HCiM: ADC-Less Hybrid Analog-Digital Compute in Memory Accelerator for Deep Learning Workloads

Shubham Negi, Utkarsh Saxena, Deepika Sharma and Kaushik Roy

PURDUE UNIVERSITY

CoCoSys
CENTER FOR THE CO-DESIGN OF COGNITIVE SYSTEMS

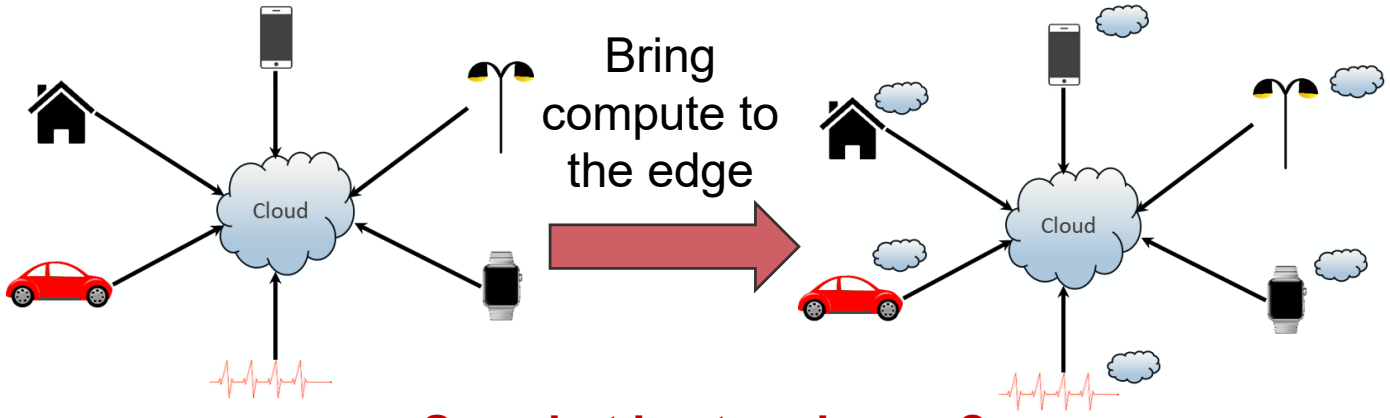
ASIA SOUTH PACIFIC DESIGN AUTOMATION CONFERENCE DAC

22nd January 2025

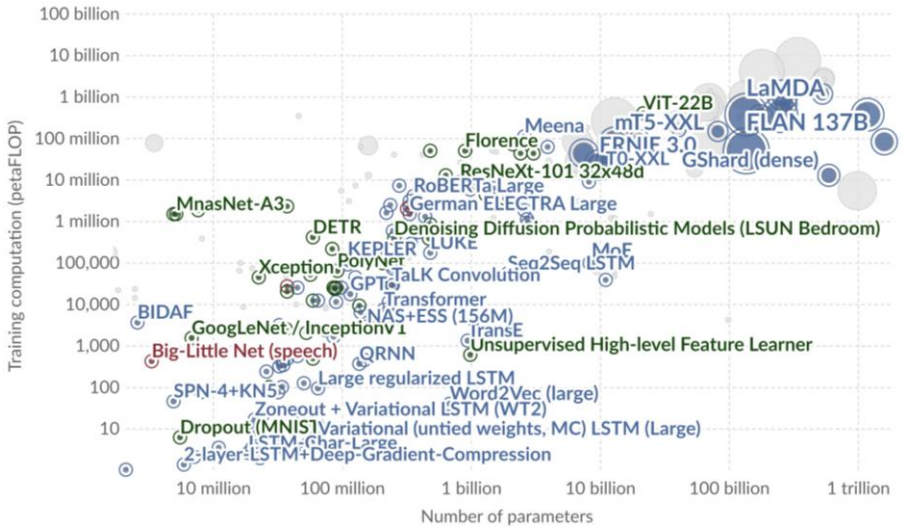
Overview

- Introduction and Background
- Challenges
- Proposed Hardware Algorithm Co-design Approach
  - Two Stage Quantization
  - Hybrid Analog-Digital CiM Accelerator
- Results
- Conclusion

Motivation : Compute in Memory

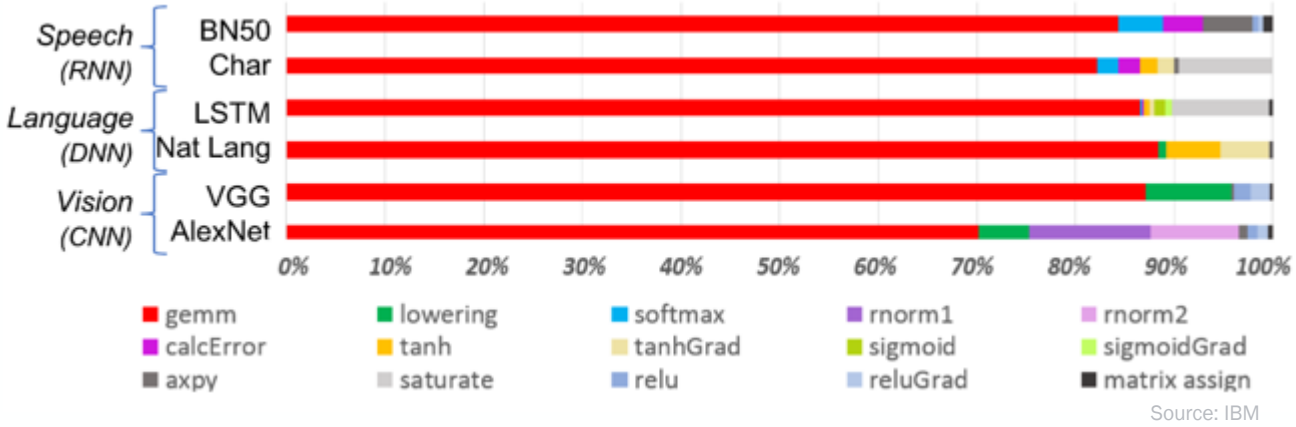


So, what is stopping us?



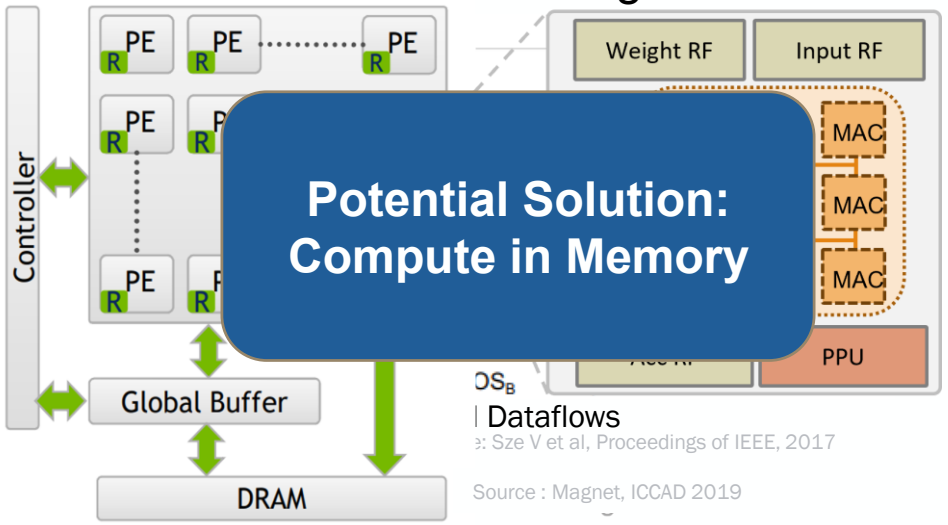
Source: OurWorldInData.org/artificial-intelligence

