MF-RALU: design of an efficient multi-functional reversible arithmetic and logic unit for processor design on field programmable gate array platform

Girija Sanjeevaiah[1], Sangeetha Bhandari Gajanan[2]
[1]Department of Electronics and Communications, Dr. Ambedkar Institute of Technology, Bangalore, India
[2]Department of Electronics and Communications, RNS Institute of Technology, Bangalore, India

Article Info

Article history:

Received Jan 11, 2023
Revised Apr 2, 2023
Accepted Apr 7, 2023

Keywords:

Arithmetic logic unit
Arithmetic unit
Logical unit
Processor
Reversible gates

ABSTRACT

Most modern computer applications use reversible logic gates to solve power dissipation issues. This manuscript uses an efficient multi-functional reversible arithmetic and logical unit (MF-RALU) to perform 30 operations. The 32-bit MF-RALU includes arithmetic, logical, complement, shifters, multiplexers, different adders, and multipliers. The multi-bit reversible multiplexers are used to construct the MF-RALU structure. The Reduced instruction set computer (RISC) processor is designed to realize the functionality of the MF-RALU. The MF-RALU can perform its operation in a single clock cycle. The 1-bit RALU is developed and compared with existing approaches with improvements in performance metrics. The 32-bit reversible arithmetic units (RAUs) and reversible logical units (RLUs) are constructed using 1-bit RALU. The MF-RALU and RISC processor are synthesized individually in the Vivado environment using Verilog-HDL and implemented on Artix-7 field programmable gate array (FPGA). The MF-RALU utilizes a <11% chip area and consumes 332 mW total power. The RISC processor utilizes a <3% chip area and works at 483 MHZ frequency by consuming 159 mW of total power on Artix-7 FPGA.

Corresponding Author:

Girija Sanjeevaiah
Department of Electronics and Communications Engineering, Dr. Ambedkar Institute of Technology
Bangalore, India
Email: girija.pari@gmail.com

1. INTRODUCTION

Digital circuit designs face significant issues like excess heat and power dissipation from consumers and manufacturers, resulting in high information losses. There will be a dramatic reduction in energy/power losses by introducing the new fabrication process and a higher level of integrations over a period. The reversible logic gates dissipate zero energy during computation was introduced by Bennett in 1973 [1]. While computing, the number of bits removed is proportional to the energy dissipated. He concludes that irreversible gate computation's efficiency is the same as reversible gates. The reversible circuits are constructed using a set of reversible gates [1], [2]. Quantum computation uses reversible circuits to solve problems and speed up system efficiency. There is an exponential improvement in hardware architecture's performance using quantum computations, which are adopted in most industrial applications [3]. The reversible gates (RG) are used extensively in low-power applications, wireless networks, digital signal processing (DSP), and quantum computing applications. The RGs reduce the power dissipation based on the number of RG used, usage of ancilla inputs (AI), generated garbage output (GO) and obtained quantum cost (QC). The gate count (GC) specifies how many reversible gates were employed to create the circuit or module. About the cost of the primitive gate, QC is an RG cost. The QC calculation relies on generating the appropriate number of primitive RGs to create a reversible circuit. The input ports must be either constant zero or one for certain reversible circuits to function. In reversible circuits/gates, this zero or one is regarded as AI. Certain reversible circuits provide outputs not utilized in the subsequent circuit realization and are referred to as GO.

The RGs are represented based on Boolean functions, including matrix, truth-table, cycle expansion, binary decision diagram (BDD), and positive polarity Reed-Muller (PPRM). Many syntheses approach, cycle-based, BDD-based, search-based, transformation-based, programming language-based, and exclusive sum of products (ESOPs), are used in reversible logic circuits to realize the functionality and performance analysis [4]. The most commonly used reversible logic gates are the Feynman gate (FG) [5], Peres gate (PG) [6], Fredkin gate (FRG) [7], and Toffoli gate (TG) [8], and many more. These gates are used to construct reversible-based digital circuits, arithmetic and logical unit (ALU), adders, multipliers, and floating-point units to improve power dissipation and hardware efficiency. The reversible ALU (RALU) performs multiple arithmetic and logical (AL) operations. The RALU is constructed using three methodologies: dedicated design approach, control-based approach, and multiplexer (MUX) based approach.

The novel reversible gates are introduced to construct the RALU using an enhanced carry look-ahead adder (CLA) and realize the cost and delay parameters [9]. The quantum circuit-based NCV library is used to construct the n-bit RALU, which performs addition, subtraction, AND, OR, XOR, XNOR, NAND, and NOR operations [10]. The control unit-based RALU is constructed using FG, FRG, and Haghparast Navi gate (HNG), providing a lower propagation delay than the conventional ALU method [11]. The control unit-based 8-bit RALU is built using control output gate (COG), which perform 12 AL operations [12]. Field programmable gate array (FPGA) based 32-bit RALU is designed using basic RGs (FG, PG, FRG, and TG), which performs 8 AL operations and improves the 5.1% power dissipation than the conventional ALU approach [13]. The MUX-based RALU is constructed using FG, FRG, and HNG in the quantum dot cellular automata (QCA) library environment. The design performs 16 AL operations and improves the QC than the existing approaches [14].

As technology advances, more people are using digital electronic devices. The latest generation of digital devices is less expensive, more portable, and faster than their previous ones while using less power. Component size reduction for performance improvement has resulted in increased power losses. Reversible logic circuits are one method to cut down on power dissipation. Reversible logic circuits can meet the needs of high-speed computing because they are often used in quantum computing. Different techniques for ALU are already available in both irreversible and reversible forms. With improved progress in hardware restrictions, multi-functional RALU is the subject of substantially less research. The suggested works offer superior remedies to the issues mentioned above.

In this manuscript, an efficient multi-functional reversible arithmetic and logical unit (MF-RALU) is designed for reduced instruction set computer (RISC) processor and realize the performance metrics in detail. The work is developed using Verilog-HDL with Vivado IDE environment on Artix-7 FPGA. The Artix-7 FPGA is a test platform to implement and prototype the complete work. The contribution of the proposed work is listed as follows:

The proposed 32-bit MF-RALU performs arithmetic, logical complements, shifters, multiplexer, different adders, and multiplier operations. The MF-RALU can perform 30 AL operations, executed individually based on mode input in a single clock cycle latency. The 32-bit reversible adders like reversible ripple carry adder/subtractor (RRCA/RRCS), reversible carry look-ahead adder (RCLA), reversible carry skip adder (RCSKA), reversible carry select adder (RCSA), and reversible kogge stone adder (RKSA) are constructed and used as part of MF-RALU.The different 32-bit reversible multipliers, like reversible array multiplier (RAM), reversible modified booth multiplier (RMBM), and reversible Wallace tree multiplier (RWM), are constructed and used as part of MF-RALU. Lastly, the RISC processor is constructed using MF-RALU to realize the functionality and performance evaluation.

The manuscript is organized as follows: the existing works of the RALU and its approaches are discussed with performance metrics in section 1. The overview of the reversible gates is highlighted in section 2. The proposed MF-RALU and its submodules are discussed along with the RISC processor in section 3. Section 4 discusses the simulation results and their performance metrics. Finally, it concludes the overall work with improvement in section 5.

