

A Novel Multiplier using Vedic Mathematics and Booth Encoding

Srividya B V¹, T Kiran Kumar²

^{1,2}Assistant Professor, Dept of Telecommunication, Dayananda Sagar College Engineering, Bangalore, India

SAMPLE ABSTRACT:

In this paper a comparative study of multiplier is done for speed. The concept used is combination of “UrdhvaTiryagbhyam” algorithm and Booth encoding for multiplication. The “UrdhvaTiryagbhyam” algorithm is an ancient Indian Vedic mathematics, which is used for multiplication to improve the speed and area [1]. The Booth encoded multipliers are smaller and faster compared to their non encoded counterparts [6].

The approached architecture for a multiplier uses Booth encoder, Vedic Multiplier, along with parallel adders. The proposed multiplier is coded in VHDL, synthesized and simulated using Xilinx ISE 10.1 software. This multiplier is implemented on Spartan 3 FPGA devices XC3S50- 5pq208. The performance evaluation results in terms of speed and device utilization are compared with Vedic multiplier and Booth Encoded multiplier. The proposed design has speed improvements as compared to the other multipliers.

Keywords: Vedic Multiplier, Booth Encoding, Multiplier, Multiplication, VHDL, FPGA, Synthesis

1. INTRODUCTION

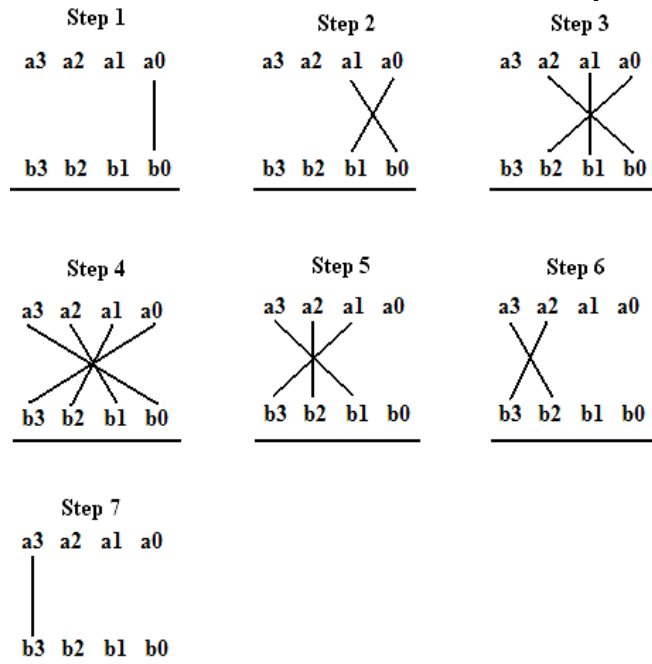
Many digital signals processing operation requires several multiplication and for the same, we need very fast multiplier for a wide range of requirements for hardware and speed. Vedic mathematics was rediscovered in the early twentieth century from ancient Indian sculptures (Vedas). The Multiplier Architecture is based on the Vertical and Crosswise algorithm of ancient Indian Vedic Mathematics. Vedic mathematics simplifies and optimizes the conventional mathematical algorithm. Booth encoding was proposed to accelerate serial multiplication. Partial products are chosen by considering a pair of digits along with the most significant digit from the previous pair. The multiplier is encoded using radix-2 or radix-4 Booth algorithm [2]. The advantage of this encoding is that it generates only half of the partial products compared to the multiplier which does not. However the benefit comes from speed hardware trade off.

We have developed a new architecture using “Urdhva Tiryagbhyam” 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, and finally conclusion and references are given in section 6 and 7 respectively.

2. VEDIC MULTIPLIER USING ‘URDHVA TIRYAGBHYAM’ SUTRA.

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



58
59

60 **Figure 1: Line Diagram for multiplication of two 4-bit numbers**

61
62
63
64
65
66

To illustrate the multiplication algorithm for binary let us consider a multiplication of two binary numbers $a_3a_2a_1a_0$ and $b_3b_2b_1b_0$. The parallelism in generation of partial products and their summation is explained in Figure 1.

