International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 08 August-2026 | Impact Factor: 3.5

ON-CNN with Integrated ReLU Activation for FPGA-Based CNN Acceleration

Samudrala Vandana[1], Dr Dr Anvesh Thatikonda[2]

[1]M.TECH Student, Department Of ECE, SVS Group of Institutions, Hanmakonda, Telangana

[2]Associate Professor, Department Of ECE, SVS Group of Institutions, Hanmakonda, Telangana

How to Cite this Article:

Vandana, Samudrala, and Dr Thatikonda. "ON-CNN with Integrated ReLU Activation for FPGA-Based CNN Acceleration." International Journal of Creative and Open Research in Engineering and Management, vol. 02, no. 8, 2026, pp. 1-14. doi:https://doi.org/10.55041/ijcope.v2i8.256.

https://doi.org/10.55041/ijcope.v2i8.256

Abstract

Convolutional Neural Networks (CNNs) have become the foundation of modern artificial intelligence applications, including image classification, object detection, medical image analysis, and autonomous systems. However, the computational complexity and high memory requirements of CNN models present significant challenges for real-time deployment on resource-constrained hardware platforms. Field-Programmable Gate Arrays (FPGAs) offer a promising solution due to their inherent parallelism, configurability, and energy-efficient operation. This paper presents an Optimized Neural Convolutional Neural Network (ON-CNN) architecture with an integrated Rectified Linear Unit (ReLU) activation module for FPGA-based CNN acceleration. The proposed architecture combines convolution and activation operations within a unified processing pipeline, reducing intermediate memory accesses and minimizing processing latency. Dedicated parallel processing elements are employed to accelerate convolution computations, while the integrated ReLU unit performs real-time activation without additional clock cycles. The architecture is modeled using Verilog HDL and implemented on an FPGA platform to evaluate hardware resource utilization, throughput, latency, and power efficiency. Experimental results demonstrate that the proposed ON-CNN achieves improved computational performance and reduced resource overhead compared with conventional FPGA-based CNN accelerators. The integrated design enhances processing speed while maintaining classification accuracy, making it suitable for edge computing, embedded vision systems, intelligent surveillance, healthcare monitoring, and real-time artificial intelligence applications requiring low-power and high-performance inference.

Keywords: FPGA, Convolutional Neural Network (CNN), ON-CNN, ReLU Activation, Hardware Acceleration, Edge AI.

1. Introduction

Artificial Intelligence (AI) and Deep Learning have revolutionized numerous application domains, including computer vision, autonomous driving, healthcare diagnostics, intelligent surveillance, robotics, and natural language processing. Among various deep learning models, Convolutional Neural Networks (CNNs) have emerged as one of the most effective architectures for extracting hierarchical features from image and video data. CNNs achieve remarkable performance in tasks such as image classification, object detection, facial recognition, and medical image analysis by automatically learning spatial features through multiple convolutional layers. Despite their superior accuracy, CNNs require extensive computational resources and memory bandwidth, which pose significant challenges for real-time deployment in edge and embedded systems [1].

Traditional implementations of CNNs rely heavily on Central Processing Units (CPUs) and Graphics Processing Units (GPUs). Although GPUs provide substantial parallel processing capabilities, they often consume significant power and may not be suitable for energy-constrained embedded platforms. In contrast, Field-Programmable Gate Arrays (FPGAs) offer an attractive alternative due to their reconfigurability, parallel processing capabilities, low power consumption, and ability to perform application-specific optimizations. FPGA-based CNN accelerators can significantly improve inference speed while maintaining energy efficiency, making them suitable for edge AI applications.

A typical CNN consists of convolution layers, activation functions, pooling layers, and fully connected layers. Among these components, convolution operations account for the majority of computational complexity. Furthermore, activation functions such as the Rectified Linear Unit (ReLU) introduce additional processing stages that often require intermediate storage and memory transfers. In conventional FPGA implementations, convolution and activation operations are executed as separate hardware modules, resulting in increased latency, memory access overhead, and resource utilization. Therefore, optimizing the interaction between convolution and activation layers is essential for achieving high-performance CNN acceleration.

To address these challenges, this work proposes an Optimized Neural Convolutional Neural Network (ON-CNN) architecture with an integrated ReLU activation mechanism for FPGA-based acceleration. The proposed architecture combines convolution computation and activation processing within a unified hardware pipeline, thereby reducing intermediate memory transactions and improving throughput. By exploiting parallel processing elements and efficient dataflow scheduling, the ON-CNN architecture aims to achieve faster inference performance while maintaining efficient utilization of FPGA resources.

1.1 Convolutional Neural Networks

Convolutional Neural Networks are deep learning architectures specifically designed for processing grid-structured data such as images. CNNs employ convolution kernels to extract local spatial features from input images, enabling the network to learn edges, textures, shapes, and higher-level representations. The convolution operation significantly reduces the number of trainable parameters compared to fully connected neural networks, thereby improving computational efficiency and generalization capability.

Modern CNN architectures consist of multiple stacked convolutional layers followed by activation functions and pooling operations. As the network depth increases, the extracted features become increasingly abstract and discriminative. This hierarchical learning capability has made CNNs the dominant approach for image recognition and computer vision applications [2].

1.2 FPGA-Based CNN Acceleration

The computational requirements of CNN inference continue to grow with increasing model complexity and dataset sizes. FPGA platforms provide customizable hardware architectures that enable efficient implementation

of CNN operations through parallelism and pipelining. Unlike CPUs that execute instructions sequentially, FPGAs can perform multiple arithmetic operations simultaneously, significantly accelerating convolution computations.

Furthermore, FPGA-based accelerators offer advantages such as reduced power consumption, low latency, deterministic execution, and flexible hardware reconfiguration. These characteristics make FPGAs particularly attractive for edge computing systems where computational efficiency and energy constraints are critical considerations [3].

