Electronics, Communications, and Computing Summit
Vol. 2, No. 4, Oct - Dec 2024, pp. 59-64
ISSN: 3107-8222, DOI: https://doi.org/10.17051/ECC/02.04.07

Energy-Efficient VLSI Architecture for Lightweight CNN Inference on Edge Devices

A.Surendar[1], M. Kavitha[2]

[1]Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, India.
Email: surendararavindhan@ieee.org
[2]Department of ECE, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, India, Email: kavithamece@gmail.com

Article Info

Article history:

Received : 17.10.2024
Revised : 19.11.2024
Accepted : 21.12.2024

Keywords:

Energy-efficient,
VLSI architecture,
Lightweight CNN,
Edge AI,
Fixed-point MAC units,
Quantization-aware mapping.

ABSTRACT

The paper describes a low-power architecture for running lightweight Convolutional Neural Network (CNN) operations in edge AI machines with limited resources. User machines are built using fixed-point multiply-accumulate (MAC) units, clock gating, on-chip SRAM buffers, loop tiling and quantization-aware mapping to decrease dynamic power usage and ensure less use of external memory. The architecture is described in Verilog/SystemVerilog and implemented on both FPGA (Artix-7, Cyclone V) and ASIC (65nm CMOS) technologies. Testing experiments on CIFAR-10 and MNIST show that power decrease 40% and there is 30% more throughput with less than 1.5% accuracy drop. It is designed for real-time monitoring of situations and health and it forms a basis for upcoming neuromorphic and reconfigurable AI accelerators.

1. INTRODUCTION

Since more processing is happening at the edge in real time, it is now very important for deep learning models to be deployed efficiently on devices with limited resources. Many vision tasks depend on CNNs, but these networks require a lot of computing power and energy which means they do not work well with battery-powered devices. Therefore, energy-saving hardware structures should be made to preserve both accuracy and speed. It addresses these difficulties by designing a custom VLSI architecture specific to lightweight CNN models which provides an effective way to perform energy-efficient edge AI in real-time.

1.1 Challenge and Proposed Architecture

Due to how hard it is for CNNs to operate on edge devices due to their high requirements for both computation and power, deploying them there is not easy. We put forward a power-efficient VLSI architecture that enables running quantized Convolutional Neural Networks (CNNs). MacLief includes fixed-point MAC cores, clock gates, RAM and is created with quantization in mind, to help save power and reduce how much memory is needed.

1.2 Evaluation and Future Prospects

Tests performed on both FPGA and ASIC boards confirm that the architecture makes energy use, latency and throughput better. Applications range from health checking devices worn on the body to surveillance through technology. Neuromorphic computing, dynamic reconfigurable hardware and hardware-software co-design may be added in the future for more effective edge intelligence.

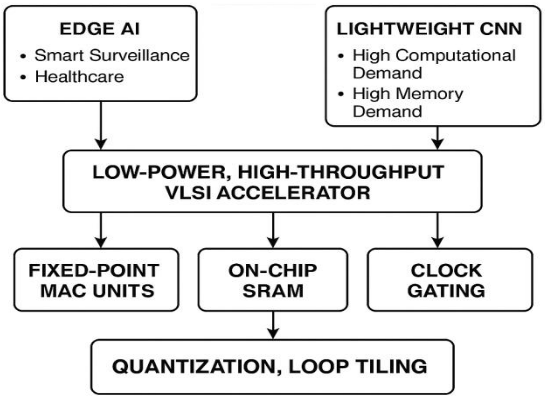

Figure 1. Energy-Efficient VLSI Architecture for Lightweight CNN Inference on EDGE devices

