ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Optimal Fault-Tolerant F2G and FRG Multiplier Using K-Algorithm

Ganga Bhavana
M.Tech VLSI,
Department of ECE,
CMR Institute of Technology.

Mrs.G.Shanmuga Priya, M.E
Associate Professor,
Department of ECE,
CMR Institute of Technology.

Mr.M.Gurunadha Babu
Professor,
Department of ECE,
CMR Institute of Technology.

Abstract:

Fault-tolerant multipliers with low power consumption are the need of today"s computing systems and reversible circuits have emerged as an efficient solution for ultra-low power design. In this paper, we propose an improved Booth"s recoding algorithm named as K-Algorithm for signed multiplication which reduces the hardware complexity. Efficient multiplier architecture for implementation of K-Algorithm is also proposed. Based on Booth"s recoding algorithm, we design 4-bit reversible multipliers, with and without fault-tolerance. We also design a fault-tolerant reversible multiplier using K-Algorithm. We analyze all the three proposed designs using the reversible logic design metrics: quantum cost, number of ancilla bits, number of garbage outputs, and number of gates. We then compare our reversible multiplier designs with the existing designs reported in literature and find that on an average, our K-Algorithm based fault-tolerant design reduces the quantum cost by 33% and other design metrics are also considerably reduced. Overall, the proposed fault-tolerant reversible multiplier based on Kalgorithm is most optimal as compared to all other designs.

Keywords:

K-Algorithm; Booth"s Recoding; Multiplier Architecture, Reversible Circuits,Fault-Tolerant Reversible Multiplier.

I. INTRODUCTION:

For a past few decades, irreversible circuits comprising of irreversible gates have remained the backbone of the VLSI design. However, Landauer proved that these irreversible circuits generate × 2 joules energy per bit information loss, where is Boltzman"s constant and is the absolute temperature. Reversible circuits have emerged as an efficient solution for ultra-low power design, dissipating zero energy in ideal conditions.

This zero information loss is due to a unique one-to-one mapping between inputs and outputs in a reversible logic enabled circuit. Reversible logic has many applications in low-power CMOS design, quantum computing, DNA computing and nanotechnology. Fault-tolerance is an essential property of reliable VLSI design. Whether irreversible or reversible, these circuits must be able to continue operation even when some of their components become faulty. In reversible logic, fan-outs and loops are not allowed and therefore, parity preserving used in conventional irreversible logic to detect error is hard to implement. In order to prevent errors at the outputs, has introduced the concept of conservative reversible gates with hamming weight of the input equal to that of the output. These gates are also called as fault-tolerant reversible gates. Multiplier circuits are used extensively in computing systems such as DSP hardware.

Therefore, it is necessary to optimize multipliers such that the processing units have high speed low-power consumption Fault-tolerance. In this paper, we propose an efficient and optimal design of a fault-tolerant multiplier using reversible logic. We make the following contributions:

• Our work is motivated by the fact that most of the literature comprises design of reversible multipliers based on conventional multiplication algorithms and there is very less research directed towards improvements in the multiplication algorithm. We show the advantages of improved Booth"s recoding (K-Algorithm) over the conventional Booth"s algorithm.
• We further propose multiplier architecture for implementation of K-Algorithm that results in hardware efficient design.
• We present three different designs: B-ReM (Reversible Multiplier based on Booth"s recoding), BFT-ReM and KFT-ReM (Fault-Tolerant Reversible Multipliers based on Booth"s Recoding and K-Algorithm respectively). To the best of our knowledge, we are the first to emphasize the use of Booth"s recoding in reversible computation.

• Finally, we analyze all the three reversible multiplier designs and compare them with the designs existing in literature. It is established that KFT-ReM design is most optimal in terms of quantum cost and there is a 33% reduction in quantum cost, on an average.

II.REVERSIBLE LOGIC GATES:

A reversible logic gate is an n-input n-output logic device with one-to-one mapping. This helps to determine the outputs from the inputs and also the inputs can be uniquely recovered from the outputs. Also in the synthesis of reversible circuits direct fan-out is not allowed as one–to-many concept is not reversible. However fan out in reversible circuits is achieved using additional gates. A reversible circuit should be designed using minimum number of reversible logic gates.From the point of view of reversible circuit design, there are many parameters for determining the complexity and performance of circuits.

•The number of Reversible gates (N): The number of reversible gates used in circuit.
•The number of constant inputs (CI): This refers to the number of inputs that are to be maintained constant at either 0 or 1 in order to synthesize the given logical function.
•The number of garbage outputs (GO): This refers to the number of unused outputs present in a reversible logic circuit. One cannot avoid the garbage outputs as these are very essential to achieve reversibility.
•Quantum cost (QC): This refers to the cost of the circuit in terms of the cost of a primitive gate. It is calculated knowing the number of primitive reversible logic gates (1*1 or 2*2) required to realize the circuit.
•Gate levels (GL): This refers to the number of levels in the circuit which are required to realize the given logic functions.
•Total Reversible Logic Implementation Cost(TRLIC): Let, in a reversible logic circuit there are NG reversible logic gates, CI constant inputs, GO number of garbage outputs, and have a quantum cost QC. Then the TRLIC is given a Reduction of these parameters is the bulk of the work involved in designing a reversible circuit. In this, an improved design of reversible multiplier with respect to its previous counterparts is proposed. Multiplier circuits play an important role in computational operation using computers.

There are many arithmetic operations which are performed, on a computer ALU, through the use of multipliers. Design and implementation of digital circuits using reversible logic has attracted popularity to gain entry into the future computing technology.

A. Basic reversible logic gates:

•Feynman Gate: Figure1 shows a 2*2 Feynman gate . Quantum cost of a Feynman gate is 1.Feynman gate is called as Controlled NOT gate or CNOT gate. It is equivalent to single control input tofili gate.

