ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Design and Implementation of Advanced Modified Booth Encoding Multiplier

B.Sirisha
M.Tech Student,
Department of Electronics and communication Engineering, GDMM College of Engineering and Technology.

G.Swarna Kumari
Asst.Professor,
Department of Electronics and communication Engineering, GDMM College of Engineering and Technology.

ABSTRACT:

This paper presents the design and implementation of Advanced Modified Booth Encoding (AMBE) multiplier for both signed and unsigned 32 - bit numbers multiplication. The already existed Modified Booth Encoding multiplier and the Baugh-Wooley multiplier perform multiplication operation on signed numbers only. Where as 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 AMBE 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 AMBE multiplier is obtained. The Carry Save Adder (CSA) tree and the final Carry Look ahead (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 this in turn reduces power dissipation and cost of a system.

KEY WORDS:

Modified Booth Encoding multiplier, CSA,CLA, Signed-unsigned.

I.INTRODUCTION:

Multiplication is a most commonly used operation in many computing systems. Infact multiplication is nothing but addition since, multiplicand adds to itself multiplier no.of times gives the multiplication value between multiplier and multiplicand.

But considering the fact that this kind of implementation really takes huge hardware resources and the circuit operates at utterly low speed. In order to address this so many ideas have been presented so far for the last three decades. Each one is aimed at particular improvement according to the requirement.

One may be aimed at high clock speeds and another mayb e aimed for low power or less area occupation. Either way ultimate job is to come up with an efficient architecture which can address three constraints of VLSI speed, area, and power.

Among these three speed is the one which requires special attention. If we observe closely multiplication operation involves two steps one is producing partial products and adding these partial products [3]. Thus, the speed of a multiplier hardly depends on how fast generate the partial products and how fast we can add them together. If the number of partial products to be generated are of less then it is indirectly means that we have achieved the speed in generating partial products. Booth's algorithms are meant for this only.

To speed up the addition among the partial products we need fast adder architectures. Since the multipliers have a significant impact on the performance of the entire system, many high performance algorithms and architectures have been proposed [1-12]. 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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

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 [4] 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 [7,8,9] 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 [10] 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 AMBE multiplier.

II.CONVENTIONAL MODIFIED BOOTH MULTIPLIER:

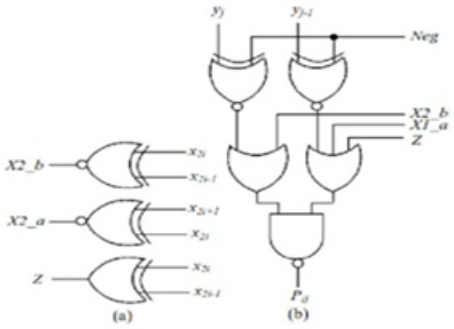
1.Algorithm of the Modified Booth Multiplier:

Multiplication consists of three steps: 1) the first step to generate the partial products; 2) the second step to add the generated partial products until the last two rows are remained; 3) the third step to compute the final multiplication results by adding the last two rows.

The modified Booth algorithm reduces the number of partial products by half in the first step. We used the modified Booth encoding (MBE) scheme proposed in [2]. It is known as the most efficient Booth encoding and decoding scheme. To multiply X by Y using the modified Booth algorithm starts from grouping Y by three bits and encoding into one of {-2, -1, 0, 1, 2}. Table I shows the rules to generate the encoded signals by MBE scheme and Fig. 1 (a) shows the corresponding logic diagram. The Booth decoder generates the partial products using the encoded signals as shown in Fig. 1(b).

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



MBE scheme. (a) Simple encoder (b) Decoder.

