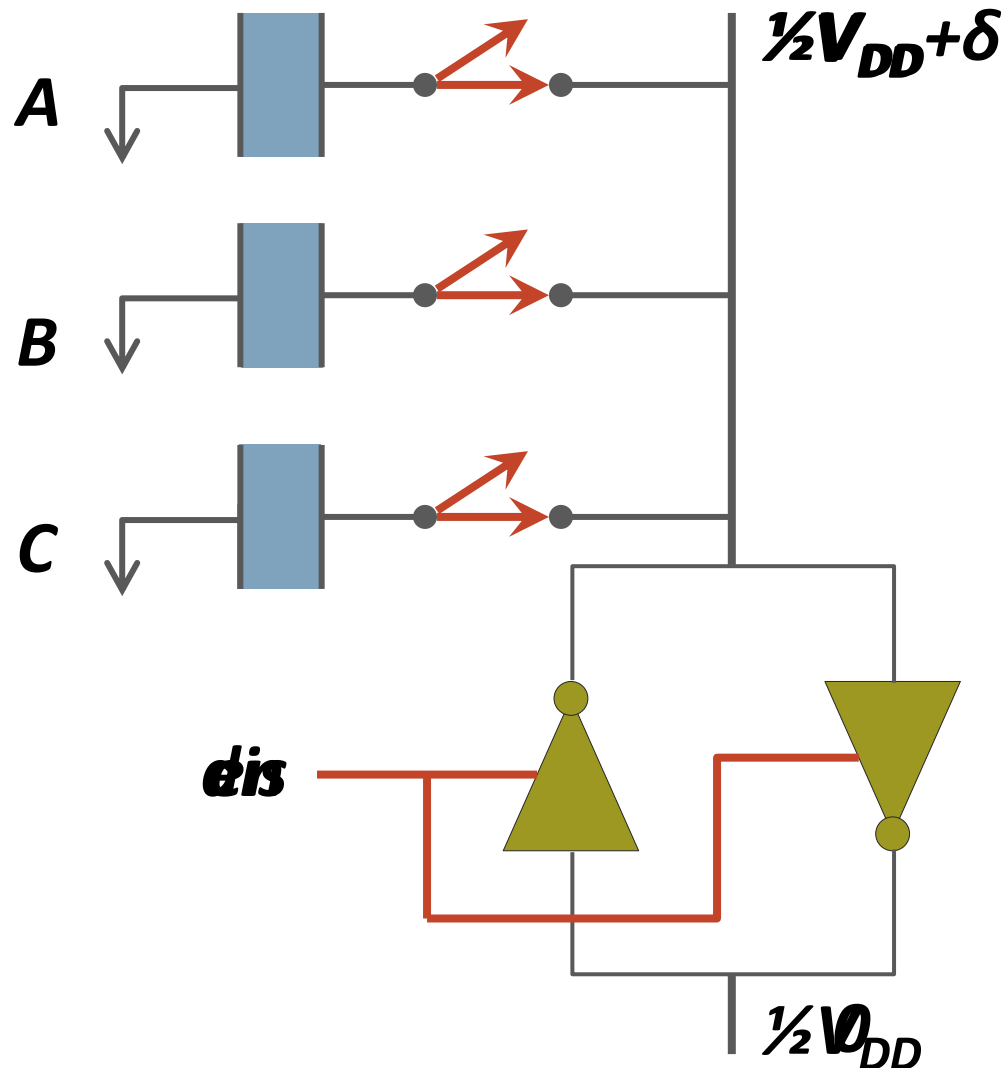


# (Truly) In-Memory Computation

---

- We can support in-DRAM 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.
- New memory technologies enable even more opportunities
  - Memristors, resistive RAM, phase change mem, STT-MRAM, ...
  - Can operate on data with minimal movement

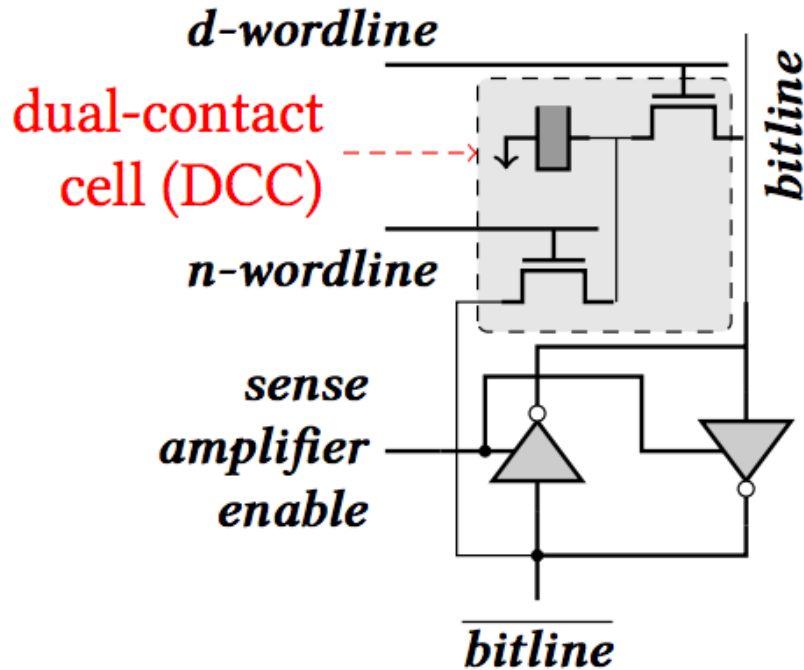
# In-DRAM AND/OR: Triple Row Activation



**Final State**  
 **$AB + BC + AC$**

**$C(A + B) +$   
 **$\sim C(AB)$****

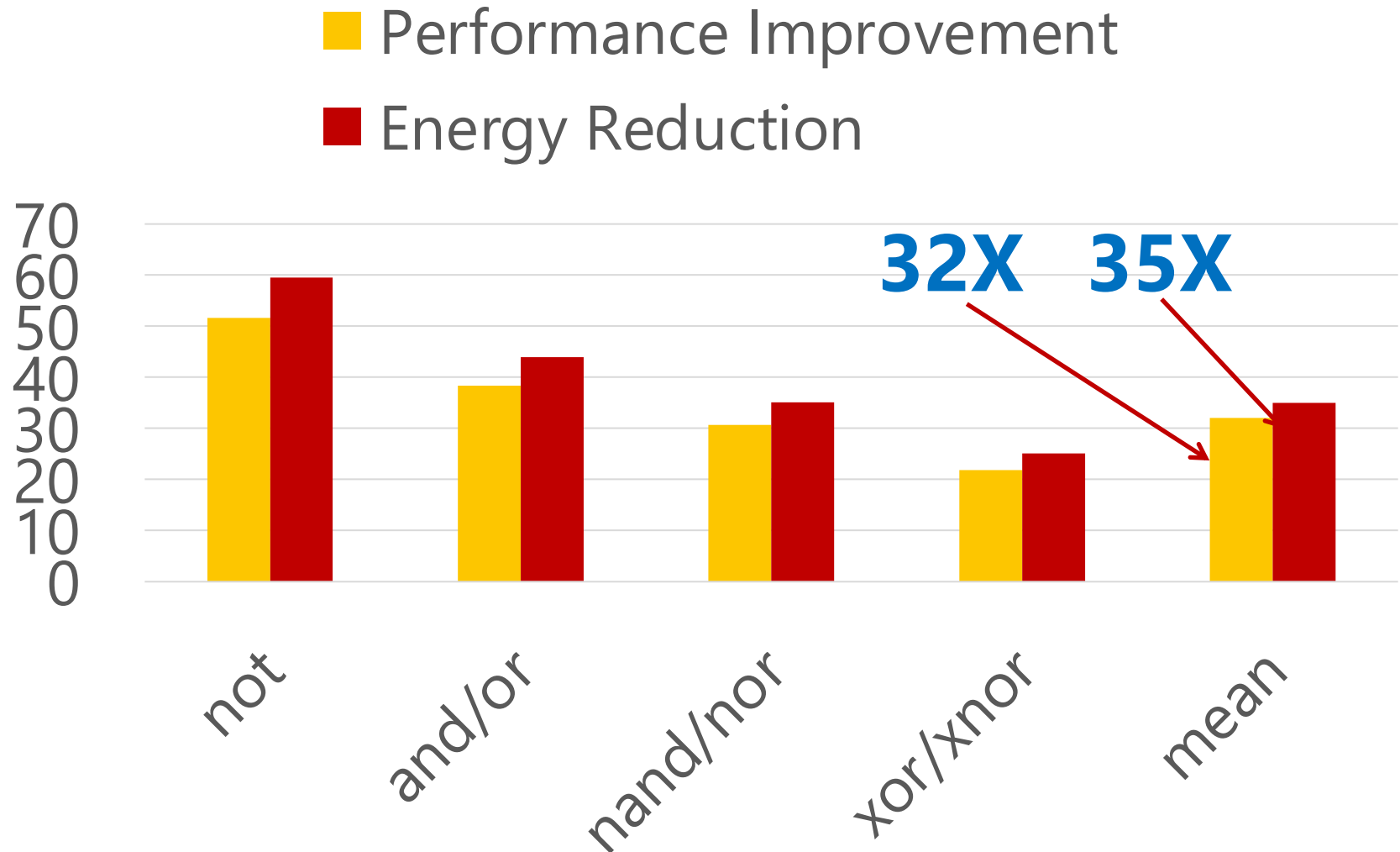
# In-DRAM NOT: Dual Contact Cell



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

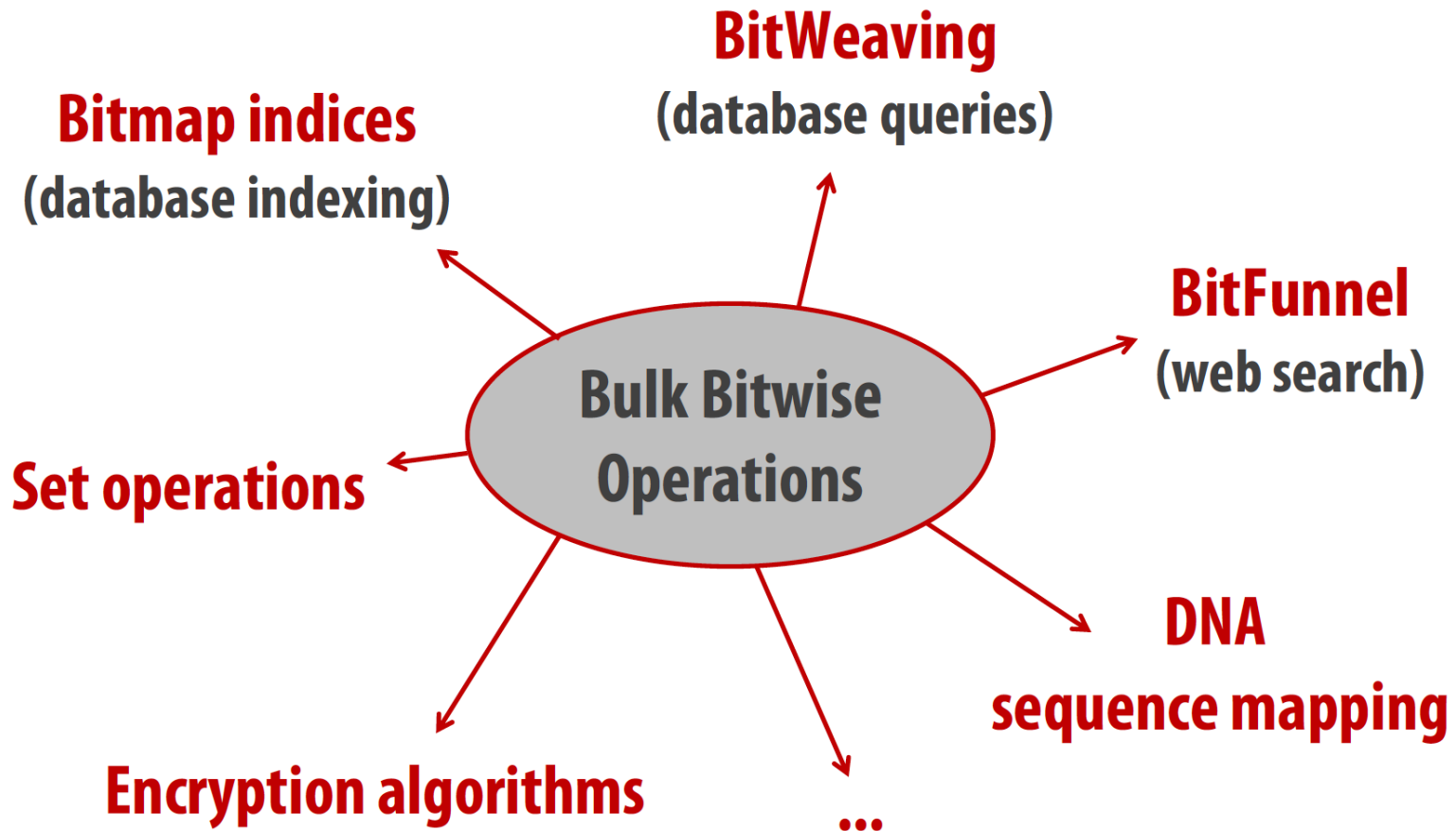
Idea:  
Feed the  
negated value  
in the sense amplifier  
into a special row

# Ambit vs. DDR3: Performance and Energy

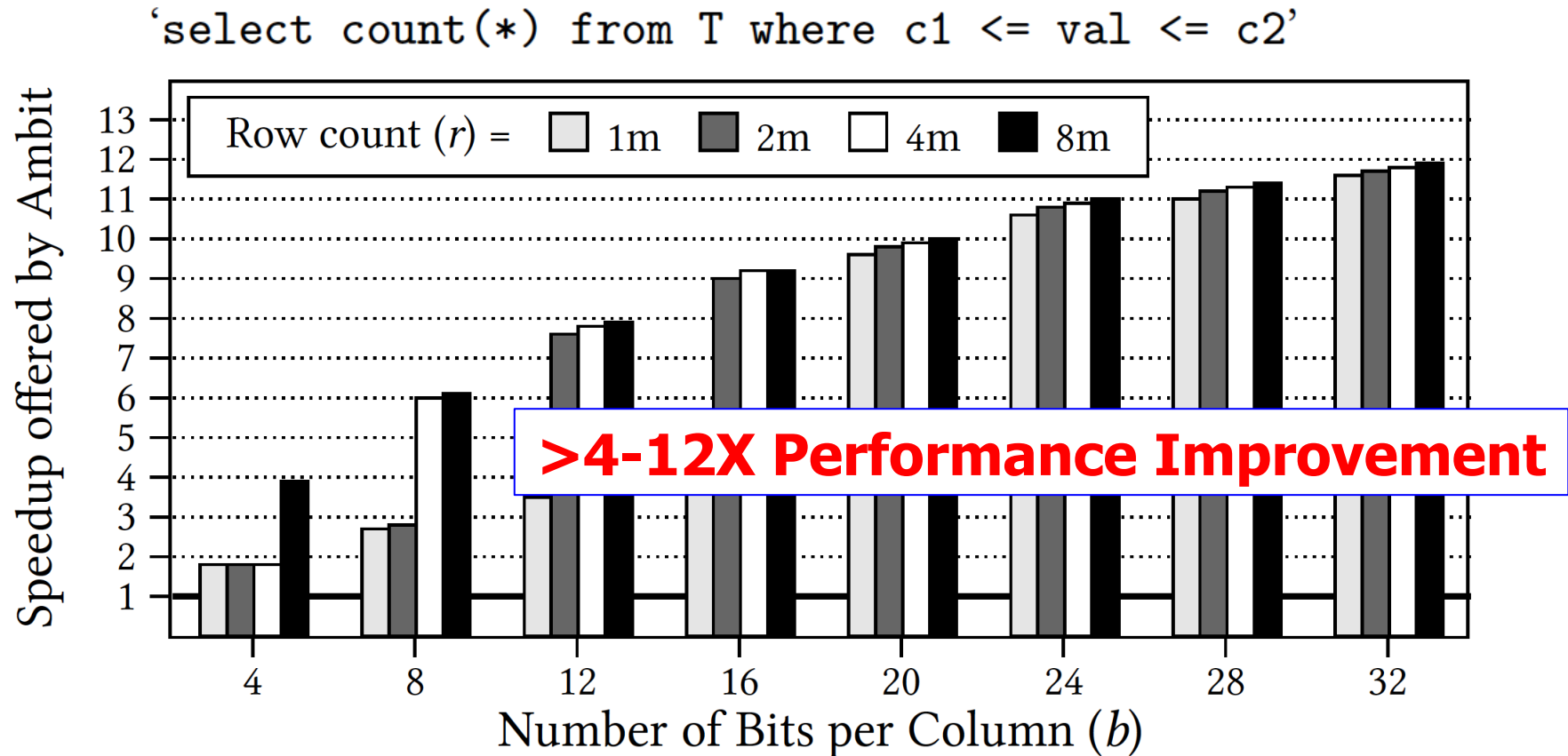


# Bulk Bitwise Operations in Workloads

---



# In-DRAM Acceleration of Database Queries



**Figure 11: Speedup offered by Ambit over baseline CPU with SIMD for BitWeaving**

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



# 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

# More on Ambit

---

- Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, and Todd C. Mowry,  
["Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology"](#)  
*Proceedings of the 50th International Symposium on Microarchitecture (MICRO)*, Boston, MA, USA, October 2017.  
[\[Slides \(pptx\) \(pdf\)\]](#) [\[Lightning Session Slides \(pptx\) \(pdf\)\]](#) [\[Poster \(pptx\) \(pdf\)\]](#)

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

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

<sup>1</sup>Microsoft Research India   <sup>2</sup>NVIDIA Research   <sup>3</sup>Intel   <sup>4</sup>ETH Zürich   <sup>5</sup>Carnegie Mellon University

# In-DRAM Bulk Bitwise Execution

---

- Vivek Seshadri and Onur Mutlu,  
**"In-DRAM Bulk Bitwise Execution Engine"**  
*Invited Book Chapter in Advances in Computers*, to appear  
in 2020.  
[Preliminary arXiv version]

## In-DRAM Bulk Bitwise Execution Engine

Vivek Seshadri  
Microsoft Research India  
visesha@microsoft.com

Onur Mutlu  
ETH Zürich  
onur.mutlu@inf.ethz.ch

# 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

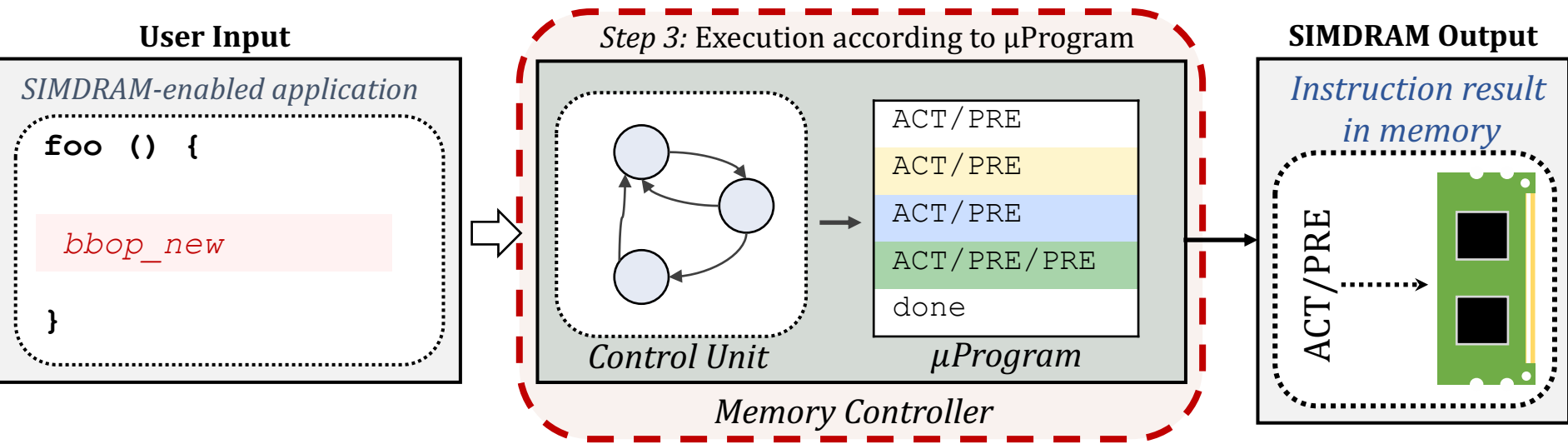
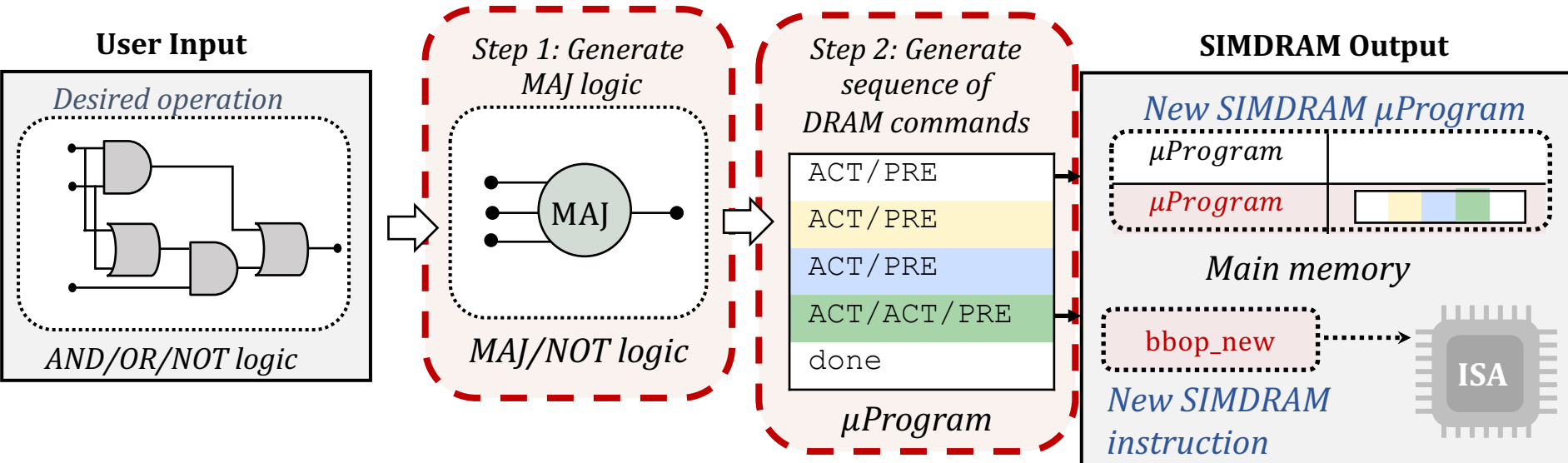
|                                    |                                   |                             |                                  |
|------------------------------------|-----------------------------------|-----------------------------|----------------------------------|
| *Nastaran Hajinazar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

<sup>1</sup>ETH Zürich

<sup>2</sup>Simon Fraser University

<sup>3</sup>University of Illinois at Urbana–Champaign

# SIMDRAM Framework: Overview



# SIMDRAM Key Results

Evaluated on:

- 16 complex in-DRAM operations
- 7 commonly-used real-world applications

**SIMDRAM provides:**

- **88×** and **5.8×** the **throughput** of a **CPU** and a **high-end GPU**, respectively, over **16 operations**
- **257×** and **31×** the **energy efficiency** of a **CPU** and a **high-end GPU**, respectively, over **16 operations**
- **21×** and **2.1×** the **performance** of a **CPU** and a **high-end GPU**, over **seven real-world applications**

**SAFARI**



# More on SIMD RAM

---

- 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 <sup>1,2</sup> | *Geraldo F. Oliveira <sup>1</sup> | Sven Gregorio <sup>1</sup>  | João Dinis Ferreira <sup>1</sup> |
| Nika Mansouri Ghiasi <sup>1</sup>  | Minesh Patel <sup>1</sup>         | Mohammed Alser <sup>1</sup> | Saugata Ghose <sup>3</sup>       |
|                                    | Juan Gómez-Luna <sup>1</sup>      | Onur Mutlu <sup>1</sup>     |                                  |

<sup>1</sup>ETH Zürich

<sup>2</sup>Simon Fraser University

<sup>3</sup>University of Illinois at Urbana–Champaign

# MIMDRAM: More Flexible Processing using DRAM

---

- **Appears at HPCA 2024**     <https://arxiv.org/pdf/2402.19080.pdf>

## **MIMDRAM: An End-to-End Processing-Using-DRAM System for High-Throughput, Energy-Efficient and Programmer-Transparent Multiple-Instruction Multiple-Data Computing**

Geraldo F. Oliveira<sup>†</sup>     Ataberk Olgun<sup>†</sup>     Abdullah Giray Yağlıkçı<sup>†</sup>     F. Nisa Bostancı<sup>†</sup>  
Juan Gómez-Luna<sup>†</sup>     Saugata Ghose<sup>‡</sup>     Onur Mutlu<sup>†</sup>

<sup>†</sup> *ETH Zürich*

<sup>‡</sup> *Univ. of Illinois Urbana-Champaign*

*Our **goal** is to design a flexible PUD system that overcomes the limitations caused by the large and rigid granularity of PUD. To this end, we propose MIMDRAM, a hardware/software co-designed PUD system that introduces new mechanisms to allocate and control only the necessary resources for a given PUD operation. The key idea of MIMDRAM is to leverage fine-grained DRAM (i.e., the ability to independently access smaller segments of a large DRAM row) for PUD computation. MIMDRAM exploits this key idea to enable a multiple-instruction multiple-data (MIMD) execution model in each DRAM subarray (and SIMD execution within each DRAM row segment).*

# MIMDRAM: Executive Summary

**Problem:** Processing-Using-DRAM (PUD) suffers from three issues caused by DRAM's large and rigid access granularity

- Underutilization due to data parallelism variation in (and across) applications
- Limited computation support due to a lack of interconnects
- Challenging programming model due to a lack of compilers

**Goal:** Design a flexible PUD system that overcomes the three limitations caused by DRAM's large and rigid access granularity

**Key Mechanism:** MIMDRAM, a hardware/software co-design PUD system

- **Key idea:** leverage fine-grained DRAM for PUD operation
- **HW:** - simple changes to the DRAM array, enabling concurrent PUD operations
  - low-cost interconnects at the DRAM peripherals for data reduction
- **SW:** - compiler and OS support to generate and map PUD instructions

**Key Results:** MIMDRAM achieves

- **14.3x, 30.6x, and 6.8x** the energy efficiency of state-of-the-art PUD systems, a high-end CPU and GPU, respectively
- Small area cost to a DRAM chip (**1.11%**) and CPU die (**0.6%**)

# In-DRAM Lookup-Table Based Execution

João Dinis Ferreira, Gabriel Falcao, Juan Gómez-Luna, Mohammed Alser, Lois Orosa, Mohammad Sadrosadati, Jeremie S. Kim, Geraldo F. Oliveira, Taha Shahroodi, Anant Nori, and Onur Mutlu,

**"pLUTo: Enabling Massively Parallel Computation in DRAM via Lookup Tables"**

*Proceedings of the 55th International Symposium on Microarchitecture (MICRO)*, Chicago, IL, USA, October 2022.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Longer Lecture Slides \(pptx\)](#) ([pdf](#))]

[[Lecture Video](#) (26 minutes)]

[[arXiv version](#)]

[[Source Code](#) (Officially Artifact Evaluated with All Badges)]

***Officially artifact evaluated as available, reusable and reproducible.***



## pLUTo: Enabling Massively Parallel Computation in DRAM via Lookup Tables

João Dinis Ferreira<sup>§</sup>

Gabriel Falcao<sup>†</sup>

Juan Gómez-Luna<sup>§</sup>

Mohammed Alser<sup>§</sup>

Lois Orosa<sup>§∇</sup>

Mohammad Sadrosadati<sup>§</sup>

Jeremie S. Kim<sup>§</sup>

Geraldo F. Oliveira<sup>§</sup>

Taha Shahroodi<sup>‡</sup>

Anant Nori<sup>\*</sup>

Onur Mutlu<sup>§</sup>

<sup>§</sup>ETH Zürich

<sup>†</sup>IT, University of Coimbra

<sup>∇</sup>Galicia Supercomputing Center

<sup>‡</sup>TU Delft

<sup>\*</sup>Intel

# In-DRAM Physical Unclonable Functions

---

- Jeremie S. Kim, Minesh Patel, Hasan Hassan, and Onur Mutlu,  
**"The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency-Reliability Tradeoff in Modern DRAM Devices"**  
*Proceedings of the 24th International Symposium on High-Performance Computer Architecture (HPCA)*, Vienna, Austria, February 2018.  
[[Lightning Talk Video](#)]  
[[Slides \(pptx\)](#)] [[pdf](#)] [[Lightning Session Slides \(pptx\)](#)] [[pdf](#)]  
[[Full Talk Lecture Video](#) (28 minutes)]

## The DRAM Latency PUF:

Quickly Evaluating Physical Unclonable Functions

by Exploiting the Latency-Reliability Tradeoff in Modern Commodity DRAM Devices

Jeremie S. Kim<sup>†§</sup>

Minesh Patel<sup>§</sup>

Hasan Hassan<sup>§</sup>

Onur Mutlu<sup>§†</sup>

<sup>†</sup>Carnegie Mellon University

<sup>§</sup>ETH Zürich



# In-DRAM True Random Number Generation

---

- Jeremie S. Kim, Minesh Patel, Hasan Hassan, Lois Orosa, and Onur Mutlu,  
**"D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput"**

*Proceedings of the 25th International Symposium on High-Performance Computer Architecture (HPCA), Washington, DC, USA, February 2019.*

[[Slides \(pptx\)](#) ([pdf](#))]

[[Full Talk Video](#) (21 minutes)]

[[Full Talk Lecture Video](#) (27 minutes)]

***Top Picks Honorable Mention by IEEE Micro.***

## D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput

Jeremie S. Kim<sup>‡§</sup>

Minesh Patel<sup>§</sup>

Hasan Hassan<sup>§</sup>

Lois Orosa<sup>§</sup>

Onur Mutlu<sup>§‡</sup>

<sup>‡</sup>Carnegie Mellon University

<sup>§</sup>ETH Zürich



# In-DRAM True Random Number Generation

---

- Ataberk Olgun, Minesh Patel, A. Giray Yaglikci, Haocong Luo, Jeremie S. Kim, F. Nisa Bostanci, Nandita Vijaykumar, Oguz Ergin, and Onur Mutlu,  
**"QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips"**  
*Proceedings of the 48th International Symposium on Computer Architecture (ISCA)*, Virtual, June 2021.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Short Talk Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (25 minutes)]  
[[SAFARI Live Seminar Video](#) (1 hr 26 mins)]

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

Ataberk Olgun<sup>§†</sup>   Minesh Patel<sup>§</sup>   A. Giray Yağlıkçı<sup>§</sup>   Haocong Luo<sup>§</sup>  
Jeremie S. Kim<sup>§</sup>   F. Nisa Bostancı<sup>§†</sup>   Nandita Vijaykumar<sup>§⊙</sup>   Oğuz Ergin<sup>†</sup>   Onur Mutlu<sup>§</sup>  
<sup>§</sup>ETH Zürich   <sup>†</sup>TOBB University of Economics and Technology   <sup>⊙</sup>University of Toronto

# In-DRAM True Random Number Generation

---

- F. Nisa Bostanci, Ataberk Olgun, Lois Orosa, A. Giray Yaglikci, Jeremie S. Kim, Hasan Hassan, Oguz Ergin, and Onur Mutlu,

## **"DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators"**

*Proceedings of the 28th International Symposium on High-Performance Computer Architecture (HPCA)*, Virtual, April 2022.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Short Talk Slides \(pptx\)](#) ([pdf](#))]

## **DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators**

F. Nisa Bostanci<sup>†§</sup>

Ataberk Olgun<sup>†§</sup>

Lois Orosa<sup>§</sup>

A. Giray Yağlıkçı<sup>§</sup>

Jeremie S. Kim<sup>§</sup>

Hasan Hassan<sup>§</sup>

Oğuz Ergin<sup>†</sup>

Onur Mutlu<sup>§</sup>

<sup>†</sup>*TOBB University of Economics and Technology*

<sup>§</sup>*ETH Zürich*

# DRAM Chips Are Already (Quite) Capable!

---

- **Appears at HPCA 2024**    <https://arxiv.org/pdf/2402.18736.pdf>

## Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis

İsmail Emir Yüksel    Yahya Can Tuğrul    Ataberk Olgun    F. Nisa Bostancı    A. Giray Yağlıkçı  
Geraldo F. Oliveira    Haocong Luo    Juan Gómez-Luna    Mohammad Sadrosadati    Onur Mutlu

ETH Zürich

*We experimentally demonstrate that COTS DRAM chips are capable of performing 1) functionally-complete Boolean operations: NOT, NAND, and NOR and 2) many-input (i.e., more than two-input) AND and OR operations. We present an extensive characterization of new bulk bitwise operations in 256 off-the-shelf modern DDR4 DRAM chips. We evaluate the reliability of these operations using a metric called success rate: the fraction of correctly performed bitwise operations. Among our 19 new observations, we highlight four major results. First, we can perform the NOT operation on COTS DRAM chips with 98.37% success rate on average. Second, we can perform up to 16-input NAND, NOR, AND, and OR operations on COTS DRAM chips with high reliability (e.g., 16-input NAND, NOR, AND, and OR with average success rate of 94.94%, 95.87%, 94.94%, and 95.85%, respectively). Third, data pattern only slightly*

# DRAM Chips Are Already (Quite) Capable!

---

- <https://arxiv.org/pdf/2312.02880.pdf>

## **PULSAR: Simultaneous Many-Row Activation for Reliable and High-Performance Computing in Off-the-Shelf DRAM Chips**

Ismail Emir Yuksel   Yahya Can Tugrul   F. Nisa Bostanci   Abdullah Giray Yaglikci   Ataberk Olgun  
Geraldo F. Oliveira   Melina Soysal   Haocong Luo   Juan Gomez Luna   Mohammad Sadrosadati  
Onur Mutlu

ETH Zurich

We propose PULSAR, a new technique to enable high-success-rate and high-performance PuM operations in off-the-shelf DRAM chips. PULSAR leverages our new observation that a carefully-crafted sequence of DRAM commands simultaneously activates up to 32 DRAM rows. PULSAR overcomes the limitations of existing techniques by 1) replicating the input data to improve the success rate and 2) enabling new bulk bitwise operations (e.g., many-input majority, *Multi-RowInit*, and *Bulk-Write*) to improve the performance.

