P&S Processing-in-Memory
Real-World Processing-in-Memory Architectures: Alibaba Hybrid Bonding PNM Engine

Dr. Juan Gómez Luna
Prof. Onur Mutlu
ETH Zürich
Spring 2022
12 May 2022
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

---

UPMEM PIM System Organization

• A UPMEM DIMM contains 8 or 16 chips
  - Thus, 1 or 2 ranks of 8 chips each

• Inside each PIM chip there are:
  - 8 64MB banks per chip: Main RAM (MRAM) banks
  - 8 DRAM Processing Units (DPUs) in each chip, 64 DPUs per rank
FIMDRAM: Chip Structure

- FIMDRAM based on HBM2

**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)

[3D Chip Structure of HBM with FIMDRAM]
FIMDRAM: System Organization (III)

- PIM units respond to standard DRAM column commands (RD or WR)
  - Compliant with unmodified JEDEC controllers
- They execute one wide-SIMD operation commanded by a PIM instruction with deterministic latency in a lock-step manner
- A PIM unit can get 16 16-bit operands from IOSAs, a register, and/or the result bus
AiM: Chip Implementation

- 4 Gb AiM die with 16 processing units (PUs)

**AiM Die Photograph**

**1 Process Unit (PU) Area**

<table>
<thead>
<tr>
<th>Component</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>0.19mm²</td>
</tr>
<tr>
<td>MAC</td>
<td>0.11mm²</td>
</tr>
<tr>
<td>Activation Function (AF)</td>
<td>0.02mm²</td>
</tr>
<tr>
<td>Reservoir Cap.</td>
<td>0.05mm²</td>
</tr>
<tr>
<td>Etc.</td>
<td>0.01mm²</td>
</tr>
</tbody>
</table>

- Reservoir Cap. 26%
- MAC 58%
- AF 11%
- Etc. 5%
AiM: System Organization

- GDDR6-based AiM architecture

Lee et al., A 1ynm 1.25V 8Gb, 16Gb/s/pin GDDR6-based Accelerator-in-Memory supporting 1TFLOPS MAC Operation and Various Activation Functions for Deep-Learning Applications, ISSCC 2022
Samsung AxDIMM (2021)

- DIMM-based PIM
  - DLRM recommendation system

AxDIMM Design: Hardware Architecture

Alibaba 3D Logic-to-DRAM
Hybrid Bonding with
Processing-near-Memory Engine
Hybrid Bonding with PnM Engine (ISSCC 2022)

ISSCC 2022 / SESSION 29 / ML CHIPS FOR EMERGING A...

29.1 184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System

Dimin Niu¹, Shuangchen Li¹, Yuhao Wang¹, Wei Han¹, Zhe Zhang², Yijin Guan², Tianchan Guan³, Fei Sun¹, Fei Xue¹, Lide Duan¹, Yuanwei Fang¹, Hongzhong Zheng¹, Xiping Jiang⁴, Song Wang⁴, Fengguo Zuo⁴, Yubing Wang⁴, Bing Yu⁴, Qiwei Ren⁴, Yuan Xie¹

¹Alibaba DAMO Academy, Sunnyvale, CA; ²Alibaba DAMO Academy, Beijing, China; ³Alibaba DAMO Academy, Shanghai, China; ⁴UnilC, Xian, China

https://doi.org/10.1109/ISSCC42614.2022.9731694
Processing-in-Memory for Machine Learning

- Memory bandwidth is not enough for many ML workloads

**Scaling Out Solution**
- More hardware
- More computation time

**Solution Needed**
- Limited by physical limit
- New chip architecture or new memory technology

<table>
<thead>
<tr>
<th>Al model computation requirement</th>
<th>Hardware computation capability</th>
<th>Memory system capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>750x / 2 years</td>
<td>3.1x / 2 years</td>
<td>1.4x / 2 years</td>
</tr>
</tbody>
</table>

- **Natural Language Processing**
- **Recommendation Systems**
- **Graph Neural Network**
- **Multi-Task Online Inference**

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
Processing-in-Memory Classification

Traditional

Memory

Computation

CPU

GPU

FPGA

2D CIM

Memory & Computation

PUM (e.g., SIMDRAI, NVM...)

