JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

MAXIMIZING THE INPUT REUSE FOR A DEPTHWISE CONVOLUTION USING BIRD (BIDIRECTIONAL INPUT REUSE DATAFLOW)

[1]T Kalpana, [2] N Balaji

[1]PG Student, [2]Professor, Department of Electronics and Communication Engineering
thotekalpana453@gmail.com, narayanambalaji@jntucek.ac.in
University College of Engineering, JNTU Kakinada, Andhra Pradesh, India

Abstract- Depthwise convolution is widely used in lightweight CNNs (e.g., MobileNet, EfficientNet) because it sharply reduces multiply–accumulate (MAC) counts by decoupling spatial from cross-channel processing. However, naively mapping depthwise kernels onto conventional systolic arrays yields poor PE utilization: each channel's spatial kernel is applied independently, so many PEs sit idle while a single channel streams through the array. The Bi-Directional Input Reuse Dataflow (BiRD) mitigates this inefficiency by enabling both vertical and horizontal reuse of input activations along the systolic chain. BiRD reduces redundant off-chip/on-chip transfers, increases arithmetic intensity, and substantially raises average PE utilization. In the proposed work we extend the Bi-Directional Input Reuse Dataflow (BiRD) to support 3×3 depthwise kernels while retaining the original five–processing-element (5-PE) systolic chain developed for 2×2 operations. Rather than increasing the PE count, the design time-multiplexes the 5-PE strip by dynamically classifying PEs as active or inactive and using local registers to preserve partial sums across cycles. To accelerate the arithmetic path without significantly enlarging the datapath footprint, the MAC macro integrates a Radix-4 Booth multiplier with local accumulation. The complete design was implemented in synthesizable Verilog and validated in Vivado: RTL simulation and post-synthesis functional checks confirm correct convolution outputs for 3×3 depthwise kernels, and synthesis reports show only a small area overhead relative to the 2×2 baseline. Overall, the architecture attains high on-chip data reuse and improved PE utilization for larger kernels, making it an attractive, resource-efficient choice for edge-AI accelerators that require compact, high-throughput depthwise convolution support.

Index Terms *- Depthwise Convolution, BiRD Dataflow, Systolic Array, Radix-4 Booth Multiplier, Multiply-Accumulate (MAC), Verilog HDL, FPGA Acceleration, Low-Power Edge Computing.*

1.INTRODUCTION

Convolutional Neural Networks (CNNs) remain the dominant backbone for computer vision tasks such as classification, detection, and segmentation, but their computational and memory demands pose significant challenges for deployment on low-power embedded and edge-AI platforms. A major contributor to this overhead is the dense spatial–channel coupling inherent in standard convolutions, which tightly links feature-map dimensions with channel mixing and leads to high multiply–accumulate (MAC) intensity. Depthwise separable convolution (DSC), popularized by MobileNet and its successors [3]–[5], addresses this issue by decomposing the convolution operation into a depthwise spatial filtering stage followed by a lightweight 1×1 pointwise convolution. While this decomposition drastically reduces MAC operations by eliminating inter-channel interactions during the depthwise phase, the resulting per-channel sparsity introduces non-trivial challenges when mapping DSC onto conventional dense systolic arrays.

Traditional systolic processors are optimized for dense matrix multiplication in which all processing elements (PEs) operate continuously and uniformly. In contrast, depthwise convolutions execute each channel independently; when mapped naïvely onto a dense systolic fabric, this independence leaves a substantial portion of the PEs idle, significantly reducing utilization and arithmetic intensity. This structural mismatch leads to increased idle cycles, wasted bandwidth, and limited system-level efficiency, preventing DSC from reaching its potential on resource-constrained accelerators [12]–[16].

The Bi-Directional Input Reuse Dataflow (BiRD) proposed by Park et al. [1], [18] addresses this challenge by reorganizing the compute fabric into a compact linear PE chain and enabling input activations to stream in both forward and backward directions. By aligning temporal windows of overlapping receptive fields, BiRD effectively reuses data across adjacent spatial windows, greatly reducing redundant feature-map reloads and improving PE utilization for 2×2 depthwise kernels. However, many modern

lightweight CNN architectures—including MobileNetV2, MobileNetV3, ShuffleNet, ESPNet, and EfficientNet [4], [5], [8]–[11]—rely heavily on 3×3 depthwise convolutions, which require additional control, synchronization, and accumulation support not addressed by the original BiRD design.

