INTERNATIONAL JOURNAL OF RESEARCH
In Advanced Engineering Technologies

Volume 2, Issue 1 SEP 2014

FPGA IMPLEMENTATION OF EFFCIENT MODIFIED BOOTH ENCODER MULTIPLIER FOR SIGNED AND UNSIGNED NUMBERS

NUNAVATH.VENNELA [(1)], A.VIKAS [(2)]

P.G.Scholor (VLSI SYSTEM DESIGN),TKR COLLEGE OF ENGINEERING[(1)]

M.TECHASSISTANT PROFESSOR,TKR COLLEGE OF ENGINEERING[(2)]

ABSTRACT-- -This paper presents the design and implementation of signed-unsigned Modified Booth Encoding (SUMBE) multiplier. The present Modified Booth Encoding (MBE) multiplier and the Baugh-Wooley multiplier perform multiplication operation on signed numbers only. The array multiplier and Braun array multipliers perform multiplication operation on unsigned numbers only. Thus, the requirement of the modern computer system is a dedicated and very high speed unique multiplier unit for signed and unsigned numbers. Therefore, this paper presents the design and implementation of SUMBE multiplier. The modified Booth Encoder circuit generates half the partial products in parallel. By extending sign bit of the operands and generating an additional partial product the SUMBE multiplier is obtained. The Carry Save Adderr (CSA) tree and the final Carry Lookahead (CLA) adder used to speed up the multiplier operation. Since signed and unsigned multiplication operation is performed by the same multiplier unit the required hardware and the chip area reduces and cost of a system.

Index Terms: Modified Booth Encoding multiplier, Baugh Wooley multiplier, Array multiplier, Braun array multiplier, CSA, CLA, partial products, Signed-unsigned.

I. INTRODUCTION

In digital computing systems multiplication is an arithmetic operation. The multiplication operation consists of producing partial products and then adding these partial products the final product is obtained. Thus the speed of the multiplier depends on the number of partial product and the speed of the adder. Since the multipliers have a significant impact on the performance of the entire system, many high performance algorithms and architectures have been proposed. The very high speed and dedicated multipliers are used in pipeline and vector computers. The high speed Booth multipliers and pipelined Booth multipliers are used for digital signal processing (DSP) applications such as for multimedia and communication systems. High speed DSP computation applications such as Fast Fourier transform (FFT) require additions and multiplications. The papers presents a design methodology for high speed Booth encoded parallel multiplier. For partial product generation, a new Modified Booth encoding (MBE) scheme is used to improve the performance of traditional MBE schemes. But this multiplier is only for signed number multiplication operation. The conventional modified Booth encoding (MBE) generates an irregular partial product array because of the extra partial product bit at the least significant bit position of each partial product row. Therefore papers presents a simple approach to generate a regular partial product array with fewer partial product rows and negligible overhead, there by lowering the complexity of partial product reduction and reducing the area, delay, and power of MBE multipliers. But the drawback of this multiplier is that it function only for signed number operands. The modified-Booth algorithm is extensively used for high-speed multiplier circuits. Once, when array multipliers were used, the reduced number of generated partial products significantly improved multiplier performance. In designs based on reduction trees with logarithmic logic depth, however, the reduced number of partial products has a limited impact on overall performance. The Baugh-Wooley algorithm is a different scheme for signed multiplication, but is not so widely adopted because it may be complicated to deploy on irregular reduction trees. Again the Baugh-Wooley algorithm is for only signed number multiplication. The array multipliers and Braun array multipliers operates only on the unsigned numbers. Thus, the requirement of the modern computer system is a dedicated and very high speed multiplier unit that can perform multiplication operation on signed as well as unsigned numbers. In this paper we designed and implemented a dedicated multiplier unit that can perform multiplication operation on both signed and unsigned numbers, and this multiplier is called as SUMBE multiplier.

