**IJCTA**, 9(25), 2016, pp. 423-431 ISSN: 0974-5572
© International Science Press

# FPGA Implementation of Complex Multiplier Based on Signed Vedic Multiplier

**Paldurai K.* and T. Saminathan****

**ABSTRACT**

To implement the hardware module of Fast Fourier Transform (FFT), Discrete Fourier Transformation (DFT), Discrete Cosine Transformation (DCT), Discrete Sine Transformation (DST) and modem broadband communications, large numbers of complex multipliers are required. This paper describes the FPGA implementation of complex multiplier based on signed Vedic Multiplier.it multiplies signed numbers in 2's compliment form and produces the result in 2's complement form. So complex multiplier based on Signed Vedic Multiplier produces output for both signed and unsigned input values. The proposed complex multiplier is compared with complex multiplier based on radix-2 Booth Multiplier. Both the complex multipliers are synthesized and simulated using ISE simulator. It is implemented on the iWave Systems Unified Learning Kit (ULK) which is Spartan 6 family xc6slx25t-2fgg484 FPGA. The Area and Maximum Combinational path delay of both the complex multipliers are compared.

***Keywords:*** Urdhva Tiryakbhyam, radix-2, 2s complement, signed binary numbers, Gauss's Algorithm

## 1. INTRODUCTION

In Digital Signal Processing (DSP) complex multiplication plays an essential role. It is used in twiddle factor multiplication to find the coefficients for DIT-FFT. A complex number multiplication consists of four real number multiplication, one addition and one subtraction for unsigned numbers. So it needs four multipliers and two adders [1]. Here a complex multiplier is proposed for signed numbers by utilizing the same four multipliers and two adders. It gives the output for both signed and unsigned numbers.

In order to represent signed binary numbers we have five methods: Signed Magnitude Representation, One's complement, Two's complement, Excess-k and Base-2. Among these Two's complement is the best way to represent signed numbers. Because, Signed magnitude has multiple representations for zero (00000000 (+0) and 10000000 ("0)) and one's complement has multiple representation for zero (00000000 (+0) and 11111111 ("0)) and it is necessary to do an end-around carry i.e. add any resulting carry back into the resulting sum.). In order to overcome these two problems we move to two's complement method, because it has no multiple representations for zeros and no needs to add end-around carry.

Rajashri K. Bhongade [1] proposed a complex multiplier for multiplication of positive complex numbers based on Booth and Vedic Multiplier. But our proposed complex multiplier works for both positive and negative complex numbers. Rupa A. Tomaskar [2] also proposed a complex multiplier for unsigned complex numbers only. Archana Fande [5] proposed a signed complex multiplier for Radix_4 FFT algorithm and compares the performance with Radix_2 FFT algorithm. M.Nagaraju [7] designed a unsigned complex multiplier based on Nikkhalam Sutra whereas in this proposal Urdhva Tiryakbhyam is being used. A block diagram for unsigned complex multiplier has been proposed by Rajashri Bhongade [10] based on Gauss's Algorithm, this block diagram is modified for signed complex multiplication in this proposal.

* Assistant Professor Department of ECE SRM University, Kattankulathur, Chennai, *Email: Paldurai.k@ktr.srmuniv.ac.in*

** Assistant Professor Department of ECE SRM University, Kattankulathur, Chennai, *Email: Saminathan.t@ktr.srmuniv.ac.in*

## 2. BOOTH'S MULTILPLICATION ALGORITHM

Two's complement method is the best way to represent signed numbers. So Booth's multiplication algorithm multiplies two signed bit numbers in two's complement notation. It calculates the signed output by repeatedly adding two values named A and S to a product P, then perform Arithmetic Right Shift (ARS) on P.

### 2.1. Steps Involved in Booth Multiplication

Take two numbers M and R, where M is the multiplier and R is the multiplicand. Let x and y be the total number of bits in M and R respectively.

