P&S Processing-in-Memory

Real-World Processing-in-Memory Architectures: Samsung HBM-PIM Architecture

Dr. Juan Gómez Luna
Prof. Onur Mutlu

ETH Zürich
Spring 2023
6 April 2023
Processing in DRAM Engine

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

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

---

**CPU** (x86, ARM, RV...) ➔ **DDR Data bus**

---


Recall: 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
Samsung HBM-PIM, a.k.a. FIMDRAM
Samsung Develops Industry’s First High Bandwidth Memory with AI Processing Power

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.”
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 Kwon¹, Suk Han Lee¹, Jaehoon Lee¹, Sang-Hyuk Kwon¹, Je Min Ryu¹, Jong-Pil Son¹, Seongil O¹, Hak-Soo Yu¹, Haesuk Lee¹, Soo Young Kim¹, Youngmin Cho¹, Jin Guk Kim¹, Jongyoon Choi¹, Hyun-Sung Shin¹, Jin Kim¹, BengSeng Phuah¹, HyoungMin Kim¹, Myeong Jun Song¹, Ahn Choi¹, Daeho Kim¹, SooYoung Kim¹, Eun-Bong Kim¹, David Wang², Shinhaeng Kang¹, Yuhwan Ro³, Seungwoo Seo³, JoonHo Song³, Jaeyoun Youn³, Kyomin Sohn¹, Nam Sung Kim¹

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

https://doi.org/10.1109/ISSCC42613.2021.9365862
Hardware Architecture and Software Stack for PIM Based on Commercial DRAM Technology

Industrial Product

Sukhan Lee§1, Shin-haeng Kang§1, Jaehoon Lee1, Hyeonsu Kim2, Eojin Lee1, Seungwoo Seo2, Hosang Yoon2, Seungwon Lee2, Kyoungwhan Lim1, Hyunsung Shin1, Jinhyun Kim1, Seongil O1, Anand Iyer3, David Wang3, Kyomin Sohn1 and Nam Sung Kim§1

1Memory Business Division, Samsung Electronics
2Samsung Advanced Institute of Technology, Samsung Electronics
3Device Solutions America, Samsung Electronics

https://doi.org/10.1109/ISCA52012.2021.00013
Aquabolt-XL: Samsung HBM2-PIM with in-memory processing for ML accelerators and beyond


Samsung Electronics

https://doi.org/10.1109/HCS52781.2021.9567191
Background: High Bandwidth Memory (HBM)

- HBM stacks **DRAM layers** and a **buffer layer**
  - The buffer layer contains I/O circuitry, self-test, test/debug
- DRAM layers and buffer layer communicate using **Through Silicon Vias (TSVs)**

- The buffer layer is connected to a host processor via a **silicon interposer**

- 1 HBM2 die comprises 4 pseudo channels (pCHs) each with 4 bank groups
  - An access transfers a 256-bit data block over 4 64-bit bursts over one pCH
FIMDRAM: Exploiting Bank Parallelism

- HBM bandwidth is not enough for many ML workloads
  - BLAS-1 and BLAS-2 are typically memory bound

Kwon et al., A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications, ISSCC 2021
FIMDRAM: Chip Structure

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)
Chip Implementation

- Mixed design methodology to implement FIMDRAM
  - Full-custom + Digital RTL
FIMDRAM: System Organization (I)

- HBM2 vs. FIMDRAM
FIMDRAM: System Organization (II)

- Design goals:
  - 1. Support DRAM and PIM-DRAM mode for versatility
  - 2. Minimize the engineering cost of redesigning DRAM banks and sub-arrays

- Thus, PIM unit at I/O boundary of bank
  - 1 PIM unit for each 2 banks
  - 16 16-bit SIMD floating-point units (FPUs) per PIM unit

---

Lee et al., Hardware Architecture and Software Stack for PIM Based on Commercial DRAM Technology, ISCA 2021
SIMD Processing

- Single instruction operates on multiple data elements
  - In time or in space
- Multiple processing elements (PEs), i.e., execution units

- Time-space duality
  - **Array processor**: Instruction operates on multiple data elements at the same time using different spaces (PEs)
  - **Vector processor**: Instruction operates on multiple data elements in consecutive time steps using the same space (PE)
Array vs. Vector Processors

