High Speed-Low Power Radix-8 Booth Decoded Multiplier

Praveen Kumar Patil
M.Tech Scholar
Department of Electronics and Communication
Maulana Azad National Institute of Technology
Bhopal, India

Laxmi Kumre
Assistant Professor
Department of Electronics and Communication
Maulana Azad National Institute of Technology
Bhopal, India

ABSTRACT
This paper proposed a new method for adding sum and carry using carry look-ahead adder at the final stage of the radix-8 booth decoding multiplier. In a conventional radix-8 booth decoded multiplier, full adders and half adders are used to add sum and carry. After partial product reduction using booth decoding, the partial product rows are required to add for final result. In this method carry look-ahead adders (CLAs) are used to add reduced partial product generated after decoding the multiplier bits. The carry look-ahead adder generates the carry and sum simultaneously. 5 bit and 8 bit carry look-ahead adders are used to add reduced partial product terms in proposed circuit. The proposed method is used to implement 8bit multiplication using radix-8 booth decoded multiplier. The circuit is designed and simulated using cadence virtuoso EDA tool at 180nm CMOS technology. Simulation results shows power reduction by 11.48 % and propagation delay reduction by 33.06 % as compared to conventional method.

Keywords
Booth Multiplier, Radix-8, Booth Decoder, Partial Product, Carry Look-ahead Adder

1. INTRODUCTION
Now days Analog systems are replacing by digital systems because of its high speed performance, takes less area and less power dissipation [1]. Multiplication is one of the most important and basic arithmetic operation that constitute programs. In fact 8.72% of all instructions in typical scientific programs are based on multiplication operation [2]. Many multipliers have been proposed in the past with consideration of small area, low power and high performance. Multiplication is achieved by the addition of a certain number of partial products rows. Each partial product row is generated by multiply the multiplier bit one by one to multiplicand. In a simple multiplier, the generated partial products rows are equal to the number of bits in multiplier. For example, in 8×8 bit multiplication, it will produce 8 partial product rows. It will take more adders and more time.

To improve the performance of the multiplier, Booth multiplier is mostly used multiplier. The number of partial products rows that must be added to give the multiplication's result can be reduced by using Booth decoding. In Booth multiplier, the numbers of reduced partial products rows are depend on the grouping done at multiplier bits [3][4]. These groups of multiplier perform the selected operation on multiplicand. In booth multiplier grouping is done by 2 bits, 3 bits, 4 bits and so on. Higher order booth decoding reduces the number of partial product rows by a greater by decoding larger groups of multiplier bits.

This multiplication process is completed in 3 steps. First step: multiplier bits are divided in groups then these groups are fed to decoder at where it will indicate that which operation is to perform on multiplicand. Second step: here indicated operation performs on the multiplicand and it will generate the partial products. Third step: Now generated partial products are adding with adders [4].

After generation of these partial products, for adding them, many techniques are used with difference performances. In this proposed paper one of new techniques used that is carry look-ahead adder. CLA reduces the propagation path delay and increase the performance of the system, which is mostly required in digital systems.

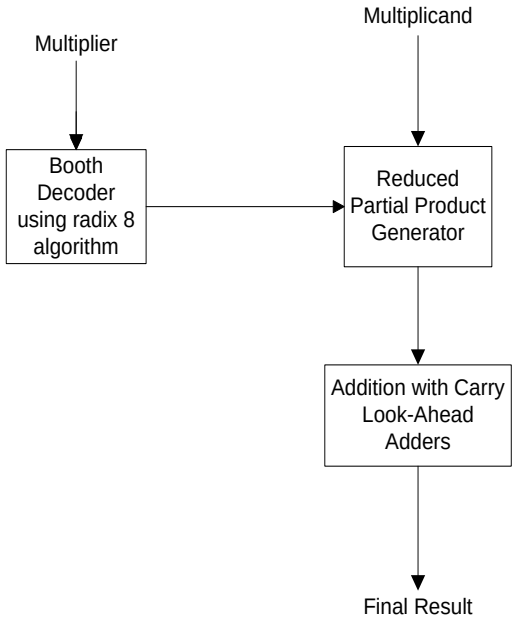

Figure 1: Proposed Booth Multiplication Process

2. RADIX-8 BOOTH MULTIPLIER

This Booth multiplier is known as radix-8 because it perform the 8 different types of operations on the multiplicand that are +M, +2M, +3M, +4M, –4M, –3M, –2M and –M where M denotes the Multiplicand. All the multiples with except 3M are easily obtainable, by simply shifting and complementing. The generation of the 3M (3× multiplicand), which is referred to as a hard multiple, cannot be obtained by simple shifting and complementation. It can be produce either M+2M or 4M–M. Here in this project, it is produced by M+2M. For example of 8×8 bit multiplication, a simple multiplier generates the 8 partial product rows, but by radix-8 booth multiplier it is reduced to 3. It means that radix-8 booth multiplier reduces the partial product rows by N/3 where 'N' in number of bits in multiplier [5] - [9].