1. Determine the values of A and S, and the initial value of P. All of these numbers should have a length equal to (x + y + 1).
   - i. A: To find A Fill the most significant x bits with the value of M and fill the remaining (y + 1) bits with zeros.
   - ii. S: Take 2's complement of M, which gives -M. To Calculate S fill the most significant x bits with the value of -M then fill the remaining (y + 1) bits with zeros.
   - iii. P: To find P fill the most significant x bits with zeros. To the right of this, append the value of R then fill the least significant bit with a zero.
2. Then determine the two least significant bits of P and perform the following steps.
   - i. If they are 01, calculate the value of P + A. Ignore Most Significant Bit (MSB) in case of overflow.
   - ii. If they are 10, calculate the value of P + S. Ignore Most Significant Bit (MSB) in case of overflow.
   - iii. If they are 00, no operations required. Use P directly in the next step.
   - iv. If they are 11, no operations required. Use P directly in the next step.
3. Perform Arithmetic Right Shift to value obtained in the 2nd step. Consider this is the new P value.
4. Take the new P value and repeat steps 2 and 3 until they have been done y times.
5. Drop the least significant (rightmost) bit from P. This is the product of M and R.

Find 6 × (–7), with M = 6 and R = –7, and $x = 4$ and $y = 4$:

- M = 0110, -M = 1010, R = 1001
- A = 0110 0000 0
- S = 1010 0000 0
- P = 0000 1001 0

Do the necessary steps:

a. P = 0000 1001 0. The last two bits are 10.
   - i. P=1010 1001 0. P = P+S.
   - ii. P = 1101 0100 1. Arithmetic Right Shift.

b. P = 1101 0100 1. The last two bits are 01.
   - i. P = 0011 0100 1. P = P+A.
   - ii. P = 0001 1010 0. Arithmetic Right Shift.

c. P = 0001 1010 0. The last two bits are 00.
   - i. P = 0000 1101 0. Arithmetic Right Shift.

d. P = 0000 1101 0. The last two bits are 10.
   - i. P=1010 1101 0. P = P+S.
   - ii. P = 1101 0110 1 Arithmetic Right Shift.

e. The product is 11010110 which is -42.

## 3. VEDIC MULTIPLIER

Vedic mathematics is mainly based on 16 sutras.VM is based on the 14th sutra which is Urdhva Tiryakbhyam (Vertically and Crosswise). 2 × 2 Multiplier is the basic building block of Vedic Multiplier. It multiply two 2-bit binary number and produces 4-bit number as a result. It is shown in Fig 3. it uses 4 AND gates and two Half Adders (HA).

### 3.1. 8 × 8 Vedic Multiplier

The architecture for the 8 × 8 unsigned VM is shown in Fig 3.For 8-bit Multiplier design, first the basic block 2 × 2 bit multiplier is designed (Fig 2), then a 4 × 4 block is designed using this 2 × 2 block, then a 8 × 8 block is designed using this 4 × 4 block and three 4 Bit Ripple Carry Adder as shown in below figure.