# DRAM Chips Are Already (Quite) Capable!

---

- **Appears at DSN 2024**



## **Simultaneous Many-Row Activation in Off-the-Shelf DRAM Chips: Experimental Characterization and Analysis**

İsmail Emir Yüksel<sup>1</sup>   Yahya Can Tuğrul<sup>1,2</sup>   F. Nisa Bostancı<sup>1</sup>   Geraldo F. Oliveira<sup>1</sup>  
A. Giray Yağlıkçı<sup>1</sup>   Ataberk Olgun<sup>1</sup>   Melina Soysal<sup>1</sup>   Haocong Luo<sup>1</sup>  
Juan Gómez-Luna<sup>1</sup>   Mohammad Sadrosadati<sup>1</sup>   Onur Mutlu<sup>1</sup>

<sup>1</sup>*ETH Zürich*   <sup>2</sup>*TOBB University of Economics and Technology*

# The Capability of COTS DRAM Chips

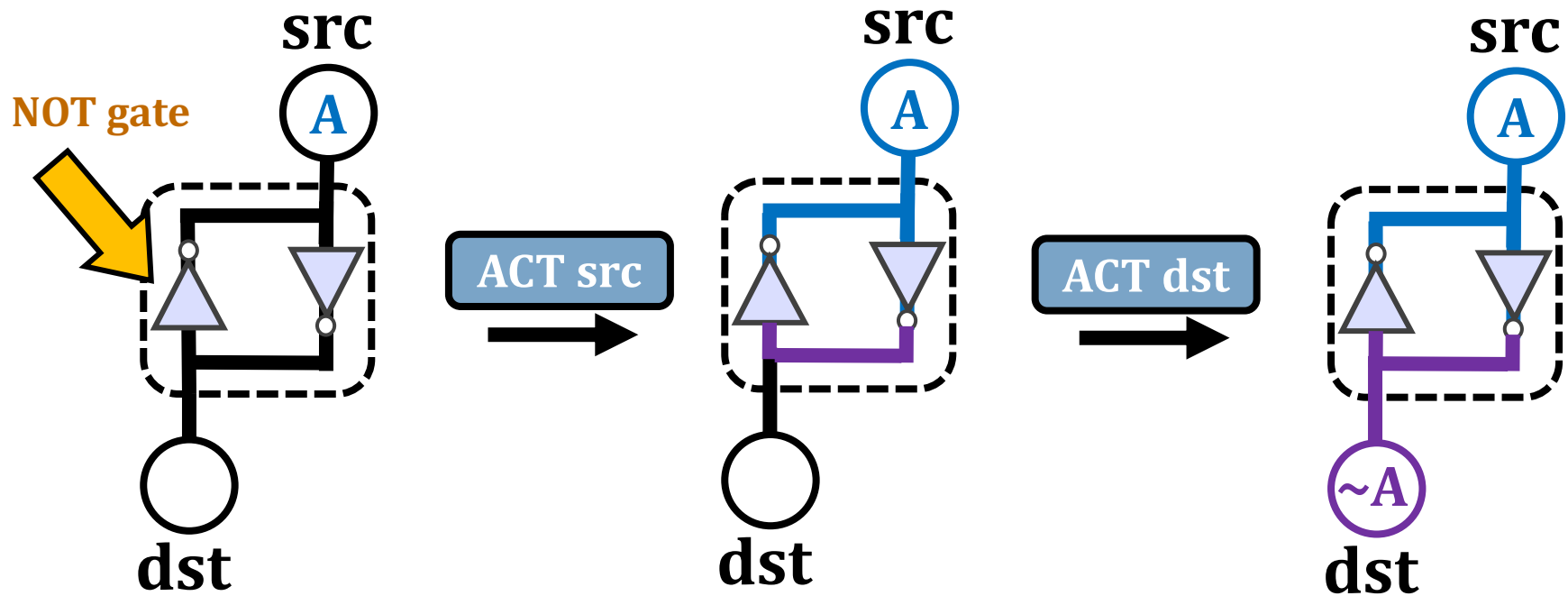
We demonstrate that COTS DRAM chips:

- 1 Can simultaneously activate up to 48 rows in two neighboring subarrays
- 2 Can perform **NOT operation** with up to **32 output operands**
- 3 Can perform up to **16-input AND, NAND, OR, and NOR** operations



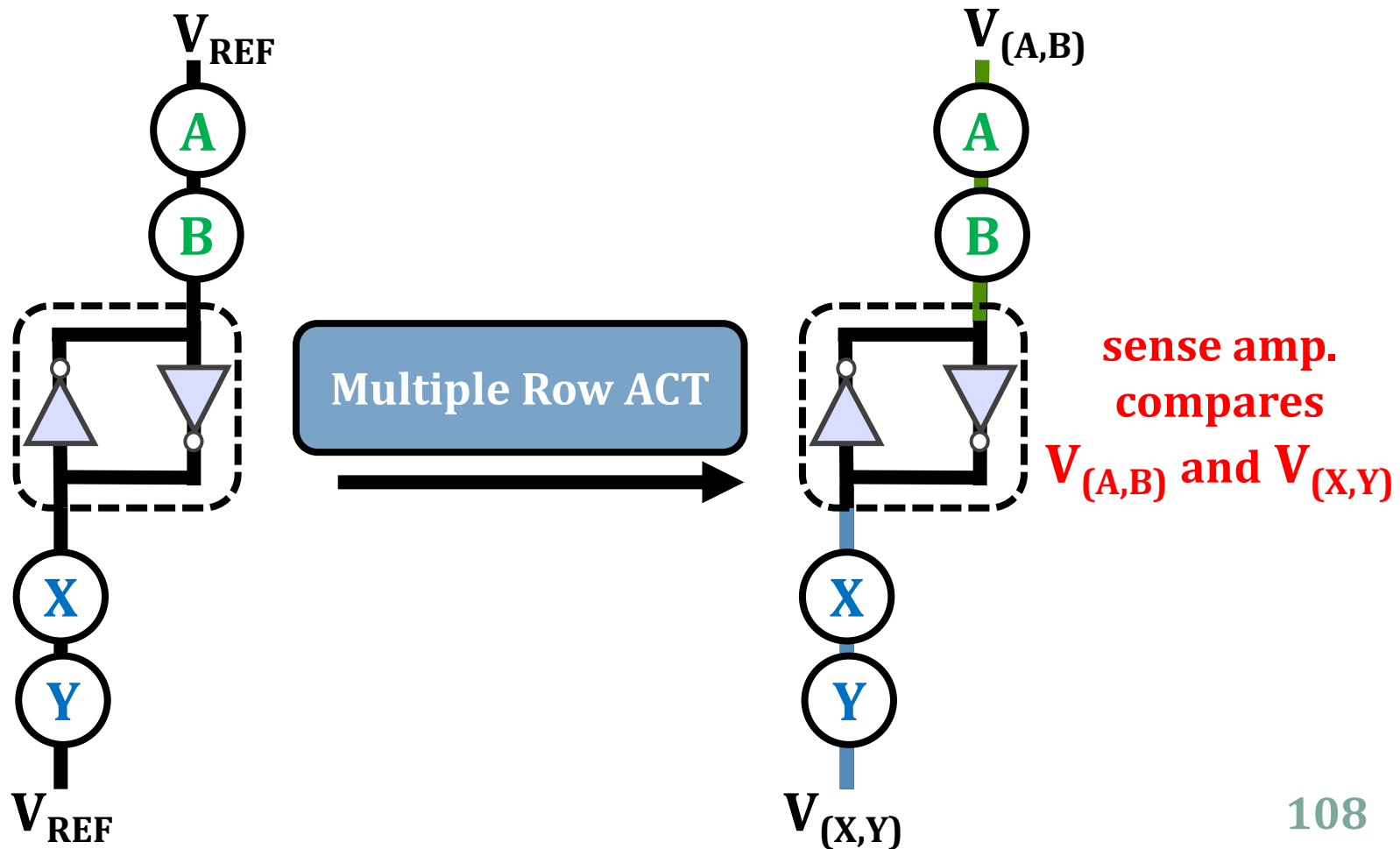
# Key Idea: NOT Operation

Connect rows in neighboring subarrays through a **NOT gate** by simultaneously activating rows



# Key Idea: NAND, NOR, AND, OR

**Manipulate the bitline voltage** to express  
**a wide variety of functions** using  
multiple-row activation in neighboring subarrays



# FC-DRAM: Executive Summary

- **Motivation: Processing-using-DRAM** can alleviate the performance and energy bottlenecks caused by **data movement**
  - **Prior works** show that **existing DRAM chips** can perform **three-input majority** and **two-input AND and OR** operations
- **Problem: Proof-of-concept** demonstrations on **commercial off-the-shelf (COTS)** DRAM chips do not provide
  - **functionally-complete** operations (e.g., NAND or NOR)
  - **NOT operation**
  - **AND** and **OR** operations with **more than two inputs**
- **Experimental Study: 256 DDR4 chips** from **two major manufacturers**
- **Key Results:**
  - COTS DRAM chips can perform **NOT** and **{2, 4, 8, 16}-input AND, NAND, OR, and NOR** operations with **very high reliability (>94% success rate)**
  - **Data pattern** and **temperature** only slightly affect the reliability of these operations (**<1.98% decrease in success rate**)

# In-Flash Bulk Bitwise Execution

---

- Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira, Mohammad Sadrosadati, Rakesh Nadig, David Novo, Juan Gómez-Luna, Myungsuk Kim, and Onur Mutlu, **"Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory"**  
*Proceedings of the 55th International Symposium on Microarchitecture (MICRO)*, Chicago, IL, USA, October 2022.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Longer Lecture Slides \(pptx\)](#)] [[pdf](#)]  
[[Lecture Video](#) (44 minutes)]  
[[arXiv version](#)]

## Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

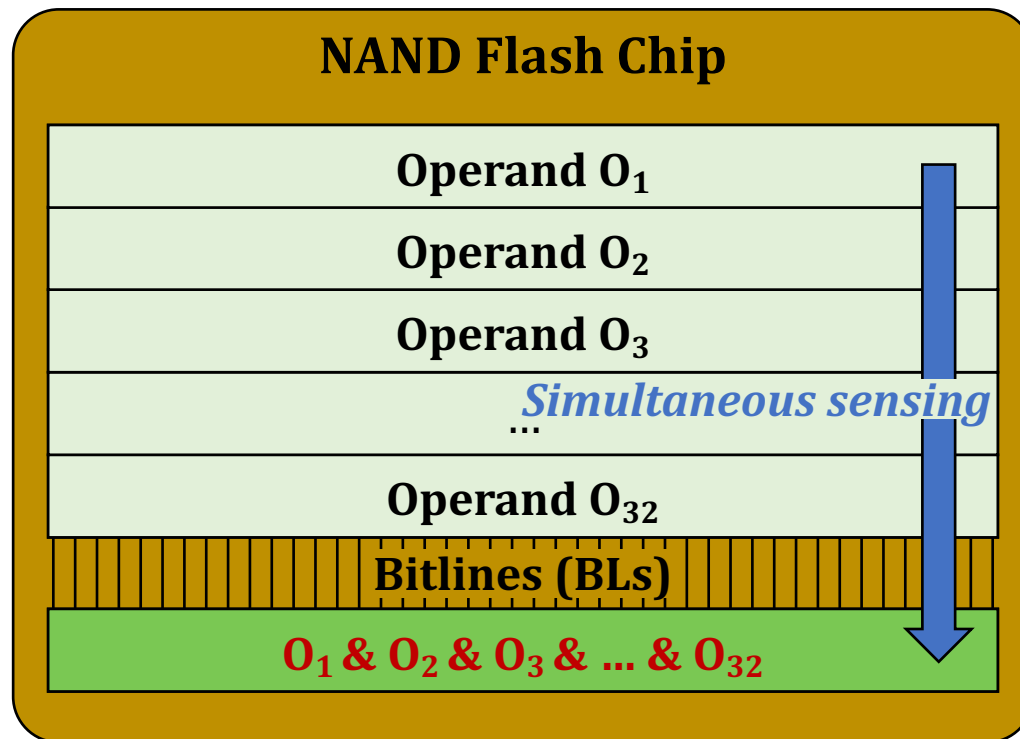
Jisung Park<sup>§∇</sup> Roknoddin Azizi<sup>§</sup> Geraldo F. Oliveira<sup>§</sup> Mohammad Sadrosadati<sup>§</sup>  
Rakesh Nadig<sup>§</sup> David Novo<sup>†</sup> Juan Gómez-Luna<sup>§</sup> Myungsuk Kim<sup>‡</sup> Onur Mutlu<sup>§</sup>

<sup>§</sup>ETH Zürich    <sup>∇</sup>POSTECH    <sup>†</sup>LIRMM, Univ. Montpellier, CNRS    <sup>‡</sup>Kyungpook National University

# Flash-Cosmos: Basic Ideas

## ▪ Flash-Cosmos enables

- Computation on multiple operands with a single sensing operation
- Accurate computation results by eliminating raw bit errors in stored data



# Multi-Wordline Sensing (MWS): Bitwise AND

## ■ Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

A bitline reads as '**1**' only when all the target cells store '**1**'  
→ Equivalent to the bitwise AND of all the target cells

*Operate  
as a resistance (1)  
or an open switch (0)*

WL<sub>2</sub>  
WL<sub>3</sub>  
WL<sub>4</sub>

Result: 0

0

0

0

BL<sub>1</sub>

BL<sub>2</sub>

BL<sub>3</sub>

BL<sub>4</sub>

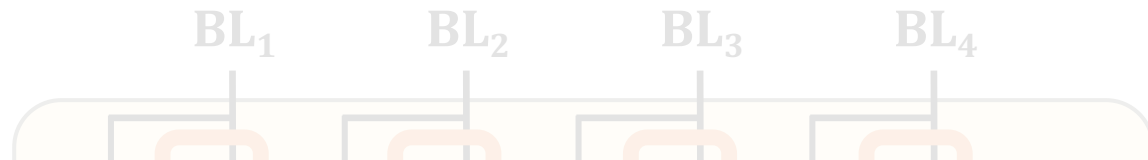


# Multi-Wordline Sensing (MWS): Bitwise AND

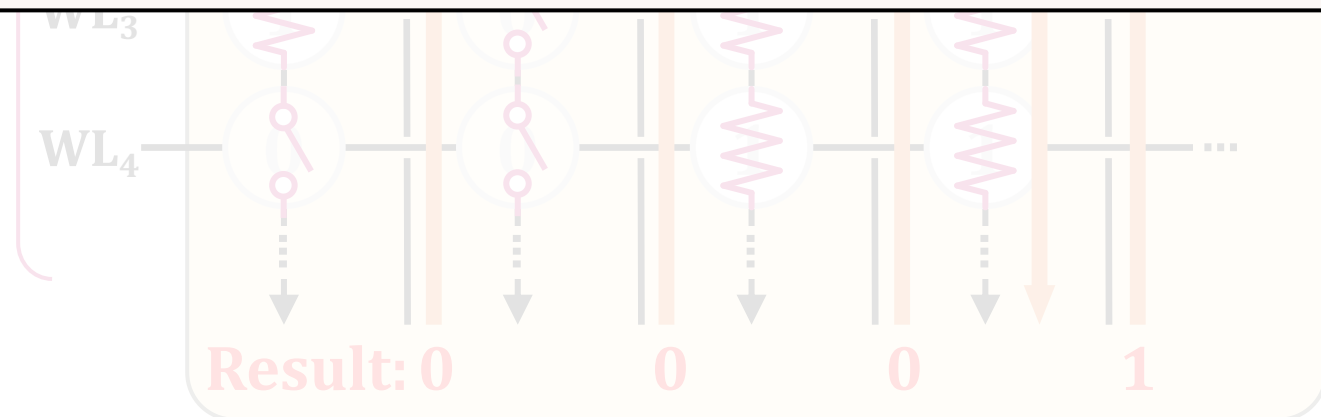
- Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs



**Flash-Cosmos (Intra-Block MWS)** enables bitwise AND of multiple pages in the same block via a single sensing operation



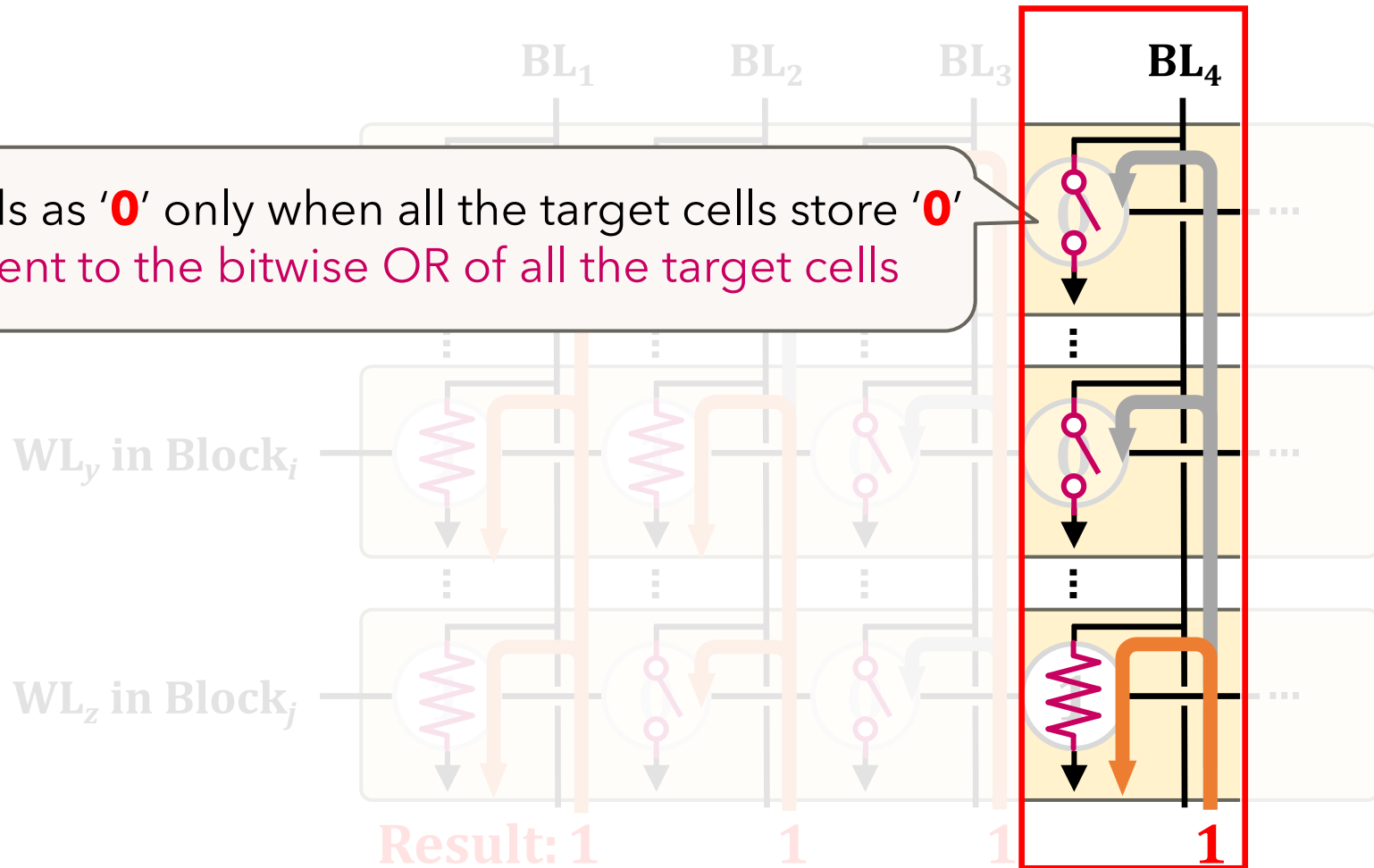
# Multi-Wordline Sensing (MWS): Bitwise OR

- **Inter-Block MWS:**

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs

A bitline reads as '**0**' only when all the target cells store '**0**'  
→ Equivalent to the bitwise OR of all the target cells



# Multi-Wordline Sensing (MWS): Bitwise OR

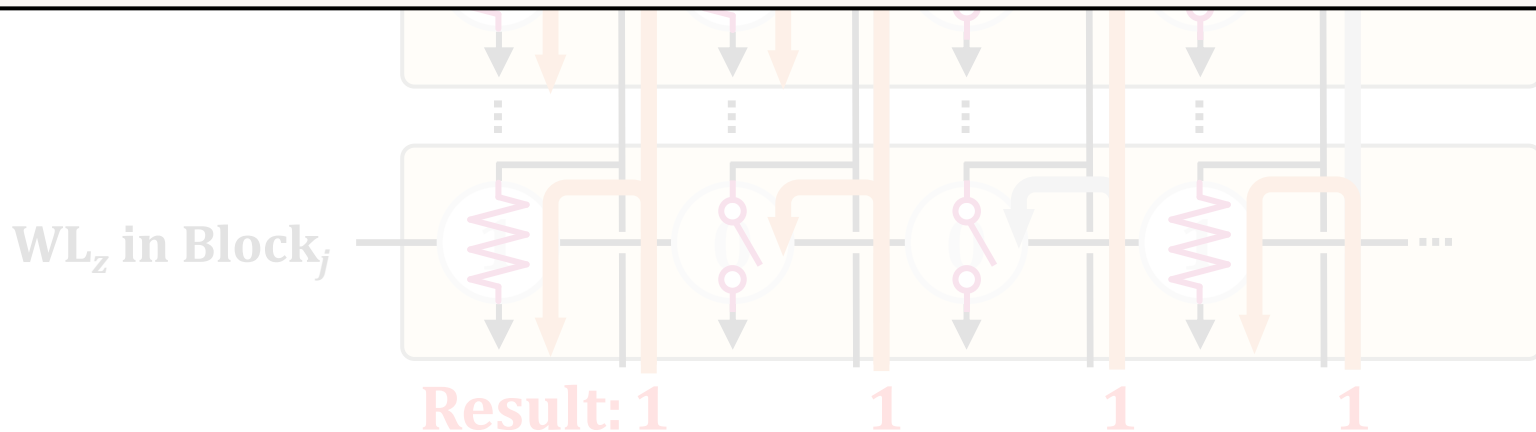
- Inter-Block MWS:

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs



**Flash-Cosmos (Inter-Block MWS)** enables  
bitwise OR of multiple pages in different blocks  
via a single sensing operation



# Other Types of Bitwise Operations

**Flash-Cosmos** also enables  
other types of bitwise operations  
(NOT/NAND/NOR/XOR/XNOR)  
leveraging **existing features** of NAND flash memory

## Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park<sup>§∇</sup> Roknoddin Azizi<sup>§</sup> Geraldo F. Oliveira<sup>§</sup> Mohammad Sadrosadati<sup>§</sup>  
Rakesh Nadig<sup>§</sup> David Novo<sup>†</sup> Juan Gómez-Luna<sup>§</sup> Myungsuk Kim<sup>‡</sup> Onur Mutlu<sup>§</sup>

<sup>§</sup>*ETH Zürich*    <sup>∇</sup>*POSTECH*    <sup>†</sup>*LIRMM, Univ. Montpellier, CNRS*    <sup>‡</sup>*Kyungpook National University*



<https://arxiv.org/abs/2209.05566.pdf>

# Results: Real-Device Characterization

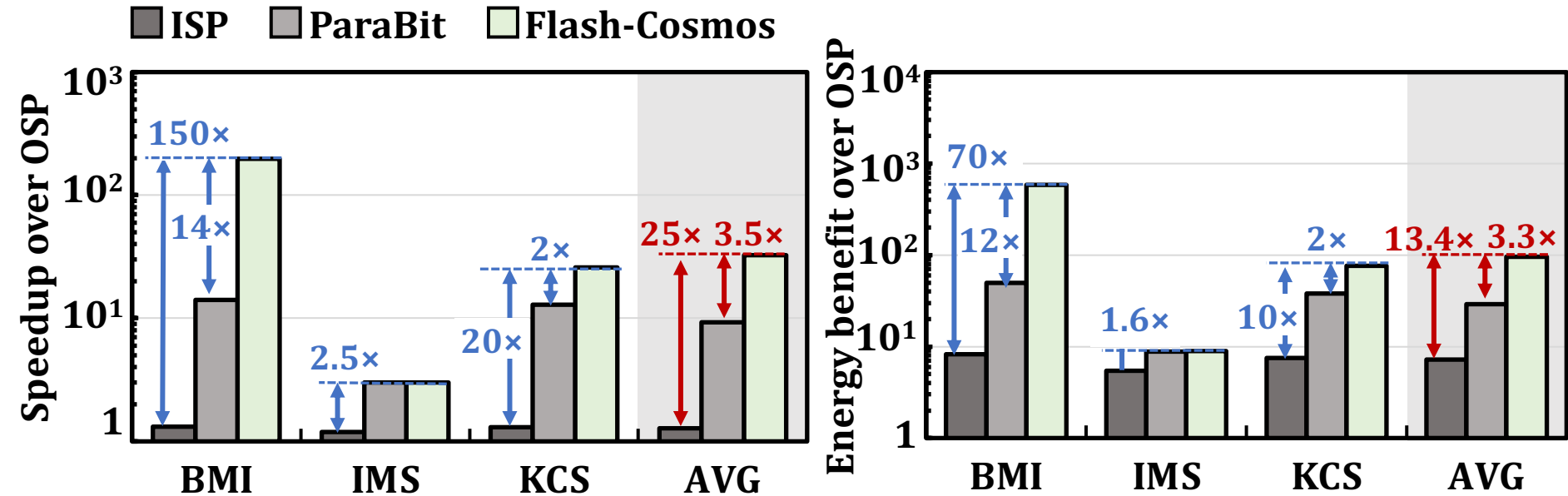
---

No changes to the cell array  
of commodity NAND flash chips

Can have many operands  
(AND: up to 48, OR: up to 4)  
with small increase in sensing latency (< 10%)

ESP significantly improves  
the reliability of computation results  
(no observed bit error in the tested flash cells)

# Results: Performance & Energy



Flash-Cosmos provides **significant performance & energy benefits** over all the baselines

The larger the number of operands,  
the higher the performance & energy benefits



# Pinatubo: RowClone and Bitwise Ops in PCM

---

## **Pinatubo: A Processing-in-Memory Architecture for Bulk Bitwise Operations in Emerging Non-volatile Memories**

Shuangchen Li<sup>1\*</sup>, Cong Xu<sup>2</sup>, Qiaosha Zou<sup>1,5</sup>, Jishen Zhao<sup>3</sup>, Yu Lu<sup>4</sup>, and Yuan Xie<sup>1</sup>

University of California, Santa Barbara<sup>1</sup>, Hewlett Packard Labs<sup>2</sup>

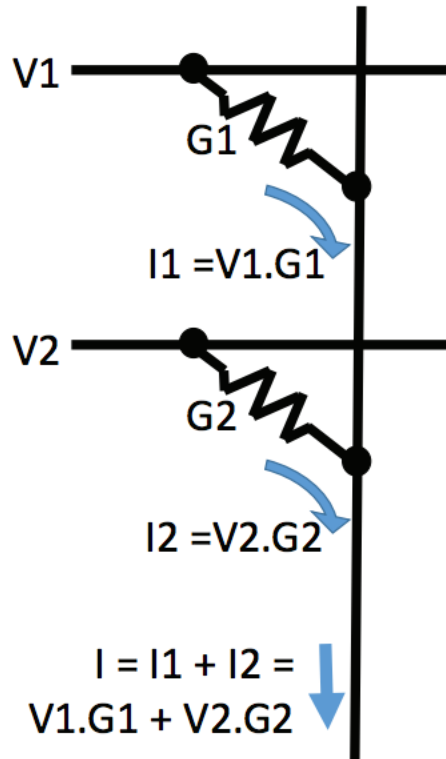
University of California, Santa Cruz<sup>3</sup>, Qualcomm Inc.<sup>4</sup>, Huawei Technologies Inc.<sup>5</sup>  
{shuangchenli, yuanxie}@ece.ucsb.edu<sup>1</sup>

# Other Readings on Processing using NVM

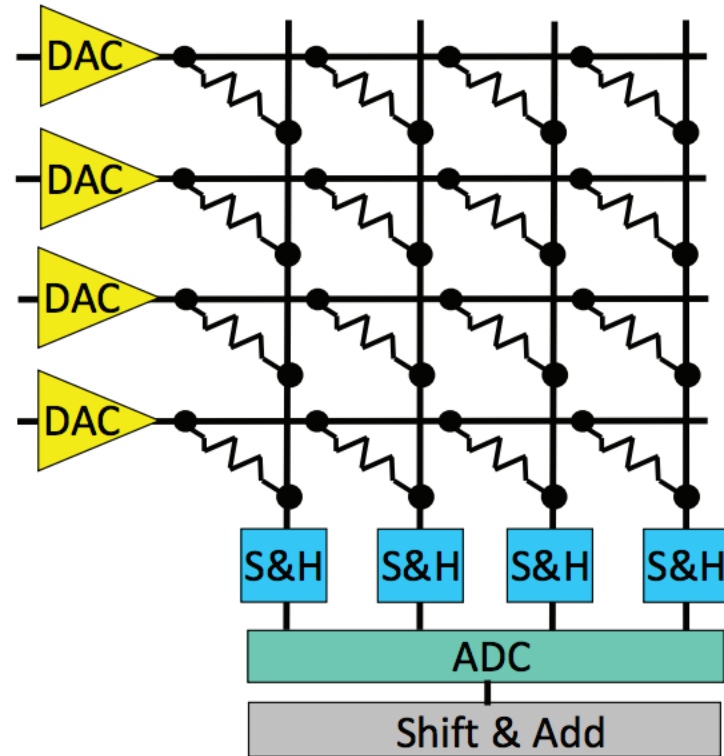
---

- Shafiee+, “ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars”, ISCA 2016.
- Chi+, “PRIME: A Novel Processing-in-memory Architecture for Neural Network Computation in ReRAM-based Main Memory”, ISCA 2016.
- Prezioso+, “Training and Operation of an Integrated Neuromorphic Network based on Metal-Oxide Memristors”, Nature 2015
- Ambrogio+, “Equivalent-accuracy accelerated neural-network training using analogue memory”, Nature 2018.

# In-Memory Crossbar Computation



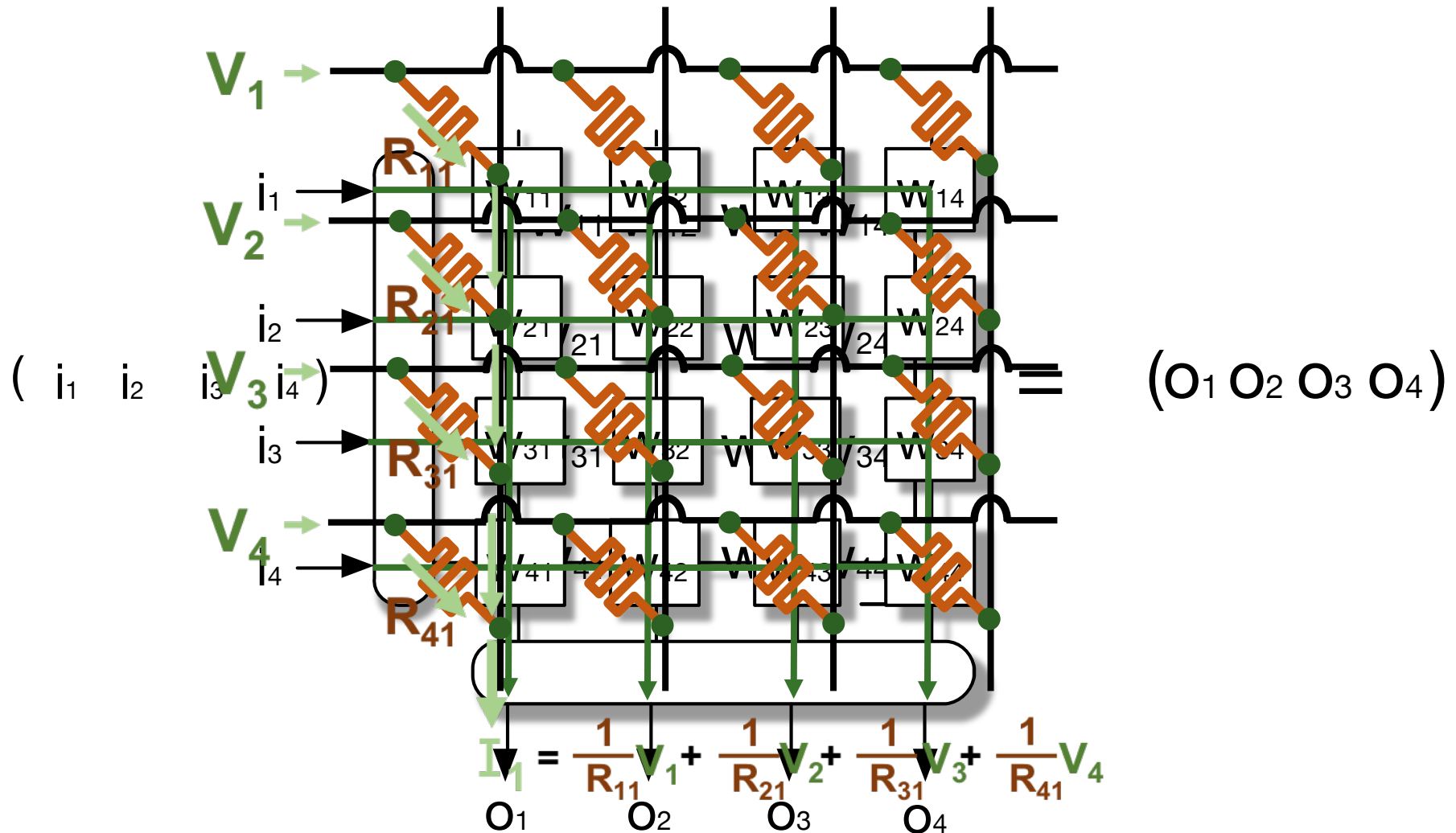
(a) Multiply-Accumulate operation



(b) Vector-Matrix Multiplier

Fig. 1. (a) Using a bitline to perform an analog sum of products operation. (b) A memristor crossbar used as a vector-matrix multiplier.

# In-Memory Crossbar Computation



# Processing in Memory: Two Approaches

1. Processing using Memory
2. **Processing near Memory**

# PIM Review and Open Problems

---

## A Modern Primer on Processing in Memory

Onur Mutlu<sup>a,b</sup>, Saugata Ghose<sup>b,c</sup>, Juan Gómez-Luna<sup>a</sup>, Rachata Ausavarungnirun<sup>d</sup>

*SAFARI Research Group*

<sup>a</sup>*ETH Zürich*

<sup>b</sup>*Carnegie Mellon University*

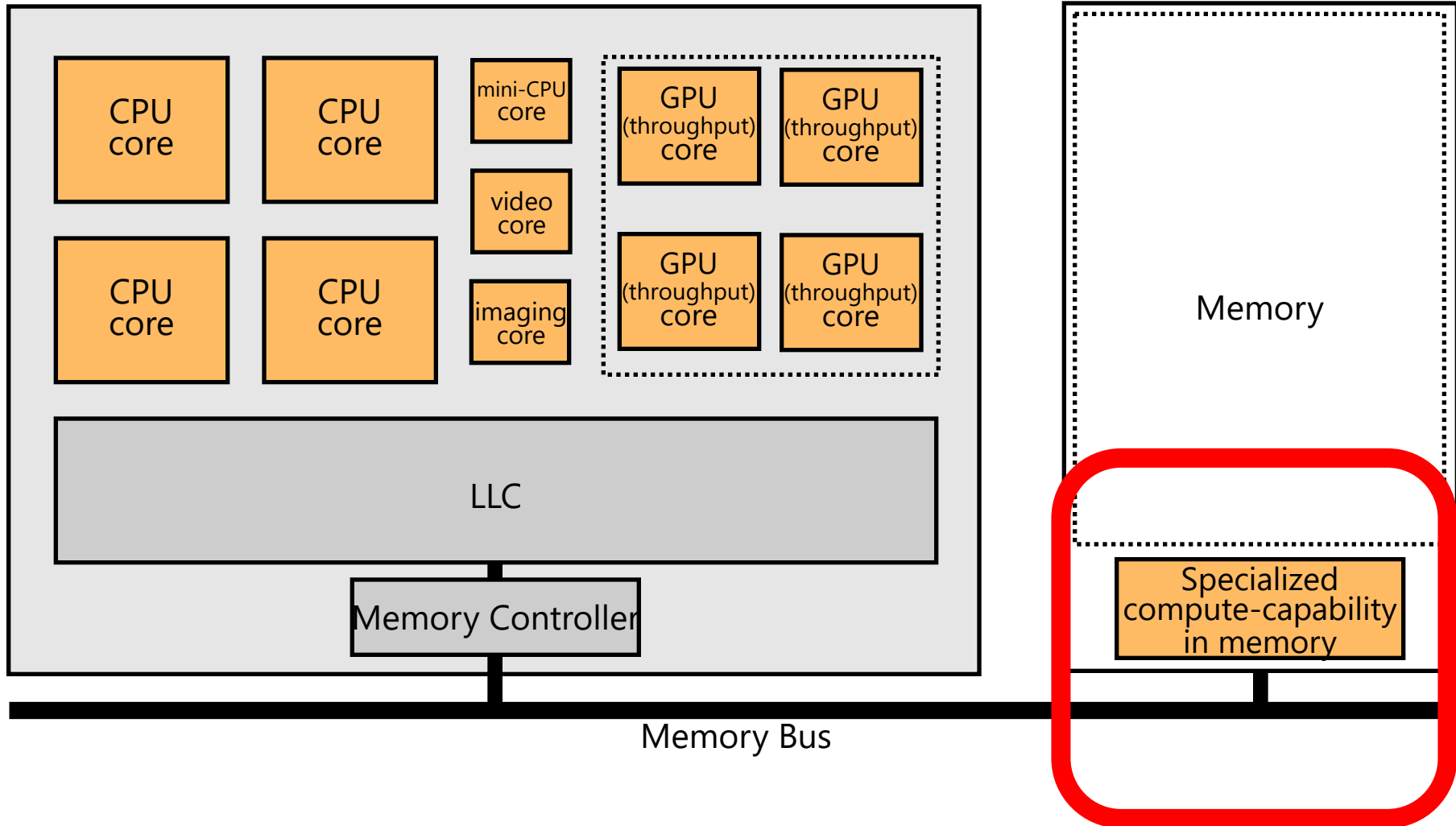
<sup>c</sup>*University of Illinois at Urbana-Champaign*

<sup>d</sup>*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.*



# Mindset: Memory as an Accelerator



**Memory similar to a "conventional" accelerator**

# Accelerating In-Memory Graph Analytics

- Large graphs are everywhere (circa 2015)



36 Million  
Wikipedia Pages



1.4 Billion  
Facebook Users

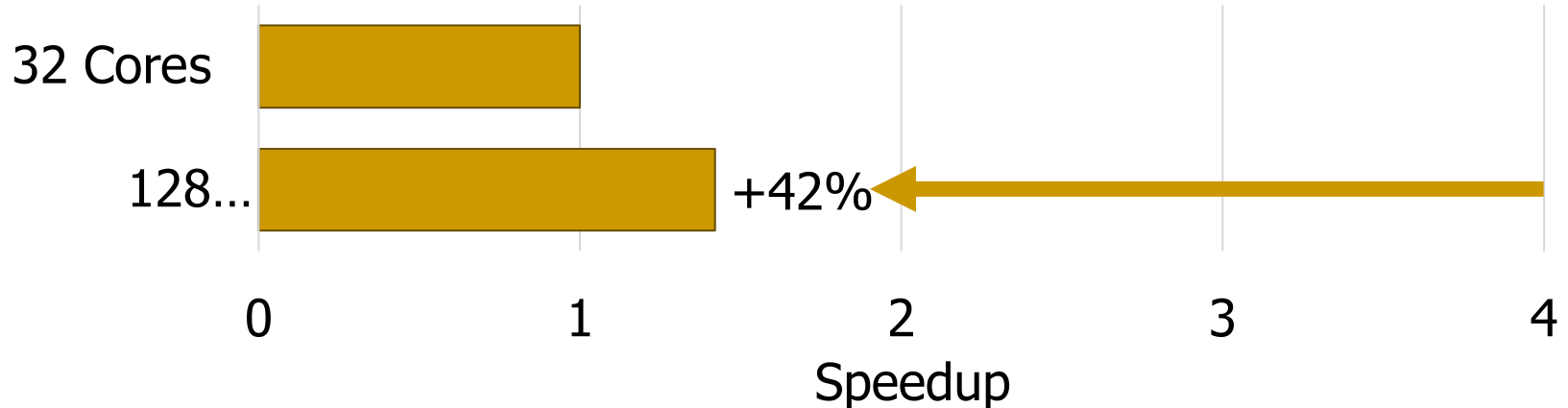


300 Million  
Twitter Users



30 Billion  
Instagram Photos

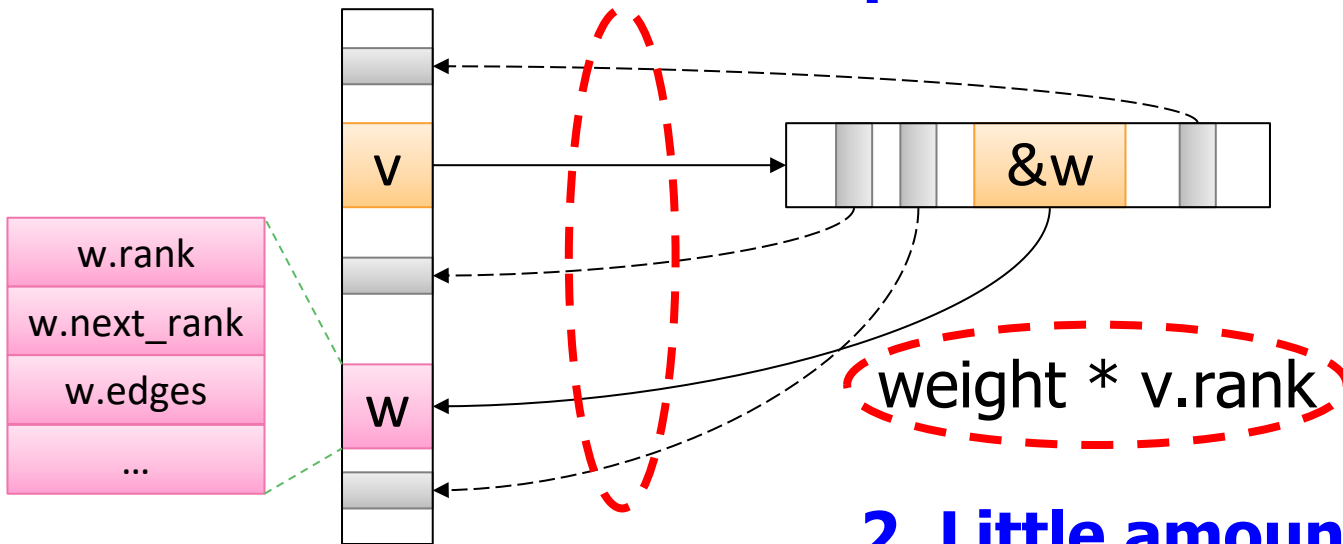
- Scalable large-scale graph processing is challenging