67
68
69

3. BOOTH ENCODING

70
71
72
73

In the array multipliers, the partial products are computed by observing one bit of the multiplier at a time. In general, a n -digit by n -digit array multiplier would require $n(n-2)$ full adders and n half adders. So, the number of components required increases quadratically.

74
75
76
77
78
79
80

Booth encoding was proposed to accelerate serial multiplication. Partial products are chosen by considering a pair of digits along with the most significant digit from the previous pair. The multiplier (X) is decoded into several select lines and driven across partial product row. The select lines control Booth selectors that choose the appropriate multiple of Y for each partial product. The negative partial products must be sign extended to be summed correctly. The Booth encoded multipliers are smaller and faster compared to their non encoded counterparts [6].

81
82

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

83
84
85
86
87
88
89

TABLE 1: Radix-4 Booth Encoding.

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

The motivation for this work was obtained by comparing the Booth Multiplier with Vedic Multiplier. The delay of Vedic multiplier was 20% reduced compare to Booth Encoded Multiplier. Since Vedic multiplier is faster [4][7], the idea of combining both the methods should further reduce the delay.

4. PROPOSED ARCHITECTURE

Figure 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:

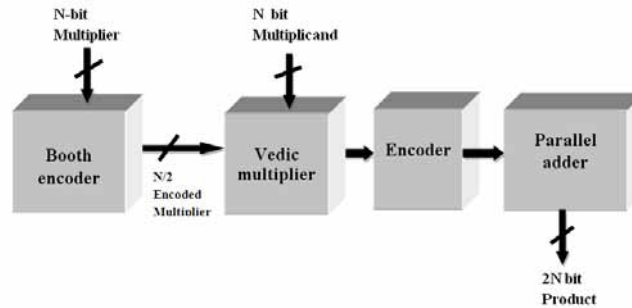


Figure-2: Proposed Architecture

The multiplicand is $X_3 X_2 X_1 X_0$ and the multiplier is $Y_3 Y_2 Y_1 Y_0$. The multiplier gets encoded as Y_{N1} and Y_{N0} . With the new encoded multiplier all the partial products are computed parallelly. After the application of "Urdhva Tiryakbhyam" Sutra and Booth encoding the partial products obtained are

$$\begin{aligned}
 P_0 &= X_0 Y_{N0} \\
 P_1 &= X_1 Y_{N0} \\
 P_2 &= X_2 Y_{N0} + X_0 Y_{N1} \\
 P_3 &= X_3 Y_{N0} + X_1 Y_{N1} \\
 P_4 &= X_2 Y_{N1} \\
 P_5 &= X_3 Y_{N1}
 \end{aligned}$$

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

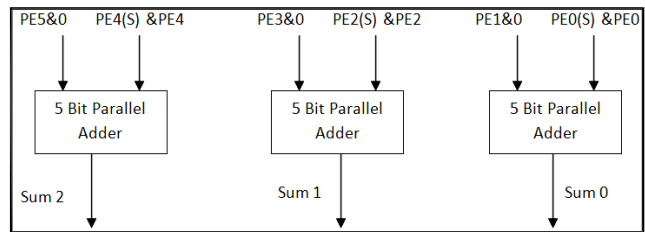


Figure-3 : Parallel addition of encoded values.

From Figure 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. Figure-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 thrice left shifted Sum2.

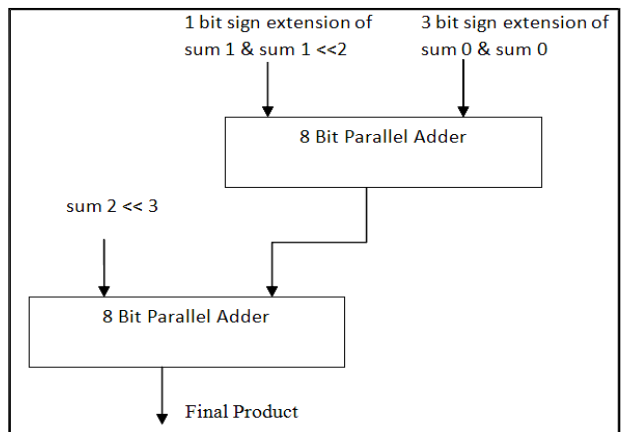


Figure-4: Addition for Final Product

This proposed architecture reduces the delay considerably compared to both Booth encoding algorithm [5] and Vedic Mathematics algorithm [3]. The carries generated by the individual parallel adders are not passed on to the next stage of adders.

