A Word-Level Multi-Precision Systolic Array Architecture for Accelerating Deep Neural Networks

Xinjun Zhou, Fen Ge, Hui Chen, Weiqiang Liu
Nanjing University of Aeronautics and Astronautics, Nanjing, China
zhouxinjun@nuaa.edu.cn

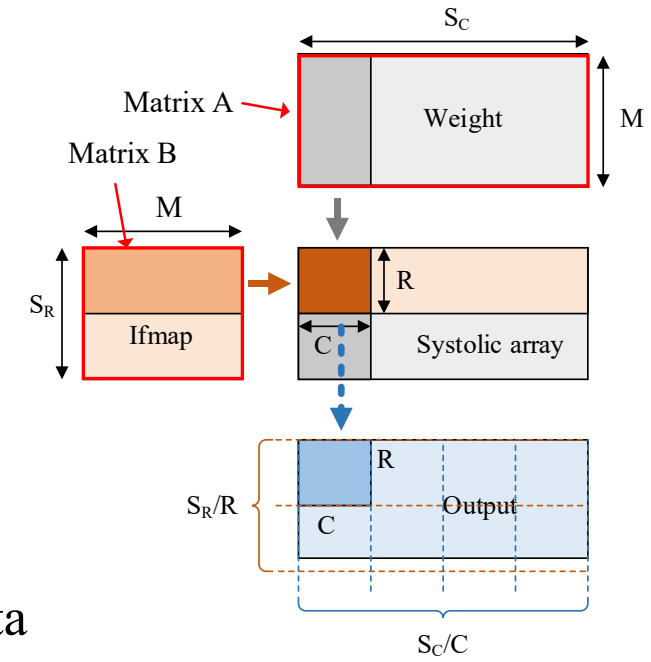
29 June 2026

Contents

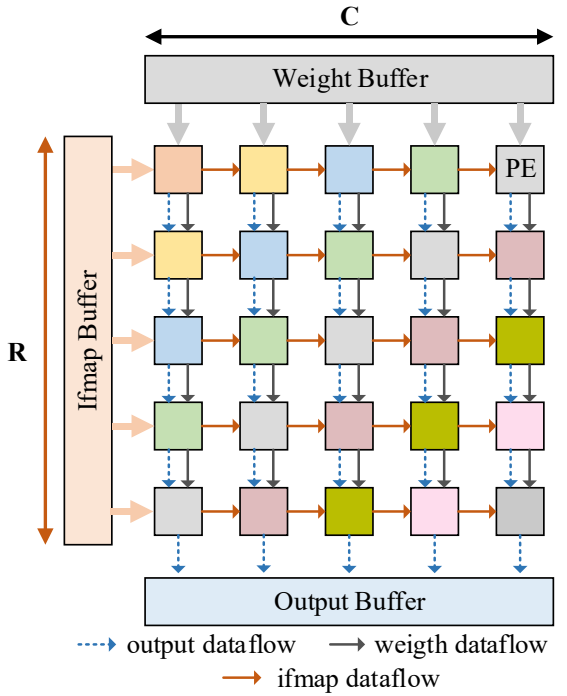
Introduction and Motivation

➢ Why DNN Acceleration Relies on Systolic Arrays?

- DNN computation is mainly composed of matrix multiply–accumulate operations such as GEMM and convolution.
- Systolic arrays enable regular and scalable data reuse through local communication between adjacent PEs.
- Compared with frequent DRAM accesses, on-chip data movement can significantly reduce energy consumption and control overhead.



(a) Matrix Computations in Systolic Array



(b) Conventional Systolic Array

Introduction and Motivation

➢ Analysis of Systolic Array Dataflow and Runtime Modeling

Conventional systolic array is used to derive the runtime as:

OS mode: $T_{os} = (2R + C + M - 2) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

IS mode: $T_{IS} = (2R + C + S_R - 2) \cdot \left\lceil \frac{M}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

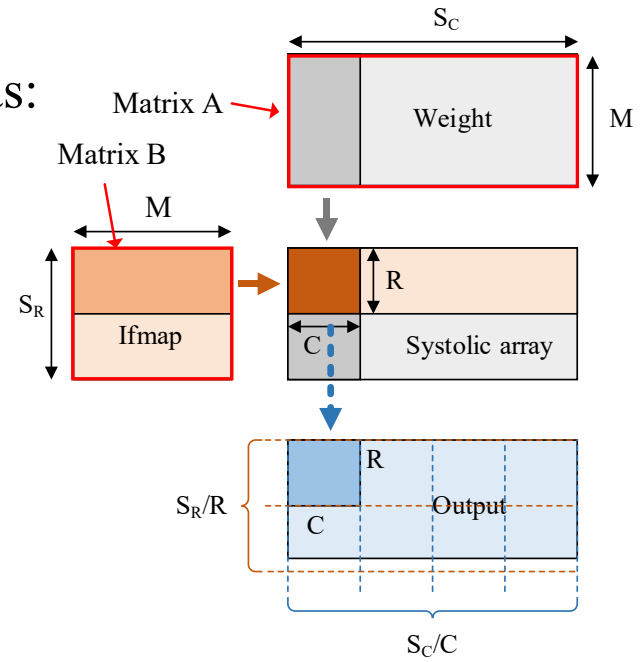
WS mode: $T_{ws} = (2R + C + S_R - 2) \cdot \left\lceil \frac{S_C}{R} \right\rceil \cdot \left\lceil \frac{M}{C} \right\rceil$

Pre-filling time as: $T_p = (R + C - 1) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

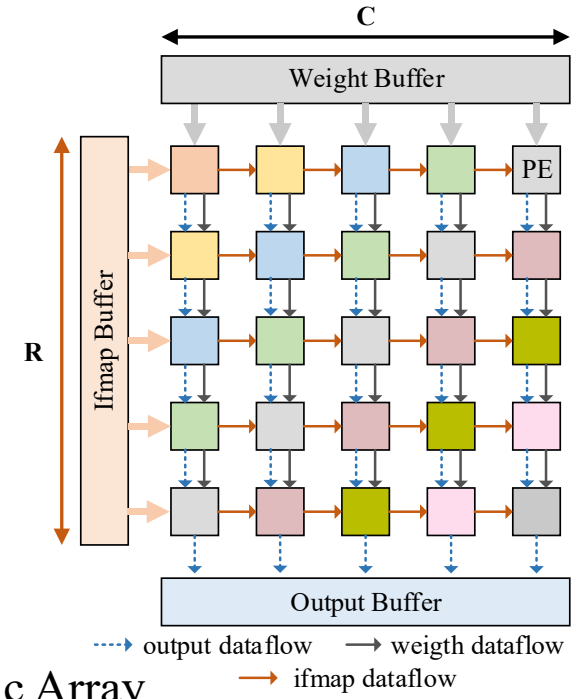
Computation time for each PE as: $T_c = (M - 1) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

