# Design of Computing-in-Memory: Analog vs. Digital

**Tony Tae-Hyoung Kim**

**School of Electrical and Electronic Engineering**

**Nanyang Technological University**

# Outline

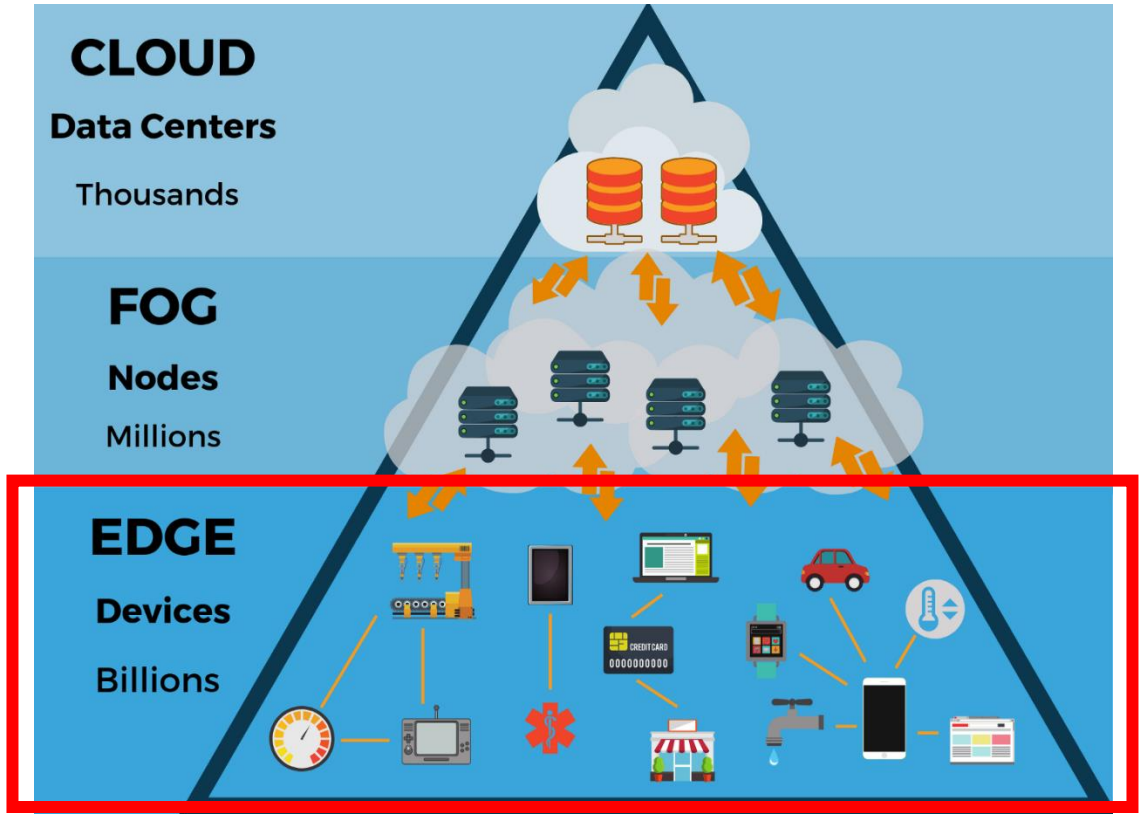
- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
- Summary

# Outline

- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
- Summary

# Machine Learning for Edge Devices

- Tiny machine learning for IoT devices has wide applications.




[Source: eBizSolutions]

# Ubiquitous IoT Devices and Embedded ML

[MIT HAN Lab]

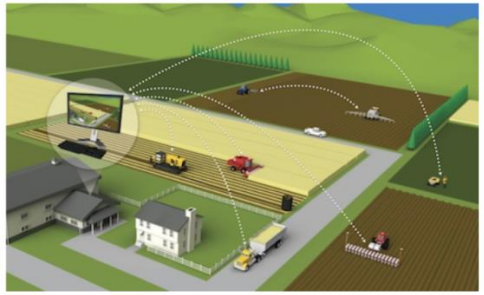
Smart Retail


Personalized Healthcare

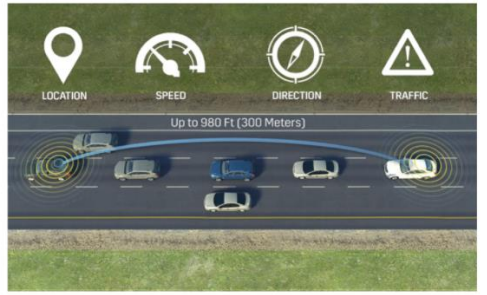
Smart Home


Smart Manufacturing

Precision Agriculture



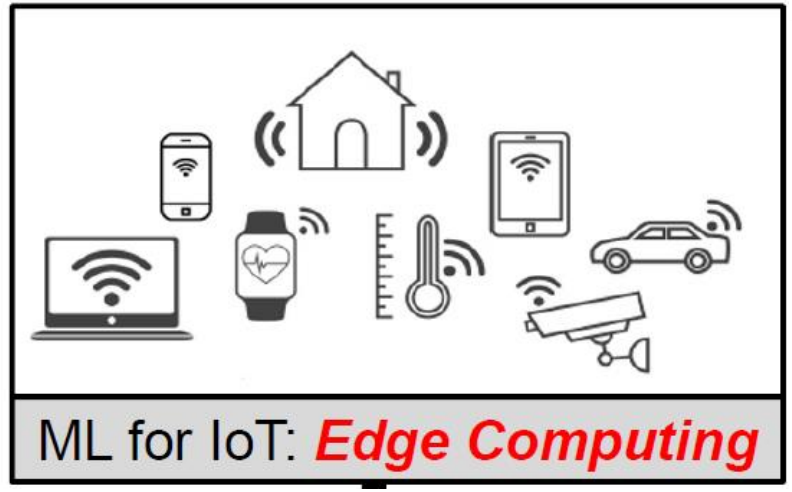
Autonomous Driving




- IoT devices and embedded ML models increasingly ubiquitous in the world

# Machine Learning for Edge Computing

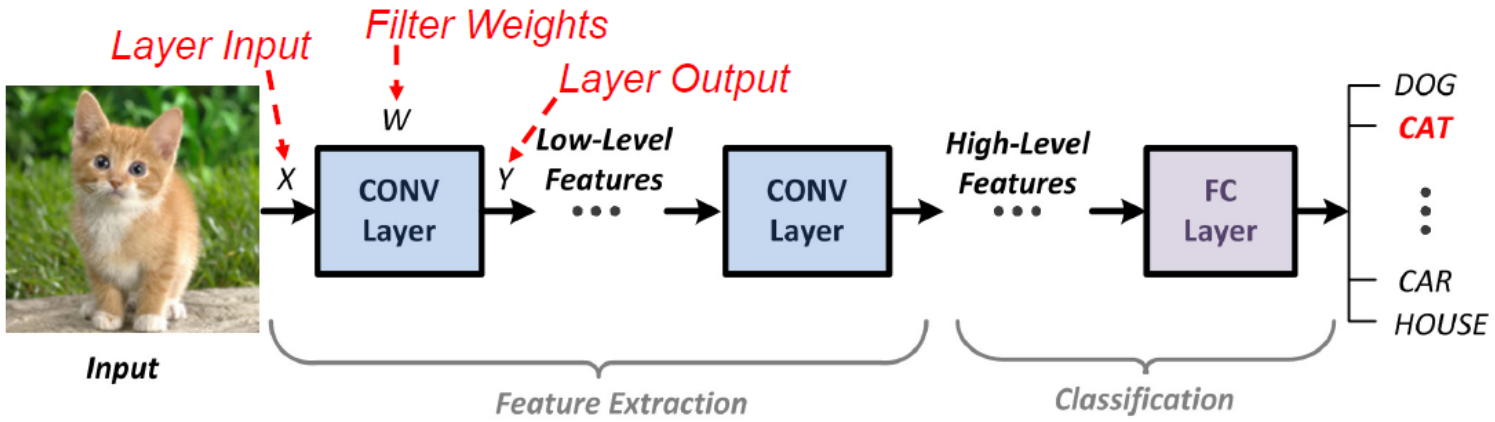
[Biswas, ISSCC'18]

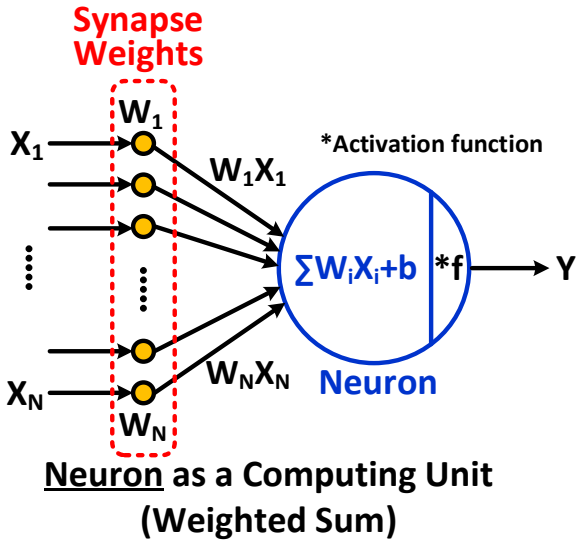

- Face Recognition
- Speech Recognition
- Image Classification
- Object Detection
- High Power Consumption
  - AlphaGo (1MWatt) vs Human (20W)

- Faster Local Decisions
- Less Communication
- More Secure (local data)
- Requirements:
  - Low power consumption
  - Low storage capacity
  - Real-time processing

# Convolutional Neural Networks (CNNs)




[Biswas, ISSCC'18]




Key Operation: Multiplication & Accumulation (MAC) using filter weights and layer inputs

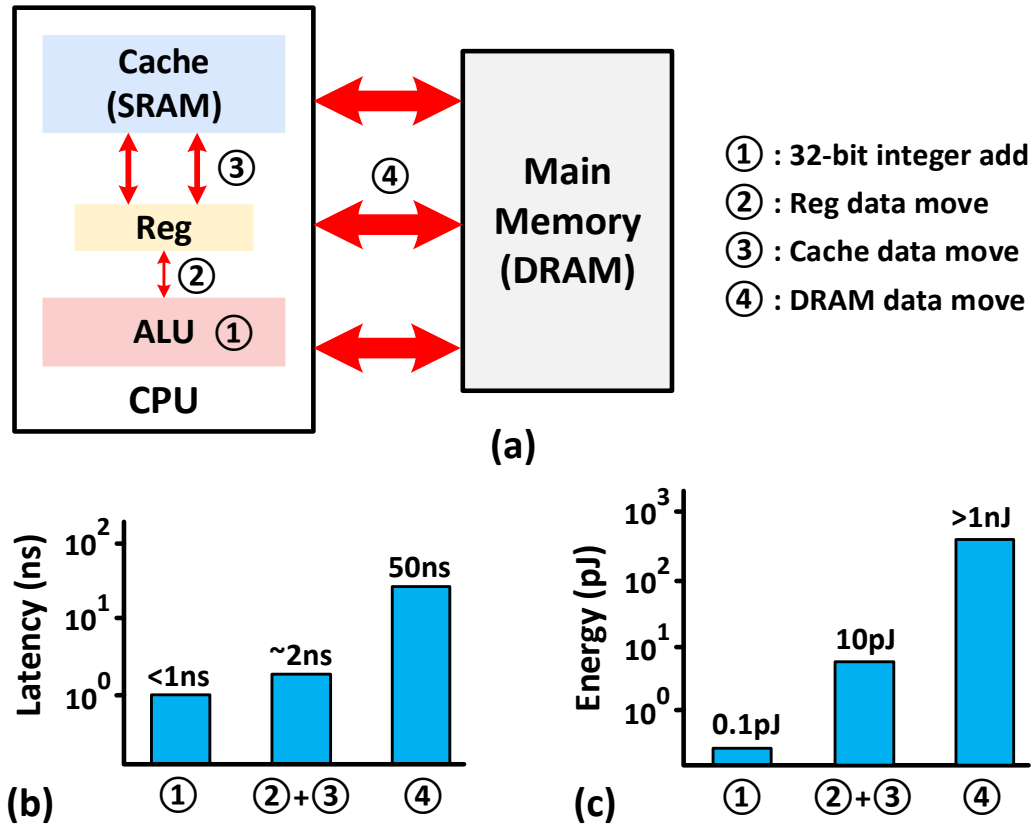
# Popular DNN Models

[Sze, NeurIPS'19]

| Metrics | LeNet-5 | AlexNet | VGG-16 | GoogLeNet (v1) | ResNet-50 | EfficientNet - B4 |
|---|---|---|---|---|---|---|
| Top-5 error (ImageNet) | n/a | 16.4 | 7.4 | 6.7 | 5.3 | 3.7 |
| Input Size | 28x28 | 227x227 | 224x224 | 224x224 | 224x224 | 380x380 |
| **# of CONV Layers** | **2** | **5** | **16** | **21 (depth)** | **49** | **96** |
| # of Weights | 2.6k | 2.3M | 14.7M | 6.0M | 23.5M | 14M |
| # of MACs | 283k | 666M | 15.3G | 1.43G | 3.86G | 4.4G |
| **# of FC layers** | **2** | **3** | **3** | **1** | **1** | **65** |
| # of Weights | 58k | 58.6M | 124M | 1M | 2M | 4.9M |
| # of MACs | 58k | 58.6M | 124M | 1M | 2M | 4.9M |
| **Total Weights** | **60k** | **61M** | **138M** | **7M** | **25.5M** | **19M** |
| **Total MACs** | **341k** | **724M** | **15.5G** | **1.43G** | **3.9G** | **4.4G** |
| **Reference** | **Lecun**, *PIEEE* 1998 | **Krizhevsky**, *NeurIPS* 2012 | **Simonyan**, *ICLR* 2015 | **Szegedy**, *CVPR* 2015 | **He**, *CVPR* 2016 | **Tan**, *ICML* 2019 |

- Larger and deeper DNN models: not suitable for IoT devices

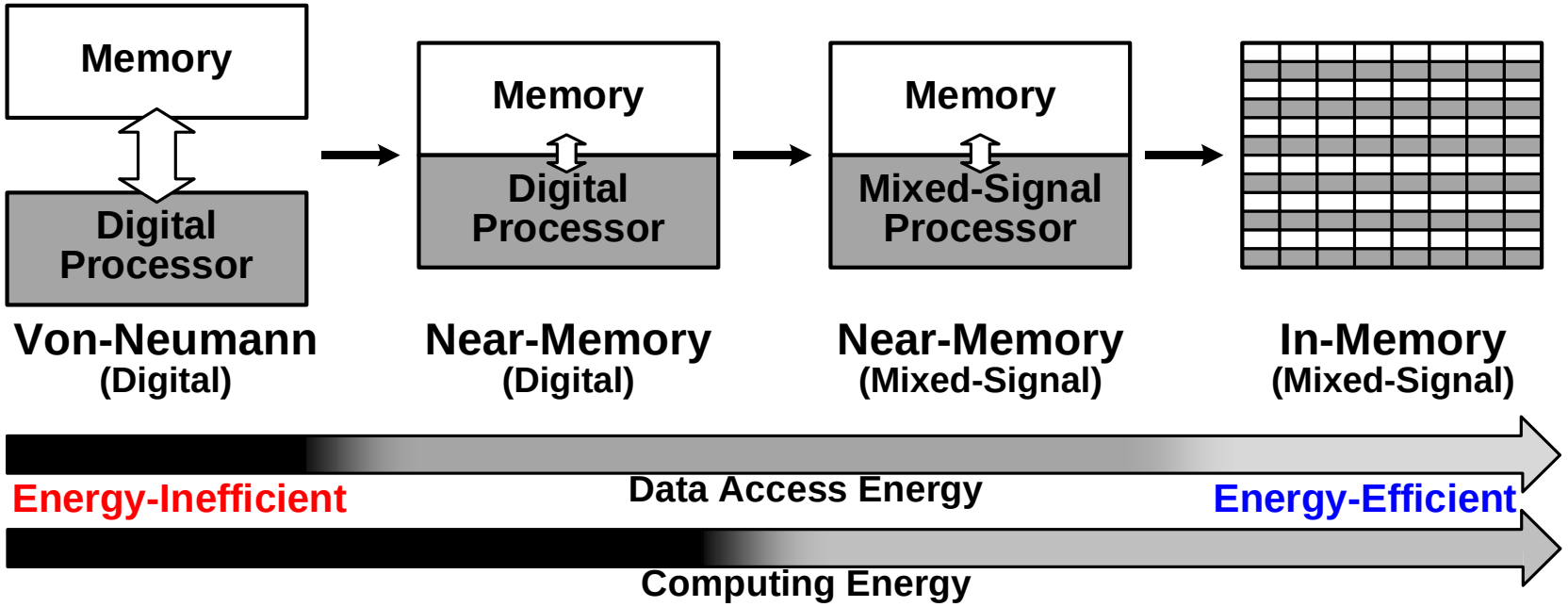
# Conventional Computing Architecture




[M. Horowitz, ISSCC'14]

- Von Neumann Architecture: data movement across memory layers and system bus
  - Long latency, high power consumption, hardware cost

# Architecture vs. Energy Efficiency



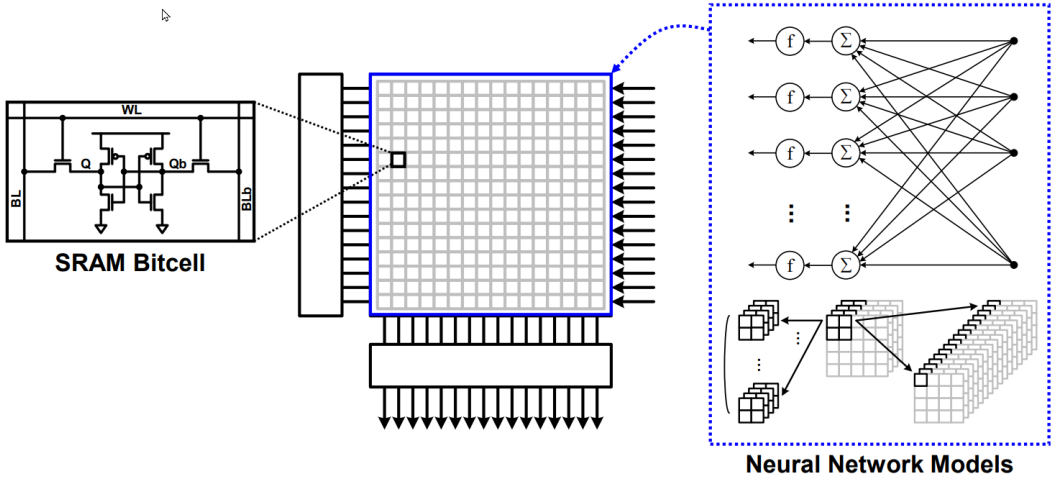

- Von-Neumann architecture: computation bottleneck and excessive energy consumption due to memory access.
- Computing-in-memory: high energy efficiency and high performance with massive parallelism