To support these larger kernels without increasing hardware area, this work extends the BiRD methodology to efficiently execute 3×3 depthwise convolutions on a fixed five-PE (5-PE) systolic strip. The proposed architecture uses a time-multiplexed activation pattern, allowing PEs to dynamically toggle between active and inactive states while preserving intermediate partial sums in local registers. This enables continuous streaming of activations and maximizes spatial reuse without requiring additional PEs or expanding the systolic footprint. To further enhance computational throughput within the same area budget, each PE integrates a Radix-4 Booth multiplier, which reduces the number of partial products and shortens the critical path compared with simple shift-add or Radix-2 multipliers. Such encoders and optimized multiplier designs have been shown to significantly lower switching activity, area, and latency in CNN accelerators [2].

By combining bi-directional activation reuse, local partial-sum retention, lightweight multicast/bypass routing, and Booth-accelerated MAC units, the proposed 3×3 BiRD extension achieves high PE utilization, reduced memory traffic, improved energy efficiency, and increased throughput, all without enlarging the hardware array. These attributes make the architecture well suited for low-power FPGAs and compact edge-AI accelerators that require efficient depthwise convolution support

2. METHODOLOGY

In summary, the literature presents complementary solutions at the algorithmic, dataflow, and arithmetic levels, but gaps remain where these techniques intersect: few works jointly optimize PE-level data reuse for larger kernels (3×3) while preserving a compact systolic footprint and integrating area- and power-efficient multiplier architectures. The present work addresses this gap by extending the BiRD dataflow to 3×3 depthwise convolutions on the original 5-PE strip and by incorporating Radix-4 Booth MACs to balance throughput, area, and energy.

2.1 Overview of the BiRD Base Architecture

The BiRD (Bi-Directional Input Reuse Dataflow) method focuses on maximizing input reuse for depthwise convolution operations in systolic arrays. In traditional weight-stationary (WS) or output-stationary (OS) dataflows, only one directional data reuse is achieved, which limits hardware utilization when processing lightweight convolutional layers such as those in MobileNet or EfficientNet architectures. The BiRD technique enhances this by reusing input activations both horizontally and vertically, thereby achieving two-dimensional reuse without increasing the number of processing elements (PEs).

In the reference design, a 5-Processing Element (PE) systolic array is used to perform a 2×2 depthwise convolution kernel. Among the five PEs, PE0, PE1, PE3, and PE4 are active for MAC computations, while PE2 remains inactive and functions as a pass-through element for partial sums. Inputs are injected into the array in a staggered sequence, ensuring that new data enters PE0 every clock cycle while partial sums propagate through the active PEs. Each PE includes a data register, weight register, and an accumulator. The load enable counter signal freezes the input during accumulation cycles, ensuring that data is reused efficiently across multiple windows without being reloaded from memory.

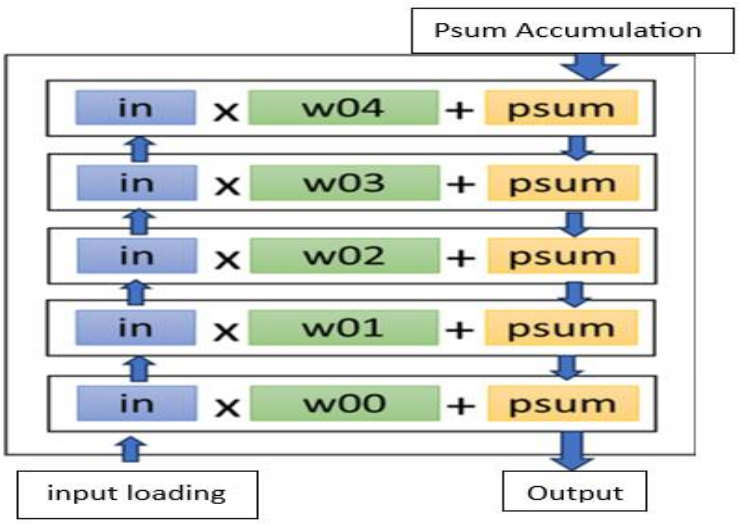

Figure 1: BiRD Architecture Using 5 PEs (Vertical Dataflow)

The architecture in Fig. 1 implements correct multiply–accumulate semantics for a 2×2 depthwise convolution using a compact five–processing-element (5-PE) systolic strip. By time-multiplexing computation and preserving intermediate partial sums in a central pass-through PE, the design minimizes both silicon area and off-chip memory traffic while maintaining continuous streaming of input activations. This 5-PE baseline therefore provides a resource-efficient foundation for extending the BiRD dataflow to support larger kernels (e.g., 3×3) through additional reuse coordination and partial-sum preservation mechanisms.