Table 1: Operation to be performed on multiplicand

Group of Multiplier bits	Operation to beperform on Multiplicand
0000	0
0001	1 x Multiplicand
0010	1 x Multiplicand
0011	2 x Multiplicand
0100	2 x Multiplicand
0101	3 x Multiplicand
0110	3 x Multiplicand
0111	4 x Multiplicand
1000	-4 x Multiplicand
1001	-3 x Multiplicand
1010	-3 x Multiplicand
1011	-2 x Multiplicand
1100	-2 x Multiplicand
1101	-1 x Multiplicand
1110	-1 x Multiplicand
1111	0

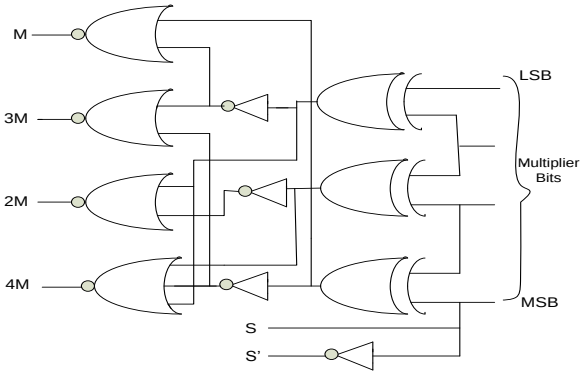

Figure 2: Radix-8 Booth Decoder [10]

Using figure 2 and figure 3, the partial products are generated. In figure 2, multiplier bits are used and then desired operations perform on multiplicand as shown in figure 3 to generate partial product.

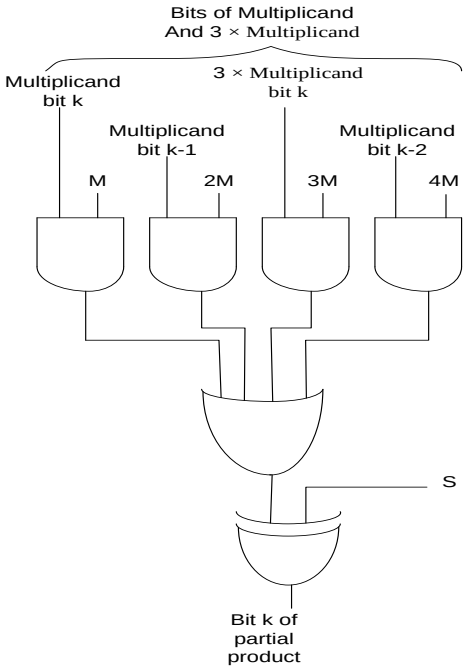

Figure 3: PP Generator of Radix-8 Booth multiplier [10]

Now apply the algorithm on these generated partial products. Algorithm for radix-8 Booth multiplication [11] as below:

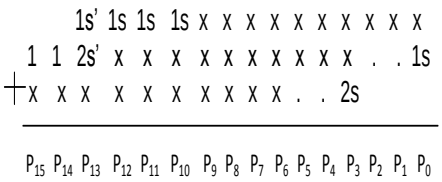

Figure 4: Radix-8 Algorithm

Where ‘x’ denotes the partial product,

‘S’ denotes the sign bit of particular group of multiplier bits.

3. CARRY LOOK-AHEAD ADDER

In a multiplier the overall computation time is mostly dominated by the final adder stage. This presents a particular problem when the multiplier is pipelined because the delay introduced by the adders. The problem may be solved by also pipelining the adder, but if the usual pipelined ripple-carry adder having a triangular structure is used then this will increase the flush time of the multiplier by an amount which may prove unacceptable. The carry look-ahead adder has the advantages of a short flush time and small stage delays, which allow the booth multiplier to be implemented with approximately equal stage delays throughout [12].