Output readout time as: $T_o = R \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

Total time = pre-fill + compute + output



(a) Matrix Computations in Systolic Array

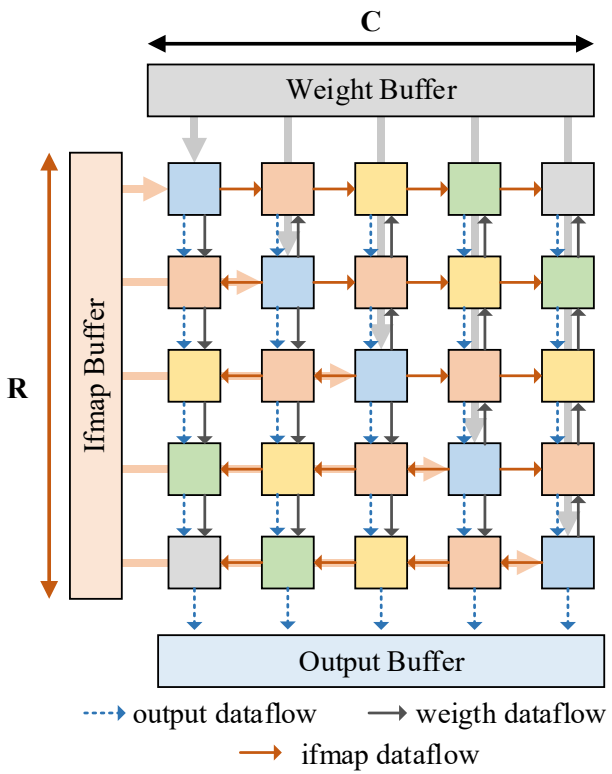


(b) Conventional Systolic Array

Small tiled matrices make pre-filling time a major runtime bottleneck.

Introduction and Motivation

➢ Pre-filling time improvement



Axon

Pre-filling time as:

$$T_p = \max(R, C) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$$

Systolic array runtime as:

$$T_{os} = (R + M + \max(R, C) - 1) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$$

$$T_{IS} = (R + S_R + \max(R, C) - 1) \cdot \left\lceil \frac{M}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$$

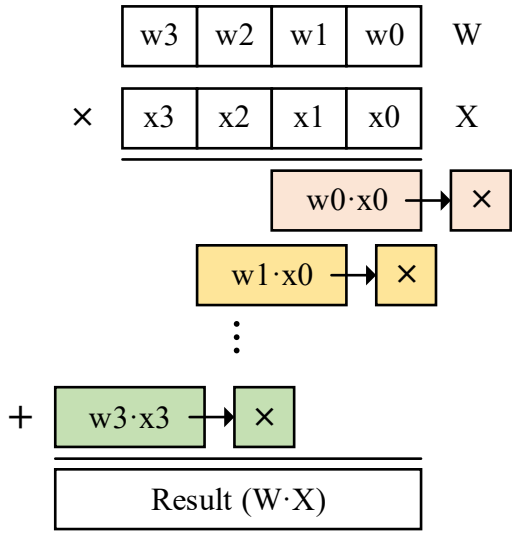
$$T_{ws} = (R + S_R + \max(R, C) - 1) \cdot \left\lceil \frac{S_C}{R} \right\rceil \cdot \left\lceil \frac{M}{C} \right\rceil$$

Axon still leaves room for further pre-filling time reduction

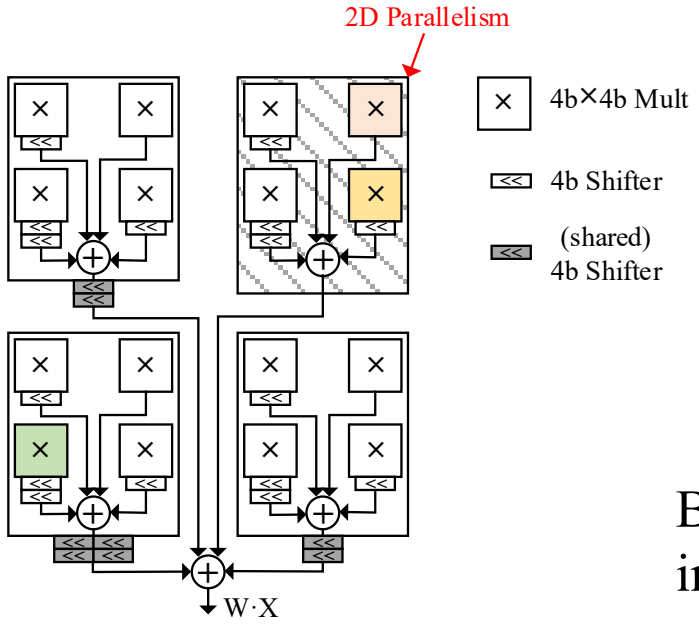
Nayan, Md Mizanur Rahaman, et al. "Axon: A novel systolic array architecture for improved run time and energy efficient gemm and conv operation with on-chip im2col." *2025 Design, Automation & Test in Europe Conference (DATE)*. IEEE, 2025.