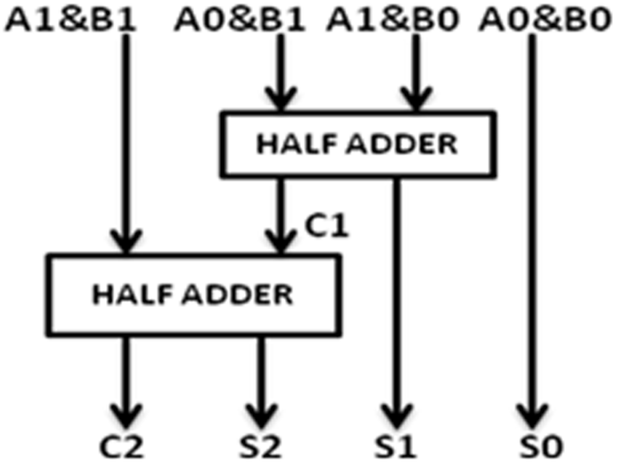


Figure 1: 2 × 2 Multiply Block

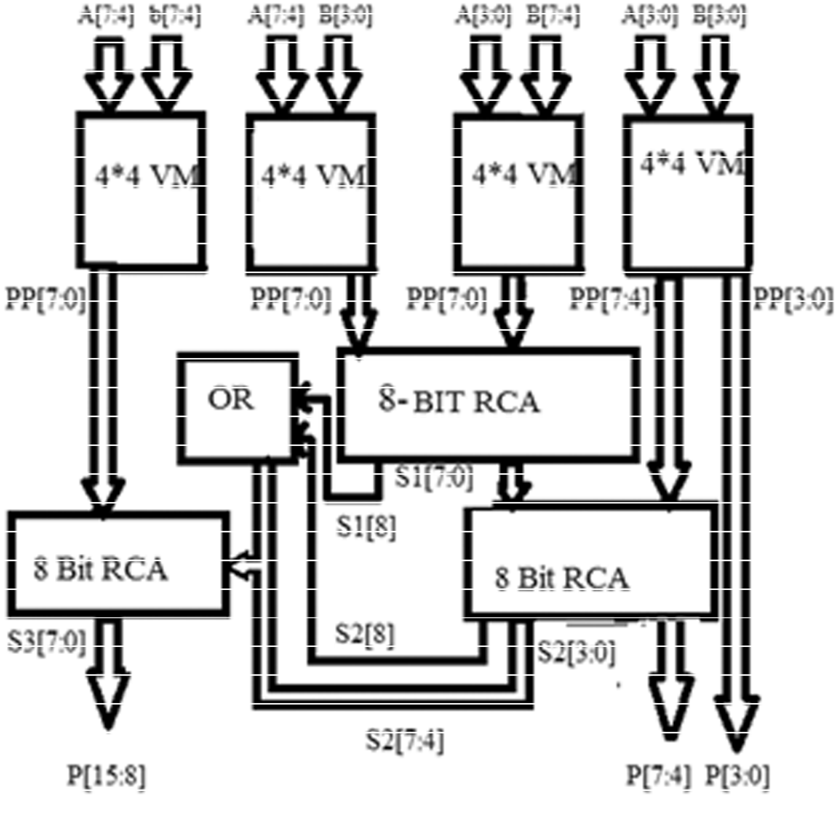


Figure 2: 8 × 8 unsigned VM

### 3.2. 8-Bit Ripple Carry Adder (RCA)

The architecture for 8-bit RCA is shown in Fig 4. It adds two 8-bit binary numbers and produces 9 bit binary number as a result. So it consists of 1 Half Adder (HA) and8 Full Adders (FA).

## 4. FLOWGRAPH FOR SIGNED MULTIPLIER

Find 15 × (“12):

* a = 01111; b = 10100;

- MSB of a = 0, so consider the next four bits as it as.
- MSB of b = 1, so take 2s compliment of next 4 bits, it is 1100.
- Give the above 4 bits to a unsigned VM and get the output as 10110100.

