SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V13I4P110

Volume 13 Issue 4, 132-146, April 2026

Original Article

Data Path Reallocation and Silicon Footprint Minimization for a 4-Bit ALU Using E-FRIMA and DW-FGCG Techniques

Shylaja V[1, 2], N. Kannan[1], T. Y. Satheesha[3]

[1]*Department of ECE, CMR University, Bangalore, Karnataka, India.*
[2]*Department of ECE, Bangalore Institute of Technology, Bangalore, Karnataka, India.*
[3]*School of Computer Science and Engineering, REVA University, Bengaluru, Karnataka, India.*

[1]*Corresponding Author : shylaja.v@cmr.edu.in*

Received: 09 January 2026 Revised: 10 February 2026 Accepted: 11 March 2026 Published: 30 April 2026

Abstract *- Minimizing silicon footprint and optimizing transistor count are important for improving the power efficiency and performance while designing the 4-bit Arithmetic Logic Unit (ALU). Yet, conventional studies have not performed data path reallocation in ALU, which has increased its resource utilization and power consumption. Thus, this paper presents an efficient Excoffier Functional Redundancy Identification and Merging Algorithm (E-FRIMA) and Drop wave Fine-Grained Clock Gating (Dw-FGCG)-based design of data path reallocation and silicon footprint minimization-aware 4-bit ALU. Initially, for performing ALU operations, the functional requirements are provided. Next, by employing the Zagros Karnaugh Map (ZKM), logic level reduction is done. After constructing the map, data path reallocation is performed based on E-FRIMA to reduce resource utilization and power consumption. Then, by using Return-To-Zero Asynchronous Logic (RTZ-AL), the errors in the reallocated data path are minimized. Now, through Dw-FGCG, the transistor receives a clock signal, thereby enabling the clock signal when needed and disabling it when not in use. Next, by using the Power-Skewed Gaussian Adaptive Network-based Fuzzy Inference System (PS-GANFIS), the defects are identified. After that, based on Chen Bird Baker Crayfish Clock Tree Synthesis Optimization (C2B-2CTSO), the identified defects are mitigated. Eventually, a 4-bit ALU design with minimized silicon footprint is obtained. As per the experimental outcomes, the proposed framework reduced the area to 76%, thus outperforming existing methods.*

Keywords *- Data Path Reallocation, 4-bit Arithmetic Logic Unit (ALU), ALU Area Overhead Reduction, Logic Level Reduction, Zagros Karnaugh Map (ZKM).*

1. Introduction

A fundamental component of any processor is an ALU. It executes arithmetic and logical operations. Usually, a combination of numerous digital circuits merged to perform data processing instructions is termed ALU (i.e., arithmetic and logical operations) in the Central Processing Unit (CPU) of any computer [2]. Usually, an ALU requires one or two operands upon which it works and generates outcomes. Likewise, the timing response of an ALU depends on the complexity and manner in which the circuit is designed [24]. The demand for ALUs that provide high performance while consuming minimal power and occupying a smaller silicon footprint has increased owing to advancements in computing systems [14]. The silicon footprint is the physical area occupied by a circuit on a semiconductor chip. It is an essential factor in Very Large-Scale Integration (VLSI) design because it affects the power consumption, manufacturing costs, and performance. A larger silicon footprint has more material usage, higher production costs, and power dissipation because of the longer interconnects and greater parasitic effects [30].

Recently, modern processors have demanded complex arithmetic operations, including floating-point calculations and vector processing, which increase the transistor count required to execute these functions [26]. The main challenge in designing an ALU is to diminish the transistor count and circuit complexity without compromising performance. Most ALU designs involve complex circuit structures and a maximum transistor count, leading to high power dissipation and fabrication costs [6]. Therefore, it is important to improve the design of the ALU. Recently, many improved techniques have been implemented to design an ALU with a smaller silicon footprint, lower transistor count, and circuit complexity [8]. The Vedic mathematics method was employed in previous studies to implement the optimal ALU [13]. Likewise, to design an ALU with less circuit complexity,

certain prevailing studies used basic reversible gates and Cadence tools [7, 11].

Similarly, Moore's law was utilized to effectively reduce the transistor count while designing the ALU [19]. Likewise, some prevailing works utilized the combination of clock gating and one-hot coding technique for the optimal design of ALU and ensuring less switching activity [29]. In addition, to design a 4-bit ALU with low power dissipation, the Threshold Adaptive Memristor (TEAM) model Complementary Metal-Oxide-Semiconductor (CMOS) logic was employed [12]. Similarly, for the optimal design of an ALU, some existing works utilized the Gate Diffusion Input (GDI) technique and a 17T full adder [4, 22]. However, the existing studies failed to perform data path reallocation in the ALU, thereby increasing the resource utilization along with the power consumption of the ALU. To overcome this shortcoming, a novel data path reallocation and silicon footprint minimization-aware implementation of a 4-bit ALU utilizing E-FRIMA and Dw-FGCG is proposed in this study.

1.1. Problem Statement

Existing studies related to the design of a 4-bit ALU with a minimized silicon footprint have certain drawbacks, which are explained below.

- None of the prevailing works concentrated on data path reallocation in ALU, thus increasing the resource utilization and power consumption of ALU. In addition, this may lead to poor security and low performance.
- In [25], clock gating was not considered; therefore, all ALU components received the clock signal continuously, which increased circuit complexity and generated more heat.
- The annealing process introduced more defects [21], requiring additional circuits for error correction and stability enhancement. In addition, it increases the transistor count to compensate for the signal degradation, thus increasing the silicon footprint.
- Ternary logic diminishes the total number of transistors needed per gate compared with binary logic. The prevailing [9] had an area overhead owing to the requirement of additional voltage regulators, level shifters, and signal stabilizers to prevent errors.
- Owing to the greater number of logic levels in the design of the ALU, the transistor count increased, and a larger silicon area was required.

1.2. Objectives

The key objectives of the proposed methodology are as follows.

- E-FRIMA was established to perform data path reallocation in the ALU, thus reducing the resource utilization and power consumption of the ALU.
- Dw-FGCG is employed for clock gating, thereby diminishing circuit complexity and power dissipation.
- PS-GANFIS was used to identify the defects, and C2B-2CTSO was introduced to mitigate the defects and reduce the transistor count and the silicon footprint.
- RTZ-AL is utilized to reduce the ALU area overhead by minimizing errors.
- ZKM was introduced to reduce the number of logic levels in the ALU design, thereby reducing the transistor count and area overhead.

The remainder of this paper is arranged as follows: the literature survey is presented in Section 2, the proposed framework is illustrated in Section 3, the results and discussion are presented in Section 4, and finally, Section 5 presents the conclusion of the proposed model with future work.

2. Literature Review

In [25], a low-power ALU-enabled sliced processor was presented. Here, using Gated Diffusion Input (GDI) logic and Modified GDI (MGDI) logic, optimal and low-power ALUs were implemented. Likewise, to reduce power consumption, a recursive parallel self-timed adder was utilized. The model significantly reduces the transistor count and power dissipation. However, the model failed to consider clock gating, thereby increasing circuit complexity and producing more heat.

In [21], a model for designing a neuromorphic circuit with minimum power consumption was proposed. To optimally implement the neuromorphic circuit, the Izhikevich (IZH) mathematical model was used. This research significantly decreased the power consumption and maintained the same computational speed level. However, the annealing process introduces more defects, requiring additional circuits for error correction and stability enhancement. Likewise, it increases the transistor count to compensate for the signal degradation, thus increasing the silicon footprint.

In [9], the authors presented a framework named the optimal implementation of ALU with noise tolerance, along with power-optimized ternary combinational circuits. To reduce the power consumption and transistor count, a ternary multiplexer and a ternary half adder were used. Each ternary circuit employed carbon nanotube field-effect transistor technology to achieve all levels. The model maintained an excellent low noise sensitivity and reduced the delay. Nevertheless, owing to the utilization of additional voltage regulators, level shifters, and signal stabilizers, the model has area overhead.

In [17], a model for designing an ALU for photonic circuits was presented. MicroRing Resonators (MRRs) with low losses were used to design an ALU with a reduced silicon footprint. Here, the electro-optic full adder/subtractor, as well as '3' input odd-even parity checkers, were employed. Their research achieved high reliability, computational speed,

efficiency, performance, and accuracy. However, it has struggled to implement complex arithmetic operations.

In [20], an optimal hardware implementation model was proposed. In this work, the ALU design integrated a radix-4 multiplier (i.e., having efficiency in both multiplication and reduction) to maximize the ALU efficiency and reduce the additional hardware requirements.

Here, the optimized ALU executed operations, such as addition, squaring, and multiplication. The model boosted the overall computational speed and improved performance. However, owing to the handling of signed digits, extra switching activity was introduced, which affected power efficiency.