# Outline

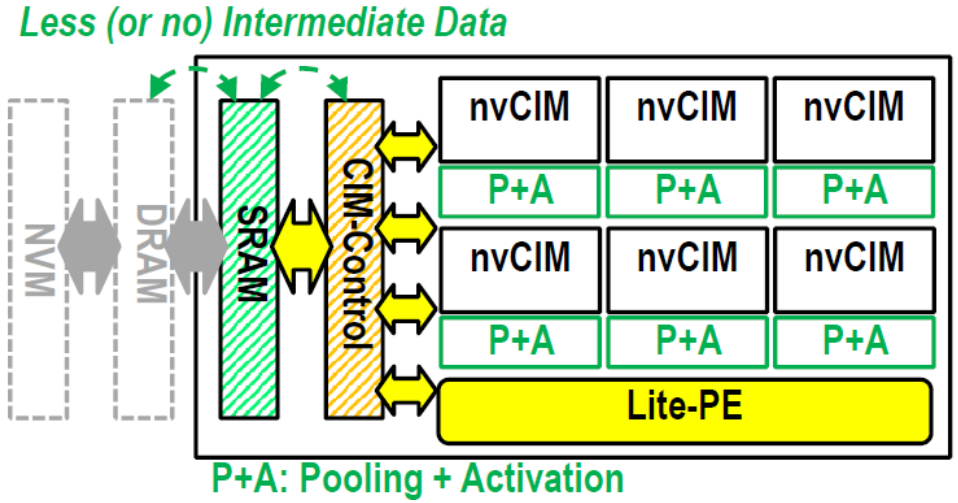
- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
- Summary

# Computing-in-Memory: Basics

- Computation of MACs inside of memory
- Features
  - Activation of multiple rows
  - Analog bitline voltage or current for representing MAC results
  - Digitization using Analog-to-digital converters



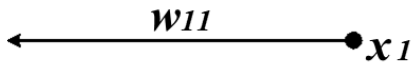

[Kim, ISCAS'21 Tutorial]

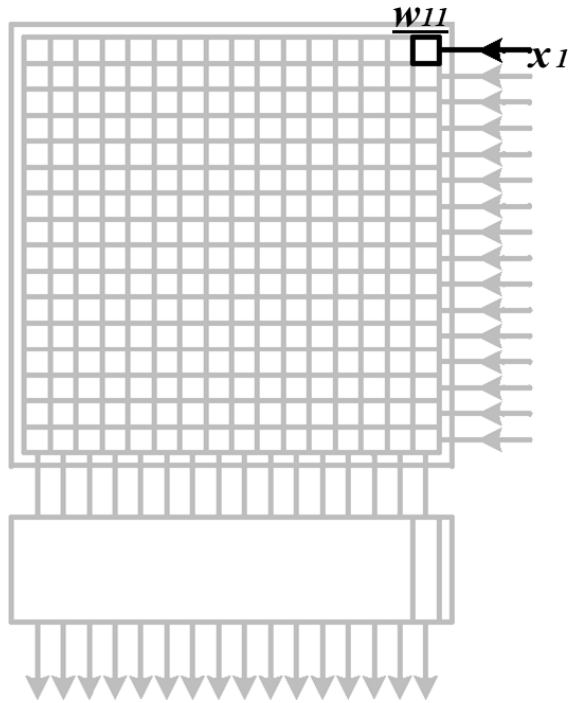



[Chen, ISSCC'18]

# Synapse: SRAM Bitcell

[Kim, ISCAS'21 Tutorial]

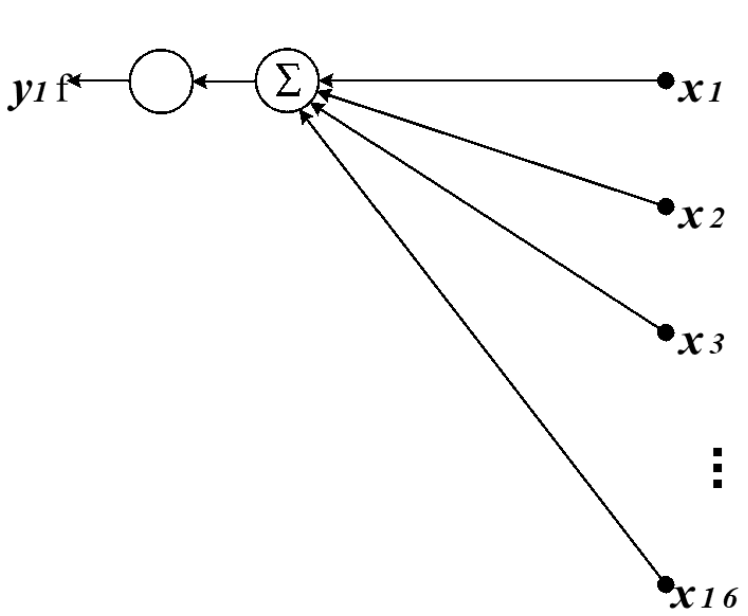


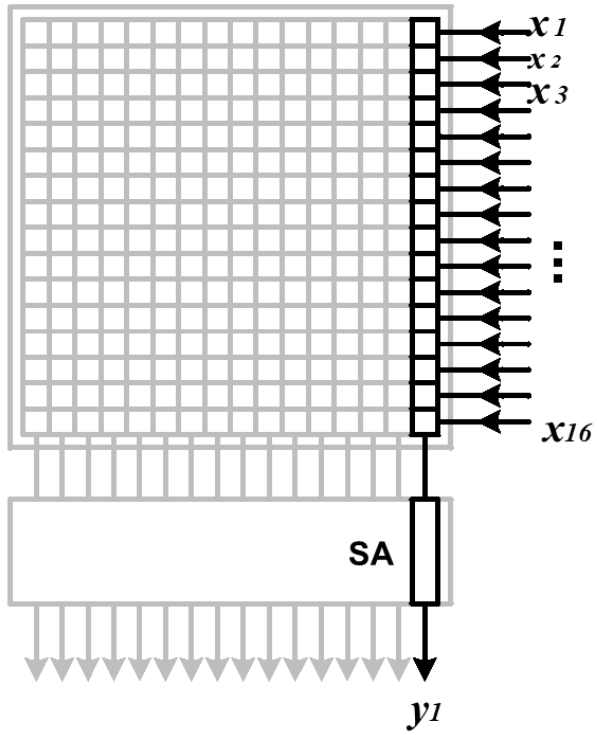




- A (binary) synapse can be mapped to a single SRAM bitcell
  - Multiplication in the SRAM bitcell

# Synapse: SRAM Bitcell

[Kim, ISCAS'21 Tutorial]

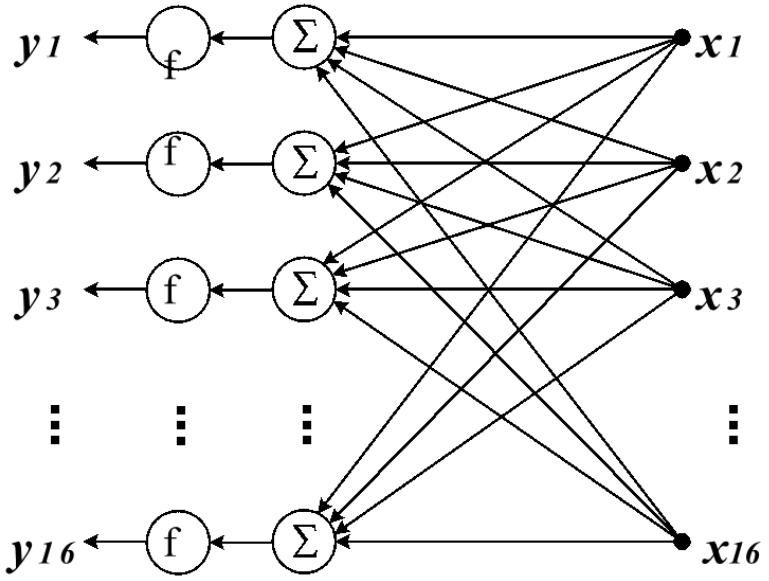


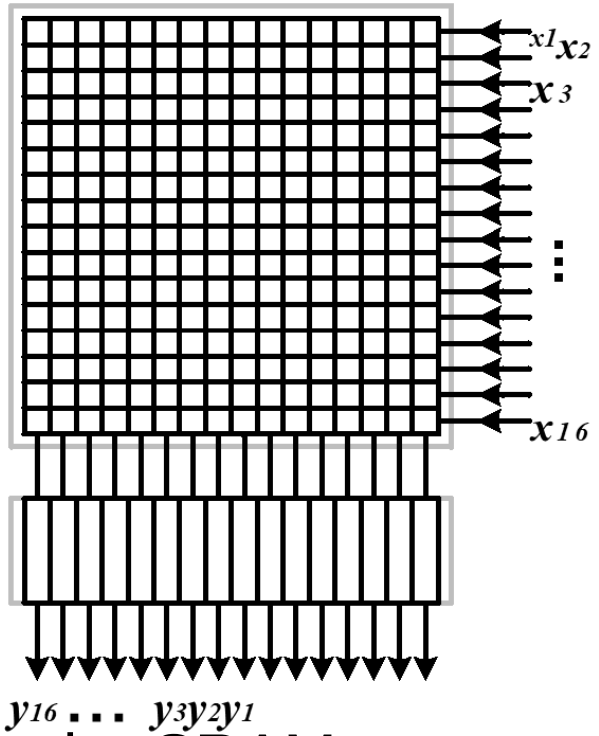




- A neuron is mapped to a column of bitcells
  - A dot-product between input and weight & an activation is performed

# Synapse: SRAM Bitcell

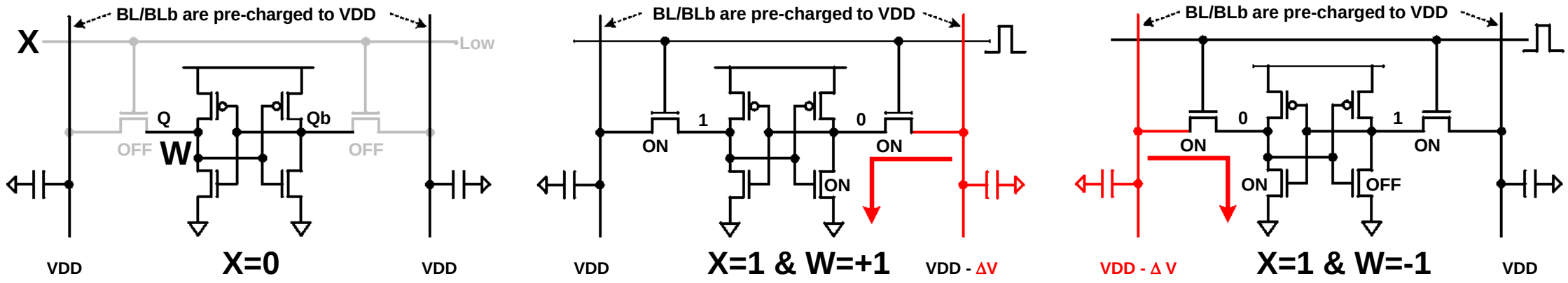
[Kim, ISCAS'21 Tutorial]




- A feedforward neural network is mapped to entire SRAM macro
  - Many parallel dot-products = a matrix (weight) vector (input) multiplication

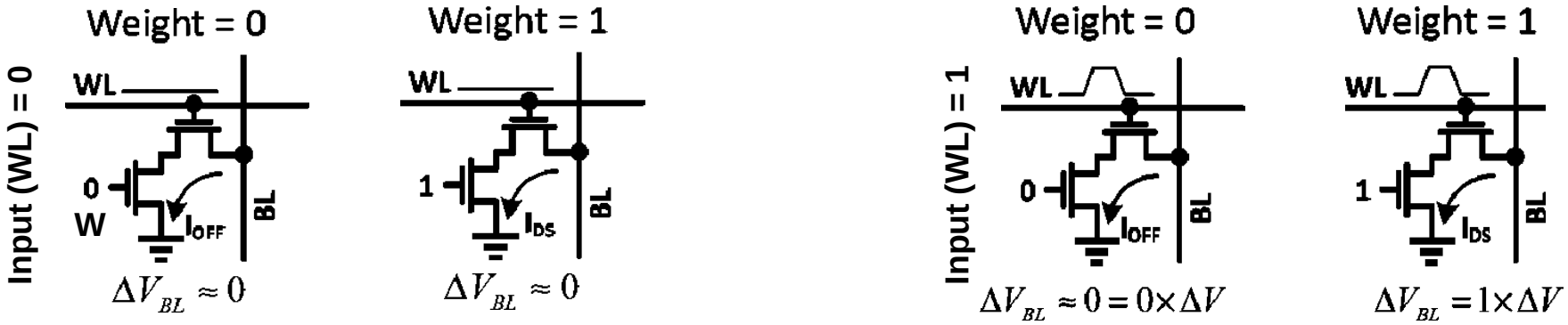
# Binary MAC Operation in 6T SRAM Cell




- Differential bitlines
  - Input X = 0/1 and weight W = -1/+1
  - Three different voltage differences: 0 V, ΔV, and -ΔV

| X \ W | +1 | -1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | -ΔV | ΔV |

# Binary MAC Operation in Single-ended BL




[Dong, ISSCC'20]

- Single-ended bitline
  - Input X = 0/1 and weight W = 0/1
  - Two different voltage differences: 0 V and $\Delta$V

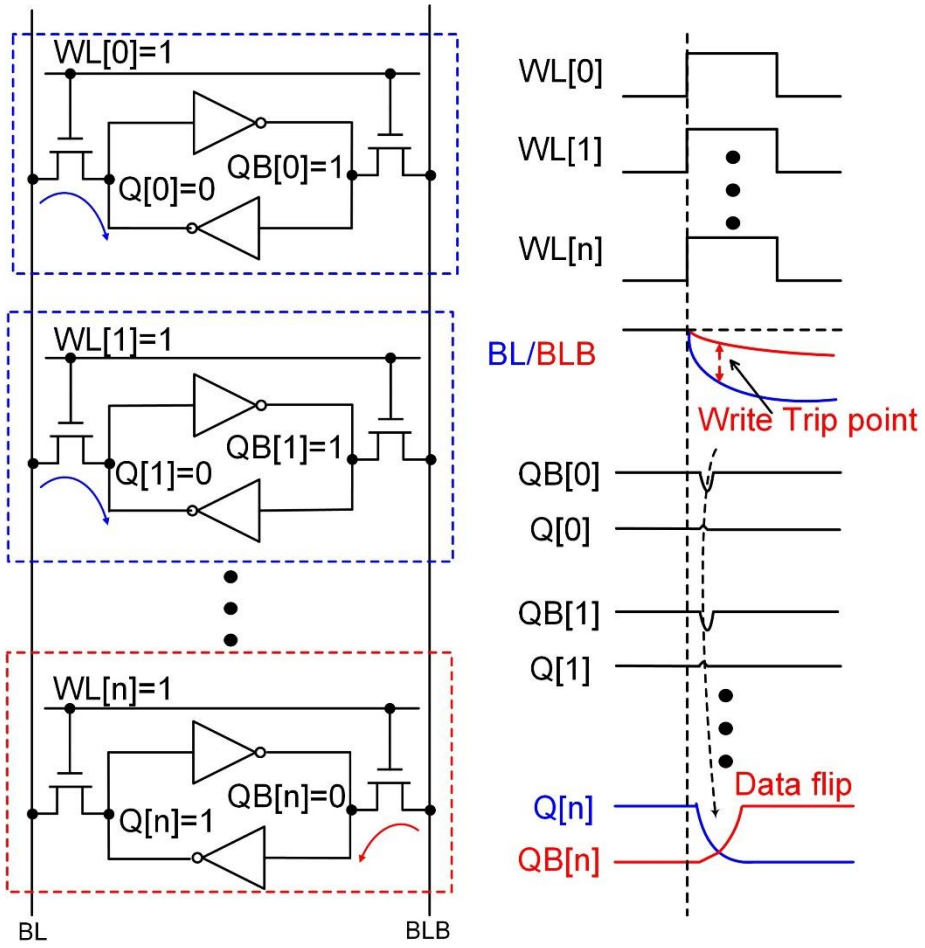
| X \ W | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | $\Delta$V |

# Computing-in-Memory: Challenges

- Disturbance during MAC operation
- Bit cell area
- Narrow dynamic range for linearity
- Limited precision
- ADC area/power overhead
- Limited reconfigurability

# Computing-in-Memory: Challenges

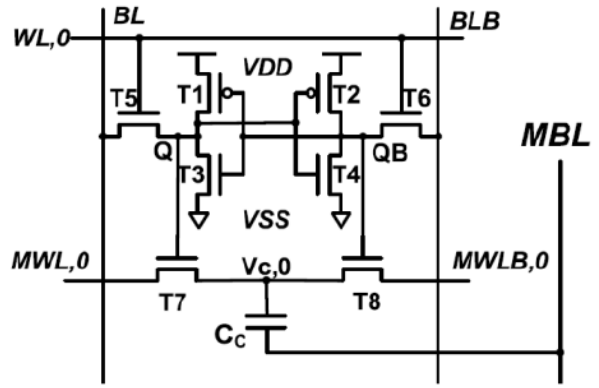
- **Disturbance during MAC operation**
  - Internal nodes affected by bitline voltage
  - Data flip due to multiple enabled SRAM cells and a wide bitline voltage range
- Bit cell area
- Narrow dynamic range for linearity
- Limited precision
- ADC area/power overhead
- Limited reconfigurability



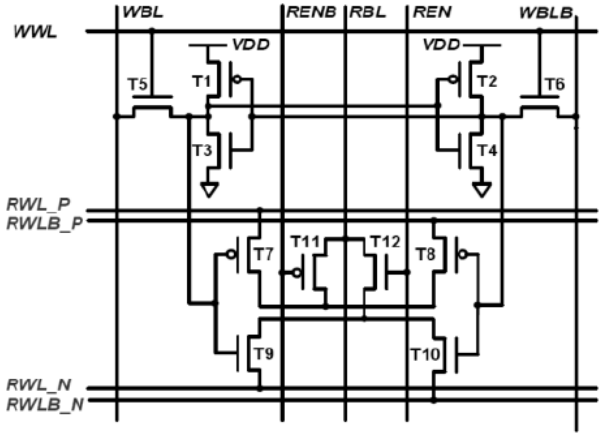

[Si, JSSC'20]

