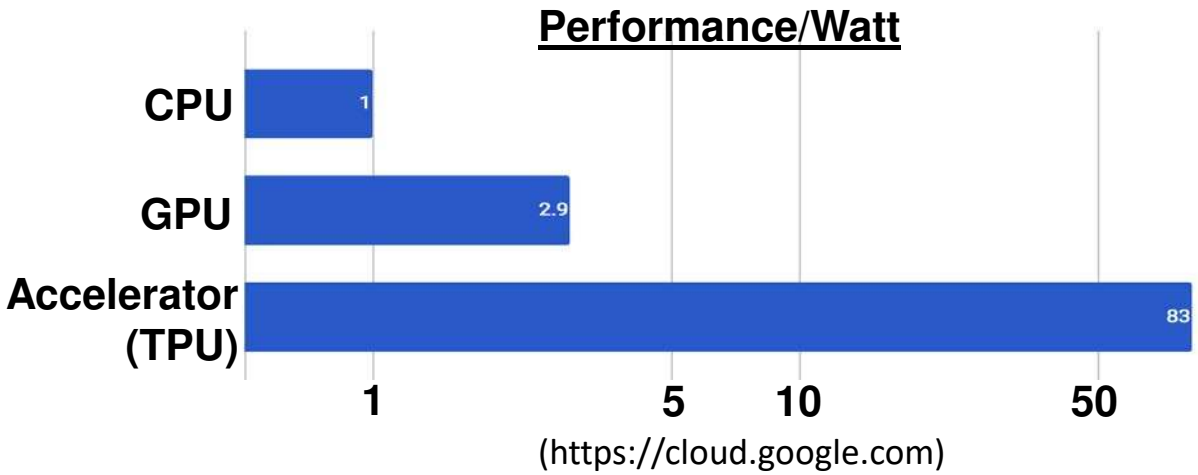
A Programmable Embedded Microprocessor for Bit-scalable In-memory Computing

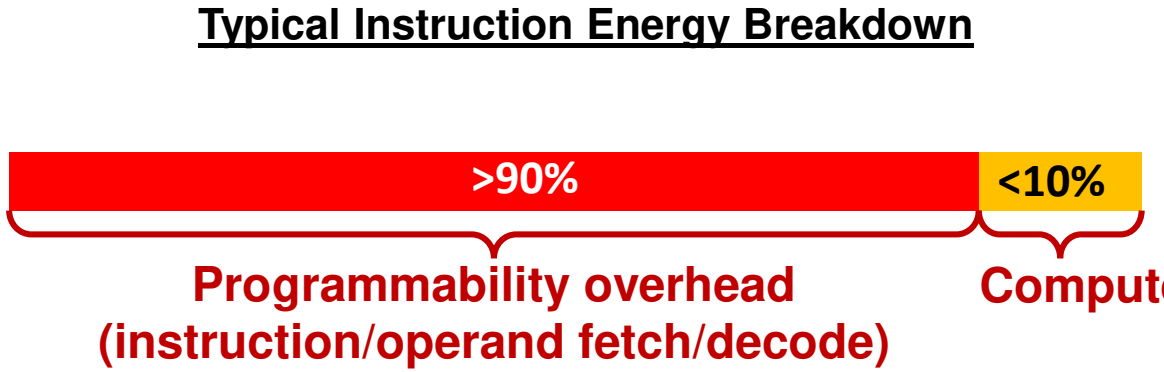
Hongyang Jia (hjia@princeton.edu),
H. Valavi, Y. Tang, J. Zhang, N. Verma

HotChips 2019

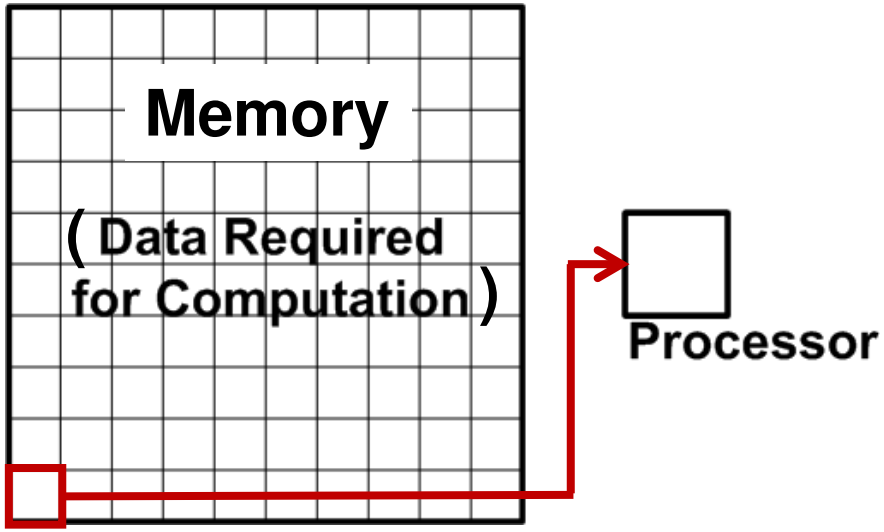
Digital Acceleration

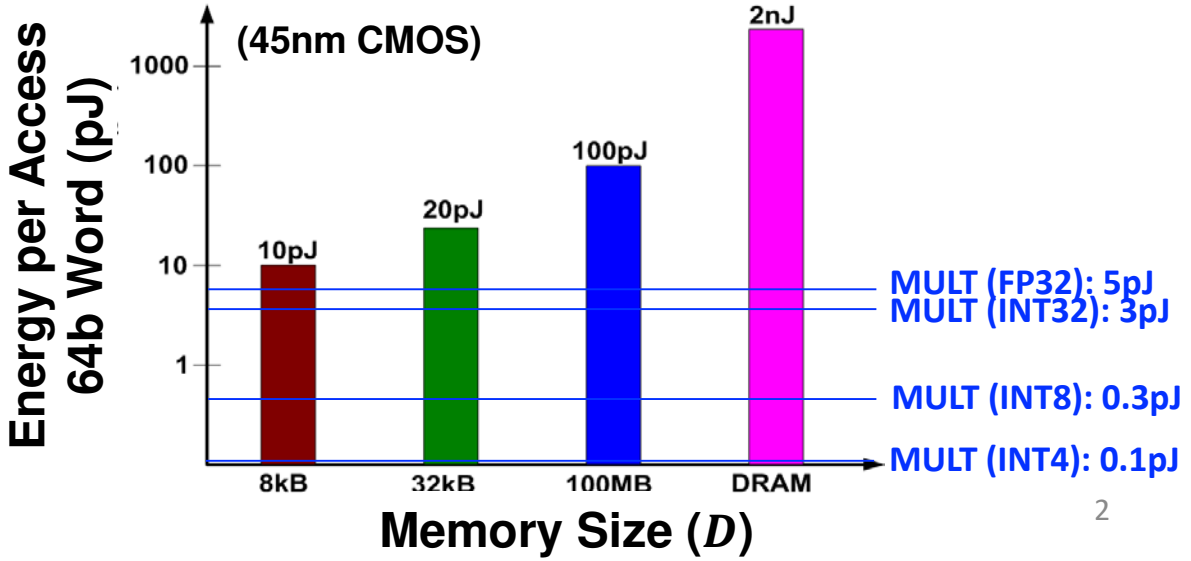
- 10-100× gains in compute energy efficiency & speed





- … BUT, not data movement & memory accessing



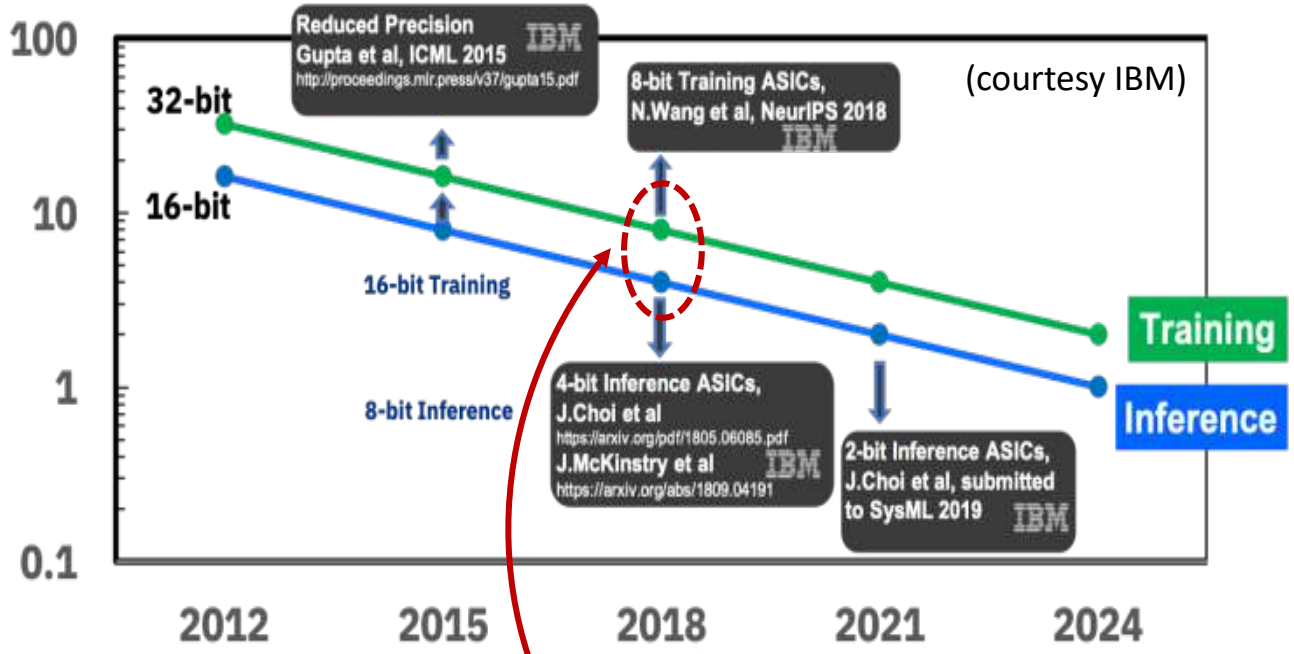


Focusing on Embedded Memory (SRAM)

- **Even models for Edge AI can be large**

Data Type	Application	Model	# Params
Vision	Image classification	ResNet-50	26M
	Object detection	SSD300	24M
Language	Keyword spotting	Deep Speech 2	38M
	Machine translation	Base Transformer	50M

- **…BUT, reducing bit precision makes feasible to store on chip**



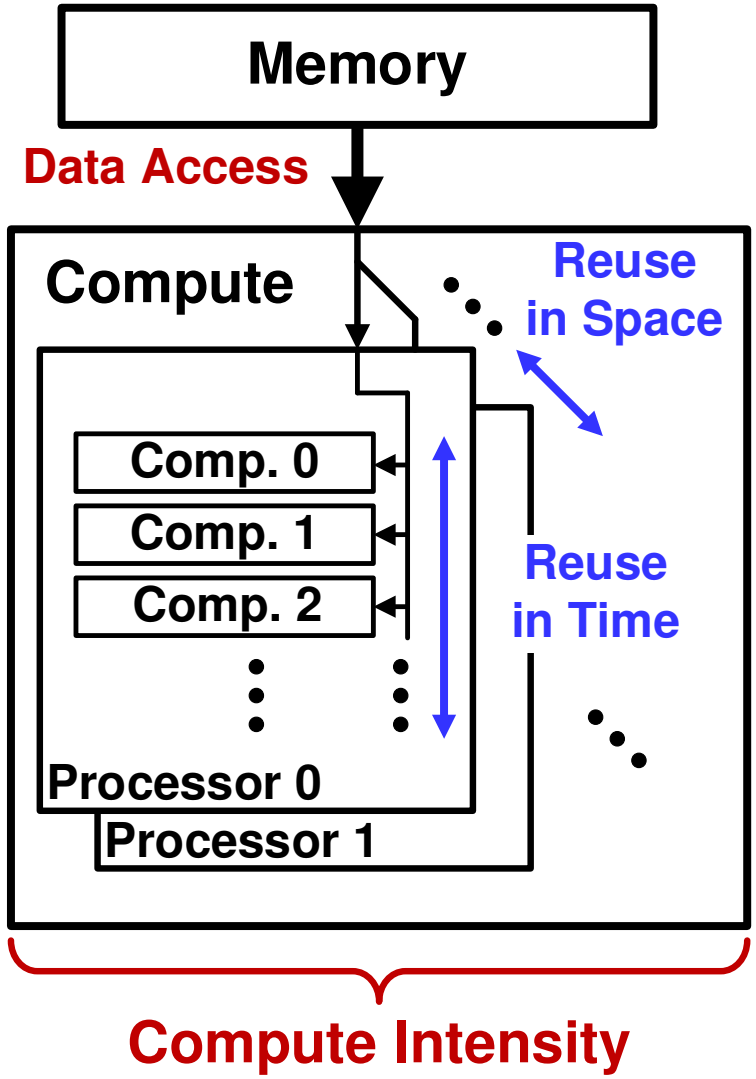
28nm: <20mm² embedded SRAM
16nm: <12mm² embedded SRAM

Outline

1. **Basics of In-Memory Computing (IMC)**
   - Amortizing data movement
2. **IMC challenges**
   - Compute SNR due to analog operation
3. **High-SNR capacitor-based (charge-domain) IMC**
4. **Programmable heterogeneous IMC architecture**
   - Bit-scalability, integration in memory space, near-memory data path
5. **Prototype measurements**
   - Chip measurements, SW libraries, neural-network demonstrations

