HPACuma Research Group

ALDEBARAN Research Group

UNIVERSIDAD DE MÁLAGA | uma.es

Processing in Memory

Oscar Plata

oplata@uma.es

Dept. Arquitectura de Computadores
Universidad de Málaga

HPACuma Research Group
ALDEBARAN Research Group

UCM, May 26, 2021

Data is Key in Modern and Future Workloads

- **Important workloads are all data intensive**
  - Artificial Intelligence, Machine Learning, Genomics, Time Series Analysis, In-Memory Databases, Graph/Tree Processing, Data Analytics, UHD Video Analysis…
- **Such workloads require fast and efficient processing for large amounts of data**
- **Data is increasing…**

Data overwhelms modern computers

Data $\Longrightarrow$ Performance and energy bottleneck

Computing Behaviour of Modern Workloads

- **Frequent memory access patterns in modern workloads**
  - **Streaming** access (lack of data reuse)
  - **Strided** access (with large strides)
  - **Non regular** access (even random), specially when operating on sparse data structures

Graph processing

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

PageRank algorithm

Genome sequence alignment

```
sp = C[Q[m]]
ep = C[Q[m]+1]
for i from m-1 to 1 step -1{
  sp = LF(Q[i],sp)
  ep = LF(Q[i],ep)
}
```

Backward search exact matching

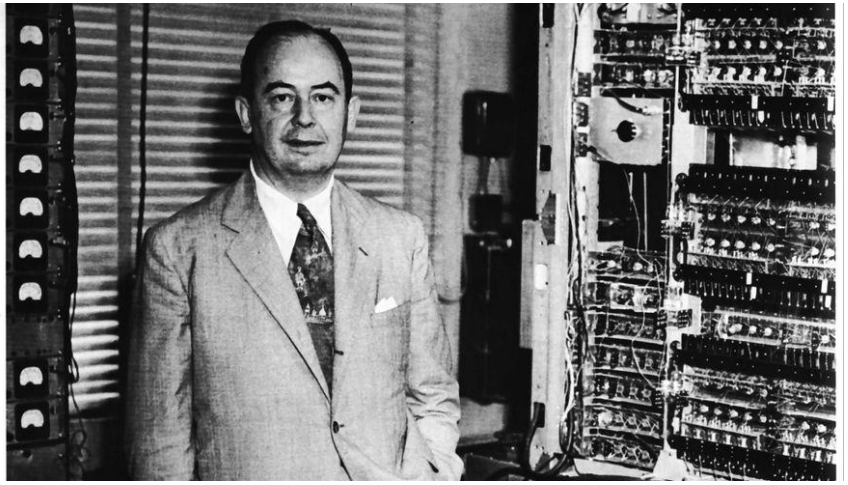
- **Usually, little amount of computation (low operational intensity)**

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

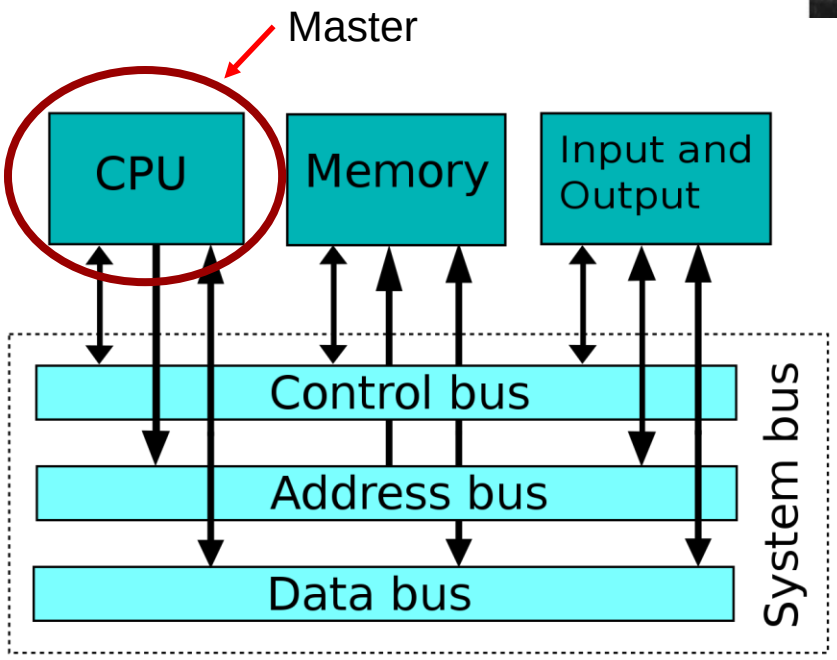
Accelerating Sequence Alignments Based on FM-Index Using the Intel KNL Processor, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2020

Conventional Computing System

- **Four key components**
  - Computation
  - Memory
  - Input/Output
  - Communication



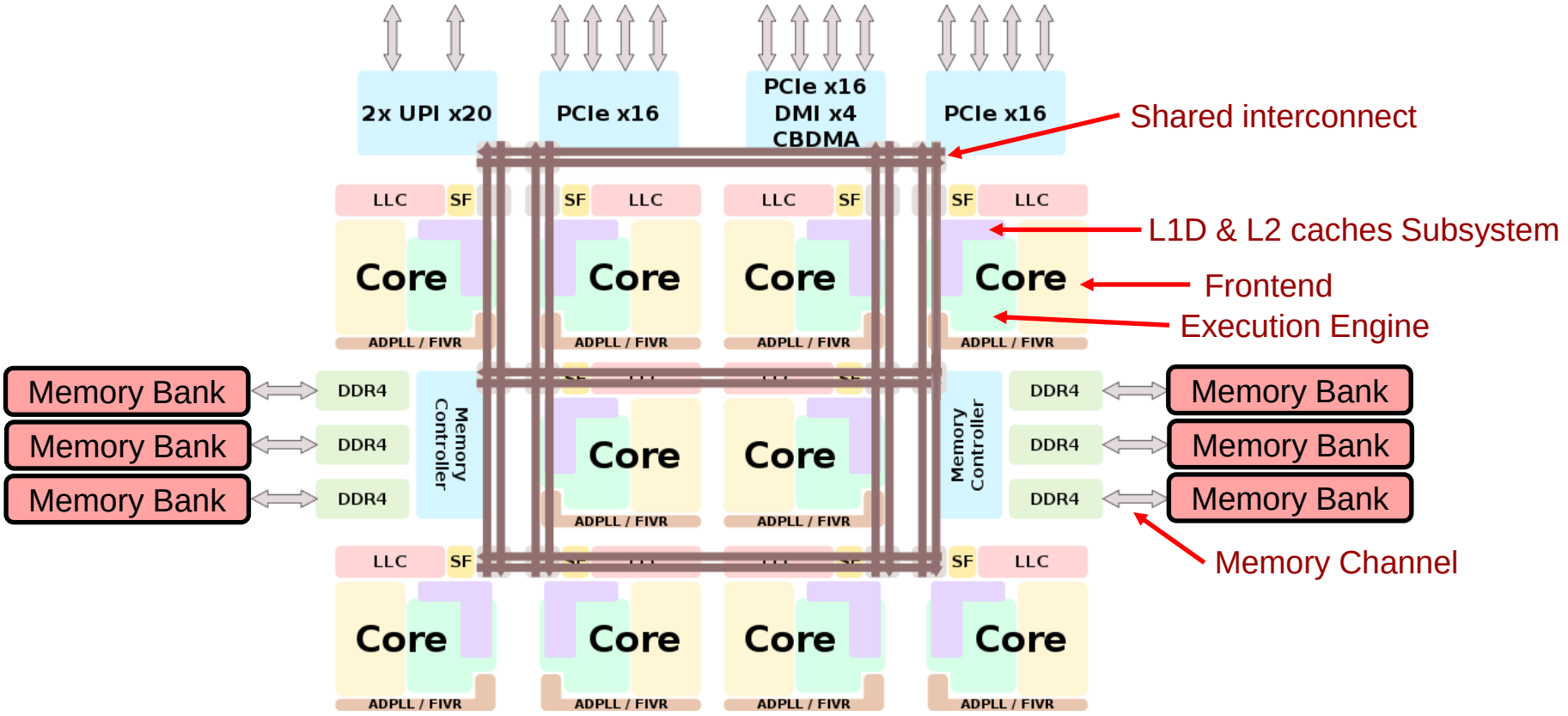
John Louis von Newmann & UNIVAC



Processor-Centric Design

- **All data is processes in the CPU**
  - At great system cost
- **Processors are heavily optimized and considered the master**
- **Data storage units are dumb**

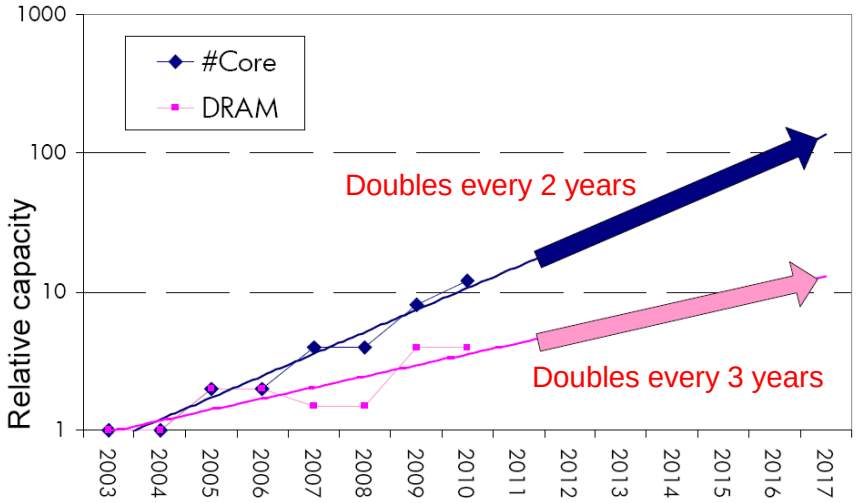
A Processor-Centric Design (I)

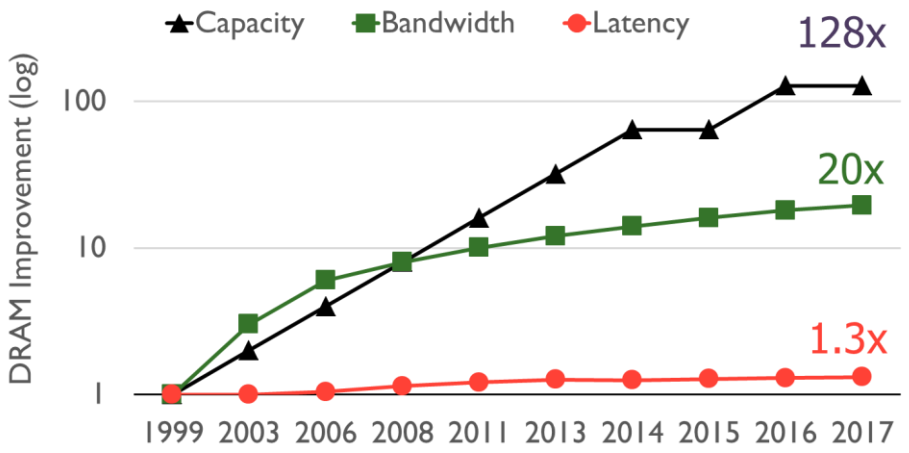


A great part of the processor is dedicated to **storing** and **moving** data

Main Memory Trends (I)

- **Difficulty in scaling memory: capacity, bandwidth, latency (DRAM)**
  - **Multi-core**: increasing number of cores asking for data
  - **Data-intensive applications**: increasing demand for data





Memory capacity per core drops by 30% every 2 years

Trends worse for **memory bandwidth per core**

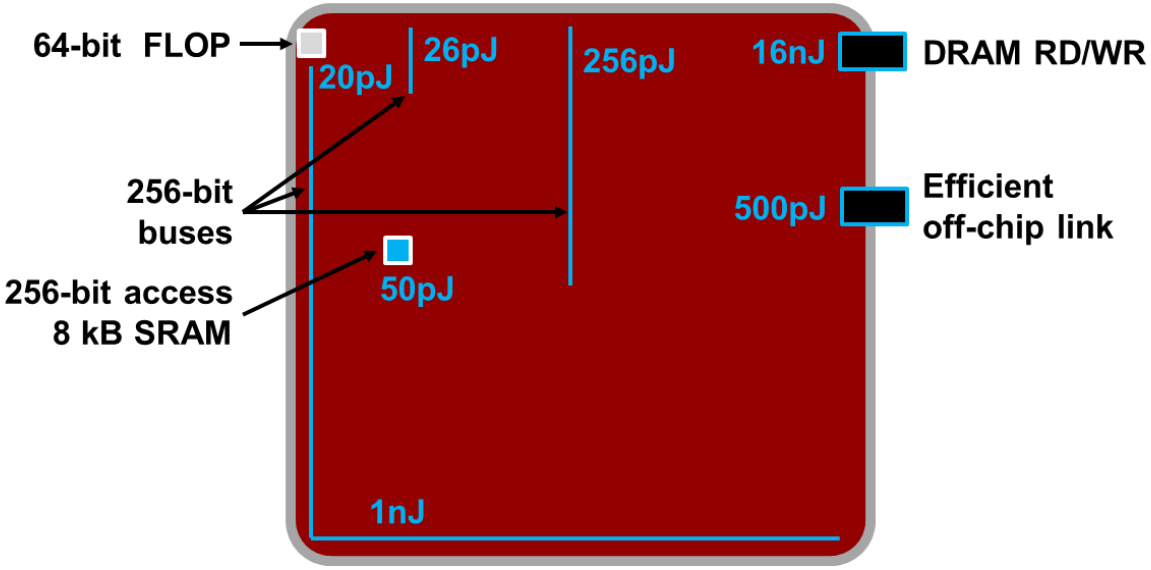
Scaling for DRAM capacity, bandwidth, latency

Disaggregated Memory for Expansion and Sharing in Blade Servers, ISCA 2009

Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization, SIGMETRICS 2016

Main Memory Trends (II)

- **Data movement dominates computation in term of energy/power**
  - **Scientific apps**: **~40%** of the total energy is spent on data movement
  - **Mobile apps**: **~35%** of the total energy is spent on data movement
  - **Consumer apps**: **~62%** of the total energy is spent on data movement



28nm nVIDIA (*Source: W. Dally*)

Quantifying the Energy Cost of Data Movement in Scientific Applications, IISWC 2013

Quantifying the Energy Cost of Data Movement for Emerging Smart Phone Workloads on Mobile Platforms, IISWC 2014

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks, ASPLOS 2018

Main Memory Trends (III)

- **DRAM scaling has become increasingly difficult**
  - Cell leakage current, cell reliability, manufacturing difficulties…
  - Difficult to continue the improvements in capacity, energy and latency
- **Emerging memory technologies are promising**
  - **Reduced-latency DRAM**: RLDRAM, TL-DRAM…
  - **Low-power DRAM**: LPDDR4…
  - **3D-stacked RAM**: HBM, HMC…
  - **Non-volatile memory (NVM)**: PCM, MRAM, STT-MRAM, ReRAM…

A Processor-Centric Design (II)

- **The system is grossly imbalanced**
  - Processing is done only in one place (execution engines)
  - The rest of the system just stores and moves data
  - Modern workloads cause huge data movement
- **The processor becomes overly complex**
  - Required to tolerate data accesses from memory
    - » Complex multi-level cache hierarchies
    - » Multiple complex hardware prefetching techniques
    - » High amounts of multithreading
    - » Complex out-of-order (OoO) execution mechanisms

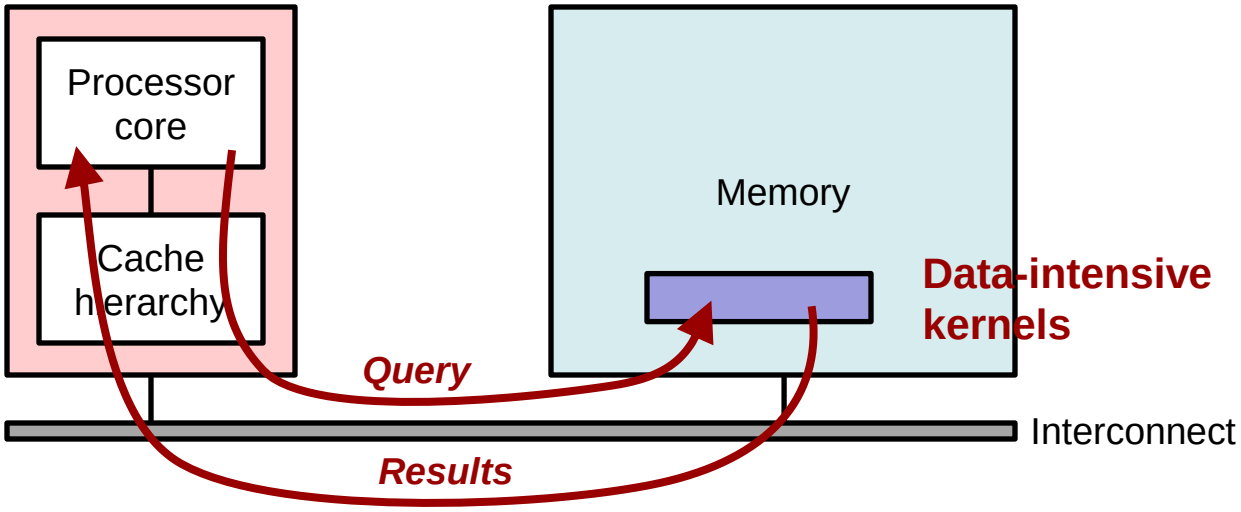
Data movement causes energy waste and latency

Processor complex mechanisms to tolerate latency, that causes additional energy waste ... **and latency**

Data-intensive apps make many techniques **innefficient**

A Data-Centric Design

- **Enable computation with minimal data movement**
- **Compute where data resides**
- **Make computing architectures more data-centric**



Data-Centric Computing

PIM
Processing-in-Memory

CIM
Computation-in-Memory

- **Memory technologies for PIM**
- **Approaches to implementing PIM**

Memory Technologies for PIM

- **3D-stacked memory architectures**

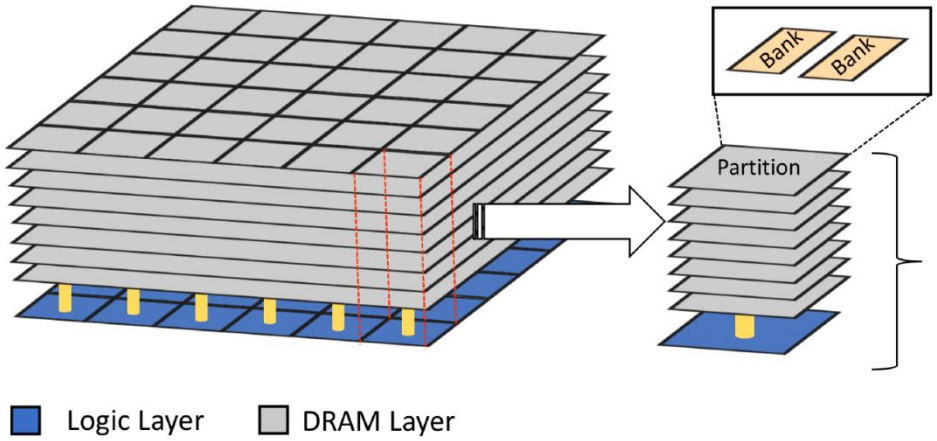
Segment	DRAM Standards & Architectures
Commodity	DDR3, DDR4
Low-Power	LPDDR3, LPDDR4
Graphics	GDDR5, GDDR6
Performance	eDRAM, RLDRAM
3D-Stacked	MCDRAM, HBM, HMC

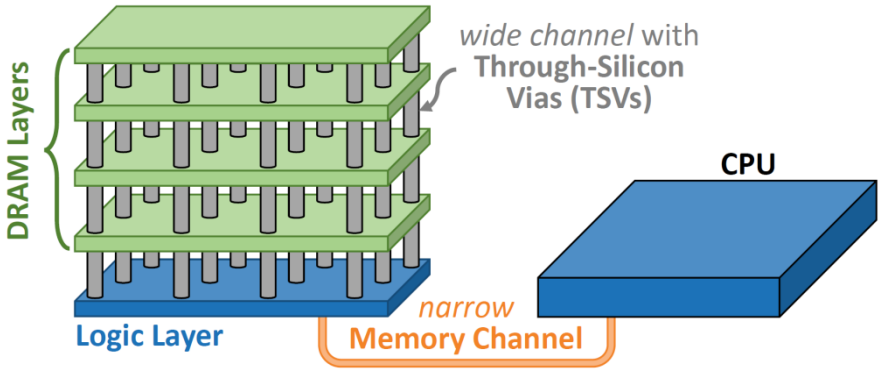
- **Non-volatile random-access memories (NVRAM)**

Technology	NVRAM Architectures
Resistive-Phase	PCM (Phase-Change Memory)
Resistive-Magnetic	MRAM (magnetoresistive RAM) STT-MRAM (Spin-Transfer Torque MRAM)
Resistive-Memristor	ReRAM (Resistive RAM)

