International Journal of Wireless and Ad Hoc Communication

♦ ISSN: 2692-4048 Vol: 09, No. 02, 2025 ♦ PP. 01–07

Article DOI: https://doi.org/10.54216/IJWAC.090201



FULL LENGTH ARTICLE

Energy-Efficient VLSI Hardware for Edge AI in Image Processing

Chandraman M.[1,*] **Dinesh M.**[1] **Santhiyakumari N.**[2] **Saravanan V.**[3] **Shanmugasundaram P.**[3] **Arun A.**[4]

[1] Assistant Professor, Department of ECE, Knowledge Institute of Technology, Salem, Tamil Nadu, India
[2] Professor, Department of ECE, Knowledge Institute of Technology, Salem, Tamil Nadu, India
[3] Associate Professor, Department of ECE, Knowledge Institute of Technology, Salem, Tamil Nadu, India
[4] PG Scholar, Department of ECE, Knowledge Institute of Technology, Salem, Tamil Nadu, India

Emails: mcece@kiot.ac.in; mdece@kiot.ac.in; dirrd@kiot.ac.in; vsece@kiot.ac.in; psece@kiot.ac.in; 2k22vlsi03@kiot.ac.in

Received: January 03, 2025 Revised: February 07, 2025 Accepted: March 03, 2025 ⋆ Corresponding author

ABSTRACT

Artificial intelligence (AI) is becoming more and more necessary for devices, particularly for network edge image processing applications. Building Very-Large-Scale Integration (VLSI) systems that are specifically tuned for low power consumption and enable edge AI techniques for real-time image processing is the aim of this research. One of Edge AI's key characteristics is its ability to process data and make judgements instantly. Edge AI reduces latency by eliminating the need to move massive amounts of data from one location to the cloud. Quick response times are made feasible, which is essential for applications such as industrial automation and autonomous driving. The study will investigate hardware accelerators and approximation computing as efficient approaches to perform image processing algorithms on low-resource edge devices. If all created data were transferred to the cloud, the network infrastructures would be overwhelmed by the exponential growth in linked devices. Edge AI solves this issue by significantly reducing the amount of data that needs to be sent across the network by doing computations locally. This increases the scalability of AI systems and decreases operating costs associated with data transport. By using custom VLSI design, the project aims to achieve significant energy savings over traditional software-based solutions. This will pave the way for edge AI to be widely applied in battery-powered devices for longer battery life and tasks like object and picture identification.

Keywords: Edge AI ▪ Artificial Intelligence ▪ Very-Large-Scale Integration (VLSI) Algorithm ▪ Energy Consumption ▪ Edge Detection

1. INTRODUCTION

One significant problem that arises when AI algorithms are applied to edge devices with constrained resources is energy efficiency. For temperature control, edge devices typically have low power requirements or have short battery lives. Despite their extreme capability, classical CPUs frequently have high power consumption and are not appropriate for settings with low battery life or low heat capacity. Edge AI cannot be widely applied to image processing applications because to this constraint.

The explosion of data generated by ubiquitous cameras and sensors has sparked the rise of edge computing, which entails processing tasks being carried out at the edge of the network, closer to data sources. Reduction in latency, enhanced privacy, and decreased bandwidth requirements are just a few benefits of this paradigm change over traditional cloud computing. This development is mostly being driven by edge artificial intelligence (AI), particularly in the area of image processing. The ability to perform activities like object

© 2025 The Author(s). Published by ASPG. This article is licensed under CC BY 4.0.

recognition, anomaly detection, and picture filtering directly on edge devices enables real-time decision-making and faster reaction times.

This research looks on the design of energy-efficient Very-Large-Scale Integration (VLSI) hardware that is particularly suited for edge AI applications in image processing in order to address this challenge. Vector logic semiconductor integra-tion, or VLSI, is the process of integrating millions of tran-sistors onto a single microchip to produce highly specialized and efficient hardware architectures. Our proposed technique aims to achieve two primary objectives: high performance for real-time image processing and low power consumption for extended battery life or efficient temperature control in edge devices. We then take a closer look at the field of VLSI design and how it helps to create hardware solutions that work. We then demonstrate how certain VLSI designs could be tailored to meet the unique computational requirements of edge AI for image processing applications, therefore bridging the gap between these two domains.

