Analysis of High Speed Parallel Multiplier

[1]Manthan J. Trivedi, [2]Vimal H. Nayak, [3]Mohmmed G. Vayada
[1]P.G. Student, [2]Assistant Professor, [3]Assistant Professor.
Electronics & Communication
Silver Oak College of Engineering & Technology (SOCET), Gujarat Technological University- Ahmedabad, India.

***Abstract* - This research paper represents the analysis of Advanced Modified Booth Encoder (AMBE) parallel Multiplier with the already invented Booth multiplier and Modified Booth Encoding Multiplier. The existed Booth and Baugh Wooley Multipliers are used for only signed numbers, while array multipliers uses only for unsigned numbers. Modern Computer system needs a very high speed parallel multiplier which is used for signed and unsigned numbers. This multiplier is obtained by extending a sign bit from Modified Booth Encoder and generates an additional partial product. The Carry Save Adder tree (CSA) and Carry Look Ahead Adder (CLA) are used to add all partial products and generates the final product. This multiplier uses for both signed and unsigned numbers so total chip area reduces and power reduces as well. The Advanced Modified Booth Encoding parallel multiplier is simulated using Verilog-HDL language in Xilinx 13.2ISE simulator and implements on Spartan 3E board.**

***Index Terms* - Modified Booth Encoding Multiplier, CSA, CLA, signed-unsigned numbers.**

I. INTRODUCTION

Multiplication is the most commonly used in every step of the world. It is nothing but the addition by the multiplicand adds multiplier no. of times. But, it takes very large hardware resources and simulation takes very large time for the final output. The main aim in VLSI field to make a multiplier which increase the speed and reduces the area and power. The multiplication mainly consists of two steps i) To generate Partial products and ii) To add all these partial products until the final output **[1-7]**.

The High Speed Booth & Pipelined multipliers are used in DSP applications, like multimedia and communication system. Booth Algorithm provides multiplying binary integers in 2's complement format and generates partial product by recoding the bits **[4]**. The numbers of partial products in Booth multiplier are equal to the numbers of multiplicand & multiplier bits. The Modified Booth Encoding Multiplier reduces the partial products by factor of 2 **[2]**. The main drawback of this multiplier was not using the sign bit, irregular partial product and used only for the signed numbers.

The Braun array multiplier also reduces the partial product to increase the speed of multiplier but it is used only for unsigned numbers **[8]**. So, in present system high speed multiplier which can perform on signed and unsigned number a well. So, Advanced Modified Booth Encoding Algorithm is being used for both signed and unsigned numbers.

II. BOOTH ALGORITHM

Booth algorithm provides a procedure for multiplying binary integers in signed-2's complement representation. According to the multiplication procedure, strings of 0's in the multiplier require no addition but just shifting and a string of 1's in the multiplier from bit weight 2^k to weight 2^m can be treated as $2^{k+1} - 2^m$ **[2,5]** .To recode the Booth Multiplier, following rules are followed.

- Append the given multiplier with a zero to the LSB side.
- Make group of two bits in the overlapped way.

The recoding of bits in Booth multiplier is shown below in Table 1.

Table 1 Booth Recoding Rules

Q_n	Q_{n-1}	Recoded Bits	Operation Performed
0	0	0	Shift
0	1	+1	Add
1	0	-1	Substract
1	1	0	Shift

Table 1 Shows how the Shifting, Addition and Subtraction performance is done. The Original Booth Algorithm has two disadvantages: i) The number of Add and Subtract operation varies. ii) The algorithm becomes inefficient when there are isolated 1's.

III. MODIFIED BOOTH ENCODING ALGORITHM

Modified Booth Encoding Algorithm (MBE) overcomes the disadvantages of the Original Booth Algorithm. MBE reduces the partial products by a factor of 2.If there are n bits of multiplicand and multiplier then MBE algorithm generates n/2 partial products where Booth generates n. The Modified Booth Encoder recodes the multiplier bits by overlapping of 3 bits.

