JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DESIGN AND IMPLEMENTATION OF HALF ADDER AND FULL ADDER USING REVERSIBLE LOGIC GATES

[1*]Samrit Bhowmik, [1#]Rishiraj Dutta, [2]Asima Adak , [3]Anindya Sen

[1*&1#] Student, Assistant Professor, Professor
Department of Electronics and Communication Engineering
Heritage Institute of Technology, Kolkata, India

Abstract : In the realm of digital electronics, efficient and versatile circuit design is paramount. This paper presents a novel approach to designing a Full Adder circuit using unconventional gates, namely the Peres gate, Fredkin gate, NOT gate, and Feynman gate. These gates have distinct characteristics that make them suitable for complex circuitry, offering potential advantages in terms of speed, power efficiency, and scalability. The Peres gate, a three-input quantum gate, allows for reversible computation and is employed to enable arithmetic operations in our Full Adder circuit. The Fredkin gate, known for its ability to perform reversible data swapping, plays a pivotal role in managing carry operations within the circuit. The NOT gate is a fundamental element for inverting binary inputs, while the Feynman gate serves as a versatile unit capable of performing multiple logical operations. This project aims to demonstrate the feasibility of constructing a Full Adder circuit using these unconventional gates and to explore the potential advantages they bring to digital circuitry. Through detailed simulations and physical prototyping, we evaluate the performance of this innovative Full Adder design in terms of speed, power consumption, and scalability. Our findings suggest that the use of Peres, Fredkin, NOT, and Feynman gates in a Full Adder circuit can lead to more efficient and adaptable digital systems, paving the way for advancements in modern computing and signal processing technologies

Index Terms **- Reversible logic gate, Half adder, full adder**

I. INTRODUCTION

Reversible logic gates have recently gained popularity due to their potential for low power consumption, the ability to reduce heat dissipation in digital circuits, low complexity, minimal latency, high speed, less information loss in a circuit, and so on. This property has led to its application in a wide range of developing technologies, including quantum computing, nanotechnology, bioinformatics, cryptography, and so on. Because of the problems caused by ordinary logic gates, reversible logic gates were introduced into the scenario. Many challenges arose as a result of irreversibility, which is best defined as "if an output does not uniquely define the input, the device is said to be logically irreversible." Among the significant concerns with conventional logic gates are elements such as power consumption, heat generation, and information loss, all of which are adverse to the performance.
A Full Adder is a fundamental digital circuit used in computer arithmetic and logic operations. Its primary function is to perform addition on binary numbers, considering both the current bit and the carry from the previous bit. The circuitry of a Full Adder typically consists of three inputs: two binary digits to be added (A and B) and a carry input (Cin), and two outputs: the sum (S) and the carry output (Cout). The circuit combines these inputs to produce the sum of the bits (S) at its outputs while generating a carry output (Cout) that can be propagated to the next Full Adder in a chain for multi-bit addition.
Full Adders are crucial components in digital processors, calculators, and other devices that require arithmetic operations. Their ability to handle carry propagation makes them essential in arithmetic circuits, contributing to the foundation of modern computing. Additionally, Full Adders are used in various applications, including binary arithmetic, data encryption, and even in constructing more advanced components like multipliers and processors.
In 2017, a group worked on the project of [1] designing multiplexer using reversible logic gates. There they explained A Multiplexer (MUX) is a type of combinational logic circuit. It has multiple input lines, one output line, and one or more select lines. A multiplexer with 2N inputs has N select lines that are used to select data or binary information available on any of the input lines and steer it to the output line; thus, the multiplexer is known as a 2N x 1 multiplexer, 2N - to -1 multiplexer, or 2 N: 1 multiplexer and it is known as a Data selector. They used Fredkin gate and RMUX1 gate to make the reversible multiplexer. They proposed a new way of designing multiplexer using reversible logic gates where they reduced the garbage output to 2 and the quantum cost to 4. In May 2023, a group proposed a design of [2] 4:2 priority encoder where they used only one SRK reversible logic gate. SRK gate comprises of 3 logic gates that are AND, XOR and NOT gates. They first used these logic gates to design SRK gate and then the used this SRK gate to design the 4:2 priority encoder. The priority encoder is a combinational logic device with 2n inputs and n outputs. All of the input lines are prioritized in a specific order. When multiple input lines are active at the same time, the output is created based on the priority of the input lines. It is used to resolve issues with binary encoders that result

in improper output when a large number of input lines are active high. If more than one input line is active high at the same time, this encoder prioritizes each input level and assigns the priority level to each input. The output of this encoder corresponds to the input with the highest priority. In order to construct the, only the input with the highest priority is considered. They designed the priority encoder with the lowest quantum cost of 12 and the garbage output of 4. In 2012, another group also did the same project of designing 4:2 priority encoder [3]. There they used 3 Fredkin gates to make this design and if seen more prominently the total number of logic gates used in that project was 21 because a Fredkin gate can be made by using 4 AND gates, 2 XOR gates and 1 NOT gate, in this way 3 Fredkin gates are taken into use that means 21(7*3 = 21) gates are used. This design was little bit less effective than the current one [2]. This design has a total quantum cost of 15 where the current one has the total quantum cost of 12. But it also has total constant input as 3 and total garbage output as 4. In 2017, a group of students designed a reversible full adder [4], where they used total 9 gates which is a combination of 3 different logic gates that are Toffoli gate, Peres gate and Feynman gate. This design had the quantum cost of 28 and garbage output as 7.

In 2020, a student proposed two designs of full adder [5] [A & B], in which the first design[5A] consists of 3 gates which is a combination of one Toffoli gate, one Peres gate and one Feynman gate. This design has a quantum cost of 10 and garbage output as 4. In their second design [5B], they used 3 reversible gates of a combination of 2 Peres gates and one Feynman gate. But this one has a quantum cost of 9 that is 1 lesser than the previous one and garbage output as 5 that is 1 greater than the previous design. In past years, a group of two students introduced [6] a synthesis approach which can cope with large functions. Their basic idea is to create a Binary Decision Diagram for the function to be synthesized and afterwards substituting each node by a cascade of Toffoli or elementary quantum gates, respectively. Since BDDs may include shared nodes causing fan-outs (which are not allowed in reversible logic), also substitutions including an additional circuit line are proposed. In June 2015, a group presented [7] different types of reversible Encoder using Feynman and Fredkin gates. In this paper they calculated quantum costs of different types of reversible encoders. The quantum costs of 4:2, 8:3 and 16:4 reversible encoder is 8,19 and 48 respectively. They are used in various fields such low power VLSI design, optical computing, Nanotechnology, Quantum computer, Design of low power arithmetic and data path for digital signal processing (DSP). In April 2012, a group [8] compared conventional and reversible logic gates. They also realized the addition and subtraction operations using reversible DKG gate and also compared it with conventional logic gates. The author explained that a 4*4 reversible DKG gate can work singly as a reversible full adder and a reversible full subtractor. The author also compared the results of conventional adder with reversible on Xilinx 9.1 software [6].

