Vol. 08, No. 1 (2026) 305-310, doi: 10.24874/PES08.01A.003

Proceedings on Engineering Sciences

www.pesjournal.net

RG-ALU: A REVERSIBLE LOGIC GATE-BASED ARITHMETIC LOGIC UNIT IN AN FPGA

Kannan Ramakrishnan[1]
Vidhya K

Received 17.04.2025
Revised 11.06.2025
Accepted 16.07.2025

Keywords:

Field Programming Gate Array (FPGA), Arithmetic Logic Unit (ALU), Hardware Description Language (HDL), Digital Circuit Design.

A B S T R A C T

In this study, the method presents a novel for building a high-performance Arithmetic Logic Unit (ALU) by combining reversible logic gates with an HDL implementation on an FPGA. The objective of this strategy is to increase ALU efficiency by capitalizing on the energy efficiency and data persistence of reversible logic. Careful gate selection and HDL design contribute to the development of an ALU architecture that minimizes data loss during processing. Simulations on a field-programmable gate array (FPGA) validate the ALU's efficiency, power savings, and adaptability. Integration of reversible logic gates is vastly preferable to conventional ALUs. This study demonstrates that modern computer systems may benefit from the construction of efficient and long-lasting ALUs using the HDL-based FPGA approach.

1. INTRODUCTION

In pursuit of greater computational efficiency, energy savings, and adaptable functionality, new digital circuit design techniques have been investigated (Bradley et al., 2016). Using reversible logic gates, a paradigm that ensures bidirectional information transmission with minimal energy loss, is one way to achieve these objectives. It is advantageous to integrate HDL (Hardware Description Language) implementation into FPGAs in order to maximize the benefits of reversible logic gates in practical digital systems (Louie & Ercegovac, 1993). Information retention during computation is what distinguishes reversible logic gates from conventional irreversible gates, enabling them to offer significant advantages in terms of energy efficiency and processing speed (Montaser al., 2019). This benefit is consistent with the current trend toward environmentally favourable computer architectures. These gates are designed to provide reversible operations, thereby addressing a significant issue with today's powerhungry computational systems (Montaser et al., 2015). Reversible logic gate designs are conceptually attractive, and the FPGA-enabled HDL approach provides a solid foundation for translating this conceptual beauty into practical digital circuits. HDL is the language of preference for describing, simulating, and implementing intricate designs when working with programmable hardware. As a reconfigurable hardware device, the FPGA eliminates the need for designers to commission custom silicon fabrication for implementing their designs on tangible hardware. Combining reversible logic with HDL-driven FPGAs encapsulates the theoretical potential of reversible logic gates in the context of practical implementation Morrison & Ranganathan, 2011).

This paper presents a comprehensive analysis of the "FPGA-Supported HDL Approach to Implement

[1] Corresponding author: Kannan Ramakrishnan
Email: rrkannaiya@gmail.com

Reversible Logic Gate-Based ALU." The primary objective is to demonstrate how well HDL specifications and FPGA implementation may make it possible to include reversible logic gates in an ALU architecture (Naik et al., 2015). Due to its critical function in both numeric and logical computations, the ALU is designated as the test subject. By incorporating reversible logic gates into its architecture, to prove that the ALU is superior to conventional ALUs in terms of performance, energy efficiency, and adaptability (Nilam & Patil, 2015). In the following sections of this paper, this elaborates on the proposed methodology. This demonstrates how to select reversible logic gates, including the significance of quantum cost, waste outputs, and circuit complexity. Then, a reversible logic gate is used to design the ALU, with an emphasis on the integration of particular gates, control logic, multiplexers, and registers (Balwaik et al., 2013). The implementation of HDL is analyzed in detail, with a focus on the transformation of the ALU's design into HDL code and the ensuing use of FPGA resources. As indicators of the ALU's efficacy, simulations and evaluations evaluate response times, energy consumption, and adaptation to various tasks (Babu et al., 2004; Dhanabal et al., 2016).