# Computing-in-Memory: Challenges

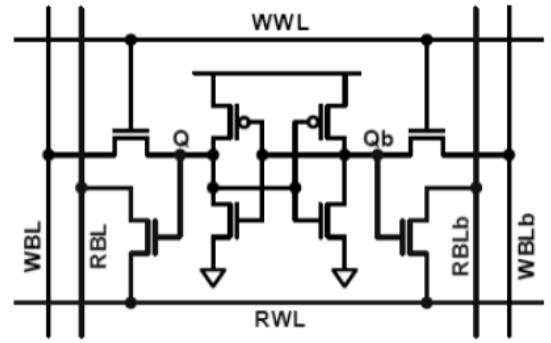
- Disturbance during MAC operation
- **Bit cell area**
  - Decoupled bitcells for removing disturbance
  - Additional TRs increasing area overhead
- Narrow dynamic range for linearity
- Limited precision
- ADC area/power overhead
- Limited reconfigurability




[Jiang, ESCIRC'19]




[Yin, JSSC'20]

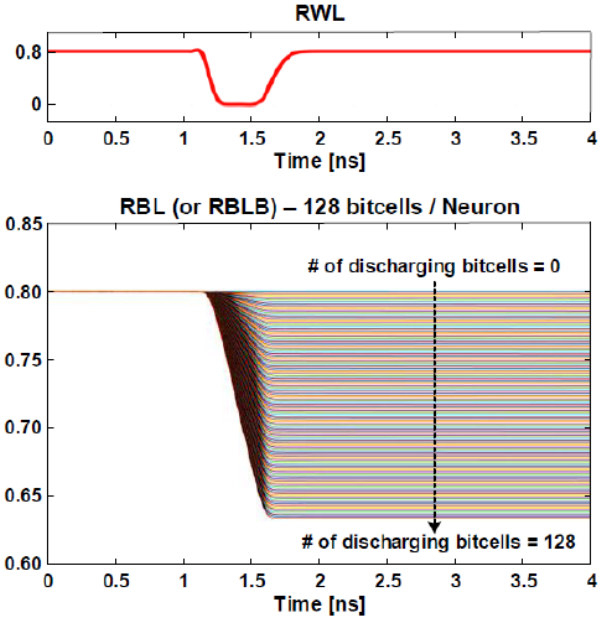



[Yu, CICC'20]

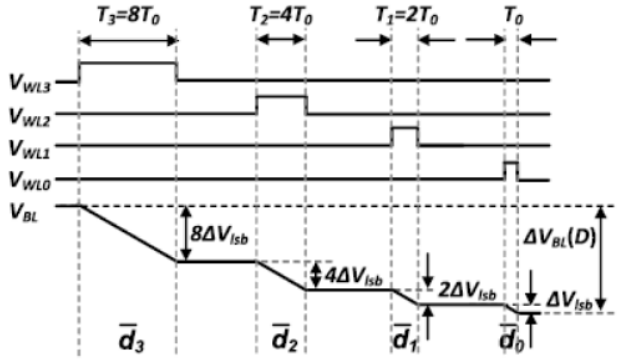
# Computing-in-Memory: Challenges

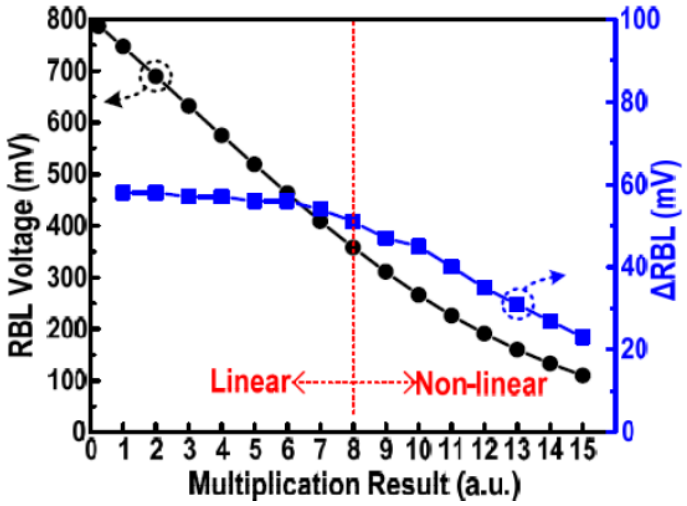
- Disturbance during MAC operation
- Bit cell area
- **Narrow dynamic range for linearity**
  - Nonlinear voltage step depending on MAC value
  - Limited dynamic range: 200~300mV
- Limited precision
- ADC area/power overhead
- Limited reconfigurability

[Yu, CICC'20]




[Kang, JSSC'18]



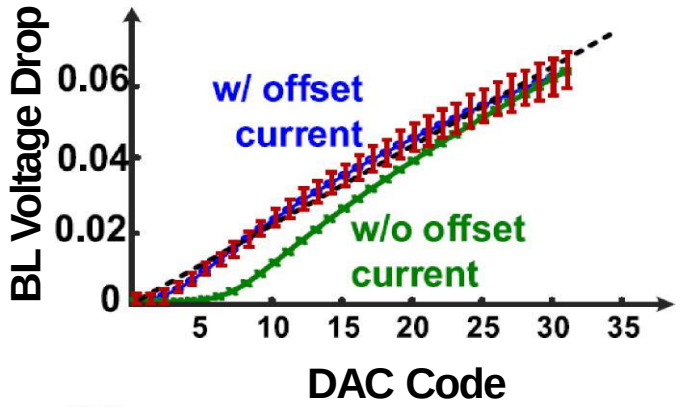



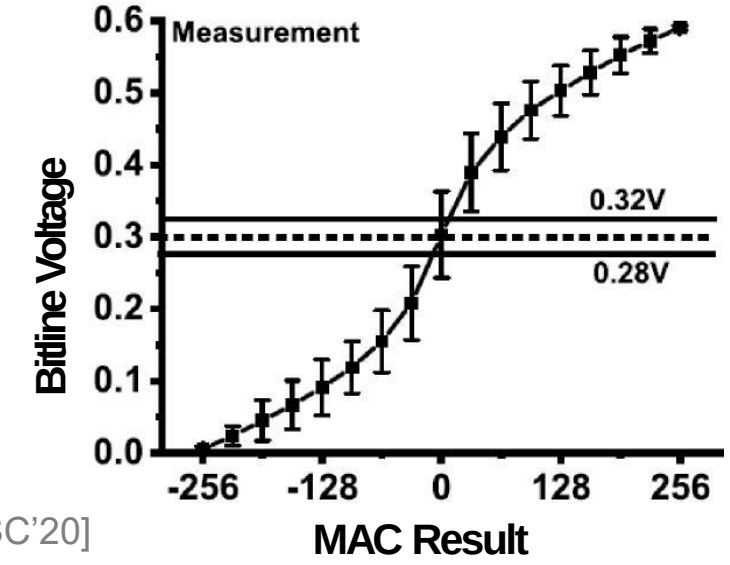

[Dong, ISSCC'20]

# Computing-in-Memory: Challenges

- Disturbance during MAC operation
- Bit cell area
- Narrow dynamic range for linearity
- **Limited precision**
  - Nonlinear MAC results caused by analog processing
  - PVT variation impacts on bitline voltage
- ADC area/power overhead
- Limited reconfigurability

[Zhang, JSSC'17]



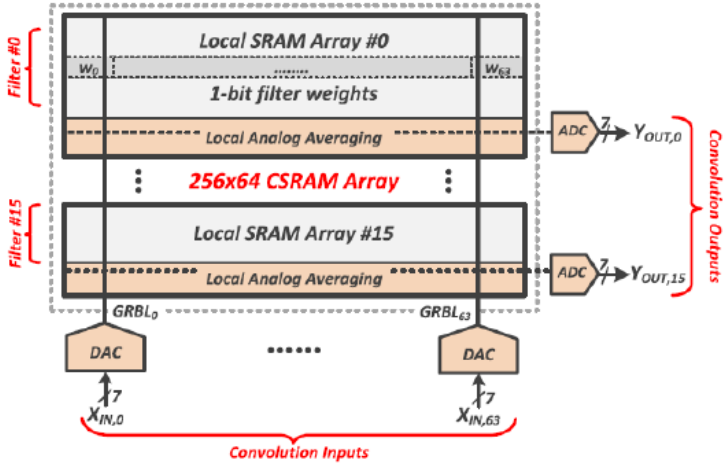



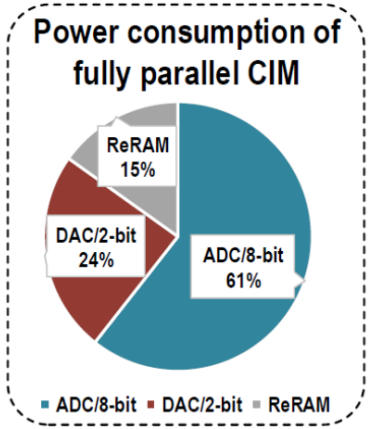

[Yin, JSSC'20]

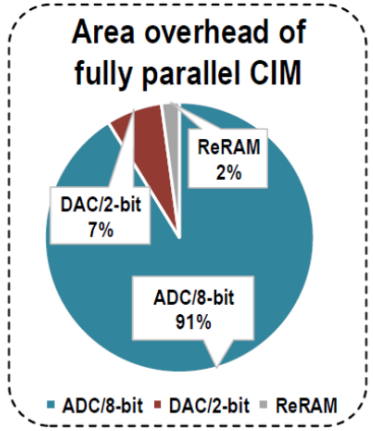
# Computing-in-Memory: Challenges

- Disturbance during MAC operation
- Bit cell area
- Narrow dynamic range for linearity
- Limited precision
- **ADC area/power overhead**
  - Multiple DACs and ADCs
  - Leading to energy/area overhead
- Limited reconfigurability

[Biswas, JSSC'19]



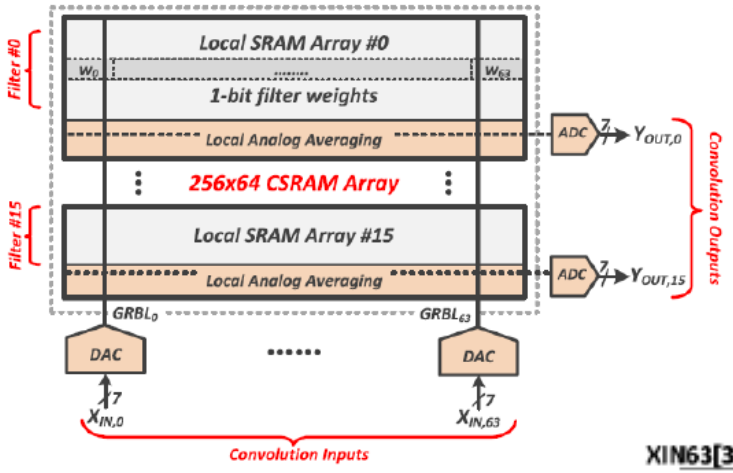

[Liu, ISSCC'20]

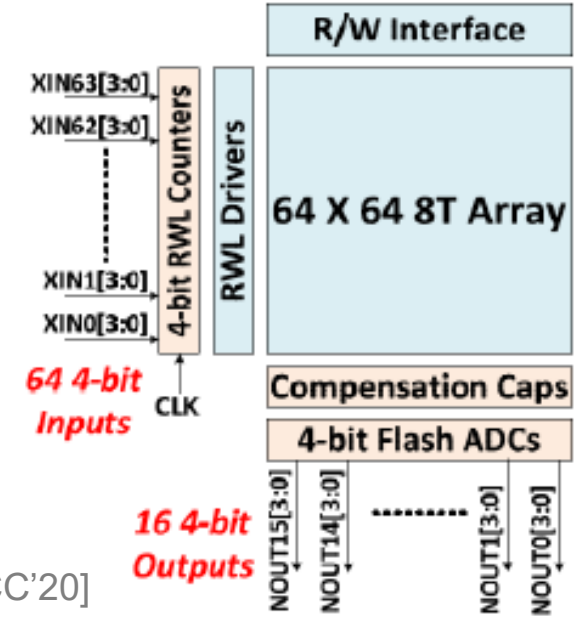
1152*128 ReRAM array, 2-bit DAC and 8-bit ADC are adopted in this evaluation

# Computing-in-Memory: Challenges

- Disturbance during MAC operation
- Bit cell area
- Narrow dynamic range for linearity
- Limited precision
- ADC area/power overhead
- **Limited reconfigurability**
  - Fixed input, weight, and output
  - Difficult to reconfigure due to analog processing

[Biswas, JSSC'19]



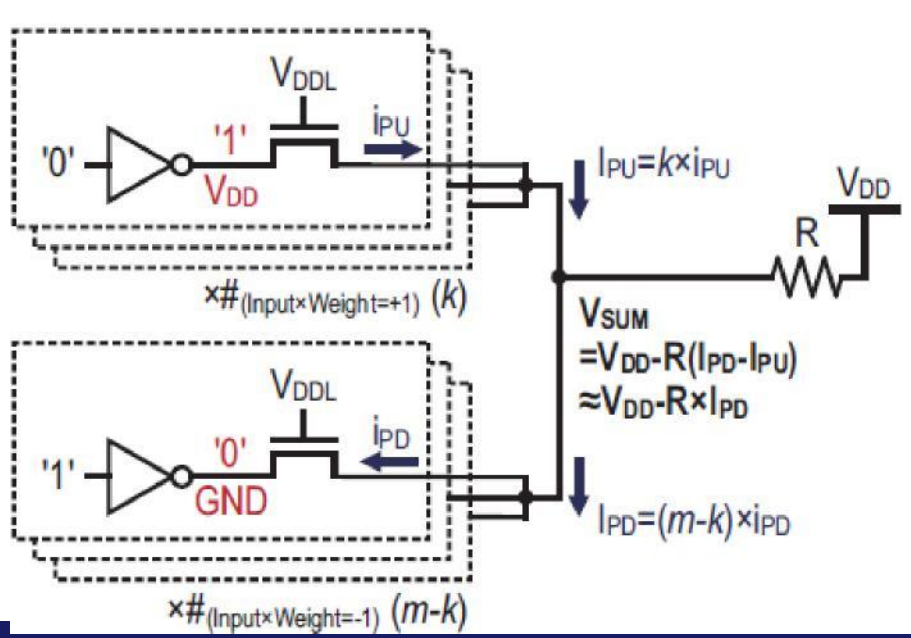

[Dong, ISSCC'20]

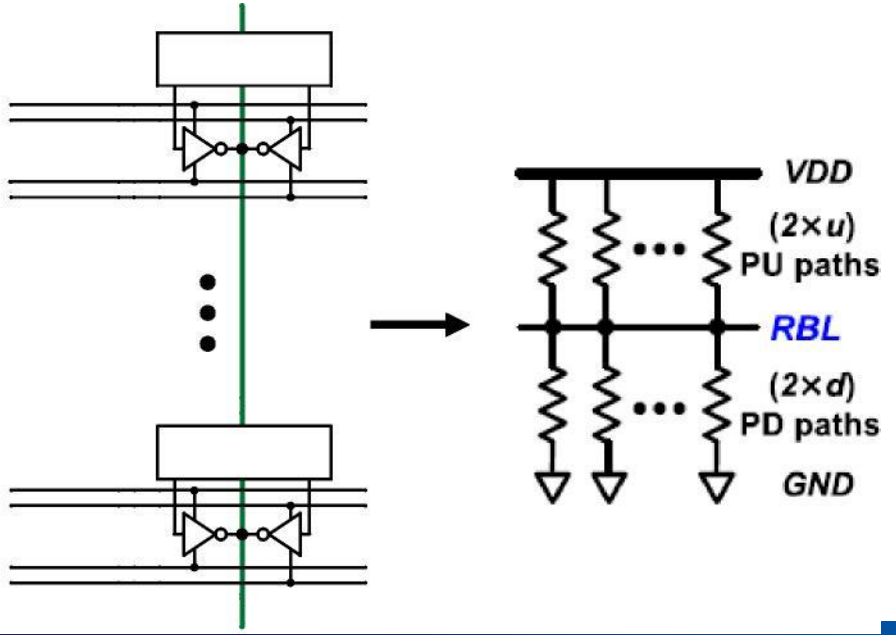
# Outline

- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
  - Analog CIM
  - Digital CIM
  - ReRAM CIM
- Summary

# Accumulate using Pull-Up/Down Drivers

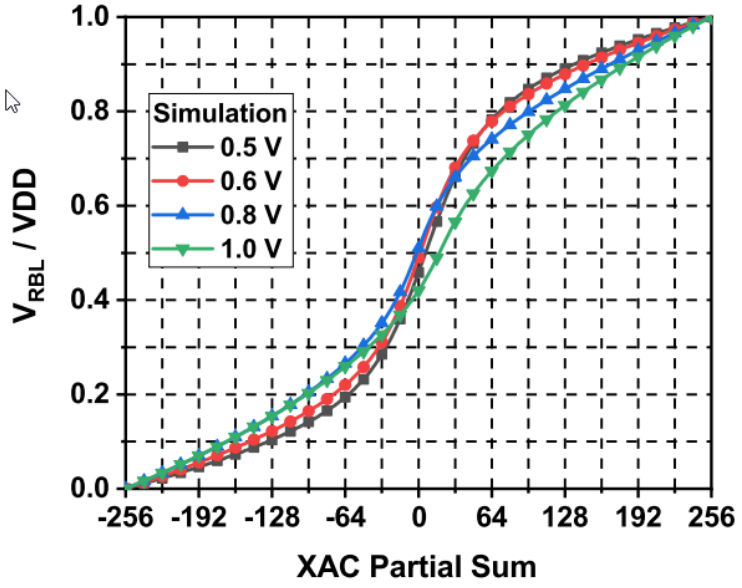
- ‘Pull-up’ and ‘Pull-down’ resistance determined by cell data
- Bitline voltage dependency on ‘pull-up’ and ‘pull-down’ resistance distribution
- Wide bitline swing at the cost of short circuit current

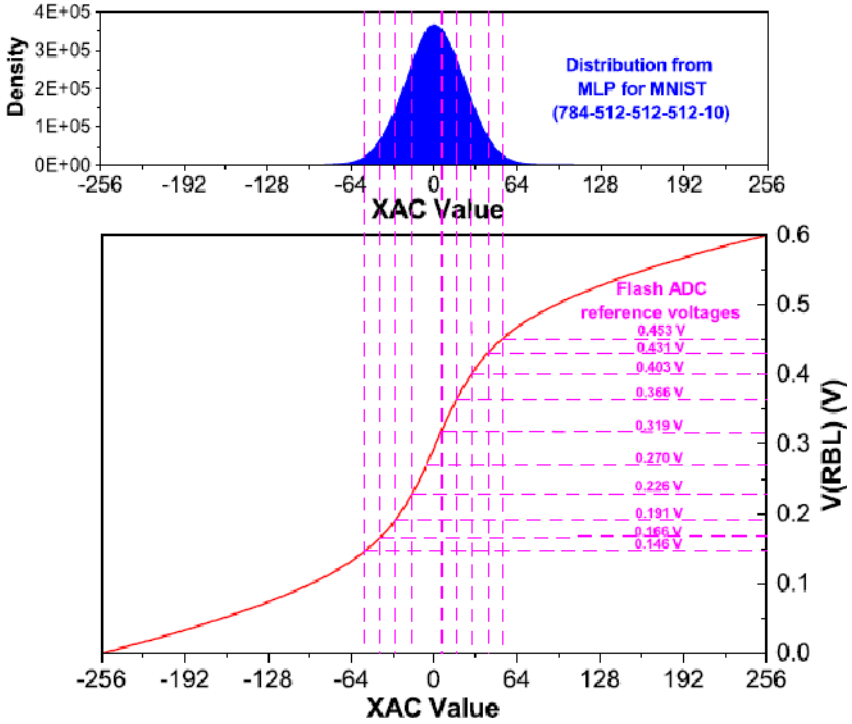


# Accumulate using Pull-Up/Down Drivers

- Nonlinear relationship between MAC result and bitline voltage
- Nonlinearity dependency on supply voltage
- Nonlinear reference voltages for linear ADC