1.1 About Reversible Logic Gates

The logic of reversibility was developed to address the many problems that emerge with irreversible logic gates, such as information loss, higher power dissipation, and heat creation after computing.

Reversibility in computing means that knowledge of the computational states can always be retrieved and applied as needed.

Reversibility requirements include:

- ***Physical Reversibility****:* Physical reversibility is a process that wastes no energy and generates no heat.
- ***Logical Reversibility:*** Logical reversibility is a process in which any early-stage information can be recovered by computing backward or un-computing the results.

Reversible logic gates are circuits in which numbers of outputs are equal to the number of inputs and there is one to one correspondence between the vector of inputs and outputs, it not only determines the output from the input but also uniquely recovers the input from the output. Some important terminologies [2], [9], [10],[11], in reversible logic gate are:

- **Constant Input:** Constant input are the inputs that is to be maintained at either 0 or 1 in order to synthesize the given logical function.
- **Garbage Output:** Garbage output is the term used to describe the unused output that is present in a reversible logic circuit. Garbage output cannot be avoided because it is necessary to accomplish reversibility.
- **Quantum Cost:** Quantum cost is a cost of circuit with respect to the cost of primitive logic gate.
- **Flexibility:** Flexibility is the universality of reversible logic gate and can be used universally to implement additional functionality.
- **Gate Level:** Gate level is the number of levels present in the circuit which are required to realize the given logic functions.
- **Hardware Complexity:** The total amount of AND, OR, and EXOR operations present in the circuit is referred to as the hardware complexity.

1.2 Some Basic Reversible Logic Gates

There are many types of reversible logic gates in present day, some of them [10], [12], [13] are mentioned below:

NOT GATE –It is a 1*1 gate with quantum cost 0. Its input is defined as "A" and output is defined as "P".

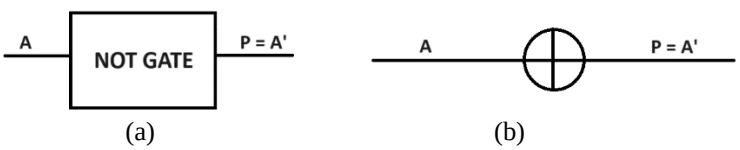
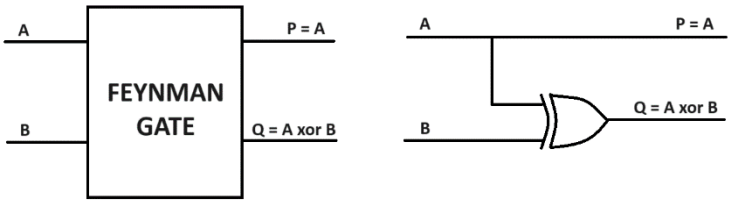


Figure 1: ***Not Gate(a) Block Diagram (b) Schematic Diagram***

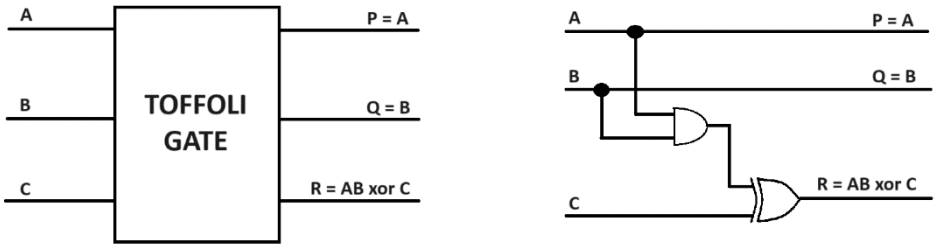
FEYNMAN GATE – It is a 2*2 gate with quantum cost 1. It is also called as Controlled NOT gate (CNOT). Its Input is defined as "A" and "B" and its output is defined as "P" and "Q".



(a) (b)

Figure 2:Feynman Gate (a) Block Diagram (b) Schematic Diagram

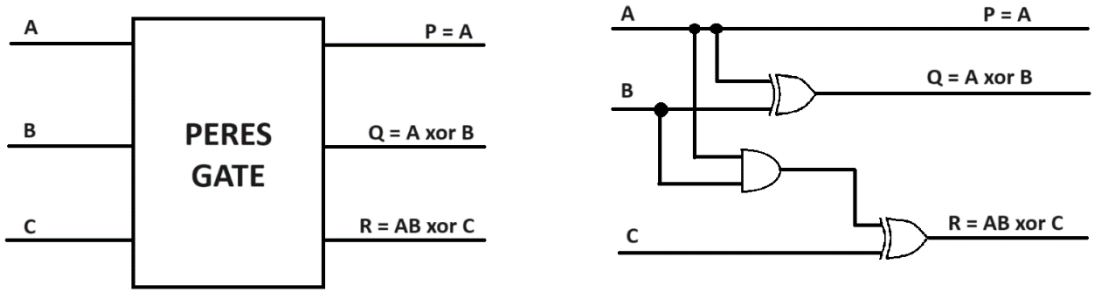
TOFFOLI GATE –It is a 3*3 gate with quantum cost 5. It is a universal logic gate. Its inputs are named as "A", "B" and "C" and its outputs are named as "P", "Q" and "R".