Introduction and Motivation

➢ Precision-Scalable PE



16b×16b multiplication



Fusion unit

$$X = \sum_{i=0}^{3} x_i 2^{4i} \qquad W = \sum_{j=0}^{3} w_j 2^{4j}$$

$$X \cdot W = \sum_{i=0}^{3} \sum_{j=0}^{3} (x_i \cdot w_j) 2^{4(i+j)}$$

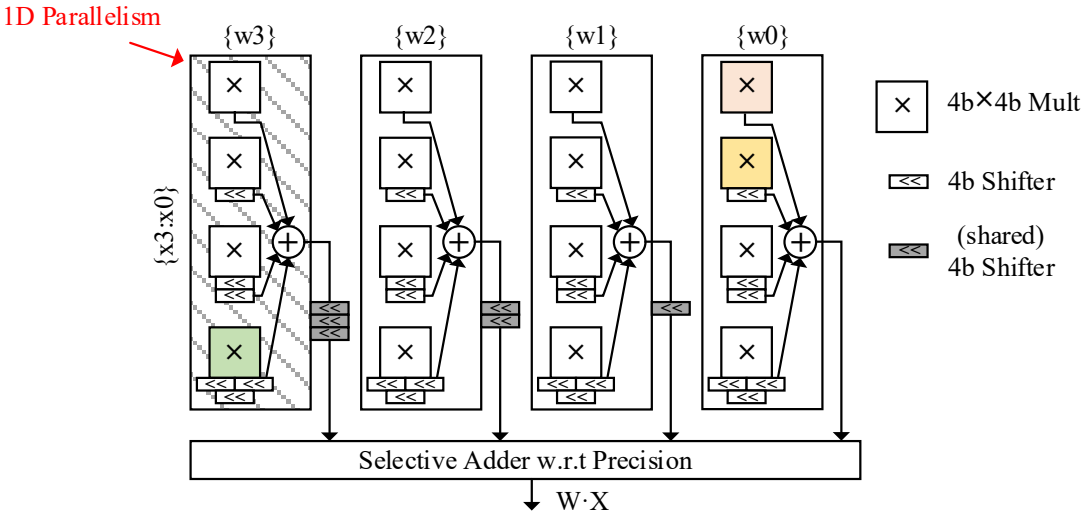
BitFusion decomposes high-precision multiplications into multiple low-precision operations using an array of small multipliers, enabling support for 4-bit, 8-bit, and 16- bit computations through dynamic fusion of compute units

BitFusion's 2D parallelism structure suffers from insufficient MAC utilization at low precision

Sharma, Hardik, et al. "Bit fusion: Bit-level dynamically composable architecture for accelerating deep neural network." *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2018.

Introduction and Motivation

➢ MAC utilization improvement



Flexblock core

Flexblock improves the MAC organization by transforming the original 2D parallelism into a 1D parallelism structure

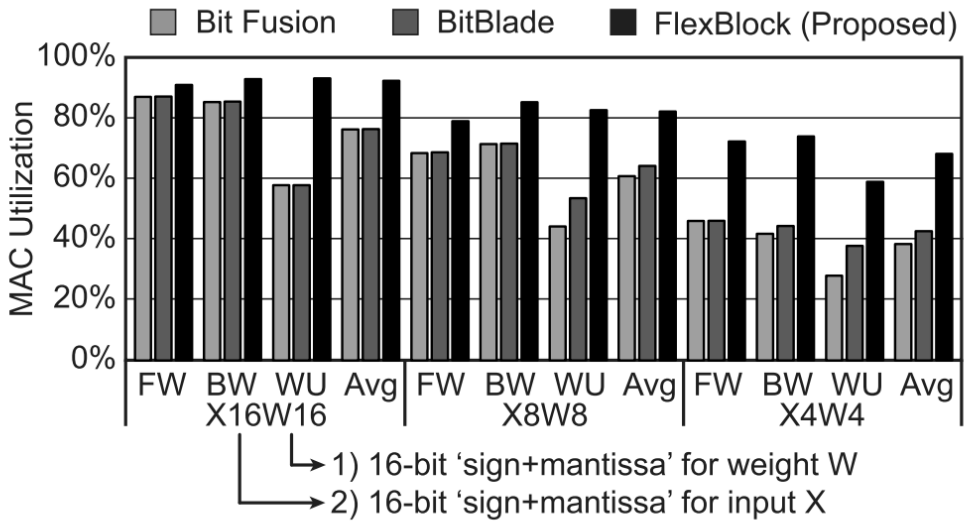
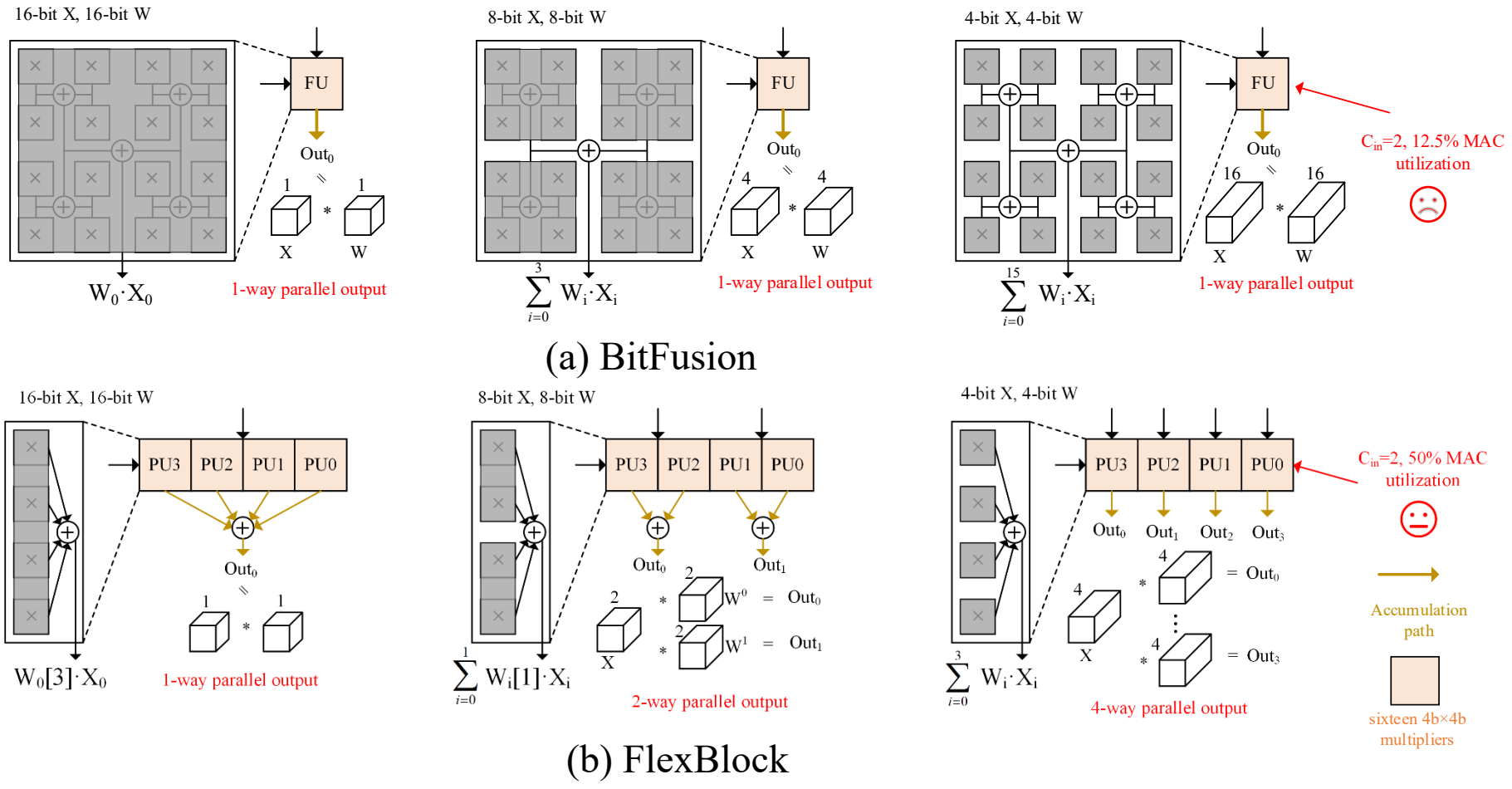


