ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

PERFORMANCE COMPARISON OF HIGHER RADIX BOOTH MULTIPLIER USING 45nm TECHNOLOGY

JasbirKaur[1], Sumit Kumar[2]

Asst. Professor, Department of E & CE, PEC University of Technology, Chandigarh, India[1]

P.G. Student, Department of E & CE, PEC University of Technology, Chandigarh, India[2]

ABSTRACT: This paper includes implementation and comparison of different parameter of higher radix booth multiplier. Comparison is made taking both multiplier and multiplicand of 60 bit each and output product of 120 bit using radix2, radix4, radix8, radix16 and radix32.All these multiplier were designed in verilog programming language and synthesized on cadence RTL schematic. Comparison is made between delay, area and power utilization of a higher radix multiplier. A Conventional Booth Multiplier consists of the Booth Encoder, the partial-product tree and carry propagate adder [2, 3]. Different schemes are addressed to improve the area and circuit speed effectively. From the result it is observed that in most of the parameter performance of radix 4 booth multiplier is more efficient than other multiplier.

KEYWORDS: Radix-2, Radix-4, Radix-8, Radix-16 and Radix-32 Booth Encoding multiplier

I. INTRODUCTION

Multipliers are key components of many high performance systems such as FIR filters, Microprocessor, digital signal processors, etc. A systems performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system. Furthermore, it is generally the most area consuming. Hence, optimizing the speed, area and power of the multiplier is a major design issue however area and speed are usually conflicting constraints so that improving speed results mostly in larger areas [5,6].Generally booth multiplier is used to increase the speed of circuit by encoding the number that are multiplied. Booth Multiplier reduces number of iteration step to perform multiplication as compare to Conventional multiplier. Radix 8 booth encoding multiplier reduces the no. of partial products by one third, N/3.Booth algorithm 'scans' the multiplier operand and skips chains of the algorithm can reduce the number of additions required to produce the result compared to Conventional Multiplication algorithm, where each bit of the multiplier is multiplied with the multiplicand and the partial products are aligned and added together [4]. More interestingly the number of additions is data dependent.

II. COMMON FEATURES OF MULTIPLIERS

(i) Counter flow Organization: A novel multiplier organization is introduced, in which the data bits flow in one direction and the Booth commands are piggy backed on the acknowledgments flowing in the opposite direction.
(ii) Merged Arithmetic/Shifter Unit: An architectural optimization is introduced that merges the arithmetic operations and the shift operation into the same function unit, thereby obtaining significant improvement in area, energy and speed.

(iii) Overlapped Execution: The entire design is pipelined at the bit-level, which allows overlapped execution of Proceedings of multiple iterations of the Booth algorithm, including across successive multiplications. As a result, both the cycle time per Booth iteration as well as the overall cycle time per multiplication are significantly improved

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

(iv) Modular Design: The design is quite modular, which allows the implementation to be scaled to arbitrary operand widths without the need for gate resizing, and without incurring any overhead on iteration time.

(v) Precision-Energy Trade-Off: Finally, the architecture can be easily modified to allow dynamic specification of operand widths, i.e., successive operations of a given multiplier implementation could operate upon different word length

It was then taken a step further in this analysis by designing and synthesizing Radix-8 Booth Encoding multiplier, Radix-16 Booth Encoding multiplier. The number of partial products reduces as we proceed to higher radix booth multiplier. Radix-8 Booth Encoding multiplier will encounter a reduction of N/3 in the partial products while Radix-16 Booth Encoding multiplier reduces its number of partial product by N/4.The algorithm to produce the partial products gets a lot more complicated as we proceed to higher Radix-based Booth Encoding multiplier.

III. RADIX-2 BOOTH MULTIPLIER

Radix-2 Booth Multiplier type, implemented using asynchronous circuits [6,7]. An iterative implementation was chosen, as opposed to a combinational array type, for higher area efficiency. A Booth implementation was chosen so as to uniformly handle signed as well as unsigned operands. Booth algorithm gives a procedure for multiplying binary integers in signed –2's complement representation. The booth algorithm with the following example: Example: $2_{10} \times (-4)_{10}$ or $0010_2 \times 1100_2$

