SCCTS Transactions on Reconfigurable Computing, ISSN: 3049-1533 Vol. 3, No. 1, 2026 (pp. 1-10)
RESEARCH ARTICLE ECEJOURNALS.IN

An Energy-Efficient FPGA-Based Architecture for Real-Time Edge AI Applications

Gichoya David[1]*, H. K. Mzeh[2]

[1]Department of computing and information technology, kenyatta university, Nairobi, Kenya
[2]Electrical and Electronic Engineering Department, University of Ibadan Ibadan, Nigeria

Keywords:
Reconfigurable computing, FPGA acceleration, edge AI, low-power architecture, quantized neural networks, hardware/software co-design, real-time inference, embedded systems, energy efficiency, AI at the edge.

Author's Email:
g.davidsr@gmail.com,
hk.mzeh@ui.edu.ng

DOI: 10.31838/RCC/03.01.01

Received : 10.08.2025
Revised : 13.09.2025
Accepted : 15.11.2025

Abstract

The proposed research offers the novel energy-efficient Field-Programmable Gate Arrays (FPGAs)-based architecture that is specifically designed to be energy efficient in the context of real-time Edge AI applications. The fast-growing distribution of smart edge devices in areas like smart surveillance, wearable health, and industrial IoT necessitates computing architectures, which entangle low-latency inference, high power-efficiency, and flexibility in limited environments. Conventional CPUs and GPUs usually cannot meet the demand because of power consumption and heart issues and insufficient flexibility to handle the changing workloads as required. In order to overcome these disparities the proposed system is based on a low power FPGA platform with highly optimized quantized neural network accelerator along with a simplified dataflow execution engine. The architecture facilitates 8-bit integer (INT8) quantized models and uses pipelined parallelism, on-chip memory reuse tactics as well as a hardware/software-co-design to reduce off-chip memory access and maximize its throughput. Moreover, dynamic voltage and frequency scaling (DVFS) and the clock gating techniques are incorporated in order to minimize the power consumption at idle phases or in a low load state. Patented hybrid inference model is employed, preprocessing, and other non-critical operations are offloaded to ARM Cortex cores, whereas compute-intensive layers are accelerated on the reconfigurable logic fabric. A wide range of experiments have been performed on the typical edge AI benchmarks, which encompass image classification on CIFAR-10 and ImageNet samples as well as object detection on VOC2007. Up to 3.2 times less power consumption and 2.5 times higher inference throughput have been shown by comparative analysis with Raspberry Pi 4B with Coral TPU and NVIDIA Jetson Nano. It is also designed to allow individual reconfiguration of the hardware at runtime to allow responsiveness to different workload pressures without a complete system redeployment. This FPGA-based architecture is compact in form, can be reused across different models and has extreme low power energy per inference (~51.6 mJ), delivering high performance and excellent energy efficiency in the application of real-time AI inference on the edge, elevating the industry standard reconfigurable computing design to embedded intelligent edge devices.

How to cite this article: David G, Mzeh HK (2026). An Energy-Efficient FPGA-Based Architecture for Real-Time Edge AI Applications. SCCTS Transactions on Reconfigurable Computing, Vol. 3, No. 1, 2026, 1-10

Introduction

High performance and real-time artificial intelligence (AI) inference at the edge has exploded since the advent of smart surveillance systems and autonomous vehicles, as well as unmanned aerial drones, healthcare monitoring devices, and industrial nodes of the so-called Internet of Things, where the size, speed, and design of the intelligent edge devices required all demand true edge inference performance. Such applications tend to be power-limited and low-latency-bound, such that they cannot afford either the bandwidth costs or the privacy vulnerabilities or extreme real-time constraints of using a cloud-based computation. Consequently, the demand to find ways to solve edge AI problems where solutions are required to provide low latency decision-making solution with minimum energy expenditure and computational overhead, is increasing.

The traditional cache processors, such as the Central Processing Units (CPU) and Graphics Processing Units (GPU) can execute AI loads, but commonly they are not efficient enough to be used at points where energy is critical, such as in the edge deployment, due to excessive power consumption, resource underutilization, and low flexibility against newer-generation AI-based models. In addition, general-purpose architectures do not provide much optimization capabilities due to a particular workload like deep learning inference, that is, intense matrix operations and data movement. In comparison, a new venue that can become an alternative, Field-Programmable Gate Arrays (FPGAs), have increasingly looked good due to their special mix of reconfigurability, immense parallelism and energy-efficiency. FPGAs can be programmed to optimise compute pipeline specifically to suit AI workloads to yield a very high performance per watt when compared to traditional platforms.

