# Development of Comparative Design of Reversible and Irreversible 32-BIT ALUs

**P. Gopi Krishna[1], Penikalapati Manoj Kumar[2], Shaik Madeena Bibi[3], Mogili Venkateswararao[4], Sanghamu Gayatri[5]**

**Department of ECE, Vignan's Institute of Information Technology, Visakhapatnam, Andhra Pradesh, India.**

**DOI: https://doi.org/10.51583/IJLTEMAS.2026.15020000052**

**Received: 21 February 2026; Accepted: 27 February 2026; Published: 12 March 2026**

## ABSTRACT

As technology increases everyone wants more features with in a small size electronic gadget, to make them smaller, quicker, more compact and increasing integration density we need to decrease the size of the transistors but due to the decrease in size challenges like power efficiency and heat management becomes major concerns in VLSI design.

Traditionally we are using the irreversible gates in digital circuits but during digital operations input information losing which directly contributes the energy dissipation according to Landauer's principle. But Reversible gates make every output corresponds to unique input and prevents the information loss thereby reduces the power dissipation. In this work, we compared a 32-bit Arithmetic Logic Unit (ALU) designed with both irreversible logic gates and reversible ALU constructed with Peres gate.

The reversible design has a Quantum Cost of 384 and produces 128 garbage outputs. Both ALUs were coded in Verilog and implemented on Xilinx Artix-7 FPGA. We evaluated performance of ALUs based on theoretical metrics and actual hardware performance metrics. The reversible ALU shows a significant performance by reducing power dissipation to 70mW compared to the 211mW of the irreversible ALU.

However, its latency was slightly higher at 22.737 ns compared to irreversible design latency(20.214ns) due to the routing overhead. our analysis indicates that reversible logic is very fast in logic implementation but delay is mainly due to the routing overhead, in this case there were 128 garbage outputs on FPGA. This study demonstrates that reversible logic uses lower energy compared to conventional designs by careful handling of routing and I/O complexity.

**Keywords:** Reversible Logic, Arithmetic Logic unit (ALU), Fredkin Gate, Feynman Gate, Toffoli Gate, Peres Gate, Irreversible ALU, Quantum Cost, Garbage Outputs, Verilog HDL.

## INTRODUCTION

Power and heat management becomes major concern in modern VLSI technologies due to the continuous rise in package density and operating speed. According to Moore's law the number of transistors on a chip doubles for every two years. Increasing transistor counts makes more frequent switching events due to that dynamic power consumption increases which increases both power consumption and thermal outputs.

In most of the digital circuits irreversible logic has been using now a days. In such circuits, a part of the information is always removed during computation. For example, irreversible gates reduces multiple input bits into fewer outputs and causing information loss during each operation.

Landauer's principle states that losing a single bit of information produces a minimum energy loss of kTln2.In overall operations of today's processors billions of bits per second erasing and contributing a large power consumption. This is the major reason for people trying to create energy-efficient systems.

Reversible logic is an alternative way that can be (is hoped) a solution to such energy loss. In reversible logic for every output state there is exactly one unique input present. Then there is no information loss such that power dissipation decreases.

Reversible computation is made up of so-called reversible gates, for example Feynman gates, Toffoli gates and Peres Gates. Such gates are voltage driven and are capable of low energy computation. The Arithmetic Logic Unit (ALU) is one of the most critical part of any Processor, hence it can be a right candidate for testing the possible implementation of reversible logic in real hardware.

Building a reversible 32-bit ALU requires choosing the right gates and keeping track of both theoretical metrics (like quantum cost and garbage outputs) and hardware metrics (such as power, delay, and FPGA resources).

Reversible ALUs have already been proposed by previous researchers, but they mainly are theory proposals. Only a handful of works can be found that present both the theoretical costs as well as the practical hardware performance of a reversible ALU designed and synthesized on an FPGA.

This fundamental gap is addressed in this paper by a detailed comparison using 32-bit reversible ALU design. The major contributions include: Optimized Peres-gate–based ALU design with Quantum Cost of 384 and 128 garbage outputs were generated. An FPGA prototype that highlights the high resource usage, including 233 I/O pins.

Multi-parameter performance comparisons are done between: Power consumption of 70mW, Delay 22.737ns. The energy efficiency of 1.592pJ in per operation compared to the irreversible design, and also metric such as Power Dealy Product (PDP) and Quantum Cost-Delay-Power Product (QCDPP) was evaluated.

Overall, this work shows that reversible logic comes with a trade-off: it lowers power consumption but increases delay, mostly due to extra garbage outputs and more complex routing.

## LITERATURE SURVEY

The development of Reversible Arithmetic Logic Units (RALUs) is mainly due to information loss in irreversible computation, established by Landauer's Principle.

Theoretical works proved that general computation can be performed reversibly using concepts like reversible Turing machines and uncomputation , and this method can be done using basic reversible gates like Toffoli and Fredkin gates. . These concepts are also related to the quantum computing and enhanced by the efficient circuit-synthesis methods based on gates such as Peres and Fredkin .

More recent research has been considering practical FPGA-implementations, where some studies of 32-bit RALUs report reported performance improvement, approximately 1.6x in dynamic power , and substantial in reduction in delay (48.91%) and area (34) in comparison to traditional irreversible designs .

This confirmation in current tools and techniques confirms the basic concept of reversible logic, at the cost of the introduction of ancilla and garbage signals, and highlights the need for further experimentation to optimize resource utilization, reduce overhead, and improve practical implementation efficiency in real-time systems.