2. CONVENTIONAL MBE MULTIPLIERS

The new MBE recorder was designed according to the following analysis. Table1 presents the truth table of the new encoding scheme. The Z signal makes output zero to compensate the in correct X2_b and neg signals. Fig. 1presents the circuit diagram of the encoder and decoder. The encoder generates X1_b, X2_b,Z signals by encoding the three x-signals. The y LSB signal is the LSB of the y signals and the combination with x - signals to determine the Row_LSB and the Neg_cin signals. Similarly,y msb Is combined with x-signals to determine the sign extension signals. Fig. 2 shows an overview of the partial product array for an 8 * 8 multiplier. The sign extension circuitry developed. The convention al MBE partial products array has two drawbacks: 1) an additional partial product term at the (n-2)th bit position; 2) poor performance at the LSB-part. To remedy the two drawbacks,thenewequations for the Row_LSB and Neg_cin can be written as(1) and (2) respectively

$$Row\_LSB = y_{LSB}(x2i + x2i) \quad (1)$$

$$Neg\_cin_i = x_{2i+1}\overline{(x_{2i-1} + x_{2i-1})}\,\overline{(x_{2i-1} + y_{LSB})}\,\overline{(x_{2i} + y_{LSB})} \quad (2)$$

TABLE 1: Truth Table of MBE Scheme

$b_{i+1}\ b_i\ b_{i-1}$	Value	X1_a	X2_b	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

Fig. 2(a) has widely been adopted in parallel multiplier Since it can reduce the number of partial product rows to be added by half, thus reducing the size and enhancing the speed of reduction tree. However, as shown Fig. 1(a), the conventional MBE algorithm generates n/2 + 1 partial product rows rather than n/2 due to the extra partial product bit (neg bit) at the least significant bit position of each partial product row for negative encoding, leading to an irregular partial product array and a complex reduction tree. Therefore the modified booth multiplier with a regular partial product array produces a very regular partial product array, as shown in Fig. 3. Not only each Neg is shifted to left and replaced by Ci but also the last neg bit is removed this approach reduces the partial product n/2+1 to n/2 by incorporating the last
neg bit into the sign extension bits of the first partial product row, and almost no overhead is introduced to the partial product array and fewer partial product rows result in a small and fast reduction tree, so that area, delay, and power of MBE multipliers can further be reduced.

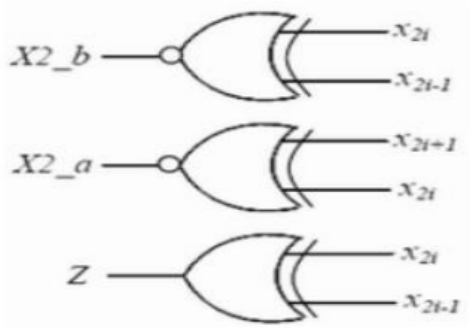


Fig: Decoder For MBE

3. PROPOSED SUMBE MULTIPLIER

The main goal of this paper is to design and implement 16x16 multiplier for signed unsigned numbers using MBE technique. Table2 shows that truth table of MBE scheme. From table2 the MBE logic and considering other conditions the Boolean expression for one bit partial product generator
is given by equation 3.

$b_{i+1}\ b_i\ b_{i-1}$	Value	X1_a	X2_b	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	0

Fig: Truth table of MBE scheme

$$P_{ij} = \overline{(\overline{a_i}+b_{i+1}+\overline{b_{i-1}}+b_i)\,(a_{i-1}+\overline{b_{i+1}}+\overline{b_i}+\overline{b_{i+1}}+b_{i-1}+b_i)}$$

