北京大学
PEKING UNIVERSITY

智能硬件体系结构

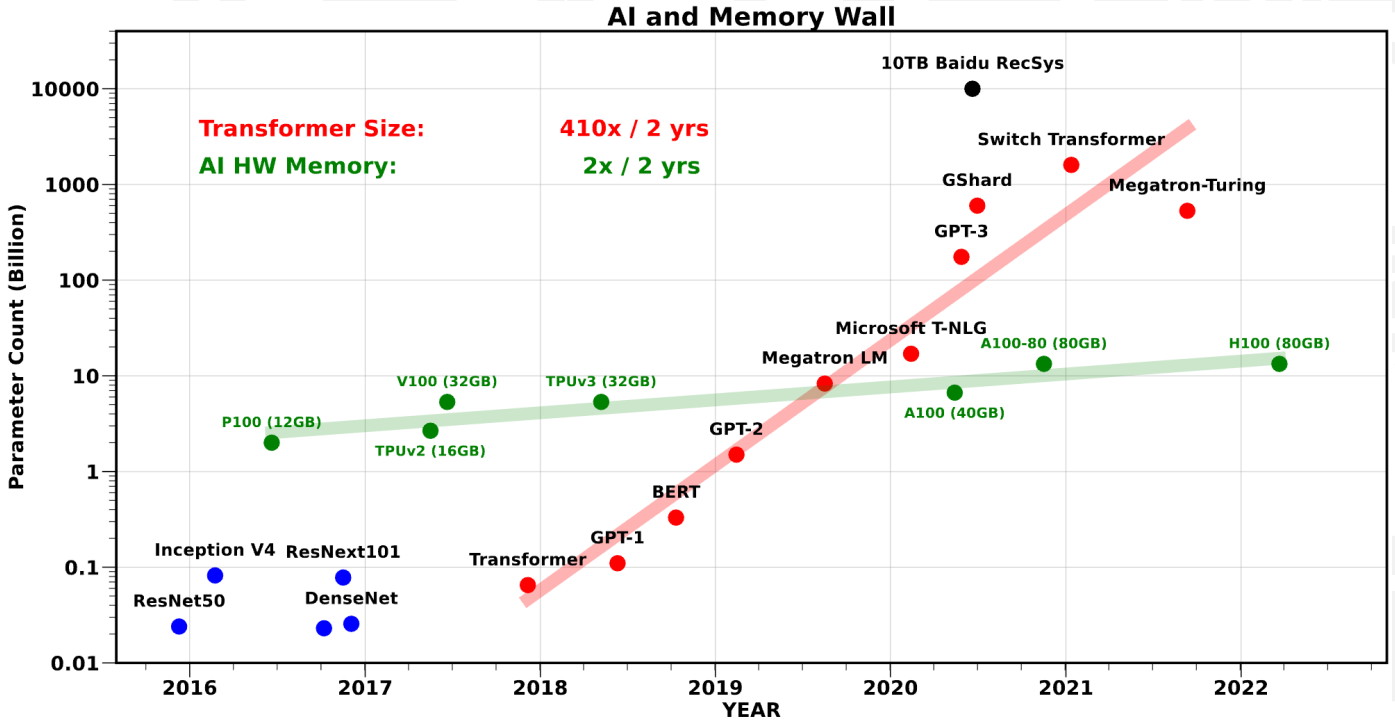
第十三讲：存算一体简介

主讲：陶耀宇、李萌

2025年秋季

人工智能模型的发展与挑战

- 随着人工智能的快速发展，神经网络模型**规模指数级增长**，对人工智能处理器的计算、存储和传输带宽需求也显著提升

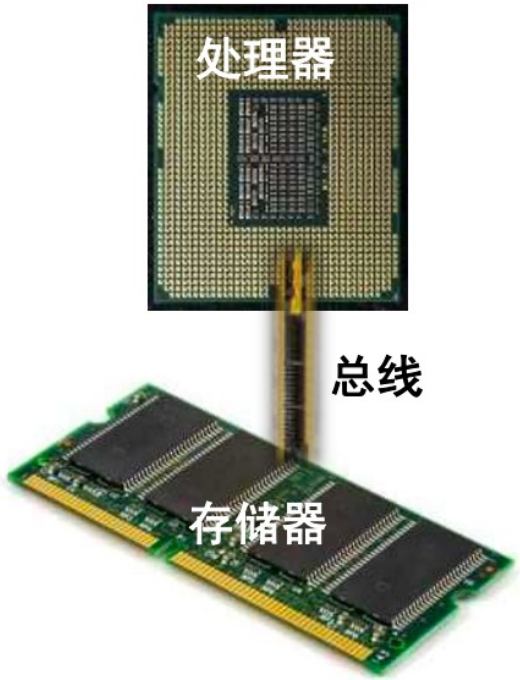


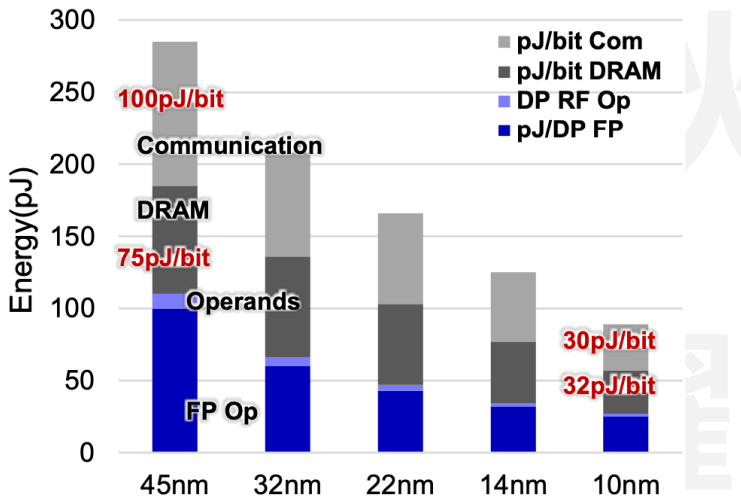
时间	模型	模型参数量 (GB)	模型算力需求 (TFLOPs-day)
2012	AlexNet	10^{-2}	10^{-3}
2015	ResNets	10^{-1}	10^{-1}
2017	Transformer	1	10^{1}
2020	GPT-3	10^{2}	10^{6}
2023	ChatGPT	10^{3}	10^{7}

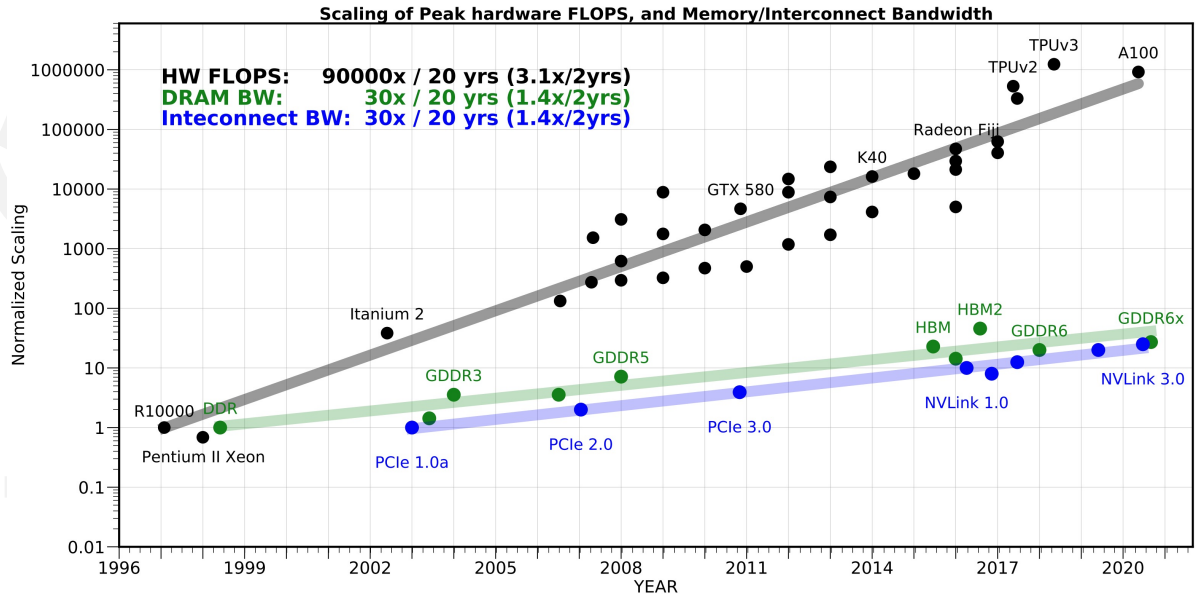
https://github.com/amirgholami/ai_and_memory_wall

人工智能模型的发展与挑战

- 然而，现有的冯诺依曼架构采用存储和计算分离的架构，面临“存储墙”和“功耗墙”瓶颈
  - 内存带宽/互连带宽增长与算力增长速度不一致（30× vs 90,000×），特别是对于先进节点
- 如何克服层“存储墙”瓶颈，进一步提升AI处理器算力和能效成为重要问题



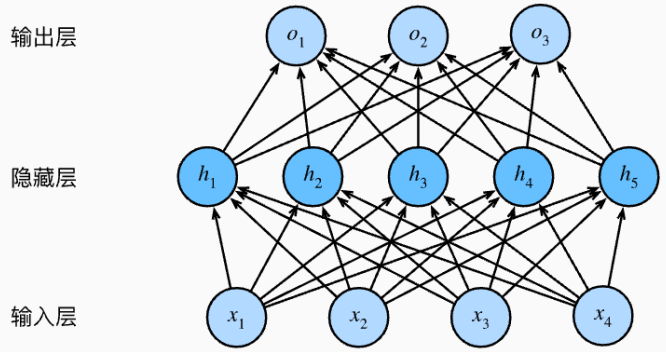




https://github.com/amirgholami/ai_and_memory_wall

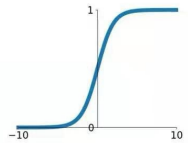
人工智能模型的发展与挑战

- 回顾：什么样的算子更容易受到“存储墙”影响？



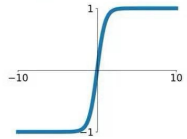
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



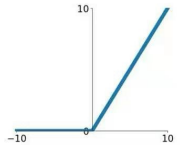
tanh

$\tanh(x)$



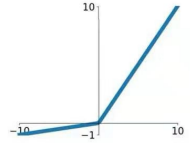
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