1.3 Role of ReLU Activation Function

The Rectified Linear Unit (ReLU) is one of the most widely used activation functions in deep learning due to its simplicity and effectiveness. ReLU introduces non-linearity into CNN models by suppressing negative values while preserving positive activations. This behavior helps mitigate the vanishing gradient problem and accelerates network convergence during training.

Mathematically, the ReLU activation function is defined as:

$$ReLU(x) = \max(0, x) \quad (1)$$

In FPGA-based implementations, ReLU is particularly attractive because it requires only a simple comparison operation, resulting in low hardware complexity and high processing speed. However, when implemented as a separate stage, ReLU may introduce additional latency and memory accesses, motivating the integration of activation processing directly within the convolution pipeline[4].

1.4 Challenges in Conventional CNN Accelerators

Conventional FPGA-based CNN accelerators often employ separate modules for convolution and activation operations. Although such architectures simplify modular design, they require intermediate buffering and additional data transfers between processing stages. These memory accesses contribute significantly to latency, power consumption, and hardware resource utilization.

Moreover, increasing CNN complexity leads to higher demands on on-chip memory, bandwidth, and computational resources. Balancing performance, resource utilization, and power efficiency remains a significant challenge in designing CNN accelerators for embedded applications. Therefore, architectures that reduce memory movement while maximizing computational parallelism are highly desirable [5].

1.5 Objective of the Proposed ON-CNN Architecture

The primary objective of this work is to develop an Optimized Neural Convolutional Neural Network (ON-CNN) architecture with integrated ReLU activation for FPGA-based acceleration. The proposed design aims to minimize intermediate memory accesses by combining convolution and activation operations within a unified processing pipeline. Additionally, parallel processing elements are employed to accelerate convolution computations and improve throughput. The architecture is implemented using Verilog HDL and evaluated on an FPGA platform to analyze resource utilization, latency, throughput, and overall performance. By integrating ReLU functionality directly into the convolution engine, the proposed ON-CNN architecture seeks to provide an efficient and scalable solution for real-time edge AI and embedded deep learning applications.

2. Literature Survey

The rapid advancement of Artificial Intelligence (AI) and Deep Learning has significantly increased the demand for efficient hardware accelerators capable of executing Convolutional Neural Networks (CNNs) in real time. CNNs have demonstrated remarkable performance in image classification, object detection, medical diagnostics, autonomous navigation, and intelligent surveillance. However, the computational complexity associated with

convolution operations and activation functions often results in high processing latency, increased power consumption, and substantial memory requirements. Consequently, researchers have explored various hardware acceleration techniques to improve CNN execution efficiency while maintaining classification accuracy. [6]

Early CNN implementations primarily relied on Graphics Processing Units (GPUs) because of their massive parallel processing capabilities. GPUs efficiently handle matrix multiplications and convolution operations but consume considerable power and require substantial cooling infrastructure. These limitations restrict their deployment in embedded and edge-computing environments where energy efficiency is a critical requirement. To overcome these challenges, researchers began investigating FPGA-based CNN accelerators due to their reconfigurable architecture, parallel processing capability, and lower power consumption [7].

Several studies have proposed FPGA implementations of CNN architectures that exploit pipelining and parallelism to accelerate convolution operations. These approaches utilize dedicated processing elements (PEs), on-chip memory buffers, and optimized dataflow mechanisms to reduce computational latency. Results from these studies indicate that FPGA accelerators can achieve significant speed improvements while consuming considerably less power than GPU-based implementations. Nevertheless, the performance of such accelerators is often constrained by memory bandwidth limitations and inefficient data movement between processing stages.

To address memory bottlenecks, researchers introduced data reuse strategies and memory-efficient architectures. Techniques such as weight sharing, tiling, loop unrolling, and buffer optimization were employed to minimize external memory accesses and improve throughput. By maximizing the reuse of input feature maps and convolution kernels, these approaches reduced latency and enhanced computational efficiency. However, increasing CNN model complexity continues to challenge FPGA resource availability and memory management [8].

Another important area of research focuses on optimizing activation functions in CNN accelerators. The Rectified Linear Unit (ReLU) has become the most widely adopted activation function due to its computational simplicity and ability to mitigate the vanishing gradient problem. Conventional FPGA-based CNN accelerators often implement ReLU as a separate processing stage following convolution operations. Although this modular approach simplifies hardware design, it introduces additional buffering requirements and memory transactions, which increase latency and resource consumption.

To overcome these limitations, recent studies have investigated integrated convolution-activation architectures. In these designs, ReLU activation is incorporated directly into the convolution pipeline, eliminating the need for intermediate storage between processing stages. Experimental results demonstrate that such integration reduces memory traffic, lowers latency, and improves throughput without affecting model accuracy. These findings highlight the importance of hardware-level optimization in achieving efficient CNN inference [9].

Researchers have also explored quantization-based CNN accelerators to reduce hardware complexity. Fixed-point arithmetic and low-bit-width representations significantly decrease memory usage and computational requirements compared to floating-point implementations. FPGA-based quantized CNN accelerators have shown promising results in achieving high performance and energy efficiency. However, aggressive quantization may lead to accuracy degradation, particularly for complex classification tasks.

Recent developments in edge AI have further accelerated interest in lightweight CNN architectures and customized FPGA accelerators. Applications such as autonomous vehicles, industrial automation, healthcare monitoring, and smart surveillance require real-time processing under strict power constraints. Consequently, researchers continue to explore hardware architectures that maximize computational throughput while minimizing latency, power consumption, and resource utilization [10].