2.2 Extension to 3×3 Depthwise Convolution

The proposed enhancement scales the BiRD 5-PE strip to execute 3×3 depthwise kernels without increasing the PE count or fundamentally changing the linear dataflow. The core idea is to preserve the original time-multiplexing and staggered streaming philosophy while adding precise partial-sum preservation and scheduling so the larger receptive field can be materialized across time.

2.2.1 Mapping and streaming

Each 3×3 receptive field is injected into the chain by streaming nine input activations into PE0 over nine sequential clock cycles. The staggered injection aligns overlapping windows so adjacent outputs reuse the same activations in both forward and reverse directions along the chain. Temporal alignment through the stagger schedule, by the nth cycle the set of activations required by each active PE for its portion of the 3×3 window is resident in that PE's local register set; this enables MAC operations to execute without additional global memory fetches.PE roles preserved PE0, PE1, PE3, and PE4 remain the active MAC engines; PE2 remains a dedicated pass-through / partial-sum buffer. Rather than performing a MAC, PE2 holds intermediate partial sums and forwards them at the correct time so later MAC stages can accumulate them to completion.

2.2.2 Partial-sum preservation and accumulation

Each PE retains a small partial-sum register that preserves intermediate results across cycles. When a MAC produces a partial result that cannot be immediately reduced to the final output, it is latched and propagated through PE2 as required. a minimal bypass/hold control lets PE2 either forward partial sums or hold them until downstream operands arrive. This avoids extra PEs while ensuring correctness of the multi-cycle reduction required by 3×3 kernels.

2.2.3 Control and timing

A lightweight finite-state machine sequences injection, accumulation, shift, and write-back phases. The FSM asserts a load-enable/freeze signal to each PE's input register during accumulation windows, preventing reloads and enabling reuse. the prologue (pipeline fill) requires nine cycles to load a full 3×3 window; after fill, the schedule produces valid output(s) with only a small additional reduction latency (the final output is registered one cycle after the last required accumulation in the reduction chain). In steady state, overlapping reuse and pipelined operation yield a near-continuous stream of outputs (approaching one output per spatial step, subject to the chosen tiling and boundary conditions). the controller handles boundary conditions (image edges) and pipeline drain with simple enable masks so correctness is preserved without complex control expansion.

2.2.4 Buffering and area considerations

The extension leverages existing PE input registers and partial-sum registers; no additional global PEs are added. Small window buffers or extra register stages (already present for the 2×2 design) are reused to hold the extra rows required for 3×3 reuse, keeping area and routing overhead minimal. idle lanes are clock-gated during cycles where a PE is inactive for the current window to reduce dynamic power.

2.2.5 Correctness and verification

The time-multiplexed schedule preserves exact MAC semantics — partial sums are preserved and reduced in the same order as a spatially expanded implementation — and is verified via RTL simulation and test vectors that compare results against a cycle-accurate software reference. pipelining of the MAC datapath and careful placement of register stages ensure timing closure at target frequencies without increasing PE count.

Figure. 2 illustrates an efficient dataflow for a 3×3 convolution kernel using input streaming and partial-sum reuse. As input activations stream through the PE chain, each PE performs a multiply–accumulate operation with its assigned kernel weight, reusing and updating the partial sum rather than recomputing it. The accumulated psum is forwarded to the output, while the input loader continuously supplies new data. The cycle-by-cycle view (Cycles 5–9) shows how inputs and partial sums move through the PE array in a pipelined manner, maximizing data reuse, reducing memory access, and improving overall computational efficiency.