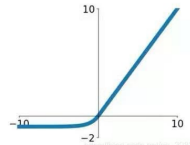


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



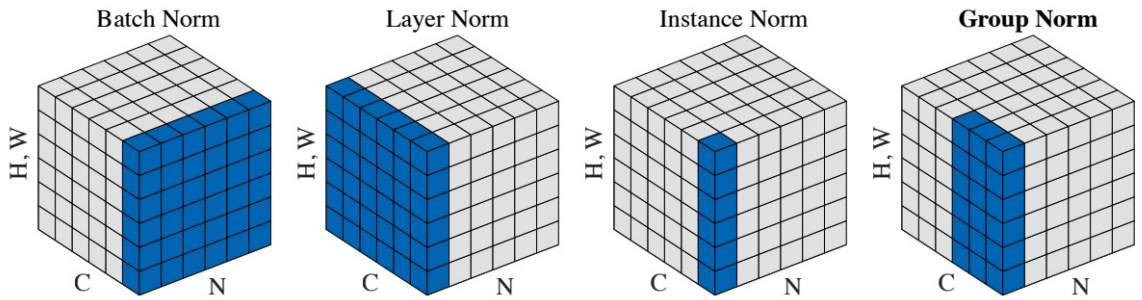
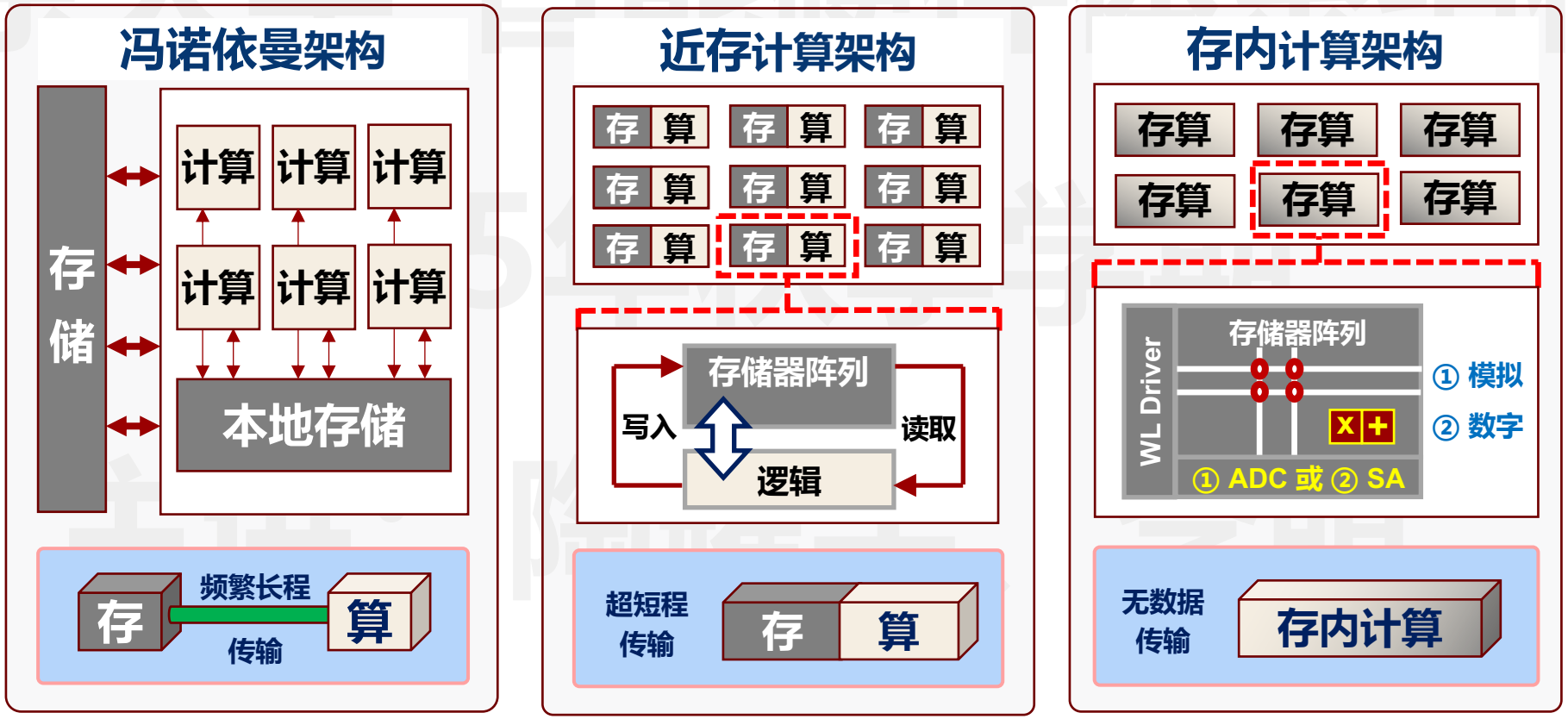


Figure 2. **Normalization methods**. Each subplot shows a feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed by aggregating the values of these pixels.

存算一体新架构

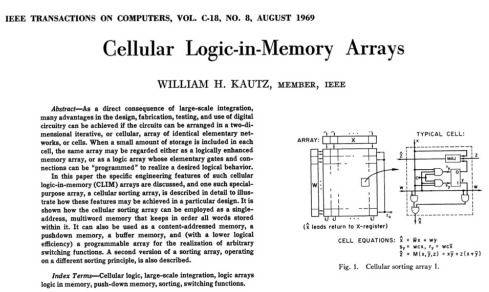
- 针对于访存瓶颈的算子，**存算一体技术**成为打破算力瓶颈的重要途径
- **近存计算**（NDP/PNM）：存储阵列一般无需改动，只提供数据存储功能，计算模块安放在阵列附近
- **存内计算**（PIM）：存储器件可以参与计算操作，这通常意味着存储阵列需要改动来支持计算



DRAM近存计算发展

- 思考：“存储墙”主要在哪里?
- 从学术研究（1969-2016）到商业化探索（2016至今），DRAM近存计算得意于工艺能力的进步和AI应用的驱动

• 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.

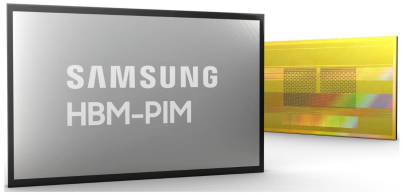
Index Terms—Cellular logic, large-scale integration, logic arrays logic in memory, push-down memory, sorting, switching functions.

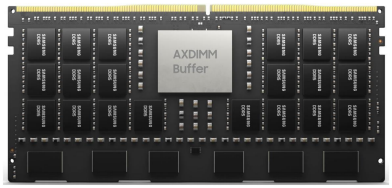
Fig. 1. Cellular sorting array I.

UPMEM PIM-DIMM

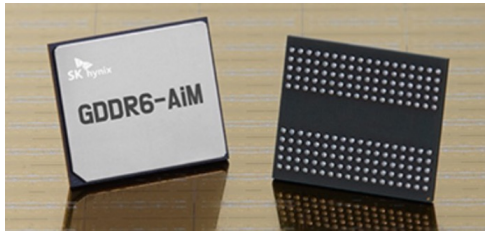
DDR4 2400 R-DIMM module

三星PIM-HBM，AxDIMM

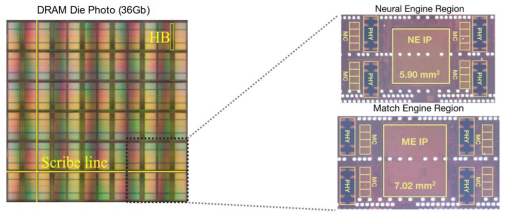




海力士GDDR6-PIM



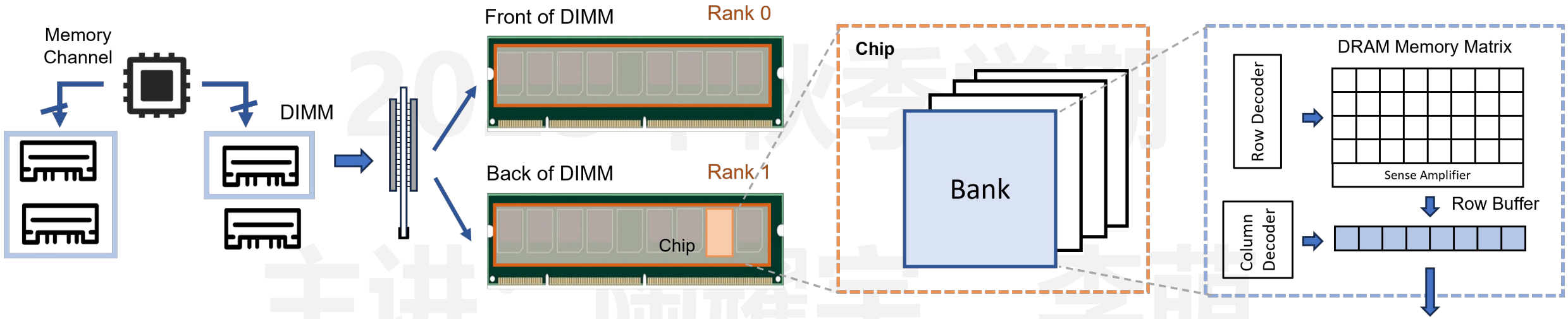
阿里 HB-PNM



1969 2016 2019 2021 2022

DRAM内存架构简介

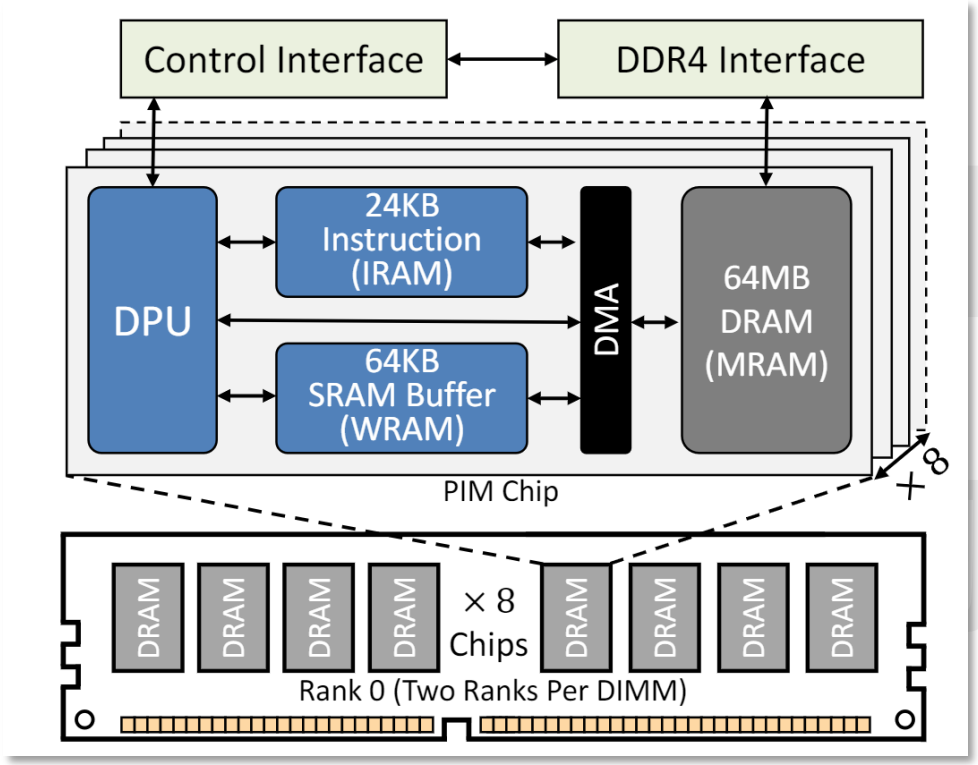
- DRAM的内存架构和组织形式
  - DRAM channel, DIMM, rank, chip, bank, column/row decoder
- 内部带宽 vs 外部带宽