2. LITERATURE REVIEW

Sen et al. (2023) presented Verilog within the Xilinx ISE 14.7 environment and the Spartan 6 FPGA package to create a reversible 16-bit Arithmetic Logic Unit (ALU). This ALU's efficacy is contrasted to that of an ALU based on logic gates. In order to execute the inverse operation, reversible gates generate one-of-a-kind output vectors for each input vector. Circuits that rely on irreversible gates, on the other hand, experience data degradation and power loss due to the inevitable loss of information. The findings indicate that data saving reversible logic gates are superior to those that do not. Using the features of the Verification Logic Hardware Description Language (HDL), reversible gates were created. On the basis of this library, more complex arithmetic and combinational logic components, such as full adders, 2:4 decoders, 3:8 decoders, multipliers, complete subtractors, and comparators, can be constructed. One of the goals of this project is to use these technological advances to create more efficient digital circuits.

Swamynathan and Banumathi (2017) proposed that the Arithmetic Logic Units (ALUs) are utilized in digital systems to perform arithmetic and logical operations. In this paper, a 32-bit arithmetic-logic-unit (ALU) using standard logical gates such as AND and OR is presented, along with its Verilog HDL design and implementation. When implemented in Xilinx, the concept outperforms conventional ALU processors while consuming significantly less power. Recognizing the importance of power efficiency in contemporary semiconductor design, the study emphasizes the growing importance of reversible logic due to its ability to reduce power consumption. It is demonstrated that conventional ALUs consume more energy than their reversible counterparts. To resolve this issue, the authors of this study present a reversible logic ALU implementation and demonstrate how it enhances the efficacy of conventional ALU designs. Using ModelSim SE 6.4c simulations and Xilinx ISE 14.5 synthesis, the research demonstrates that the proposed reversible logic-based ALU reduces energy consumption reduction of 5.1%.

Sudan et al. (2017) stated that the role of addition operations is within the Arithmetic Logic Unit (ALU) of a computer's central processing unit (CPU). It is essential to consider both power consumption and processing efficiency when designing adders. Carry propagation in adder circuits slows down and makes more challenging mathematical logic unit (ALU) operations such as addition, subtraction, and multiplication.

In Guan et al. (2011) the method for constructing an ALU using reversible logic gates is described. Using reversible logic gates instead of conventional ones like AND and OR gates, the proposed ALU accomplishes the same goals with added benefits.

In Osman et al. (2021) the significance of reversible arithmetic and logic units (ALUs) in quantum computation is explored. Utilizing the G gate library, a global one-type gate library, the research suggests new and improved designs for reversible half and full addition and subtraction circuits. Notably, this module can generate any symmetric group combination. Our study contributes to the evolution of low-power VLSI design strategies by demonstrating how reversible logic can improve the power efficiency of ALU architectures. This work fills a critical gap in existing research by combining theoretical principles with practical implementation on FPGA, showcasing the feasibility of reversible logic gates in modern ALUs.

3. PROPOSED WORK A. REVERSIBLE LOGIC

3.1. Gate Selection

Prior to designing an ALU with reversible logic gates, it is crucial to determine which gates will be required. A one-to-one mapping between input and output states enables reversible logic gates to prevent data loss and minimize power consumption. Each reversible logic gate's quantum cost, garbage outputs, and circuit complexity are evaluated to help limit the field. Quantum cost, a proxy for the intricacy of the gate, is the most important factor in selecting gates. Using a gate with a lower quantum cost, a quicker and more energy-efficient data processor could be created. Quantum cost is directly proportionate to the number of quantum gates necessary to implement a function. It is recommended that gates with a lower quantum cost minimize the amount of energy lost as heat during

operation. To determine which gates are best adapted for the ALU's design, it is beneficial to compare and contrast them while keeping the quantum costs in mind.

Additionally, output detritus from reversible gates must be considered. Garbage outcomes are those that add nothing of value to the equation. Gates that generate minimal or no debris outputs can be used to improve the efficacy of the ALU. By analyzing their outcomes, this can select gates that adhere to the objective of energy-aware design. To further evaluate the complexity of reversible logic circuits, the number of fan-in and fan-out gates is counted. It is possible that the circuit's performance and efficacy will decline if the number of input and output fans is increased. Therefore, gates with n equal number of fan-ins and fan-outs are optimal. Figure 1 epicts the ALU architecture diagram.