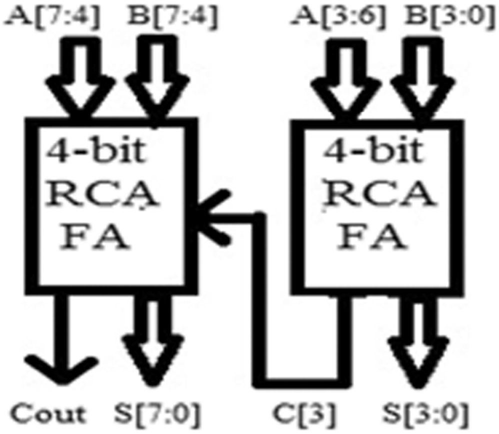


**Figure 3: 8 bit Ripple Carry Adder (RCA)**

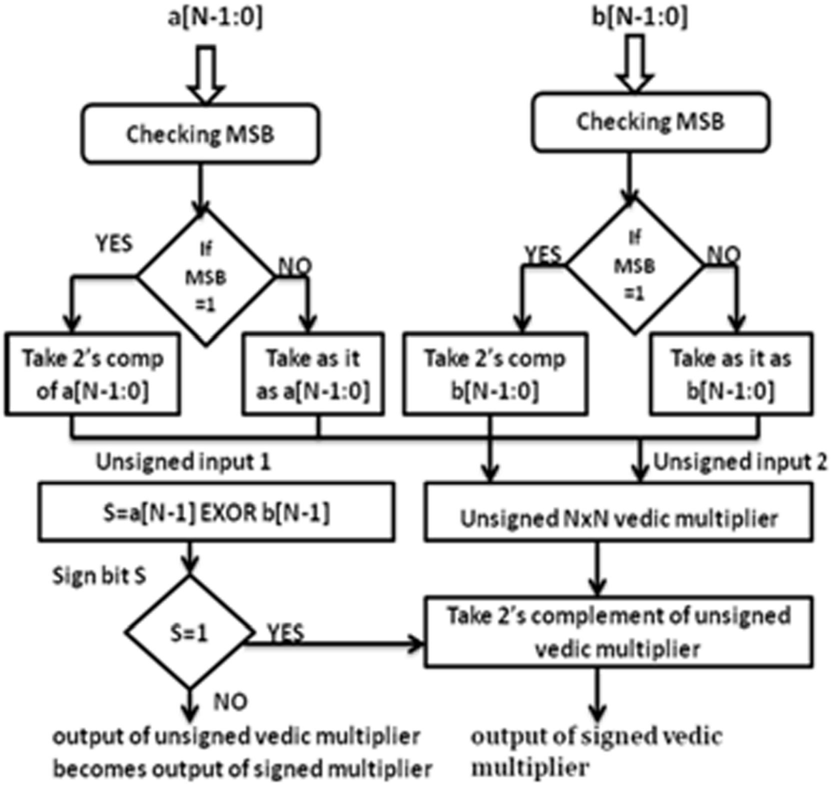

a[N-1:0]
b[N-1:0]
Checking MSB
Checking MSB
YES
If MSB =1
NO
YES
If MSB =1
NO
Take 2's comp of a[N-1:0]
Take as it as a[N-1:0]
Take 2's comp b[N-1:0]
Take as it as b[N-1:0]
Unsigned input 1
Unsigned input 2
S=a[N-1] EXOR b[N-1]
Unsigned NxN vedic multiplier
Sign bit S
S=1
YES
Take 2's complement of unsigned vedic multiplier
NO
output of unsigned vedic multiplier becomes output of signed multiplier
output of signed vedic multiplier

**Figure 4: Flowchart for Signed VM**

- Calculate the sign bit by EXOR the MSB of the multiplier and multiplicand. Here 0 EXOR 1 is 1.
- Sign bit is 1, so take the 2s compliment of output of the unsigned multiplier. Otherwise consider the unsigned multiplier output as the final output.
- Combine the sign bit and the output from 2scomplement or unsigned multiplier, which gives the final result as 101001100, which is -180.

## 5. FLOWGRAPH FOR PROPOSED SIGNED COMPLEX MULTIPLIER

The process of multiplication of complex number involves two parts, Real (R) and Imaginary (I) .Lets take two signed complex number ±a±jc and ±b±jd.Consider the multiplication of these two complex numbers are R+jI

$$R+jI = (\pm a \pm jc)\ (\pm b \pm jd)$$
$$= ((\pm ab) - (\pm cd)) + j((\pm ad) + (\pm bc))$$

So the real part of the output can be given by ((±ab) – (±cd)) and the imaginary part of the output can be given by ((±ad) + (±bc)).

### 5.1. Algorithm to find Real Part

Consider a, b, c and d are 4 bit 2s complement signed numbers. So each have the size of [3:0]. The flowchart is shown in fig. 5.

- Find the signed multiplication of A = a× b and B = c × d by using either Booth or Signed VM. So A and B have the size of [7:0]
- Take 2s complement for B.
- Add A and 2s complement of B by 8 Bit Ripple carry Adder (RCA).It gives 9 bit as a result.