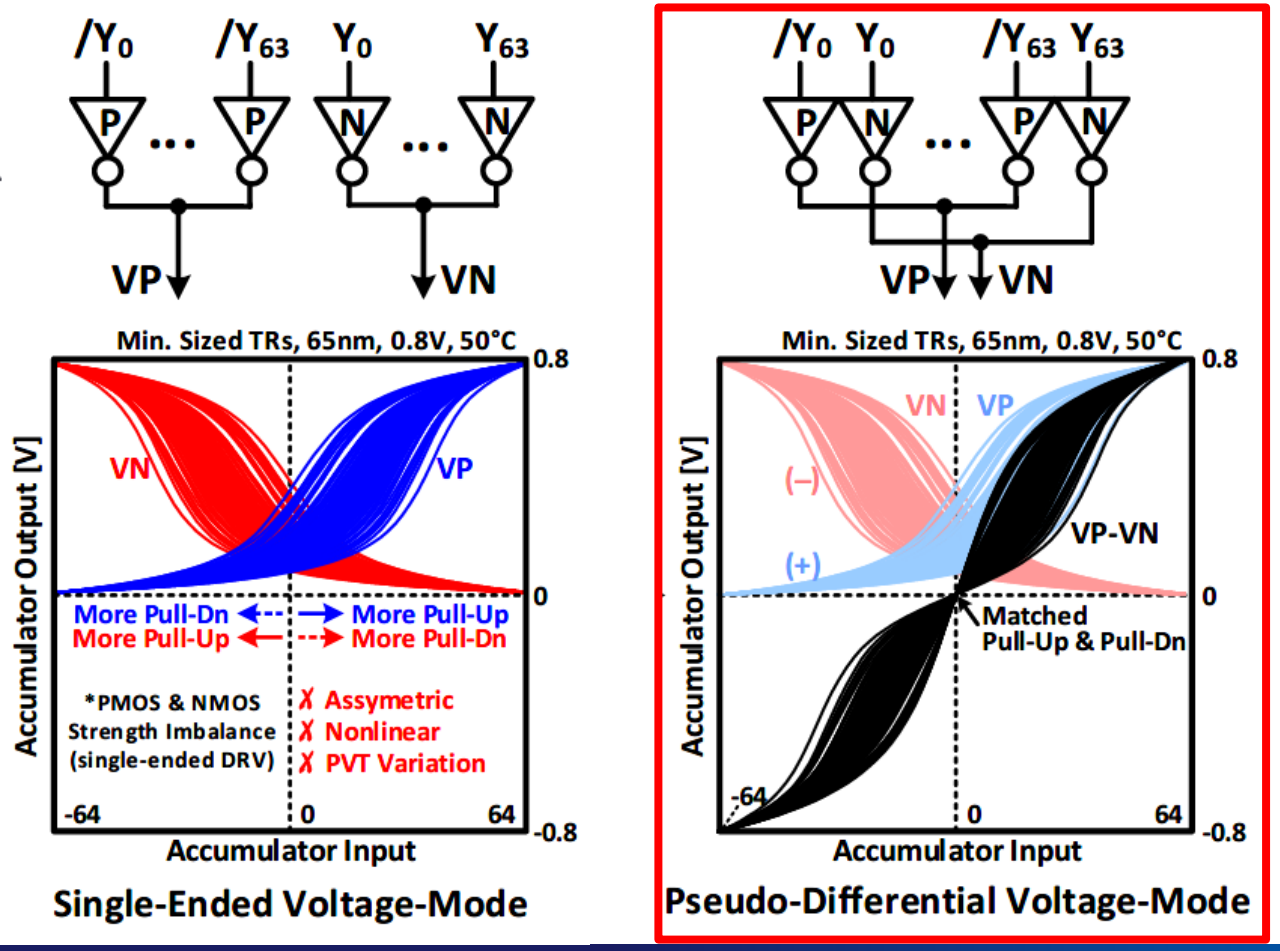




[Yin, JSSC'20]

# Differential Accumulation

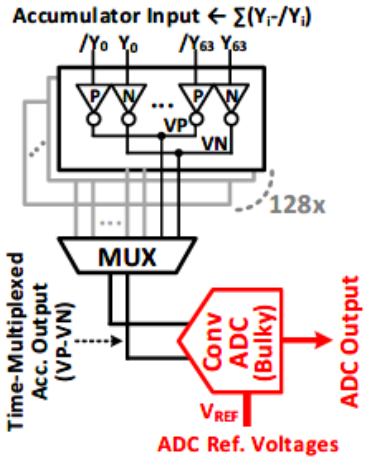
[Kim, A-SSCC'19]

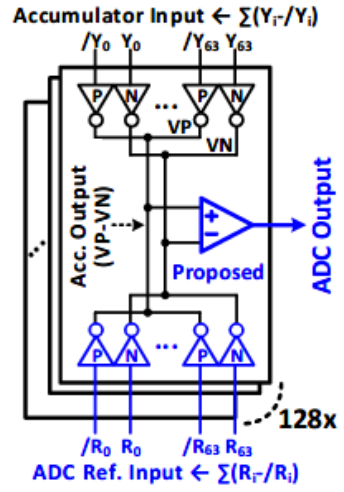
- Nonlinear single-ended accumulation
- Pseudo differential accumulation
- Improved linearity
- Reference voltage generated by replica bitcells



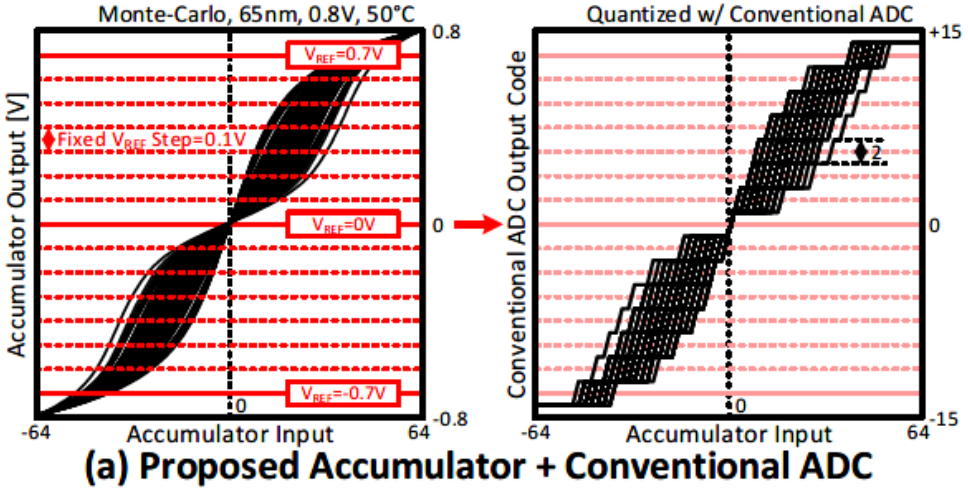
# ADC Using Replica Bitcells

- Reference voltage generate by replica bitcells
- Better linearity and variation tolerance

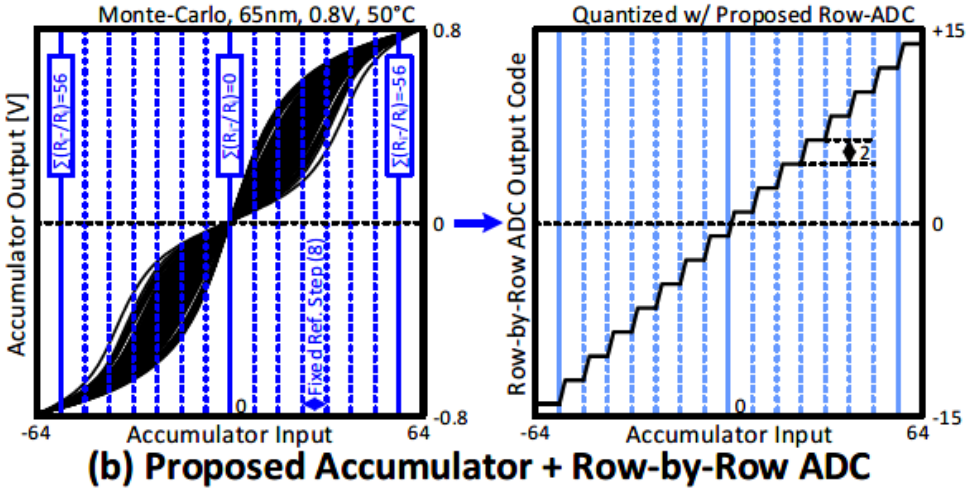



[Kim, A-SSCC'19]




(a) Proposed Accumulator + Conventional ADC

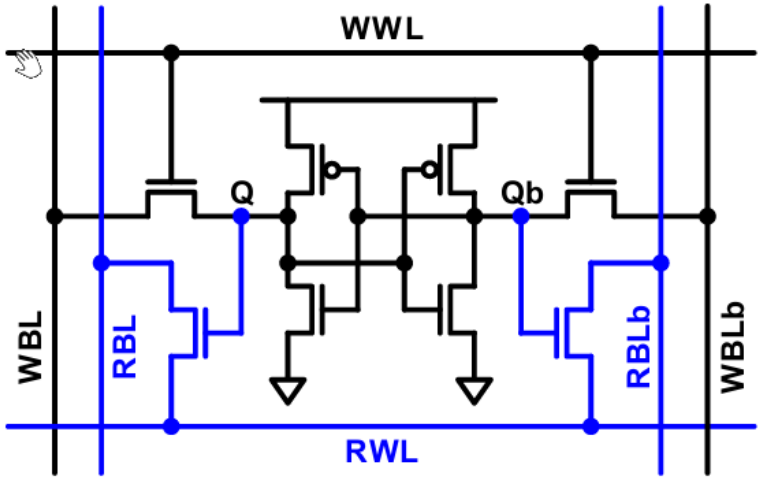



(b) Proposed Accumulator + Row-by-Row ADC

# Current-based Accumulation

- Decoupled 8T SRAM cell for disturb-free MAC operation
- Differential bitline for MAC operation
- Combined cell current is converted to bitline voltage

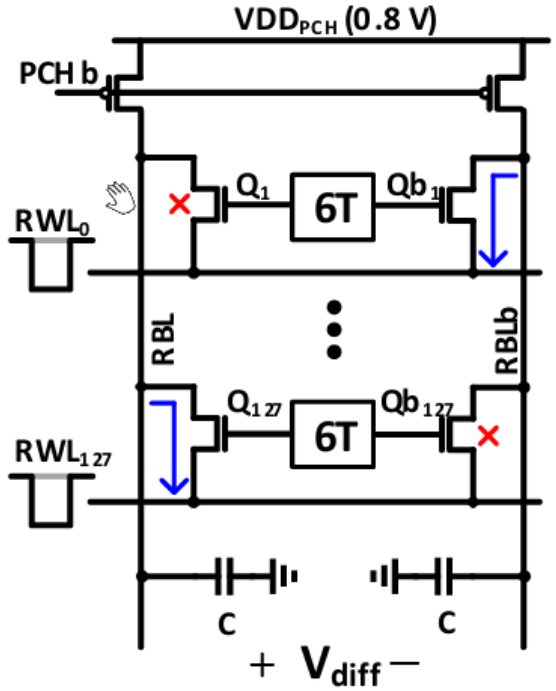
[Yu, CICC'20]




**Logic Table**

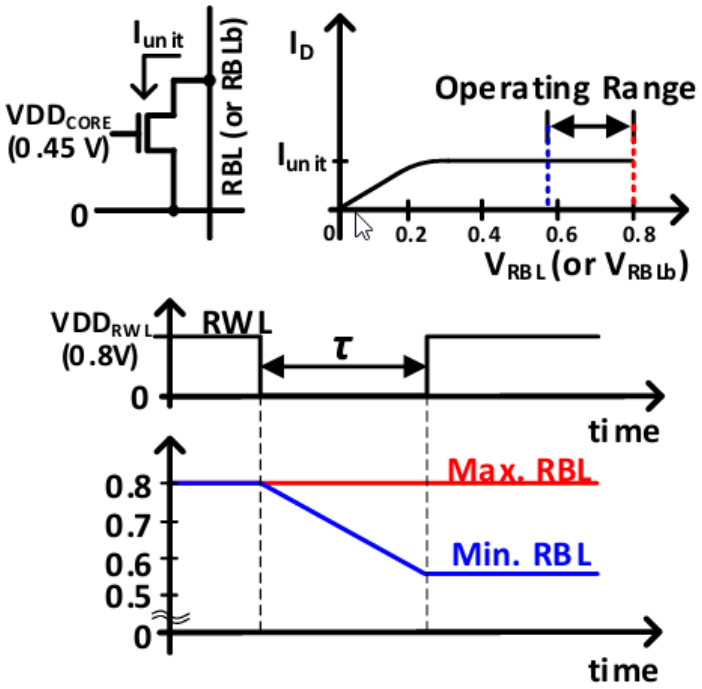
| Weight (Q,Qb) \ Input (RWL) | 0 (RWL=H) | 1 (RWL= ⊓ ) |
|---|---|---|
| -1 (Q=H, Qb=L) | 0 (No change) | -1 (-ΔV ) |
| +1 (Q=L, Qb=H) | 0 (No change) | +1 (+ΔV) |

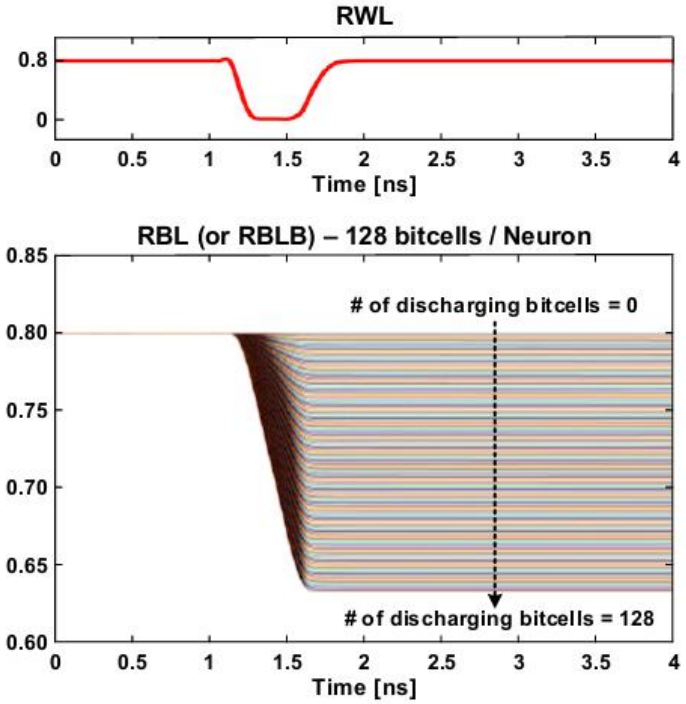
( ): Circuit behavior representation



# Current-based Accumulation

- Lower core supply voltage for constant unit current ($I_{unit}$)
- Limited bitline dynamic range (~200 mV) for linearity
- RWL pulse width control for target dynamic range

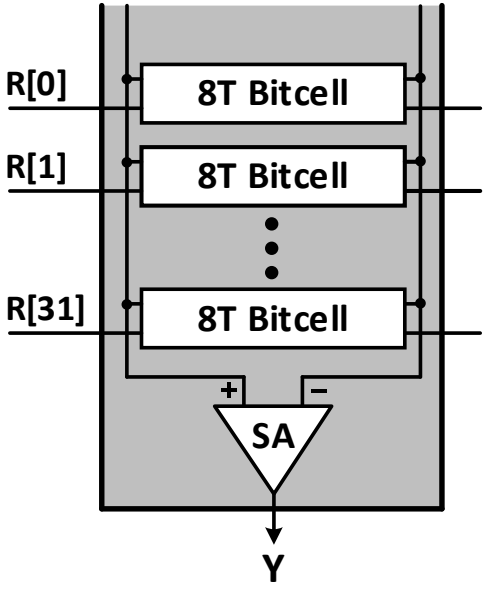


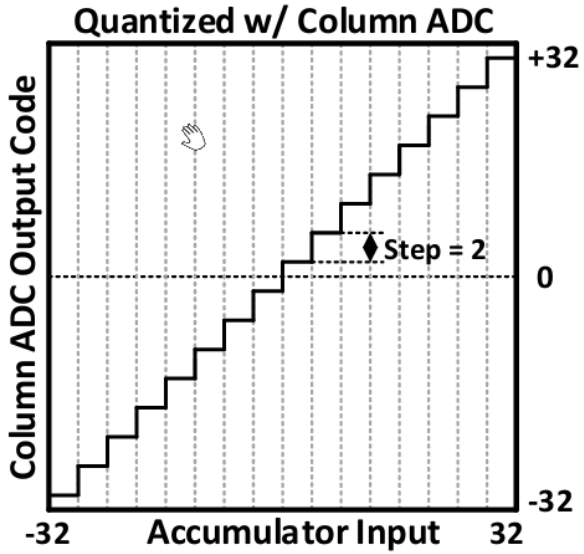



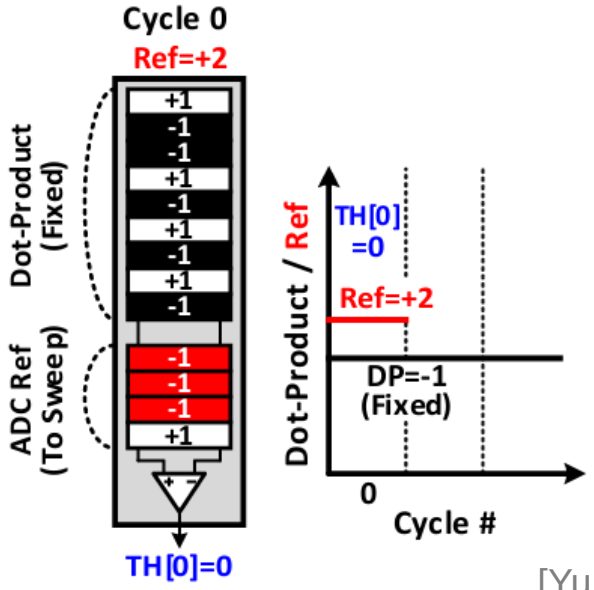

[Yu, CICC'20]