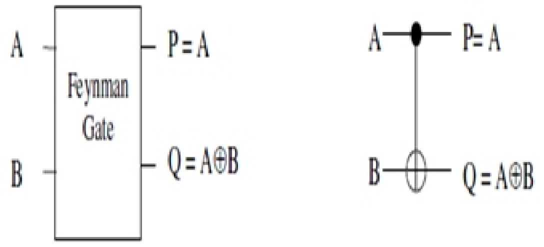


Fig1. Feyman gate and It's Symbolic representation.

•Toffoli Gate: Figure 2 shows a 3*3 Toffoli gate The input vector is I(A, B, C) and the output vector is O(P,Q,R). The outputs are defined by P=A, Q=B, R=A(B xor C). Quantum cost of a Toffoli gate is 5. It has two control inputs.

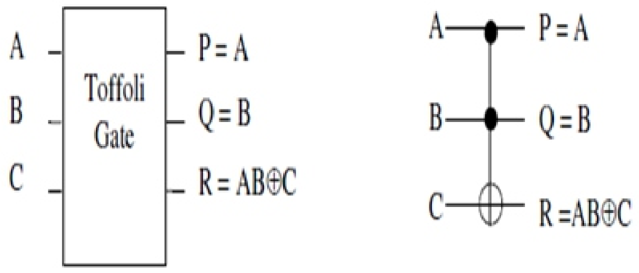


Fig2. Toffoli gate and its symbolic representation.

•Peres Gate:

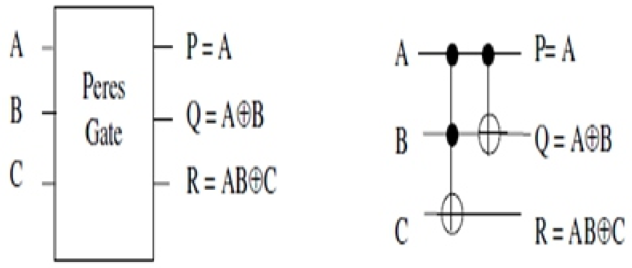


Fig3. Peres Gate.

•BVPPG gate: BVPPG gate is a 5 * 5 reversible gate and its logic diagram is as shown in figure .

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Its quantum cost is 10. Ffoli representation of the BVPPG gate is a shown. The truth table of BVPPG is as shown in the Table -1.

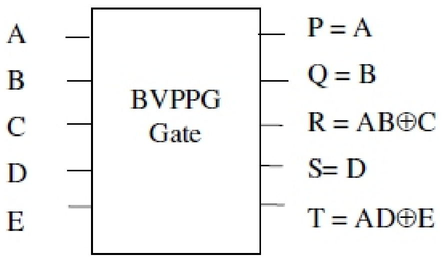


Fig4. BVPPG Gate

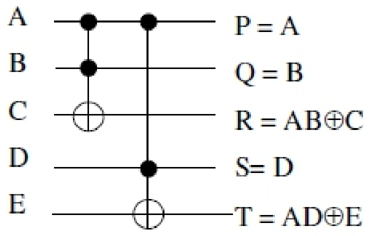


Fig5. Toffoli gate

The BVPPG gate is used to construct the partial product generator which has resulted in least number of gates, least quantum cost and least number of garbage outputs. The two product terms are available at the outputs R and T of the BVPPG gate with C and E inputs maintained constant at 0. The other outputs namely P, Q and S are used for fan-out of the multiplier operands as shown in figure.. This reduces the number of external fan-out gates to zero in our design which is main design feature. The proposed design is compared with the existing designs.

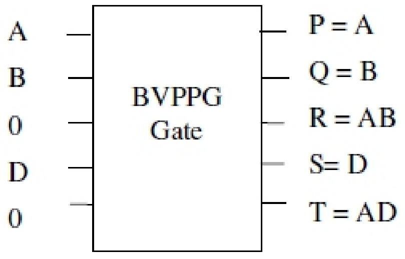


Fig 6. BVPPG Gate Producing Product Terms And Duplication Of Inputs.

•CNOT GATE: CNOT gate is also known as controlled-not gate. It is a 2*2 reversible gate. The CNOT gate can be described as:

Iv = (A, B) ; Ov = (P= A, Q= A B)

Iv and Ov are input and output vectors respectively.m Quantum cost of CNOT gate is 1. Figure shows a 2*2 CNOT gate and its symbol.

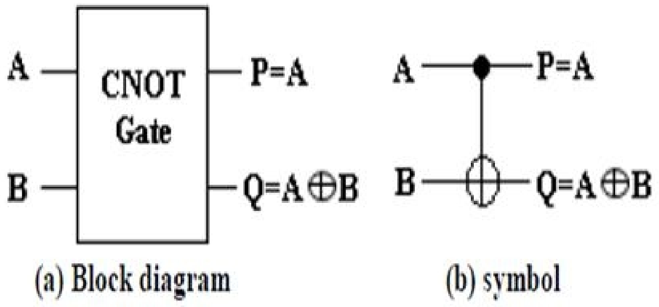


Fig7. CNOT Gate.

•NFT Gate: It is a 3x3 gate and its logic circuit and its quantum implementation is as shown in the figure. It has quantum cost five.

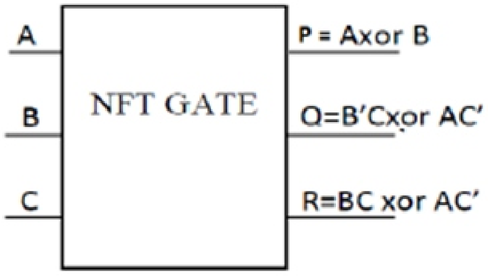
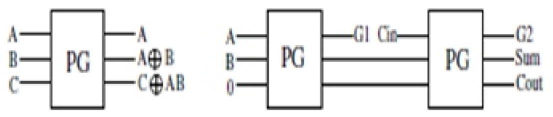


Fig8. NFT Gate.

III. BACKGROUND AND RELATED WORK:

In this section, we start with the basic concepts of reversible logic, quantum realization of reversible circuits, and fault-tolerance in order to understand the fault-tolerant reversible circuit design. Since our focus is to design fault-tolerant reversible multiplier, we also review here the related work reported in literature.