2D PNM

Memory

Computation

UPMEM

A&M

2.5D PNM

Memory

Computation

Interposer

AxDIMM

PUM: Process Near Memory
CIM: Compute In Memory

3D HB-PNM

Memory

Computation

3D TSV-PNM

Memory

Memory

Memory

Computation

PSM: Process Near Memory
CIM: Compute In Memory

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
5.2. Two Approaches: Processing Using Memory (PUM) vs. Processing Near Memory (PNM)

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

Table 1: Summary of enabling technologies for the two approaches to PIM used by recent works. Adapted from [309].

<table>
<thead>
<tr>
<th>Approach</th>
<th>Enabling Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing Using Memory</td>
<td>SRAM</td>
</tr>
<tr>
<td></td>
<td>DRAM</td>
</tr>
<tr>
<td></td>
<td>Phase-change memory (PCM)</td>
</tr>
<tr>
<td></td>
<td>Magnetic RAM (MRAM)</td>
</tr>
<tr>
<td></td>
<td>Resistive RAM (RRAM)/memristors</td>
</tr>
<tr>
<td>Processing Near Memory</td>
<td>Logic layers in 3D-stacked memory</td>
</tr>
<tr>
<td></td>
<td>Silicon interposers</td>
</tr>
<tr>
<td></td>
<td>Logic in memory controllers</td>
</tr>
</tbody>
</table>

**Processing using memory (PUM)** exploits the existing memory architecture and the operational principles of the memory circuitry to enable operations within main memory with minimal changes. PUM makes use

---


PIM Review and Open Problems

A Modern Primer on Processing in Memory

Onur Mutlu\textsuperscript{a,b}, Saugata Ghose\textsuperscript{b,c}, Juan Gómez-Luna\textsuperscript{a}, Rachata Ausavarungnirun\textsuperscript{d}

SAFARI Research Group

\textsuperscript{a}ETH Zürich
\textsuperscript{b}Carnegie Mellon University
\textsuperscript{c}University of Illinois at Urbana-Champaign
\textsuperscript{d}King Mongkut’s University of Technology North Bangkok

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun, "A Modern Primer on Processing in Memory"

HB-PNM: Overall Architecture (I)

- 3D-stacked logic die and DRAM die vertically bonded by hybrid bonding (HB)
HB-PNM: Overall Architecture (II)

- Match engine and neural engine for matching and ranking in a recommendation system

![Overall Architecture Diagram](image-url)

- Image / Item Query
- Classification
- Object Detection
- Feature Extraction
- Coarse-grained Matching
- Fine-grained Ranking
- Top-K Results

**DRAM Die**
- 1Gb DRAM
- 1Gb DRAM
- 1Gb DRAM
- 1Gb DRAM

**Logic Die**
- Dual-mode Interface
- Match Engine (ME)
- Neural Engine (NE)
- Control
- Dataflow
- Control
- Activation
- Matching
- Top-K
- GEMM
- Transpose

**Typical Recommendation System**
- A two-step recommendation system
- Feature Generation
- Classification, object detection and feature extraction
- Computation-bound
- Typically executed on GPU
- Matching & Ranking
- Coarse-grained matching and fine-grained ranking
- Memory-bound
- Typically executed on CPU and commercial DRAM as external memory
- Consumes most latency (89.87%) and energy (82.97%)
- Requires high-bandwidth, large-capacity and energy-efficient memory
Recommendation Systems
Feature Generation + Matching & Ranking

- Recommendation system
  - Feature generation
    - Classification, object detection, feature extraction
    - Compute-bound
    - Good fit for GPU
  - Matching and ranking
    - Coarse-grained matching, fine-grained ranking
    - Memory-bound
      - Most latency (89.87%) and energy (82.97%)
      - Typically run on CPU

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
Candidate recommendations are retrieved and then ranked.

Covington et al., Deep Neural Networks for YouTube Recommendations, RecSys 2016


Overview of Recommendation Models

- **Personalized recommendation**: recommend content to users, e.g., Facebook’s DLRM recommendation system

---

**Dense features**: continuous inputs in vectors and matrices are processed by typical DNN layers (e.g., fully connected layers)

**Overview of Recommendation Models**

- **Personalized recommendation**: recommend content to users, e.g., Facebook’s DLRM recommendation system

---

**Sparse features**: for categorical inputs; processed by indexing large embedding tables

---

Overview of Recommendation Models

- **Personalized recommendation**: recommend content to users, e.g., Facebook’s DLRM recommendation system

Embedding tables are organized as a set of potentially millions of vectors: lookup and pooling operations represent sparse features learned during training and generally exhibit **Gather-Reduce pattern**, via Caffe2’s **SparseLengths (SLS)** operators.