Fig. 5. Comparison of the MAC utilization during the training of MobileNetV1 with mini-batch size of 16 using various precision-scalable MAC arrays (Bit Fusion [11], BitBlade [12] and the proposed FlexBlock).

Introduction and Motivation

➢ Why MAC utilization improvement?



FlexBlock improves MAC utilization, but still achieves only 50% utilization in 4-bit mode.

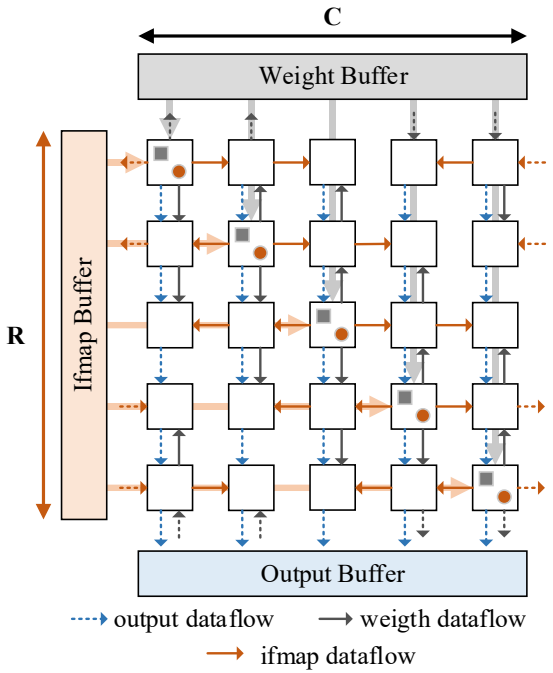
Noh, Seock-Hwan, et al. "Flexblock: A flexible dnn training accelerator with multi-mode block floating point support." *IEEE Transactions on Computers* 72.9 (2023): 2522-2535.

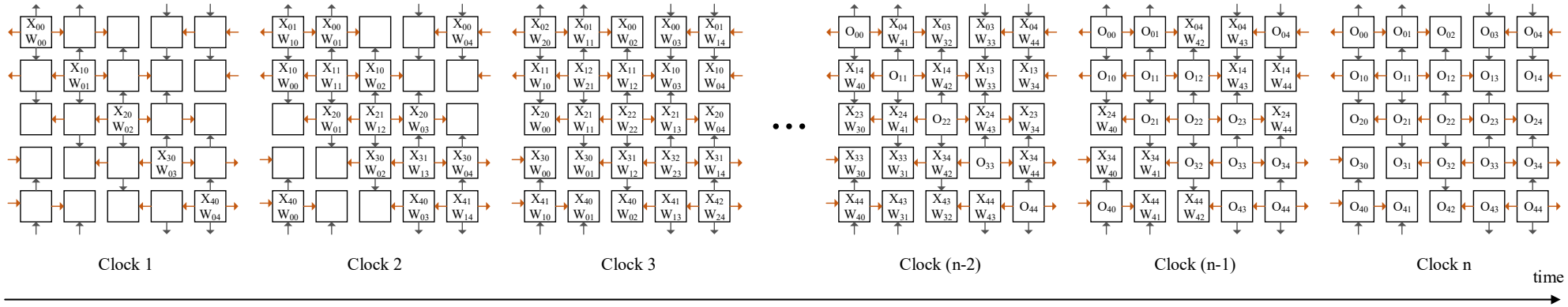
Contents

- Introduction and Motivation
- Proposed Approach
- Hardware Architecture
- Evaluation
- Conclusion

Proposed Approach

➢ Ring-based Diagonal Dataflow





Temporal evolution of the ring-based diagonal dataflow

Proposed ring-based diagonal dataflow

Overall time as: $T_{os} = (R + M + \lceil \max(R, C) \rceil - 1) \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

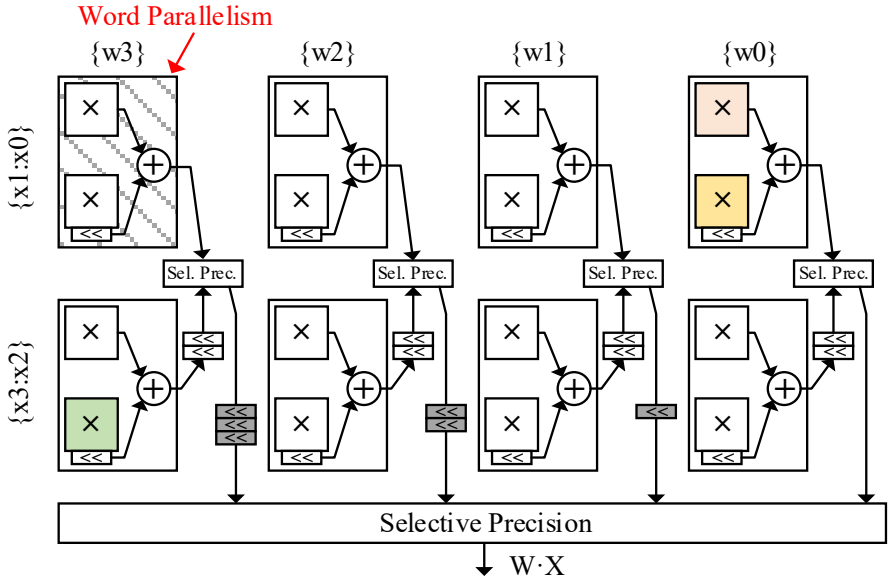
$T_{IS} = (R + S_R + \lceil \max(R, C) \rceil - 1) \cdot \left\lceil \frac{M}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