A carry look-ahead adder is a type of adder used in digital arithmetic operations. A carry look-ahead adder improves speed by reducing the amount of time required to determine carry bits. The CLA calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits. The operation perform by CLA for adding 4 bit numbers is as-

If A= A[3]A[2]A[1]A[0] and B=B[3]B[2]B[1]B[0] then A+B calculation using CLA

$P(0)= A(0) \oplus B(0)$, $G(0)= A(0) .B(0)$,

$P(1)= A(1) \oplus B(1)$, $G(1)= A(1) . B(1)$,

$P(2)= A(2) \oplus B(2)$, $G(2)= A(2) .B(2)$,

$P(3)= A(3) \oplus B(3)$, $G(3)= A(3) .B(3)$,

$C(1)= G(0) + [P(0) . Cin]$

$C(2)= G(1) + [P(1) .G(0)] + [P(1) .P(0) . Cin]$

$C(3)= G(2) + [P(2) . G(1)] + [P(2) . P(1) . G(0)] + [P(2) .P(1) . P(0) . Cin]$

$C(4)= G(3) + [P(3) . G(2)] + [P(3) . P(2) . G(1)] + [P(3) . P(2). P(1) . G(0)] + [P(3) . P(2) . P(1) . P(0) . Cin]$

And final A+B is given as

$S(0)= P(0) \oplus Cin$

$S(1)= P(1) \oplus C(1)$

$S(2)= P(2) \oplus C(2)$

$S(3)= P(3) \oplus C(3)$

$S(4)= Cout = C(4)$

Where ‘P’ ‘G’ and ‘C’ are signals and taken as reference and ‘S’ is the sum of the adder.

And so on for higher bits carry look-ahead adders.

4. IMPLEMENTATION AND RESULTS

After partial product generated, there are many techniques to add them to generate final product like using half adder and full adder, wallace tree, carry save adder, ripple carry adder etc. But in all these technique carry is propagates from LSB to MSB that increases propagation delay in the circuit. proposed Booth Decoded Multiplier final addition is done with CLAs as shown in Figure 1. Carry look-ahead adder (CLA) is the technique in which carry is not propagates rather it is simultaneously generates for all the columns to be added [13]. In this paper 8x8 bit multiplication is shown using radix-8 booth multiplication, where three partial product rows are generated. First these rows are added using conventional method that is using half adder and full adder to generates final product and for comparison with implemented circuit. Now the generated partial product rows are adding using 8 bit CLA and 5 bit CLA. Result comparison is shown in table below, where both circuits are simulated with V=1.8 V and transition time of 340nSec with W/L of 240nm/180nm in cadence virtuoso ETA tool.