DLRM Performance Characterization

- Identifying **key performance bottlenecks** for the DLRM system

SparseLengths (SLS) operators:
- **Low FP intensity**
- Larger batch size:
  - Higher memory footprint
  - Higher memory intensity

The memory bandwidth can easily be saturated by embedding operations especially as both the batch size and the number of threads increase.

Feature Generation + Matching & Ranking

- Recommendation system
  - Feature generation
    - Classification, object detection, feature extraction
    - Compute-bound
    - Good fit for GPU
  - Matching and ranking
    - Coarse-grained matching, fine-grained ranking
    - Memory-bound
      - Most latency (89.87%) and energy (82.97%)
      - Typically run on CPU
Matching & Ranking

- **Coarse-grained matching**
  - Binary feature vectors with 512 dimensions
  - Distance calculation
  - Top-1000 items selected from 40K items

- **Fine-grained ranking**
  - Features with 8 bits x 1024 dimensions
  - 3-layer MLP (2048-256-64-1) for similarity prediction
  - Top-100 ranking results from 1K items

---

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
3D Logic-to-DRAM Hybrid Bonding
HB-PNM: Overall Architecture (I)

- 3D-stacked logic die and DRAM die vertically bonded by hybrid bonding (HB)
HB-PNM: Chip Implementation

- 3D Logic-to-DRAM Hybrid Bonding
  - Face-to-face hybrid wafer bonding

![Diagram of 3D Logic-to-DRAM Hybrid Bonding]

**Figure 29.1.1:** Motivations and comparison of state-of-the-art PNM/CIM architectures.

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

**Figure 29.1.3:** Overall architecture of PNM logic. Detailed flow of typical recommendation system.

**Figure 29.1.4:** Detailed design of Match Engine (ME), showing internal data-path micro-architecture of AddGen, distance calculator, and top-K engine.

**Figure 29.1.5:** Detailed design of Neural Engine (NE), showing internal datapath, interface modules, micro architecture of VPU and GEMM, FSM of control modules and lock-step debug module.

**Figure 29.1.6:** Illustration of FPGA-based evaluation platform, comparison with prior near-memory processing designs, and end-to-end performance evaluation of our HB chip and CPU-DRAM system on recommendation application.
HB-PNM: Hybrid-Bonding Interconnection

- 3D Logic-to-DRAM Hybrid Bonding
  - Face-to-face hybrid wafer bonding
    - Logic and memory manufactured independently: this avoids the challenges of integrating logic into memory chips
  - Cu-Cu direct fusion with low bonding temperature (<350ºC)
  - Much denser vias than other 3D-stacking technologies
    - Low pitch size (3 um) vs. HBM microbumps (35 um)
    - High inter-layer bandwidth (1.38 TB/s) vs. HBM2E (460 GB/s)

1 Kim et al. Signal Integrity and Computing Performance Analysis of a Processing-In-Memory of High Bandwidth Memory (PIM-HBM) Scheme, IEEE TCPMT, 2021
2 https://product.skhynix.com/products/dram/hbm/hbm2e.go

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
HB-PNM: DRAM Die and Logic Die

- DRAM die and logic die

<table>
<thead>
<tr>
<th>DRAM Die</th>
<th>Logic Die</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technology</strong></td>
<td><strong>Technology</strong></td>
</tr>
<tr>
<td>25nm</td>
<td>55nm</td>
</tr>
<tr>
<td><strong>Area</strong></td>
<td><strong>Area</strong></td>
</tr>
<tr>
<td>Total* 602.22 mm²</td>
<td>Total* 602.22 mm²</td>
</tr>
<tr>
<td>Neural Engine 32 mm²</td>
<td>Neural Engine 5.90 mm²</td>
</tr>
<tr>
<td>Match Engine 32 mm²</td>
<td>Match Engine 7.02 mm²</td>
</tr>
<tr>
<td><strong>Voltage</strong></td>
<td><strong>Voltage</strong></td>
</tr>
<tr>
<td>1.1 V</td>
<td>1.2 V</td>
</tr>
<tr>
<td><strong>Frequency (max)</strong></td>
<td><strong>Frequency</strong></td>
</tr>
<tr>
<td>150 MHz</td>
<td>300 MHz</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td><strong>Power</strong></td>
</tr>
<tr>
<td>300 mW per 1Gb</td>
<td>977.70 mW</td>
</tr>
<tr>
<td><strong>Bandwidth</strong>**</td>
<td><strong>Precision</strong></td>
</tr>
<tr>
<td>153.60 GB/s / 1.38 TB/s</td>
<td>INT8</td>
</tr>
</tbody>
</table>

