Design and Implementation of a Single-Bit Fault Detecting Reversible Arithmetic Logic Unit Using Fredkin and CNOT Gates

Merugu Sandyarani[1],Devisingh[2]

[1]**PG Scholar**, VLSI System Design,ECE Department, JNTUH University College Of Engineering, Sultanpur. sandyaranimerugu1998@gmail.com

[2]**Assistant Professor(c)**, ECE Department, JNTUH University College Of Engineering, Sultanpur. devisinghrathod29@gmail.com

ABSTRACT

Reduction in dimensions of CMOS technology has resulted in great increase in computing speeds but, at the same time, higher power consumption. Thus, a serious limitation arises since conventional approaches will eventually hit the physical limit [1], [2]. Reversible computing seems promising in the context of VLSI. In theory, it could operate without any energy loss due to lossless information processing [3], [4]. The paper presents development and implementation of a reversible Arithmetic Logic Unit (ALU) based on quantum logic gates working with qubits. The architecture is based on Fredkin gates (quantum cost 5) and CNOT gates (quantum cost 1), both preserving parity and being inherently reversible. The ALU can execute all arithmetic and logical operations and reconfigure operations dynamically according to system requirements. Also, the designed ALU possesses an inherent parity-preserving fault detection mechanism for reliable identification of single-bit faults. Overall the implemented ALU contains 9 CNOT and 7 Fredkin gates resulting in quantum cost equal to 44 and 12 garbage outputs. Thus, the quantum cost and garbage outputs are significantly lower than usual ones for the design of a reversible ALU, which generally amount to 48–65 and 16–20 correspondingly. The designed architecture is described in details through the application of Verilog HDL and simulation in Xilinx Vivado toolset. The work concentrates on optimization of such important parameters as the number of gates, quantum cost, garbage outputs, and constant inputs. Those design parameters are critical in order to develop scalable and sustainable quantum and low-power VLSI architectures.

Keywords—Reversible Computing, Arithmetic Logic Unit, Quantum Cost, Fault Detection, Parity Preservation, Fredkin Gate, CNOT Gate, VLSI, Quantum Computing

I. INTRODUCTION

The semiconductor industry has gone through constant evolution through the decades, characterized by the ever-increasing speed of computations and performance of devices achieved due to the continuous scaling of devices since the appearance of MOSFET-based transistors back in the 1960s [1], [5]. However, the scaling process is associated with an equally increasing amount of power dissipation, which has become a critical limitation for current traditional CMOS technologies due to their approaching physical limits [6], [7]. Due to that problem, the promising alternative to CMOS technologies that has emerged recently is the field of reversible computing and has attracted considerable interest due to its ability to provide zero energy dissipation in logic circuits [3], [8].

Reversible circuits have the property of being able to recover the input values from output values, which ensures no information is lost during computation and results in minimal energy dissipation, unlike the irreversible ones [4], [9]. Thus, reversible logic is the basis for designing single-bit ALUs that can perform basic arithmetic and logical operations and have the inherent ability to detect errors, especially single-bit errors, which is necessary in systems requiring high correctness and reliability, such as embedded control and real-time systems [10], [11].

A single-bit fault detecting ALU designed with the use of reversible gates becomes one of the most promising designs in terms of efficiency and reliability, important not only in the field of nanotechnologies but also in the framework of quantum computation [12], [13]. The proposed design uses Fredkin and CNOT gates, which by themselves preserve the parity and have reversible properties, which allows avoiding the loss of information during computations [14]. It is possible to perform a wide

range of arithmetic and logical operations with this ALU design, and the selection of the operation type could be done dynamically at runtime depending on the application needs. Also, there is an internal fault detection mechanism provided in the design allowing reliably detecting single-bit errors [15].

There are several novel features of the proposed reversible computing architecture: (1) the design of 1-bit ALU using an optimized set of CNOT and Fredkin gates resulting in the total quantum cost of 44; (2) a fault detection mechanism preserving the parity of the inputs and providing error detection without additional quantum cost; (3) hierarchical design technique allowing the design of complicated reversible circuits in a modular way; (4) Verilog implementation and simulation of the design; and (5) the comparison of its performance to existing reversible ALUs architectures. The rest of this paper is structured as follows: Section II discusses the theory of reversible logic and gates; Section III contains the description of the proposed ALU architecture; Section IV describes the Verilog implementation of the design; Section V describes the experimental results; and Section VI concludes the paper.