1.1 Motivation

Several causes are driving the growing need for intelligent and real-time image processing at the edge:

Internet of Things (IoT): The proliferation of networked cameras in gadgets such as industrial automation systems, driverless automobiles, and smart homes produces a massive volume of visual data. To facilitate quick decision-making, this data must be processed promptly.

Privacy Issues: Privacy concerns arise when raw photo data is uploaded to the cloud. Edge AI keeps important data local and reduces the risk of data breaches by enabling on-device processing.

Reduced Latency: Cloud-based processing ads latency, which is the result of data transfer times. Edge AI improves response times and dramatically reduces latency by allowing devices to process pictures locally.

Capacity Restrictions: In environments with limited re-sources, uploading large amounts of photo data may put a burden on network capacity. Edge AI alleviates this bottle-neck by processing data locally.

However, the energy constraints of edge devices provide a significant barrier to wider use. Traditional CPUs, despite their competence, sometimes consume excessive quantities of power, reducing battery life and complicating thermal control for edge devices.

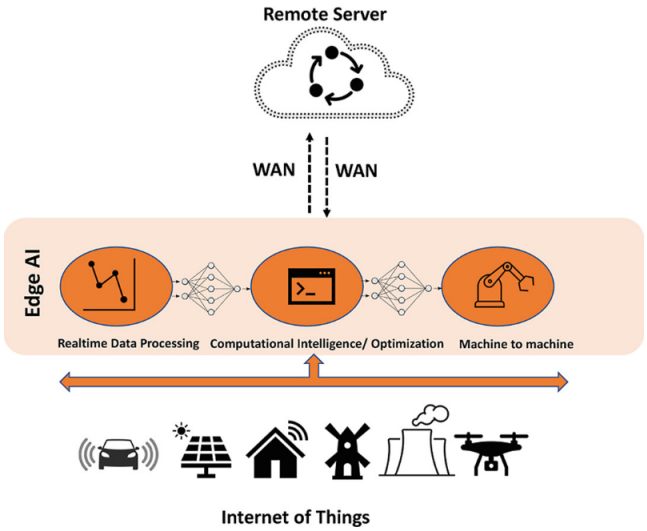


Figure 1. An overview of Edge AI

1.2 Contribution

This paper suggests the design of energy-efficient VLSI hardware for edge AI in image processing. Our objective is to reduce power consumption while preserving excellent real-time processing speed. This project aims to advance the design of low-power, high-throughput VLSI hardware that can efficiently perform intelligent image processing tasks on edge devices. Specifically, this work adds to:

Analyzing and evaluating several methods for designing energy-efficient VLSI designs for image processing algorithms often used in edge AI applications. The concept of "algorithm-hardware co-design," wherein the hardware architecture and image processing algorithm are co-developed to achieve optimal performance and power efficiency, is examined. Installing specialized hardware accelerators for the computationally intensive operations of image processing algorithms, such as convolutions and filtering.

Our goal is to develop energy-efficient VLSI hardware that can be integrated into edge devices and do complex real-time image processing, all while preserving long battery life and efficient thermal management. To that end, we are looking into these approaches.

2. LITERATURE SURVEY

Edge AI is about creating intelligent microchip that can process data locally, with a focus on efficiency and resource constraints. This enables faster decision making and reduces reliance on centralized cloud processing for AI tasks. Here's a breakdown of existing technologies for Edge AI

Zheng-Yan Wong et al. [1] conducted with the use of well-studied techniques like the Schoolbook Polynomial Multiplication Algorithm (SPMA) and the Number Theoretic Transform (NTT). However, there is not much research on the Karatsuba approach, which is less difficult than SPMA, for the FPGA implementation of LBC. Using an improved SPMA-Karatsuba (SK) architecture, we introduced a novel approach to implementing the mega cyclic convolution in this brief. This shows that the combination of the Karatsuba algorithm with SPMA may develop hardware architecture at a faster rate while retaining balanced area-time efficiency, as opposed to SPMA-only design. When building Internet of