Exploding computational complexity of Deep Learning models



Source: IBM

Breakdown of operation type across ML workloads

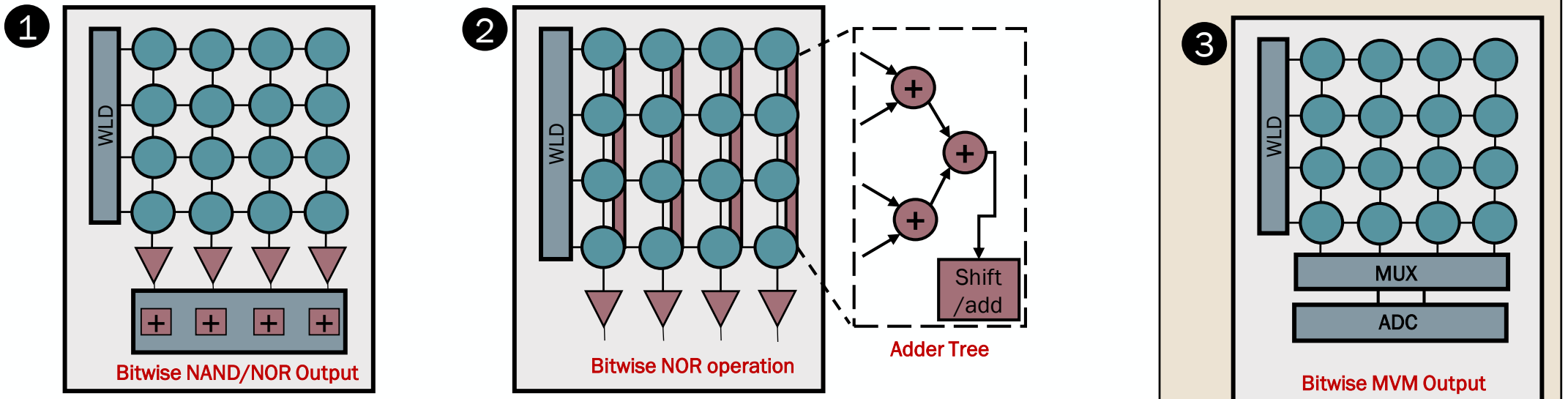


Dataflows

: Sze V et al, Proceedings of IEEE, 2017

Source : Magnet, ICCAD 2019

CiM Accelerators : Types



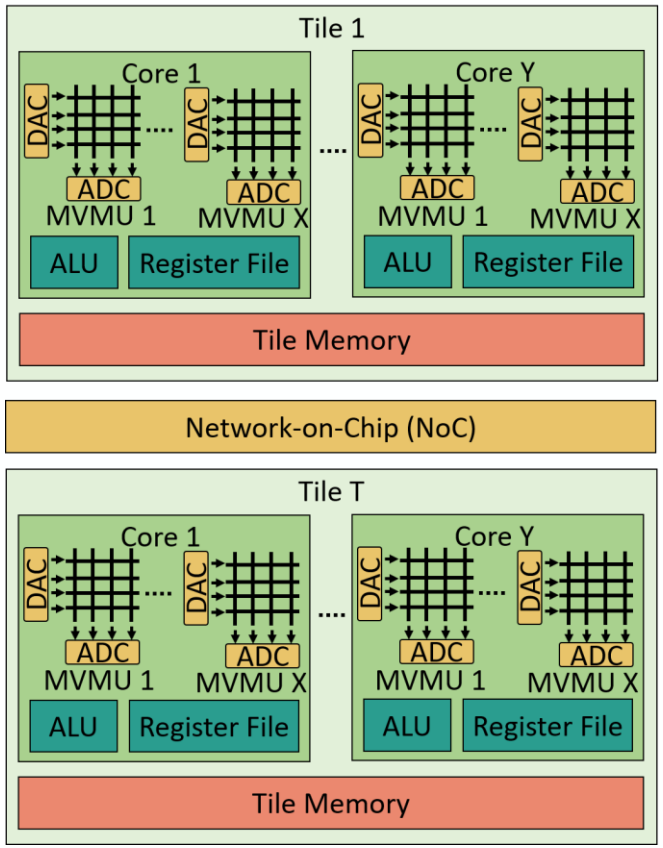
Digital Compute In Memory

Chih et al. ISSCC 2021

Analog Compute In Memory

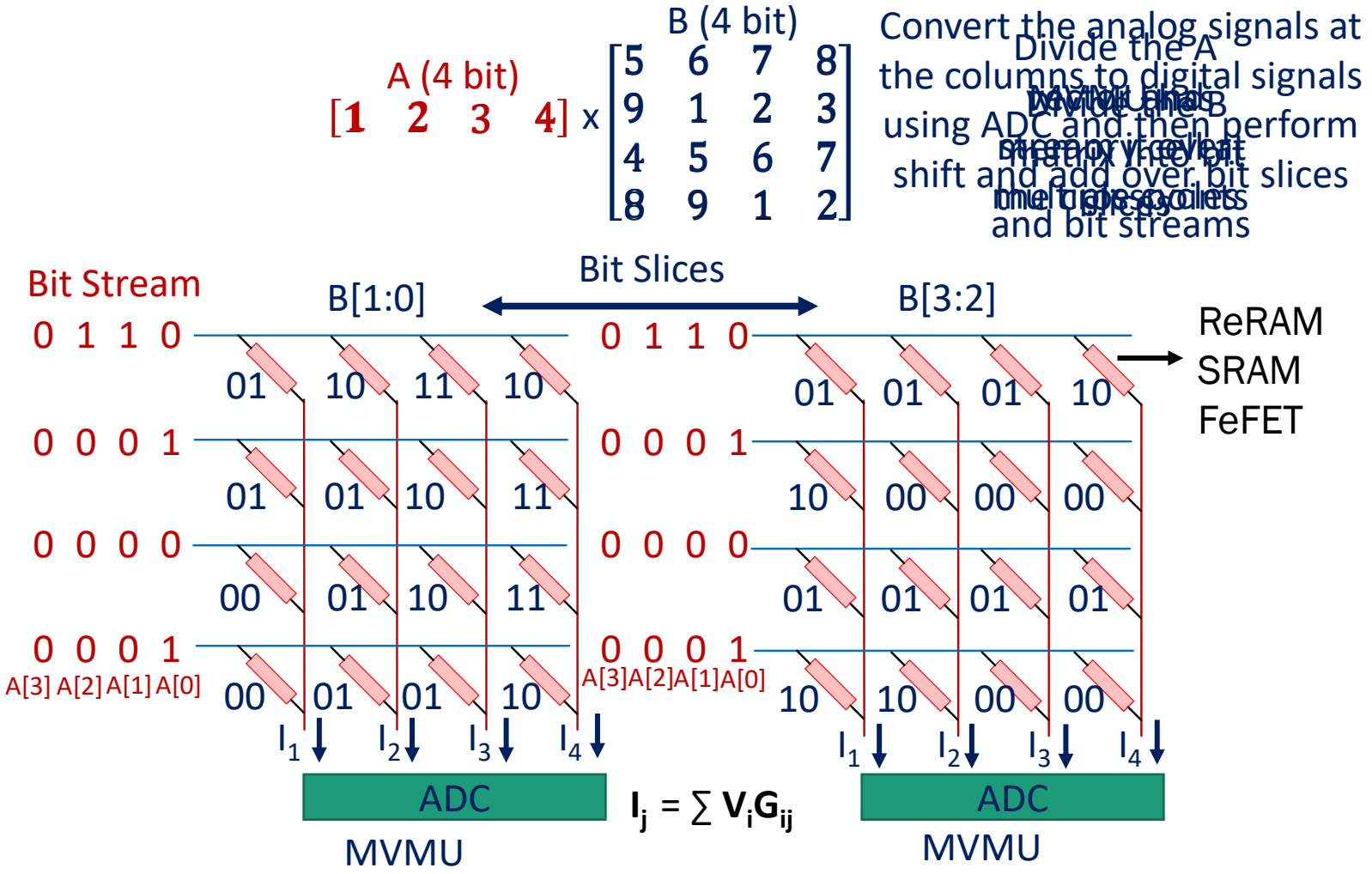
- Digital CIM
  - Bitwise multiplication operation followed by an accumulation in the peripherals
  - Multiple word lines activated to perform bitwise MVM operation in Analog domain
- Analog CIM
  - Reduction performed in analog domain
  - Vector-Matrix multiplication followed by reduction in MVM tree