The new MBE recorder [2] was designed according to the following analysis. Table (1) presents the truth table of the new encoding scheme. The Z signal makes the output zero to compensate the incorrect X2_b and Neg signals. Fig. 1 presents the circuit diagram of the encoder and decoder. The encoder generates X1_b, X2_b, and Z signals by encoding the three x-signals. The y LSB signal is the LSB of the y signal and is combined with x-signals to determine the Row_LSB and the Neg_cin signals. Similarly, yMSB is combined with x- signals to determine the sign extension signals.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

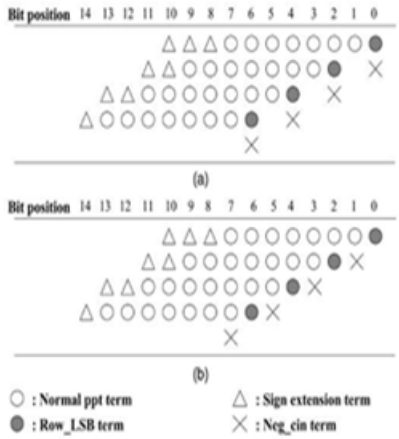


Fig. 2. 8× 8 MBE partial product array. (a) Traditional MBE partialproduct array. (b) New MBE partial product array.

The Fig. 2(a) has widely been adopted in parallel multipliers since it can reduce the number of partial product rows to be added by half, thus reducing the size and enhancing the speed of the reduction tree. However, as shown in 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 multipliers with a regular partial product array [4] produce a very regular partial product array, as shown in Fig. 3. Not only each negi is shifted left and replaced by ci but also the last neg bit is removed. This approach reduces the partial product rows from 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 generator. More regular partial product array and fewer partial product rows result in a small and fast reduction tree, so that the area, delay, and power of MBE multipliers can further be reduced.

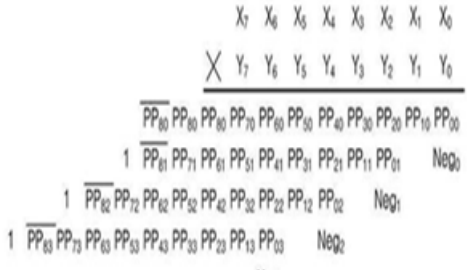


Fig. 3 Generated partial products and sign extension scheme

Fig. 3 shows the generated partial products and sign extension scheme of the 8-bit modified Booth multiplier. The partial products generated by the modified Booth algorithm are added in parallel using the Wallace tree until the last two rows are remained. The final multiplication results are generated by adding the last two rows. The carry propagation adder is usually used in this step.

2.2.Architecture of the Modified Booth Multiplier:

Fig. 4 shows the architecture of the commonly used modified Booth multiplier. The inputs of the multiplier are multiplicand X and multiplier Y. The Booth encoder encodes input Y and derives to the encoded signals as shown in Fig. 1 (a). The Booth decoder generates the partial products according to the logic diagram in Fig. 1 (b) using the encoded signals and the other input X. The Wallace tree computes the last two rows by adding the generated partial products. The last two rows are added to generate the final last two rows by adding the generated partial products. The last two rows are added to generate the final multiplication results using the carry look-ahead adder (CLA).

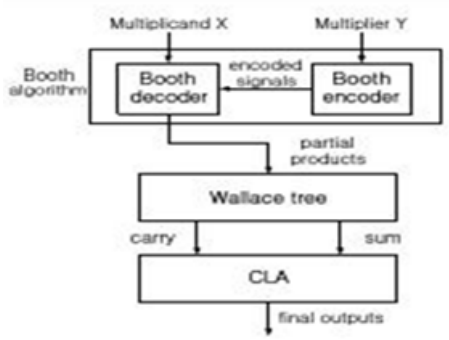


Fig. 4 Architecture of the modified Booth multiplier.

III. PROPOSED AMBE MULTIPLIER:

The main goal of this paper is to design and implement 32×32 multiplier for signed and unsigned numbers using MBE technique. Table 2 shows the truth table of MBE scheme. From table 2 the MBE logic diagram is implemented as shown in Fig. 5. Using the MBE logic and considering other conditions the Boolean expression for one bit partial product generator is given by the equation 1.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Table 2: Truth Table of MBE Scheme.