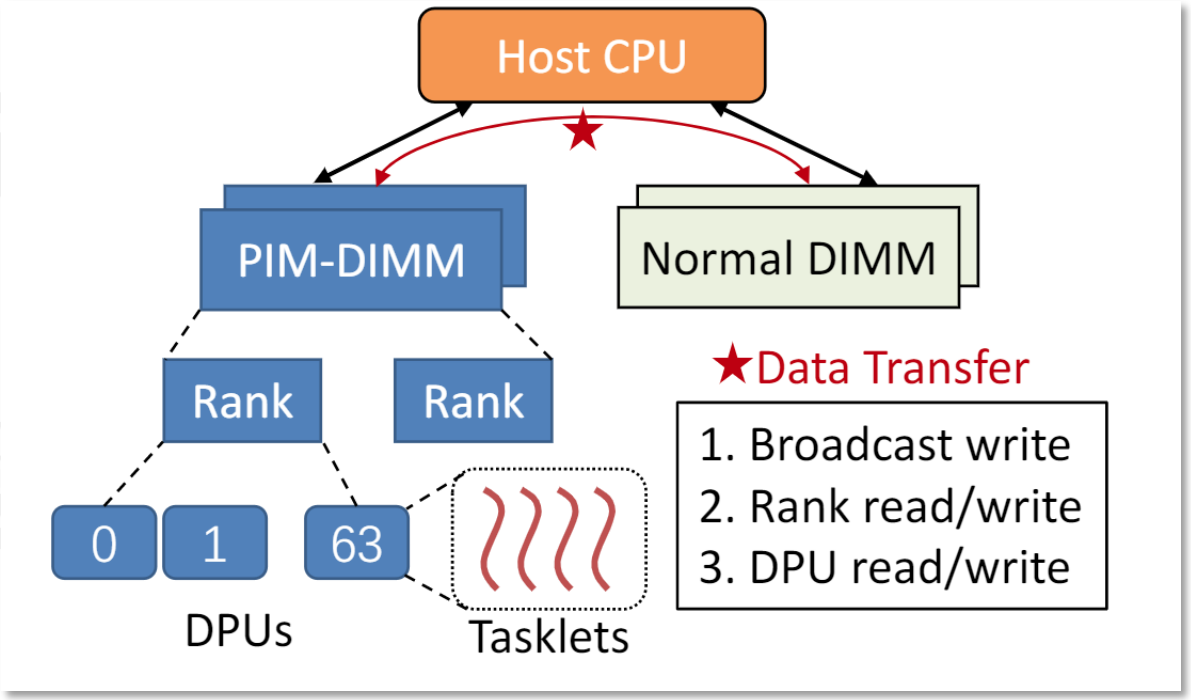


Reference: Prof. Onur Mutlu's computer architecture course

UPMEM PIM-DIMM

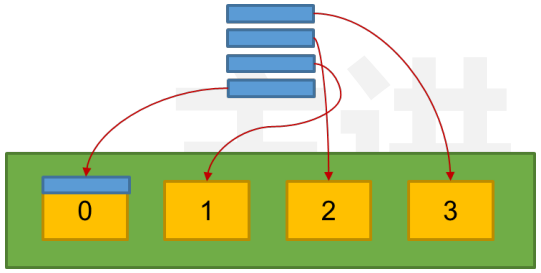
- UPMEM PIM-DIMM架构
  - 单条8GB，128个计算核心（DPU），64KB便签存储器
  - 内部带宽约为100 GB/s（DDR4带宽<20 GB/s）



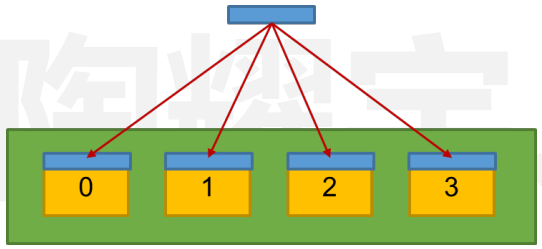


UPMEM PIM-DIMM

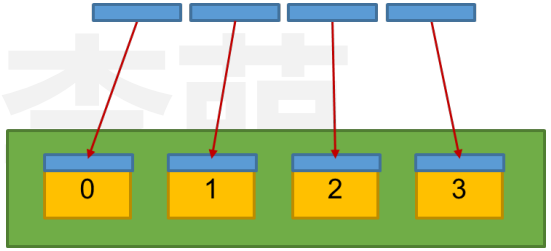
- UPMEM PIM-DIMM系统集成
  - 为PIM-DIMM分配了独立的物理地址空间
  - 数据传入包括serial、parallel和广播模式
- 资源分配/调度：
  - Tasklet (线程)：一个DPU 可以运行24个线程，有各自的stack
  - DPU：64MB Main RAM，DPU之间无法直接进行数据传递
  - Rank: 64个DPU，可以进行并行CPU-DPU数据传输



CPU-DPU serial



CPU-DPU parallel



CPU-DPU broadcast

UPMEM PIM-DIMM

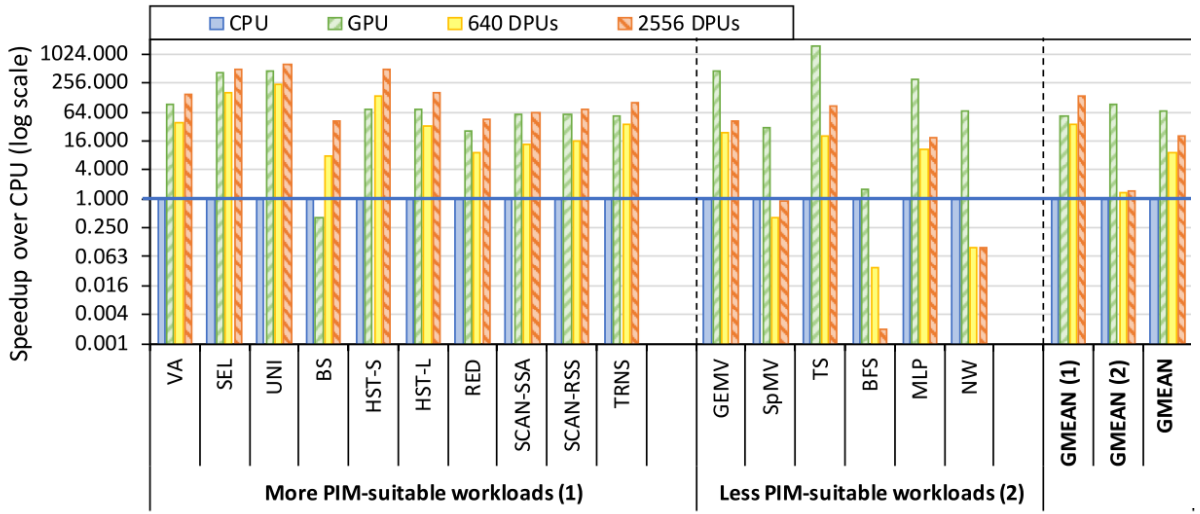
- **性能测试**

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

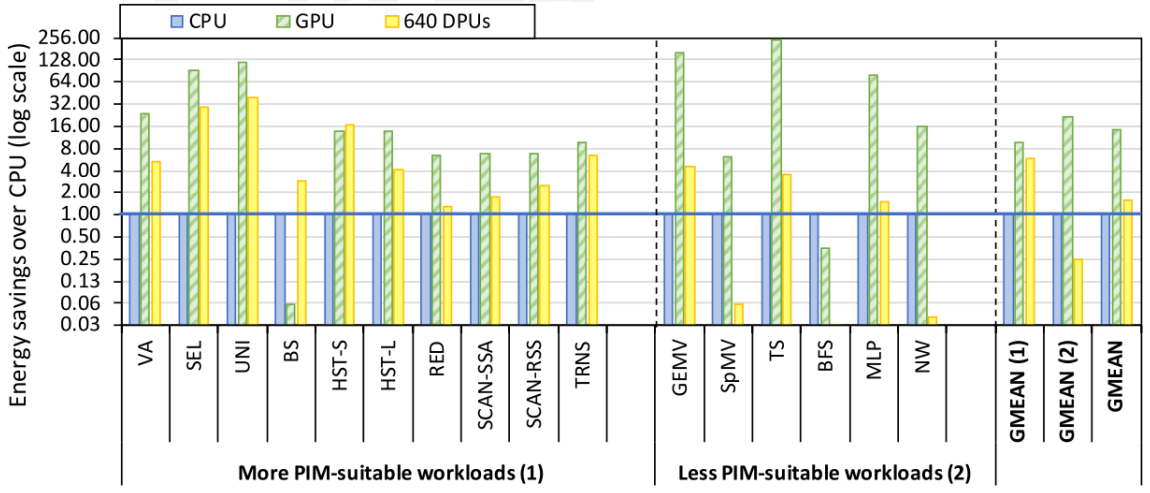
System	Process Node	Processor Cores			Memory		TDP
		Total Cores	Frequency	Peak Performance	Capacity	Total Bandwidth	
Intel Xeon E3-1225 v6 CPU [106]	14 nm	4 (8 threads)	3.3 GHz	26.4 GFLOPS*	32 GB	37.5 GB/s	73 W
NVIDIA Titan V GPU [192]	14 nm	80 (5,120 SIMD lanes)	1.2 GHz	12,288.0 GFLOPS	12 GB	652.8 GB/s	250 W
2,556-DPU PIM System	2x nm	2,556[9]	350 MHz	894.6 GOPS	159.75 GB	1.7 TB/s	383 W†
640-DPU PIM System	2x nm	640	267 MHz	170.9 GOPS	40 GB	333.75 GB/s	96 W†

*$Estimated\ GFLOPS = 3.3\ GHz \times 4\ cores \times 2\ instructions\ per\ cycle.$

†$Estimated\ TDP = \frac{Total\ DPUs}{DPUs/chip} \times 1.2\ W/chip$ [52].