# Bitcell-based Column ADC

- 32 replica bit cells for sweeping reference
- 1-5 bit output precision controlled by # of cycles
- TH[0]: sense amplifier output with highest reference



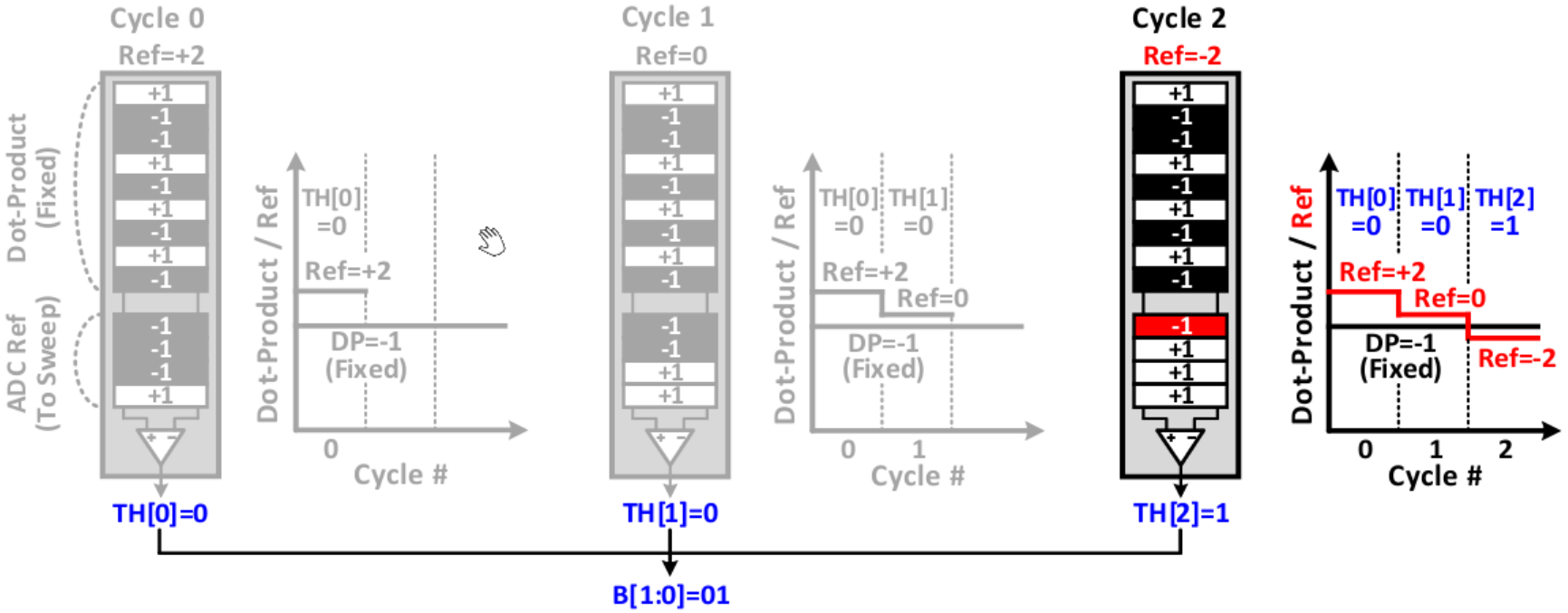




[Yu, CICC'20]

# Bitcell-based Column ADC

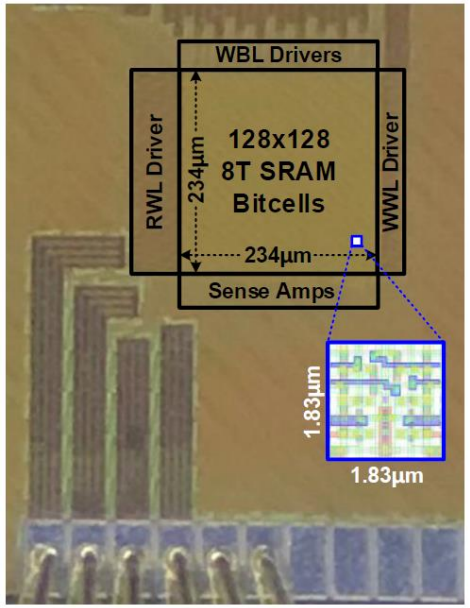
- Incrementing reference by writing more ‘1s’ in the replica bitcells
- TH[2]: sense amplifier output with lower reference
- Digitized output generation using sense amplifier output over cycles




[Yu, CICC’20]

# Measurement Summary

- Summary and Comparison
  - Cycle-based reconfigurable output precision
  - Low voltage operation for energy-constrained IoT devices




| Process | 65nm |
|---|---|
| Supply Voltage | 0.8V (RWL/PCH) /0.45V (SRAM) |
| Operating Frequency | 200/400MHz |
| Bitcell | 8T SRAM |
| Bitcell Size | 1.83x1.83µm$^2$ |
| Array Size | 128 x 128 (16K) |
| Efficiency (1bit OP) | 2.04fJ @ 200M 1.99fJ @ 400M |
| ML Algorithm | MLP (2-layer) 784-128-128-10 |
| Dataset /Accuracy | MNIST/96.2% (-0.4% vs. baseline) |

| | SOVC'16 [2] | ISSCC'18 [3] | SOVT'18 [4] | This Work |
|---|---|---|---|---|
| Technology | 130nm | 65nm | 65nm | 65nm |
| Bitcell | 6T SRAM | 10T SRAM | 12T SRAM | 8T SRAM |
| Accumulation | Current-Mode | Voltage-Mode | Voltage-Mode | Current-Mode |
| Array Size | 128x128 | 64x256 | 256x64 | 128x128 |
| Input/Out. Bit# | 5/1 | 6/6 | 1.59/3.46 | 1/1-5 |
| Weight Bit # | 1 | 1 | 1 | 1 |
| Energy-Efficiency [TOPS/W] | 11.5 | 51.3 | 139 | 490-15.8** |
| # ADCs/Neurons | N/A | 16/256 | 1/64 | 128/128 |
| ML Algotithm | SVM | CNN | MLP | MLP |
| ML Dataset | MNIST | MNIST | MNIST | MNIST |
| Accuracy | 90% | 98% | 98.3% | 96.2%* |

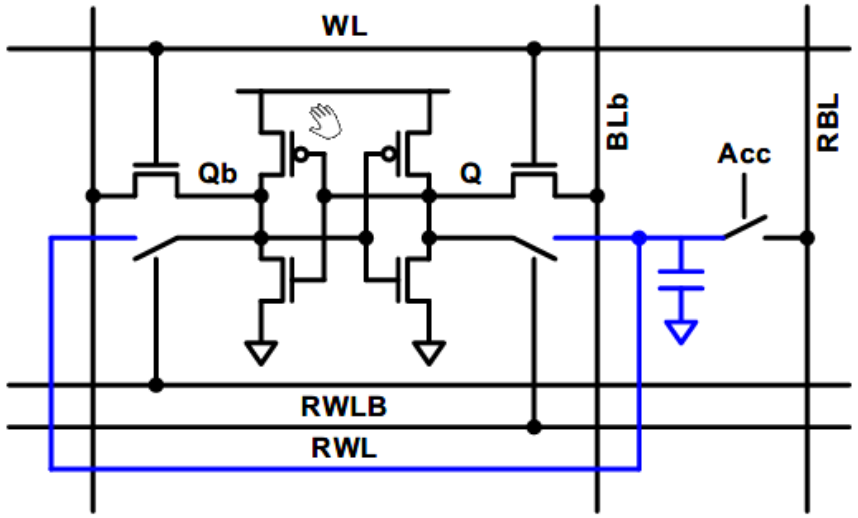
*Accuracy based-on MC (1K runs) sim. results (σ=6.35mV) **1-5b (1-31cycles/OP), 200MHz

[Yu, CICC'20]

# Charge Sharing vs Capacitive Coupling

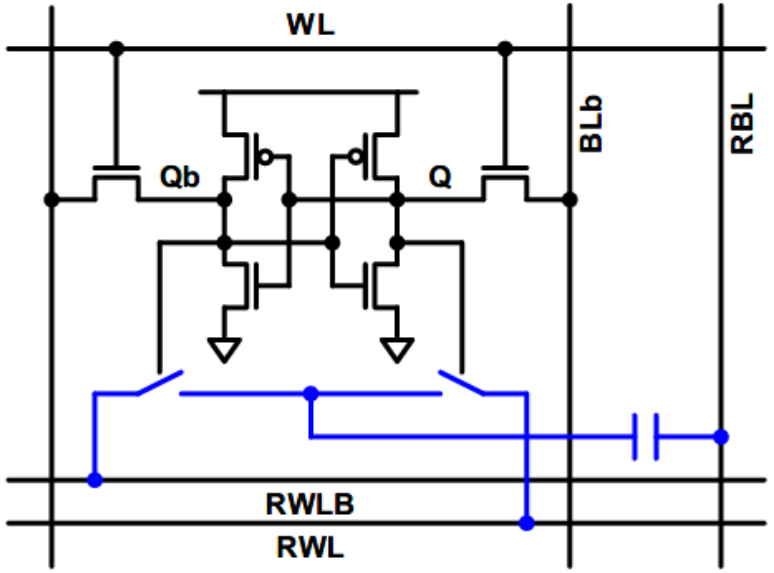
- Analog MAC result using either charge-sharing or capacitive-coupling
- Unit capacitor implementation using metal layers

**Bitcell with Charge-Sharing**




[Valavi, JSSC'19]

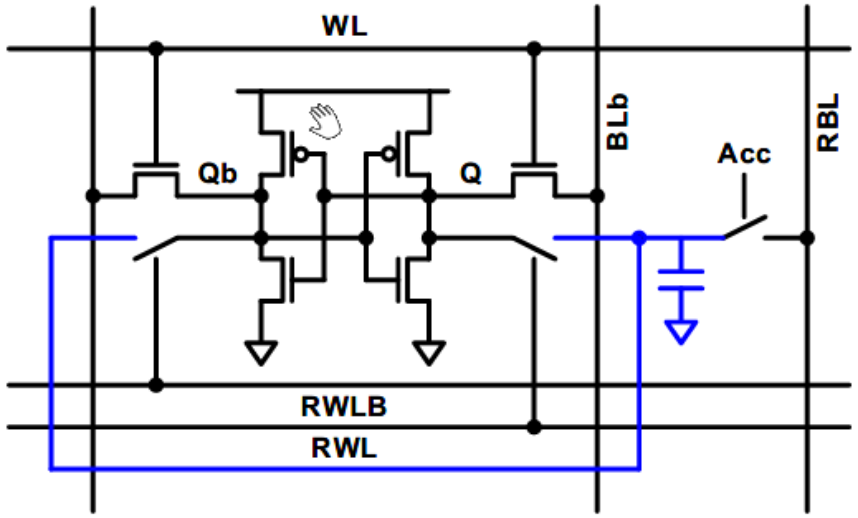
**Bitcell with Charge-Coupling**



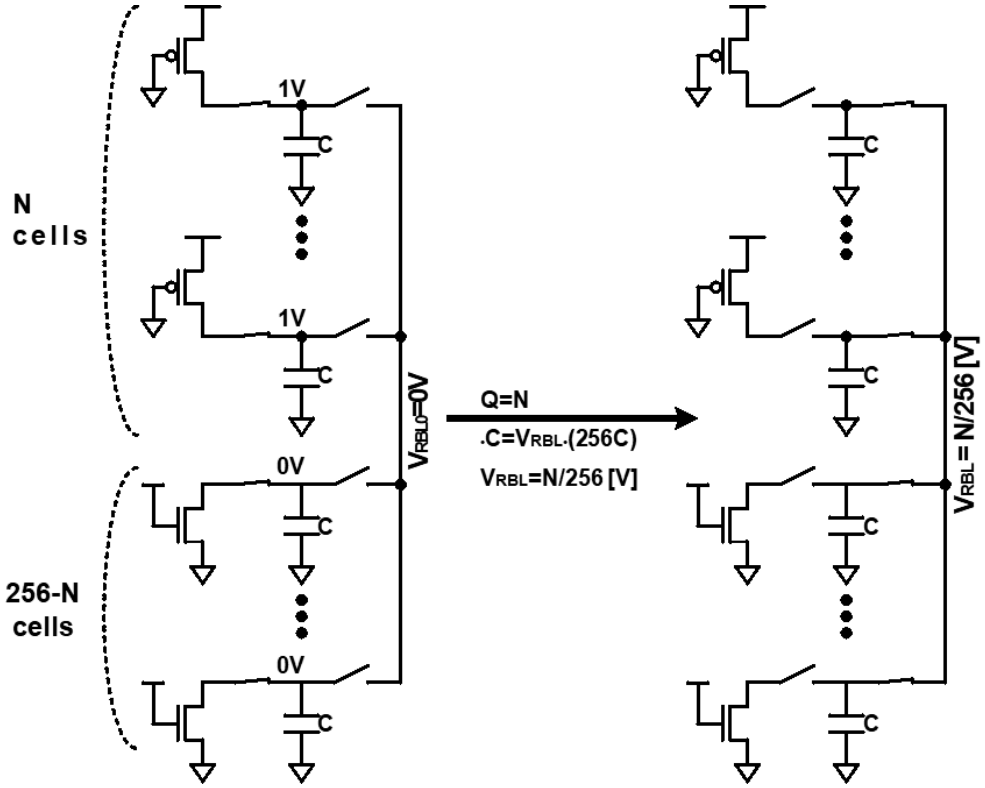

[Jiang, ESSCIRC'19]

# Accumulation Using Charge-Sharing

- During accumulation, all the unit capacitors are connected to the shared bitline.
- Distributed unit capacitors generate averaged bitline voltage.

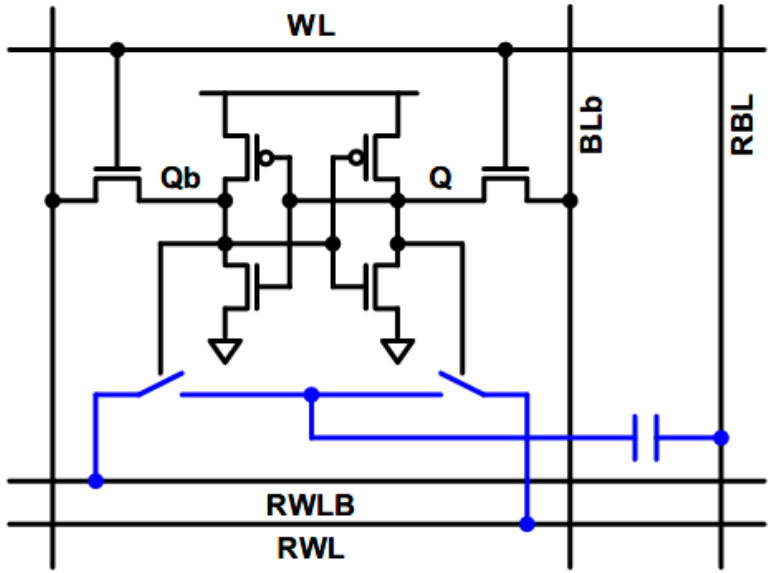


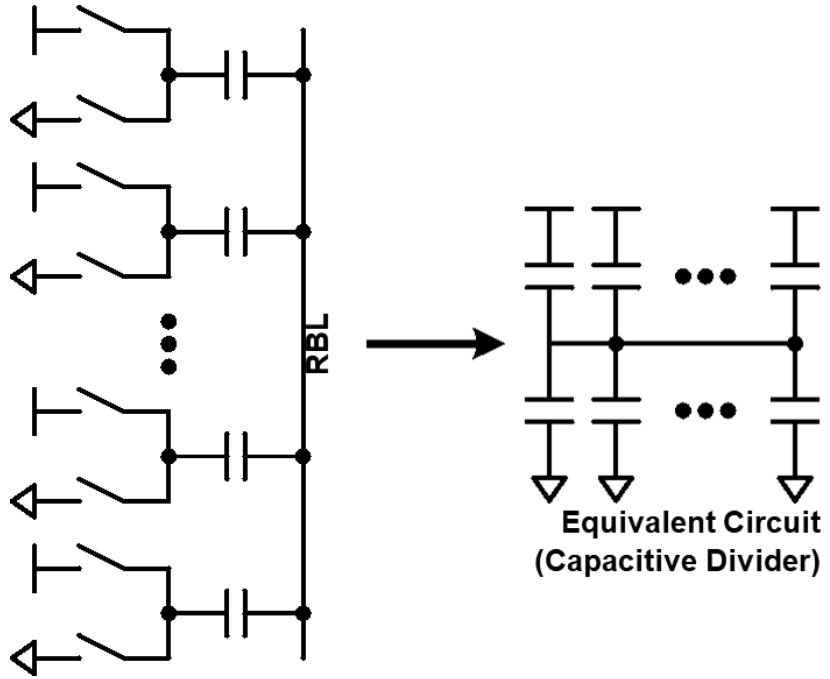

Multiply switches OFF / Accumulate switch ON



# Accumulation Using Capacitive Coupling

- Charge redistribution through capacitive coupling
- Analog MAC result generated through a simple capacitive divider

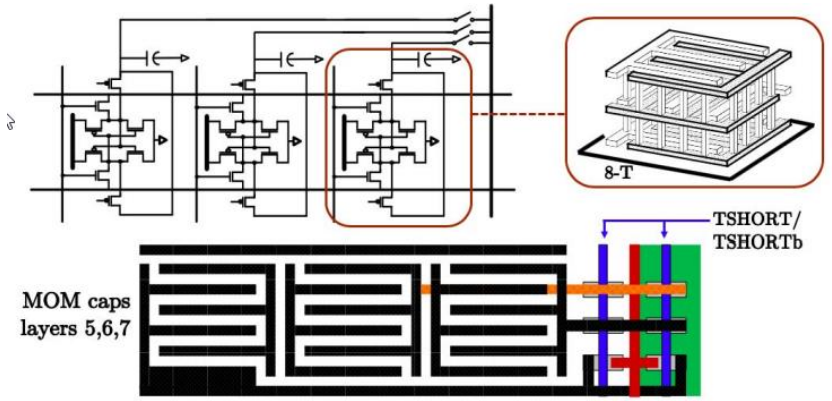


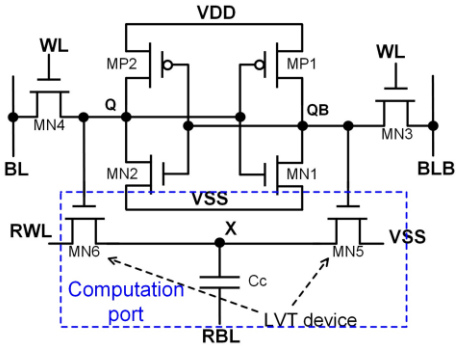



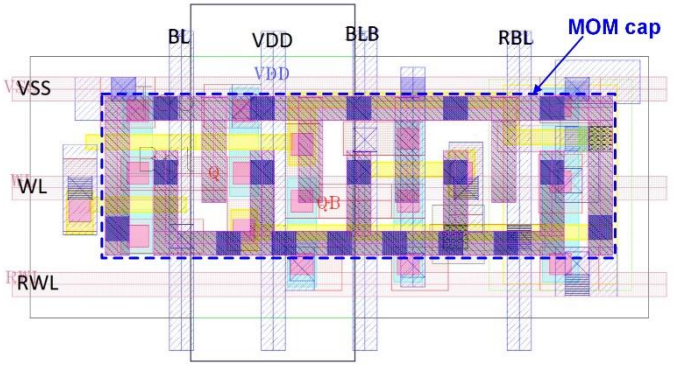
# Bitcell Layouts with Unit Capacitor

- Minimizing area overhead by implementing capacitors on top of transistors
- Area overhead dominated by additional switches
- MOMCAP: 1~2fF per bitcell area
- MOSCAP: for higher capacitance