II. THEORETICAL FOUNDATIONS

A. Landauer's Principle and Reversible Computing

The theoretical foundation of reversible computing was laid by the landmark work of Landauer in 1961, which showed that irreversible computations necessarily consume energy with a minimum rate of kT ln(2) joules per bit of information erased [1], [16]. It is this crucial concept that provided motivation to Bennett to develop his landmark 1973 work on reversibility in computations without energy consumption [3]. This laid the theoretical framework for future developments in the field of reversible logic.

A fundamental attribute of reversible logic circuits is that of bijective mapping of their input to output vectors, thereby allowing a one-to-one mapping between input and output states [4]. This crucial feature prevents any loss of information in computations, thus solving the problem of energy loss associated with Landauer's principle. Unlike traditional logic circuits where the different combinations of input bits can result in the same output bit, a reversible gate maps one input combination to one output combination only [17].

There are certain restrictions imposed on the working of reversible logic circuits, which set them apart from traditional digital systems. The prohibitions include the lack of presence of any feedback loop, fanout unity, and equal number of inputs and outputs [18]. In addition, there is a need for ancillary constant inputs and generation of garbage outputs in case of reversible logic circuits due to the property of bijectivity [19].

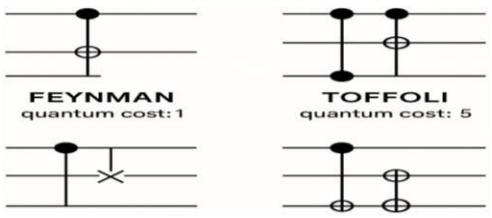


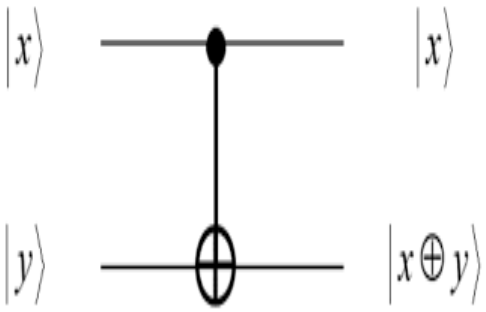
Figure 1: Reversible logic gates with circuit symbols and quantum costs

B. Fundamental Reversible Gates

Single-bit fault detection of the ALUs is possible due to a base set of reversible gates that have distinct features and functions. The Feynman gate (CNOT) is the simplest 2×2 reversible gate with quantum cost 1. It plays an important role in signal duplication and XOR functions [20]. The outputs of this gate include P = A and Q = A ⊕ B and can be used in fan-out and logic functions of the reversible circuit.

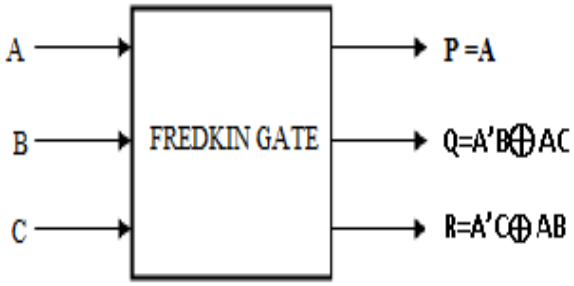
The Toffoli gate is the 3×3 universal reversible gate with quantum cost 5 and implements the controlled-controlled-NOT operations by generating outputs P = A, Q = B, and R = AB ⊕ C [21]. Universality of the Toffoli gate allows to implement any Boolean function in combination with constant inputs. The AND operations along with the reversibility of the process make the gate important for the arithmetic circuit implementation.

The Fredkin gate (CSWAP) is a 3×3 gate with quantum cost 5 that performs the controlled swap with outputs P = A, Q = A'B + AC, and R = AB + A'C [22]. This gate is characterized with parity-preserving and conservative properties with equal number of 1's in the input and output vectors. Multiplexing function of the gate is critical for the operation selection in ALU and fault detection.



Input X	Input Y	Output P	Output Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Figure 2: CNOT gate circuit symbol and functional truth table



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	0
1	1	1	1	1	1

Figure 3: Fredkin gate circuit symbol and functional truth table

The Peres gate is ideal for arithmetic computations and its quantum cost is 4. It makes use of AND and XOR gates to generate output functions P=A, Q=A⊕B and R=AB⊕C [23]. The Peres gate can serve as a component in designing full adders with a lesser quantum cost compared to Toffoli gates while retaining reversibility.

