Journal of Advances in Mathematics and Computer Science

26(6): 1-9, 2018; Article no.JAMCS.37931
ISSN: 2456-9968
(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

A Novel Multiplier Using Vedic Mathematics and Booth Encoding

B. V. Srividya[1*] and T. Kiran Kumar[1]

[1]Department of Telecommunication, Dayananda Sagar College of Engineering, Bangalore, India.

Authors' contributions

This work was carried out in collaboration between both authors. Author TKK designed the block diagram and performed the analysis. Author BVS designed the algorithm, managed VHDL coding, literature survey and wrote the first draft of the manuscript. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2018/37931
Editor(s):
(1) Qiankun Song, Department of Mathematics, Chongqing Jiaotong University, China.
(2) Dariusz Jacek Jakóbczak, Assistant Professor, Chair of Computer Science and Management in this Department, Technical University of Koszalin, Poland.
(3) Tian-Xiao He, Professor, Department of Mathematics and Computer Science, Illinois Wesleyan University, USA.
Reviewers:
(1) Ufuk Çelik, Bandirma Onyedi Eylul University, Turkey.
(2) Rachmad Vidya Achmad, Bandung Institute of Technology, Indonesia.
(3) Gaurav Purohit, CSIR CEERI, India.
Complete Peer review History: http://www.sciencedomain.org/review-history/23616

Short Research Article

Received: 13th October 2017
Accepted: 20th December 2017
Published: 13th March 2018

Abstract

In this paper a comparative study of multiplier is done for speed. The concept used in the proposed algorithm is the combination of "Urdhva Tiryagbhyam" algorithm and Booth encoding for performing multiplication. The "Urdhva Tiryagbhyam" algorithm is an ancient Indian Vedic mathematics, which is used for multiplication to improve the speed and area [1]. The multipliers based on the concept of Booth encoding are compact and have significantly a higher speed when compared to their non encoded counterparts [2].

The approached architecture for a multiplier uses Booth encoder, Vedic Multiplier, along with parallel adders. The coding for the proposed multiplier is carried out using the hardware descriptive language namely VHDL. Subsequently the code is simulated and synthesized using Xilinx ISE 10.1 software. This multiplier is implemented on Spartan 3 FPGA devices XC3S50- 5pq208. The performance metrics for comparing the Booth encoded multiplier and the Vedic multiplier is the speed of operation and the device utilization summary, when both the algorithms are implemented on FPGA. It has been observed that the proposed design has speed improvements as compared to the other multipliers.

**Corresponding author: E-mail: srividyabv@gmail.com;*

Keywords: Vedic multiplier; booth encoding; multiplier; multiplication; VHDL; FPGA; synthesis.

1 Introduction

Many digital signals processing operation requires several multiplication and for the same, we need very fast multiplier for a wide range of requirements for hardware and speed [3]. Vedic mathematics was rediscovered in the early twentieth century from ancient Indian sculptures (Vedas).The Multiplier Architecture is based on the Vertical and Crosswise algorithm of ancient Indian Vedic Mathematics. Vedic mathematics simplifies and optimizes the conventional mathematical algorithm [4].

For serial multiplication, Booth encoding based multipliers was proposed. In this method, the partial products are generated by choosing a pair of digits along with the most significant digit from the preceding pair. The multiplier is encoded using radix-2 or radix-4 Booth algorithm [5]. The advantage of Booth encoding is that it generates only half of the partial products in comparison to the other multipliers. However the benefit comes from speed hardware trade off.

We have developed a new architecture using "Urdhva Tiryakbhyam" Sutra and Booth encoding. The final parallel adders implemented in the proposed architecture reduce propagation delay significantly. It is believed that our architecture may set new path for future research.

An organization of this paper is as: section 2 describes the Vedic multiplier using "Urdhva Tiryagbhyam" sutra. Section 3 describes the multiplier using Booth Encoding algorithm. Section 4 shows proposed architecture using carry save adders for 4 bit multipliers. Section 5 describes results, comparison with the Booth encoded multiplier and Vedic multiplier. Conclusion and references are given in section 6 and 7 respectively.