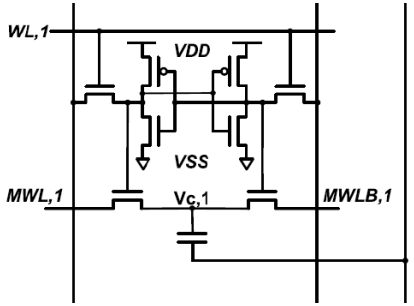
[Valavi, JSSC'19]

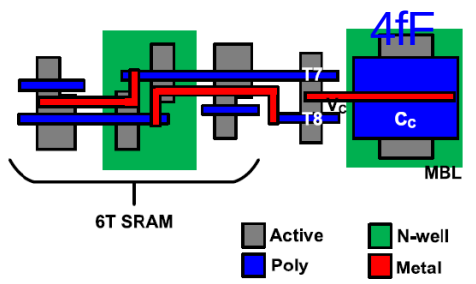



[Sharma, ISCAS'21]




[Jiang, ESSCIRC'19]

# Analog Computing-in-Memory: Summary

- Pros
  - High energy-efficiency by minimizing data transfer between memory and Pes
  - Massive parallelism for achieving high throughput
- Cons
  - Significant power & area overhead in DAC/ADC
  - Limited precision due to analog MAC result
  - Limited reconfigurability
- Advanced Techniques
  - Decoupled bitcell structure for removing disturbance
  - Capacitive bitcell for improving linearity in MAC

# Outline

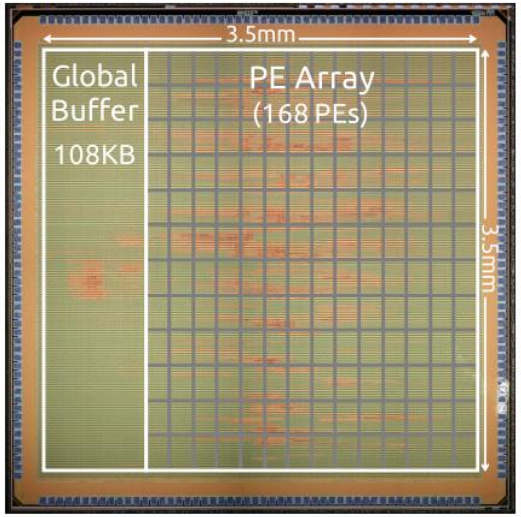
- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
  - Analog CIM
  - Digital CIM
  - ReRAM CIM
- Summary

# Digital Computing-In-Memory

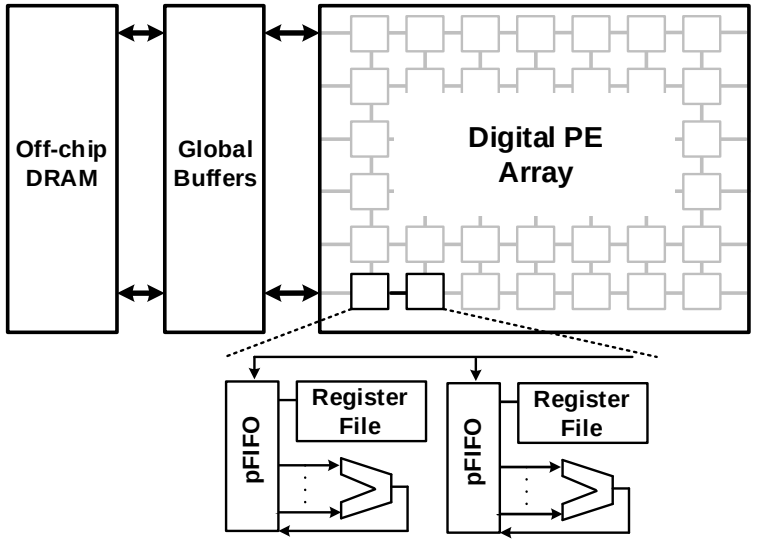
- No degradation in Precision
- Fully Reconfigurable Weights/Inputs
  - By changing column MAC size/operation cycles
- Bit-Serial Computation
  - Reduced hardware area
  - Throughput/latency issue mitigated by parallelism
- Unique Number Representation
  - Two’s complement & binary-weighted signed number

# Digital CNN Accelerator

- Eyeriss: on of the first digital CNN accelerators
- Processing elements implemented with digital circuits
- MAC operation in the digital domain: no accuracy degradation



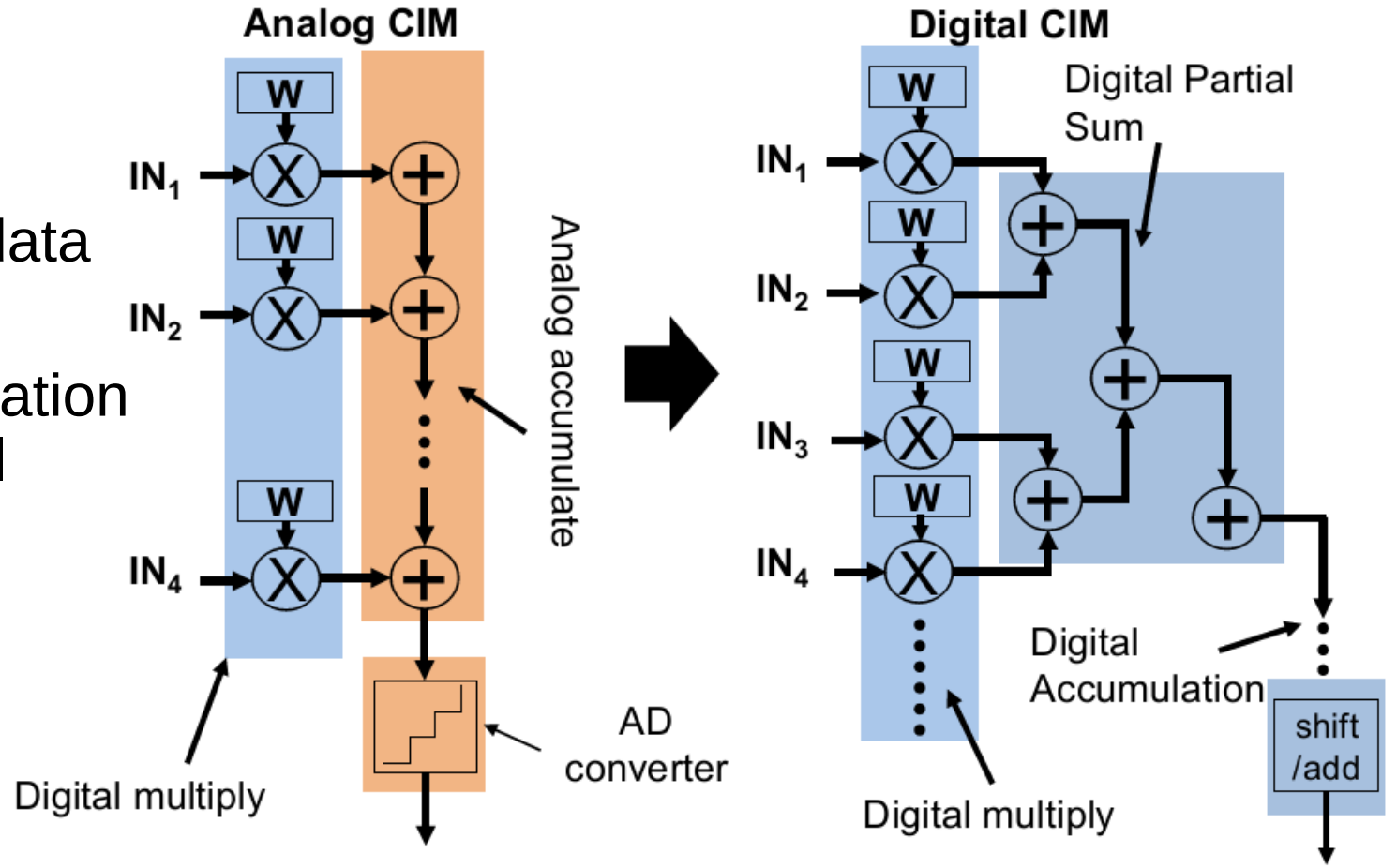

[Chen, ISSCC'16]




[Kim, ESSCIRC'19]

# Analog CIM vs. Digital CIM

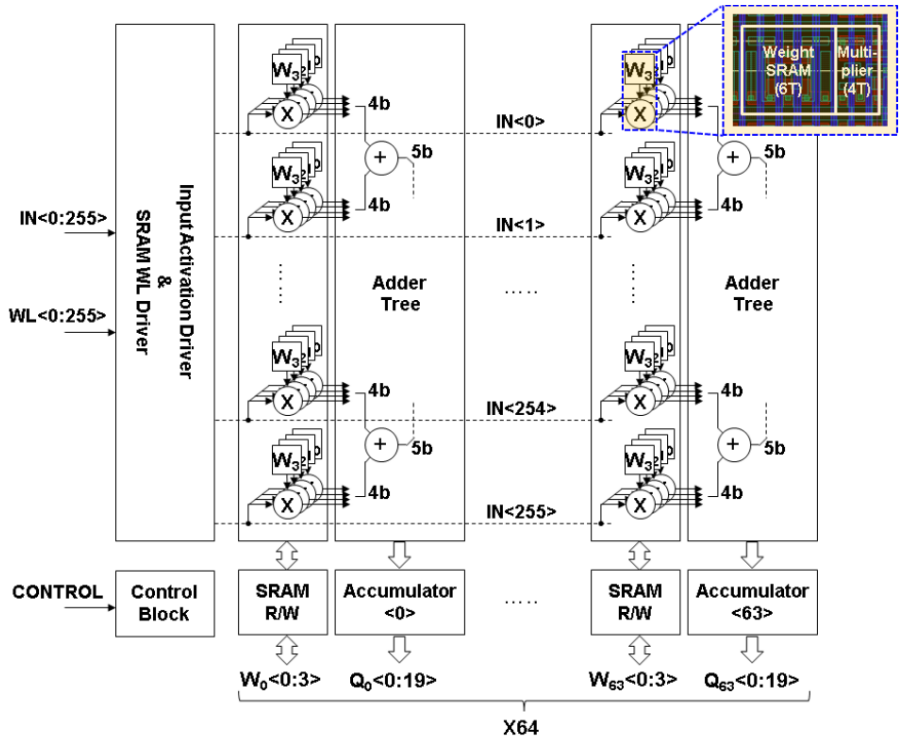
- Digital approach to avoid inaccuracy
- Massively parallel MAC operation to enable weight/data reuse for energy saving
- Energy-efficiency MAC operation by bit-serial multiply/ parallel adder




[Chih, ISSCC'21]

# Digital CIM Macro Architecture

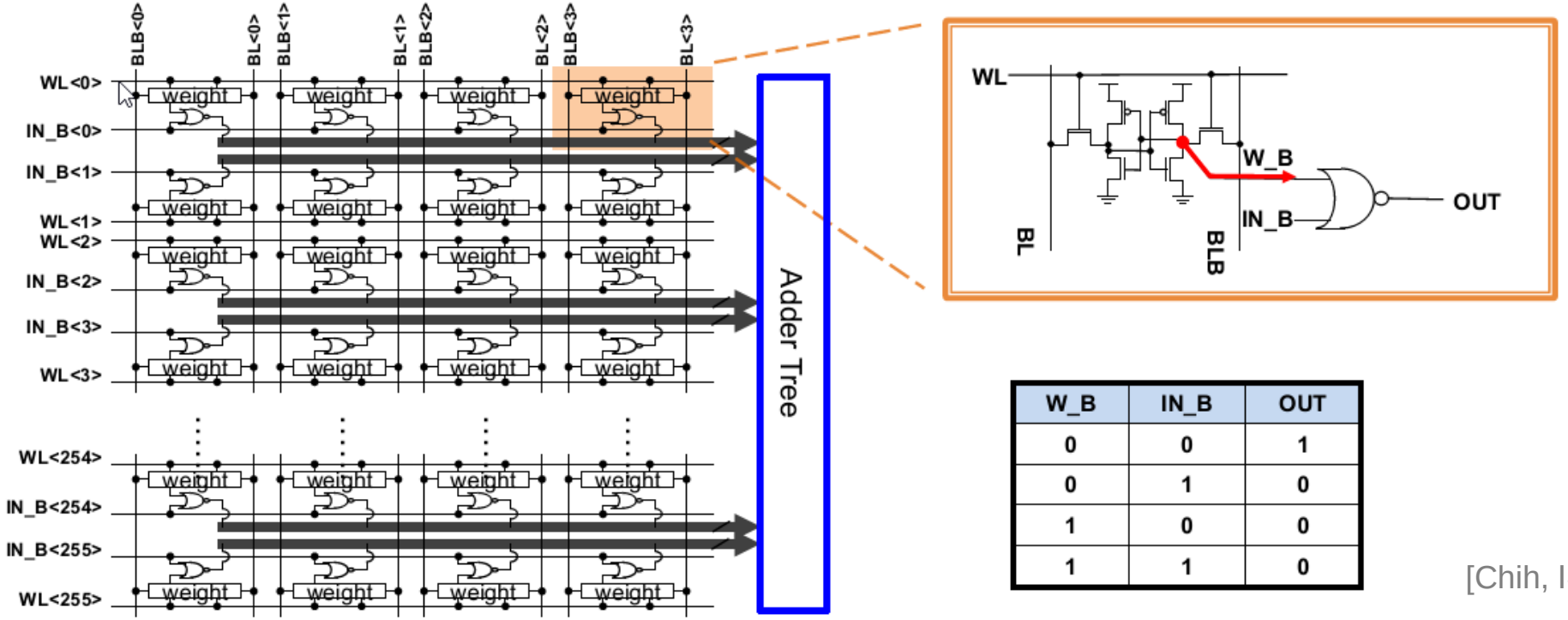
- Fixed weight precision (4bit) and adder-tree
  - Simpler digital computing-in-memory macro
  - Higher parallelism (but less reconfigurability and density is low)




[Chih, ISSCC'21]

# Digital CIM Array Circuit

- 6T SRAM bitcell for weight storage
- NOR for 1-bit multiplier
- NOR output sent to adder tree for accumulation

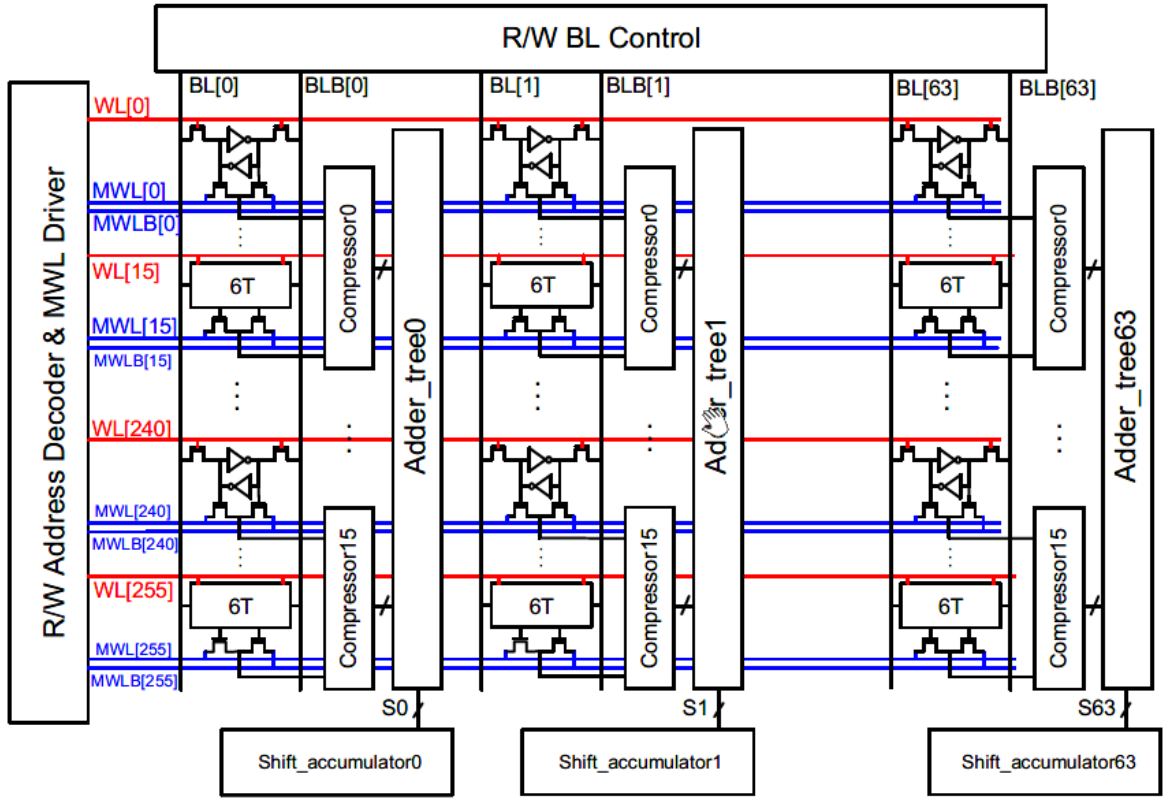



| W_B | IN_B | OUT |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

[Chih, ISSCC'21]