A.Reversible Circuits and Their Quantum Realizations:



(a) (b)
Fig9. (a) Half Adder and (b) Full Adder Using Peres Gates.

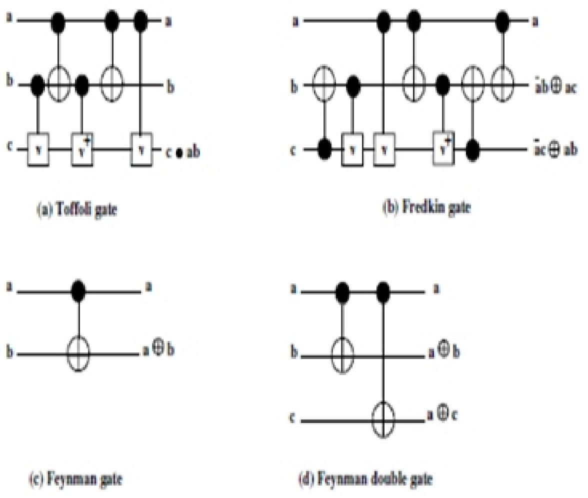


Fig10. Quantum Realization of Prominent Reversible

The reversible circuit design uses the basic principle of reversible logic which states that a logic function:→of n -input pattern = {   , . . . ,   } is reversible iff its number of inputs (n) is equal to number of outputs (k) and there is a unique mapping (called bijective mapping) between the input and output patterns. An irreversible function with inputs and outputs can be made reversible by adding constant input(s) with preset value(s) and garbage output(s). Reversible circuits are the realizations of reversible functions in which fan-outs and loops are not allowed. As a result, a reversible circuit can be constructed by cascading of reversible gates. The value of target bit is determined by the patterns of the control bits. Though a large number of reversible gates have been reported in literature [5], the most promising amongst them are Toffoli gate [6], Fredkin gate [7], Feynman gate [8] and Feynman double gate (F2G) [9]. These gates are shown in Fig9. We also consider Peres gate which is used in efficient realization of half-adder and full adder. Reversible half-adder and full adder designs using Peres gates are shown in Fig10. The concept of reversible logic plays an important role in realization of quantum circuits which are inherently reversible.

The cost of a reversible circuit is an important design metric and is known as quantum cost. The quantum cost of a reversible circuit is defined as the number of quantum operations required to realize the circuit. It depends on the number of control lines in each gate. In quantum circuits, each circuit line is represented by a qubit instead of a bit and its state can be expressed as $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $|0\rangle$ and $|1\rangle$ denote pure logic states 0 and 1, respectively, and α and β are complex numbers such that $|\alpha|2+ |\beta|2 = 1$. In view of the properties of quantum circuits, any reversible gate can be decomposed into a cascade of quantum gates, namely NOT gate (inversion of a single qubit), CNOT (controlled-NOT) gate in which the target qubit is inverted if the single control bit is 1, controlled V gate (square root of NOT), and controlled V+ gate to perform the inverse operation of V gate. Moreover, each such gate has a quantum cost of 1 which means that the cost of a quantum circuit is equal to the number of quantum gates used. Figure 3 shows the quantum realizations of some of the prominent reversible gates. The quantum costs of Feynman Gate, Fredkin gate and Feynman Double Gate are 1, 5 and 2 respectively. Other important design metrics in reversible computing are number of gates, ancilla bits (constant inputs), garbage outputs etc.

B. Fault-Tolerance:

Fault-tolerance is an important mechanism in designing fault-tolerant reversible circuits in order to increase reliability. A reversible gate is fault-tolerant if the Hamming weight (number of logical ones) of its input(s) equals the Hamming weight of its output(s). If the input and output patterns of a fault-tolerant gate (also called conservative) are $I = \{i1, . . . , in\}$ and $O = \{o1, . . . , on\}$, then the following properties must be preserved:

$$I \leftrightarrow O \tag{1}$$

$$i_1 \oplus i_2 \oplus \ldots \oplus i_n = o_1 \oplus o_2 \oplus \ldots \oplus o_n \tag{2}$$

It can be observed from above that the conservative reversible gates are zero-preserving and ones-preserving and effectively permute their input(s) from their output(s). The fault-tolerant gates are also known as Parity Preserving gates and the popular ones are Fredkin (FRG), Feynman double gate (F2G), New Fault Tolerant Gate (NFT) and MIG which have unique mapping between input and output patterns. Recently designed fault-tolerant full adders using MIG [3] and NFT gates [10] are shown in Figure 4 and 5 respectively. We design fault-tolerant reversible circuits using these gates only.

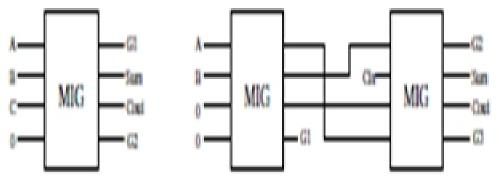


Fig11. Fault-tolerant Adders using MIG gate: (a) Half Adder (b) Full Adder.

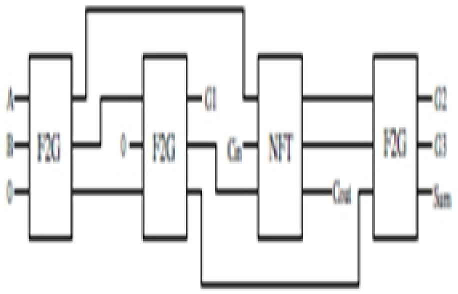


Fig12. Fault-tolerant Full Adder using NFT gate

x_i	x_{i-1}	y_i	Operations
0	0	0	in the middle of string of 0. No operation.
0	1	1	end of string of 1. Add A to partial product.
1	0	-1	beginning of string of 1. Subtract A from partial product.
1	1	0	in the middle of string of 1. No operation .

