International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-2, December 2019

Design of Multiplier through Modified Booth Algorithm with Mig Gate

Ripunjai Gurava, P. Suresh Babu

Abstract*: This paper compares different Booth multipliers i.e., Radix-2, 4, 8 is designed using a new carry look-Ahead adder (CLA). In this Delay and Power have been compared and the main aim behind the project is developing Booth multiplier using Reversible Logic Gate (RLG).While comparing with the normal multiplication, Modified Booth Algorithm gives the less amount of delay as the number of partial products gets reduced. In this process CLA is used to reduce the overall multiplier Delay.The reversible logic is considered because it reduces the circuit complexity, loss of information and power consumption. In this paper a new CLA architecture is proposed in place of the existing CLA architecture which exhibits a high performance of computation, power consumption and area. In this architecture, Delay and power consumption of the design are reported. This new architecture is simulated and synthesized using Xilinx ISE environment.

Keywords* : Booth multiplier, Carry Look-Ahead adder (CLA), Reversible Logic Gates (RLG).

I. INTRODUCTION

Now a days the reversible logic is important due to its main advantages while comparing with normal CMOS technology, such as possibility in low-power consumption.
R.Landauer has proposed that the loss of information in new technology circuits will cause energy dissipation according to that it will generate heat $kT * \ln(2)$[1] i.e., which dissipate some energy for each bit of information loss. According to Frank reversible logic, operations which are used by computers can widely reuse the signal energy that hypothetically can approach arbitrary around 100%. A gate which produces an 'n' number of outputs for 'n' number of inputs is known to be reversible logic gate. For mapping and each input instance with a single output instance, the reversible gates are cascaded to form a reversible network. Many reversible new logic gates had been developed and existed in the literature, where as in the synthesis of each reversible logic circuits differs parallel. In this paper, a multiplier through booth's algorithm with MIG reversible gate is designed and capable of working with unsigned and signed numbers

Revised Manuscript Received on December 30, 2019.
***** Correspondence Author
Ripunjai Gurava*, M.Tech(VLSI System design) , VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India.
P. Suresh babu, Ph. D, Acharya Nagarjuna University, Guntur, India.

BASIC REVERSIBLE GATES

1.1 Feynman Gate: This FG gate is reversible gate which is known as Controlled NOT and mostly used for fan-out purposes. The inputs (a, b) and outputs (P, Q) P= a, $Q = a \oplus b$. Its total quantum cost is one.[7]

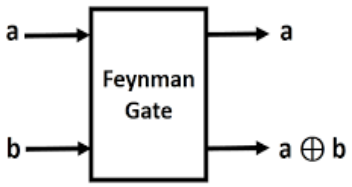

Figure 1: Feynman gate

1.2 Toffoli Gate: The input vectors $I(A,B,C)$ and its output vectors $O(P,Q,R)$ shows the $3 * 3$ Toffoli gate [3]. The outputs were defined as $P = A, Q = B, R = AB \oplus C$.Its Quantum cost is five.[7]

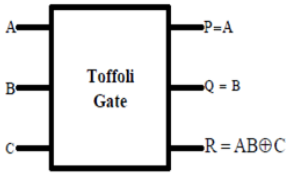

Fig 2: Toffoli gate

1.3 Fredkin Gate: The $3 * 3$ input vectors $I(A,B,C)$ and its output vectors $O(P,Q,R)$ shows the $3 * 3$ Fredkin gate [4].The outputs were defined as $P = A$, $Q = \bar{A}B \oplus AC$ and $R = \bar{A}C \oplus AB$. Its Quantum cost is five.[7]

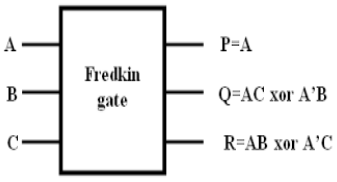

Fig 3: Fredkin Gate

1.4 Peres Gate: The $3 * 3$ input vectors$I(A,B,C)$ and its output vectors $O(P,Q,R)$ shows the $3 * 3$ Peres gate[5]. The outputs were defined as $P = A$, $Q = a \oplus b$ and $R = ab \oplus c$. Its Quantum cost is four.[7]