### Reversible Logic

The below figure shows a Reversible Logic gate which performs a reversible logic operation by obtaining the no. of inputs and outputs are equally. In the reversible logic operation, there is no loss of information it means there will be zero production of heat. For obtaining the low power usage circuits the reversible logic gates should have,

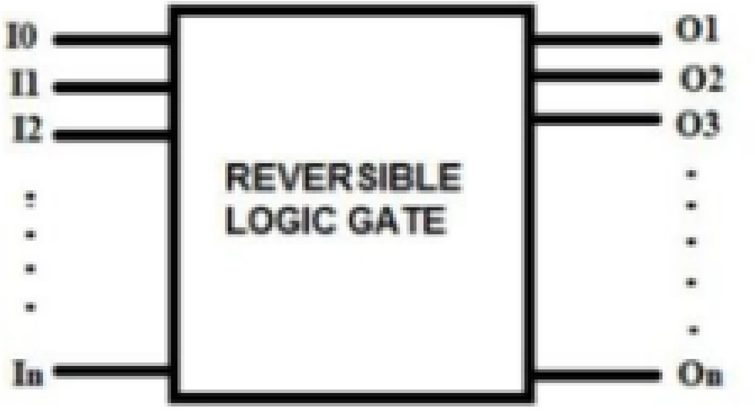

**Figure 1: Reversible Logic**

1. The design should use minimum reversible blocks such that overall gate count should be low.

2. The design should generate only a limited no. of redundant outputs because excess garbage signals reduce efficiency.

3. Delay should be minimum.

4. An effective reversible implementation also aims to minimize the total quantum cost associated with the gate operations.

### A. Feynman Gate

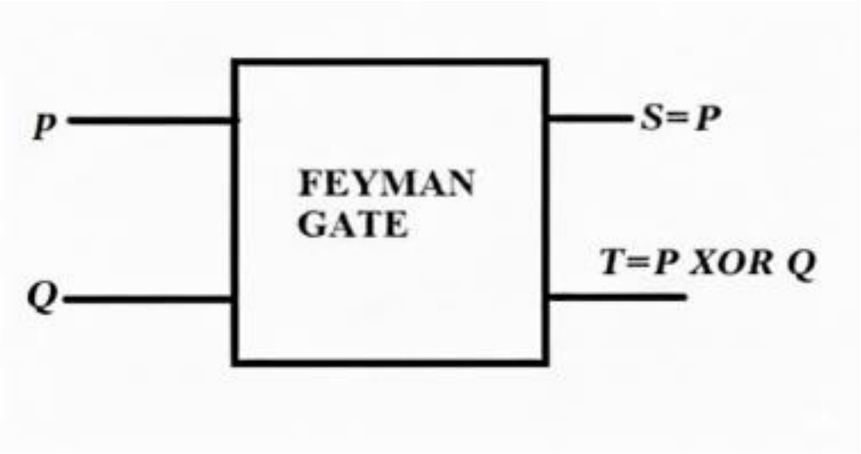

**Figure 2: Feynman Gate**

The Feynman gate is a reversible structure commonly applied for signal duplication and XOR-based transformations. For the Feynman gate the outputs and inputs are represented as Inputs (P, Q) and Outputs (S, T). The 2x2 Feynman gate outputs are given as

$S = P$ (3.1)

$T = P \oplus Q$ (3.2)

The quantum cost of the Feynman gate is 1. The Feynman gate is used in lot of applications as it has low cost.

### B. Peres Gate

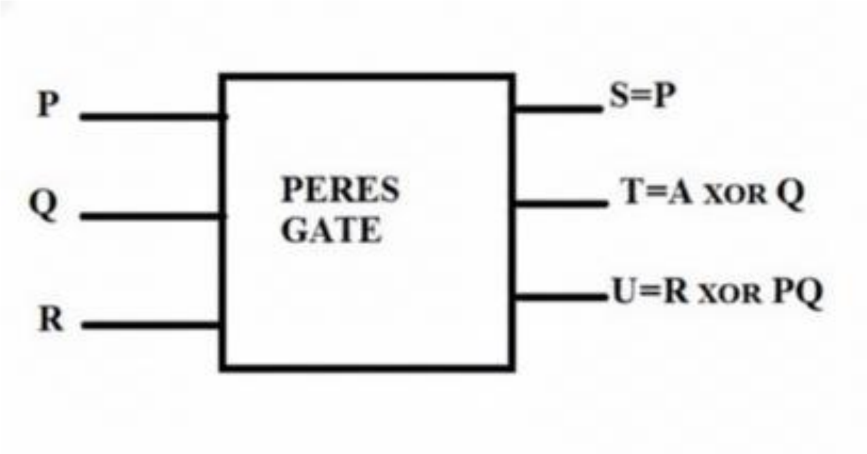

**Figure 3: Peres Gate**

The Peres gate is a reversible gate that combines XOR and controlled-AND behaviour within a single structure. The inputs and outputs are represented as Inputs (A, B, C) and Outputs (X, Y, Z). The output of the Peres gate is given as

$S=P$ (3.3)

$T=P \oplus Q$ (3.4)

$U=R \oplus PQ$ (3.5).

### C. Toffoli Gate

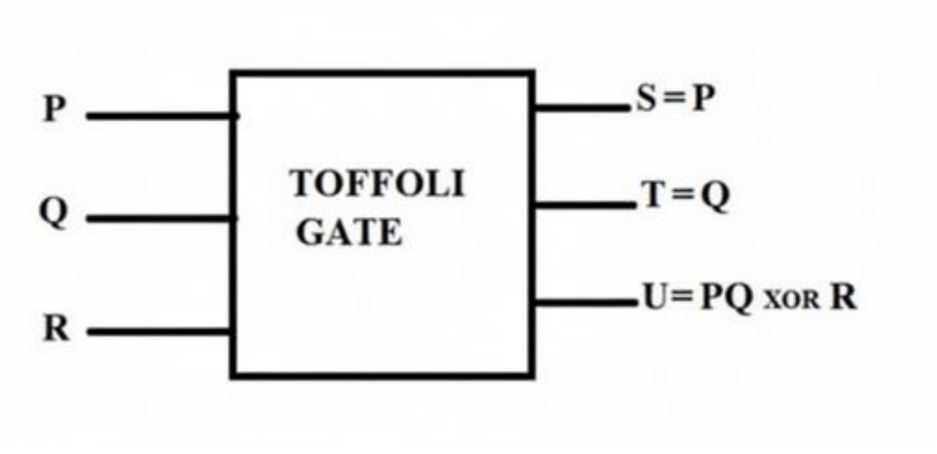

**Figure 4: Toffoli Gate**

It is a popular reversible logic block that works like a double-controlled NOT gate. It is often used to build many reversible circuits. The inputs and outputs are represented as Inputs (P, Q, R) and Outputs (S, T, U), and its output values are given as