Analog CiM Accelerator

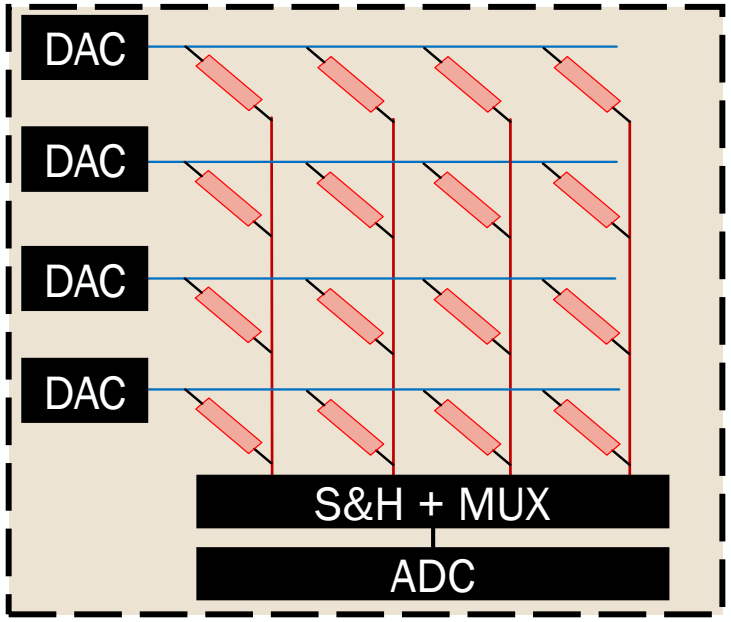


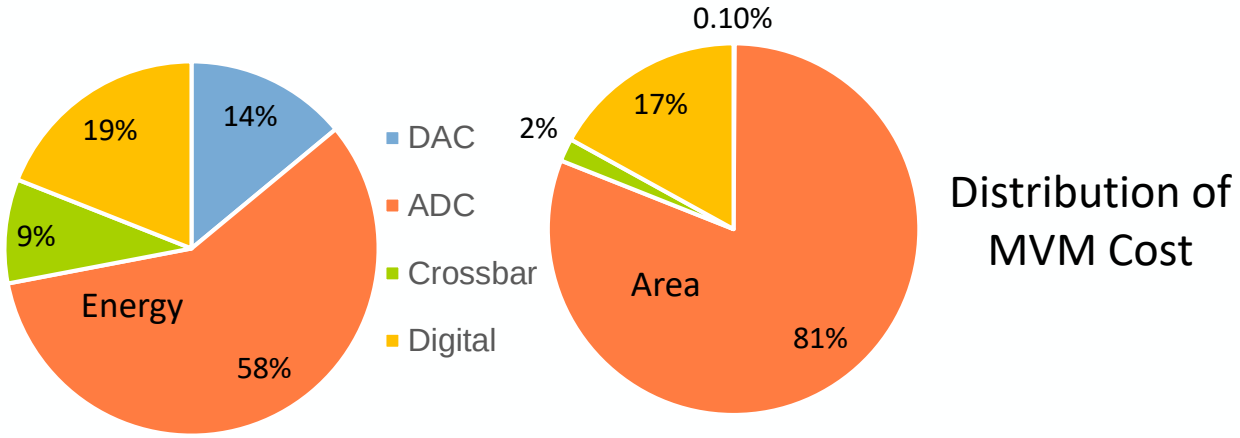
PUMA: CiM Accelerator

Source: Ankit A et al, ASPLOS, 2019



Challenge : Analog to Digital Converter Overhead



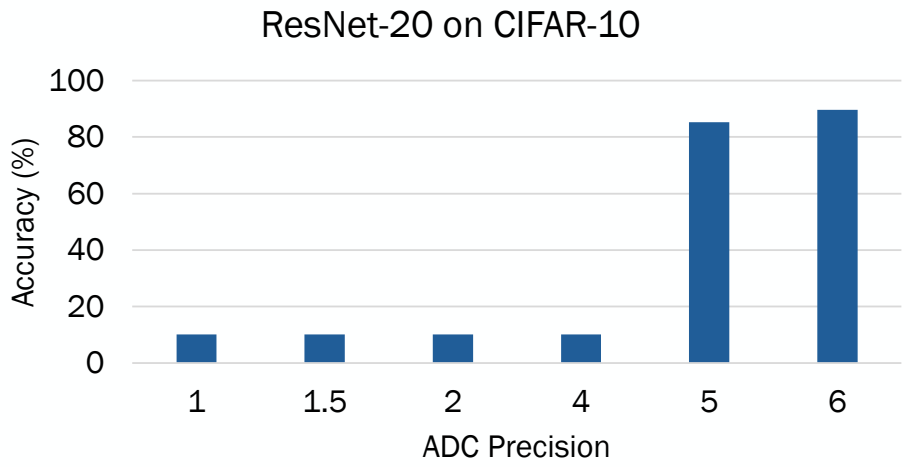


Source: S. Huang et al, ASP-DAC, 2021

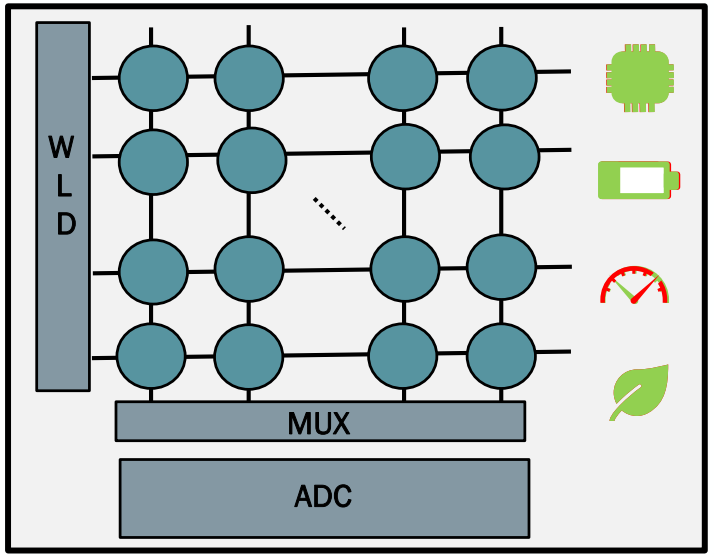
- Hardware efficiency bottlenecked by ADCs
- CiM macro area heavily dominated by ADCs
- Sharing ADCs across multiple memory array columns affect hardware throughput

Reducing ADC Overhead

- Reduce ADC precision to reduce overhead posed by the ADCs.
  - Reducing ADC precision reduces ADC area, power, energy, latency.
  - Reduced ADC area allows more ADCs in a single CiM macro improving throughput.
- Reducing ADC precision quantizes the partial sum.

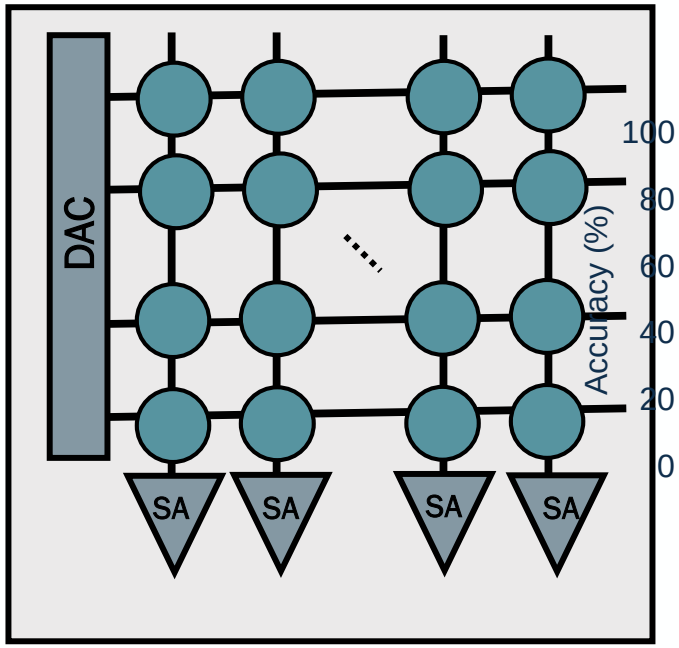


Quantizing partial sum reduces DNN accuracy.