In [27], the recommended high-speed Field-Programmable Gate Array (FPGA) implementation framework was proposed. Here, an ultralow-latency modular multiplier was employed to implement a high-speed FPGA model. A memory access method was introduced by scheduling the input and output of the ALU in two identical Random Access Memories (RAMs), thus diminishing latency. Based on the results, the model achieved superior area efficiency. Nevertheless, owing to high-speed computation, the model exhibited increased dynamic power consumption.

In [3], a heterogeneous ALU architecture model was presented. Scaling ripple carry adders were used to design an energy-efficient and high-performance heterogeneous ALU architecture. Finer-grained task scheduling was achieved based on input operand size-based and energy-constraint-based controls. As per the experimental results, the proposed model saved energy excellently. Nevertheless, because of the integration of multiple types of ALU cores, the silicon footprint and transistor count increased.

In [10], a CNFET-enabled ternary ALU model was proposed. Here, the ALU model comprises a function-select block, transmission gate block, and functional modules. The functional modules were designed based on the 2:1 multiplexer-based design approach, thus neglecting the requirements of the decoders. The model had a lower power consumption. However, the research introduced additional complexity regarding the circuit design, control logic, and signal processing.

In [23], an energy-efficient ternary ALU was designed in a Carbon Nanotube Field-Effect Transistor (CNFET). Ternary logic was used to design an ALU with a reduced circuit overhead. Likewise, the presence of symmetric literals across several single- and dual-shift operators, in addition to subtraction operations, led to the optimized implementation of adder/subtractor modules. The model obtained superior outcomes with respect to power consumption and device count. However, a greater number of logic levels in the ALU design increases the transistor count.

In [1], a model for designing a robust single-layer Quantum-dot Cellular Automata (QCA) technology ALU was presented. Here, for the optimal design of the ALU, a fault-tolerant 3-input majority gate with ten simple and rotated cells was used. The model was more tolerant to single-cell omission and extracellular deposition defects. However, the performance of the model was highly sensitive to temperature variations.

Recent studies highlight the growing importance of efficient hardware architectures and optimization techniques in modern electronic systems. In [36], the authors discuss interdisciplinary advances in nanoscience that support compact and high-performance engineering designs. Research in emphasizes advanced antenna and communication technologies requiring optimized hardware resources.

Additionally, the study on SRAM optimization highlights the significance of power efficiency and fault tolerance. These works motivate techniques like E-FRIMA and DW-FGCG for datapath reallocation and silicon footprint minimization in ALU design.

Based on the reviewed literature, several research gaps remain unaddressed in the design of low-power and high-performance ALUs. Although existing approaches effectively reduce power consumption, transistor count, or delay using techniques such as GDI/MGDI logic, ternary logic, CNFETs, photonic circuits, FPGA optimizations, and emerging technologies like QCA, most models focus on optimizing one or two performance metrics in isolation.

Power-efficient designs often overlook critical aspects such as clock gating, thermal management, and dynamic power reduction under high-speed operation. Similarly, high-speed, noise-tolerant ALUs introduce substantial area overhead due to additional components, such as voltage regulators, level shifters, stabilizers, or multiple ALU cores.

Emerging paradigms, including ternary logic, neuromorphic circuits, photonic ALUs, and QCA-based designs, improve efficiency or robustness but suffer from increased design complexity, scalability limitations, temperature sensitivity, or restricted support for complex arithmetic operations.

Furthermore, several implementations rely on additional circuitry for error correction, control logic, or signal stability, leading to increased silicon footprint and reduced energy benefits. Hence, there exists a clear research gap in developing a unified ALU architecture that simultaneously achieves low power consumption, reduced transistor count, minimal thermal impact, scalability for complex arithmetic operations, and robustness against noise and environmental variations without incurring significant area or design complexity overhead.

3. Proposed Data Path Reallocation and Silicon Footprint Minimization-based 4-bit ALU Design Methodology

The proposed E-FRIMA is employed for datapath reallocation. The proposed ZKM algorithm was utilized to reduce the number of logic levels in the ALU. Similarly, to minimize errors, the proposed RTZ-AL was established. The proposed PS-GANFIS was used to identify defects while performing the ALU operations. Similarly, to mitigate the identified defects, the proposed C2B-2CTSO is utilized. The proposed Dw-FGCG is introduced to provide clock gating. An architectural diagram of the proposed model is shown in Figure 1.

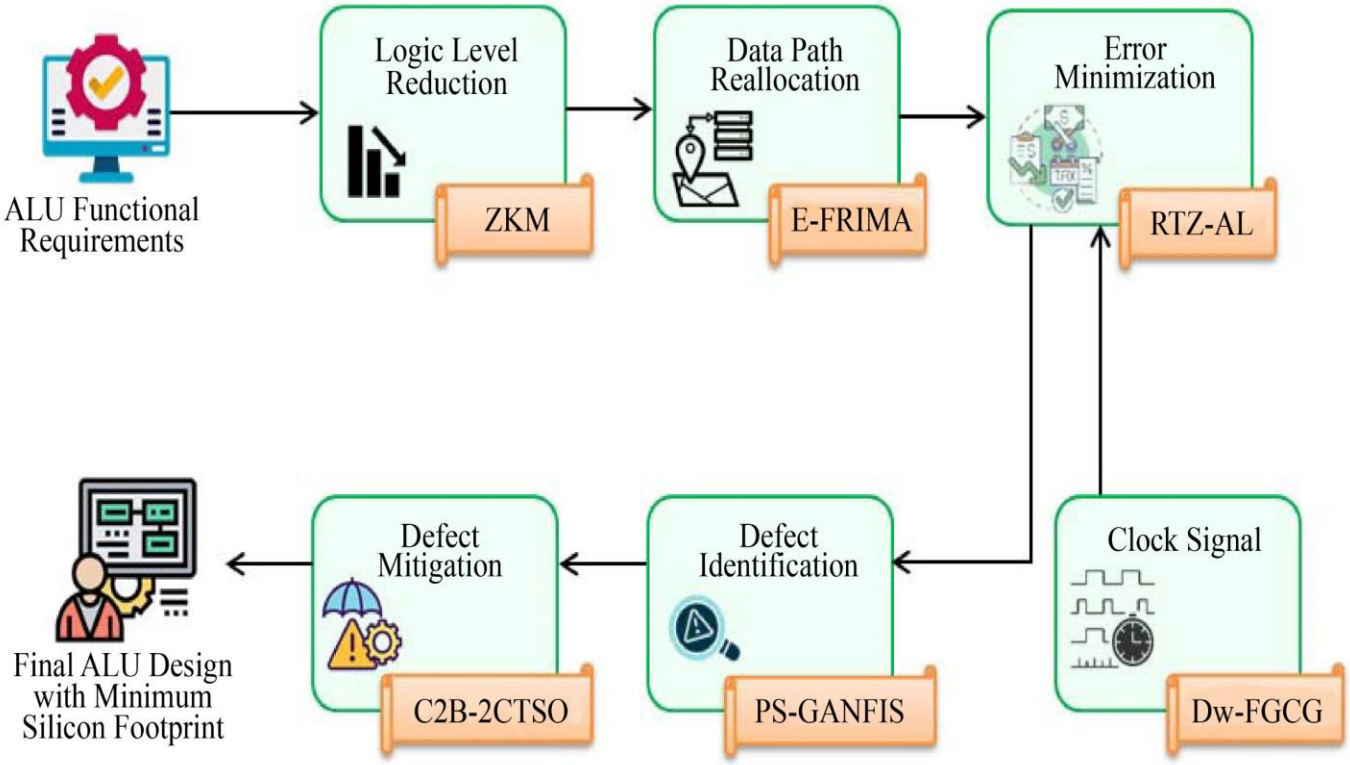


Fig. 1 Architectural diagram of the proposed model

The proposed methodology effectively designed a 4-bit ALU with a minimized silicon footprint, circuit complexity, area overhead, and transistor count.

The processes involved in the proposed model are explained briefly in the following sections.

3.1. ALU Functional Requirements

Initially, for performing the ALU operation, the functional requirements (i.e., arithmetic operations such as addition, subtraction, multiplication, and division; logical operations such as AND, OR, XOR, NOT, NAND, and NOT; bitwise and shift operations; and comparison operations) are provided. The ALU functional requirements $\left(\Im_x^{ALU}\right)$ are defined as,

$$\Im_x^{ALU} \rightarrow \left[\Im_1^{ALU}, \Im_2^{ALU}, \Im_3^{ALU}, \dots\dots, \Im_\kappa^{ALU}\right] \quad where \quad x = (1\ to\ \kappa) \quad (1)$$

$\Im_\kappa^{ALU}$ requires several functional ALU requirements.