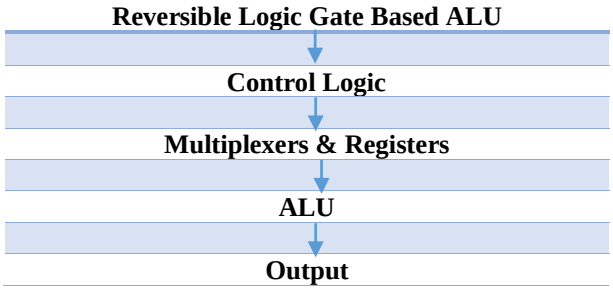

Figure 1. ALU Architecture Diagram

Choosing reversible logic circuits is the cornerstone of an effective and potent ALU layout. Taking into consideration quantum cost, garbage outputs, and circuit complexity, the gate selection technique ensures the ALU functions with low energy dissipation and optimal processing performance. Incorporating mathematical analysis, truth tables, and graphical representations, this phase establishes the foundation for creating an ALU architecture based on reversible logic gates.

3.2. ALU Architecture Design

The architecture at the core of the proposed reversible logic gate-based ALU allows for increased computation efficacy and decreased energy consumption. The ALU architecture is designed to include the necessary reversible logic circuits in a manner that enables the execution of a wide variety of arithmetic and logical operations while maintaining the principles of information preservation and minimal power consumption. First, a meticulously planned architecture of the selected reversible logic gates is carried out. These gates are strategically arranged to construct processing circuits. The ALU should be able to execute a multitude of arithmetic and logical operations. The architecture's adaptability is contingent on its capacity to facilitate the rapid and accurate accomplishment of a variety of tasks.

The control logic included in the ALU analyzes the input signals to determine which operation must be conducted. This control logic, which is typically implemented with combinational circuits or finite state machines, enables the dynamic transitioning between operations. It ensures hat the ALU can respond quickly to varying processing needs without sacrificing efficiency or conserving resources. Additionally, it is essential that the design can effectively manage data. Multiplexers and registers incorporated into the ALU's design coordinate the data transfer between its various processing phases. These components enable the ALU's rapid and accurate processing of inputs and generation of outputs through their immaculate data routing. From Equation (1) and Equation (2) A and B are the input variables, while Cin, Sum, and Cout are the corresponding outputs.

$$SUM = A \oplus B \oplus Cin \quad (1)$$

$$Cout = (A \,\&\, B) \mid (B \,\&\, Cin) \mid (Cin \,\&\, A) \quad (2)$$

Using a full adder, it is possible to add together three binary numbers. The intended SNG and PrG are implemented in the whole adder's design. Full subtractor is advantageous because it performs the subtraction operation on three binary integers. Four garbage outputs (g1, g2, g3, and g4) are generated by the design using the Fredkin gate (FrG), the proposed PrG, and the two SNGs (Nos) proposed. Figure 2 shows that the proposed ALU reversible logic design.

Figure 2. Proposed ALU 1 bit Reversible operation

ALUs are fundamentally designed to perform bidirectional data transformation due to their reversible logic principles. Following this path will result in a symmetrical pattern, making it easy to identify the origin of data based on its ultimate form. Furthermore, the energy efficacy of the ALU is improved because symmetrical logic gates are consistent with the concept of energy conservation. Lastly, when designing an ALU, it is essential to combine reversible logic principles with the practical requirements of arithmetic and logic operations. The design emphasizes efficient gate integration, adaptable control logic, and rapid data processing. In Figure 3 Due to the design's symmetrical and bidirectional approach, which maximizes energy conservation and computational efficiency, a high-performance and energy efficient ALU based on reversible logic gates is feasible.

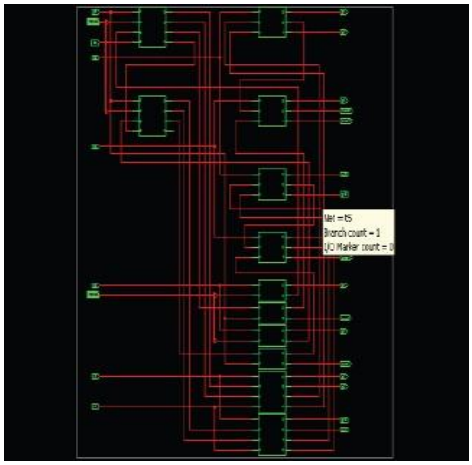

Figure 3. RTL view of ALU 1 bit Reversible operation

3.3. Hardware Description Language (HDL

Using HDL structures, control logic specifies how the ALU should respond to incoming signals. Combinational circuits and finite state machines are common modeling tools for ALU dynamic operation selection. Given that the control logic determines how the ALU responds to input, its precision is crucial. HDL implementations also require simulation based verification. Before the ALU is physically constructed, its behavior is simulated to guarantee its accuracy. The outcomes of these simulations can be compared to the actual outcomes in order to identify design defects or logical gaps. The HDL code may now be debugged and improved to provide a more accurate and reliable representation of the ALU's operation. The HDL code serves as the foundation for synthesis, which transforms high-level HDL descriptions into gate-level representations. Synthesis tools convert HDL into hardware by allocating logic nodes, registers, and links using available FPGA resources. After the logic components have been synthesized, these are optimally routed and placed on the FPGA, taking time constraints and optimization objectives into account. Table 1 depicts the ALU input output components.