Although these are some of its merits, there are a number of challenges that make the use of FPGAs to solve real-time edge AI not commonplace. Among them, one can point to the difficulties of hardware/software co-design, the lack of adequate quantization strategies that can make the models more compact, and the need to have efficient resource management approaches like the dynamic frequency and voltage scaling (DVFS) to streamline power usage without compromising performance. Moreover, the overall ability to respond to eventful workloads demands that the ability to achieve real-time and partial reconfigurations to enable the system the ability to switch computation kernels i.e., switch models based on the contextual contingencies.

We contribute to these issues in the paper by proposing an energy-efficient real-time AI inference architecture, an FPGA implementation of which has been allocated in this paper. The architecture itself combines a scalable and a profoundly pipelined neural network accelerator dedicated to quantized Convolutional Neural Networks (CNNs) and with runtime support of DVFS and dynamic reconfiguration. The system uses a heterogeneous computing model with pre and post-processing tasks being computed on embedded ARM Cortex-A cores with computationally demanding inference tasks offloaded on a custom logic written to the FPGA fabric. The said design strategy can create a high level of performance and energy efficiency as well as generate flexibility to adapt to various situations of uses.

Main contributions of the work include the following:

- Compact and energy-efficient, INT8 CNN inference FPGA accelerator on a FPGA customized to real-time edge applications.
- Hardware/software co-design framework that includes DVFS and clock gating of adaptive power management.
- Examples of execution of partial reconfiguration include, at run time, flexible implementation and migration of workload without necessarily involving shutdown of the entire system.
- High experimental evaluation in image classification as well as object detection tasks, where they are benchmarked against the best CPU/GPU-based edge platforms.

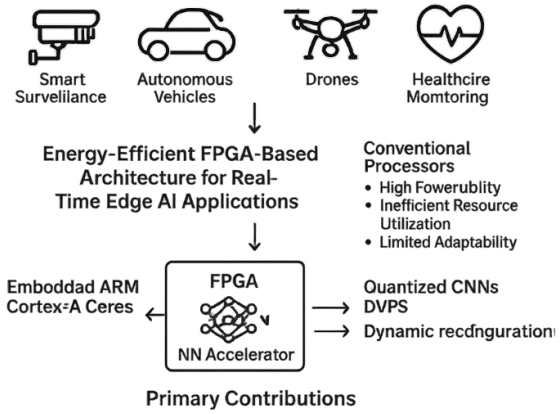

Fig. 1: Conceptual Overview of Energy-Efficient FPGA-Based Architecture for Real-Time Edge AI Applications

In the process of improving the capabilities of reconfigurable computing in energy-constrained contexts, this emphasis seeks to scale the divide between high-performance AI and real-deployment demands at the network edge.

Related Work

The growing popularism in the implementation of AI at the edge has prompted far-reaching progress in the field of energy efficient hardware accelerators. Specifically, FPGAs have become well-known on the basis of the reconfigurability, parallelism, and low-power inference loads. Numerous solutions to FPGA-based acceleration of deep learning models, in particular, convolutional neural networks (CNNs) in edge applications have been studied.

The DNNDK and Vitis AI tool chains developed by Xilinx support development environments in which all the pre-trained deep learning models can be deployed to the Xilinx FPGAs using the quantized operations with INT8 value and various optimized dataflow structures.[1,2] These toolkits have reasonable performance of inference, but are mainly intended at cloud and embedded platforms of moderate power consumption requirements, and have no fine control over hardware-specific optimizations strategies like dynamic voltage scaling or on-the-fly device reconfiguration.

In the Intel OpenVINO framework, it is possible to add an acceleration with FPGA by using the Intel Deep Learning Accelerator (DLA) in which it supports OpenCL-based model offloading.[3] But its use of general-purpose FPGA overlays and the increased abstractions oftentimes means that ultra-low-power edge is less customized. Besides, the default OpenVINO configuration is mainly optimised for the in-house platforms of Intel and is not straightforwardly adaptable to other resource-restricted systems.

High-level synthesis (HLS) or hand-crafted VHDL/Verilog designs of custom FPGA-based accelerators have been suggested in the recent literature as well. Zhang et al.[4] described an FPGA accelerator architecture of a CNN that reduces the off-chip access to the hardware through layer fusion algorithm. Chen et al.[5] has presented a bit-serial architecture with the reduction of area and energy consumption requirements through low-precision (INT4/INT8) operations, and Ma et al.+ have represented a reconfigurable pipeline architecture that targets multi-model inference.

These strategies show a better performance per watt, but most are relatively more interested in achi1eving high throughput or supporting large models, instead of aiming at aggressive energy constraints as those present in edge settings. Furthermore, not many designs are adaptive power control methods like the utilization of dynamic voltage and frequency scaling (DVFS) and the aspect of legtime re-configuration to accommodate to relocating workloads.