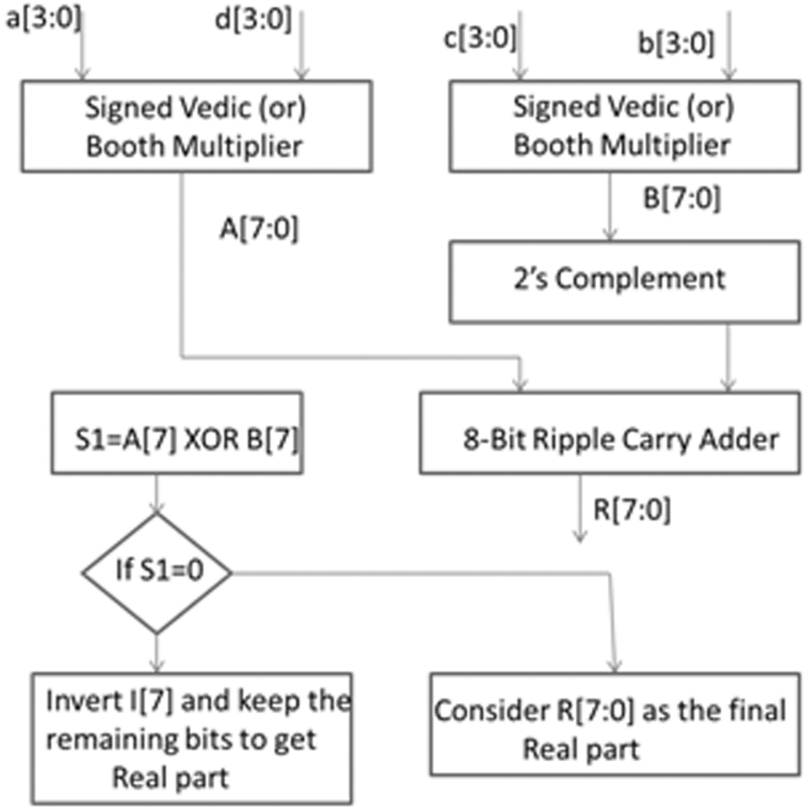

a[3:0]
d[3:0]
c[3:0]
b[3:0]
Signed Vedic (or) Booth Multiplier
Signed Vedic (or) Booth Multiplier
A[7:0]
B[7:0]
2's Complement
S1=A[7] XOR B[7]
8-Bit Ripple Carry Adder
R[7:0]
If S1=0
Invert I[7] and keep the remaining bits to get Real part
Consider R[7:0] as the final Real part

**Figure 5: Flowchart to find Real part of proposed complex multiplier**

- Take XOR between the Most Significant Bits (MSB) of A and B. Consider this is S1.
- If s1 = 0, invert the MSB of RCA and keep the remaining bits of RCA to get the final Real part. Otherwise consider the 9 bit RCA output as the real part (R) of the output.

### 5.2. Algorithm to find Imaginary Part

The flowchart to find Imaginary part is shown in fig. 6.

- Find the signed multiplication of A = a× d and B = b × c by using either Booth or Signed VM. So A and B have the size of [7:0].
- Add A and B by * Bit Ripple carry Adder (RCA). It gives 9 bit as a result.
- Take XOR between the Most Significant Bit (MSB) of A and B. Consider this is S2.
- If s2 = 1, invert the MSB of RCA and keep the remaining bits of RCA to get the final Imaginary part. Otherwise consider the 9 bit RCA output as the Imaginary part (R) of the output.

## 6. RESULTS AND DISCUSSIONS

The proposed Signed complex multiplier based on Radix_2 Booth and Signed Vedic Multiplier are coded in Verilog, synthesized and simulated using ISE simulator. It is implemented on the iWavesystems Unified Learning Kit Spartan6 family xc6slx25t-2fgg484 FPGA. Table 1 shows the comparison of no. of slices used by both the multipliers for same number of input and output bonds.