3.2. Logic Level Reduction

Subsequently, to reduce the logic level in the ALU design, the ALU functional requirements $\left(\Im_x^{ALU}\right)$ are constructed as a map by employing ZKM. Usually, K-maps (KM) provide a perfect and intuitive way to group terms and simplify Boolean functions. However, KM requires manual observation and grouping, which introduces errors. To address this problem, the Zagros function was employed to determine the size of KM. The working process of the ZKM model is derived as follows:

Initially, KM (K_m) is created $\left(\Im_x^{ALU}\right)$ based on the Zagros function, which adjusts the grouping excellently and makes necessary corrections. It is expressed as,

$$K_m = \sum_{x=1}^{\kappa}\left[\Im_x^{ALU}\, sin\left(\Im_x^{ALU}\right) + cos\left(\Im_x^{ALU}\right)\right] \quad (2)$$

The truth table values of the Boolean functions were then placed into the(K_m). Next, the adjacent 1s were grouped into power-of-two-sized blocks. Here, all groups must be large while covering all 1s. The common variables were determined for each group. The condition for writing the Boolean expression (CB) is then provided as

$$CB = \begin{cases} if \quad \vartheta = \text{Constant 1 in } a \text{ group} & Then \quad \text{included} \\ if \quad \vartheta = \text{Constant 0 in } a \text{ group} & Then \quad \text{its complement is included} \\ if \quad \vartheta = \text{Changes within } a \text{ group} & Then \quad \text{eliminated} \end{cases} \tag{3}$$

Here, ϑ denotes a variable. Finally, by summing all grouped terms, a simplified Boolean function is formed. Therefore, the logic-level reduction outcomes are denoted L_r.

3.3. Data Path Reallocation

Then, by using Excoffier Functional Redundancy Identification and Merging Algorithm (E-FRIMA), the data path reallocation is performed for the$L_r$for diminishing the resource utilization and power consumption of ALU. Usually, the Functional Redundancy Identification and Merging Algorithm (FRIMA) optimizes the datapath by diminishing unwanted computations. However, the Hamming distance only identifies bitwise differences, not the functional redundancy in FRIMA.

To solve this issue, the Excoffier distance is used instead of the Hamming distance in the FRIMA. The step-by-step working process of the proposed E-FRIMA is as follows.

- ✓ Equivalent functional units were detected. If two functional units (i.e., $L_l$and L_y) are equivalent, their outputs match for all inputs. It is expressed as,

$$\hbar_l(I) \equiv \hbar_y(I) \quad \forall I \tag{4}$$

Where $\hbar_l(I)$ and $\hbar_y(I)$ denote the Boolean functions computed by L_l and L_y, respectively, and Idenotes the input operands. If the above condition in Equation (4) is satisfied, then the functional units perform identical operations [35].

- ✓ Unused functional units were detected. If the functional units are not employed in any ALU operation cycle, they are considered unused. The usage function (Us) is expressed as

$$Us(L_r) = \sum_{\tau=1}^{\varepsilon} Z_r(\tau) \tag{5}$$

$$\Xi = \begin{cases} if \quad L_r \ is \ used & Z_r(\tau) = 1 \\ else & Z_r(\tau) = 0 \end{cases} \tag{6}$$

Here, $Z_r(\tau)$ outlines the usage status, ε indicates the total execution time, and Ξ illustrates unused functional unit detection outcomes. Suppose $Z_r(\tau) = 0$it is an unused functional unit. Unused functional units were removed from the ALU.

- ✓ Furthermore, functional redundancy was captured based on a functional similarity graph. Here, the similarity function $\left(Sim(L_l, L_y)\right)$ was estimated using the Excoffier distance instead of the Hamming distance. It is given as,

$$Sim(L_l, L_y) = \sqrt{\sum_I \left(\hbar_l(I) - \hbar_y(I)\right)^2} \tag{7}$$

The functional units (i.e., $L_l$and L_y) are redundant during $\left(Sim(L_l, L_y) = 0\right)$. Then, L_y it is merged intoL_l, and one unit is removed. The updated functional unit set (U) is provided as

$$U = L_r - \{L_y\} \tag{8}$$

Subsequently, the data path reallocation matrix was updated to ensure that all operations were perfectly assigned to the remaining functional units. It is determined as,

$$\Re_{\partial t} = Op(U) \tag{9}$$

Here, $Op(U)$ indicates optimized mapping. The reallocated data path is denoted by $\Re_{\partial t}$. The pseudocode for E-FRIMA is as follows.

Algorithm-1: Pseudocode for E-FRIMA

Input: Logic-level reduction outcome (L_r)
Output: Reallocated data path $(\Re_{\partial t})$

Begin
  Initialize (L_r)
  For (L_r)
    Identify equivalent functional units.
      $\hbar_l(I) \equiv \hbar_y(I) \quad \forall I$
    Compute Usage function
      $\boldsymbol{Us(L_r) = \sum_{\tau=1}^{\varepsilon} Z_r(\tau)}$
    Discovering unused functional Units
      $\Xi =$
      $\begin{cases} if \quad L_r \ is \ used & Z_r(\tau) = 1 \\ else & Z_r(\tau) = 0 \end{cases}$
    Estimation of the similarity function $\left(Sim(L_l, L_y)\right)$
    The functional unit set is updated.
      $U = L_r - \{L_y\}$
    Update
      $\Re_{\partial t} = Op(U)$
  End For
  Obtaining the Reallocated data path $(\Re_{\partial t})$
End

Therefore, the proposed E-FRIMA reallocates the data paths superiorly.

3.4. Error Minimization

Subsequently, based on Return-To-Zero Asynchronous Logic (RTZ-AL), the errors present in the reallocated data

path $(\Re_{\partial t})$ are minimized, thus reducing the ALU area overhead. Typically, Asynchronous Logic (AL) generates less electromagnetic noise and effectively reduces timing errors. Nevertheless, owing to the need for handshaking circuits with Dual-Rail Encoding (i.e., request-acknowledge lines), the hardware complexity of AL increases. To overcome this problem, return-to-zero (RTZ) encoding is used instead of Dual-Rail Encoding. The structural layout of the proposed RTZ-AL is shown in Figure 2.

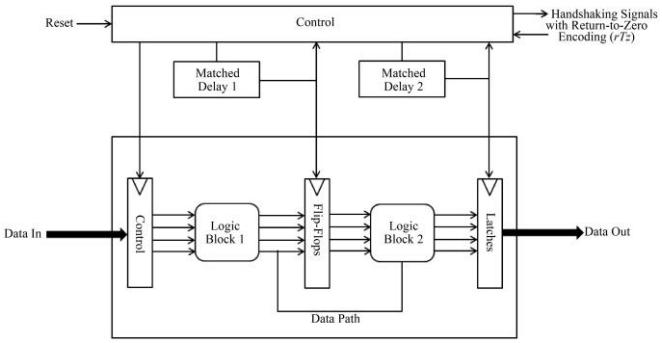


Fig. 2 Structural layout of the proposed RTZ-AL

The working principle of RTZ-AL is described as follows:

➲ The asynchronous gate operation is defined as follows:

$$V = \eta(\Re_{\partial t}) \tag{10}$$

Here, $\eta(\Re_{\partial t})$ implies a Boolean function, which identifies the time at which the output V changes.

➲ Then, the data transfer between the components is managed by the handshaking protocols, thus diminishing the errors that arise owing to clock variations. A two-phase handshake between sender (ςe) and receiver $(R\varepsilon)$ is estimated using the following equations: To reduce the hardware complexity of AL, RTZ encoding (rTz) is employed with handshaking circuits. It is provided as,

$$\varsigma e \xrightarrow{rTz} R\varepsilon : \partial Y \tag{11}$$

$$R\varepsilon \xrightarrow{rTz} \varsigma e : Ak \tag{12}$$

$$\varsigma e \xrightarrow{rTz} R\varepsilon : S \tag{13}$$

$$rTz = \begin{cases} if \;\; \Re_{\partial t} = 1 \;\; (for\ half\ clock\ cycle) & Then \;\; \alpha\tau \\ otherwise & Then \;\; idl \end{cases} \tag{14}$$

Here, ∂Y designates that the data are ready, Ak indicates the acknowledged process, S indicates the reset process, $\alpha\tau$ represents the active high level, and idl indicates the idle state.