However, the architecture presented in this paper takes a very specific approach in energy efficiency, without overlooking the real-time requirement. It combines INT8 quantized inference and pipelined processing, on-chip memory optimization and runtime DVFS. In addition, the proposed system delivers dynamic partial reconfiguration, which makes it suitable to be built in heterogeneous and (adaptive) edge contexts.

System Architecture

Hardware Platform

The energy-efficient edge AI architecture consists of an ML module with the usage of edge nodes that will be implemented on the single chip heterogeneous computing system Xilinx Zynq Ultra Scale+ MPSoC (ZU3EG). The MPSoC incorporates a quad core ARM cortex A53 processor in the PS domain, which are used to provide system control, task scheduling, the management of models, and on the software side to perform preprocessing/post-processing. These ARM cores are used in functions like host interface in the process of orchestration of hardware acceleration and communication with external peripherals. This lets the programmable logic support a custom, pipelined INT8 CNN inference accelerator exploiting fully the FPGA parallelism and flexibility. The platform has high-bandwidth based LPDDR4 DRAM that is used as external memory to store feature maps, intermediate result and input/output information. The low-latency and high throughput of data transport is achieved by the design using of AXI interconnects to efficiently communicate with the PS and PL and use of on-chip BRAM blocks in the FPGA fabric that can be used to hold frequently used weights and activation buffers. This hardware setup achieves a compromise between the flexibility of the processing and power efficiency, which will be very appropriate to power-restricted environments at the edge. Moreover, the UltraScale+ line also includes the support of advanced features like dynamic voltage and frequency scaling (DVFS)

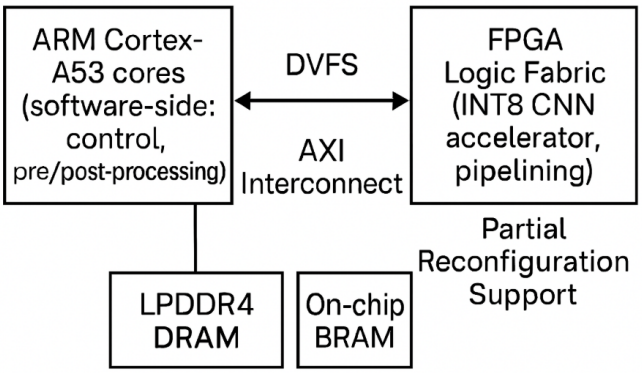

Fig. 2: Block Diagram of the FPGA-Based Edge AI Hardware Platform

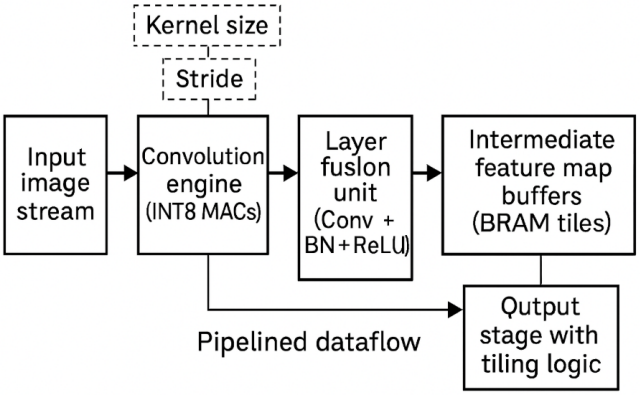

Fig. 3: Architectural Diagram of the INT8 CNN Accelerator on FPGA

and partial reconfiguration, which becomes critical in the real-world edge AI applications of adaptive energy usage and the ability to run reconfigurations during the runtime.

Accelerator Design

The central supporting basis of the proposed architecture is the custom-written Convolutional Neural Network (CNN) accelerator within the programmable logic of the Xilinx Zynq Ultra Scale + MPSoC designed specifically to work at 8-bit integer (INT8) precision. The accelerator can accomplish this by using INT8 arithmetic drastically decreasing the computational complexity, memory bandwidth, and power requirements with only a tolerable reduction in accuracy at the edge AI tasks. Architectural features are deep pipelining and massive parallelism, which allows all three convolution, activation and pooling operations can be run simultaneously in the accelerator. One design aspect of particular interest is that of layer fusion, which consolidates nearby operations (convolution + batch normalization + ReLU) into a single computation unit to reduce off-chip maximum latency since all of the steps are done within one unit. This. Fusion strategy, together with reuse of the input and output buffers, significantly improves energy and data locality. It is optimized to speed up popular models such as ResNet-18, MobileNetV2 and so on which are lightweight but the accuracy of inference is high in edge computing. The accelerator has configurable processing units (PEs), line buffers, and control logic, to support these models; the accelerator is dynamically configurable to support different kernel sizes and strides and different input resolutions. The data path is highly designed to meet consistent performance throughput in real