# Key Bottlenecks in Graph Processing

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

**1. Frequent random memory accesses**



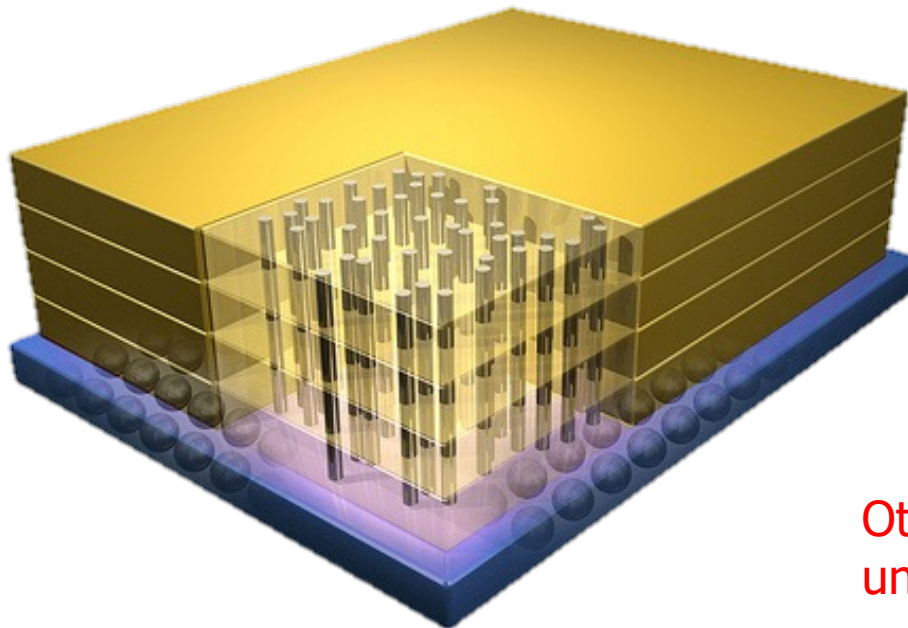
**2. Little amount of computation**

# Opportunity: 3D-Stacked Logic+Memory

---



Hybrid Memory Cube  
C O N S O R T I U M



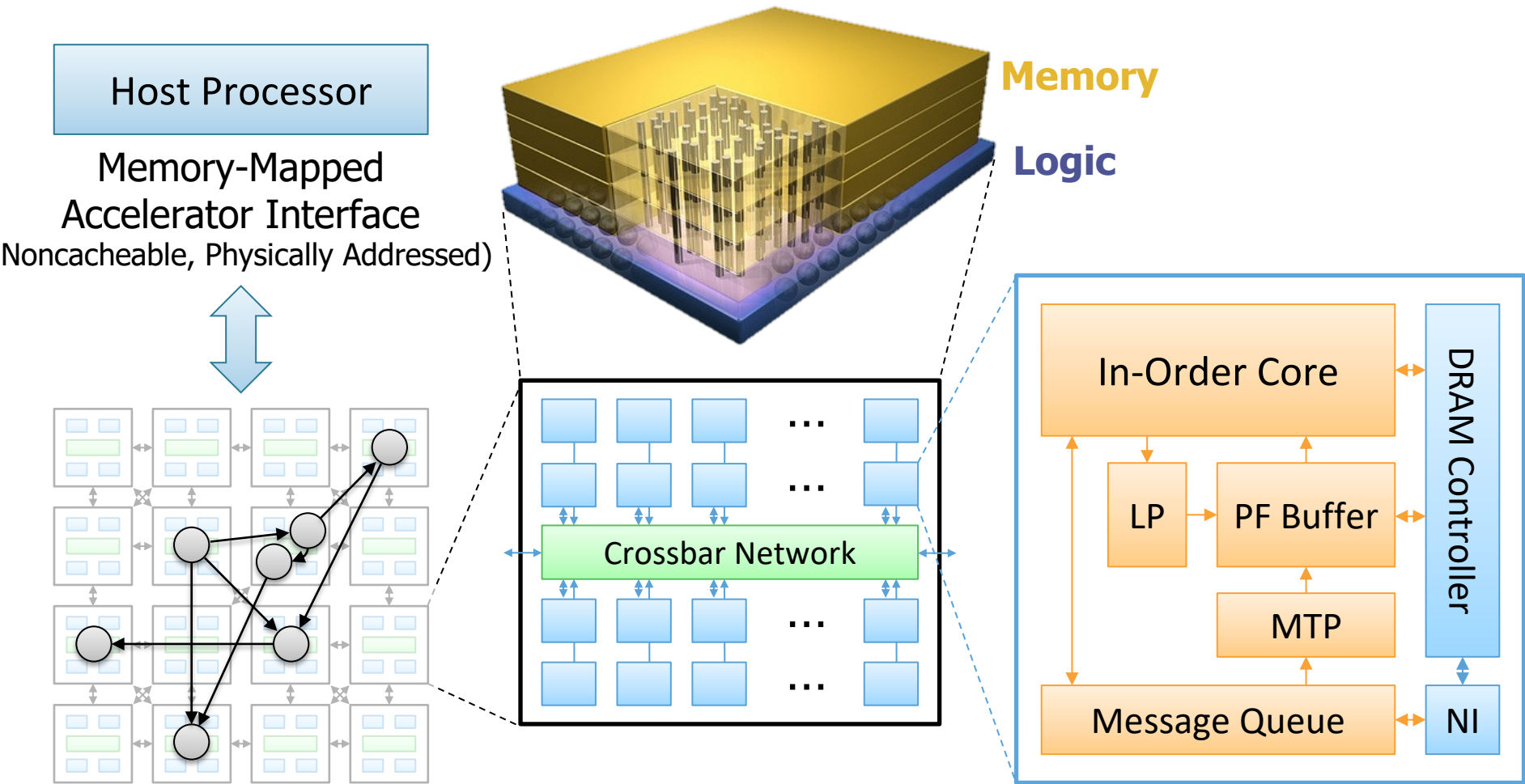
Memory

Logic

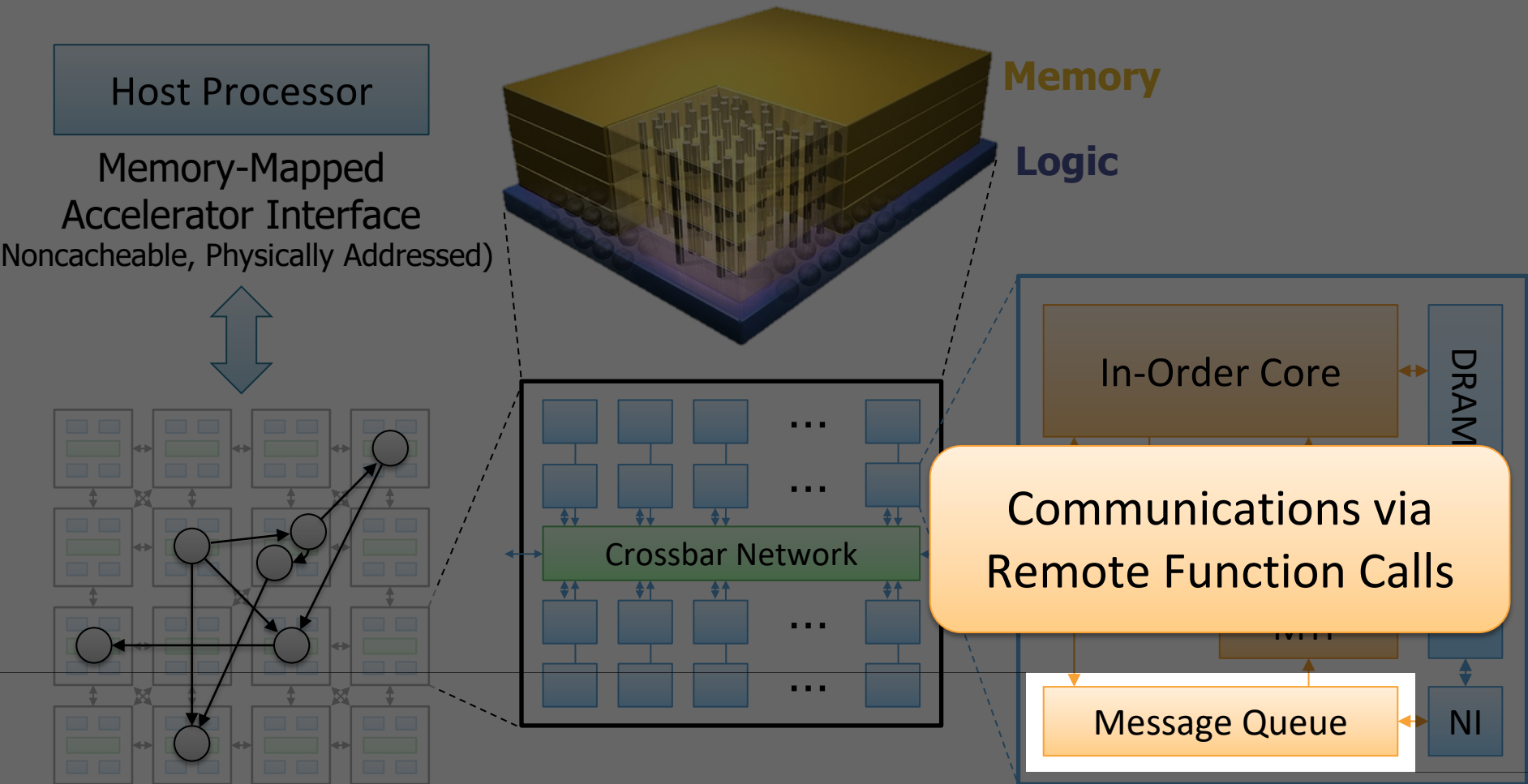
Other "True 3D" technologies  
under development

# Tesseract System for Graph Processing

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

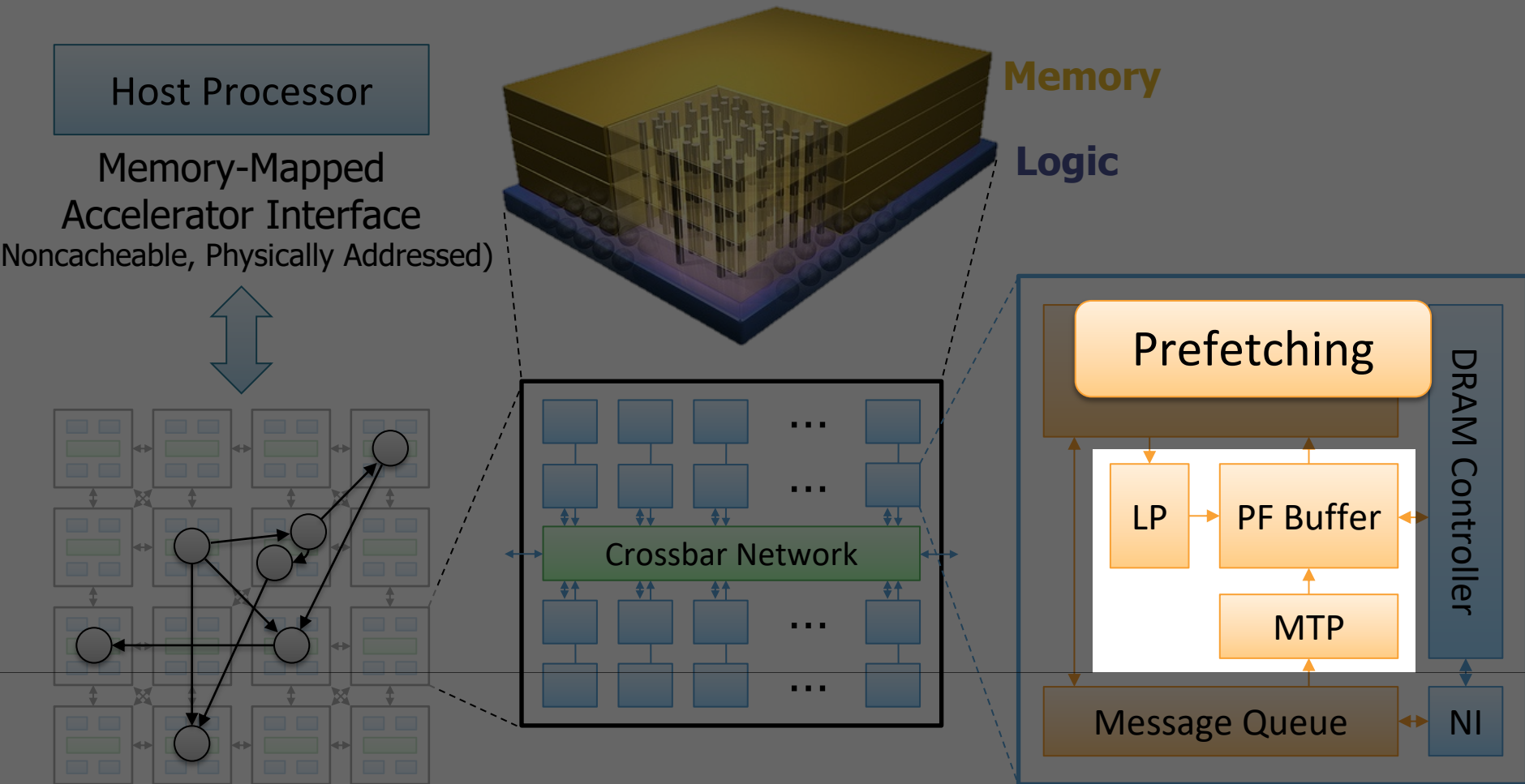


# Tesseract System for Graph Processing



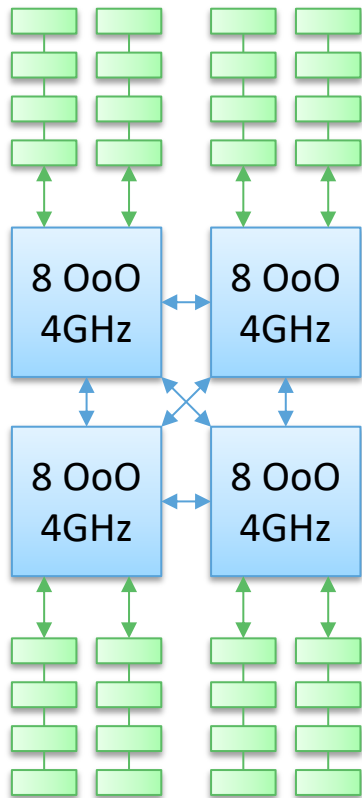


# Tesseract System for Graph Processing



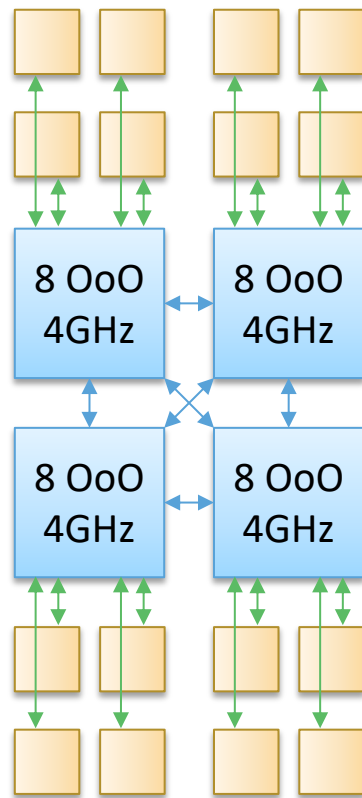
# Evaluated Systems

DDR3-OoO



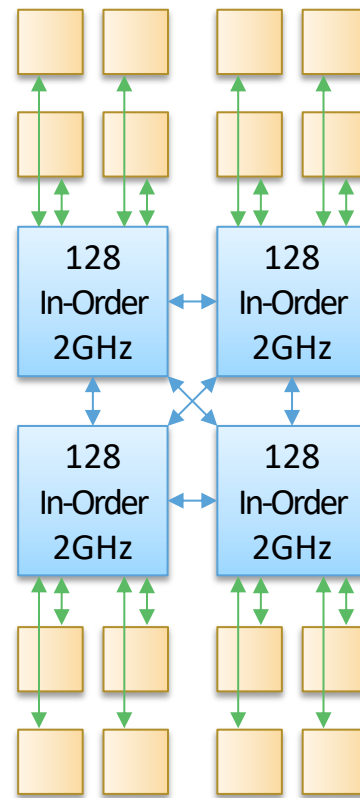
102.4GB/s

HMC-OoO



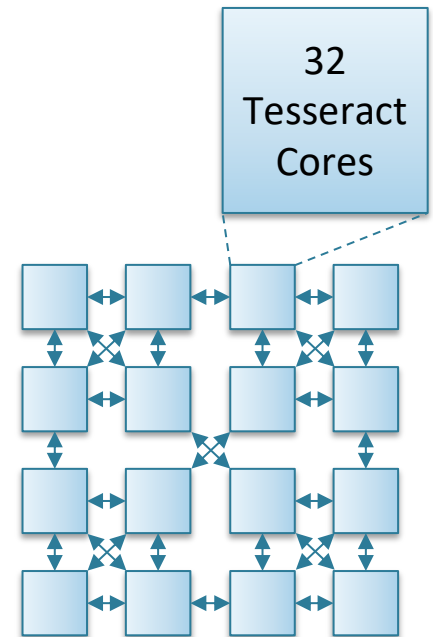
640GB/s

HMC-MC



640GB/s

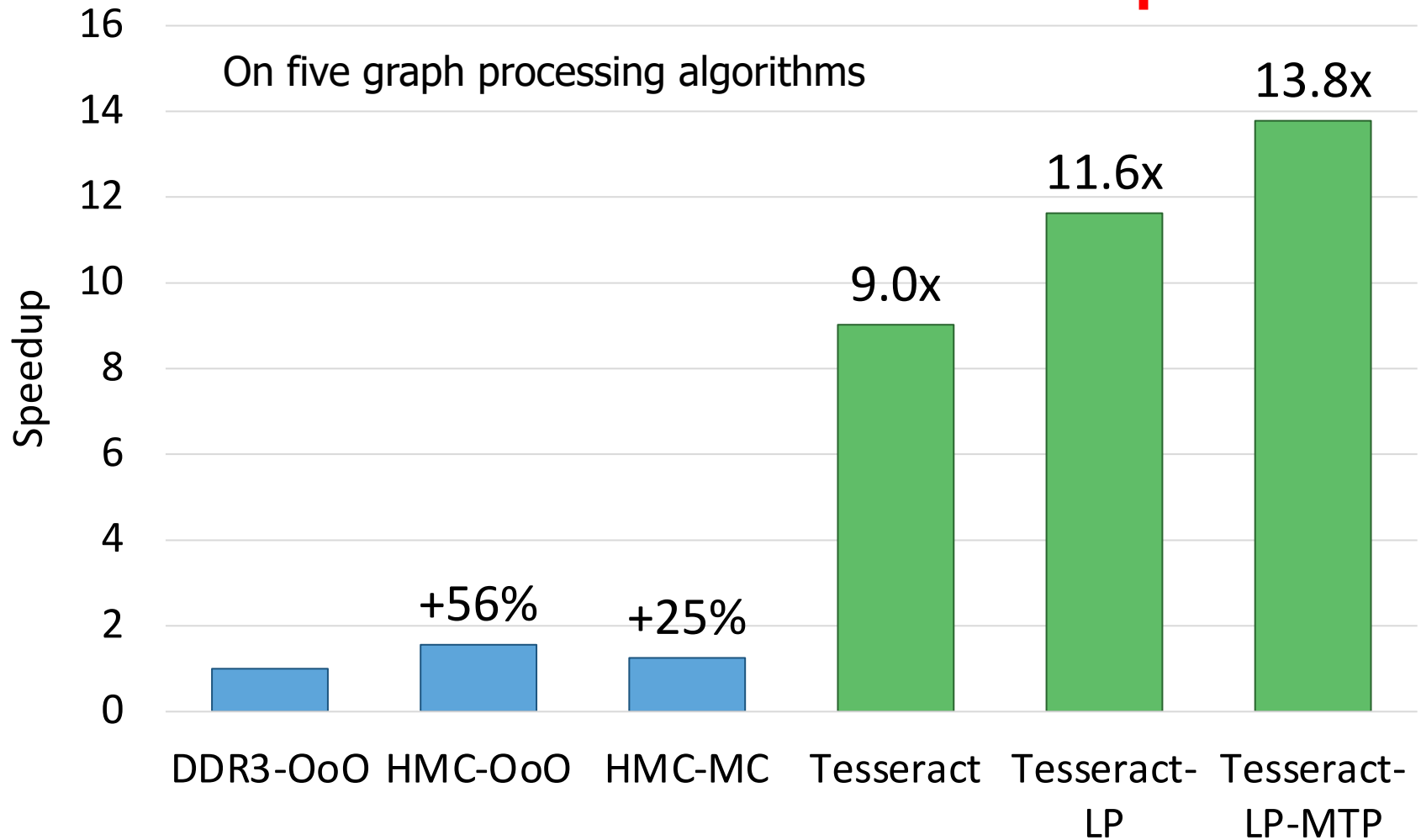
**Tesseract**



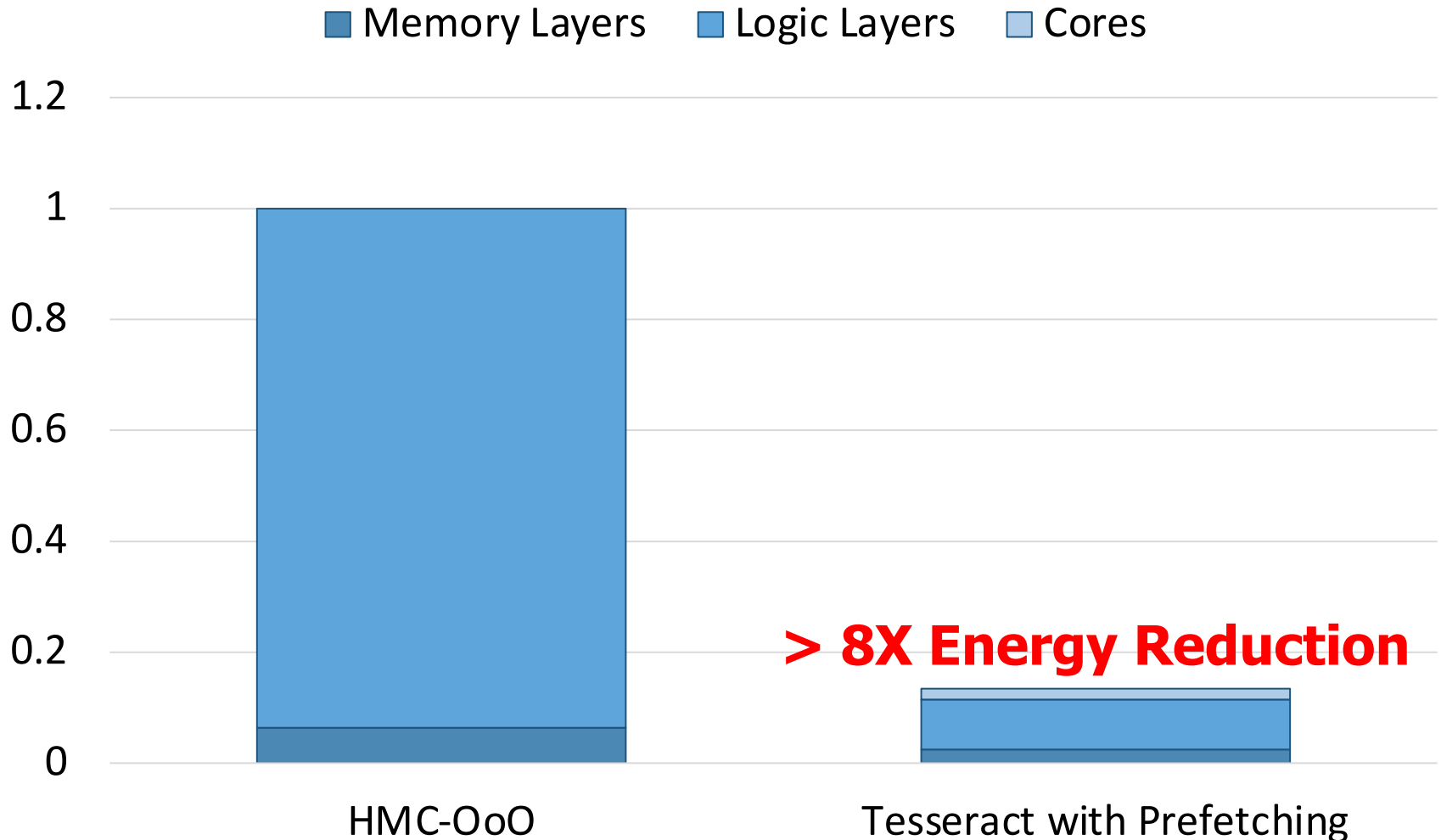
**8TB/s**

# Tesseract Graph Processing Performance

**>13X Performance Improvement**



# Tesseract Graph Processing System Energy



# More on Tesseract

---

- Junwhan Ahn, Sungpack Hong, Sungjoo Yoo, Onur Mutlu, and Kiyoun 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 (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)]*  
***Top Picks Honorable Mention by IEEE Micro.***  
***Selected to the ISCA-50 25-Year Retrospective Issue covering 1996-2020 in 2023 (Retrospective (pdf) Full Issue).***

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

Junwhan Ahn   Sungpack Hong<sup>§</sup>   Sungjoo Yoo   Onur Mutlu<sup>†</sup>   Kiyoun Choi  
junwhan@snu.ac.kr, sungpack.hong@oracle.com, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr

Seoul National University   <sup>§</sup>Oracle Labs   <sup>†</sup>Carnegie Mellon University

# Accelerating Graph Pattern Mining

- Maciej Besta, Raghavendra Kanakagiri, Grzegorz Kwasniewski, Rachata Ausavarungnirun, Jakub Beránek, Konstantinos Kanellopoulos, Kacper Janda, Zur Vonarburg-Shmaria, Lukas Gianinazzi, Ioana Stefan, Juan Gómez-Luna, Marcin Copik, Lukas Kapp-Schwoerer, Salvatore Di Girolamo, Nils Blach, Marek Konieczny, Onur Mutlu, and Torsten Hoefler,

## **"SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems"**

*Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.*

[[Slides \(pdf\)](#)]

[[Talk Video](#) (22 minutes)]

[[Lightning Talk Video](#) (1.5 minutes)]

[[Full arXiv version](#)]

## **SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems**

Maciej Besta<sup>1</sup>, Raghavendra Kanakagiri<sup>2</sup>, Grzegorz Kwasniewski<sup>1</sup>, Rachata Ausavarungnirun<sup>3</sup>, Jakub Beránek<sup>4</sup>, Konstantinos Kanellopoulos<sup>1</sup>, Kacper Janda<sup>5</sup>, Zur Vonarburg-Shmaria<sup>1</sup>, Lukas Gianinazzi<sup>1</sup>, Ioana Stefan<sup>1</sup>, Juan Gómez-Luna<sup>1</sup>, Marcin Copik<sup>1</sup>, Lukas Kapp-Schwoerer<sup>1</sup>, Salvatore Di Girolamo<sup>1</sup>, Nils Blach<sup>1</sup>, Marek Konieczny<sup>5</sup>, Onur Mutlu<sup>1</sup>, Torsten Hoefler<sup>1</sup>

<sup>1</sup>ETH Zurich, Switzerland  
Thailand

<sup>2</sup>IIT Tirupati, India

<sup>3</sup>King Mongkut's University of Technology North Bangkok,  
<sup>4</sup>Technical University of Ostrava, Czech Republic

<sup>5</sup>AGH-UST, Poland

# PIM for 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.*

[[Slides \(pptx\) \(pdf\)](#)] [[Lightning Session Slides \(pptx\) \(pdf\)](#)] [[Poster \(pptx\) \(pdf\)](#)]

[[Lightning Talk Video](#) (2 minutes)]

[[Full Talk Video](#) (21 minutes)]

## **Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks**

Amirali Boroumand<sup>1</sup>

Saugata Ghose<sup>1</sup>

Youngsok Kim<sup>2</sup>

Rachata Ausavarungnirun<sup>1</sup>

Eric Shiu<sup>3</sup>

Rahul Thakur<sup>3</sup>

Daehyun Kim<sup>4,3</sup>

Aki Kuusela<sup>3</sup>

Allan Knies<sup>3</sup>

Parthasarathy Ranganathan<sup>3</sup>

Onur Mutlu<sup>5,1</sup>



# Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

**Amirali Boroumand**

Saugata Ghose, Youngsok Kim, Rachata Ausavarungnirun,  
Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela,  
Allan Knies, Parthasarathy Ranganathan, Onur Mutlu

**SAFARI**

**Carnegie Mellon**

**Google**



SEOUL  
NATIONAL  
UNIVERSITY

**ETH** zürich

# Consumer Devices



**Consumer devices are everywhere!**

**Energy consumption is  
a first-class concern in consumer devices**



# Popular Consumer Workloads



**Chrome**

Google's web browser



**TensorFlow Mobile**

Google's machine learning  
framework

**VP9**



**Video Playback**

Google's **video codec**

**VP9**

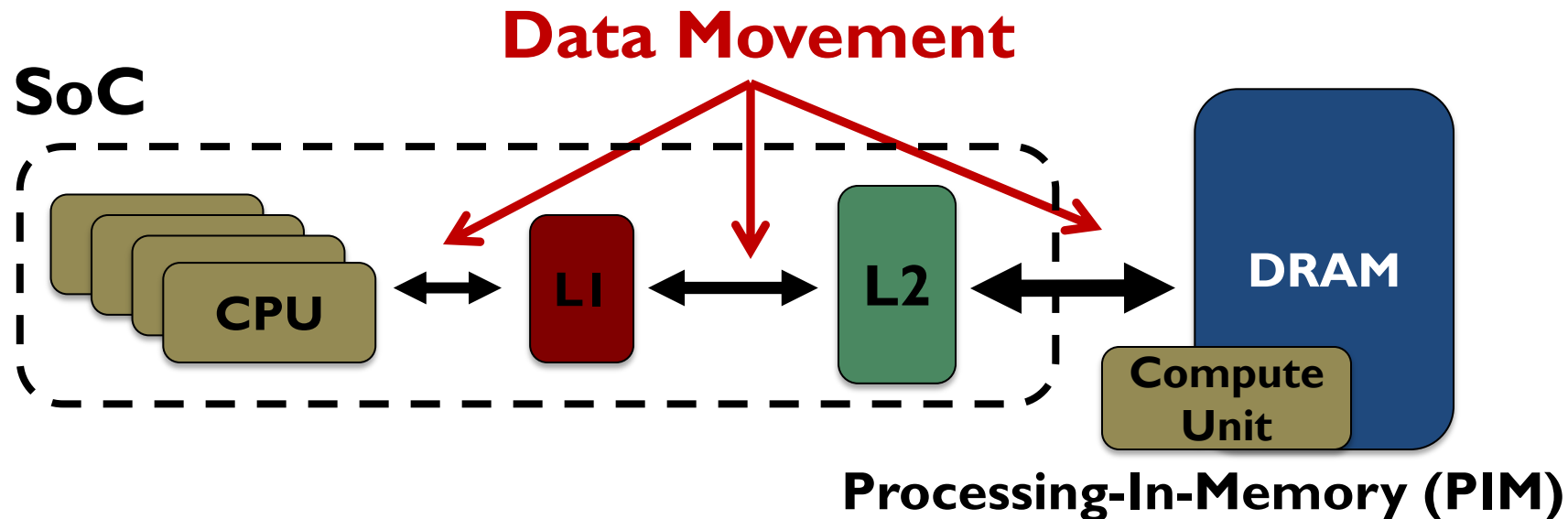


**Video Capture**

Google's **video codec**

# Energy Cost of Data Movement

**1<sup>st</sup> key observation:** **62.7%** of the total system energy is spent on **data movement**



**Potential solution:** move computation **close to data**

**Challenge:** limited area and energy budget

# Using PIM to Reduce Data Movement

**2<sup>nd</sup> key observation:** a significant fraction of the **data movement** often comes from **simple functions**

We can design lightweight logic to implement these simple functions in **memory**

Small embedded  
low-power core



Small fixed-function  
accelerators



Offloading to PIM logic reduces energy and improves performance, on average, by 2.3X and 2.2X

# Workload Analysis



**Chrome**

Google's web browser



**TensorFlow Mobile**

Google's machine learning  
framework



**Video Playback**

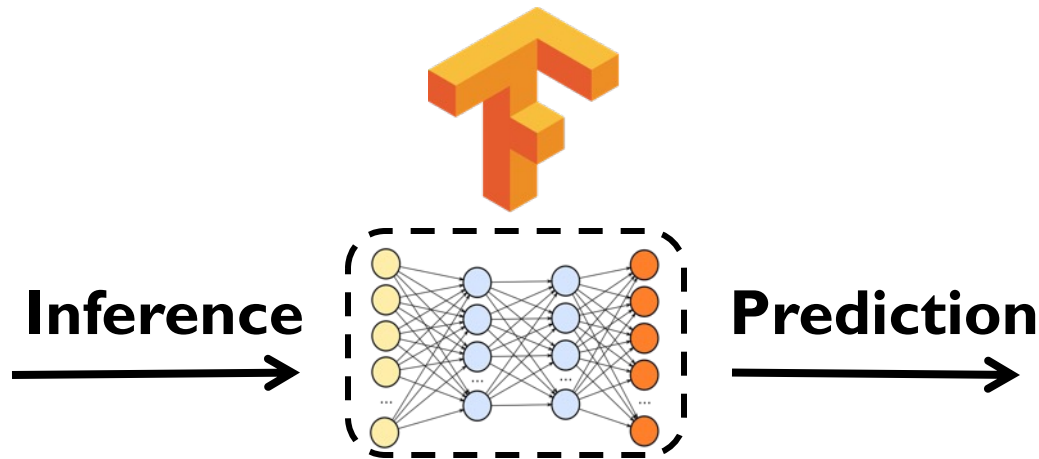
Google's **video codec**



**Video Capture**

Google's **video codec**

# TensorFlow Mobile



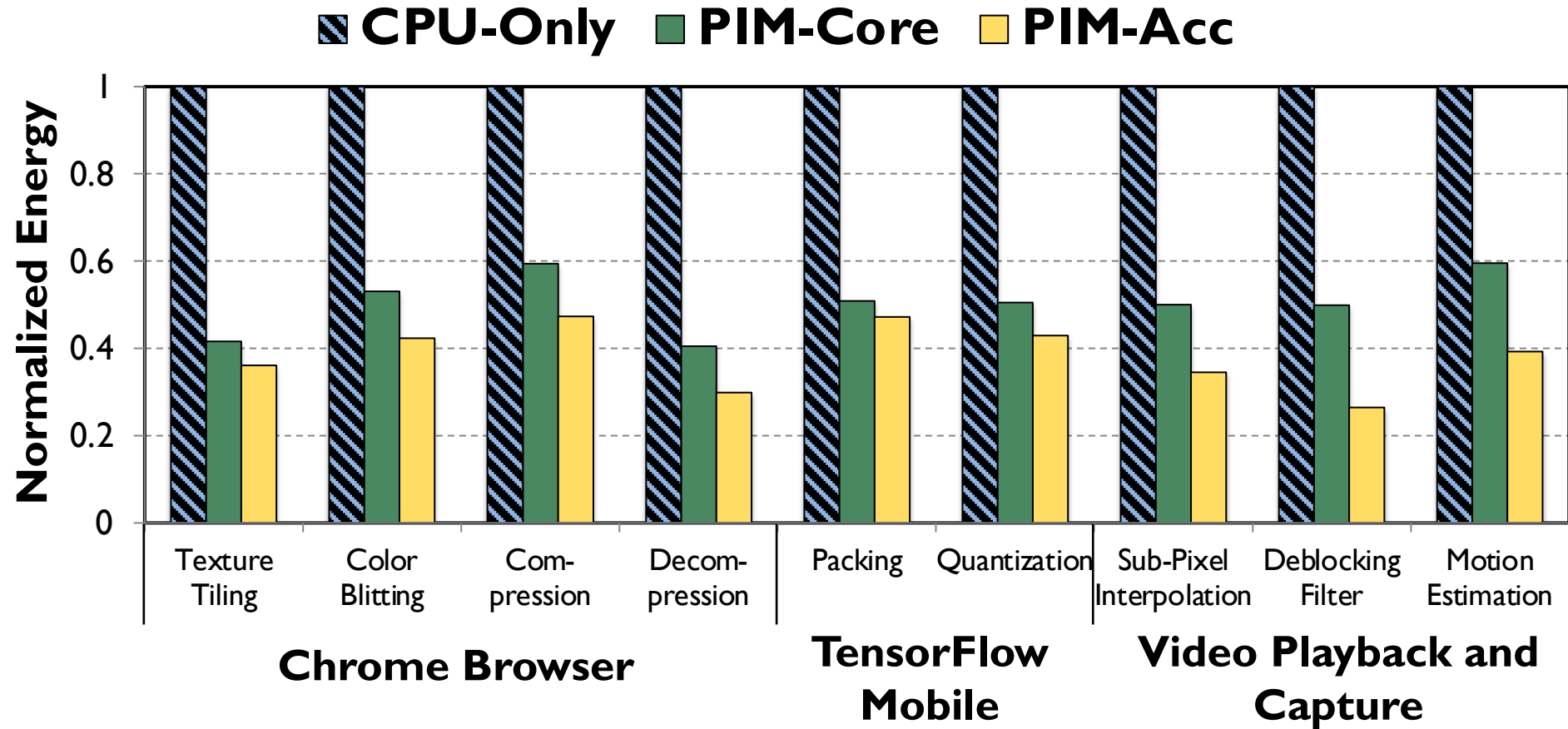
**57.3%** of the inference energy is spent on data movement