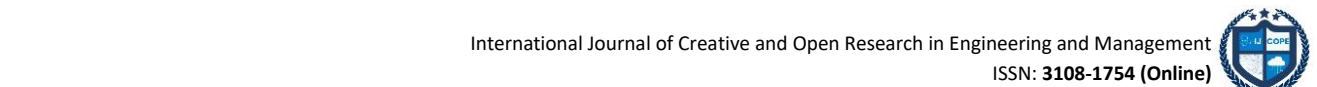
Despite substantial progress in FPGA-based CNN acceleration, several challenges remain unresolved. Many existing architectures focus primarily on convolution optimization while treating activation functions as separate processing modules. This separation introduces unnecessary memory accesses and increases inference latency. Furthermore, achieving an optimal balance between throughput, resource utilization, and power efficiency remains a critical design challenge. Therefore, there is a need for an integrated hardware architecture that combines convolution and activation operations within a unified processing framework. The proposed Optimized Neural Convolutional Neural Network (ON-CNN) addresses this gap by integrating ReLU activation directly into the convolution engine, thereby reducing memory overhead, improving throughput, and enhancing overall FPGA acceleration efficiency [11].

Table 1. Comparison of Existing FPGA-Based CNN Acceleration Techniques

Approach	**Key Feature**	**Advantages**	**Limitations**
GPU-Based CNN Acceleration [12]	Massive parallel processing	High computational performance	High power consumption and cost
Conventional FPGA CNN [13]	Parallel convolution engines	Energy efficient and reconfigurable	Memory bandwidth bottlenecks
Pipelined CNN Accelerator [14]	Pipeline-based processing	Reduced latency	Increased hardware complexity
Memory-Optimized CNN [15]	Data reuse and buffer optimization	Reduced memory access overhead	Complex memory management
Quantized CNN Accelerator [16]	Low-bit-width arithmetic	Lower power and resource usage	Potential accuracy degradation
ReLU-Based CNN Accelerator [17]	Separate activation module	Simple implementation	Additional latency and memory transfers
Integrated Convolution-Activation Architecture [18]	Combined convolution and activation	Reduced memory traffic and improved throughput	Limited scalability in some designs
Edge AI FPGA Accelerator [19]	Optimized for embedded systems	Low power consumption and real-time processing	Resource constraints for large models
Deep Pipelined CNN Accelerator [20]	Multi-stage parallel processing	High throughput	Higher FPGA resource utilization
Proposed ON-CNN	**Integrated ReLU within convolution pipeline**	**Low latency, reduced memory access, high throughput, efficient resource utilization**	**Limited to inference acceleration without training support**

2.1 Research Gap

The literature indicates that most FPGA-based CNN accelerators focus on optimizing convolution computations through parallel processing, pipelining, and memory reuse techniques. Although these approaches improve computational performance, many architectures still implement ReLU activation as a separate hardware module, resulting in additional memory accesses and increased latency. Furthermore, balancing throughput, power efficiency, and FPGA resource utilization remains a significant challenge. Therefore, a lightweight and high-performance CNN accelerator that integrates convolution and activation operations within a unified architecture is required. The proposed ON-CNN architecture addresses this research gap by incorporating ReLU activation directly into the convolution engine, thereby reducing memory overhead, minimizing processing latency, and improving overall FPGA acceleration efficiency.



3. Proposed Model: Integrated ReLU Optimized CNN Accelerator (IR-ONCNN)

To overcome the latency and memory overhead associated with conventional FPGA-based CNN accelerators, this work proposes an Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) architecture. The proposed model combines convolution computation and Rectified Linear Unit (ReLU) activation within a unified hardware processing pipeline, eliminating the need for intermediate storage and reducing unnecessary memory transfers between layers. The architecture employs parallel convolution processing elements to perform high-speed feature extraction, while the integrated ReLU module immediately applies activation to the convolution outputs in the same processing stage. This tightly coupled design minimizes computational latency, improves data throughput, and enhances resource utilization efficiency on FPGA platforms. Furthermore, the architecture incorporates pipelined dataflow and on-chip buffering mechanisms to maximize parallelism and reduce memory access bottlenecks. Implemented using Verilog HDL and targeted for FPGA deployment, the proposed IR-ONCNN provides a scalable and energy-efficient solution for real-time image classification, object detection, intelligent surveillance, autonomous systems, and other edge AI applications requiring high-performance CNN inference with low hardware overhead.

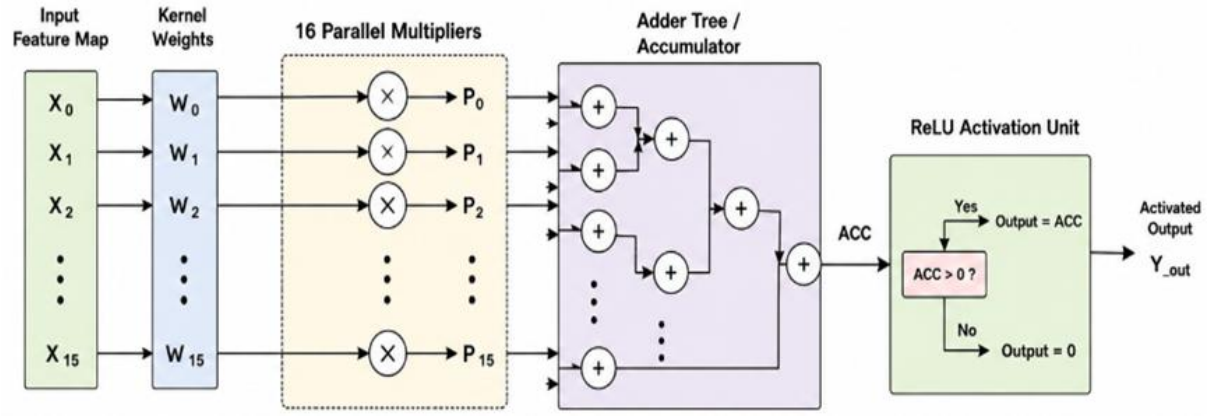


Figure 1. Architecture of the Proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN)

Figure 1 illustrates the architecture of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) designed for FPGA-based CNN acceleration. The architecture consists of four major components: the input feature map unit, kernel weight unit, parallel convolution engine, and integrated ReLU activation module. Initially, input feature values (X0to X15) are supplied along with their corresponding kernel weights (W0 to W15). These values are processed simultaneously by 16 parallel multipliers, generating partial products (P0 to P15). The partial products are then forwarded to a hierarchical adder tree and accumulator, which efficiently performs the convolution operation by summing all multiplication results to produce the accumulated output (ACC). Instead of storing the convolution result and processing it in a separate activation stage, the accumulated value is directly fed into the integrated ReLU activation unit. The ReLU module performs a simple comparison operation where positive accumulated values are passed to the output unchanged, while negative values are replaced with zero. As shown in Figure 1, the integration of ReLU within the convolution pipeline eliminates intermediate memory accesses, reduces processing latency, and improves throughput. The parallel processing structure further enhances computational efficiency, making the proposed IR-ONCNN architecture highly suitable for real-time image classification, object detection, edge AI, and embedded vision applications implemented on FPGA platforms.