Table I: Consecutive Bits of Multiplier Vs. Operations:

Popular ones are Fredkin (FRG), Feynman double gate (F2G), New Fault Tolerant Gate (NFT) and MIG which have unique mapping between input and output patterns. Recently designed fault-tolerant full adders using MIG [3] and NFT gates [10] are shown in Figure 4 and 5 respectively. We design fault-tolerant reversible circuits using these gates only.

III.FAULT-TOLERANT REVERSIBLE MULTIPLIER:

In this section, we first present a 4-bit reversible multiplier based on Booth‟s recoding algorithm and then redesign it for fault-tolerance. The choice of Booth‟s algorithm for multiplier design is due to its better performance than other algorithms, especially when there are consecutive 0‟s or 1‟s in the multiplier and when it is a signed multiplication [11]. The multiplication speed can be increased by replacing the corresponding sequence of additions with a subtraction at the least significant end and an addition in the position immediately to the left of its most significant end, i.e.

$$2| + 2|-1 + 2|-2 + \dots + 2K = 2|+1 - 2K \qquad (3)$$

Booth exploits this recoding property in order to perform a fast multiplication. Consider a multiplication $A \times X$, where $A =$ $a-1a-2$ . . . $a1a0$ is the multiplicand and X = $x-1x-2$ . . . $x1x0$ is the multiplier. The multiplier is recoded into Booth‟s recoded form using a conversion truth table, wherein we take $x-1 = 0$. We check every two consecutive bits in X at a time and perform the respective operations using Table I.

The signed value of X can be represented as

$$Val(X) = -x_{n-1} \times 2^{n-1} + \sum_{i=0}^{n-2} x_i \times 2^i \qquad (4)$$

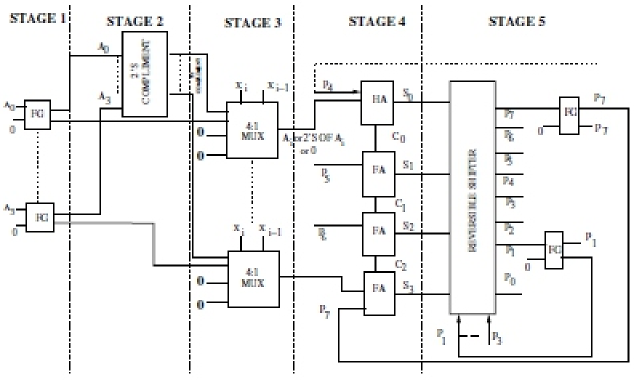


Fig.13. Booth's Recoding based Reversible Multiplier (B-ReM).

$$\begin{aligned} A \times Val(X) &= (x_{-1} - x_0) \times A \times 2^0 + (x_0 - x_1) \times A \times 2^1 \\ &\quad +(x_1 - x_2) \times A \times 2^2 + \dots \\ &\quad +(x_{n-2} - x_{n-1}) \times A \times 2^{n-1} \\ &= A \times [-x_{n-1} \times 2^{n-1} + \sum_{i=0}^{n-2} x_i \times 2^i] \end{aligned} \qquad (5)$$

It should be noted that a negative multiplicand, represented by signed 2‟s complement, needs to be complemented again for subtraction. The design of a 4-bit signed reversible multiplier based on this Booth‟s recoding algorithm is given below.

A. Reversible Multiplier using Booth's Recoding (B-ReM):

The proposed 4-bit reversible multiplier design based on Booth‟s recoding. The design consists of the following five stages:

Stage1: Copying Multiplicand‟s Bits This stage is used to copy each bit of the multiplicand using 4 Feynman gates. The first output of each Feynman gate goes to 2‟s complement block and then the output of 2‟s complement block goes to each multiplexer block. The second output of Feynman gate goes directly to the multiplexer block. This stage has 4 constant inputs and a quantum cost of 4 without any garbage output.

Stage 2: Multiplicand‟s 2‟s Complement Computation In this stage, we first invert each bit of the multiplicand using 4 NOT gates. The quantum cost of each NOT gate is 1. The 2‟s complement is computed by adding 1 to the LSB of 1‟s complement. This requires, as Figure 5.2 shows, 4 half adders for addition and carry propagation. The quantum cost of 1 Peres Gate and hence of a half adder which requires single Peres Gate is 4. This stage uses a total of 8 gates. The quantum cost of this stage comes out to be 20 and the number of garbage outputs and constant inputs are 5 each.

Stage3: Reversible Multiplexer Design In the third stage of the proposed multiplier, we design a reversible multiplexer which performs the required operations as per Table I. Depending on the consecutive.

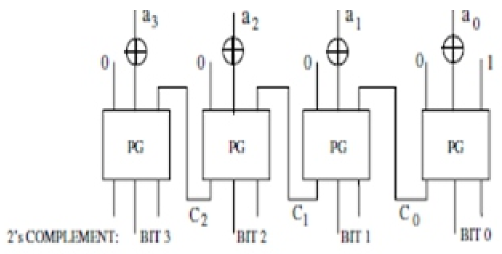
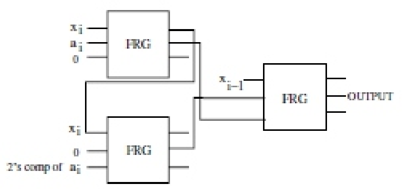


Fig14. Complement Computation of the 4-bit Multiplicand.



bits of multiplier (x_{i-1}), the multiplexer modifies the multiplicand as follows:

- If $x_i = 0$ and $x_{i-1} = 0$ then the input to the adder block is all 0s (0000 in the case of 4-bit multiplier).
- If $x_i = 0$ and $x_{i-1} = 1$ then the input to the adder block is the multiplicand itself.
- If $x_i = 1$ and $x_{i-1} = 0$ then 2‟s complement of multiplicand is the input to the adder block.
- If $x_i = 1$ and $x_{i-1} = 1$ then the input to the adder block is again all 0s. In our multiplexer design, we use three 2: 1 multiplexers

•as shown in Figure 7. Each 2 : 1 multiplexer can be implemented using single Fredkin gate. This multiplexer is used to select one of the above as an input to the adder block. The total number of gates used in this stage is 12. The quantum cost and the garbage outputs of this stage come out to be 60 and 20 respectively. The number of constant inputs are 8.