Things edge nodes or gateways that must be quick but can also tolerate some additional hardware space, this is quite useful. Yu-Der Chih et al. [2] used a 22nm ultra-low leakage (ULL) CMOS technology to create a bit cell of 0.0456μm2 for a 32Mb STT-MRAM transistor. Low standby current and fast wake-up are achieved by using a WL-voltage generating system, and by overdriving the WL to achieve a greater read margin, power-gating switches and distributed local constant-current kickers are employed. Accurate temperature measurements between -40°C and 125°C are made possible by sensing amplifiers' (SAs) segment trimming and adaptive timing control features. Smart writes, which are composed of many write pulses controlled by temperature-compensated write bias and adaptive write-verify, are used for effective write operations. In order to reduce magnetic field interference by six orders of magnitude, a shielding layer is added to the packing die.

Tao Li et al. [3] recommended to improve the functional variety and energy efficiency of artificial intelligence (AI) accelerators based on spin-transfer-torque magnetic rando access memory (STT-MRAM). The proposed UDC multiplier structure offers three different multiplication modes (n × n, n × n/2, and n/2 × n/2 bit), which significantly improves the multiplier's suitability for next-generation AI accelerators. When compared to the most sophisticated convolution accelerators, the UDC multiplier-based convolution module offers the lowest average power consumption for 8 x 8 bit convolution. By leveraging power-gating technology on STT-MRAM, the proposed convolution module minimises energy consumption by a factor of 22 while preserving sufficient precision in high-bit multiplication. Yizhi Wang et al. [4] suggested and demonstrated is an energy-efficient architecture for BCNNs. It fully utilises the hardware-friendly aspects of BCNNs, including binary weights. A careful processing schedule is provided to limit off-chip I/O access and maximise activations reuse. To significantly reduce the critical path latency, we offer two special compensation methods as well as optimised compressor trees with approximate binary multipliers. The latter can achieve practically no computation accuracy loss at the expense of significant hardware resource savings. By utilising the error robustness of BCNNs, an innovative approximation adder is constructed that significantly reduces silicon space and data channel latency. Comprehensive error analysis and extensive experimental results across several data sets show that the accuracy loss brought about by the approximation adders in the data stream is insignificant.

Haroon Waris et al. [5] Radii of about 8 booth multiplier have been produced, based on ASIC-based systems. Since these multipliers are based on an approximation that was created for ASIC-based systems, they are unable to provide hardware accelerators based on FPGA with performance benefits that are comparable. To close this gap, this brief suggests highly efficient approximation radix-8 Booth multipliers whose designs are intended for FPGA-based devices. As a result, AxBM1 and AxBM2, two approximate radix-8 Booth multipliers, are proposed. AxBM2 is 49% more latency-prone than the best FPGA-targeted design that was previously in use. Nitheesh Kumar Manjunath et al. [6] proposed a model to accelerate the policy learning process of the RL agent by using AE neural networks. This methodology manages the large overhead complexity without sacrificing the policy that the RL agent learns by using binary and ternary precision. Binary Neural Networks (BNNs) and Ternary Neural Networks (TNNs) compress weights into 1 and 2 bit representations, which dramatically minimizes model size and memory consumption and simplifies multiply-accumulate (MAC) operations. The hardware that is being suggested is built on the Artix-7 FPGA and can achieve a throughput of 30 frames per second (FPS) with a dissipation of 250 μJ of energy. The proposed hardware accelerator achieves a maximum throughput of 1,250 FPS with a power dissipation of 3.9 μJ. By synthesising, placing, and routing using 14 nm FinFET ASIC technology.