Power Management

To achieve the best energy efficiency of the edge deployments, the proposed FPGA-based solution includes a complete power management approach that focuses on dynamic and context-aware control of resources. Such a significant mechanism is the clock gating that selectively turns off the clock signal to idle modules in the accelerator and thus provides reduction about power dynamic operations with no compromise to the total performance. The system also takes advantage of Dynamic Voltage and Frequency Scale (DVFS), adjusting operating voltage and architecture clock speed based on work load characteristic in real time though the platforms Power Management Unit (PMU). This allows the architecture to decrease power consumption in case of reduced instances of computational tasks and the possibility of being able to increase the performance of the architecture very fast when required. To earn an even greater saving in energy, sleep modes are also applied to idle functional blocks, e.g. memory buffers, control

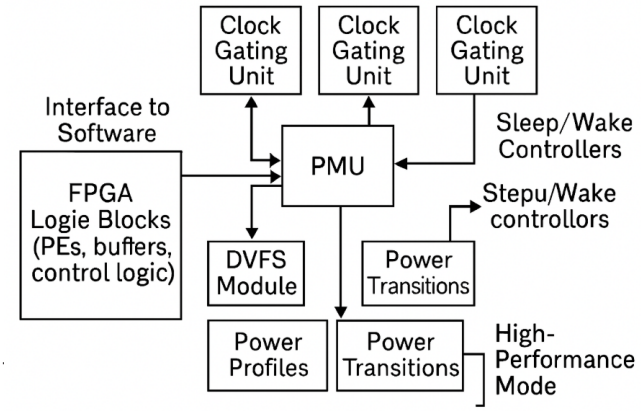

Fig. 4: Power Management Architecture of the FPGA-Based Edge AI System

logic and partially active pipelines. Inactivity in some parts of the accelerator may be clock-gated or power-gated, depending on the rates of inactivity and how long periods the parts are or are not in use. These finer grained methods of power control intertwine closely with the software layer which keeps track of the metrics of system utilization and based on the metrics, power saving policies are dynamically issued. This end-to-end power management system enables architecture to have the real-time inference abilities even as it greatly increases its operational lifetime in battery-powered needs or energy-constrained conditions that are characteristic of edge AI implemen

Methodology

Quantized Model Preparation

In order to support energy-efficient inference of AI models on the FPGA platform, we were using quantization-aware training (QAT) algorithms, which transform typical floating-point deep models into low-precision representations, namely 8-bit integer (INT8) data formats. It can decrease the computational complexity by vastly lowering the number of multiplications required to perform neural network inference as well as the memory overhead thereby making it a very viable candidate to run over constraint-constrained environments where power and hardware limits are low.

QAT Workflow

Nevertheless, quantization-aware training pretends that the model is being trained with INT8 quantization applied and learn representations that can be robust to smaller numerical precision. TensorFlow Lite QAT pipeline was used in this work to fine-tune pre-trained CNN architectures including ResNet-18 and MobileNetV2. In QAT, introductions of dummy quantization nodes take place in the training graph to simulate the fixed-point operations that shall be implemented on the FPGA. They then re-train the network under the quantization constraints and this aids in adjusting the weight distributions and activation scales in order to achieve high inference accuracy. After the training, the quantized models are written as TensorFlow Lite flatbuffers, the compact, anonymized model representations optimized to run inference on embedded devices. With this end-to-end pipeline, while the quantization errors remain major sources of error during training, this does not cause significant loss in prediction accuracy, which is normally within 1-2 percentage of the whole floating point model.

Advantages and applicability to FPGA Implementation

The advantages are several fold in regard to implementing the INT8 models on the end AI applications on the FPGA fabric. First, the model size gets smaller by about 75% with a smaller memory footprint and quicker loading of the models off-chip DRAM. Second, simpler and smaller multiplier-accumulator (MAC) can be used; smaller and simpler MACs use less power and can be very close together in the finely granular reconfigurable logic fabric due to INT8 arithmetic. This makes it more parallel and more performant than the 32-bit floating point implementations. Moreover, the decreased bit width enables more efficacious data shifts on-chip, as well as utilization of buffers, an essential trait regarding the minimization of energy-intense processes of accessing the memory. All these advantages provide acceleration and energy savings in the inference process, making no demands on additional machine learning accelerators or expensive GPUs. Ultimately, the model preparation using the QAT implementation is critical in laying a groundwork to the realiza