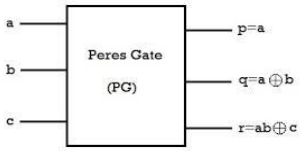

Fig 4: Peres Gate

1.5 Modified IG Gate (MIG): The $4*4$ input vectors $I(A,B,C,D)$ and the output vector $O(P,Q,R,S)$ show the $4*4$ MIG gate. The output were defined as $P=A$, $Q=A\oplus B$, $R=AB\oplus C$ and $S=A\bar{B}\oplus D$.[8]

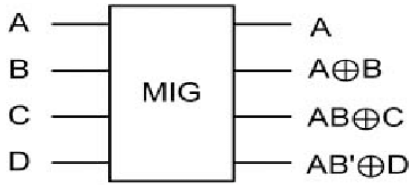

Fig 5: MIG Gate

II. REVIEW OF BOOTH'S ALGORITHM

a. MODIFIED BOOTH ALGORITHM

(i) RADIX-4 MULTIPLIER

The Modified Booth encoding (MBE) or Modified Booth's algorithm (MBA), has been proposed by O. L. Macsorley in the year 1961.To generate the partial products, recoding method is used for the implementation and generation of parallel multipliers, as it gives the parallel encoding scheme. The best way to understand the speed multipliers is by improving parallelism and then the calculation stages have been reduced. The distinctive form of the Booth calculation i.e., Radix-2 has two negatives which are:

(i) The shift, subtract and add operations converts the variables by then it becomes difficult for designing the parallel multipliers.

(ii) By this isolated 1's algorithm it becomes incompetent. These difficulties are resolved using Modified Radix-4 Booth algorithm.

Radix-4 encoding decreases the multiplier digits by a factor of two, as the quantity of digits in multiplier will reduce from$16\ to\ 8$. In this calculation, the multiplier is made into gatherings of three successive digits where the peripheral digit in each gathering is imparted to the furthest digit of the adjacent group. Every one of these gatherings of three double digits at that point compares with one of the numbers from the set $\{2,1,0,-1,-2\}$. Each recoder creates an $3-bit$ output. Booth's algorithm is a most dominant procedure for both unsigned and signed number calculation. For Radix-4 a n-bit binary number can be translated as $\frac{n}{2}$ –digit, and for Radix-8 an-bit binary number can be translated $\frac{n}{3}$ –digit and so on. The primary drawback of the Radix-2 calculation is that the procedure mandate for k shifts and a normal of $\frac{k}{2}-$ additions of a k bit multiplier. The shifting and addition of operations are exceptionally hard for structuring the parallel multipliers. The Radix-4 modified Booth calculation will overcome these confinements happened in Radix-2 calculation. If the operand size is larger than 16 bit then Radix-4 Booth algorithm can be generally utilized. The multiplier is encoded into 2's complement format so then the computing design stages i.e., it decreases the quantity of partial items and then it is added to $k/2$.

(ii) RADIX-8 MULTIPLIER

Radix -8 Booth multiplier will use $4-$ bit encoding technique for the reduction of partial products.

Radix-8 Booth recoding is as similar as Radix-4 algorithm, but in Radix-8, it is codified as a signed digits, the groups of four bits is taken rather than triplets as seen in Radix-4 algorithm.

Radix-8 booth algorithm will reduce the quantity of partial products from $k$to k$/3$, here k is defined as total number of multiplier bits. Radix-8 Modified Booth Multiplier can be preferred for its high performance than radix-4 Modified Booth Multiplier. Because radix-8 will have less number of partial items than the radix -4. So, the time span was diminished in radix-8 contrasted with the radix-4 multiplier.