# DIMC Macro Architecture

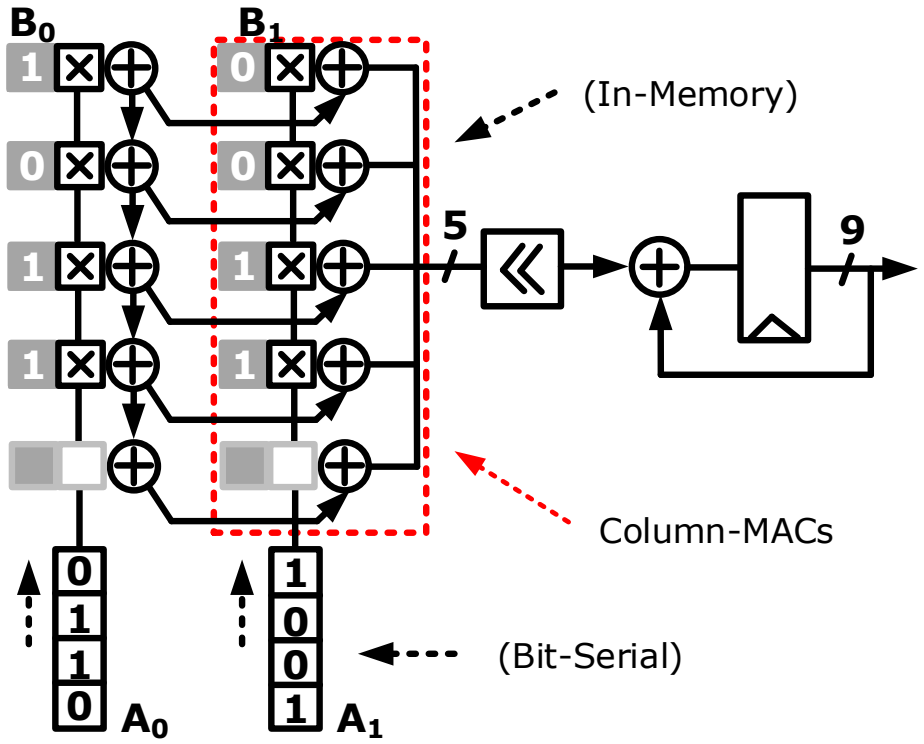
- Macro size: 256x64
- Perform a binary vector-matrix dot-product in one cycle
- A column integrates:
  - 256 binary multiply cells
  - 16 approximate compressors
  - One 16-input adder tree
  - One shift accumulator
- Compressor: small accuracy degradation for high area efficiency




[Wang, ISSCC'22]

# Bit-Serial Digital Computing-in-Memory

- Full Digital Implementation: Free from analog variation and ADC overhead.
- Area/Energy Efficiency Comparable to Analog Accelerators

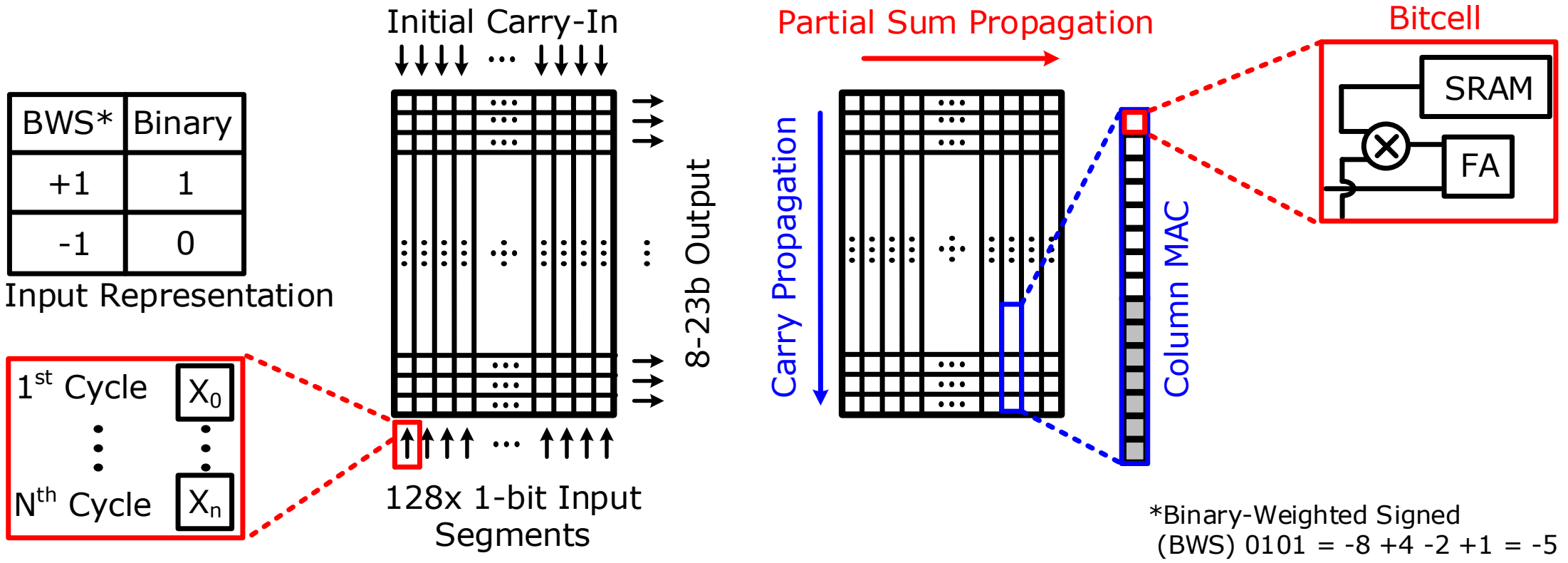



[Kim, ESSCIRC'19]

# Bit-Serial Digital Computing-in-Memory

- Operation diagram

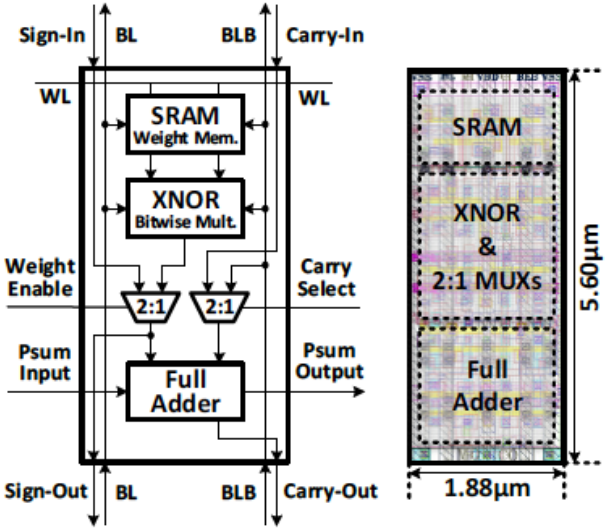
[Kim, ESSCIRC'19]

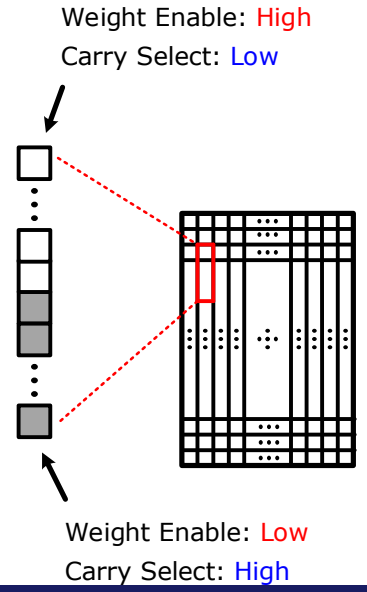


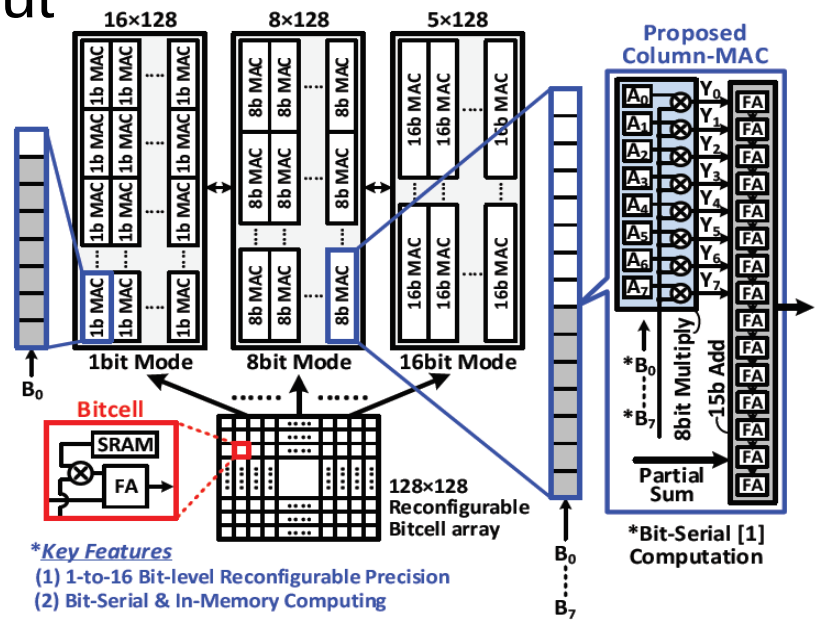
# Bit-Serial Digital Computing-in-Memory

- Bitcell Configuration
  - Weight Enable = 'High': XOR enable
  - Carry Select = 'High': Carry-in from upper bitcell
- Bitcell Mode: Multiple-and-Accumulate (MAC) & Accumulate-only
- Reconfigurable Column-MAC with serialized input

[Kim, ESSCIRC'19]



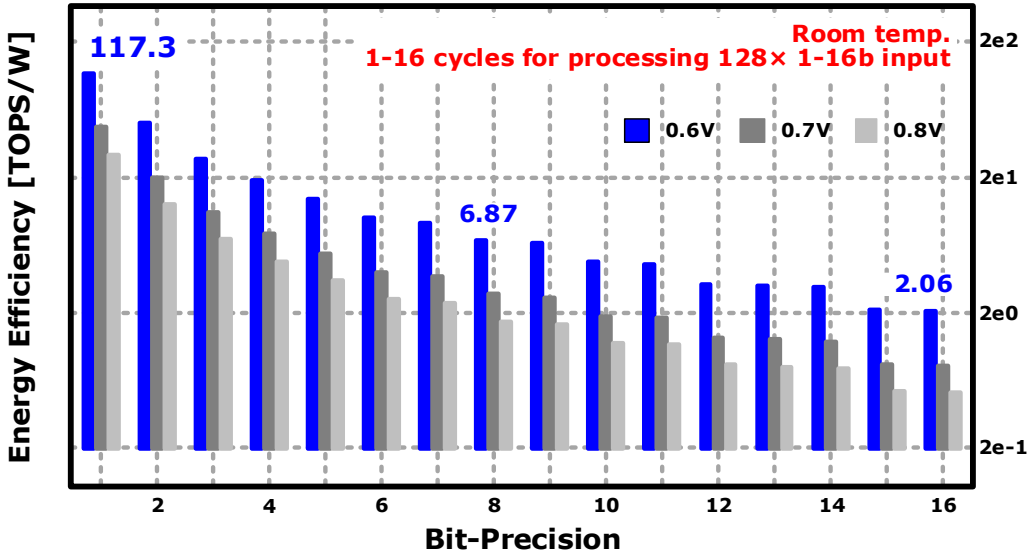



# Bit-Serial Digital Computing-in-Memory

- Reconfigurability
  - Higher energy efficiency at lower bit-precision
  - Energy-aware bit-precision control in IoT devices

| Reconfigurability | 1-16b Weight / 1-16b Input | | | |
|---|---|---|---|---|
| Bitcell & Register | 128×128 Bitcell Array with 16 Register Columns | | | |
| Bit-Precision** | 1b/1b | 1b/16b | 16b/1b | 16b/16b |
| Operation Cycles | 1 | 16 | 1 | 16 |
| Column MACs | 16×128 | 16×128 | 5×128 | 5×128 |
| Max Frequency | 138MHz | 138MHz | 75.8MHz | 75.8MHz |
| Latency | 0.12μs | 1.92μs | 0.22μs | 3.59μs |
| Throughput | 567GOPS | 35.4GOPS | 97GOPS | 6.1GOPS |
| Energy Efficiency | 156TOPS/W | 9.7TOPS/W | 22TOPS/W | 1.4TOPS/W |

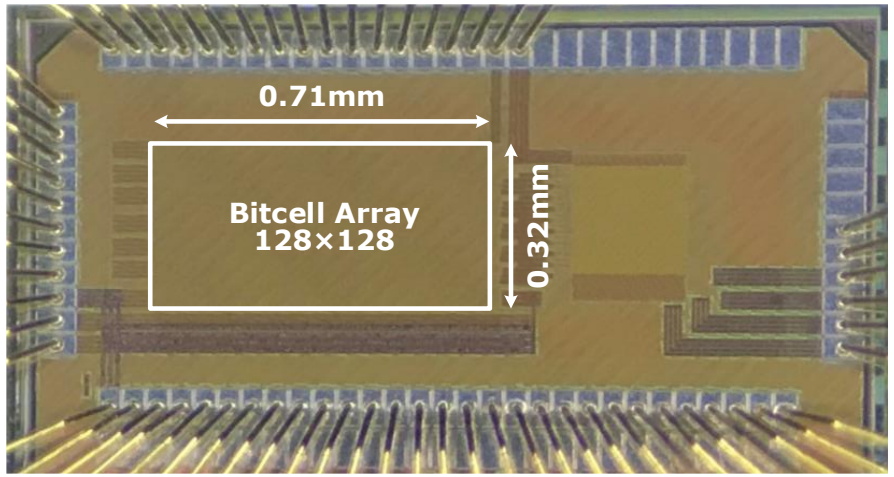
*Simulated (65nm, TT, 0.8V, 50°C) **Bit-precision setting (Weight/Input)




[Kim, ESSCIRC'19]

# Measurement Summary

| | [3]Envision ISSCC'17 | [5]UNPU ISSCC'18 | [7] VLSI'18 | This Work |
|---|---|---|---|---|
| **Technology** | 28nm | 65nm | 14nm | **65nm** |
| **Supply Voltage** | 0.65-1.1V | 0.63-1.1V | 0.28-0.9V | **0.6-0.8V** |
| **Multiply Precision** | 1-16/N bit (N=1,2,4) | 1-to-16bit | FP16b INT8/16b | **1-to-16bit** |
| **Accumulate Precision** | 48/N bit | 32bit | FP32b INT24/48b | **8-to-23bit** |
| **Reconfigurability** | Reconfig. Multiplier | Bit-Serial | Fixed Bits (8,16,24,48) | **Column MAC Bit-Serial** |
| **MAC Array** | N×256 | 12×12 | 4×4 | **16×128(1b) 5×128(16b)** |
| **MAC Area [μm$^2$]** | N/A | N/A | 1480 | **84.2(1b) 242.1(16b)** |
| **Energy per MAC [pJ/MAC]** | N/A | 0.055(1b) 1.26(16b) | N/A | **0.017(1b) 0.78(16b)** |
| **Min. Energy Eff. [TOPS/W]** | 0.26 | 3.08(16b) | 0.55(16b) | **2.06(16b)** |
| **Max. Energy Eff. [TOPS/W]** | 10 | 50.6(1b) | 11.3(8b) | **117.3(1b)** |




Die micrograph

[Kim, ESSCIRC'19]

# Comparison: Analog PIM vs Digital PIM

## Analog PIM

- High energy efficiency
- High throughput with massive parallelism

- Data Conversion overhead
- Limited reconfigurability
- Limited output precision

## Digital PIM

- High output precision
- Good reconfigurability

- Lower performance
- Lower energy efficiency
- Large bitcell size and low density
- Lower throughput than analog PIM

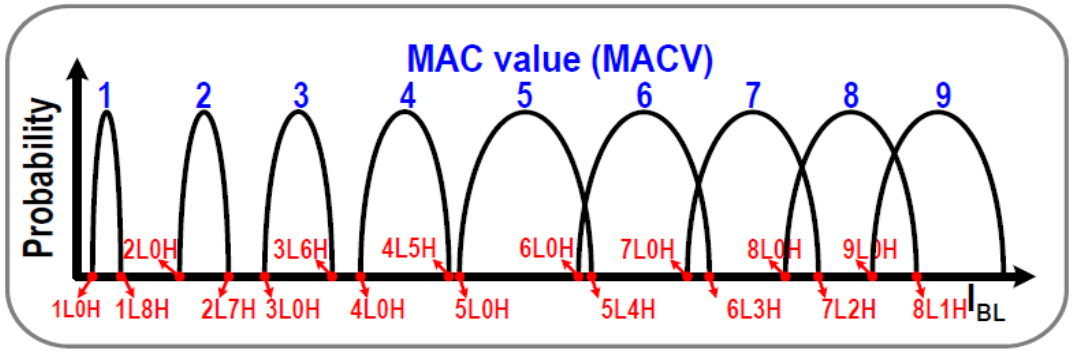
# Outline

- Introduction
- Computing-in-memory Basics and Challenges
- State-of-the-arts Computing-in-memory
  - Analog CIM
  - Digital CIM
  - ReRAM CIM
- Summary

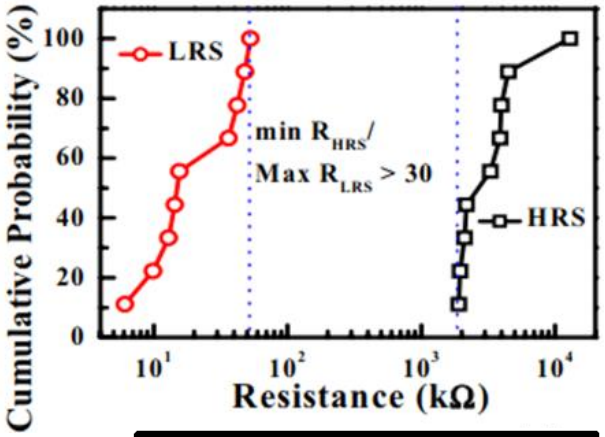
# Challenges in ReRAM CIM

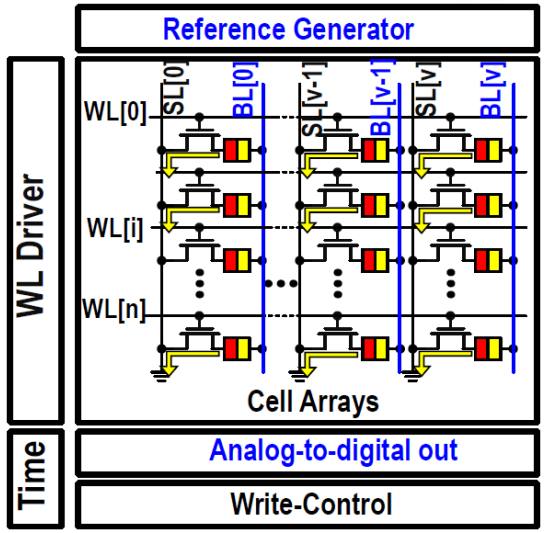
[Chen, ISSCC'18]

- Large variations in resistance
- Non-zero bitline current for data '0' after multiplication
- Overlap between MAC values for small sensing margin




| Input: WL (IN) | Weight (W) | Product (INxW) | $I_{MC}$ |
|---|---|---|---|
| 0 | 0 (HRS) | 0 | 0 |
| 0 | 1 (LRS) | 0 | 0 |
| 1 | 1 (LRS) | 1 | $I_{LRS}$ |
| 1 | 0 (HRS) | 0 | $I_{HRS}$ |