**54.4%** of the **data movement** energy comes from packing/unpacking and quantization

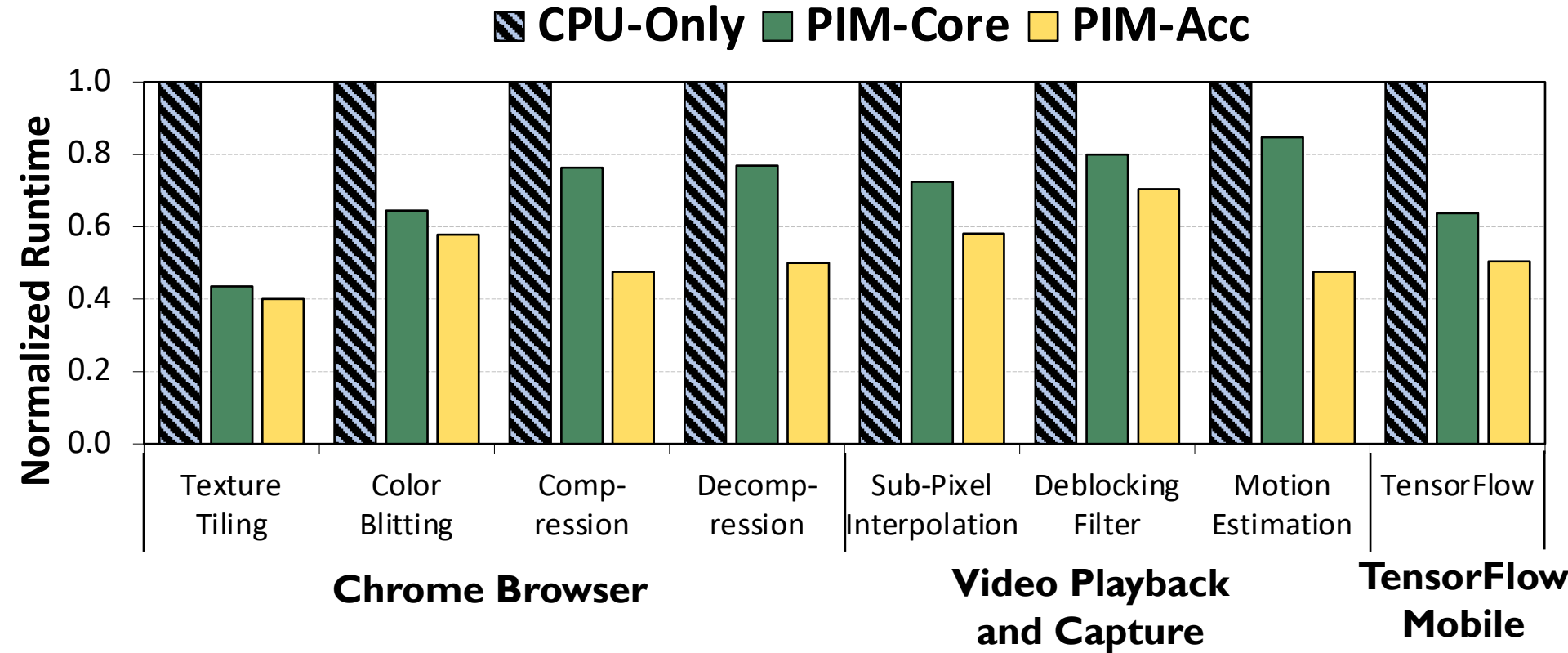


# Normalized Energy



**PIM core and PIM accelerator reduce**  
**energy consumption on average by 49.1% and 55.4%**

# Normalized Runtime



Offloading these kernels to **PIM core** and **PIM accelerator** reduces **program runtime** on average by **44.6%** and **54.2%**

# More on PIM for 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.*

[[Slides \(pptx\) \(pdf\)](#)] [[Lightning Session Slides \(pptx\) \(pdf\)](#)] [[Poster \(pptx\) \(pdf\)](#)]

[[Lightning Talk Video](#) (2 minutes)]

[[Full Talk Video](#) (21 minutes)]

## **Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks**

Amirali Boroumand<sup>1</sup>

Saugata Ghose<sup>1</sup>

Youngsok Kim<sup>2</sup>

Rachata Ausavarungnirun<sup>1</sup>

Eric Shiu<sup>3</sup>

Rahul Thakur<sup>3</sup>

Daehyun Kim<sup>4,3</sup>

Aki Kuusela<sup>3</sup>

Allan Knies<sup>3</sup>

Parthasarathy Ranganathan<sup>3</sup>

Onur Mutlu<sup>5,1</sup>

# Accelerating Neural Network Inference

---

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,  
**"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"**  
*Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Virtual, September 2021.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (14 minutes)]

## Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand<sup>†◇</sup>

Geraldo F. Oliveira<sup>\*</sup>

Saugata Ghose<sup>‡</sup>

Xiaoyu Ma<sup>§</sup>

Berkin Akin<sup>§</sup>

Eric Shiu<sup>§</sup>

Ravi Narayanaswami<sup>§</sup>

Onur Mutlu<sup>\*†</sup>

<sup>†</sup>*Carnegie Mellon Univ.*

<sup>◇</sup>*Stanford Univ.*

<sup>‡</sup>*Univ. of Illinois Urbana-Champaign*

<sup>§</sup>*Google*

<sup>\*</sup>*ETH Zürich*

# Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

**Amirali Boroumand**

**Saugata Ghose**

**Berkin Akin**

**Ravi Narayanaswami**

**Geraldo F. Oliveira**

**Xiaoyu Ma**

**Eric Shiu**

**Onur Mutlu**

**PACT 2021**

**SAFARI**

**Carnegie Mellon**



UNIVERSITY OF  
**ILLINOIS**  
URBANA-CHAMPAIGN



**ETH** zürich

# Executive Summary

**Context:** We extensively analyze a state-of-the-art edge ML accelerator (Google Edge TPU) using 24 Google edge models

- Wide range of models (CNNs, LSTMs, Transducers, RCNNs)

**Problem:** The Edge TPU accelerator suffers from **three challenges:**

- It operates **significantly below** its peak throughput
- It operates **significantly below** its theoretical energy efficiency
- It **inefficiently** handles memory accesses

**Key Insight:** These shortcomings arise from **the monolithic design** of the Edge TPU accelerator

- The Edge TPU accelerator design does not account for **layer heterogeneity**

**Key Mechanism:** A new framework called **Mensa**

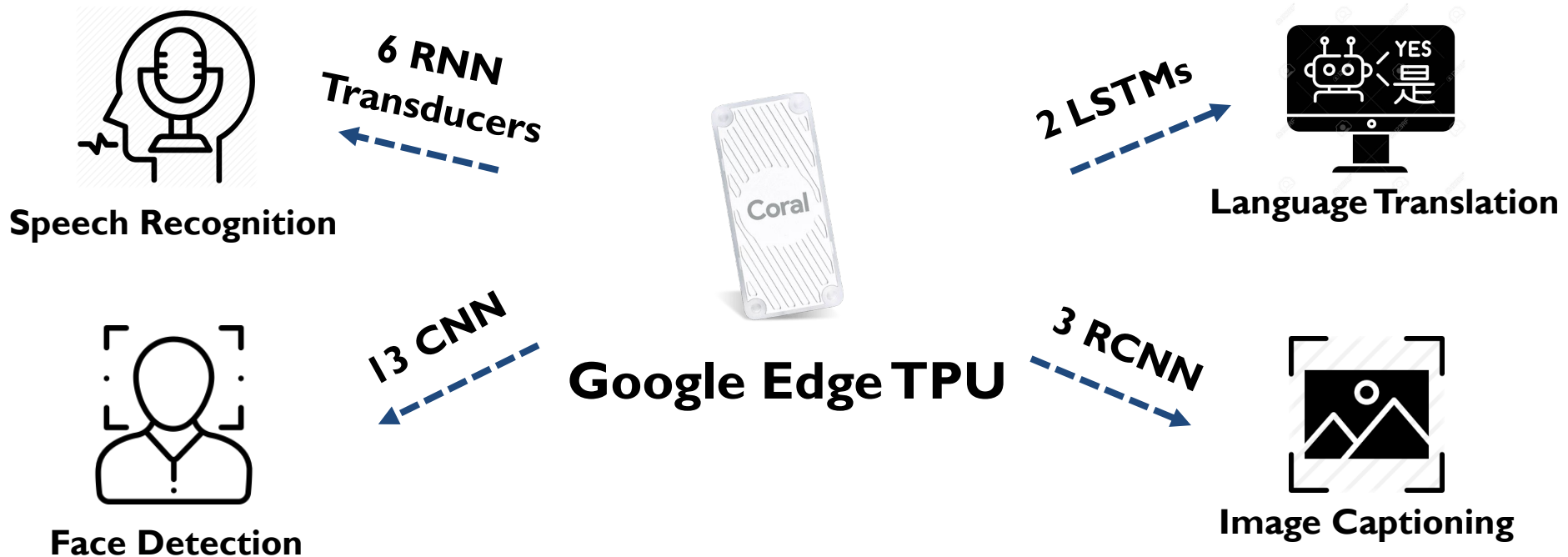
- Mensa consists of heterogeneous accelerators whose dataflow and hardware are specialized for specific families of layers

**Key Results:** We design a version of Mensa for Google edge ML models

- Mensa improves performance and energy by **3.0X** and **3.1X**
- Mensa reduces cost and improves area efficiency

# Google Edge Neural Network Models

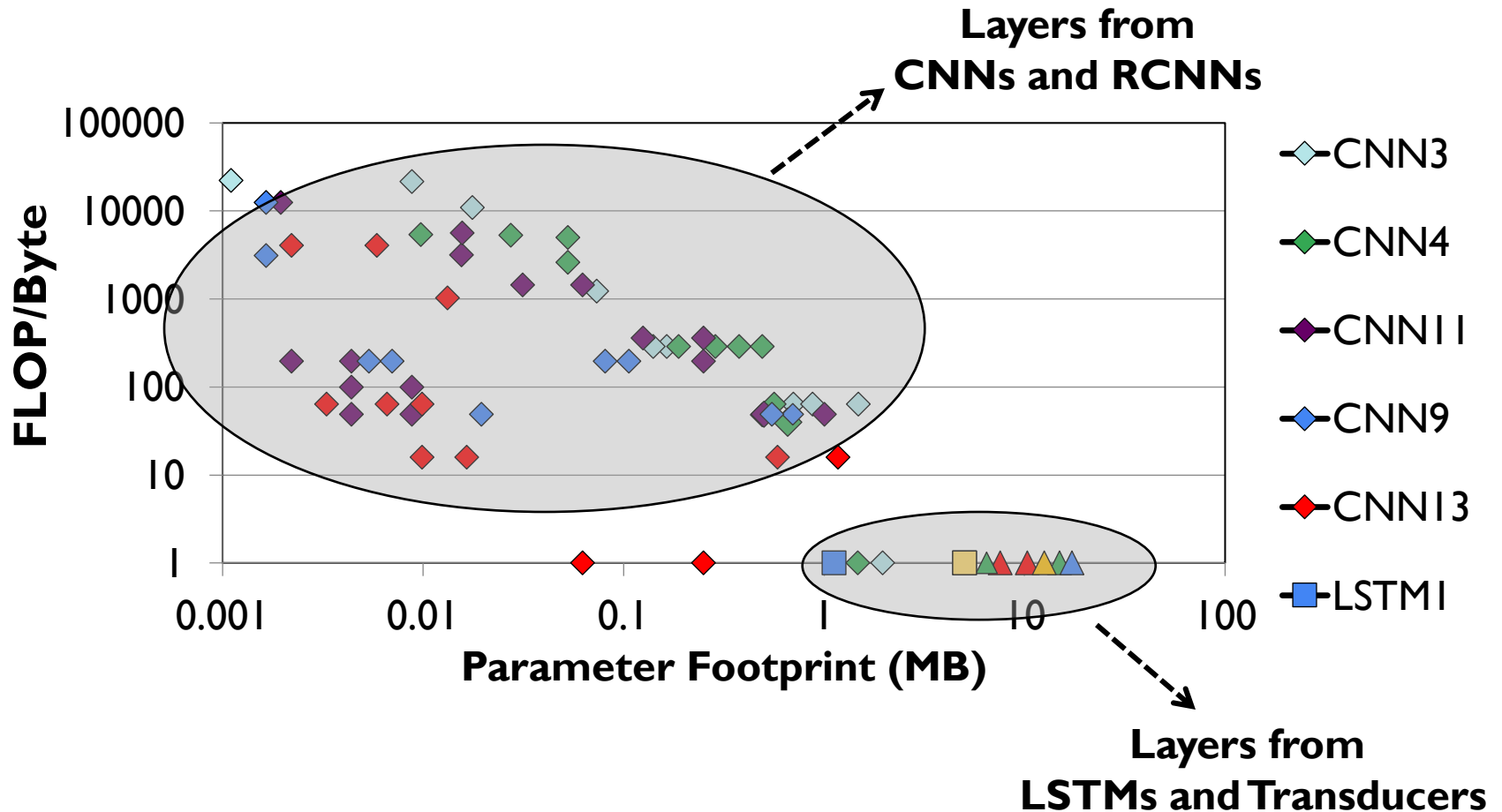
We analyze inference execution using 24 edge NN models





# Diversity Across the Models

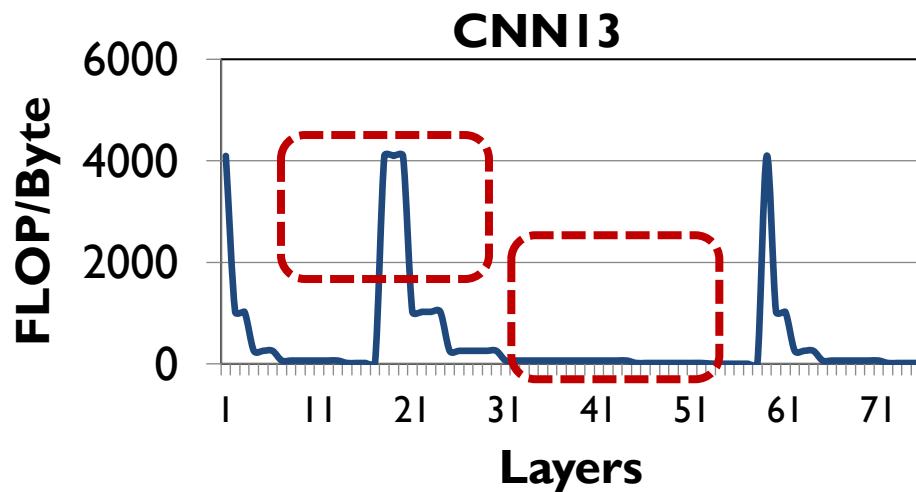
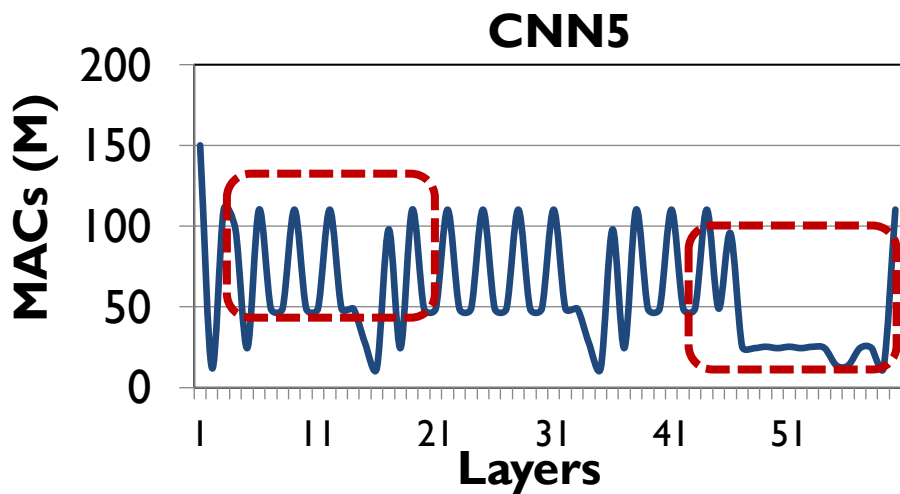
**Insight I:** there is **significant variation** in terms of layer characteristics **across the models**



# Diversity Within the Models

**Insight 2:** even **within** each model, layers exhibit **significant variation** in terms of layer characteristics

For example, our analysis of edge **CNN** models shows:



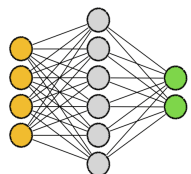
Variation in **MAC intensity**: up to **200x** across layers

Variation in **FLOP/Byte**: up to **244x** across layers

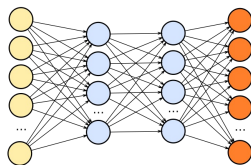
# Mensa High-Level Overview

## Edge TPU Accelerator

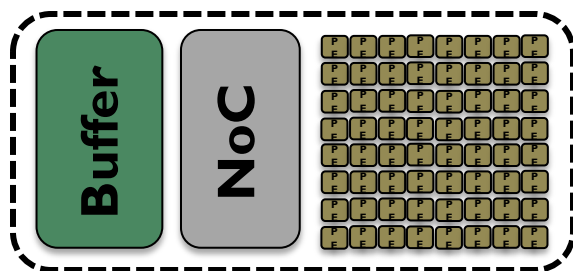
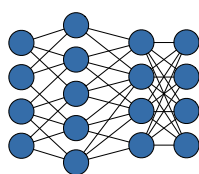
Model A



Model B



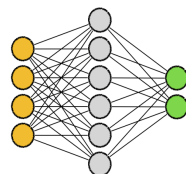
Model C



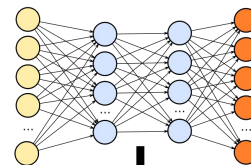
Monolithic Accelerator

## Mensa

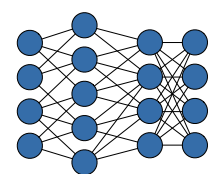
Model A



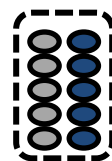
Model B



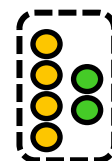
Model C



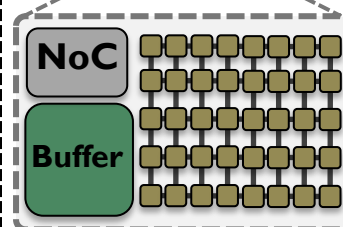
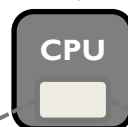
Family 1



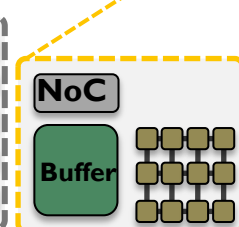
Family 2



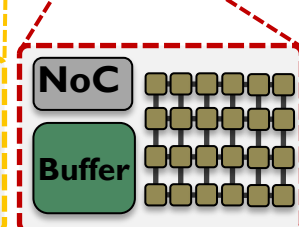
Family 3



Acc. 1



Acc. 2

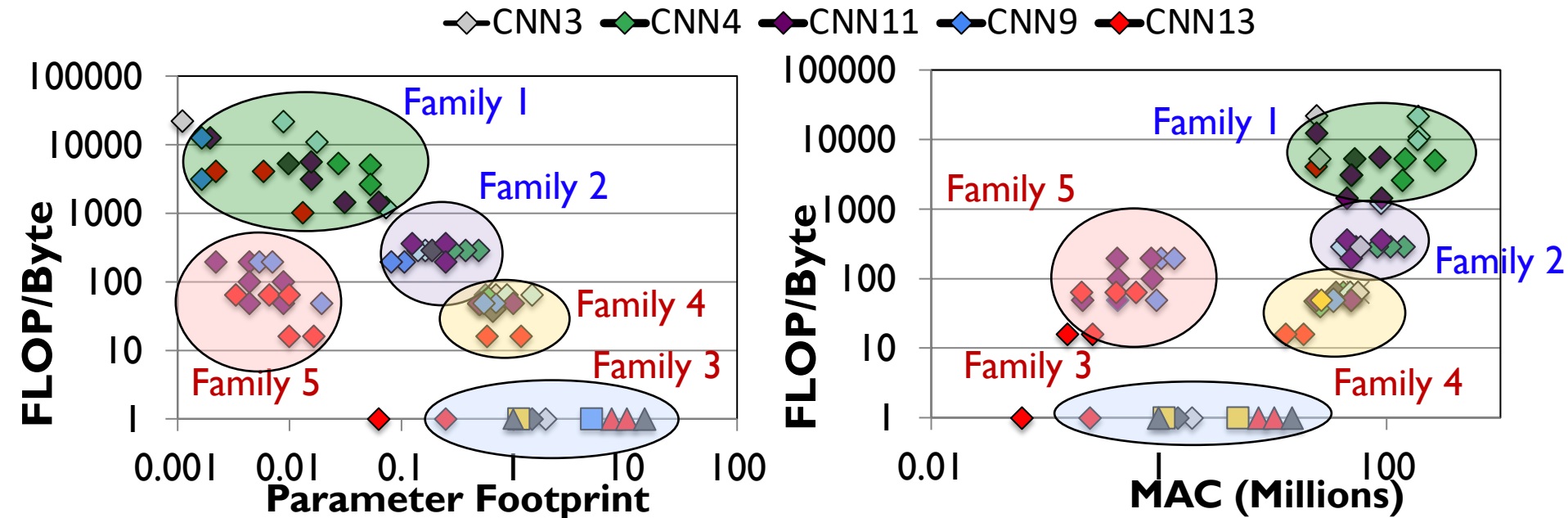


Acc. 3

Heterogeneous Accelerators

# Identifying Layer Families

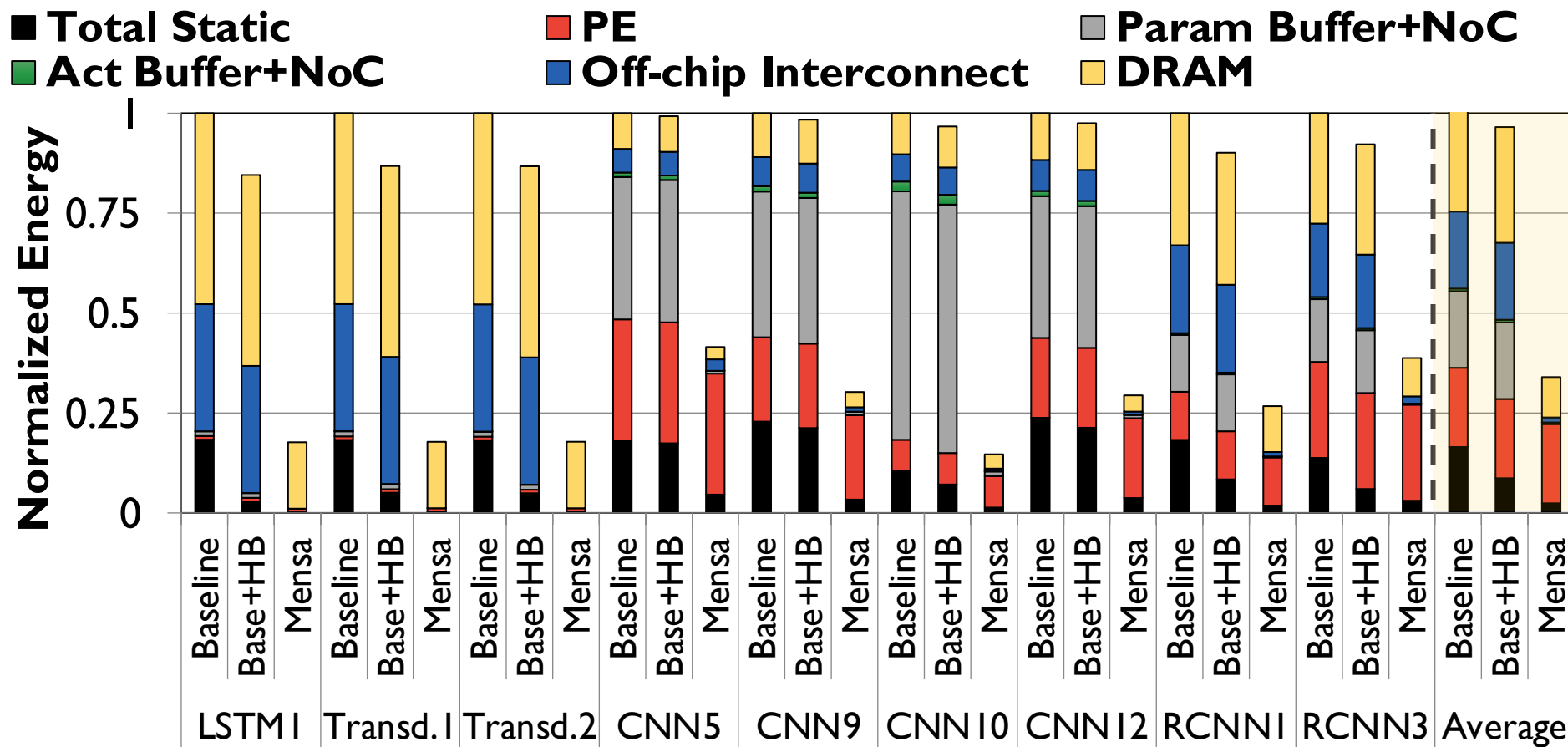
**Key observation: the majority of layers group into a small number of layer families**



**Families 1 & 2: low parameter footprint, high data reuse and **MAC** intensity**  
→ compute-centric layers

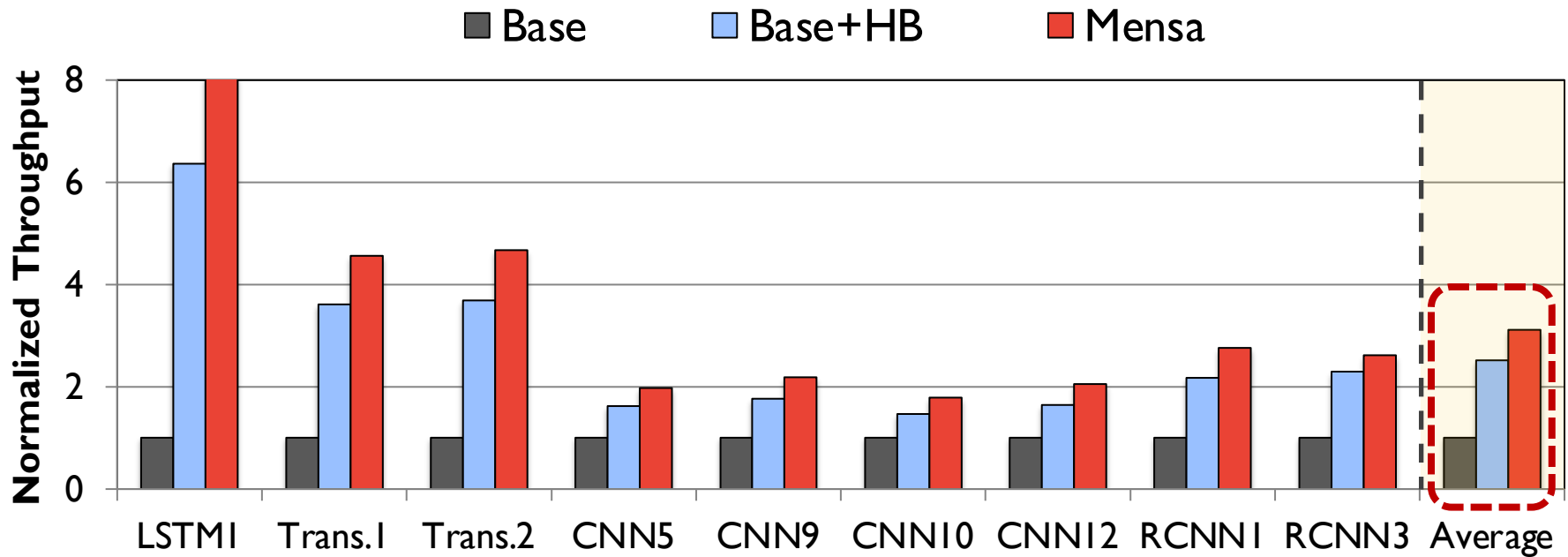
**Families 3, 4 & 5: high parameter footprint, low data reuse and **MAC** intensity**  
→ data-centric layers

# Mensa: Energy Reduction



**Mensa-G reduces energy consumption by 3.0X**  
compared to the baseline Edge TPU

# Mensa: Throughput Improvement



**Mensa-G improves inference throughput by 3.1X**  
compared to the baseline Edge TPU

# Mensa: Highly-Efficient ML Inference

---

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,  
**"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"**  
*Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Virtual, September 2021.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (14 minutes)]

## Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand<sup>†◇</sup>

Geraldo F. Oliveira<sup>\*</sup>

Saugata Ghose<sup>‡</sup>

Xiaoyu Ma<sup>§</sup>

Berkin Akin<sup>§</sup>

Eric Shiu<sup>§</sup>

Ravi Narayanaswami<sup>§</sup>

Onur Mutlu<sup>\*†</sup>

<sup>†</sup>*Carnegie Mellon Univ.*

<sup>◇</sup>*Stanford Univ.*

<sup>‡</sup>*Univ. of Illinois Urbana-Champaign*

<sup>§</sup>*Google*

<sup>\*</sup>*ETH Zürich*



# In-Storage Genomic Data Filtering [ASPLOS 2022]

---

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,  
**"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"**  
*Proceedings of the 27th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Virtual, February-March 2022.  
[[Lightning Talk Slides \(pptx\)](#)] ([pdf](#))  
[[Lightning Talk Video](#) (90 seconds)]

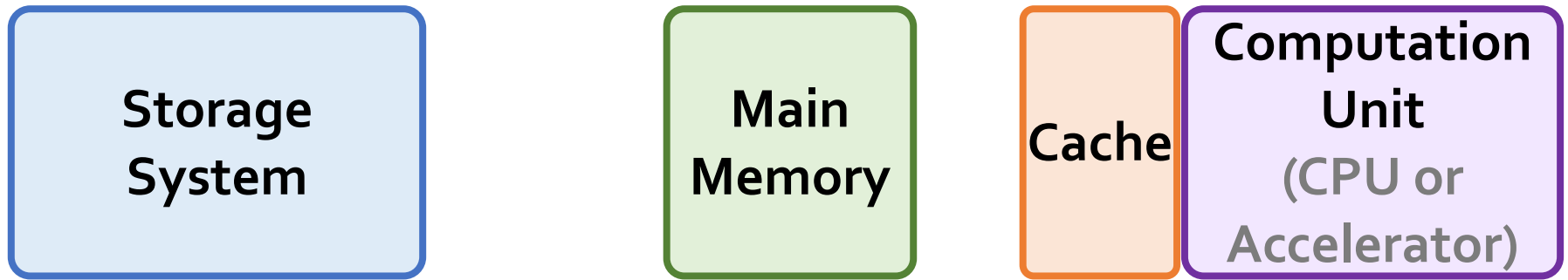
## GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi<sup>1</sup> Jisung Park<sup>1</sup> Harun Mustafa<sup>1</sup> Jeremie Kim<sup>1</sup> Ataberk Olgun<sup>1</sup>  
Arvid Gollwitzer<sup>1</sup> Damla Senol Cali<sup>2</sup> Can Firtina<sup>1</sup> Haiyu Mao<sup>1</sup> Nour Almadhoun Alserr<sup>1</sup>  
Rachata Ausavarungnirun<sup>3</sup> Nandita Vijaykumar<sup>4</sup> Mohammed Alser<sup>1</sup> Onur Mutlu<sup>1</sup>

<sup>1</sup>ETH Zürich <sup>2</sup>Bionano Genomics <sup>3</sup>KMUTNB <sup>4</sup>University of Toronto

# Genome Sequence Analysis

**Data Movement from Storage**

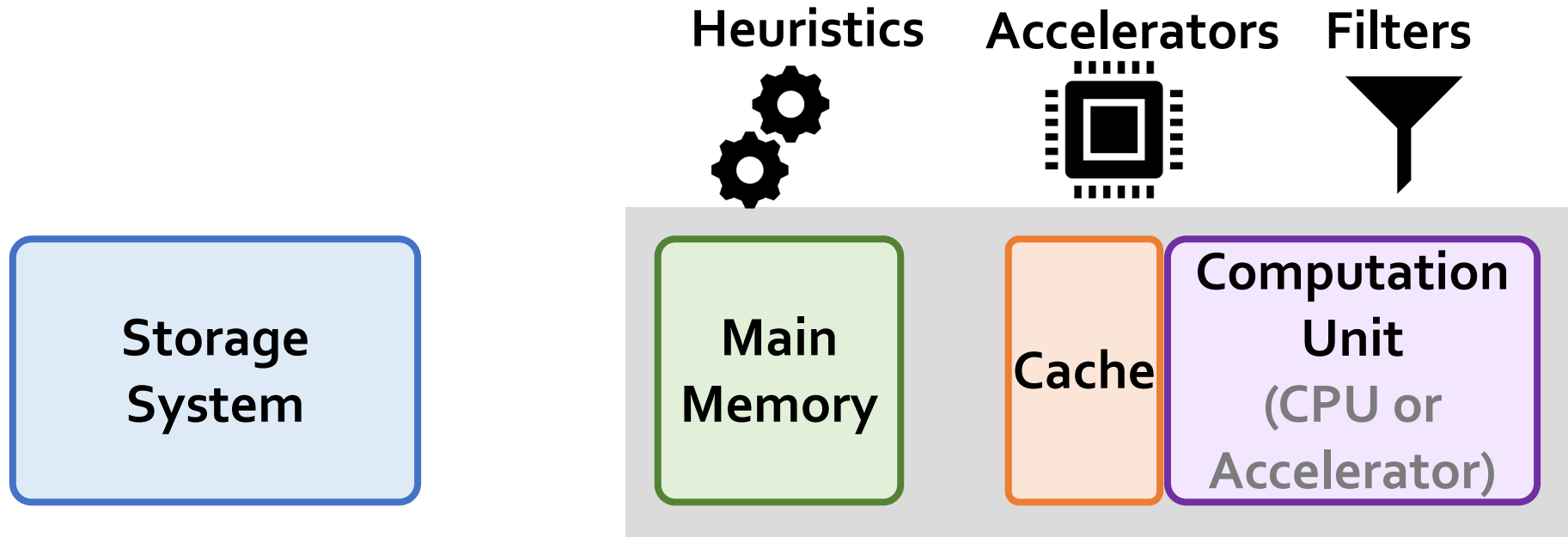


**Computation overhead**



**Data movement overhead**

# Compute-Centric Accelerators



Computation overhead

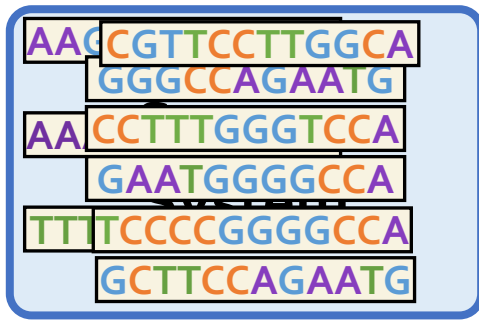


Data movement overhead

# Key Idea: In-Storage Filtering



*Filter reads that do **not** require alignment inside the storage system*



**Filtered Reads**

**Main  
Memory**

**Cache**

**Computation  
Unit  
(CPU or  
Accelerator)**

## **Exactly-matching** reads

Do not need expensive approximate string matching during alignment

## **Non-matching** reads

Do not have potential matching locations and can skip alignment

# GenStore



*Filter reads that do **not** require alignment  
inside the storage system*

GenStore-Enabled  
Storage  
System

Main  
Memory

Cache

Computation  
Unit  
(CPU or  
Accelerator)



Computation overhead



Data movement overhead

GenStore provides significant speedup (1.4x - 33.6x) and  
energy reduction (3.9x - 29.2x) at low cost

# In-Storage Genomic Data Filtering [ASPLOS 2022]

---

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,  
**"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"**  
*Proceedings of the 27th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Virtual, February-March 2022.  
[[Lightning Talk Slides \(pptx\)](#)] ([pdf](#))  
[[Lightning Talk Video](#) (90 seconds)]

## GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi<sup>1</sup> Jisung Park<sup>1</sup> Harun Mustafa<sup>1</sup> Jeremie Kim<sup>1</sup> Ataberk Olgun<sup>1</sup>  
Arvid Gollwitzer<sup>1</sup> Damla Senol Cali<sup>2</sup> Can Firtina<sup>1</sup> Haiyu Mao<sup>1</sup> Nour Almadhoun Alserr<sup>1</sup>  
Rachata Ausavarungnirun<sup>3</sup> Nandita Vijaykumar<sup>4</sup> Mohammed Alser<sup>1</sup> Onur Mutlu<sup>1</sup>

<sup>1</sup>ETH Zürich <sup>2</sup>Bionano Genomics <sup>3</sup>KMUTNB <sup>4</sup>University of Toronto

# DAMOV Analysis Methodology & Workloads

---

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

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

LOIS OROSA, ETH Zürich, Switzerland

SAUGATA GHOSE, University of Illinois at Urbana-Champaign, USA

NANDITA VIJAYKUMAR, University of Toronto, Canada

IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland

MOHAMMAD SADROSADATI, Institute for Research in Fundamental Sciences (IPM), Iran & ETH Zürich, Switzerland

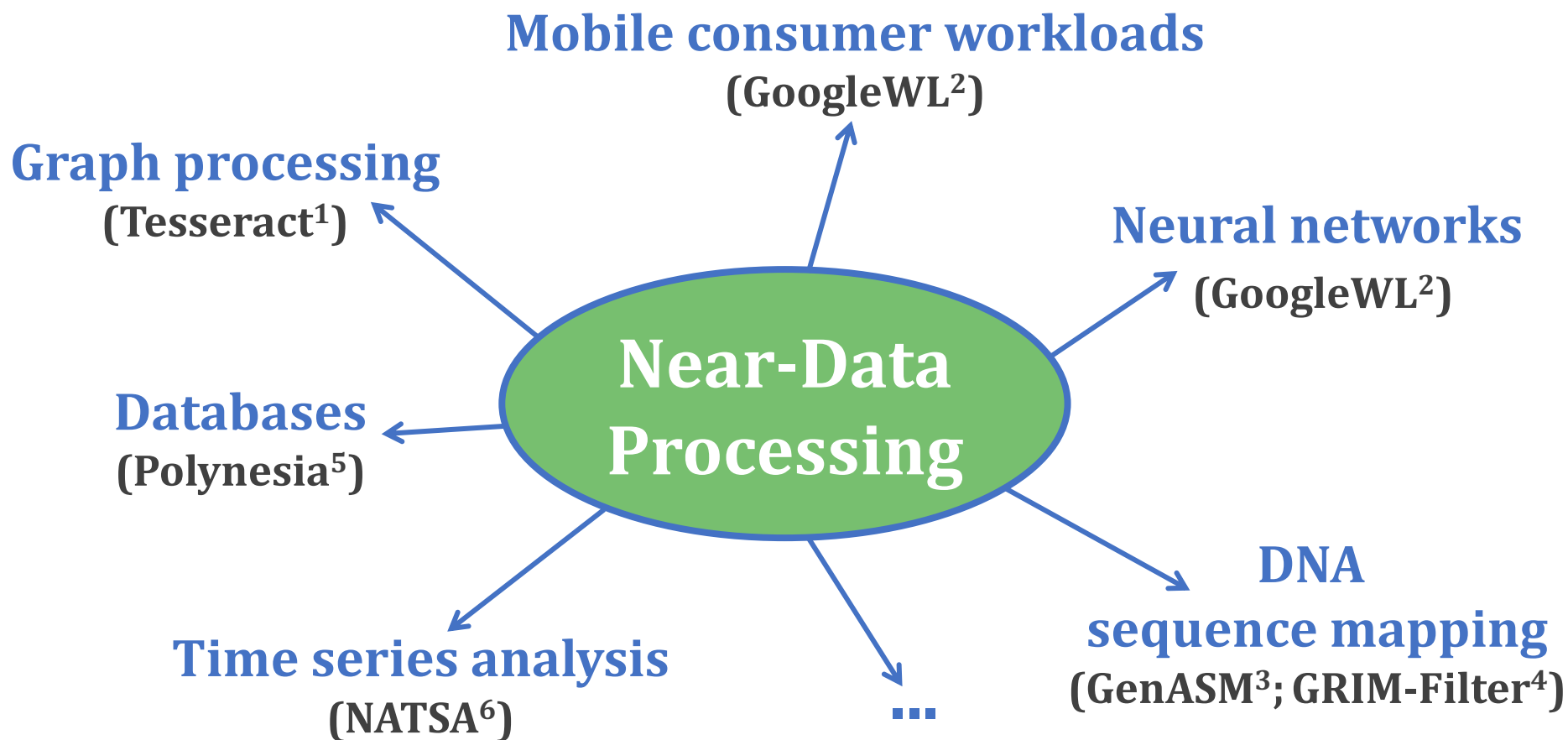
ONUR MUTLU, ETH Zürich, Switzerland

Data movement between the CPU and main memory is a first-order obstacle against improving performance, scalability, and energy efficiency in modern systems. Computer systems employ a range of techniques to reduce overheads tied to data movement, spanning from traditional mechanisms (e.g., deep multi-level cache hierarchies, aggressive hardware prefetchers) to emerging techniques such as Near-Data Processing (NDP), where some computation is moved close to memory. Prior NDP works investigate the root causes of data movement bottlenecks using different profiling methodologies and tools. However, there is still a lack of understanding about the key metrics that can identify different data movement bottlenecks and their relation to traditional and emerging data movement mitigation mechanisms. Our goal is to methodically identify potential sources of data movement over a broad set of applications and to comprehensively compare traditional compute-centric data movement mitigation techniques (e.g., caching and prefetching) to more memory-centric techniques (e.g., NDP), thereby developing a rigorous understanding of the best techniques to mitigate each source of data movement.