This section describes the existing works of reversible ALU using different methodologies and for various applications. The RALU is presented by Nayana and Sujatha [15] for 1-bit and 4-bit architectures. Reversible logic circuits are used in work to investigate the 8B/10B encoding and decoding system. The RALU can carry out eight AL functions. To build the RALU, SN gate (SNG), PG, and FGs are used. Comparisons between the 1-bit/4-bit RALU and conventional methods show delay, QC, and GOs improvements. On the Artix-7 FPGA, the encoding and decoding achieve delays of 1.8 ns and 2.06 ns, respectively. Amirthalakshmi and Raja [16] offer the 8-bit RALU design for applications that involve nanoscale processors. Complementary metal-oxide semiconductor (CMOS) technology is used to design the framework. The circuit uses a single electron transistor and includes gates such as AND, OR, NAND, NOR, and NOT gates (single electron transistor (SET)). The RALU can carry out seven AL activities. A SET module employing dual key gate pair (DKGP) gates is utilized in this design. The work examines the outcomes of the simulation and the comparison investigation. Krishna *et al.* [17] introduced 32-bit RALU with low-power and area-optimized functionalities on the FPGA platform. For RALU development, work uses the reversible decoder, adder/subtractor, and multiplexer components. The 351 slices are obtained by the RALU, which uses 1.17 mW of dynamic power. RALU outperforms the irreversible ALU regarding chip area by 91% and dynamic power by 1.6%.

The innovative RALU employing QCA with area reduction is presented by Oskouei and Ghaffari [18]. The work uses double FGs to examine reversible XOR, OR, full adder, and multiplexer. The RALU can carry out four AL activities. Using the QCA environment, simulation outcomes of every sub-module and RALU are explored. RALU's built-in self-test (BIST) is one of the aspects mentioned by Gaur *et al.* [19]. The control unit and FA are used to build RALU. RALU can carry out 14 AL activities. To actualize the fault-tolerant capabilities in RALU, parity preservation features are added. The work compares RALU with current methods showing GC and QC advances. For 1-bit designs, fault-detection-based RALU is described by Bhusal *et al.* [20]. To build RALU, the design allows reversible gates such as controlled not gate (CNOT), FG, double FG (DFG), PG, TG, and FR gates and carries out 16 AL activities. The work evaluates outcome measures, including AI, GO, GC, QC, and simulation.

The 8-bit and 16-bit RALU for low-power solutions on the FPGA platform is presented by Chandralekha *et al.* [21]. FG, PG, and R-I gates are used to develop the control unit-based RALU. On the Virtex-7 FPGA, the 8-bit RALU receives 5.819 W, whereas the 16-bit RALU receives a total power of 8.424 W. The 32-bit RALU with low-power and area-optimized characteristics is presented by Jose *et al.* [22] for DSP applications. Control units and FA are used to build the RALU. The RALU can carry out 12 AL activities. The RALU and traditional ALU are contrasted in work in terms of chip area and power. The RALU draws 220 and 80.01 W of power on the FPGA platform. On the QCA framework, Norouzi et al. [23] demonstrate RALU employing HNG and FRG. Twenty AL activities can be carried out using the RALU. The RALU contains 480 cells, 12 majority votes, and 15 clock phases. The RALU is 6% faster and 28% better than the previous RALU on the QCA framework. The 32-bit ALU is described by Patidar and Gupta [24] using the QCA framework for high-speed computing. The RALU can carry out four AL activities. The design of ALU is set to n bits. For 16-bit and 32-bit systems, ALU uses 149 and 300 meV of energy, respectively. Additionally, the work examines the area, latency, energy consumption, and cell count for various bit designs. The RALU is presented by Safaiezadeh *et al.* [25], employing the QCA platform. Basic reversible gates like FG, FRG, and BS1 block are utilized to build RALU. The BS1 block carries out the operation of majority voter logic. 16 AL activities can be carried out using RALU. The RALU contrasts with current ALUs' better QC, latency, and cell count.

2. REVERSIBLE GATES

The quantum gate libraries are used to construct the reversible gates and can perform more than one operation. Many reversible gates were available to build digital circuits for low-power modern computer applications. The list of reversible gates used in the proposed multi-functional RALU is tabulated in Table 1. The table contains reversible gates and their functionality using input and output.

Table 1. List of Reversible gates used in proposed work

Reversible Gates	Inputs	Outputs
Feynman Gate (FG)	(a, b)	$p = a; q = a \oplus b;$
Peres Gate (PG)	(a, b, c)	$p = a; q = a \oplus b; r = ab \oplus c;$
Fredkin Gate (FRG)	(a, b, c)	$p = a; q = a`b \oplus ac; r = a`c \oplus ab;$
MCF Gate (AND)	(a, b, 0)	$p = g_1; q = ab; r = g_2;$
MCF Gate (OR)	(a, 1, b)	$p = g_1; q = a + b; r = g_2;$
Modified Fredkin Gate (MFG)	(a, b, c)	$p = a; q = ab; r = a`b \oplus ac;$
MUX using MFG	(s, b, c)	$p = g_1; q = g_2; r = out;$
MFR Gate	(a, b, c)	$p = a; q = ab` \oplus ac`; r = a`c \oplus ab;$
Universal Gate (UG)	(a, b, c)	$p = (a + b) \oplus c; q = b; r = ab \oplus c;$
COG Gate	(a, b, c)	$p = a; q = a`b \oplus ac; r = bc \oplus b`c`;$

The FG performs XOR operation with a QC of 1 using a 2×2 form. The PG uses a QC of 4 with a 3×3 format. The FRG uses a QC of 5 with a 3×3 structure. The FRG is further used to perform AND OR logical operations with modifications. The multiple controlled Fredkin (MCF) gate acts as reversible AND OR gates with input ports change in FRG. The FRG is further altered and used as MFG and MFR gates with QC of 4 and 5. The MFG acts as a reversible multiplexer (RMUX) by considering the first input port as a select line (s). The universal gate (UG) used a QC of 5 and constructed a 3×3 format to support the most logical operations. The COG gate is built with a 3x3 structure and uses a QC of 4.

The MF-RALU uses a basic reversible adder/ subtractor to construct advanced reversible arithmetic operations. The basic reversible adder/subtractor structure is illustrated in Figure 1. The reversible half adder/subtractor (HAS) is built using one PG and two FGs, as shown in Figure 1(a). The reversible full adder/subtractor (FAS) is constructed using two PGs, and two FGs, shown in Figure 1(b). The reversible HAS and FAS use the QC of 6 and 10, respectively. The control input (as) acts as a select line: If as is '0', it performs addition or subtraction operations in reversible HAS and FAS units.