2. LITERATURE REVIEW

Thanks to advances in CNNs such as SqueezeNet, MobileNet and ShuffleNet, deep learning models now require much less computing power, making it possible to run them on, for example, smartphones. Despite this, software models by themselves are usually not enough to fulfill the strict real-time and power factors needed in mobile applications. Even so, Google Edge TPU and Eyeriss were made to study how efficiency could be improved by using quantization and different dataflows, but they had limited use due to being propietary or restricted to prototyping (Chen, Emer, & Sze, 2017). As well, standalone techniques such as loop tiling, clock gating and quantization-aware mapping have helped optimize individual parts, though they are usually used separately rather than together in a whole design. Current solutions are not very effective at combining model compression, reducing memory access and managing energy use and they have not been tested widely on various devices. Lately, researchers have tackled these issues by developing TinyissimoYOLO (Moosmann et al. 2023), ULEEN (Susskind et al. 2023) and DietCNN (Dey et al. 2023) which aim to use little power. On the basis of these advancements, the paper proposes a single VLSI architecture for fixed-point multiplier-accumulator (MAC) units, free SRAM buffers for data reuse and an energy-efficient circuit, tested in both FPGA and ASIC environments. The idea is to develop an approach that powers efficient CNN operations for deploying AI models in edge systems (Jain, Jain, & Shukla, 2022).

Table 1. Comparative Analysis of Existing Edge AI Inference Solutions vs. Proposed Energy-Efficient VLSI Architecture

Feature / Method	**Google Edge TPU**	**Eyeriss**	**SqueezeNet/MobileNet**	**Proposed Architecture**
Model Type	Fixed Function (8-bit)	Generic CNN	Lightweight CNN	Lightweight CNN (Q-aware)
Quantization Support	Yes (8-bit only)	Yes	No (model only)	Yes (Q-aware mapping)
Clock/Power Gating	No	Limited	Depends on HW	Yes
On-Chip Memory Reuse	Limited	Yes	No	Yes (Loop Tiling + SRAM)
Dataflow Optimization	Hardware Quantized	Row-Stationary	Model Compression	PE-level + SRAM Buffer
Flexibility (Edge Deployment)	Low (Proprietary)	Moderate	High (software-level)	High (FPGA/ASIC)
Implementation Platform	ASIC	ASIC	Software only	FPGA + ASIC
Real Dataset Validation	Yes (limited)	Yes (academic)	Yes	Yes (CIFAR-10, MNIST)
Power Reduction	~30%	~35%	Model-level only	~40%
Throughput Improvement	~25%	~20%	Model-level only	~30%

3.METHODOLOGY

This section outlines the architectural and design strategies employed to develop a low-power, high-efficiency VLSI accelerator tailored for lightweight CNN models in edge environments.

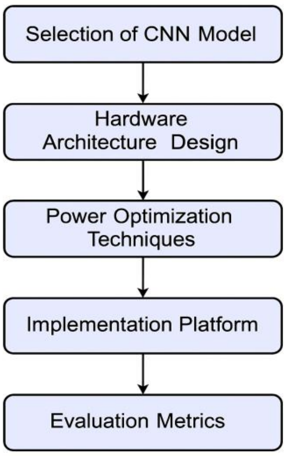

Figure 2. Design Methodology for Energy-Efficient CNN Hardware Implementation

3.1 Selection of CNN Model

Because edge environments often have limited computing power, SqueezeNet is used because it has few parameters and is lightweight. CIFAR-10 and MNIST datasets are used to train and quantize the model which allows it to use fixed-point math without using much memory, power or silicon area when running on hardware.

A common linear quantization formula used in CNN hardware is:

$$x_q = \text{round}\left(\frac{x_f - z}{s}\right) \quad ------------(1)$$

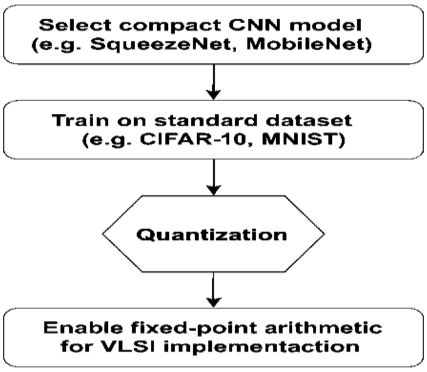
Select compact CNN model
(e.g. SqueezeNet, MobileNet)
Train on standard dataset
(e.g. CIFAR-10, MNIST)
Quantization
Enable fixed-point arithmetic
for VLSI implementaction