(a) (b)

Figure 3:Toffoli Gate (a) Block Diagram (b) Schematic Diagram

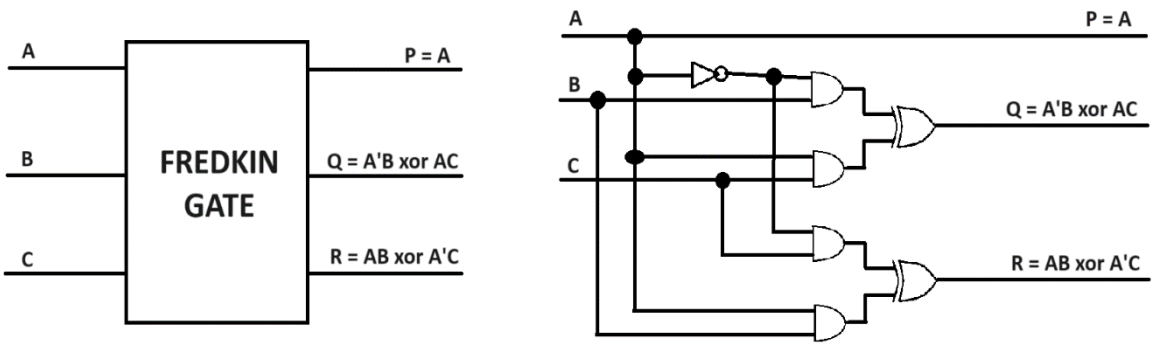
PERES GATE – It is a 3*3 gate and its quantum cost is 4. Its inputs are named as "A", "B" and "C" and its outputs are named as "P", "Q" and "R".



(a) (b)

Figure 4: Peres Gate (a) Block Diagram (b) Schematic Diagram

FREDKIN GATE – It is a 3*3 gate. It has quantum cost 5. Its inputs are "A", "B" and "C" and its outputs are "P", "Q" and "R".

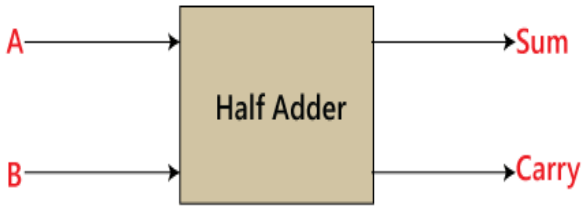


(a) (b)

Figure 5: Fredkin Gate (a) Block Diagram (b) Schematic Diagram

1.3 Half Adder and Full Adder

The Half-Adder is a basic building block of adding two numbers as two inputs and produce out two outputs. The adder is used to perform OR operation of two single bit binary numbers. The **augent** and **addent** bits are two input states, and **'carry'** and **'sum** 'are two output states of the half adder.



Inputs		Outputs	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(a) (b)

Figure 6: Half Adder (a) Block Diagram (b) Truth table

Construction of Half Adder Circuit:

In the block diagram, we have seen that it contains two inputs and two outputs. The **augent** and **addent** bits are the input states, and **carry** and **sum** are the output states of the half adder. The half adder is designed with the help of the following two logic gates:

A) 2-input Exclusive-OR Gate or Ex-OR Gate [14]

B) 2-input AND Gate [15].

A) 2- input Ex-OR Gate:

The **Sum** bit is generated with the help of the **Exclusive-OR** or **Ex-OR**or **XOR** Gate[14].

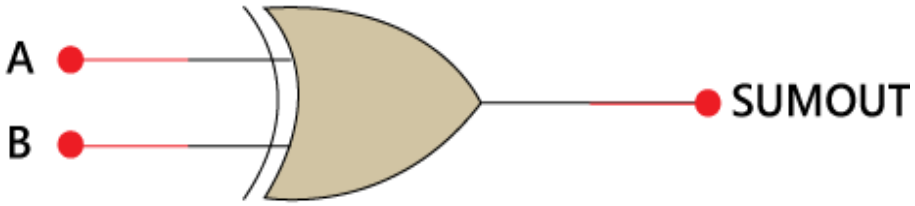


Figure 7: EX-OR(XOR) Gate Schematic Diagram

The above is the symbol of the **EX-OR(commonly said as XOR gate and denoted by ⊕symbol)** gate. In the above diagram, 'A' and 'B' are the inputs, and the 'SUMOUT' is the final outcome after performing the XOR operation of both numbers. The truth table of the EX-OR gate (***Figure 8***)is as follows:

Input		Output
A	B	SUMOUT
0	0	0
0	1	1
1	0	1
1	1	0

Figure 8: Truth table of EX-OR Gate

From the above table, it is clear that the EX-OR gives the result 1 when both of the inputs are different. When both of the inputs are the same, the EX-OR gives the result 0.

B) 2-input AND Gate:

The XOR gate is unable to generate the carry bit. For this purpose, we use another gate called AND Gate [15]. The AND gate gives the correct result of the carry.

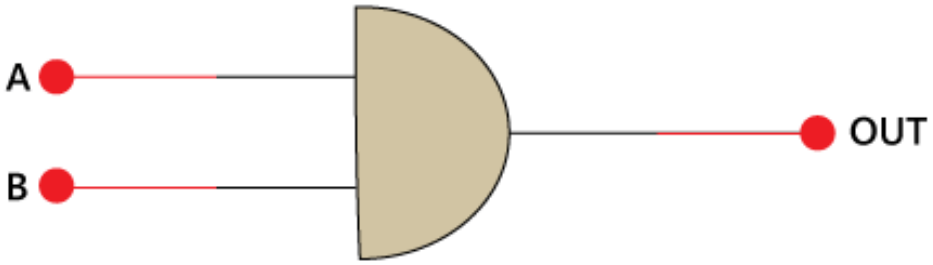


Figure 9: AND Gate Schematic Diagram

The above is the symbol of the **AND** gate. In the above diagram, 'A' and 'B' are the inputs, and 'OUT' is the final outcome after performing AND operation of both numbers.There is the following truth table of AND Gate(***Figure 10)***:

Input		Output
A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

Figure 10: AND Gate Truth Table

From the above table, it is clear that the AND gate gives the result 1 when both of the inputs are 1. When both of the input are different and 0, the AND gate gives the result 0. So, the Half Adder is designed by combining the 'XOR' and 'AND' gates and provide the sum and carry.