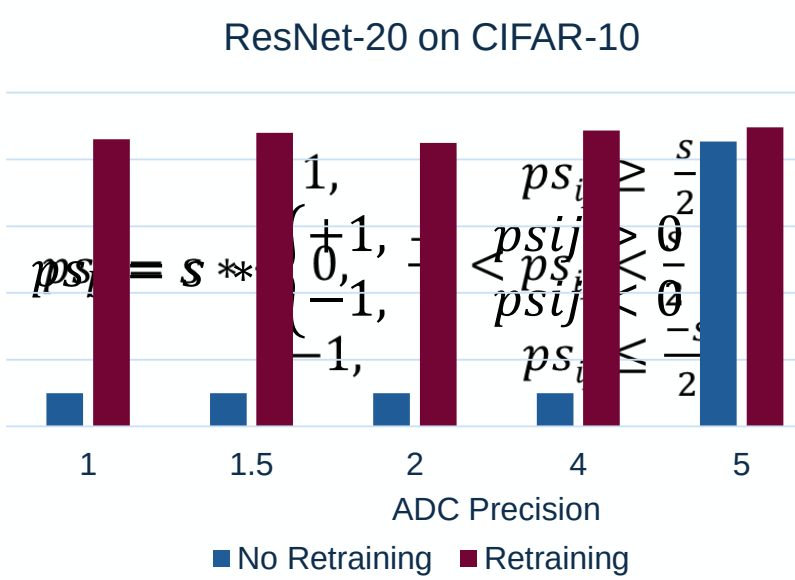


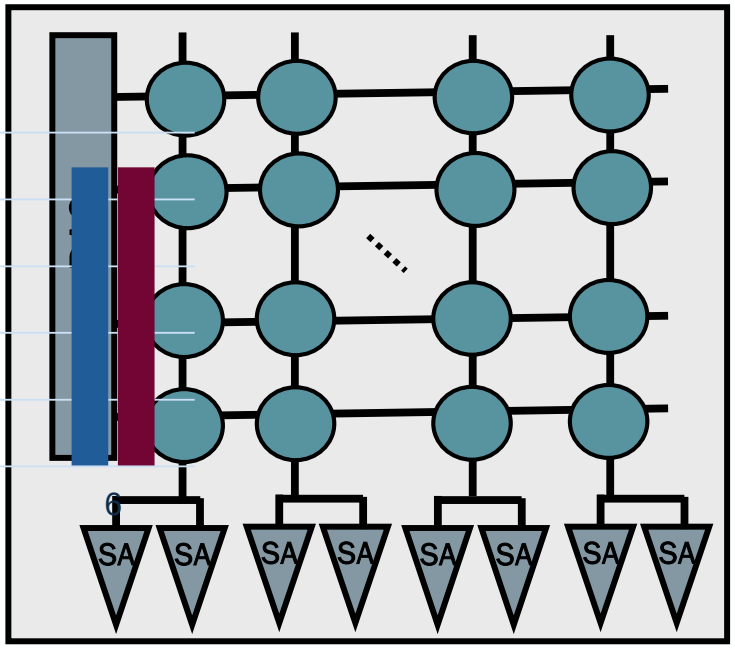
Quantization aware Training

- Train the DNN Model with partial-sum quantization in the training loop
- When partial-sum is quantized to binary → ADC-Less CiM
- When partial-sum is quantized to ternary → Near ADC-Less CiM



ADC-Less CiM



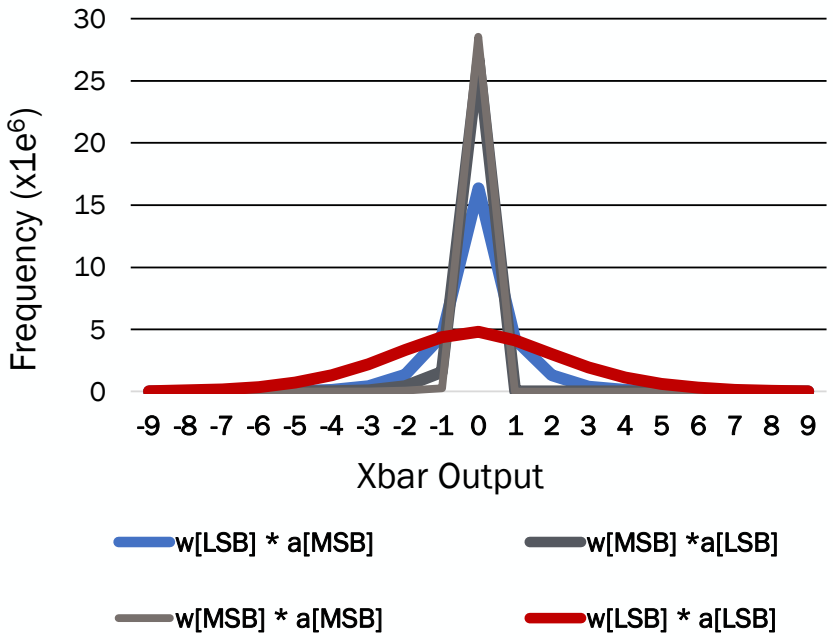


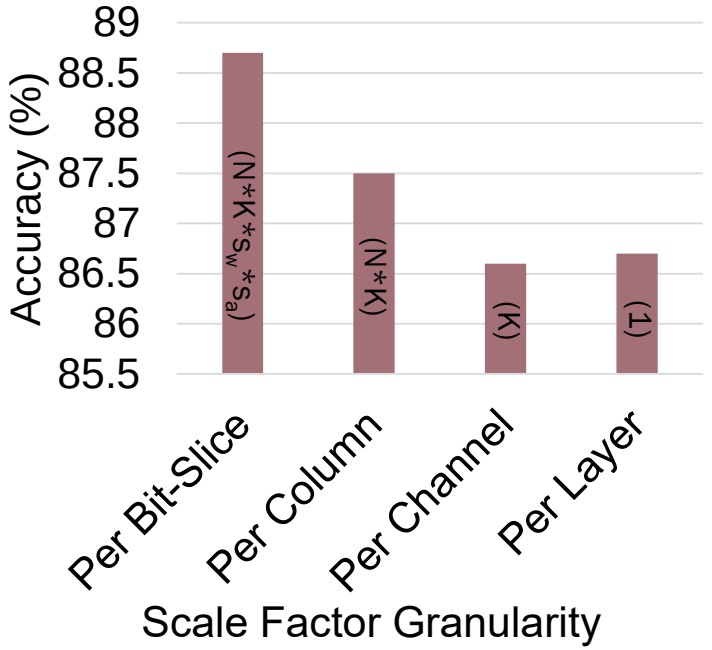
Near ADC-Less CiM

Source: Saxena U Partial-Sum Quantization for Near ADC-Less Compute-In-Memory Accelerators, ISLPED, 2023

Accuracy vs Scale Factor Granularity

- Scale factor granularity plays a big role in the final accuracy achieved
- Reducing number of scale factors results in accuracy drop



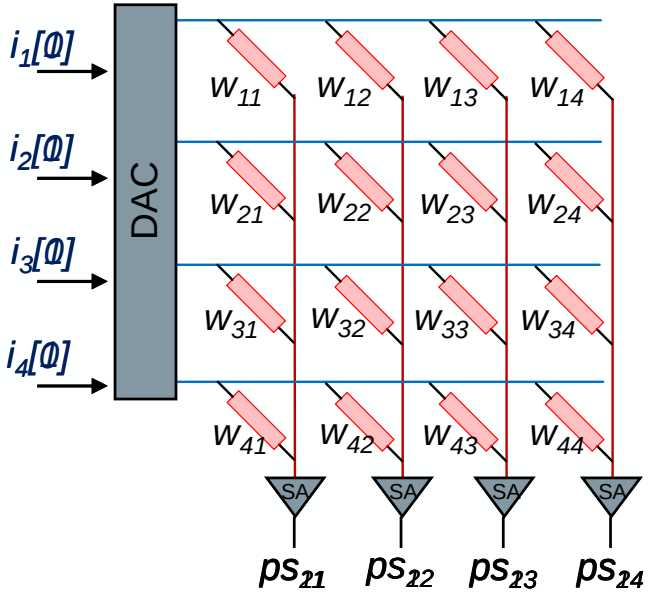


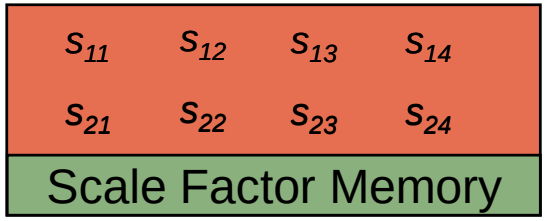
Scale factors are crucial for ultra-low precision quantization