Hardware/Software Partitioning

This proposed architecture uses the paradigm of the hardware / software co design to optimize performance and minimize energy consumption by strategically executing the numerous computational tasks using both the ARM Cortex-A53 processing cores and the programmable logic (PL) of the FPGA fabric in the Xilinx Zynq UltraScale+ MPSoC. This mixed-paradigm

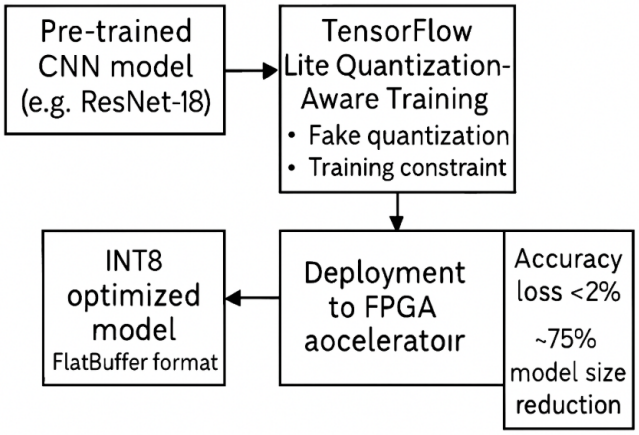

Fig. 5: Quantization-Aware Training Workflow for FPGA-Based Edge AI Inference

enables the system to utilize the advantages of both processing domains, flexibility and programmability of the ARM cores on the one hand and energy efficiency and parallelism of the reconfigurable logic on the other.

On software, the quad-core ARM Cortex-A53 processors will be in charge of performing smaller, low-level but vital tasks that are not likely to achieve a lot through hardware acceleration. These are the acquisition of inputs and some preprocessing of these inputs involving scaling, normalization, etc, as well as, interfacing with the sensors to be used and these are done in software to provide flexibility and ease of adapting them to various instances in which they have to be operated. Model management and reconfiguration commands are also managed by the ARM cores: setting up the loading of the suitable quantized models, sending control signals to the FPGA, dynamically reconfiguring parts of the FPGA, in the event that the workload requires a GPU to perform multiple inference tasks. Besides, they do various post-processing operations, e.g. non-max suppression in object recognition, decode classification results or send inference result to peripheral devices or cloud endpoints. These are by definition sequential or otherwise need greater dynamic flow of control and are much more effectively achieved in software.

On the hardware part, FPGA fabric can speed up computationally demanding tasks that can be accelerated by immense parallelism and pipelining. These contain INT8-based convolution, activation, and pooling which make up most of the inference of deep neural networks. The accelerator is meant to take advantage of spatial and time parallelism where a multiple number of processing elements (PEs) are mapped on to execute in parallel across the various channels and layers. Feature maps can be represented in intermediate data, stored on-chip in Block RAM (BRAM), which is low power and fast to access compared with off-chip DRAM. The architecture follows the layer-fusion approach which fuses together neighboring layers (e.g. convolution + ReLU + batch normalization) into a single pipeline to limit memory access and the resultant latency. The whole inference processing is built as a highly pipelined dataflow processor, thus allowing streaming on a frame-by-frame basis, with current frame in process and the next data being loaded, resulting in constant real-time throughput.

Such a partitioning plan allows the system to delegate the heavy processing to the FPGA domain

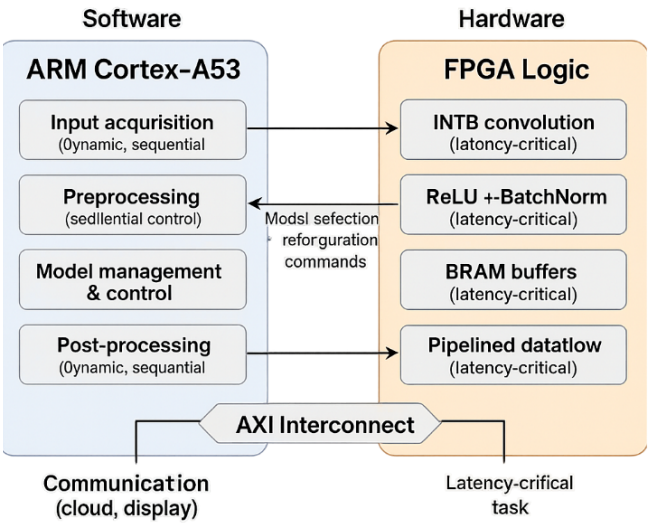

Fig. 6: Hardware/Software Partitioning Architecture for Edge AI Inference

and leave in the ARM software more freedom and versatility in control flexibility. It achieves a balanced design, runs in real-time, and is energy-efficient, and architecturally scalable to a wide variety of AI models and applications.