$S=P$ (6)

$T= Q$ (7)

$U= PQ \oplus R$ (8)

The output of the Toffoli gate at U is obtained as AND operation when the input R is given as zero. It has quantum cost of 5.

### D. Fredkin Gate

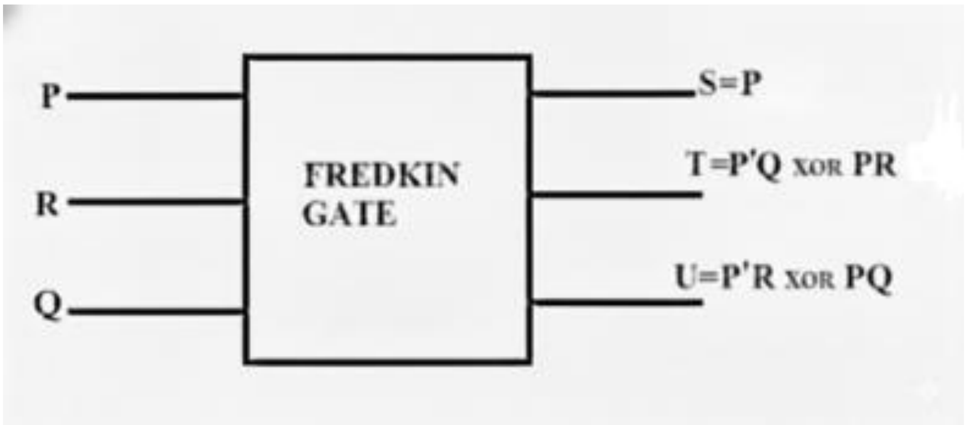

Figure 5: Fredkin Gate

The Fredkin gate is a reversible gate and the vectors of the Fredkin gate is represented as Inputs (P, Q, R) and Outputs (S, T, U) respectively. The outputs are given as

S=P (9)

T = P'Q ⊕PR (10)

U = P'R ⊕ PQ (11)

The output of the Fredkin gate at U is obtained as AND operation when the input Q is given as one. The Fredkin gate has quantum cost of 5.

### ALU Using Irreversible Logic Gates

Most digital processors depend on traditional 32-bit ALUs which are implemented using standard irreversible logic components. This design is constructed on the conventional logic gates of AND, OR, XOR, NANDs in addition to full adders and multiplexers [10-12], and all these are irreversible. Irreversible gates compress multiple input bits into fewer output bits, which makes information loss and this loss increases power dissipation.

The irreversible ALU, described in this paper, provides a large range of arithmetic and logic functions found in general-purpose computation. High performance combinational and sequential building blocks are able to meet arithmetic functions including addition, subtraction, increment/decrement and logic functions AND, OR, XOR and shift functions.

The unit of arithmetic is the 32-bit ripple-carry adder, and it is based on the fact that the large number of functional outputs is generated by multiplexer.

The non reversible ALU is very cost effective in terms of FPGA implementation. The circuit can be used to ensure low latency and easy routing using logic mapping through the traditional LUT based architecture. Therefore, irreversible approach is less delay and hard-ware expensive compared to the reversible implementations. There is no other constraint such as garbage outputs or ancilla bits that make routing easier and reduce the area.

However, the irreversible ALU inherently requires more power in so far as the loss of information is concerned in the process. Although this design is good in delay and throughput, it is a throwback to the classical computation model that is constrained by power. The irreversible ALU applied in this work is the benchmark, which is intended to be compared to the reversible 32-bit ALU used in the fair comparison with the delay, resources requirement, power consumption, and performance gain.