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	0

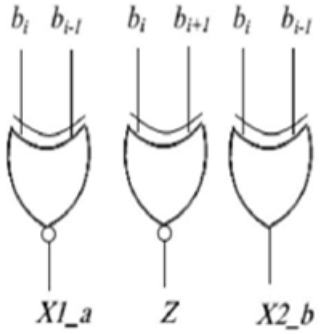


Fig. 5. Logic diagram of MBE.

$$pij = (ai \oplus bi+1+bi-1 \oplus bi)\,(\,ai-1 \oplus bi+1+bi \oplus bi+1+\,bi-1 \oplus$$

Equation(1) is implemented as shown in Fig. 6. 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. 6. The negative partial products are converted into 2's complement by adding a negate (Ni) bit.

An expression for negate bit is given by the Boolean equation 2. This equation is implemented as shown in Fig. 7. The required signed extension to convert 2's complement signed multiplier into both signed-unsigned multiplier is given by the equations 3and 4. For Boolean equations 5 and 6 the corresponding logic diagram is shown in Fig. 8.

$$Ni = bi+1(\overline{bi-1\ bi}) \quad \text{---(2)}$$
$$a8 = s\_u \cdot a7 \quad \text{---(3)}$$
$$b8 = s\_u \cdot b7 \quad \text{---(4)}$$

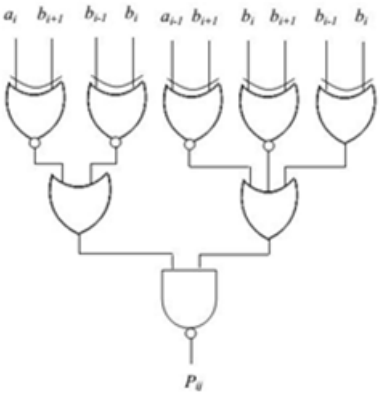


Fig. 6. Logic diagram of 1-bit partial product generator.

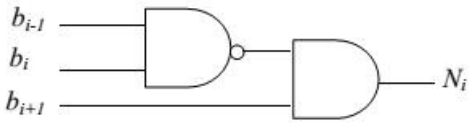


Fig. 7. Logic diagram of negate bit generater.

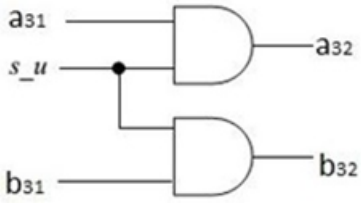


Fig.8. Logic Diagram of Sign Converter

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 Signunsign 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 extended bit of both multiplicand and multiplier should be extended with 0, that is a32= a33= b32= b33= 0. It is required that when the operation is signed multiplication the sign extended bit depends on whether the multiplicand 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 the sign extended bits should be generated are s_u = 1, a31=1,b31= 0, a32= a33 =1, and b32= b33=0. Andwhen the multiplicand operand is positive and multiplier operand is negative the sign extended bits should be generated are s_u = 1, a31=0, ,b31= 1, a32= a33 =0, and b32= b33=1. Table 3 shows the SUMBE multiplier operation.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Table 3: SUMBE operation:

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

Fig. 10. shows the partial products generated by partial product generater circuit which is shown in Fig. 6. There are 17-partial products with sign extension and negate bit Ni. All the 17-partial products are generated in parallel.