Step1: Making the Booth table

I. From the two numbers, pick the number with the smallest difference between a series of consecutive numbers, and make it a multiplier. i.e., 0010 — From 0 to 0 no change, 0 to 1 one change, 1 to 0 another change, and so there are two changes on this one, 1100 — From 1 to 1 no change, 1 to 0 one change, 0 to 0 no change, so there is only one change on this one. Therefore, multiplication of 2 × (–4) can be taken in which $(2)_{10}(0010_2)$ is the multiplicand and $(-4)_{10}$ (1100_2) is the multiplier.
II. Let X = 1100 (multiplier) and Y = 0010 (multiplicand) Take the 2's complement of Y and call it –Y so –Y = 1110
III. Load the X value in the table.
IV. Load 0 for X-1 value it should be the previous first least significant bit
V. Load 0 in U and V rows which will have the product of X and Y at the end of operation.
VI. Make four rows for each cycle; this is because we are multiplying four bits numbers.
Step 2: Booth Algorithm requires examination of the multiplier bits, and shifting of the partial product. Prior to the shifting, the multiplicand may be added to partial product, subtracted from the partial product, or left unchanged according to the following rules: Look at the first least significant bits of the multiplier "X", and the previous least significant bits of the multiplier "X - 1".

I 00 and 11 Shift only. 01 Add Y to U, and shift 10 Subtract Y from U, and shift or add (-Y) to U and shift

II Take U & V together and shift arithmetic right shift which preserves the sign bit of 2's complement number. Thus a positive number remains positive, and a negative number remains negative.

III Shift X circular rights shift because this will prevent us from using two registers for the X value. B.

IV.Radix 4 BOOTH MULTIPLIER

The Modified Booth Multiplier was proposed by O. L. Macsorley in 1961. The recoding method is widely use to generate the partial products for implementation of large parallel multipliers, which adopts the parallel encoding scheme [2]. One of the solutions of realizing high speed multipliers is to enhance parallelism which helps to decrease the number of subsequent stages. The original version of Booth algorithm (Radix-2) had two drawbacks:

• The number of add subtract operations and the number of shift operations becomes variable and becomes inconvenient in designing parallel multipliers.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

• The algorithm becomes inefficient when there are isolated 1's [3].

These problems can be overcome by Modified Booth algorithm (MBA). In MBA process three bits at a time are recoded. Recoding the multiplier in higher radix is a powerful way to speed up standard Booth multiplication algorithm. In each cycle a greater number of bits can be inspected and eliminated therefore total number of cycles required to obtain products get reduced [4]. Number of bits inspected in radix r is given by n = 1 + log2r.

Algorithm for radix 4 Booth multiplier is given below: in each cycle of radix-4 algorithm 3 bits get inspected. Procedure for implementing radix-4 algorithm is as follows

• Append a 0 to the right of LSB

• Extend the sign bit 1 position if necessary to ensure that n is even

• According to the value of each vector, find

each partial product Radix-4 encoding reduces the total number of multiplier digits by a factor of two, which means in this case the number of multiplier digits will reduce from 16 to 8 [5]. Booth's recoding method does not propagate the carry into subsequent stages. This algorithm groups the original multiplier into groups of three consecutive digits where the outermost digit in each group is shared with the outermost digit of the adjacent group. Each of these groups of three binary digits then corresponds to one of the numbers from the set {2, 1, 0,-1,-2}. Each recoding produces a 3-bit output where the first bit represents the number 1 and the second bit represents this number 2. The third and final bit indicates whether the number in the first or second bit is negative. Let the two numbers to be multiplied are X = 34 and Y = -42.

Multiplicand X = 34 = 00100010

Multiplicand Y= -42=11010110(2's Complement)

X*Y = -1428

TABLE 1. MODIFIED BOOTH EXAMPLE

									0	0	1	0	0	0	1	0	34
									1	1	0	1	0	1	1	0	-42
						1	1	1	1	0	1	1	1	1	0	0	PP1
				1	1	0	0	1	0	0	0	1	0	0			PP2
		1	1	0	0	0	1	0	0	0	1	0					PP3
1	0	1	1	1	0	1	1	1	1	0							PP4
1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	0	0	-1428

V. RADIX-8 BOOTH ENCODER MULTIPLIER

Radix 8 booth encoding multiplier uses 4-bit encoding scheme. As a result of that no. of Partial product reduce by one third factor. but the circuit complexity also increases as compare to previous version of booth multiplier.Radix-8 booth multiplier also lacks in most parameter like delay, speed from radix 4 booth multiplier due to the complexity of the circuit.Radix-8 booth multiplier which scan strings of four bits An algorithm similar to previous radix algorithm is made for radix8 booth multiplier. After encoding the multiplier the resulting partial product will be 0,y,+2y,+3y,+4y,-4y,-3y,-2y,-y where y represent the multiplicand. We have to represent all the no. in 2's complement form

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

VI. RADIX-16 BOOTH ENCODER MULTIPLIER

To reduce the number of partial products added while multiplying the multiplicand higher radix Booth Encoding algorithm is one of the most well-known techniques used[13].Radix-16 Booth algorithm which scan strings of five bits with the algorithm given below: (1) Extend the sign bit 1 position if necessary to ensure that n is even. (2) Append a 0 to the right of the LSB of the multiplier. (3) According to the value of each vector, each Partial Product will be0,y,+2y,+3y,+4y,+5y,+6y,+7y,+8y,-8Y,-7y,-6y,-5y,-4y,-3y,-2Y,-Y. The multiplication of y is done by shifting y by one bit to the left. Thus, in any case, in designing n-bit parallel multipliers, only n/4 partial products are generated [12].