2 Vedic Multiplier Using 'Urdhva Tiryagbhyam' Sutra

The "Urdhva Tiryagbhyam" Sutra [1] is a general multiplication formula applicable to all cases multiplication. "Urdhva" and "Tiryagbhyam" words are derived from Sanskrit literature. "Urdhva" means "Vertically" and "Tiryagbhyam" means crosswise.

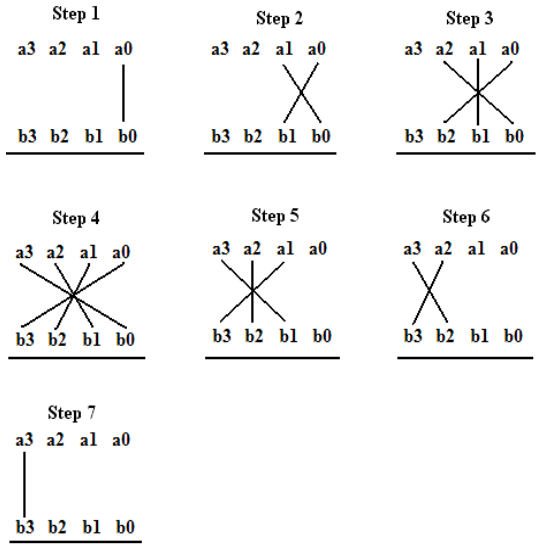

Fig. 1. Line Diagram for multiplication of two 4-bit numbers

To illustrate the multiplication algorithm for binary let us consider a multiplication of two binary numbers a3a2a1a0 and b3b2b1b0. The parallelism in generation of partial products and their summation is explained in Fig. 1.

3 Booth Encoding

In array multipliers, the computation of the partial products is based on considering one bit of the multiplier at a time. An n-bit by n-bit array multiplier requires 'n' half adders and 'n (n-2)' full adders. Hence it can be observed that the number of adders have increased quadratically with an increase in the size of the multiplier.

Booth encoding method was proposed to expedite the serial multiplication. In this method, the Partial products are generated by considering a pair of digits along with the most significant digit from the preceding pair. The multiplier represented as (X) is decoded into distinct select lines and driven across the row, where partial products are generated. These select lines control Booth selectors that choose the appropriate multiple of Y for each partial product. In case of negative numbers, the negative partial products are sign extended in order to be summed precisely. The Booth encoded multipliers are compact and faster compared to the other popular multipliers [2].

X is the multiplicand and Y is the multiplier. Every Addition/Subtraction/NOP is followed by right shift by 2 bits.

Table 1. Radix-4 booth encoding

Multiplier (Y)	Encoding	Action to be taken
000	0	NOP
001	+1	Add X
010	+1	Add X
011	+2	Add X twice
100	-2	Add 2s complement of X two times
101	-1	Add 1s complement of X
110	-1	Add 1s complement of X
111	0	NOP

The motivation for this work was obtained by making a survey of fast multipliers that could be used in encryption algorithms. In the survey, many authors have compared the Booth Multiplier with the Vedic Multiplier. It has been observed from the Survey that the delay of Vedic multiplier was 20% reduced in comparison to the Booth Encoded Multiplier.

Also in an NXN Vedic multiplier, 'n' bits of the multiplier are considered for generating the partial product. But since, Parallelism is involved in the generation of the partial products. Vedic multiplier is found to have a lower delay [6].

Whereas in an NXN Booth encoded multiplier, only 'n/2' bits of the multiplier are considered for generating the partial product. But the partial products are generated serially.

In the proposed algorithm, the multiplier is encoded using Booth encoding, in order to reduce the number of multiplier bits to 'n/2'. In addition to this, Parallelism is also involved in the generation of the partial products. Hence the benefits of the Vedic multiplier and that of the Booth encoder are combined together in order to further reduce the delay.

4 Proposed Architecture

Fig. 2 shows the proposed architecture for multiplication of N bit binary number using Booth encoding and Vedic mathematics.