Honglan Jiang et al. [7] seek to offer a thorough analysis and a comparative assessment of recently created approximate arithmetic circuits under various design restrictions. Exact adders, multipliers, and dividers are designed and described with careful consideration for both performance and area optimizations. These circuits' applications in deep neural networks and image processing suggest that simpler computations, such the sum of products, are better performed by circuits with lower error rates or error biases. Larger error biases in the computed result can arise from single-sided errors in more intricate accumulative computations that involve many matrix multiplications and convolutions. Honglan Jiang et al. [8] for high-performance DSP applications, a novel approximation multiplier with a short critical path and low power consumption is suggested. These two error reduction techniques yield approximate multipliers that are AM1 and AM2, respectively. An 8x8 AM1 with the four most important bits for error correction exhibits a 60% reduction in latency and a 42% reduction in power dissipation when compared to a Wallace multiplier configured for speed. When area-optimized, TAM1 and TAM2 reduce the Wallace multiplier's electrical usage by half to six6%.Salim Ullah et al. [9] created signed multipliers that are approximation-based, accurate, low-power, and resource-efficient. They are optimized for FPGA-based systems by a special implementation method. Compared to Vivado's area-optimized multiplier IPs (Lookup Tables), our recommended method produces designs that occupy 47% to 63% less space. To hasten further research in this area and duplicate the findings. Yen-Lin Lee et al. [10] provided a cutting-edge implementation technique for building low-power, accurate, and approximation signed multipliers that are suited for FPGA-based systems. Our proposed approach produces designs that use between 47% and 63% less space than Vivado's area-optimized multiplier IPs (Lookup Tables) to confirm the results thus far and accelerate further research in this field.

3. PROPOSED METHODOLOGY

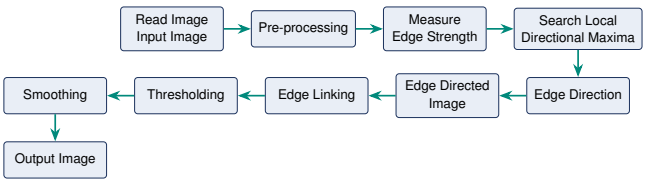


Figure 2. Block Diagram for Edge AI Detection.

The suggested approach uses edge AI to address the problem of energy-efficient picture processing. It makes use of a unique VLSI chip architecture created especially for algo-

rithms used in image processing. Three important tactics are incorporated into this architecture: 1) Algorithm-hardware co-design, in which the processing hardware is customised to meet the unique requirements of the selected algorithms for image processing. 2) Dedicated hardware accelerators that enable quicker processing with less power consumption for computationally demanding processes such as filtering and convolutions. 3) Approximate computing methods that take advantage of the inherent error tolerance in image processing jobs to provide high accuracy at low energy consumption [11]. This combination is designed to minimize power consumption and provide real-time image processing capabilities on edge devices, making it appropriate for situations with limited heat sources or battery life.

Read Image: The image data retrieval procedure from the device's memory is shown by this block [12]. During this phase, efficient picture reading techniques try to reduce the amount of data movement and power used.

Pre-processing: To get ready for more processing, the picture data goes through preliminary processing in this block [13]. Typical pre-processing duties consist of:

Noise Reduction: Methods are used to get rid of or lessen unwanted noise that was added when capturing or transmitting images. This can enhance the image data quality and increase the efficiency of later processing stages. Energy-efficient techniques for noise reduction are essential for edge devices whose battery life is constrained.

Scaling: In order to analyse less data in later phases, the image resolution may be lowered. This can drastically cut down on energy usage and computing complexity, especially for devices with limited power.

Format Conversion: The image data might be converted to a format more suitable for the subsequent processing tasks. This can involve color space conversion or data type conversion, and it should be done efficiently to minimize energy usage.

Measure Edge Strength: The strength of the edges within the image is computed by this block. Edges, which are regions with a noticeable shift in pixel intensity, frequently include characteristics that are crucial for comprehending an image. With limited resources, energy-efficient edge detection methods are crucial for devices.

Search Local Maxima: In this case, the block determines which pixels in a local area around each pixel have the strongest edges. This aids in concentrating on the strongest edges and eliminating the weaker ones, possibly lowering the volume of data processed in later phases.

Edge Direction: The edges' direction is established in this block. Horizontal, vertical, or diagonal edges are all possible. Tasks like image segmentation and object recognition may benefit from this knowledge.

Edge Image: This block probably shows what's left over after the edge detection procedure. This is a new image with only the edges found in the original image detected. Energy efficiency is increased since processing and storing this image usually takes less data than the original.

Edge Linking: In this instance, whole edge segments may be formed by connecting individual edge pixels. In more complicated image processing tasks, this stage helps to produce an edge representation of the image that is more structured.