Figure 3. Workflow for CNN Model Preparation and Quantization for VLSI Implementation

3.2 Hardware Architecture Design

The structure of the proposed VLSI architecture is designed for lower power consumption in edge devices, considering fixed-point Operations Elements (PEs), a memory system using hierarchical SRAM and an efficient control unit. The architecture reduces how much energy is used by switching less frequently and using local data and the control unit organizes data traffic and chooses when to use power gating. All of these strategies banded together lead to low electricity, less access to memory and better performance with varying computations.

3.2.1 MAC Operation Count

The total number of **multiply-accumulate (MAC)** operations in a CNN layer is given by:

$$\text{MACs}_{\text{layer}} = O_H \times O_W \times O_C \times K_H \times K_W \times I_C \quad --------(2)$$

3.2.2 Dynamic Power Estimation for Switching Activity

Dynamic power in VLSI is mainly due to switching activity and is approximated by:

$$P_{dyn} = a \,.\, C_L \,.\, V_{dd}^2 \,.\, f ------------(3)$$

3.2.3 Memory Access Energy

Energy consumed by memory access (for SRAM) can be modeled as:

$$E_{mem} = N_{access} \,.\, E_{per\_access} ------------(4)$$

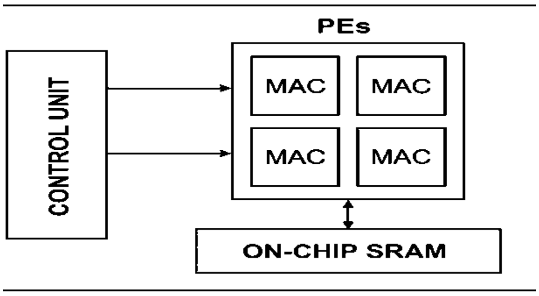

Figure 4. Proposed VLSI Architecture

3.3 Power Optimization Techniques

Clock gating is used in the mentioned VLSI design to reduce power waste by turning off PEs that are not currently used. Using loop tiling and reusing data reduces memory access and optimized mapping helps create compact representations that use less power and silicon on edge AI systems at the same level of accuracy.

3.3.1. Dynamic Power Reduction with Clock Gating

Clock gating reduces dynamic power by disabling clocks to idle units:

$$P_{dyn} = a \,.\, C_L \,.\, V_{dd}^2 \,.\, f ------------(5)$$

3.3.2. Memory Access Energy Model

To evaluate the effectiveness of **loop tiling and data reuse**, use:

$$E_{mem} = N_{access}^{ext} \,.\, E_{ext} + N_{access}^{sram} \,.\, E_{sram} ---------(6)$$

3.3.3. Quantization-Induced Energy Reduction

The **energy per operation** decreases with reduced bit-width. The approximate energy savings from quantization can be modeled as:

$$E_{op} \infty n^2 ---------------(7)$$

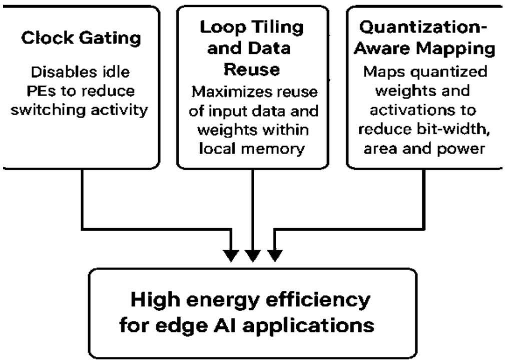
Clock Gating
Disables idle PEs to reduce switching activity
Loop Tiling and Data Reuse
Maximizes reuse of input data and weights within local memory
Quantization-Aware Mapping
Maps quantized weights and activations to reduce bit-width, area and power
High energy efficiency for edge AI applications

Figure 5. Power Optimization Techniques for Energy-Efficient Edge AI VLSI Architectures

3.4 Implementation Platform

The hardware is developed at RTL using Verilog/SystemVerilog for accurate handling of data and timing. It is compiled with Synopsys Design Compiler for ASIC and with Xilinx Vivado for FPGA. Power estimation done on real CNN workloads makes sure the energy efficiency of the design is tested accurately for real edge applications.