Stage4: Half Adder and Full Adder Design This stage, consisting of one half adder and three full adders, adds the output of multiplexer block to the four Most Significant Bits (MSBs) of partial product. In this design, we use half adders and full adders realized using Peres Gates as depicted in Figure 2. The quantum cost of full adder is 8. The quantum cost of this stage comes out to be 28. Furthermore, the number of gates used and garbage outputs are 7 each. The number of constant inputs in this stage is 3.

Stage 5: Parallel-In Parallel-Out Reversible Shifter Parallel-In Parallel-Out shifter (PIPO) is used to perform the arithmetic right shift operation on the partial product generated after the addition operation. The PIPO shifter design is shown in Figure 6.1. This shifter is asynchronous and can be implemented using seven Fredkin gates and one Feynman gate. The quantum cost and the number of garbage outputs of this last stage of the multiplier comes out to be 36 and 2 respectively. The number of constant inputs is 3 and the number of gates is 8.

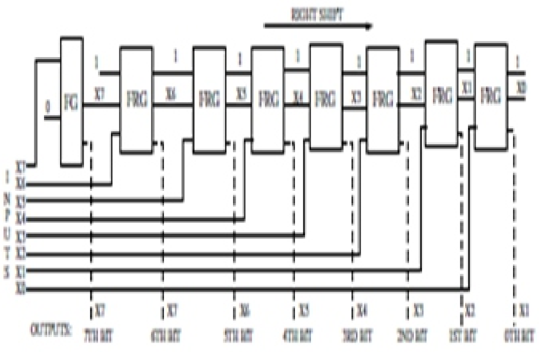


Fig16. Parallel-In Parallel-Out Shifter Design.

This completes our design of Booth‟s recoding based reversible multiplier (B-ReM). It consists of 46 gates, 41 garbage outputs, 30 constant inputs and has a quantum cost of 155. However, this design is not fault-tolerant as it uses non-fault tolerant reversible gates.

B. Fault-Tolerant Multiplier Design (BFT-ReM):

We now propose a fault-tolerant design of the reversible Booth"s multiplier. In this design, we use fault-tolerant reversible gates at each of the five stages shown in Figure 6. Thus, all the stages are converted to their respective fault tolerant counterparts by replacing the non-fault tolerant gates with the fault-tolerant ones. We replace all Feynman gates by Feynman double gates. Further, because NOT gate and Fredkin gate are parity preserving, there is no need to replace them. Finally, to complete the fault tolerant design of the proposed multiplier, we use half adders and full adders implemented using MIG gates. These fault-tolerant adders are shown in Figure 4. Since MIG gate has a quantum cost of 7, the quantum cost of fault-tolerant half-adder and full adder are 7 and 14 respectively. The number of garbage outputs and ancilla inputs of half-adder are 2 and 1 respectively, and those of full adder are 3 and 2 respectively. This completes the design of the fault-tolerant reversible multiplier based on Booth"s recoding algorithm (BFT-ReM). It consists of a total of 46 gates, 60 garbage outputs, 51 constant inputs and a quantum cost of 200.

IV. K-ALGORITHM AND ARCHITECTURE:

In this section, we propose an improved Booth"s recoding algorithm, named as K (Keshuv)-Algorithm. In Booth"s algorithm, depending on the two consecutive bits of multiplier (x_i and x_{i-1}), either the multiplicand or its two"s complement or zero is passed to the adder. Table I shows that the multiplier X is transformed from a binary number with digit set [0, 1] to a binary signed-digit number Y with a digit set [−1, 1] in Booth"s Recoding. However, in our proposed algorithm, we perform the multiplication without using the digit set [−1, 1] for . Instead we use the digit set [0, 1] itself. Table II shows y_{new_i} and the respective operations depending on the consecutive multiplier bits. „–" here represents a don"t care situation. As evident from Table II, in case of the y_{new_i} value of 0 and 1, we pass the multiplicand (A) as it is to the addition stage. However, we perform addition (in the case of y_{new_i} value 0) and subtraction (in the case of y_{new_i} value 1) using an efficient adder/subtractor hardware. It can also be observed that y_{new_i} is same as x_i except for the first and the last case for which y_{new_i} is don"t care. Therefore, we take y_{new_i} as x_i without any alteration. This implies that we are not transforming the multiplier X to any other number. Now, we propose a 4×4 multiplier architecture as shown in Figure 10. Initially, the contents of partial product are $0000a_3a_2a_1a_0$.

Our K-Algorithm does not evaluate the 2"s complement of multiplicand to be sent separately to the multiplexer beforehand. It selects only between multiplicand bits or 0s. This replaces the use of 4 : 1 multiplexer in original Booth"s Algorithm by 2 : 1 multiplexer. The select line used in this multiplexer is $x_i \oplus x_{i-1}$. When the select line i.e. $x_i \oplus x_{i-1}$ is low (for the first and last case of Table II), 0000 is passed, merely shifting the partial product to the right by 1 bit. However, when this line is high (for the second and third case), the multiplicand i.e. $a_3a_2a_1a_0$ is passed to the adder through an XOR gate, exactly in agreement with the table. The carry-in of the full adder and the other input to this XOR gate is y_{new_i} i.e. x_i. The reason behind this choice is that XOR gate is a programmable inverter. For the second case, y_{new_i} is 0 and thus, the multiplicand is not inverted by XOR gate and is passed as it is to the full adder with a carry-in of 0. Hence, it gets added to the partial product. For the third case, y_{new_i} is 1 and thus, the multiplicand gets inverted before getting passed to the full adder. Since, the carry-in is also 1, it leads to subtraction of multiplicand from the partial product (Addition of 2"s complement). Further, in the first and the last case, O is either added or subtracted. This just shifts the partial product by 1 bit as 2"s complement of 0 is again 0.