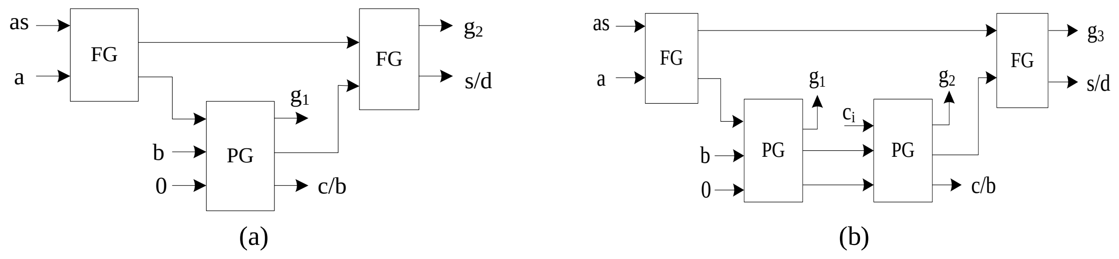


Figure 1. Reversible adder/subtractor used in MF-RALU: (a) reversible HAS (b) reversible FAS

3. MULTI-FUNCTIONAL REVERSIBLE ALU FOR PROCESSOR

The multi-functional reversible ALU (MF-RALU) is constructed using a reversible arithmetic unit, logical unit, complements (1's and 2's), barrel shifter (right and left), multiplexers (MUXs), reversible adders, and multipliers are illustrated in Figure 2. The MF-RALU operations and their functionality is tabulated in Table 2. The 32-bit data width is considered for all the 30 ML-RALU operations on the input side.

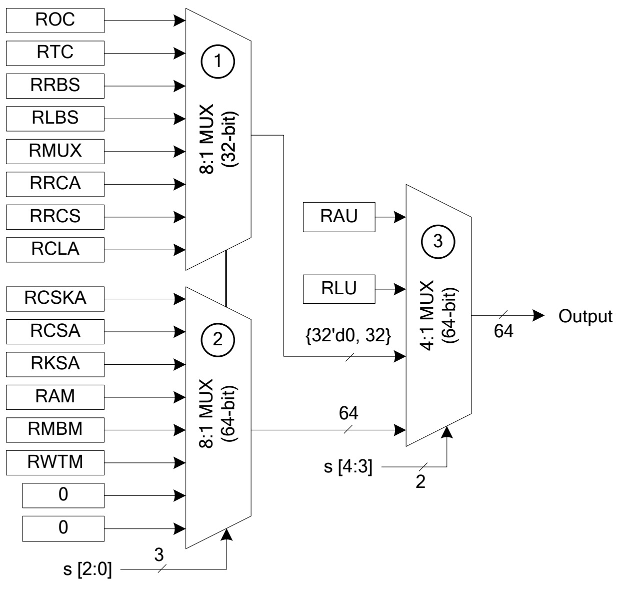


Figure 2. Multi-functional reversible ALU (MF-RALU)

The reversible adders produce 33-bit output, and the multiplier has a 64-bit output. So three MUXs like, (1) 32-bit MUX (8:1), (2) 64-bit MUX (8:1), and (3) 64-bit MUX (4:1), are used to generate the final MF-RALU output. The first MUX performs the eight operations (ALU input from 16th to 23rd), and the second MUX performs the six operations (ALU input from 24th to 29th). These two MUXs outputs, along with reversible arithmetic unit (RAU) and reversible logical unit (RLU) outputs, act as inputs to the 3rd MUX to produce the final MF-RALU output. The RAU performs eight operations and uses 0th to 7th ALU inputs. The RLU performs eight operations and uses 8th to 15th ALU inputs. The RAU and RLU produce 32-bit output individually and are appended with 32 zeros at the MSB side to form 64-bit input to the 3rd MUX. Similarly, The First MUX produces 32-bit output and is appended with 32 zeros at MSB-side to form 64-bit input. The detailed description of the submodules of the MF-RALU is explained in the following sub-sections.

Table 2. MF-RALU operations and its functionality

ALU input	Operations	Function	ALU input	Operations	Function
0	Transfer	B		Multifunctional Operations	
1	Increment	B + 1	16	1's Complement (ROC)	~A
2	Addition	A + B	17	2's Complement (RTC)	~A +1
3	Add with carry	A + B + 1	18	Right Barrel Shifter (RRBS)	A >> Shift
4	Subtract	A` + B	19	Left Barrel Shifter (RLBS)	A << Shift
5	Subtract with barrow	A' + B + 1	20	Multiplexer (RMUX)	s ? A : B
6	Decrement	B - 1	21	Ripple Carry Adder (RRCA)	A + B
7	Transfer	B	22	Ripple Carry Subtractor (RRCS)	A - B
8	Bitwise OR	A | B	23	Carry Look Ahead Adder (RCLA)	A + B + Ci
9	Bitwise NOR	~ (A | B)	24	Carry Skip Adder (RCKSA)	A + B + Ci
10	Buffer	A | B	25	Carry Select Adder (RCSA)	A + B + Ci
11	Bitwise AND	A & B	26	Kagge-Stone Adder (RKSA)	A + B + Ci
12	Bitwise Negation	~A	27	Array Multiplier (RAM)	A * B
13	Bitwise XOR	A ^ B	28	Modified Booth Multiplier (RMBM)	A * B
14	Bitwise XNOR	~ (A ^ B)	29	Wallace Tree Multiplier (RWM)	A * B
15	Bitwise NAND	~ (A & B)			

3.1. Reversible arithmetic and logical unit (RALU)

The 1-bit RALU mainly contains RAU, RLU, and reversible MUX (2:1). The representation of RALU is illustrated in Figure 3. The 3-bit select line (s [2:0]) performs the corresponding arithmetic and logical operations. The 2:1 RMUX receives the inputs from RAU and RLU and provides the RALU output (fo) based on select line (s [3]). The block diagram of the 1-bit RAU is shown in Figure 4. The RAU contains one MFR gate and two PGs. The RAU has two inputs (a and b) with a 3-bit select line (s) at the input side. The RAU produces the AU output (fo) and carry output (co) at the output side. The eight arithmetic operations are performed based on the 3-bit select line in RAU as per Table 2. The RAU uses three reversible gates and one AI and produces the four GO.

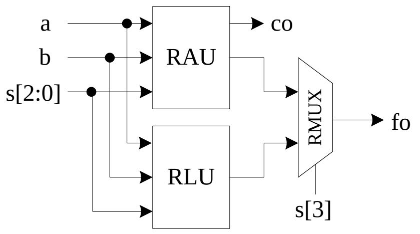


Figure 3. 1-bit Reversible ALU

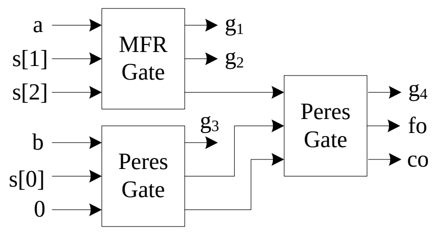


Figure 4. 1-bit Reversible Arithmetic Unit (RAU)

The block diagram of the 1-bit RLU is shown in Figure 5. The RLU contains five FGs, one PG, one UG, and seven COG gates. The RLU has two inputs (a, b) with a 3-bit select line (s) at the input side. The RLU produces the LU output (fo) at the output side. The eight logical operations are performed based on the 3-bit select line in RLU as per Table 2. The RLU uses 14 reversible gates with 7 AI and produces 11 GO. The RAU uses the 3-bit select line, which acts as ALU input from 0 to 7 in MF-RALU. Similarly, The RLU uses the 3-bit select line, which acts as ALU input from 8 to 15 in MF-RALU. The carry output (co) of the 1-bit RAU acts as 1st select line (s [0]) in the next 1-bit RAU and continued further to construct the 32-bit RAU. The 32-bit RLU is built by using a 1-bit RLU 32-times in parallel. The 32-bit RALU is made using 32-bit RAU and RLU along with 32-bit MUX (2:1). The same applies to constructing the n-bit RALU. The different reversible adders and multipliers are discussed in the next section.