3D-Stacked Memory Architectures

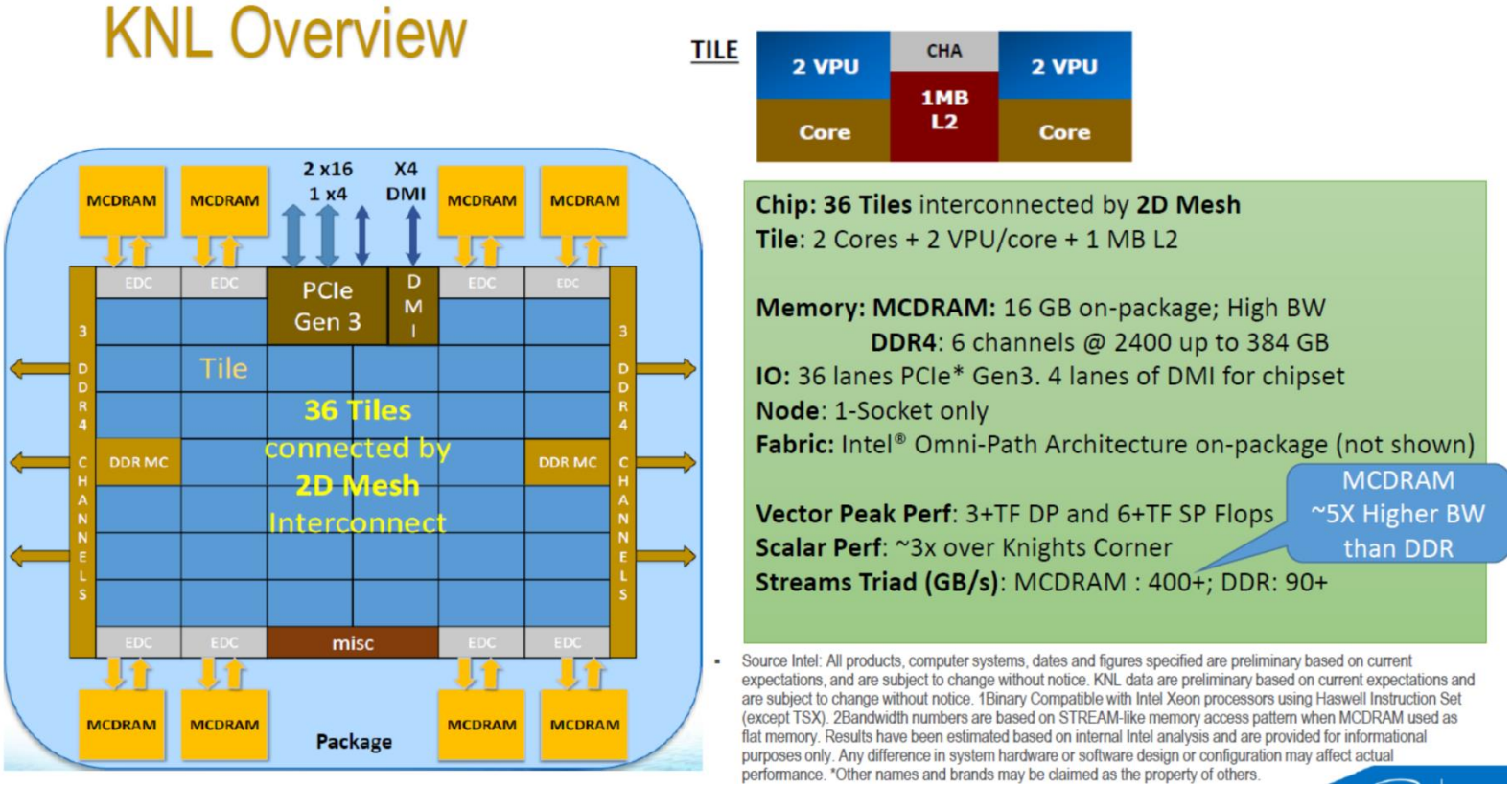
- **Multiple layers of DRAM memory are stacked**
  - The layers are partitioned into banks
  - Same banks in different layers are connected using TSVs (Through-Silicon Vias)
  - Thousands of TSVs provides internal **ultra high memory bandwidth**
  - Examples: HMC (Hybrid Memory Cube), HBM (High Bandwidth Memory), MCDRAM (Multi-Channel DRAM)





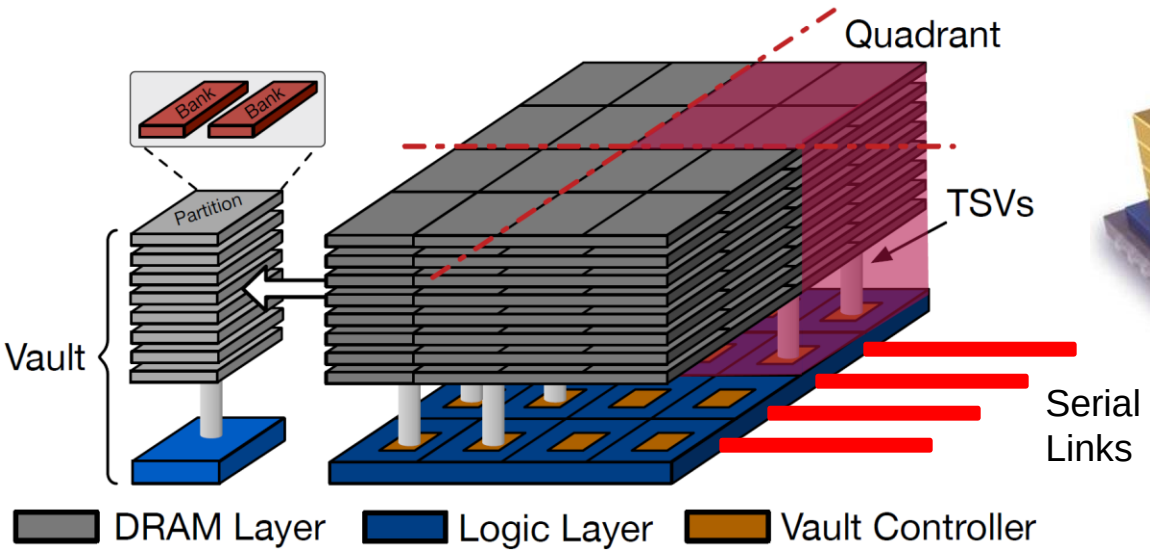
Multi-Channel DRAM (MCDRAM) - 2013/18

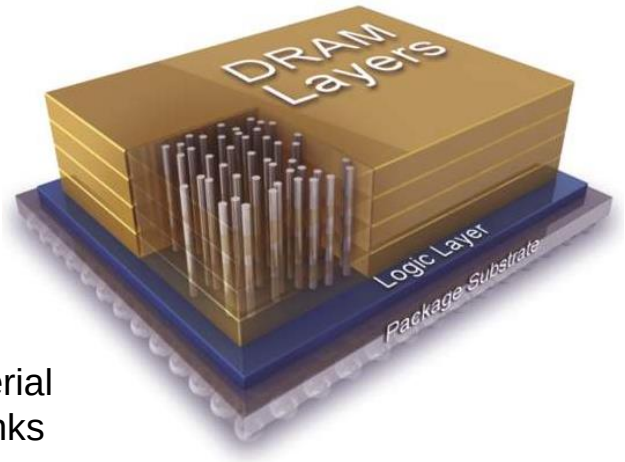
- **3D-stacked DRAM used in Intel Xeon Phi processor (Knights Landing)**
  - Variant of HMC developed with Micron
  - Ultra high bandwidth: ~400 GB/s (DDR4 bandwidth: ~90 GB/s)
  - Similar latency than DRAM DIMMs



Hybrid Memory Cube (HMC) - 2011/18

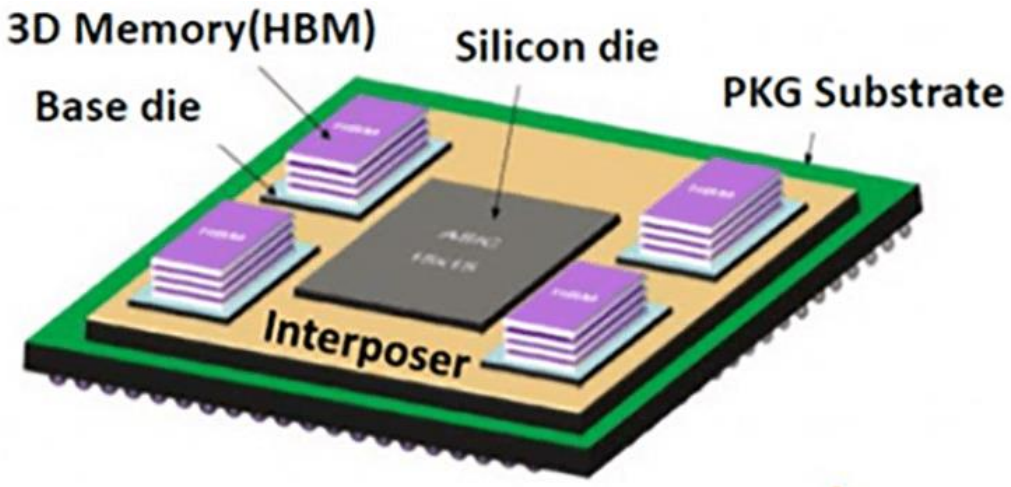
- **3D-stacked DRAM developed by Micron and Samsung**
  - HMC uses standard DRAM cells
  - HMC interface is **non-compatible** with DDRx
  - Serial link: 8/16 SerDes @ 10-15 Gbps (half- or full-duplex)
    - » 4-link@15Gbps fd HMC ⇨ 240 GB/s bandwidth
    - » 8-link@10Gbps fd HMC ⇨ 320 GB/s bandwidth





High Bandwidth Memory (HBM) - 2013

- **3D-stacked DRAM developed by Samsung, AMD and SK Hynix**
  - HBM uses standard DRAM cells (up to 8)
  - HMC interface is **compatible** with DDR4 or GDDR5
  - Ultra wide memory bus
    - » A 4-DRAM HBM stack has 128-bit 8 channels (=1024-bit bus)
    - » Overall package bandwidth: 128 GB/s
  - HBM2 doubles the package bandwidth of HBM (256 GB/s)



Memory Technologies for PIM

- **3D-stacked memory architectures**

Segment	DRAM Standards & Architectures
Commodity	DDR3, DDR4
Low-Power	LPDDR3, LPDDR4
Graphics	GDDR5, GDDR6
Performance	eDRAM, RLDRAM
3D-Stacked	MCDRAM, HBM, HMC

- **Non-volatile random-access memories (NVRAM)**

Technology	NVRAM Architectures
Resistive-Phase	PCM (Phase-Change Memory)
Resistive-Magnetic	MRAM (magnetoresistive RAM) STT-MRAM (Spin-Transfer Torque MRAM)
Resistive-Memristor	ReRAM (Resistive RAM)

Non-Volatile RAM (NVRAM)

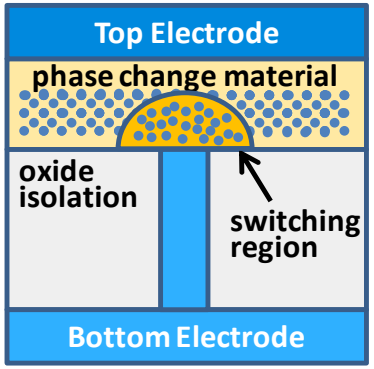
- **Byte-addressable resistive non-volatile random-access memory**
  - Emerging technologies
    - » Phase-change memory (**PCM**)
    - » Magnetoresistive RAM (**MRAM**) and Spin-Transfer Torque MRAM (**STT-MRAM**)
    - » Metal-oxide resistive RAM (**RRAM** or **ReRAM**) or memristors

	PCM	STT-MRAM	RRAM	DRAM	SRAM
Density (F^2)	4-30	6-50	<4	6-10	>100
W energy /bit (pJ)	~10	~0.1	~0.1	~0.01	~0.001
Read time (ns)	<10	<10	<10	~10	0.1-0.3
Write time (ns)	~50	<10	<10	~10	0.1-0.3
Retention	Years	Years	Years	10-20 ms	As V applied
Endurance (cyc.)	10^8-10^{15}	$>10^{15}$	10^8-10^{12}	$>10^{16}$	$>10^{16}$

An Overview of In-memory Processing with Emerging Non-volatile Memory for Data-intensive Applications, arXiv 2019

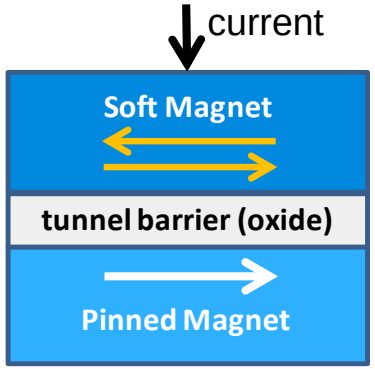
Non-Volatile RAM (NVRAM)

- **Promising resistive memory technologies**
  - Random access, resistive, non volatile, no erase before write



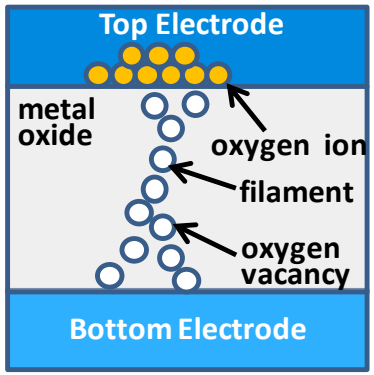
PCM

Phase Change Memory



STT-MRAM

Spin Transfer Torque Magnetic Random Access Memory

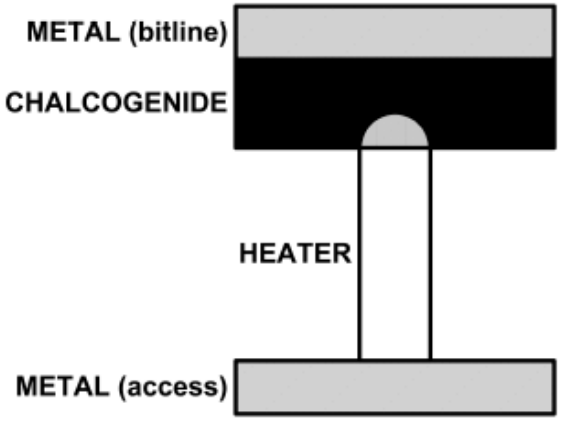


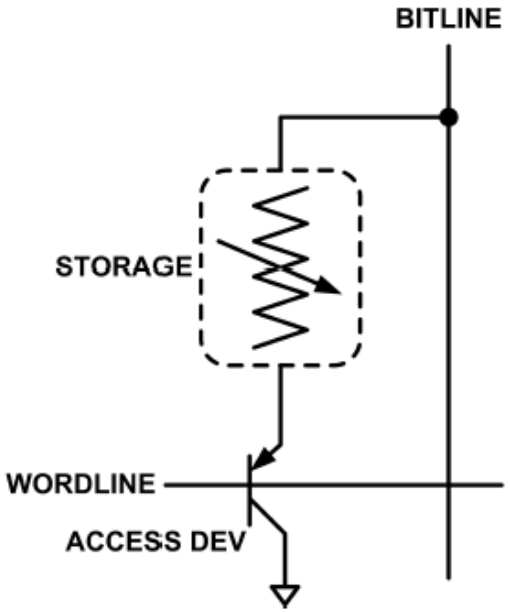
RRAM, ReRAM

Resistive Switching Random Access Memory

PCM: Phase Change Memory (I)

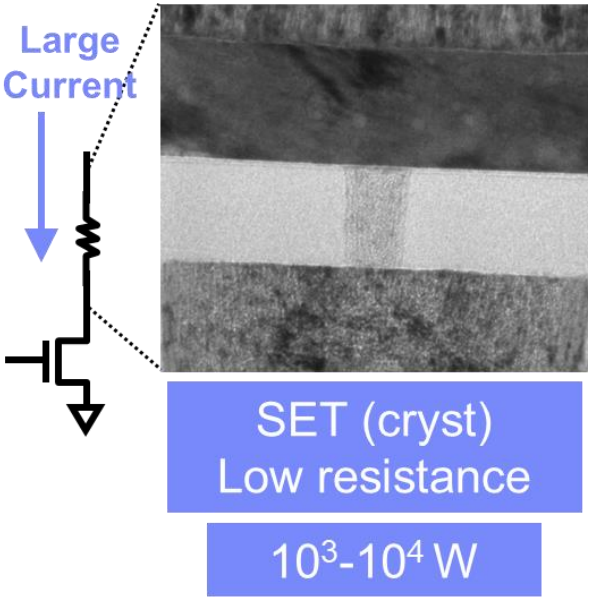
- **Inject current to change phase of chalcogenide glass by heat**
  - **Amorphous**: Low optical reflexivity and **high electrical resistivity**
  - **Crystalline**: High optical reflexivity and **low electrical resistivity**
- **PCM cell can be switched between states reliably and quickly**

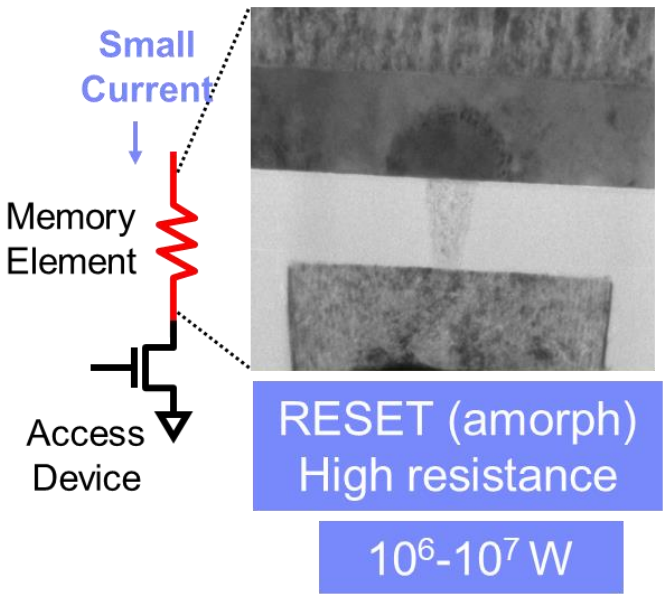


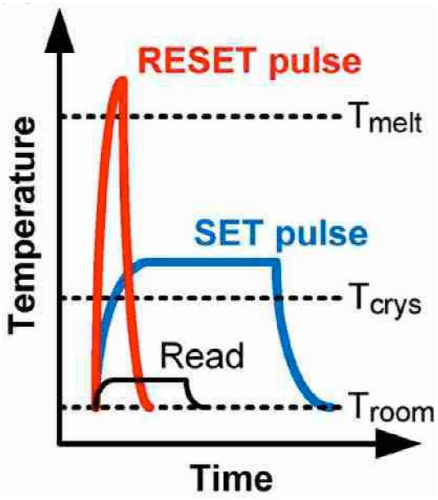


PCM: Phase Change Memory (II)

- **Write operation**
  - **SET**: sustained current to heat cell above **crystalline temperature** (T_{crys})
  - **RESET**: cell heated above **melt temperature** (T_{melt}) and quenched
- **Read operation**
  - Detect phase (amorphous/crystalline) via material resistance (high/low)



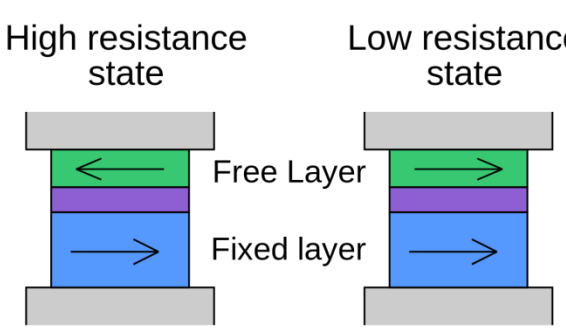


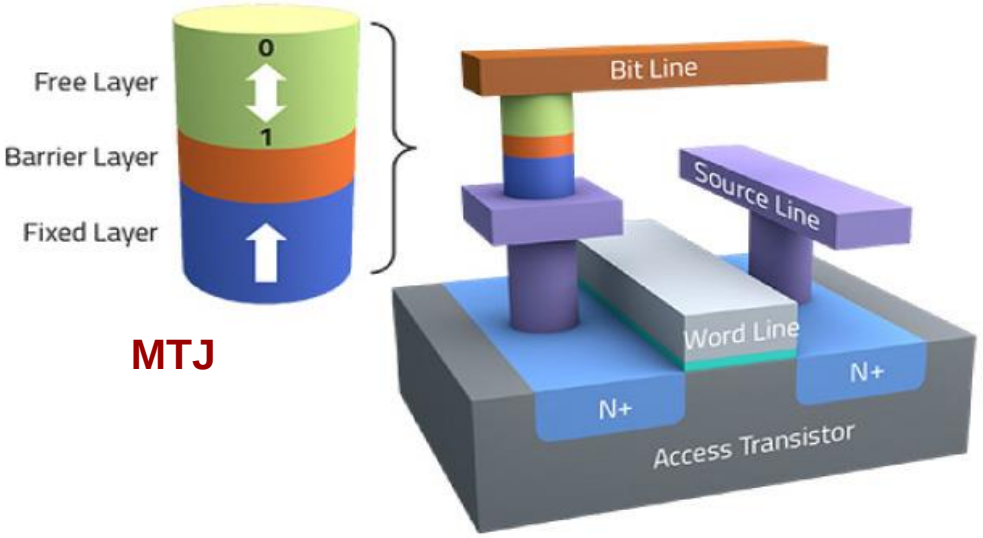


Phase Change Memory: From Devices to Systems, Morgan & Claypool Pub. 2011

STT-MRAM: Spin-Transfer Torque MRAM(I)

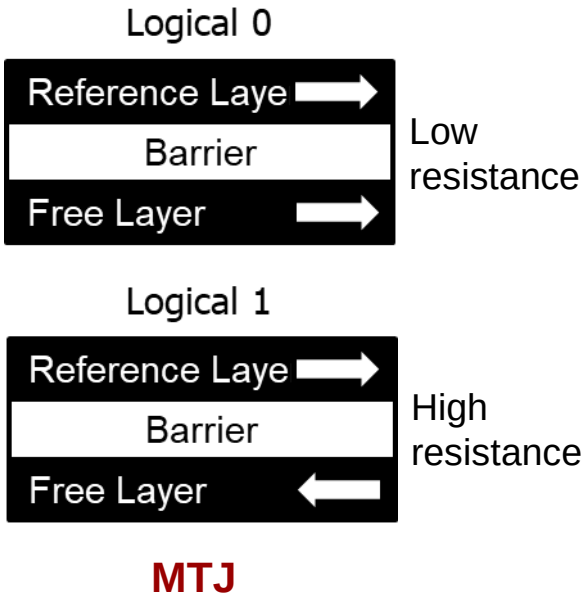
- **Inject current to change magnetic polarity**
- **Based on a MTJ (Magnetic Tunnel Junction) device**
  - **Reference (fixed) layer (RL)**: with a fixed magnetic orientation
  - **Free layer (FL)**: with a variable magnetic orientation
- **Magnetic orientation of the free layer determines logical state**
  - FL parallel to RL: MTJ with **low resistance**
  - FL anti-parallel to RL: MTJ with **high resistance**

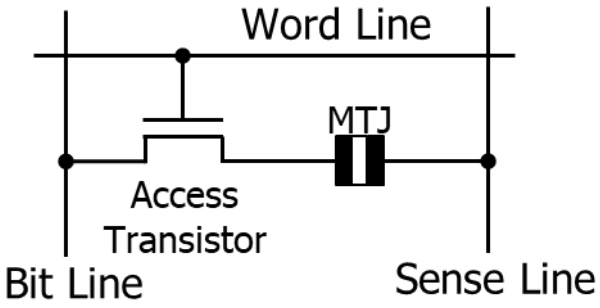




STT-MRAM: Spin-Transfer Torque MRAM(II)

- **Write operation**
  - Push large current through MTJ to change orientation of free layer
- **Read operation**
  - Sense current flow