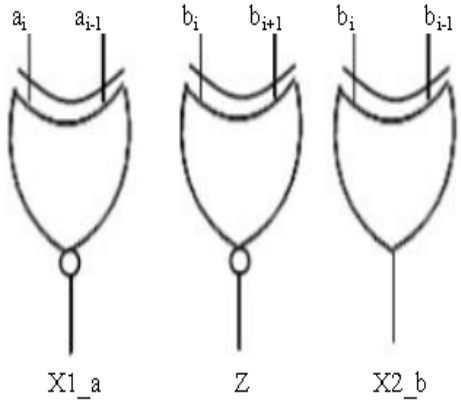


Fig: Logic Diagram of MBE

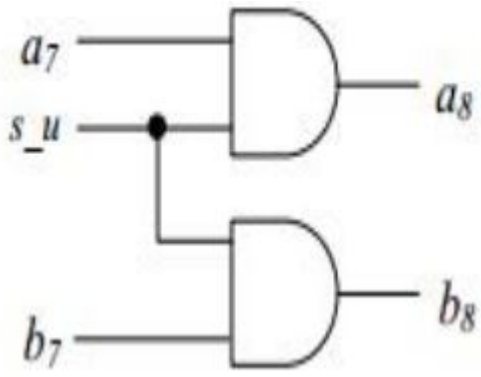
Equation 3 is implemented as shown in Fig. 3.The SUMBE multiplier does not separately consider the encoder and the decoder logic, but instead implemented as a single unit called partial product generator as shown in Fig. 3.The negative partial products are converted into 2's complement by adding a negative (N_i) bit. An expression for negate bit is given by the Boolean equation4. This equation is implemented as shown in Fig. 4. The required signed extension to convert 2's complement singed multiplier into both signed-unsigned multipliers is given by the equation 3 and 4. For Boolean equations 5 and 6 the corresponding logic diagram is shown in Fig. 5

.

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

$$a_{16} = s\_u.a_{15}$$

$$b_{16} = s\_u.b_{15}$$



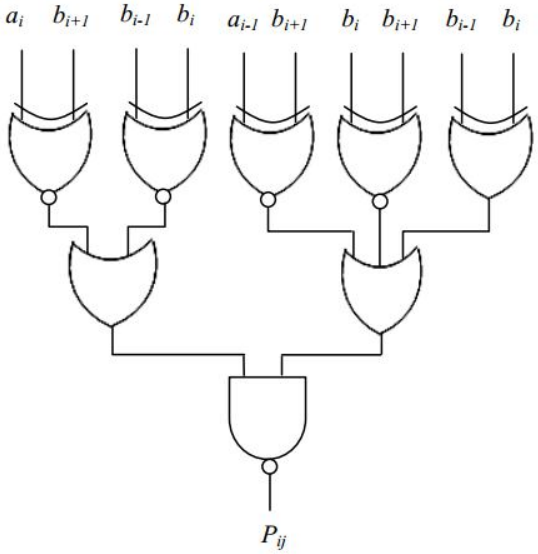


Fig: Logic diagram of 1-bit partial product generator

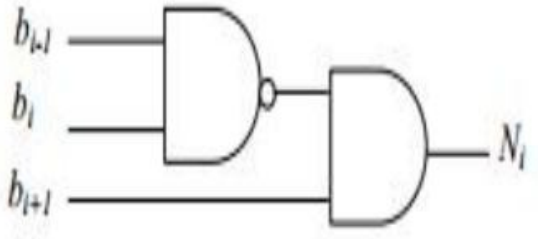


Fig: Logic Diagram of negate bit generator

Fig: Logic Diagram of sign converter

The working principle of sign extension that converts signed multiplier signed - unsigned multiplier as follows. One bit control signal called signed-unsigned (s_u) bit is used to indicate whether the multiplication operation is signed number or unsigned number. When sign - unsigned (s_u) = 0 it indicates unsigned number multiplication, and when

s_u=1, it indicates signed number multiplication. It is required that when the operation is unsigned multiplication the sign the extended bit both multiplication and multiplier should be extended with 0, that is a 16=a17=b16=b17= 0.Is required that when the operation is signed multiplication the sign extended bit depends on whether the multiplication is negative or the multiplier is negative or both the operands are negative. For this when the multiplicand operand is negative and multiplier operand is positive operand is negative and multiplier operand is positive the sign extended bit should be generated are s_u=1, a15= 1, b15= 0, a16= a17=0, and b16= b17=1.Table 3 shows the SUMBE multiplier operation.

Sign-unsigned	Type of operation
0	Unsigned multiplication
1	Signed multiplication

Fig:SUMBE operation

Shows the partial products generated by partial product generator circuit which is shown in Fig. 3. There are 5-partial products which sign extension and negate bit Ni.all the 9-partial products are generated in parallel.