3.2. Reversible adders

The different reversible adders like RRCA/RRCS, RCLA, RCSKA, RCSA, and RKSA are discussed in this section. The 32-bit RRCA/RRCS is constructed using a single HAS and thirty-one FAS units. This work does not consider carrying input (ci) using a single HAS. The carry output (c/b) of the 1st FAS carried input to the 2nd FAS and continued till the 31st-FAS to generate the final carry output. Each HAS/FAS produces the individual sum/borrow output. The control input (as) represents '0' for addition and '1' for subtraction operation.

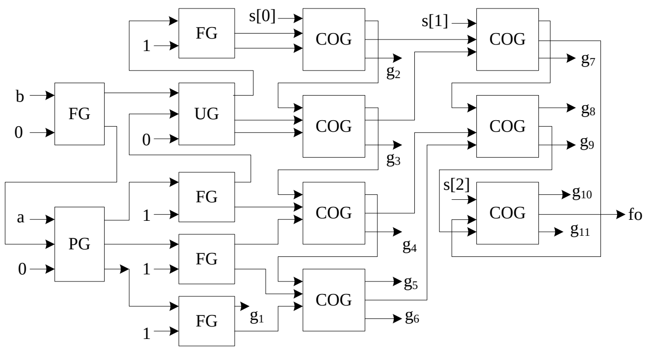


Figure 5. 1-bit Reversible Logical Unit (RLU)

3.2.1. RCLA

The n-bit RCLA is illustrated in Figure 6. It mainly contains the *n* number of FAS units and RCLA logic unit. The RCLA logic unit produces generate, propagate, and next carry generation bits. Each next carry generation bit from the RCLA logic unit is used further in the following FAS as a carry input. The same operation is processed till the last carry output generation.

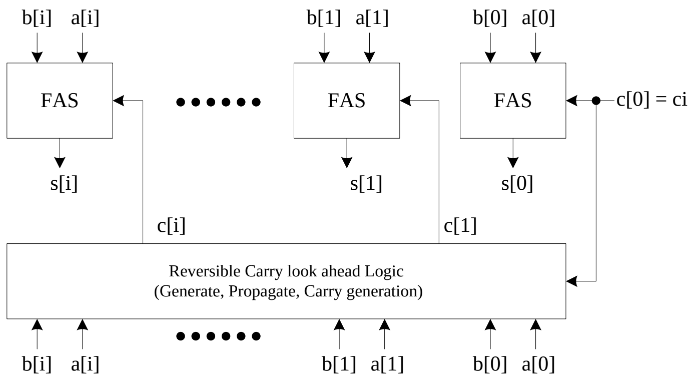


Figure 6. Reversible CLA (n-bit)

In this work, 'n' is fixed to 32, and 'i' is varied from 0 to 31. The (1) to (3) represent generating, propagating, and next carry generation bits. The generate bits are designed using MCF gate (AND) and propagate bits are designed using MCF gate (OR). The carry generation bits are generated using MCF (AND and OR) gates.

$$\text{Generate:}\quad g[i] = a[i].b[i] \tag{1}$$

$$\text{Propagate:}\quad p[i] = a[i] + b[i] \tag{2}$$

$$\text{Carry Generation:}\quad c[i+1] = g[i] + p[i].c[i] \tag{3}$$

3.2.2. RCSKA

The RCSKA is an adder to improve the worst-case delay scenarios. 32-bit RCSKA is illustrated in Figure 7. It mainly contains eight 4-bit FAS units and eight 4-bit RCSL units. The 4-bit RCSL unit uses both MCF (AND and OR) gates. The carry output (co) of the 4-bit RCSL unit is represented in (4). The 1st 4-bit FAS produces the carry output, acting as '*ca*' in the 1st 4-bit RCSL unit. The 1st 4-bit RCSL unit has to carry skip output (*cs*), which carries input into the 2nd FAS and RCSL units. The process continued until the 8th RCSL unit, which generates the final carry output (co). The eight 4-bit FAS units produce the final 32-bit sum output after concatenation.

$$\mathrm{co} = (((\mathrm{a}[0] + \mathrm{b}[0]).(\mathrm{a}[1] + \mathrm{b}[1])).((\mathrm{a}[2] + \mathrm{b}[2]).(\mathrm{a}[3] + \mathrm{b}[3]).\mathrm{ci} + \mathrm{ca} \quad (4)$$

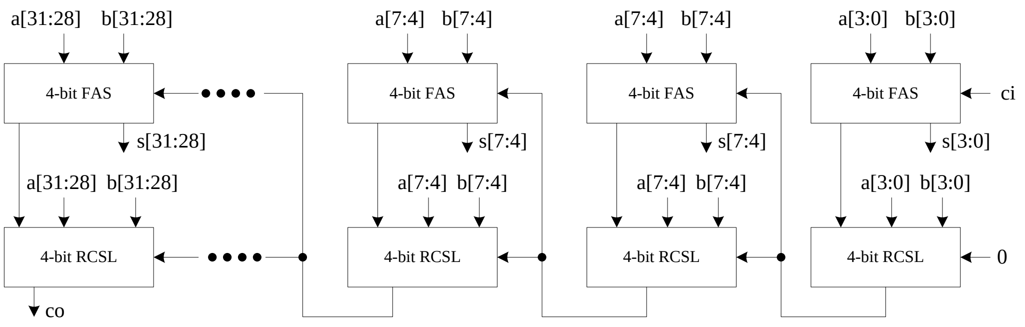


Figure 7. 32-bit RCSKA

3.2.3. RCSA

The RCSA is the fastest adder among other adders. The 8-bit RCSA is illustrated in Figure 8. It mainly contains four 4-bit FAS and four multiplexers. The first and second 4-bit FAS produces two intermediate sum and carry outputs by providing '0' and '1' as carry inputs, respectively. The first MUX receives two intermediate sum outputs and produces the first 4-bit of RCSA sum output (s [3:0]). Similarly, the second MUX receives two intermediate carry outputs, making carry input to the next stage MUXs. The exact process is repeated one more time to generate the next 4-bit of RCSA sum output (s [7:4]) and final carry output (co). The 32-bit RCSA is constructed using four 8-bit RCSA. Each 8-bit RCSA carry output acts as input to the next 8-bit RCSA. The 4th 8-bit RCSA produces the final carry output of 32-bit RCSA.