Pre-filling latency is further reduced as: $T_p = \lceil \max(R, C)/2 \rceil \cdot \left\lceil \frac{S_R}{R} \right\rceil \cdot \left\lceil \frac{S_C}{C} \right\rceil$

$T_{ws} = (R + S_R + \lceil \max(R, C) \rceil - 1) \cdot \left\lceil \frac{S_C}{R} \right\rceil \cdot \left\lceil \frac{M}{C} \right\rceil$

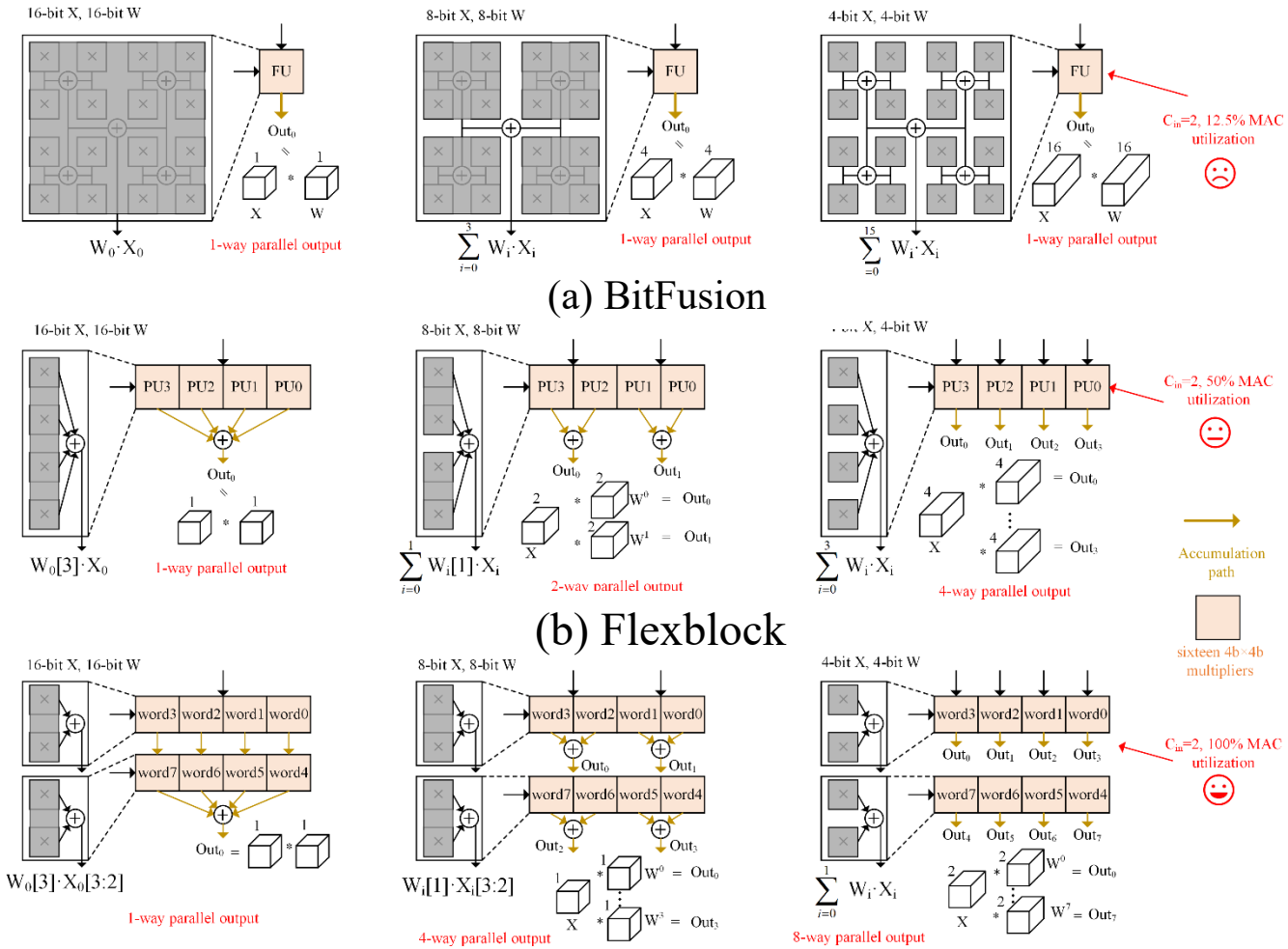
Proposed Approach

➢ Word-parallel PE



Proposed word-parallel PE

Enables 100% MAC utilization even in 4-bit mode



(a) BitFusion

(b) Flexblock

(c) word-parallel PE

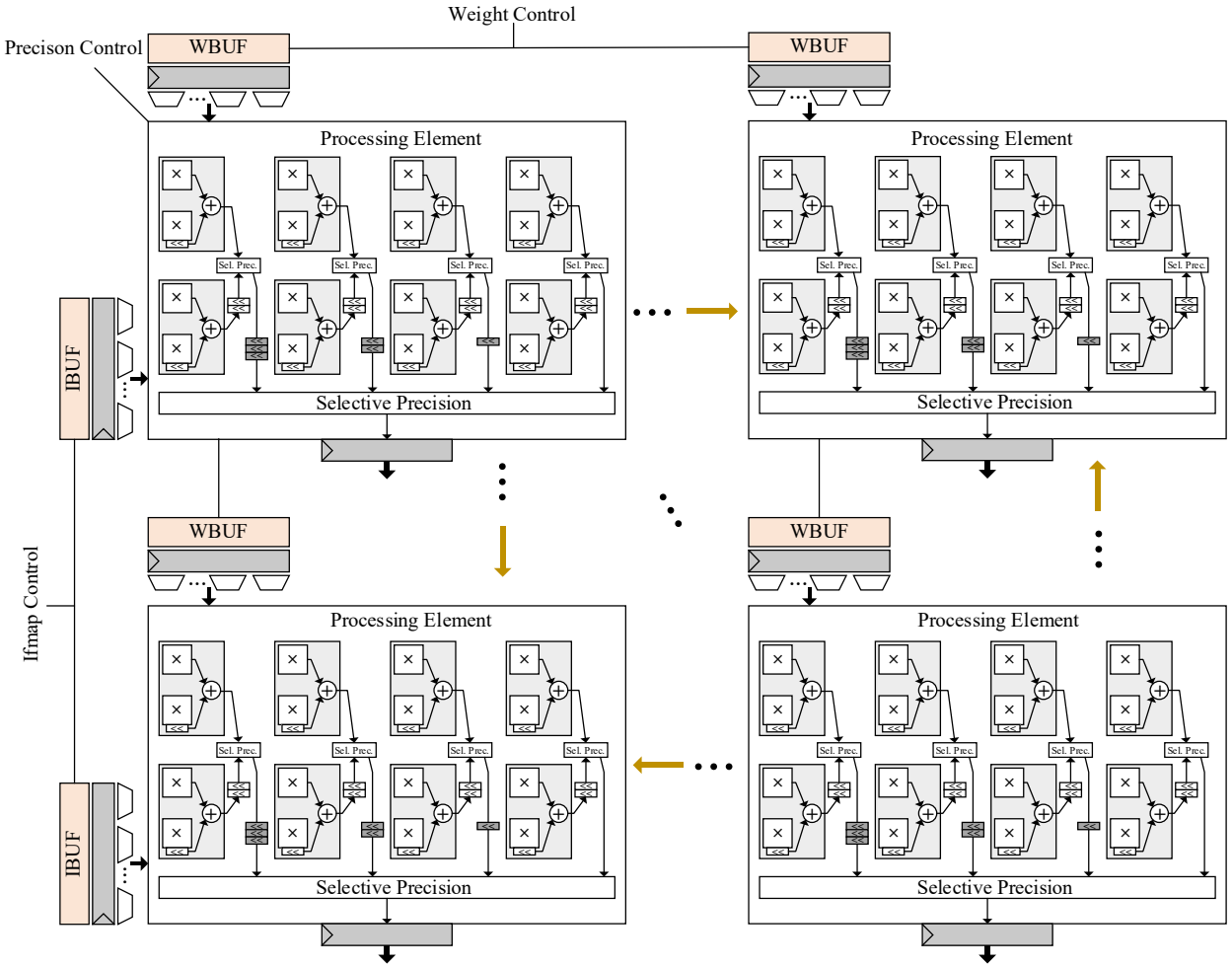
PE structures under different precision modes

Contents

- Introduction and Motivation
- Proposed Approach
- Hardware Architecture
- Evaluation
- Conclusion

Hardware Architecture

➢ Overall Architecture