Typical Ways to Amortize Data Movement

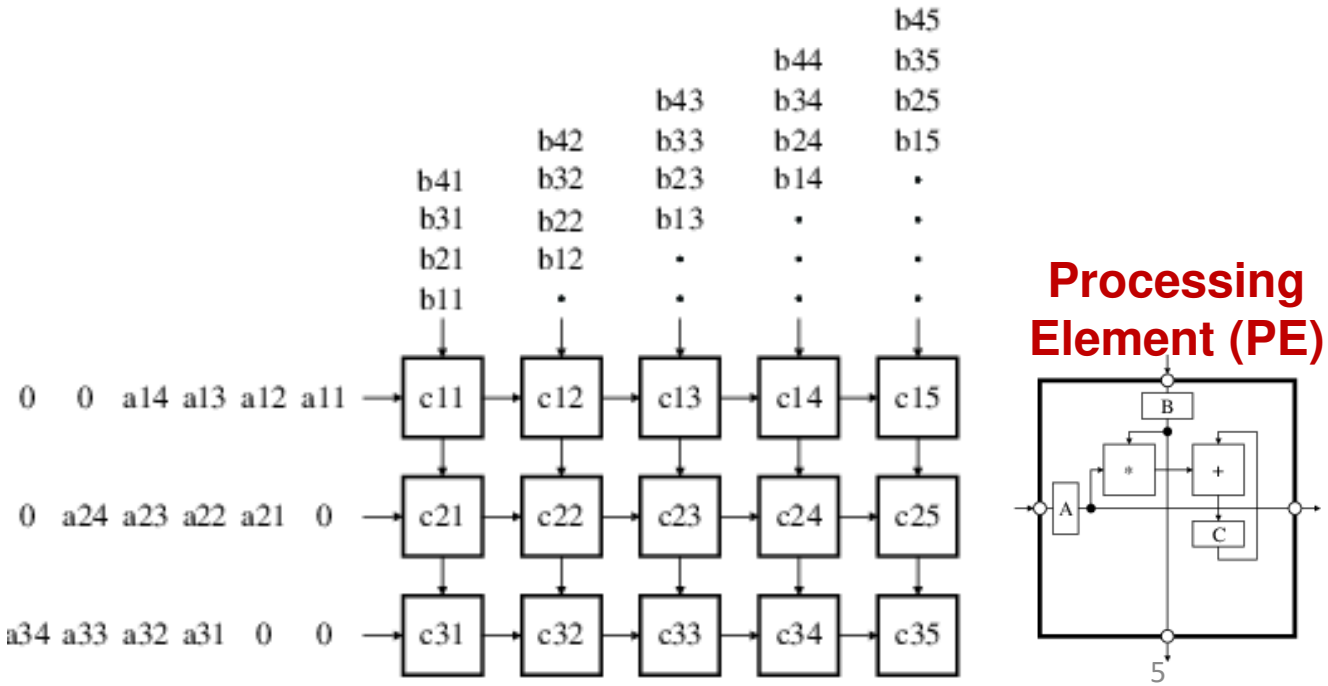
- **Reuse accessed data for compute operations**



- **Specialized (memory-compute integrated) architectures**

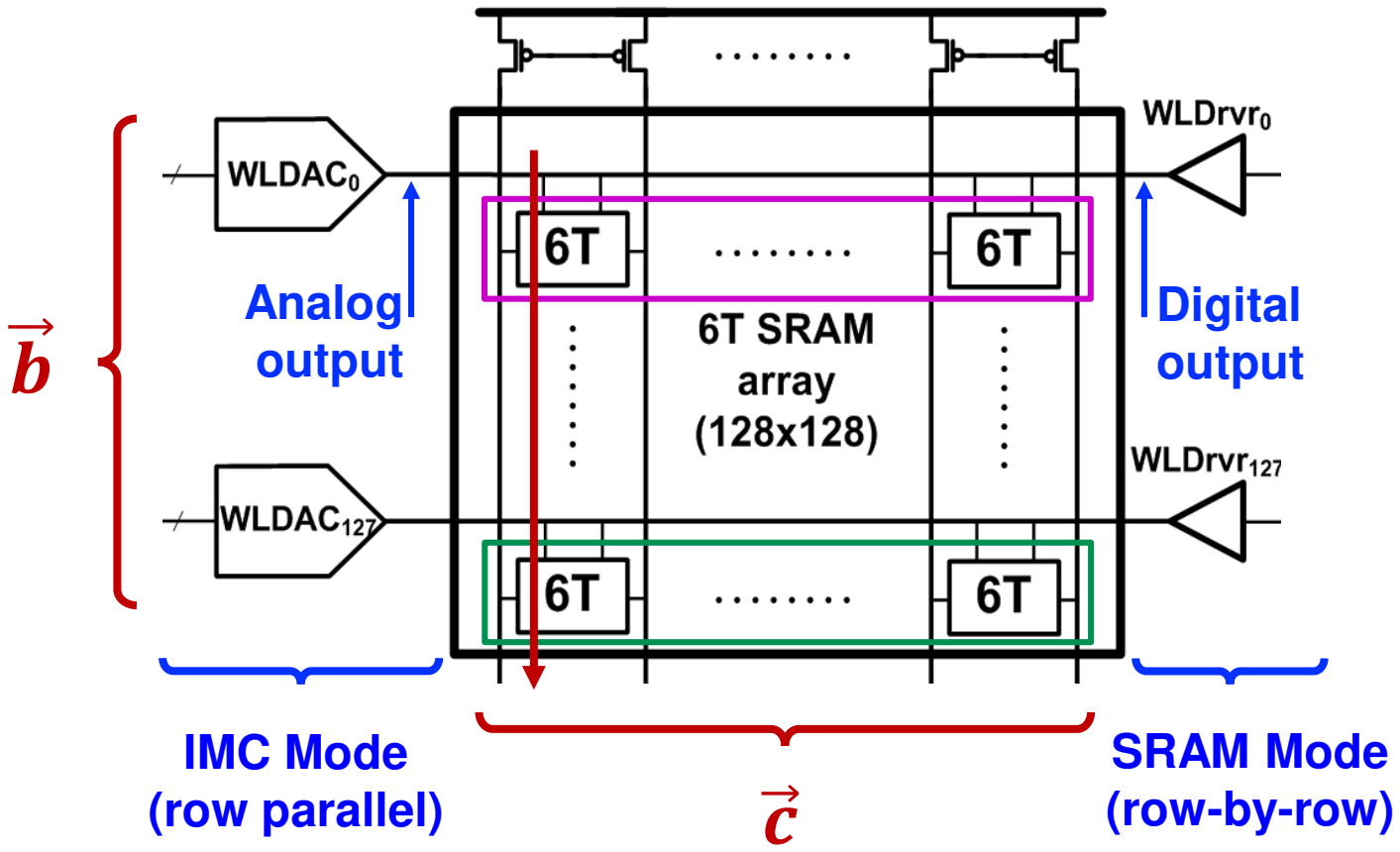
$$\vec{c} = A \times \vec{b}$$

$$\begin{bmatrix} c_1 \\ \vdots \\ c_M \end{bmatrix} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,N} \\ \vdots & \ddots & \vdots \\ a_{M,1} & \cdots & a_{M,N} \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix}$$

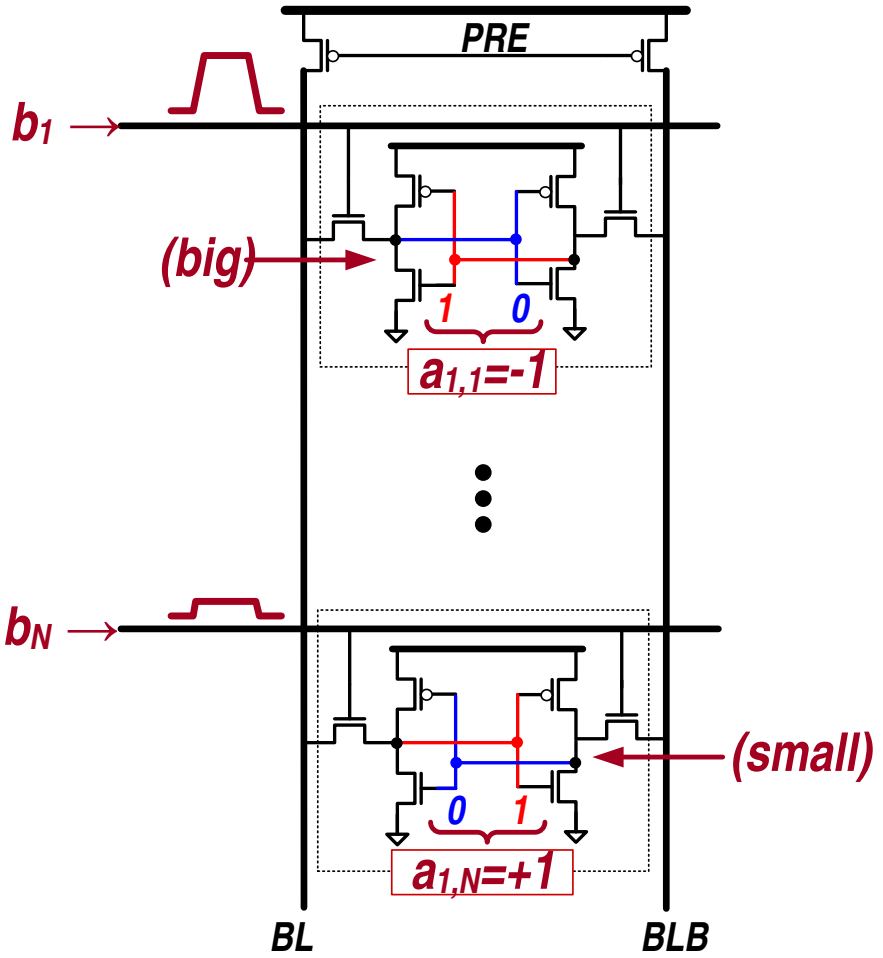


In-Memory Computing (IMC) to Amortize Data Movement

$$\vec{c} = A \times \vec{b} \Rightarrow \begin{bmatrix} c_1 \\ \vdots \\ c_M \end{bmatrix} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,N} \\ \vdots & \ddots & \vdots \\ a_{M,1} & \cdots & a_{M,N} \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix}$$



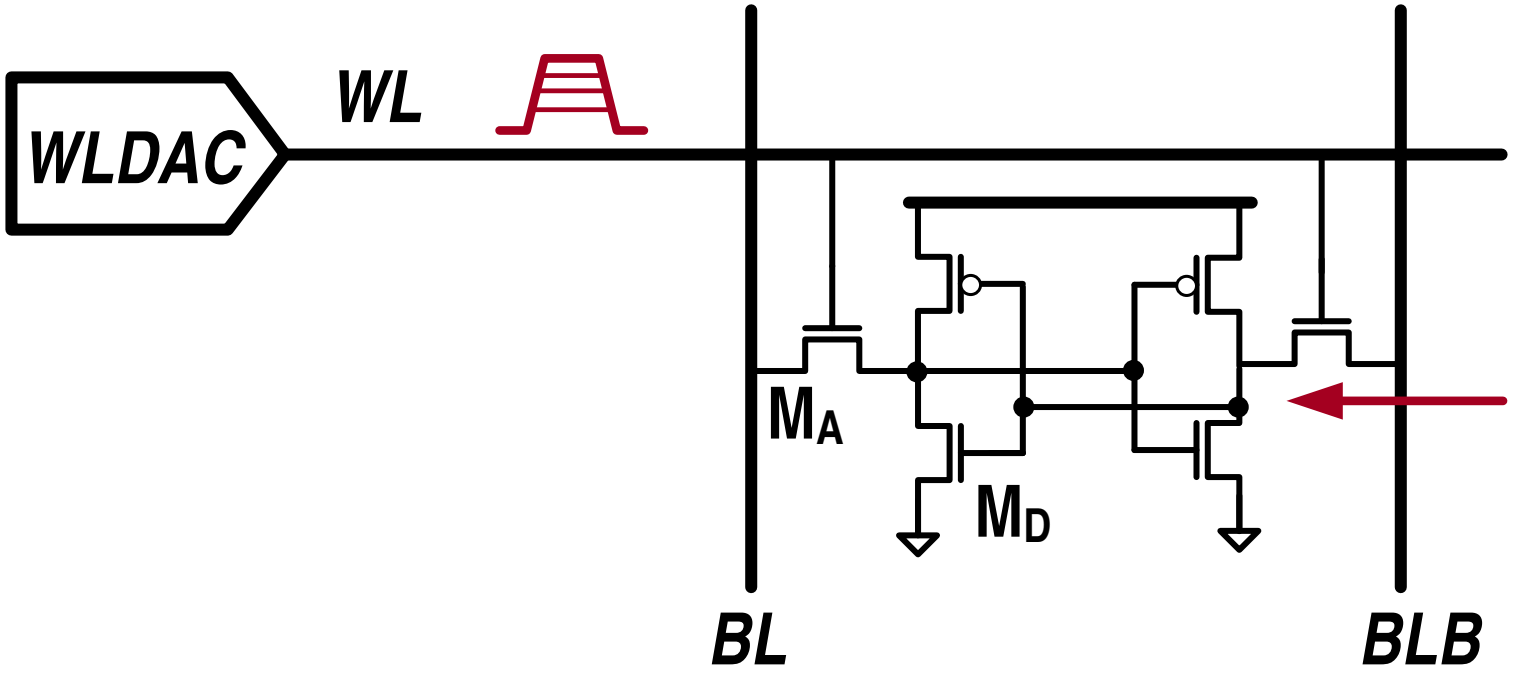
[J. Zhang, *VLSI*'16][J. Zhang, *JSSC*'17]