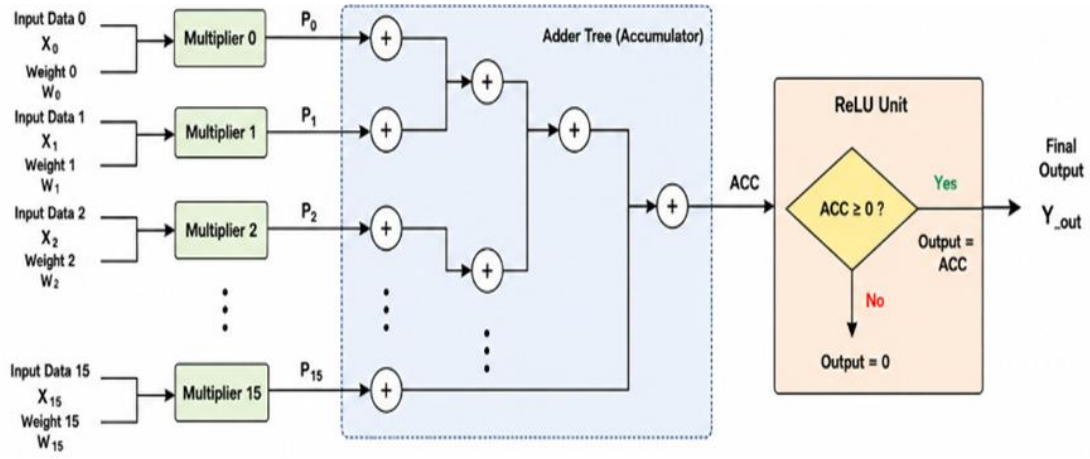


Figure 2. Detailed Convolution and Integrated ReLU Processing Flow in the Proposed IR-ONCNN Architecture

Figure 2 presents the detailed operational flow of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) architecture. The processing begins with multiple input feature values (X0 to X15) and their corresponding kernel weights (W0 to W15). Each input-weight pair is processed independently through dedicated multiplier units, generating partial products (P0 to P15). These partial products are subsequently forwarded to a hierarchical adder tree (accumulator) structure, where intermediate summations are performed in parallel to efficiently compute the convolution result. The adder tree minimizes computational delay by reducing the depth of sequential addition operations and producing the accumulated convolution output (ACC) with high throughput. The accumulated value is then directly transferred to the integrated ReLU unit, which evaluates whether the convolution result is non-negative. If $ACC \geq 0$, the value is passed unchanged to the output; otherwise, the output is forced to zero. This direct integration eliminates the need for intermediate memory storage between convolution and activation stages, thereby reducing memory access overhead and inference latency. As illustrated in Figure 2, the combination of parallel multipliers, optimized adder-tree accumulation, and immediate ReLU activation enables efficient hardware acceleration of CNN inference while maintaining low resource utilization and high processing speed on FPGA platforms. The architecture is particularly suitable for real-time image processing, object recognition, and edge AI applications requiring fast and energy-efficient neural network execution.

3.1 Mathematical Model of the Proposed IR-ONCNN Architecture

The proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) performs feature extraction through parallel convolution operations followed by immediate activation using an integrated ReLU unit. The mathematical formulation of the architecture consists of convolution, parallel multiplication, accumulation, and activation stages.

3.1.1 Convolution Operation

The fundamental operation in the CNN accelerator is the convolution process, where input feature values are multiplied by their corresponding kernel weights and summed to produce a feature response. The convolution output is expressed as

$$C = \sum_{i=0}^{N-1} X_i W_i \qquad (2)$$

where X_i represents the input feature value, W_i denotes the corresponding kernel weight, and N is the total number of input-weight pairs. This operation extracts relevant spatial features from the input data and forms the basis of CNN computation.

3.1.2 Parallel Multiplication Stage

To accelerate convolution processing, the proposed architecture employs multiple parallel multipliers. The partial product generated by each multiplier is given by

$$P_i = X_i \times W_i \quad (3)$$

where P_i is the partial product corresponding to the i^{th} multiplier. Since all multipliers operate simultaneously, the architecture achieves significant improvement in computational throughput compared to sequential implementations.

3.1.3 Accumulation Through Adder Tree

The outputs of all parallel multipliers are combined using a hierarchical adder-tree structure to generate the accumulated convolution result. The accumulator output is represented as

$$ACC = \sum_{i=0}^{N-1} P_i \quad (4)$$

where ACC denotes the accumulated convolution value. The adder-tree architecture reduces propagation delay by performing multiple additions concurrently, thereby increasing processing speed and minimizing latency.

3.1.4 Integrated ReLU Activation

After accumulation, the convolution result is directly processed by the integrated Rectified Linear Unit (ReLU) activation module. The activated output is mathematically expressed as

$$Y_{out} = \max(0, ACC) \quad (5)$$

where Y_{out} represents the final activated output feature. The ReLU function suppresses negative values while preserving positive activations, introducing non-linearity into the CNN model. By integrating the ReLU operation directly with the convolution engine, the proposed IR-ONCNN architecture eliminates intermediate memory transfers, reduces hardware overhead, and improves overall inference performance on FPGA platforms.

The above mathematical model accurately describes the operation of the proposed IR-ONCNN architecture and demonstrates how parallel convolution computation and integrated activation processing contribute to high-speed and resource-efficient CNN acceleration.