In the proposed architecture the N bit multiplier is encoded into a N/2 bit multiplier using radix 4 Booth encoding. To illustrate with an example, let us consider 4 bit multiplication:

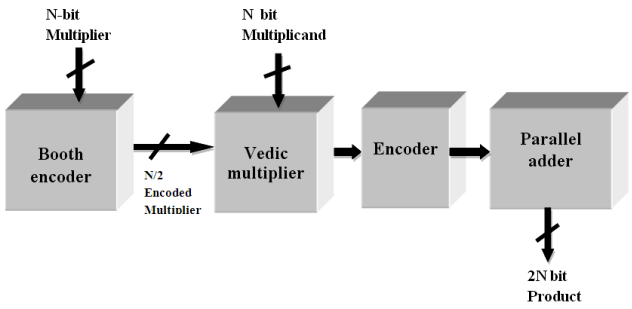

Fig. 2. Proposed Architecture

The multiplicand X is represented as X3 X2 X1 X0 and the multiplier Y is represented Y3 Y2 Y1 Y0. By applying the Booth encoding technique, the multiplier Y gets encoded as YN1 and YN0. With the new encoded multiplier all the partial products are computed parallelly using the Vedic multiplier. After the application of "Urdhva Tiryakbhyam" Sutra and Booth encoding the partial products obtained are:

P0 = X0 YN0
P1 = X1 YN0
P2 = X2 YN0 + X0 YN1
P3 = X3 YN0 + X1 YN1
P4 = X2 YN1
P5 = X3 YN1

The encoding of the partial products after the application of both the algorithms is discussed in Table 2.

Table 2. Encoded values

Partial Products (P5 to P0)	Encoded Values (PE)
0	0000
1	0001
2	0010
3	0011
4	0100
-4	1100
-3	1101
-2	1110
-1	1111

The encoded values for each of the partial products
P5 to P0 are PE5 PE4 PE3 PE2 PE1 PE0.

Figs. 3 and 4 shows the final product obtained after the encoded values are shifted and fed to the Parallel adder.

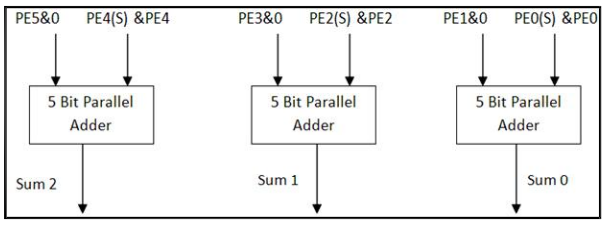

Fig. 3. Parallel addition of encoded values

From Fig. 3, it is shown that PE0, PE2 and PE4 are sign extended by one bit. Where as PE1, PE3 and PE5 are left shifted by one bit position before applying to the 5 bit parallel adder.

Fig. 4 shows the final product obtained from the 8bit parallel adder. The 8bit parallel adder takes 3 bit sign extension of Sum0 along with twice left shifted and 1 bit sign extension of Sum1.The second stage parallel adder gets its input from the first stage sum and also four times left shifted Sum2.

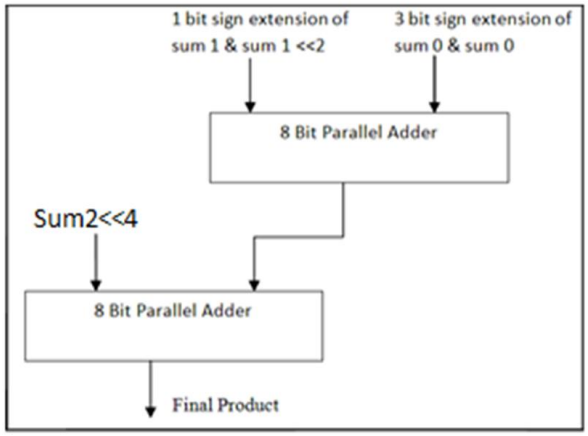

Fig. 4. Addition for final product

This proposed architecture reduces the delay considerably compared to both Booth encoding algorithm [7] and Vedic Mathematics algorithm [8]. The carries generated by the individual parallel adders are not passed on to the next stage of adders.