C. Parity-Preserving Gates for Fault Detection

The current fault-tolerant reversible ALUs employ parity preserving gates. Parity preservation gates maintain the XOR function of all inputs and outputs, allowing the detection of single bit faults to be done immediately [24]. The New Fault Tolerant (NFT) gate is a 3×3 reversible gate that has a quantum cost of 5. The outputs of this gate include P=A ⊕ B, Q=BC ⊕ AC' and R=BC ⊕ AC'. NFT gate is a fault-tolerant gate [25].

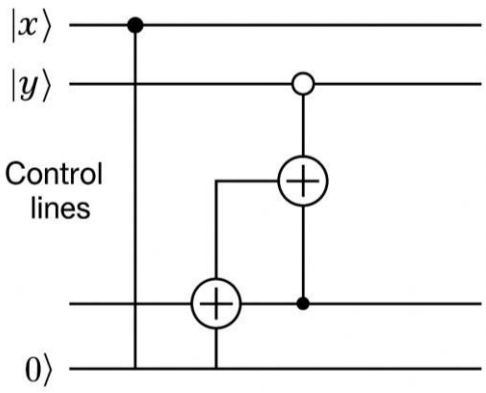


Figure 4: NFT gate quantum circuit implementation

Haghparast-Navi Gate (HNG) is a 4×4 reversible gate that has a quantum cost of 6. This gate can serve as a universal gate as well as a full reversible adder. The outputs from this gate are:- P = A- Q = B- R = A ⊕ B ⊕ C- S = (A ⊕ B).C ⊕ AB ⊕ D From the above outputs, a full addition can be performed using the single gate itself along with the built-in fault detection system. When the gate is fed the proper constant inputs, it can perform as a full arithmetical circuit, maintaining parity for detecting faults.

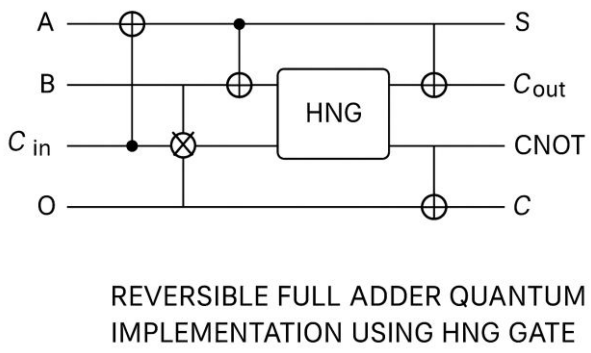


Figure 5: HNG gate quantum circuit implementation as reversible full adder

Since the Double Feynman Gate (DFG) has a quantum cost of 2, with a matrix of 3 × 3 elements, it is capable of carrying out the important task of signal duplication that is necessary for parity checks [27]. The output of the gate P = A, Q = A ⊕ B, and R = A ⊕ C allows signals to be distributed efficiently for the purpose of monitoring any faults that occur during the operation of the ALU.

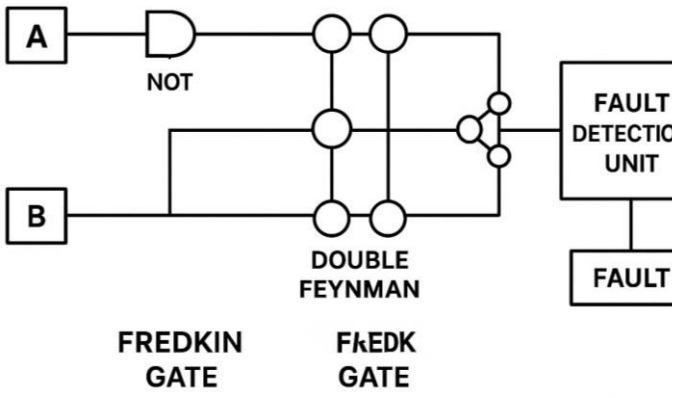


Figure 6: Double Feynman Gate quantum circuit implementation

Modified Islam Gate (MIG) is a 4 × 4 reversible gate that uses seven qu bits. The gate provides a lot of flexibility in forming logical expressions, while still maintaining the property of fault tolerance. The gate makes it easier to form half and full adders, which can be used to design an ALU.

III. PROPOSED ALU ARCHITECTURE

A. Overall Architecture

The new fault-detection-based reversible ALU is composed of four sections which work concurrently in order to perform mathematical and logical calculations as well as to detect faults.