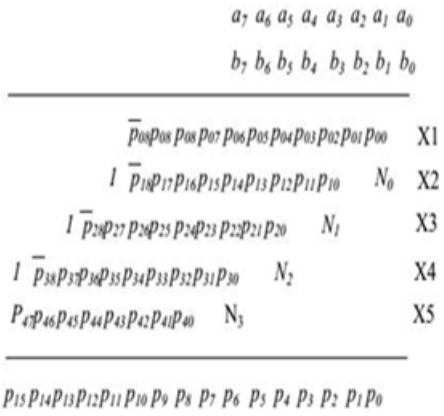
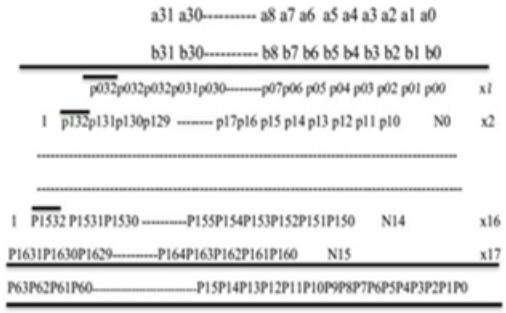


Fig. 9. 8×8 multiplier for signed-unsigned number.



In Fig. 10 there are 17-partial products namely X1, X2, X3, X4, X5,X6,X7,X8,X9,X10,X11,X12,X13,X14,X15,X16 and X17. These partial products are added by the Carry Save Adders (CSA) and the final stage is Carry Look ahead (CLA) adder as shown in Fig. 11. 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 15+16 = 31 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.

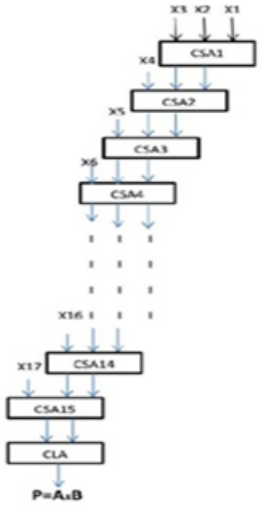


Fig. 11. Partial product adder logic.

IV. SIMULATION RESULTS:

Verilog code is written to generate the required hardware and to produce the partial product, for CSA adder, and CLA adder. After the successful compilation the RTL view generated is shown in Fig.12

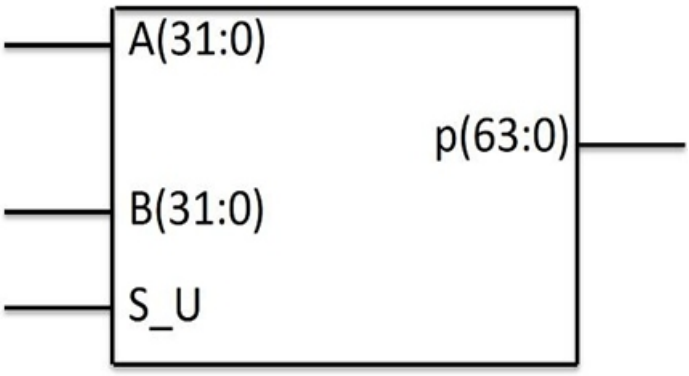
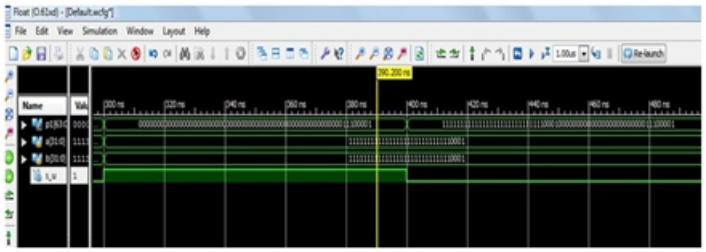