VII. RADIX-32 BOOTH MULTIPLIER

Radix32 booth multiplier reduces the no. of partial products by one fifth. An algorithm similar to previous radix algorithm is made for radix 32 booth multiplier .The booth table generated as a result of the algorithm is given below

Table2.RADIX-32 BOOTH ENCODING MULTIPLIER

MULTIPLIER	RECODING
000000, 111111	0
000001, 000010	1 x Multiplicand
000011, 000100	2 x Multiplicand
000101, 000110	3 x Multiplicand
000111, 001000	4 x Multiplicand
001001, 001010	5 x Multiplicand
001011, 001100	6 x Multiplicand
001101, 001110	7 x Multiplicand
001111, 010000	8 x Multiplicand
010001, 010010	9 x Multiplicand
010011, 010100	10 x Multiplicand
010101, 010110	11 x Multiplicand
010111, 011000	12 x Multiplicand
011001, 011010	13 x Multiplicand
011011, 011100	14 x Multiplicand
011101, 011110	15 x Multiplicand
011111	16 x Multiplicand
100000	-16 x Multiplicand
100001, 100010	-15 x Multiplicand
100011, 100100	-14 x Multiplicand
100101, 100110	-13 x Multiplicand
100111, 010111	-12 x Multiplicand
101001, 101010	-11 x Multiplicand
101011, 101100	-10 x Multiplicand
101101, 101110	-9 x Multiplicand
101111, 110000	-8 x Multiplicand
110001, 110010	-7 x Multiplicand
110011, 110100	-6 x Multiplicand
110101, 110110	-5 x Multiplicand
110111, 111000	-4 x Multiplicand
111001, 111010	-3 x Multiplicand
111011, 111100	-2 x Multiplicand
111101, 111110	-1 x Multiplicand

To booth recode the multiplier, we consider the bits in block of six bit such that each block overlap the previous block by one bit.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

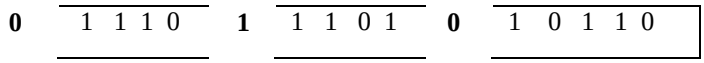

Fig 1. Grouping of bits in Radix-32 multiplier

VIII. RESULTS AND DISCUSSION

Simulation tool used

The radix 2, radix4, radix8, radix 16, radix 32 results are firstly designed using Xilinx fpga tool in verilog HDL programming language. But Xilinx could not support high bit multiplier such as 60×60 bit booth multiplier i.e 60 bit multiplier and multiplicand and 120 bit output results so these results are obtained using the Cadence RTL schematic tool. Different parameter i.e area, power and delay of different booth multiplier are then analyzed.

AREA ANALYSIS

From the result, we observed that Radix-4 Booth encoding multiplier outperformed the other radix based encoding multiplier and has very less area compared to others. As the no. of cells reduces area reduced to about 85.64% from radix-2 to radix-4 booth encoding multiplier. Area of other higher radix increase as we go to higher radix. Radix 4 and Radix-8 have nearly equal area

TABLE 3 AREA COMPARISON

S.N0	BOOTH MULTIPLIER	NO. OF CELLS	TOTAL AREA
1	RADIX 2	9426	27569
2	RADIX 4	1690	3959
3	RADIX8	2280	4359
4	RADIX16	14672	32973
5	RADIX32	25703	57091

POWER ANALYSIS

From the result we observed that Radix-8 booth encoding multiplier (60×60)is having very low power dissipation as compare to other radix multiplier. power consumption of radix -8 booth multiplier is 26.68% less as compared to radix-4 booth multiplier .These results are true only for higher bit multiplier but if we consider 12×12 booth multiplier then radix-4 booth multiplier is having less power consumption then radix-8 booth multiplier.