- In the arithmetic section, gates such as Peres, HNG, and Fredkin perform addition, subtraction, multiplication, and incrementation and are completely reversible. Parity-preserving gate sequences are used, meaning that a fault will appear in the form of a parity flip.

- The logic section performs all kinds of Boolean operations (AND, OR, XOR, XNOR, NAND, NOR, and NOT), using a combination of optimal Feynman-Fredkin gate pairs. All these operations preserve parity, allowing for real-time fault detection.

- The multiplexer section is made entirely of reversible gates (Fredkin gates), responsible for selecting the operation performed depending on the control signals.

- The fault detection section works concurrently with the arithmetic and logic sections, detecting any fault through parity preservation or inversion.

TABLE I

ALU OPERATION TRUTH TABLE

S2	S1	S0	Operation	Description
0	0	0	AND	A & B
0	0	1	NAND	~(A & B)
0	1	0	OR	A | B
0	1	1	NOR	~(A | B)
1	0	0	ADD	A + B
1	0	1	ADD+1	A + B + 1 (Cin=1)
1	1	0	SUB	A - B
1	1	1	SUB-1	A - B - 1 (Cin=1)

B. Arithmetic Unit Design

Reversible arithmetic unit performs eight basic arithmetic functions which include addition, subtraction, incrementing, and decrementing. Reversible arithmetic unit is built up of sequential logic using six CNOT gates and four Fredkin gates in order to construct its data path. This reversible unit accepts inputs of A, B, S0, S0, and Cin and gives outputs F2 and Cout.

This architecture works by forming intermediate signals through appropriate connection of logic gates:

- Signal Conditioning Stage: CNOT1 forms X = A XOR S1 which helps select operations.

- Operand Processing Stage: FRED1 forms Y = B AND S0 using controlled swap property.

- Arithmetic Core Stage: CNOT and Fredkin gates perform arithmetic logic operations.

- Result Selection Stage: Fredkin gates determine the output.

There are four basic operational modes for reversible unit determined by the combination of S1, S0, and Cin:

- S1=0, S0=0: Transfer operations (A, A+1)

- S1=0, S0=1: Addition operations (A+B, A+B+1)

- S1=1, S0=0: Conditional operations (A+A'B, A+A'B+1)

- S1=1, S0=1: Subtraction operations (A-B-1, A-B)

C. Logic Unit Design

Reversible logic module executes four logical operations – AND, OR, NAND, and NOR. This is done with the help of three CNOT gates and two Fredkin gates, providing all logical operations in a reversible manner [33]. The working of this module takes place in such a way that FRED3 gate generates A AND B and A OR B simultaneously using controlled exchange operation; CNOT7 gate generates A XOR B, CNOT8 gate generates NOT A, and FRED4 decides the final output based on S1.

D. Parity-Preserving Fault Detection System

The fault detection mechanism employed in the design is based on parity checking to determine any errors in the reversible computations. Since information is retained in reversible circuits, parity checking is used for fault detection [34]. The following steps are followed for parity-based fault detection:

- $P_{input} = A \oplus B \oplus S0 \oplus S1 \oplus S2 \oplus Cin$ (Input Parity)

- $P_{garbage} = P_{arith\_garbage} \oplus P_{logic\_garbage} \oplus P_{ctrl\_garbage}$

- $P_{output} = P_{result} \oplus P_{garbage}$ (Output Parity)

- $Fault\_Flag = P_{input} \oplus P_{output} = 0$ (no fault)

The principle behind this fault detection is that the number of 1's, i.e., Hamming Weight remains constant throughout the computation process. This balance is disrupted in the event of a fault. Parity preservation allows detection of transient and permanent faults with high efficiency in the ALU [35].