**Array Processor**
- LD0, LD1, LD2, LD3
- AD0, AD1, AD2, AD3
- MU0, MU1, MU2, MU3
- ST0, ST1, ST2, ST3

**Vector Processor**
- LD, ADD, MUL, ST

**Instruction Stream**
- LD VR ← A[3:0]
- ADD VR ← VR, 1
- MUL VR ← VR, 2
- ST A[3:0] ← VR

**Time**
- LD0
- AD0
- MU0
- ST0
- LD1
- AD1
- MU1
- ST1
- LD2
- AD2
- MU2
- ST2
- LD3
- AD3
- MU3
- ST3

**Space**
- Same op @ same time
- Different ops @ same space
- Same op @ space
- Different ops @ time
Lecture on SIMD Processing and GPUs

Tensor Core Microarchitecture (Volta)

- Each warp utilizes two tensor cores
- Each tensor core contains two “octets”
  - 16 SIMD units per tensor core (8 per octet)
  - 4x4 matrix-multiply and accumulate each cycle per tensor core

Unlike conventional SIMD, register contents are not private to each thread, but shared inside the warp.

HetSys Course: Lecture 2: SIMD Processing and GPUs (Spring 2022)

https://youtu.be/hEqk6UMQT0U
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

Lee et al., Hardware Architecture and Software Stack for PIM Based on Commercial DRAM Technology, ISCA 2021
FIMDRAM: Bank-level Parallelism

- Unlike standard DRAM devices, **all banks can be accessed concurrently for 8x higher bandwidth** (with 16 banks/pCH)
- In **AB-PIM mode**, a memory command triggers a PIM instruction in the CRF
FIMDRAM: Programmable Computing Unit (I)

- Control: Instruction sequence manager
- Pipeline of 5 stages
  - 1. Fetch/decode
  - 2. Load 256-bit data from even or odd bank (optional)
  - 3. MUL
  - 4. ADD
  - 5. Writeback to GRF
FIMDRAM: Programmable Computing Unit (II)

- Interface unit to control data flow
- Execution unit
- Register group
  - CRF (command): Instruction buffer
  - GRF (general): Weights and accumulation
  - SRF (source): Constants for MAC
### FIMDRAM: Instruction Set Architecture (I)

- 9 RISC-style 32-bit instructions

<table>
<thead>
<tr>
<th>Type</th>
<th>CMD</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating Point</td>
<td>ADD</td>
<td>FP16 addition</td>
</tr>
<tr>
<td></td>
<td>MUL</td>
<td>FP16 multiplication</td>
</tr>
<tr>
<td></td>
<td>MAC</td>
<td>FP16 multiply-accumulate</td>
</tr>
<tr>
<td></td>
<td>MAD</td>
<td>FP16 multiply and add</td>
</tr>
<tr>
<td>Data Path</td>
<td>MOVE</td>
<td>Load or store data</td>
</tr>
<tr>
<td></td>
<td>FILL</td>
<td>Copy data from bank to GRFs</td>
</tr>
<tr>
<td>Control Path</td>
<td>NOP</td>
<td>Do nothing</td>
</tr>
<tr>
<td></td>
<td>JUMP</td>
<td>Jump instruction</td>
</tr>
<tr>
<td></td>
<td>EXIT</td>
<td>Exit instruction</td>
</tr>
</tbody>
</table>
Operation sequence for matrix vector computing

- Input and output data are accessible to the host in conventional DRAM operation
### FIMDRAM: Data Flow

- Data flow controlled by operation mode and bit RA13

<table>
<thead>
<tr>
<th>Index</th>
<th>Mode</th>
<th>CMD</th>
<th>RA13</th>
<th>Data Movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>WR</td>
<td>L</td>
<td>Data write to cell array</td>
</tr>
<tr>
<td>2</td>
<td>Normal</td>
<td>WR</td>
<td>H</td>
<td>Not available</td>
</tr>
<tr>
<td>3</td>
<td>Normal</td>
<td>RD</td>
<td>L</td>
<td>Data read to cell array</td>
</tr>
<tr>
<td>4</td>
<td>Normal</td>
<td>RD</td>
<td>H</td>
<td>Not available</td>
</tr>
<tr>
<td>5</td>
<td>FIM</td>
<td>WR</td>
<td>L</td>
<td>Data movement from PCU block to cell array</td>
</tr>
<tr>
<td>6</td>
<td>FIM</td>
<td>WR</td>
<td>H</td>
<td>Data write to PCU register</td>
</tr>
<tr>
<td>7</td>
<td>FIM</td>
<td>RD</td>
<td>L</td>
<td>Data movement from cell array to PCU block</td>
</tr>
<tr>
<td>8</td>
<td>FIM</td>
<td>RD</td>
<td>H</td>
<td>Not available</td>
</tr>
</tbody>
</table>