5. RESULTS AND DISCUSSION

Table-3 shows comparison of the proposed architecture with Vedic Mathematics Multiplier [3] and Booth Encoding Multiplier [5].

TABLE-3: Synthesis Report

Device Spartan2 XC3S50: 5pq208	4 X 4 Multiplier Using Vedic Mathematic s	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[3], which has 10.134ns for logic and 7.632 ns for routing and Booth Encoding Multiplier[5], which has 12.313 ns for logic and 9.896ns for routing, giving a delay reduction of 25% and 40 % respectively.

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.

X	0101	0111	1010	0101
Y	1010	0111	1010	
YN0	-2	-1	-2	
YN1	-1	2	-1	
PE0	1110	1111	0000	1110
PE1	0000	1111	1110	0000
PE2	1101	0001	0000	1101
PE3	0000	0010	0001	1101
PE4	1111	0010	0000	1111
PE5	0000	0000	0010	1111
FINALPRODUCT	-30	49	70	-60

Figure 5. Simulation result for 4 bit proposed multiplier

X	01111111	01111111	11111111	01111111
Y	11111111	01111111	11111111	
YN0	-1	-1		
YN1	0	0		
YN2	0	0		
YN3	0	2	0	
PE0	11111	11111		
PE1	11111	11111		
PE2	11111	11111		
PE3	11111	11111		
PE4	11111	11111		
PE5	11111	11111		
PE6	11111	00001	11111	
PE7	00000	00010	00001	11111
PE8	00000	00010	00000	00000
PE9	00000	00010	00000	
PE10	00000	00010	00000	
PE11	00000	00010	00000	
PE12	00000	00010	00000	
PE13	00000	00000	00010	00000
FinalProduct	-127	16129	32385	-255

Figure 6: Simulation Result for Proposed 8-bit multiplier

Figure 5 shows the ModelSim results for 4 bit proposed multiplier for X = 0111, Y = 0111 yielding a product of 49 10. Similarly for X= Y = 1111, the product is -15

Figure 6 shows the ModelSim results for 8 bit proposed multiplier for X = 11111111, Y = 01111111 yielding a product of 32385 to base 10. Similarly for X= 01111111, and Y = 11111111 the product is -127. In Figures 5 and 6 PE_n each is an encoded value. The proposed architecture assumes X to be an unsigned number and Y to be a signed number.

6. CONCLUSIONS

The proposed multiplier architecture shows speed improvements over multiplier architecture presented in Vedic mathematics [3] and Booth Encoding [5]. 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.

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, pp.65-68.

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

229
 230 [3] Shamim Akhter " VHDL implementation of fast NxN multiplier based on Vedic
 231 mathematic" , Circuit theory and design 2007, ECCTD 2007, 18th European Conference,
 232 Issue date 27-30 Aug. 2007, pg 472-475.
 233
 234 [4] Prabha S Kasliwal, BP Patil, DK Gautam, "Performance Evaluation of Squaring
 235 Operation by Vedic Mathematics", IETE Journal of research, Year 2011, Volume 57, Issue
 236 1, Pg 39-41.
 237
 238 [5] A.D. Booth, " A signed Binary Multiplication technique" Qarterly J. Mechan. Appl. Math.,
 239 Vol IV, 1951
 240
 241 [6] David Villeger, Vojin G Oklobdzija, "Evalution of Booth Encoding Techniques for
 242 Parallel Multiplier Implementation.
 243
 244 [7] Sushma R Huddar, Sudhir Rao, Kalpana M, Surabhi Mohan, " Novel high speed vedic
 245 mathematics multiplier using compressors", 978-1-4673-5090-7/13/\$31.00 ©2013 IEEE, Pp No.
 246 465-469
 247
 248 [8] Kapse, Pooja R, Komal M, "Review on a compressor design and implementation of
 249 multiplier using Vedic mathematics" International Journal of advanced research in computer
 250 and communication Engineering, Vol 6, Issue 2, February 2017, Pp No. 86-90
 251
 252