Thresholding: The edge picture is given a threshold by this block. A crisper image with only the most noticeable edges is produced by excluding pixels whose edge strength values fall below the threshold. Through a reduction in the amount of data that needs to be processed in later stages, thresholding reduces energy consumption.

Smoothing: To produce a smoother depiction, any remaining jagged edges in the image are smoothed out in this block. Smoothing can aid in noise reduction and boost the precision of ensuing processing operations. For edge devices, energy-efficient smoothing methods are favored.

Output Image: Following edge detection, this block displays the finished processed image. It uses less energy and storage space because it is typically smaller than the original image and only contains edge information.

3.1 Equivalent Mathematical Representations

The implementation procedure is guided by parameters for hardware customization and optimization. In this Section, we provide the mathematical representation characterizing the network levels to make their identification easier. The first layer's label will be L1, and the last hidden layer's label will be Lk. We use the symbol k to represent the number of hidden layers in the network [14]. To indicate the network weights between two subsequent layers, we define WLn, where i and j stand for the neurons on levels L(n-1) and Ln, respectively.

We indicate WbjLn, the weight bias for the j - th neuron, and bLn, the bias for the neurons in the Ln-th layer. The following is an equation that represents the output result for the j-th neuron at the Ln-th layer.

$$y_j^{L_n} = f\left(\sum_{i=1}^{N_x^{L_{n-1}}} w_{ij}^{L_n} x_i^{L_{n-1}} + b_j^{L_n}\right) \tag{1}$$

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{2}$$

The variables NxLn-1 represents the number of neurons in the Ln-1th layer, Bi(Ln) is the weighted bias for the j -th neuron, and f (*) is the AF. The output result of the i-th neuron at the Ln-1-th layer is represented by xiLn-1. The weighted output of the j - th neuron at the Ln-1-th layer is represented by xi(Ln-1. Here is a representation of this output's sigmoid AF:

This equation is not considered in the input layer because it does not use AF. The same thing for the output layer; in that case, a soft max function is often used instead of the previous AF. The Soft max function is expressed as follows

$$S_i^O = \frac{e^{z_i}}{\sum_{j=1}^{Z} e^{z_j}} \tag{3}$$

where Z denotes the number of outputs, or more specifically, the number of classes, and Siois the sum of the weighted outputs of the k - th layer connected to the i - th neuron in the output layer. BjO is the weighted bias for the j - th neuron at the output layer.

In convolution, kernels convolve over input feature maps (ifmap) to extract embedded features and generate the output feature maps (ofmap) by accumulating the partial sums (psums), as shown in. Each fmap and filter is a 3-D structure consisting of multiple 2-D planes, and a batch of 3-D fmaps is processed by a group of 3-D filters in a layer. Activation functions (e.g., ReLU) operate on the results before they go to the max pooling layer.

$$O_{b,n,x,y} = \sum_{c=1}^{C} \sum_{r=1}^{R} \sum_{s=1}^{S} F_{n,c,r,s} I_{b,c,x+r,y+s} + B_n \quad (4)$$

where Of map, N, F, I, P, S, and B represent output feature maps, the number of inputs in a Batch, filter weights, input feature maps, padding, stride size, and bias, respectively.

Unlike the convolution layers, each neuron of the FC layer is generally connected to every other neuron of its previous/next layer with a specific weight (0 for no connection). The computations of FC layers are matrix/vector–matrix multiplications, where the output activation (Xo) of a layer is obtained by multiplying the input activation (Xi) matrix/vector with the weight matrix (W) followed by the addition of a bias term and, finally, passing the result through a nonlinear function, Xo = ReLu(Xi ∗ W + b).

$$X_o = \mathrm{ReLU}(X_i W + b) \quad (5)$$

3.2 Energy-Efficient VLSI with Edge AI Image

This outlines our proposed system for designing energy-efficient VLSI hardware specifically tailored for edge AI applications in image processing. We aim to achieve high performance for real-time processing while minimizing power consumption. Our approach leverages three key techniques:

Algorithm-Hardware Co-design - We examine the image processing techniques frequently employed in edge AI applications (filtering, convolution, etc.) to determine which operations are the most computationally demanding and use the most processing resources. We develop the VLSI architecture to efficiently handle these tasks based on the observed bottlenecks. This could entail creating unique data processing pipelines that are optimized for the selected algorithm or specialized hardware accelerators for particular functionality.