Architecture of Modified Booth Encoding Algorithm

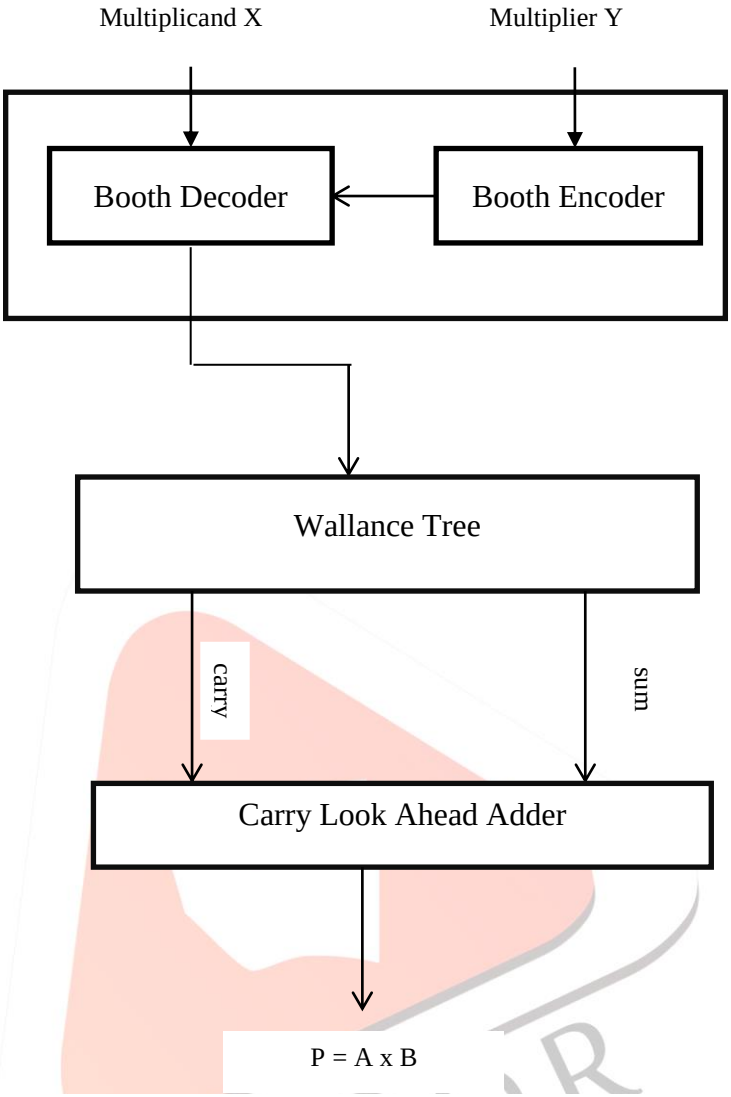

Figure 1 Architecture of Modified Booth Encoding Multiplier

Figure 1 represents the architecture diagram of Modified Booth Encoding parallel Multiplier. The inputs of the multiplier are X (Multiplicand) and Y(Multiplier). The Booth Encoder encodes the input Y and derives the encoded signals by overlapping of 3 bits {-2, -1, 0, 1, 2}. The Modified Booth Encoding Algorithm follows the below equation.

$$\textbf{Value} = -2\times b_{i+1} + b_i + b_{i-1} \quad (1)$$

Equation 1 represents how the value of the encoder can be generated from 3 overlapping bits. The truth table for Modified Booth Encoder is shown below.

Table 2 Truth Table of Modified Booth Encoder

b_{i+1}	b_i	b_{i-1}	Value
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	2
1	0	0	-2
1	0	1	-1
1	1	0	-1
1	1	1	0

IV. Proposed Advanced Modified Booth Encoding Multiplier (AMBE)

By extending a sign bit in the Previous algorithm, an extra partial product is generated and to use for both signed and unsigned numbers. The AMBE also follows the same architecture diagram that have been shown in figure 1.There are Booth Encoder, Booth

Decoder, Wallance Tree adder and Carry Look Ahead adder blocks are used. The Booth Encoder encodes the multiplier bits and generates the encoded signals {-2, -1, 0, 1, 2}.