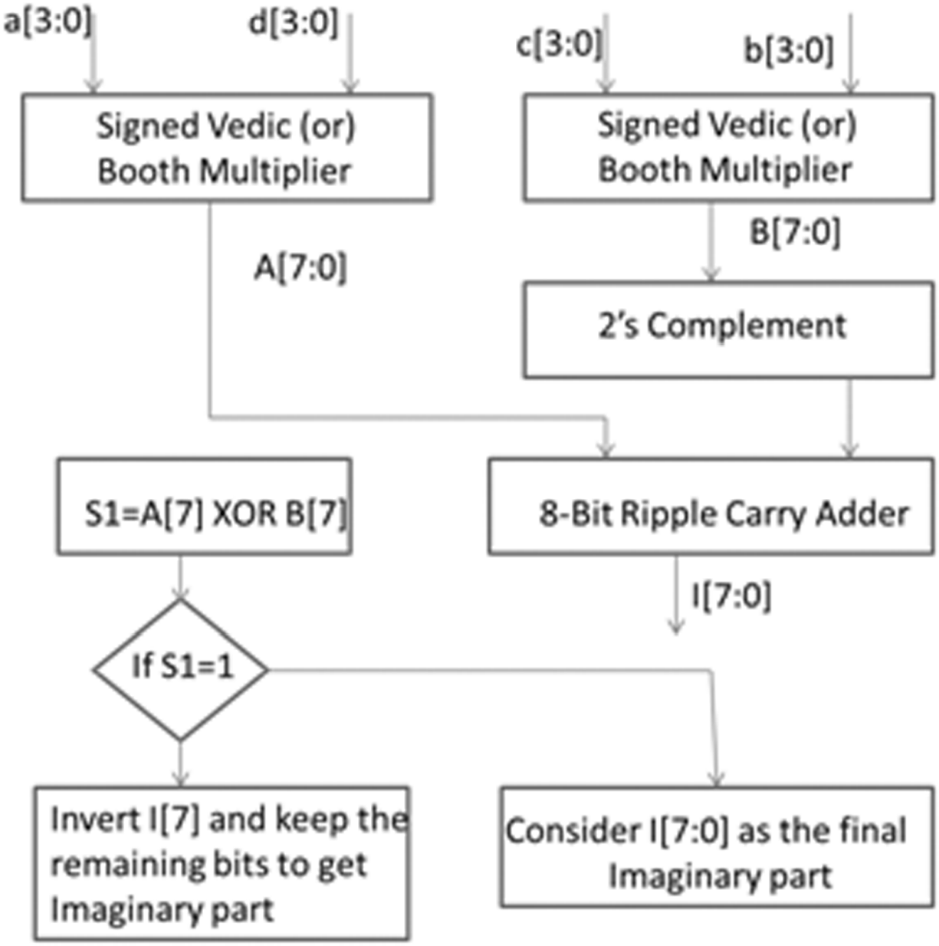

a[3:0]
d[3:0]
c[3:0]
b[3:0]
Signed Vedic (or) Booth Multiplier
Signed Vedic (or) Booth Multiplier
A[7:0]
B[7:0]
2's Complement
S1=A[7] XOR B[7]
8-Bit Ripple Carry Adder
I[7:0]
If S1=1
Invert I[7] and keep the remaining bits to get Imaginary part
Consider I[7:0] as the final Imaginary part

**Figure 6: Flowchart to find Imaginary part of proposed complex multiplier.**

**Table 1**
**Area Comparison.**

| *Multiplier* | *Complex multiplier based on Signed VM* | *Complex multiplier based on Booth* | *Percentage of Area Reduced (%)* |
|---|---|---|---|
| 8 × 8 | 683 | 830 | 17.7 |