III.EXISTING DESIGN

The existing design of booth multiplier been developed with the reversible logic with B and C cell as this cells consists of different logic gates in which Radix-2 with $4-bit$ and $8-bit$ been implemented. In C cell TS-3 and Fredkin logic gates are used as this cell will generate the necessary control signal named as D and H and the total C cells is called as Controlled signal generator as the name implies it controls the block. The B cell is designed with the MTSG, TS-3 Gate and Peres Gate. This 3 gates been implemented in a single block though this all acts as adder/ subtractor so it is named as adder/ subtractor block. This B and C cells are placed in an order to form of trapezium. When the Multiplicand (MD) is given as a input to the FG gate it is been complemented and the partial product generated been given to the B cell by which the addition been carried out and the output of one B cell is given to the another B cell as a input after the total addition process been done the actual output is generated [6].

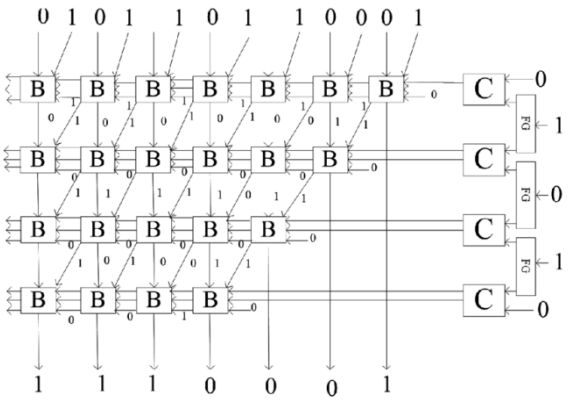

Fig 6: Array multiplier using Reversible logic

IV.PROPOSED REVERSIBLE BOOTH'S MULTIPLIER

In this module, booth's multiplier is implemented with a new reversible logic carry look-ahead adder (CLA). Again Radix-2 was designed with new CLA as this is the better approach than the existing system though the speed and power has been reduced.Radix-4 and Radix-8 are also implemented by the same reversible CLA and the multipliers (i.e., Radix-2, 4, 8) are compared with Power and Delay.

i. REVERSIBLE CARRY LOOK-AHEAD ADDER

The impediment factor is time, as the ripple carry adder takes to engender the carry. So we have considered carry look-ahead adder (CLA) as it calculates the carry signal flag ahead of time, in view of the input signals. These outcomes will reduce in carry propagation time.

To comprehend that how CLA works, first we need to implement Boolean expression for the propagator (P) and generator(G) for the full adder which is givenas:

$P_i = A_i \oplus B_i$ Carry propagator

$G_i = a_i . b_i$ Carry generator

And both this propagated and generated signals will depend just on the input bits.

The accompanying new articulations for the output i.e., the carry-out and sum are given by:

$S_i = P_i \oplus C_{i-1}$

$C_{i+1} = G_i + P_i . C_i$

So the equations for $4-$bit adder is given as:

$C_1 = G_0 + P_0 . C_0$

$C_2 = G_1 + P_1 . C_1 = G_1 + P_1 (G_0 + P_0 . C_0) = G_1 + P_1 . G_0 + P_1 . P_0 . C_0$

$C_3 = G_2 + P_2 . C_2 = G_2 + P_2 . G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

$C_4 = G_3 + P_3 . C_3 = G_3 + P_3 . G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$

By the above expressions it show that C_2, C_3 and C_4 will not only depends on its previous carry-in. So C_4 don't require to wait for the propagation of C_3. As earlier as C_0 is calculated, C_4 can be reached to steady state. This is continued for C_2 and C_3.