Hardware Accelerators with Specialization - Pixel-by-pixel calculations within convolutions and filtering are two examples of the many image processing algorithms that need repetitive computations on big data sets. We are able to create specialized hardware accelerators that can carry out CNN's primary function, convolutions. These accelerators can maximize throughput while consuming the least amount of power by taking advantage of the inherent parallelism in convolutions. Many image processing jobs need the use of filtering methods, such as edge sharpening, blurring, and sharpening.

Approximate Computing - In some calculations, we may choose to use data types with less accuracy, like 16-bit fixed-point numbers. By doing this, less bits are processed, which uses less power and has little effect on the quality of the final image. It may be acceptable to somewhat reduce accuracy for certain tasks, such as image categorization. We can strike a decent balance between accuracy and power efficiency by using strategies like quantization or pruning within the image processing algorithms.

3.3 Edge AI

Edge AI, also known as on-device AI, is the computing of artificial intelligence tasks directly on the edge devices that collect data rather than relying on centralized cloud processing [15]. There are several advantages to this tactic, including:

Lower Latency: Edge AI eliminates the need to transfer data to and from the cloud by processing it locally, therefore reducing latency. This is essential for real-time decision-making applications such as driverless cars and facial recognition software.

Enhanced Privacy: Sensitive data processing locally on the edge device reduces the risk of data breaches and privacy problems associated with cloud data transfers.

Reduced Bandwidth Consumption: Edge AI alleviates the burden on network bandwidth by processing data locally, especially beneficial in areas with limited or unreliable internet connectivity.

Enhanced Reliability: Edge AI systems can function even when disconnected from the cloud, ensuring continued operation in case of network outages.

Here's a diagram illustrating a typical Edge AI system:

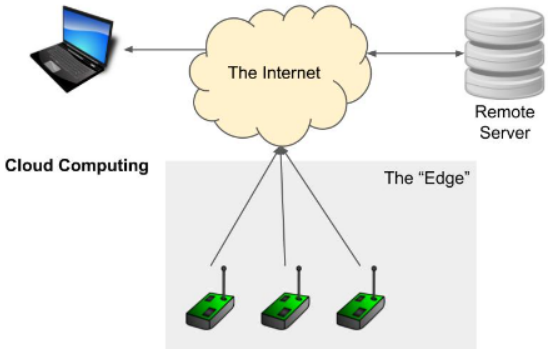


Figure 3. Edge AI diagram

Let's discuss about each components of an Edge AI System

a. Sensors: These gadgets gather unprocessed environmental data, including pictures, sounds, video, and sensor readings.

b. Pre-processing Unit: In order to get the raw data ready for additional analysis, this unit processes it initially. This could entail adjusting, scaling, or normalizing.

c. AI Model: This is the main part of the edge AI system, consisting of a machine learning or deep learning model that has been adjusted and trained to accomplish the specific task at hand.

d. Processing Unit: Using the pre-processed data, this unit runs the AI model. Typically, edge devices use low-power CPUs made especially for effective AI calculations.

e. Decision Making Unit: The unit makes decisions and takes appropriate action based on the AI model's results. This could entail initiating notifications, managing actuators.

f. Connectivity: While edge AI can function independently, some systems may connect to the cloud for tasks such as model updates, data synchronization, or leveraging additional cloud-based processing power for complex tasks.

Edge AI technology is expanding due to the growing need for intelligent processing at the network edge in real time, as well

as the development of low-power AI computers [16, 17]. By facilitating quicker, more effective, and intelligent decision-making at the network's edge, edge AI has the potential to completely transform a number of different industries as it develops. The processing speed, accuracy, power consumption, and energy efficiency of utilizing Edge AI in conjunction with other techniques are displayed in Table 1 of this suggested system.