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
HB-PNM Architecture
HB-PNM Architecture

- DRAM die composed of 6x6 1Gb DRAM cores
  - 8 banks per core
  - 128-bit I/O per bank
  - On-chip ECC (8 Mb per 128 Mb)

![Diagram of 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022](image)

HB imposes design constraints on location of memory controllers (MC) and PHY
HB-PNM: Logic Die

- **Match engine and neural engine** for matching and ranking in a recommendation system
  - Direct access to their counterpart DRAM blocks
  - Access to other DRAM blocks via on-chip bus
HB-PNM Logic Die: Dual-mode Interface

- **Dual-mode interface** can switch between
  - All 8 banks in lock-step for full bandwidth
  - Single channel (1 of 8 banks)
**Neural Engine: Interface Bridge**

- **Support for single-channel mode and lockstep mode**
- **Read/write counters to support burst requests**

Niu et al., 184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
HB-PNM: Match Engine

- Responsible for coarse-grained matching

Overall Architecture

- Memory
  - 4 x 1 Gb blocks with 4096 bits I/O
  - 38.4 GB/s on-chip bandwidth per block

- Compute
  - Match Engine: Coarse-grained Matching
  - Neural Engine: Fine-grained Ranking
  - Dual-mode Interface

Niu et al., 184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
**Match Engine: Address Generator**

- **AddGen** generates the address of the input query
  - Mode selection for **different access patterns**
  - Configurable via registers
Match Engine: Distance Calculator

- **Distance calculator** obtains similarity between input query and feature vectors
  - It computes *Hamming distance of two 512-bit vectors*
  - Distance is filtered by root of max-heap
Match Engine: Top-K Engine (I)

- **Max-heap hardware block** and data structure with 1000 nodes `<address, distance>` for the 1000 shortest distances
  - New input every two cycles

```
DRAM-0
Memory Controller

I/F Bridge

QSPI

SPI bridge

REGs

AddrGen

ME Ctrl

Distance Calculator

FIFO

Max-heap Top-K Engine

DRAM-1
Memory Controller

I/F Bridge

QSPI

SPI bridge

REGs

AddrGen

ME Ctrl

GPIO

Start

Reset

Odd clk cycle? N

Input? Y

Y

Root.Value = Distance
Root.Metadata = Addr

For all nodes on odd layer

< child nodes?

Y

> left child nodes?

N

Swap
(Current, Left Child)

Current = Left Child

For all nodes on even layer

< child nodes?

Y

Swap
(Current, Right Child)

Current = Right Child
```
Match Engine: Top-K Engine (II)

- **Max-heap hardware block** and data structure with 1000 nodes <address, distance> for the 1000 shortest distances

Distance calculator

```
Addr   Data   Query Data
Bit-wise Hamming Distance
Partial Popcount 0  Partial Popcount 1
Add
```

Output

Root of Max-Heap

Top-K

```
Start
Reset

Odd clk cycle? N
Input?

Y

Y

Root.Value = Distance
Root.Metadata = Addr

For all nodes on odd layer

< child nodes?

Y

< left child nodes?

Y

Swap (Current, Left Child)
Current = Left Child

N

Swap (Current, Right Child)
Current = Right Child
```

If Distance < left child, swap left child;
Else, swap right child
HB-PNM: Neural Engine

- Responsible for similarity prediction for fine-grained ranking

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
Neural Engine: Vector Processing Unit

- **Activations** based on LUTs
  - Support for GeLU and Exp

- **Transpose**
  - Transpose 16x16 matrix with ping-pong array
  - 2D register file array
  - Row-based writes and column-based reads

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
Neural Engine: GEMM

- 32x32 INT8 fully-pipelined systolic array
  - Partial sums accumulated in INT32 accumulator

![Diagram of Neural Engine Architecture](image)
Lecture on Systolic Arrays

Digital Design & Computer Arch. - Lecture 19: VLIW, Systolic Arrays, DAE (ETH Zürich, Spring 2021)

2,724 views • Streamed live on May 7, 2021

Onur Mutlu Lectures
20.1K subscribers

https://youtu.be/UtLy4Yagdys?t=2948
Neural Engine: Finite State Machine

- Five working states and one idle state
  - Each working state is for one instruction
HB-PNM: Key Feature Summary