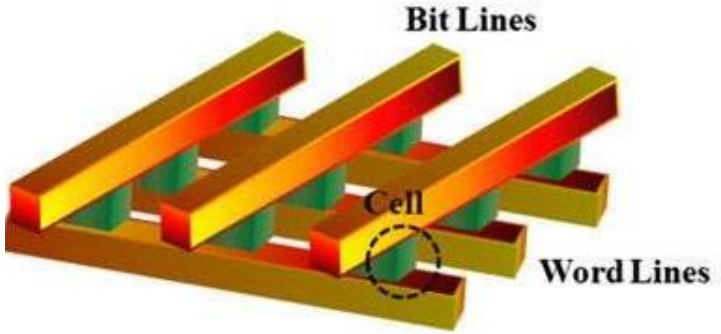


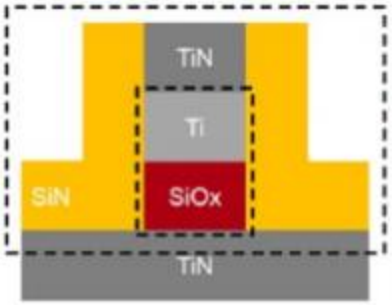


Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative, ISPASS 2013

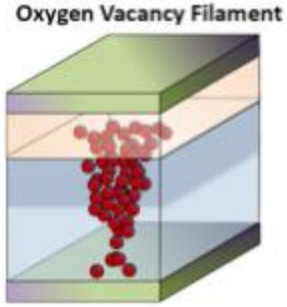
RRAM/ReRAM: Resistive RAM(I)

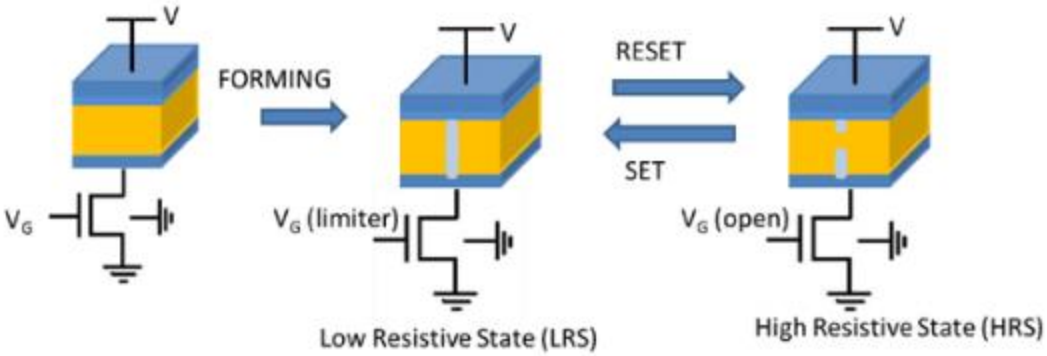
- **Inject current to change atomic structure**
- **Based on a memristor device**
  - **Memristor:** solid-state device with two resistive states (low/high resistance)





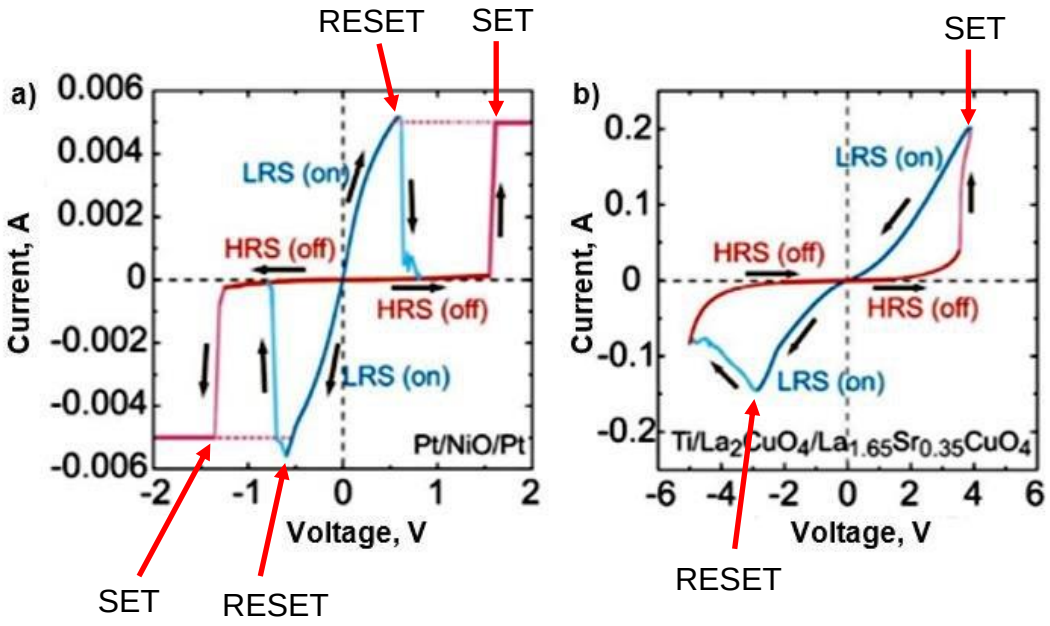
ReRAM memory





RRAM/ReRAM: Memristor (I)

- **Passive electronic element that change resistance depending on the electrical field applied**
  - **Unipolar memristor**: the polarity of the operating voltage **IS NOT** relevant
  - **Bipolar memristor**: the polarity of the operating voltage **IS** relevant



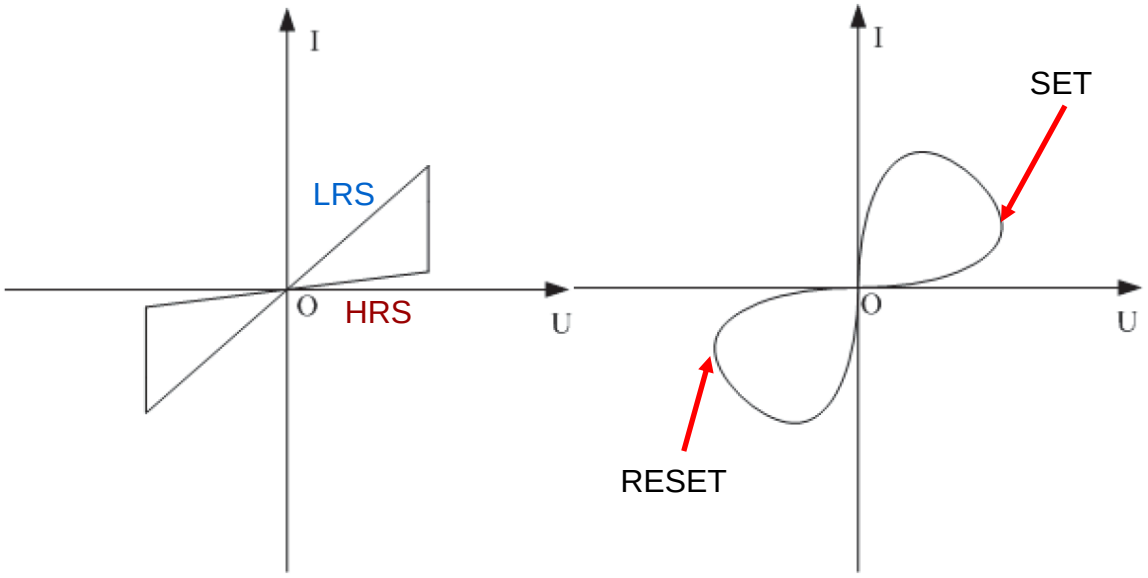
LRS: Low Resistance State

HRS: High Resistance State

Unipolar Memristor **Bipolar Memristor**

RRAM/ReRAM: Memristor (II)

- **Passive electronic element that change resistance depending on the electrical field applied**
  - **Digital memristor**: exhibits **two** neat resistance states: LRS and HRS
    - » Codifies 1 bit per memristor
  - **Analog memristor**: exhibits **multiple** resistance windows between LRS and HRS
    - » Codifies several bits (small number) per memristor



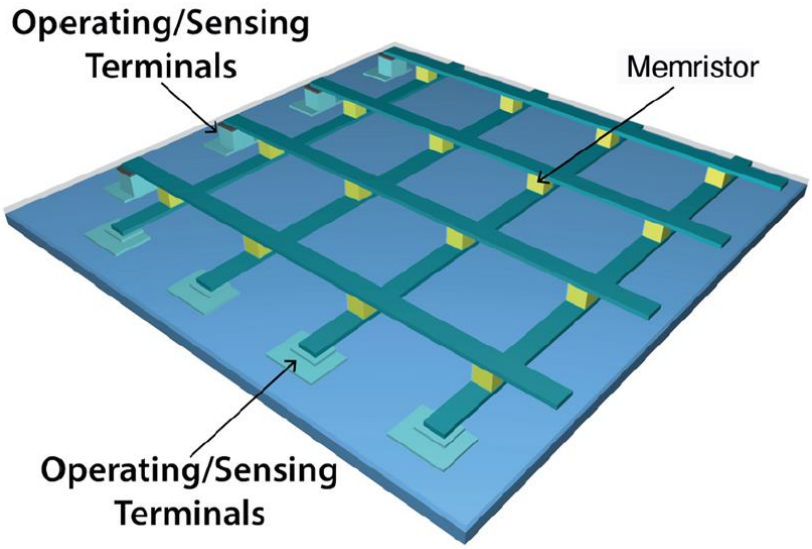
Digital Memristor **Analog Memristor**

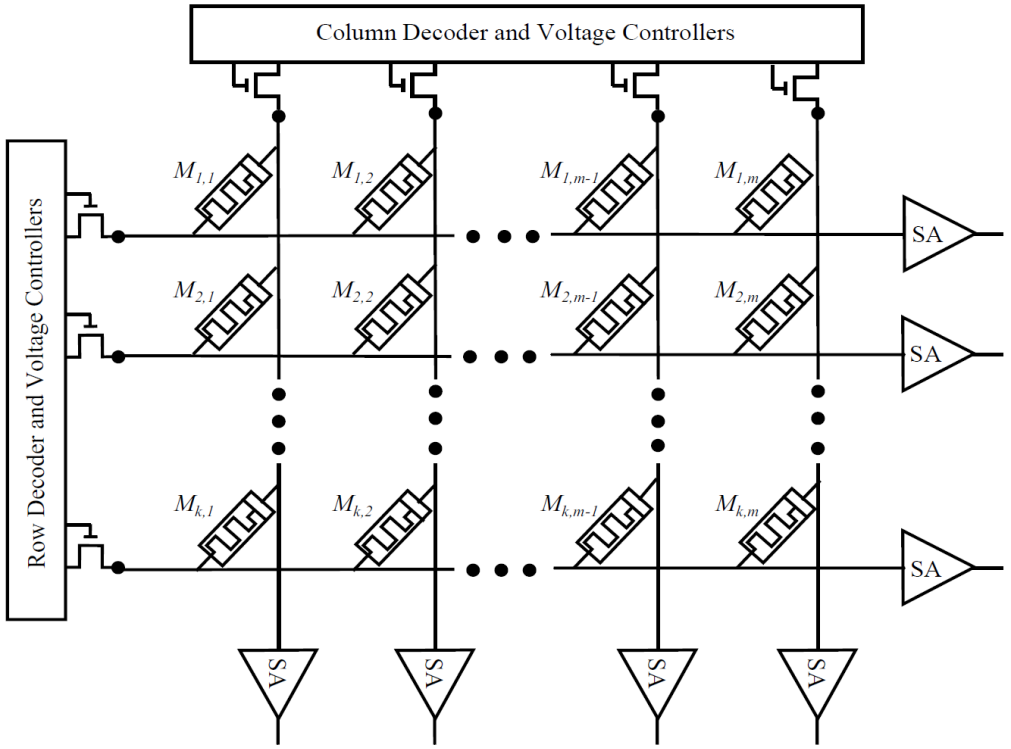
LRS: Low Resistance State

HRS: High Resistance State

RRAM/ReRAM: Memristor Crossbar

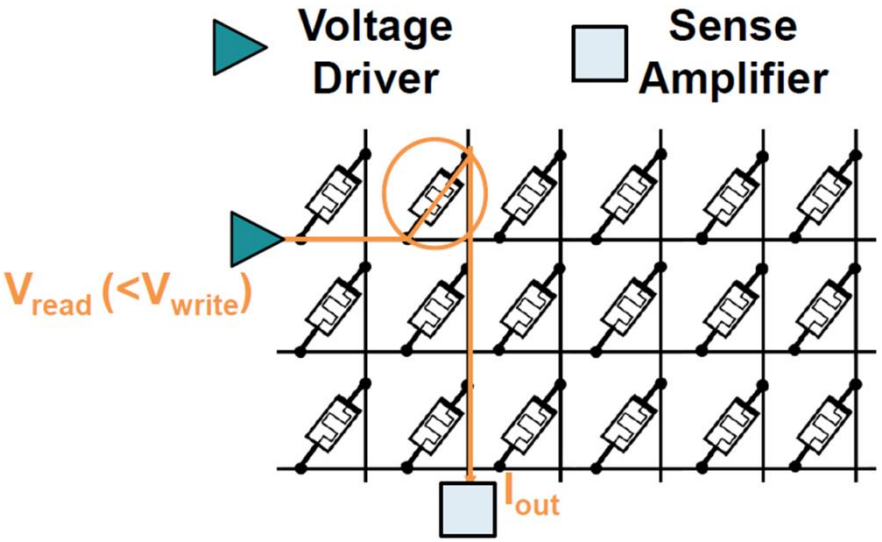
- **Memristor cells in a RRAM are usually arranged as a crossbar**

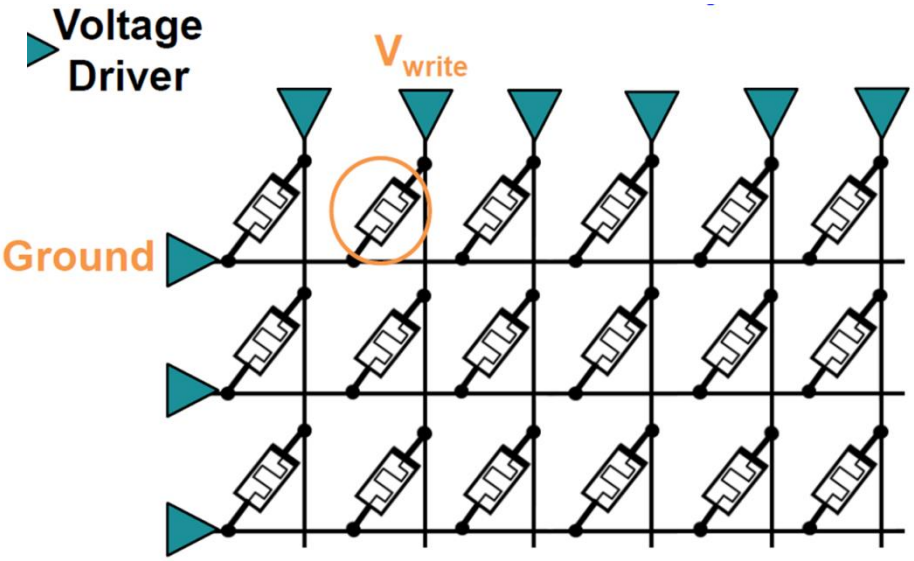




RRAM/ReRAM: Memristor Read/Write

- **Read/Write operations in a RRAM**





Data-Centric Computing

PIM
Processing-in-Memory

CIM
Computation-in-Memory

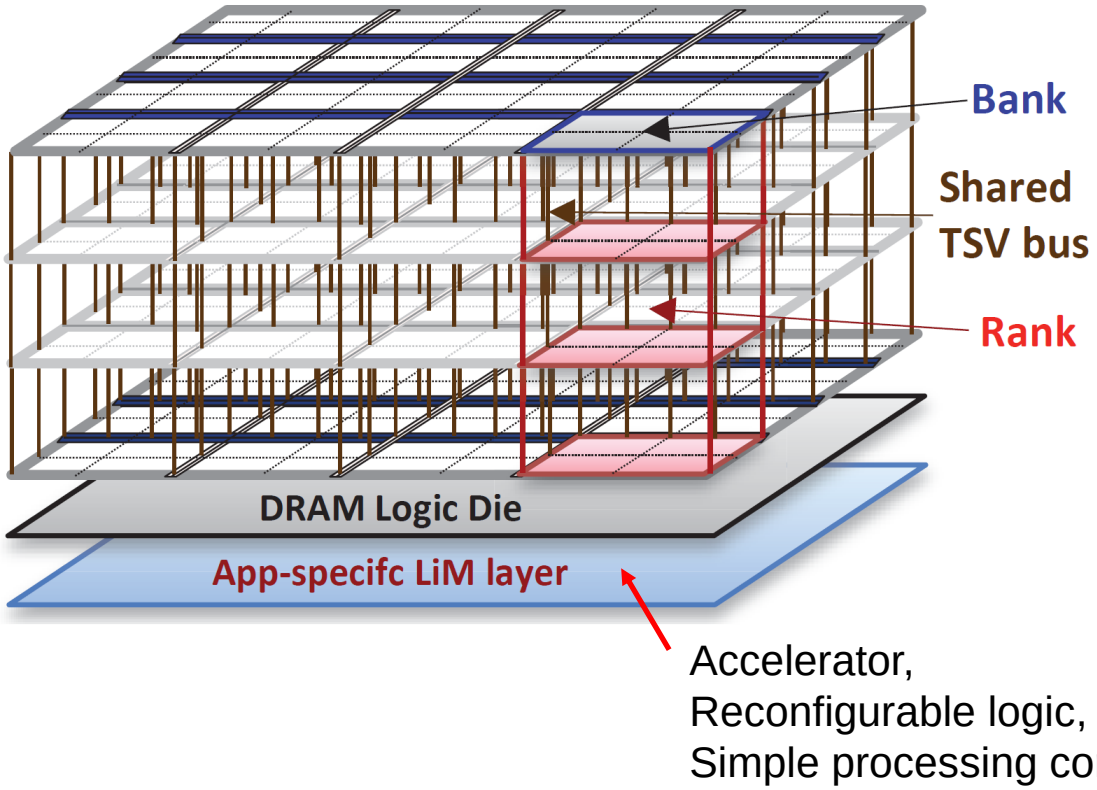
- **Memory technologies for PIM**
- **Approaches to implementing PIM**

Approaches to Implementing PIM

- **PNM: Processing Near Memory**
  - Logic layer in 3D-stacked memory
  - Function-specific logic connected to 3D-stacked memory (via logic interface)
  - Silicon interposers (connected directly to TSVs)
  - Logic in memory controllers
- **PIM/PUM: Processing In/Using Memory**
  - Logic inside SRAM
  - Logic inside DRAM
  - PCM with logic (in cells and/or periphery)
  - STT-MRAM with logic (in cells and/or periphery)
  - RRAM with logic (in cells and/or periphery)

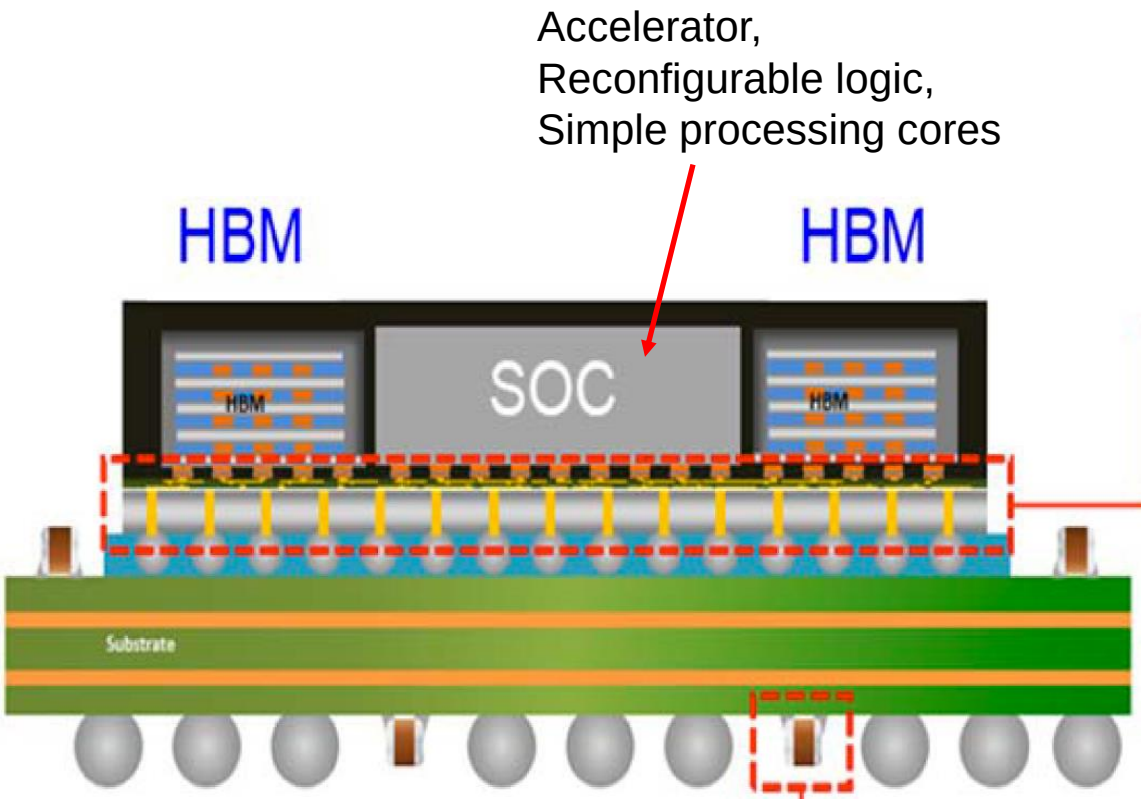
Processing Near Memory (PNM)

- **Logic layer in 3D-stacked memory**



Processing Near Memory (PNM)