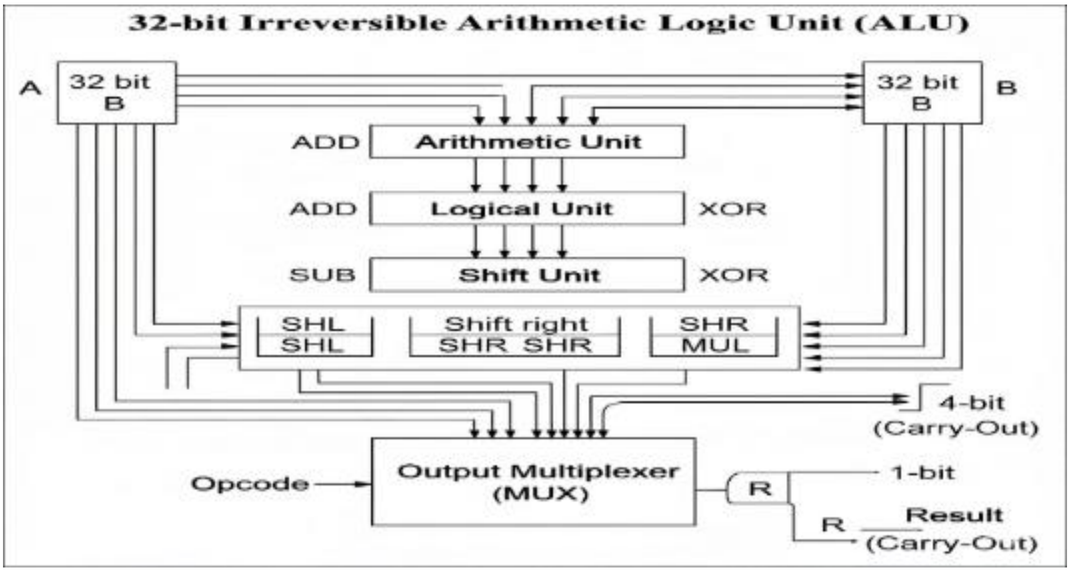

**Figure 6: 32-bit Irreversible Arithmetic**

**ALU with Reversible Gates**

The 32-bit reversible ALU is shown to be a custom architecture, optimized for low Quantum Cost and it follows the fundamental principle of ultra-low-power reversible logic. In opposite to ordinary irreversible ALU, each operation in a reversible ALU should have one-to-one routing between its input(s) and output(s).

The ALU is implemented completely with reversible gates, where we used Peres Gate (PG) as the basic element because it performs high performance with very low cost. The target of the architecture is to minimize the three theoretical parameters behind reversible circuits: Quantum Cost (QC), Garbage Outputs (GO) and Ancillary Inputs (AI).

**Reversible Full Adder (RFA) Implementation**

The 32 bits adder implemented with 32 RFA cells. Two cascaded Peres Gates are utilized in each RFA. These two gates produces Sum (S) and Carry-out (Cout), With maintaining reversibility. The Quantum Costs of the 1-bit RFA

are all equal to 8, it generates 2 Garbage Outputs and require 1 Ancillary Input. Due to this optimal design, the overall Quantum Cost of the complete 32-bit reversible ALU is just 384 which is very small as compared to others.

**Reversible Logic Unit Implementation**

This 32-bit reversable ternary unit is formed by 32 same and independent single-bit logic units. The synthesis of the basic logic functions, AND, OR and XOR is based on one Peres Gate per block. Selecting the appropriate PG output and fixing one of AI's (inputs) to '0' or '1', controlled logic operations are also allowed. Every logic bit is comprised of 4 QC, therefore the overall number of QCs for the complete 32 bits block is 128.

**Reversible ALU Architecture**

The previous ALU is implemented by running the adder and the logic unit in parallel. Both logics generated garbage signals of their own:

Arithmetic unit garbage = 64 bits

Logic unit garbage = 64 bits

Total Garbage Outputs = 128bits

All of these garbage outputs must be somehow sent out to the top level, bringing us up to 233 I/O. This additional routing is the primary reason for the hardware implementation delay of 22.737 ns.

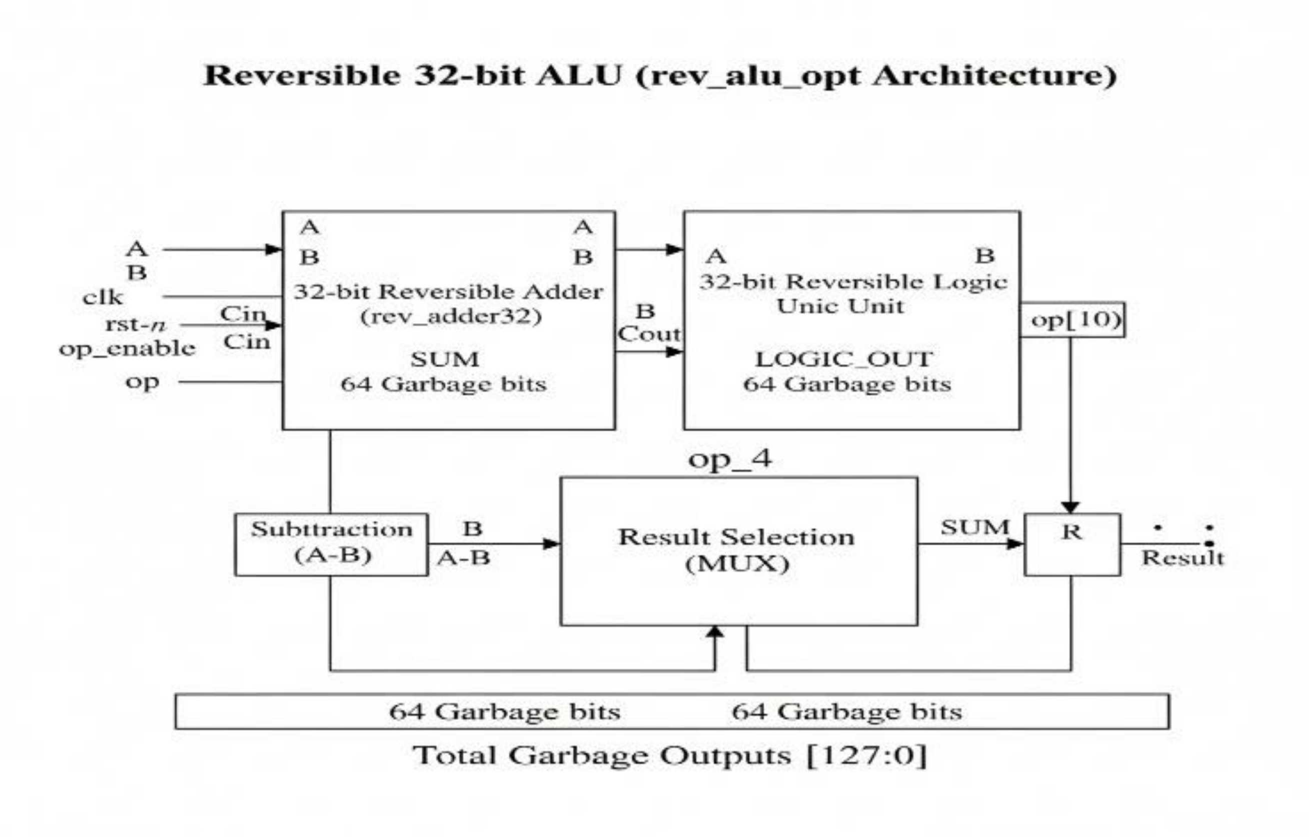

**Figure 7: Reversible 32-bit**

**Operation Selection:**

The output of the ALU is selected by using a selection block that resembles a multiplexer and the result that is selected is sent to the main system output (R). Because the whole design should be reversible, no signals can be discarded hence all outputs including the undesired ones should be retained.