Table 3 Truth Table for AMBE Multiplier

b_{i+1}	b_i	b_{i-1}	Value	X1_a	X2_a	Z	Neg
0	0	0	0	1	0	1	0
0	0	1	1	0	1	1	0
0	1	0	1	0	1	0	0
0	1	1	2	1	0	0	0
1	0	0	-2	1	0	0	1
1	0	1	-1	0	1	0	1
1	1	0	-1	0	1	1	1
1	1	1	0	1	0	1	1

Table 3 represents the truth table of Advanced Modified Booth Encoding Parallel Multiplier. The Booth encoder encodes the the signals X1_a, X2_a, and Z. The logic diagram for Booth Encoder is shown in **Fig.2.**

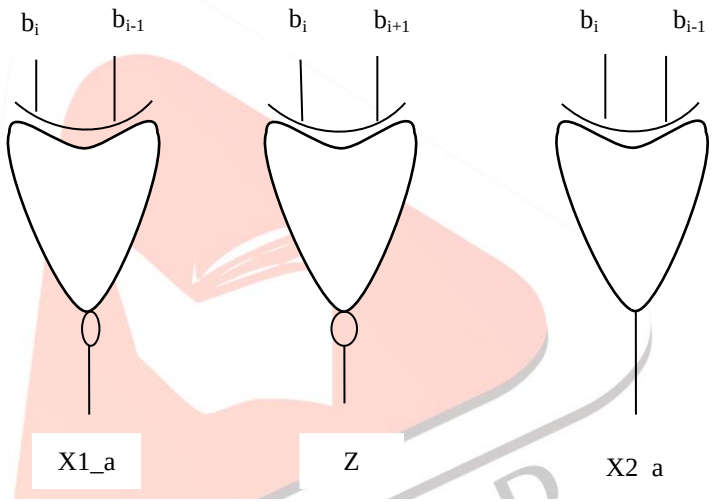

Figure 2 Logic Diagram of Booth Encoder

Using MBE logic, Booth decoder generates the partial product bit which is given by **Equation 2.**

$$pij = (a_i \oplus b_{i+1} + b_{i-1} \oplus b_i)\,(\,a_{i-1} \oplus b_{i+1} + b_i \oplus b_{i+1} + b_{i-1} \oplus b_i\,) \qquad (2)$$

Equation 2 is implemented as shown in **Fig.3.** The AMBE multiplier does not separately consider the encoder and the decoder logic, but instead implemented as a single unit called partial product generator as shown in **Fig. 3**. The negative partial products are converted into 2's complement by adding a negate (Ni) bit. The Negative bit is represented by **Equation 3**. The Equation 3 is implemented as shown in **Fig.4**.The sign extension bit is converted to signed-unsigned number by **Equations 4** and **Equations 5**. The logic Diagram of **Equation 4** and **Equation 5** are implemented as shown in **Fig.5**.