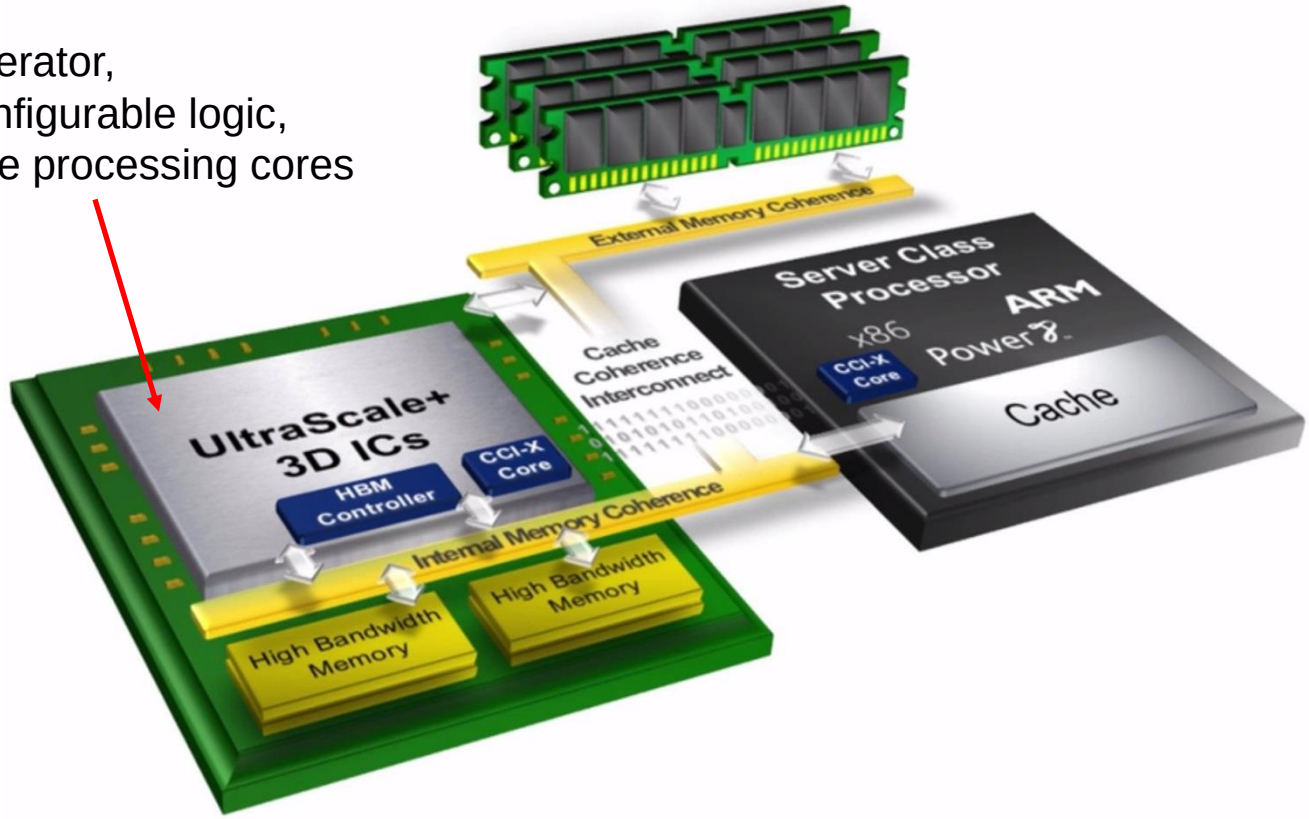
- **Silicon interposer**
  - SOC connected directly to the TSVs in HBM



Processing Near Memory (PNM)

- **Function-specific logic connected to 3D-stacked memory**
  - Processing logic connected HBM channel interface via an HBM controller

Accelerator,
Reconfigurable logic,
Simple processing cores



Approaches to Implementing PIM

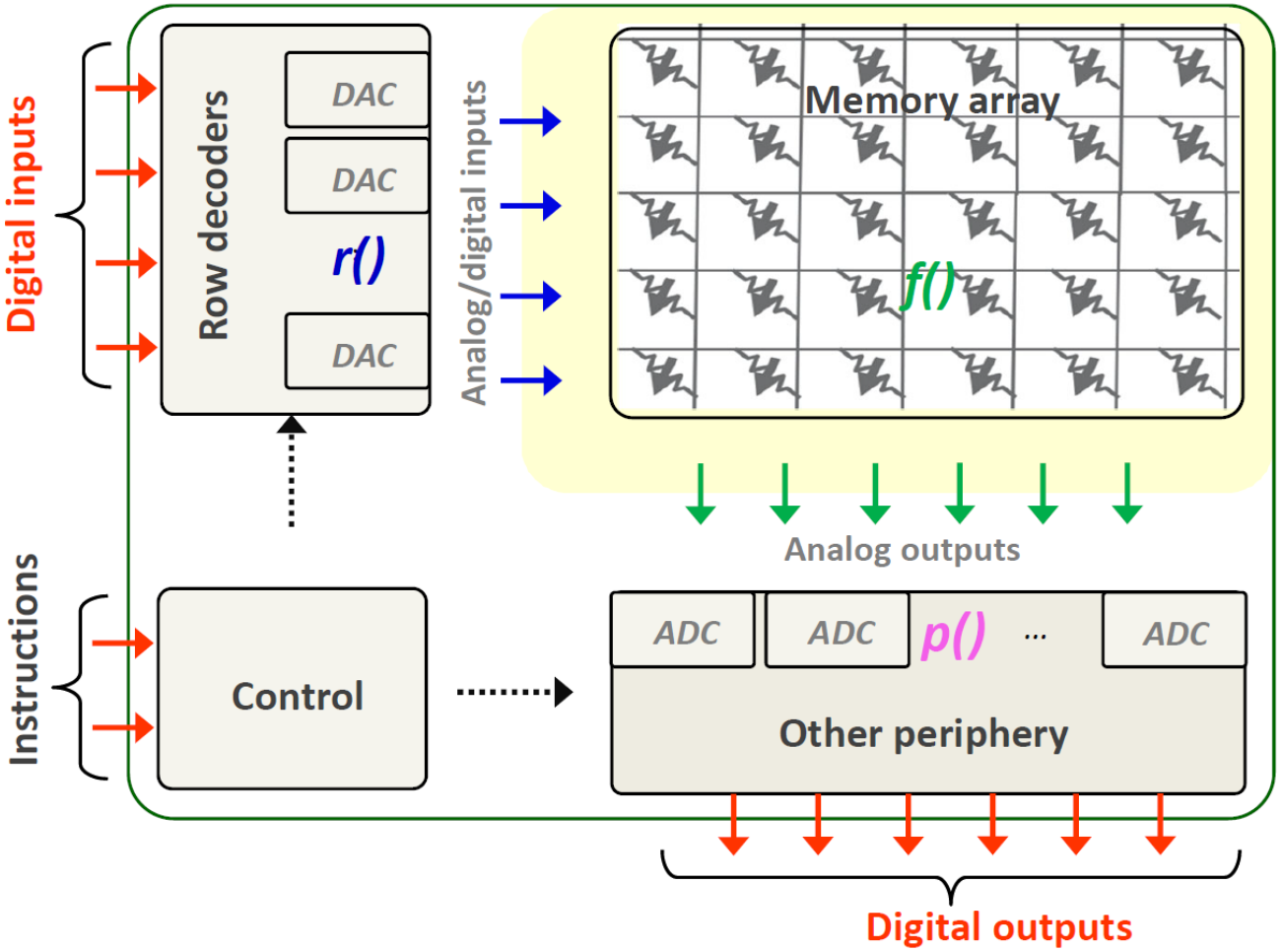
- **PNM: Processing Near Memory**
  - Logic layer in 3D-stacked memory
  - Function-specific logic connected to 3D-stacked memory (via logic interface)
  - Silicon interposers (connected directly to TSVs)
  - Logic in memory controllers
- **PIM/PUM: Processing In/Using Memory**
  - Logic inside DRAM
  - PCM with logic (in cells and/or periphery)
  - STT-MRAM with logic (in cells and/or periphery)
  - RRAM with logic (in cells and/or periphery)

Processing In/Using Memory (PIM/PUM)

- **Exploits the memory architecture to enable operations with minimal changes**
  - Makes use of memory cells or cell arrays to perform useful computation
- **A wide range of different functions are enabled**
  - Fully parallel bulk or fine-grained operations
    - » Data copy/initialization
    - » Bulk bitwise operations
    - » Simple arithmetic operations (addition, multiplication, comparison, majority…)

PIM/PUM on RRAM: Computation

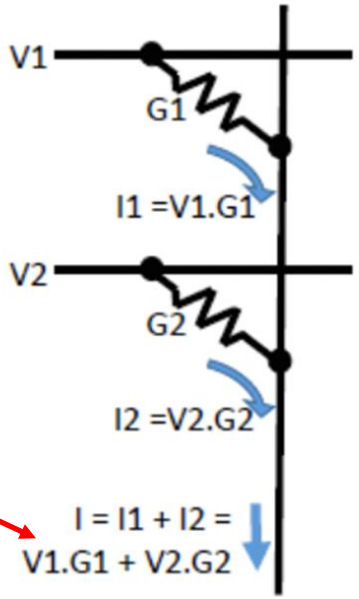
- **Single or multiple functions**



PIM/PUM on RRAM: Analog Computation

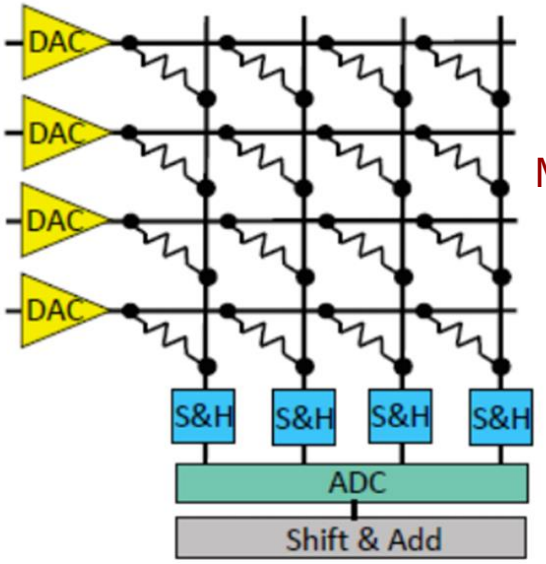
- **Analog memristor operations**
  - The analog nature of memristors is used to codify data as resistance value
- **Example: vector-matrix operations**

Scalar product of vectors V=(V1,V2) and G=(G1,G2)



Using a bitline to perform analog "sum of products" (Multiply-accumulate) operation

Vector

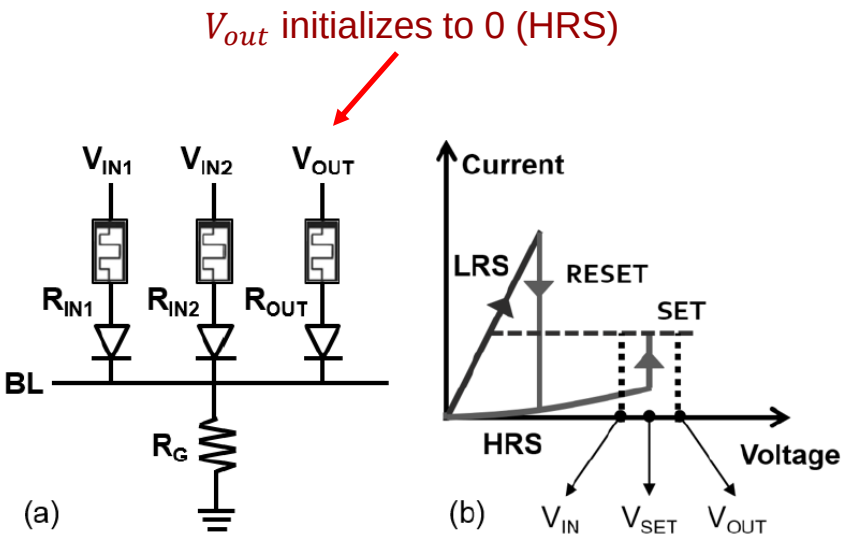


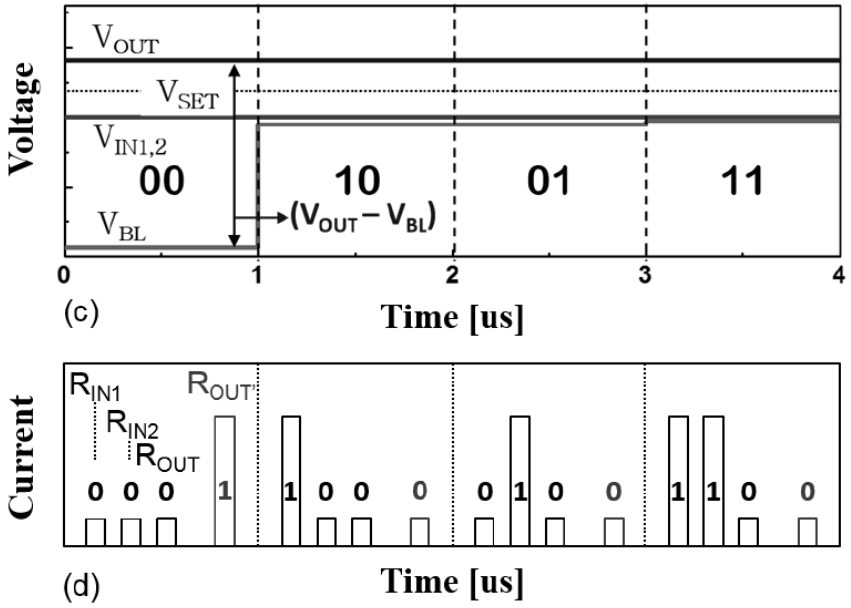
Matrix

Memristor crossbar for vector-matrix multiplier

PIM/PUM on RRAM: Digital Computation (I)

- **Digital memristor operations**
  - Data is codified as resistance values: LRS → 1, HRS → 0
- **Example: boolean NOR using unipolar memristors**

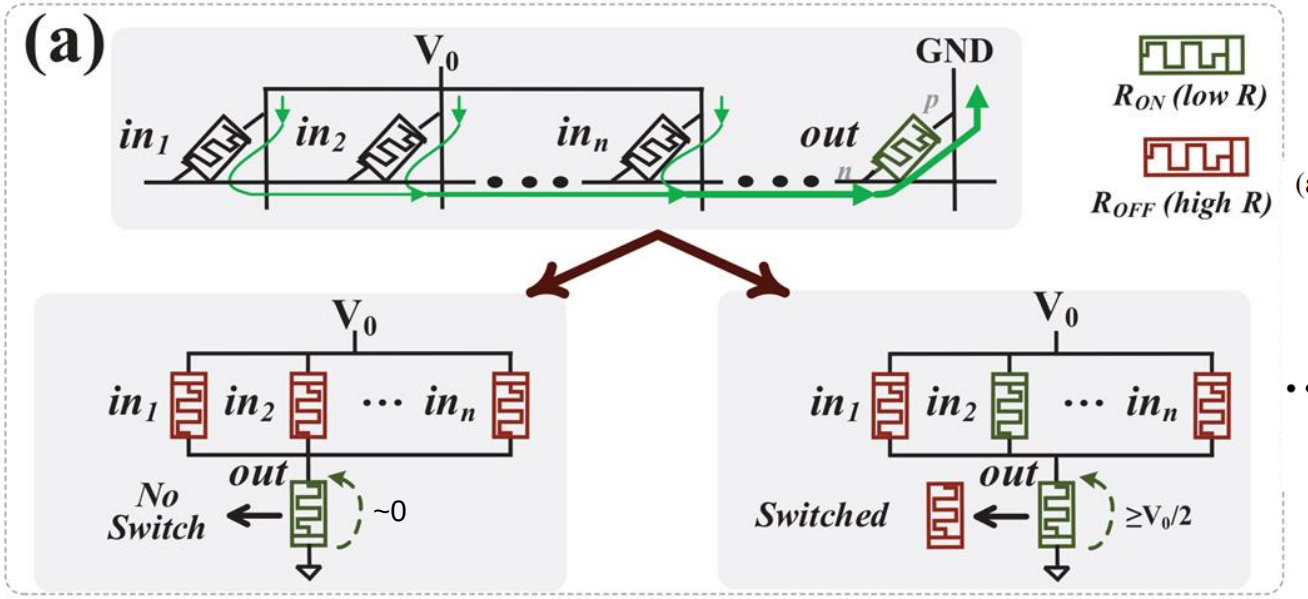


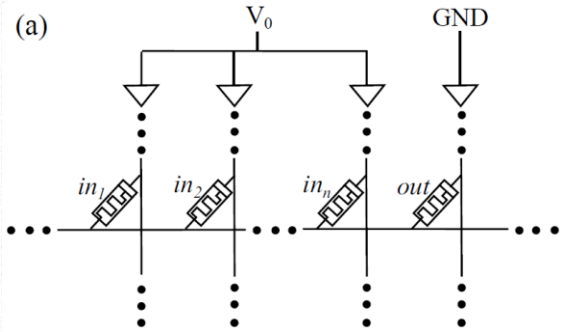


UPIM : Unipolar Switching Logic for High Density Processing-in-Memory Applications, GLSVLSI 2019

PIM/PUM on RRAM: Digital Computation (II)

- **Digital memristor operations**
  - Data is codified as resistance values: LRS → 1, HRS → 0
- **Example: boolean NOR using bipolar memristors**
  - Step 1: set 'out' to 1 (LRS)
  - Step 2: apply V_0 to 'in' and GND to 'out' (being $V_0/2 > V_{RESET}$)

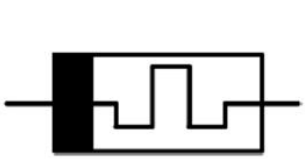




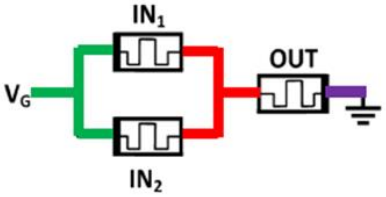
Logic Design Within Memristive Memories Using Memristor-Aided loGIC (MAGIC), IEEE Transactions on Nanotechnology 2016

PIM/PUM on RRAM: Digital Computation (III)

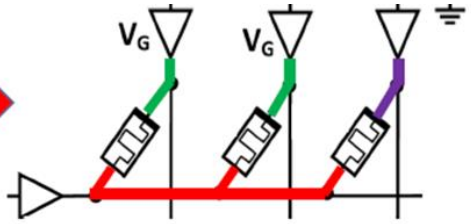
- **Processing in/using memory: SIMD parallelism**



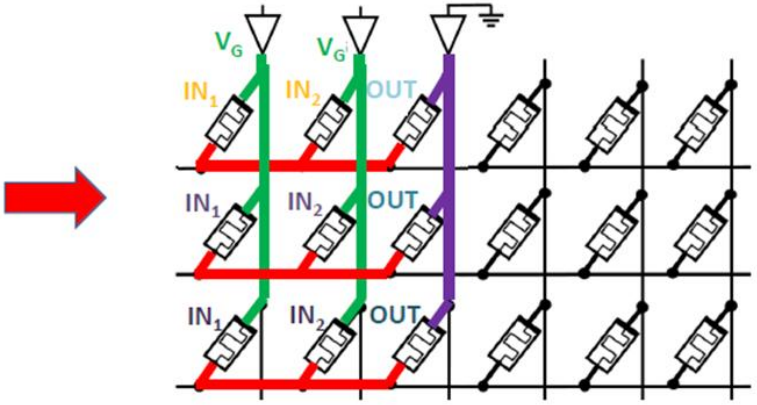
A memristor memory cell



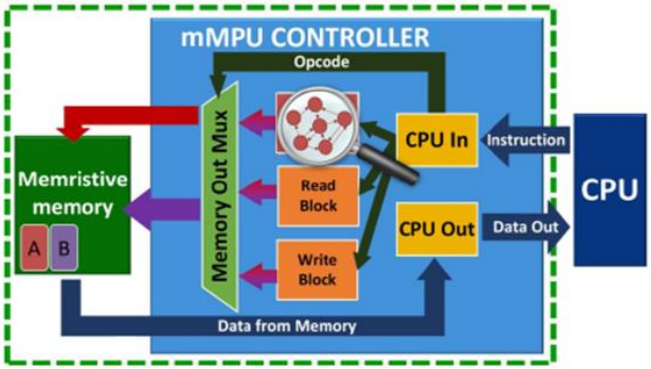
NOR logic Gate (MAGIC)



Crossbar Compatible



SIMD computing in memory

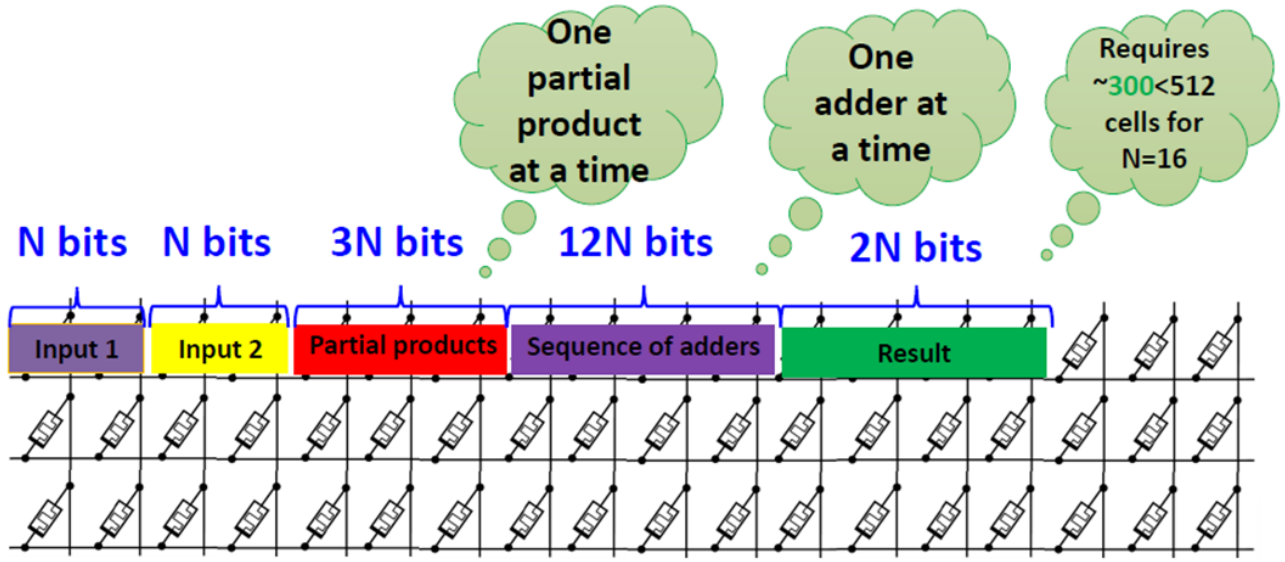


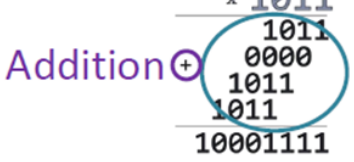
Control & Data

True Processing in Memory

PIM/PUM on RRAM: Digital Computation (IV)

- **Example: N-bit fixed-point multiplication**
  - Possible high latency but massive parallelism





- **Showcase PIM implementation of complex operation**
- N=16 → **~3700 NOR operations** ($O(N^2)$)
- Naïve column allocation: **~3700 Cells** ($O(N^2)$)
- Optimized manual/automictic allocation: **~300 Cells!** ($O(N)$)

Total*:
Latency: ~3700 steps
Area (max): ~3700 cells
Area (min): ~300 cells
Energy: ~3700 operations

* Per element, ignoring initialization

Data-Centric Computing

Own Research
PNM Solutions

- **NATSA: PNM accelerator for time series analysis**
- **FM-Index: PNM accelerator for genome sequence alignment**