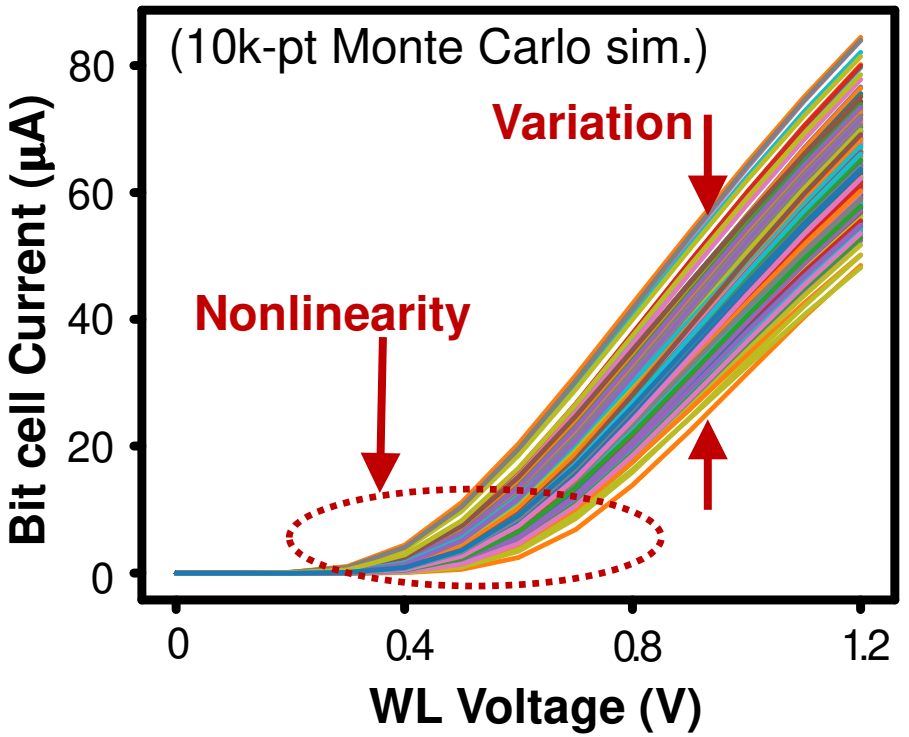


→ **Critical tradeoff:**
energy/throughput vs. SNR

Limited by Analog Computation

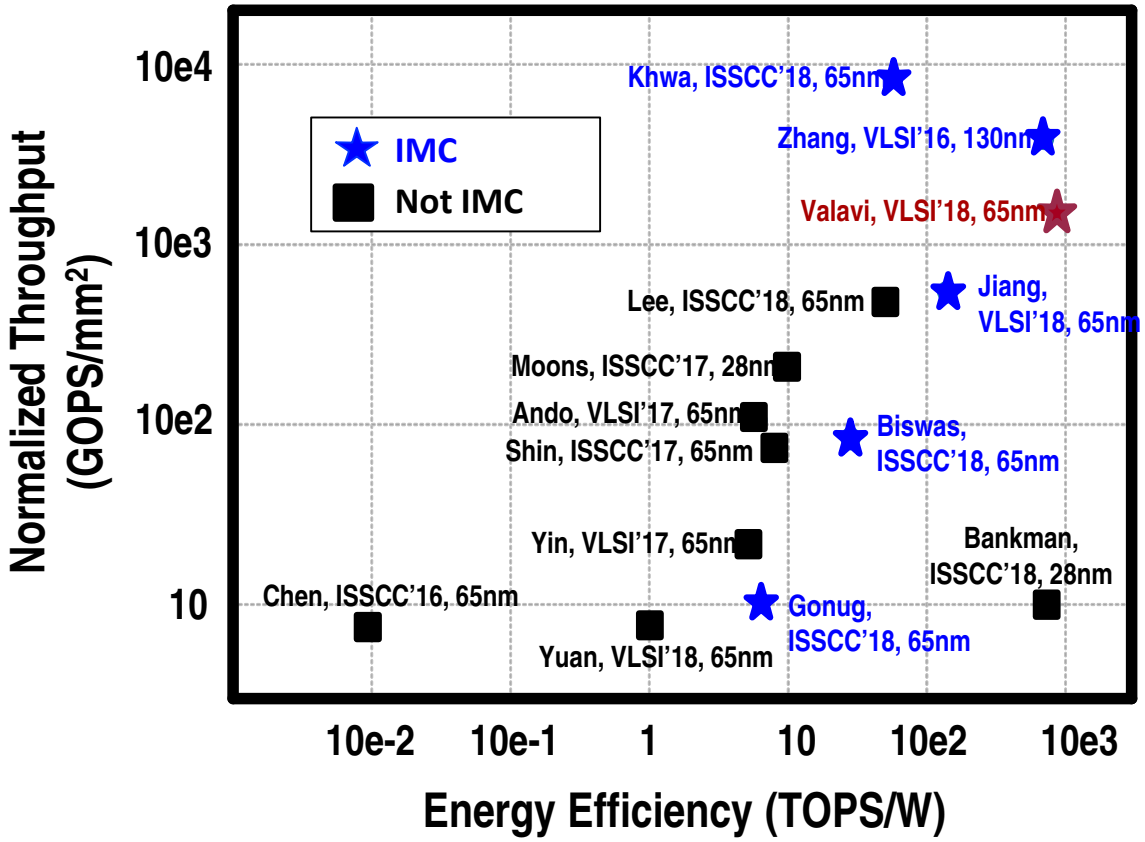
- **Use analog circuits to 'fit' computation in bit cells**
  ⟶ Compute SNR limited by circuit non-idealities (nonlinearity, variation, noise)

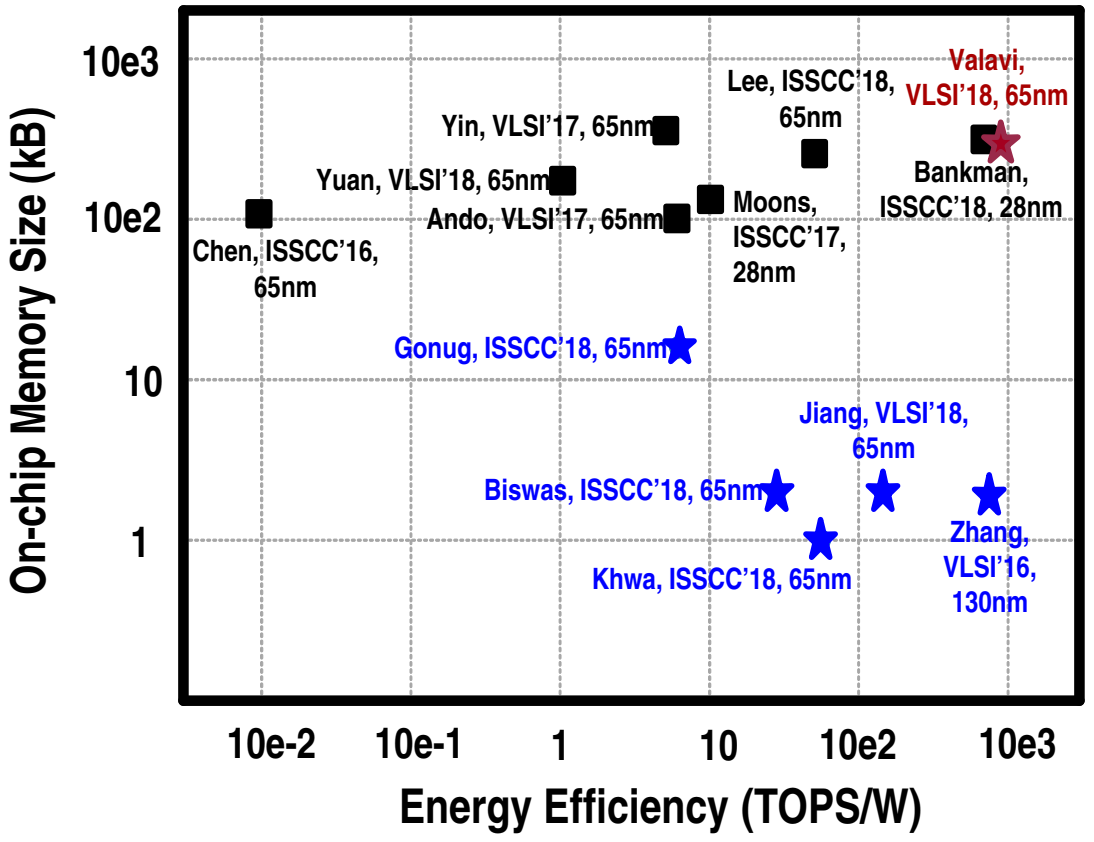




Where does IMC Stand Today?

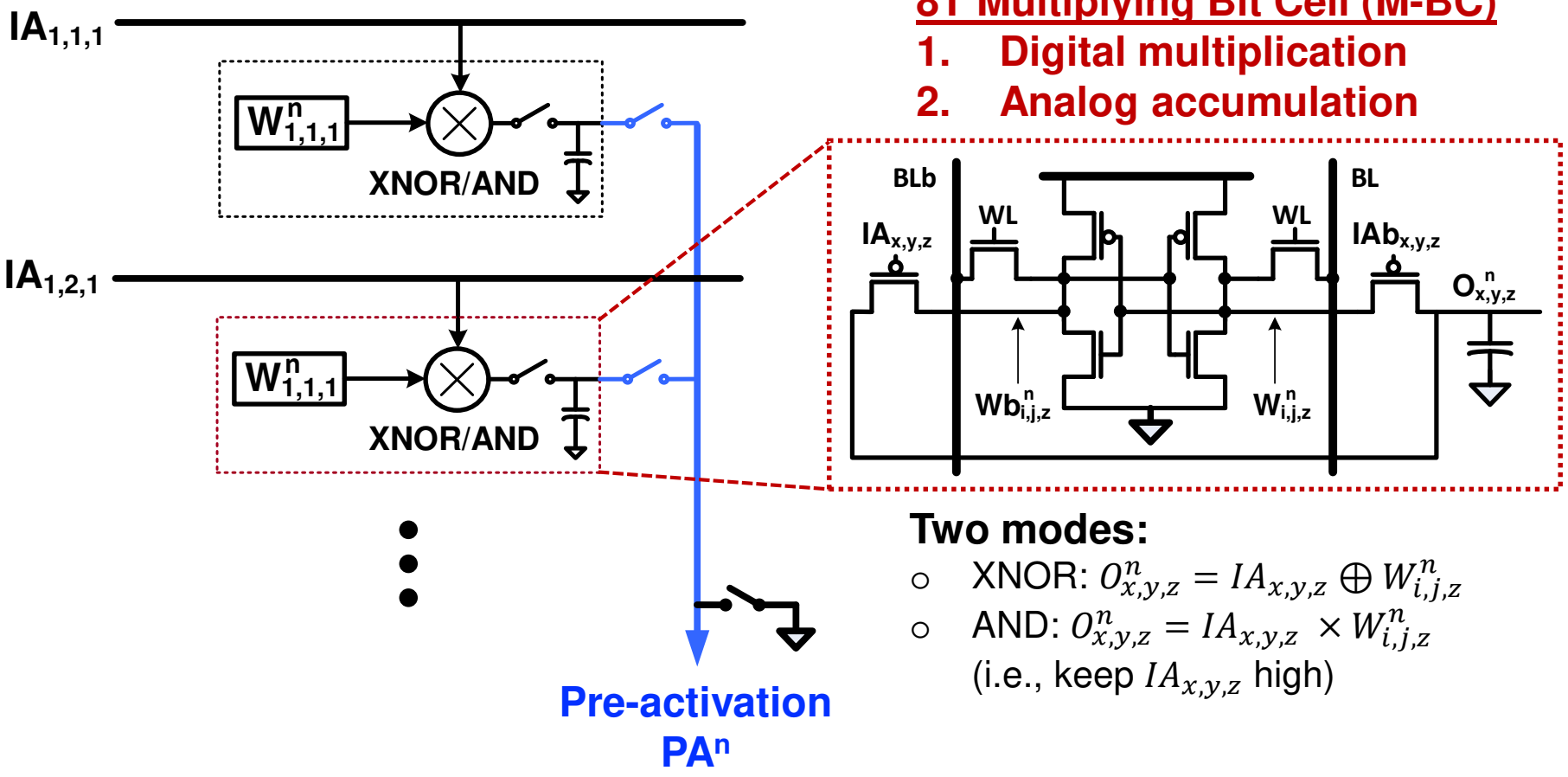
- **Potential for 10× higher energy efficiency & throughput**
- **… BUT, limited scalability (size, workloads, architectures)**





Moving to High-SNR Analog Computation

- **Charge-domain computation based on capacitors**
  ⟶ capacitances set by geometric parameters, well controlled in advanced nodes

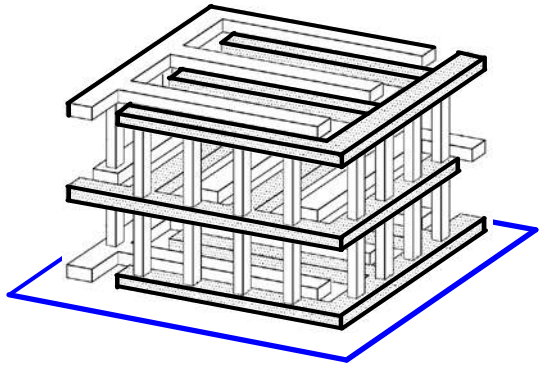


8T Multiplying Bit Cell (M-BC)

1. **Digital multiplication**
2. **Analog accumulation**

Two modes:

- XNOR: $O^n_{x,y,z} = IA_{x,y,z} \oplus W^n_{i,j,z}$
- AND: $O^n_{x,y,z} = IA_{x,y,z} \times W^n_{i,j,z}$
  (i.e., keep $IA_{x,y,z}$ high)

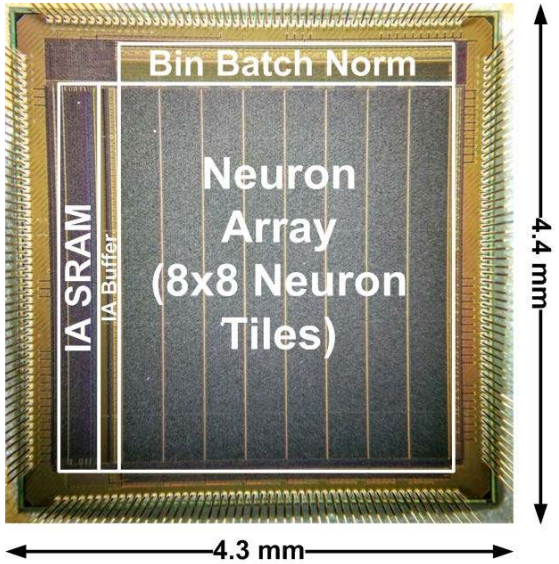


Metal cap. above bit cell

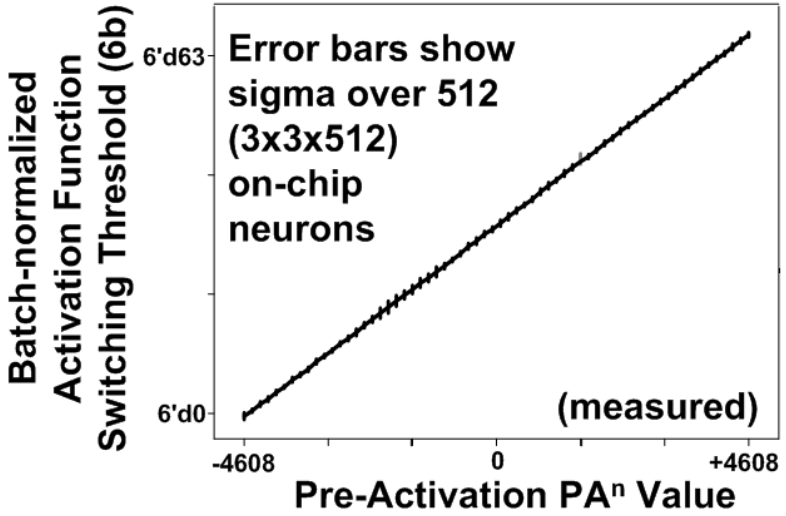
- 1.8× area of 6T cell
- ~10x smaller than equal digital circuit