With this goal in mind, we perform the first large-scale characterization of a wide variety of applications, across a wide range of application domains, to identify fundamental program properties that lead to data movement to/from main memory. We develop the first systematic methodology to classify applications based on the sources contributing to data movement bottlenecks. From our large-scale characterization of 77K functions across 345 applications, we select 144 functions to form the first open-source benchmark suite (DAMOV) for main memory data movement studies. We select a diverse range of functions that (1) represent different types of data movement bottlenecks, and (2) come from a wide range of application domains. Using NDP as a case study, we identify new insights about the different data movement bottlenecks and use these insights to determine the most suitable data movement mitigation mechanism for a particular application. We open-source DAMOV and the complete source code for our new characterization methodology at <https://github.com/CMU-SAFARI/DAMOV>.



# When to Employ Near-Data Processing?



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

[2] Boroumand+, "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks," ASPLOS, 2018

[3] Cali+, "GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis," MICRO, 2020

[4] Kim+, "GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies," BMC Genomics, 2018

[5] Boroumand+, "Polynesia: Enabling Effective Hybrid Transactional/Analytical Databases with Specialized Hardware/Software Co-Design," arXiv:2103.00798 [cs.AR], 2021

[6] Fernandez+, "NATSA: A Near-Data Processing Accelerator for Time Series Analysis," ICCD, 2020

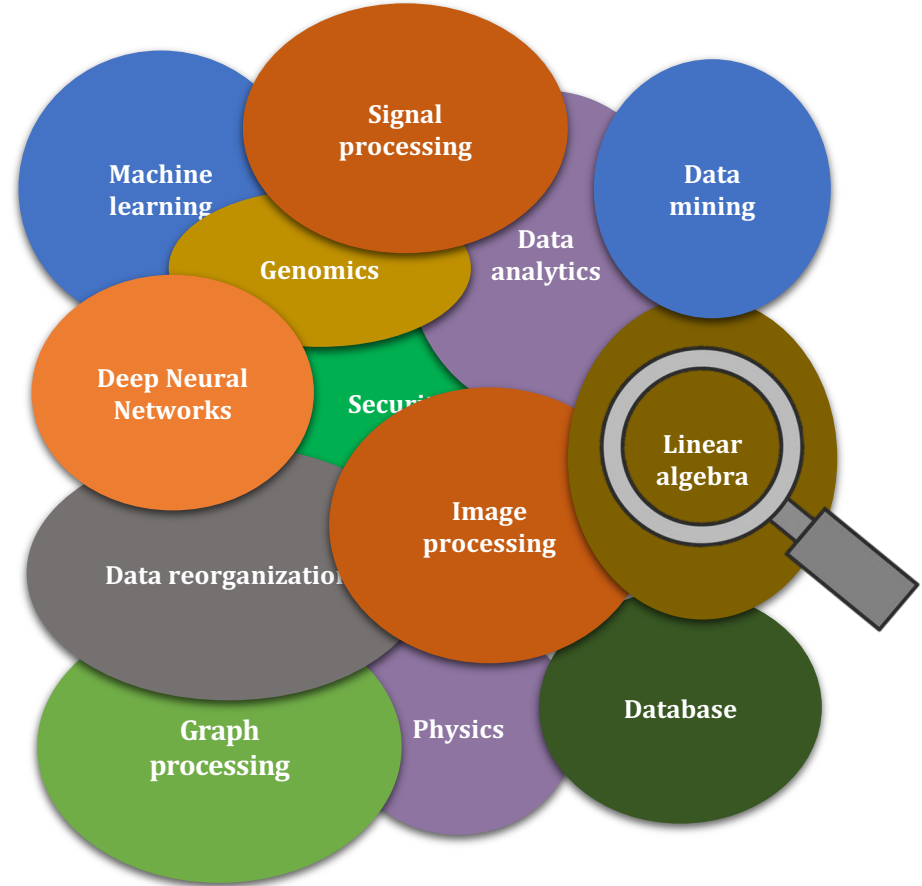
# Step 1: Application Profiling

- We analyze 345 applications from distinct domains:

- Graph Processing
- Deep Neural Networks
- Physics
- High-Performance Computing
- Genomics
- Machine Learning
- Databases
- Data Reorganization
- Image Processing
- Map-Reduce
- Benchmarking
- Linear Algebra

...

**SAFARI**



# Step 3: Memory Bottleneck Analysis

**Six classes of  
data movement bottlenecks:**

each class  $\leftrightarrow$  data movement  
mitigation mechanism

## Memory Bottleneck Class

1a: *DRAM  
Bandwidth*

1b: *DRAM Latency*

1c: *L1/L2  
Cache Capacity*

2a: *L3 Cache  
Contention*

2b: *L1 Cache  
Capacity*

2c: *Compute-Bound*

# DAMOV is Open Source

- We open-source our **benchmark suite** and our **toolchain**

CMU-SAFARI / DAMOV

<> Code Issues Pull requests Actions Projects Security Insights Settings

main 1 branch 0 tags

Go to file

Add file

Code

About



DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing. Described by Oliveira et al. (preliminary version at <https://arxiv.org/pdf/2105.03725.pdf>)

Readme

Releases

No releases published  
[Create a new release](#)

Packages

No packages published  
[Publish your first package](#)

Languages



omutlu Update README.md

ce1b4ea 17 days ago 5 commits

simulator

Cleaning

19 days ago

README.md

Update README.md

17 days ago

get\_workloads.sh

DAMOV -- first commit

19 days ago

README.md

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

DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing.

The DAMOV benchmark suite is the first open-source benchmark suite for main memory data movement-related studies, based on our systematic characterization methodology. This suite consists of 144 functions representing different sources of data movement bottlenecks and can be used as a baseline benchmark set for future data-movement mitigation research. The applications in the DAMOV benchmark suite belong to popular benchmark suites, including [BWA](#), [Chai](#), [Darknet](#), [GASE](#), [Hardware Effects](#), [Hashjoin](#), [HPCC](#), [HPCG](#), [Ligra](#), [PARSEC](#), [Parboil](#), [PolyBench](#), [Phoenix](#), [Rodinia](#), [SPLASH-2](#), [STREAM](#).

DAMOV-SIM

DAMOV  
Benchmarks

SAFARI

# DAMOV is Open Source

- We open-source our [benchmark suite](#) and our [toolchain](#)

CMU-SAFARI / DAMOV

<> Code Issues Pull requests Actions Projects Security Insights Settings

main 1 branch 0 tags

Go to file

Add file

Code

About

DAMOV is a benchmark suite and a

Get DAMOV at:

<https://github.com/CMU-SAFARI/DAMOV>

README.md

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

DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing.

The DAMOV benchmark suite is the first open-source benchmark suite for main memory data movement-related studies, based on our systematic characterization methodology. This suite consists of 144 functions representing different sources of data movement bottlenecks and can be used as a baseline benchmark set for future data-movement mitigation research. The applications in the DAMOV benchmark suite belong to popular benchmark suites, including [BWA](#), [Chai](#), [Darknet](#), [GASE](#), [Hardware Effects](#), [Hashjoin](#), [HPCC](#), [HPCG](#), [Ligra](#), [PARSEC](#), [Parboil](#), [PolyBench](#), [Phoenix](#), [Rodinia](#), [SPLASH-2](#), [STREAM](#).

Readme

### Releases

No releases published  
[Create a new release](#)

### Packages

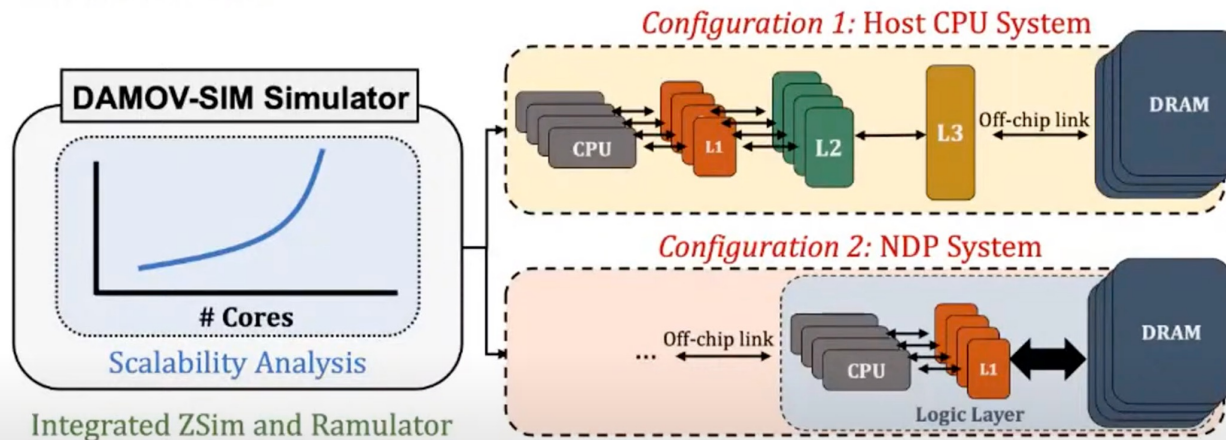
No packages published  
[Publish your first package](#)

### Languages

# More on DAMOV Analysis Methodology & Workloads

## Step 3: Memory Bottleneck Classification (2/)

- **Goal:** identify the specific sources of data movement bottlenecks



- **Scalability Analysis:**
  - 1, 4, 16, 64, and 256 out-of-order/in-order host and NDP CPU cores
  - 3D-stacked memory as main memory

SAFARI DAMOV-SIM: <https://github.com/CMU-SAFARI/DAMOV> 30

SAFARI Live Seminar: DAMOV: A New Methodology & Benchmark Suite for Data Movement Bottlenecks

352 views • Streamed live on Jul 22, 2021

18 0 SHARE SAVE ...



Onur Mutlu Lectures  
17.7K subscribers

ANALYTICS

EDIT VIDEO

[https://www.youtube.com/watch?v=GWideVyo0nM&list=PL5Q2soXY2Zi\\_tOTAYm--dYByNPL7JhwR9&index=3](https://www.youtube.com/watch?v=GWideVyo0nM&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9&index=3)

# More on DAMOV Methods & Benchmarks

---

- Geraldo F. Oliveira, Juan Gomez-Luna, Lois Orosa, Saugata Ghose, Nandita Vijaykumar, Ivan fernandez, Mohammad Sadrosadati, and Onur Mutlu,  
**"DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks"**  
**IEEE Access**, 8 September 2021.  
*Preprint in arXiv*, 8 May 2021.  
[[arXiv preprint](#)]  
[[IEEE Access version](#)]  
[[DAMOV Suite and Simulator Source Code](#)]  
[[SAFARI Live Seminar Video](#) (2 hrs 40 mins)]  
[[Short Talk Video](#) (21 minutes)]

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

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

LOIS OROSA, ETH Zürich, Switzerland

SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA

NANDITA VIJAYKUMAR, University of Toronto, Canada

IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland

MOHAMMAD SADROSADATI, ETH Zürich, Switzerland

ONUR MUTLU, ETH Zürich, Switzerland



# Ramulator 2.0 for PIM Systems

---

- Haocong Luo, Yahya Can Tugrul, F. Nisa Bostanci, Ataberk Olgun, A. Giray Yaglikci, and Onur Mutlu,  
**"Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator"**  
*Preprint on **arxiv**, August 2023.*  
[[arXiv version](#)]  
[[Ramulator 2.0 Source Code](#)]

## Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator

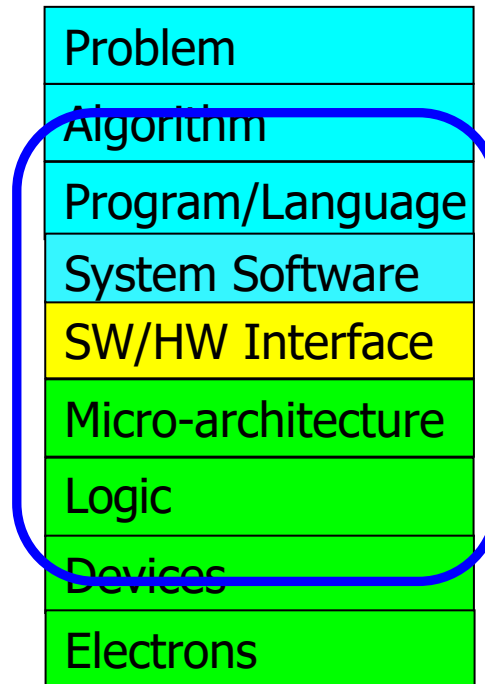
Haocong Luo, Yahya Can Tuğrul, F. Nisa Bostancı, Ataberk Olgun, A. Giray Yağlıkçı, and Onur Mutlu

<https://arxiv.org/pdf/2308.11030.pdf>

# We Need to Revisit the Entire Stack

---

- With a **memory-centric mindset**



**We can get there step by step**

# PIM Review and Open Problems

---

## A Modern Primer on Processing in Memory

Onur Mutlu<sup>a,b</sup>, Saugata Ghose<sup>b,c</sup>, Juan Gómez-Luna<sup>a</sup>, Rachata Ausavarungnirun<sup>d</sup>

*SAFARI Research Group*

<sup>a</sup>*ETH Zürich*

<sup>b</sup>*Carnegie Mellon University*

<sup>c</sup>*University of Illinois at Urbana-Champaign*

<sup>d</sup>*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.*

# PIM Review and Open Problems (II)

---

## A Workload and Programming Ease Driven Perspective of Processing-in-Memory

Saugata Ghose<sup>†</sup>   Amirali Boroumand<sup>†</sup>   Jeremie S. Kim<sup>†§</sup>   Juan Gómez-Luna<sup>§</sup>   Onur Mutlu<sup>§†</sup>

<sup>†</sup>*Carnegie Mellon University*

<sup>§</sup>*ETH Zürich*

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, to appear in November 2019.*

[Preliminary arXiv version]

# Processing in Memory: Adoption Challenges

1. Processing **using** Memory
2. Processing **near** Memory

## How to Enable Adoption of Processing in Memory

# Potential Barriers to Adoption of PIM

---

1. **Applications & software** for PIM
2. Ease of **programming** (interfaces and compiler/HW support)
3. **System** and **security** support: coherence, synchronization, virtual memory, isolation, communication interfaces, ...
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**

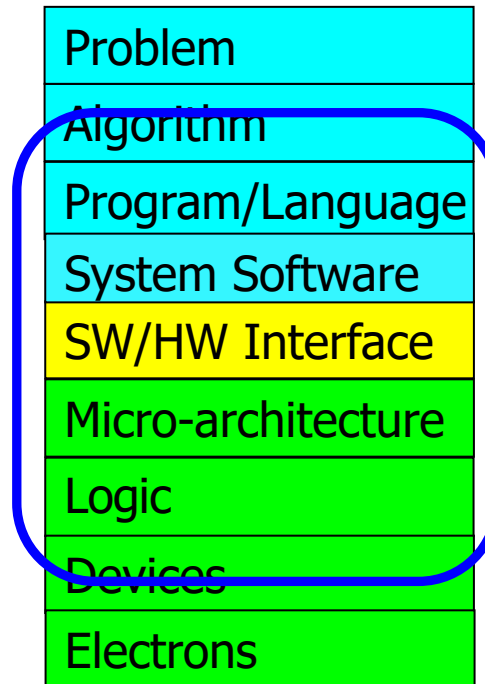
---



# We Need to Revisit the Entire Stack

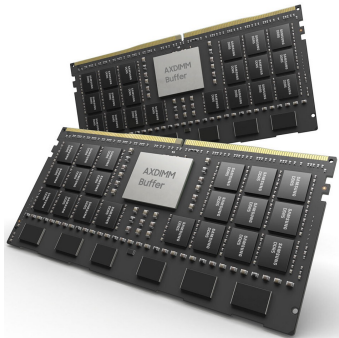
---

- With a **memory-centric mindset**

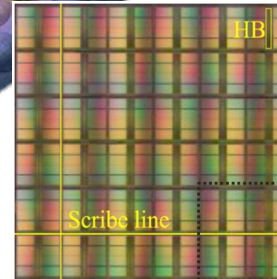
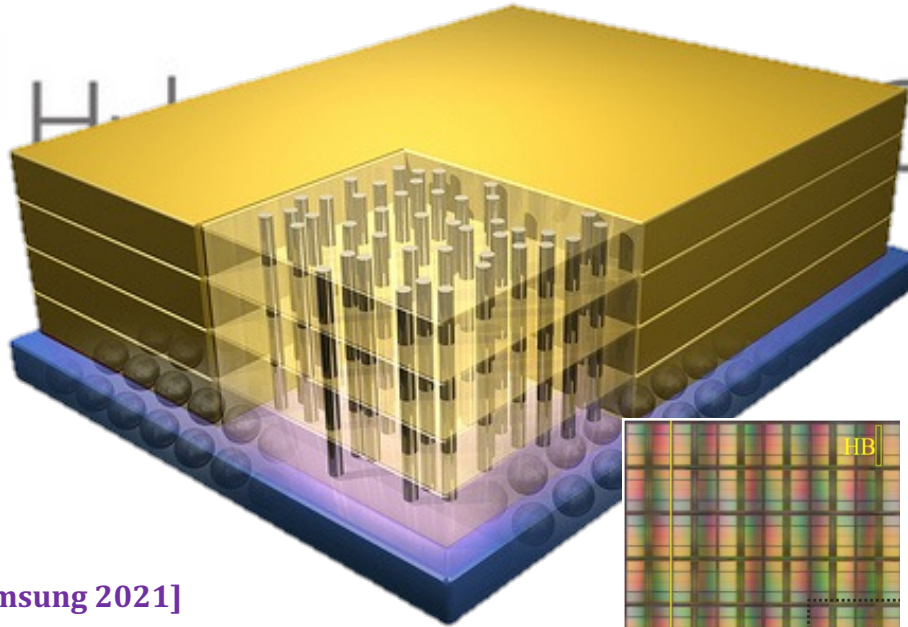


**We can get there step by step**

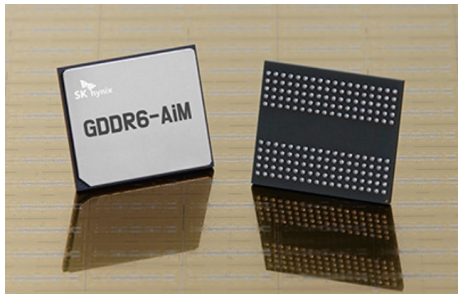
# Processing-in-Memory Landscape Today



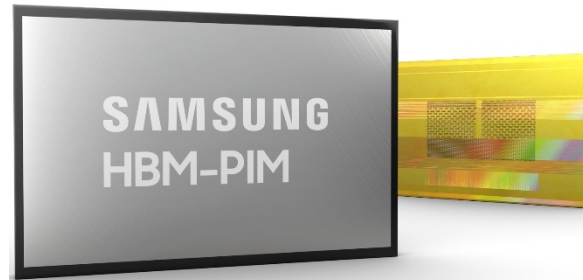
[Samsung 2021]



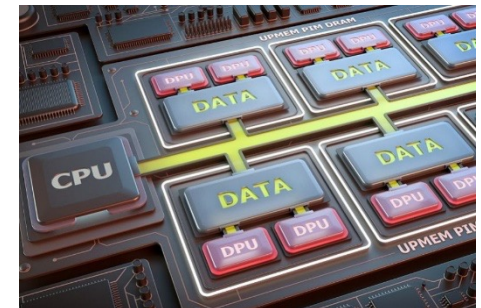
[Alibaba 2022]



[SK Hynix 2022]



[Samsung 2021]



[UPMEM 2019]

# Adoption: How to Keep It Simple?

---

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoun 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<sup>†</sup>   Kiyoun Choi

junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr

Seoul National University

<sup>†</sup>Carnegie Mellon University

# Adoption: How to Ease Programmability? (I)

---

- Geraldo F. Oliveira, Alain Kohli, David Novo, Juan Gómez-Luna, Onur Mutlu,  
**“DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures,”**  
in *PACT SRC Student Competition*, Vienna, Austria, October 2023.

## **DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures**

Geraldo F. Oliveira\*

Alain Kohli\*

David Novo‡

Juan Gómez-Luna\*

Onur Mutlu\*

\**ETH Zürich*

‡*LIRMM, Univ. Montpellier, CNRS*

# Adoption: How to Ease Programmability? (II)

---

- Jinfan Chen, Juan Gómez-Luna, Izzat El Hajj, YuXin Guo, and Onur Mutlu,  
**"SimplePIM: A Software Framework for Productive and Efficient Processing in Memory"**  
*Proceedings of the 32nd International Conference on Parallel Architectures and Compilation Techniques (PACT), Vienna, Austria, October 2023.*

## SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen<sup>1</sup>   Juan Gómez-Luna<sup>1</sup>   Izzat El Hajj<sup>2</sup>   Yuxin Guo<sup>1</sup>   Onur Mutlu<sup>1</sup>  
<sup>1</sup>ETH Zürich   <sup>2</sup>American University of Beirut

# Adoption: How to Ease Programmability? (III)

---

- Geraldo F. Oliveira, Juan Gomez-Luna, Lois Orosa, Saugata Ghose, Nandita Vijaykumar, Ivan fernandez, Mohammad Sadrosadati, and Onur Mutlu,  
**"DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks"**  
**IEEE Access**, 8 September 2021.  
*Preprint in arXiv*, 8 May 2021.  
[[arXiv preprint](#)]  
[[IEEE Access version](#)]  
[[DAMOV Suite and Simulator Source Code](#)]  
[[SAFARI Live Seminar Video](#) (2 hrs 40 mins)]  
[[Short Talk Video](#) (21 minutes)]

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

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

LOIS OROSA, ETH Zürich, Switzerland

SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA

NANDITA VIJAYKUMAR, University of Toronto, Canada

IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland

MOHAMMAD SADROSADATI, ETH Zürich, Switzerland

ONUR MUTLU, ETH Zürich, Switzerland

# Adoption: How to Maintain Coherence? (I)

---

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,  
"**LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory**"  
*IEEE Computer Architecture Letters* (**CAL**), June 2016.

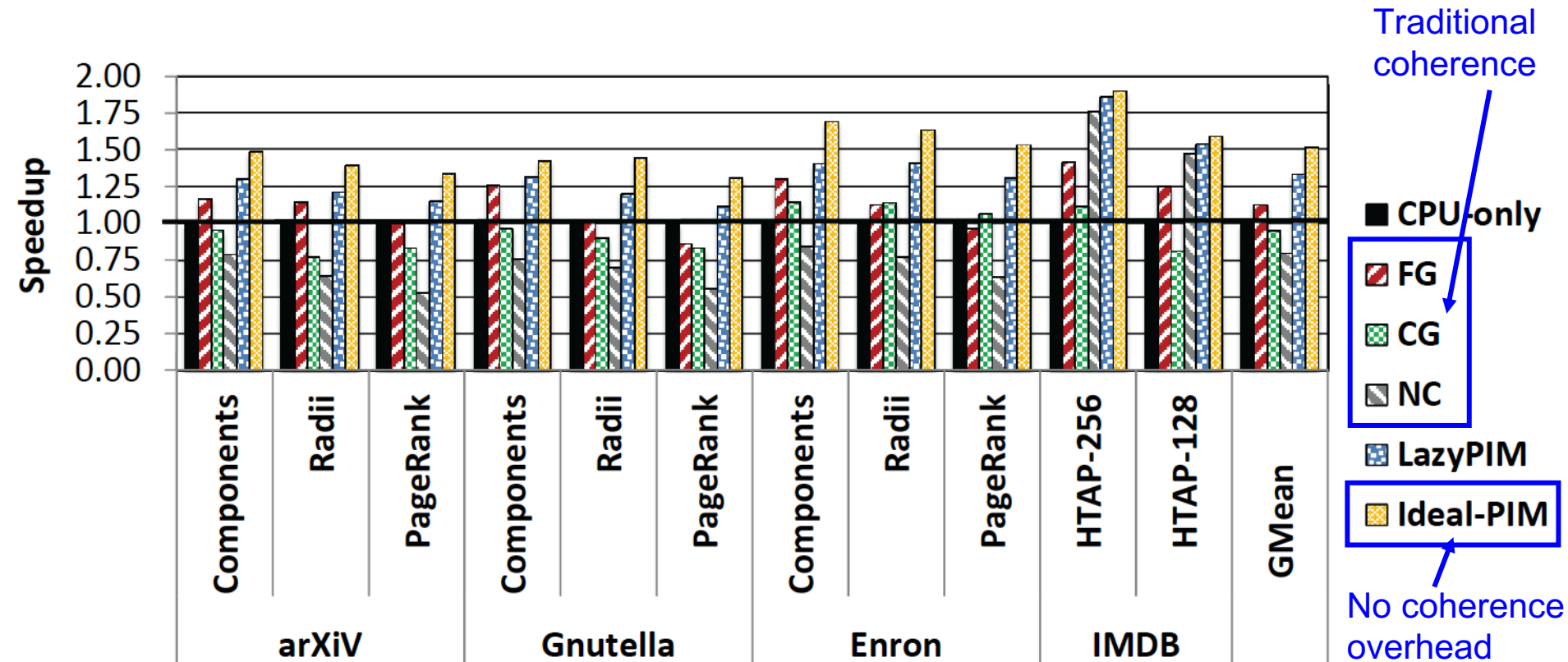
## LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory

Amirali Boroumand<sup>†</sup>, Saugata Ghose<sup>†</sup>, Minesh Patel<sup>†</sup>, Hasan Hassan<sup>†§</sup>, Brandon Lucia<sup>†</sup>,  
Kevin Hsieh<sup>†</sup>, Krishna T. Malladi<sup>\*</sup>, Hongzhong Zheng<sup>\*</sup>, and Onur Mutlu<sup>††</sup>

<sup>†</sup>Carnegie Mellon University   <sup>\*</sup>Samsung Semiconductor, Inc.   <sup>§</sup>TOBB ETÜ   <sup>‡</sup>ETH Zürich



# Challenge: Coherence for Hybrid CPU-PIM Apps



# Adoption: How to Maintain Coherence? (II)

---

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,  
**"CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators"**

*Proceedings of the 46th International Symposium on Computer Architecture (ISCA), Phoenix, AZ, USA, June 2019.*

## CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators

Amirali Boroumand<sup>†</sup>

Saugata Ghose<sup>†</sup>

Minesh Patel<sup>★</sup>

Hasan Hassan<sup>★</sup>

Brandon Lucia<sup>†</sup>

Rachata Ausavarungnirun<sup>†‡</sup>

Kevin Hsieh<sup>†</sup>

Nastaran Hajinazar<sup>◇†</sup>

Krishna T. Malladi<sup>§</sup>

Hongzhong Zheng<sup>§</sup>

Onur Mutlu<sup>★†</sup>

<sup>†</sup>Carnegie Mellon University

<sup>★</sup>ETH Zürich

<sup>‡</sup>KMUTNB

<sup>◇</sup>Simon Fraser University

<sup>§</sup>Samsung Semiconductor, Inc.

# Adoption: How to Support Synchronization?

---

- Christina Giannoula, Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas, Ivan Fernandez, Juan Gómez-Luna, Lois Orosa, Nectarios Koziris, Georgios Goumas, Onur Mutlu, **"SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures"**  
*Proceedings of the 27th International Symposium on High-Performance Computer Architecture (HPCA)*, Virtual, February-March 2021.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Short Talk Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (21 minutes)]  
[[Short Talk Video](#) (7 minutes)]

## **SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures**

Christina Giannoula<sup>†‡</sup> Nandita Vijaykumar<sup>\*‡</sup> Nikela Papadopoulou<sup>†</sup> Vasileios Karakostas<sup>†</sup> Ivan Fernandez<sup>§‡</sup>  
Juan Gómez-Luna<sup>‡</sup> Lois Orosa<sup>‡</sup> Nectarios Koziris<sup>†</sup> Georgios Goumas<sup>†</sup> Onur Mutlu<sup>‡</sup>  
<sup>†</sup>*National Technical University of Athens*    <sup>‡</sup>*ETH Zürich*    <sup>\*</sup>*University of Toronto*    <sup>§</sup>*University of Malaga*

# Adoption: How to Support Virtual Memory?

---

- Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,  
["Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"](#)  
*Proceedings of the 34th IEEE International Conference on Computer Design (ICCD)*, Phoenix, AZ, USA, October 2016.

## Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh<sup>†</sup>   Samira Khan<sup>‡</sup>   Nandita Vijaykumar<sup>†</sup>  
Kevin K. Chang<sup>†</sup>   Amirali Boroumand<sup>†</sup>   Saugata Ghose<sup>†</sup>   Onur Mutlu<sup>§†</sup>  
<sup>†</sup>*Carnegie Mellon University*   <sup>‡</sup>*University of Virginia*   <sup>§</sup>*ETH Zürich*

# Adoption: Evaluation Infrastructures

---

- Haocong Luo, Yahya Can Tugrul, F. Nisa Bostanci, Ataberk Olgun, A. Giray Yaglikci, and Onur Mutlu,  
**"Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator"**  
*Preprint on **arxiv**, August 2023.*  
[[arXiv version](#)]  
[[Ramulator 2.0 Source Code](#)]

## Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator

Haocong Luo, Yahya Can Tuğrul, F. Nisa Bostancı, Ataberk Olgun, A. Giray Yağlıkçı, and Onur Mutlu

<https://arxiv.org/pdf/2308.11030.pdf>

# Methodologies, Workloads, and Tools for Processing-in-Memory: Enabling the Adoption of Data-Centric Architectures

**Geraldo F. Oliveira and Onur Mutlu**

[geraldofojunior@gmail.com](mailto:geraldofojunior@gmail.com)

<https://geraldofojunior.github.io/>

**SAFARI**

**ETH** zürich

# Processing-in-Memory: Challenges

**To fully support PIM systems, we need to develop:**

- 1 **Workload characterization methodologies and benchmark suites targeting PIM architectures**
- 2 **Frameworks that can facilitate the implementation of complex operations and algorithms using PIM primitives**
- 3 **Compiler support and compiler optimizations targeting PIM architectures**
- 4 **Operating system support for PIM-aware virtual memory, memory management, data allocation and mapping**
- 5 **End-to-End System-on-Chip Design Beyond DRAM**

**The lack of tools and system support for PIM architectures limit the adoption of PIM systems**



# An Example: SimplePIM Framework

---

- Jinfan Chen, Juan Gómez-Luna, Izzat El Hajj, YuXin Guo, and Onur Mutlu,  
**"SimplePIM: A Software Framework for Productive and Efficient Processing in Memory"**  
*Proceedings of the 32nd International Conference on Parallel Architectures and Compilation Techniques (PACT), Vienna, Austria, October 2023.*

## SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen<sup>1</sup>   Juan Gómez-Luna<sup>1</sup>   Izzat El Hajj<sup>2</sup>   Yuxin Guo<sup>1</sup>   Onur Mutlu<sup>1</sup>  
<sup>1</sup>ETH Zürich   <sup>2</sup>American University of Beirut

# Programming a PIM System

---

- PIM programming is challenging

- Manage data movement between host DRAM and PIM DRAM
  - Parallel, serial, broadcast, and gather/scatter transfers
- Manage data movement between PIM DRAM bank and scratchpad
  - 8-byte aligned and maximum of 2,048 bytes
- Multithreaded programming model
- Inter-thread synchronization
  - Barriers, handshakes, mutexes, and semaphores

## Our Goal

Design a **high-level programming framework** that abstracts these hardware-specific complexities and provides **a clean yet powerful interface** for ease of use and **high program performance**

# The SimplePIM Programming Framework

---

- SimplePIM provides standard abstractions to build and deploy applications on PIM systems
  - Management interface
    - Metadata for PIM-resident arrays
  - Communication interface
    - Abstractions for host-PIM and PIM-PIM communication
  - Processing interface
    - Iterators (`map`, `reduce`, `zip`) to implement workloads

# Productivity Improvement (I)

- Example: Hand-optimized histogram with UPMEM SDK

```
... // Initialize global variables and functions for histogram
int main_kernel() {
    if (tasklet_id == 0)
        mem_reset(); // Reset the heap
    ... // Initialize variables and the histogram
    T *input_buff_A = (T*)mem_alloc(2048); // Allocate buffer in scratchpad memory

    for (unsigned int byte_index = base_tasklet; byte_index < input_size; byte_index += stride) {
        // Boundary checking
        uint32_t l_size_bytes = (byte_index + 2048 >= input_size) ? (input_size - byte_index) : 2048;
        // Load scratchpad with a DRAM block
        mram_read((const __mram_ptr void*)(mram_base_addr_A + byte_index), input_buff_A, l_size_bytes);
        // Histogram calculation
        histogram(hist, bins, input_buff_A, l_size_bytes/sizeof(uint32_t));
    }
    ...
    barrier_wait(&my_barrier); // Barrier to synchronize PIM threads
    ... // Merging histograms from different tasklets into one histo_dpu
    // Write result from scratchpad to DRAM
    if (tasklet_id == 0)
        if (bins * sizeof(uint32_t) <= 2048)
            mram_write(histo_dpu, (__mram_ptr void*)mram_base_addr_histo, bins * sizeof(uint32_t));
        else
            for (unsigned int offset = 0; offset < ((bins * sizeof(uint32_t)) >> 11); offset++) {
                mram_write(histo_dpu + (offset << 9), (__mram_ptr void*)(mram_base_addr_histo +
                    (offset << 11)), 2048);
            }
    return 0;
}
```

# Productivity Improvement (II)

- Example: SimplePIM histogram

```
// Programmer-defined functions in the file "histo_filepath"
void init_func (uint32_t size, void* ptr) {
    char* casted_value_ptr = (char*) ptr;
    for (int i = 0; i < size; i++)
        casted_value_ptr[i] = 0;
}

void acc_func (void* dest, void* src) {
    *(uint32_t*)dest += *(uint32_t*)src;
}

void map_to_val_func (void* input, void* output, uint32_t* key) {
    uint32_t d = *(uint32_t*)input;
    *(uint32_t*)output = 1;
    *key = d * bins >> 12;
}

// Host side handle creation and iterator call
handle_t* handle = simple_pim_create_handle("histo_filepath", REDUCE, NULL, 0);

// Transfer (scatter) data to PIM, register as "t1"
simple_pim_array_scatter("t1", src, bins, sizeof(T), management);

// Run histogram on "t1" and produce "t2"
simple_pim_array_red("t1", "t2", sizeof(T), bins, handle, management);
```

# Productivity Improvement (III)

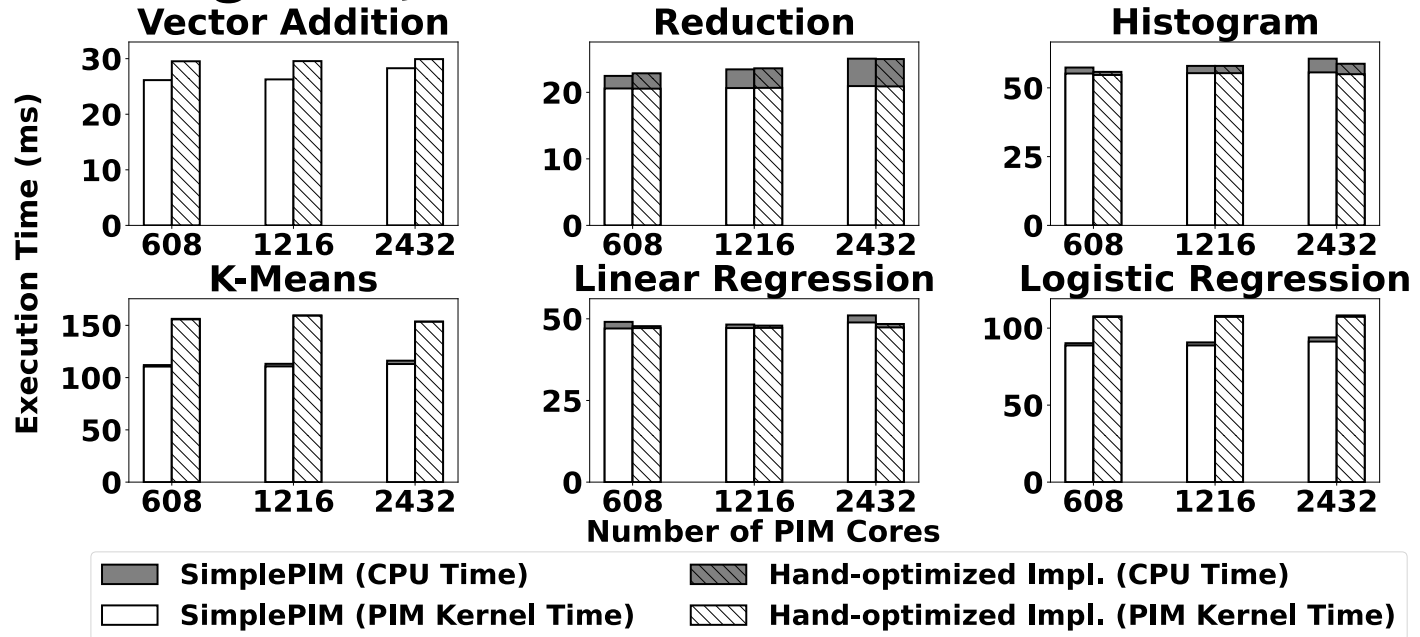
- Lines of code (LoC) reduction

|                     | SimplePIM | Hand-optimized | LoC Reduction |
|---------------------|-----------|----------------|---------------|
| Reduction           | 14        | 83             | 5.93×         |
| Vector Addition     | 14        | 82             | 5.86×         |
| Histogram           | 21        | 114            | 5.43×         |
| Linear Regression   | 48        | 157            | 3.27×         |
| Logistic Regression | 59        | 176            | 2.98×         |
| K-Means             | 68        | 206            | 3.03×         |

SimplePIM reduces the number of lines of effective code by a factor of 2.98× to 5.93×

# Performance Evaluation

- Weak scaling analysis



SimplePIM achieves **comparable performance** for reduction, histogram, and linear regression

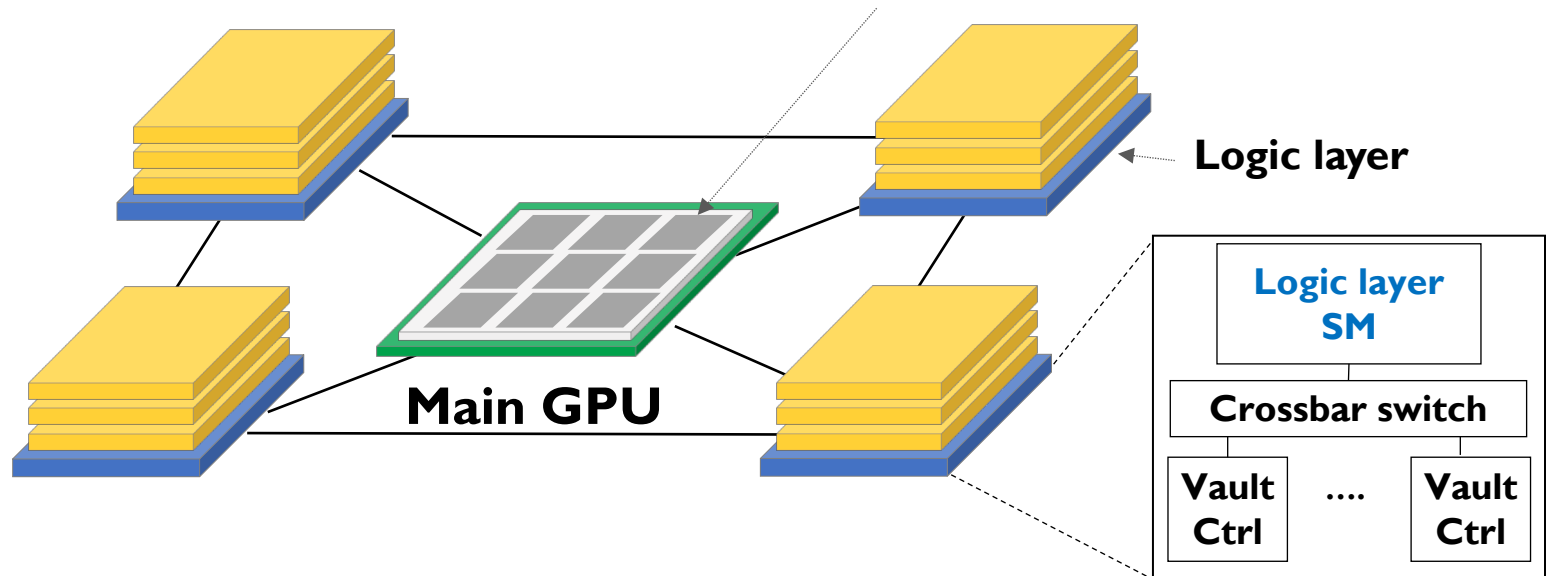
SimplePIM **outperforms hand-optimized implementations** for vector addition, logistic regression, and k-means by 10%-37%



# Truly Distributed GPU Processing with PIM

**3D-stacked memory  
(memory stack)**

**SM (Streaming Multiprocessor)**



```
__global__
void applyScaleFactorsKernel( uint8_T * const out,
                             uint8_T const * const in, const double *factor,
                             size_t const numRows, size_t const numCols )
{
    // Work out which pixel we are working on.
    const int rowIdx = blockIdx.x * blockDim.x + threadIdx.x;
    const int colIdx = blockIdx.y;
    const int sliceIdx = threadIdx.z;

    // Check this thread isn't off the image
    if( rowIdx >= numRows ) return;

    // Compute the index of my element
    size_t linearIdx = rowIdx + colIdx*numRows +
        sliceIdx*numRows*numCols;
```

# Accelerating GPU Execution with PIM (I)

---

- Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu, and Stephen W. Keckler, **"Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems"**

*Proceedings of the 43rd International Symposium on Computer Architecture (ISCA), Seoul, South Korea, June 2016.*

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Session Slides \(pptx\)](#) ([pdf](#))]

## Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems

Kevin Hsieh<sup>‡</sup> Eiman Ebrahimi<sup>†</sup> Gwangsun Kim<sup>\*</sup> Niladrish Chatterjee<sup>†</sup> Mike O'Connor<sup>†</sup>  
Nandita Vijaykumar<sup>‡</sup> Onur Mutlu<sup>§‡</sup> Stephen W. Keckler<sup>†</sup>

<sup>‡</sup>Carnegie Mellon University <sup>†</sup>NVIDIA <sup>\*</sup>KAIST <sup>§</sup>ETH Zürich

# Accelerating GPU Execution with PIM (II)

---

- Ashutosh Pattnaik, Xulong Tang, Adwait Jog, Onur Kayiran, Asit K. Mishra, Mahmut T. Kandemir, Onur Mutlu, and Chita R. Das,  
**"Scheduling Techniques for GPU Architectures with Processing-In-Memory Capabilities"**  
*Proceedings of the 25th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Haifa, Israel, September 2016.

## Scheduling Techniques for GPU Architectures with Processing-In-Memory Capabilities

Ashutosh Pattnaik<sup>1</sup>    Xulong Tang<sup>1</sup>    Adwait Jog<sup>2</sup>    Onur Kayiran<sup>3</sup>  
Asit K. Mishra<sup>4</sup>    Mahmut T. Kandemir<sup>1</sup>    Onur Mutlu<sup>5,6</sup>    Chita R. Das<sup>1</sup>

<sup>1</sup>Pennsylvania State University    <sup>2</sup>College of William and Mary  
<sup>3</sup>Advanced Micro Devices, Inc.    <sup>4</sup>Intel Labs    <sup>5</sup>ETH Zürich    <sup>6</sup>Carnegie Mellon University

# Accelerating Linked Data Structures

---

- Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,  
["Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"](#)  
*Proceedings of the 34th IEEE International Conference on Computer Design (ICCD)*, Phoenix, AZ, USA, October 2016.

## Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh<sup>†</sup> Samira Khan<sup>‡</sup> Nandita Vijaykumar<sup>†</sup>  
Kevin K. Chang<sup>†</sup> Amirali Boroumand<sup>†</sup> Saugata Ghose<sup>†</sup> Onur Mutlu<sup>§†</sup>  
<sup>†</sup>Carnegie Mellon University    <sup>‡</sup>University of Virginia    <sup>§</sup>ETH Zürich

# Accelerating Dependent Cache Misses

---

- Milad Hashemi, Khubaib, Eiman Ebrahimi, Onur Mutlu, and Yale N. Patt, **"Accelerating Dependent Cache Misses with an Enhanced Memory Controller"**

*Proceedings of the 43rd International Symposium on Computer Architecture (ISCA), Seoul, South Korea, June 2016.*

*[Slides (pptx) (pdf)]*

*[Lightning Session Slides (pptx) (pdf)]*

## Accelerating Dependent Cache Misses with an Enhanced Memory Controller

Milad Hashemi\*, Khubaib<sup>†</sup>, Eiman Ebrahimi<sup>‡</sup>, Onur Mutlu<sup>§</sup>, Yale N. Patt\*

*\*The University of Texas at Austin    <sup>†</sup>Apple    <sup>‡</sup>NVIDIA    <sup>§</sup>ETH Zürich & Carnegie Mellon University*

# Accelerating Runahead Execution

---

- Milad Hashemi, Onur Mutlu, and Yale N. Patt,  
**"Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads"**  
*Proceedings of the 49th International Symposium on Microarchitecture (MICRO), Taipei, Taiwan, October 2016.*  
[\[Slides \(pptx\) \(pdf\)\]](#) [\[Lightning Session Slides \(pdf\)\]](#) [\[Poster \(pptx\) \(pdf\)\]](#)  
***Best paper session.***

## Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads

Milad Hashemi\*, Onur Mutlu<sup>§</sup>, Yale N. Patt\*

\**The University of Texas at Austin*    <sup>§</sup>*ETH Zürich*

# Accelerating Climate Modeling

---

- Gagandeep Singh, Dionysios Diamantopoulos, Christoph Hagleitner, Juan Gómez-Luna, Sander Stuijk, Onur Mutlu, and Henk Corporaal,

**"NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling"**

*Proceedings of the 30th International Conference on Field-Programmable Logic and Applications (FPL)*, Gothenburg, Sweden, September 2020.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]

[[Talk Video](#) (23 minutes)]

***Nominated for the Stamatis Vassiliadis Memorial Award.***

## NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling

Gagandeep Singh<sup>a,b,c</sup>

Dionysios Diamantopoulos<sup>c</sup>

Christoph Hagleitner<sup>c</sup>

Juan Gómez-Luna<sup>b</sup>

Sander Stuijk<sup>a</sup>

Onur Mutlu<sup>b</sup>

Henk Corporaal<sup>a</sup>

<sup>a</sup>Eindhoven University of Technology

<sup>b</sup>ETH Zürich

<sup>c</sup>IBM Research Europe, Zurich



# Accelerating Approximate String Matching

- Damla Senol Cali, Gurpreet S. Kalsi, Zülal Bingöl, Can Firtina, Lavanya Subramanian, Jeremie S. Kim, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu, **"GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis"**  
*Proceedings of the 53rd International Symposium on Microarchitecture (MICRO), Virtual, October 2020.*  
[[Lighting Talk Video](#) (1.5 minutes)]  
[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]  
[[Talk Video](#) (18 minutes)]  
[[Slides \(pptx\)](#) ([pdf](#))]

## GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali<sup>†⌘</sup> Gurpreet S. Kalsi<sup>⌘</sup> Zülal Bingöl<sup>▽</sup> Can Firtina<sup>◇</sup> Lavanya Subramanian<sup>‡</sup> Jeremie S. Kim<sup>◇†</sup>  
Rachata Ausavarungnirun<sup>○</sup> Mohammed Alser<sup>◇</sup> Juan Gomez-Luna<sup>◇</sup> Amirali Boroumand<sup>†</sup> Anant Nori<sup>⌘</sup>  
Allison Scibisz<sup>†</sup> Sreenivas Subramoney<sup>⌘</sup> Can Alkan<sup>▽</sup> Saugata Ghose<sup>\*†</sup> Onur Mutlu<sup>◇†▽</sup>  
<sup>†</sup>Carnegie Mellon University   <sup>⌘</sup>Processor Architecture Research Lab, Intel Labs   <sup>▽</sup>Bilkent University   <sup>◇</sup>ETH Zürich  
<sup>‡</sup>Facebook   <sup>○</sup>King Mongkut's University of Technology North Bangkok   <sup>\*</sup>University of Illinois at Urbana-Champaign

# Accelerating Sequence-to-Graph Mapping

- Damla Senol Cali, Konstantinos Kanellopoulos, Joel Lindegger, Zülal Bingöl, Gurpreet S. Kalsi, Ziyi Zuo, Can Firtina, Meryem Banu Cavlak, Jeremie Kim, Nika Mansouri Ghiasi, Gagandeep Singh, Juan Gomez-Luna, Nour Almadhoun Alserr, Mohammed Alser, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,  
**"SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping"**  
*Proceedings of the 49th International Symposium on Computer Architecture (ISCA)*, New York, June 2022.  
[[arXiv version](#)]

## SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping

Damla Senol Cali<sup>1</sup> Konstantinos Kanellopoulos<sup>2</sup> Joël Lindegger<sup>2</sup> Zülal Bingöl<sup>3</sup>  
Gurpreet S. Kalsi<sup>4</sup> Ziyi Zuo<sup>5</sup> Can Firtina<sup>2</sup> Meryem Banu Cavlak<sup>2</sup> Jeremie Kim<sup>2</sup>  
Nika Mansouri Ghiasi<sup>2</sup> Gagandeep Singh<sup>2</sup> Juan Gómez-Luna<sup>2</sup> Nour Almadhoun Alserr<sup>2</sup>  
Mohammed Alser<sup>2</sup> Sreenivas Subramoney<sup>4</sup> Can Alkan<sup>3</sup> Saugata Ghose<sup>6</sup> Onur Mutlu<sup>2</sup>

<sup>1</sup>Bionano Genomics   <sup>2</sup>ETH Zürich   <sup>3</sup>Bilkent University   <sup>4</sup>Intel Labs  
<sup>5</sup>Carnegie Mellon University   <sup>6</sup>University of Illinois Urbana-Champaign

# Accelerating Basecalling + Read Mapping

---

- Haiyu Mao, Mohammed Alser, Mohammad Sadrosadati, Can Firtina, Akanksha Baranwal, Damla Senol Cali, Aditya Manglik, Nour Almadhoun Alserr, and Onur Mutlu,  
**"GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping"**  
*Proceedings of the 55th International Symposium on Microarchitecture (MICRO)*,  
Chicago, IL, USA, October 2022.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Longer Lecture Slides \(pptx\)](#)] [[pdf](#)]  
[[Lecture Video](#) (25 minutes)]  
[[arXiv version](#)]

## **GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping**

Haiyu Mao<sup>1</sup> Mohammed Alser<sup>1</sup> Mohammad Sadrosadati<sup>1</sup> Can Firtina<sup>1</sup> Akanksha Baranwal<sup>1</sup>  
Damla Senol Cali<sup>2</sup> Aditya Manglik<sup>1</sup> Nour Almadhoun Alserr<sup>1</sup> Onur Mutlu<sup>1</sup>  
<sup>1</sup>ETH Zürich <sup>2</sup>Bionano Genomics

# Accelerating Time Series Analysis

---

- Ivan Fernandez, Ricardo Quisiant, Christina Giannoula, Mohammed Alser, Juan Gómez-Luna, Eladio Gutiérrez, Oscar Plata, and Onur Mutlu,  
**"NATSA: A Near-Data Processing Accelerator for Time Series Analysis"**  
*Proceedings of the 38th IEEE International Conference on Computer Design (ICCD)*, Virtual, October 2020.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (10 minutes)]  
[[Source Code](#)]

## NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez<sup>§</sup>

Ricardo Quisiant<sup>§</sup>

Christina Giannoula<sup>†</sup>

Mohammed Alser<sup>‡</sup>

Juan Gómez-Luna<sup>‡</sup>

Eladio Gutiérrez<sup>§</sup>

Oscar Plata<sup>§</sup>

Onur Mutlu<sup>‡</sup>

<sup>§</sup>*University of Malaga*

<sup>†</sup>*National Technical University of Athens*

<sup>‡</sup>*ETH Zürich*

# Accelerating Graph Pattern Mining

- Maciej Besta, Raghavendra Kanakagiri, Grzegorz Kwasniewski, Rachata Ausavarungnirun, Jakub Beránek, Konstantinos Kanellopoulos, Kacper Janda, Zur Vonarburg-Shmaria, Lukas Gianinazzi, Ioana Stefan, Juan Gómez-Luna, Marcin Copik, Lukas Kapp-Schwoerer, Salvatore Di Girolamo, Nils Blach, Marek Konieczny, Onur Mutlu, and Torsten Hoefler,

## **"SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems"**

*Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.*

[[Slides \(pdf\)](#)]

[[Talk Video](#) (22 minutes)]

[[Lightning Talk Video](#) (1.5 minutes)]

[[Full arXiv version](#)]

## **SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems**

Maciej Besta<sup>1</sup>, Raghavendra Kanakagiri<sup>2</sup>, Grzegorz Kwasniewski<sup>1</sup>, Rachata Ausavarungnirun<sup>3</sup>, Jakub Beránek<sup>4</sup>, Konstantinos Kanellopoulos<sup>1</sup>, Kacper Janda<sup>5</sup>, Zur Vonarburg-Shmaria<sup>1</sup>, Lukas Gianinazzi<sup>1</sup>, Ioana Stefan<sup>1</sup>, Juan Gómez-Luna<sup>1</sup>, Marcin Copik<sup>1</sup>, Lukas Kapp-Schwoerer<sup>1</sup>, Salvatore Di Girolamo<sup>1</sup>, Nils Blach<sup>1</sup>, Marek Konieczny<sup>5</sup>, Onur Mutlu<sup>1</sup>, Torsten Hoefler<sup>1</sup>

<sup>1</sup>ETH Zurich, Switzerland  
Thailand

<sup>2</sup>IIT Tirupati, India

<sup>3</sup>King Mongkut's University of Technology North Bangkok,  
<sup>4</sup>Technical University of Ostrava, Czech Republic

<sup>5</sup>AGH-UST, Poland

# Accelerating HTAP Database Systems

---

- Amirali Boroumand, Saugata Ghose, Geraldo F. Oliveira, and Onur Mutlu,  
**"Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design"**  
*Proceedings of the 38th International Conference on Data Engineering (ICDE)*,  
Virtual, May 2022.  
[[arXiv version](#)]  
[[Slides \(pptx\)](#) ([pdf](#))]  
[[Short Talk Slides \(pptx\)](#) ([pdf](#))]

## Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design

Amirali Boroumand<sup>†</sup>      Saugata Ghose<sup>◇</sup>      Geraldo F. Oliveira<sup>‡</sup>      Onur Mutlu<sup>‡</sup>  
<sup>†</sup>*Google*      <sup>◇</sup>*Univ. of Illinois Urbana-Champaign*      <sup>‡</sup>*ETH Zürich*



# Mensa: Highly-Efficient ML Inference

---

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,  
**"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"**  
*Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Virtual, September 2021.  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[Talk Video](#) (14 minutes)]

## Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand<sup>†◇</sup>

Geraldo F. Oliveira<sup>\*</sup>

Saugata Ghose<sup>‡</sup>

Xiaoyu Ma<sup>§</sup>

Berkin Akin<sup>§</sup>

Eric Shiu<sup>§</sup>

Ravi Narayanaswami<sup>§</sup>

Onur Mutlu<sup>\*†</sup>

<sup>†</sup>*Carnegie Mellon Univ.*

<sup>◇</sup>*Stanford Univ.*

<sup>‡</sup>*Univ. of Illinois Urbana-Champaign*

<sup>§</sup>*Google*

<sup>\*</sup>*ETH Zürich*



# Accelerating Data-Intensive Workloads

---

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoun 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<sup>†</sup>   Kiyoun Choi

junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr

Seoul National University

<sup>†</sup>Carnegie Mellon University

# FPGA-based Processing Near Memory

---

- Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu, ["FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications"](#) *IEEE Micro* (**IEEE MICRO**), 2021.

## FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh<sup>◇</sup> Mohammed Alser<sup>◇</sup> Damla Senol Cali<sup>✕</sup>

Dionysios Diamantopoulos<sup>▽</sup> Juan Gómez-Luna<sup>◇</sup>

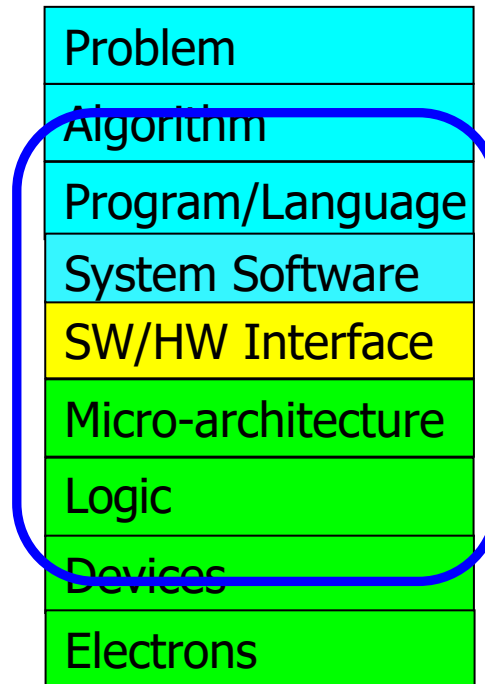
Henk Corporaal<sup>\*</sup> Onur Mutlu<sup>◇✕</sup>

<sup>◇</sup>*ETH Zürich*    <sup>✕</sup>*Carnegie Mellon University*

<sup>\*</sup>*Eindhoven University of Technology*    <sup>▽</sup>*IBM Research Europe*

# We Need to Revisit the Entire Stack

---



**We can get there step by step**

## Processing-in-Memory in the Real World

# PIM Tutorial at ISCA 2024

## ISCA 2024 Memory-Centric Computing Systems Tutorial

Saturday, June 29, Buenos Aires, Argentina

**Organizers:** Geraldo F. Oliveira, Dr. Mohammad Sadrosadati, Ataberk Olgun, Professor Onur Mutlu

**Program:** <https://events.safari.ethz.ch/isca24-memorycentric-tutorial/>

Overview of PIM | PIM taxonomy  
PIM in memory & storage  
Real-world PNM systems  
PUM for bulk bitwise operations  
Programming techniques & tools  
Infrastructures for PIM Research  
Research challenges & opportunities





# Real PIM Tutorials [MICRO'23, ISCA'23, ASPLOS'23, HPCA'23]

- June, March, Feb : Lectures + Hands-on labs + Invited talks



## ISCA 2023 Real-World PIM Tutorial

Search

[Recent Changes](#) [Media Manager](#) [Sitemap](#)

Trace: • [start](#)

### Real-world Processing-in-Memory Systems for Modern Workloads

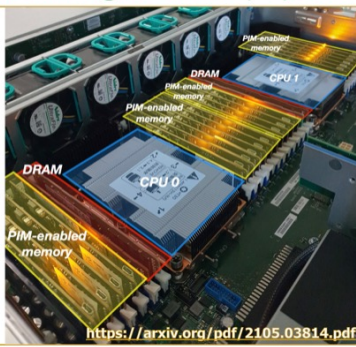
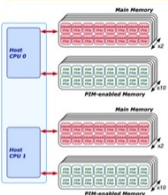
#### Tutorial Description

Processing-in-Memory (PIM) is a computing paradigm that aims at overcoming the data movement bottleneck (i.e., the waste of execution cycles and energy resulting from the back-and-forth data movement between memory units and compute units) by making memory compute-capable.

Explored over several decades since the 1960s, PIM systems are becoming a reality with the advent of the first commercial products and prototypes.

A number of startups (e.g., UPMEM, Neuroblade) are already commercializing real PIM hardware, each with its own design approach and target applications. Several major vendors (e.g., Samsung, SK Hynix, Alibaba) have presented real PIM chip prototypes in the last two years. Most of these architectures have in common that they place compute units near the memory arrays. This type of PIM is called processing near memory (PNM).

#### 2,560-DPU Processing-in-Memory System



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

#### Table of Contents

- [Real-world Processing-in-Memory Systems for Modern Workloads](#)
- [Tutorial Description](#)
- [Organizers](#)
- [Agenda \(June 18, 2023\)](#)
- [Lectures \(tentative\)](#)
- [Hands-on Labs \(tentative\)](#)
- [Learning Materials](#)

PIM can provide large improvements in both performance and energy consumption for many modern applications, thereby enabling a commercially viable way of dealing with huge amounts of data that is bottlenecking our computing systems. Yet, it is critical to (1) study and understand the characteristics that make a workload suitable for a PIM architecture, (2) propose optimization strategies for PIM kernels, and (3) develop programming frameworks and tools that can lower the learning curve and ease the adoption of PIM.

This tutorial focuses on the latest advances in PIM technology, workload characterization for PIM, and programming and optimizing PIM kernels. We will (1) provide an introduction to PIM and taxonomy of PIM systems, (2) give an overview and a rigorous analysis of existing real-world PIM hardware, (3) conduct hand-on labs about important workloads (machine learning, sparse linear algebra, bioinformatics, etc.) using real PIM systems, and (4) shed light on how to improve future PIM systems for such workloads.

<https://events.safari.ethz.ch/isca-pim-tutorial/>

# Real PIM Tutorial [ISCA 2023]

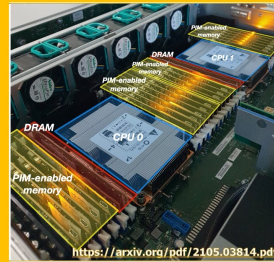
## ■ June 18: Lectures + Hands-on labs + Invited talks

### ISCA 2023 Real-World PIM Tutorial Sunday, June 18, Orlando, Florida

Organizers: Juan Gómez Luna, Onur Mutlu, Ataberk Olgun  
Program: <https://events.safari.ethz.ch/isca-pim-tutorial/>



Overview PIM | PNM | UPMEM PIM |  
PNM for neural networks |  
PNM for recommender systems |  
PNM for ML workloads |  
How to enable PIM? | PUM prototypes  
**Hands-on Labs:** Benchmarking |  
Accelerating real-world workloads



International Symposium on Computer Architecture (ISCA)

## Real-world Processing-in-Memory Systems for Modern Workloads

<https://www.youtube.com/live/GIb5EgSrWk0?feature=share>

Room: Magnolia 16  
Marriott World Center Orlando  
Orlando, FL, USA  
July 18th, 2023

**SAFARI** zoom

ISCA 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

Onur Mutlu Lectures  
33.9K subscribers

Subscribed

57

Share

Download

Clip

...

1,687 views Streamed live on Jun 18, 2023 Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)

[https://www.youtube.com/  
live/GIb5EgSrWk0](https://www.youtube.com/live/GIb5EgSrWk0)

[https://events.safari.ethz.ch/  
isca-pim-tutorial/](https://events.safari.ethz.ch/isca-pim-tutorial/)

### Tutorial Materials

| Time            | Speaker                             | Title                                                                                                               | Materials                                                                                  |
|-----------------|-------------------------------------|---------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| 8:55am-9:00am   | Dr. Juan Gómez Luna                 | Welcome & Agenda                                                                                                    | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 9:00am-10:20am  | Prof. Onur Mutlu                    | Memory-Centric Computing                                                                                            | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 10:20am-11:00am | Dr. Juan Gómez Luna                 | Processing-Near-Memory: Real PNM Architectures / Programming General-purpose PIM                                    | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 11:20am-11:50am | Prof. Izzat El Hajj                 | High-throughput Sequence Alignment using Real Processing-in-Memory Systems                                          | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 11:50am-12:30pm | Dr. Christina Giannoula             | SparseP: Towards Efficient Sparse Matrix Vector Multiplication for Real Processing-In-Memory Systems                | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 2:00pm-2:45pm   | Dr. Sukhan Lee                      | Introducing Real-world HBM-PIM Powered System for Memory-bound Applications                                         | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 2:45pm-3:30pm   | Dr. Juan Gómez Luna / Ataberk Olgun | Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components / PUM Prototypes: PiDRAM | <a href="#">(PDF)</a> <a href="#">(PPT)</a><br><a href="#">(PDF)</a> <a href="#">(PPT)</a> |
| 4:00pm-4:40pm   | Dr. Juan Gómez Luna                 | Accelerating Modern Workloads on a General-purpose PIM System                                                       | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 4:40pm-5:20pm   | Dr. Juan Gómez Luna                 | Adoption Issues: How to Enable PIM?                                                                                 | <a href="#">(PDF)</a> <a href="#">(PPT)</a>                                                |
| 5:20pm-5:30pm   | Dr. Juan Gómez Luna                 | Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture                                | <a href="#">(Handout)</a><br><a href="#">(PDF)</a> <a href="#">(PPT)</a>                   |



# Real PIM Tutorial [ASPLOS 2023]

## ■ March 26: Lectures + Hands-on labs + Invited talks

ASPLOS 2023 Real-World PIM Tutorial

Real-world Processing-in-Memory Systems for Modern Workloads

**Tutorial Description**

Processing-in-Memory (PIM) is a computing paradigm that aims at overcoming the data movement bottleneck (i.e., the waste of execution cycles and energy resulting from the back-and-forth data movement between memory units and compute units) by making memory compute-capable.

Explored over several decades since the 1960s, PIM systems are becoming a reality with the advent of the first commercial products and prototypes.

A number of startups (e.g., UPMEM, Neuroblade) are already commercializing real PIM hardware, each with its own design approach and target applications. Several major vendors (e.g., Samsung, SK Hynix, Alibaba) have presented real PIM chip prototypes in the last two years. Most of these architectures have in common that they place compute units near the memory arrays. This type of PIM is called processing near memory (PNM).

**2,560-DPU Processing-in-Memory System**

PIM can provide large improvements in both performance and energy consumption for many modern applications, thereby enabling a commercially viable way of dealing with huge amounts of data that is bottlenecking our computing systems. Yet, it is critical to (1) study and understand the characteristics that make a workload suitable for a PIM architecture, (2) assess realistic strategies for PIM kernels, and (3)

### Tutorial Materials

| Time            | Speaker                                            | Title                                                                                                | Materials                                                                                        |
|-----------------|----------------------------------------------------|------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| 9:00am-10:20am  | Prof. Onur Mutlu                                   | Memory-Centric Computing                                                                             | <a href="#">(PDF)</a><br><a href="#">(PPT)</a>                                                   |
| 10:40am-12:00pm | Dr. Juan Gómez Luna                                | Processing-Near-Memory: Real PNM Architectures Programming General-purpose PIM                       | <a href="#">(PDF)</a><br><a href="#">(PPT)</a>                                                   |
| 1:40pm-2:20pm   | Prof. Alexandra (Sasha) Fedorova (UBC)             | Processing in Memory in the Wild                                                                     | <a href="#">(PDF)</a><br><a href="#">(PPT)</a>                                                   |
| 2:20pm-3:20pm   | Dr. Juan Gómez Luna & Ataberk Olgun                | Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components           | <a href="#">(PDF)</a><br><a href="#">(PPT)</a><br><a href="#">(PDF)</a><br><a href="#">(PPT)</a> |
| 3:40pm-4:10pm   | Dr. Juan Gómez Luna                                | Adoption issues: How to enable PIM?<br>Accelerating Modern Workloads on a General-purpose PIM System | <a href="#">(PDF)</a><br><a href="#">(PPT)</a><br><a href="#">(PDF)</a><br><a href="#">(PPT)</a> |
| 4:10pm-4:50pm   | Dr. Yongkee Kwon & Eddy (Chanwook) Park (SK Hynix) | System Architecture and Software Stack for GDDR6-AiM                                                 | <a href="#">(PDF)</a><br><a href="#">(PPT)</a>                                                   |
| 4:50pm-5:00pm   | Dr. Juan Gómez Luna                                | Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture                 | <a href="#">(Handout)</a><br><a href="#">(PDF)</a><br><a href="#">(PPT)</a>                      |

ASPLOS 2023 Tutorial

Real-world Processing-in-Memory Systems for Modern Workloads

**Accelerating Modern Workloads on a General-purpose PIM System**

Dr. Juan Gómez Luna  
Professor Onur Mutlu

ETH Zürich SAFARI

ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

Sunday, March 26, 2023

ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

Onur Mutlu Lectures  
32.1K subscribers

Subscribed

33

Share

Clip

Save

views Streamed 7 days ago Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)

LOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

<https://events.safari.ethz.ch/asplos-2023/>

<https://www.youtube.com/watch?v=oYCaLcT0Kmo>

<https://events.safari.ethz.ch/asplos-pim-tutorial/>

# Real PIM Tutorial [HPCA 2023]

## ■ February 26: Lectures + Hands-on labs + Invited Talks

HPCA 2023 Real-World PIM Tutorial

Search

Recent Changes Media Manager Sitemap

Trace: - start

### Real-world Processing-in-Memory Architectures

**Tutorial Description**

Processing-in-Memory (PIM) is a computing paradigm that aims at overcoming the data movement bottleneck (i.e., the waste of execution cycles and energy resulting from the back-and-forth data movement between memory units and compute units) by making memory compute-capable.

Explored over several decades since the 1960s, PIM systems are becoming a reality with the advent of the first commercial products and prototypes.

A number of startups (e.g., UPMEM, Neuroblade, Mythic) are already commercializing real PIM hardware, each with its own design approach and target applications. Several major vendors (e.g., Samsung, SK Hynix, Allbaba) have presented real PIM chip prototypes in the last two years.

#### 2,560-DPU Processing-in-Memory System

Most of these architectures have in common that they place compute units near the memory arrays. But, there is more to come: Academia and Industry are actively exploring other types of PIM by, e.g., exploiting the analog operation of DRAM, SRAM, flash memory and emerging non-volatile memories.

PIM can provide large improvements in both performance and energy consumption, thereby enabling a commercially viable way of dealing with huge amounts of data that is bottlenecking our computing systems. Yet, it is critical to examine and research adoption issues of PIM using especially learnings from real PIM systems that are available today.

This tutorial focuses on the latest advances in PIM technology. We will (1) provide an introduction to PIM and taxonomy of PIM systems, (2) give an overview and a rigorous analysis of existing real-world PIM hardware, (3) conduct hand-on labs using real PIM systems, and (4) shed light on how to enable the adoption of PIM in future computing systems.