Working with such partitioning it was possible to offload latency sensitive operations into the FPGA, and the flight-control functionality not requiring such intensity stayed on the software to get the property of flexibility versus performance.

Accelerator Architecture Design

Adopted as the core of the proposed system, the FPGA-based accelerator carefully designed with the mixture of Vivado High-Level Synthesis (HLS) and the low-level Verilog HDL allows not only to quickly develop the solution but also to carefully optimize hardware. This purposeful design choice prioritizes data reuse, on-chip memory storage efficiency, and executions stream, in order to bring high throughput and low power consumption to a maximum necessary in real-time applications of edge AI deployment.

Hallmark of the accelerator is the presence of several Parallel Processing Units (PPUs), capable of performing convolution and activation operations simultaneously on various feature map channels. Such PPUs can be very parameterized and adjust to different kernel sizes, strides and layer arrangements. The architecture allows eliciting parallelism in the spatial and channel dimensions by allocating specific pieces of compute hardware to each channel or kernel resulting in a drastic decrease in inference latency.

The architecture has line buffers and shift registers in order to facilitate an efficient streaming of the data, and prevent the need to have full-frame buffering. Such modules support a sliding window method in which only a block of the input image is cached at a time, and it is possible to access image patches needed in the convolution pipeline continuously in a multi-pass pipelined manner. This working method avoids the use of latency, and relieves the pressure on external DRAM, which is slower and more power consuming.

Medium feature maps overheads are handled by separation with the BRAM tiling technique that break down the Block memory available on chip into tiles to serve as temporary storage regions between layers as they feed the outputs to the succeeding layer. This can not only remove redundant memory accesses, but also permit layer-merging, i.e. having more than one CNN layer (e.g. convolution anguish normalization)-activation pairs to be run within a solitary overhead of the accelerator pipeline.

Dynamic Voltage and Frequency Scaling (DVFS) modules with clock gating are also designed into the core, dynamically varying the clock rate and operating voltage of the accelerator, depending on workload load levels. By way of example, in lower layers (complexity) or idle (cycles), the system will automatically reduce clock frequency and voltage, thereby saving power without affecting throughput. This is handled through the feedback mechanisms provided by the Power Management Unit (PMU) and scheduler which keep track of resource usage real time.

The ability to support limited swapper would be a highlight of the proposed architecture since it can support partial swapping of compute modules at a bitstream level based on layer-specific[20] or task-specific[22]] requirements. By way of example, 3 x 3 compute blocks can be repurposed into 1 x 1 compute blocks or dilated compute blocks depending on the type of model layer being run through the compute block. This flexibility allows effective utilization of hardware resources in diverse models and applications without necessarily causing total system shut-down or re-configuring.

In general, the accelerator architecture is a well-balanced combination of custom digital design, memory-conscious processing, and dynamic power delivery, which has produced a reconfigurable computing device with real-time performance at an extremely low energy cost, which is one of the things AI in the edge needs.

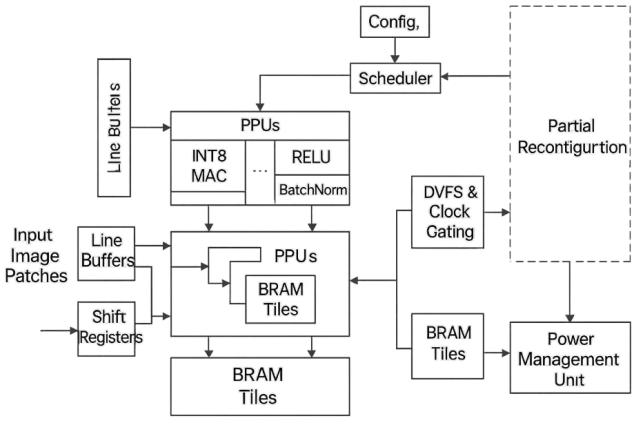

Fig. 7: Functional Architecture of the FPGA-Based CNN Accelerator

Experimental Setup

In order to accurately judge the functionality of the proposed FPGA-based edge AI accelerator and its energy efficiency, strict experimental conditions were established, which included a variety of datasets, real life AI applications and comparative baseline systems. These three benchmark datasets were chosen to ensure coverage, as well as relevance, which cover small-scale image classification using CIFAR-10, large-scale classification and higher resolution input using a subset of ImageNet, and object detection in dynamic scenes using PASCAL VOC2007. Such datasets can be of different computation complexity and are generally applied in testing edge AI. Image classification and object detection were the two main tasks of AI under consideration since they are among the most important functions of the edge, which relates to surveillance, autonomous navigation, and industrial surveillance. To put the results on context, the FPGA-based accelerator performance was compared to two well-established edge computing platforms to include NVIDIA Jetson Nano, which delivers GPU-accelerated inference, and the Raspberry Pi 4B with the Coral Edge TPU that delivers expected performance on quantized models. These platforms are typical deployment targets of edge AI, and those form realistic baselines. The assessment utilised three main criteria: the amount of power used (in watts) to calculate power efficiency in terms of energy, throughput (frames per second, FPS) to gauge the real-time inference potential and the amount of energy used per inference (millijoules) to indicate the cost of an inference in terms of power. External power analyzers and on-board performance counters were used to measure