[H. Valavi, *VLSI'18*]

Previous Demonstration: 2.4Mb, 64-tile IMC



Neuron Transfer Function



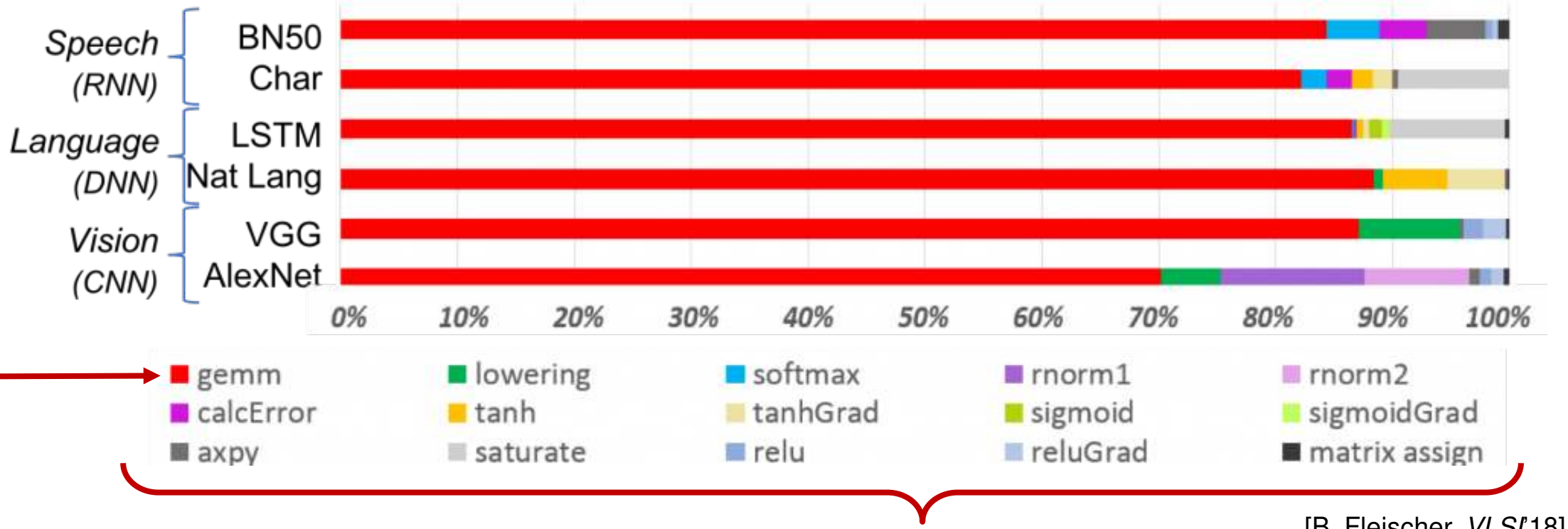
	Moons, ISSCC'17	Bang, ISSCC'17	Ando, VLSI'17	Bankman, ISSCC'18	Valavi, VLSI'18
Technology	28nm	40nm	65nm	28nm	65nm
Area (mm^2)	1.87	7.1	12	6	17.6
Operating VDD	1	0.63-0.9	0.55-1	0.8/0.8 (0.6/0.5)	0.94/0.68/1.2
Bit precision	4-16b	6-32b	1b	1b	1b
on-chip Mem.	128kB	270kB	100kB	328kB	295kB
Throughput (GOPS)	400	108	1264	400 (60)	18,876
TOPS/W	10	0.384	6	532 (772)	866

- 10-layer CNN demos for MNIST/CIFAR-10/SVHN at energies of 0.8/3.55/3.55 µJ/image
- Equivalent performance to software implementation

[H. Valavi, *VLSI'18*]

Need Programmable Heterogeneous Architectures

- **Matrix-vector multiply is only 70-90% of operations**
  ⟶ IMC must integrate in programmable, heterogenous architectures

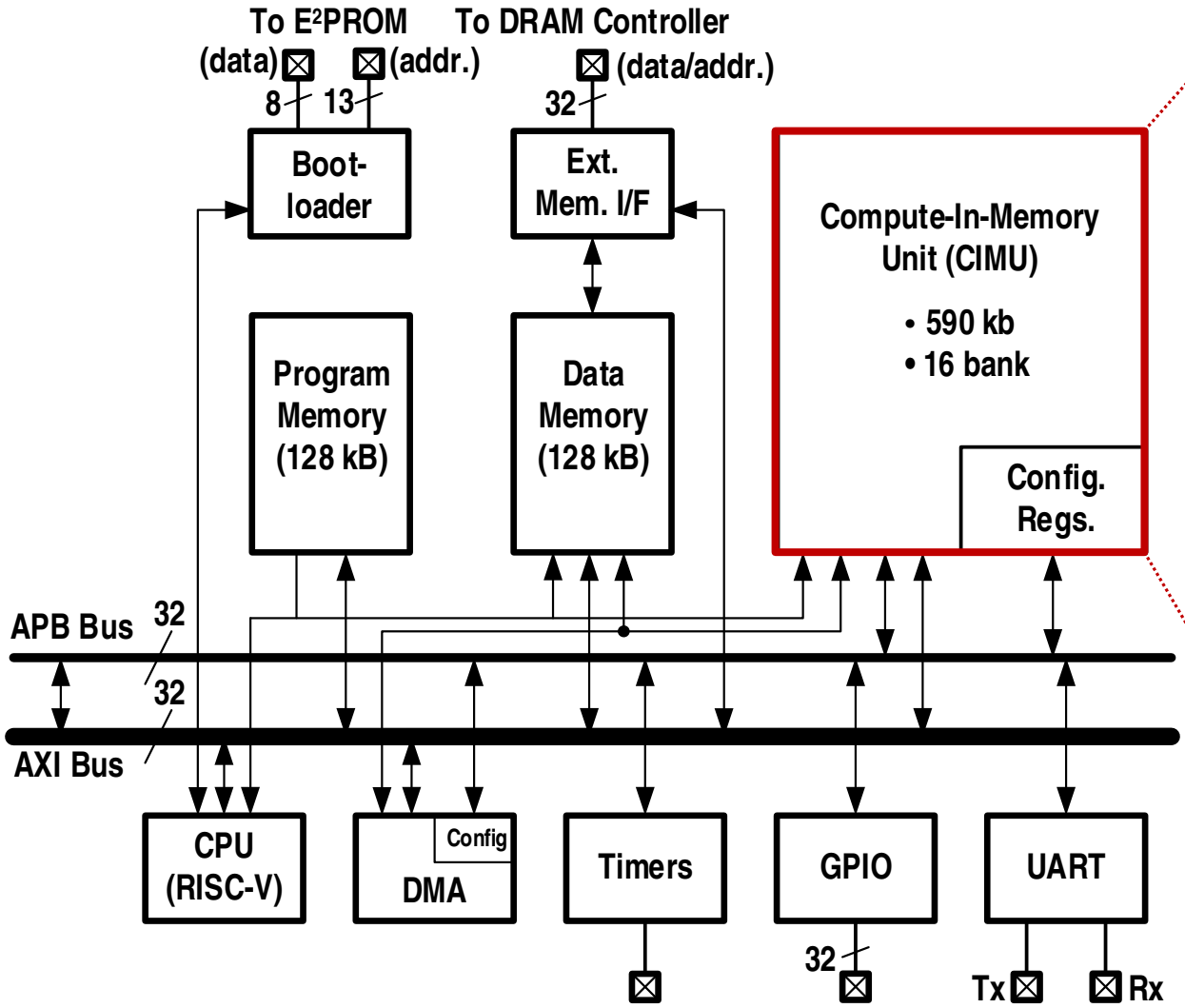


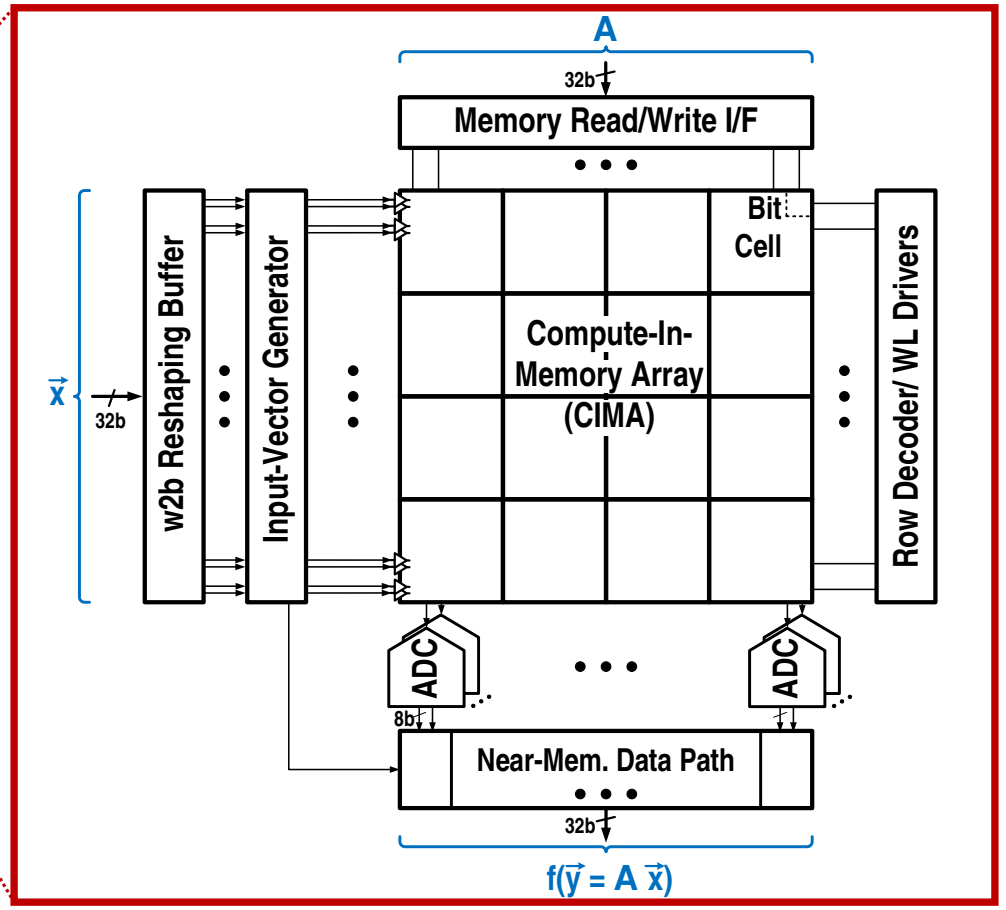
[B. Fleischer, *VLSI*'18]

General matrix multiply with many elements (IMC)

Single/few-word operands (traditional, near-mem. acceleration)