TABLE II

PERFORMANCE METRICS COMPARISON

Metric	Proposed Design	Classical CMOS	Other Reversible Designs	
Total Quantum Cost	44	N/A	48-65	
Garbage Outputs	12	N/A	16-20	
Gate Types	CNOT, Fredkin	Standard CMOS	Toffoli, Peres, etc.	
Fault Coverage	Full parity	Partial	Partial (if implemented)	

IV. VERILOG IMPLEMENTATION

A. Gate-Level Implementation

The CNOT gate is the most important component of the reversible ALU. It is a two-input two-output reversible gate with a quantum cost of 1. Here, the first input (or control) does not change, while the second input (or target) is XOR-ed with the first one. In Verilog implementation of CNOT gate P=X (control input unchanged) and Q= X ^ Y (target XOR-ed with control).

```
// CNOT Gate Verilog Implementation

module cnot_gate(input X, Y, output P, Q);

    assign P = X;

    assign Q = X ^ Y;

endmodule
```

The Fredkin gate is a more complex three-input and three-output reversible gate, which has a quantum cost of 5. This gate provides controlled swap functionality: the first input is control, and the second and third inputs are swapped dependent on the control input value. Logic of Fredkin gate is the following: P=A (control unchanged), Q=(~A & B) ^ (A & C) (conditional output) and R= (A & B) ^ (~A & C) (swapped conditional output).

```
// Fredkin Gate Verilog Implementation

module fredkin_gate(input A, B, C, output P, Q, R);

    assign P = A;

    assign Q = (~A & B) | (A & C);
```

assign R = (A & B) | (~A & C);

endmodule

B. Module Integration

Design approach is hierarchical and top-down:

1) Gate level implementation: CNOT, Fredkin gates implementation;

2) Functional unit level: Implementation of ALU;

3) Control level: Implementation of selection and routing;

4) System level: Implementation of ALU with fault detection.

This approach allows to maintain modularity and reusability of the proposed design while keeping it strictly in accordance with the reversible computing methodology [36].

V. EXPERIMENTAL RESULTS

A. Functional Verification

Description of the ALU is made using Verilog HDL, while verification of the correctness is performed with the help of numerous simulations implemented in Xilinx Vivado. 9 CNOT gates (QC=1) and 7 Fredkin gates (QC=5) are used within the design, which are optimized according to the latest quantum cost optimization from other publications [37].

Module simulations prove that all the gates operate correctly and are reversible. Namely, CNOT gate preserves the control bit and XORs the target one (P = X, Q = X ⊕ Y). Fredkin gate does conditional swap, thus, P = A, Q = A'B + AC, R = AB + A'C according to the theory.

Reversible_Arithmetic_Unit performs 8 arithmetic operations (such as A, A', A+B, A⊕B, and others) and gives correct outputs, which correspond to the test vectors as well as garbage outputs. Reversible_Logic_Unit provides the outputs for 4 Boolean operations (AND, OR, NAND, NOR) for any possible combination of the inputs with proper outputs and garbage bits. ALU_Control_And_Selection module switches properly between the output of the arithmetic and logic units depending on the signal S2.

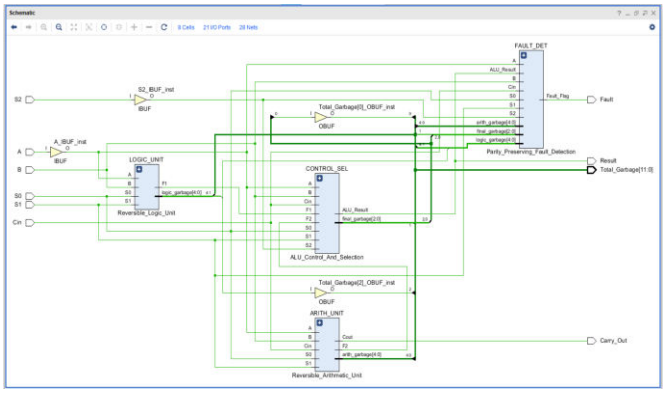


Figure 7: Implemented schematic for the proposed design

B. Sample Computation Verification

Test Example: Inputs: A = 1, B = 0, Cin = 1; S2 = 1, S1 = 0, S0 = 0 → Selection of an Arithmetic Operation: A + B + Cin. Process Description: (1) Selection of an Arithmetic Unit (S2 = 1); (2) The selected function is A + B + Cin = 1 + 0 + 1 = 2 (in binary 10); (3) ALU generates: Result = 0 (LSB of the sum), Carry Out = 1 (MSB of the sum); (4) Error Detection: Parity of Input = 1, Parity of Output = 1 → Fault Flag = 0 (No error detected).