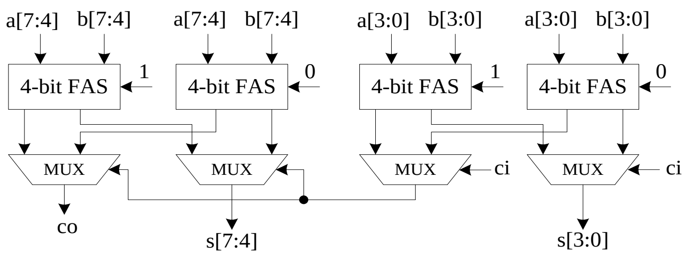


Figure 8. 8-bit RCSA

3.2.4. RKSA

The RKSA is one of the parallel prefix adders, which improves combinational delay more than the above adders. The 32-bit RKSA is constructed using one 32-bit Square box (SB), five big circles (BCs), seven small circles (SCs), and two Triangle (T) units. The SB produces the intermediate generate (g) and propagate (p) bits. The 30-bit, 28-bit, 24-bit, 16-bit, and 1-bit BCs create the final generate (g) and propagate (p) bits. The three 1-bit, 2-bit, 3-bit, 7-bit, and 15-bit SCs produce the final carry output (co). The 1-bit Triangle (T) and 31-bit Triangle (T) calculate final 32-bit sum output. The functional description of SB, BC, SC, and T units are represented in (5) to (8):

Square box (SB): Propogate $\rightarrow p[i] = a[i] \oplus b[i]$; Generate $\rightarrow g[i] = a[i].b[i]$ (5)

Big Circle (BC): Propogate $\rightarrow p = p[i].p[i-1]$; Generate $\rightarrow g = (p[i].g[i-1]) + g[i]$ (6)

Small Circle (SC): Carry generation $\rightarrow c[i] = g[i]$ (7)

Triangle (T): Sum $\rightarrow s[i] = p[i] \oplus c[i-1]$ (8)

3.3. Reversible multipliers

The different 32-bit reversible multipliers like RAM, RMBM, and RWM are discussed in this section. The RAM is used to multiply two binary numbers by using an array of half and full adders. The example of 2×2 RAM is illustrated in Figure 9. The 2-bit inputs (a, b) generate the 4-bit product output concerning the four MCF gates (AND) and two HASs. The higher-order RAMs are constructed similarly to 2×2 RAM. The 4×4 RAM uses four 2×2 RAMs, one 4-bit FAS, and two 6-bit FAS units. Similarly, 8×8 RAM is constructed using four 4×4 RAMs, one 8-bit FAS, and two 12-bit FAS units. 16×16 RAM uses four 8×8 RAMs, one 16-bit FAS, and two 24-bit FAS units. Lastly, 32×32 RAM is constructed using four 16×16 RAMs, one 32-bit FAS, and two 48-bit FAS units in a similar 2×2 RAM structure.

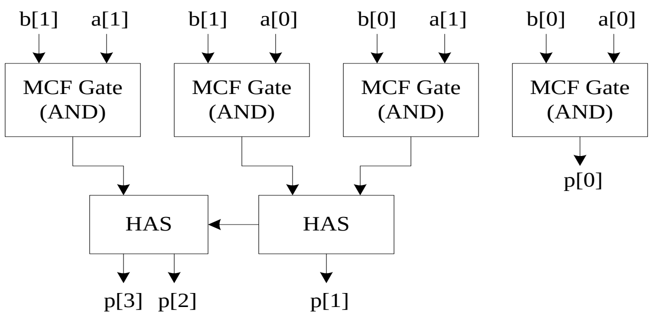


Figure 9. Example of the 2×2 RAM

3.3.1. RMBM

The 32-bit RMBM performs signed and unsigned multiplication operations in two's complement forms. The RMBM improves the area overhead more than other multipliers. It mainly contains a booth partial product generator (BPPG), carry save adder (CSA) based compressor, and 64-bit FAS unit, illustrated in Figure 10.

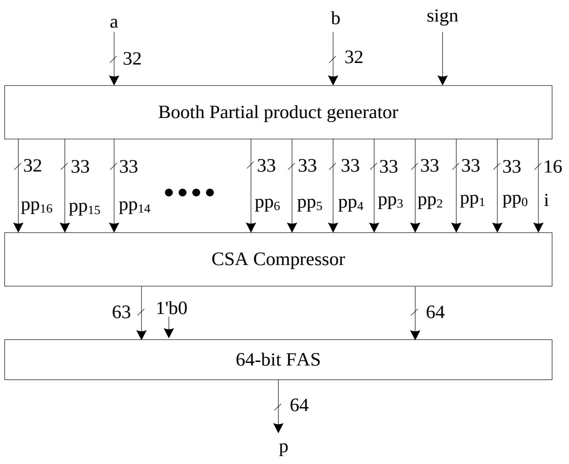


Figure 10. 32-bit RMBM

The 1-bit control signal (sign) performs signed multiplication when it is '1' or unsigned in RMBM. The BPPG generates 17 different partial products (pp_0 to pp_{16}) using a number of FGs and four MUXs. The CSA compressor receives the partial products, generates the 64-bit sum and 63-bit carry bits, and is input to the 64-bit FAS unit. The 64-bit FAS unit produces the final 64-bit product (p) output of RMBM.

3.3.2. RWM

The RWM improves speed and power dissipation more than the other multipliers. The 32-bit RWM contains sixteen 8×8 Wallace tree multipliers (WM8) and fifteen 64-bit FAS to generate final 64-bit product (p). The 8×8 WM is constructed using eight 8-bit MCF gates (AND), sixteen 1-bit HAS, and 47 1-bit FAS to produce the 16-bit product output.

3.4. Reduced instruction set computer (RISC) Processor using MF-RALU

The RISC processor is designed to realize the functionality of MF-RALU, illustrated in Figure 11. It mainly contains fetching unit (FU), a decoding unit (DU), data memory (DM), and MF-RALU. The FU includes a program counter (PC) and instruction memory (IM). The PC receives the processor's 8-bit data input (pro_in) and performs the counting operation. The FU is used to fetch the instruction from IM via PC and pass it to the DU. The DU decodes fetched instruction operands. It generates 5-bit ALU input (alu_in) and two 5-bit operands (R_{dx} and R_{dy}). The DM reads and stores in user read-only memory (ROM) data as per instruction operands in 32 memory locations. The DM produces 32-bit register data (R_x and R_y) and acts as an address value to read the ROM user data. The MF-RALU receives two registers' data and executes the 30 multi-functional arithmetic and logical operations as per Table 2. MF-RALU provides the instruction set to perform different operations. The MF-RALU output is considered the final RISC processor output (alu_out).

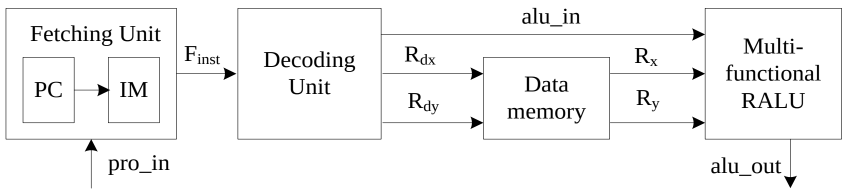


Figure 11. Simple RISC Processor using MF-RALU

4. RESULTS AND DISCUSSION

The results of the 1-bit RALU and MF-RALU for the RISC processor are discussed in this section. The Xilinx Vivado environment with Verilog-HDL is used to construct the design modules. The modules are synthesized on Artix-7 FPGA (XC7A100T-3 CSG324). The hardware performance metrics like area (LUTs), delay, and total power is generated after place and route operation. The simulation results of the 1-bit RAU and RLU in 1-bit RALU is represented in Figure 12 and 13, respectively. The RAU and RLU perform the arithmetic and logical operations as per Table 2 based on select line (s) in 1-bit RALU. The different combinations of two inputs (a and b) are analyzed to understand the functionality of both RAU and RLU in RALU.

The performance summary of 1-bit RALU and its sub-modules are tabulated in Table 3. The RAU utilizes a QC of 13, one LUT with a delay of 1.065 ns. Similarly, the RLU utilizes a QC of 42 and one LUT with a delay of 1.061 ns. The RALU is constructed using RAU and RLU with 2:1 RMUX. So RALU employs GC of 18, AI of 8, GO of 17, QC of 59, and 2 LUTs with a delay of 1.075 ns.