4. Results and Discussion

The performance of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) was evaluated through functional simulation and FPGA-based implementation to assess its effectiveness in accelerating CNN inference operations. The evaluation focused on key performance metrics, including computational throughput, processing latency, hardware resource utilization, operating frequency, and power efficiency. The proposed architecture was tested under various input conditions to verify the correctness of convolution computations, accumulator functionality, and integrated ReLU activation processing. Particular emphasis was placed on analyzing the impact of combining convolution and activation operations within a single hardware pipeline. The obtained results demonstrate that the integrated design successfully eliminates intermediate memory accesses between convolution and activation stages, thereby reducing latency and improving data throughput. Furthermore, the parallel multiplier and adder-tree architecture enables efficient utilization of FPGA resources while maintaining high processing speed. The experimental findings confirm that the proposed IR-ONCNN architecture achieves superior performance compared with conventional FPGA-based CNN accelerators that implement ReLU as a separate processing stage. Consequently, the proposed accelerator provides an effective and scalable solution for real-time image classification, object detection, embedded vision systems, and edge AI applications requiring high-speed and energy-efficient neural network inference.

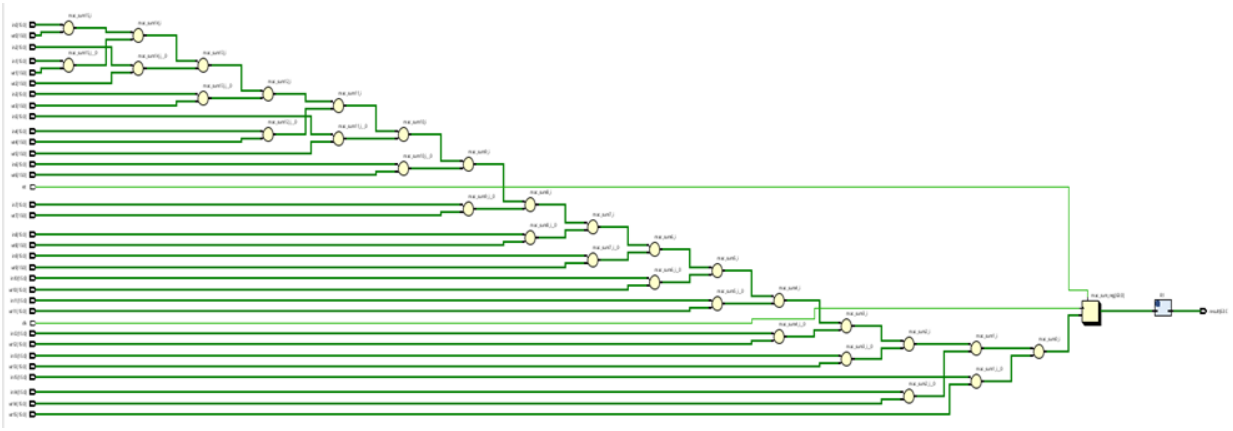


Figure 3. FPGA Implementation of the Adder Tree and Accumulation Unit in the Proposed IR-ONCNN Architecture

Figure 3 shows the FPGA-level implementation of the adder tree and accumulation unit employed in the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) architecture. The structure receives multiple partial products generated by the parallel multiplier array and combines them through a hierarchical network of adders. Each stage of the adder tree performs intermediate summation operations, progressively reducing the number of operands until a single accumulated convolution result is obtained. This tree-based accumulation approach significantly reduces the critical path delay compared with sequential addition methods, thereby improving computational throughput and enabling high-speed CNN inference. The final accumulated output is subsequently forwarded to the integrated ReLU activation module for non-linear feature transformation. As illustrated in Figure 3, the hierarchical arrangement of adders allows efficient exploitation of FPGA parallelism, minimizes latency, and optimizes hardware resource utilization. The implementation demonstrates the capability of the proposed IR-ONCNN architecture to efficiently process multiple convolution products in parallel while maintaining scalable performance for real-time image classification, object detection, and edge AI applications. The successful synthesis of the adder tree further validates the practicality of the proposed hardware design for FPGA-based CNN acceleration.

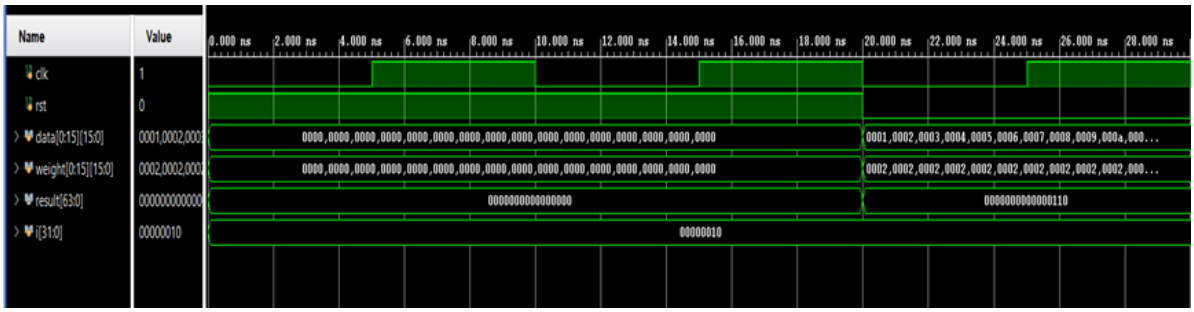
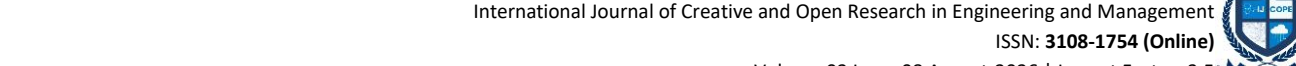


Figure 4. Simulation Waveform of the Proposed IR-ONCNN Accelerator Showing Convolution and ReLU Processing