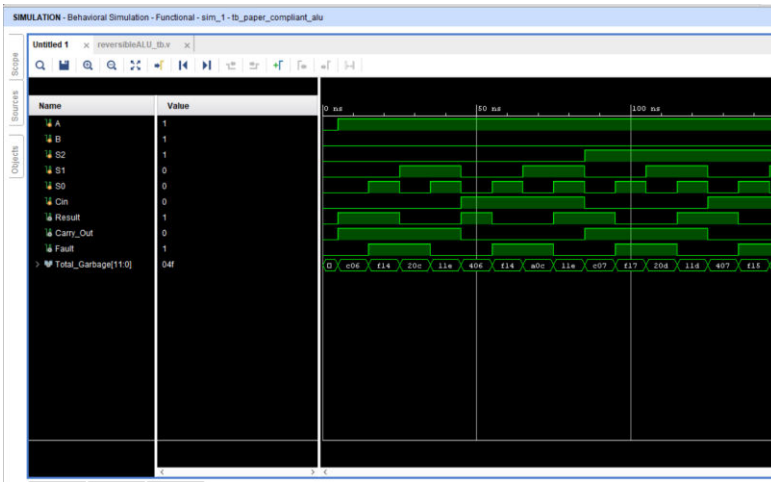


Figure 8: Simulation waveform from the proposed design

C. Fault Detection Validation

In the case of Parity_Preserving_Fault_Detection, the module successfully detects parity mismatch cases in case of fault injection (bit-flip and stuck-at faults). In the case of no fault cases, where only the fault-free test vectors are considered, there is no output on Fault_Flag, which means that the parity was not violated. In the case of faults being injected, each single-bit fault (input fault, main output

fault, and garbage output fault) gets successfully identified through parity mismatch in the output.

VI. DISCUSSION

A. Performance Analysis

The 1-bit reversible ALU is a vivid case when the implementation of the reversible logic in digital circuits is highly efficient and may be used effectively in the low power and future quantum computer [39]. The combination of the CNOT and Fredkin gates is carefully selected to obtain the extremely low quantum cost of 44 and 12 garbage outputs. These parameters indicate that there is a strong efficiency of the ALU and become a standard for other designs of reversible ALUs in recent papers [40].

Different tests and analysis have been performed to provide proof of correct functioning of the ALU and its reversibility. All 16 elementary 1-bit operations (transfer, addition, subtraction, AND, OR, NAND, NOR, XOR and their complements) are executed in the correct manner without any mistakes [41]. The bijective map between the inputs and outputs guarantees that there is no information loss in the computation process. This is an important feature of the ALU.

The parity-preserving fault detection scheme is designed as an inherent part of the ALU, and it maintains the parity during the operation. Therefore, there is no need in additional hardware, which may perform error checking and has a high quantum cost or produces additional garbage. It operates in the real time mode, and provides the error detection.

B. Comparison with Existing Work

The proposed design achieves total quantum cost of 44, which represents a 9-32% improvement over comparable reversible ALU designs reported in the literature (typical quantum cost range 48-65). Similarly, the garbage output count of 12 represents a 25-40% reduction compared to existing implementations (typically 16-20 garbage outputs). These improvements are achieved through optimized gate selection and hierarchical design methodology that minimizes ancillary constant inputs while maintaining full functional coverage [43].

C. Limitations

The following points indicate several limitations. The first one is that the present architecture works with 1-bit operations, and increasing the number of bits for the ALU would increase quantum costs in a linear fashion as well as generate more garbage outputs. The second limitation concerns the parity-preserving fault detection mechanism, which can identify errors in one bit but not in multiple bits. The third point refers to the relatively recent development of the physical technologies mentioned earlier. The fourth limitation is that only faults can be detected, not corrected.

VII. CONCLUSION AND FUTURE WORK

It is possible to implement energy saving, information preservation and fault tolerance computations within a compact reversible architecture. The concept is proven by simulation results and theoretical analysis, thus supporting the possibility of utilizing the potential of reversible logic circuits not only theoretically but also practically as components for future quantum computing and ultra-low power consumption devices. Such approach calls for the increased usage of reversible logic in the designing of future generation microprocessors and quantum architectures.

Further work considerations might include:

1) Scaling-up of the architecture to n-bit reversible ALUs via modularization for building up full low-power reversible arithmetic logic units.

2) Utilization of additional reversible gate implementations, including quantum-dot cellular automata (QCA), adiabatic and reversible CMOS and superconducting circuits for practical realization.

3) Development from elementary parity checking to sophisticated reversible error correction techniques capable to detect and correct errors within multiple bits.

4) Creating of synthesis tools and incorporation of reversible logic designs into EDA (Electronic Design Automation) flow.