Table II: Multiplier Consecutive Bits Vs. Operations (Modified):

x_i	x_{i-1}	y_{new_i}	Operations
0	0	–	No operation at all.
0	1	0	End of string of 1's in X. Pass A as it is to the addition stage.
1	0	1	Beginning of string of 1's in X. Pass A as it is to the addition stage.
1	1	–	No operation at all.

The original Booth"s algorithm does not generate the variable carry-in that is required by the adder/subtractor hardware. It evaluates y_i as 1 for addition and as −1 for subtraction (a value that cannot be supplied as carry-in). Our algorithm, on the other hand, proposes y_{new_i} that can be directly supplied as carry-in to the full adder and to the XOR gates. Therefore, the proposed algorithm implements Booth"s Recoding with much smaller hardware. In effect, we are able to replace 4 : 1 multiplexers by 2 : 1 multiplexers and we now save the hardware required to implement 2"s complement unit just by using few XOR gates. This completes the description of the proposed K-Algorithm and the corresponding architecture.

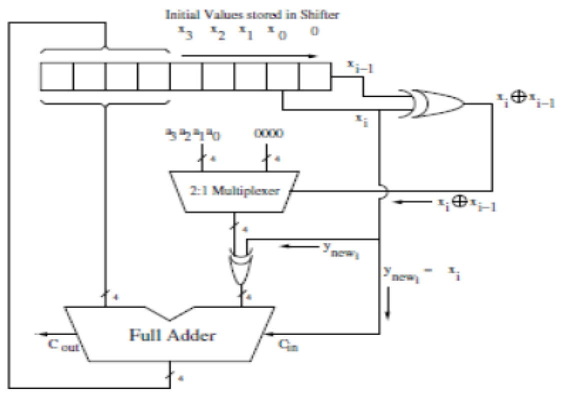


Fig17. Proposed Multiplier Architecture for K-Algorithm Implementation.

A. Optimal Fault-Tolerant Multiplier:

In this section, we propose an optimal and hardware efficient design of a fault-tolerant reversible multiplier based on the proposed K-Algorithm (KFT-ReM). In our design, we use Single NFT gate Full Adder (SNFA). The NFT Gate is fault tolerant with a quantum cost of 5. This adder, as shown in Figure 5, uses a total of 4 gates: 3 Feynman double gates and 1 NFT gate. Thus, this adder has a quantum cost of 11 and is fault-tolerant with 2 ancilla inputs and 3 garbage outputs. Fig7 shows the detailed design of 4 × 4 fault-tolerant reversible multiplier based on K-Algorithm. The exclusive OR function is performed by Feynman Double Gate and each Fredkin Gate serves as a single 2 : 1 multiplexer. We use the PIPO shifter described earlier (Section 3).

The initial contents of the shifter are 0000ı3ı2ı1ı00. The least significant bit is 0 since ı−1 is 0. The figure shows the shifter in its very first stage. As the shifter is asynchronous and Parallel-in Parallel out in nature, its outputs are fed back to the inputs through at least one gate (since forming loops in reversible logic is not permitted). For instance, Fredkin gate output ı4 is fed back to ı3 through a Feynman Double Gate. The proposed multiplier thus, uses a total of 12 Fredkin Gates, 23 Feynman Double Gates and 4 NFT gates i.e. a total of 39 fault-tolerant gates, thereby generating a fault-tolerant design. The number of garbage outputs and ancilla inputs are 34 and 30 respectively. The quantum cost of this multiplier is 126.

V.PERFORMANCE AND OPTIMALITY ANALYSIS:

In this section, we compare the K-Algorithm based reversible multiplier with other reversible multipliers reported in literature. Cost metrics of some of the recent designs of reversible multipliers are tabulated in Table III where, QC signifies Quantum Cost.

Table III shows that the proposed K-Algorithm based reversiblev4×4 multiplier is the most optimal design in terms of quantum cost out of all the existing designs (both fault-tolerant as well as non fault-tolerant). The reduction in quantum cost varies from around 8% to 56% which on an average comes out to be around 33%. The multiplication is fast because the shifter used in this design is asynchronous.

This makes it efficient in time. Since, it requires lesser number of gates when compared to those of multiplier based on the Booth‟s recoding algorithm, [12], [14], [15] and [16] etc., it is efficient in space as well. Some designs, having lesser number of gates [13], are not as space optimized as they appear. This is because they use a number of 4×4 gates (e.g. HNG gate). We, on the other hand, have used only fault-tolerant 3×3 gates in our design, thereby optimizing our design in space. Furthermore, gates like HNG Gate, Peres Gate etc. are not parity preserving and hence their designs are not fault-tolerant.

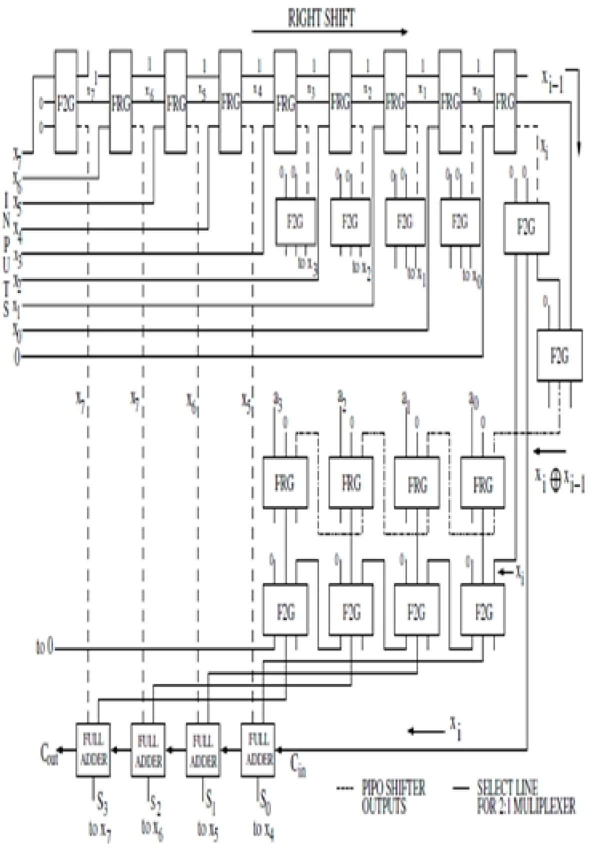


Fig18. Fault-tolerant Reversible 4 × 4 Multiplier (KFT-ReM).