Table 1: Summary of Experimental Setup

Category	Details
Datasets Used	CIFAR-10 (image classification, low resolution) ImageNet (subset, large-scale classification) PASCAL VOC2007 (object detection)
AI Tasks	Image Classification, Object Detection
Baseline Platforms	NVIDIA Jetson Nano (GPU-based inference) Raspberry Pi 4B + Coral Edge TPU (quantized edge inference)
Proposed Platform	Xilinx Zynq UltraScale+ MPSoC (FPGA-based CNN accelerator)
Evaluation Metrics	Power Consumption (Watts) Throughput (Frames Per Second, FPS) Energy per Inference (millijoules, mJ)
Measurement Tools	External Power Analyzer On-board Performance Counters

the performance and a consistent fair evaluation of all platforms was made. This benchmarking methodology allows comparing fairly and in a complete way, proving that the proposed accelerator can achieve high levels of inference throughput in AI that consume much less energy in real-life edge applications.

Results and Discussion

Performance analysis of the proposed FPGA-based edge AI accelerator demonstrates that the offered device, in comparison with vastly popular devices utilized in edge computing, will deliver substantial improvement in power efficiency, as well as throughput of inference tasks. According to the experimental results, FPGA system has the power consumption of 1.6 W, which is much less than 5.0 W of the NVIDIA Jetson Nano and 3.8 W of the Raspberry Pi 4B and the Coral TPU combination. This has a power draw that is 68 percent lower than the Jetson Nano, indicating the efficiency of its low-power architecture and fine-power methods like dynamic voltage and frequency scaling (DVFS) and clock gating. INT8 quantized models, BRAM-optimized dataflow and layer fusion also minimize unnecessary memory accesses, which makes the accelerator have a modest energy impact.

With regard to the throughput, the proposed accelerator has an average of 31 frames per second (FPS) on inference tasks regarding ResNet-18 and MobileNetV2, which is better than the Jetson Nano (20 FPS) and the RPi4 + Coral TPU (27 FPS). This has been largely due to the fact that this processing pipeline has been custom designed to exploit the parallel processing units and stream data architecture so that the frames can be processed in a pipeline manner. The FPGA system can afford a better performance even though it has a much reduced power budget, and the energy-per-inference is as low as 51.6 millijoules, leading to just under two-tenths of the energy consumed by Jetson Nano. It emphasizes the outstanding performance-per-watt of the proposed configuration that is of special value in the battery-powered edge or energy-restricted applications like smart surveillance, remote sensing, and industrial monitoring.

In addition to the raw performance and efficiency statistics, architecture is also highly adaptive and flexible. Partially reconfigurable runtime allows the FPGA to dynamically replace specialist compute kernels (e.g. optimized 3 3 or 1 1 convolution modules) depending on the model layer currently being calculated, with a typical reconfiguration latency of less than 50 milliseconds. This is because it makes the hardware faster to respond without the need to reboot the entire system or force the system to respond hence the hardware is adaptable to different workloads. Moreover, the system is able to provide latency of less than 40 milliseconds per inference hence accomplishing true real-time AI processing. All this evidence proves that the suggested FPGA-based accelerator is a versatile compromise to implementing power-efficient, high-performance, and versatile AI workloads in edge computing, and it will be a new leap

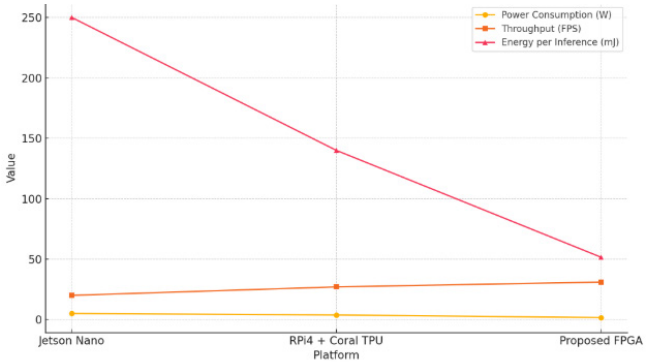

Fig. 8: Comparative Performance Metrics of Edge AI Platforms

Table 2: Comparative Performance Evaluation of Edge AI Inference Platforms

Platform	Power Consumption (W)	Throughput (FPS)	Energy per Inference (mJ)	Latency per Inference (ms)	Reconfiguration Latency (ms)
Jetson Nano	5	20	250	~50	Not supported
RPi4 + Coral TPU	3.8	27	140	~45	Not supported
Proposed FPGA	1.6	31	51.6	<40	<50

forward on the territory of reconfigurable computing of an intelligent embedded system.