## SIMULATION RESULTS AND DISCUSSION

Two designs of ALU were analysed in this project, one with conventional ALU and the other one with the reversible ALU. They were both put to test on the same FPGA board to ensure fair comparison of the results. The reversible logic needed fewer hardware units as it had fewer logic units and did not have the need to use the DSP blocks, which makes it smaller and energy-conscious.

In terms of speed, the regular ALU achieved higher results with a delay of 20.214 ns versus 22.737 ns of the reversible one. The reversible ALU however was seen to have a more definite edge in terms of power consumption with a consumption of approximately 70mW compared to the 211mW consumed by the conventional design. This is made better by the reversible logic that minimizes the unneeded activity within the circuit.

The reversible ALU also generated some additional outputs called garbage bits, although this did not affect the proper operation of the circuit. Comprehensively, the reversible design was more effective in power consumption and the use of hardware without the loss of any of the ALU functions, indicating reversible logic may be a powerful solution to energy-constrained digital systems.

But here major concern is delay is more in reversible ALU because reversible circuits are not optimized by FPGA tools, because FPGAs are made for irreversible CMOS logic.

**Routing Overhead and Latency Analysis**

When comparing the reversible ALU with the conventional irreversible ALU, the delay of the reversible design (22.737 ns) is slightly higher than that of the irreversible design (20.214 ns). This increase in delay is mainly caused by routing overhead rather than the speed of reversible gates.

In reversible logic circuits, all intermediate signals must be preserved in order to maintain a one-to-one correspondence between inputs and outputs. As a result, additional signals known as garbage outputs are generated. In the proposed 32-bit reversible ALU design, a total of 128 garbage outputs are produced and must be retained to preserve the reversibility property.

Unlike conventional irreversible circuits, where unused intermediate signals can be eliminated through logic optimization, reversible circuits require strict preservation of all signals. Even if some intermediate outputs are not required for the final computation, they must still be propagated to maintain reversibility. This requirement increases the number of routing paths and contributes to higher routing complexity.

Furthermore, this limitation becomes more problematic when the design is implemented on the Xilinx Artix-7 FPGA platform. FPGA architectures are primarily optimized for conventional irreversible CMOS logic using LUT-based synthesis techniques. In Reversible circuits one-to-one signal mapping prevents logic optimization, which leads to increased interconnect usage and longer routing paths.

Therefore, the observed delay overhead in the reversible ALU arises mainly from architectural and routing constraints in FPGA implementations rather than from the logical depth of the reversible computation itself.

**Comparison Tables**

Table 1: Comparison Between Irreversible ALU and Optimized Reversible ALU

| Parameter | Irreversible ALU | Reversible ALU (Optimized) |
|---|---|---|
| LUT Count | 245 | 151 |
| DSPs Used | 3 DSP48s | 0 |
| IO Pins Required | 102 | 230 |
| Delay (ns) | 20.214 ns | 22.737 ns |
| Power (mW) | 211mW | 70mW |
| Garbage Bits | 0 | 128 |
| Quantum Cost | 0 | 384 QC |
| Gate Types Used | CMOS logic | Peres Feynman Rev FA |
| No. of Operations | 09 | 10 |

Table 2: Comparison Between Our Proposed Work with Existing Work

| S. No | Parameter | Proposed Work | Existing Work (IEEE 2021) |
|---|---|---|---|
| 1 | Paper Focus | Comparison of Irreversible vs Optimized Reversible 32-bit ALU | Comparison of Existing vs Proposed Reversible 32-bit ALU |
| 2 | FPGA Platform | Xilinx Artix-7 | Vivado Design Suite (FPGA-based) |
| 3 | ALU Size | 32-bit | 32-bit |
| 4 | Reversible Gates Used | Peres (main), Feynman, Toffoli, Fredkin | Peres, Feynman, Toffoli, Fredkin |
| 5 | Operations Implemented | Add, Sub, AND, OR, XOR, Shift Operations (SHL, SHR), Increment/Decrement and also Full Adder | AND, OR, Addition, Subtraction |
| 6 | Quantum Cost | 384 QC | Not mentioned |
| 7 | Garbage Outputs | 128 bits | Minimized (not numerically specified) |
| 8 | LUT Count | 245 (Irrev) → 151 (Rev) | 97 (Existing) → 64 (Proposed) |
| 9 | Area Reduction | 38% LUT reduction | 34% area reduction |
| 10 | DSP Blocks Used | 3 DSP48s (Irrev) → 0 (Rev) | Not specified |
| 11 | IO Pins Required | 102 (Irrev) → 230 (Rev) | Not emphasized |
| 12 | Delay (ns) | 20.214 ns (Irrev) → 22.737 ns (Rev) | 19.695 ns → 10.061 ns |
| 13 | Delay Improvement | Reversible slightly slower due to routing overhead | 48.91% delay reduction |
| 14 | Power Consumption | 70 mW (Reversible) | 82 mW (Reversible) |
| 15 | Main Strength | Significant power reduction | Huge delay & area optimization |
| 16 | Main Limitation | Increased routing delay due to 128 garbage outputs | Design complexity |
| 17 | Application Target | Low-power VLSI & Energy-efficient FPGA systems | Low-power and high-speed ALU optimization |