NATSA: PNM Accelerator for Time Series Analysis

2020 IEEE 38th International Conference on Computer Design (ICCD)

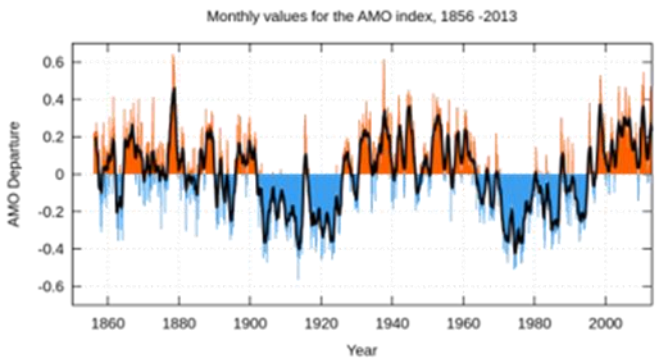
NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez[§] Ricardo Quislant[§] Christina Giannoula[†] Mohammed Alser[‡]
Juan Gómez-Luna[‡] Eladio Gutiérrez[§] Oscar Plata[§] Onur Mutlu[‡]

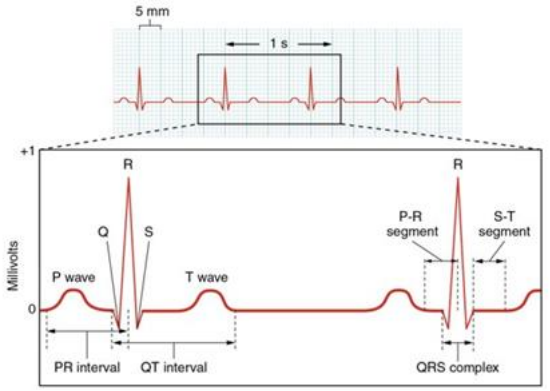
[§]*University of Malaga* [†]*National Technical University of Athens* [‡]*ETH Zürich*

NATSA: Motivation

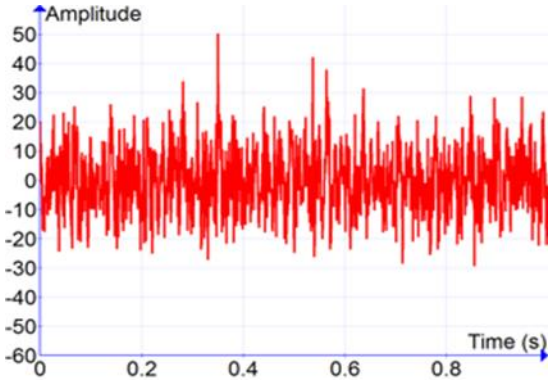
- **Time series analysis has many applications**



Climate change

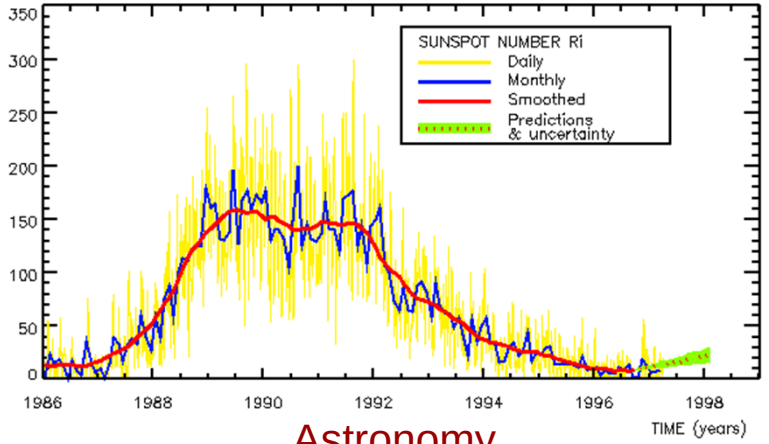


Medicine



Signal processing

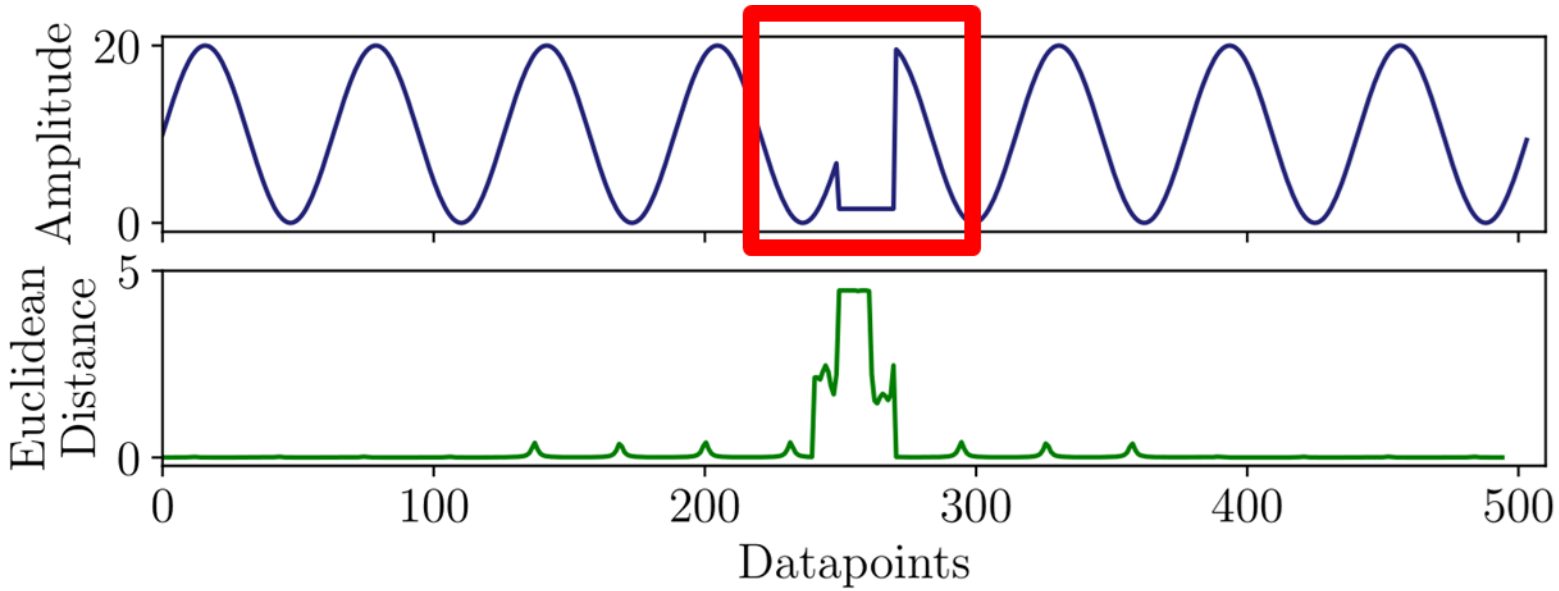
Economics



Astronomy

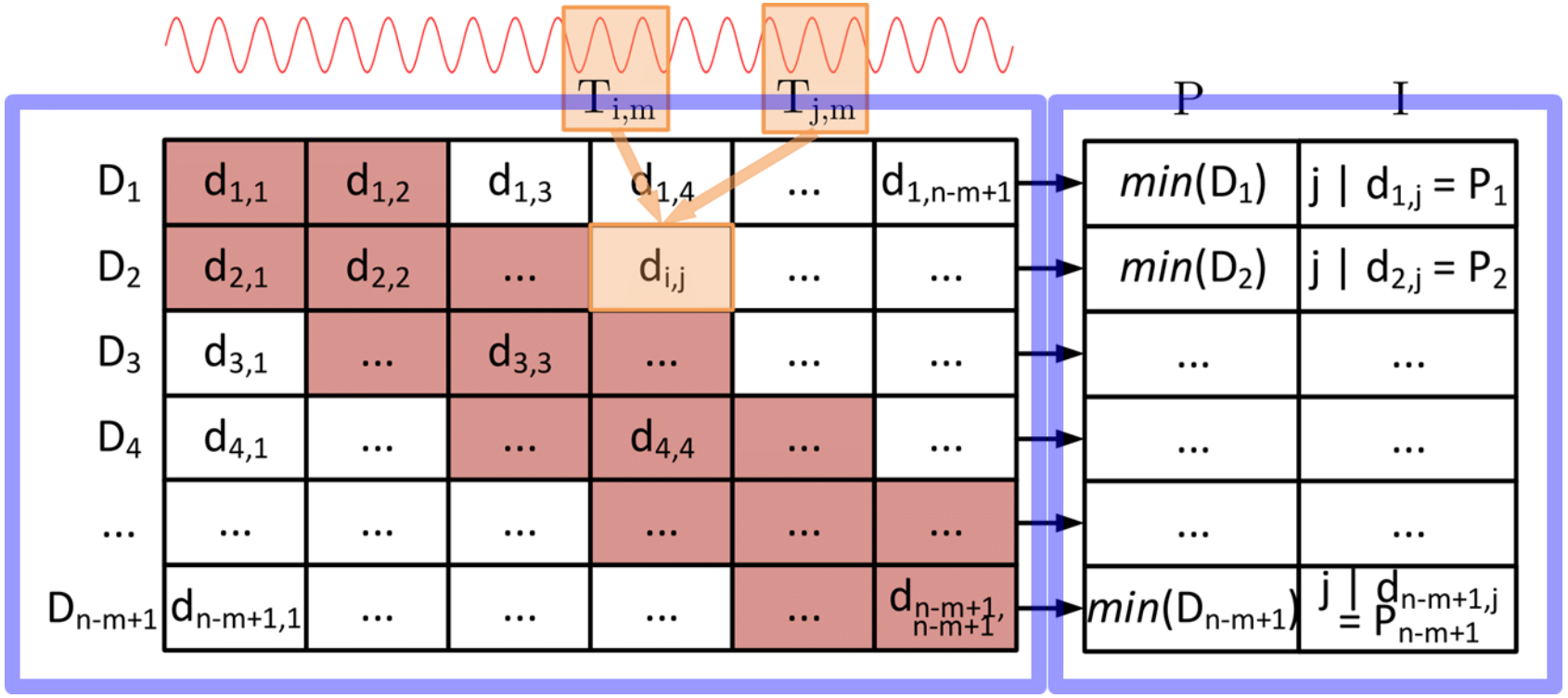
NATSA: Motifs and Discords

- **Given a sliced time series into subsequences**
  - **Motif** discovery focuses on finding **similarities**
  - **Discord** discovery focuses on finding **anomalies**
- **Naive example of anomaly detection**



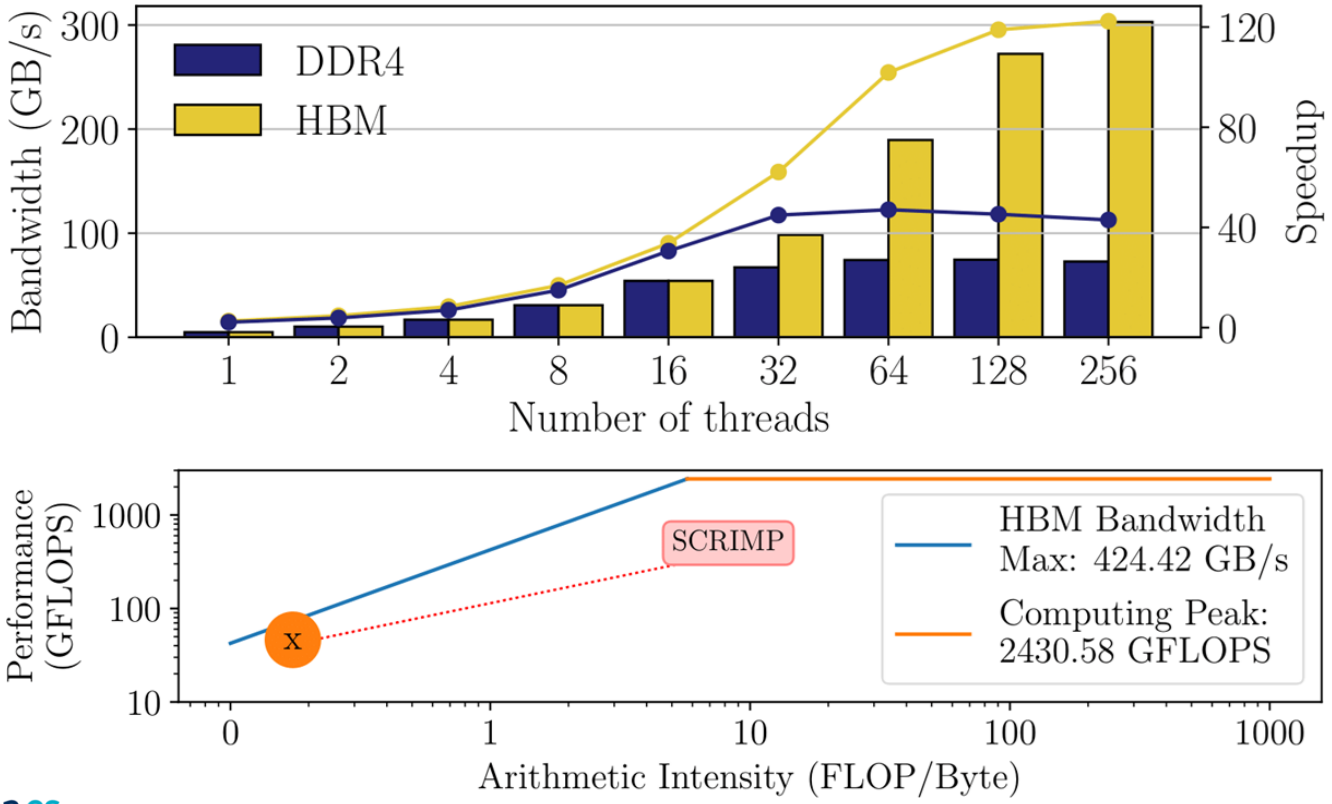
NATSA: Matrix Profile (MP)

- **MP: Algorithm and open source tool for motif and anomaly discovery**
- **A single input parameter: subsequence length**



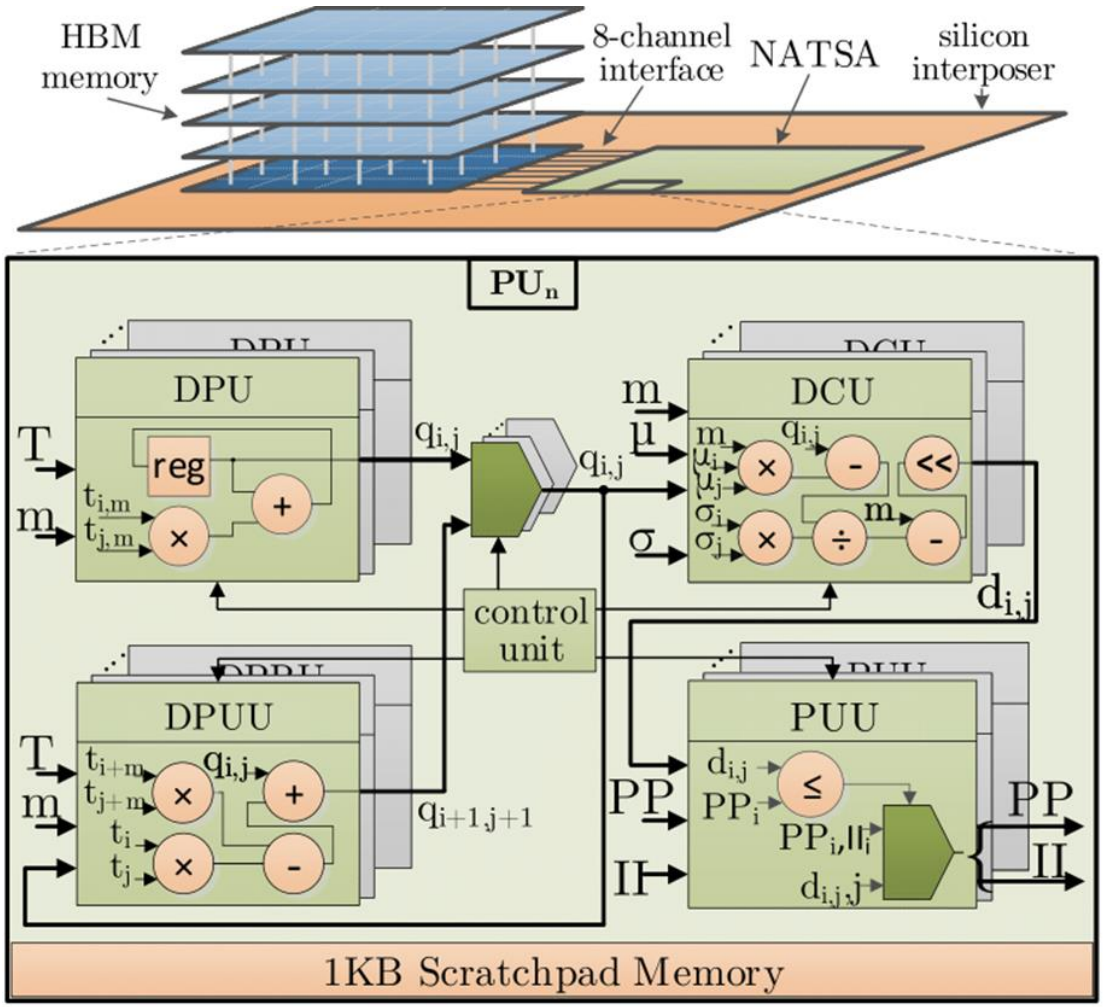
NATSA: Matrix Profile (MP) - SCRIMP

- **SCRIMP: state-of-the-art CPU matrix profile implementation**
- **SCRIMP on Intel Xeon Phi KNL**
  - **SCRIMP is heavily bottlenecked by data movement**



NATSA: PNM Accelerator for SCRIMP

- **NATSA: fully exploit HBM memory bandwidth**
  - NATSA consists of multiple processing units (PUs)
  - PUs compute batches of diagonals of the distance matrix in a vectorized way
- **Hardware components**
  - **DPU**: dot product unit
  - **DCU**: distance compute unit
  - **DPUU**: dot product update unit
  - **PUU**: profile update unit



NATSA: Evaluation

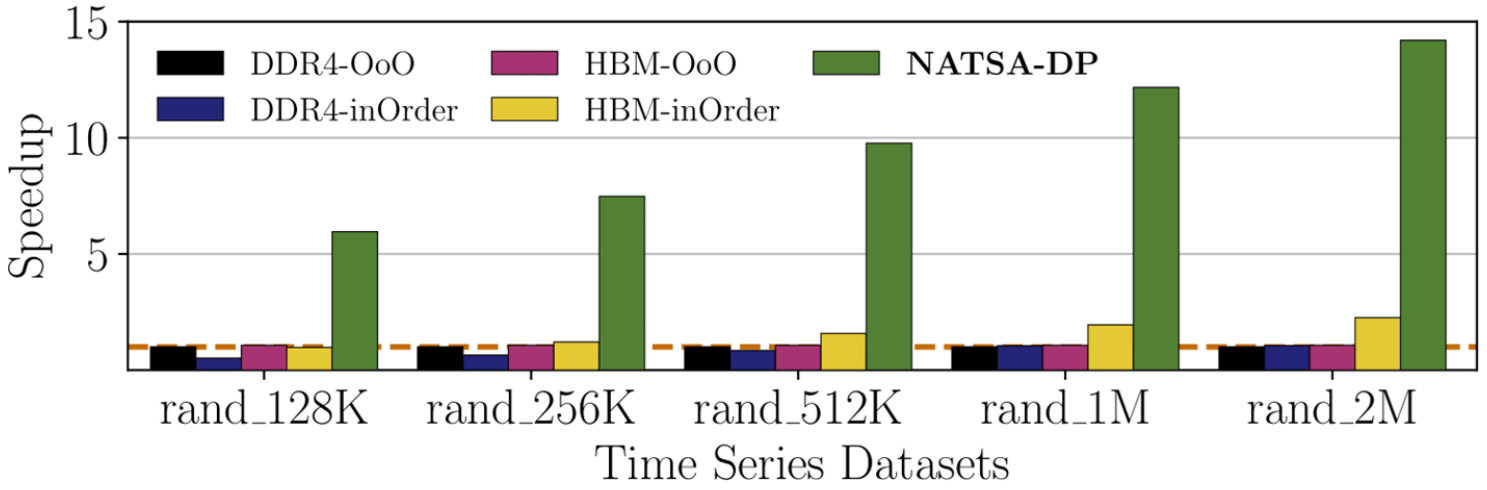
- **Simulated platforms**
  - ZSIM + Ramulator, McPAT, Aladdin + gem5, Micron power calculator

Hardware	Cores / PUs	Caches (L1 / L2 / L3)	Memory
DDR4-OoO	8 OoO @ 3.75 GHz	32KB / 256KB / 8MB	16 GB DDR4-2400
DDR4-inOrder	64 in-order @ 2.5 GHz	32KB / - / -	16 GB DDR4-2400
HBM-OoO	8 OoO @ 3.75 GHz	32KB / 256KB / 8MB	4 GB HBM2
HBM-inOrder	64 in-order @ 2.5 GHz	32KB / - / -	4 GB HBM2
NATSA	48 PUs @ 1 GHz	48KB (Scratchpad)	4 GB HBM2

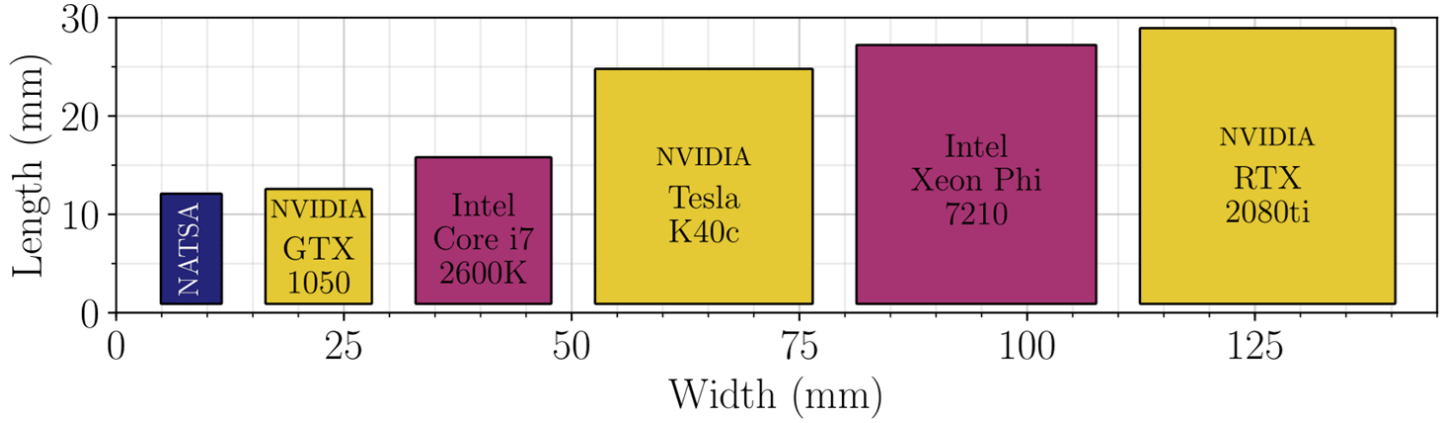
- **Real hardware platforms**
  - Intel Xeon Phi KNL, NVIDIA Tesla K40c, NVIDIA GTX 1050