➲ Likewise, RTZ-AL uses delay-intensive circuits to prevent glitches due to hazards. Next, the error-free operation of RTZ-AL is verified as

$$st(h+1) = st(h) + Inc \cdot \phi \tag{15}$$

Here, $st(h)$ indicates the state marking at time h, Inc portrays the incidence matrix, and ϕ determines the firing vector. Equation (15) ensures that the system avoids incorrect state transitions:

➲ Next, the probability of error (prb_{error}) owing to a timing violation is computed as follows:

$$prb_{error} = exp^{-\frac{C_{mrg}}{C_{clk}}} \tag{16}$$

Here, C_{mrg} designates the available timing slack and C_{clk} signifies the clock period. In RTZ-AL, timing was not fixed. Then, (prb_{error}) approaches zero, thus minimizing the errors excellently. After minimizing the errors, an error-free data path is denoted as E_v.

i) Clock Signal

To ensure that the clock signal is enabled when needed and disabled when not in use, the clock signal is sent to the transistor (T) through Drop Wave Fine-Grained Clock Gating (DW-FGCG). Usually, Fine-Grained Clock Gating (FGCG) significantly reduces energy consumption without affecting the performance. However, the overall system timing is impacted in FGCG owing to the increase in clock-enabled signal propagation delays. To overcome this issue, the drop wave function is employed in FGCG for switching activity [34], thus diminishing the signal propagation delays. The working process of the Dw-FGCG is explained below.

❄ Initially, the dynamic power consumption (P) of the clocked circuit (clc) is defined as:

$$P = \varsigma \ddot{C} W^2 fq \tag{17}$$

Here, ς indicates the switching activity factor, $\ddot{C}$ signifies the load capacitance, W indicates the supply voltage, and fq designates the clock frequency. The main objective of clock gating is to reduce ς power consumption.

❄ Then, the enable signal for clock gating is determined and formulated as

$$clc_{gat}(v) = clc_{org}(v) * \zeta(v) \tag{18}$$

Here, $clc_{gat}(v)$ designates the gated clock signal, $clc_{org}(v)$ outlines the original clock signal, and $\zeta(v)$ signifies the enable signal. Here, $\zeta(v)$ becomes 1 when the clock is active and 0 when it is disabled.

❄ After applying the clock signal, the effective switching activity factor (ς_{ef}) is estimated. The drop wave function was used to reduce the signal propagation delay. This is mathematically expressed as follows:

$$\varsigma_{ef} = \varsigma \cdot pb(\zeta = 1) * D \tag{19}$$

$$D = -\frac{1+cos\left(12\sqrt{\left(clc_{org}\right)^2+\left(clc_{gat}\right)^2}\right)}{0.5\left(\left(clc_{org}\right)^2+\left(clc_{gat}\right)^2\right)+2} \quad (20)$$

Where$pb(\zeta = 1)$ outlines the probability, and ζ is 1. Clock gating degrades the average activity by$pb(\zeta = 1)$, thus reducing power consumption. In addition, $pb(\zeta = 1)$ it is less than one for fine-grained gating.

- Usually, the Dw-FGCG applies gating at individual transistor levels. For each transistor, the clock-enabling condition is expressed as:

$$\zeta(v) = A(T) \quad (21)$$

Here, $A(T)$ indicates the clock signal activity in the transistor. Therefore, the Dw-FGCG ensures that only the required transistor receives a clock to save power.

3.5. Defect Identification

Then, using the Power-Skewed Gaussian Adaptive Network-based Fuzzy Inference System (PS-GANFIS), the defects present while performing ALU operations are identified. Usually, an Adaptive Network-based Fuzzy Inference System (ANFIS) can learn complex patterns from defect data, thus enhancing detection accuracy.

However, when the input features increase, the number of fuzzy rules increases exponentially, making the training process of ANFIS more difficult. To address this issue, a Power-Skewed Gaussian membership function is utilized in the ANFIS. The mathematical expression for the proposed PS-GANFIS is as follows:

Primarily, if-then conditions are employed to compute the fuzzy rules (F) and are defined as

$$F \Rightarrow \begin{cases} if \quad \rho\partial \leq 10\% \quad \text{of} \quad \xi \quad Then \quad \beta \\ if \quad \rho\partial \geq 10\% \quad \text{of} \quad \xi \quad Then \quad nD \end{cases} \quad (22)$$

Here, $(\rho\partial)$ indicates the propagation delay, ξ denotes the expected delay, β denotes the presence of signal degradation owing to defects, and nD indicates the absence of defects.

The five layers of PS-GANFIS are explained as follows.

Layer 1: The first layer of the PS-GANFIS is the fuzzification layer. Here, each node was adjusted to a function parameter. To avoid the harder training process of ANFIS, the Power-Skewed Gaussian membership function (χ) is utilized. The fuzzification layer $\left(J_{1,b}\right)$ is formulated as

$$J_{1,b} = \chi(\rho\partial) \quad (23)$$

$$\chi = exp^{-\frac{(\rho\partial - j)^2}{2\sigma^2}} \cdot (1 + \psi\rho\partial)^{\theta} \quad (24)$$

Here, exp the exponential function j signifies the center or mean of the Gaussian function, σ indicates the standard deviation, ψ indicates the asymmetry factor, and θ represents the power term.

Layer 2: The second layer of PS-GANFIS is the rule layer. Each node was fixed, and the firing strength was assessed for all rules. The rule layer $\left(J_{2,b}\right)$ is expressed as:

$$J_{2,b} = \gamma_b \xrightarrow{F} \chi(\rho\partial) \quad Here \quad b = 1,2 \quad (25)$$

Where γ_b signifies the firing strength.

Layer 3: The third layer is known as the normalization layer. The firing strength (γ_b) was normalized. The normalization layer outcomes $\left(J_{3,b}\right)$ are specified as

$$J_{3,b} = \bar{\gamma}_b = \frac{\gamma_b}{\sum_b \gamma_b} \quad (26)$$

Here, $\bar{\gamma}_b$ denotes the normalized firing strength.

Layer 4: The fourth layer is the defuzzification layer. Each node has a function. It is written as,

$$J_{4,b} = \bar{\gamma}_b \cdot (F) = \gamma_b(\varphi_b\rho\partial + M_b) \quad (27)$$

Here, $J_{4,b}$ specifies the defuzzification layer outcomes, φ_b and M_b signifies the parameter sets.

Layer 5: The last layer of PS-GANFIS is the output layer. Here, by summing all arriving signals from the previous node, the final outcome is obtained. The output layer outcome $\left(J_{5,b}\right)$ is defined as

$$J_{5,b} = \sum_b \bar{\gamma}_b \cdot (F) = \frac{\sum_b \bar{\gamma}_b \cdot (F)}{\sum_b \bar{\gamma}_b} \quad (28)$$

Therefore, the defect identification outcomes (DI) are written as

$$DI = [\beta, nD] \quad (29)$$

The pseudocode for PS-GANFIS is as follows.

Algorithm 2: Pseudocode for PS-GANFIS.

Input: Propagation delay $(\rho\partial)$
Output: Defect identification outcomes: (DI)

Begin
  Initialize $(\rho\partial)$
  For $(\rho\partial)$
    Compute fuzzy rules
    $F \Rightarrow \begin{cases} if \quad \rho\partial \leq 10\% \quad \text{of} \quad \xi \quad Then \quad \beta \\ if \quad \rho\partial \geq 10\% \quad \text{of} \quad \xi \quad Then \quad nD \end{cases}$
    Implementation of five-layer operations
    Fuzzification layer performance

$$\chi = exp^{-\frac{(\rho\partial - j)^2}{2\sigma^2}} \cdot (1 + \psi\rho\partial)^{\Theta}$$

Find the estimate rule layer.

$$J_{2,b} = \gamma_b \xrightarrow{F} \chi(\rho\partial) \quad Here \quad b = 1,2$$

Evaluation of the normalization layer $(J_{3,b})$

Perform the defuzzification layer.

$$J_{4,b} = \bar{\gamma}_b \cdot (F) = \gamma_b(\varphi_b\rho\partial + M_b)$$

Compute

$$J_{5,b}$$

End For

Obtain Defect identification outcomes: $DI = [\beta, nD]$

End

Therefore, the proposed PS-GANFIS effectively identifies defects while performing ALU operations.

3.6. Defect Mitigation

If a defect is present β, it is mitigated using Chen Bird Baker Crayfish Clock Tree Synthesis Optimization (C2B-2CTSO). Commonly, Clock Tree Synthesis (CTS) effectively mitigates annealing-induced defects, leakage current, and additional circuit requirements by balancing the skew, optimizing power [33], and improving timing stability. However, unpredictable clock jitter can occur in CTS because of variations in clock arrival times across the circuit, which degrades the reliability of the circuit. To address this problem, Clock Skew was optimized using CrayFish Optimization (CFO). CFO employs adaptive movements and directional updates to obtain the best solutions. However, the CFO can still become stuck in the local optima. To avoid this, a Baker map was used instead of a random uniform distribution. In addition, the Chen Bird function is used to optimize the buffer insertion count in the CTS. The working process of C2B-2CTSO is as follows.

- Initially, the clock delay (H_k) is computed at each node as

$$H_k = R_k N_k \tag{30}$$

Here, R_k exhibits the resistance of the path and N_k specifies the capacitance at that node. To prevent timing failure defects, the delay was balanced at all clock nodes.