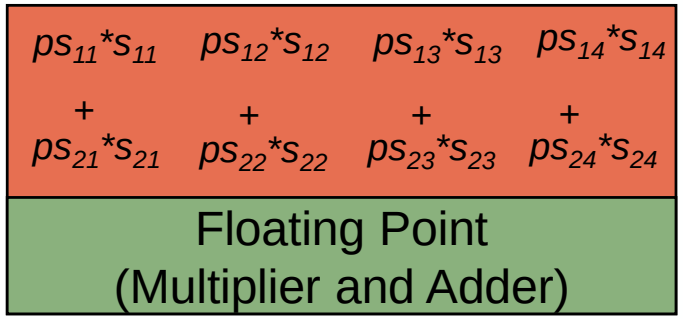
N : Number of Xbars || K: Output channel dimension || s_a : Number of activation bit-slices || s_w : Number of weight bit-slices

Source: Saxena U Partial-Sum Quantization for Near ADC-Less Compute-In-Memory Accelerators, ISLPED, 2023

Deploying on ADC-Less CiM Accelerator

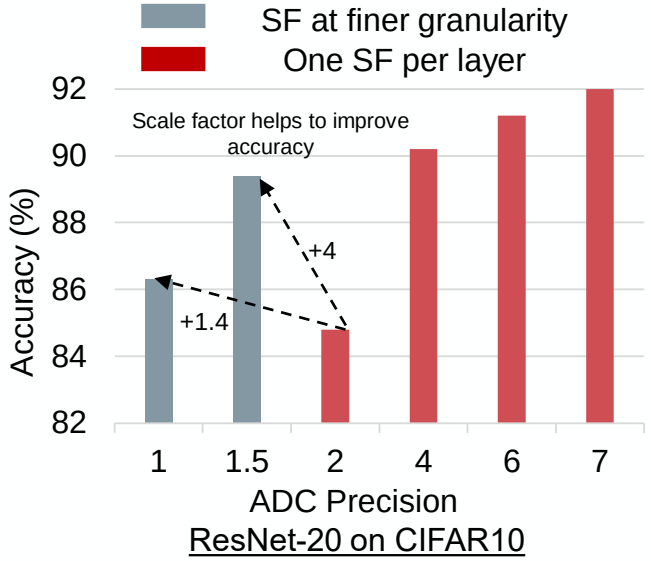


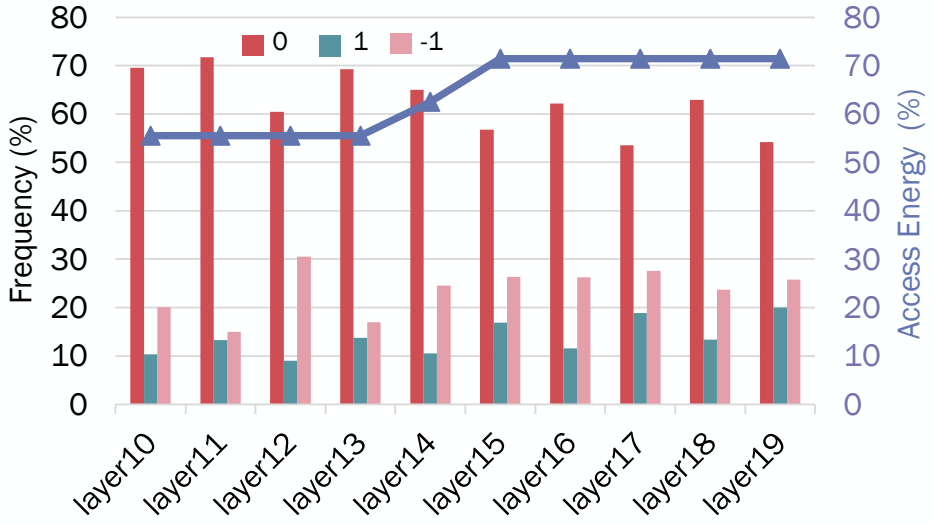




- Partial sum quantization algorithm uses **floating point scale factors**
- Quantized partial sums need to be **dequantized** before accumulating partial sums from crossbars
- This dequantization requires **floating point arithmetic units**

Scale Factor Overhead





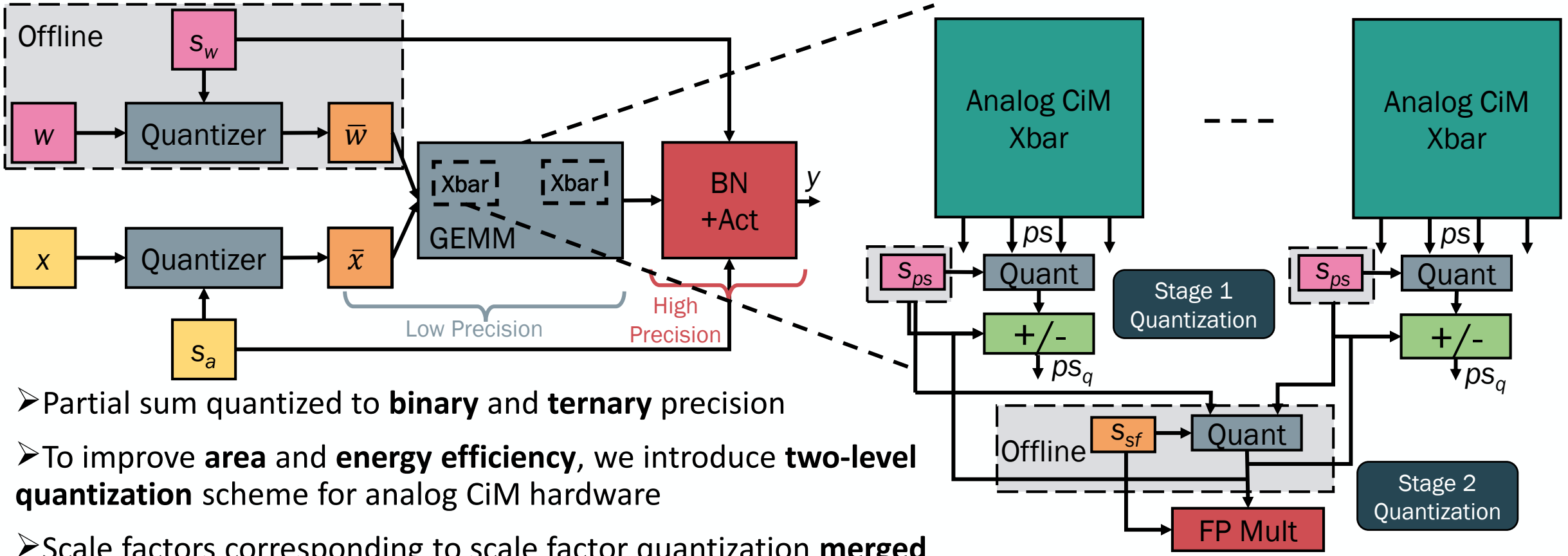
- Overheads:
  - Need power hungry **floating point multipliers** to dequantize the partial sums before adding them from different crossbars
  - Scale factors requires very **high data movement energy**

Is it possible to process these scale factors efficiently?

- Opportunities:
  - Quantized values ∈ {+1,-1,0} therefore we only need **adders/subtractors** instead of multipliers
  - Computation corresponding to quantized value 0 in case of ternary quantization can be **skipped**