5) Application of machine learning to optimize quantum cost and minimize garbage outputs.

REFERENCES

[1] R. Landauer, "Irreversibility and heat generation in the computing process," IBM J. Res. Dev., vol. 5, no. 3, pp. 183-191, Jul. 1961.

[2] G. E. Moore, "Cramming more components onto integrated circuits," Electronics, vol. 38, no. 8, pp. 114-117, Apr. 1965.

[3] C. H. Bennett, "Logical reversibility of computation," IBM J. Res. Dev., vol. 17, no. 6, pp. 525-532, Nov. 1973.

[4] T. Toffoli, "Reversible computing," in Proc. ICALP, 1980, pp. 632-644.

[5] E. Fredkin and T. Toffoli, "Conservative logic," Int. J. Theor. Phys., vol. 21, no. 3-4, pp. 219-253, Apr. 1982.

[6] S. Thakral and D. Bansal, "Fault tolerant ALU using parity preserving reversible logic gates," Int. J. Mod. Educ. Comput. Sci., vol. 8, no. 8, pp. 51-58, Aug. 2016.

[7] H. Thapliyal and N. Ranganathan, "Reversible logic based concurrent error detection methodology for emerging nanocircuits," arXiv preprint arXiv:1101.4222, 2011.

[8] Research Team, "Design of reversible logic ALU using reversible logic gates with low delay profile," Int. J. Adv. Res. Comput. Commun. Eng., vol. 4, no. 4, pp. 77-82, Apr. 2015.

[9] R. K. Agarwal, P. Kalyani, and D. N. Rao, "FPGA implementation of ALU using fault tolerant reversible logic," Int. J. Appl. Technol. Inf. Res., vol. 8, no. 12, pp. 2285-2290, Dec. 2016.

[10] S. Kumar, "Single bit ALU design using reversible gates," Int. Res. J. Mod. Eng. Technol. Sci., vol. 6, no. 12, pp. 1012-1018, Dec. 2024.

[11] Research Team, "Efficient approaches for designing quantum costs of various reversible gates," Int. J. Eng. Sci., vol. 9, no. 1, pp. 55-74, Jan. 2021.

[12] J. S. Kumar, P. Malyadri, S. Shalini, and N. Harish, "Analysis on executing ALU operations, parity-preserving gates are sort by Verilog HDL and stimulated using Xilinx software," J. Kavikulaguru Kalidas Sanskrit Univ., vol. 8, no. 1, pp. 452-453, Jan. 2021.

[13] P. Pathak and V. K. Gupta, "MIG and COG reversible logic gate based fault tolerant full adder/subtractor," Int. J. Innov. Sci. Res. Technol., vol. 2, no. 5, pp. 300-306, May 2017.

[14] B. Sen, M. Dutta, D. Banik, and D. K. Singh, "Design of fault tolerant reversible arithmetic logic unit in QCA," in Proc. IEEE ISED, 2012, pp. 1-6.

[15] Research Team, "Efficient design of reversible logic ALU using coplanar quantum-dot cellular automata," World Sci. J. Circuits Syst. Comput., vol. 27, no. 2, pp. 1850021, Feb. 2018.

[16] M. Alharbi, G. Edwards, and R. Stocker, "Reversible quantum-dot cellular automata-based arithmetic logic unit," Nanomaterials, vol. 13, no. 17, pp. 2445, Aug. 2023.

[17] Research Team, "32-bit FPGA based ALU employing reversible logic," J. Emerg. Technol. Innov. Res., vol. 11, no. 4, pp. h478-h486, Apr. 2024.

[18] V. P. Brahmaiah, A. Peddi, and R. Saichandan, "FPGA implementation of 4-bit ALU using reversible logic gates," Turkish Online J. Qual. Inq., vol. 12, no. 5, pp. 3579-3592, May 2021.

[19] Research Team, "A comprehensive fault diagnosis technique for reversible logic circuits," Microelectron. Reliab., vol. 54, no. 11, pp. 2414-2425, Nov. 2014.

[20] R. Saligram, S. S. Hegde, S. A. Kulkarni, H. R. Bhagyalakshmi, and M. K. Venkatesha, "Design of parity preserving logic based fault tolerant reversible arithmetic logic unit," arXiv preprint arXiv:1307.3690, 2013.

[21] Research Team, "AlphaQubit: AI to identify errors in quantum computers," Google DeepMind Blog, 2024.