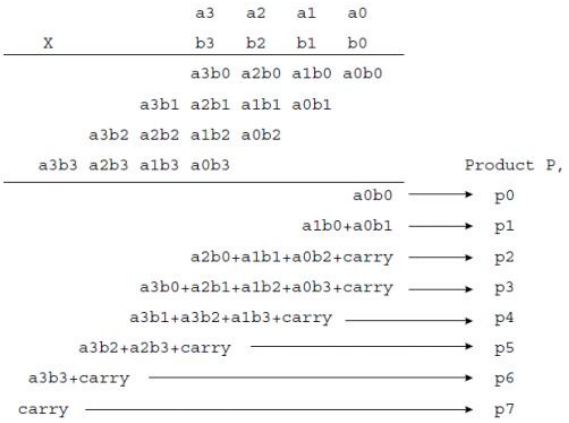


Fig: multiplier for signed-unsigned numbers using bugh vally

there are 9-partial products namely X1, X2, X3,X4, X5, X6, X7, X8and X9. These are 9-partial products are added by the carry save adders (CSA) and the final stage is carry look ahead (CLA) adder as shown in Fig.7.Each CSA adder takes three inputs and produce sum and carry in parallel. There are three CSAs, five partial products are added by the CSA tree and finally when there are only two outputs left out then finally CLA adder is used to produce the final result. Assuming each gate delay an unit delay, including partial product generator circuit delay, then the total through the CSA and CLA is 3+4=7 unit delay. Thus with present very large scale integration (VLSI) the total delay is estimated around 0.7 nanosecond and the multiplier operates at giga hertz frequency.

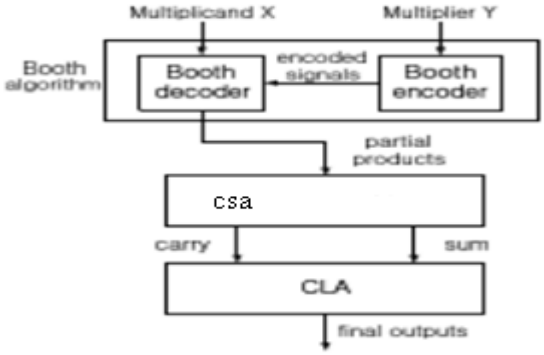


Fig: Architecture of the modified Booth multiplier

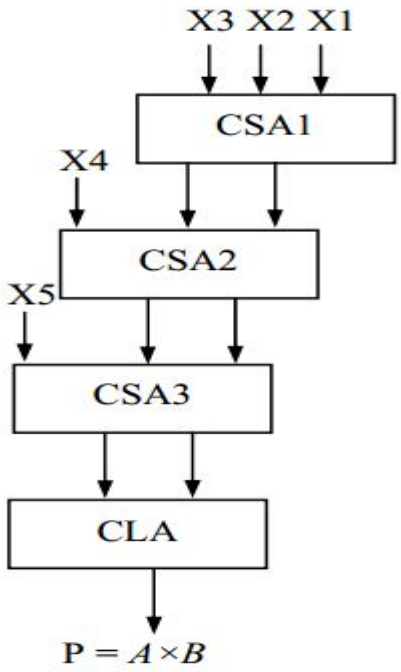
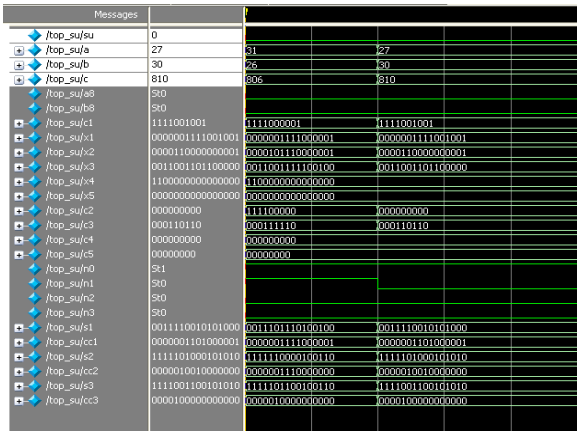