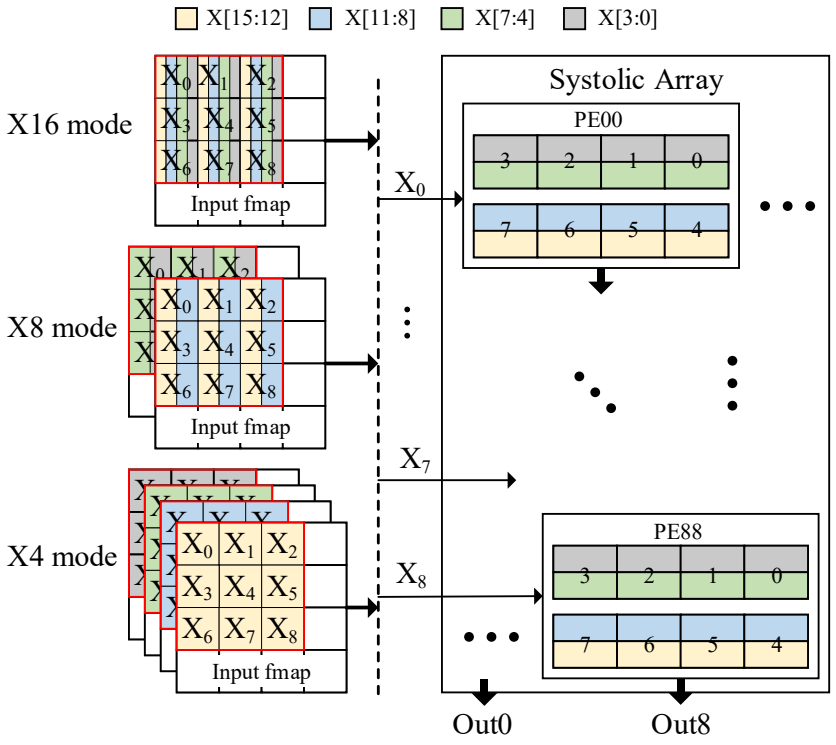


- The overall architecture adopts a two-dimensional homogeneous PE array, where each PE integrates local computation and storage resources.
- IBUFs distribute ifmaps along the rows, while WBUFs supply weights along the columns.
- A global precision control signal is broadcast to all PEs, enabling configurable 4/8/16-bit computation.
- Outputs are collected through column-wise accumulators and bottom output buffers.

Hardware Architecture

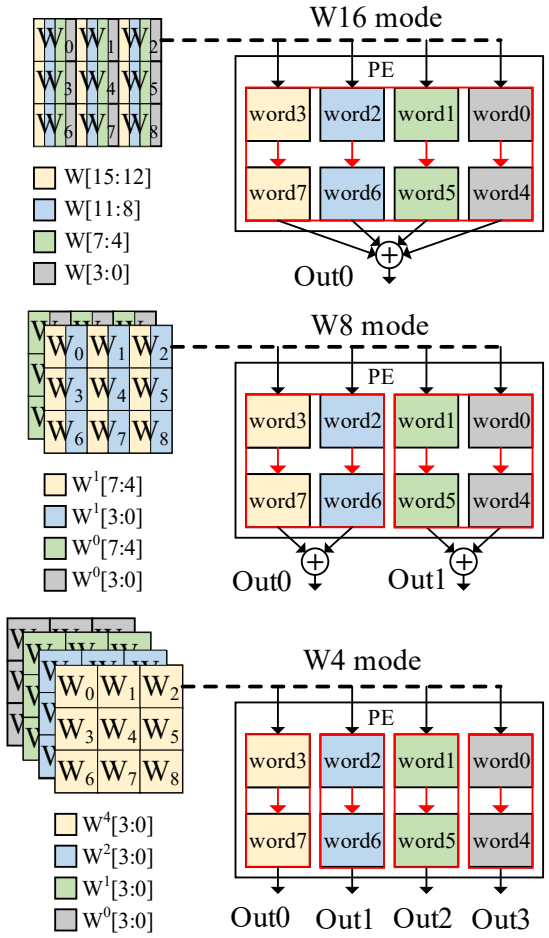
➢ Reconfigurable Mapping

- **X16:** Split each ifmap into four 4-bit units for full accumulation.
- **X8:** Split each ifmap into two 4-bit units, enabling two parallel ifmap groups.
- **X4:** Map each ifmap to one 4-bit unit, enabling up to four parallel ifmap groups.