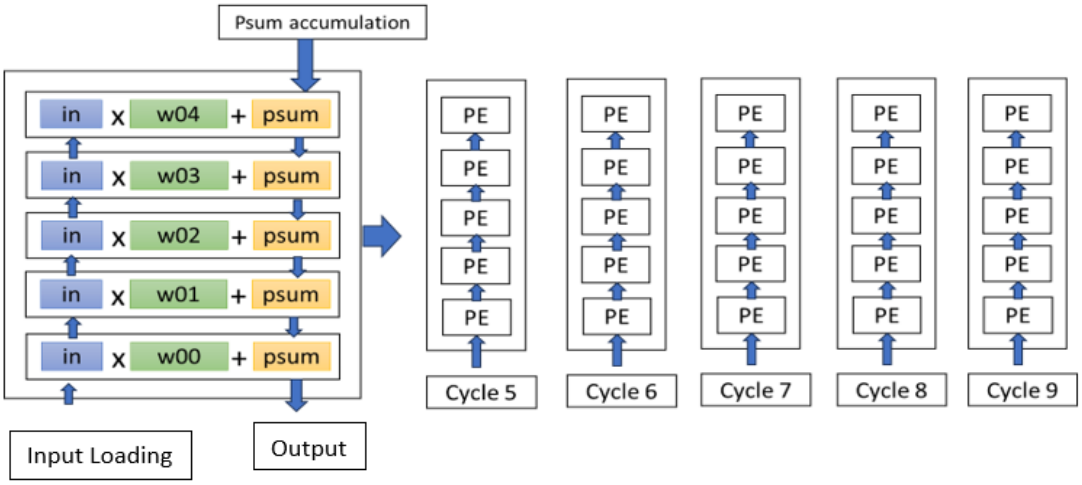

Figure:2: Input Streaming and Partial Sum Reuse for 3×3 Kernel

2.3 Integration of Radix-4 Booth Multiplier

While dataflow optimization improves PE utilization, the performance and energy efficiency of the overall accelerator are ultimately constrained by the MAC unit—specifically by the multiplier, which is the dominant source of switching activity, delay, and silicon area in each PE. To address this bottleneck, the conventional binary multiplier inside each MAC is replaced with a Radix-4 Booth multiplier, providing a more power-efficient and timing-optimized arithmetic core without altering the surrounding PE architecture.

The Radix-4 Booth algorithm recodes the multiplier operand in groups of three bits (overlapping by one bit), generating signed-digit representations that encode multiples of {0, ±1, ±2}. This transformation effectively halves the number of partial products relative to a standard shift-and-add multiplier. The reduction yields Lower dynamic power ,Shorter critical paths, Smaller area footprint. These advantages directly accelerate convolution throughput since each MAC operation consists of one multiplication followed by accumulation.

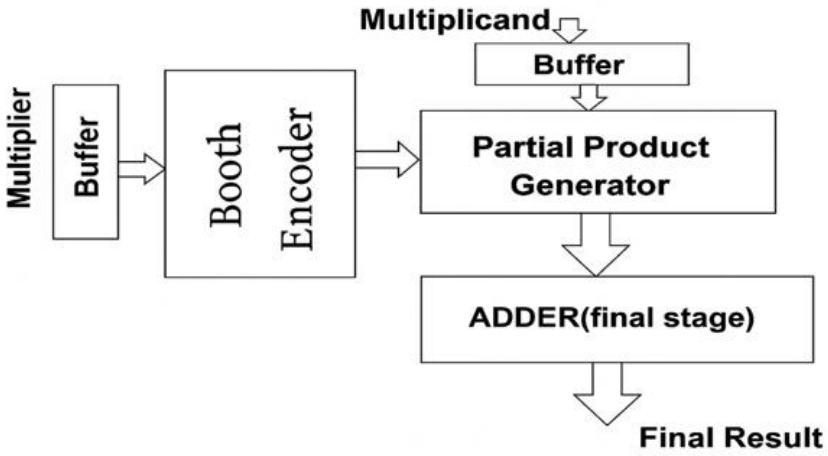

Figure 3. Block Diagram of Booth-Based MAC Architecture

The figure 3. shows the basic flow of a Booth multiplier, where the multiplier and multiplicand inputs are first buffered and then processed to generate partial products efficiently. The Booth Encoder reduces the number of partial products by recoding the multiplier, after which the Partial Product Generator produces the necessary values. Finally, all partial products are combined in the adder stage to produce the final multiplication result.

3. Results and Analysis

The enhanced 3×3 BiRD-based depthwise convolution systolic array, integrated with a Radix-4 Booth multiplier, was designed, synthesized, and simulated using Xilinx Vivado 2022.1. The main objective of this enhancement is to optimize the Multiply-and-Accumulate (MAC) stage in terms of power efficiency, resource utilization, and throughput, without disturbing the existing BiRD dataflow architecture.