性能对比



功耗对比

Juan et al. ***Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture***

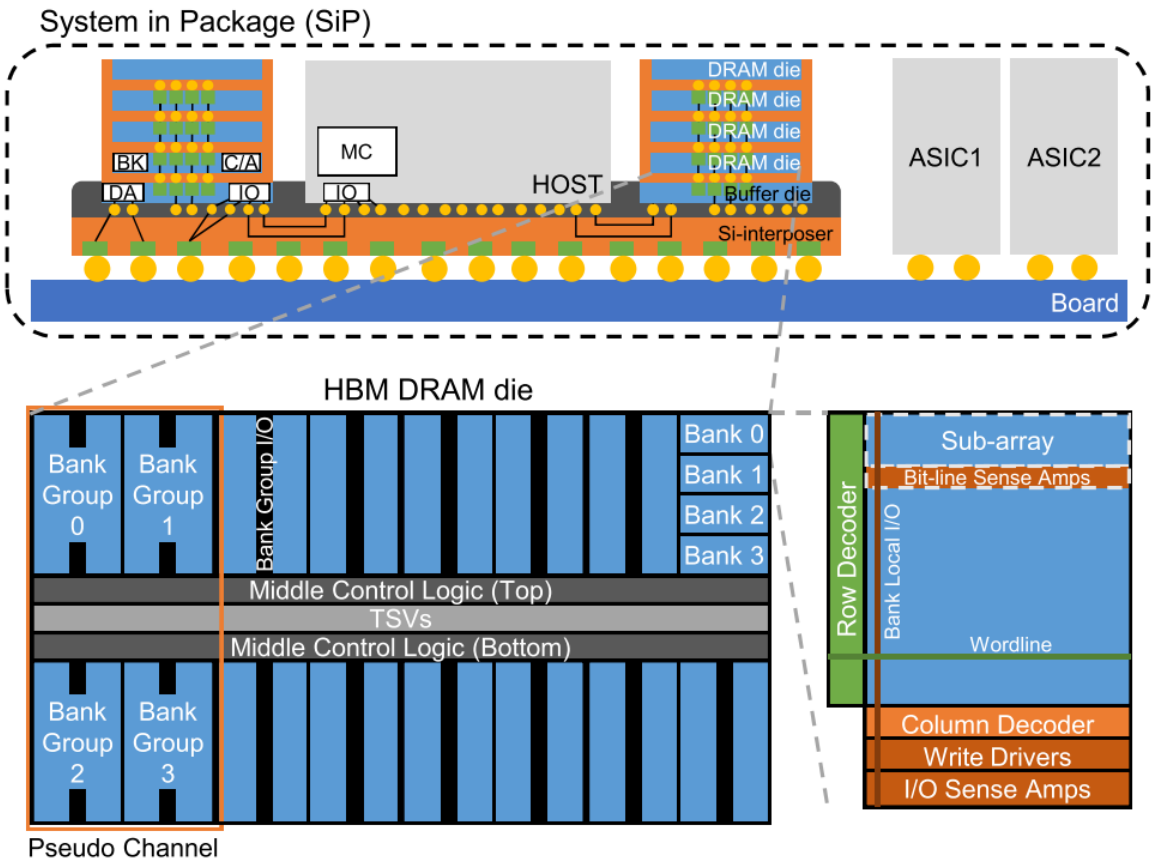
UPMEM PIM-DIMM

北京大学 PEKING UNIVERSITY

- 优势
  - Scalable memory bandwidth，相对于HBM等容量更大 （同代对比）
  - 通用编程模型，且兼容目前的server系统
- 缺点
  - 无法当作普通main-memory 使用：CPU 和Memory 都有内存控制器，协调困难，存在 Memory interleaving 、Cache coherence 问题和系统支持问题 （虚拟内存等）
  - 数据传输性能是瓶颈，难以处理需要频繁同步的应用
  - 应用切分困难：总共包含 2048个DPU，64MB local DRAM/DPU，DPU之间通信代价高
  - 无SIMD / FP 单元，浮点算力太低，在深度学习应用场景受限

Samsung's PIM-HBM

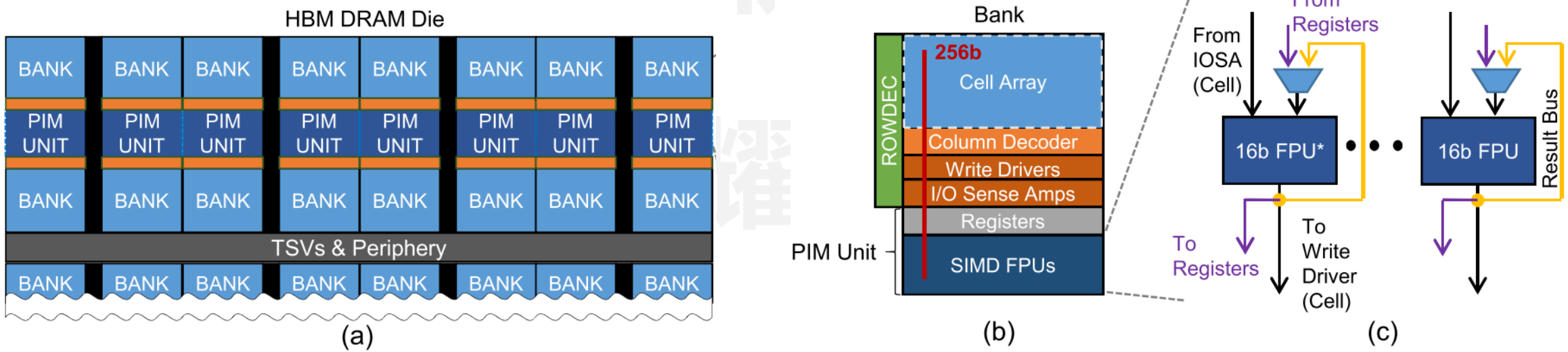
- **HBM的带宽对于访存密集型深度学习算子仍然存在不足**



- **HBM2 架构：**
  - **4 DRAM Die**
  - **4 pseudo channels (pCH) per Die**
  - **4 bank group per pCH**
  - **4 banks per bankgroup**
  - **256-bit data block over 4 64-bit over one pCH**
  - **256GB/s (1GHz IO)**

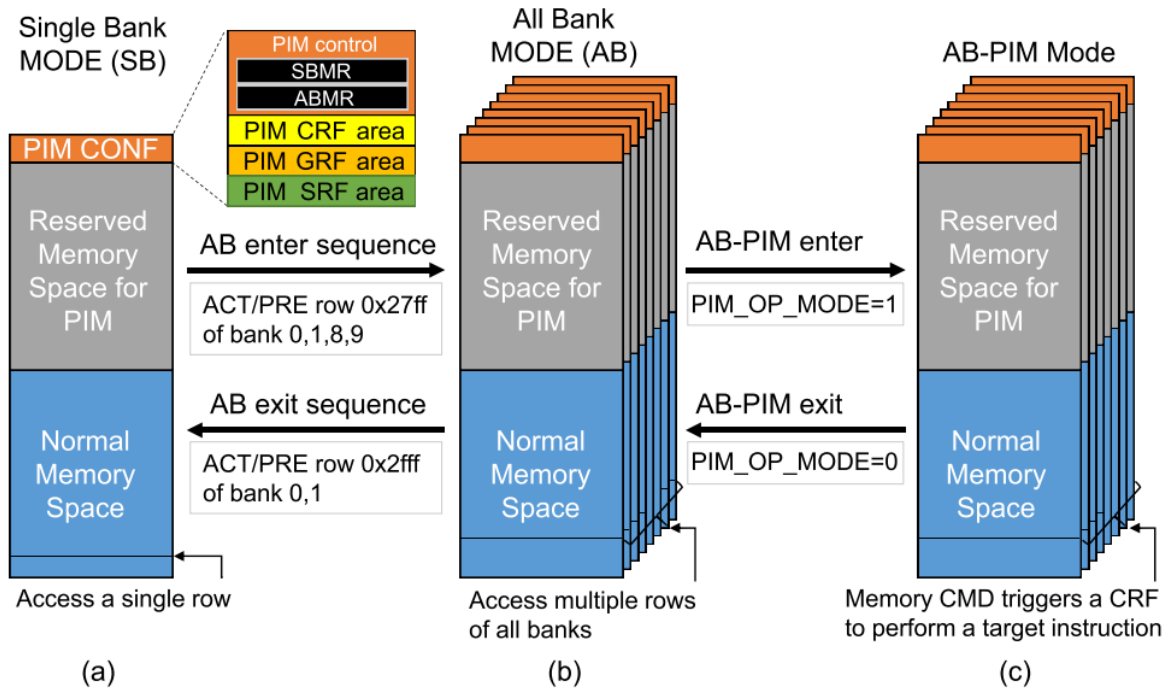
Samsung's PIM-HBM

- **设计目标**： 支持PIM和普通HBM模式，维持原本的DRAM bank和阵列设计
- 在Bank的I/O boundary处设置PIM单元
  - Bank row数量减半，两个bank共享PIM单元，每个PIM单元内包含16 x 16bit的SIMD单元
- PIM单元由标准的DRAM column commands（RD、WD）控制
  - 和UPMEM相似，指令提前load进memory，用RD指令操作数地址，一个channel所有bank可以并行读出4096个bit，RD指令同时激活PIM单元的计算



Samsung's PIM-HBM

- PIM-HBM 三种操作模式
  - Single Bank Mode
    - 标准DRAM 模式
    - 每次访问一个channel中的一个Bank
  - All Bank Mode
    - 一个DRAM 指令可以同时控制多个Bank
    - 8x higher on-chip bandwidth
  - All Bank PIM Mode
- 通过写high row地址以及PIM mode 寄存器来切换mode