- The clock skew $(\delta_{kd})$is the difference in the arrival time of the clock signal between two points (i.e., H_k and H_d). It is equated as,

$$\delta_{kd} = |H_k - H_d| \tag{31}$$

Next, based on Baker CrayFish Optimization (BCFO), $\delta_{kd}$the circuit was optimized to improve circuit reliability.

3.7. Clock Skew Optimization

The crayfish population is initialized with respect to its position. Here, the $(\delta_{kd})$is considered as the initialized population. The initialized population $(Y_{p,q})$ is defined as

$$Y_{p,q} = lp_q + (up_q - lp_q) \times Bk \tag{32}$$

$$Bk = \begin{cases} \frac{Y_{p,q}}{c} & 0 \le Y_{p,q} < c \\ \frac{Y_{p,q} - c}{1 - c} & c \le Y_{p,q} \le 1 \end{cases} \tag{33}$$

Here, lp_q and up_q indicate the lower and upper bounds, respectively, Bk designates the baker map function that prevents the local optima issue, and c is the parameter. Then, the fitness function (G) is calculated by considering the buffer count as a minimum $(min(\beta\mu))$, and is formulated as

$$G \rightarrow min(\beta\mu) * G(Y_{p,q}) \tag{34}$$

Usually, changes in temperature affect crayfish behavior. The temperature of the crayfish located in the environment (Tp) is given as

$$Tp = rf \times 15 + 20 \tag{35}$$

Here, rf represents a random number. If (Tp) is high, then the crayfish choose a cool place. Likewise, crayfish exhibit foraging behavior at the appropriate(Tp)time. Similarly, temperature affects the feeding behavior of crayfish. The crayfish intake (B) is modeled as

$$B = X_1 \times \left(\frac{1}{\sqrt{2\times\pi\times\Phi}} \times exp\left(-\frac{(Tp - \Psi)^2}{2\Phi^2}\right)\right) \tag{36}$$

Here, Φ and X_1 controlled the intake of crayfish at various temperatures and Ψ defined the most suitable temperature for crayfish.

3.7.1. Exploration

The crayfish chose to join the cave when the temperature exceeded 30. Therefore, the location of crayfish during summer $(Y_{p,q}^{new1})$ is specified as

$$Y_{p,q}^{new1} = Y_{p,q} + X_2 \times rf \times (Y_{cave} - Y_{p,q}) \tag{37}$$

Where X_2 indicates the decreasing cave, and Y_{cave} denotes the cave chosen by the crayfish.

3.7.2. Exploitation: (Competition Stage)

Now, cra

yfish compete with others and adjust their location with respect to the location of another crayfish. It is expressed as,

$$Y_{p,q}^{new2} = Y_{p,q} - Y_{i,q} + Y_{cave} \tag{38}$$

Here, Y_{cave} signifies the updated position of crayfish during the competition stage, and $Y_{i,q}$ is a random individual crayfish.

3.7.3. Foraging Stage

At the appropriate temperature, crayfish exhibit foraging behavior. Here, the crayfish moves towards the food and finds it. Then, it determines the food size. If the food size is large, it carries it and eats it within a few walking feet. The foraging behavior of crayfish $\left(Y_{p,q}^{new3}\right)$ is modeled as

$$Y_{p,q}^{new3} = Y_{p,q} + Y_{fd} \times B \times (cos(2 \times \pi \times rf) - sin(2 \times \pi \times rf)) \quad (39)$$

Here, Y_{fd} designates the location of the food. This position update continues until convergence is achieved. Thus, the optimized clock skew is denoted by $\ddot{\delta}_{opt}$.

- Similarly, unwanted capacitance is reduced, and the clock tree is optimized to prevent high power consumption owing to the defects. The probability of failure $\left(Q_{fl}\right)$ in a path is then determined as

$$Q_{fl} = `1 - (1 - z)^{\varpi} \quad (40)$$

Here, z signifies the probability of failure per clock segment and ϖ signifies the number of clock segments. If $\left(Q_{fl}\right)$ is present, redundant paths are added to improve the robustness of the clock tree. Thereafter, jitter and process variations were controlled.

- Finally, based on the Chen Bird function, the **b**uffer insertion count $(\beta\mu)$ was optimized as follows:

$$g \xrightarrow{\beta\mu} \left(\frac{\left(\frac{n_{dy}}{\rho}\right)^{\frac{1}{3}} (N_{load})^{\frac{2}{3}}}{N_{buf}} \right) + 1 \quad (41)$$

Here, g indicates the optimized buffer insertion count, n_{dy} signifies the clock tree delay, ρ implies the parameter, N_{load} denotes the total load capacitance, and N_{buf} represents the capacitance in one buffer. Finally, a 4-bit ALU design with a minimized silicon footprint, transistor count, and circuit complexity was obtained using the proposed methodology.

4. Results and Discussion

Here, performance validation and comparative evaluation of the proposed model and conventional methods are performed to demonstrate the dependability of the proposed model. In addition, the proposed model was implemented in the working platform of MATLAB.

4.1. Performance Validation for Data Path Reallocation

Here, the performance validation of the proposed E-FRIMA as well as the existing FRIMA, Redundancy Addition and Removal (RAR), Resource Sharing and Functional Unit Merging Algorithm (RS-FUMA), and Quine–McCluskey Algorithm (QMA) is performed as follows:

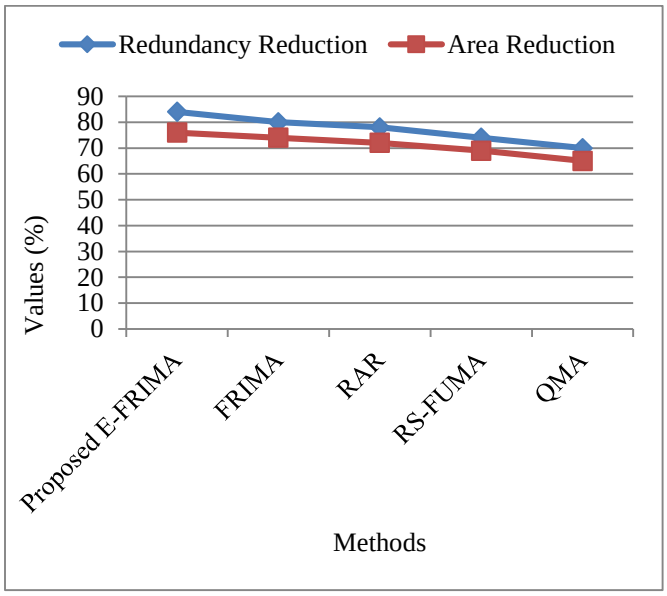
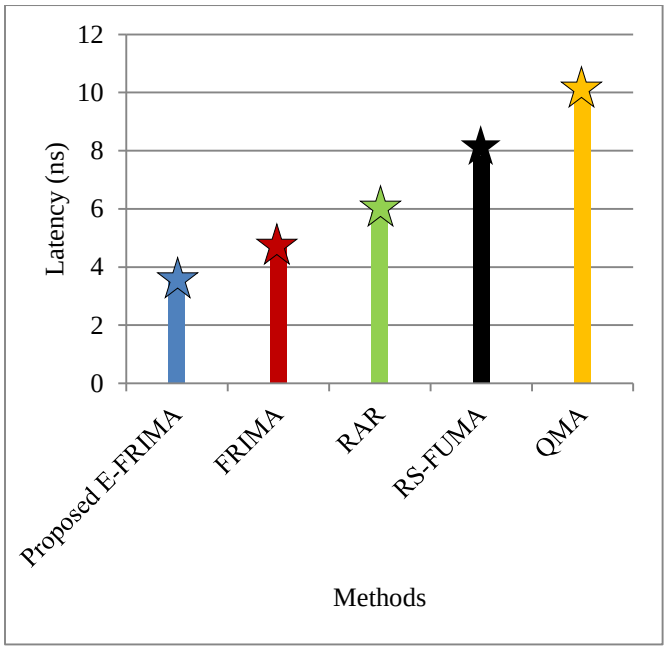
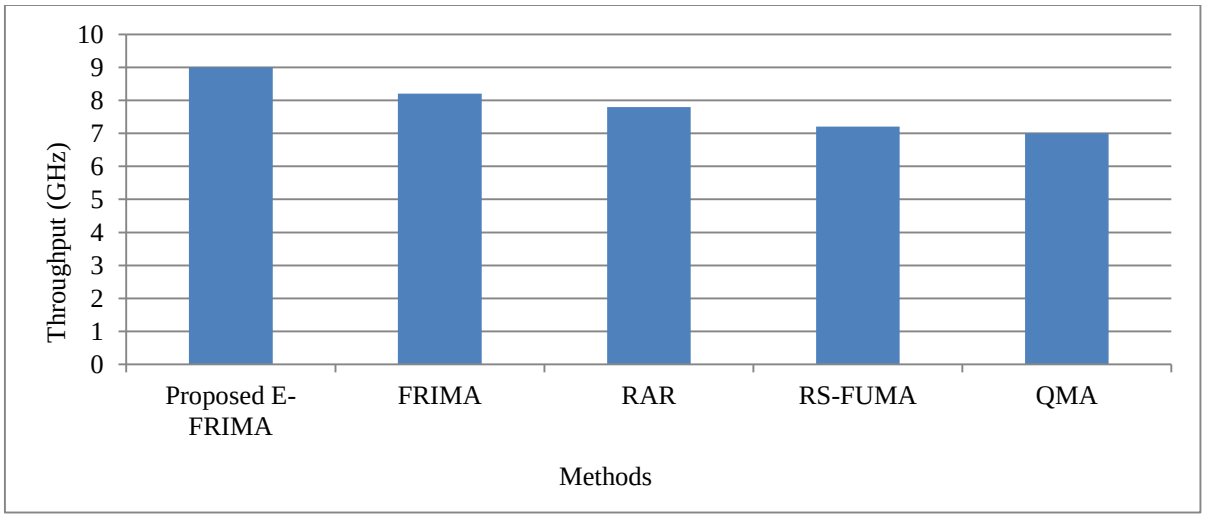