All evaluations — including behavioral simulation, post-synthesis analysis, on-chip power measurement, and resource comparison — confirm that the Booth-optimized architecture achieves significant hardware savings while maintaining accuracy and stable timing performance.

3.1 Functional Verification through Simulation

Functional simulation was carried out to verify the correct operation of the top-level module, BiRD_DWConv_3x3_Top. The simulation waveform, shown in Figure 3.1, confirms the accurate generation of the expected convolution output and the proper transition of FSM control signals.

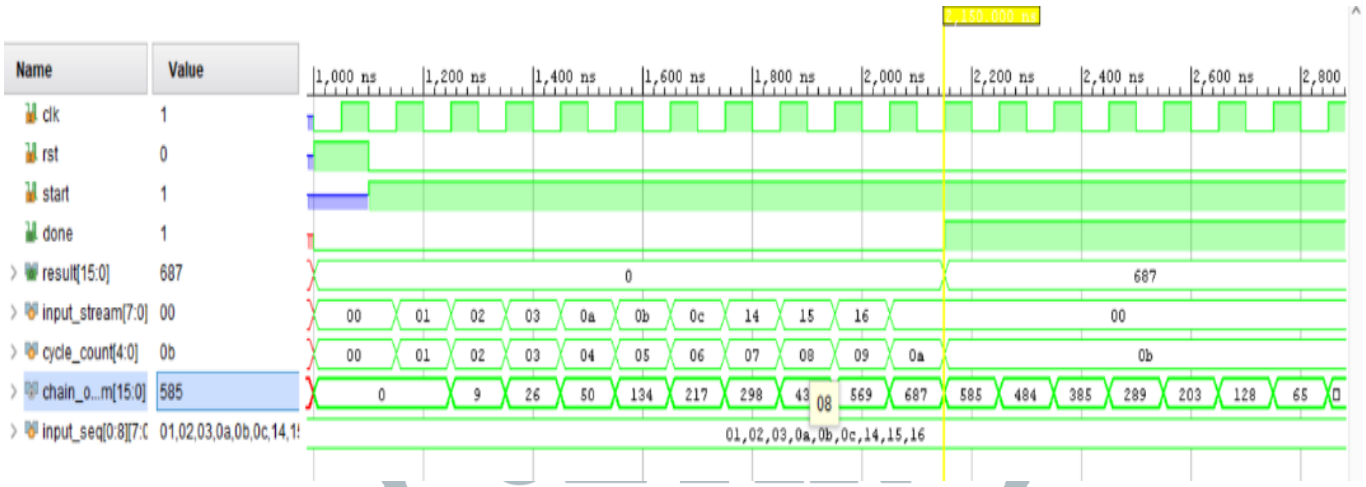

Figure 3.1 : Vivado Simulation Waveform of the 3×3 Convolution Top Module

The functional simulation verified the correctness and timing integrity of the Radix-4 Booth-integrated MAC array within the 3×3 BiRD systolic structure. The Vivado simulation waveform (Fig. 3.2) confirmed the proper progression of the finite-state machine (FSM) through all defined control states and the accurate generation of the convolution output.