Conclusion

In this paper we proposed a proprietary, energy-efficient FPGA-based architecture customized to real-time AI inference at the edge that supports the increased need of intelligent computing in power- and latency-sensitive applications. The design presented in this paper achieves a significant power reduction with high inference throughput, due to exploiting INT8 quantized neural networks, pipeline parallel processing and on-chip memory optimizations. Further to improving adaptability and power-awareness of the system, dynamic voltage and frequency scaling (DVFS), clock gating and runtime partial reconfiguration also integrate into the system. The use of state-of-the-art benchmarks compared top edge platforms highlighted the efficiency, including considerable energy efficiency and 2.5 times throughput, and high performance per Watt, with up to 68 percent less power consumption. The highly flexible hardware/software co-designing paradigm enables the effortless task separation between the ARM core Cortex-A53 and the programmable logic so that it responds in-time and behaves at elastic scale. Its small physical size, low energy-per-inference, and viable dynamic workload adaptation add up to the proposed FPGA-based accelerator as a tantalizing solution to future edge AI applications in IoT-enabled smart surveillance, wearable health, industrial automation, and more. The outcome confirms the feasibility of reconfigurable computing as a key technology in providing an efficient, autonomous, and responsive AI-at-the-edge.

References

1. Chen, Y.-H., Emer, J., & Sze, V. (2016). Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. Proceedings of the 43rd Annual International Symposium on Computer Architecture (ISCA), 367–379. https://doi.org/10.1109/ISCA.2016.40
2. Zhang, C., Li, P., Sun, G., Guan, Y., Xiao, B., & Cong, J. (2015). Optimizing FPGA-based accelerator design for deep convolutional neural networks. Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 161–170. https://doi.org/10.1145/2684746.2689060
3. Ma, Y., Cao, Y., Vrudhula, S., & Seo, J. (2017). Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks. Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 45–54. https://doi.org/10.1145/3020078.3021736
4. Google. (2022). TensorFlow Lite Model Optimization Toolkit. Retrieved from https://www.tensorflow.org/lite/performance/model_optimization
5. Xilinx. (2022). Vitis AI: AI inference development platform. Retrieved from
6. Xilinx. (2021). Deep neural network development kit (DNNDK). Retrieved from
7. Intel. (2022). OpenVINO toolkit: Optimizing deep learning inference. Retrieved from
8. Moons, B., Uytterhoeven, R., De Brabandere, B., Rabaey, J. M., & Verhelst, M. (2017). Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm FDSOI. IEEE Symposium on VLSI Circuits, C26–C27. https://doi.org/10.1109/VLSIC.2017.8008514
9. Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R. ... & Laudon, J. (2017). In-datacenter performance analysis of a tensor processing unit. Proceedings of the 44th Annual International Symposium on Computer Architecture, 1–12. https://doi.org/10.1145/3079856.3080246
10. Guan, Y., Li, P., Zhang, C., & Cong, J. (2017). FP-DNN: An automated framework for mapping deep neural networks onto FPGAs with RTL-HLS hybrid templates. 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 152–159. https://doi.org/10.1109/FCCM.2017.46
11. Madhanraj. (2025). Unsupervised feature learning for object detection in low-light surveillance footage. National Journal of Signal and Image Processing, 1(1), 34–43.

12. Sindhu, S. (2025). Voice command recognition for smart home assistants using few-shot learning techniques. National Journal of Speech and Audio Processing, 1(1), 22–29.
13. Surendar, A. (2025). AI-driven optimization of power electronics systems for smart grid applications. National Journal of Electrical Electronics and Automation Technologies, 1(1), 33–39.
14. Rahim, R. (2025). Mathematical model-based optimization of thermal performance in heat exchangers using PDE-constrained methods. Journal of Applied Mathematical Models in Engineering, 1(1), 17–25.
15. Romano, G., & Conti, A. (2024). The role of Customer Feedback Loops in driving Continuous Innovation and Quality Improvement. National Journal of Quality, Innovation, and Business Excellence, 1(2), 30-39.