Programmable IMC

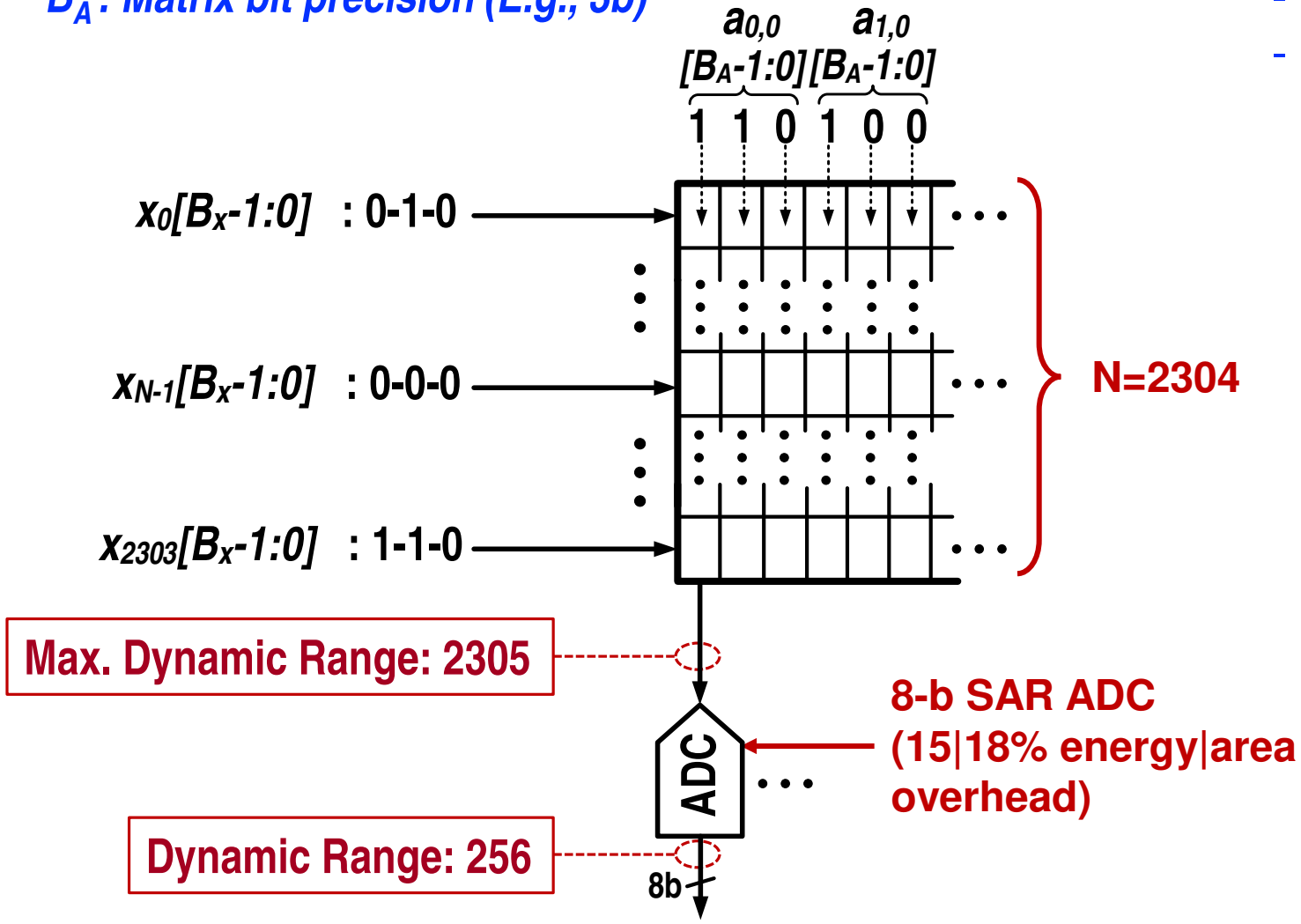




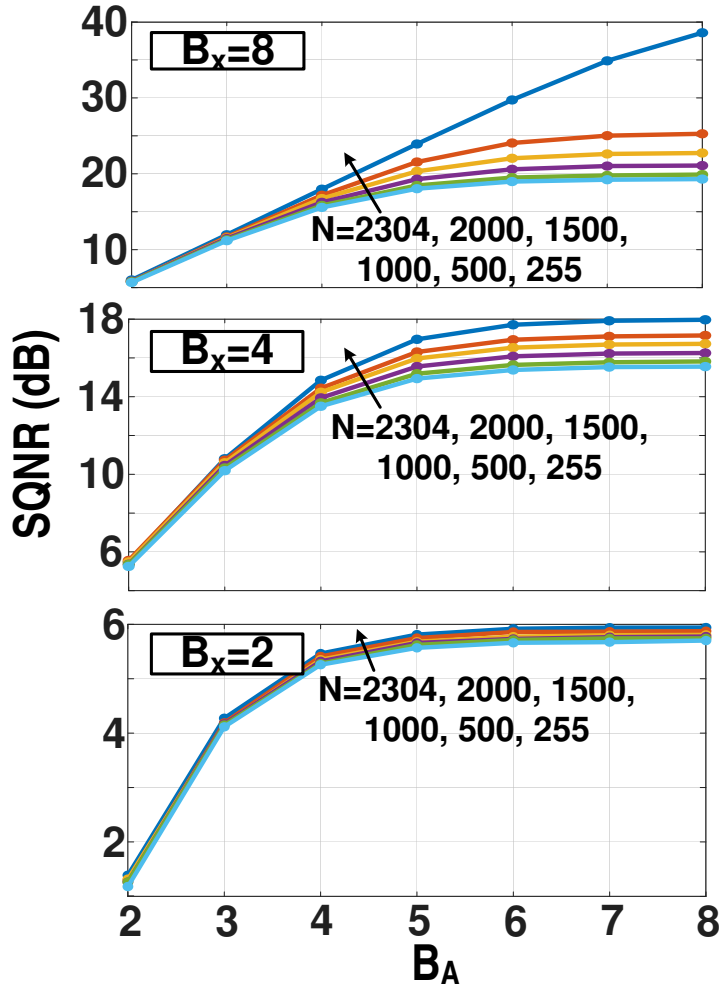
1. Interfaces to standard processor memory space
2. Digital near-mem. accelerator (element compute)
3. Bit scalability from 1 to 8 bits

Bit-Parallel/Bit-Serial (BP/BS) Multi-bit IMC

B_X : Input-vector bit precision (E.g., 3b)
B_A : Matrix bit precision (E.g., 3b)

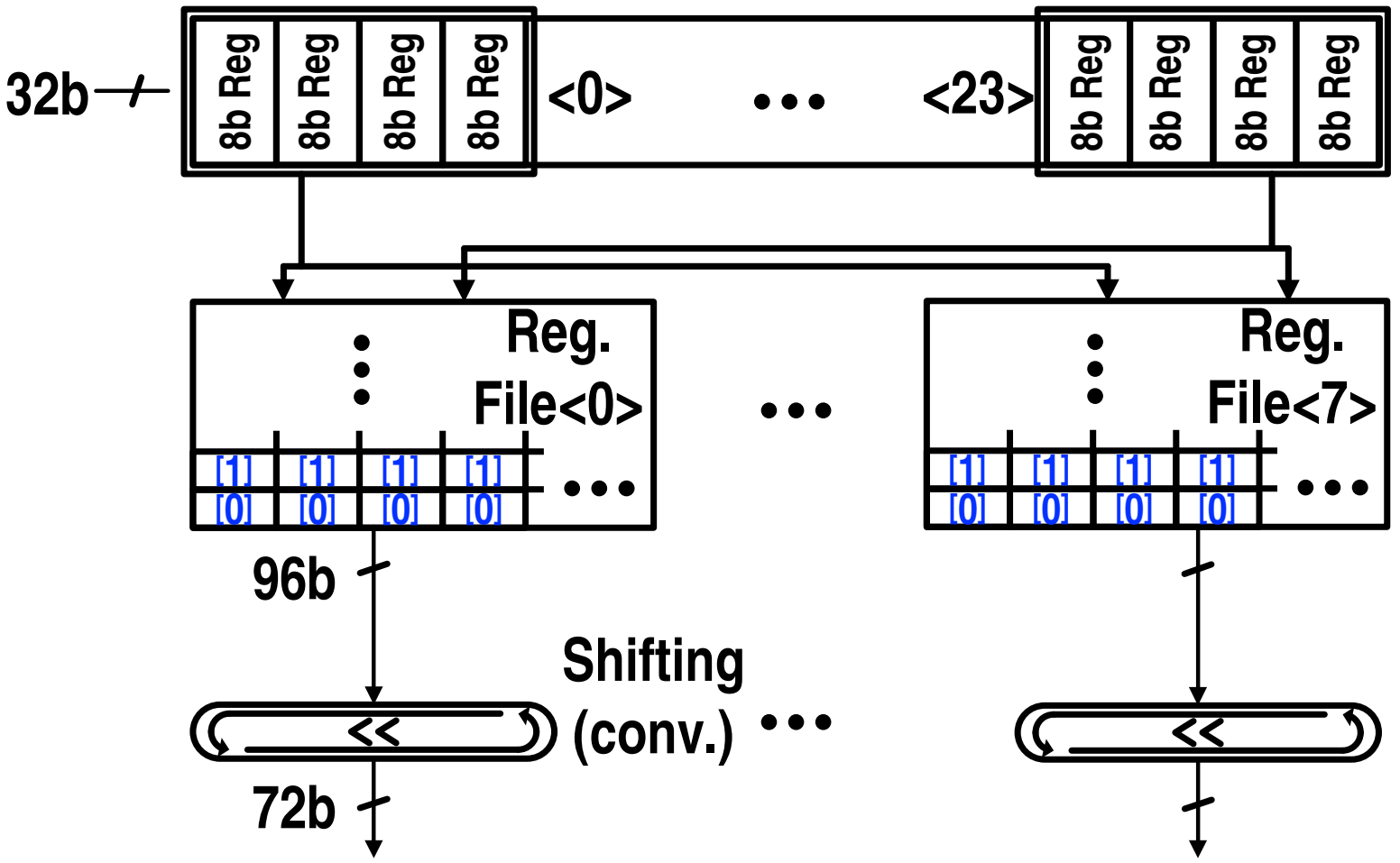


- **SQNR different that standard INT compute**
  - rounding effects are well modeled
  - SQNR is high at precisions of interest



Word-to-bit (w2b) Reshaping Buffer

(B_x : Input-vector bit precision, E.g., 2b)



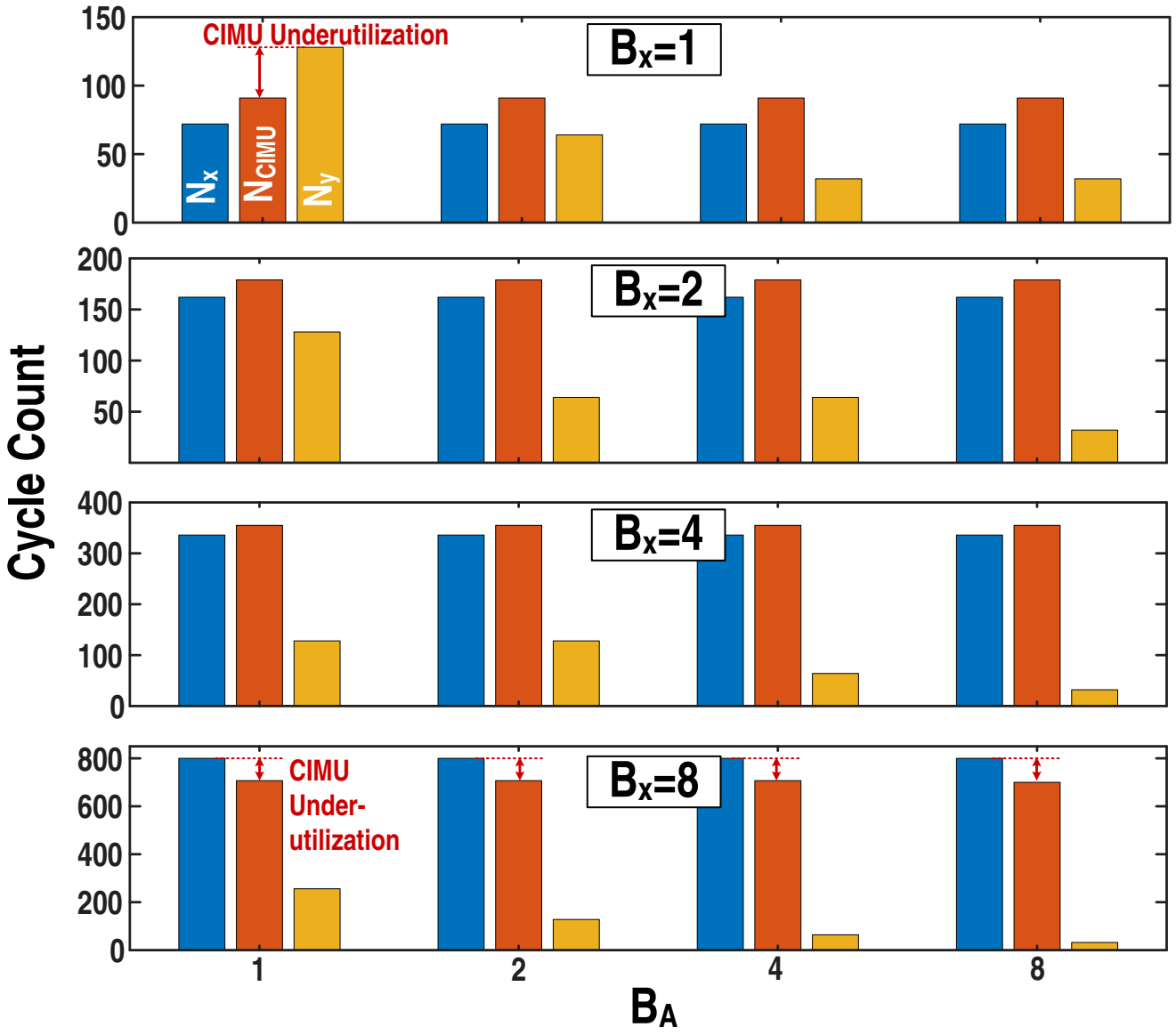
Highly configurable connection network

Sequenced data for bit-serial compute

Circular interface for configurable convolutional striding

Sparsity-proportional energy savings

Data-transfer Analysis



N_x: number of cycles to transfer $\vec{x}$

N_{CIMU}: number of cycles for CIMU compute

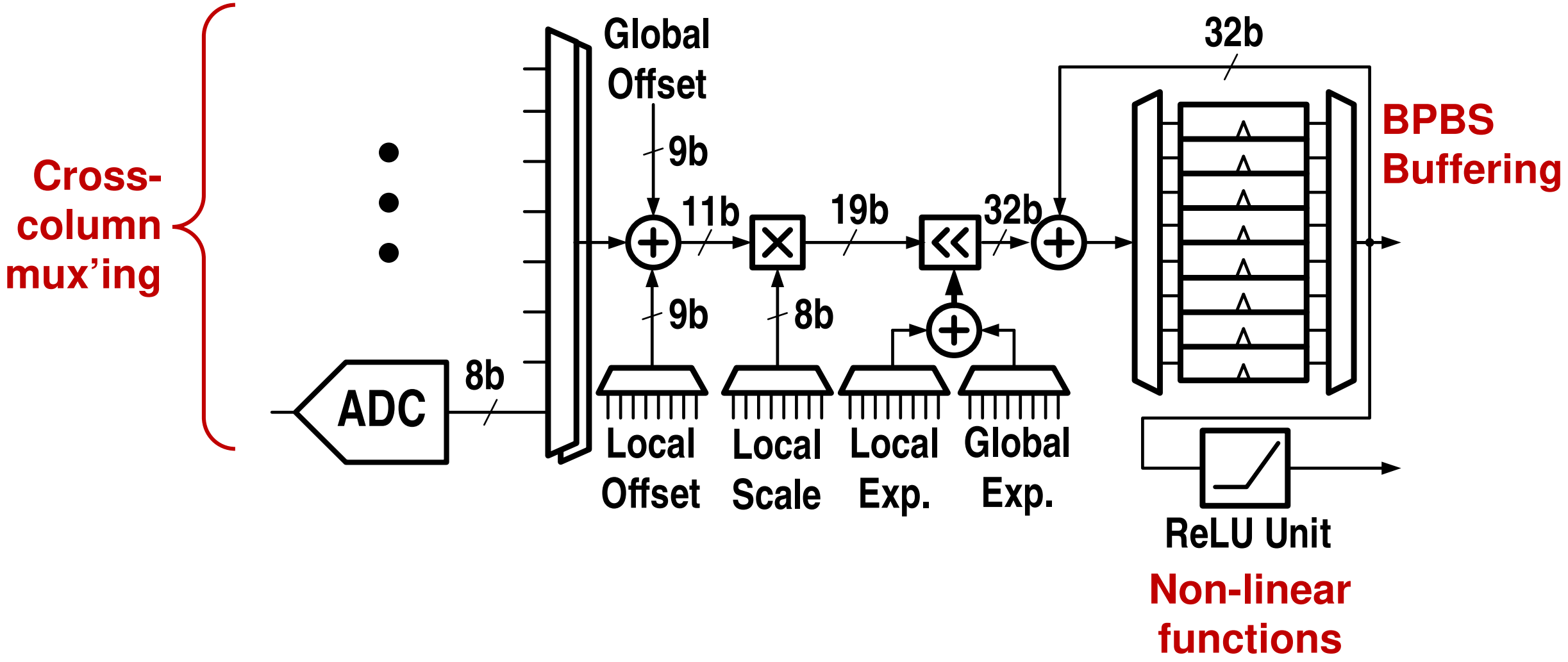
N_y: number of cycles to transfer $\vec{y}$