Fig. 3 Performance assessment of redundancy reduction and area reduction

The performance assessment of the proposed E-FRIMA and the prevailing methods regarding redundancy and area reduction is shown in Figure 3. The proposed E-FRIMA achieved high redundancy reduction (84%) and area reduction (76%). However, existing FRIMA and RS-FUMA obtained low redundancy reductions of 80% and 74%, respectively. Likewise, the existing RAR and QMA attained low area reductions of 72% and 65%, respectively. The proposed E-FRIMA employs the Excoffier distance to identify functional redundancy, thereby effectively reallocating the data paths.



(a)



(b)

Fig. 4 Graphical representation of (a) latency and (b) throughput

In Figures 4 (a) and (b), a graphical representation of the proposed E-FRIMA and conventional methods, including FRIMA, RAR, RS-FUMA, and QMA, in terms of latency and throughput is shown. To improve the performance of data path reallocation, the Excoffier distance was modified with the existing FRIMA. The proposed E-FRIMA achieved a low latency of 3.4 nanoseconds (ns) and a high throughput of 8.9 GigaHertz (). Similarly, conventional methods, such as FRIMA, RAR, RS-FUMA, and QMA, attained a high average latency of 7.075ns and a low average throughput of 7.5475GHz. Therefore, the results proved that the proposed model is superior to conventional methods.

Table 1. Energy consumption and leakage power reduction analysis

Techniques	Energy Consumption (pJ)	Leakage Power Reduction (%)
Proposed E-FRIMA	12	74
FRIMA	32	72
RAR	45	68
RS-FUMA	56	65
QMA	89	63

The Excoffier distance is modified with the existing FRIMA for efficient datapath reallocation, thus reducing power and energy consumption. Table 1 shows the energy consumption and leakage power reduction analysis of the proposed E-FRIMA and prevailing techniques. The proposed E-FRIMA consumed less energy (12 pJ) and achieved a high leakage power reduction of 74%. However, existing techniques, such as FRIMA, RAR, RS-FUMA, and QMA, consume maximum energy of 32pJ, 45pJ, 56pJ, and 89pJ, respectively. Likewise, existing methods have achieved poor leakage power reduction. Therefore, the effectiveness of the proposed model is proven.

4.2. Performance Assessment for Defect Identification

The performance of the proposed PS-GANFIS is compared with the prevailing methods to demonstrate the reliability of the proposed model.

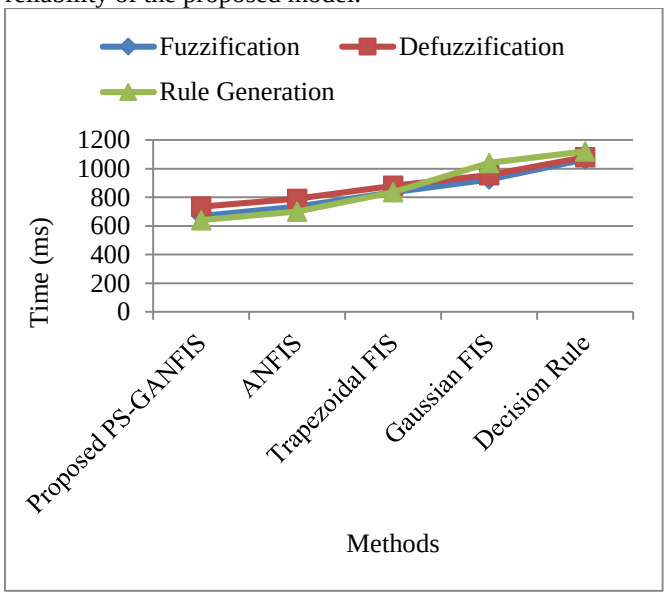


Fig. 5 Performance validation based on performance metrics.

Regarding the fuzzification time, defuzzification time, and rule generation time, the performance of the proposed PS-GANFIS is validated using the prevailing techniques in Figure 5. Regarding fuzzification, defuzzification, and rule generation time, the proposed PS-GANFIS took less time, 673ms, 730ms, and 642ms; while the prevailing ANFIS, Trapezoidal Fuzzy Inference System (Trapezoidal FIS), Gaussian FIS, and Decision Rule took a maximum value of 888ms, 926.5ms, and 923.5ms, respectively. Here, owing to the usage of the Power-Skewed Gaussian membership function, the proposed PS-GANFIS excellently identified the defects while performing ALU operations [31].

4.3. Performance Analysis for Logic Level Reduction

Here, a performance analysis is performed for the proposed ZKM and existing techniques, including KM, Espresso Heuristic Logic Minimizer (EHLM), Petrick's method (PM), and Binary Decision Diagrams (BDDs), as follows:

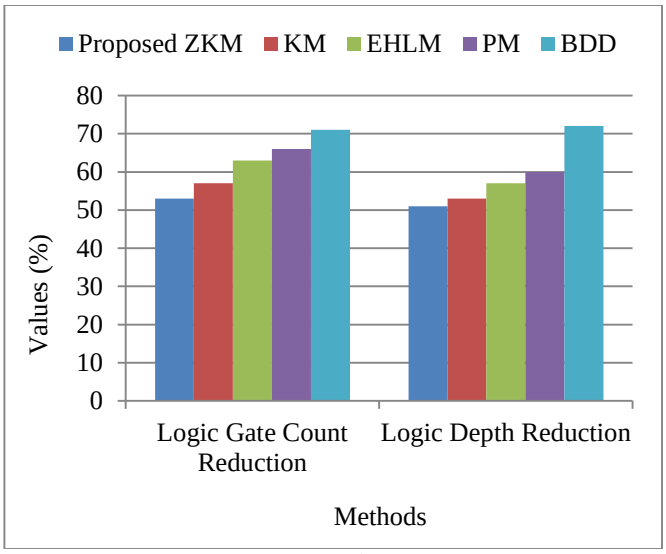


Fig. 6 Graphical analysis of logic-level reduction

In Figure 6, regarding the logic gate count reduction and logic depth reduction, the proposed ZKM performance is analogous to the prevailing techniques. Owing to the inclusion of the Zagros function, the proposed ZKM achieved a high logic gate count reduction (68%) and logic depth reduction (69%). However, the conventional EHLM and BDDs achieved low logic gate count reductions of 62% and 58%, respectively. Similarly, the conventional KM and PM methods achieved low logic depth reductions of 67.2% and 61.45%, respectively. Therefore, the proposed ZKM method significantly reduces the number of logic levels in the ALU design [32].