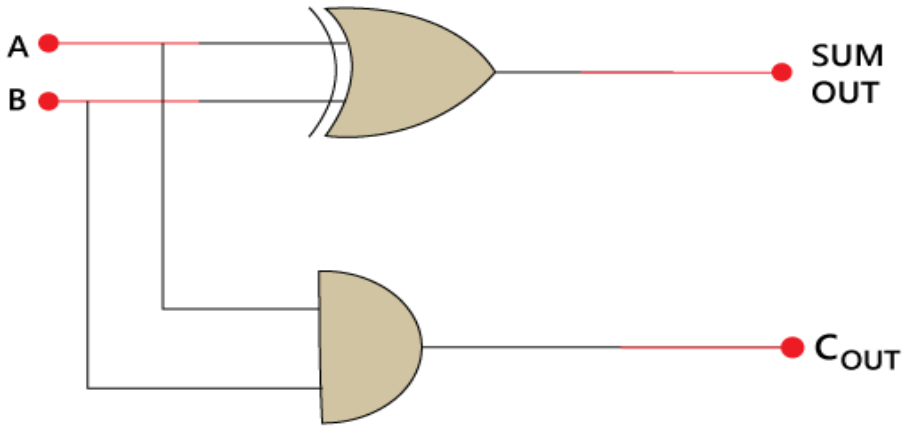


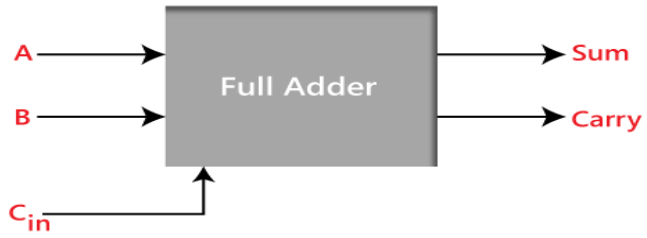
Figure 11:Half Adder Schematic Diagram

There is the following **Boolean expression** of **Half Adder circuit**:

Sum= A ⊕ B

Carry= A . B

The half adder is used to add only two numbers. To overcome this problem, the full adder was developed. The full adder is used to add three 1-bit binary numbers A, B, and carry C. The full adder has three input states and two output states i.e., sum and carry.



(a)

Inputs			Outputs	
A	B	C_{in}	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(b)

Figure 11: Full Adder (a) Simplified Block Diagram (b) Truth Table

In the above table,

1. 'A' and' B' are the input variables. These variables represent the two significant bits which are going to be added
2. 'C_{in}' is the third input which represents the carry. From the previous lower significant position, the carry bit is fetched.
3. The 'Sum' and 'Carry' are the output variables that define the output values.
4. The eight rows under the input variable designate all possible combinations of 0 and 1 that can occur in these variables.

The SOP form can be obtained with the help of K-map as:

Note: In the K-Map(Figure 12), the inputs A, B and C are written as X, Y and Z.

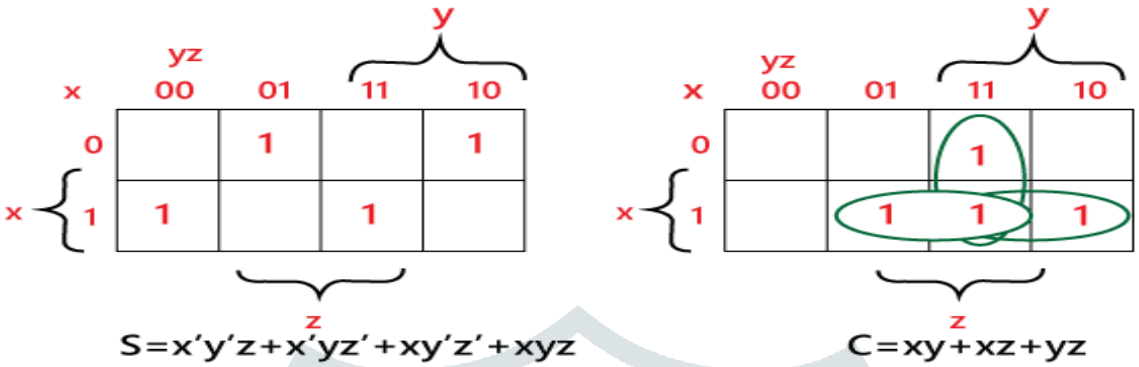


Figure 12: K-MAP For Full Adder (a) SUM (b) CARRY

SUM = X'Y'Z+X'YZ+XY'Z'+XYZ = (X ⊕ Y) ⊕ Z

CARRY = XY+XZ+YZ

Construction of Full Adder Circuit:

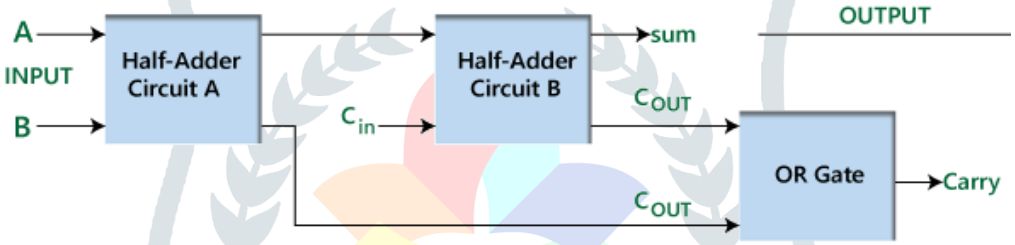


Figure 13:Full Adder Block Diagram

The above block diagram(Figure 13) describes the construction of the Full adder circuit. In the above circuit, there are two half adder circuits that are combined using the OR gate. The first half adder has two single-bit binary inputs A and B. As we know that, the half adder produces two outputs, i.e., Sum and Carry. The 'Sum' output of the first adder will be the first input of the second half adder, and the 'Carry' output of the first adder will be the second input of the second half adder. The second half adder will again provide 'Sum' and 'Carry'. The final outcome of the Full adder circuit is the 'Sum' bit. In order to find the final output of the 'Carry', we provide the 'Carry' output of the first and the second adder into the OR gate. The outcome of the OR gate will be the final carry out of the full adder circuit.The MSB is represented by the final 'Carry' bit.
The full adder logic circuit can be constructed using the **'AND'** and **the 'XOR gate'** with an**'OR gate'**

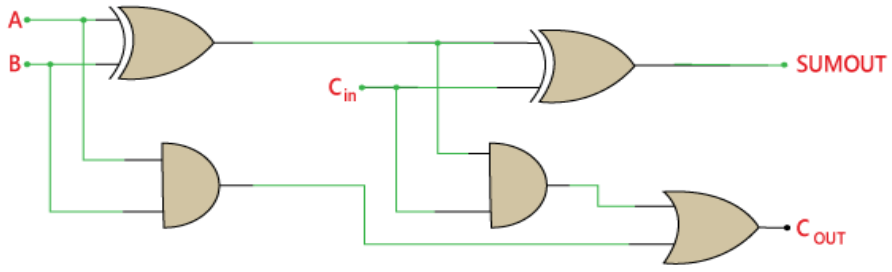


Figure 14: Full Adder Circuit Diagram

The actual logic circuit of the full adder is shown in the above diagram. The full adder circuit construction can also be represented in a Boolean expression.