VI. RESULTS:

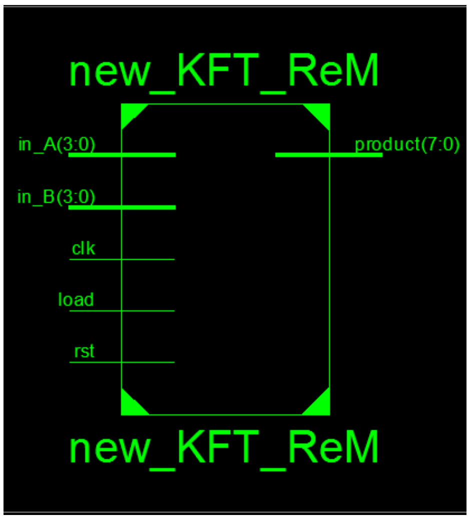


Fig19. RTL Schematic.

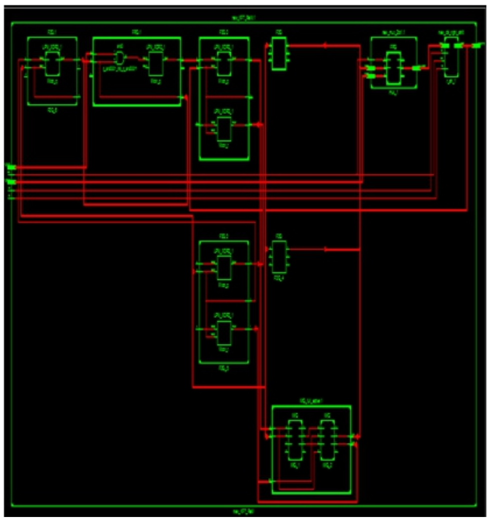


Fig20. Internal RTL Schematic.

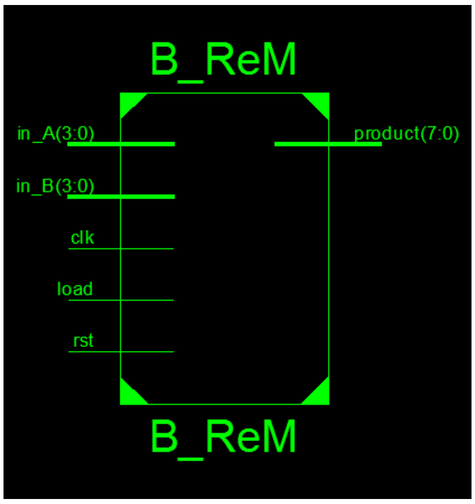


Fig21. Proposed RTL Schematic.

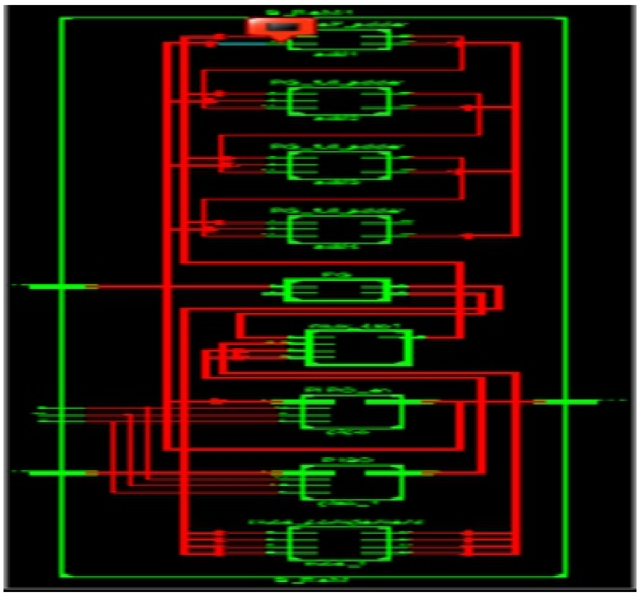
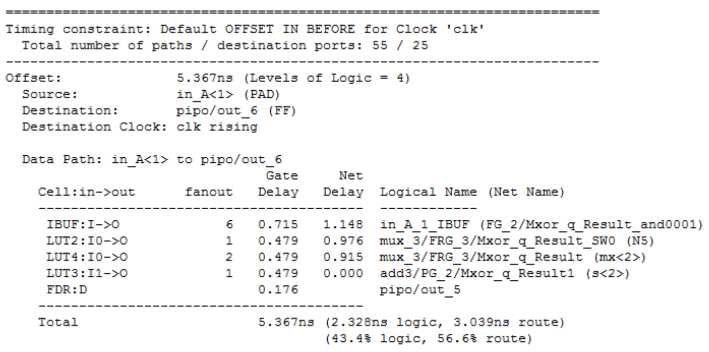


Fig22. Proposed Internal RTL Schematic.