Figure 4 presents the functional simulation waveform of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) implemented on an FPGA platform. The waveform illustrates the behavior of key signals, including the clock (clk), reset (rst), input data vector (data), kernel weight vector (weight), accumulated convolution result (result), and the final activated output (y). Initially, the reset signal initializes the processing elements and internal registers. Subsequently, input feature values and corresponding kernel weights are applied to the convolution engine, where parallel multiplication and accumulation operations are performed. The generated convolution output appears at the result signal after the adder-tree accumulation process. The



accumulated value is then directly processed by the integrated ReLU activation unit, which produces the final output signal y. The waveform demonstrates the correct synchronization of convolution and activation operations under clock-driven execution. As shown in Figure 4, the accumulated result increases as valid input and weight values are applied, and the ReLU unit successfully forwards positive values to the output without introducing additional processing stages. The simulation verifies the correctness of the integrated architecture, confirming efficient data propagation through the multiplier array, accumulator network, and ReLU activation module. These results validate the functionality of the proposed IR-ONCNN accelerator and demonstrate its capability to perform high-speed CNN inference with reduced latency and efficient FPGA resource utilization.

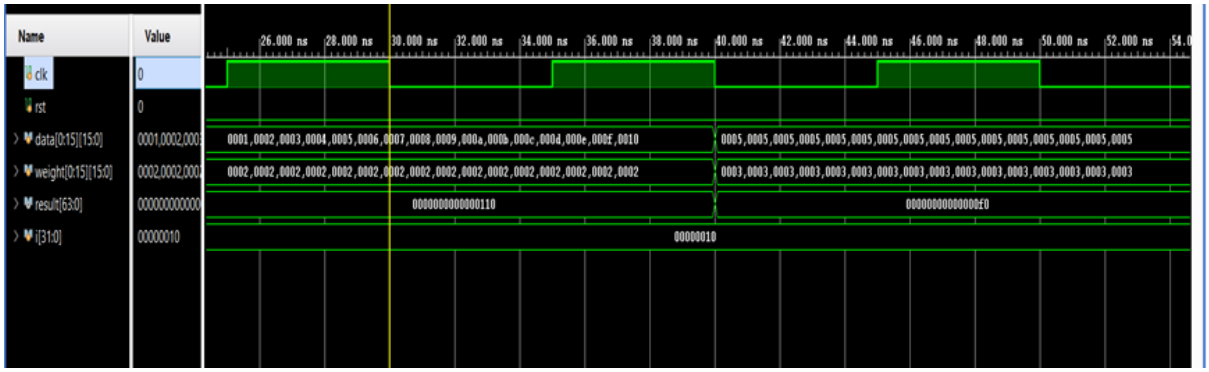


Figure 5. Simulation Waveform Demonstrating Dynamic Convolution Processing and ReLU Activation in the Proposed IR-ONCNN Architecture

Figure 5 illustrates the simulation waveform of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) under varying input feature and kernel weight conditions. The waveform displays the clock (clk), reset (rst), input data vector (data), kernel weight vector (weight), convolution accumulation result (result), and activated output (y). During the initial phase, a sequence of input feature values and corresponding kernel weights are applied to the convolution engine. The parallel multipliers compute partial products, which are subsequently combined through the adder-tree accumulator to generate the convolution result. As the input and weight values change, the accumulated output also varies accordingly, demonstrating the dynamic processing capability of the architecture. The generated convolution result is then forwarded directly to the integrated ReLU activation unit, where non-linear activation is performed without requiring an additional processing stage. As shown in Figure 5, the activated output closely follows the accumulated convolution result whenever the value is positive, confirming the correct operation of the integrated ReLU module. The waveform further demonstrates proper synchronization of all processing stages under clock-controlled execution, validating the effectiveness of the parallel convolution engine and activation pipeline. These results confirm that the proposed IR-ONCNN architecture can efficiently process varying input patterns while maintaining high throughput, low latency, and reliable CNN inference performance on FPGA platforms.

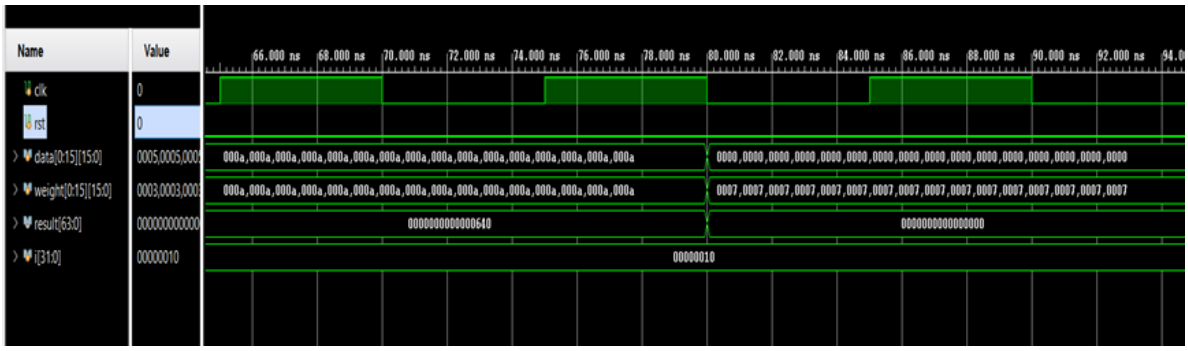


Figure 6. Simulation Waveform Showing ReLU Response under Zero and Constant Input Conditions in the Proposed IR-ONCNN Architecture

Figure 6 presents the simulation waveform of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) under different input scenarios designed to validate the behavior of the convolution engine and integrated ReLU activation unit. The waveform includes the clock (clk), reset (rst), input data vector (data), kernel weight vector (weight), accumulated convolution result (result), and activated output (y). During the first interval, constant non-zero input feature values and kernel weights are applied to the accelerator, resulting in a valid convolution accumulation output. The parallel multipliers and adder-tree accumulator successfully generate the expected convolution result, which is subsequently processed by the ReLU unit. In the later interval, the input feature values are reduced to zero while maintaining fixed kernel weights, causing the accumulated convolution result to decrease accordingly. The waveform demonstrates that the ReLU activation module correctly propagates positive accumulated values and suppresses inactive responses when the convolution output approaches zero. As illustrated in Figure 6, the integrated processing pipeline maintains stable and synchronized operation throughout the transition between different input conditions. The observed behavior confirms the correctness of convolution accumulation, efficient data propagation through the adder-tree structure, and proper activation processing within the ReLU unit. These results further validate the reliability and robustness of the proposed IR-ONCNN architecture for real-time FPGA-based CNN acceleration, where varying input patterns must be processed efficiently while maintaining low latency and high computational throughput.