Sum:

- Perform the XOR operation of input A and B.
- Perform the XOR operation of the outcome with carry. So, the sum is (A XOR B) XOR C_{in} which is also represented as:**(A ⊕ B) ⊕ C_{in}**

Carry:

- Perform the 'AND' operation of input A and B.
- Perform the 'XOR' operation of input A and B.

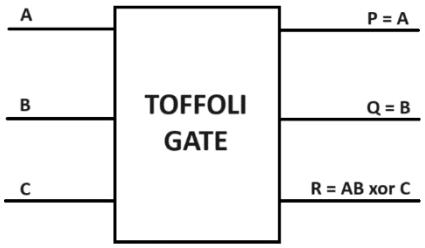
- Perform the 'OR' operations of both the outputs that come from the previous two steps.So the 'Carry' can be represented as:**A.B + (A ⊕ B)**

2. RESULTS AND DISCUSSION

2.1 DESIGN OF HALF AND FULL ADDER USING REVERSIBLE LOGIC GATES

In our design, we used 4 types of reversible logic gates. That are Toffoli Gate, Fredkin Gate and Feynman Gate and Not Gate. From the truth table of Full Adder, we have the outputs:
SUM = X'Y'Z+X'YZ+XY'Z'+XYZ = (X ⊕ Y) ⊕ Z
CARRY = XY+XZ+YZ
So, to understand how to produce **Sum** and **Carry** from Toffoli, Fredkin and Feynman Gate, let us first understand truth table of these 3 gates. First, we will understand the truth table of Toffoli Gate:

(a)



(b)

INPUT			OUTPUT		
A	B	C	P=A	Q=B	R=AB XOR C
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	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Figure 15:Toffoli gate (a) Block Diagram (b) Truth Table

Toffoli gate is made up of two logic gates that is a XOR gate and an AND gate. XOR gate gives the output 1 when any of the two inputs are 1. It gives output as 0 when both of the inputs are either 0 or 1. Another gate that is AND which gives output 1 when both the inputs are 1 and otherwise all the outputs are 0.The Toffoli gate, also known as the Controlled-Controlled-Not (CCNOT) gate, is a three-qubit quantum gate used in quantum computing. It is a controlled gate, meaning that its operation depends on the state of one or more control qubits. The Toffoli gate performs a controlled-controlled-X operation, which means it applies a Pauli-X (bit-flip) gate to the target qubit if and only if both control qubits are in the state |1⟩. Here's how it works: The Toffoli gate has three qubits: two control qubits (usually labelled as C1 and C2 and one target qubit T.

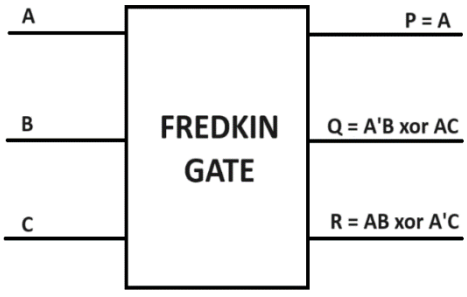
- If both C1 and C2 are in the 1 state, the Toffoli gate applies an X gate to the target qubit T, flipping its state 0 to 1 or vice versa. Mathematically, this can be represented as:$C1C2T \rightarrow C1C2(C1 \oplus C2)T$

Here, ⊕ denotes the XOR (exclusive OR) operation. The XOR result is used to determine whether the target qubit is flipped or not.

- If either or both of the control qubits C1 and C2 are in the 0 state, no operation is performed on the target qubit, and its state remains unchanged.

Now, we will discuss about the 2nd gate that is Fredkin gate.

(a)



(b)

INPUT			OUTPUT		
A	B	C	P=A	Q=A'B+AC	R=AB+A'C
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	1
1	1	1	1	1	1

Figure 16:Fredkin gate (a) Block Diagram (b) Truth Table

The Fredkin gate, also known as the Controlled-Swap (C-SWAP) gate, is a three-qubit reversible gate in quantum computing. It operates as a controlled gate, meaning its action depends on the state of one or more control qubits. The primary function of the Fredkin gate is to swap the states of two target qubits based on the state of a control qubit. Here's how it works:
The Fredkin gate has three qubits: one control qubit (C) and two target qubits (T1 and T2).

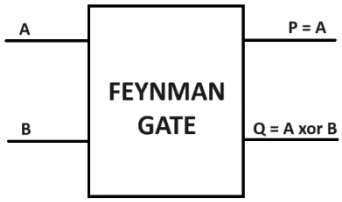
1. If the control qubit (C) is in the 1 state, the Fredkin gate swaps the states of the two target qubits (T1 and T2). In other words, if T1 is in state a and T2 is in state b, after applying the Fredkin gate, T1 will be in state b, and T2 will be in state

a.Mathematically, this operation can be represented as: C T1 T2 →C (T1 ⊕ (C ∧ (T2 ⊕ T1)) (T2 ⊕ (C ∧ (T2 ⊕ T1)).Here, ⊕ represents the XOR (exclusive OR) operation, and ∧ represents the AND operation.

2. If the control qubit (C) is in the 0 state, the Fredkin gate leaves the states of the two target qubits (T1 and T2) unchanged.The Fredkin gate is notable for its ability to perform a conditional swap operation. Depending on the state of the control qubit, it either swaps the states of the target qubits or leaves them as they are.

Lastly, we will discuss about Feynman gate which is used to design a Full Adder.

(a)



(b)

INPUT		OUTPUT	
A	B	P=A	Q=A XOR B
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Figure 17:Feynman gate (a) Block Diagram (b) Truth Table

Feynman gate is a 2*2 reversible gate as shown in Figure 17. The input vector is I(A, B) and the output vector is O(P, Q). The outputs are defined by P=A, Q=A ⊕ B. Quantum cost of a Feynman gate is 1. Feynman Gate (FG) can be used as a copying gate. Since a fan-out is not allowed in reversible logic, this gate is useful for duplication of the required outputs.
Ultimately, we have now understood the mechanisms of all the reversible gates that we used to design a Full Adder. Now its time to see how we have designed our Half and Full Adder Circuits. In the design of Half Adder we used one Toffoli gate and one Feynman gate.