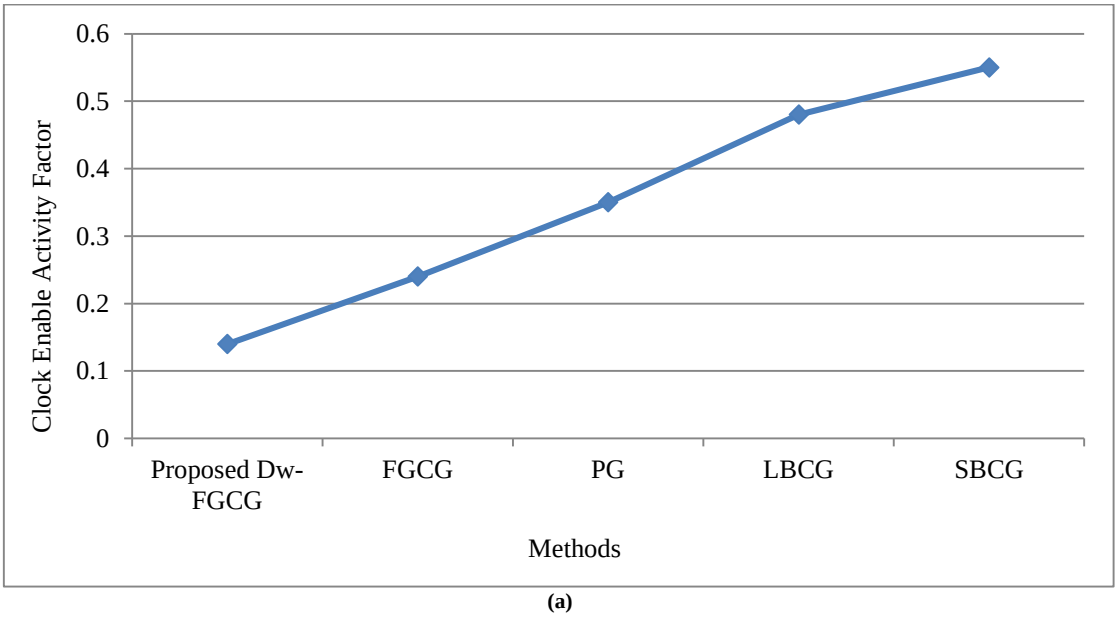
4.4. Performance Estimation for Clock Gating

The performance of the proposed Dw-FGCG is compared with the prevailing techniques to demonstrate the trustworthiness of the proposed model.

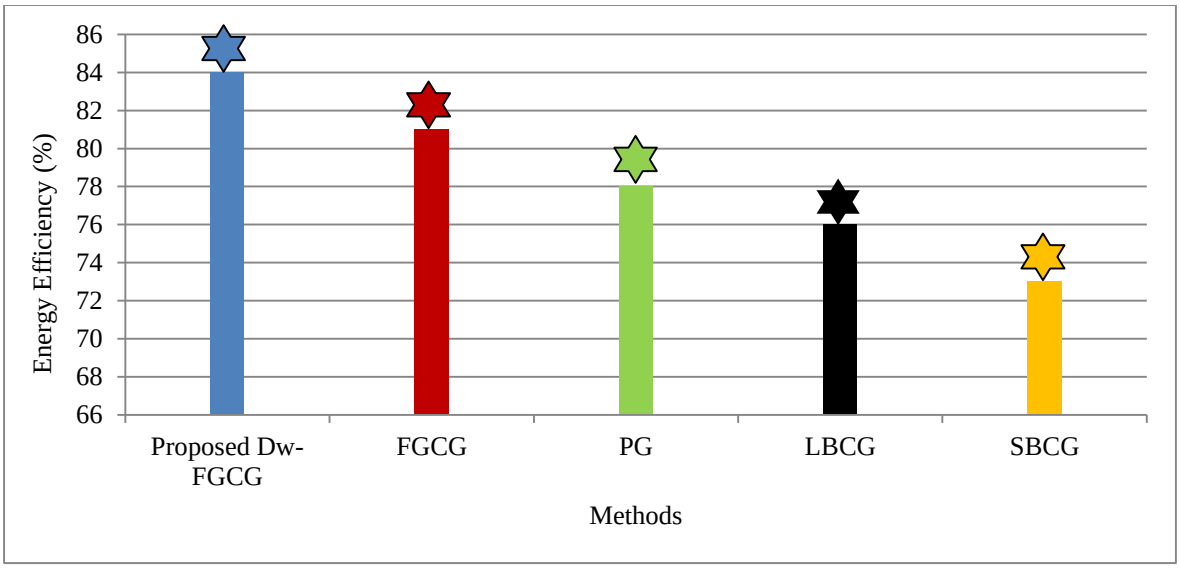
Table 2. Flip-Flop Utilization Reduction and Switching Activity Reduction Analysis

Methods	Flip-Flop Utilization Reduction (%)	Switching Activity Reduction (%)
Proposed Dw-FGCG	56	78
FGCG	52	75
PG	50	72
LBCG	48	70
SBCG	43	67

The proposed Dw-FGCG employs a drop-wave function to proficiently provide clock gating to the transistors, thus diminishing the switching activity and signal propagation delays. The flip-flop utilization reduction and switching activity reduction analysis of the proposed Dw-FGCG and the existing methods are presented in Table 2. The proposed Dw-FGCG achieved a high flip-flop utilization reduction and switching activity reduction of 56 and 78%, respectively. However, the existing methods, such as FGCG, Power Gating (PG), Latch-Based Clock Gating (LBCG), and Synthesis-Based Clock Gating (SBCG), achieved a low average flip-flop utilization reduction (48.25%) and switching activity reduction (71%). Therefore, the efficacy of the proposed model is demonstrated.



(a)



(b)

Fig. 7 Comparative validation of (a) clock-enabled activity factor, and (b) energy efficiency

The comparative validations of the proposed Dw-FGCG and prevailing methods with respect to the clock-enable activity factor and energy efficiency are displayed in Figures 7 (a) and (b). The proposed Dw-FGCG achieved a low clock-enable activity factor of 0.134 and a high energy efficiency of 83%. Similarly, the prevailing PG and SBCG attained high clock-enabled activity factors of 0.345 and 0.547, respectively. Furthermore, the prevailing FGCG and LBCG attained low energy efficiencies of 81 and 76%, respectively. Here, to smooth the transition of the clock-enabled signal and reduce the signal propagation delay, the drop wave function is modified with the existing FGCG.

4.5. Comparative Evaluation

A comparative evaluation was performed for the proposed and related works.

Table. 3 Comparative evaluation

Ref. No.	Aim	Techniques	Merit	Demerits
Propo -sed Model	Designing data path reallocation and silicon footprint minimization-aware 4-bit ALU	E-FRIMA and Dw-FGCG	The proposed model excellently reallocated the data path while designing the ALU, thus reducing resource utilization.	However, it failed to integrate the cache coherence with ALUs for better performance in multicore systems.
[16]	Implementation of reversible ALU	HNG and Ferdkin gates	The model proficiently decreased the delay.	Yet, the research introduced additional latency and complexity.
[18]	Designing a reversible ALU by QCA	Reversible gates	It attained low cellular consumption, high speed, and low occupation area.	The model was highly sensitive to fabrication defects, thermal fluctuations, and external noise.
[15]	ALU with low hardware complexity	FS-GDI	The research consumed low energy.	Yet, it had unpredictable delays and power fluctuations.
[28]	Designing an optimal ALU	Reversible logic	It excellently minimized the problem of heat dissipation.	However, the model had a high propagation delay.
[5]	Low power and reduced transistor count magnetic ALU	Hybrid STT-MTJ/CMOS circuit	The research had reduced power dissipation, delay, and device count.	It had increased area overhead.

A comparative evaluation of the proposed and related methods is presented in Table 3. Here, while designing the ALU, the proposed E-FRIMA and Dw-FGCG excellently reallocated the data path, thus reducing resource utilization. However, the existing Hybrid New Gate (HNG) and Ferdkin gates introduce additional latency and complexity. Likewise, prevailing reversible gates are highly sensitive to fabrication defects, thermal fluctuations, and external noise. Similarly, the conventional Full Swing Gate Diffusion Input (FS-GDI) approach had unpredictable delays and power fluctuations. Similarly, the existing reversible logic has a high propagation delay. Equally, the area overhead was augmented by the prevailing hybrid Spin-Transfer Torque Magnetic Tunnel Junction (STT-MTJ)/CMOS circuit. Therefore, the proposed model is superior to existing methods.

5. Conclusion

Here, a novel E-FRIMA- and Dw-FGCG-enabled implementation of data path reallocation and silicon footprint minimization-aware 4-bit ALU is presented. Essential processes such as logic level reduction, data path reallocation, error minimization, clock gating, defect identification, and defect mitigation were performed in the proposed methodology to obtain a silicon-footprint-minimized ALU design. The proposed E-FRIMA achieved high area reduction (76%) and leakage power reduction (74%), which proved the effectiveness of the proposed model. Similarly, for clock gating, the proposed Dw-FGCG obtained high energy efficiency and switching activity reductions of 83% and 78%, respectively. Similarly, for defect identification, the proposed PS-GANFIS required less fuzzification and defuzzification times of 673ms and 730ms, respectively, thus proving its low time complexity. Thus, the proposed model achieved high reliability and trustworthiness. The comparative evaluation of the proposed Dw-FGCG and prevailing methods further highlights the effectiveness of the proposed model. The Dw-FGCG achieved a low clock-enable activity factor of 0.134 and a high energy efficiency of 83%, outperforming the existing PG, SBCG, FGCG, and LBCG methods, which exhibited higher activity factors and lower energy efficiency. Similarly, while designing the ALU, the proposed E-FRIMA and Dw-FGCG effectively reallocated the data path, reducing resource utilization compared to existing approaches, such as HNG and Ferdkin gates, reversible gates, FS-GDI, Reversible logic, and Hybrid STT-MTJ/CMOS circuits, which suffer from additional latency, fabrication sensitivity, unpredictable delays, high propagation delay, and increased area overhead. This comprehensive comparative analysis demonstrates that the proposed model consistently delivers superior performance, energy efficiency, and reliability, establishing it as a significant advancement over conventional ALU design methodologies. The proposed model failed to integrate the cache coherence with ALUs for better performance in multicore systems, even though it performed data path reallocation while designing ALUs to reduce resource utilization. Enhanced techniques will be developed in the future to integrate cache coherence with ALUs to further improve their performance in multicore systems.