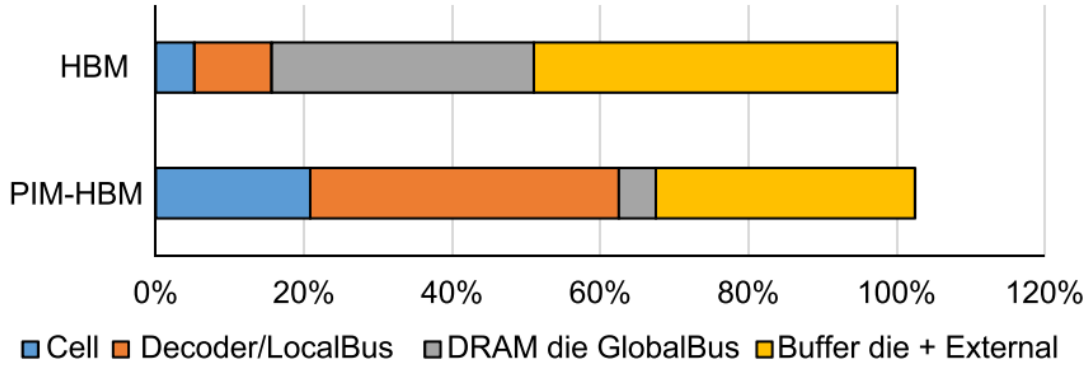


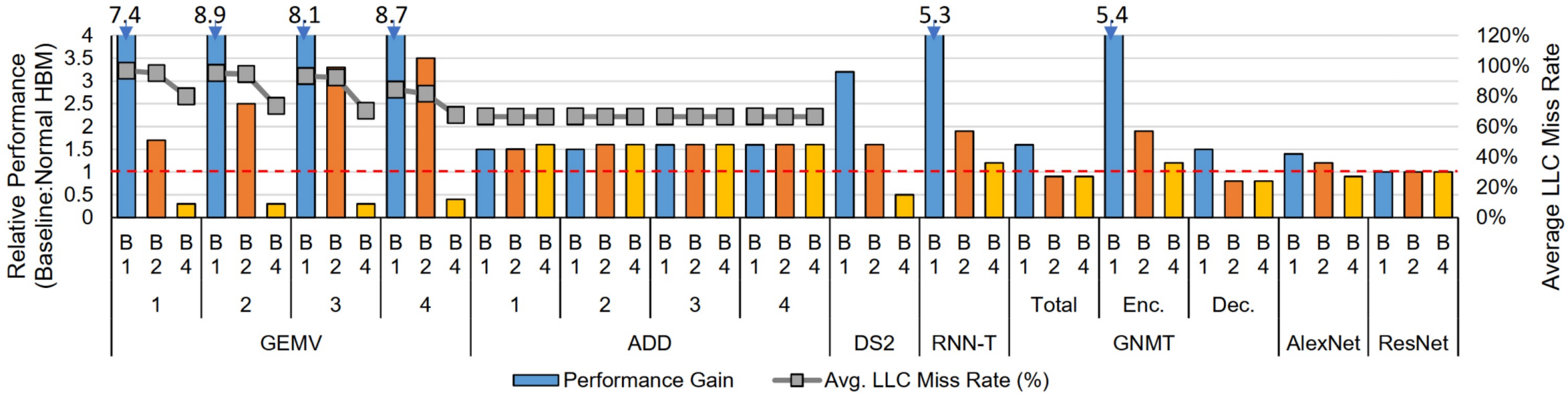
Samsung's PIM-HBM

- **访存密集型应用上加速比明显**

TABLE VI: Microbenchmark.

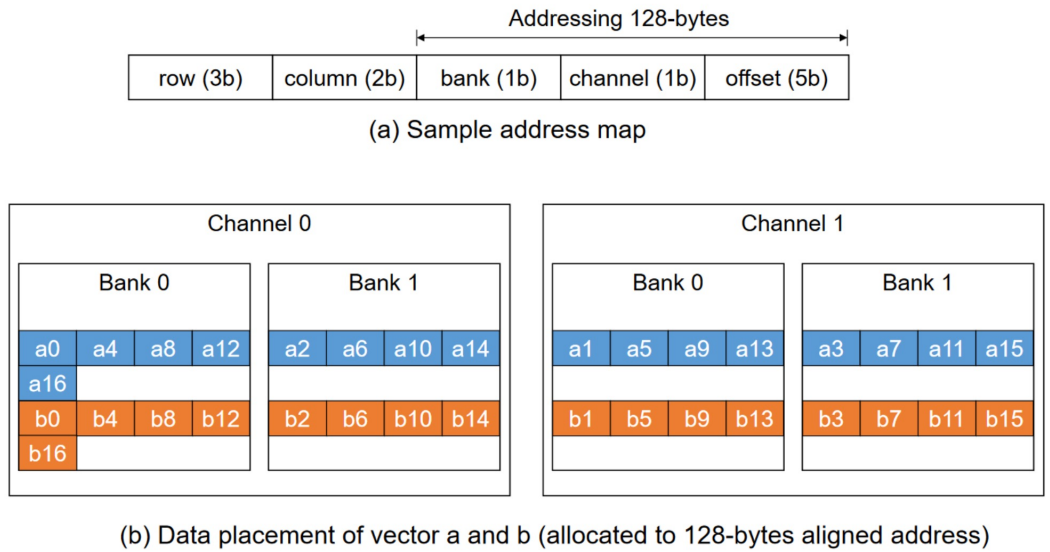
Name	GEMV Dim.	Name	ADD Dim.
GEMV1	1k × 4k	ADD1	2M
GEMV2	2k × 4k	ADD2	4M
GEMV3	4k × 8k	ADD3	8M
GEMV4	8k × 8k	ADD4	16M



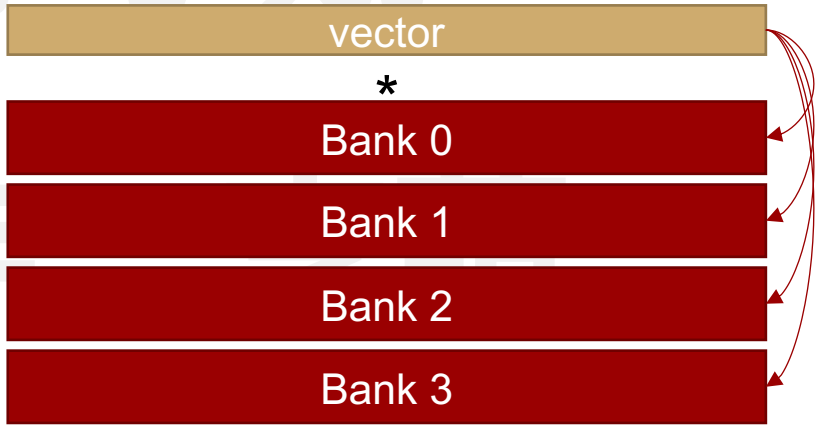


Samsung's PIM-HBM

- **Data Interleaving 问题：**
  - **对于elementwise 应用，interleaving 不影响计算结果**
  - **对于GEMV类应用，权重矩阵需要软件层进行重排**
- **控制粒度问题：**
  - **PIM计算和Host计算无法细粒度pipeline**
  - **多cube后bank数太多，小规模算子利用率低**
- **数据Replica 问题**
  - **以GEMV 为例，Host 需要向每个bank 写入输入**
- **PIM间没有直接的数据通路，需要Host进行转发**

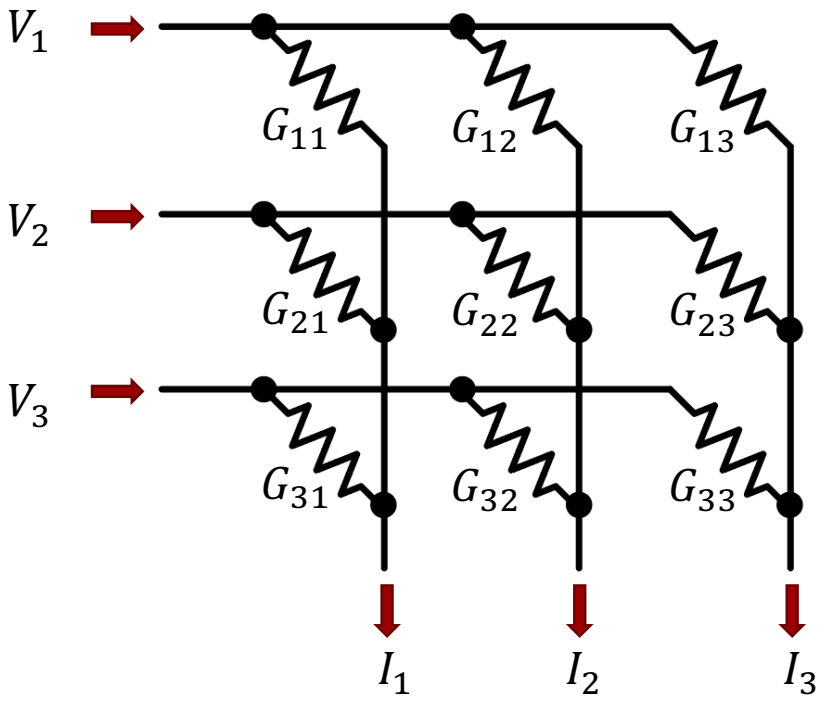


数据interleaving



存内计算

- 存内计算可以分为**模拟**存内计算和**数字**存内计算
- 模拟存内计算利用基尔霍夫电流定律，可以高效实现矩阵向量乘法



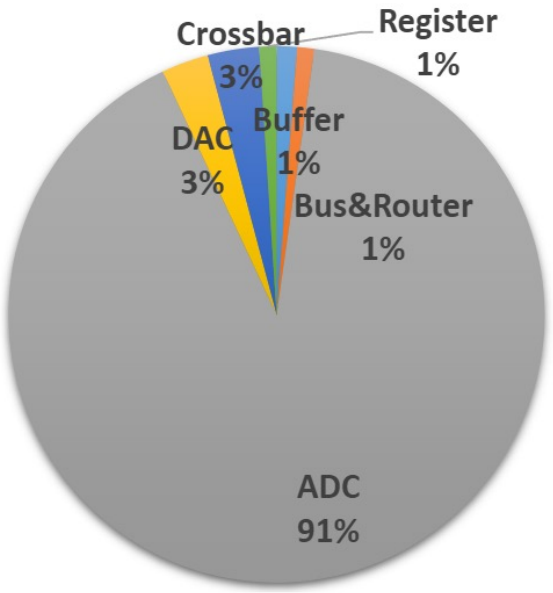
inputs　weights　outputs

$$[V_1 \quad V_2 \quad V_3]\begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{21} & G_{22} & G_{23} \\ G_{31} & G_{32} & G_{33} \end{bmatrix} = [I_1 \quad I_2 \quad I_3]$$