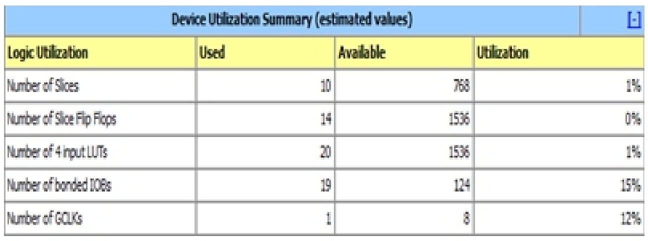


```
=========================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 55 / 25
-------------------------------------------------------------------------
Offset:              5.367ns (Levels of Logic = 4)
  Source:            in_A<1> (PAD)
  Destination:       pipo/out_6 (FF)
  Destination Clock: clk rising

  Data Path: in_A<1> to pipo/out_6
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             6   0.715   1.148  in_A_1_IBUF (FG_2/Mxor_q_Result_and0001)
     LUT2:I0->O            1   0.479   0.976  mux_3/FRG_3/Mxor_q_Result_SW0 (N5)
     LUT4:I0->O            2   0.479   0.915  mux_3/FRG_3/Mxor_q_Result (mx<2>)
     LUT3:I1->O            1   0.479   0.000  add3/PG_2/Mxor_q_Result1 (s<2>)
     FDR:D                     0.176          pipo/out_5
    ----------------------------------------
    Total                      5.367ns (2.328ns logic, 3.039ns route)
                                       (43.4% logic, 56.6% route)
```

Fig23. Acquired Delay.

Table3. Area utilization:



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	10	768	1%
Number of Slice Flip Flops	14	1536	0%
Number of 4 input LUTs	20	1536	1%
Number of bonded IOBs	19	124	15%
Number of GCLKs	1	8	12%

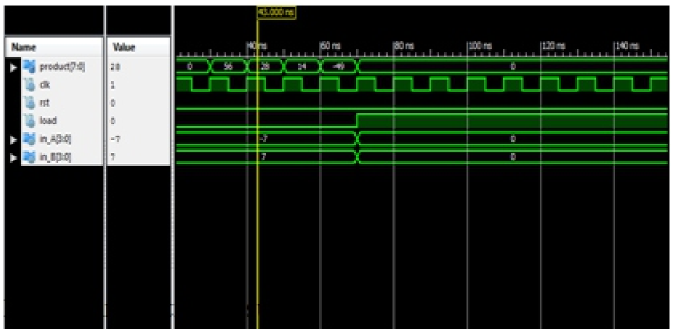


Fig24. Simulation Results.

VII. CONCLUSION:

In this paper, we proposed the K-Algorithm i.e. improved Booth"s recoding algorithm that significantly reduces the hardware required for signed multiplication. Multiplier architecture for efficient implementation of K-Algorithm is also proposed. We designed three reversible multipliers, namely B-ReM, BFT-ReM and KFT-ReM and analyzed them with respect to the reversible logic design metrics. It is found that our fault-tolerant designs (BFT-ReM and KFT-ReM) are much more optimal than other designs reported in literature. We further establish that using K-Algorithm to design a faul ttolerant reversible multiplier causes the design to be most optimal in quantum cost and reduces the number of gates, ancilla inputs and garbage outputs considerably. Specifically, on an average, there is a 33% reduction in quantum cost in case of KFT-ReM, thus making it the most optimal design. Hence, K-Algorithm proves to be very useful in optimizing the reversible as well as conventional multiplier designs with equal number of merits.

VIII. REFERENCES:

[1] R. Landauer, "Irreversibility and heat generation in the computing process," IBM Journal of Research and Development, vol. 5, no. 7, pp. 183–191, 1961.

[2] C. Bennet, "Logical reversibility of computation," IBM Journal of Research and Development, vol. 17, no. 1, pp. 525–532, 1973.

[3] M. Islam et al., "Synthesis of fault tolerant reversible logic circuits," CoRR, vol. abs/1008, pp. 33–40, 2010.

[4] J. Bruce et al., "Efficient adder circuits based on a conservative reversible logic gate," in Proc. IEEE ISVLSI 2002, Pittsburgh, PA, apr 2002, pp. 83–88.

[5] P. Kemtopf, "Synthesis of multipurpose reversible logic gates," in Euromicro Symposium on Digital System Design (DSD"02), Dortmund, Germany, sep 2002, pp. 259–267.

[6] T. Toffoli, "Reversible computing," MIT Lab for Computer Science, vol. 85, pp. 632–644, 1980.

[7] E. Fredkin and T. Toffoli, "Conservative logic," Int"l Journal of Theoretical Physics, vol. 21, pp. 219–253, 1982.

[8] R. Feynman, "Quantum mechanical computers," Optics News, vol. 11, no. 2, pp. 11–20, 1985.

[9] B. Parhami, "Fault tolerant reversible circuits," in Proc. 40th Asimolar Conference Signals, Systems and Computers, Pacific Grove, CA, 2006, pp. 1726–1729.

[10] S. Mitra and A. Chowdhury, "Minimum cost fault tolerant adder circuit in reversible logic synthesis," in 25th International Conference on VLSI Design, Hyderabad, India, jan 2012, pp. 334–339.

[11] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs. New York: Oxford University Press, 2000.

[12] S. Babazadeh and M. Haghparast, "Design of a nanometric fault tolerant reversible multiplier circuit," Journal of Basic and Applied Scientific Research, vol. 2, no. 2, pp. 1355–1361, 2012.

[13] M. Haghparast et al., "Optimized reversible multiplier circuits," Journal of Circuits, Systems and Computers, vol. 18, no. 2, pp. 311–323, 2009.

[14] M. H. M. et al., "Design of a novel reversible multiplier circuit using hng gate in nanotechnology," World Applied Science Journal, vol. 3, no. 6, pp. 974–978, 2008.

[15] M. Shams et al., "Novel reversible multiplier circuit in nanotechnology," World Applied Science Journal, vol. 3, no. 5, pp. 806–810, 2008.

[16] H. Thapliyal and M. Srinivas, "Novel reversible multiplier architecture using reversible tsg gate," in IEEE International Conference on Computer Systems and Applications, 2006, pp. 100–103.

[17] H. Thapliyal et al., "A reversible version of 4 × 4 bit array multiplier with minimum gates and garbage outputs," in The 2005 International Conference on Embedded Systems and Applications (ESA"05), Las Vegas, USA, 2005, pp. 106– 114