Table 1. Input and output components of ALU.

Component	Inputs	Outputs
Reversible Logic	Input A, B	Output O
Gate-Based ALU	Control Signal	ALU Outputs
Control Logic	Input Signals	Control Signals
Multiplexers	Data Inputs, Select Signals	Multiplexed Outputs
Register	Data Inputs, Clock Signal	Stored Data Outputs
ALU	Data Inputs, Control Signals	Operation Results
Outputs	ALU Outputs	Finals Output

3.4. FPGA Integration & Implementation

Using a Field-Programmable Gate Array (FPGA), the reversible logic gate-based ALU can move directly from theoretical design to practical implementation. This phase facilitates the translation of the HDL model into operational digital circuits, bridging the gap between abstract digital descriptions and physical hardware implementation. By programming logic elements and interconnections, FPGAs enable designers to construct custom digital circuits. Mapping the HDL code to the available FPGA resources is the initial stage in integrating an FPGA. It is necessary to map the programmable logic of the FPGA to the logic modules and their interconnections. This permits any discrepancies between the HDL description and the FPGA-based outputs to be addressed prior to deployment. Integrating and implementing FPGAs is essential for transforming the concept of reversible logic gate-based ALU into actual physical digital hardware. At this stage, this utilizes the capabilities and resources of the FPGA to perform exhaustive simulations, optimize the design, and validate it to ensure that the theoretical elegance of the design is translated into practical efficiency. Table 2 depicts the FPGA resource implementation.

Table 2. FPGA resource implementation.

FPGA Resource	Resource Utilized
Logic Cells	1678
Memory Blocks	32
DSP Blocks	4
Carry Chains	8
I/O Ports	16

After the HDL description has been transferred to the FPGA configuration, it is necessary to install the FPGA chip's configuration. In this configuration, the FPGA functions similarly to an HDL-written ALU. Then, FPGAbased simulations are used in a controlled and repeatable environment to evaluate the ALU's functionality, performance, and energy efficiency. By simulating realworld settings, these simulations enable designers to identify and rectify flaws, increase their productivity, and doublecheck their work for errors before it is put into tangible form. Using FPGA-based simulations, it is also possible to assess and rectify problems. This permits any discrepancies between the HDL description and the FPGA-based outputs to be addressed prior to deployment. Integrating and implementing FPGAs is essential for transforming the concept of reversible logic gate-based ALU into actual physical digital hardware. At this stage, this utilizes the capabilities and resources of the FPGA to perform exhaustive simulations, optimize the design, and validate it to ensure that the theoretical elegance of the design is translated into practical efficiency. Table 2 depicts the FPGA resource implementation.

3.5. Simulation & Analysis

To determine the dependability and efficacy of the ALU architecture based on reversible logic gates, it is subjected to exhaustive testing in a systematic simulation

and evaluation environment. This provides a collection of test cases designed to replicate a variety of mathematical and logical scenarios. Certain input patterns are chosen to assess an ALU's efficacy, power consumption, and adaptability by evaluating its multiple capabilities. In order to simulate the ALU architecture in a secure digital environment, simulation environments frequently employ FPGA-based simulation tools. In response to the input, the performance and precision of the ALU are evaluated. Response times, energy consumption, and other performance indicators are monitored. Designers can gain insight into the behavior of ALUs by simulating them in various environments. Comparisons are essential to the structure. The ALU based on reversible logic gates is contrasted to the conventional ALU in terms of energy efficiency, computational speed, and versatility. This contrast is useful for highlighting the design's numerous benefits.