Acknowledgment

The authors acknowledge the support from CMR University, BIT, Bengaluru, and REVA UNIVERSITY, India, for the facilities provided to carry out the research.

Data Availability

The labeled datasets used to support the findings of this investigation can be obtained from the corresponding author upon request.

References

[1] Seyed-Sajad Ahmadpour, Mohammad Mosleh, and Saeed Rasouli Heikalabad, "The Design and Implementation of a Robust Single-Layer QCA ALU using a Novel Fault-Tolerant Three-Input Majority Gate," *Journal of Supercomputing*, vol. 76, pp. 10155-10185, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[2] Hamza Mohammed Ridha Al-Khafaji et al., "Performance optimization of the Nano-Scale Carry-Skip Adder based on Quantum Dots and its Application in the Upcoming Internet of Things," *Optik*, vol. 287, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Alok Anand, Ivan Khokhlov, and Abhishek Anand, "Heterogeneous ALU Architecture -- Power-Aware System," *arXiv preprint*, pp. 1-8, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Shubham Anand, and S. Indu, "A Low Power and High Speed 8-bit ALU Design using 17T Full Adder," *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, pp. 514-519, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[5] Prashanth Barla, Vinod Kumar Joshi, and Somashekara Bhat, "A Novel Low Power and Reduced Transistor Count Magnetic Arithmetic Logic Unit Using Hybrid STT-MTJ/CMOS Circuit," *IEEE Access*, vol. 8, pp. 6876-6889, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[6] Muskan Bhusal, R. Rohith, and R. Sakthivel, "Single-Bit Fault-Detecting ALU Design using Reversible Gates," *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Vellore, India, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[7] Saroja S. Bhusare et al., "Optimized Reversible Arithmetic and Logic Unit," *International Conference on Communication and Intelligent Systems*, pp. 641-653, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Vuppala Chandralekha et al., "Design of 8 bit and 16 bit Reversible ALU for Low-Power Applications," *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)*, Greater Noida, pp. 477-480, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[9] Katyayani Chauhan, and Deepika Bansal, "Noise-Tolerant and Power-Optimized Ternary Combinational Circuits for Arithmetic Logic Units," *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 11, pp. 1-11, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[10] Sharvani Gadgil, and Chetan Vudadha, "Design of CNTFET-Based Ternary ALU Using a 2:1 Multiplexer-based Approach," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 661-671, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[11] Cissy Jose, T.D. Subash, and Simi P. Thomas, "FPGA Implementation of Dynamic Power and Area-Optimized Reversible ALU for Various DSP Applications," *Materials Today: Proceedings*, vol. 24, no. 3, pp. 2044-2053, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Ajay Kumar, Bobby Kumawat, and Neeta Pandey, "Design of 4-bit ALU using TEAM Memristor Model and CMOS Logic," *2020 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, India, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Dhanunjay Lachireddy, and S.R. Ramesh, "Power and Delay Efficient ALU using a Vedic Multiplier," *Advances in Electrical and Computer Technologies*, vol. 672, pp. 703-711, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[14] Konstantina Miteloudi et al., "PQ.V.ALU.E: Post-Quantum RISC-V Custom ALU Extensions on Dilithium and Kyber," *International Conference on Smart Card Research and Advanced Applications*, pp. 190-209, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Mohsen A.M. El-Bendary, and M. Ayman, "Various Nano-Scale Technologies: Lower Hardware Complexity of ALU Realizing Utilizing FS-GDI Approach in 45 and 130 nm," *International Journal of Applied Mathematics and Machine Learning*, vol. 14, no. 1, pp. 9-34, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Maliheh Norouzi, Saeed Rasouli Heikalabad, and Fereshteh Salimzadeh, "Reversible ALU using HNG and Ferdkin Gates in QCA Nanotechnology," *International Journal of Circuit Theory and Applications*, vol. 48, no. 8, pp. 1291-1303, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[17] Assylkhan Nurgali et al., "Realization of Arithmetic Logic Units Using Electro-Optic Microring Resonators in Photonic Circuits," *IEEE Access*, vol. 13, pp. 11021-11028, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[18] Saeed Mirzajani Oskouei, and Ali Ghaffari, "Designing a New Reversible ALU using QCA to Reduce the Occupation Area," *Journal of Supercomputing*, vol. 75, pp. 5118-5144, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Amrut Anilrao Purohit, Mohammed Riyaz Ahmed, and R. Venkata Siva Reddy, "Design of Area Optimized Arithmetic and Logical Units for the Microcontroller," *2020 IEEE VLSI Device Circuit and System (VLSI DCS)*, Kolkata, India, pp. 335-339, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[20] Asher Sajid et al., "An Optimized Hardware Implementation of a Non-Adjacent-Form Algorithm Using a Radix-4 Multiplier for Binary Edwards Curves," *Applied Sciences*, vol. 14, no. 1, pp. 1-18, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Maria Sapounaki, and Athanasios Kakarountas, "A Novel Low-power Neuromorphic Circuit based on Izhikevich Model," *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, Thessaloniki, Greece, pp. 1-4, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[22] Shubham Sarkar et al., "8-Bit ALU Design using m-GDI Technique," *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, Tirunelveli, India, pp. 17-22, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[23] Trapti Sharma, and Laxmi Kumre, "Energy-Efficient Ternary Arithmetic Logic Unit Design in CNTFET Technology," *Circuits, Systems, and Signal Processing*, vol. 39, pp. 3265-3288, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] Jitesh R. Shinde, Sanjeev Sharma, and Lipsa Dash, "An Optimization Design approach for Arithmetic Logic Unit," *International Conference on Intelligent Computing and Communication Technologies*, pp. 706-716, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[25] N. Subbulakshmi et al., "ALUSGDI: Low-Power Arithmetic Logic Unit-Based Sliced Processor using GDI and MGDI," *Measurement: Sensors*, vol. 28, pp. 1-9, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[26] Shaveta Thakral, and Dipali Bansal, "Novel Reversible ALU Architecture Using DSG Gate," *Ambient Communications and Computer Systems*, pp. 149-156, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[27] Jing Tian, Bo Wu, and Zhongfeng Wang, "High-Speed FPGA Implementation of SIKE based on Ultra-Low-Latency Modular Multiplier," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 9, pp. 3719-3731, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[28] Kamal K. Upadhyay et al., "Design and Performance Analysis of All-Optical Reversible Full Adder, as ALU," *Proceedings of the National Academy of Sciences India Section A - Physical Sciences*, vol. 90, pp. 899-909, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[29] Gaurav Verma, "Design and Analysis of ALU for Low-Power IOT-Centric Processor Architectures," *2020 Global Conference on Wireless and Optical Technologies (GCWOT)*, Malaga, Spain, pp. 1-5, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[30] Ankita Yadav, and Varsha Bendre, "Design and Verification of 16 Bit RISC Processor Using Vedic Mathematics," *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Pune, India, pp. 759-764, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[31] Chiraz Khedhiri et al., "A Self-checking CMOS Full adder in Double Pass Transistor Logic," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 2, pp. 1-4, 2012. [Google Scholar] [Publisher Link]

[32] Asheesh Shah, A. Mazyad, and A.K.Ramani, "Metric-Driven Framework for Processor Verification," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 2, pp. 1-4, 2010. [Google Scholar] [Publisher Link]

[33] Ismail Assayad et al., "Novel Architecture Prototyping Framework with Generic Property Verification for Sub-Architectures," *Engineering Letters*, vol. 29, no. 2, pp. 634-644, 2021. [Google Scholar] [Publisher Link]

[34] Rekha K. James et al., "Performance Analysis of Reversible Fast Decimal Adders," *Proceedings of the World Congress on Engineering and Computer Science*, San Francisco, USA, pp. 234-239, 2007. [Google Scholar] [Publisher Link]

[35] Abdullah Yildiz et al., "Crucial Topics in Computer Architecture Education and a Survey of Textbooks and Papers," *IAENG International Journal of Computer Science*, vol. 47, no. 3, pp. 404-419, 2020. [Google Scholar] [Publisher Link]

[36] Ajay Sudhir Bale et al., "Nanosciences Fostering Cross Domain Engineering Applications," *Materials Today: Proceedings*, vol. 43, no. 6, pp. 3428-3431, 2021. [CrossRef] [Google Scholar] [Publisher Link]