$$N_i = b_{i+1}\,(\overline{b_{i-1}\ b_i}) \qquad (3)$$

$$a_8 = s\_u * a_7 \qquad (4)$$

$$b_8 = s\_u * b_7 \qquad (5)$$

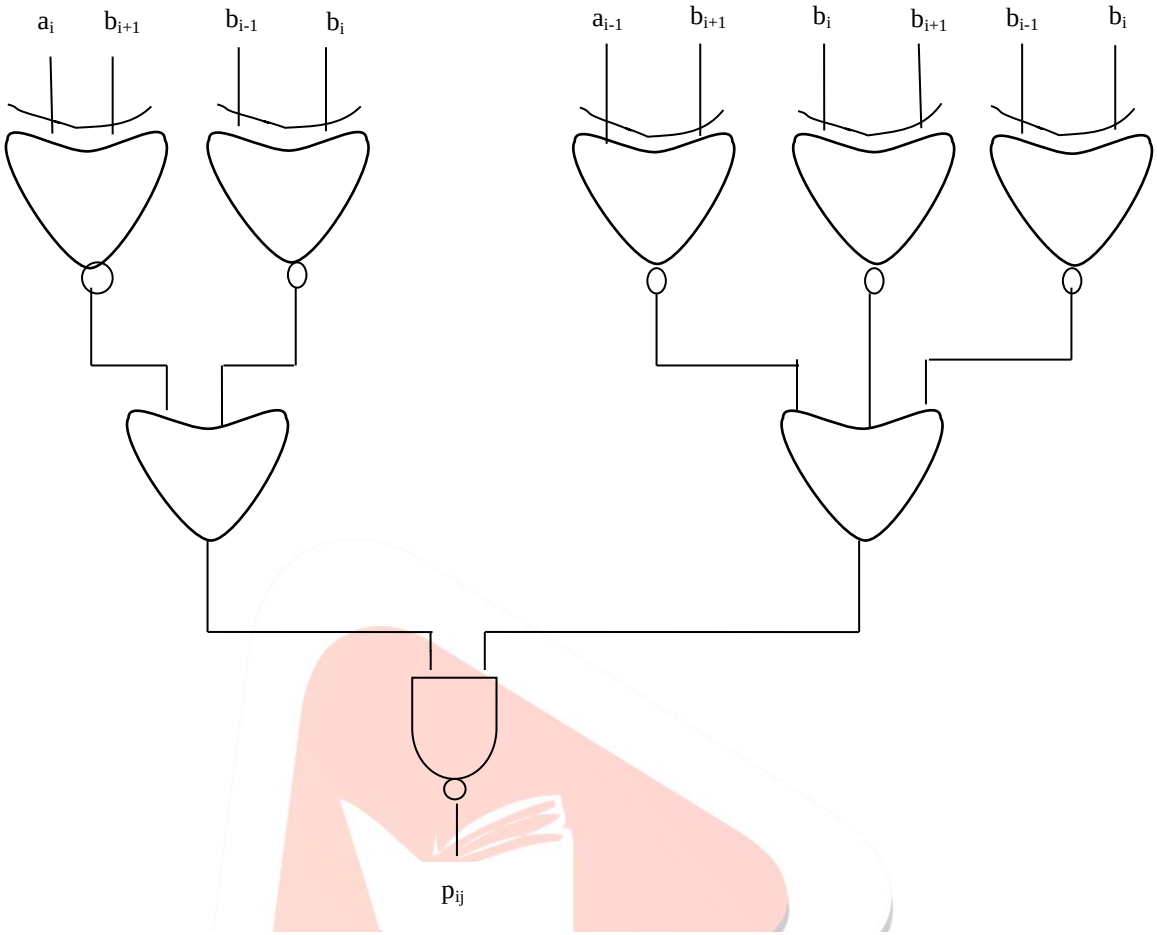

Figure 3: 1-bit partial product generator

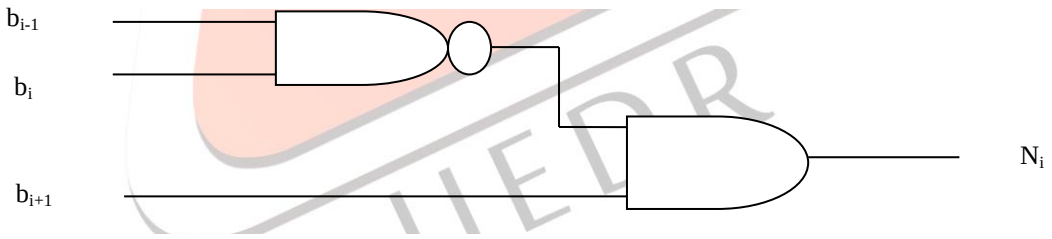

Figure 4: Logic Diagram of Negative bit generator

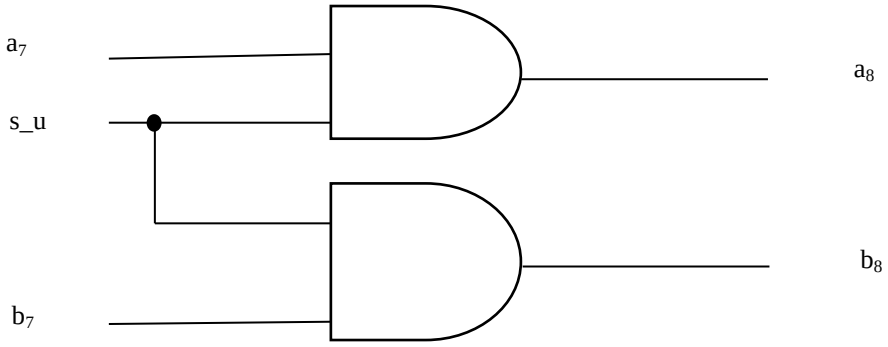

Figure 5: Logic Diagram of sign converter

The sign Extension converter is used to convert signed multiplier into signed- unsigned multiplier by inserting s_u bit to indicate whether signed multiplier or unsigned multiplier. If s_u bit = 0, it indicates unsigned number multiplication. If s_u bit =1, it indicates signed number multiplication. It is very necessary that during unsigned multiplication, multiplicand and multiplier both

are extended with 0 "i.e. $a_8=a_9=b_8=b_9=0$". During signed multiplication, the sign extended bit depends on whether multiplicand is negative, multiplier is negative or both are negative. When the multiplicand is negative and multiplier is positive then $a_7=1$, $b_7=0$, $s\_u=1$, $a_8=a_9=1$, $b_8=b_9=0$. When the multiplicand is positive and multiplier is negative then $s\_u=1$, $a_7=0$, $b_7=1$, $a_8=a_9=0$, $b_8=b_9=1$.

$a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0$