TABLE 4 POWER UTILIZED BY BOOTH MULTIPLIER

S.NO	BOOTH MULTIPLIER	LEAKAGE POWER(µwatt)	DYNAMIC POWER(µwatt)	TOTAL POWER(µwatt)
1	RADIX 2	0.499598	1783.491548	1783.991146
2	RADIX 4	0.076715	175.154257	175.230972
3	RADIX8	0.077178	138.333716	138.410893
4	RADIX16	0.564135	1106.480729	1107.044864
5	RADIX32	1.041318	1464.460923	1465.502241

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

TIMING DELAY ANALYSIS

The best case given by delay analysis results shows that Radix-4 Booth encoding multiplier outperformed the other Radix – based Booth multiplier and has very less delay. A delay decreased of nearly 10 ns occurred during the transition between Radix-2 and Radix-4 booth encoding multiplier. Delay of radix-8 booth multiplier is also nearly equal to radix-4 booth multiplier .the Delay of Radix-8 booth multiplier slightly increases by 0.817 ns on transition from radix 4 booth multiplier. We experienced a delay of 11.7 ns from Radix-8 to Radix-16 and a raised of 1.2 ns from Radix-16 to Radix-32 Booth encoding multiplier

TABLE 5 DELAY OF BOOTH MULTIPLIER

S.NO	BOOTH MULTIPLIER	TIME DELAY(ns)
1	RADIX 2	23.382
2	RADIX 4	13.993
3	RADIX8	14.810
4	RADIX16	26.581
5	RADIX32	27.729

IX. CONCLUSION

In this paper, five types of Radix based Booth Encoding multiplier i.e Radix-2, Radix-4, Radix-8 Radix-16 and Radix-32 Booth encoding multiplier were designed. They are designed firstly in Xilinx fpga tool using verilog HDL language. Then they are synthesized on cadence RTL schematic tool having 45 nm technology library.

From the result obtained, it is found that in 60×60 Booth encoded multiplier radix-4 booth encoding multiplier has higher speed then among other radix based multiplier. The other higher radix based multiplier has less delay when compared to radix-2 booth multiplier. Delay of radix-8 booth multiplier is nearly equal to radix-4 multiplier. From the result of area analysis it is found that radix-4 also dominates here with having very less number of cells and area. Radix-4 booth multiplier also has less complexity as compare to other higher radix booth multiplier. Similarly from the result of power dissipation, we found that radix-8 booth multiplier has very less power dissipation then other booth multiplier. But for low bit booth multiplier such as 12×12 radix4 booth multiplier has less power dissipation then rival radix-8 based booth multiplier.

REFRENCES

[1] N. Jiang and D. Harris, "Parallelized Radix-2 Scalable Montgomery Multiplier", submitted to IFIP Intl. Conf. on VLSI, 2007.
[2] D. Kudeeth., "Implementation of low-power multipliers", Journal of low-power electronics, vol. 2, 5-11, 2006.
[3] Y.N. Ching, "Low-power high-speed multipliers", IEEE Transactions on Computers, vol. 54, no. 3, pp. 355-361, 2005.
[4] D. Harris, R. Krishnamurthy, M. Anders, S. Mathew, and S. Hsu, "An improved unified scalable radix-2 Montgomery multiplier" , Proc. 17th IEEE Symp. Computer Arithmetic, pp. 172-178, 2005.
[5] K. Kelley and D. Harris, "Parallelized very high radix scalable Montgomery multipliers", Proc. Asilomar Conf. Signals, Systems, and Computers, pp. 1196-1200, Nov. 2005.
[6] J. Hensley,A. Lastra and M. Singh, "An area and energy efficient asynchronous booth multiplier for mobile devices", In Proc. Int. Conf. Computer Design (ICCD), 2004.
[7] A. Efthymiou,W. Suntiamorntut, J. Garside, and L. Brackenbury. "An asynchronous, iterative implementation of the original Booth multiplication algorithm. In Proc. Int. Symp", On Advanced Research in Asynchronous Circuits and Systems, pages 207–215. IEEE Computer Society Press, Apr. 2004.
[8] P. D. Chidgupkar and M. T. Karad, "The Implementation of Vedic Algorithms in Digital Signal Processing", Global J. of Engg. Edu, vol. 8, no. 2, pp. 153–158, 2004.
[9] M. Sheplie, "High performance array multiplier", IEEE transactions on very large scale integration systems, vol. 12, no. 3, pp. 320-325, 2004.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 1, Januray 2016

[10] Hsin Lei Lin, Robert C. Chang, Ming-Tsai, "Design of a Novel Radix-4 Booth Multiplier", IEEE Asia-Pacific Conference on Circuits and Systems, December 6-9, 2004.
[11] A. S. Prabhu, and V. Elakya , "Design of Modified Low Power Booth Multiplier", IEEE Conference Publication, pp 1-6, 2012.