NATSA: Evaluation: Performance and Area

- **Performance**

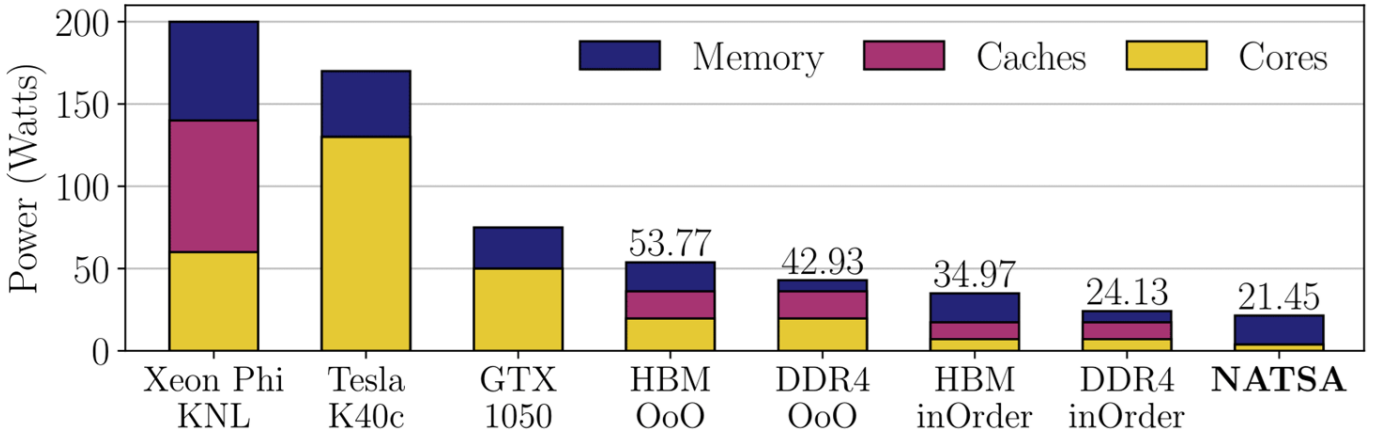


- **Area**

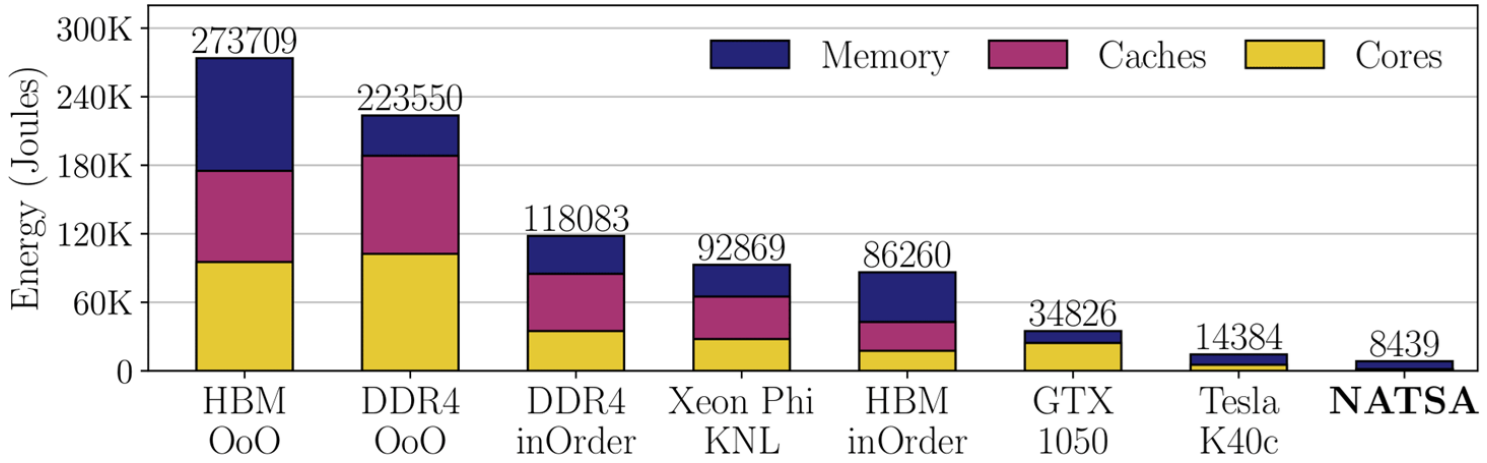


NATSA: Evaluation: Performance and Area

- **Power consumption**



- **Energy consumption**

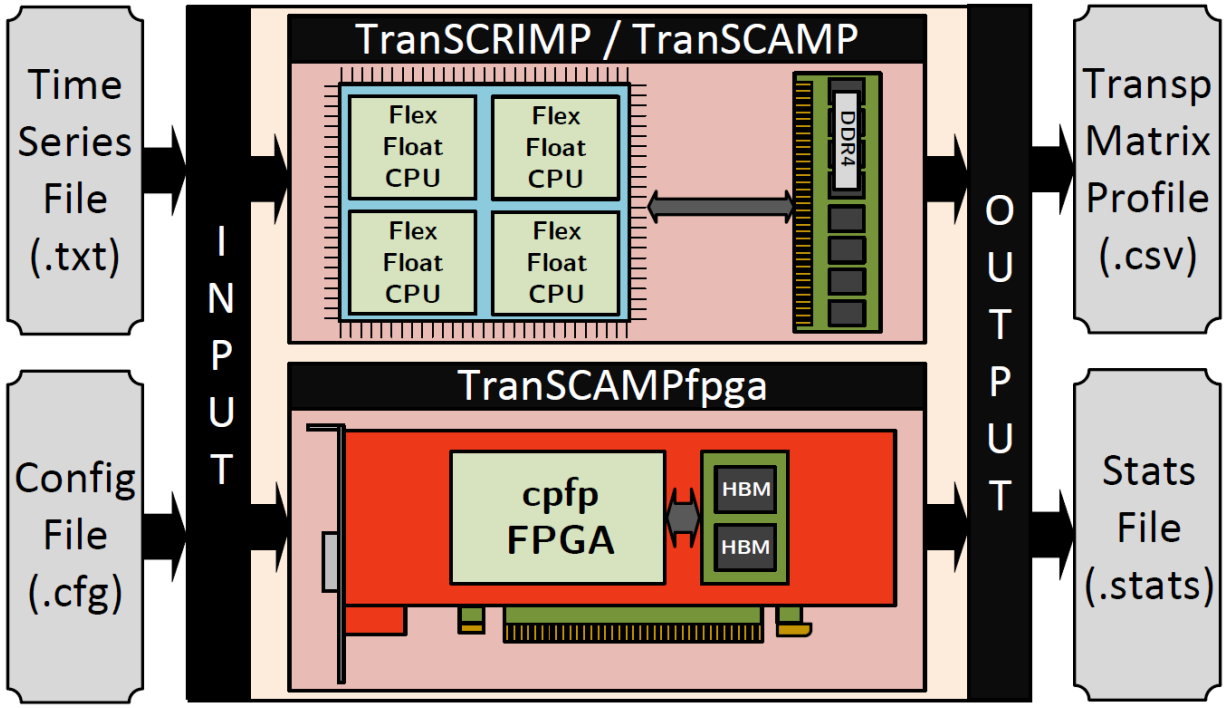


TraTSA: Transprecise Accelerator for Time Series Analysis

IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. ?, NO. ?, MONTH YEAR

TraTSA: A Transprecision Framework for Efficient Time Series Analysis

Ivan Fernandez, Ricardo Quislant, Sonia González-Navarro, Eladio Gutierrez and Oscar Plata



FM-Index: PNM Accelerator for Genome Sequence Alignment

IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, VOL. 17, NO. 4, JULY/AUGUST 2020

Accelerating Sequence Alignments Based on FM-Index Using the Intel KNL Processor

Jose M. Herruzo, Sonia González-Navarro, Pablo Ibáñez-Marín, Víctor Viñals-Yúfera, Jesús Alastruey-Benedé, and Oscar Plata

The Journal of Supercomputing
https://doi.org/10.1007/s11227-021-03661-3

Enabling fast and energy-efficient FM-index exact matching using processing-near-memory

Jose M. Herruzo[1] · Ivan Fernandez[1] · Sonia González-Navarro [1] · Oscar Plata[1]

FM-Index: Motivation

- **Genome sequence alignment is very relevant in bioinformatics**

- **There is a variety of sequence alignment tools**
  - Bowtie, Bowtie2, BWA, HiSAT2, SOAP, CUS…
- **Modern sequencing tools generate a lot of data**
- **Many sequence aligners are based on the FM-index data structure**

FM-Index: Data Structure

- **FM-index data structure**

- Data structures
  - C array
  - Occ table
- Main operation: Rank query

$$LF(Q[i], u) = C[Q[i]] + Occ[Q[i], u]$$

c	$	A	C	G	T
C	0	1	3	4	5

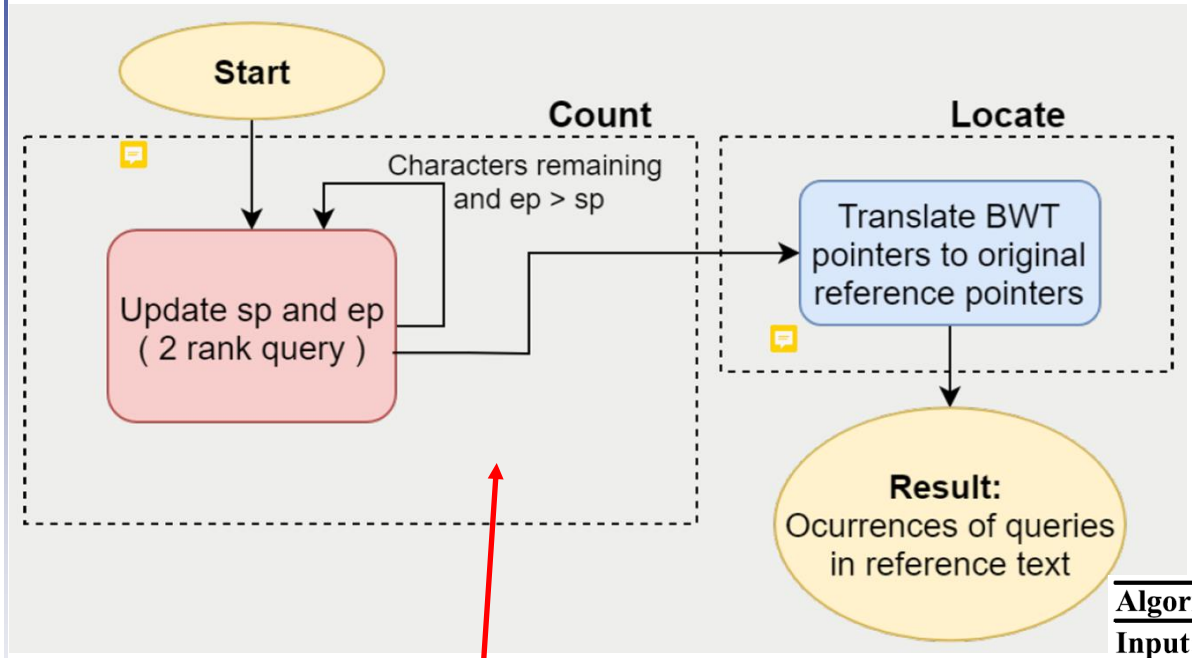
C Table

	T	G	T	A	T	C	$	A
	1	2	3	4	5	6	7	8
$	0	0	0	0	0	0	1	1
A	0	0	0	1	1	1	1	2
C	0	0	0	0	0	1	1	1
G	0	1	1	1	1	1	1	1
T	1	1	2	2	3	3	3	3

Occ Table

FM-Index: Algorithm

- **FM-index exact matching algorithm (backward search)**



Most time-consuming part
Random memory access pattern
Memory bound

Algorithm BS: Backward Search Based on FM-index

Input: FM-index of T (C & Occ), Q query, n=|T|, p=|Q|

Ouput: (sp,ep): Interval pointers of Q in T

begin

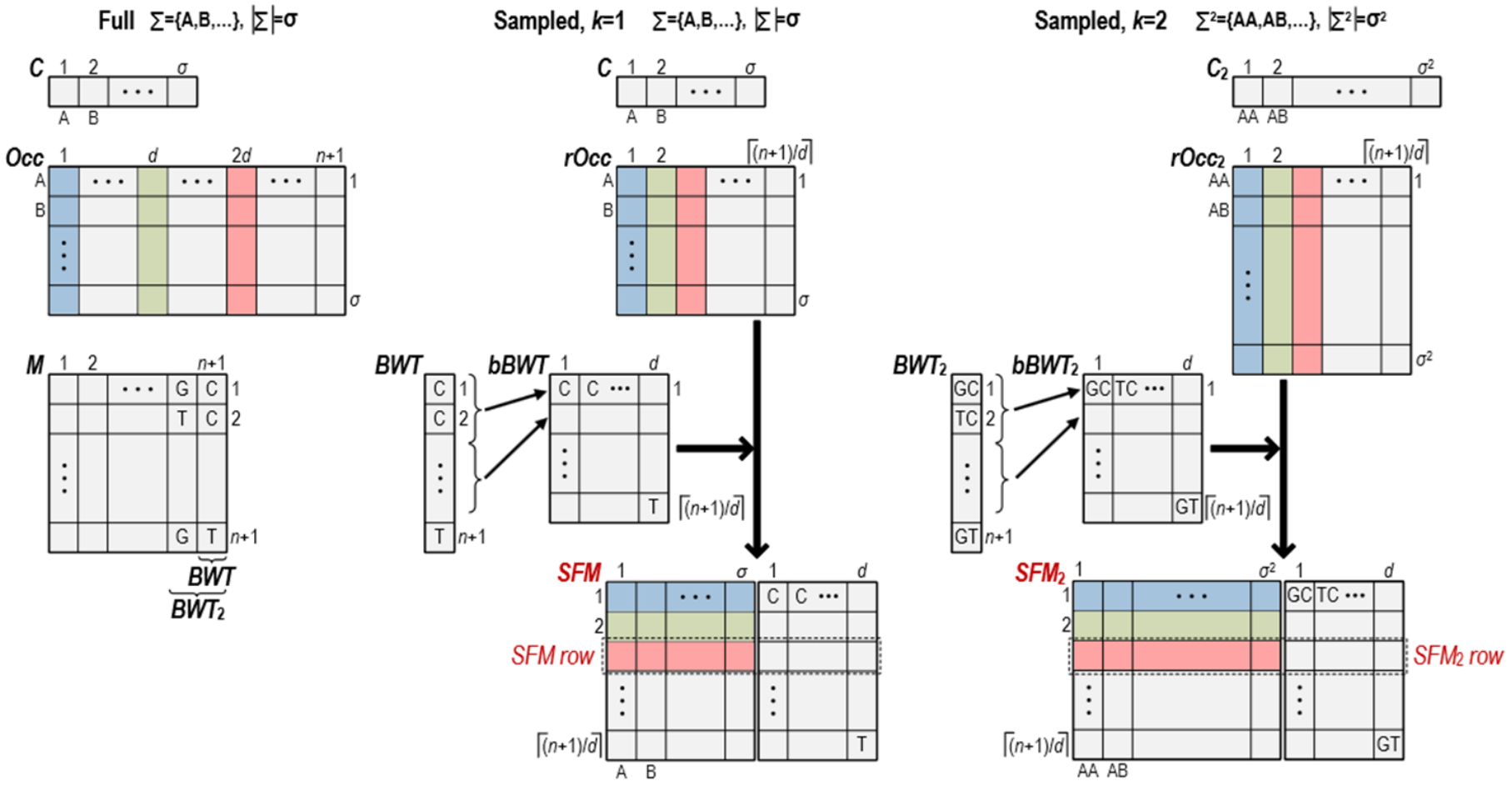
1: $sp = C[Q[p]]$
2: $ep = C[Q[p]+1]$
3: **for** i **from** p-1 **to** 1 **step** -1
4: $sp = LF(Q[i],sp)$
5: $ep = LF(Q[i],ep)$
6: **end for**
7: **return** (sp+1,ep)

end

} 2 LFM-chains

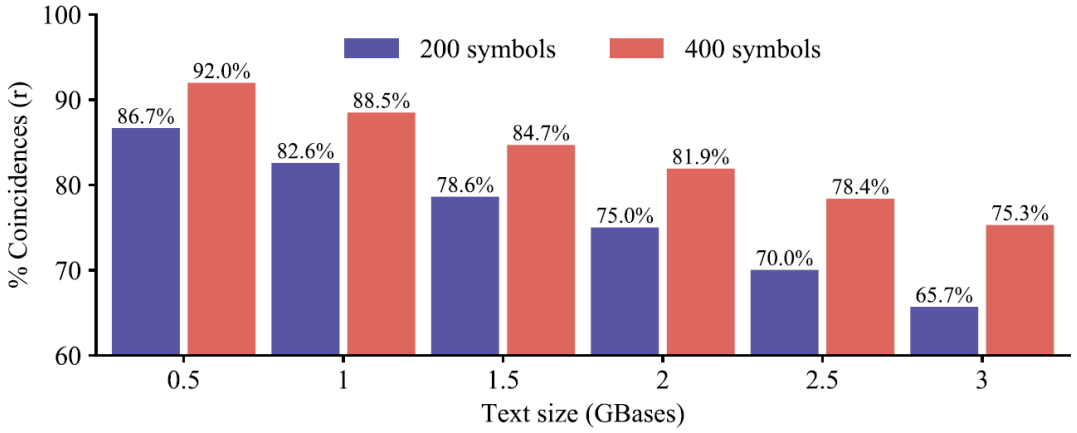
FM-Index: Algorithm Optimizations

- **FM-index exact matching algorithm (backward search)**



FM-Index: Split Bit-Vector Sampled

- **Minimizing memory bandwidth consumption**



Algorithm BS: Backward Search Based on FM-index

Input: FM-index of T (C & Occ), Q query, n=|T|, p=|Q|

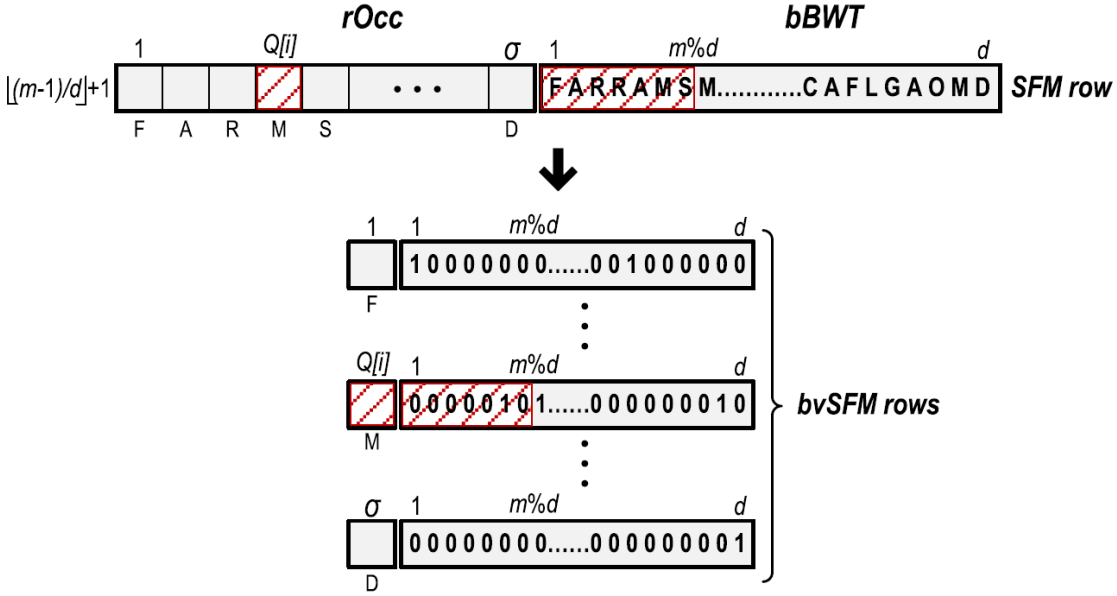
Ouput: (sp,ep): Interval pointers of Q in T

begin

1: $sp = C[Q[p]]$
2: $ep = C[Q[p]+1]$
3: **for** i **from** p-1 **to** 1 **step** -1
4: $sp = LF(Q[i],sp)$
5: $ep = LF(Q[i],ep)$
6: **end for**
7: **return** (sp+1,ep)