Reconfigurable mapping of ifmaps under different precision modes.

- **W16:** Split each weight into four 4-bit units aligned with ifmap units.
- **W8:** Split each weight into two 4-bit units, enabling more parallel outputs with low-precision ifmaps.
- **W4:** Map each 4-bit weight to a 4-bit × 4-bit multiplier, supporting up to 8 parallel outputs.



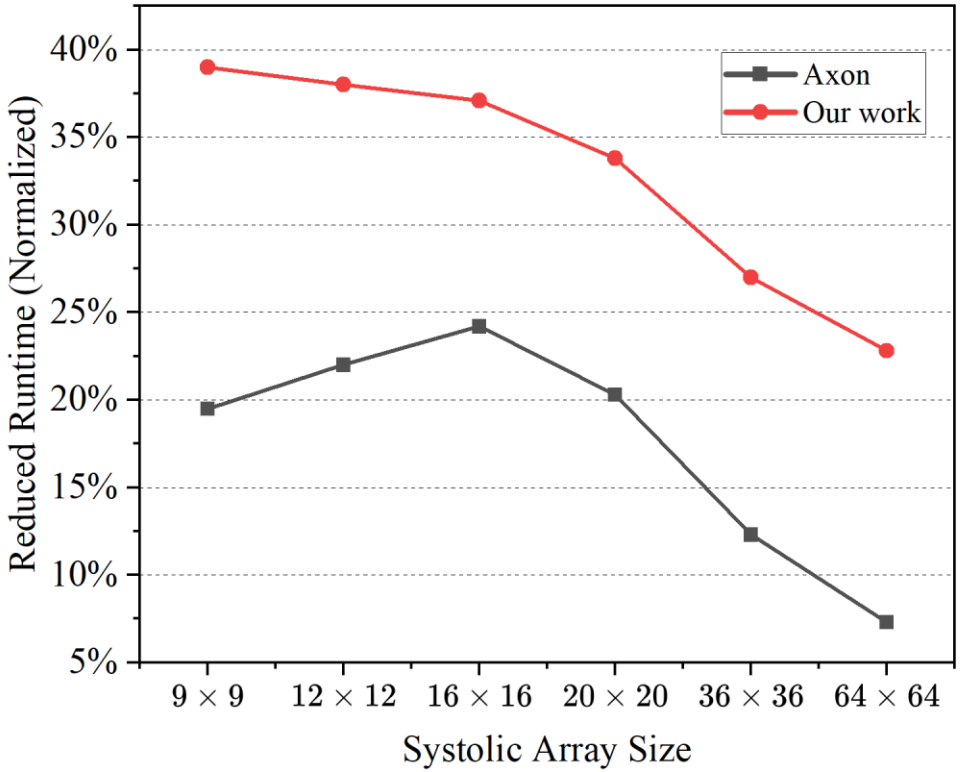
Reconfigurable mapping of weights under different precision modes.

Contents

- Introduction and Motivation
- Proposed Approach
- Hardware Architecture
- Evaluation
- Conclusion

Evaluation

➢ Runtime Analysis of Systolic Array Dataflow



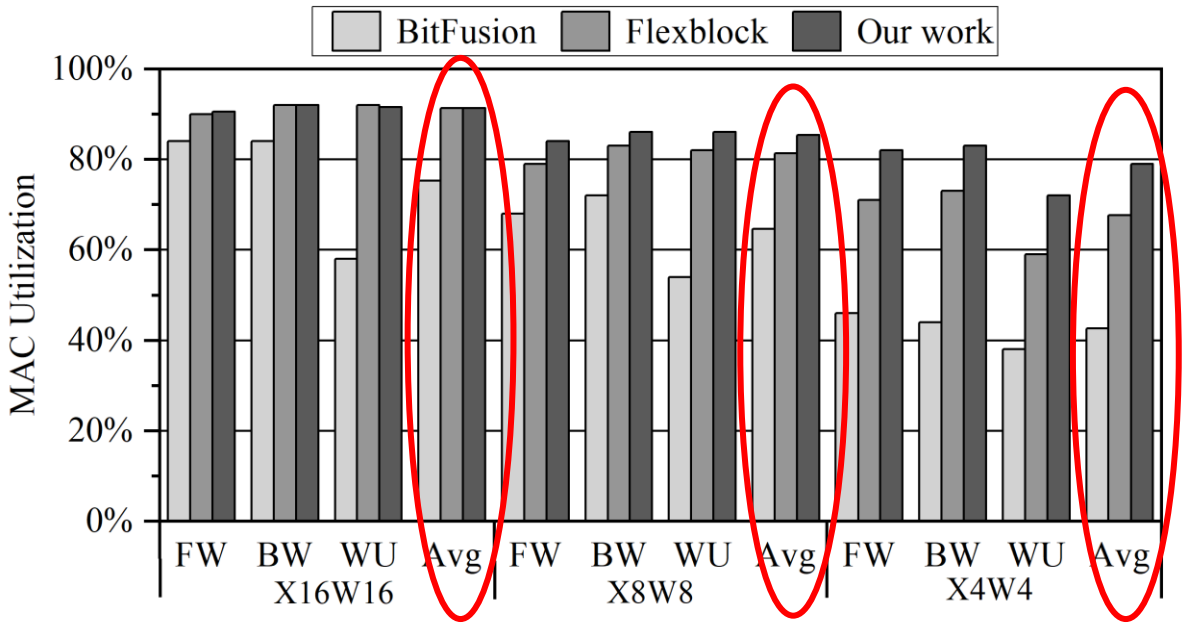
- Matrix multiplication: **49 × 16** and **16 × 49**
- Array size varies from **9 × 9** to **64 × 64**
- Runtime is normalized to the conventional systolic array baseline

The proposed ring-based diagonal dataflow achieves larger runtime reduction, improving runtime by **22.8%–39.0%**, compared with **7.3%–24.2%** for AXON.