- Comparison table

<table>
<thead>
<tr>
<th></th>
<th>2D CIM *</th>
<th>UPMEM PIM **</th>
<th>A100 GPU ***</th>
<th>FIMDRAM ****</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Memory</td>
<td>SRAM</td>
<td>DDR4</td>
<td>HBM2</td>
<td>HBM2</td>
<td>LPDDR4</td>
</tr>
<tr>
<td>Technology (Memory/Logic)</td>
<td>16nm</td>
<td>2xn,m / 2xn,m</td>
<td>1y# / 7nm</td>
<td>20nm / 20nm</td>
<td>25nm / 55nm</td>
</tr>
<tr>
<td>Capacity</td>
<td>4.5 Mb</td>
<td>8GB / DIMM</td>
<td>80GB</td>
<td>6GB / cube</td>
<td>4.5GB</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>-</td>
<td>128GB/s / DIMM</td>
<td>1935GB/s</td>
<td>1200GB/s / cube#</td>
<td>38.4GB/s / 1Gb</td>
</tr>
<tr>
<td>Frequency (Logic)</td>
<td>200MHz</td>
<td>500MHz</td>
<td>1410MHz</td>
<td>300MHz</td>
<td>300MHz</td>
</tr>
<tr>
<td>Bandwidth/Capacity (a.u.)</td>
<td>-</td>
<td>16</td>
<td>24.2</td>
<td>200</td>
<td>307</td>
</tr>
</tbody>
</table>

** F. Devaux et al, Hotchip 2019
*** J. Choquette et al, Hotchip 2020
**** Y. C. Kwon et al, ISSCC 2021
Processing-in-Memory Classification

Traditional

2D CIM

2D PNM

2.5D PNM

3D HB-PNM

3D TSV-PNM

PNM: Process Near Memory
CIM: Compute In Memory
HB: Hybrid Bonding
TSV: Through-silicon Via

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
5.2. Two Approaches: Processing Using Memory (PUM) vs. Processing Near Memory (PNM)

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

<table>
<thead>
<tr>
<th>Approach</th>
<th>Enabling Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing Using Memory</td>
<td>SRAM, DRAM, Phase-change memory (PCM), Magnetic RAM (MRAM), Resitive RAM (RRAM)/memristors</td>
</tr>
<tr>
<td>Processing Near Memory</td>
<td>Logic layers in 3D-stacked memory, Silicon interposers, Logic in memory controllers</td>
</tr>
</tbody>
</table>

Processing using memory (PUM) exploits the existing memory architecture and the operational principles of the memory circuitry to enable operations within main memory with minimal changes. PUM makes use


Similarities and Differences among Current PIM Systems

- **Similarities**
  - Current real-world processing-in-memory architectures follow a processing-near-memory approach
  - All based on DRAM memory

- **Differences**
  - Near-bank (UPMEM, FIMDRAM, AiM, HB-PNM) vs. near-chip (AxDIMM)
  - General-purpose (UPMEM) vs. special-function (FIMDRAM, AiM, HB-PNM)
  - FGMT (UPMEM) vs. SIMD (FIMDRAM, AiM, AxDIMM) vs. systolic array (HB-PNM)
  - Natively integer (UPMEM, HB-PNM) vs. floating point (FIMDRAM)
    - FP16 (FIMDRAM) vs. BF16 (AiM) vs. FP32 (AxDIMM)
  - DDR4 (UPMEM, AxDIMM) vs. LPDDR4 (HB-PNM) vs. HBM2 (FIMDRAM) vs. GDDR6 (AiM)
Processing-in-Memory Classification

Traditional

CPU

FPGA

GPU

Computation

Memory

2D CIM

PUM (e.g., SIMDRAm, NVM…)

Memory & Computation

2D PNM

UPMEM

AiM

FIMDRAM

AxDIMM

2.5D PNM

Memory

Interposer

Computation

3D HB-PNM

Memory

Computation

3D TSV-PNM

Memory

Memory

Memory

Computation

PNM: Process Near Memory
CIM: Compute In Memory

HB: Hybrid Bonding
TSV: Through-silicon Via

Niu et al., 184QPS/W 64Mb/mm2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System, ISSCC 2022
processing-using-memory in real dram chips

computedram: in-memory compute using off-the-shelf drams

fei gao
feig@princeton.edu
department of electrical engineering
princeton university

georgios tziantzioulis
georgios.tziantzioulis@princeton.edu
department of electrical engineering
princeton university

david wentzlaff
wenzlaf@princeton.edu
department of electrical engineering
princeton university

SoftMC: Open Source DRAM Infrastructure


- Flexible
- Easy to Use (C++ API)
- Open-source
  
github.com/CMU-SAFARI/SoftMC
RowClone & Bitwise Ops in Real DRAM Chips

Figure 4: Timeline for a single bit of a column in a row copy operation. The data in $R_1$ is loaded to the bit-line, and overwrites $R_2$.