B_X: bit precision of $\vec{x}$ elements

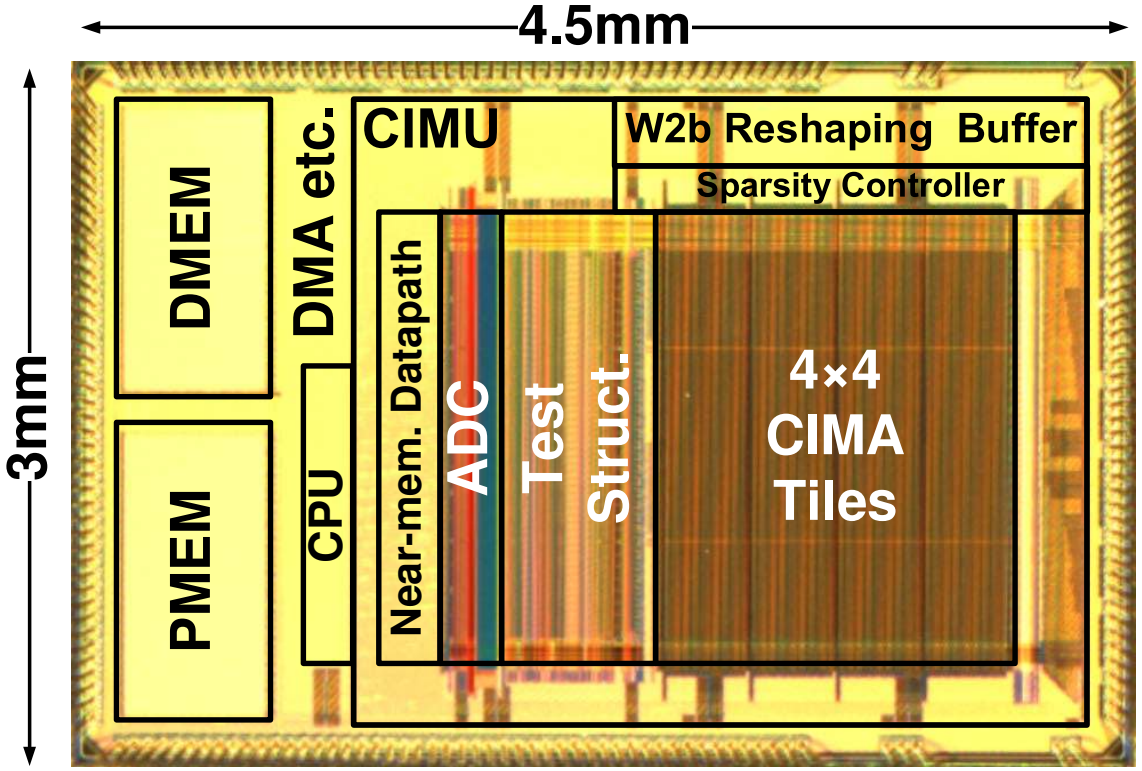
B_A: bit precision of A elements

(33k cycles to load the whole CIMU memory)

Near-memory Datapath (NMD)



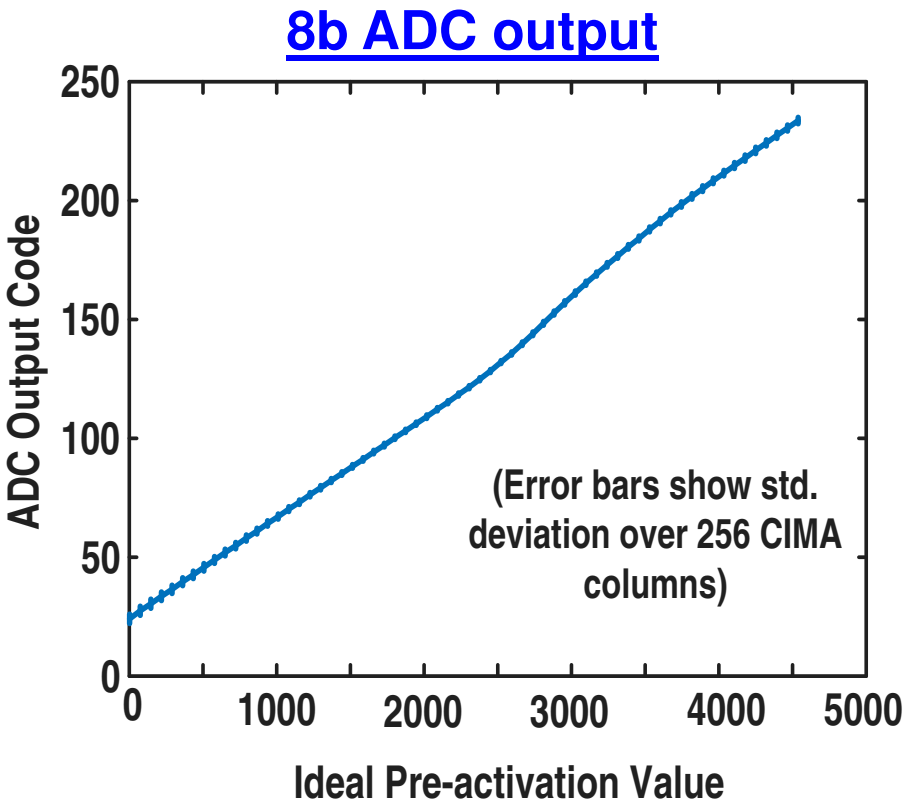
Prototype



Technology (nm)	65
CIMU Area (mm^2)	8.5
V_{DD} (V)	1.2|0.85
On-chip mem. (kB)	74
Bit precision (b)	1-8
Thru.put (1b-GOPS/mm^2)	0.26|0.10
Energy Eff. (1b-TOPS/W)	192|400

- **Recent work has moved to advanced CMOS nodes**
  → Observe energy/density scaling like digital, while maintaining analog precision

Column Computation

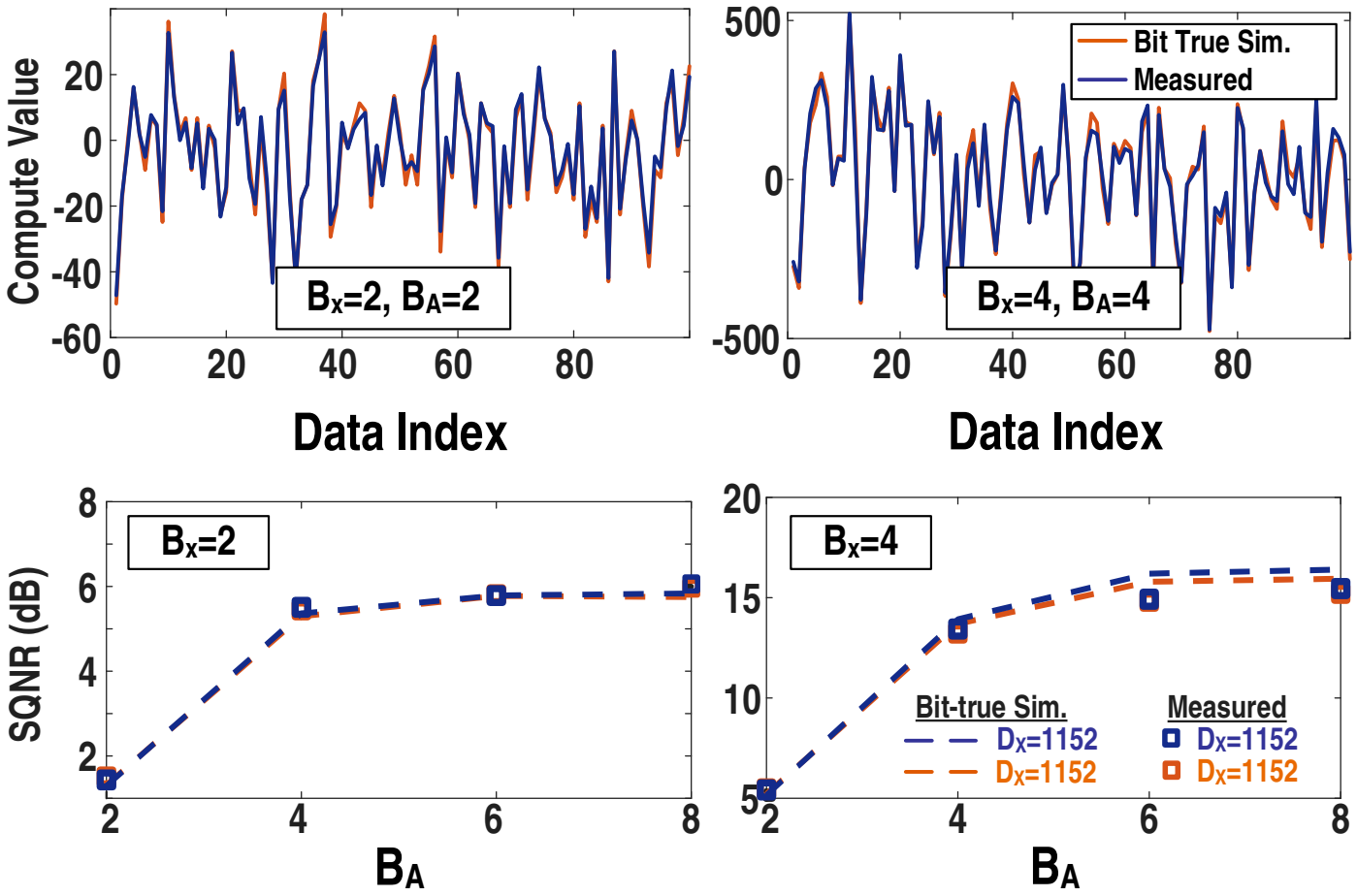


Summary			
Tech. (nm)	65	F_{CLK} (MHz)	100 | 40
V_{DD} (V)	1.2 | 0.7/0.85	Total area (mm^2)	13.5
Energy Breakdown @V_{DD} = 1.2V | 0.7V (P/D MEM, Reshap. Buf), 0.85V (rest)			
CPU (pJ/instru)	52 | 26	CIMA[1] (pJ/column)	20.4 | 9.7
DMA (pJ/32b-transfer)	13.5 | 7.0	ADC[1] (pJ/column)	3.56 | 1.79
Reshap. Buf.[1] (pJ/32b-input)	35 | 12	Dig. Datapath[1] (pJ/output)	14.7 | 8.3