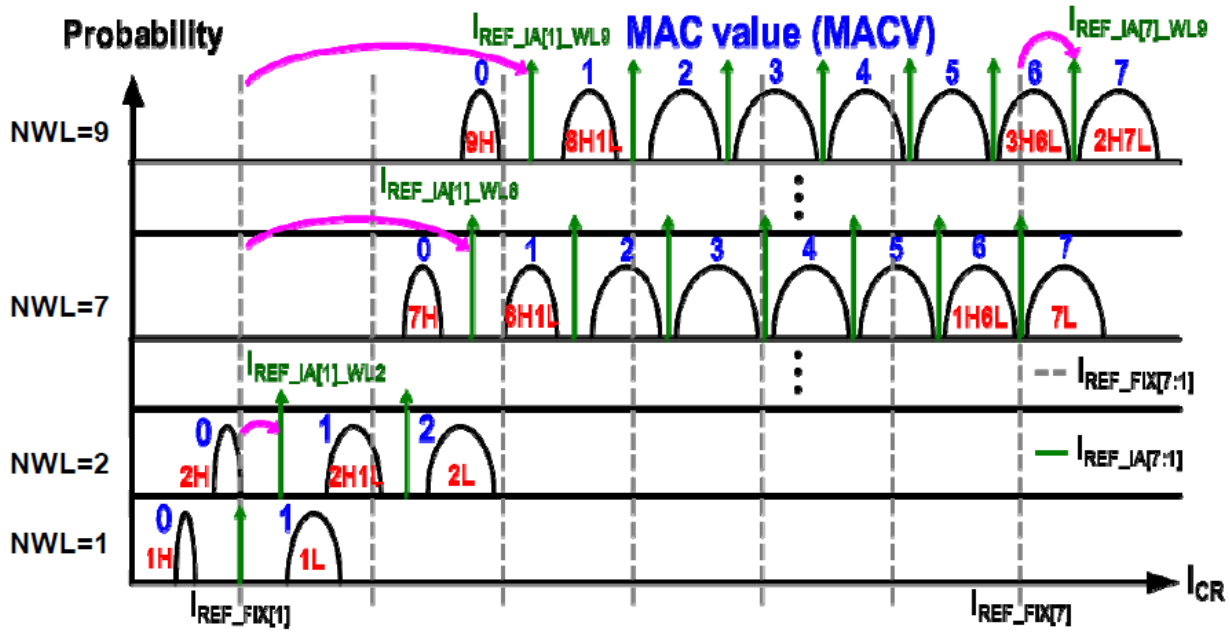


# Input-aware Dynamic Reference

[Chen, ISSCC'18]

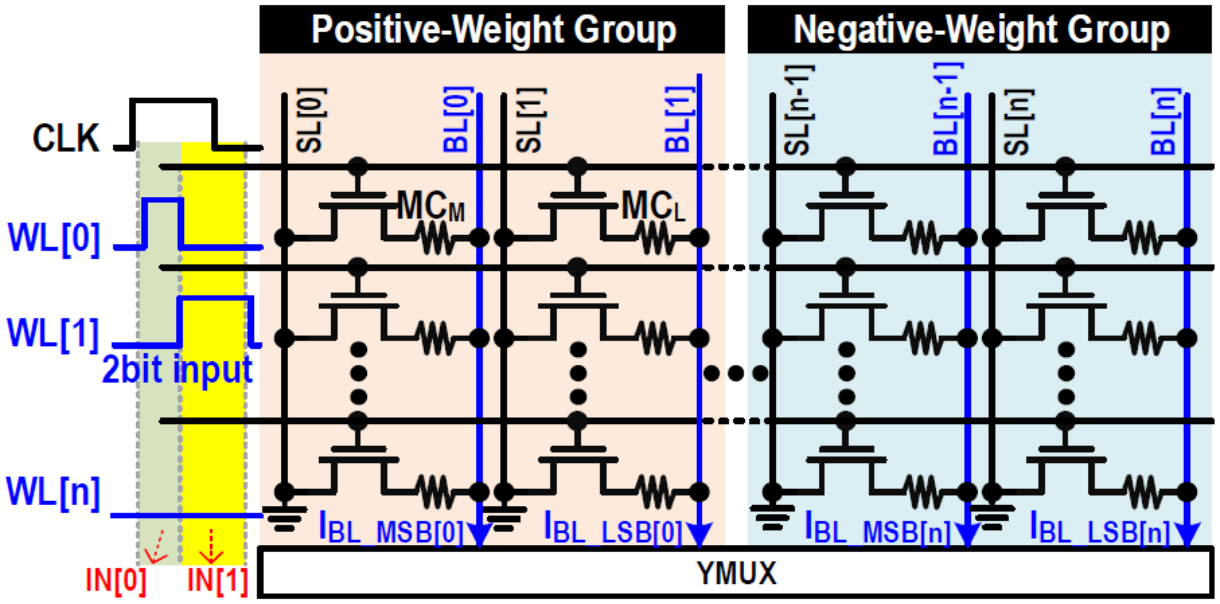
- Input-aware reference current
- Reference current separation
- Input aware replica rows



# Serial-Input Non-Weighted Product

[Xue, ISSCC'19]

- Multibit weight storage in 1T1R cells
  - Non-weighted current accumulation in cell array
  - Peripheral circuits deal with weighted values



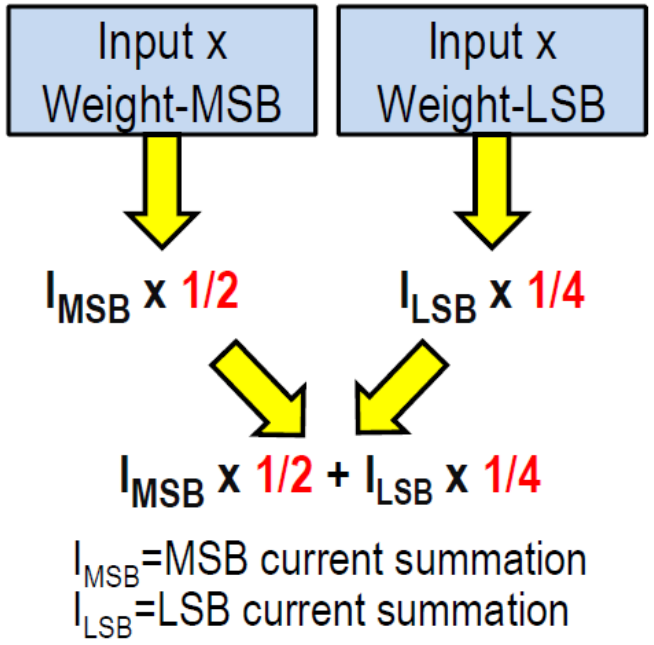

| Positive Weight Group | | |
|---|---|---|
| $MC_M$ | $MC_L$ | Weight value(W) |
| LRS(+2) | LRS(+1) | +3 |
| LRS(+2) | HRS(0) | +2 |
| HRS(0) | LRS(+1) | +1 |
| HRS(0) | HRS(0) | 0 |

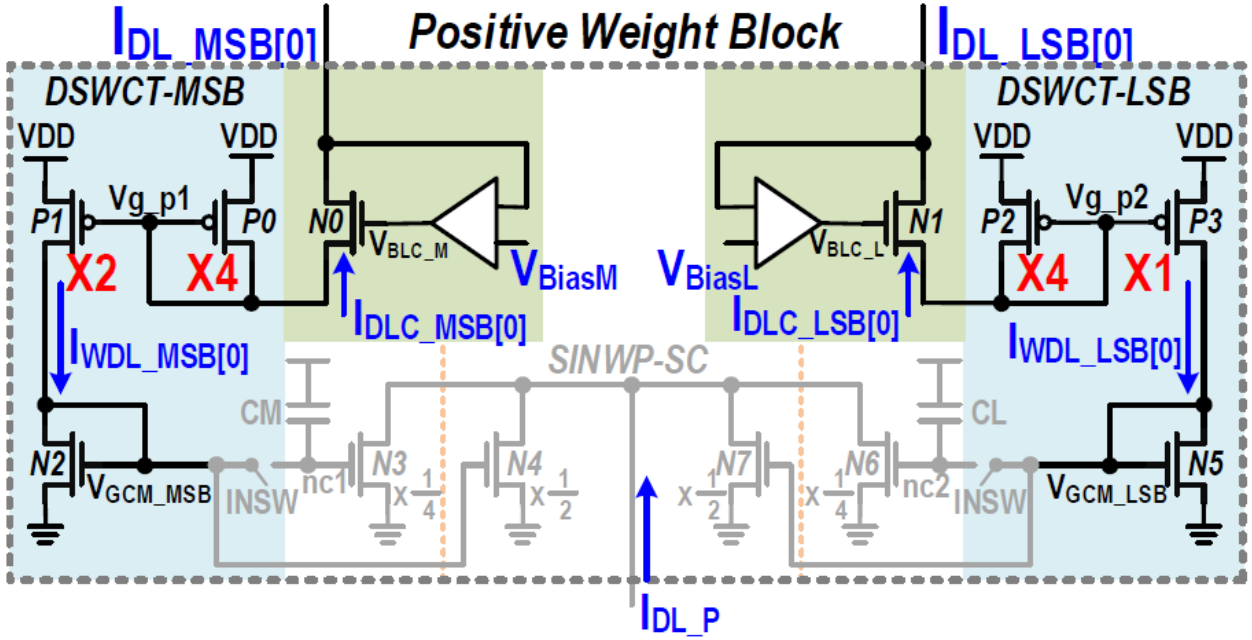
| Negative Weight Group | | |
|---|---|---|
| $MC_M$ | $MC_L$ | Weight value(W) |
| HRS(0) | HRS(0) | 0 |
| HRS(0) | LRS(-1) | -1 |
| LRS(-2) | HRS(0) | -2 |
| LRS(-2) | LRS(-1) | -3 |

# Serial-Input Non-Weighted Product

[Xue, ISSCC'19]

- Down-scaling current mirror ratio
  - Process 2-bits weight values
  - Reduce summation current range and read-path current

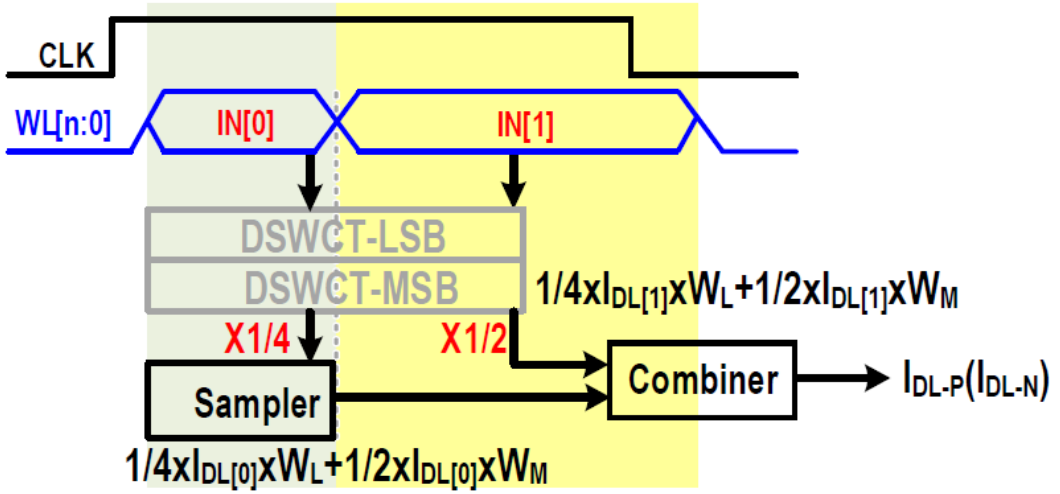


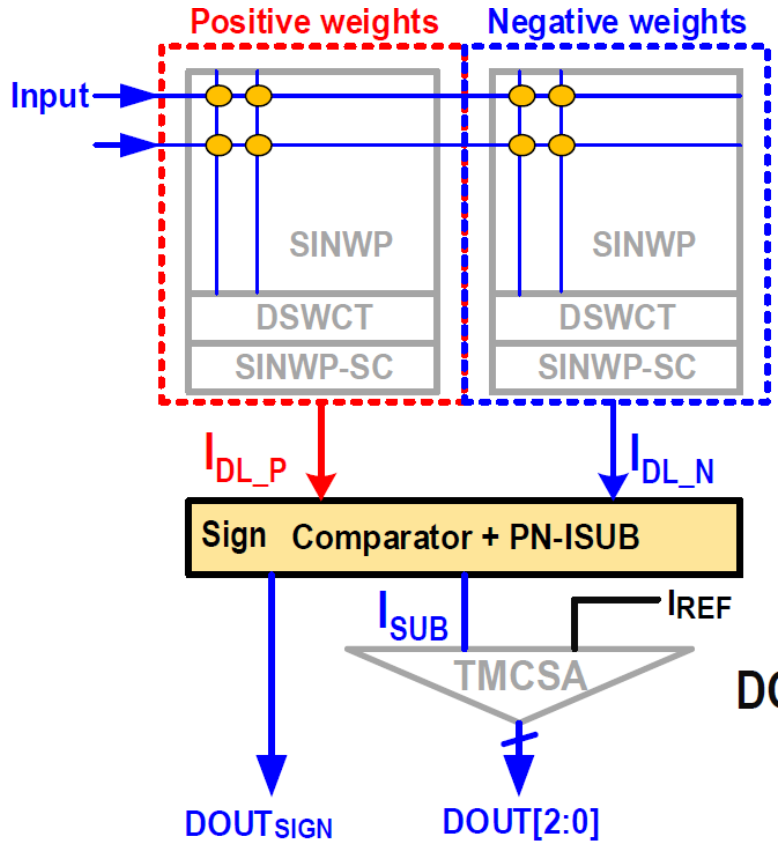
# Serial-Input Non-Weighted Product

[Xue, ISSCC'19]

- SINWP - Sampler and Combiner
  - Phase1: sample MAC of IN[0]
  - Phase2: combine MACs of IN[1] and IN[0]
  - Current subtraction with reduced range




$$I_{DL\text{-}P}(I_{DL\text{-}N}) = 1/4 \times [1/4\times(I_{DL[0]}+I_{DL[1]})\times W_L + 1/2\times(I_{DL[0]}+I_{DL[1]})\times W_M] + 1/2 \times [1/4\times(I_{DL[0]}+I_{DL[1]})\times W_L + 1/2\times(I_{DL[0]}+I_{DL[1]})\times W_M]$$



# Summary

- Neural Networks have promising opportunities in various energy-constrained smart applications.
- Computing-in-Memory is a critical research area for improving energy efficiency of neural networks by orders of magnitude.
- Analog and digital computing-in-memory designs have its own advantages and limitations.
- Computing-in-memory using emerging non-volatile memory is promising in energy-constrained IoT applications.
- Computing-in-memory design is not mature yet and needs more comprehensive research.

# References


- A. Biswas and A. P. Chandrakasan, “CONV-SRAM: An energy-efficient SRAM with in-memory dot-product computation for low-power convolutional neural networks,” IEEE JSSC, vol. 54, no. 1, pp. 217–230, Jan. 2019
- V. Sze, “Efficient Processing of Deep Neural Network: from Algorithms to Hardware Architectures,” in Tutorial, 35th Conference on Neural Information Processing Systems (NIPS), Dec. 2019.
- M. Horowitz, "Computing's energy problem (and what we can do about it)," ISSCC, pp. 10-14, 2014.
- Y. -H. Chen et al., "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," ISSCC, pp. 262-263, 2016.
- L. Deng, et al., "Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey," in Proceedings of the IEEE, vol. 108, no. 4, pp. 485-532, April 2020.
- J.-Y. Kim, B. Kim and T. Kim, “Revolutionizing AI Hardware with Processing-in-Memory: From Architectures to Circuits,” in Tutorial, IEEE ISCAS, May 2021
- W. -H. Chen et al., "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," ISSCC, pp. 494-496, 2018.
- Q. Dong et al., “A 351TOPS/W and 372.4GOPS Compute-in-Memory SRAM Macro in 7nm FinFET CMOS for Machine-Learning Applications,” ISSCC, pp. 242-243, Feb. 2020.

# References


- X. Si et al., "A Twin-8T SRAM Computation-in-Memory Unit-Macro for Multibit CNN-Based AI Edge Processors," IEEE JSSC, vol. 55, no. 1, pp. 189-202, Jan. 2020.
- Z. Jiang et al., "C3SRAM: In-Memory-Computing SRAM Macro Based on Capacitive-Coupling Computing," ESSCIRC, pp. 131-134, Sep. 2019.
- S. Yin et al., “XNOR-SRAM: In-Memory Computing SRAM Macro for Binary/Ternary Deep Neural Networks,” JSSC, vol. 55, no. 6, pp. 1733-1743, June 2020.
- C. Yu et al., "A 16K Current-Based 8T SRAM Compute-In-Memory Macro with Decoupled Read/Write and 1-5bit Column ADC," CICC, pp. 1-4, Apr. 2020.
- M. Kang et al., "A Multi-Functional In-Memory Inference Processor Using a Standard 6T SRAM Array," JSSC, vol. 53, no. 2, pp. 642-655, Feb. 2018.
- J. Zhang, Z. Wang, and N. Verma, “In-memory computation of a machine-learning classifier in a standard 6T SRAM array,” JSSC, vol. 52, no. 4, pp. 915–924, Apr. 2017.
- Q. Liu et al., "A Fully Integrated Analog ReRAM Based 78.4TOPS/W Compute-In-Memory Chip with Fully Parallel MAC Computing," ISSCC, pp. 500-502, 2020.
- H. Kim, Q. Chen and B. Kim, "A 16K SRAM-Based Mixed-Signal In-Memory Computing Macro Featuring Voltage-Mode Accumulator and Row-by-Row ADC," A-SSCC, pp. 35-36, Nov. 2019.

# References


- C. Yu et al., "A Zero-Skipping Reconfigurable SRAM In-Memory Computing Macro with Binary-Searching ADC," ESSCIRC, pp. 131-134, 2021.
- H. Valavi et al., "A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute," JSSC, vol. 54, no. 6, pp. 1789-1799, June 2019.
- Y. Chih et al., "An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," ISSCC, pp. 252-254, Feb. 2021.
- D. Wang et al., "DIMC: 2219TOPS/W 2569F2/b Digital In-Memory Computing Macro in 28nm Based on Approximate Arithmetic Hardware," ISSCC, pp. 266-268, 2022.
- W. -H. Chen et al., "A 65nm 1Mb non-volatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," ISSCC, Feb. 2018, pp. 494-496
- C. -X. Xue et al., "A 1Mb Multibit ReRAM Computing-In-Memory Macro with 14.6ns Parallel MAC Computing Time for CNN Based AI Edge Processors," ISSCC, Feb. 2019, pp. 388-390