The general expression given as

$C_{i+1} = G_i + P_i . G_{i-1} + P_i P_{i-1} G_{i-2} + \cdots \ldots \ldots + P_i P_{i-1 \ldots\ldots} P_1 P_0 C_0$

The new carry look-ahead adder implemented using MIG gate. A reversible logic gate consists of 4*4 input and output. In this MIG gate we require only 2 outputs for carry propagate and carry generator and the remaining 2 outputs we consider as garbage value.

Consider MIG gate with 4 inputs a, b, c, d and 4 outputs p, q, r, s as p is considered as carry propagate and q is considered as carry generator and the remaining two outputs i.e., r, s is considered as garbage outputs, because these are not useful at present.

ii. PROPOSED DESIGN OF 8 BIT BOOTH MULTIPLIER

An 8x8 reversible Booth's multiplier is instigated by different Radix i.e., Radix-2, 4, 8. As the process of implementing all the radix is same but the only change we have to observe is encoding and the multiplier digits are made into groups are different. The partial product generation in Booth multiplication is based on recoding. Multiplication using booth algorithm depends on partial products as this can be generated for group of consecutive $1's$ and $0's$ called as booth recoding. The width of the partial products depends on radix scheme. So the use of few partial products provides low power and area. Booth algorithm treats both negative and positive consistently. As a k-bit binary number can be defined as $\frac{n}{2}-$digit for Radix-4 number, a $\frac{n}{3}-$digit for Radix-8 number and so on. The main drawback of Radix-2 algorithm is, that this process requires $k-$shifts and $k/2$ number of additions for n bit multiplier. This process for the designing of parallel multipliers is inconvenient. To overcome this limitations Radix-4 algorithm is developed.

In this a new CLA is developed using Reversible gate.MIG gate is used as reversible gate as when compared to other reversible gates i.e., Fredkin, Peres. MIG gate can generate both carry propagator as well as carry generator by which the number of gates have been reduced.

a. RADIX-4 BOOTH'S ALGORITHM

We considered the multiplier bits into three blocks, as Radix-4 encodes multiplier bits into $[-2,2]$. Grouping starts from LSB and the first block will consist only 2 bits and the last LSB bit will be assumed to be 0.The Radix-4 consists functional operator for booth encoding as shown in below table.

Table 1: Radix-4 Encoding table

B	Zn	Partial Product
000	0	0
001	1	1x Multiplicand
010	1	1x Multiplicand
011	2	2x Multiplicand
100	-2	-2x Multiplicand
101	-1	-1x Multiplicand
110	-1	-1x Multiplicand
111	0	0

b. RADIX-8 BOOTH'S ALGORITHM

Radix-8 encodes multiplier bits into [-4, 4]. Radix-8 algorithm is same as Radix-4 but when compared to grouping we consider group of four bits instead of triplets. Radix-8 condenses the sum of partial products to $n/3$. This gives the decrease in time for calculation.In all type of radix the MSB bit is considered as a sign extension value. As 0 defines for the unsigned operation, 1 defines for the signed operation. The Radix-4 consists functional operator for booth encoding as shown in table 2.

Table 2: Radix-8 Encoding table

b_{3i+2}	b_{3i+1}	b_{3i}	b_{3i-1}	PP
0	0	0	0	0
0	0	0	1	$+A$
0	0	1	0	$+A$
0	0	1	1	$+2A$
0	1	0	0	$+2A$
0	1	0	1	$+3A$
0	1	1	0	$+3A$
0	1	1	1	$+4A$
1	0	0	0	$-4A$
1	0	0	1	$-3A$
1	0	1	0	$-3A$
1	0	1	1	$-2A$
1	1	0	0	$-2A$
1	1	0	1	$-A$
1	1	1	0	$-A$
1	1	1	1	0

iii. PARTIAL PRODUCT GENERATOR

A value or a variable product formed by increasing the multiplicand by every single digit of the multiplier when it has more than one digit.

Partial products are taken as intermediate steps for the calculation of larger product values. This product generator was intended the product by multiplying the multiplicand by $[0,1,-1,2,-2,3,-3,4,-4]$ etc.