Figure 5: Logical AND in ComputeDRAM. $R_1$ is loaded with constant zero, and $R_2$ and $R_3$ store operands ($0$ and $1$). The result ($0 = 1 \land 0$) is finally set in all three rows.
Bitline is above $V_{dd}/2$ when R2 is activated.
Bitwise AND in ComputeDRAM

T1 very short
Sense amps are not activated

T2 very short
PRE cannot close R1
R3 will appear on the address bus
ACT(R2) will activate R3 and R2
Figure 9: (a) Schematic diagram of our testing framework. (b) Picture of our testbed. (c) Thermal picture when the DRAM is heated to 80 °C.
Experimental Methodology (II)

Table 1: Evaluated DRAM modules

<table>
<thead>
<tr>
<th>Group ID: Vendor_Size_Freq(MHz)</th>
<th>Part Num</th>
<th># Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>SKhynix_2G_1333</td>
<td>HMT325S6BFR8C-H9</td>
<td>6</td>
</tr>
<tr>
<td>SKhynix_4G_1066</td>
<td>HMT451S6MMP8G-C7</td>
<td>2</td>
</tr>
<tr>
<td>SKhynix_4G_1066</td>
<td>HMT451S6MMP8G-C7</td>
<td>2</td>
</tr>
<tr>
<td>SKhynix_4G_1066</td>
<td>HMT451S6MMP8G-C7</td>
<td>4</td>
</tr>
<tr>
<td>SKhynix_4G_1066</td>
<td>HMT451S6MMP8G-C7</td>
<td>2</td>
</tr>
<tr>
<td>Samsung_4G_1066</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Samsung_4G_1066</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Micron_2G_1333</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Micron_2G_1333</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Elpida_2G_1333</td>
<td>EBJ21UE8BDS0-DJ-F</td>
<td>2</td>
</tr>
<tr>
<td>Nanya_4G_1333</td>
<td>NT4GC64B8HG0NS-CG</td>
<td>2</td>
</tr>
<tr>
<td>TimeTec_4G_1333</td>
<td>78AP10NUS2R2-4G</td>
<td>2</td>
</tr>
<tr>
<td>Corsair_4G_1333</td>
<td>CMSA8GX3M2A1333C9</td>
<td>2</td>
</tr>
</tbody>
</table>

32 DDR3 Modules
~256 DRAM Chips
Proof of Concept (I)

- How they test these memory modules:
  - Vary $T_1$ and $T_2$, observe what happens.

**SoftMC Experiment**

1. Select a random subarray
2. Fill subarray with random data
3. Issue ACT-PRE-ACTs with given $T_1$ & $T_2$
4. Read out subarray
5. Find out how many columns in a row support either operation
   - Row-wise success ratio
Each grid represents the success ratio of operations for a specific DDR3 module.
ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs

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

Georgios Tziortziolis
georgios.tziortziolis@princeton.edu
Department of Electrical Engineering
Princeton University

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

PnM and PuM Working Synergistically

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. [Older arXiv version]

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

Maciej Besta1, Raghavendra Kanakagiri2, Grzegorz Kwasniewski1, Rachata Ausavarungnirun3, Jakub Beránek4, Konstantinos Kanellopoulos1, Kacper Janda5, Zur Vonarburg-Shmaria1, Lukas Gianinazzi1, Ioana Stefan1, Juan Gómez Luna1, Marcin Copik1, Lukas Kapp-Schwoerer1, Salvatore Di Girolamo1, Marek Konieczny5, Onur Mutlu1, Torsten Hoefler1

1 ETH Zurich 2 IIT Tirupati 3 King Mongkut’s University of Technology North Bangkok
4 Technical University of Ostrava 5 AGH-UST
Upcoming Lectures

- PUM architectures and prototypes
- Case studies
  - SpMV on UPMEM PIM architecture
  - Neural network accelerators for the edge
  - Hybrid transactional and analytical processing (HTAP) databases
- Enabling the adoption of PIM
P&S Processing-in-Memory

Real-World Processing-in-Memory Architectures: Alibaba Hybrid Bonding PNM Engine

Dr. Juan Gómez Luna
Prof. Onur Mutlu
ETH Zürich
Spring 2022
12 May 2022