$b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$

$p_{08}\ \overline{p_{08}}\ p_{08}\ p_{07}\ p_{06}\ p_{05}\ p_{04}\ p_{03}\ p_{02}\ p_{01}\ p_{00}$ X1

$1\ \overline{p_{18}}\ p_{17}\ p_{16}\ p_{15}\ p_{14}\ p_{13}\ p_{12}\ p_{11}\ p_{10}$ N_0 X2

1----------p_{20} N_1 X3

1 ----------p_{30} N_2 X4

$P_{47}\ p_{46}\ p_{45}\ p_{44}\ p_{43}\ p_{42}\ p_{41}\ p_{40}$ N_3 X5

$p_{15}\ p_{14}\ p_{13}\ p_{12}\ p_{11}\ p_{10}\ p_9\ p_8\ p_7\ p_6\ p_5\ p_4\ p_3\ p_2\ p_1\ p_0$

Figure 6 : 8 x 8 Multiplier for signed-unsigned numbers

Figure 6 represents the partial products generated from **Fig.3**. There are 5 partial products and Negative bit generator. These partial products are X1, X2, X3, X4 and X5. These partial products are added by Wallance tree Adder which uses Carry Save Adder circuit. It adds partial products until final two outputs are left. Carry Save Adder (CSA) tree diagram is shown in **Fig. 7** below. When there are only two outputs are left, Carry Look Ahead (CLA) Adder is used to do the final addition and give the Final product. Take each gate delay as an unit delay then including partial products, total delay is calculated as 3+4=7. During simulation in VLSI delay is calculated in nano second and operated on Giga Hertz frequency.