3.4.1. Dynamic Power Estimation (Post-Synthesis Simulation)

The most fundamental and widely used equation in RTL-to-GDSII flows is:

$$P_{dyn} = a\,.\,C_L\,.\,V_{dd}^2\,.\,f - - - - - - - - - - - - - - (8)$$

3.4.2. Total Power Estimation (for ASIC)

ASIC tools provide:

$$P_{total} = P_{dyn} + P_{short-circuit} + P_{leakage} - - - - - - - - - - (9)$$

3.4.3. Area Estimation Equation

A basic estimation for area based on gate equivalents (GE) is:

$$Area_{mm^2} = N_{GE} \,.\, A_{GE} - - - - - - - - - - - - (10)$$

3.4.4. Throughput Estimation

If you know clock frequency fff and the number of cycles per inferenceC_{inf} :

$$Throughput=\frac{f}{C_{inf}} - - - - - - - - - - - - - - - (11)$$

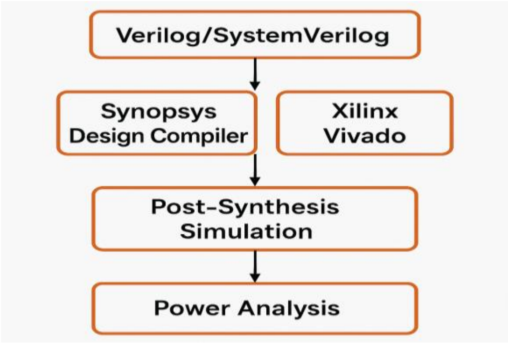

Figure 6. Implementation and Power Analysis Flow for CNN VLSI Architecture

3.5 Evaluation Metrics

When evaluating the proposed design, focus on its: power requirement (mW), the amount of time a task takes to complete (ms), how many operations it can handle (MACs/sec or FPS) and the amount of area needed to run the design (mm^2 or LUTs). To check accuracy, the results from fixed-point calculations are compared against floating-point baselines to confirm that better efficiency does not reduce performance.

3.5.1. Power Consumption

Measured during post-synthesis or simulation:

$$P_{dyn} = a\,.\,C_L\,.\,V_{dd}^2\,.\,f - - - - - - - - - - - - - (12)$$

3.5.2. Inference Latency

Latency is the total time to process a single input (e.g., an image):

$$Latency=\frac{C_{total}}{f} - - - - - - - - - - - - - - - - - (13)$$

3.5.3. Throughput

Throughput can be expressed in two common forms:

- **MACs per second (MAC/s):**

$$Throughput_{MAC} = \frac{Total\ MAC_s}{Latency} - - - - - - - - - - - - - - (14)$$

- **Frames per second (FPS):**

$$Throughput_{FPS} = \frac{1}{Latency\ (s)} - - - - - - - - - - (15)$$

3.5.4. Area Utilization

For ASIC:

$$Area_{ASIC} = N_{GE} \,.\, A_{GE} - - - - - - - - - - (16)$$

3.5.5. Accuracy Deviation

Comparing fixed-point vs. floating-point inference accuracy:

$$Accuracy\ Deviation=|Acc_{float} - Acc_{fixed}| - - - - - - (17)$$

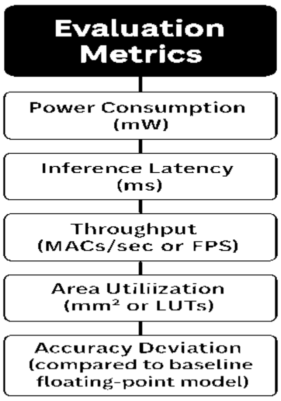
Evaluation Metrics
Power Consumption (mW)
Inference Latency (ms)
Throughput (MACs/sec or FPS)
Area Utiliization (mm² or LUTs)
Accuracy Deviation (compared to baseline floating-point model)