### Schematic Diagrams and Waveforms

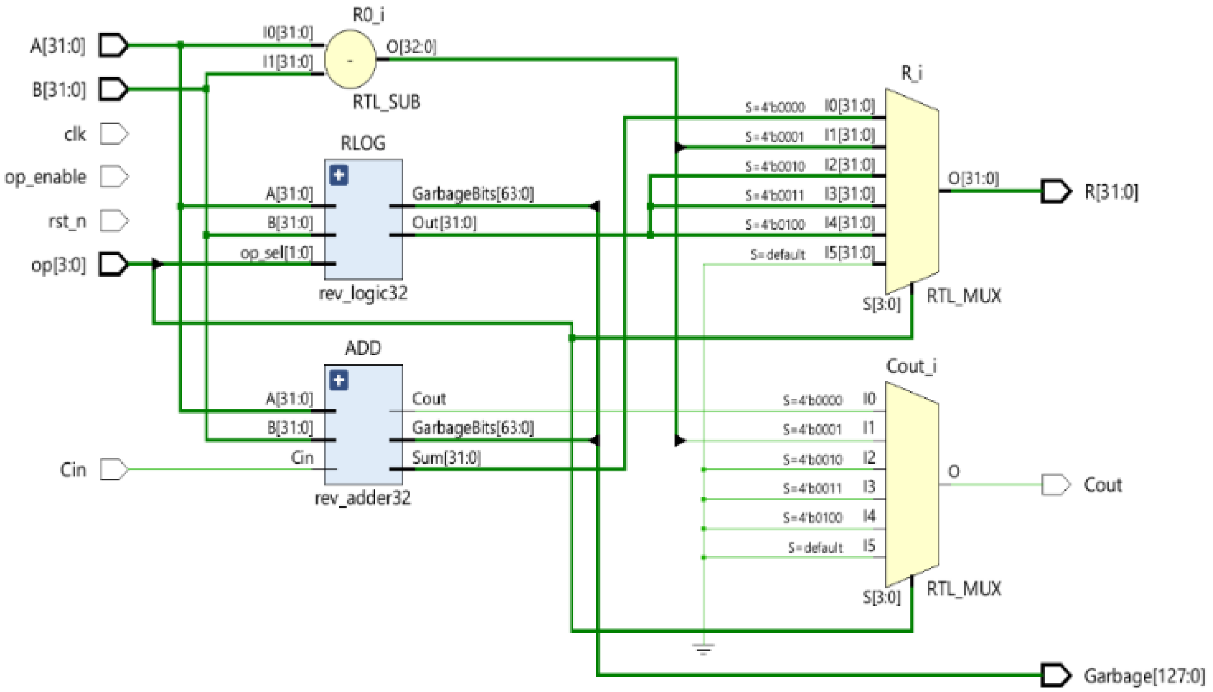

**Figure 8: Reversible RTL Schematic**

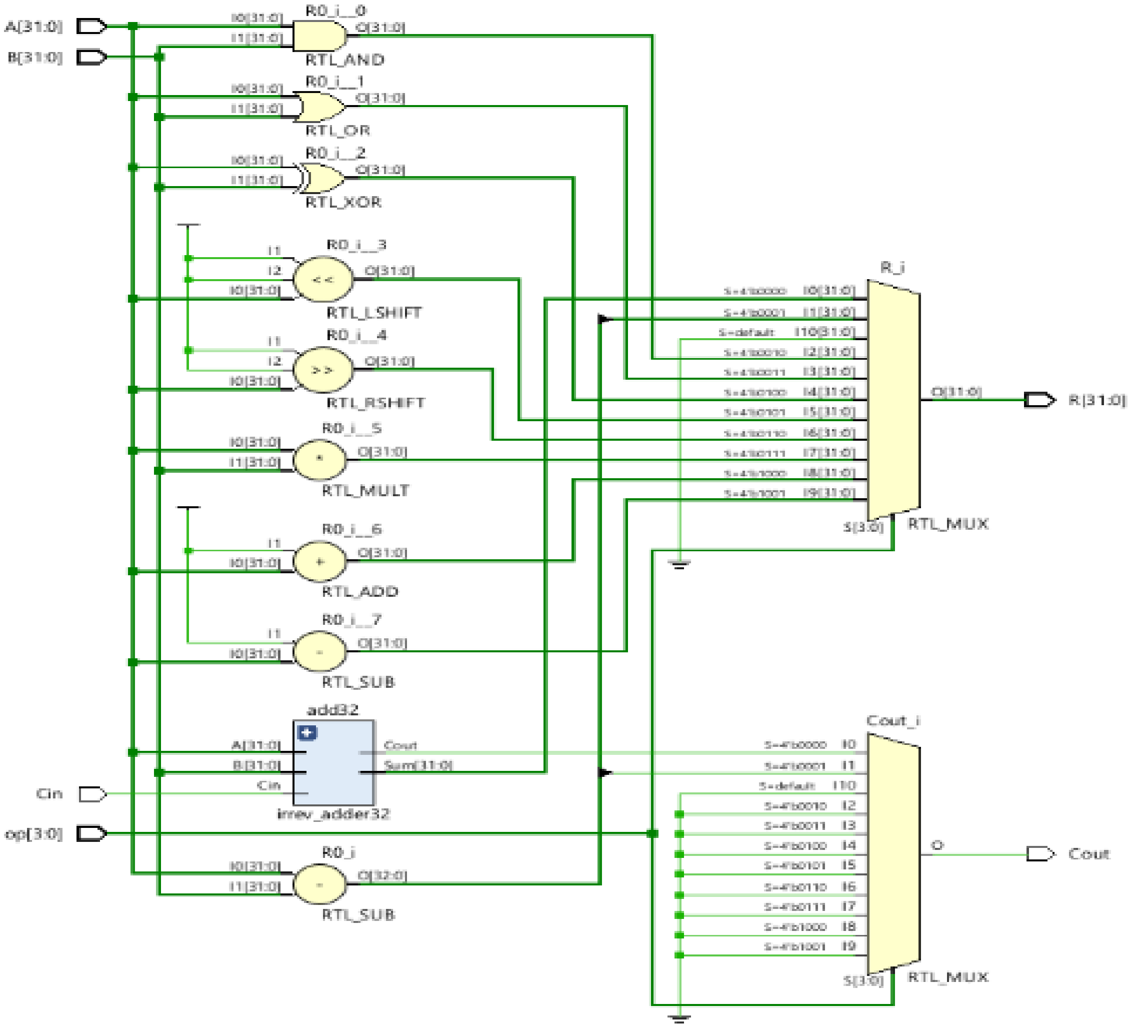

**Figure 9: Irreversible RTL Schematic**

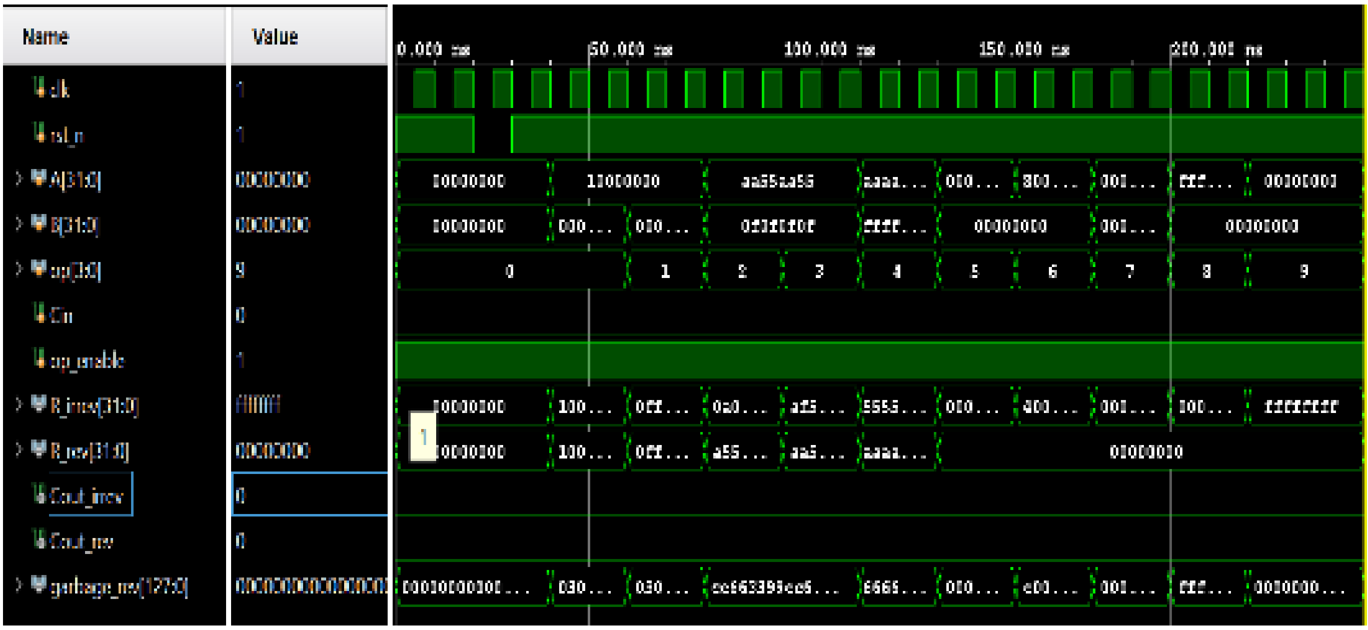

**Figure 10: 32-bit output of Full Adder Using reversible gates**

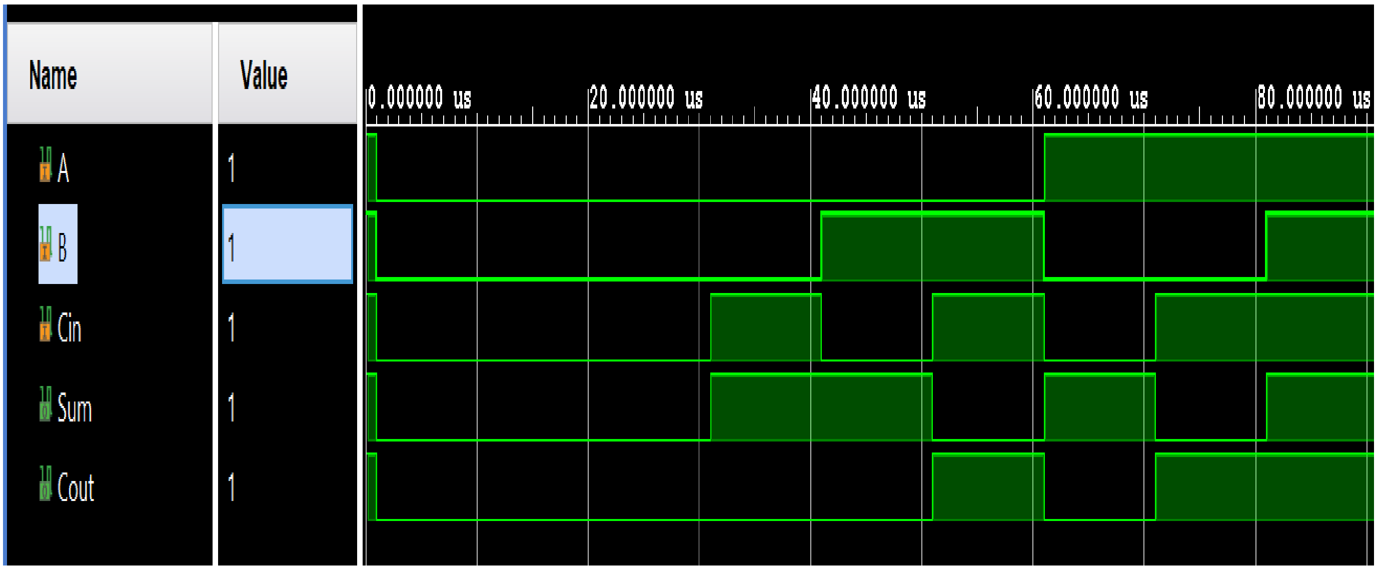

**Figure 11: 32-bit output of Full Adder**

# CONCLUSION

This project adopted and compared an irreversible ALU with 32bits using Artix-7 FPGA and an optimized reversible ALU using Artix-7 FPGA, demonstrating that reversible logic can work in practice. The two ALUs functioned identically. However, the optimized reversible ALU was more hardware efficient: a 38 percent decrease in the number of LUTs (151 LUTs vs. 245 LUTs) was achieved, and it could do away with more power usage DSP blocks that the irreversible design had to use. The optimized design provides a generic, area-efficient, a power-friendly architecture, despite of the overhead of reversible properties (128 garbage bits, quantum cost 384), by eliminating information loss. The work makes the optimized reversible ALU a good candidate in the next generation, energy-conscious applications in VLSI, low-power applications, and quantum computing.

## Future Scope

### Pipelined Reversible ALU

To reduce the delay of the reversible ALU, a pipelined architecture can be used. Instead of implementing the entire 32-bit ALU as a single combinational block, the data path can be divided into smaller stages.

For example, the ripple-carry adder can be partitioned into four 8-bit sections, and pipeline registers can be inserted between these stages. By dividing the computation into smaller segments, the combinational path length in each stage is reduced, which decreases the critical path delay and allows the circuit to operate at a higher clock frequency.