To illustrate with an example: Let the first operand, the multiplicand be $X=x3x2x1x0=(0101)_2=(5)_{10}$; The second operand, the multiplier be chosen as $Y=y3y2y1y0=(1010)_2=(-6)_{10}$;

According to the proposed algorithms, the multiplier is encoded using Booth encoder as Yn0 and Yn1. Yn0 and Yn1 are 3-bits in size.

Yn0=y1, y0, 0 i.e zero is the least significant bit.

Yn1=y3, y2, y1.

According to the illustration $Yn0= (100)_2$. Yn0 is encoded as (-2) according to Table 1. Similarly, $Yn1= (101)_2$. Yn1 is encoded as (-1) according to Table 1.

The encoded multiplier (Yn1, Yn0) and the multiplicand (X3, X2, X1, X0) are used to compute the partial products parallelly using the concept of Vedic multiplication.

P0 = X0 YN0=1*(-2)=(-2)=$(1110)_2$
P1 = X1 YN0=0*(-2)=0=$(0000)_2$
P2 = X2 YN0 + X0 YN1=(1*-2)+(1*(-1))=(-2)+(-1)=$(1101)_2$
P3 = X3 YN0 + X1 YN1=0*(-2)+0*(-1)=0=$(0000)_2$
P4 = X2 YN1=1*(-1)=$(1111)_2$
P5 = X3 YN1=0*(-1)=(0)=$(0000)_2$

From Fig. 3, it is shown that PE0, PE2 and PE4 are sign extended by one bit. Whereas PE1, PE3 and PE5 are left shifted by one bit position before applying to the 5 bit parallel adder.

Sum0= Sign extended value of PE0+ Left shifted value of PE1= $(11110)_2$+$(00000)_2$=$(11110)_2$
Sum1= Sign extended value of PE2+ Left shifted value of PE3= $(11101)_2$+$(00000)_2$=$(11101)_2$
Sum2= Sign extended value of PE4+ Left shifted value of PE3= $(11111)_2$+$(00000)_2$=$(11111)_2$

From Fig. 4, the 8bit parallel adder takes 3 bit sign extension of Sum0 along with twice left shifted and 1 bit sign extension of Sum1.The second stage parallel adder gets its input from the first stage sum and also four times left shifted Sum2.

Final Product is obtained in two stages: Stage a result + Stage b result

Stage a result: 3 bit sign extension of Sum0 along with twice left shifted and 1 bit sign extension of Sum1

i.e. $(11111110)_2$+ $(11110100)_2$= $(11110010)_2$

Stage b result: Stage a result + four times left shifted value of Sum2.

i.e. $(11110010)_2$+ $(11110000)_2$=$(11100010)_2$=$(-30)_{10}$

Since, the most significant bit is 1; the final product is in the 2's complement form.

This yields the final product to be $(00011110)_2$= $(30)_{10}$

5 Results and Discussion

Table 3 shows comparison of the proposed architecture with Vedic Mathematics Multiplier [8] and Booth Encoding Multiplier [7].

Table 3. Synthesis report

Device Spartan2 XC3S50: 5pq208	**4 X 4 Multiplier Using Vedic Mathematics**	**4 X 4 Multiplier Using Booth Encoding**	**4 X 4 Multiplier Using proposed Architecture**
Delay	17.754 ns	22.209 ns	13.3 ns
Number of Slices	18/768	27/768	38/768
Number of LUTs	33/1536	33/1536	69/1536
Levels of Logic	9	14	7

The proposed architecture for a N X N multiplier for a value of N = 4 is found to have a delay of 13.3 ns, with 8.168 ns for the logic and 5.132 ns for routing. The speed is optimized compared to Vedic Mathematics multiplier [8], which has 10.134ns for logic and 7.632 ns for routing and Booth Encoding Multiplier [7], which has 12.313 ns for logic and 9.896ns for routing, giving a delay reduction of 25% and 40 % respectively.