Fig: Partial product adder logic

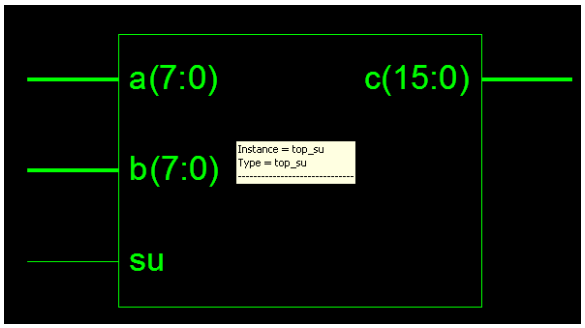
IV. SIMULATION RESULTS

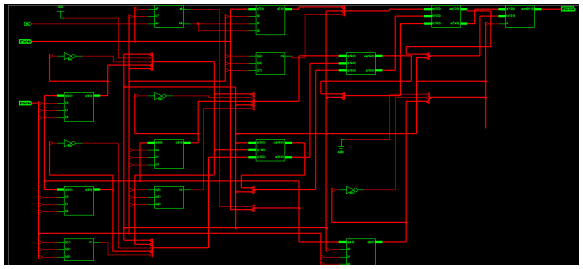
Verilog code is written to generate the required hardware and to produce the partial product, for CSA adder, and CLA added.



SYNTHESIS REPORT:

Device Utilization Summary (estimated values)		
Logic Utilization	Used	Available
Number of Slices	91	4656
Number of 4 input LUTs	161	9312
Number of bonded IOBs	33	232





V. CONCLUSION

In all multiplication operation product is obtained by adding partial products. Thus the final speed of the multiplier circuit depends on the speed of the adder circuit and the number of JOURNAL OF COMPUTERS, VOL. 5, NO. 10, partial products generated. If radix 8 Booth encodind technique is used then there are only 3 partial products and for that only one CSA and a CLA is required to produce the final product.

REFERENCES

[1] W. –C. Yeh and C. –W. Jen, "High Speed Booth encoded Parallel Multiplier Design," IEEE transactions on computers, vol. 49, no. 7, pp. 692-701, July 2000.

[2] Shiann-Rong Kuang, 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.

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

[4] Soojin Kim and Kyeongsoon Cho "Design of High-speed Modified Booth Multipliers Operating at GHz Ranges" World Academy of Science, Engineering and Technology 61 2010.

[5] Magnus Sjalander and Per Larson-Edefors. "The Case for HPM-Based Baugh-Wooley Multipliers," Chalmers University of Technology, Sweden, March 2008.

[6] Z Haung and M D Ercegovac, "High performance Low Power left to right array multiplier design" IEEE trans.Computer, vol 54 no3, page 272-283 Mar 2005.

[7] Hsing-Chung Liang and Pao-Hsin Huang, "Testing Transition Delay Faults in Modified BoothMultipliers by Using C-testable and SIC Patterns"IEEE2007, 1-4244-1272-2/07.

[8] Aswathy Sudhakar, and D. Gokila, "Run-Time Reconfigurable Pipelined Modified Baugh-Wooley Multipliers," Advances in Computational Sciences and Technology ISSN 0973-6107 Volume 3 Number 2 (2010) pp. 223–235.

[9] Myoung-Cheol Shin, Se-Hyeon Kang, and In-Cheol Park, "An Area- Efficient Iterative Modified-Booth Multiplier Based on Self-Timed Clocking," Industry, and Energy through the project System IC 2010, and by IC Design Education Center (IDEC).

[10] Leandro Z. Pieper, Eduardo A. C. da Costa, Sérgio J. M. de Almeida, "Efficient Dedicated Multiplication Blocks for2´s Complement Radix-2m Array Multipliers,"

OCTOBER 2010.

[11] C R Baugh and B. A Wooley, " A two's complement parallel array multiplication algorithm," IEEE Transaction on Computers, Vol. 22, n0.12,pp 1045-1047, Dec.1973.

[12] Neil H E Weste, David Harris, Ayan Banerjee, "CMOS VLSI Design A circuits and Systems Perspective " Third edition, Pearson Education, pp.347-349.

[13] Pucknell Douglas A, Eshraghan, Kamran, "Basic VLSI

Design,"Third edition 2003, PHI Publication, pp.242-243.

[14] A Chandrakasan and R Brodersen, " Low power CMOS digital design," IEEE J solid state circ, vol 27 no. 4, April 1992, pp. 473-484.

[15] C. S Wallace, "A suggestion for a fast multiplier" IEEE Transaction on Electronic Computers, pp-14-17, Feb-1974.

[16] I Koren, Computer Arithmetic Algorithms. Englewood Cliffs, New Jersey, Prentice Hall, 1993, pp.99-123.

[17] M D Ercegovac, and T Lang, " Fast multiplication without carry propagate addition" IEEE Transaction on computers, vol.39, no.11, pp. 1385-1390. Nov 1990