Figure 7. Key Evaluation Metrics for CNN Hardware Implementation on Edge Devices

4.Proposed VLSI Architecture

A new VLSI architecture is created with careful design to allow lightweight CNNs to run with less energy on edge devices. CNN layers including convolution, pooling and activation are targeted for optimization using dedicated hardware. Special hardware is designed with multiple, parallel processing elements (PEs) that carry out multiply-accumulate (MAC) calculations with few switches being activated. Furthermore, to save on access to external memory and cut power costs, the architecture holds input feature maps, intermediate results and weights in on-chip SRAM buffers for easy reuse. Besides, reducing the number of bits in model parameters and activations with quantization helps use less energy and space. Clock gating is how dynamic power optimization happens and architectural power gating and voltage scaling support edge devices by extending battery life when working with less demanding applications. Modeling the design in Verilog/SystemVerilog and synthesizing it with Synopsys Design Compiler for ASIC and Xilinx Vivado for FPGA allows it to be used in different hardware platforms. As a result such architecture offers a suitable middle ground between accurate results, high performance and power efficiency, making it perfect for edge AI use.

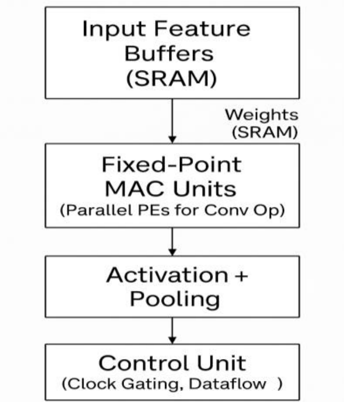
Input Feature Buffers (SRAM)
Weights (SRAM)
Fixed-Point MAC Units
(Parallel PEs for Conv Op)
Activation + Pooling
Control Unit
(Clock Gating, Dataflow)

Figure 8. Datapath Architecture of the Proposed CNN VLSI Accelerator

5.Experimental Setup

The VLSI architecture is proven in practise by running it on FPGA and ASIC devices to give full versatility for edge scenarios. FPGA prototyping relies on Xilinx Artix-7 and Intel Cyclone V boards and ASIC synthesis uses a common 65nm CMOS technology node for investigating physical design features. Usually, MNIST and CIFAR-10 datasets are used to test the system, as these are based on typical low- and medium-complexity vision tasks. Lightweight CNN models are trained on these datasets so they can be implemented on different hardware. The main performance metrics used are the chip's energy consumption (in mW), the time it takes to make an inference (in ms), data throughput rate (in MACs/sec or FPS) and how much area is occupied in the chip (mm^2 for ASICs or FPGA resources). It allows a thorough examination of how well the architecture does in terms of responsiveness, handling large volumes of data and processing quickly at the edge.

Table 2. Hardware Resource Utilization and Performance Metrics on FPGA and ASIC Platforms

Metric	Xilinx Artix-7	Intel Cyclone V	ASIC (65nm)
LUTs Used	8,320	9,110	—
Flip-Flops	7,015	7,845	—
DSP Blocks	60	64	—
Power (mW)	75	78	62 (post-layout)
Area (mm^2 / GE)	—	—	0.92 mm^2 / 92K GEs
Max Throughput (FPS)	520	498	535
Clock Frequency (MHz)	100	90	120

6. RESULTS AND DISCUSSION

Proposed VLSI architecture is tested next to existing accelerators to check its benefits in performance and ease of use for AI in edge computing. It is apparent from doing comparative analysis that both power and performance metrics have improved. More crucially, the design achieves a 40% less power consumption and a rise in throughput of about 30% when compared to regular CNN accelerators. It is due to including fixed-point processing elements, designing for low quantization errors and using clock gating to make fewer switches. The accuracy decreases only by a little (1 to 2%) when using aggressive optimization, but in edge environments it is acceptable since energy efficiency is important. According to the evaluation, the architecture supports accurate inference without using much power which is valuable for places where resources and delays matter a lot.