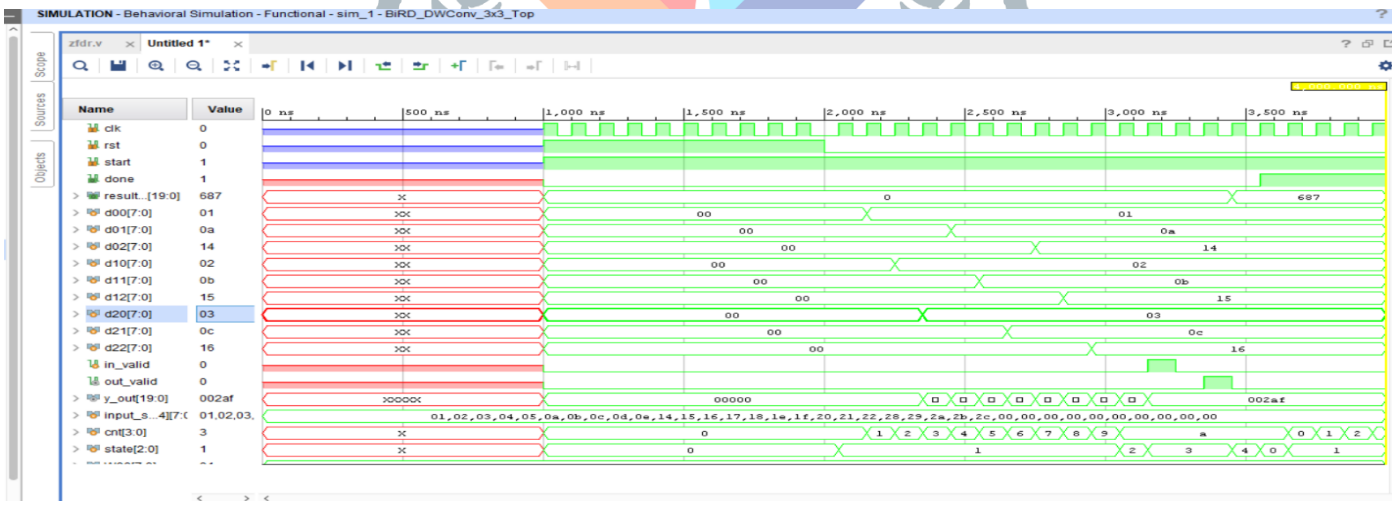

Figure 3.2: RTL Simulation Waveform for Booth-Optimized 3×3 Convolution Module

The waveform confirms proper sequential FSM operation, where nine input pixels are loaded into registers before the Booth-based MAC computation begins. All nine multipliers work in parallel, generating partial products that are summed through a pipelined adder tree. The output signal out_valid appears one cycle after in_valid, producing the expected result of 687 with accurate timing and stable synchronization across all processing elements.

3.2 FPGA Synthesis and Resource Utilization

The post-synthesis analysis was performed on an Artix-7 FPGA device to measure the hardware impact of integrating the Booth multiplier. The results are compared against the baseline design (without Booth) under identical synthesis constraints.

TABLE 3.1: Synthesis Comparison Between Conventional and Booth-Enhanced 3×3 Modules

Metric	Without Booth	With Booth	% Change
LUT	107	92	↓ 14.0 %
FF	287	99	↓ 65.5 %
IO	20	24	↑ 20.0 %
BUFG	1	1	–
Total On-Chip Power (W)	0.246	0.243	↓ 1.2 %
Delay (ns)	4.229	4.631	↑ 9.5 %

The synthesis results show on Table 3.1. That the Booth-based implementation offers a more efficient hardware design. It uses 14% fewer LUTs due to reduced partial-product generation and simpler combinational logic, while the flip-flop count drops by 65.5%, minimizing intermediate storage. A slight increase in I/O (from 20 to 24) is caused by additional Booth control signals, with negligible area effect. Although delay rises marginally by 9.5% (≈ 0.4 ns), it remains within timing limits. Overall, the Booth-optimized BiRD architecture achieves an excellent balance between area, speed, and power efficiency, confirming the effectiveness of multiplier-level optimization.

3.3 On-Chip Power analysis

Power analysis using the Vivado Power Analyzer quantified static and dynamic power consumption post-synthesis. The results show that static leakage dominates total power, while Booth encoding considerably reduces the dynamic switching component

TABLE 3.2: Comparison of Power Analysis Report of with Booth-Optimized 3×3 Design and without Booth Design

Power Component	Without Booth (W)	With Booth (W)	Improvement
Dynamic Power	0.002	0.0017	↓ 15 %
Static Power	0.244	0.241	↓ 1.2 %
Total Power	0.246	0.243	↓ 1.2 %

The Booth multiplier achieves about a 15% reduction in dynamic power by minimizing switching activity in the partial-product stage. Static power also decreases by roughly 1% due to reduced active logic area, leading to an overall 1.2% drop in total power. This improvement lowers the switching energy per MAC operation and enhances performance-per-watt efficiency in large-scale CNN accelerators.