5.1 Simulation result

The N x N multiplier, for N= 4, 8 is designed in VHDL and its functionality is being verified for all the possible input combination by writing test bench.

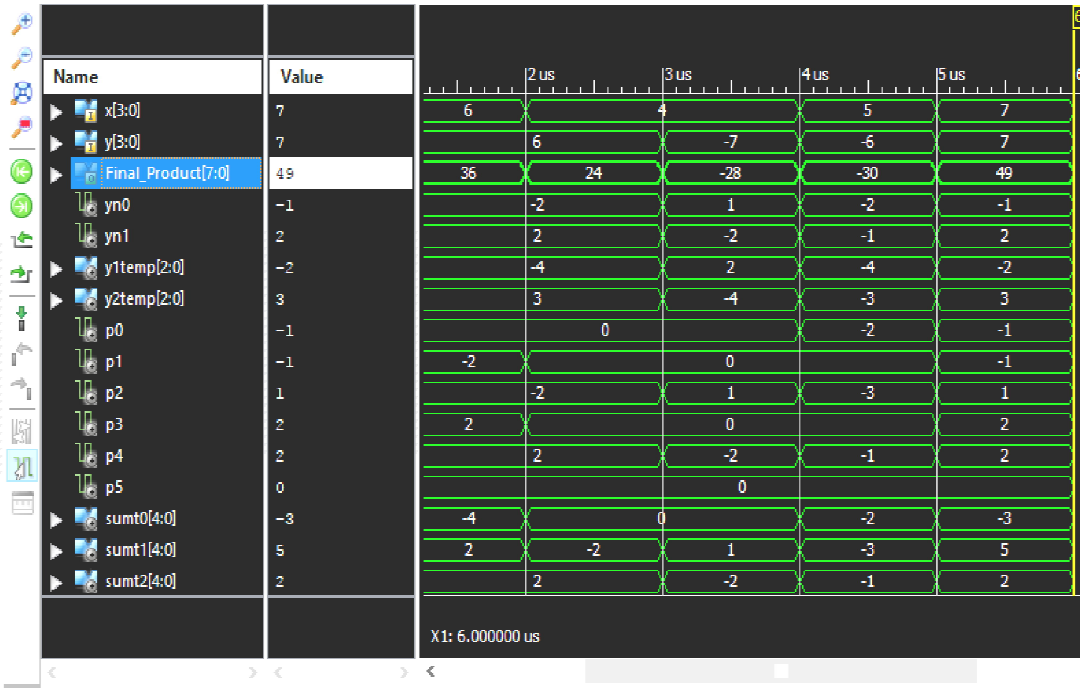

Fig. 5. Simulation result for 4 bit proposed multiplier

Fig. 5 shows 4 different values of the operands X and Y.

The first being X= $(0101)_2$= $(5)_{10}$, Y=$(1010)_2$=$(-6)_{10}$. The operand Y is encoded according to Table 1 as YN0=-2 and YN1=-1. The six partial products generated parallelly are PE0, PE1, PE2, PE3, PE4 and PE5. The final product is $(-30)_{10}$.

The second set of operands are X= $(0111)_2$= $(7)_{10}$, Y= $(0111)_2$= $(7)_{10}$. This generates the final product as 49. The operand Y is encoded as 2 and -1. The partial products are generated using the Vedic multiplier.

The Proposed algorithm is extended to perform multiplication on 8-bit operands. The Simulation results are shown in Fig. 6.

Fig. 6 shows the ModelSim results for 8 bit proposed multiplier for X = $(11111111)_2$=$(255)_{10}$, Y =($01111111)_2$=$(127)_{10}$, yielding a product of 32385 to base 10. The multiplier Y is encoded as 2, 0, 0,-1 using Booth encoding. The Partial products from PE0 to PE15 (PE14 and PE15 are missing in the figure, due to lack of clarity) are obtained using Vedic multiplier.