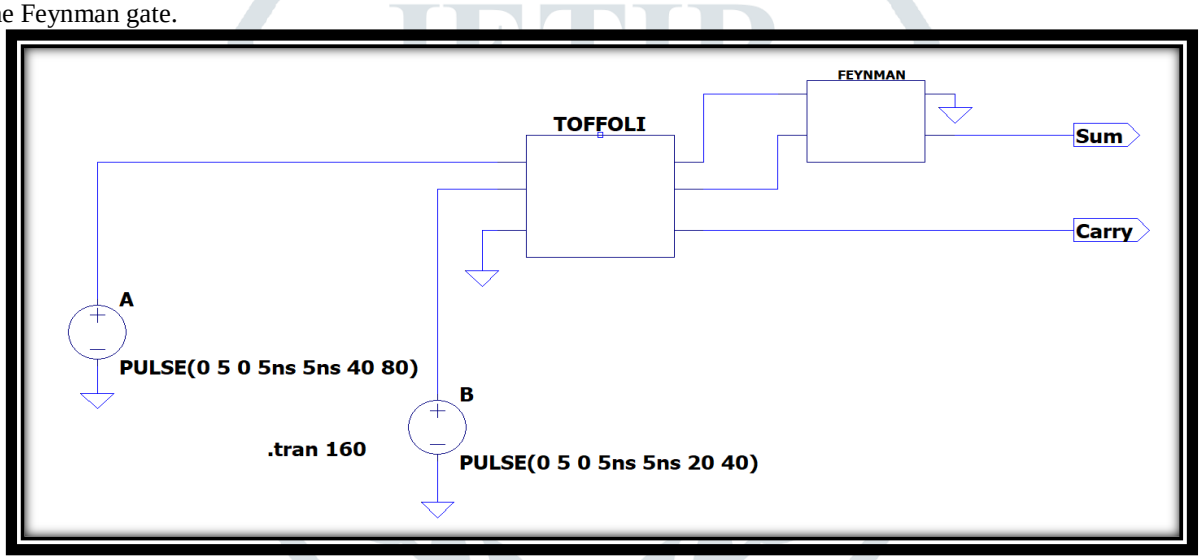


Figure 18: Half Adder Block Diagram

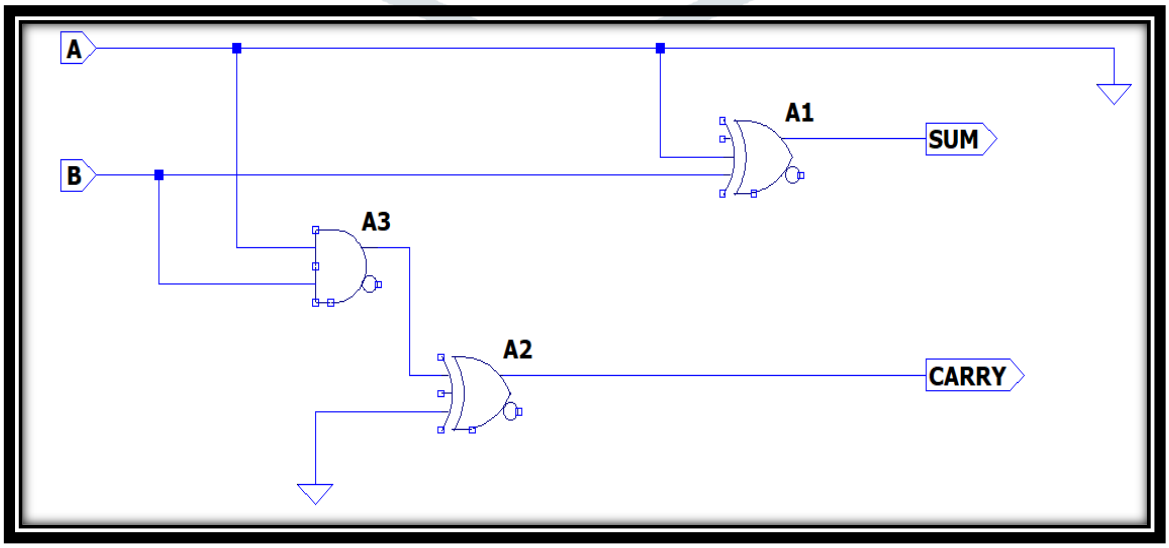


Figure 19: Half Adder Gate level Implementation using LT Spice

Using this half adder circuits we are now going to implement the Full Adder for which we need two of these half adder circuit and one Fredkin Gate to get the carry output.