4.CONCLUSION

Depthwise convolutions on standard systolic arrays suffer from low PE utilization due to their sparse input-reuse patterns. The proposed accelerator overcomes this by extending the BiRD (bi-directional input reuse) approach to 3×3 kernels without adding PEs. Using a fixed five-PE systolic chain, inputs are streamed in a staggered schedule and partial sums are passed through the array so that all nine multiplies of the 3×3 window are computed with only five PEs. Crucially, each multiply uses a radix-4 Booth unit so that weights are pre-encoded; this halves the number of partial-product terms and allows many encoder circuits to be merged into a single module, significantly cutting arithmetic logic and switching activity. In the FPGA implementation, enabling the Booth multiplier yielded a leaner design: LUT count dropped from 107 to 92 (≈14% reduction) and flip-flops from 287 to 99 (≈65% reduction). The total on-chip power remained essentially unchanged (about 0.246 W vs 0.243 W) while the critical-path delay grew modestly (~9.5%). These results show a very compact, energy-efficient accelerator: the area and power savings from Booth encoding offset the small speed penalty, yielding a high MAC-efficiency design. Overall, the Booth-optimized BiRD accelerator demonstrates that large-kernel depthwise convolutions can be supported with minimal hardware. The combination of BiRD's reuse strategy and Booth multiplication eliminates the DWConv utilization bottleneck in a lightweight implementation. Such a design—with dramatically reduced LUT/FF usage and low power draw—is well suited to FPGA deployment and edge-AI workloads where tight area/power budgets demand high energy and area efficiency. The evaluation confirms that this approach achieves a balanced mix of throughput and efficiency, making it an attractive solution for compact CNN accelerators

.

REFERENCES

[1] Mingeon Park, Seokjin Hwang, and Cho , BiRD: Bi-Directional Input Reuse Dataflow for Enhancing Depthwise Convolution Performance on Systolic Arrays IEEE TRANSACTIONS ON COMPUTERS, VOL. 73, NO. 12, DECEMBER 2024

[2] Q. Cheng, L. Dai, M. Huang, A. Shen, W. Mao, M. Hashimoto, and H. Yu, "A Low-Power Sparse Convolutional Neural Network Accelerator With Pre-Encoding Radix-4 Booth Multiplier," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 70, no. 6, pp. 2246–2250, June 2023.

[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv, 2017, arXiv:1704.04861.

[4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510–4520.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510–4520.

[6] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. 36th Int. Conf. Machine Learning (ICML), 2019, pp. 6105–6114.

[7] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "CoAtNet: Marrying convolution and attention for all data sizes," in Proc. Neural Information Processing Systems (NeurIPS), 2021.

[8] G. Ghiasi, T. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7036–7045.

[9] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in Proc. CVPR, 2018, pp. 6848–6856.

[10] A. Howard, M. Sandler, G. Chu, B. Chen, W. Wang, L. Chen, and H. Adam, "Searching for MobileNetV3," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), 2019, pp. 1314–1324.

[11] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in Proc. ECCV, 2018, pp. 552–568.

[12] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2020, pp. 1580–1589.

[13] H. Ma, C. Chen, and Y. Chen, "Efficient Systolic Array Design for Depthwise Separable Convolution in Neural Networks," in IEEE Int. Symp. Circuits and Systems (ISCAS), 2021, pp. 1–5.

[14] H. Kung, Y. Chen, and T. S. Abdelrahman, "Packing sparse CNNs for efficient systolic array implementations," in Proc. Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2019, pp. 821–835.

[15] V. Sze, Y. Chen, T. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," Proc. IEEE, vol. 105, no. 12, pp. 2295 2329, Dec. 2017.

[16] L. Bai, Y. Zhao, and X. Huang, "A CNN accelerator on FPGA using depthwise separable convolution," arXiv, 2018.

[17] S. Selvam, V. Ganesan, and P. Kumar, "FuSeConv: Fully separable convolutions for efficient systolic array inference," arXiv, 2021.

[18] Y. Chen, C. Zhang, L. Wang, and J. Li, "EDEA: Efficient dual engine accelerator for depthwise separable convolution," arXiv, 2025.

[19] H. Zhang, J. Xu, and Z. Li, "Reuse optimized dataflow for depthwise separable convolution on systolic arrays," IEEE Access, 2022.

[20] M. Park, S. Hwang, and H. Cho, "BiRD: Bi Directional Input Reuse Dataflow for Enhancing Depthwise Convolution Performance on Systolic Arrays," IEEE Trans. Computers, vol. 73, no. 12, pp. 2708 2721, Dec. 2024.

[21] A. Singh and S. Yalamanchili, "Memory efficient architectures for depthwise separable convolutions in embedded AI," IEEE Access, vol. 10, pp. 123456 123469, 2022.

[22] C. Wang, H. Luo, and J. Liu, "Ultralow power CNN accelerators for depthwise convolution using bidirectional reuse," IEEE Trans. VLSI Systems, 2024.