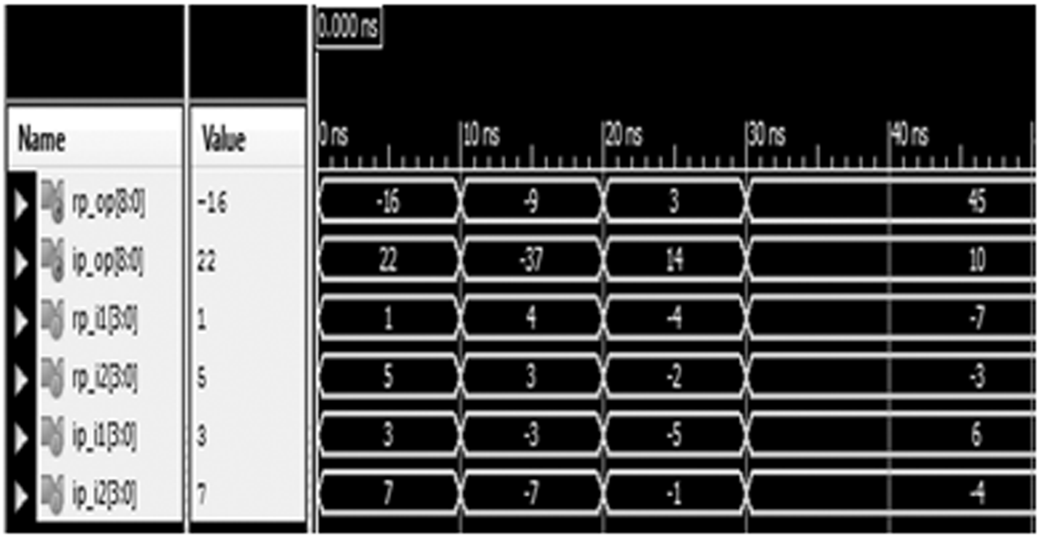
**Table 2**
**Path Delay Comparison**

| *Multiplier* | *Complex multiplier based on Signed VM (ns)* | *Complex multiplier based on Booth (ns)* | *Percentage of Delay Reduced (%)* |
|---|---|---|---|
| 8 × 8 | 30.264 | 37.972 | 20.3 |

TABLE 2 shows the comparison of maximum combinational path delay of both the multipliers.

From the above two table, one can infer that the proposed signed complex multiplier based on Signed Vedic Multiplier has low path delay and it utilizes minimum no. of slices when compared to proposed signed complex multiplier based on Booth Multiplier.

### 6.1. Output Waveform for proposed Complex Multiplier



| Name | Value |
|---|---|
| rp_op[8:0] | -16 |
| ip_op[8:0] | 22 |
| rp_i1[3:0] | 1 |
| rp_i2[3:0] | 5 |
| ip_i1[3:0] | 3 |
| ip_i2[3:0] | 7 |

Figure 7: Output waveform

### 6.2. Implementation on ULK

Suppose if we want to multiply 3-j3 and 2+j7, which gives 27+j15. The real part of the output is 00011011 (27), which is shown in fig. 8 and the imaginary part of the output is 00001111 (15) which is shown in fig.9.

## 7. CONCLUSIONS

The aim to design a high speed complex multiplier for both signed and unsigned numbers has been achieved. We discussed about two multiplication techniques namely radix_2 Booth multiplication and signed Vedic Multiplication. In the proposed complex multiplier based on signed Vedic multiplier, the speed is optimized subsequently by reducing the maximum combinational path delay by 20.3% when compared to complex multiplier based on Booth