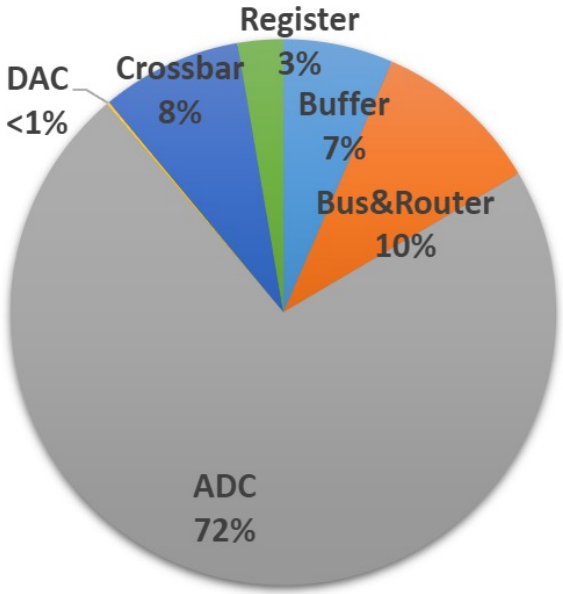
- 存内计算介质可以采用SRAM、DRAM、Flash或者其他新型存储器

模拟存内计算面临的问题

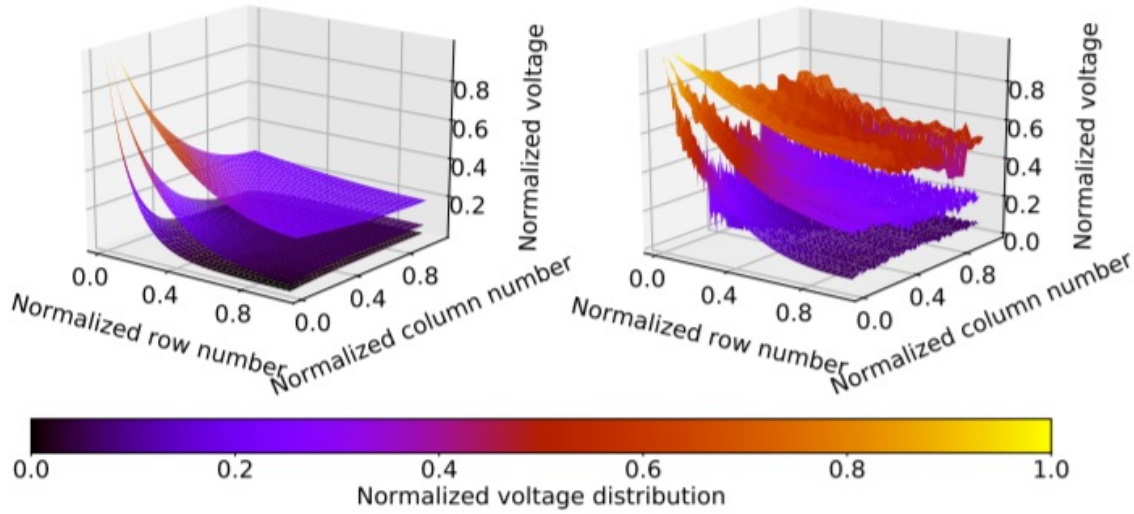
- **模拟存内计算主要面临以下挑战：**
  - **ADC/DAC的开销很大**
  - **众多非理想效应，例如IR drop、stuck-at faults、高阻态和低阻态等**



(a) Energy breakdown.



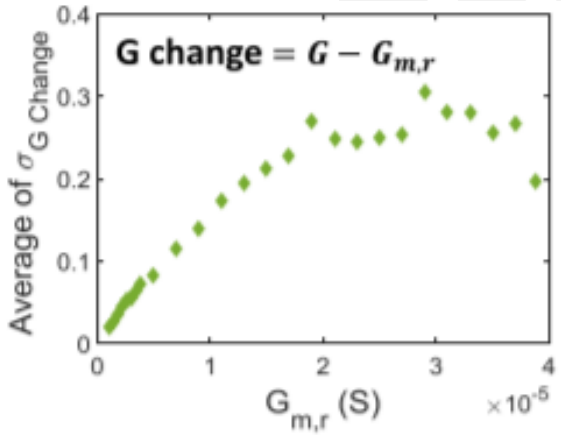
(b) Area breakdown.

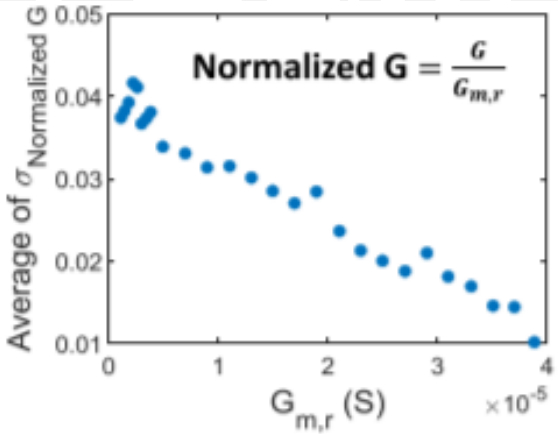


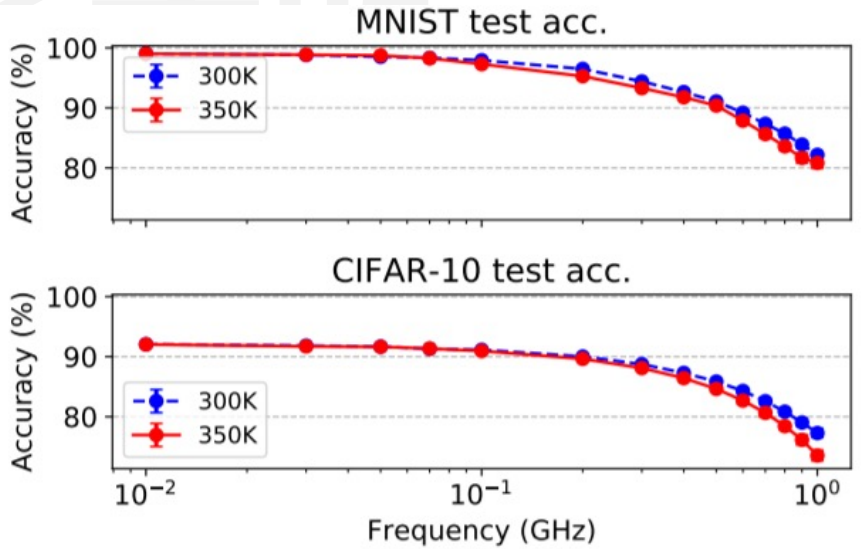
IR Drop

模拟存内计算面临的问题

- 模拟存内计算主要面临以下挑战：
  - ADC/DAC的开销很大
  - 众多非理想效应，例如IR drop、stuck-at faults等
  - 数据保持错误、编程错误等
  - 本征噪声，例如热噪声、随机电报噪声等 → 如何避免噪声影响成为了重要的研究方向

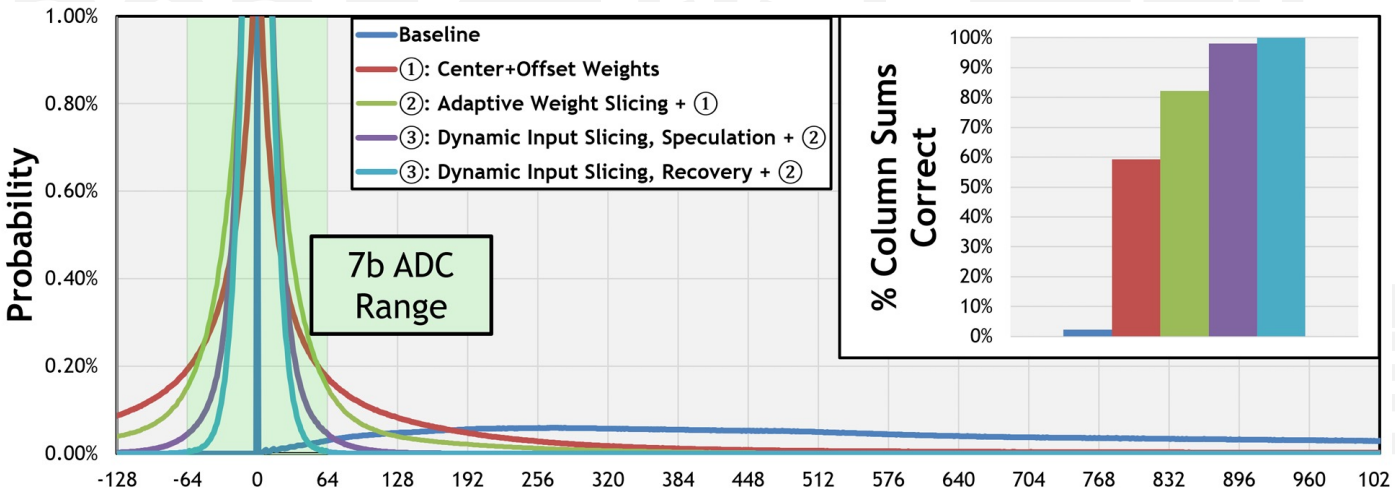






代表性架构RAELLA

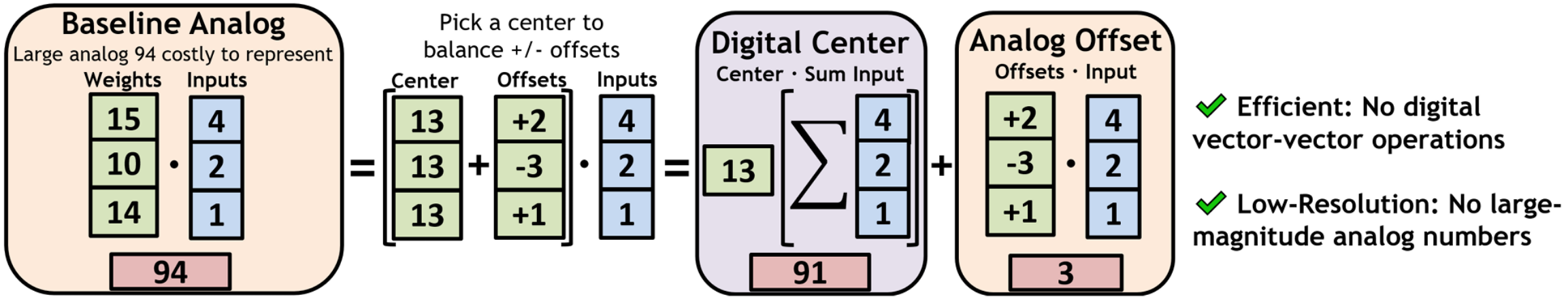
- MIT Vivienne Sze组发表于ISCA 2023
- 基于RRAM的存内计算架构，文章核心在于不进行重训练的前提下，降低ADC精度
  - 原有方法需要调整模型参数或者直接使用低精度ADC，往往需要重训练或造成计算误差
- 提出一系列方法，降低输出结果的范围，包括 (1)center + offset weights，(2) adaptive weight slicing, (3) dynamic input slicing



Tanner Andrulis et al., *RAELLA: Reforming the Arithmetic for Efficient, Low-Resolution, and Low-Loss Analog PIM: No Retraining Required!*, ISCA 2023