In this generator, multiplying with 0 defines the multiplicand is amplified by "0" . Multiplying with 1 defines the multiplicand value remains equivalent. Multiplying with -1 defines the product is the two's complement of the number. Multiplying with -2 is to move left by one bit and then the two's complement of the multiplicand esteem and multiplication with 2 defines that the multiplicand should shift left by one place. Multiplying with -4 is to move left by two bits and the two's complement of the multiplicand value and multiplying with 2 implies move left of the multiplicand by two places. And moreover we have an odd various multiple of the multiplicand, i.e., $3Y$, as this is not immediately available. To generate this we need to play out this previous addition- $2Y + Y = 3Y$. But as we know designing a multiplier and thereby the multiplicand fits to a previously known fixed of numbers which have been stored in a memory chip.

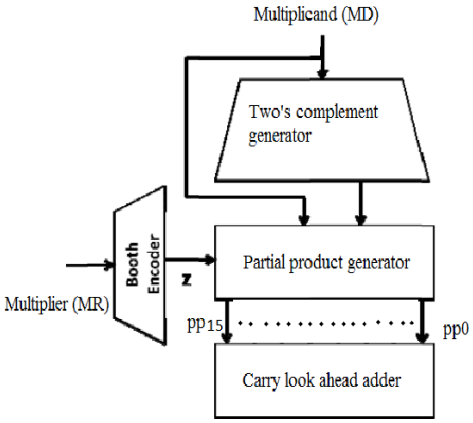

Fig 7: Booth Multiplier

V. SIMULATION AND RESULTS OF MULTIPLIER

The simulated output of Booth multiplier with its summary report and RTL schematic been shown in Fig.8, Fig.9 and Fig.10 .Comparisons of different Radix multipliers is given in table 3.