![Diagram showing data flow](image-url)

Kwon et al., A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications, ISSCC 2021
## FIMDRAM: Key Feature Summary

- Comparison table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of DRAM</strong></td>
<td>HBM2</td>
<td>LPDDR4</td>
<td>DDR4</td>
<td>HBM2</td>
</tr>
<tr>
<td><strong>Process</strong></td>
<td>20 nm</td>
<td>20 nm</td>
<td>2x nm</td>
<td>20 nm</td>
</tr>
<tr>
<td><strong>Memory density</strong></td>
<td>8GB/cube (Buffer-die + 8H 8Gb core-die)</td>
<td>8GB/chip (8H 8Gb mono die)</td>
<td>8GB/DIMM</td>
<td>6GB/cube (Buffer-die + 4H 4Gb core-die with PCU + 4H 8Gb core-die)</td>
</tr>
<tr>
<td><strong>Data rate</strong></td>
<td>2.4Gbps</td>
<td>3.2Gbps</td>
<td>2.4Gbps</td>
<td>2.4Gbps</td>
</tr>
<tr>
<td><strong>Bandwidth</strong></td>
<td>307GB/s per cube</td>
<td>25.6GB/s per chip</td>
<td>19.2GB/s per DIMM</td>
<td>307GB/s per cube</td>
</tr>
<tr>
<td><strong># of CH</strong></td>
<td>8 per cube</td>
<td>1 per chip</td>
<td>16 per DIMM</td>
<td>8 per cube</td>
</tr>
<tr>
<td><strong># of processing unit</strong></td>
<td>No</td>
<td>2048 per chip (256 per die)</td>
<td>128 per DIMM (8 per core-die)</td>
<td>128 per cube (32 per core-die)</td>
</tr>
<tr>
<td><strong>Processing operation speed</strong></td>
<td>-</td>
<td>250Mhz @simulation</td>
<td>500Mhz @ Measurement</td>
<td>300Mhz @ Measurement</td>
</tr>
<tr>
<td><strong>Peak throughput</strong></td>
<td>-</td>
<td>0.5 TOPS per chip (250MHz x 256 x 8byte)</td>
<td>0.5 TOPS per DIMM (500MHz x 128 x 8byte)</td>
<td>1.2 TFLOPS per cube (300MHz x 128 x 32byte)</td>
</tr>
<tr>
<td><strong>Operation Precision</strong></td>
<td>-</td>
<td>IN T8</td>
<td>IN T8</td>
<td>FP16</td>
</tr>
</tbody>
</table>

TFLOPS: Tera Floating Point Operations Per Second

FIMDRAM: Instruction Ordering

- One challenge is that DRAM commands may be re-ordered, and using fences is costly performance-wise
- Solution: Address Aligned Mode (AAM)
  - 8 MAC operations with 2 PIM instructions
Another Longer Lecture on HBM-PIM

FIMDRAM: Programmable Computing Unit (FIM)

- Control: Instruction sequence manager
- Pipeline of 5 stages
  - 1. Fetch/decode
  - 2. Load 256-bit data from even or odd bank (optional)
  - 3. MUL
  - 4. ADD
  - 5. Writeback to GRF

Livestream - P&S Exploring the Processing-in-Memory Paradigm for Future Computing Systems (Fall 2021)

Processing in Memory Course: Meeting 4: Real-world PIM architectures III - Fall'21

https://youtu.be/VLyYqI6Cjvc
AMD GPU with HBM-PIM

- AMD Instinct Mi100 + HBM-PIM

Upcoming Lectures

- More real-world PIM architectures
- Programming PIM systems
- Workload characterization for PIM suitability
P&S Processing-in-Memory

Real-World Processing-in-Memory Architectures:
Samsung HBM-PIM Architecture

Dr. Juan Gómez Luna
Prof. Onur Mutlu
ETH Zürich
Spring 2023
6 April 2023