Runtime improvements over conventional systolic array and Axon.

Evaluation

➢ MAC utilization for PE



Comparison of the MAC utilization during the training of MobileNetV1

- Benchmark: **MobileNetV1**
- Evaluation phases: **FW / BW / WU**
- Precision modes: **16-bit, 8-bit, and 4-bit**
- Compared with **BitFusion** and **FlexBlock**

Results:

- **16-bit:** +16% over BitFusion, comparable to FlexBlock
- **8-bit:** +21% over BitFusion, +4% over FlexBlock
- **4-bit:** +36% over BitFusion, +11% over FlexBlock

The proposed word-parallel PE maintains high MAC utilization, especially under low-precision and low-parallelism scenarios.

Evaluation

➢ Architecture Comparison

TABLE I
ARCHITECTURAL COMPARISONS BETWEEN BITFUSION-BASED CORES, FLEXBLOCK CORE, BITSYS CORES, AND PROPOSED ARCHITECTURE IN TERMS OF AREA, POWER CONSUMPTION, AND ENERGY EFFICIENCY.

Hardware Architecture	BitFusion Cores [11]	FlexBlock Cores [15]	BitSys Cores [14]	Our work
Technology	28nm	28nm	28nm	28nm
Supported Precision	4/8/16	4/8/16	4/8/16	4/8/16
Array Size	16×16 (for FIX16)	16×16 (for FIX16)	16×16 (for FIX16)	16×16 (for FIX16)
Area [μm^2]	332733	350449	342142	371864
Power Consumption [mW]	67.21 / 66.35 / 65.63	97.12 / 96.45 / 95.82	59.23 / 58.05 / 57.89	83.42 / 81.21 / 80.37
Clock Frequency	333MHz	333MHz	333MHz	333MHz
Throughput [GFLOPS]	176.26 / 110.63 / 41.25	548.75 / 209.38 / 45.22	151.34 / 78.25 / 35.21	**954.35** / **420.32** / **46.45**
Efficiency [GFLOPS/W]	2622.53 / 1667.37 / 628.52	5650.23 / 2170.87 / 471.93	2555.12 / 1347.98 / 608.22	**11332.41** / **5157.72** / 577.95

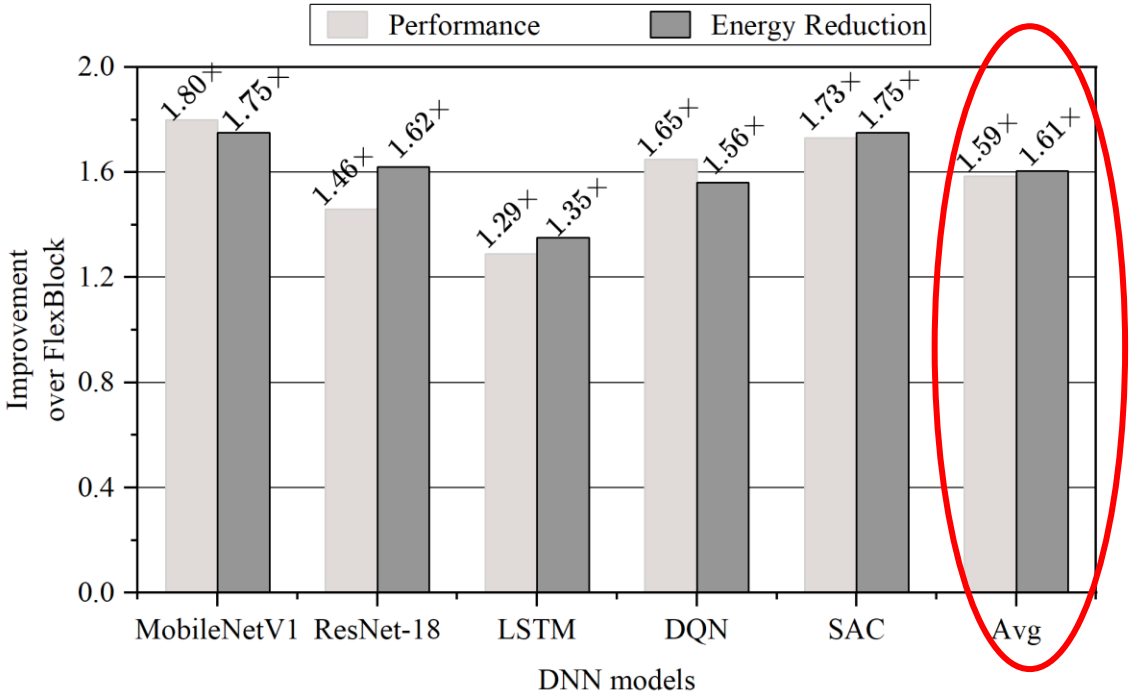
With modest area overhead and competitive power, our design achieves the highest 4-/8-bit throughput and energy efficiency.

Evaluation

➢ Performance and Energy Evaluation on DNN workloads

TABLE II
EVALUATED DNN BENCHMARKS.

DNN model	Type	Domain	Precision
MobileNetV1	CNN	Image Classification	4/8/16
ResNet-18	CNN	Image Classification	4/8/16
LSTM	RNN	Language Modeling	4/8/16
DQN	DRL	Control Policy	4/8/16
SAC	DRL	Control Policy	4/8/16



The proposed architecture consistently improves both inference performance and energy efficiency over FlexBlock.

Contents

- Introduction and Motivation
- Proposed Approach
- Hardware Architecture
- Evaluation
- Conclusion

Conclusion

- This work proposes a **word-level multi-precision systolic array** for efficient DNN acceleration.
- A **word-parallel PE** is designed to support dynamic **4/8/16-bit** computation while improving MAC utilization.
- A **ring-based diagonal dataflow** is introduced to reduce operand pre-filling latency through bidirectional propagation.
- Experimental results show up to **36% higher MAC utilization**, **6.31× higher throughput**, and **4.44× higher energy efficiency** under 4-bit precision.
- Across representative DNN workloads, the proposed architecture achieves **1.59× performance improvement** and **1.61× energy reduction**.

Thanks!