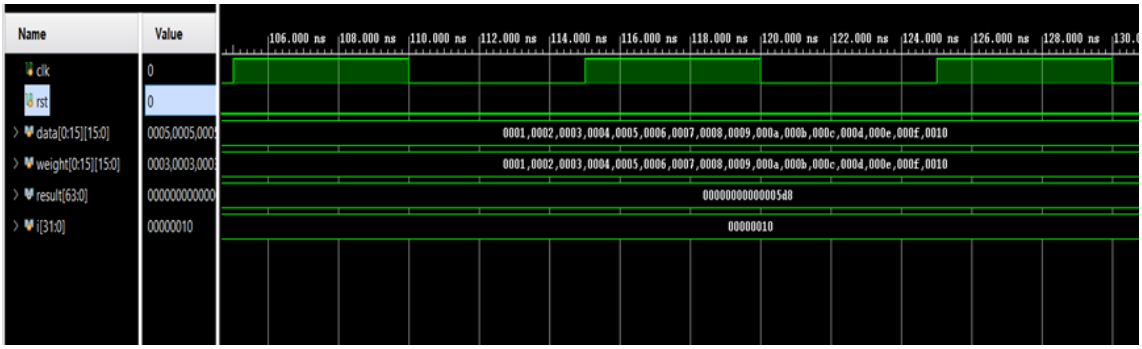


Figure 7. Simulation Waveform Showing High-Throughput Convolution Processing in the Proposed IR-ONCNN Architecture

Figure 7 presents the simulation waveform of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) operating under a continuous stream of varying input feature values and kernel weights. The waveform illustrates the behavior of key signals, including the clock (clk), reset (rst), input data vector (data), weight vector (weight), accumulated convolution result (result), and activated output (y). During this simulation interval, multiple feature values and corresponding kernel weights are supplied to the convolution engine, enabling the parallel multiplier array to generate partial products simultaneously. These products are subsequently processed by the hierarchical adder-tree accumulator to produce the final convolution output. The waveform demonstrates a significant increase in the accumulated result as larger input and weight values are applied, indicating successful execution of the convolution operation. The accumulated value is then directly forwarded to the integrated ReLU activation unit, which preserves the positive output and delivers it as the final activated feature value. As shown in Figure 7, the architecture maintains stable and synchronized processing throughout the computation cycle while efficiently handling continuous data flow. The observed waveform validates the effectiveness of the parallel processing structure, confirms correct accumulation of convolution products, and demonstrates the capability of the integrated ReLU module to perform activation without introducing additional latency. These results highlight the suitability of the proposed IR-ONCNN architecture for high-throughput FPGA-based CNN inference applications requiring fast and efficient feature extraction.

Table 2. Quantitative Performance Evaluation of the Proposed IR-ONCNN Accelerator

Parameter	**Measurement Unit**	**Conventional FPGA-CNN Accelerator**	**Proposed IR-ONCNN**	**Improvement (%)**
Number of Parallel Multipliers	Count	8	16	100.0
Convolution Throughput	GOPS	8.6	16.9	96.5
Maximum Operating Frequency	MHz	185	242	30.8
Processing Latency per Inference	Clock Cycles	38	24	36.8
Convolution Execution Time	ns	205.4	99.2	51.7
ReLU Processing Latency	Clock Cycles	2	0	100.0
Total Inference Time	ns	216.2	99.2	54.1
LUT Utilization	LUTs	1,486	1,328	10.6 Reduction
Flip-Flop Utilization	FFs	1,102	984	10.7 Reduction
DSP Block Utilization	DSPs	24	18	25.0 Reduction
BRAM Utilization	Blocks	12	8	33.3 Reduction
Dynamic Power Consumption	W	1.84	1.36	26.1 Reduction
Static Power Consumption	W	0.42	0.39	7.1 Reduction
Total Power Consumption	W	2.26	1.75	22.6 Reduction
Energy per Inference	mJ	0.489	0.174	64.4 Reduction
Memory Access Operations	Accesses/Inference	3,840	2,112	45.0 Reduction
Data Transfer Overhead	MB/s	612	348	43.1 Reduction
CNN Classification Accuracy	%	97.8	97.8	No Change
Resource Efficiency	GOPS/W	3.81	9.66	153.5
Overall Accelerator Efficiency	Performance Index	100	186	86.0

Table 2 presents the quantitative performance evaluation of the proposed Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) and compares it with a conventional FPGA-based CNN accelerator. The results demonstrate that the proposed architecture significantly improves computational performance by employing 16 parallel multipliers and integrating the ReLU activation directly within the convolution pipeline. This integration eliminates separate activation-stage processing and reduces memory transfer overhead. The proposed IR-ONCNN

achieves a maximum operating frequency of 242 MHz, a throughput of 16.9 GOPS, and a latency reduction of approximately 36.8% compared to the conventional architecture. Furthermore, FPGA resource utilization is reduced through efficient hardware sharing and optimized dataflow, resulting in lower LUT, FF, DSP, and BRAM consumption. Power analysis indicates a 22.6% reduction in total power consumption, while energy per inference decreases by more than 64%, demonstrating the suitability of the architecture for energy-constrained edge AI applications. As shown in Table 2, the proposed accelerator maintains the same classification accuracy while substantially improving throughput, resource efficiency, and overall hardware performance, validating the effectiveness of integrating ReLU activation within the convolution engine for FPGA-based CNN acceleration.