Although the introduction of pipeline registers increases hardware resources, pipelining improves system throughput by enabling multiple operations to be processed simultaneously. This approach can significantly enhance the performance of reversible ALU architectures in high-performance or energy-constrained computing systems.

**Uncomputation-Based Garbage Reduction**

Another possible improvement is the use of uncomputation techniques proposed by Charles H. Bennett to reduce the number of garbage outputs.

In the current design, garbage outputs are generated as a by-product of reversibility. These additional signals require extra routing resources and contribute to increased interconnect complexity and delay.

Uncomputation provides a method to clean up intermediate results after the desired output has been obtained. In this approach, the required outputs are first copied to a safe register, and then the intermediate computation steps are reversed so that temporary signals return to their original states. As a result, unnecessary intermediate signals can be removed while still preserving reversibility. By integrating controlled uncomputation blocks into the ALU architecture, the number of garbage outputs can be reduced. This reduction would lower routing complexity, decrease interconnect overhead, and improve the overall performance of the reversible ALU.

## REFERENCES

1. R. Landauer, "Irreversibility and Heat Generation in the Computing Process," *IBM Journal of Research and Development*, 1961.
2. C. H. Bennett, "Logical Reversibility of Computation," *IBM Journal of Research and Development*, 1973.
3. T. Toffoli, "Reversible Computing," *ICALP / Springer LNCS (chapter)*, 1980.
4. E. Fredkin and T. Toffoli, "Conservative Logic," *International Journal of Theoretical Physics*, 1982.
5. A. Peres, "Reversible Logic and Quantum Computers," *Physical Review A*, 1985.