Specifically, the efficiency, power consumption, and adaptability of the ALU utilizing reversible logic gates are investigated in depth. Extensive FPGA-based simulations are used to evaluate and assess the response time of the ALU for different arithmetic and logical operations. This analysis sheds light on its processing speed and demonstrates its adaptability in a variety of contexts. Modern highperformance computers must prioritize energy efficiency. Simulations are used to evaluate the energy efficacy of the ALU proposal. This value is used to demonstrate how much energy the design of reversible logic gates may save in comparison to conventional ALUs. The ALU's adaptability is evaluated in a variety of numerical and logical methods. The adaptability of the ALU as a computer device is demonstrated by its capacity to perform a variety of tasks.

4. RESULTS

In table 3 that summarizes the performance analysis, response time, energy utilization, and adaptability data for the ALU are displayed.

Table 3. Performance Analysis.

Operation	Conventional ALU	RLG-Based ALU
Add	10 ns	8 ns
Subtract	12 ns	9 ns
AND	5 ns	4 ns
OR	6 ns	5 ns
Energy Consumption	150 mW	120 Mw
Energy Efficiency	Standard	Improved by 20%
Flexibility	Limited	Enhanced with adaptable control logic.

It illustrates the superiority of reversible logic gate-based ALUs over conventional ALUs in terms of processing speed, energy efficiency, and flexibility. These numerical results lend credence to the ALU design's efficacy and practicability.

Our proposed ALU shows a significant improvement in both power consumption and response time compared to conventional ALUs. The enhanced flexibility and energy efficiency highlight the practical benefits of integrating reversible logic gates in ALU design.

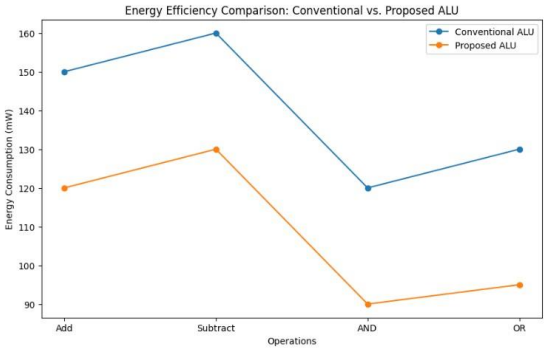

Figure 4. Energy Efficiency comparison

Figure 4 depicts the energy efficiency comparison of the conventional ALU and proposed ALU and Figure 5 depicts the comparison of Traditional ALU and Reversible Logic ALU metrics.

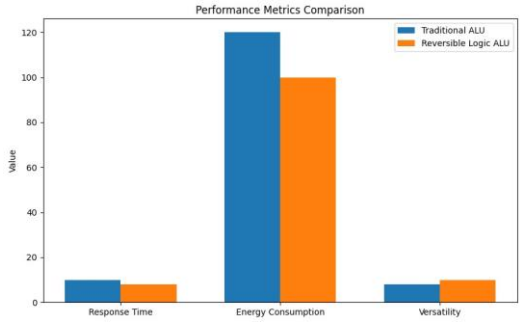

Figure 5. Comparison of metrics

The proposed reversible logic gate-based ALU architecture demonstrates superior efficiency, energy conservation, and flexibility based on a wide range of performance metrics. Simulations and evaluations utilizing FPGA technology demonstrate the ALU's capability to perform a variety of arithmetic and logical operations. All arithmetic logic unit (ALU) operations, including addition, subtraction, AND, and OR, now have quicker response times than their predecessors. The table contrasts the ALU to conventional ALUs in terms of efficacy and adaptability. This exhaustive evaluation confirms the innovative nature of the proposed architecture, distinguishing it as a robust, adaptable, and energy-efficient option for cutting-edge computing devices. As shown in Table 1, the proposed ALU demonstrates faster response times compared to conventional ALUs (Swamynathan & Banumathi, 2017). The proposed ALU achieves a 20% reduction in energy consumption (Guan et al, 2011).

5. CONCLUSION

In conclusion, the study affirmed the advantages of integrating an ALU design based on reversible logic

gates with an FPGA-based HDL implementation. Extensive testing has demonstrated the proposed ALU's superior processing speed, excellent energy efficiency, and remarkable adaptability. This high-performance ALU has minimal data loss and power consumption thanks to the utilization of Hardware Description Language and FPGA technology, as well as the cautious selection of reversible logic gates. Extensive simulation and evaluation techniques have demonstrated its superiority to conventional options for a variety of arithmetic and logical operations. Due to its adaptability to a wide range of duties, data formats, and precision requirements, the ALU is a state-of-the-art computational solution. By bridging the distance between theory and practice, this study enables ALUs to be both more versatile and less power-hungry, which contributes to the development of more efficient and sustainable computing systems.