代表性架构RAELLA

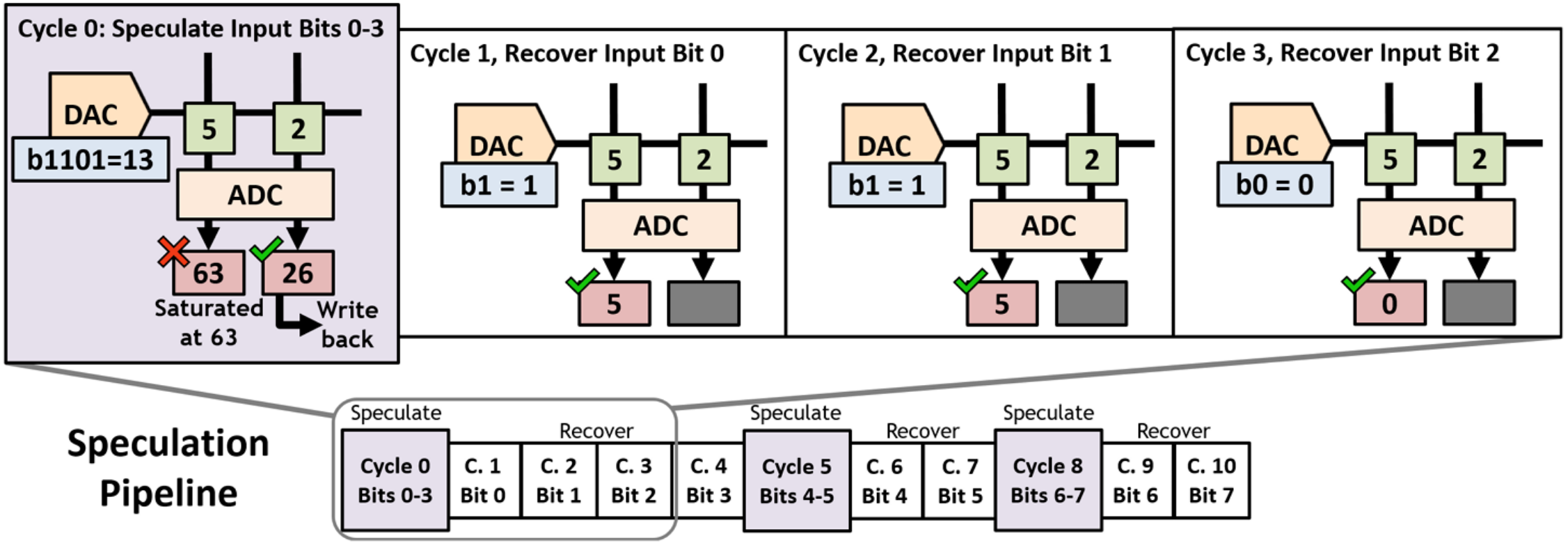
- Center + offset weights
- 通过移动0点，均衡模型权重中的正负值，并通过支持有符号数计算，降低累加结果范围



Tanner Andrulis et al., *RAELLA: Reforming the Arithmetic for Efficient, Low-Resolution, and Low-Loss Analog PIM: No Retraining Required!*, ISCA 2023

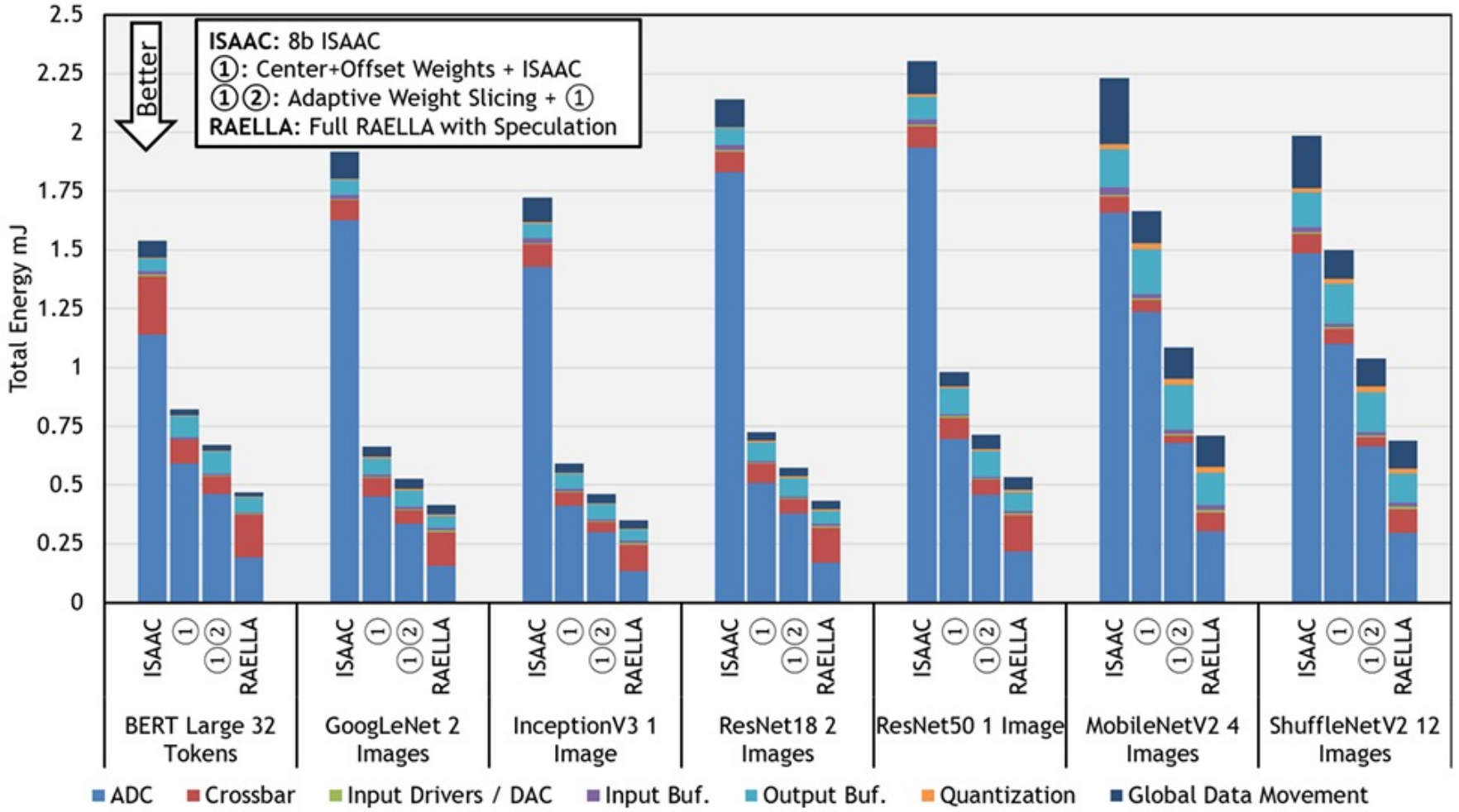
代表性架构RAELLA

- Adaptive weight slicing + dynamic input slicing
- 通过控制权重和输入的slicing比特位数，平衡计算次数与单次计算的累加结果范围



Tanner Andrulis et al., *RAELLA: Reforming the Arithmetic for Efficient, Low-Resolution, and Low-Loss Analog PIM: No Retraining Required!*, ISCA 2023

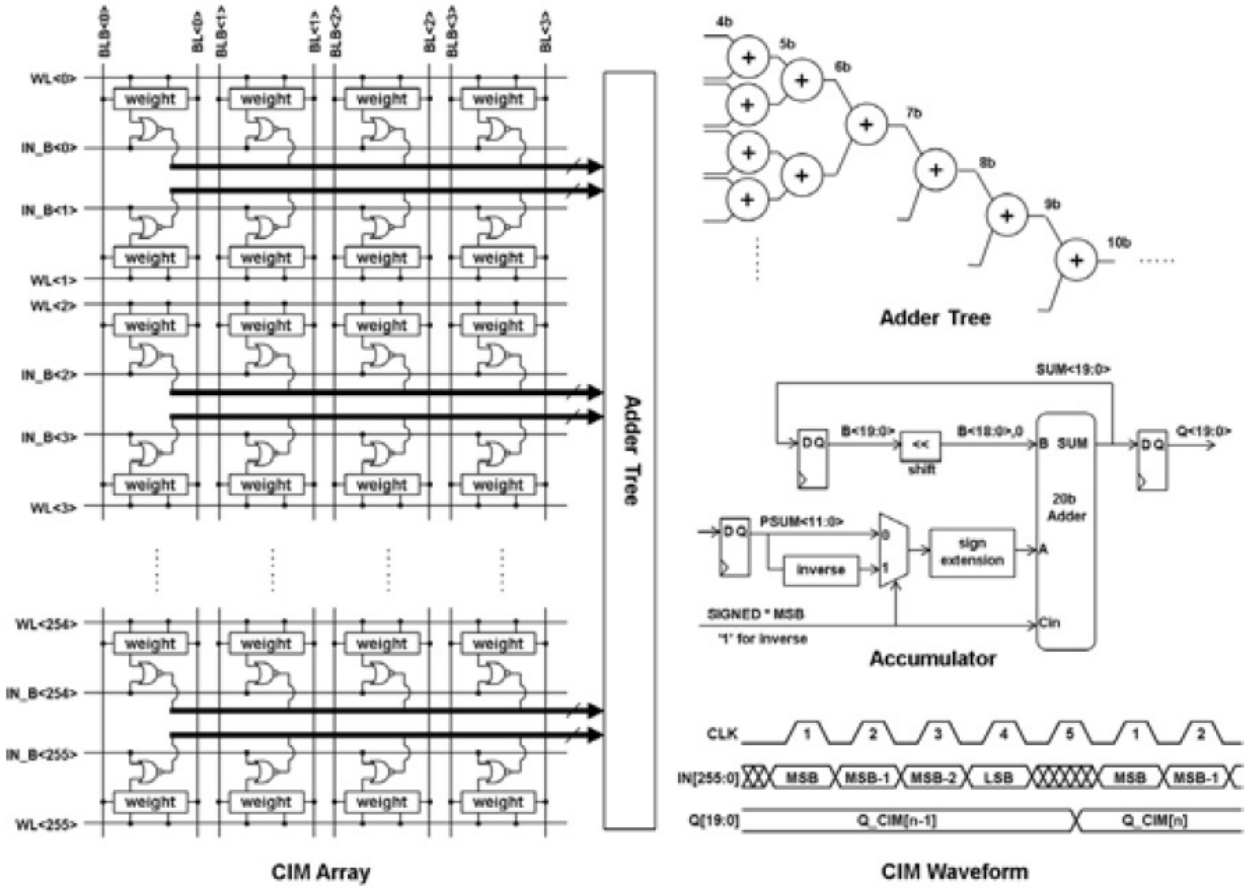
代表性架构RAELLA



Tanner Andrulis et al., ***RAELLA: Reforming the Arithmetic for Efficient, Low-Resolution, and Low-Loss Analog PIM: No Retraining Required!***, ISCA 2023