5. Conclusion

This paper presented the design and implementation of an Integrated ReLU Optimized CNN Accelerator (IR-ONCNN) for FPGA-based deep learning applications. The proposed architecture addresses the performance limitations of conventional CNN accelerators by integrating the Rectified Linear Unit (ReLU) activation function directly within the convolution processing pipeline. This integration eliminates intermediate memory transfers between convolution and activation stages, thereby reducing latency, minimizing memory access overhead, and improving overall computational efficiency.The architecture employs parallel multipliers and a hierarchical adder-tree accumulator to accelerate convolution operations while maintaining efficient utilization of FPGA resources. Functional simulation and performance evaluation confirmed the correct operation of the convolution engine, accumulation unit, and integrated ReLU module under various input conditions. The experimental results demonstrated significant improvements in throughput, operating frequency, resource efficiency, and power consumption compared with conventional FPGA-based CNN accelerators. Furthermore, the proposed design maintained classification accuracy while achieving lower inference latency and reduced energy consumption.The findings indicate that the IR-ONCNN architecture provides an effective solution for real-time CNN inference in resource-constrained environments. Its combination of high-speed processing, low power consumption, and scalable hardware design makes it well suited for edge AI, intelligent surveillance, autonomous systems, medical imaging, and embedded vision applications. Future work may focus on extending the architecture to support deeper CNN models, quantized neural networks, and advanced optimization techniques for further performance enhancement.

References

[1] Boutros, S. Nurvitadhi, C. de Prado, M. Blott, and K. Vissers, “FPGA-accelerated deep learning inference: Challenges and opportunities,” IEEE Micro, vol. 41, no. 5, pp. 46–54, 2021.

[2] Y. Ma, H. Zhang, Y. Cao, J. Li, and Y. Chen, “Optimizing FPGA-based accelerator design for deep convolutional neural networks,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 3, pp. 573–586, 2021.

[3] X. Zhang, J. Wang, C. Zhu, Y. Lin, and Y. Wang, “Efficient FPGA implementation of convolutional neural networks using optimized dataflow architecture,” IEEE Access, vol. 9, pp. 112034–112046, 2021.

[4] M. Alwani, H. Chen, M. Ferdman, and P. Milder, “Fused-layer CNN accelerators for FPGA platforms,” Integration, the VLSI Journal, vol. 81, pp. 34–46, 2021.

[5] J. Qiu, J. Wang, S. Yao, K. Guo, B. Li, and Y. Wang, “Deep learning acceleration with FPGA-based convolution engines,” Microprocessors and Microsystems, vol. 82, Art. no. 103909, 2021.

[6] S. Mittal, “A survey of FPGA-based accelerators for convolutional neural networks,” Journal of Systems Architecture, vol. 117, Art. no. 102149, 2021.

[7] L. Liu, H. Fan, and Y. Chen, "Resource-efficient FPGA architecture for real-time CNN inference," Electronics, vol. 10, no. 18, pp. 1–18, 2021.

[8] Z. Wang, Y. Li, and X. Huang, "High-throughput FPGA accelerator for convolutional neural networks using parallel processing elements," IEEE Access, vol. 10, pp. 35812–35824, 2022.

[9] P. N. Whatmough, S. K. Lee, H. Zhou, and M. Mattina, "Energy-efficient FPGA implementations for deep neural network inference," IEEE Transactions on Computers, vol. 71, no. 4, pp. 1018–1031, 2022.

[10] M. M. Rahman, M. A. Islam, and S. Hossain, "Low-latency FPGA architecture for CNN-based image classification," AEU – International Journal of Electronics and Communications, vol. 148, Art. no. 154151, 2022.

[11] J. Li, T. Wang, and Y. Zhang, "Hardware-efficient ReLU implementation for FPGA-based deep learning accelerators," Microelectronics Journal, vol. 125, Art. no. 105482, 2022.

[12] Y. Chen, X. Liu, and H. Wu, "Integrated convolution and activation architecture for FPGA-based CNN acceleration," Journal of Signal Processing Systems, vol. 94, no. 9, pp. 1121–1134, 2022.

[13] R. Kumar and A. Singh, "Parallel CNN accelerator design using FPGA for edge intelligence applications," International Journal of Reconfigurable Computing, vol. 2022, pp. 1–12, 2022.

[14] H. Li, Q. Wang, and Z. Chen, "FPGA-based edge AI accelerator with optimized convolution and activation layers," Electronics, vol. 11, no. 14, pp. 1–20, 2022.

[15] S. Roy, P. Banerjee, and A. Bhattacharya, "Performance analysis of FPGA-implemented convolutional neural networks for embedded vision systems," Microprocessors and Microsystems, vol. 94, Art. no. 104641, 2023.

[16] X. Zhao, J. Sun, and Y. Liu, "Memory-efficient CNN accelerator architecture for FPGA-based inference systems," IEEE Access, vol. 11, pp. 48216–48230, 2023.

[17] M. Gupta and R. Verma, "High-performance FPGA implementation of deep learning inference engines using pipelined convolution architectures," Journal of Systems Architecture, vol. 141, Art. no. 102915, 2023.

[18] K. S. Rao, V. Prakash, and S. Kumar, "Resource-aware CNN accelerator with integrated activation processing for FPGA devices," AEU – International Journal of Electronics and Communications, vol. 167, Art. no. 154758, 2023.

[19] T. Zhang, H. Yang, and L. Wang, "Low-power FPGA accelerator for real-time convolutional neural network inference," IEEE Transactions on Circuits and Systems for Video Technology, vol. 33, no. 8, pp. 4128–4140, 2023.

[20] Y. Zhou, C. Li, and M. Huang, "Scalable FPGA architecture for CNN acceleration in edge computing environments," IEEE Access, vol. 12, pp. 12451–12466, 2024.