Our Approach: Hardware-Software Co-Design

❑ Two Stage Quantization



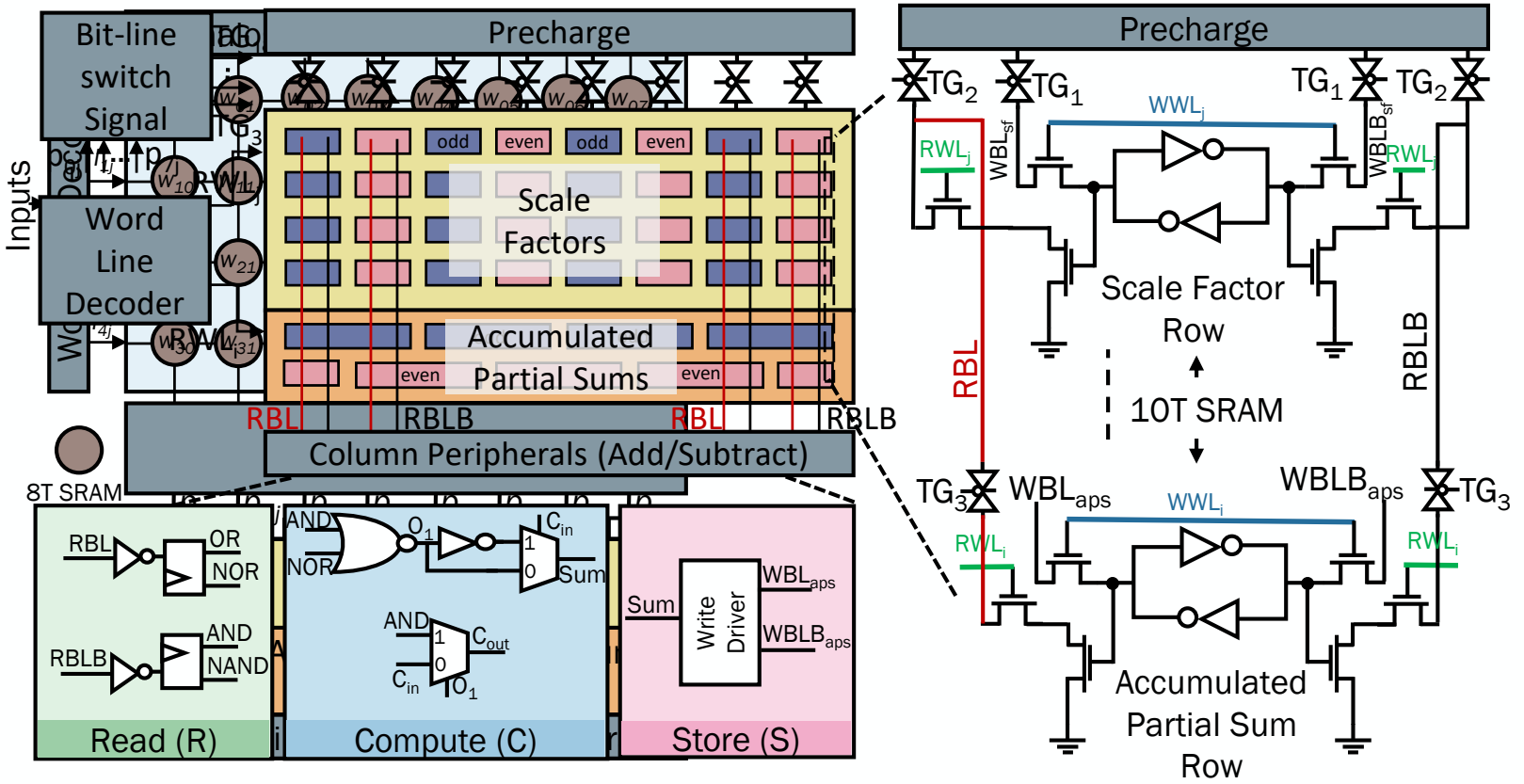
➢Partial sum quantized to **binary** and **ternary** precision

➢To improve **area** and **energy efficiency**, we introduce **two-level quantization** scheme for analog CiM hardware

➢Scale factors corresponding to scale factor quantization **merged with batch normalization layer**

Our Approach: Hardware-Software Co-Design

❑ HCiM: Hybrid Analog-Digital CiM Macro

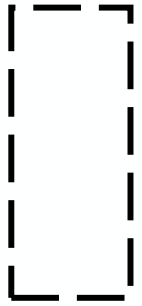


- Process scale factors **inside the memory array** itself ($APS_i = \sum(pij * \boldsymbol{s_{ij}})$)
- Accumulated partial sums and scale factors are stored in **staggered fashion** [1]
- The digital CiM macro needs to support **addition, subtraction and no operation**

[1] Agrawal A, et al. IMPULSE, IEEE Solid-State Circuits Letters. 2021 Jun 28;4:137-40.

Full Subtractor

A (SF)	B (APS)	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



$$D = A \oplus B \oplus B_{in}$$

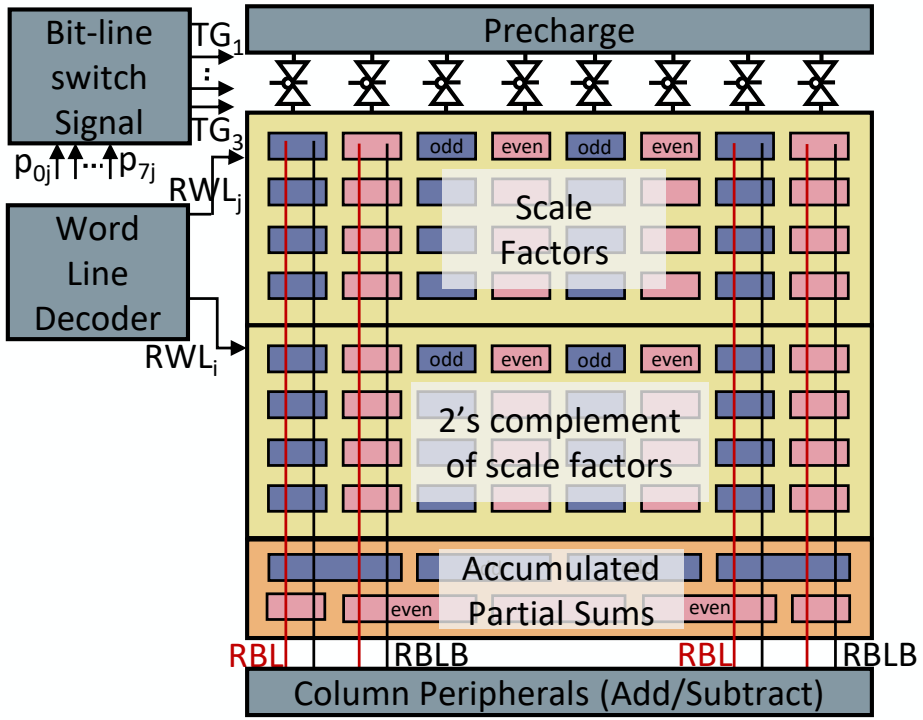
$$Bout = \bar{A}B + B_{in}\bar{A} + BB_{in}$$

How do we realize CiM subtraction?

- Difference bit (D) is same as the sum bit of a full adder
- Subtraction is a **non-commutative operation**
- When A=B, B_{out}=Bin, for other case Bout depends on inputs
- Borrow bit cannot be realized using bitwise AND, NOR