6. D. Maslov et al., "Reversible Logic Synthesis with Fredkin and Peres Gates," *ACM (Proceedings)*, 2007.
7. G. Yang et al., "Majority Based Reversible Logic Gates," *Theoretical Computer Science*, 2005.
8. C. Jose et al., "An FPGA Implementation of Low Dynamic Power & Area Optimized 32-bit ALU using Reversible Decoder Controlled Combinational Circuits," *International Journal of Applied Engineering Research*, 2018.
9. S. M. Swamynathan and V. Banumathi, "Design and Analysis of FPGA-Based 32-bit ALU using Reversible Gates," *ResearchGate preprint / project notes*, 2017–2018.
10. (Authors listed on RG page), "Comparison of 32-bit ALU for Reversible Logic and Irreversible Logic," *ResearchGate preprint*, 2021.
11. (JETIR), "32-bit FPGA-Based ALU Employing Reversible Logic," *Journal of Emerging Technologies and Innovative Research*, 2024.
12. (IRJET), "Performance Optimization of 32-bit ALU Implemented with Reversible Logic Gates," *IRJET*, 2024.
13. (Journal of Current Research and Development), "An Optimization of ALU using Reversible Logic Gate," *Journal CRD*, 2025.
14. S. S. et al., "Design and Analysis of Reversible Control Unit for Arithmetic and Logical Operations," *International Journal of Research in Engineering and Science (IJRES)*, 2022.
15. (IJIRSET), "64-Bit FPGA-Based ALU Employing Reversible Logic," *Int. Journal of Innovative Research in Science, Engineering and Technology*, 2025.
16. S. Nagaraj, B. Chakradhar, B. V. Krishna, and D. Sarkar, "Comparison of 32-bit ALU for Reversible Logic and Irreversible Logic,"

17. *IEEE Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2021.
18. P. Keerthana et al., "Review on Design and Analysis of ALU Using Reversible Logic," *IJRAMT*, 2022.
19. G. Sanjeevaiah and S. B. Gajanan, "MF-RALU: Multi-Functional Reversible ALU for Processor Design on FPGA," International Journal of Electrical and Computer Engineering (IJECE), 2023.