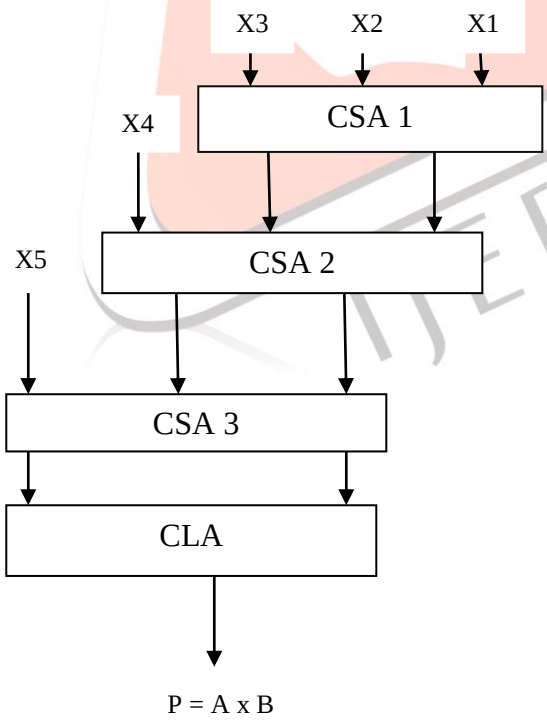

Figure 7: Partial Product Adder Logic

V. RESULTS

After designing the whole algorithm, the delay, area and power can be estimated as shown in **Table 4**.

Table 4 Comparative Results of Booth Encoder & Modified Booth Encoder

Parameters	Booth Encoder	Modified Booth Encoder
Delay	5.5ns	3.58ns
Power	0.153mW	0.76mW
Area	6603mm^2	6091mm^2

VI. CONCLUSION

In this paper, the existed Booth Multiplier and Modified Booth Encoding Multiplier and the proposed 8 x 8 signed-unsigned multiplier is presented theoretically. The delay, area and power of these all multiplier can be estimated. The proposed Advanced Modified Booth Encoding Multiplier is very much useful to increase the speed of the computer system by reducing the delay, area and power.

REFERENCES

[1] ravindra p rajput , m. n shanmukhaswamy," high speed modified booth encoder multiplier for signed and unsigned numbers", 2012 ,14th international conference on modelling and simulation, doi 10.1109/uksim.2012.99.

[2] Young-Hoseo& Dong-wook kim , a new vlsi architecture of parallel multiplier–accumulator based on radix-2 modified booth algorithm. ieee transactions on very large scale integration (vlsi) systems, vol. 18, no. 2, february 2010.

[3] Shiann-RongKuang, Jiun-Ping Wang, and Cang-Yuan Guo, "Modified Booth multipliers with a Regular Partial Product Array," IEEE Transactions on circuits and systems-II, vol 56, No 5, May 2009.

[4] Nishant Bano, "VLSI Design of Low Power Booth Multiplier", International Journal of Scientific & Engineering Research, Volume 3, Issue 2, February -2012, ISSN 2229-5518

[5] Jung-Yup Kang and Jean-Luc Gaudiot, "A simple high-speed multiplier design," IEEE Trans. on Computers, vol. 55, issue 10, Oct. pp. 1253-1258, 2006.

[6] Li- Rung Wang, Shyh-JyeJou and Chung-Len Lee, "A well-tructured Modified Booth Multiplier Design" 978-1-4244-1617-2/08/$25.00 ©2008 IEEE.

[7] Shaikh Kailash Baba& D.Rajaramesh, "Design & Implementation of Advanced Modified Booth Encoding Multiplier", International Journal of Engineering Science Invention ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726, Volume 2 Issue 8 ǁ August. 2013 ǁ PP.60-68.

[8] Wang, G., "A unified unsigned/signed binary multiplier", TheThirty-Eighth Alomar Conference on Signals, Systems andComputers, 2004, Vol. 1, pp.:513 - 516, Nov 7-10, 2004.