[1]Breakdown within CIMU accelerator

Characterization and Demonstrations

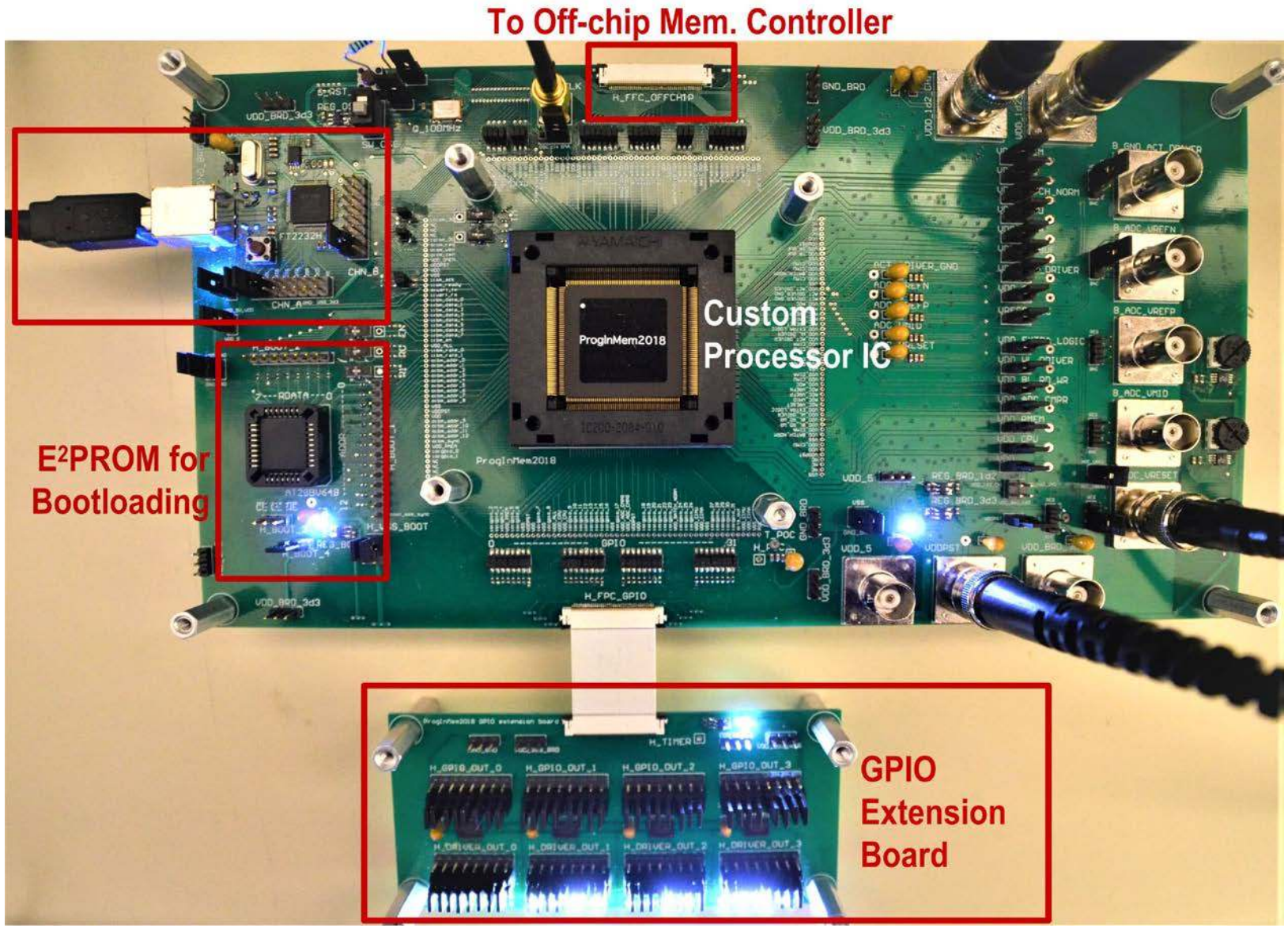
Multi-bit Matrix-Vector Multiplication



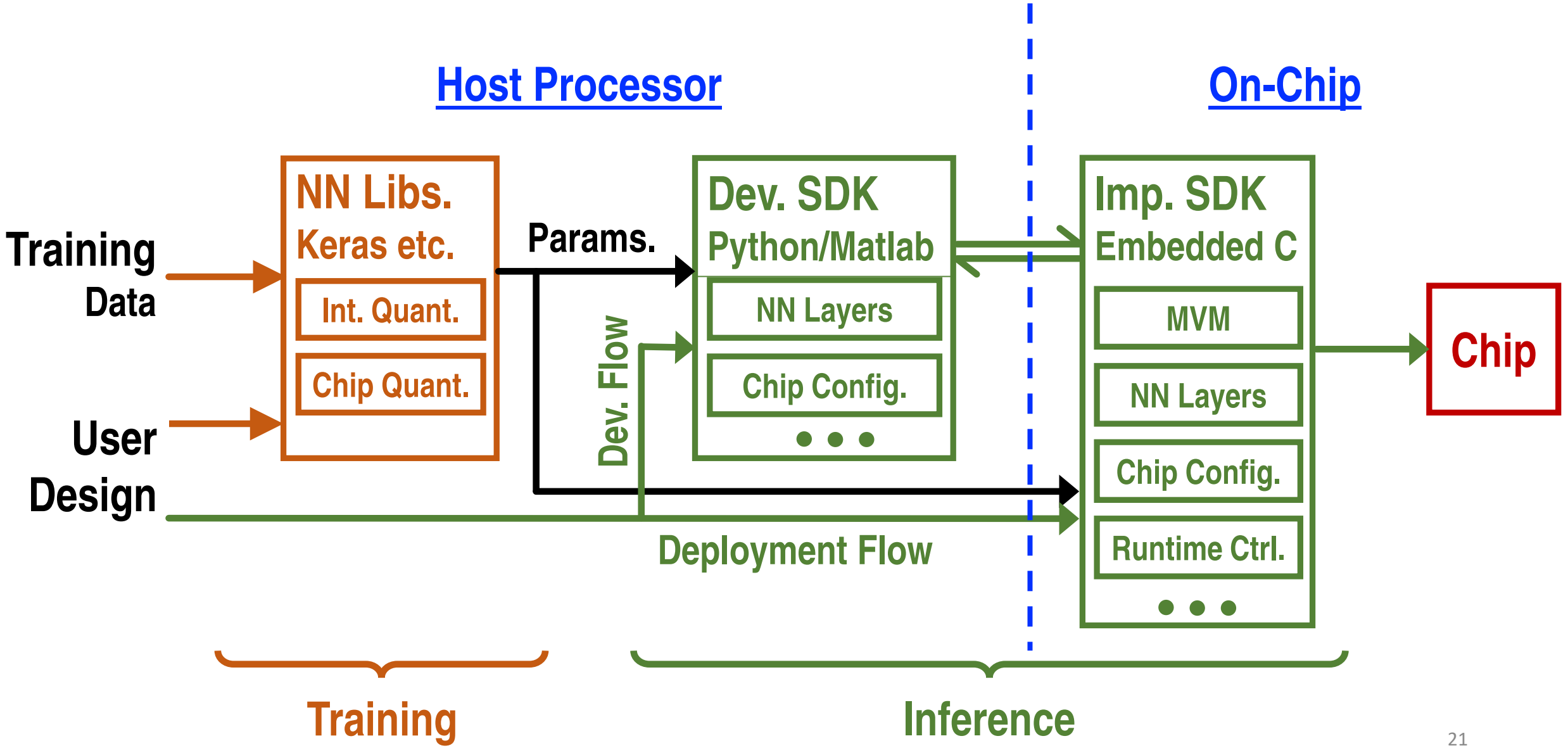
CIFAR-10 Image Classification

Neural-Network Demonstrations		
	Network A (4/4-b activations/weights)	**Network B (1/1-b activations/weights)**
Accuracy of chip (vs. ideal)	92.4% (vs. 92.7%)	89.3% (vs. 89.8%)
Energy/10-way Class.[1]	105.2 µJ	5.31 µJ
Throughput[1]	23 images/sec.	176 images/sec.
Neural Network Topology	L1: 128 CONV3 – Batch norm L2: 128 CONV3 – POOL – Batch norm. L3: 256 CONV3 – Batch. norm L4: 256 CONV3 – POOL – Batch norm. L5: 256 CONV3 – Batch norm. L6: 256 CONV3 – POOL – Batch norm. L7-8: 1024 FC – Batch norm. L9: 10 FC – Batch norm.	L1: 128 CONV3 – Batch Norm. L2: 128 CONV3 – POOL – Batch Norm. L3: 256 CONV3 – Batch Norm. L4: 256 CONV3 – POOL – Batch Norm. L5: 256 CONV3 – Batch Norm. L6: 256 CONV3 – POOL – Batch Norm. L7-8: 1024 FC – Batch norm. L9: 10 FC – Batch norm.

Development board



Application-mapping Flows



Training/Inference Libraries

1. Deep-learning Training Libraries (Keras)

Standard Keras libs:

```
Dense(units, ...)
Conv2D(filters, kernel_size, ...)
...
```

Custom libs:
(INT/CHIP quant.)

```
QuantizedDense(units, nb_input=4, nb_weight=4,
        chip_quant=False, ...)
QuantizedConv2D(filters, kernel_size, nb_input=4,
        nb_weight=4, chip_quant=False, ...)
...
```

```
QuantizedDense(units, nb_input=4, nb_weight=4,
        chip_quant=True, ...)
QuantizedConv2D(filters, kernel_size, nb_input=4,
        nb_weight=4, chip_quant=True, ...)
...
```

2. Deep-learning Inference Libraries (Python, MATLAB, C)

High-level network build (Python):

```
chip_mode = True
outputs = QuantizedConv2D(inputs,
        weights, biases, layer_params)
outputs = BatchNormalization(inputs,
        layer_params)
...
```

Function calls to chip (Python):

```
chip.load_config(num_tiles, nb_input=4,
        nb_weight=4)
chip.load_weights(weights2load)
chip.load_image(image2load)
outputs = chip.image_filter()
```

Embedded C:

```
chip_command = get_uart_word();
chip_config();
load_weights(); load_image();
image_filter(chip_command);
read_dotprod_result(image_filter_command);
```

Conclusions

Matrix-vector multiplies (MVMs) are a little different than other computations

⟶ high-dimensionality operands lead to data movement / memory accessing

Capacitor-based IMC enables high-SNR analog compute

⟶ enables scale and robust functional specification of IMC for architectural design

Programmable IMC is demonstrated (with supporting SW libraries)

⟶ physical IMC tradeoffs will drive specialized mapping/virtualization algorithms

Resent work has moved to advanced nodes

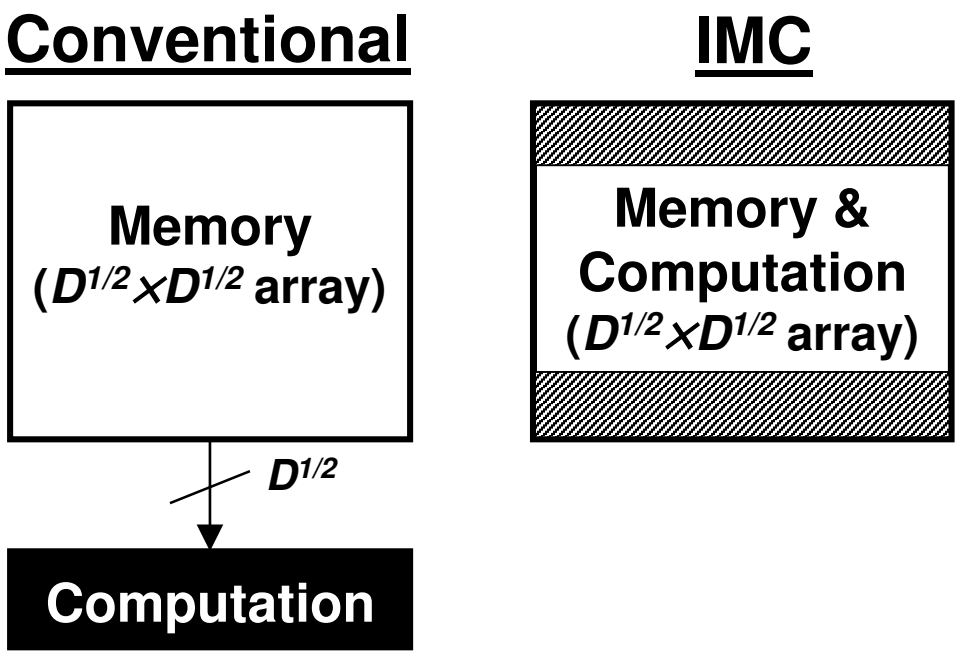
⟶ energy and density scaling similar to standard digital logic

Acknowledgements: funding provided by ADI, DARPA, NRO

Backup

IMC Tradeoffs

CONSIDER: Accessing D bits of data associated with computation, from array with $\sqrt{D}$ columns × $\sqrt{D}$ rows.

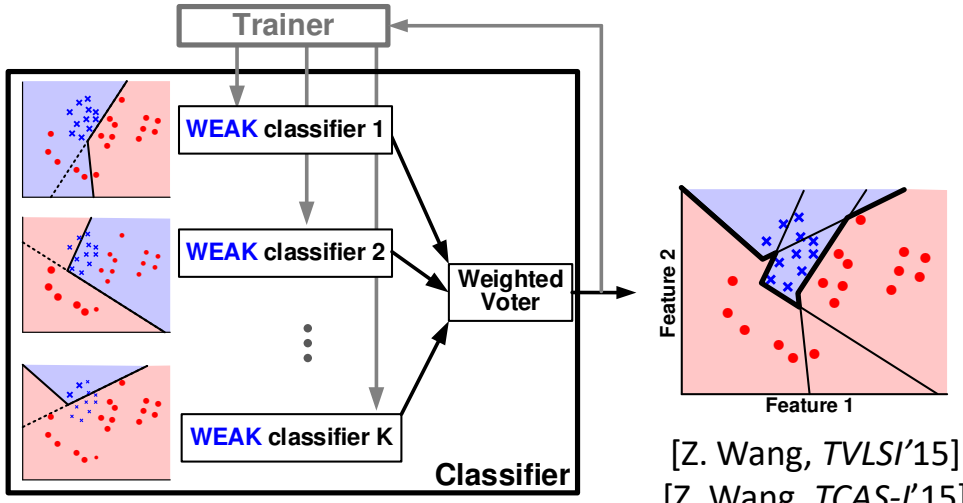


Metric	Conventional	In-memory
Bandwidth	$1/D^{1/2}$	1
Latency	D	1
Energy	$D^{3/2}$	$\sim D$
SNR	1	$\sim 1/D^{1/2}$

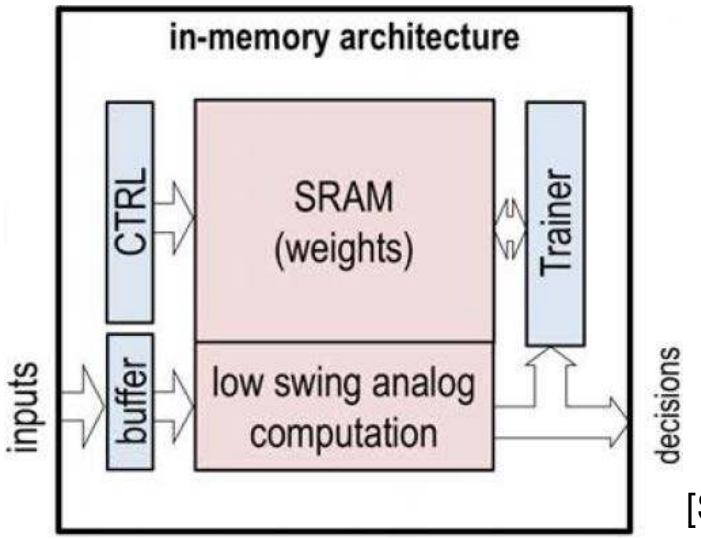
- IMC benefits energy/delay at cost of SNR
- SNR-focused systems design is critical (circuits, architectures, algorithms)

Algorithmic Co-design(?)