### Goal: Processing Inside Memory

Processor Core

Memory

Database

Graphs

Media

Query

Results

Interconnect

- Many questions ... How do we design the:
  - compute-capable memory & controllers?
  - processors & communication units?
  - software & hardware interfaces?
  - system software, compilers, languages?
  - algorithms & theoretical foundations?

HPCA 2023 Tutorial: Real-World Processing-in-Memory Architectures

Onur Mutlu Lectures

32.1K subscribers

1.8K views Streamed 1 month ago Livestream - P&S Data-Centric Architectures: Fundamentally Improving Performance and Energy (Fall 2022)

HPCA 2023 Tutorial: Real-World Processing-in-Memory Architectures

<https://events.safari.ethz.ch/real-pi...>

| Time            | Speaker                 | Title                                                                                                     | Materials                                                       |
|-----------------|-------------------------|-----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| 8:00am-8:40am   | Prof. Onur Mutlu        | Memory-Centric Computing                                                                                  | <a href="#">PDF</a> <a href="#">PPT</a>                         |
| 8:40am-10:00am  | Dr. Juan Gómez Luna     | Processing-Near-Memory: Real PNM Architectures Programming General-purpose PIM                            | <a href="#">PDF</a> <a href="#">PPT</a>                         |
| 10:20am-11:00am | Dr. Dimin Niu           | A 3D Logic-to-DRAM Hybrid Bonding Process-Near-Memory Chip for Recommendation System                      |                                                                 |
| 11:00am-11:40am | Dr. Christina Giannoula | SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures | <a href="#">PDF</a> <a href="#">PPT</a>                         |
| 1:30pm-2:10pm   | Dr. Juan Gómez Luna     | Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components                | <a href="#">PDF</a> <a href="#">PPT</a>                         |
| 2:10pm-2:50pm   | Dr. Manuel Le Gallo     | Deep Learning Inference Using Computational Phase-Change Memory                                           |                                                                 |
| 2:50pm-3:30pm   | Dr. Juan Gómez Luna     | PIM Adoption Issues: How to Enable PIM Adoption?                                                          | <a href="#">PDF</a> <a href="#">PPT</a>                         |
| 3:40pm-5:40pm   | Dr. Juan Gómez Luna     | Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture                      | <a href="#">Handout</a> <a href="#">PDF</a> <a href="#">PPT</a> |

<https://www.youtube.com/watch?v=f5-nT1tbz5w>

<https://events.safari.ethz.ch/real-pim-tutorial/>

- **October 29:** Lectures + Hands-on labs + Invited talks

## 2,560-DPU Processing-in-Memory System

**Researching a New Paradigm: An Experimental Analysis of a Host Processing-in-Memory Architecture**

JOHN GOMEZ LUNA, PhD thesis, University of Zaragoza, 2023  
 JUAN FERNANDEZ, PhD thesis, University of Zaragoza, 2023  
 CARLOS GARCIA, PhD thesis, University of Zaragoza, 2023  
 CARLOS GARCIA, PhD thesis, University of Zaragoza, 2023  
 CARLOS GARCIA, PhD thesis, University of Zaragoza, 2023

Abstract: This paper presents a new paradigm for processing-in-memory (PIM) systems, which is based on the use of a host processing-in-memory architecture. The architecture is designed to enable the execution of PIM applications on a host system, which is composed of a host CPU and a PIM-enabled memory. The host CPU is connected to the PIM-enabled memory via a high-speed interconnect. The PIM-enabled memory is composed of a large number of PIM-enabled memory blocks, which are connected to the host CPU via a high-speed interconnect. The architecture is designed to enable the execution of PIM applications on a host system, which is composed of a host CPU and a PIM-enabled memory. The host CPU is connected to the PIM-enabled memory via a high-speed interconnect. The PIM-enabled memory is composed of a large number of PIM-enabled memory blocks, which are connected to the host CPU via a high-speed interconnect.

Keywords: Processing-in-Memory, Host Processing-in-Memory, PIM-enabled Memory, High-Speed Interconnect, Host CPU, PIM-enabled Memory, High-Speed Interconnect, Host CPU, PIM-enabled Memory, High-Speed Interconnect.

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

**MICRO 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads**

**Onur Mutlu Lectures**

34.6K subscribers

Live in 74 days  
October 29 at 6:00 PM

Notify me

5
+
+
+
+

<https://events.safari.ethz.ch/micro-pim-tutorial>

## 224

# FPGA-based Processing Near Memory

---

- Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu, ["FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications"](#) *IEEE Micro* (**IEEE MICRO**), 2021.

## FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh<sup>◇</sup> Mohammed Alser<sup>◇</sup> Damla Senol Cali<sup>✕</sup>

Dionysios Diamantopoulos<sup>▽</sup> Juan Gómez-Luna<sup>◇</sup>

Henk Corporaal<sup>★</sup> Onur Mutlu<sup>◇✕</sup>

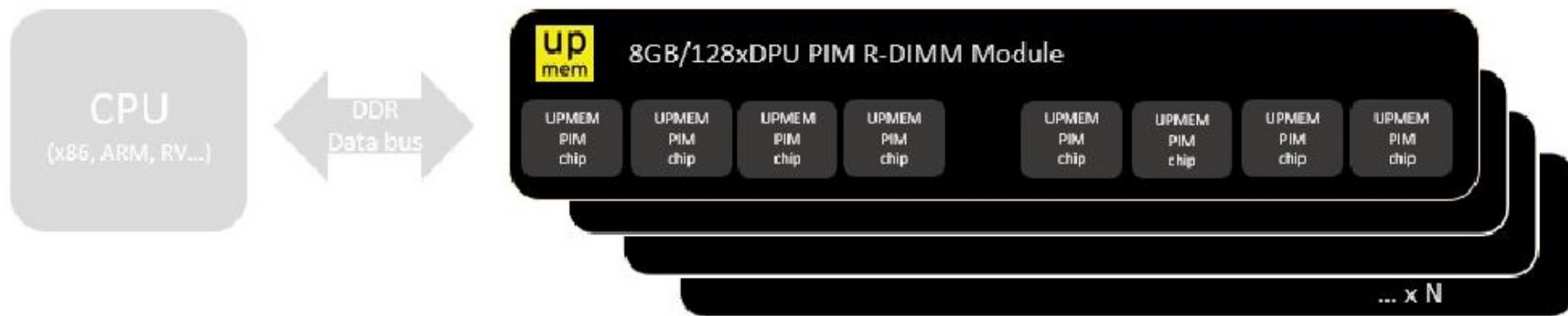
<sup>◇</sup>*ETH Zürich*    <sup>✕</sup>*Carnegie Mellon University*

<sup>★</sup>*Eindhoven University of Technology*    <sup>▽</sup>*IBM Research Europe*



# 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



# 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

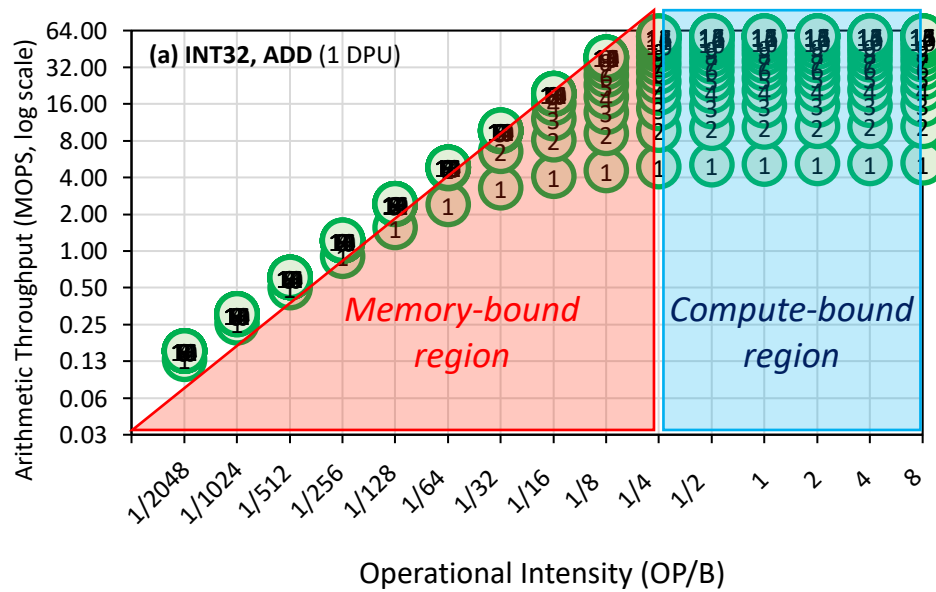
ONUR MUTLU, ETH Zürich, Switzerland

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.

# Key Takeaway 1



The throughput saturation point is as low as  $\frac{1}{4}$  OP/B, i.e., 1 integer addition per every 32-bit element fetched

## KEY TAKEAWAY 1

The UPMEM PIM architecture is fundamentally compute bound. As a result, the most suitable workloads are memory-bound.

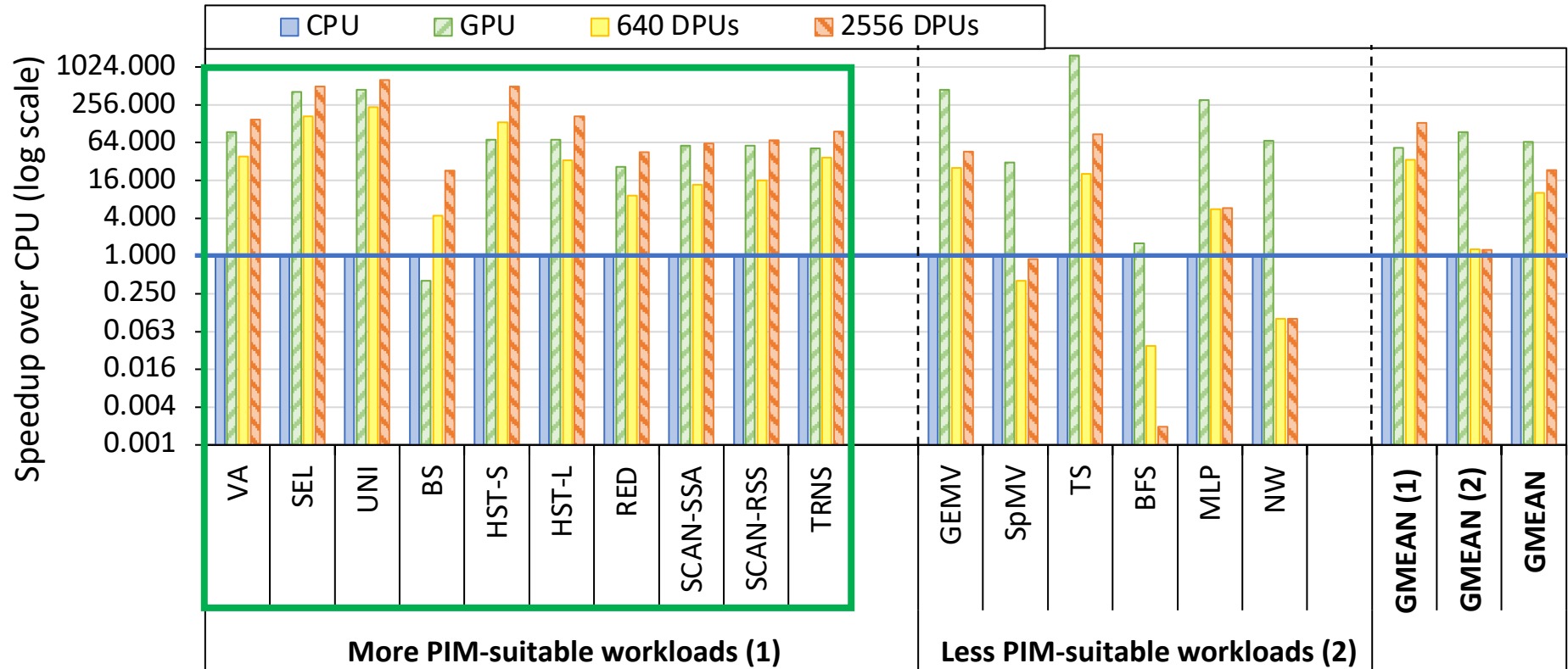


# Key Takeaway 2

Table 4: Evaluated CPU, GPU, and UPMEM-based PIM Systems.

| System                          | Process Node | Processor Cores       |           |                  | Memory    |                 | TDP                |
|---------------------------------|--------------|-----------------------|-----------|------------------|-----------|-----------------|--------------------|
|                                 |              | Total Cores           | Frequency | Peak Performance | Capacity  | Total Bandwidth |                    |
| Intel Xeon E3-1225 v6 CPU [241] | 14 nm        | 4 (8 threads)         | 3.3 GHz   | 26.4 GFLOPS*     | 32 GB     | 37.5 GB/s       | 73 W               |
| NVIDIA Titan V GPU [277]        | 14 nm        | 80 (5,120 SIMD lanes) | 1.2 GHz   | 12,288.0 GFLOPS  | 12 GB     | 652.8 GB/s      | 250 W              |
| 2,556-DPU PIM System            | 2x nm        | 2,556 <sup>†</sup>    | 350 MHz   | 894.6 GOPS       | 159.75 GB | 1.7 TB/s        | 383 W <sup>†</sup> |
| 640-DPU PIM System              | 2x nm        | 640                   | 267 MHz   | 170.9 GOPS       | 40 GB     | 333.75 GB/s     | 96 W <sup>†</sup>  |

\* Estimated GFLOPS = 3.3 GHz × 4 cores × 2 instructions per cycle.  
<sup>†</sup> Estimated TDP =  $\frac{\text{Total DPU}}{\text{DPU}_s/\text{chip}} \times 1.2 \text{ W/chip}$  [199].



## KEY TAKEAWAY 2

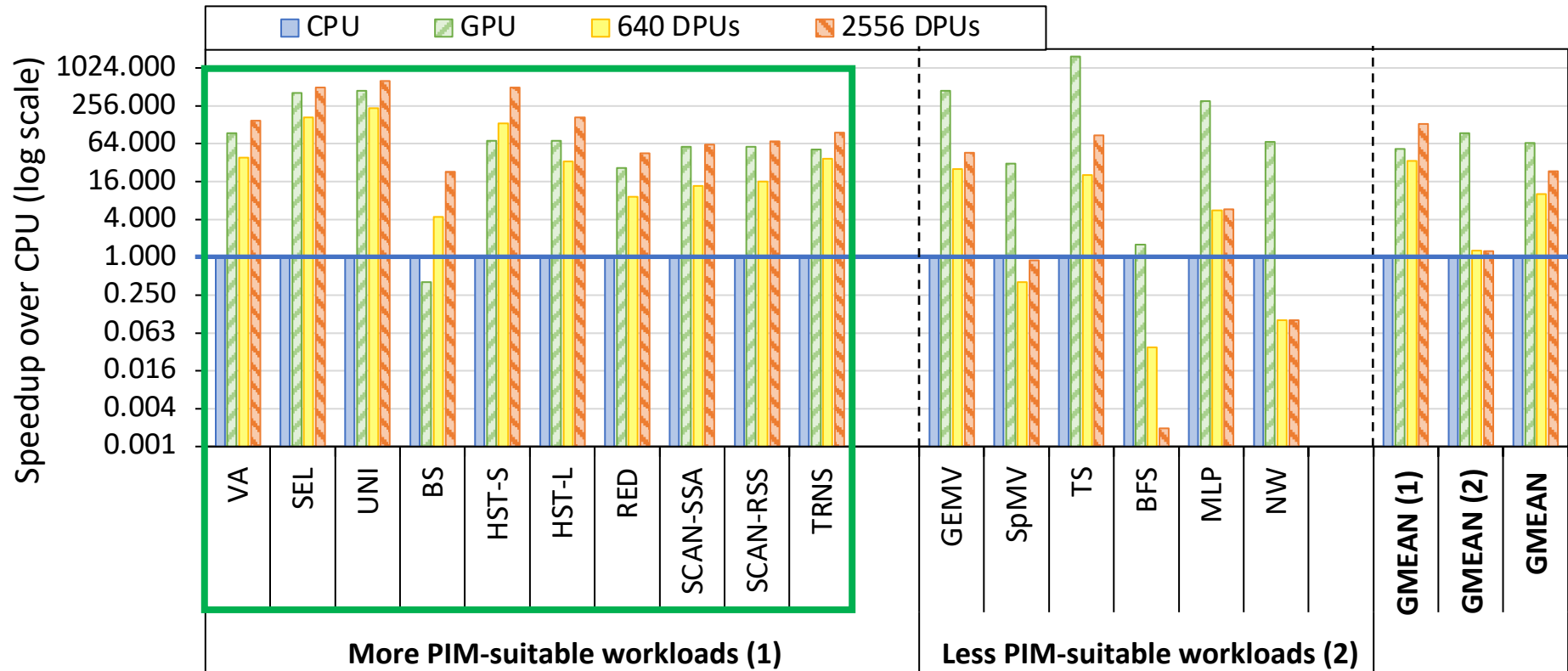
The most well-suited workloads for the UPMEM PIM architecture use no arithmetic operations or use only simple operations (e.g., bitwise operations and integer addition/subtraction).

# Key Takeaway 3

Table 4: Evaluated CPU, GPU, and UPMEM-based PIM Systems.

| System                          | Process Node | Processor Cores       |           |                  | Memory    |                 | TDP                |
|---------------------------------|--------------|-----------------------|-----------|------------------|-----------|-----------------|--------------------|
|                                 |              | Total Cores           | Frequency | Peak Performance | Capacity  | Total Bandwidth |                    |
| Intel Xeon E3-1225 v6 CPU [241] | 14 nm        | 4 (8 threads)         | 3.3 GHz   | 26.4 GFLOPS*     | 32 GB     | 37.5 GB/s       | 73 W               |
| NVIDIA Titan V GPU [277]        | 14 nm        | 80 (5,120 SIMD lanes) | 1.2 GHz   | 12,288.0 GFLOPS  | 12 GB     | 652.8 GB/s      | 250 W              |
| 2,556-DPU PIM System            | 2x nm        | 2,556 <sup>†</sup>    | 350 MHz   | 894.6 GOPS       | 159.75 GB | 1.7 TB/s        | 383 W <sup>†</sup> |
| 640-DPU PIM System              | 2x nm        | 640                   | 267 MHz   | 170.9 GOPS       | 40 GB     | 333.75 GB/s     | 96 W <sup>†</sup>  |

\*Estimated GFLOPS = 3.3 GHz × 4 cores × 2 instructions per cycle.  
<sup>†</sup>Estimated TDP =  $\frac{\text{Total DPU}}{\text{DPU}_1/\text{chip}} \times 1.2 \text{ W/chip}$  [199].



## KEY TAKEAWAY 3

The most well-suited workloads for the UPMEM PIM architecture require little or no communication across DPUs (inter-DPU communication).

# UPMEM PIM System Summary & Analysis

---

- 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

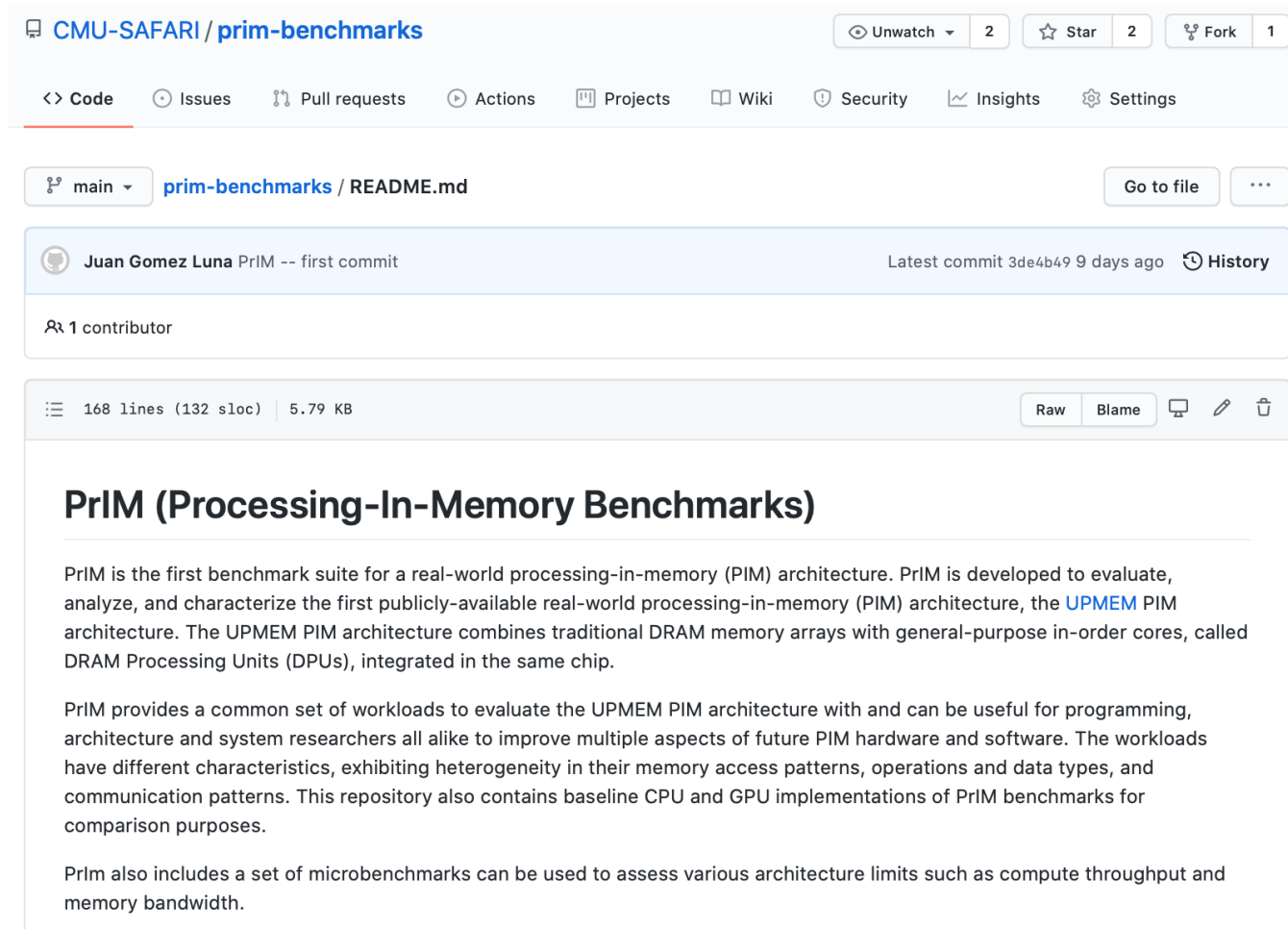
|                   |                                          |                                 |                                                    |                     |                   |
|-------------------|------------------------------------------|---------------------------------|----------------------------------------------------|---------------------|-------------------|
| Juan Gómez-Luna   | Izzat El Hajj                            | Ivan Fernandez                  | Christina Giannoula                                | Geraldo F. Oliveira | Onur Mutlu        |
| <i>ETH Zürich</i> | <i>American University<br/>of Beirut</i> | <i>University<br/>of Malaga</i> | <i>National Technical<br/>University of Athens</i> | <i>ETH Zürich</i>   | <i>ETH Zürich</i> |

# PrIM Benchmarks: Application Domains

| Domain                | Benchmark                     | Short name |
|-----------------------|-------------------------------|------------|
| Dense linear algebra  | Vector Addition               | VA         |
|                       | Matrix-Vector Multiply        | GEMV       |
| Sparse linear algebra | Sparse Matrix-Vector Multiply | SpMV       |
| Databases             | Select                        | SEL        |
|                       | Unique                        | UNI        |
| Data analytics        | Binary Search                 | BS         |
|                       | Time Series Analysis          | TS         |
| Graph processing      | Breadth-First Search          | BFS        |
| Neural networks       | Multilayer Perceptron         | MLP        |
| Bioinformatics        | Needleman-Wunsch              | NW         |
| Image processing      | Image histogram (short)       | HST-S      |
|                       | Image histogram (large)       | HST-L      |
| Parallel primitives   | Reduction                     | RED        |
|                       | Prefix sum (scan-scan-add)    | SCAN-SSA   |
|                       | Prefix sum (reduce-scan-scan) | SCAN-RSS   |
|                       | Matrix transposition          | TRNS       |

# PrIM Benchmarks are Open Source

- All microbenchmarks, benchmarks, and scripts
- <https://github.com/CMU-SAFARI/prim-benchmarks>



CMU-SAFARI / **prim-benchmarks** Unwatch 2 Star 2 Fork 1

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main prim-benchmarks / README.md Go to file ...

Juan Gomez Luna PrIM -- first commit Latest commit 3de4b49 9 days ago History

1 contributor

168 lines (132 sloc) 5.79 KB Raw Blame

## PrIM (Processing-In-Memory Benchmarks)

PrIM is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publicly-available real-world processing-in-memory (PIM) architecture, the [UPMEM](#) 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.

PrIM provides a common set of workloads to evaluate the UPMEM PIM architecture with and can be useful for programming, architecture and system researchers all alike to improve multiple aspects of future PIM hardware and software. The workloads have different characteristics, exhibiting heterogeneity in their memory access patterns, operations and data types, and communication patterns. This repository also contains baseline CPU and GPU implementations of PrIM benchmarks for comparison purposes.

PrIm also includes a set of microbenchmarks can be used to assess various architecture limits such as compute throughput and memory bandwidth.

# Understanding a Modern PIM Architecture

---

## Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System

**JUAN GÓMEZ-LUNA<sup>1</sup>, IZZAT EL HAJJ<sup>2</sup>, IVAN FERNANDEZ<sup>1,3</sup>, CHRISTINA GIANNOULA<sup>1,4</sup>,  
GERALDO F. OLIVEIRA<sup>1</sup>, AND ONUR MUTLU<sup>1</sup>**

<sup>1</sup>ETH Zürich

<sup>2</sup>American University of Beirut

<sup>3</sup>University of Malaga

<sup>4</sup>National Technical University of Athens

Corresponding author: Juan Gómez-Luna (e-mail: juang@ethz.ch).

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

<https://github.com/CMU-SAFARI/prim-benchmarks>

# Understanding a Modern PIM Architecture



The image shows a YouTube video player interface. The video title is "Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization". The speaker is Juan Gómez Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, and Onur Mutlu. The video is from the channel "Onur Mutlu Lectures" and has 18.7K subscribers. The video is titled "SAFARI Live Seminar: Understanding a Modern Processing-in-Memory Architecture" and has 2,579 views. The video is streamed live on Jul 12, 2021. The video player shows the title, speaker names, and links to the arXiv paper and GitHub repository. The video player also shows the ETH Zürich and SAFARI logos. The video player has a progress bar at 2:26 / 2:57:10. The video player has a like button with 93 likes, a comment button with 0 comments, a share button, a save button, and a bell icon. The video player has a "SUBSCRIBED" button.

**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>

ETH Zürich SAFARI

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

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

93 0 SHARE SAVE ...

 **Onur Mutlu Lectures**  
18.7K subscribers

SUBSCRIBED



# ML Training on a Real PIM System

---

## Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna<sup>1</sup> Yuxin Guo<sup>1</sup> Sylvan Brocard<sup>2</sup> Julien Legriel<sup>2</sup>  
Remy Cimadomo<sup>2</sup> Geraldo F. Oliveira<sup>1</sup> Gagandeep Singh<sup>1</sup> Onur Mutlu<sup>1</sup>

<sup>1</sup>ETH Zürich <sup>2</sup>UPMEM

## An Experimental Evaluation of Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna<sup>1</sup> Yuxin Guo<sup>1</sup> Sylvan Brocard<sup>2</sup> Julien Legriel<sup>2</sup>  
Remy Cimadomo<sup>2</sup> Geraldo F. Oliveira<sup>1</sup> Gagandeep Singh<sup>1</sup> Onur Mutlu<sup>1</sup>

<sup>1</sup>ETH Zürich <sup>2</sup>UPMEM

Short version: <https://arxiv.org/pdf/2206.06022.pdf>

Long version: <https://arxiv.org/pdf/2207.07886.pdf>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=11226s>

# ML Training on a Real PIM System

---

- Need to optimize data representation
  - (1) fixed-point
  - (2) quantization
  - (3) hybrid precision
- Use **lookup tables (LUTs)** to implement complex functions (e.g., sigmoid)
- Optimize data placement & layout for **streaming**
- Large speedups: **2.8X/27X vs. CPU, 1.3x/3.2x vs. GPU**

# ML Training on Real PIM Talk Video

## Comparison to CPU and GPU (III)

• Decision tree and K-means with Criteo 1TB dataset

**(a) Decision Tree**

| Configuration | Execution Time (ms) |
|---------------|---------------------|
| PIM Kernel    | ~10,000             |
| CPU-PIM       | ~10,000             |
| Inter PIM     | ~10,000             |
| PIM-CPU       | ~10,000             |
| CPU           | ~100,000            |
| GPU           | ~10,000             |

**(b) K-means**

| Configuration | Execution Time (ms) |
|---------------|---------------------|
| PIM Kernel    | ~10,000             |
| CPU-PIM       | ~10,000             |
| Inter PIM     | ~10,000             |
| PIM-CPU       | ~10,000             |
| CPU           | ~100,000            |
| GPU           | ~10,000             |

**PIM version of DTR is 62x faster than the CPU version and 4.5x faster than the GPU version**

**PIM version of KME is 2.7x faster than the CPU version and 3.2x faster than the GPU version**

13:39 / 15:20 • Comparison to CPU and GPU (II) >

Machine Learning Training on Memory-centric Computing Systems, Juan Gómez-Luna for ISPASS 2023



Onur Mutlu Lectures  
32.9K subscribers

Analytics

Edit video

9

Share

Download

Clip

Save

...

242 views 11 days ago Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)  
Evaluating Machine Learning Workloads on Memory-centric Computing Systems

# ML Training on Real PIM Systems

---

- Juan Gómez Luna, Yuxin Guo, Sylvan Brocard, Julien Legriel, Remy Cimadomo, Geraldo F. Oliveira, Gagandeep Singh, and Onur Mutlu,  
**"Evaluating Machine Learning Workloads on Memory-Centric Computing Systems"**  
*Proceedings of the 2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Raleigh, North Carolina, USA, April 2023.  
[[arXiv version](#), 16 July 2022.]  
[[PIM-ML Source Code](#)]  
***Best paper session.***

## An Experimental Evaluation of Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna<sup>1</sup> Yuxin Guo<sup>1</sup> Sylvan Brocard<sup>2</sup> Julien Legriel<sup>2</sup>  
Remy Cimadomo<sup>2</sup> Geraldo F. Oliveira<sup>1</sup> Gagandeep Singh<sup>1</sup> Onur Mutlu<sup>1</sup>  
<sup>1</sup>ETH Zürich <sup>2</sup>UPMEM

<https://github.com/CMU-SAFARI/pim-ml>

# SpMV Multiplication on Real PIM Systems

---

- Appears at SIGMETRICS 2022

## ***SparseP*: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems**

CHRISTINA GIANNOULA, ETH Zürich, Switzerland and National Technical University of Athens, Greece

IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

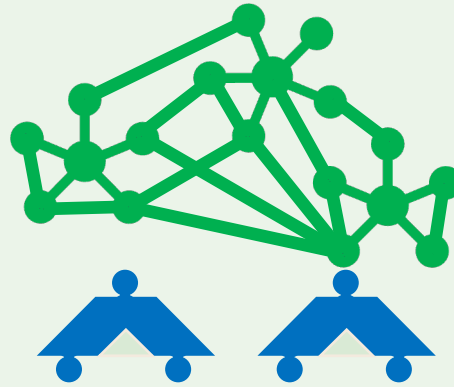
NECTARIOS KOZIRIS, National Technical University of Athens, Greece

GEORGIOS GOUMAS, National Technical University of Athens, Greece

ONUR MUTLU, ETH Zürich, Switzerland

<https://arxiv.org/pdf/2201.05072.pdf>

<https://github.com/CMU-SAFARI/SparseP>



# SparseP

Towards Efficient Sparse Matrix Vector Multiplication  
on Real Processing-In-Memory Architectures

Christina Giannoula

Ivan Fernandez, Juan Gomez-Luna,

Nectarios Koziris, Georgios Goumas, Onur Mutlu

**SAFARI** **ETH** zürich

 National Technical University of Athens  
**CSLab**



UNIVERSIDAD  
DE MÁLAGA

# SparseP: Key Contributions

## 1. Efficient SpMV kernels for current & future PIM systems

- SparseP library = 25 SpMV kernels
  - Compression, data types, data partitioning, synchronization, load balancing

SparseP is Open-Source

SparseP: <https://github.com/CMU-SAFARI/SparseP>

## 2. Comprehensive analysis of SpMV on the first commercially-available real PIM system



- 26 sparse matrices
- Comparisons to state-of-the-art CPU and GPU systems
- Recommendations for software, system and hardware designers

Recommendations for Architects and Programmers

Full Paper: <https://arxiv.org/pdf/2201.05072.pdf>



# SparseP Talk Video



The screenshot shows a YouTube video player with a presentation slide. The slide features the SparseP logo, which consists of a green network graph above two blue stylized human figures. Below the logo, the title "SparseP" is written in a large blue font, followed by the subtitle "Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures" in a smaller blue font. The presenter's name, "Christina Giannoula", is listed in red, followed by the names of other contributors: "Ivan Fernandez, Juan Gomez-Luna, Nectarios Koziris, Georgios Goumas, Onur Mutlu". At the bottom of the slide, there are logos for "SAFARI", "ETH zürich", "CSLab", and "UNIVERSITY OF MALAGA". A small video inset in the top right corner shows a woman, identified as "Christina Gian...", speaking. The video player interface includes a progress bar at 0:02 / 55:25, a volume icon, and various control buttons like play, pause, and full screen.

Processing-in-Memory Course: Lecture 11: SpMV on a Real PIM Architecture - Spring 2022

149 views • Streamed live on May 19, 2022

👍 12    🗑 DISLIKE    ➦ SHARE    ⬇ DOWNLOAD    ✂ CLIP    ≡+ SAVE    ...



Onur Mutlu Lectures  
25K subscribers

ANALYTICS

EDIT VIDEO

# More on SparseP

---

Christina Giannoula, Ivan Fernandez, Juan Gomez-Luna, Nectarios Koziris, Georgios Goumas, and Onur Mutlu,

## **"SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures"**

*Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**), Mumbai, India, June 2022.*

[[Extended arXiv Version](#)]

[[Abstract](#)]

[[Slides \(pptx\) \(pdf\)](#)]

[[Long Talk Slides \(pptx\) \(pdf\)](#)]

[[SparseP Source Code](#)]

[[Talk Video \(16 minutes\)](#)]

[[Long Talk Video \(55 minutes\)](#)]

## **SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems**

CHRISTINA GIANNOULA, ETH Zürich, Switzerland and National Technical University of Athens, Greece

IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

NECTARIOS KOZIRIS, National Technical University of Athens, Greece

GEORGIOS GOUMAS, National Technical University of Athens, Greece

ONUR MUTLU, ETH Zürich, Switzerland

**<https://github.com/CMU-SAFARI/SparseP>**

# Transcendental Functions on Real PIM Systems

---

- Maurus Item, Juan Gómez Luna, Yuxin Guo, Geraldo F. Oliveira, Mohammad Sadrosadati, and Onur Mutlu,  
**"TransPimLib: Efficient Transcendental Functions for Processing-in-Memory Systems"**  
*Proceedings of the 2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Raleigh, North Carolina, USA, April 2023.  
[[arXiv version](#)]  
[[Slides \(pptx\)](#)] [[pdf](#)]  
[[TransPimLib Source Code](#)]  
[[Talk Video](#) (17 minutes)]

## TransPimLib: Efficient Transcendental Functions for Processing-in-Memory Systems

Maurus Item  
Geraldo F. Oliveira

Juan Gómez-Luna  
Mohammad Sadrosadati

Yuxin Guo  
Onur Mutlu

ETH Zürich

<https://github.com/CMU-SAFARI/transpimlib>

# Sequence Alignment on Real PIM Systems

---

- Safaa Diab, Amir Nassereldine, Mohammed Alser, Juan Gómez Luna, Onur Mutlu, and Izzat El Hajj,  
**"A Framework for High-throughput Sequence Alignment using Real Processing-in-Memory Systems"**  
**Bioinformatics**, [published online on] 27 March 2023.  
[[Online link at Bioinformatics Journal](#)]  
[[arXiv preprint](#)]  
[[AiM Source Code](#)]

## A Framework for High-throughput Sequence Alignment using Real Processing-in-Memory Systems

Safaa Diab<sup>1</sup> Amir Nassereldine<sup>1</sup> Mohammed Alser<sup>2</sup> Juan Gómez Luna<sup>2</sup>  
Onur Mutlu<sup>2</sup> Izzat El Hajj<sup>1</sup>

<sup>1</sup>American University of Beirut <sup>2</sup>ETH Zürich

<https://github.com/CMU-SAFARI/alignment-in-memory>



## Summary

- Sequence alignment on traditional systems is limited by the **memory bandwidth bottleneck**
- **Processing-in-memory (PIM)** overcomes this bottleneck by placing cores near the memory
- Our framework, **Alignment-in-Memory (AIM)**, is a PIM framework that supports multiple alignment algorithms (NW, SWG, GenASM, WFA)
  - Implemented on UPMEM, the first real PIM system
- Results show **substantial speedups over both CPUs (1.8X-28X) and GPUs (1.2X-2.7X)**
- AIM is available at:
  - <https://github.com/CMU-SAFARI/alignment-in-memory>

# Homomorphic Operations on Real PIM Systems

---

- Harshita Gupta, Mayank Kabra, Juan Gómez-Luna, Konstantinos Kanellopoulos, and Onur Mutlu,  
**"Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System"**  
*Proceedings of the 2023 IEEE International Symposium on Workload Characterization Poster Session (IISWC)*, Ghent, Belgium, October 2023.  
[[arXiv version](#)]  
[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]  
[[Poster \(pptx\)](#) ([pdf](#))]

## Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System

Harshita Gupta\*   Mayank Kabra\*   Juan Gómez-Luna   Konstantinos Kanellopoulos   Onur Mutlu  
ETH Zürich

# Accelerating ML Training on Real PIM Systems

---

- <https://arxiv.org/pdf/2404.07164>

## **Analysis of Distributed Optimization Algorithms on a Real Processing-In-Memory System**

Steve Rhyner<sup>1</sup>   Haocong Luo<sup>1</sup>   Juan Gómez-Luna<sup>2</sup>   Mohammad Sadrosadati<sup>1</sup>  
Jiawei Jiang<sup>3</sup>   Ataberk Olgun<sup>1</sup>   Harshita Gupta<sup>1</sup>   Ce Zhang<sup>4</sup>   Onur Mutlu<sup>1</sup>  
<sup>1</sup>ETH Zurich   <sup>2</sup>NVIDIA   <sup>3</sup>Wuhan University   <sup>4</sup>University of Chicago



# Accelerating ML Training on Real PIM Systems

---

- <https://arxiv.org/pdf/2404.07164>

## 8. Conclusion

We evaluate and train ML models on large-scale datasets with centralized parallel optimization algorithms on a *real-world* PIM architecture. We show the importance of carefully *choosing* the distributed optimization algorithm that fits PIM and analyze tradeoffs. We demonstrate that *commercial* general-purpose PIM systems can be a viable alternative for many ML training workloads on large-scale datasets to processor-centric architectures. Our results demonstrate the necessity of adapting PIM architectures to enable inter-DPU communication to overcome scalability challenges for many ML training workloads and discuss decentralized parallel SGD optimization algorithms as a potential solution.