Fig. 12: RTL view of 32×32 signunsigned multiplier

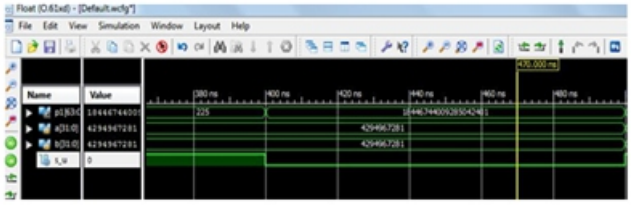
ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Fig. 13 shows the simulation result of signed-unsigned numbers Fig. 13(a) and Fig. 13(b) shows the simulation result of signed- unsigned number in binary and decimal respectively. When s_u = 1, the 32-bit operands are signed and the product of 11111111111111111111111111110001(-15)×11111111111111111111111111110001(-15)= 00000000000000000000000000000000000000000000000000011100001(225). And when the control signal s_u = 0, the 32-bit operands are unsigned and the product of 11111111111111111111111111110001(4294967281)x11111111111111111111111111110001(4294967281)=11111111111111111111111111100010000000000000000000000000011100001(18446744009285042401).

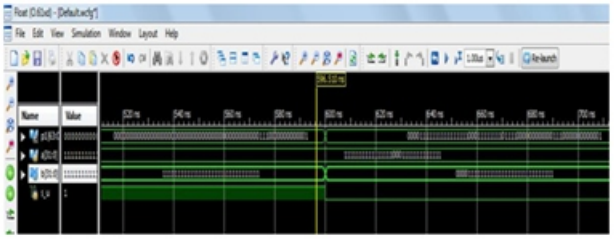
Fig. 13(c) and Fig. 13(d) shows the simulation result of signed- unsigned number in binary and decimal respectively. When s_u = 1, the 32-bit operands are signed and the product of 11111111111111100011111111111111 (-57345) × 11111111111111111111111111111111 (-1) = 00000000000000000000000000000000000000000000000011100000000000001 (57345). And when the control signal s_u = 0, the 32-bit operands are unsigned and the product of 11111111111111000111111111111111(4294909951)×00001111111111111111111111111111 (268435455)=00001111111111111111100011111110111100000000000011100000000000001(11529061068807 12705).



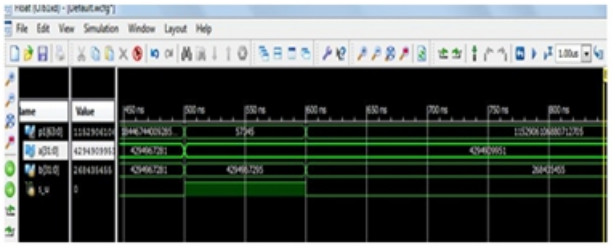
(a)



(b)



(c)



(d)

Fig. 13: Simulation results.

V. CONCLUSION:

In this paper, we present a 32-bit×32-bit advanced multiplier capable of carrying out both signed and unsigned operations. The proposed novel unified signed/unsigned multiplier was optimized in terms of speed, power consumption and silicon area by exploiting more regular partial product array, developing more efficient compression methods and combining several types of fast adders.

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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

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

[6]J. Fadavi-Ardekani, ªM×N Booth Encoded Multiplier Generator Using Optimized Wallace Trees,° IEEE Trans. VLSI Systems, vol. 1,no. 2, June 1993.

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

[8]Kim J. Y., "Multiplier to selectively perform unsigned magnitude multiplication or signed magnitude multiplication",USpatent 5,870,322, Feb 9, 1999.

[9]Hwang-Cherng Chow and I-Chyn Wey, "A 3.3V 1GHz high speed pipelined Booth multiplier," Proc. of IEEE ISCAS, vol.1, pp. 457-460, May 2002.

[10]M. Aguirre-Hernandez and M. Linarse-Aranda, "Energy-efficient high-speed CMOS pipelined multiplier," Proc. of IEEE CCE, pp. 460-464, Nov. 2008.

[11]A. D. Booth, "A signed binary multiplication technique," Quarterly J. Mechanical and Applied Math, vol. 4, pp.236-240, 1951.

[12]Kuang S. R., Wang J. P., Guo C. Y., "Modified Booth Multipliers With a Regular Partial Product Array",IEEETransactions on Circuits and Systems II: Express Briefs,Vol.56, Issue 5, pp.:404 - 408, May, 2009.

[13]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.