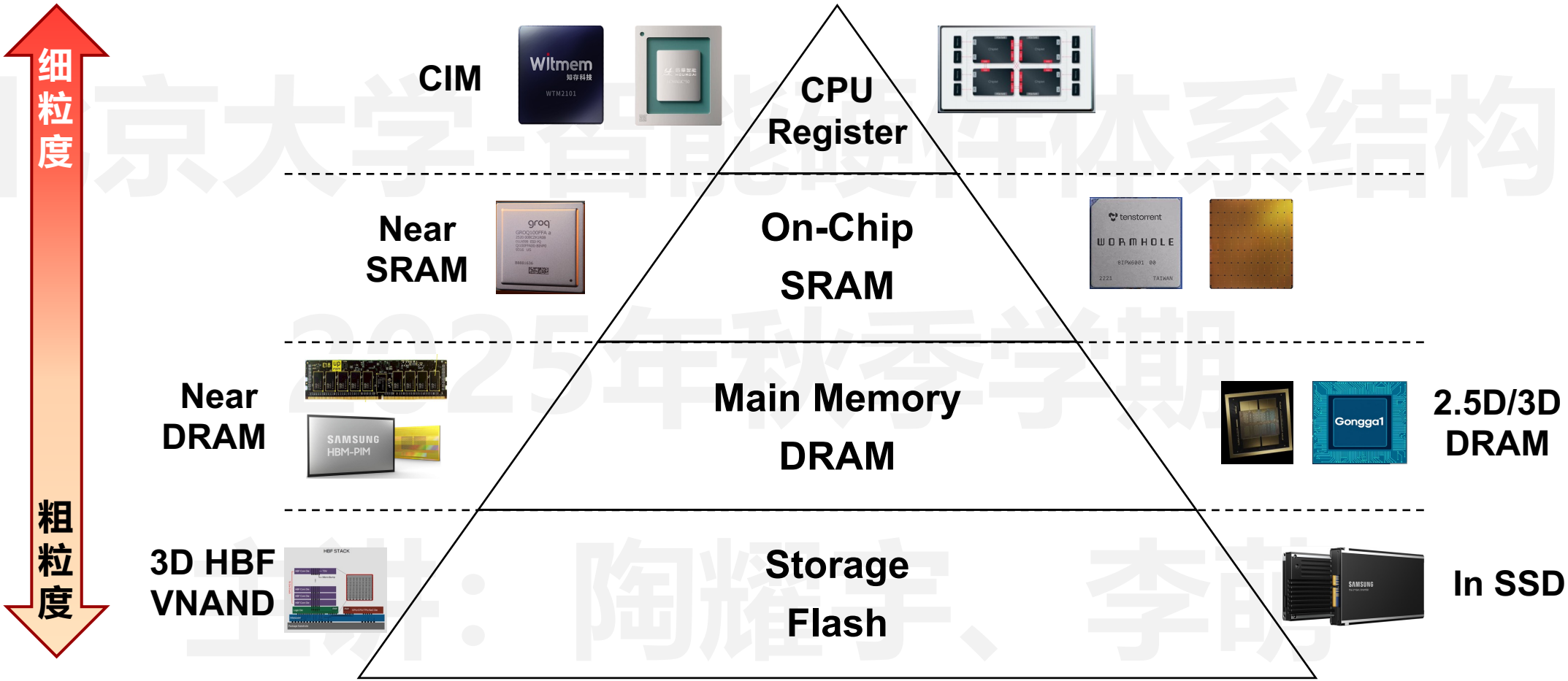
数字存内计算

- 为了克服模拟存内计算的精度等挑战，**数字存内计算**被提出，代表性架构包括SRAM数字存算



- 显著降低噪声影响，准确率更高
- 但是计算并行度较低，累加树开销较大

存算一体分类



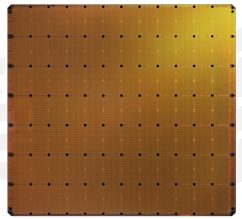
存算一体分类

■ 细颗粒度融合：提升计算效率； 粗颗粒度融合：提升访存效率

知存科技

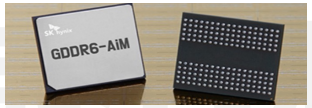
后摩智能

d-matrix

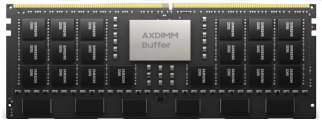


Cerabras

Groq Tenstorrent



SK Hynix



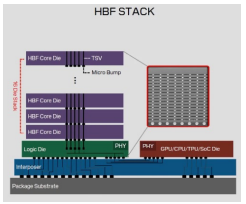
Samsung

Upmem

瑞芯微

Samsung

Nvidia

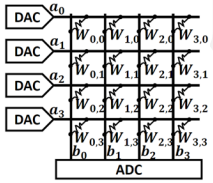


SanDisk

细颗粒度、突出计算特性 ⟷ 粗颗粒度、突出存储特性

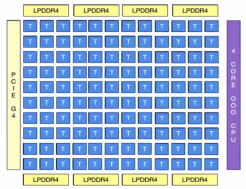
单元（Cell）级

算力/容量
~1TOPs/10KB



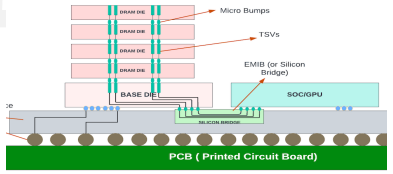
阵列（Bank)级

算力/容量
~1TOPs/1MB



芯粒/芯片级

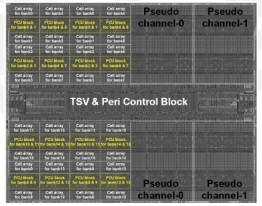
算力/容量
~1TOPs/100MB+



存算一体分类

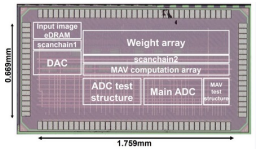
北京大学 PEKING UNIVERSITY

NDP+DRAM:
数字计算、通用
利用bank级别并行
神经网络、图计算



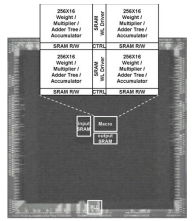
Samsung
ISSCC 2021

CIM+DRAM
低比特（1bit/cell）
模拟计算
面向神经网络

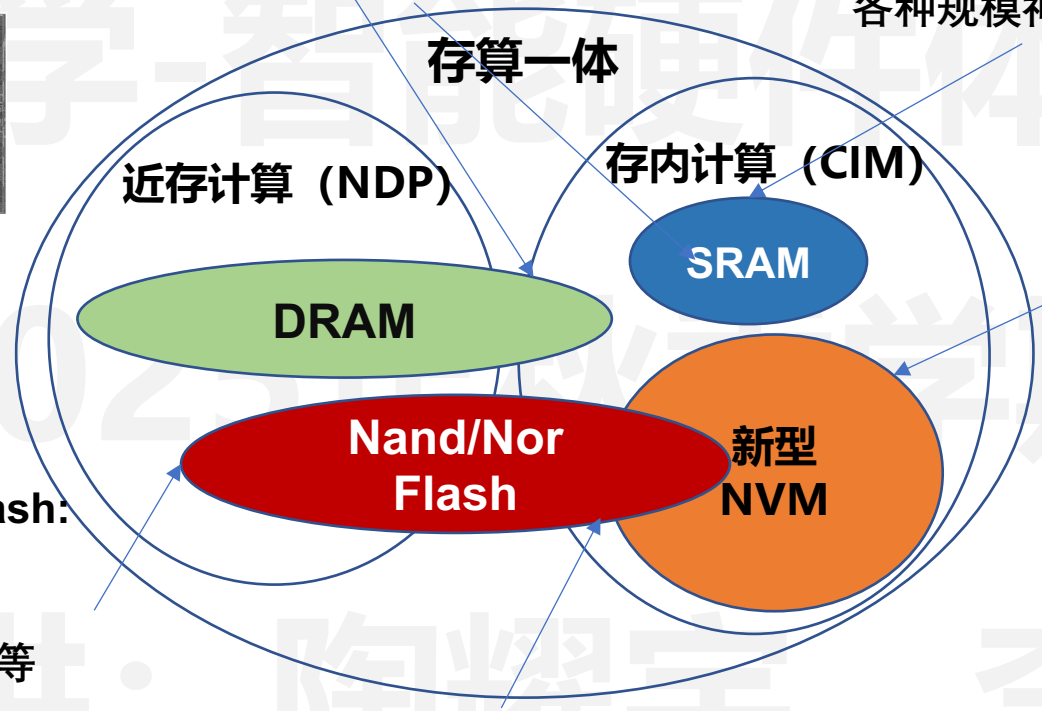


UT Austin、Intel
ISSCC 2021

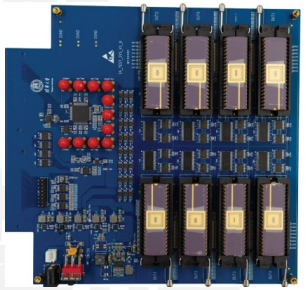
CIM+SRAM
中-高比特
数字、模拟计算
工艺成熟、频率高
各种规模神经网络



TSMC
ISSCC 2021



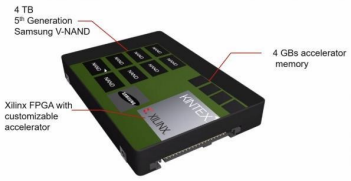
CIM+新型NVM
低、中比特（1~4bit/cell）
数字、模拟计算
工艺成熟度问题
神经网络、通用逻辑



清华大学
Nature 2020

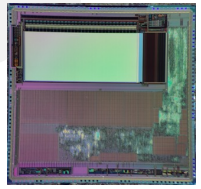
国立清华
TSMC
ISSCC 2021

NDP+NAND Flash:
数字计算、通用
SSD+计算单元
数据搜索、分析等
有商业化产品



三星 SmartSSD CSD

CIM+Nor Flash
高比特（>7bit/cell）
模拟计算、功耗低
工艺成熟、节点受限
低功耗AIoT场景



知存科技

本学期总结

北京大学 PEKING UNIVERSITY

智能硬件体系结构简介

智能芯片发展史与未来趋势

半导体晶体管与数字逻辑基础

流水线架构

指令集与流水线设计

数字逻辑与复杂计算单元

超标量与乱序执行

指令动态发射与分支预测

MIPS架构与缓存设计

近存计算与存内计算

软硬件协同设计

人工智能加速器架构

课程评估与总结

- **课程评估：扫右侧二维码或登录网上评估系统（kcpg.pku.edu.cn）**
- **希望大家后续科研学习一切顺利，也欢迎在课程结束后，同学们仍然能找我讨论有趣的问题**

- **办公室：资源西楼2213A**
- **邮箱：meng.li@pku.edu.cn**
- **电话：18701103305（微信同）**