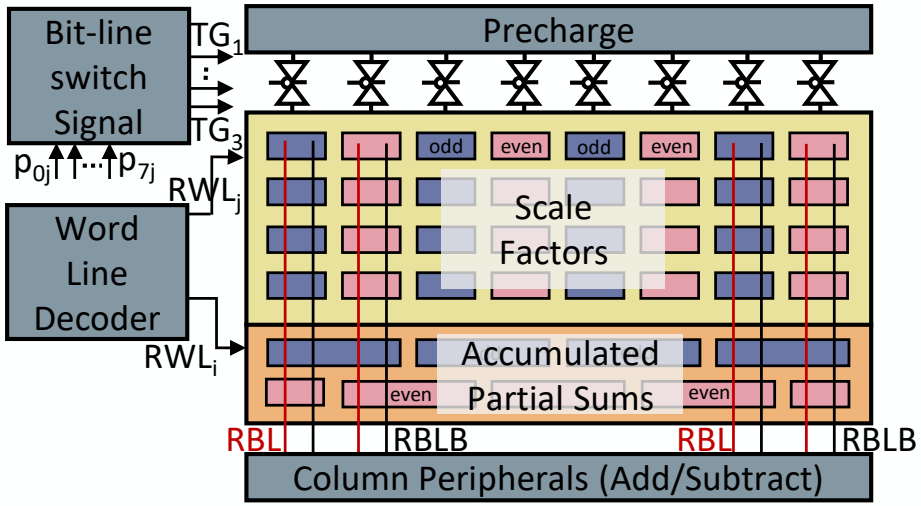
Possible Approaches

- Store 2's complement of scale factors



Need 2x memory

- Read one of the inputs in next cycle

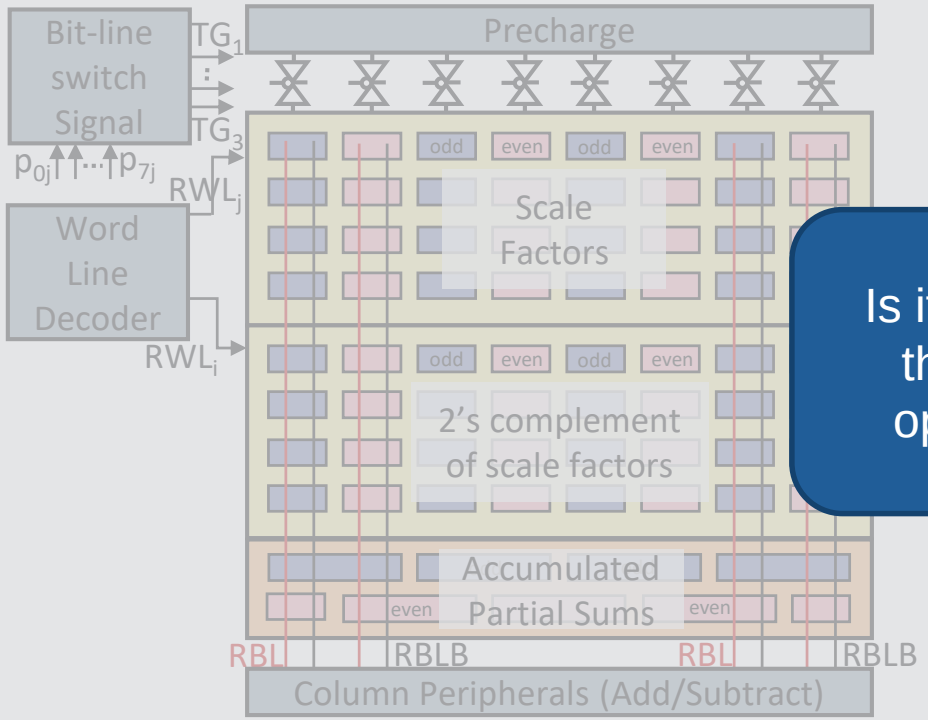


- 1st Cycle – Read AND, NOR
- 2nd Cycle – Read SF value

Need an extra cycle

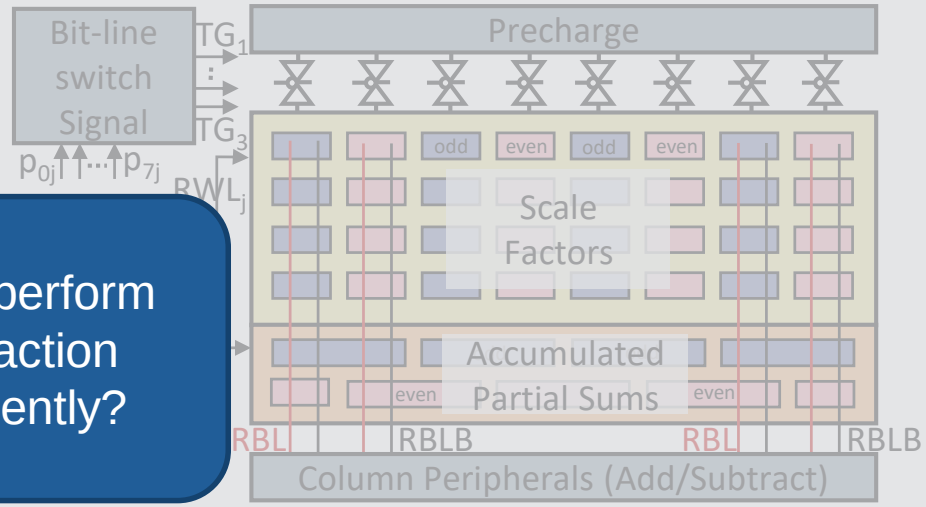
Possible Approaches

❑ Store 2's complement of scale factors



Need 2x memory

❑ Read one of the inputs in next cycle



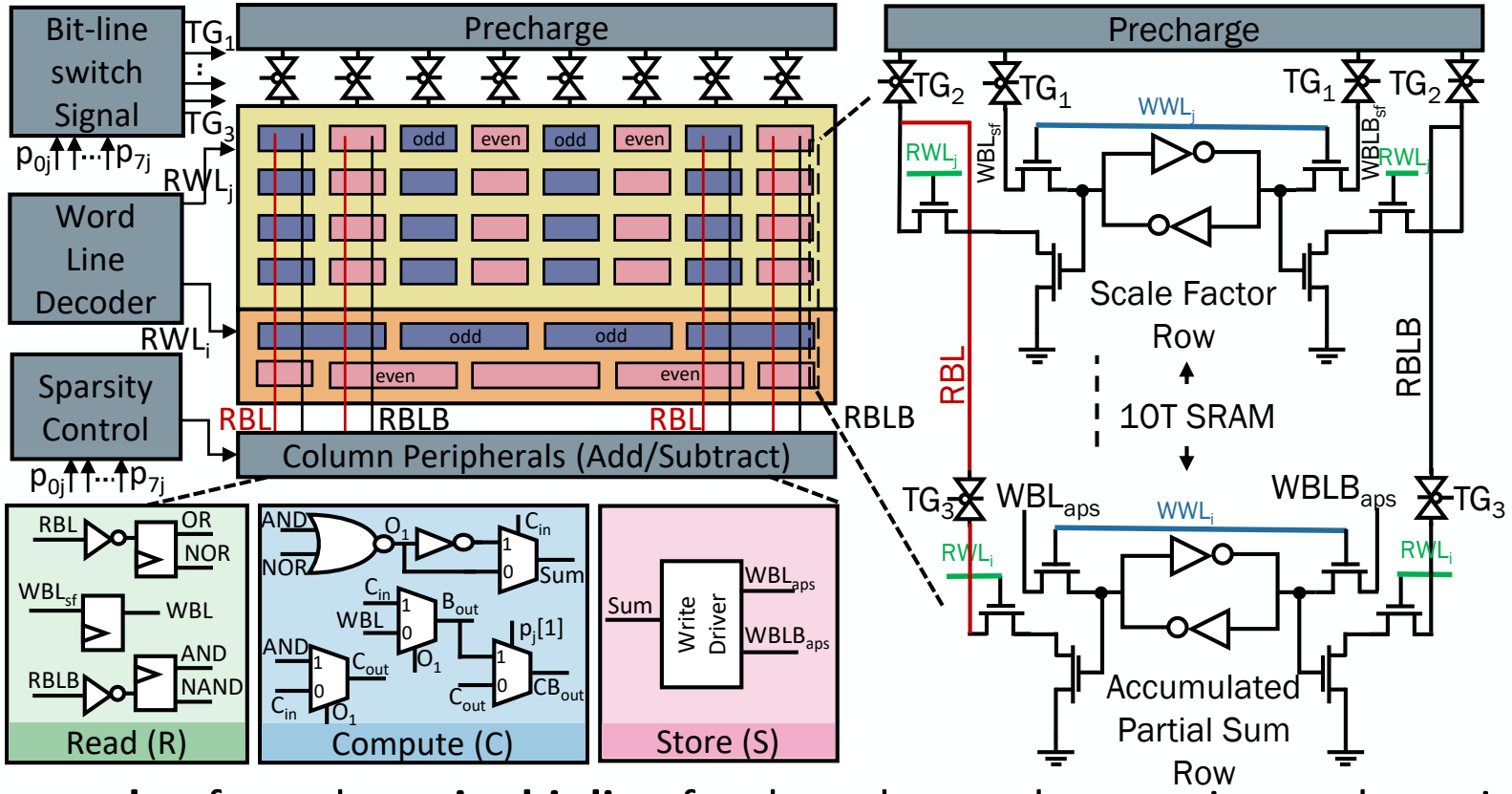
- 1st Cycle – Read AND, NOR
- 2nd Cycle – Read SF value

Need an extra cycle

Is it possible to perform this CiM subtraction operation efficiently?