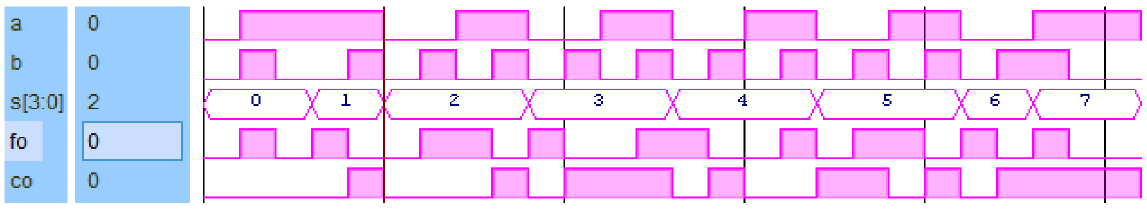


Figure 12. Simulation results of 1-bit RAU

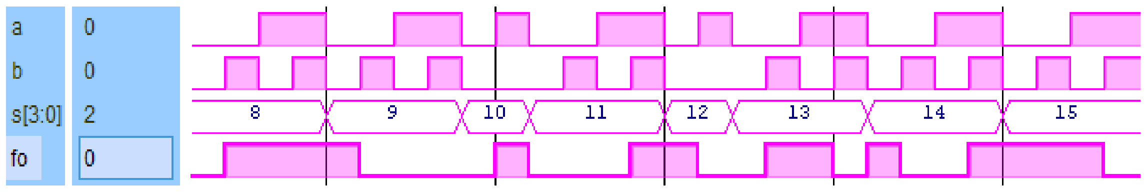


Figure13. Simulation results of 1-bit RLU

Table 3. Performance summary of the 1-bit RALU

Designs	GC	AI	GO	QC	Delay (ns)	LUTs
RAU	3	1	4	13	1.065	1
RLU	14	7	11	42	1.061	1
RALU	18	8	17	59	1.075	2

The proposed 1-bit RALU is compared with existing RALUs, tabulated in Table 4. The comparison includes the number operations, GC, AI, GO, and QC. The RALU [26] is designed by QCA based multiplexers. The work performs seven operations and utilizes a QC of 185. The deoxyribonucleic acid (DNA) based RALU [27] is designed, which performs seven operations and utilizes a QC of 61. The QCA-based RALU [28] performs 11 operations and utilizes a QC of 136. The QCA-based RALU [29] is designed using new reversible gates, which perform 16 operations and utilizes a QC of 94. The fault-tolerant RALU [30] is designed using modified reversible gates, which perform 16 operations and utilize a QC of 99. The proposed 1-bit RALU improves the QC overhead in terms of 68% than Ref [26], 3.2% than Ref [27], 56.6% than Ref [28], 37.3% than Ref [29], 10.6% than Ref [30], and 40.4% than Ref [31]. Overall the 1-bit RALU improves the performance metrics like GC, AI, GO, and QC in most approaches than existing RALU approaches [26]–[31].

Table 4. Comparison of proposed 1-bit RALU with existing RALUs

Designs	Operations	GC	AI	GO	QC
Ref [26]	7	41	10	14	185
Ref [27]	7	20	14	20	61
Ref [28]	11	40	6	11	136
Ref [29]	16	14	3	20	94
Ref [30]	16	22	13	23	66
Ref [31]	16	11	7	22	99
Proposed	16	18	8	17	59

The simulation results of RISC processor using MF-RALU are illustrated in Figure 14. The global clock is activated with active low reset (rst), and the 8-bit processor input is set to zero. The processor produces the 5-bit alu_in from the decoder unit, two 32-bit register outputs from DM, and the final 64-bit output (alu_out) from MF-RALU. For example: If the alu_in is '25', then the functionality of RCSA is performed at the output side.

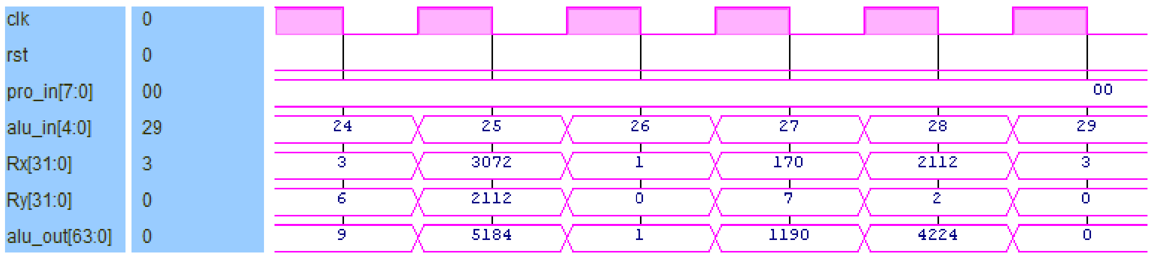


Figure 14. Simulation results of RISC processor using MF-RALU

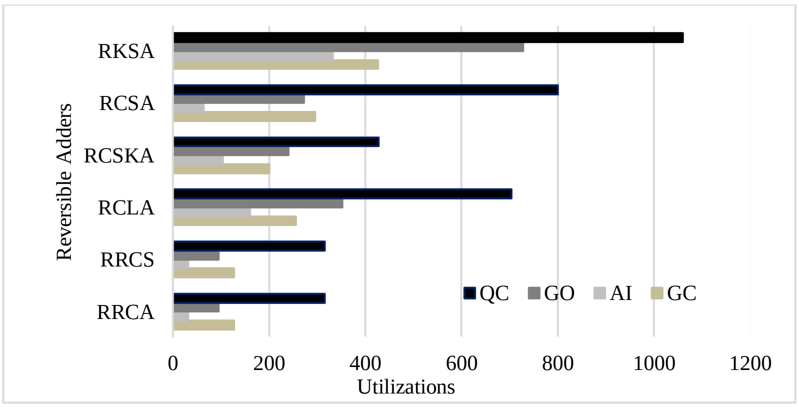
GC, AI, GO, and QC analyze the reversible circuit's performance. The performance summary of 32-bit MF-RALU and its sub-modules are tabulated in Table 5. The 32-bit RLU consumes more QC due to the usage of more gates. The RRBS/RLBS utilizes a QC of 640 for 160 GC. The RRCA/RRCS uses a QC of 316 for 127 GC. The RCLA has more GC due to generating and propagating operations in each stage. The RCLA utilizes a QC of 740 for 256 GC. The RCSKA utilizes a QC of 428 for 200 GC. The RCSA uses a number of RMUX for the carry generation process and generates a QC of 800 for 296 GC. The RKSA has many submodules (SB, BC, SC, and T), uses more GC, and produces a QC of 1060.