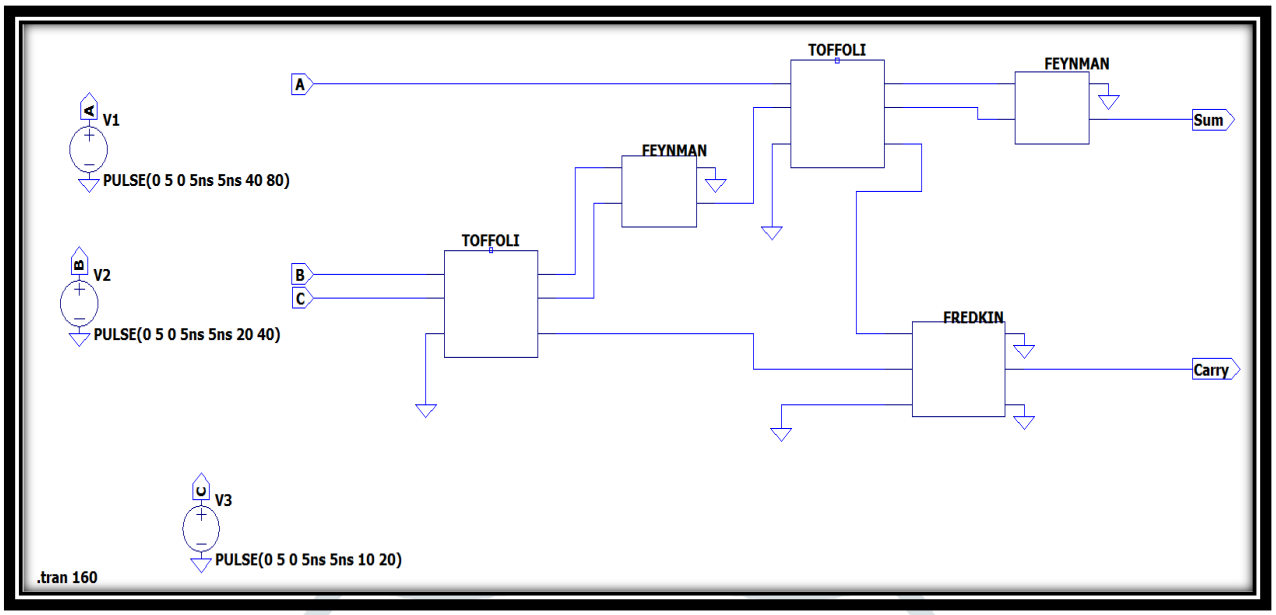


***Figure 20:* Full Adder Block Diagram**

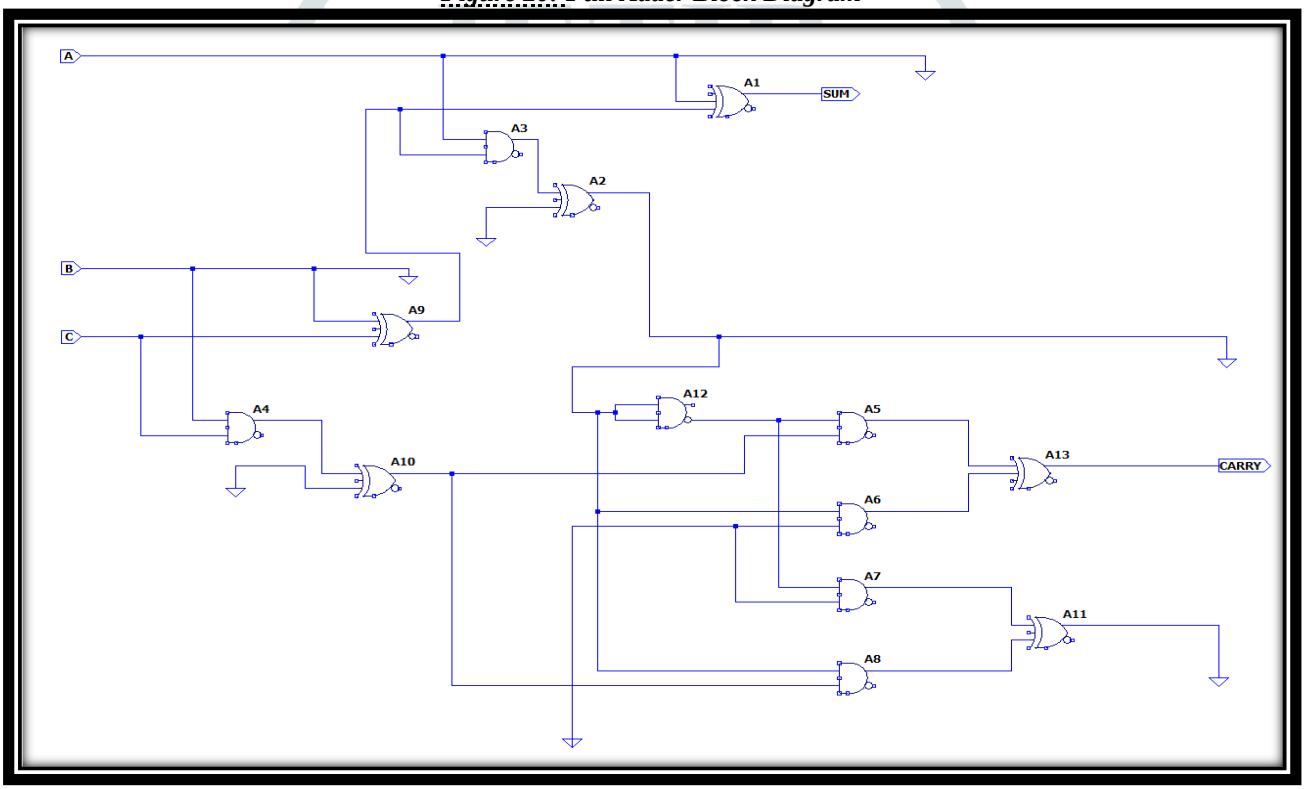


Figure 21:Full Adder Gate level Implementation using LT Spice

RESULTS –

Therefore, the Timing diagrams of the Half Adder and the Full Adder are as follows:
For our convenience the truth table of Half and Full adders corresponding to the timing diagrams are given:

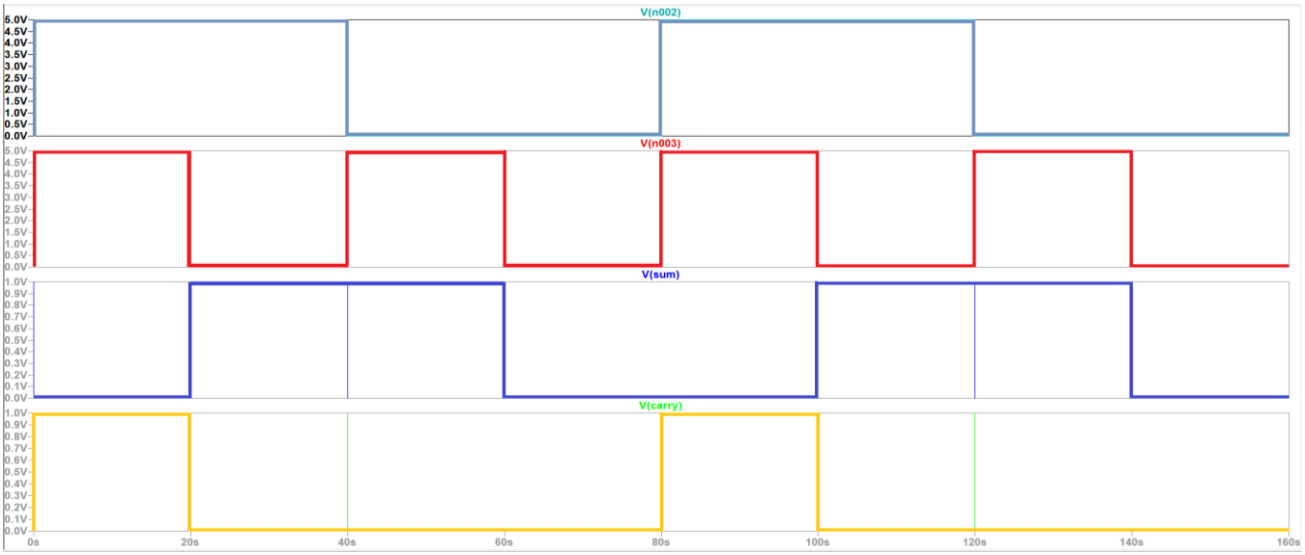


Figure 22:Timing Diagram of Half Adder using LT Spice

INPUT		OUTPUT	
A	B	SUM	CARRY
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0