# Accelerating GNNs on Real PIM Systems

---

- <https://arxiv.org/pdf/2402.16731>

## Accelerating Graph Neural Networks on Real Processing-In-Memory Systems

Christina Giannoula<sup>\*†</sup>, Peiming Yang<sup>\*</sup>, Ivan Fernandez Vega<sup>§†</sup>, Jiacheng Yang<sup>\*</sup>, Yu Xin Li<sup>\*</sup>,  
Juan Gomez Luna<sup>¶†</sup>, Mohammad Sadrosadati<sup>†</sup>, Onur Mutlu<sup>†‡</sup>, Gennady Pekhimenko<sup>\*||</sup>

<sup>\*</sup>University of Toronto      <sup>†</sup>ETH Zürich      <sup>§</sup>Barcelona Supercomputing Center

<sup>¶</sup>NVIDIA      <sup>‡</sup>Stanford      <sup>||</sup>CentML

# Accelerating GNNs on Real PIM Systems

---

- <https://arxiv.org/pdf/2402.16731>

***Abstract***—Graph Neural Networks (GNNs) are emerging ML models to analyze graph-structure data. Graph Neural Network (GNN) execution involves both compute-intensive and memory-intensive kernels, the latter dominates the total time, being significantly bottlenecked by data movement between memory and processors. Processing-In-Memory (PIM) systems can alleviate this data movement bottleneck by placing simple processors near or inside to memory arrays. In this work, we introduce PyGim, an efficient ML framework that accelerates GNNs on real PIM systems. We propose intelligent parallelization techniques for memory-intensive kernels of GNNs tailored for real PIM systems, and develop handy Python API for them. We provide hybrid GNN execution, in which the compute-intensive and memory-intensive kernels are executed in processor-centric and memory-centric computing systems, respectively, to match their algorithmic nature. We extensively evaluate PyGim on a real-world PIM system with 1992 PIM cores using emerging GNN models, and demonstrate that it outperforms its state-of-the-art CPU counterpart on Intel Xeon by on average  $3.04\times$ , and achieves higher resource utilization than CPU and GPU systems. Our work provides useful recommendations for software, system and hardware designers. PyGim will be open-sourced to enable the widespread use of PIM systems in GNNs.

# Samsung Function-in-Memory DRAM (2021)



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

Korea on February 17, 2021

Audio



Share



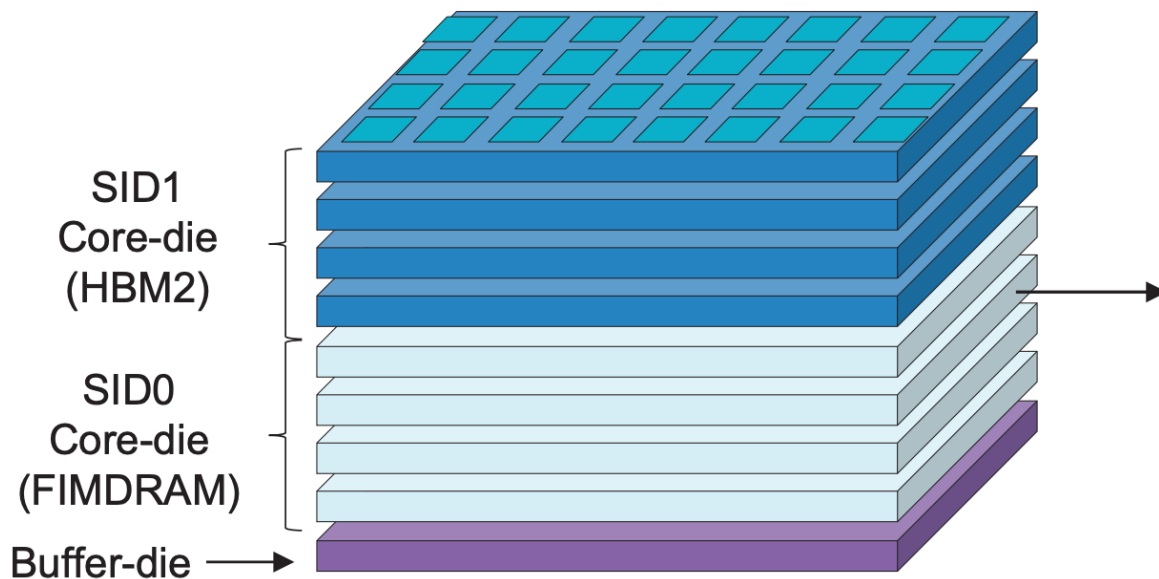
*The new architecture will deliver over twice the system performance and reduce energy consumption by more than 70%*

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."

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

<sup>1</sup>Samsung Electronics, Hwaseong, Korea

<sup>2</sup>Samsung Electronics, San Jose, CA

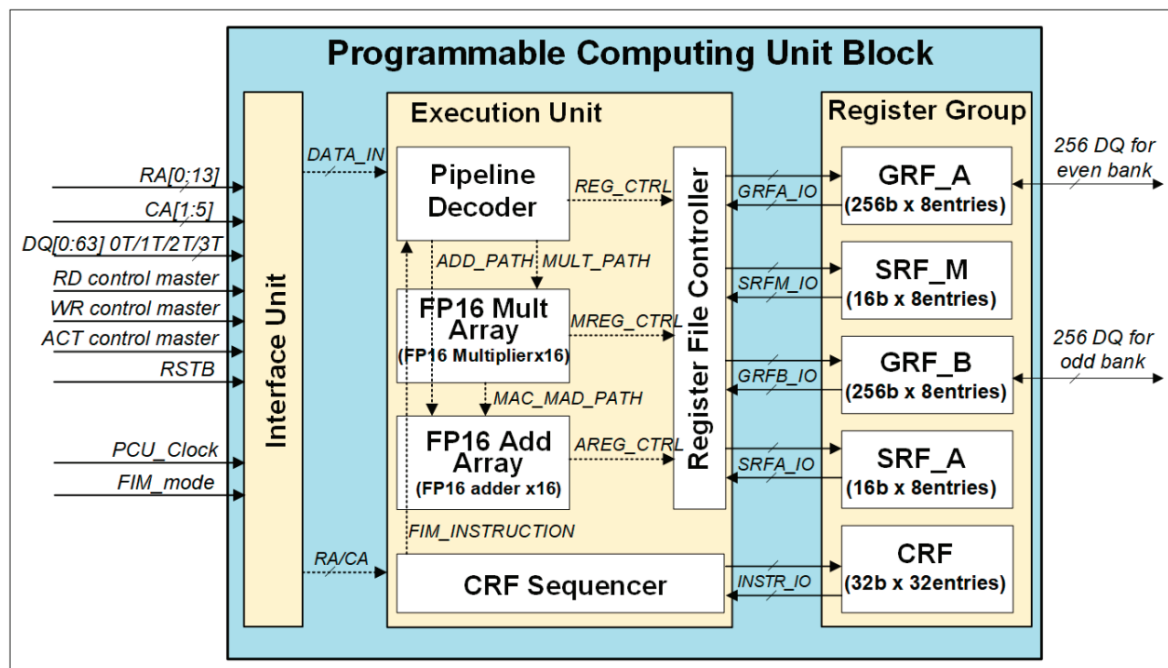
<sup>3</sup>Samsung Electronics, Suwon, Korea

# Samsung Function-in-Memory DRAM (2021)

## Programmable Computing Unit

### ■ Configuration of PCU block

- Interface unit to control data flow
- Execution unit to perform operations
- Register group
  - 32 entries of CRF for instruction memory
  - 16 GRF for weight and accumulation
  - 16 SRF to store constants for MAC operations



[Block diagram of PCU in FIMDRAM]

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

<sup>1</sup>Samsung Electronics, Hwaseong, Korea  
<sup>2</sup>Samsung Electronics, San Jose, CA  
<sup>3</sup>Samsung Electronics, Suwon, Korea

# Samsung Function-in-Memory DRAM (2021)

[Available instruction list for FIM operation]

| Type           | CMD  | Description                 |
|----------------|------|-----------------------------|
| Floating Point | ADD  | FP16 addition               |
|                | MUL  | FP16 multiplication         |
|                | MAC  | FP16 multiply-accumulate    |
|                | MAD  | FP16 multiply and add       |
| Data Path      | MOVE | Load or store data          |
|                | FILL | Copy data from bank to GRFs |
| Control Path   | NOP  | Do nothing                  |
|                | JUMP | Jump instruction            |
|                | EXIT | Exit instruction            |

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

<sup>1</sup>Samsung Electronics, Hwaseong, Korea

<sup>2</sup>Samsung Electronics, San Jose, CA

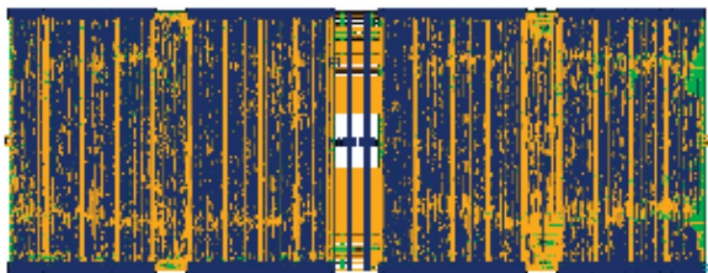
<sup>3</sup>Samsung Electronics, Suwon, Korea



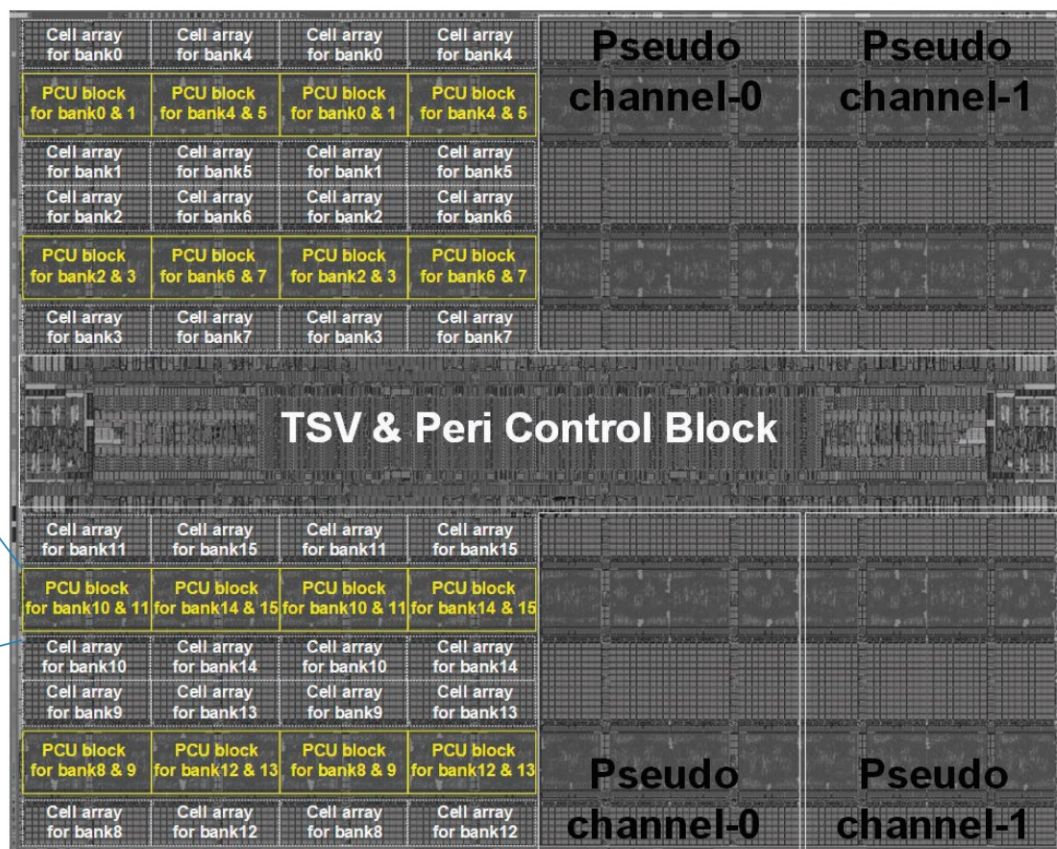
# Samsung Function-in-Memory DRAM (2021)

## 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

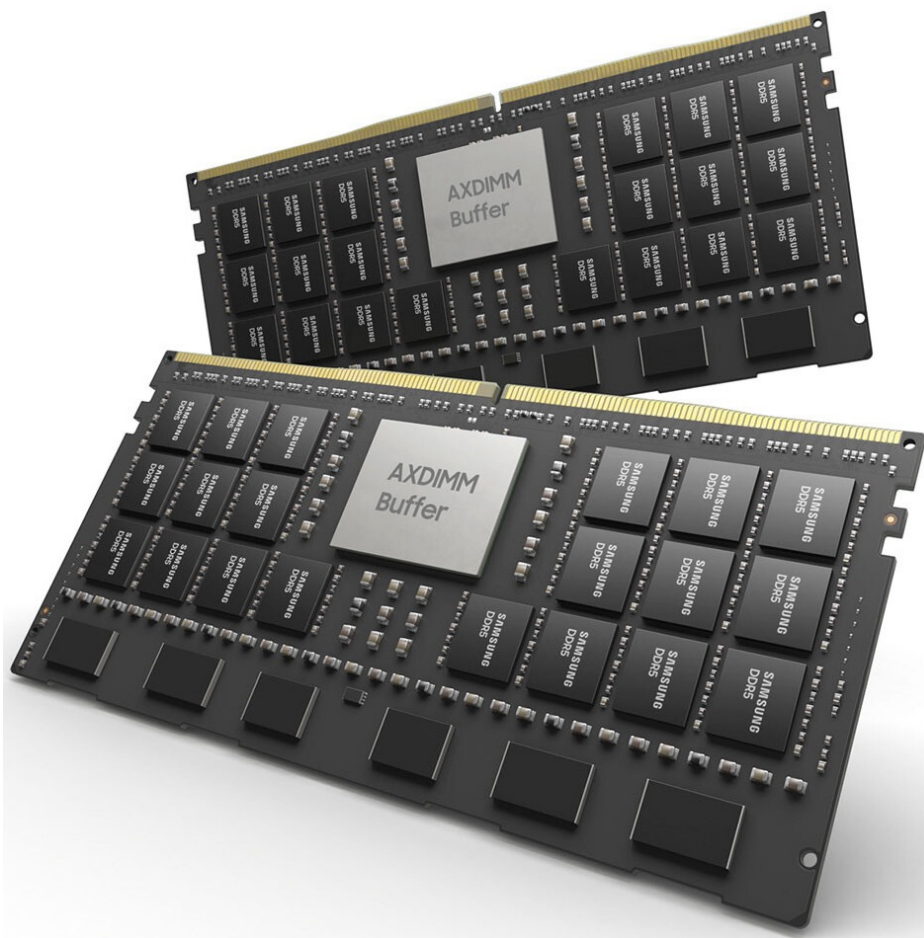
25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

Young-Cheon Kwon<sup>1</sup>, Suk Han Lee<sup>1</sup>, Jaehoon Lee<sup>1</sup>, Sang-Hyuk Kwon<sup>1</sup>, Je Min Ryu<sup>1</sup>, Jong-Pil Son<sup>1</sup>, Seongil O<sup>1</sup>, Hak-Soo Yu<sup>1</sup>, Haesuk Lee<sup>1</sup>, Soo Young Kim<sup>1</sup>, Youngmin Cho<sup>1</sup>, Jin Guk Kim<sup>1</sup>, Jongyeon Choi<sup>1</sup>, Hyun-Sung Shim<sup>1</sup>, Jin Kim<sup>1</sup>, BengSeng Phuah<sup>1</sup>, HyounMin Kim<sup>1</sup>, Myeong Jun Song<sup>1</sup>, Ahn Chai<sup>1</sup>, Daeho Kim<sup>1</sup>, SooYoung Kim<sup>1</sup>, Eun-Bong Kim<sup>1</sup>, David Wang<sup>1</sup>, Shinfaeng Kang<sup>1</sup>, Yulwan Ro<sup>1</sup>, Seungwoo Seo<sup>1</sup>, JoonHo Song<sup>1</sup>, Jaeyoun Yoon<sup>1</sup>, Kyomin Sohn<sup>1</sup>, Nam Sung Kim<sup>1</sup>

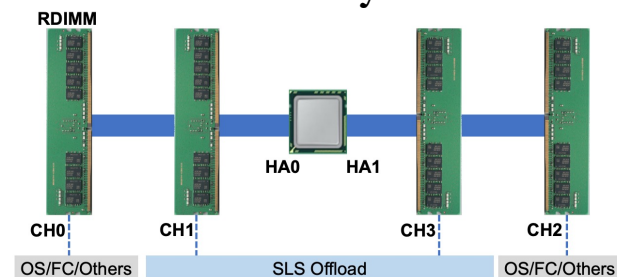
<sup>1</sup>Samsung Electronics, Hwaseong, Korea  
<sup>2</sup>Samsung Electronics, San Jose, CA  
<sup>3</sup>Samsung Electronics, Suwon, Korea

# Samsung AxDIMM (2021)

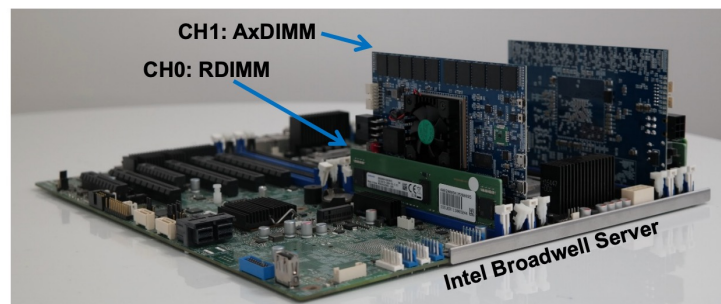
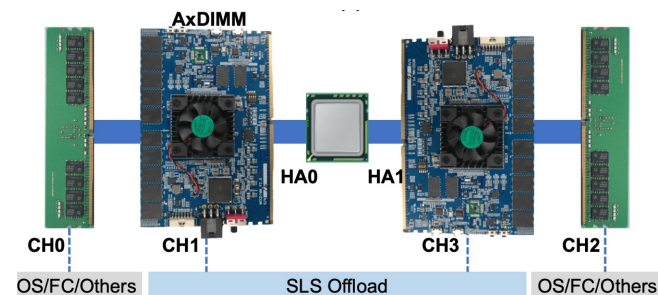
- DDRx-PIM
  - DLRM recommendation system



Baseline System



AxDIMM System





# 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](http://www.skhynix.com)) announced on February 16 that it has developed PIM\*, a next-generation memory chip with computing capabilities.

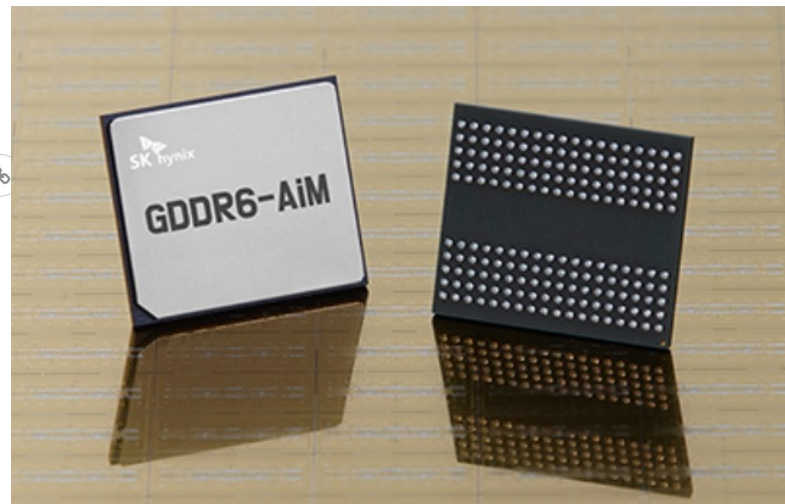
*\*PIM(Processing In Memory): A next-generation technology that provides a solution for data congestion issues for AI and big data by adding computational functions to semiconductor memory*

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 1nm 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 1nm, 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.

# SK Hynix Accelerator-in-Memory (2022)

**System Architecture and Software Stack for GDDR6-AiM**

Yongkee Kwon and Chanwook Park  
SK hynix inc.

5:42 / 6:27:38

SK hynix

ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads



Onur Mutlu Lectures  
32.1K subscribers

Analytics

Edit video

33



Share

Download

Clip

Save



1,146 views Streamed live on Mar 26, 2023 Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)

ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

<https://events.safari.ethz.ch/asplos-...>

<https://www.youtube.com/watch?v=oYCaLcT0Kmo>

# AliBaba PIM Recommendation System (2022)

ISSCC 2022 / February 24, 2022 / 8:30 AM

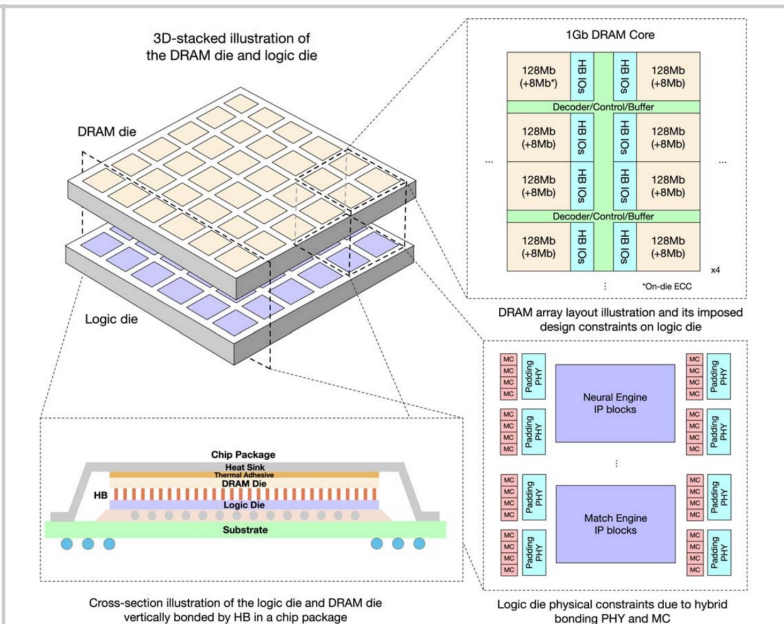


Figure 29.1.2: Illustration of 3D-stacked chip, cross-illustration of package, DRAM array layout and design blocks on logic die.

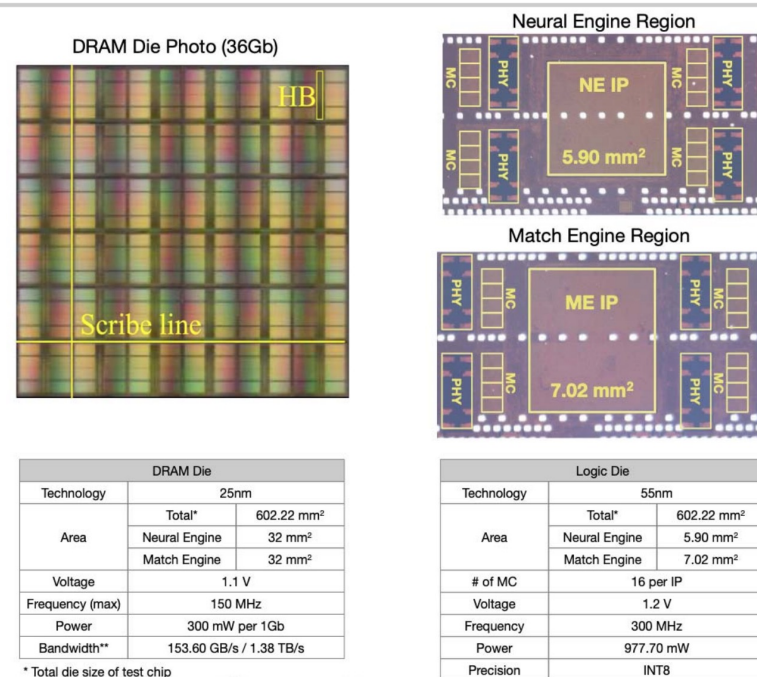


Figure 29.1.7: Die micrographs of DRAM die, NE and ME. Detailed specifications of DRAM die and logic die.

## 29.1 184QPS/W 64Mb/mm<sup>2</sup> 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System

Dimin Niu<sup>1</sup>, Shuangchen Li<sup>1</sup>, Yuhao Wang<sup>1</sup>, Wei Han<sup>1</sup>, Zhe Zhang<sup>2</sup>, Yijin Guan<sup>2</sup>, Tianchan Guan<sup>3</sup>, Fei Sun<sup>1</sup>, Fei Xue<sup>1</sup>, Lide Duan<sup>1</sup>, Yuanwei Fang<sup>1</sup>, Hongzhong Zheng<sup>1</sup>, Xiping Jiang<sup>4</sup>, Song Wang<sup>4</sup>, Fengguo Zuo<sup>4</sup>, Yubing Wang<sup>4</sup>, Bing Yu<sup>4</sup>, Qiwei Ren<sup>4</sup>, Yuan Xie<sup>1</sup>



# SK Hynix CXL Processing Near Memory (2023)

IEEE COMPUTER ARCHITECTURE LETTERS, VOL. 22, NO. 1, JANUARY-JUNE

## Computational CXL-Memory Solution for Accelerating Memory-Intensive Applications

Joonseop Sim<sup>ID</sup>, Soohong Ahn<sup>ID</sup>, Taeyoung Ahn<sup>ID</sup>,  
Seungyong Lee<sup>ID</sup>, Myunghyun Rhee, Jooyoung Kim<sup>ID</sup>,  
Kwangsik Shin, Donguk Moon<sup>ID</sup>,  
Euseok Kim, and Kyoung Park<sup>ID</sup>

**Abstract**—CXL interface is the up-to-date technology that enables effective memory expansion by providing a memory-sharing protocol in configuring heterogeneous devices. However, its limited physical bandwidth can be a significant bottleneck for emerging data-intensive applications. In this work, we propose a novel CXL-based memory disaggregation architecture with a real-world prototype demonstration, which overcomes the bandwidth limitation of the CXL interface using near-data processing. The experimental results demonstrate that our design achieves up to  $1.9\times$  better performance/power efficiency than the existing CPU system.

**Index Terms**—Compute express link (CXL), near-data-processing (NDP)

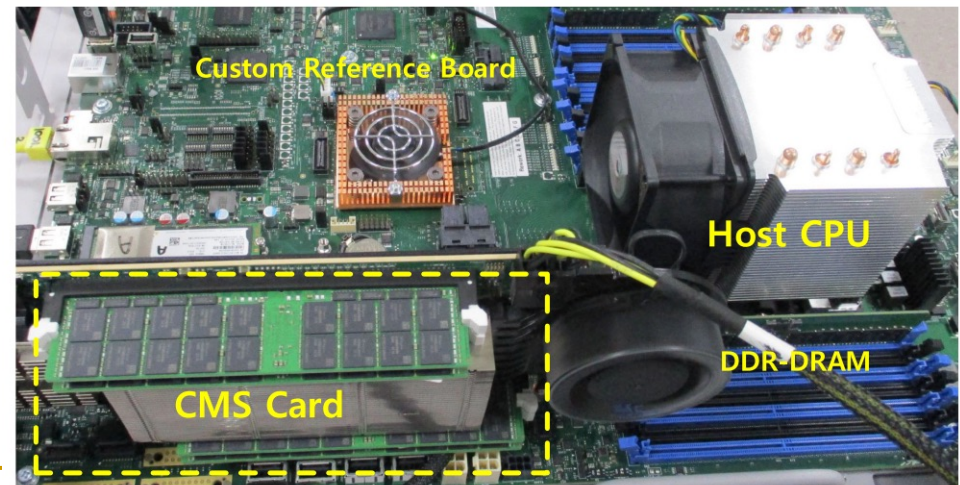


Fig. 6. FPGA prototype of proposed CMS card.

# Samsung CXL Processing Near Memory (2023)

## Samsung Processing in Memory Technology at Hot Chips 2023

By Patrick Kennedy - August 28, 2023



Samsung PIM PNM For Transformer Based AI HC35\_Page\_24



# Concluding Remarks

## Fundamentally Energy-Efficient **(Data-Centric)** Computing Architectures

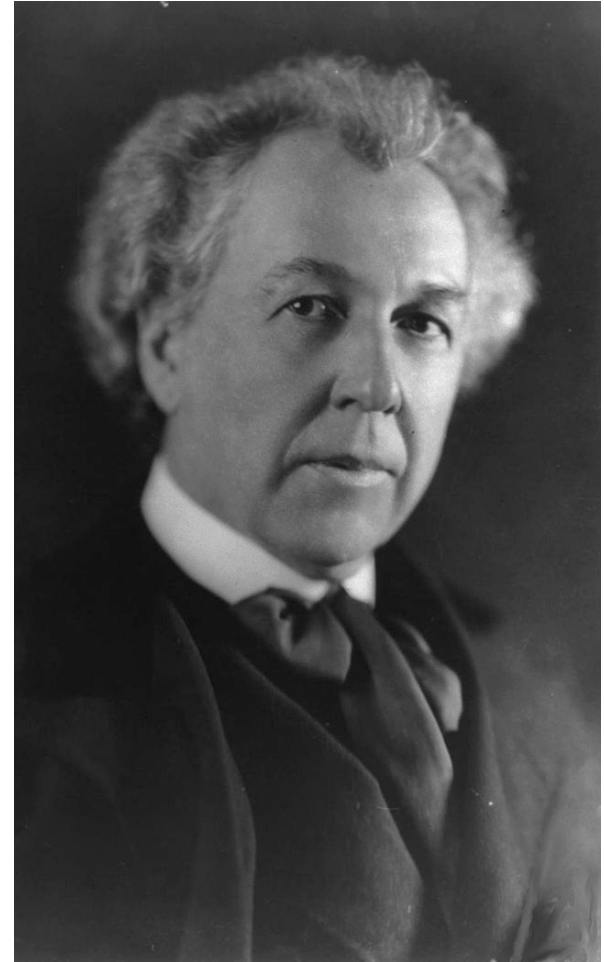
# Fundamentally High-Performance **(Data-Centric)** Computing Architectures

# Computing Architectures with Minimal Data Movement

# A Quote from A Famous Architect

---

- “architecture [...] based upon **principle**, and not upon **precedent**”



# Precedent-Based Design?

---

- “architecture [...] based upon **principle**, and not upon **precedent**”





# Principled Design

---

- “architecture [...] based upon **principle**, and not upon **precedent**”







# The Overarching Principle

---

## Organic architecture

---

From Wikipedia, the free encyclopedia

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

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



# Another Example: Precedent-Based Design

---





# Principled Design





# Another Principled Design



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

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

# Another Principled Design

---





# Principle Applied to Another Structure



Source: By 準建築人手札網站 Forgemind ArchiMedia - Flickr: IMG\_2489.JPG, CC BY 2.0

Source: <https://www.dezeen.com/2016/08/29/santiago-calatrava-oculus-world-trade-center-transportation-hub-new-york-photographs-hufton-crow/>  
<https://commons.wikimedia.org/wiki/index.php?curid=91498396>, [https://en.wikipedia.org/wiki/Santiago\\_Calatrava](https://en.wikipedia.org/wiki/Santiago_Calatrava)



# The Overarching Principle

---

## Zoomorphic architecture

---

From Wikipedia, the free encyclopedia

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

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

# Overarching Principles for Computing?

---



# Concluding Remarks

---

- It is time to design **principled system architectures** to solve the **memory problem**
- We must design systems to be **balanced, high-performance, and energy-efficient** → **memory-centric**
  - Enable computation capabilities in memory
- This can
  - Lead to **orders-of-magnitude** improvements
  - **Enable new applications & computing platforms**
  - **Enable better understanding of nature**
  - ...
- Future of **truly memory-centric computing** is bright
  - We need to do research & design across the computing stack

**Data-centric**

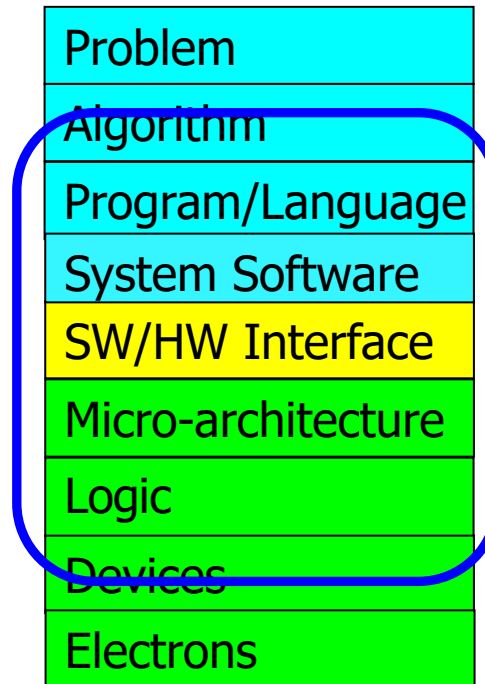
**Data-driven**

**Data-aware**

# We Need to Revisit the Entire Stack

---

- With a **memory-centric mindset**



**We can get there step by step**

# PIM Review and Open Problems

---

## A Modern Primer on Processing in Memory

Onur Mutlu<sup>a,b</sup>, Saugata Ghose<sup>b,c</sup>, Juan Gómez-Luna<sup>a</sup>, Rachata Ausavarungnirun<sup>d</sup>

*SAFARI Research Group*

<sup>a</sup>*ETH Zürich*

<sup>b</sup>*Carnegie Mellon University*

<sup>c</sup>*University of Illinois at Urbana-Champaign*

<sup>d</sup>*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.*



# Open Source Tools: SAFARI GitHub



## SAFARI Research Group at ETH Zurich and Carnegie Mellon University

Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

👤 440 followers

📍 ETH Zurich and Carnegie Mellon U...

🔗 <https://safari.ethz.ch/>

✉ [omutlu@gmail.com](mailto:omutlu@gmail.com)

🏠 Overview

📁 Repositories 98

📁 Projects

📦 Packages

👤 People 13

### 📁 ramulator Public

A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDRx, LPDDRx, GDDRx, WIOx, HBMx, and various academic proposals. Described in the...

● C++ ☆ 532 🍴 206

### 📁 prim-benchmarks Public

PrIM (Processing-In-Memory benchmarks) is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publ...

● C ☆ 126 🍴 47

### 📁 MQSim Public

MQSim is a fast and accurate simulator modeling the performance of modern multi-queue (MQ) SSDs as well as traditional SATA based SSDs. MQSim faithfully models new high-bandwidth protocol implement...

● C++ ☆ 268 🍴 143

### 📁 rowhammer Public

Source code for testing the Row Hammer error mechanism in DRAM devices. Described in the ISCA 2014 paper by Kim et al. at [http://users.ece.cmu.edu/~omutlu/pub/dram-row-hammer\\_isca14.pdf](http://users.ece.cmu.edu/~omutlu/pub/dram-row-hammer_isca14.pdf).

● C ☆ 211 🍴 42

### 📁 SoftMC Public

SoftMC is an experimental FPGA-based memory controller design that can be used to develop tests for DDR3 SODIMMs using a C++ based API. The design, the interface, and its capabilities and limitatio...

● Verilog ☆ 120 🍴 27

### 📁 Pythia Public

A customizable hardware prefetching framework using online reinforcement learning as described in the MICRO 2021 paper by Bera et al. (<https://arxiv.org/pdf/2109.12021.pdf>).

● C++ ☆ 109 🍴 34

<https://github.com/CMU-SAFARI/>

# Referenced Papers, Talks, Artifacts

---

- All are available at

<https://people.inf.ethz.ch/omutlu/projects.htm>

<https://www.youtube.com/onurmutlulectures>

<https://github.com/CMU-SAFARI/>

# Funding Acknowledgments

---

- Alibaba, AMD, ASML, Google, Facebook, Hi-Silicon, HP Labs, Huawei, IBM, Intel, Microsoft, Nvidia, Oracle, Qualcomm, Rambus, Samsung, Seagate, VMware, Xilinx
- NSF
- NIH
- GSRC
- SRC
- CyLab
- EFCL
- SNSF

Thank you!

# Memory Systems and Memory-Centric Computing

## Topic 2: Memory-Centric Computing

Onur Mutlu

[omutlu@gmail.com](mailto:omutlu@gmail.com)

<https://people.inf.ethz.ch/omutlu>

16-17 July 2024

HiPEAC ACACES Summer School 2024

**SAFARI**

**ETH** zürich