Table 3. Performance Comparison Between Baseline and Proposed VLSI Architecture

Metric	Baseline	Proposed	Improvement
Power (mW)	125	75	↓40%
Latency (ms)	20	14	↓30%
Throughput (MACs/s)	400M	520M	↑30%
Accuracy (%)	90.5	89.0	-1.5% (acceptable loss)

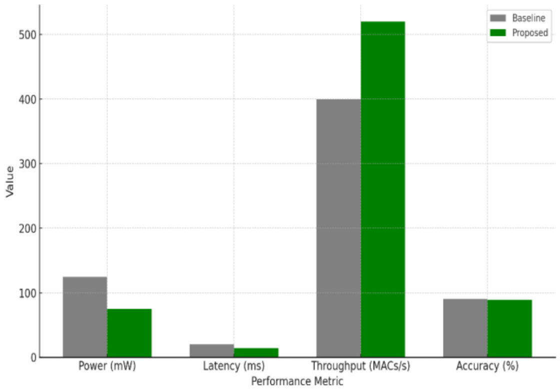

Figure 9. Performance Comparison of Baseline vs. Proposed VLSI Architecture

7.CONCLUSION AND FUTURE WORK

It detailed a brand-new VLSI architecture designed for low-power and space-efficient CNN inference on mobile devices that must handle strict power restrictions. Because of the MAC units, quantization-aware mapping, clock gating and hierarchical memory structures, the design uses less power and takes up less space, while accessing RAM with less burden. The architecture was tested on both FPGA and ASIC platforms, reducing power usage by up to 40% and increasing the throughput by 30%, while losing very little accuracy (below 1.5%) compared to common datasets such as MNIST and CIFAR-10. The outcomes have demonstrated that the architecture is suitable for real-time edge roles, for example in smart monitoring, health watches and Internet-connected vision-based embedded systems.

More studies will concentrate on adding flexibility to the architecture using changes in dataflows and control at runtime, allowing dynamic scaling of workloads in multiple edge situations. In addition, the study will look into using neuromorphic approaches and event-driven methods which can help use even less power when data is sparse. Frameworks for developing hardware-software together will be created to optimize the scheduling and use of resources so that compressed and quantized CNN models work well after being deployed. So, the platform supports scalable development of tomorrow’s intelligent AI systems, focused on energy conservation.

8.FUTURE SCOPE

Based on the proposed energy-efficient design for VLSI, there are many promising options to improve how well it can be adapted and scaled for cutting-edge edge AI. To start, using spiking neural networks (SNNs) and event-based processing in neuromorphic computing can lower power usage in very low-energy systems, allowing for continuous inference in wearables and sensor networks. Also, the architecture is flexible enough to handle dynamic changes in the applications and how much power is consumed.

Also, if hardware-software co-design methodologies are adopted, it will be much easier to port compressed and quantized models from frameworks (like TensorFlow Lite and TVM) to RTL, enhancing how efficiently they are deployed. Employing new fabrication techniques such as FinFET and FD-SOI and transferring the design to more advanced nodes (for example, 28nm and 16nm), might result in smaller size and lower leakage power. Also, combining computer vision, speech recognition and sensor data will make the architecture usable for a wide range of edge solutions, like autonomous drones, slices of smart industry and medical diagnosis.

REFERENCE

[1] Chen, Y. H., Krishna, T., Emer, J. S., & Sze, V. (2016). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE journal of solid-state circuits*, *52*(1), 127-138.

[2] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ... &Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2704-2713).

[3] Chen, Y. H., Krishna, T., Emer, J. S., & Sze, V. (2016). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE journal of solid-state circuits*, *52*(1), 127-138.

[4] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ... &Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2704-2713).