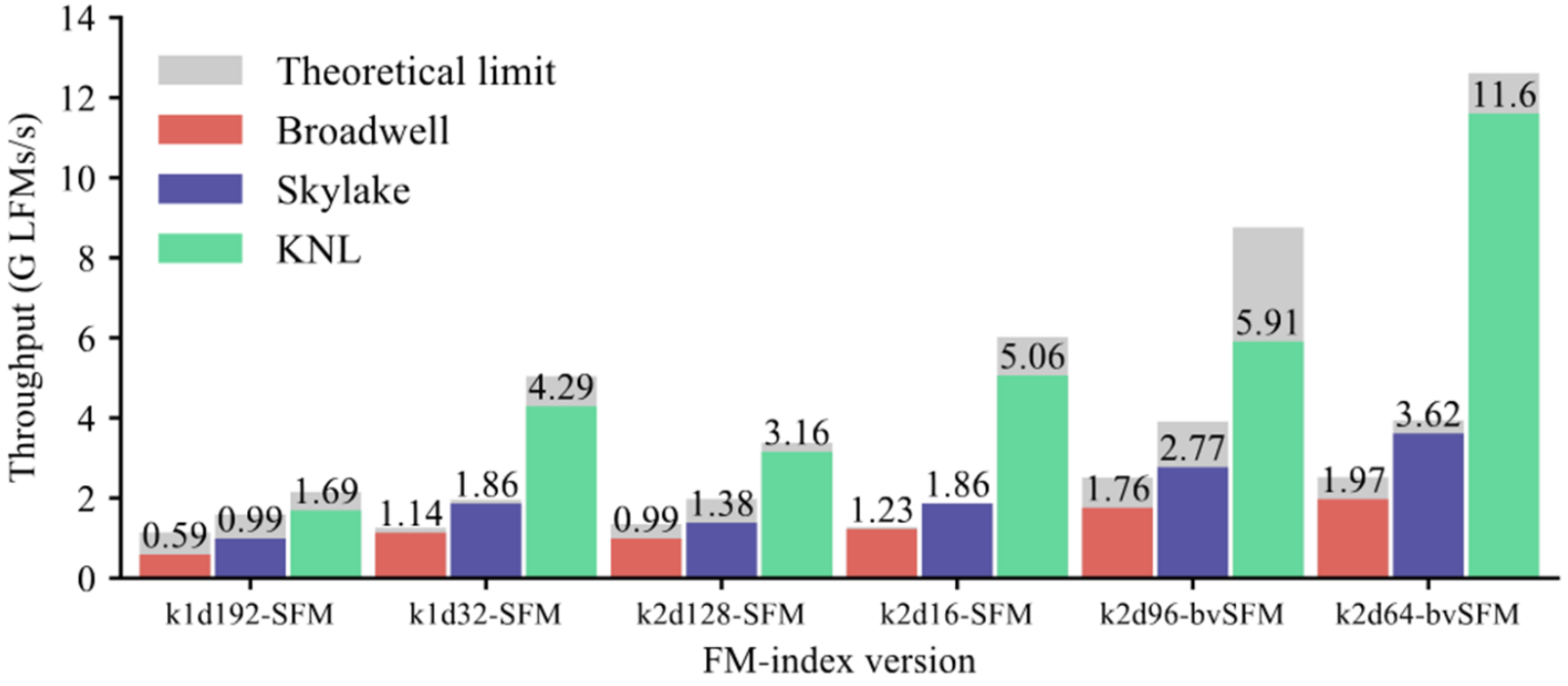
end

} 2 LFM-chains



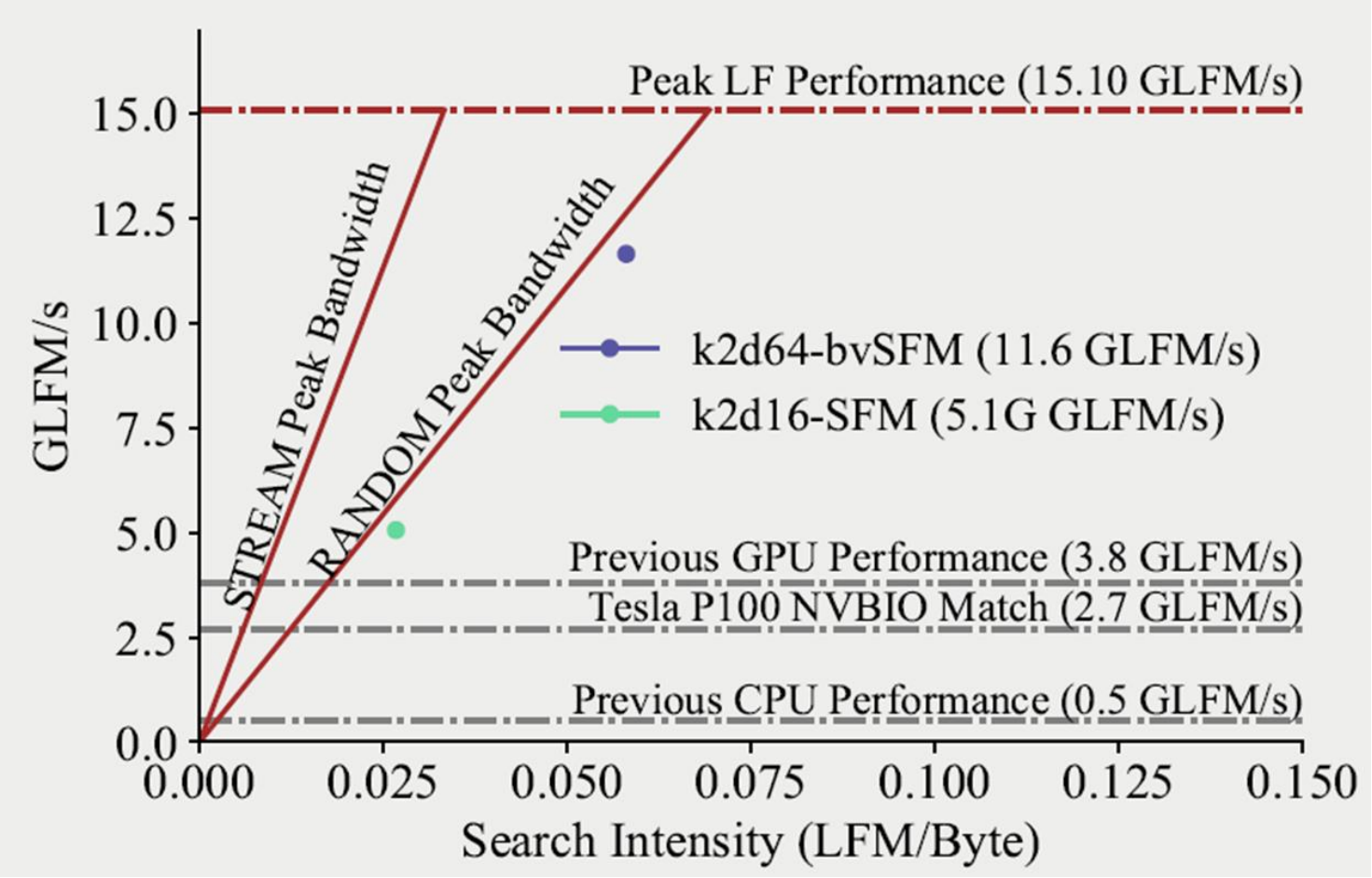
FM-Index: Evaluation

- **Experimental setup**
  - 20M 200-symbol queries generated by Mason simulation tool
  - GRCh38 (3 Gbases) human genome reference



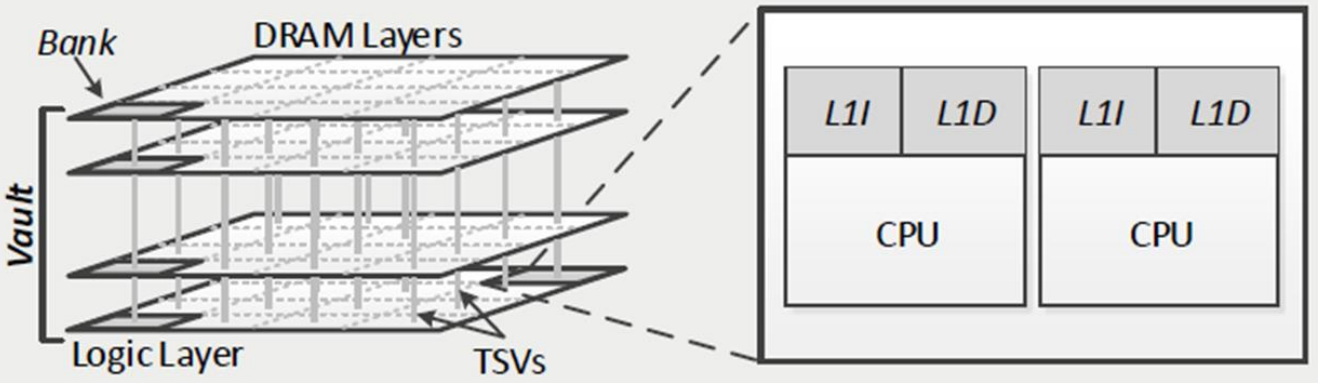
FM-Index: Evaluation

- **KNL roofline**



FM-Index: PNM Accelerator

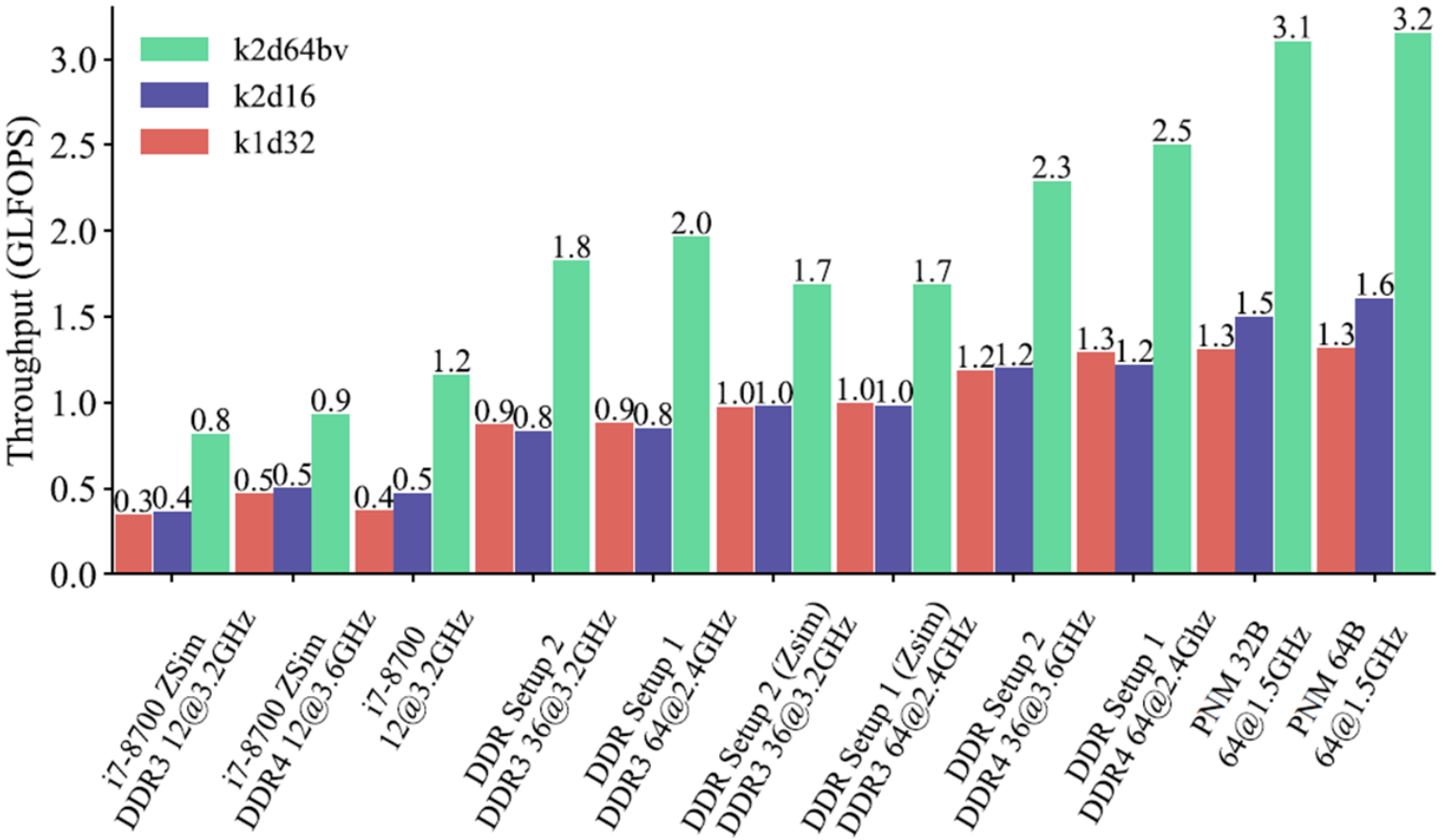
- **Logic layer in 3D-stacked memory (HMC)**



	PIM Setup
Cores	64 In Order @ 1.5 GHz
Architecture	ARM-Like
Cache block	32/64B
L1 Cache	8K/8K, 3 cycles Latency

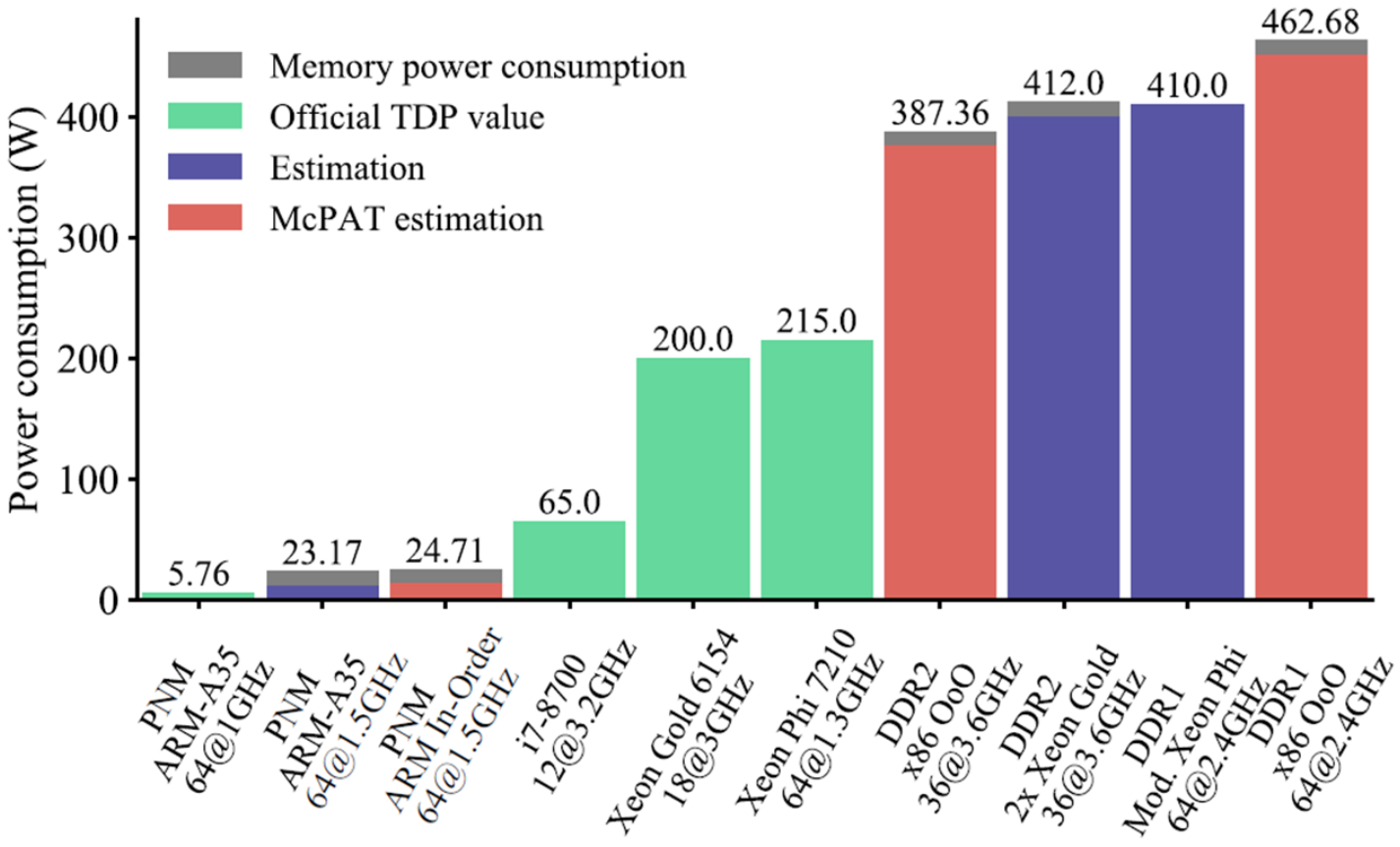
FM-Index: PNM Accelerator

- **Performance**



FM-Index: PNM Accelerator

- **Power consumption**

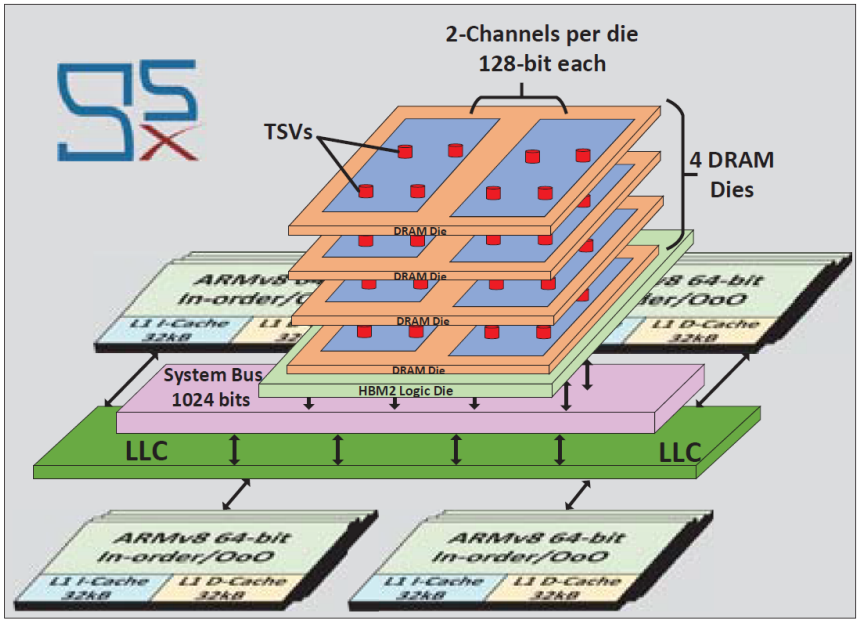


FM-Index: Architecture Exploration

TRANSACTIONS ON COMPUTERS, VOL. 14, NO. 8, FEB 2020 1

Genome Sequence Alignment - Design Space Exploration for Optimal Performance and Energy Architectures

Yasir Mahmood Qureshi, Jose Manuel Herruzo,
Marina Zapater, *Member, IEEE,* Katzalin Olcoz, *Member, IEEE,*
Sonia Gonzalez-Navarro, Oscar Plata, and David Atienza, *Fellow, IEEE,*



Data-Centric Computing

Commercial PNM Solutions

- **UPMEM**
- **Samsung FIMDRAM**

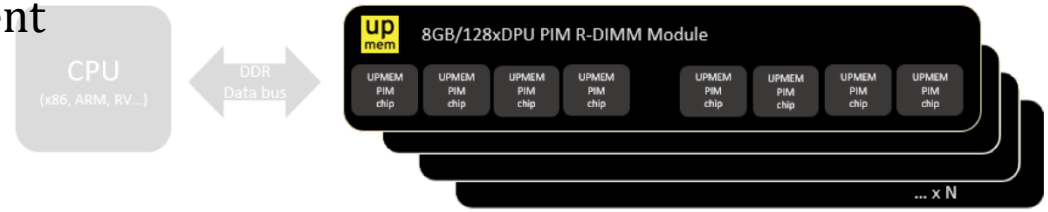
UPMEM: DRAM PIM Accelerator

Processing in DRAM Engine

- **Standard DIMM modules**

- **Key features:**
  - Relies on mature 2D DRAM fabrication process
  - DPUs support a wide variety of computations and data types
  - Threads in DPUs are independent
  - Complete software stack (C language)

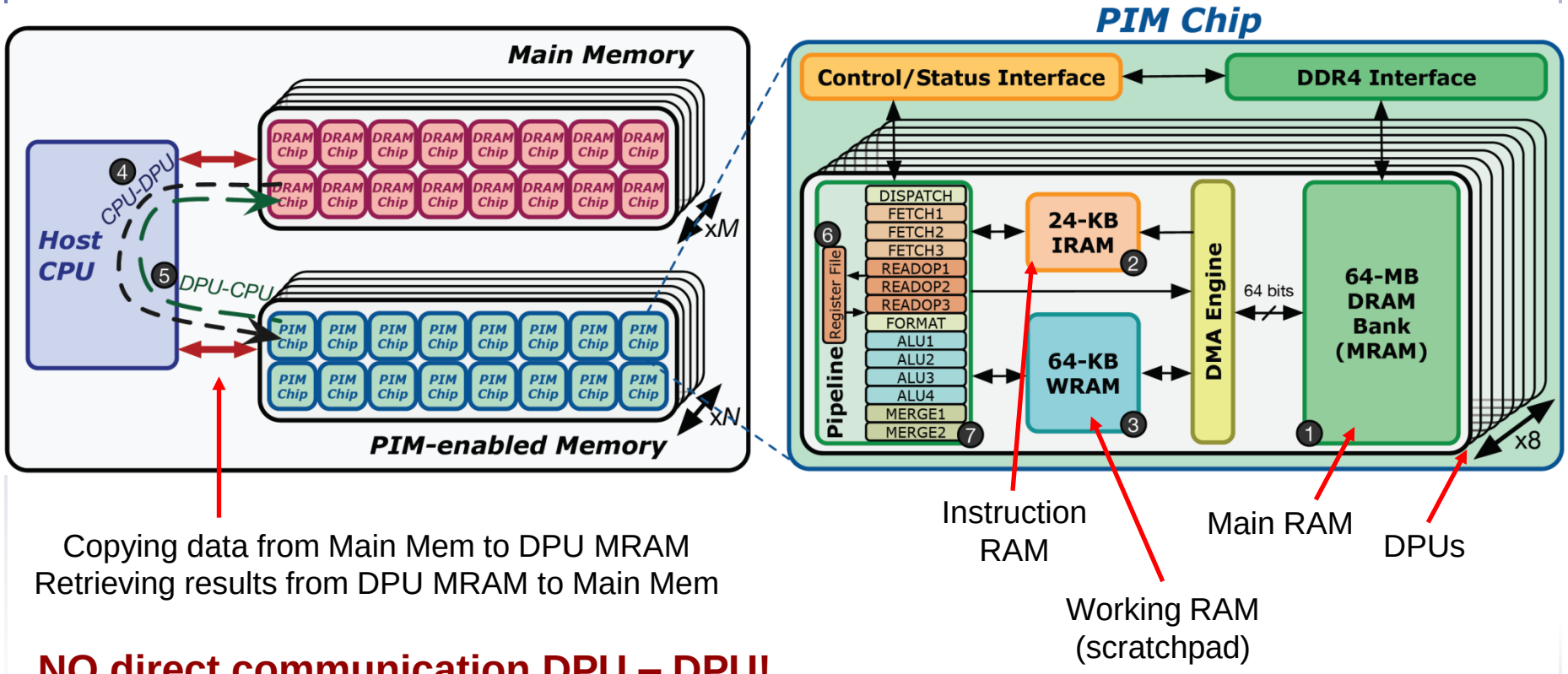
- DDR4-2400 R-DIMM modules:
  - PIM chip: 64MB + 8 DPUs @400-500MHz
  - PIM DIMM: 16 PIM chips (8GB + 128 DPUs @ 400-500MHz)
  - Server: up to 20 PIM DIMMS (160GB + 2560 DPUs)



https://www.upmem.com

UPMEM PIM System Organization

- **PIM DIMM: 8GB DRAM DDR4 2400 + 256 DPUs @ 400-500MHz**

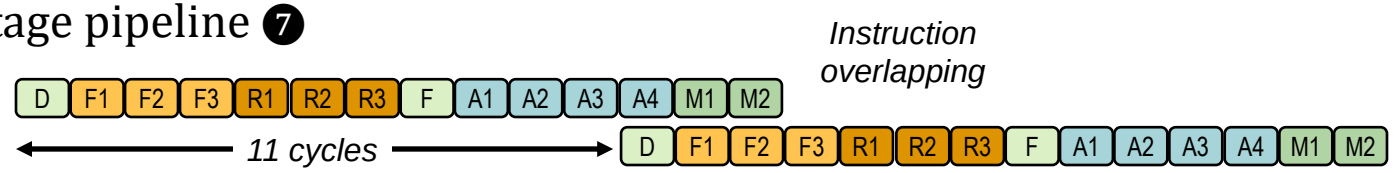


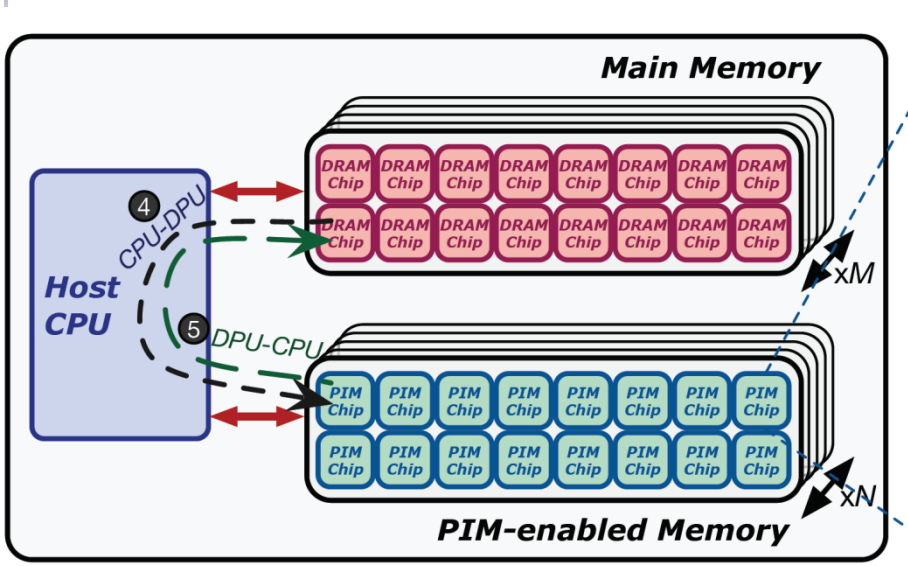
Copying data from Main Mem to DPU MRAM
Retrieving results from DPU MRAM to Main Mem