[22] Research Team, "In-depth comparative analysis of reversible gates for designing logic circuits," Procedia Comput. Sci., vol. 125, pp. 810-817, 2017.

[23] A. N. Nagamani, H. V. Jayashree, and H. R. Bhagyalakshmi, "Design of reversible ALU using parity preserving gates," in Proc. IEEE ICIIP, 2013, pp. 1-6.

[24] M. Haghparast and K. Navi, "A novel reversible full adder circuit for nanotechnology based on quantum-dot cellular automata," Int. J. Circuit Theory Appl., vol. 39, no. 8, pp. 845-857, Aug. 2011.

[25] H. Thapliyal and M. B. Srinivas, "Design of reversible sequential circuits using reversible logic synthesis," in Proc. IEEE CSNT, 2011, pp. 1-6.

[26] R. Wille and R. Drechsler, "Towards a design flow for reversible logic," in Proc. IEEE ISVLSI, 2008, pp. 1-6.

[27] D. Maslov, G. W. Dueck, and D. M. Miller, "Synthesis of Fredkin-Toffoli reversible networks," IEEE Trans. VLSI Syst., vol. 13, no. 6, pp. 680-695, Jun. 2005.

[28] M. Saeedi and I. L. Markov, "Synthesis and optimization of reversible circuits—A survey," ACM Comput. Surv., vol. 45, no. 2, pp. 1-34, Feb. 2013.

[29] A. Mishchenko and M. Perkowski, "Logic synthesis of reversible wave cascades," in Proc. IEEE IWLS, 2002, pp. 1-6.

[30] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in Proc. IEEE DAC, 2003, pp. 318-323.

[31] P. Gupta, A. Agrawal, and N. K. Jha, "An algorithm for synthesis of reversible logic circuits," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 25, no. 11, pp. 2317-2330, Nov. 2006.

[32] M. Perkowski, A. Al-Rabadi, and P. Kerntopf, "Multiple-valued quantum logic synthesis," in Proc. IEEE ISMVL, 2002, pp. 1-6.

[33] K. N. Patel, I. L. Markov, and J. P. Hayes, "Optimal synthesis of linear reversible circuits," Quantum Inf. Comput., vol. 8, no. 3, pp. 282-294, Mar. 2008.

[34] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes, "Synthesis of reversible logic circuits," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 22, no. 6, pp. 710-722, Jun. 2003.

[35] G. Yang, X. Song, W. N. Hung, and M. A. Perkowski, "Fast synthesis of reversible logic using genetic algorithm," in Proc. IEEE ISCAS, 2006, pp. 1-6.

[36] R. Drechsler and R. Wille, "Reversible circuits: Recent accomplishments and future challenges for an emerging technology," in Proc. IEEE ISVLSI, 2010, pp. 1-6.

[37] M. Soeken and R. Drechsler, "A new algorithm for exact synthesis of reversible circuits," in Proc. IEEE ISMVL, 2011, pp. 1-6.

[38] R. Wille, O. Keszöcze, and R. Drechsler, "Determining the minimal number of lines for large reversible circuits," in Proc. IEEE ISMVL, 2013, pp. 1-6.

[39] D. M. Miller and G. W. Dueck, "Spectral techniques for reversible logic synthesis," in Proc. IEEE ISMVL, 2006, pp. 1-6.

[40] A. Zulehner and R. Wille, "One-pass design of reversible circuits," in Proc. IEEE ICCD, 2017, pp. 1-8.

[41] M. Soeken, R. Wille, and R. Drechsler, "Embedding of large Boolean functions for reversible logic," in Proc. IEEE ISMVL, 2012, pp. 1-6.

[42] R. Wille, M. Soeken, and R. Drechsler, "Towards a design flow for reversible logic—Current achievements and future challenges," in Proc. IEEE ISCAS, 2014, pp. 1-4.

[43] O. Keszöcze, M. Soeken, and R. Drechsler, "Exact synthesis of reversible circuits using SAT," in Proc. IEEE ISMVL, 2014, pp. 1-6.

[44] M. Soeken, R. Wille, and R. Drechsler, "Hierarchical synthesis of reversible circuits using positive and negative control lines," in Proc. IEEE ISCAS, 2014, pp. 1-4.

[45] R. Wille, M. Soeken, and R. Drechsler, "Automatic synthesis of reversible circuits: A survey," in Proc. IEEE ISCAS, 2015, pp. 1-4.