Table 5. Performance summary of the 32-bit MF-RALU and its submodules

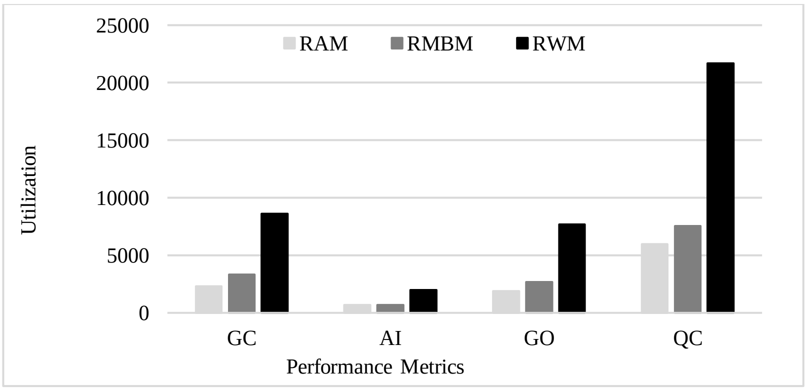
Designs	GC	AI	GO	QC	Designs	GC	AI	GO	QC
RAU	96	32	128	416	RCLA	256	160	352	704
RLU	448	224	352	1344	RCSKA	200	104	240	428
RTC	159	32	95	316	RCSA	296	64	272	800
RRBS	160	0	320	640	RKSA	427	332	728	1060
RLBS	160	0	320	640	RAM	2325	672	1881	5988
RMUX	32	0	64	128	RMBM	3323	679	2692	7538
RRCA	127	32	95	316	RWM	8625	1976	7681	21668
RRCS	127	32	95	316	MF-RALU	17657	4339	17043	45758

The RAM uses 2325 GC with a QC of 5988. The RMBM uses 3323 GC with a QC of 7538. The RWM uses fifteen 64-bit FAS, which produces more GC and QC. The RWM uses 8625 GC with a QC of 21668. The MF-RALU performs 30 operations, including arithmetic, logical, addition, and multiplication. The MF-RALU utilizes GC of 17657, AI of 4339, GO of 17043, and QC of 45758.

The performance analysis of reversible adders and multipliers is represented graphically in Figure 15. The reversible adders are illustrated in Figure 15(a), and reversible multipliers are shown in Figure 15(b). The performance metrics like GC, AI, and GO of the RRCA are better than other reversible adders. The RCSKA utilizes minor GC and produces less QC than RCLA, RCSA, and RKSA. The RAM uses minor GC, AI, GO and GO than RWM and RMBM.



(a)



(b)

Figure 15. Performance analysis of (a) reversible adders (b) reversible multipliers

The resource utilization of 32-bit reversible designs on Artix-7 FPGA is tabulated in Table 6. The RAU utilizes LUTs of 93 with a delay of 21.4 ns and consumes 116 mW of total power. The RLU employs LUTs of 32 with a delay of 21.4 ns and consumes 116 mW of total power. The RRCA/RRCS utilizes LUTs of 77 with a delay of 9.168 ns and consumes 115 mW of total power. The RCLA uses LUTs of 54 with a delay of 10.33 ns and consumes 113 mW of total power. The RCSKA utilizes LUTs of 61 with a delay of 13.112 ns and consumes 113 mW of total power. The RCSA uses LUTs of 67 with a delay of 9.28 ns and consumes 114 mW of total power. The RKSA utilizes LUTs of 114 with a delay of 3.346 ns and consumes 115 mW of total power. The RAM utilizes LUTs of 1909 with a delay of 33.51 ns and consumes 215 mW of total power. The RMBM uses LUTs of 1627 with a delay of 24.285 ns and consumes 238 mW of total power. The RWWM utilizes LUTs of 1923 with a delay of 43.641 ns and consumes 216 mW of total power. Lastly, The MMF-RALU uses LUTs of 6420 with a delay of 43.295 ns and consumes 332 mW of total power. The RCLA utilizes less area (LUTs) and power than other reversible adders. In contrast, the RKSA gives a lesser combination delay than other reversible adders. The RMBM uses less area (LUTs) and combination delay than different reversible multipliers. In contrast, the RAM gives lesser power consumption than other reversible multipliers. The MF-RALU and RISC processor utilizes <11% and <3% chip area, respectively, on Artix-7 FPGA. The RISC processor utilizes LUTs of 1557 and FFs of 78 and works at 483 MHz frequency by consuming 159 mW total power on Artix-7 FPGA.

Table 6. Resource utilization of Reversible Designs on Artix-7 FPGA

Designs	LUTs	Delay (ns)	Power (mW)	Designs	LUTs	Delay (ns)	Power (mW)
RAU	93	21.447	116	RCSA	67	9.28	114
RLU	32	1.061	110	RKSA	114	3.346	115
RRCA	77	9.168	115	RAM	1909	33.51	215
RRCS	77	9.168	115	RMBM	1627	24.285	238
RCLA	54	10.336	113	RWM	1923	43.641	216
RCSKA	61	13.112	113	MF-RALU	6420	43.295	332

5. CONCLUSION

This manuscript uses reversible gates to design an efficient 32-bit multi-functional reversible arithmetic and logical unit (MF-RALU). The MF-RALU performs 30 arithmetic and logical operations, including advanced addition and multiplications. The RISC processor is developed to verify the MF-RALU functionality. The 1-bit RALU is constructed using basic RGs, which perform 16 operations. The 1-bit RALU is compared with existing works with better improvement in reversible performance metrics (GC, AI, GO, and QC). The working operations of different 32-bit reversible adders and multipliers are discussed. The reversible carry skip adder and array multiplier utilize minor GC and QC than other advanced reversible adders and multipliers. The MF-RALU use <11% chip area (LUTs) and consumes 332 mW of total power on Artix-7 FPGA. The RISC processor utilizes a <3% chip area, works at 483 MHz frequency, and consumes a total power of 159 mW. The work can be extended to more operations and configured to higher-bit architecture.

REFERENCES

[1] C. H. Bennett, “Logical reversibility of computation,” *IBM Journal of Research and Development*, vol. 17, no. 6, pp. 525–532, Nov. 1973, doi: 10.1147/rd.176.0525.

[2] R. Khanam, A. Rahman, and Pushpam, “Review on reversible logic circuits and its application,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, May 2017, pp. 1537–1542, doi: 10.1109/CCAA.2017.8230046.

[3] M. Roetteler, K. M. Svore, D. Wecker, and N. Wiebe, “Design automation for quantum architectures,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, Mar. 2017, pp. 1312–1317, doi: 10.23919/DATE.2017.7927196.

[4] S. Thakral and D. Bansal, “A quick guide to implement reversible logic,” in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, Dec. 2018, pp. 1–5, doi: 10.1109/CCAA.2018.8777469.

[5] R. P. Feynman, “Quantum mechanical computers,” *Foundations of Physics*, vol. 16, no. 6, pp. 507–531, Jun. 1986, doi: 10.1007/BF01886518.

[6] A. Peres, “Reversible logic and quantum computers,” *Physical Review A*, vol. 32, no. 6, pp. 3266–3276, Dec. 1985, doi: 10.1103/PhysRevA.32.3266.

[7] E. Fredkin and T. Toffoli, “Conservative logic,” *International Journal of Theoretical Physics*, vol. 21, no. 3–4, pp. 219–253, Apr. 1982, doi: 10.1007/BF01857727.

[8] T. Toffoli, “Reversible computing,” in *Lecture Notes in Computer Science*, Vol 85., Berlin, Heidelberg: Springer, 1980, pp. 632–644.