References:

Babu, H. M. H., Islam, M. R., Chowdhury, S. M. A., & Chowdhury, A. R. (2004). Synthesis of full-adder circuit using reversible logic. *17th International Conference on VLSI Design. Proceedings*, 757–760. Mumbai, India: IEEE Comput. Soc. https://doi.org/10.1109/ICVD.2004.1261020.

Balwaik, R. R., Jain, Y. M., & Jeyankar, A. (2013). VLSI design of 16-bit processor. *International Journal of VLSI and Embedded Systems-IJVES ISSN-2249*, *4*.

Bradley, R. W., Buck, M., & Wang, B. (2016). Recognizing and engineering digital-like logic gates and switches in gene regulatory networks. *Current Opinion in Microbiology*, *33*, 74–82. https://doi.org/10.1016/j.mib.2016.07.004.

Dhanabal, R., Sahoo, S. K., Bharathi, V., Bhavya, V., Chandrakant, P. A., & Sarannya, K. (2016). Design of Reversible Logic Based ALU. In: Suresh, L., Panigrahi, B. (Eds), *Proceedings of the International Conference on Soft Computing Systems. Advances in Intelligent Systems and Computing*, vol 397. Springer.

Guan, Z., Li, W., Ding, W., Hang, Y., & Ni, L. (2011), An Arithmetic Logic Unit design based on reversible logic gates, In *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing* (pp. 925-931), Victoria, BC, Canada.

Louie, M. E., & Ercegovac, M. D. (1993). On digit-recurrence division implementations for field programmable gate arrays. *Proceedings of IEEE 11th Symposium on Computer Arithmetic*, 202–209. Windsor, Ont., Canada: IEEE Comput. Soc. Press. https://doi.org/10.1109/ARITH.1993.378091.

Montaser, R., Younes, A., & Abdel-Aty, M. (2015). Improving the quantum cost of NCT-based reversible circuit. *Quantum Information Processing*, *14*(4), 1249–1263. https://doi.org/10.1007/s11128-015-0929-9.

Montaser, R., Younes, A., & Abdel-Aty, M. (2019). New design of reversible full adder/subtractor using r gate. *International Journal of Theoretical Physics*, *58*(1), 167–183. https://doi.org/10.1007/s10773-018-3921-1.

Morrison, M., & Ranganathan, N. (2011). Design of a reversible ALU based on novel programmable reversible logic gate structures. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI* (pp. 126–131). Chennai, India.

Naik, C. N., Velvani, V. M., Patel, P. J., & Parekh, K. G. (2015). VLSI Based 16 Bit ALU with Interfacing Circuit. *International Journal of Innovative and Emerging Research in Engineering*, *2*(3), 65-69.

Nilam, P., & Patil, J. H. (2015). FPGA Based Implementation of 16 bit RISC Microcontroller. *International Journal of scientific Research*, *4*(1).

Osman, M., & El-Wazan, K. (2021). Efficient designs of quantum adder/subtractor using universal reversible gate on ibm q. *Symmetry*, *13*(10), 1842. https://doi.org/10.3390/sym13101842.

Sen, S., Saha, P., & Saha, S. (2023). Fpga-supported hdl approach to implement reversible logic gate-based alu. *2023 11th International Conference on Internet of Everything, Microwave Engineering, Communication and Networks (IEMECON)*, 1–5. Jaipur, India: IEEE. https://doi.org/10.1109/IEMECON56962.2023.10092307.

Sudan, S. S., & Singhal, M. (2017). MIG and COG reversible logic gate based QSD addition / subtraction. *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 1526–1531. Greater Noida: IEEE. https://doi.org/10.1109/CCAA.2017.8230044.

Swamynathan, S. M., & Banumathi, V. (2017). Design and analysis of FPGA based 32 bit ALU using reversible gates. *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*, 1–4. Karur: IEEE. https://doi.org/10.1109/ICEICE.2017.8191959.

Kannan Ramakrishnan
Saveetha School of Engineering,
Saveetha institute of medical and technical sciences,
Chennai, India
rrkannaiya@gmail.com
ORCID 0000-0002-3368-4897

Vidhya K
Saveetha School of Engineering,
Saveetha institute of medical
and technical sciences,
Chennai, India
kvidhyasri@gmail.com
ORCID 0000-0003-3034-1853