- **Chip-specific weight tuning**

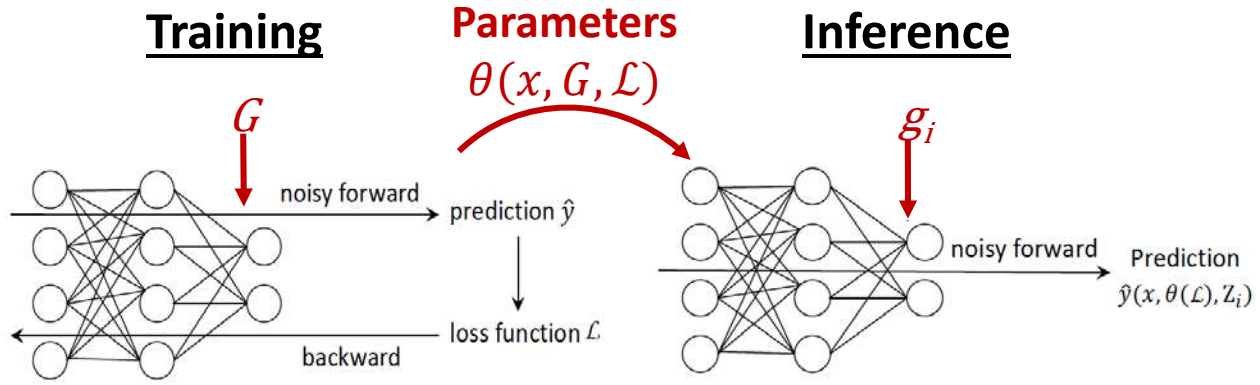


[Z. Wang, *TVLSI*'15]
[Z. Wang, *TCAS-I*'15]

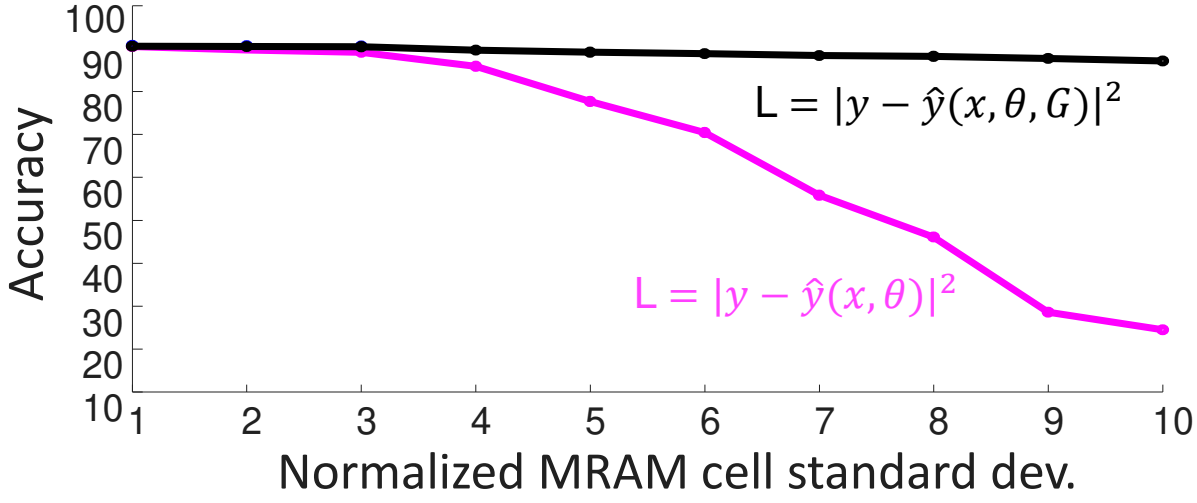


[S. Gonu., *ISSCC*'18]

- **Chip-generalized weight tuning**

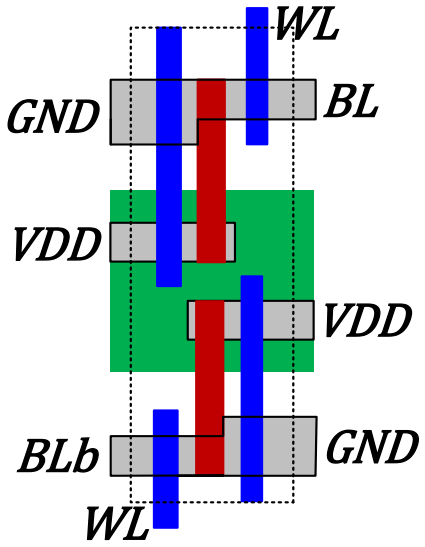


E.g.: BNN Model (applied to CIFAR-10)

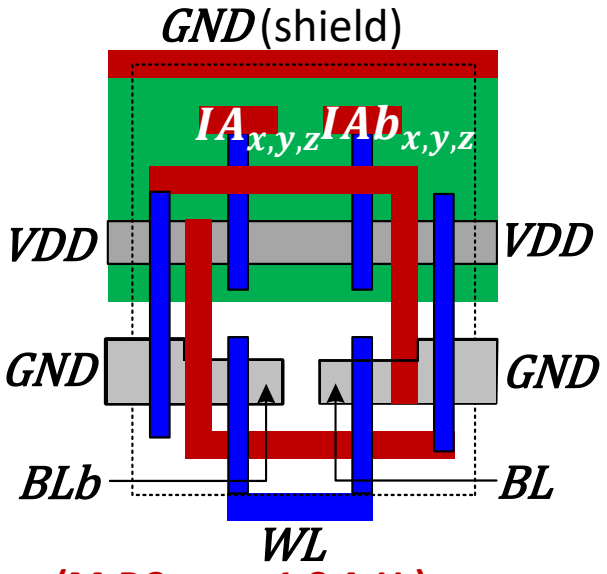


[B. Zhang, *ICASSP* 2019]

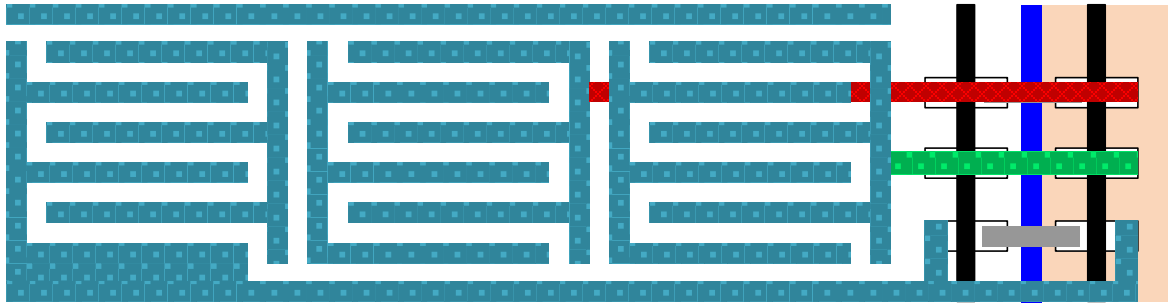
M-BC Layout



(6T area: 1.0 A.U.)

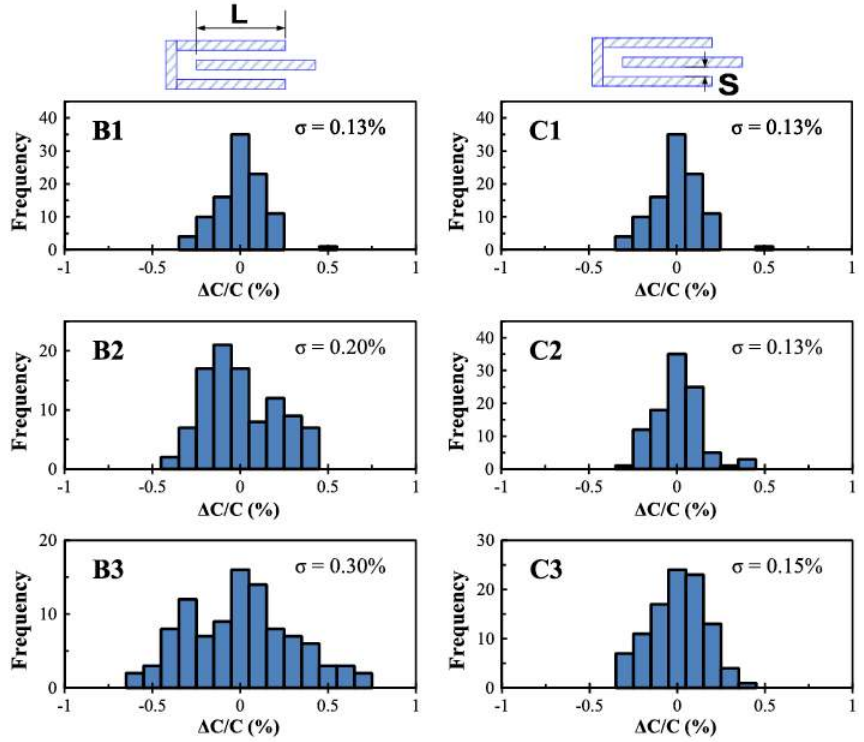


(M-BC area: 1.8 A.U.)



[H. Valavi, *VLSI*'18]

E.g., MOM-capacitor matching (130nm):



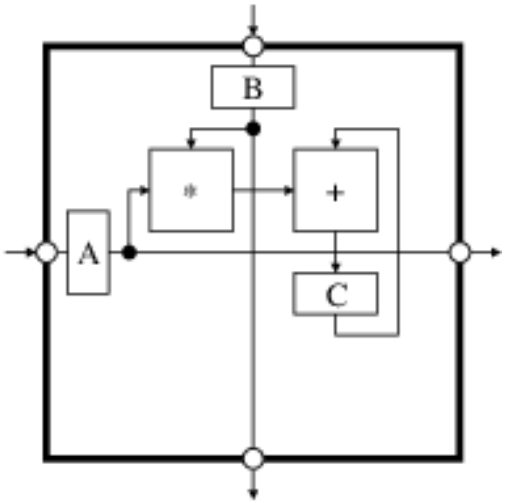
[H. Omran, *TCAS-I*'16]

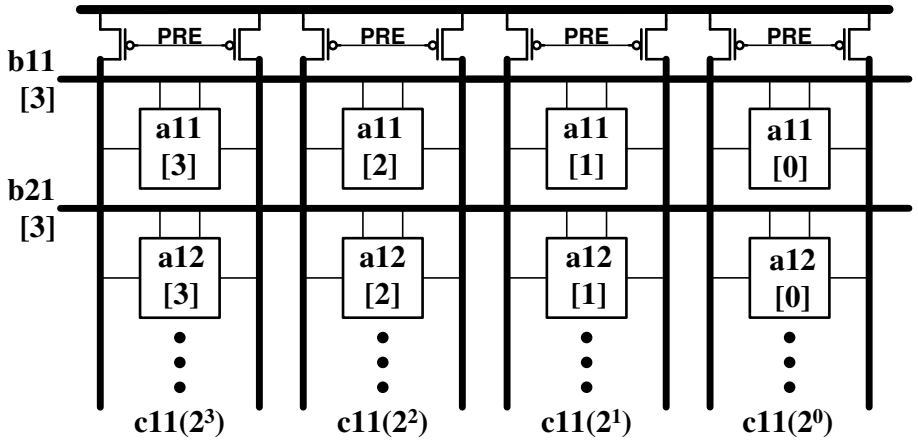
- **>14-b capacitor matching across M-BCs**
- **>14k IMC rows for matching-limited SNR**

IMC as a Spatial Architecture

Assume:

- 1k dimensionality
- 4-b multiplies
- 45nm CMOS



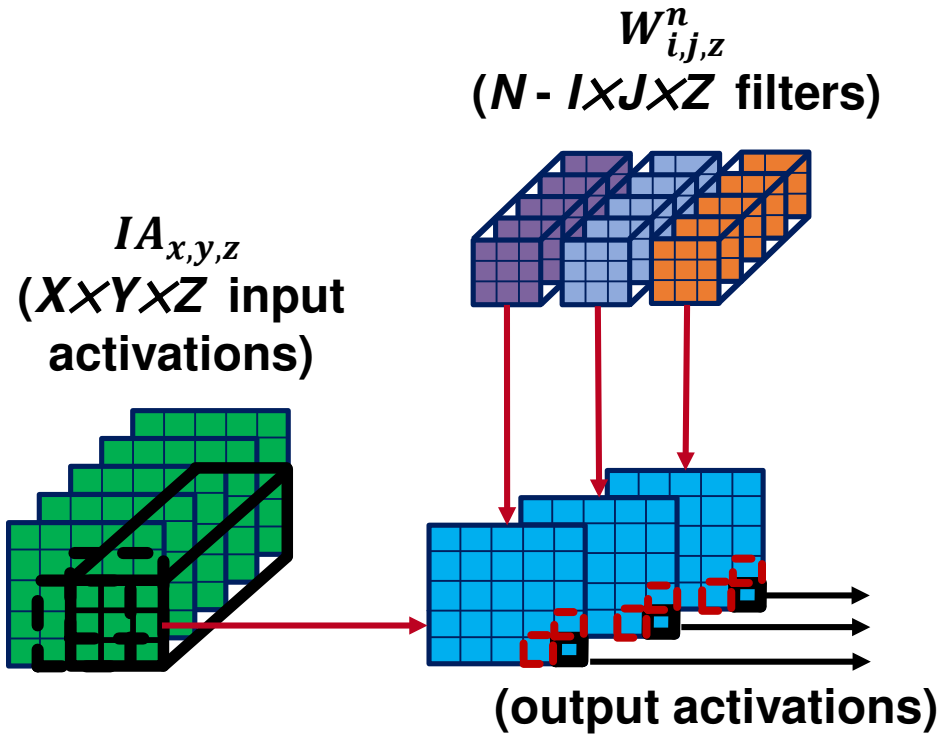


Operation	Digital-PE Energy (fJ)	Bit-cell Energy (fJ)
Storage	250	50
Multiplication	100	
Accumulation	200	
Communication	40	5
Total	**590**	**55**

Application mapping

- **IMC engines must be 'virtualized'**
  ⟶ IMC amortizes MVM costs, not weight loading
  ⟶ Need new mapping algorithms (physical tradeoffs very diff. than digital engines)

Ex.: ***X×Y*** sets reuse of filter weights



Weight Accessing:

- $E_{DRAM→IMC}$/4-bit: 40pJ
- Reuse: $X \times Y$
- $E_{MAC,4\text{-}b}$:50fJ

Activation Accessing:

- $E_{DRAM→IMC}$/4-bit: 40pJ
- Reuse: $N \times I \times J$ (10-20 lyrs
- $E_{MAC,4\text{-}b}$: 50fJ