[9] M. Morrison, M. Lewandowski, R. Meana, and N. Ranganathan, “Design of a novel reversible ALU using an enhanced carry look- ahead adder,” in *2011 11th IEEE International Conference on Nanotechnology*, Aug. 2011, pp. 1436–1440, doi: 10.1109/NANO.2011.6144406.

[10] S. Pal, C. Vudadha, P. S. Phaneendra, S. Veeramachaneni, and S. Mandalika, “A new design of an n-bit reversible arithmetic

logic unit," in *2014 Fifth International Symposium on Electronic System Design*, Dec. 2014, pp. 224–225, doi: 10.1109/ISED.2014.56.
[11] L. Gopal, N. S. Mohd Mahayadin, A. K. Chowdhury, A. A. Gopalai, and A. K. Singh, "Design and synthesis of reversible arithmetic and Logic Unit (ALU)," in *2014 International Conference on Computer, Communications, and Control Technology (I4CT)*, Sep. 2014, pp. 289–293, doi: 10.1109/I4CT.2014.6914191.
[12] A. Deeptha, D. Muthanna, M. Dhrithi, M. Pratiksha, and B. S. Kariyappa, "Design and optimization of 8 bit ALU using reversible logic," in *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, May 2016, pp. 1632–1636, doi: 10.1109/RTEICT.2016.7808109.
[13] S. M. Swamynathan and V. Banumathi, "Design and analysis of FPGA based 32 bit ALU using reversible gates," in *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*, Apr. 2017, pp. 1–4, doi: 10.1109/ICEICE.2017.8191959.
[14] A. Naghibzadeh and M. Houshmand, "Design and simulation of a reversible ALU by using QCA cells with the aim of improving evaluation parameters," *Journal of Computational Electronics*, vol. 16, no. 3, pp. 883–895, Sep. 2017, doi: 10.1007/s10825-017-1004-9.
[15] D. K. Nayana and B. K. Sujatha, "The implementation of reversible gates in a design of 1-bit, 4-bit ALU and 8b/10b encoder and decoder," *International Journal of Applied Engineering Research*, vol. 13, no. 9, pp. 6909–6914, 2018.
[16] T. M. Amirthalakshmi and S. S. Raja, "Design and analysis of low power 8-bit ALU on reversible logic for nanoprocessors," *Journal of Ambient Intelligence and Humanized Computing*, Oct. 2018, doi: 10.1007/s12652-018-1074-y.
[17] G. V. Krishna, G. S. Rao, and Y. A. Babu, "An FPGA implementation of low dynamic power and area optimized 32-bit reversible ALU," *International Journal of Applied Engineering Research*, vol. 13, no. 5, pp. 2926–2932, 2018.
[18] S. M. Oskouei and A. Ghaffari, "Designing a new reversible ALU by QCA for reducing occupation area," *The Journal of Supercomputing*, vol. 75, no. 8, pp. 5118–5144, Aug. 2019, doi: 10.1007/s11227-019-02788-8.
[19] H. M. Gaur, A. K. Singh, and U. Ghanekar, "Design of reversible arithmetic logic unit with built-in testability," *IEEE Design & Test*, vol. 36, no. 5, pp. 54–61, Oct. 2019, doi: 10.1109/MDAT.2019.2919017.
[20] M. Bhusal, R. Rohith, and R. Sakthivel, "Single bit fault detecting ALU design using reversible gates," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Feb. 2020, pp. 1–6, doi: 10.1109/ic-ETITE47903.2020.326.
[21] V. Chandralekha, L. Navya, N. Syamala, and K. Sanapala, "Design of 8 bit and 16 bit reversible ALU for low power applications," in *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)*, Oct. 2020, pp. 477–480, doi: 10.1109/ICCCA49541.2020.9250876.
[22] C. Jose, T. D. Subash, and S. P. Thomas, "FPGA implementation of dynamic power, area optimized reversible ALU for various DSP applications," *Materials Today: Proceedings*, vol. 24, pp. 2044–2053, 2020, doi: 10.1016/j.matpr.2020.03.635.
[23] M. Norouzi, S. R. Heikalabad, and F. Salimzadeh, "A reversible ALU using HNG and Ferdkin gates in QCA nanotechnology," *International Journal of Circuit Theory and Applications*, vol. 48, no. 8, pp. 1291–1303, Aug. 2020, doi: 10.1002/cta.2799.
[24] N. Patidar and N. Gupta, "An extensible architecture of 32-bit ALU for high-speed computing in QCA technology," *The Journal of Supercomputing*, vol. 78, no. 18, pp. 19605–19627, Dec. 2022, doi: 10.1007/s11227-022-04608-y.
[25] B. Safaiezadeh, E. Mahdipour, M. Haghparast, S. Sayedsalehi, and M. Hosseinzadeh, "Novel design and simulation of reversible ALU in quantum dot cellular automata," *The Journal of Supercomputing*, vol. 78, no. 1, pp. 868–882, Jan. 2022, doi: 10.1007/s11227-021-03860-y.
[26] B. Sen, M. Dutta, M. Goswami, and B. K. Sikdar, "Modular design of testable reversible ALU by QCA multiplexer with increase in programmability," *Microelectronics Journal*, vol. 45, no. 11, pp. 1522–1532, Nov. 2014, doi: 10.1016/j.mejo.2014.08.012.
[27] A. Sarker, H. M. Hasan Babu, and S. M. M. Rashid, "Design of a DNA-based reversible arithmetic and logic unit," *IET Nanobiotechnology*, vol. 9, no. 4, pp. 226–238, Aug. 2015, doi: 10.1049/iet-nbt.2014.0056.
[28] T. N. Sasamal, A. K. Singh, and A. Mohan, "Efficient design of reversible alu in quantum-dot cellular automata," *Optik*, vol. 127, no. 15, pp. 6172–6182, Aug. 2016, doi: 10.1016/j.ijleo.2016.04.086.
[29] A. Kamaraj and P. Marichamy, "Design and implementation of arithmetic and logic unit (ALU) using novel reversible gates in quantum cellular automata," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Jan. 2017, pp. 1–8, doi: 10.1109/ICACCS.2017.8014578.
[30] P. Khatter, N. Pandey, and K. Gupta, "An arithmetic and logical unit using reversible gates," in *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, Sep. 2018, pp. 476–480, doi: 10.1109/GUCON.2018.8675034.
[31] A. Kamaraj and P. Marichamy, "Design of integrated reversible fault-tolerant arithmetic and logic unit," *Microprocessors and Microsystems*, vol. 69, pp. 16–23, Sep. 2019, doi: 10.1016/j.micpro.2019.05.009.

BIOGRAPHIES OF AUTHORS

Girija Sanjeevaiah has completed her graduation in Electronics and Communication Engineering from Bangalore University and master's specialization in computer science from Visvesvaraya Technological University, Karnataka. She works as an Assistant Professor in the Electronics and Communication Engineering department of Dr. Ambedkar Institute of Technology. Her areas of interest are Reversible logic, embedded systems, and data computation. She can be contacted at email: girija.pari@gmail.com.

Dr. Sangeetha Bhandari Gajanan is an Assistant Professor in the Electronics and Communication department at RNS Institute of Technology. She received Doctorate and M. Tech degree from Visvesvaraya Technological University and B.E from UBDTCE. Her areas of research interest are low-power VLSI design and thin films. Her teaching and research experience of more than 15 years. She can be contacted at email: sangeethabg@gmail.com.