Figure 23:Truth table of Half Adder corresponding to the timing diagram

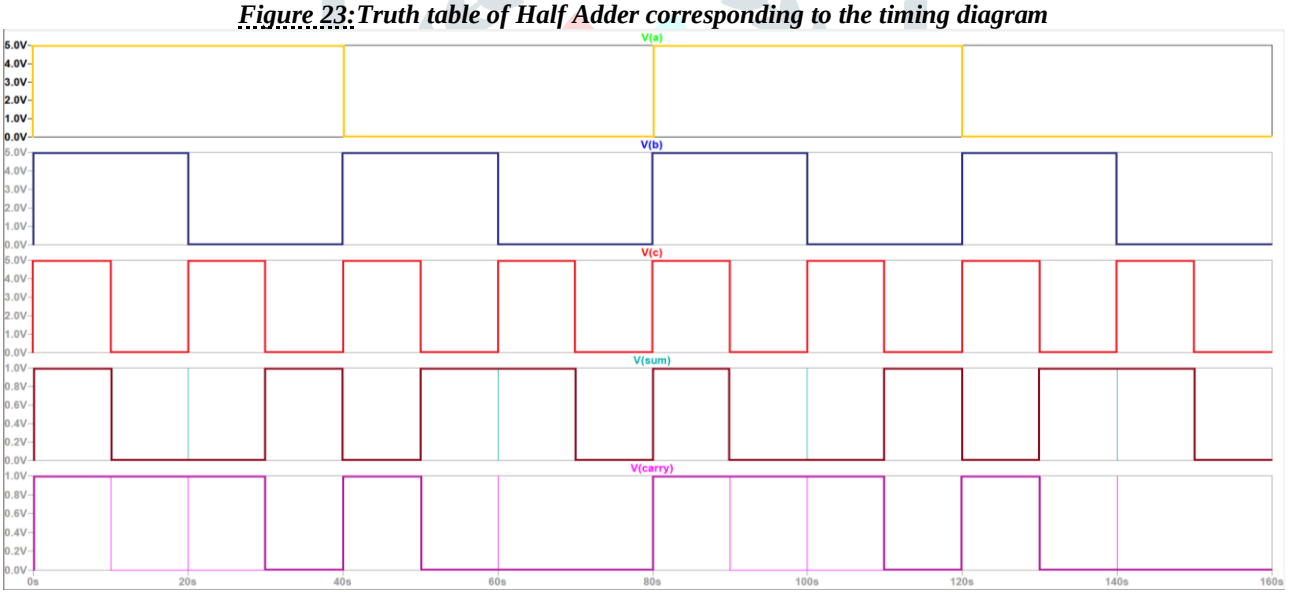


Figure 24: Timing Diagram of Full Adder using LT Spice

INPUT			OUTPUT	
A	B	C	SUM	CARRY
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Figure 25:Truth table of Full Adder corresponding to the timing diagram

Table 1: Comparing our design with the previously done designs (EXISTING DESIGN 4, 5A &5B) of Full Adder

PARAMETERS / CIRCUITS	NO. OF REVERSIBLE GATES USED	TOTAL QUANTUM COST	TOTAL GARBAGE OUTPUT
OUR DESIGN (Using 1 Fredkin, 2 Feynman & 2 Toffoli Gates)	5	17	4
EXISTING DESIGN [4]	9	28	7
EXISTING DESIGN [5A](Using 1 Feynman, 1 Peres & 1 Toffoli Gate)	3	10	4
EXISTING DESIGN [5B](Using 1 Feynman & 2 Peres Gates)	3	9	5

.

II. ACKNOWLEDGMENT

We express our heartfelt gratitude to **Arijit Goswami, Soumik Jana and** the entire faculty of the Electronics and Communication Engineering Department at Heritage Institute of Technology for unwavering support and dedication to our growth.

REFERENCES

[1] Gowthami. P, R.V.S. Satyanarayana, "Design of a Multiplexer Using Reversible Logic", 2017.

[2] Poulami Bera, Manali Bhattacharjee, Sriparna Bhattacharya (Mitra), Asima Adak, Anindya Sen, "Design of priority encoder using reversible logic gates", May 2023.

[3] Y. Syamala, A.V.N. Tilak, and K. Srilakshmi, "Testing of Reversible Combinational Circuits", Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2012.

[4] Ruqaiya Khanam Abdul Rahman Pushpam, "Review on Reversible Logic Circuits and its Application" International Conference on Computing, Communication and Automation (ICCCA2017)

[5](A &B) Sonam Dixit, "Full Adder using Reversible Logic", July 2020.

[6] Robert Wille, Rolf Drechsler, "BDD-based Synthesis of Reversible Logic for Large Functions".

[7] Sukhjeet Kaur, Amandeep Singh Bhandari, "Design and Performance Analysis of Encoders using Reversible logic gates", June 2015.

[8] B. Raghu Kanth, B. Murali Krishna, M. Sridhar, V. G. Santhi Swaroop, "A Distinguish between Reversible and Conventional logic Gates", April 2012.

[9] Subham Chowdhury,Sayantan Bose, Sarupya Chowdhury, Sauvik Mondal, "An Overview on Reversible Logic Gate & their Implementation", June 2017.

[10] Raghava Garipelly, P.Madhu Kiran, A.Santhosh Kumar, "A Review on Reversible Logic Gates and their Implementation", International Journal of Emerging Technology and Advanced Engineering ,Volume 3, Issue 3, March 2013.

[11] Md. Selim Al Mamun, David Menville, "Quantum Cost Optimization for Reversible Sequential Circuit", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 4, No. 12, 2013.

[12] Soham Bhattacharya, Anindya Sen, "A Review on Reversible Computing and it's applications on combinational circuits", June 2021.

[13] Raghava Garipelly, P.Madhu Kiran , A.Santhosh Kumar, "A Review on Reversible Logic Gates and their Implementation", March 2013.

[14] Babu M, Sathish Kumar G A, "In depth survey of XOR gate Design", July 2020.

[15] Mr. Bavit Garg, Ms. Sukhmaneet Kaur, "A review of Logic gates and its applications" May 2019.