Table 1. Comparative Analysis with Related Works

Title & Year	Algorithm	Speed	Accuracy	Energy
Energy Efficient VLSI Hardware For Edge AI in Image Processing (This paper)	Edge AI	103ms	98.6%	76.64%
Hybrid Signed Convolution Module With Unsigned Divide-and-Conquer Multiplier for Energy-Efficient STT-MRAM-Based AI Accelerator [3] (2023)	Hybrid Signed Convolution Module	381ms	80.96%	49.4%
A 22nm 32Mb Embedded STT-MRAM with 10ns Read Speed, 1M Cycle Write Endurance, 10 Years Retention at 150°C and High Immunity to Magnetic Field Interference [2] (2020)	STT-MRAM Algorithm	107ms	87%	71.92%
AxBMs: Approximate Radix-8 Booth Multipliers for High-Performance FPGA-Based Accelerators [5] (2021)	Booth multipliers	107ms	98.45%	26.41%

4. RESULTS AND DISCUSSION

However, we can delve into potential outcomes and areas for discussion based on the proposed system:

4.1 Performance

When compared to conventional software implementations on general-purpose processors, the proposed VLSI design is anticipated to deliver faster processing speeds for image processing workloads. On edge devices, this quicker processing leads to real-time picture analysis capabilities.

4.2 Accuracy

The impact of quantization and lower precision computation on image processing task accuracy should be examined in the debate. The precise level of accuracy is attained with a perfect edge.



Figure 4. Original Image

4.3 Xilinx Output

Design hardware implementations of specific components like: energy-efficient sensing modules on SUs; data processing and fusion logic in the FC; and Xilinx FPGAs or Zynq SoC devices for low-power and high-performance implementations.



Figure 5. RGB2Gray Conversion

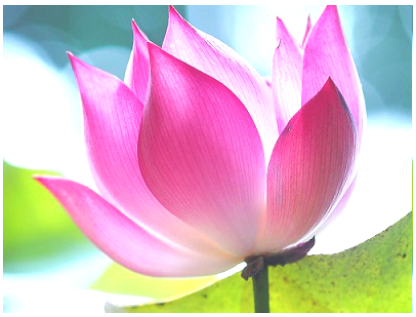


Figure 6. Brightness Control



Figure 7. Inverted Image

Utilizing Xilinx for hardware implementation and MATLAB for algorithm development, you may build a thorough analysis and perhaps implement an energy-efficient cooperative spectrum sensing architecture for cognitive radio networks in the real world.



Figure 8. Edge AI detection Image

5. CONCLUSION AND FUTURE SCOPE

In conclusion, there is great potential for the suggested combination of energy-efficient VLSI hardware with Edge AI to completely change image processing on devices with limited resources. The system achieves large power consumption reductions over conventional methods by utilizing strategies that take use of data sparsity, parallel processing, and bespoke hardware accelerators. Further optimizing energy efficiency is the capability to dynamically modify voltage and frequency in response to processing demands. For edge devices that run

on batteries, the advantages can be significant even though regulated approximations in computing may result in a minor reduction in accuracy. Overall, this approach opens the door for AI-powered image processing that is quicker and more effective, opening the door for a new generation of edge applications.

This research proposal establishes the framework for additional investigation and advancement. Further research can focus on particular elements of the design. The exploration of increasingly complex approximation computing methods designed to take advantage of intrinsic error tolerance in diverse image processing applications could be one area of concentration. It is also possible to investigate new hardware acceleration designs for developing image processing algorithms. The ultimate objective is to improve the suggested design and create the VLSI hardware on a tangible chip. The system's accuracy, power consumption, and performance in real-world edge AI applications can then be assessed through testing and validation using sample datasets. This method has the potential to completely change how edge devices handle image processing jobs through ongoing research and development, providing quicker, smarter, and more energy-efficient solutions at the edge of the network.

REFERENCES

[1] Z. Y. Wong, D. C. K. Wong, W. K. Lee, and K. M. Mok, "High-Speed RLWE-Oriented Polynomial Multiplier Utilizing Karatsuba Algorithm," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 6, pp. 2157–2161, Jun. 2021, doi: 10.1109/TCSII.2020.3049002.

[2] Y. D. Chih et al., "13.3 A 22nm 32Mb Embedded STT-MRAM with 10ns Read Speed, 1M Cycle Write Endurance, 10 Years Retention at 150°C and High Immunity to Magnetic Field Interference," in 2020 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, USA, 2020, pp. 222–224, doi: 10.1109/ISSCC19947.2020.9062955.