Our Approach: Hardware-Software Co-Design

❑ HCiM: Hybrid Analog-Digital CiM Macro



- **Read scale factor value** from the **write bit line** for the columns that require a subtraction operation
- Column peripherals **reconfigured as adder/subtractor** depending on $p \in \{+1,-1,0\}$
- We can **process multiple columns** of analog CiM crossbars in parallel
- Inherent sparsity in quantized values (p) helps to reduce energy consumption

Evaluation

❑ Accuracy Results (CIFAR-10)

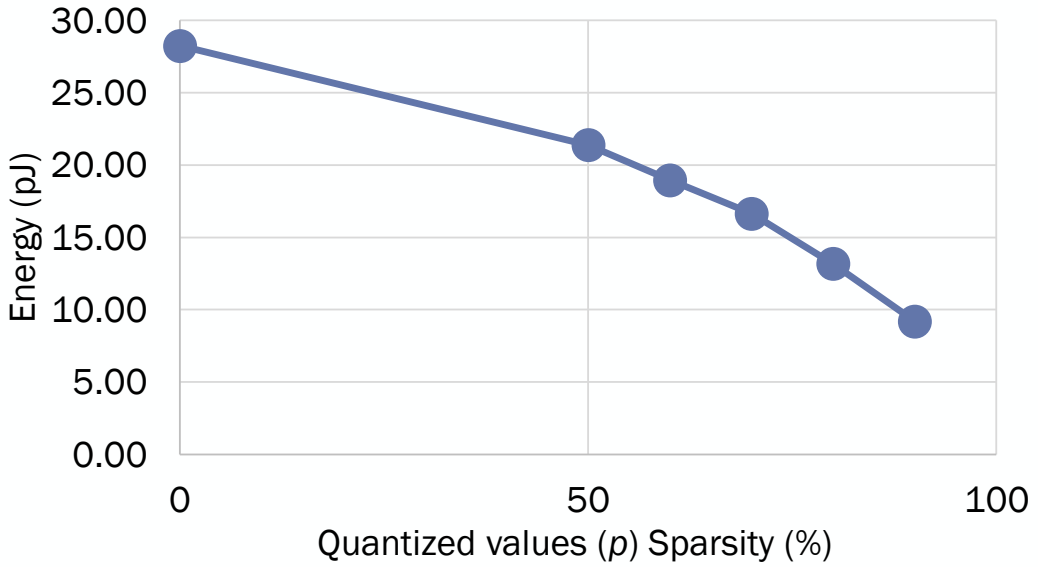
Model (Xbar Size)	ADC Precision (bits)					
	7	6	4	2	1.5	1
ResNet-20(128)	92.26	91.27	90.20	82.40	88.80	86.30
ResNet-20(64)	-	91.93	91.00	83.00	**89.80**	88.20
Wide ResNet-20(128)	93.80	93.70	92.90	88.30	92.03	91.90
Wide ResNet-20(128)	-	93.91	93.10	89.40	**92.24**	91.89

➢For accuracy evaluation we use a functional simulator that models CiM accelerator architectural details such as tiling, bit-slice and bit-stream

➢Our two-level quantization approach results in minimal drop in accuracy (~1.5%)

Evaluation

- Experimental Setup

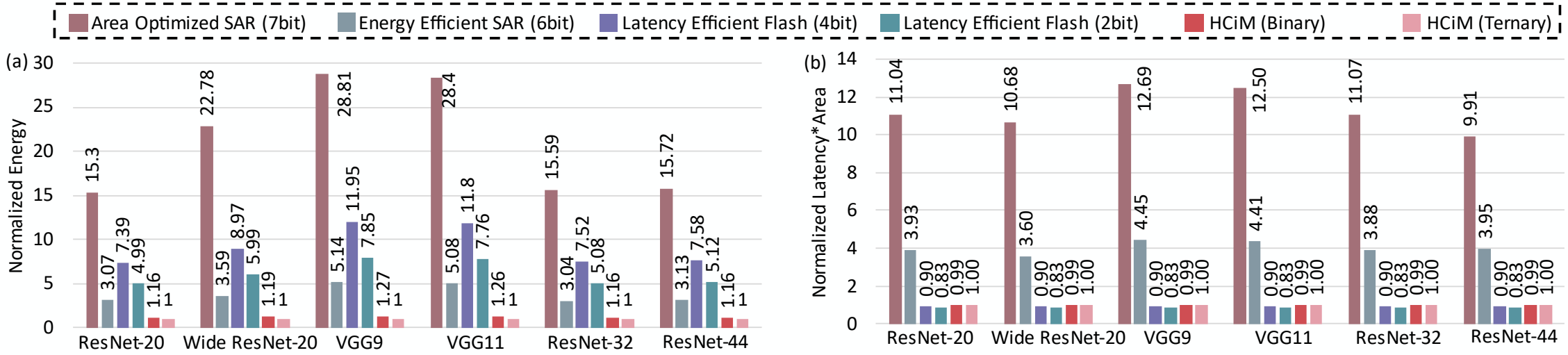


Energy to process all the columns of analog CiM crossbar with ternary quantization

- The DCiM array is designed in **65 nm technology**
- The performance results are based on schematic-level simulations
- Energy to process all the columns of analog CiM crossbar reduces with increase in sparsity of ternary quantized values (*p*)

Evaluation

❑ Performance Results (128x128)



➢ We used the cycle accurate simulator PUMA to get the energy and latency for our system level results

➢ The energy, latency and area for ADCs are estimated from the ADC survey

➢ Mapping neural networks to HCiM results in **3-28x** lower energy consumption compared to baselines and **3-12x** lower latency compared to 6-7 bit ADCs

[1] Boris Murmann. [n. d.]. ADC Performance Survey 1997-2023. [Online]. Available: https://github.com/bmurmann/ADC-survey
[2] Ankit A et al, PUMA ASPLOS, 2019

Evaluation

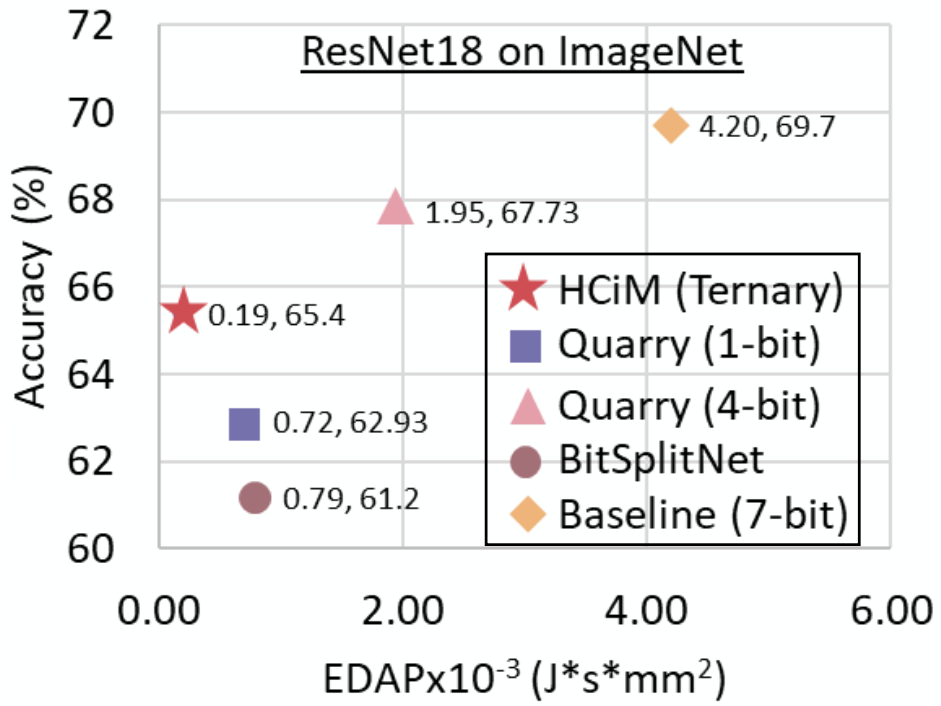
❑ Comparison of ResNet-18 mapped to HCiM with related works on ImageNet dataset

➢ Quarry (1-bit) uses per crossbar scale factor and **digital multiplier** to dequantize crossbar outputs

➢ BitSplitNet uses independent paths to process each input and weight bits

➢ Compared to Quarry with 1-bit ADC, HCiM achieves **2.5% higher accuracy** with **3.8x lower EDAP** (energy-delay-area product)

➢ Compared to BitSplitNet, HCiM has **4.2% higher accuracy** with **4.2x lower EDAP**



Summary

- The performance of neural network deployed on analog CiM accelerator is **limited by ADCs**
- Partial sum quantization introduces **floating point scale factors**
- We propose a **two-stage quantization (TSQ) algorithm** to eliminate the need for ADCs in analog CiM accelerators
- We introduced a **hybrid analog digital compute in memory macro (HCiM)** to deploy TSQ trained DNN
- Our **system-level evaluation** using a cycle-accurate simulator shows up to **28x** and **12x** reduction in energy compared to the baseline that uses 7-bit and 4-bit ADCs

Thank You!

PURDUE UNIVERSITY®