In another example X= $(01111111)_2$= $(127)_{10}$, and Y=$(11111111)_2$=$(-1)_{10}$ the product is $(-127)_{10}$. In Figs. 5 and 6 each PEn is an encoded value. The proposed architecture assumes X to be an unsigned number and Y to be a signed number.

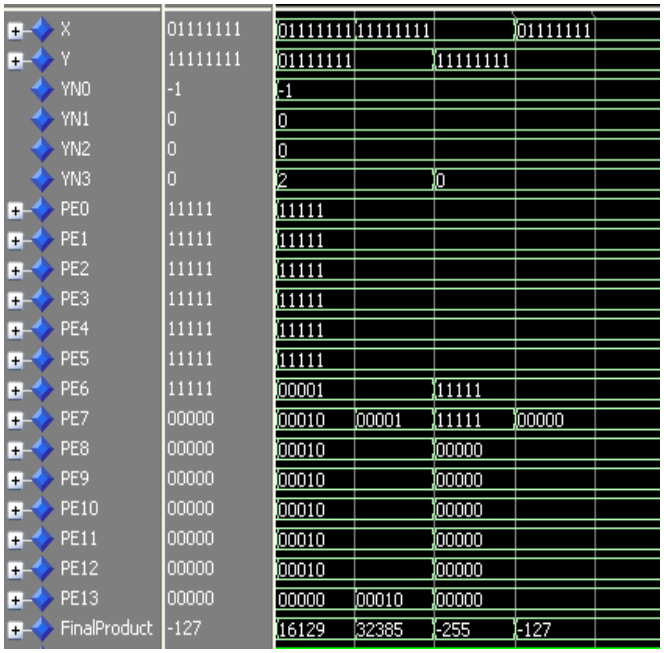

Fig. 6. Simulation Result for Proposed 8-bit multiplier

6 Conclusions

The proposed multiplier architecture which combines the benefits of Booth encoding and the benefits of Vedic multiplier shows speed improvements over multiplier architecture presented in Vedic mathematics [8] and Booth Encoding [7] individually. The N x N multiplier using the proposed architecture has resulted in delay reduction of 25% compared to Vedic Mathematics and 40% reduction in delay compared to Booth Encoded multiplier. It can be well suited for multiplication of numbers with more than 16 bit size.

This work can be further extended for area optimization and also usage of carry save adders can further reduce the delay.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Honey Durga Tiwari, Ganzorig Gankhuyag, Chan Mo Kim, Yong Beom Cho. Multiplier design based on Ancient Indian Vedic Mathematics. International SoC, Design Conference, Nov. 2008;65-68.

[2] David Villeger, Vojin G. Oklobdzija. Evalution of Booth Encoding Techniques for Parallel Multiplier Implementation.

[3] Kapse, Pooja R, Komal M. Review on a compressor design and implementation of multiplier using Vedic mathematics. International Journal of Advanced Research in Computer and Communication Engineering. 2017;6(2):86-90.

[4] Prabha S. Kasliwal, Patil BP, Gautam DK. Performance evaluation of squaring operation by Vedic mathematics. IETE Journal of Research. 2011;57(1):39-41.

[5] Jagadguru Swami Sri Bharati Krisna Tirthaji Maharaja. Vedic mathematics. Motilal Banarsidass Publishers Pvt. Ltd, Delhi; 2009.

[6] Sushma R. Huddar, Sudhir Rao, Kalpana M, Surabhi Mohan. Novel high speed Vedic mathematics multiplier using compressors. 978-1-4673-5090-7/13/$31.00 ©2013 IEEE, Pp No. 465-469.

[7] Booth AD. A signed binary multiplication technique. Qarterly J. Mechan. Appl. Math. 1951;IV.

[8] Shamim Akhter. VHDL implementation of fast NxN multiplier based on Vedic mathematic. Circuit theory and design 2007, ECCTD 2007, 18th European Conference, Issue date 27-30 Aug. 2007;472-475.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://www.sciencedomain.org/review-history/23616