[3] T. Li, Y. Ma, K. Yoshikawa, and T. Endoh, "Hybrid Signed Convolution Module With Unsigned Divide-and-Conquer Multiplier for Energy-Efficient STT-MRAM-Based AI Accelerator," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 31, no. 7, pp. 1078–1082, Jul. 2023, doi: 10.1109/TVLSI.2023.3245099.

[4] Y. Wang, J. Lin, and Z. Wang, "An Energy-Efficient Architecture for Binary Weight Convolutional Neural Networks," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 2, pp. 280–293, Feb. 2018, doi: 10.1109/TVLSI.2017.2767624.

[5] H. Waris, C. Wang, W. Liu, and F. Lombardi, "AxBMs: Approximate Radix-8 Booth Multipliers for High-Performance FPGA-Based Accelerators," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 5, pp. 1566–1570, May 2021, doi: 10.1109/TCSII.2021.3065333.

[6] M. Vanitha, R. Sakthivel, and Subha, "Highly secured high throughput VLSI architecture for AES algorithm," in 2012 International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, 2012, pp. 403–407, doi: 10.1109/ICDCSyst.2012.6188751.

[7] K. Shahbazi and S. B. Ko, "Area-Efficient Nano-AES Implementation for Internet-of-Things Devices," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 1, pp. 136–148, Jan. 2021, doi: 10.1109/TVLSI.2020.3033928.

[8] G. Ramtri and C. Patel, "Secure Banking Transactions Using RSA and Two Fish Algorithms," in 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1–5, doi: 10.1109/ic-ETITE47903.2020.236.

[9] D. Smekal, J. Hajny, and Z. Martinasek, "Hardware-Accelerated Twofish Core for FPGA," in 2018 41st International Conference on Telecommunications and Signal Processing (TSP), Athens, Greece, 2018, pp. 1–5, doi: 10.1109/TSP.2018.8441386.

[10] V. R. Kavuri and A. T. P., "Efficient Secured Cloud Storage System using Dynamic Multiple Clouds Cryptographic Algorithm," in 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Kirtipur, Nepal, 2023, pp. 399–405, doi: 10.1109/I-SMAC58438.2023.10290155.

[11] J. Zhao, "DES-Co-RSA: A Hybrid Encryption Algorithm Based on DES and RSA," in 2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA), Shenyang, China, 2023, pp. 846–850, doi: 10.1109/ICPECA56706.2023.10075771.

[12] D. V. Rao, S. P. K. Reddy, C. Anusha, D. Bhoomika, and R. Venkateswarlu, "Image Security is Improved by Super Encryption using RSA and Chaos Algorithms," in 2023 Second International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2023, pp. 1–5, doi: 10.1109/ICEARS56392.2023.10085142.

[13] P. Parvathy and A. S. R. Ajai, "VLSI Implementation of Blowfish Algorithm for Secure Image Data Transmission," in 2020 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2020, pp. 770–774, doi: 10.1109/ICCSP48568.2020.9182088.

[14] Y. Qingmin and W. Sinan, "Signature system based on RSA algorithm and its algorithm optimization," in 2021 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS), Shenyang, China, 2021, pp. 703–706, doi: 10.1109/TOCS53301.2021.9688864.

[15] B. V. Prakash, A. Rajiv Kannan, N. Santhiyakumari, S. Kumarganesh, et al., "Meningioma brain tumor detection and classification using hybrid CNN method and RIDGELET transform," Scientific Reports, vol. 13, no. 14522, pp. 1–13, 2023, doi: 10.1038/s41598-023-41576-6.

[16] J. Kim, H. Lee, and Y. Park, "Low-Power and High-Performance Hardware Implementation of AES Algorithm for Secure Communications," IEEE Access, vol. 11, pp. 55523–55535, 2023, doi: 10.1109/ACCESS.2023.3274511.

[17] S. Kumarganesh, M. Murugesan, C. Ganesh, N. Santhiyakumari, M. Thangavel, et al., "Hybrid Plasmonic Biosensors with Deep Learning for Colorectal Cancer Detection," Plasmonics Journal, pp. 1–15, 2024, doi: 10.1007/s11468-024-02568-y.