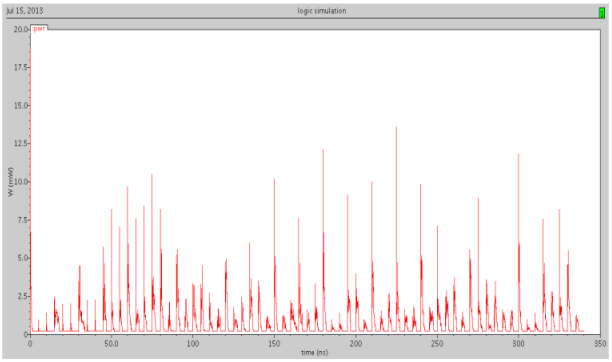

Figure 5: Average Power Dissipation Waveform

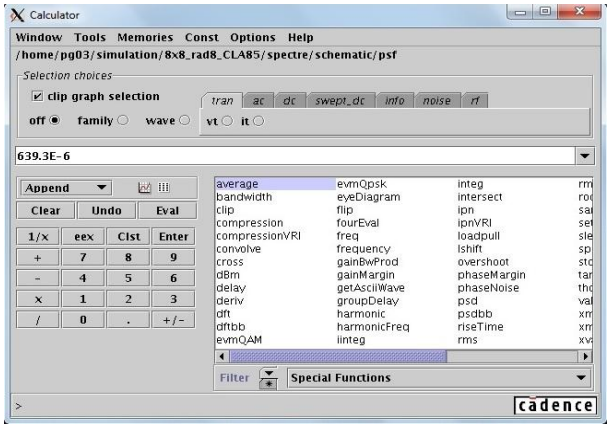

Figure 6: Average Power Dissipation

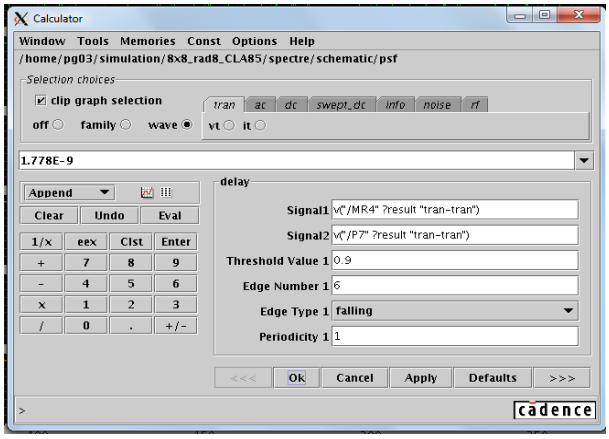

Figure 7: Propagation Path Delay

The different parameters like nodes, equations and number of accepted transition steps of both methods are shown in table 2. The average power dissipation in conventional method of adding is 722.2 µW but in proposed adding method the average power dissipation is 639.3 µW. From simulation and results, in conventional method propagation delay is 2.656 nSec and in proposed circuit propagation delay is 1.778 nSec.

Table 2: Simulation results

S. No.	Parameter	Conventional Method	Proposed Method
1.	**Nodes**	1759	2503
2.	**Equations**	8758	12478
3.	**bsim3v3**	3492	4980
4.	**vsource**	17	17
5.	**Number of Accepted Trans Steps**	22526	17940
6.	**Average Power Dissipation (µW)**	722.2	639.3
7.	**Maximum Propagation Delay (nSec)**	2.656	1.778

5. CONCLUSION

In proposed paper, the design and implementation of high performance parallel multiplier is presented usingradix-8 booth multiplier for 8x8 bit multiplication. The design for this multiplication implemented on cadence virtuoso EDA and simulated. From the simulation result, proposed design shows power dissipation and delay are reduced by 11.48% and 33.06% respectively. Hence it concludes that the performance of proposed design is better in terms of power dissipation and propagation delay as compared to the conventional design.

6. REFERENCES

[1] Jiun-Ping Wang, Shiann-Rong Kuang, Shish Chang Liang, "High Accuracy Fixed Width Modified Booth Multipliers for Lossy Applications", IEEE Trans 2011.

[2] Pouya Asadi and Keivan Navi, "A new low power 32×32 bit multiplier", World Applied Sciences Journal 2 (4): 341-347,2007.

[3] A.D. Booth, "A signed binary multiplication technique", Quart. J. Mech. Appl. Marh, vol.4, pp. 236-240, 1951. (Reprinted in [8, pp. 100-104])

[4] Razaidi Hussin, Ali Yeon Md. Shakaff, Norina Idris, Zaliman Sauli, Rizala fande CheIIsmail, and Afzan Kamaraudin, "An efficient modified Booth multiplier architecture", International Conference on Electronics Design, 978-1-4244-2315-6/08, 2008 IEEE.

[5] Ramya Muralidharan, Chip Hong Chang," Radix-4 and Radix-8 Booth encoded multi-modulas multipliers" IEEE Trans, 2013.

[6] Vignesh Kumar R., Kamala J., "High Accuracy Fixed Width Multipliers using Modified Booth Algorithm",International Conference on Modeling Optimization and Computing, Procedia Engineering 38(2012) 2491-2498.

[7] Aparna P R, Nisha Thomas, "Design and implementation of a High performance multiplier using HDL", IEEE 2012.

[8] Yajuan He, Chip Hong Chang,"A New Redundant Binary Booth Encoding for Fast 2^n-Bit Multiplier Design", IEEE Trans 2009.

[9] Philip E. Madrid, Brian Millar, Earl E. Swartzlander, "Modified Booth Algorithm for High Radix Fixed Point Multiplication", IEEE Trans 1993.

[10] Gary W. Bewick, " Fast Multiplication: Algorithms and Implementation" CSL-TR-94-617, Feb 1994.

[11] Ravindra P Rajput, M.N. Shanmukha Swamy, "High speed Modified Booth Encoder multiplier for signed and unsigned numbers", IEEE 2012.

[12] JUMP. J.R., and AHUJA,S. R., "Effective pipelining of digital systems", IEEE Trans., 1978, C-27, PP.855-865.

[13] Nirlakalla Ravi, T. Subba Rao, B. Bhaskara Rao, T. Jayachandra Prasadd, "A New Reduced Multiplication Structure for Low Power and Low Area Modified Booth Encoding Mulriplier", International Conference on Modeling Optimization and Computing, Procedia Engineering 38(2012) 2787-2771.