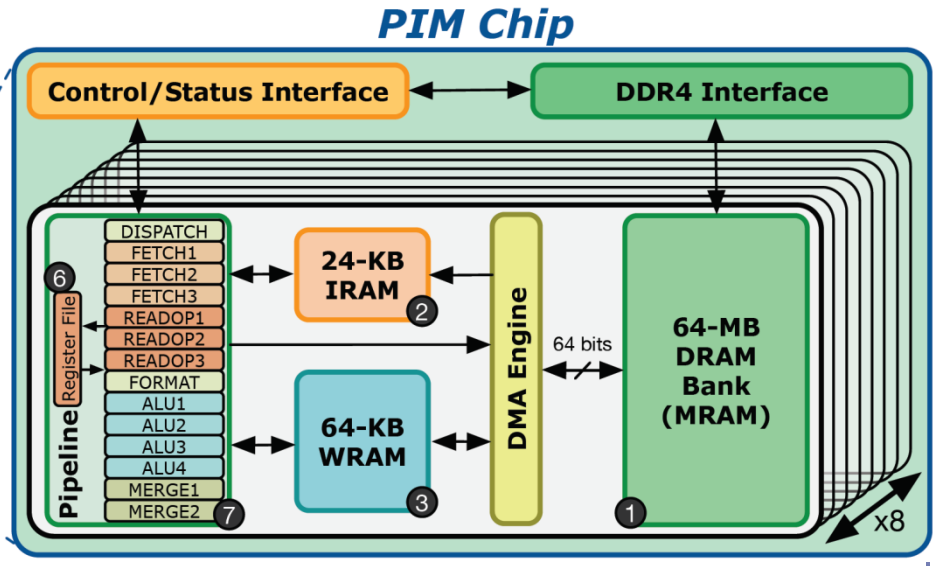
NO direct communication DPU – DPU!

UPMEM PIM DPU Architecture

- **PIM DIMM: 8GB DRAM DDR4 2400 + 256 DPUs @ 400-500MHz**
  - DPU: Multithreaded in-order 32-bit RISC core with **NO** cache hierarchy
  - 24 hardware contexts (threads), each with 24 32-bit GP registers ❻
  - Hardware threads share IRAM and WRAM (for operands)
  - 14-stage pipeline ❼

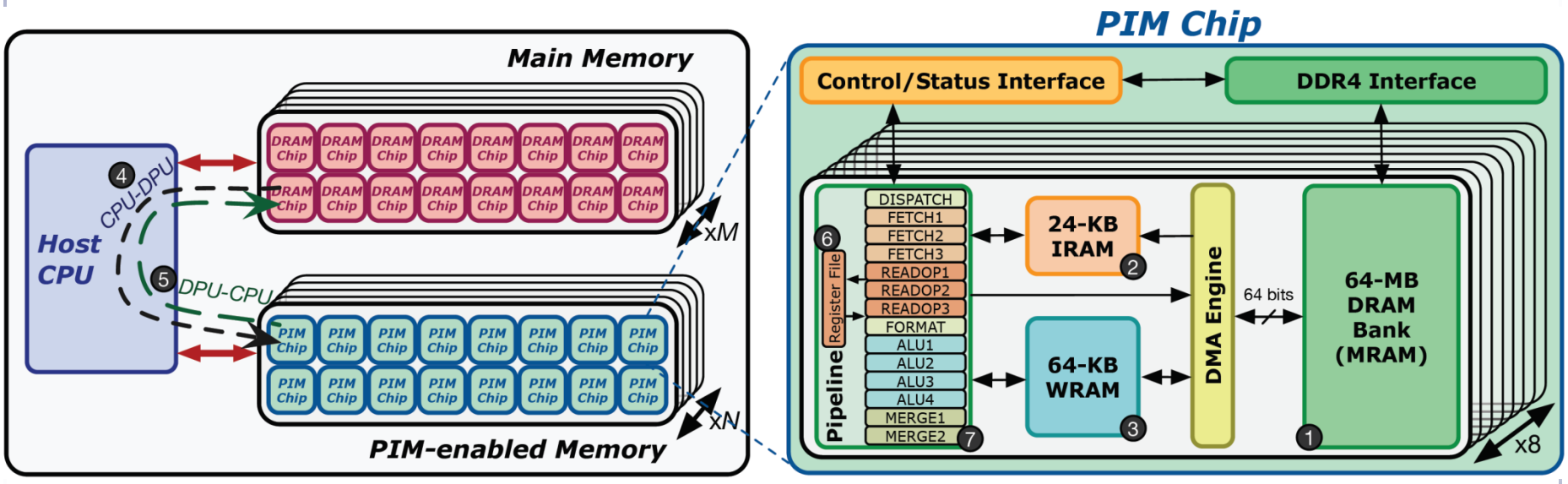






UPMEM PIM Data Throughput

- **PIM DIMM: 8GB DRAM DDR4 2400 + 256 DPUs @ 400-500MHz**
  - DPU max. frequency: 400 MHz
  - MRAM-WRAM bandwidth per DPU: 800 MB/s
  - MRAM-WRAM bandwidth per DIMM: 102 GB/s
  - Max. aggregated MRAM-WRAM bandwidth (20 DIMMS): **~2 TB/s**



UPMEM PIM Programming

- **SPMD programming model**
  - A software thread is called **tasklet**
    - » (1) All tasklets execute the same code on different data pieces
    - » (2) Tasklets can execute different control-flow paths at runtime
  - Up to 24 tasklets can run in parallel in a single DPU
  - Tasklets are statically assigned to each DPU
  - Intra-DPU tasklets can share data in MRAM and WRAM, and can synchronize
  - Inter-DPU tasklets **do not** share memory or any direct communication channel
- **Programs are written in C with library calls (UPMEM SDK)**
- **UPMEM runtime library**
  - Calls to move instructions from MRAM to IRAM
  - Calls to move data between MRAM and WRAM
  - Calls to move data between Main Memory and DPU MRAM
  - Calls for locks, barriers, handshaking, semaphores…

UPMEM PIM Evaluation (I)

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

ACM SIGMETRICS (Fall Edition) 2021

UPMEM PIM Evaluation (II)

- **PrIM Benchmarks**

Domain	Benchmark	Short name	Memory access pattern			Computation pattern		Communication/synchronization	
			Sequential	Strided	Random	Operations	Datatype	Intra-DPU	Inter-DPU
Dense linear algebra	Vector Addition	VA	Yes			add	int32_t		
	Matrix-Vector Multiply	GEMV	Yes			add, mul	uint32_t		
Sparse linear algebra	Sparse Matrix-Vector Multiply	SpMV	Yes		Yes	add, mul	float		
Databases	Select	SEL	Yes			add, compare	int64_t	handshake, barrier	Yes
	Unique	UNI	Yes			add, compare	int64_t	handshake, barrier	Yes
Data analytics	Binary Search	BS	Yes		Yes	compare	int64_t		
	Time Series Analysis	TS	Yes			add, sub, mul, div	int32_t		
Graph processing	Breadth-First Search	BFS	Yes		Yes	bitwise logic	uint64_t	barrier, mutex	Yes
Neural networks	Multilayer Perceptron	MLP	Yes			add, mul, compare	int32_t		
Bioinformatics	Needleman-Wunsch	NW	Yes	Yes		add, sub, compare	int32_t	barrier	Yes
Image processing	Image histogram (short)	HST-S	Yes		Yes	add	int32_t	barrier	Yes
	Image histogram (long)	HST-L	Yes		Yes	add	int32_t	barrier, mutex	Yes
Parallel primitives	Reduction	RED	Yes	Yes		add	int64_t	barrier	Yes
	Prefix sum (scan-scan-add)	SCAN-SSA	Yes			add	int64_t	handshake, barrier	Yes
	Prefix sum (reduce-scan-scan)	SCAN-RSS	Yes			add	int64_t	handshake, barrier	Yes
	Matrix transposition	TRNS	Yes		Yes	add, sub, mul	int64_t	mutex	Yes

UPMEM PIM Evaluation (III)

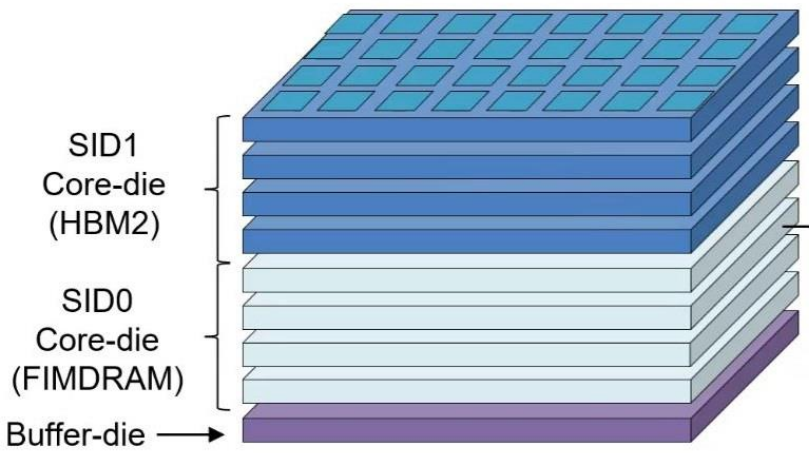
- **Key takeaways**
  - **Takeaway 1**
    - » The UPMEM PIM architecture is fundamentally compute bound
    - » As a result, the most suitable workloads are memory-bound
  - **Takeaway 2**
    - » The most well-suited workloads for the UPMEM PIM architecture use no arithmetic operations or use only simple operations (e.g., bitwise operations and integer addition/subtraction)
  - **Takeaway 3**
    - » The most well-suited workloads for the UPMEM PIM architecture require little or no communication across DRAM Processing Units (inter-DPU communication).
  - **Takeaway 4**
    - » UPMEM PIM systems outperform state-of-the-art CPUs in terms of performance and energy efficiency on most of PrIM benchmarks
    - » UPMEM PIM systems outperform state-of-the-art GPUs on a majority of PrIM benchmarks, and the outlook is even more positive for future PIM systems
    - » UPMEM PIM systems are more energy-efficient than state-of-the-art CPUs and GPUs on workloads that they provide performance improvements over the CPUs and the GPUs

Samsung FIMDRAM: PIM AI Accelerator

Processing in DRAM Engine

- **Function-in-Memory DRAM based on HBM2 (HBM-PIM)**
  - HBM2-based memory with integrated AI processors

■ FIMDRAM based on HBM2



[3D Chip Structure of HBM with FIMDRAM]

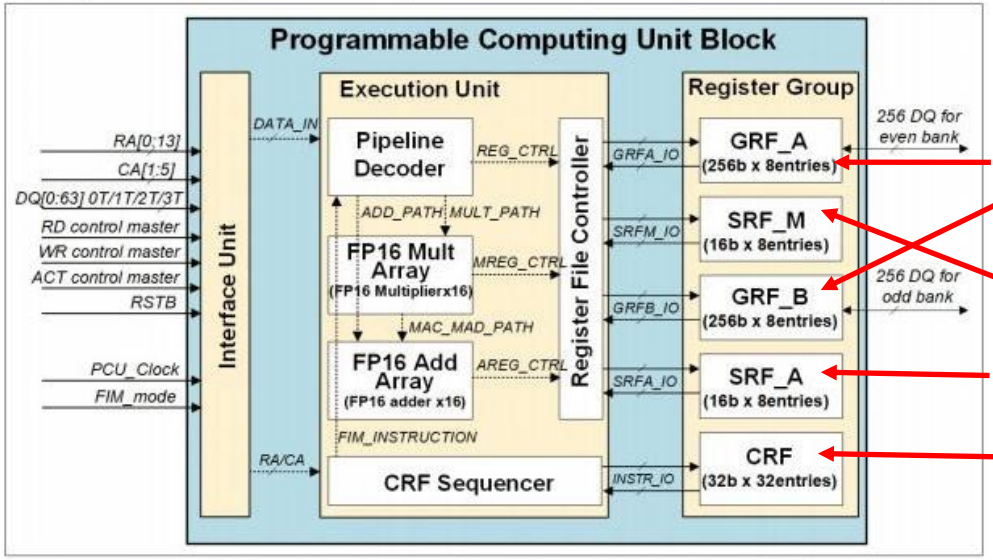
Chip Specification
128DQ / 8CH / 16 banks / BL4
32 PCU blocks (1 FIM block/2 banks)
1.2 TFLOPS (4H)
FP16 ADD / Multiply (MUL) / Multiply-Accumulate (MAC) / Multiply-and- Add (MAD)

Samsung FIMDRAM: PCU

- **Programmable Computing Unit (PCU)**
  - Interface unit to control data flow
  - Execution unit to perform operations

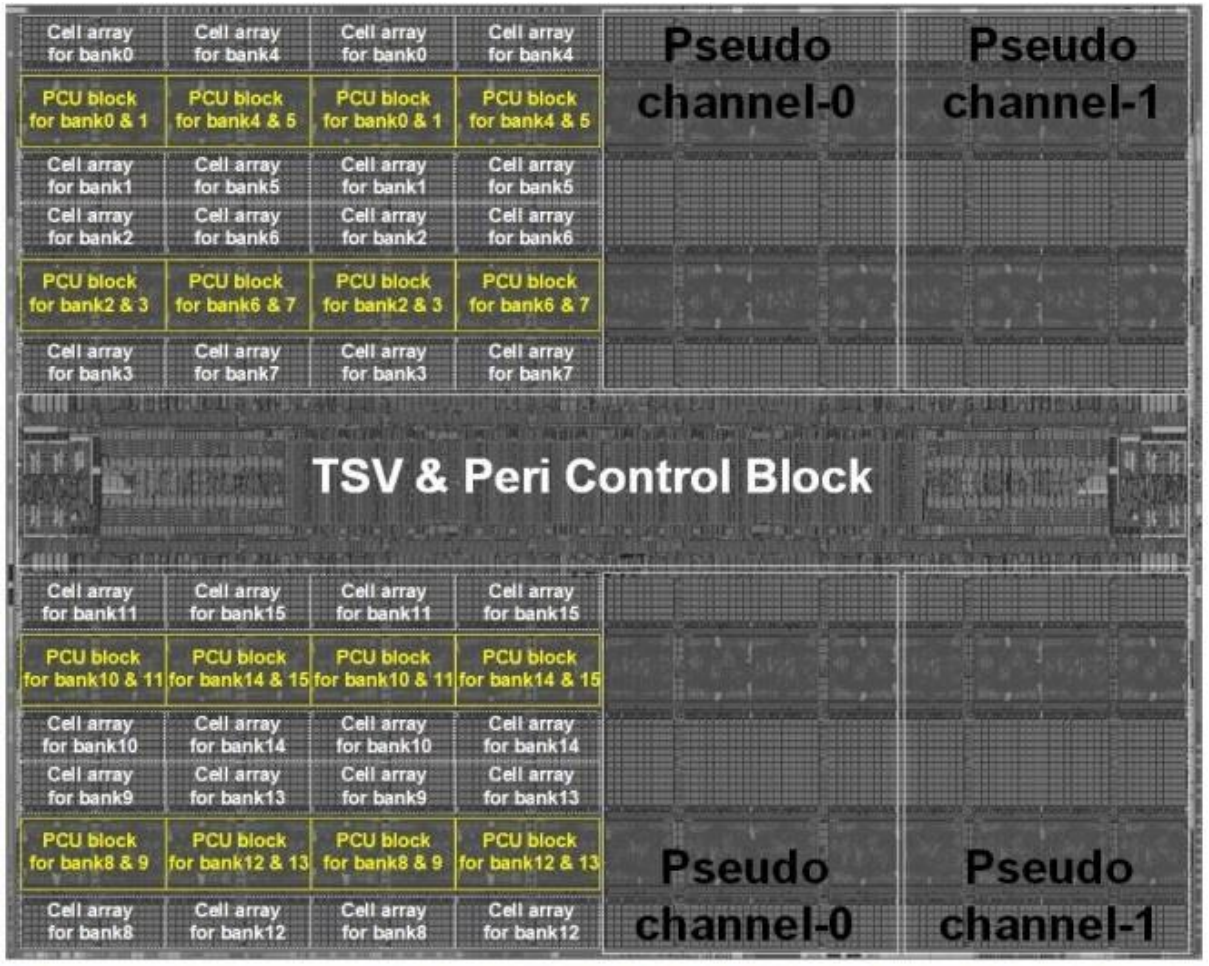
<Available instruction list for FIM operation>

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



Samsung FIMDRAM: Chip Implementation

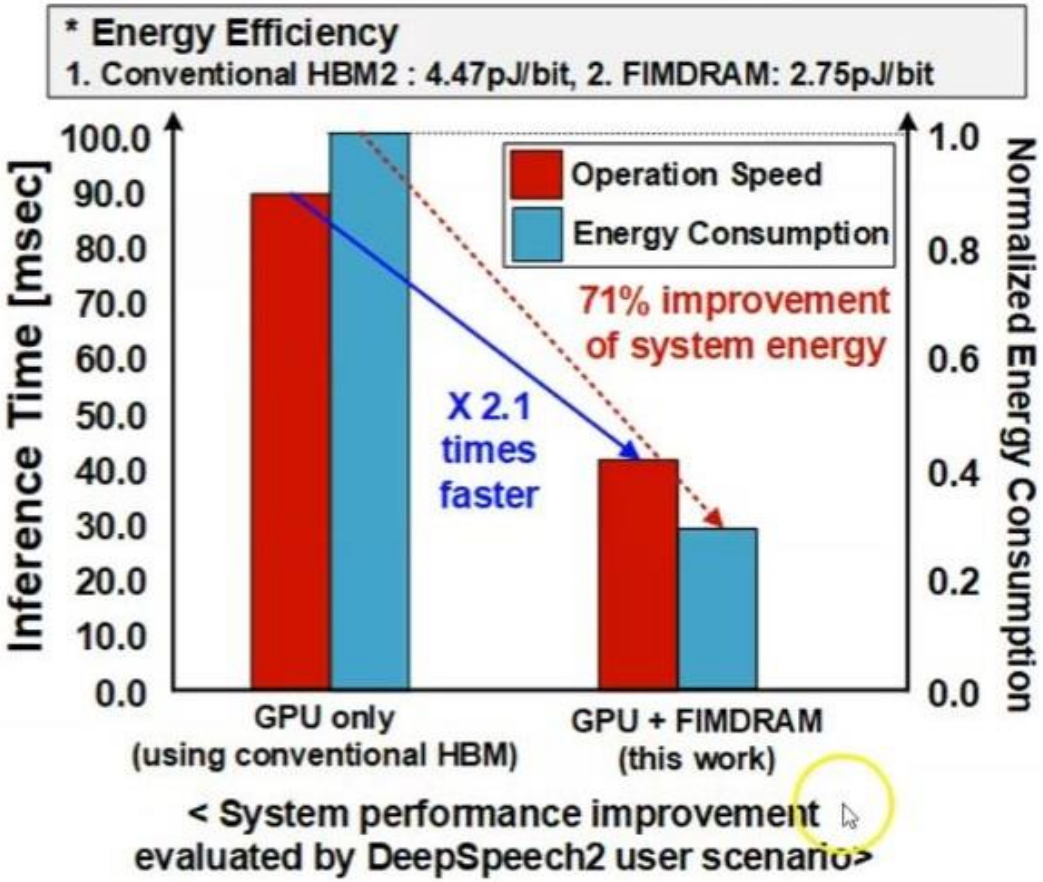
- **Mixed design methodology**
  - Full-custom + digital RTL for PCU block



Samsung FIMDRAM: Experimental Results

- **Performance and energy efficiency**

	HBM2	FIMDRAM
Type of DRAM	HBM2	HBM2
Process	20 nm	20 nm
Memory density	8GB / cube	6GB / cube
Data rate	2.4Gbps	2.4Gbps
Bandwidth	307GB/s per cube	307GB/s per cube
# channels	8 per cube	8 per cube
# processing units	No	128 per cube
Processing operation speed	-	300MHz
Peak throughput	-	1.2 TFLOPS per cube
Operation precision	-	FP16



HPACuma Research Group

ALDEBARAN Research Group

UNIVERSIDAD DE MÁLAGA | uma.es

Processing in Memory

Oscar Plata

oplata@uma.es

Dept. Arquitectura de Computadores
Universidad de Málaga

HPACuma Research Group
ALDEBARAN Research Group

UCM, May 27, 2021