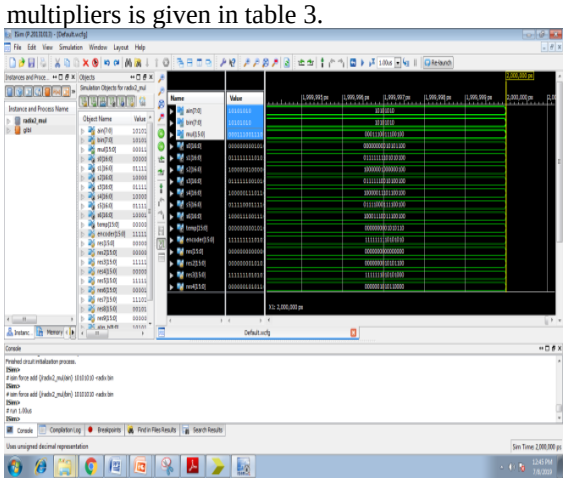

Fig 8: Simulation result of proposed 8x8 reversible multiplier

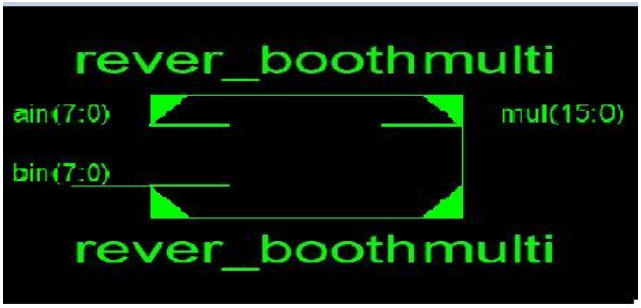

Fig 9a: Top module of proposed 8x8 reversible multiplier

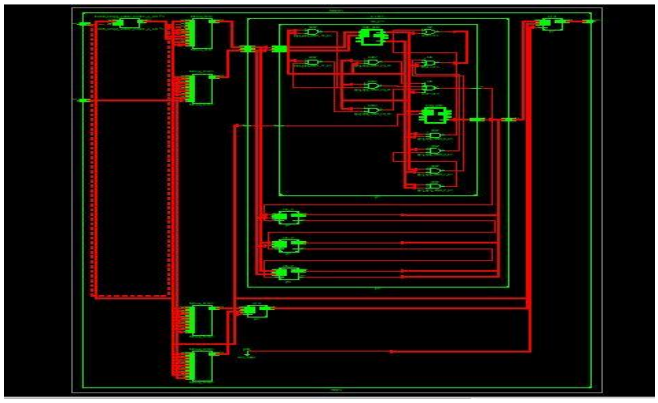

Fig 9b: RTL schematic of proposed 8x8 reversible multiplier

Table 3: comparison of all Radix

parameters	Radix-2	Radix-4	Radix-8
Delay	29.15nS	21.666nS	23.12nS
Power	33.07mW	33.59mW	42.38mW

Device Utilization Summary		
Slice Logic Utilization	**Used**	**Available**
Number of Slice Registers	0	126,800
Number of Slice LUTs	114	63,400
Number used as logic	114	63,400
Number using O6 output only	67	
Number using O5 output only	0	
Number using O5 and O6	47	
Number used as ROM	0	
Number used as Memory	0	19,000
Number used exclusively as route-thrus	0	
Number of occupied Slices	60	15,850
Number of LUT Flip Flop pairs used	114	
Number with an unused Flip Flop	114	114
Number with an unused LUT	0	114
Number of fully used LUT-FF pairs	0	114
Number of slice register sites lost to control set restrictions	0	126,800

Fig 10: Summary report

VI. CONCLUSION

This paper has presented a new multiplier structural design. The entire multiplier has been improved by eradicating the aggregation process that has the greatest delay by substituting Conventional CLA with reversible logic CLA. The proposed structure was synthesized through Xilinx ISE tool and Comparison has been done with different Radix

REFERENCES

1. R. Landauer, "Irreversibility and heat generation in the computing process," IBM J. Res. Dev., vol. 5, no. 3, pp. 183–191, Jul. 1961.
2. H. H. Babu, R. Islam, A. R. Chowdhury, and S. M. A.Chowdhury, "Reversible logic synthesis for minimization of full-adder circuit," in Euro micro Symposium on Digital Systems Design, 2003, pp. 50–54.
3. T. Toffoli, "Reversible computing," in Colloquium on Automata, Languages and Programming, 1980, pp. 632–644.
4. E. Fredkin and T. Toffoli, "Collision-based computing," vol. 21, pp.219–253, 1983.
5. A. Peres, "Reversible logic and quantum computers," Phys. Rev. A, vol. 32, pp. 3266– 3276, Dec 1985.
6. Raja Chowdhury b "On the Analysis of Reversible Booth's Multiplier" in 28th International Conference on VLSI Design and 2015 14th International Conference on Embedded Systems,2015.
7. M. Selim, "Quantum Cost Optimization for Reversible Sequential Circuit," Int. J. Adv. Comput. Sci. Appl., vol. 4, no. 12, pp. 15–21, 2013.
8. S.T.Suria," Design and Implementation of CLA Using Reversible Logic Gates" in Int. J. Ino. Research Sci. Eng. Tech., Vol. 5, Issue 5, May 2016.

AUTHORS PROFILE

Mr.G.Ripunjai has received his B.Tech degree from St.Martin's Engineering college,Hyderabad in 2017.Currently he is pursuing M.Tech(VLSI System design) in VNR Vignana Jyothi Institute of Engineering and Technology,Hyderabad. His research of interest in VLSI Design.

Mr.P.Suresh Babu has received his B.Tech degree from VR Siddhartha engineering college,Vijayawada. M.Tech(VLSI System design) degree from National institute of technology,Tiruchapally in 2001 and 2006 respectively.At present he is pursuing Ph. D in Acharya Nagarjuna University,Guntur.He has 17 years of teaching experience.His research interest are in VLSI and RF domain.