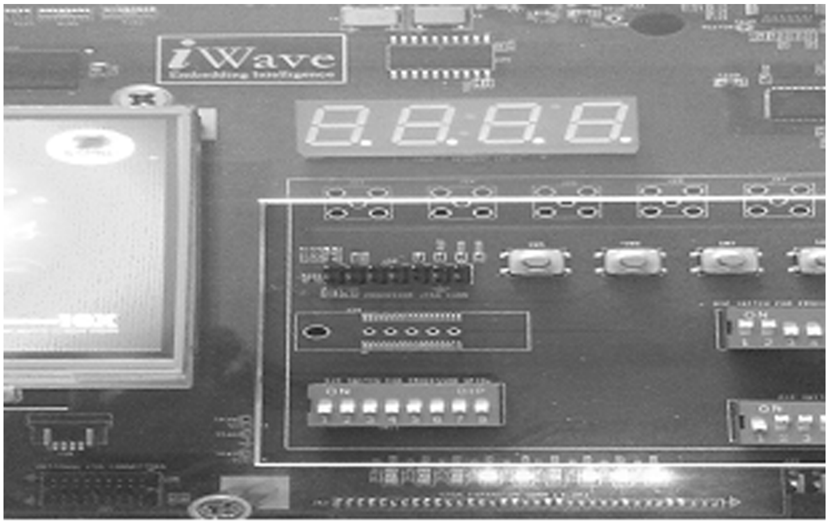


Figure 8: Real part of Output

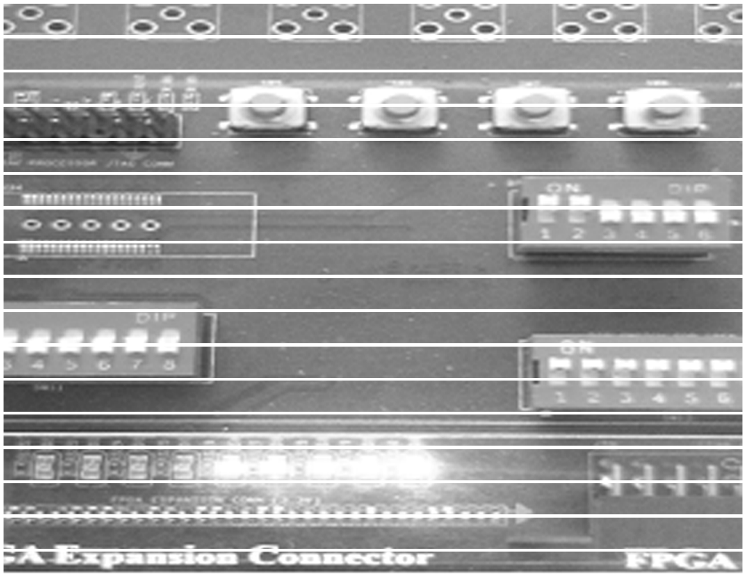


Figure 9: Imaginary part of Output

multiplication. The area is reduced by 17.7%. Further the work is to be extended up to 64 bit complex numbers. This high speed complex multiplier has its application in FFT computation in Digital Signal Processing, Transforms in Image Processing and also applications of VLSI Signal Processing.

## REFERENCES

[1] Rajashri K. Bhongade, Sharada G.Mungale and Karuna Bogawar, "Vhdl Implementation and Comparison of Complex Multiplier Using Booth's and Vedic Algorithm," COMPUSOFT, An international journal of advanced computer technology, ISSN: 2320-0790, Volume-3, Issue -3, March 2014, pp.599-603.

[2] Rupa A. Tomaskar and Gopichand D. Khandale, "FPGA Implementation of Complex Multiplier Using Urdhva Tiryakbham Sutra of Vedic Mathematics", International Journal of Engineering Research and Applications (IJERA), ISSN: 2248-9622, Volume-4, Issue -5 (Version 3), May 2014, pp.01-05.

[3] Ramalatha, M.Dayalan, K D Dharani, P Priya, and S Deoborah, "High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques", International Conference on Advances In Computational Tools for Engineering Applications (ACTEA) IEEE, pp. 600-603, July15-17, 2009, pp.600-603.

[4] Laxman S, Darshan Prabhu R, Mahesh S Shetty ,Mrs. Manjula BM and Dr. Chirag Sharma, "FPGA Implementation of Different Multiplier Architectures", International Journal of Emerging Technology and Advanced Engineering (IJETAE), ISSN:2250-2459,Volume-2,Issue -6, June 2012, pp.292-295.

[5] Archana Fande and Anil Sahu, "Efficient Implementation & Comparison of Signed Complex Multiplier on FPGA using FFT Algorithm", International Journal of Scientific Research Engineering & Technology (IJSRET), ISSN: 2278-0882, Volume-3, Issue -2, May 2014, pp.182-191.

[6] V.R.Raut and P.R.Loya, "FPGA Implementation of Low Power Booth Multiplier Using Radix-4 Algorithm", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE), ISSN: 2320-3765, Volume-3, Issue -8, August 2014, pp.11479-11486.

[7] M.Nagaraju, R.Surya Prakash and B.Vijay Bhaskar, "High Speed ASIC Design of Complex Multiplier Using Vedic Mathematics", International Journal of Engineering Research and Applications (IJERA), ISSN:2248-9622,Volume-3,Issue -1, January-February 2013, pp.1079-1084.

[8] Pushpalata Verma, K. K. Mehta "Implementation of an Efficient Multiplier based on Vedic Mathematics Using EDA Tool" International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-5, June 2012, pp.75-79.

[9] Rajashri Bhongade, S.G.Mungale and Karuna Bogavar , "Performance Evaluation of High Speed Complex Multiplier Using Vedic Mathematics," International Journal of Innovative Research in Advanced Engineering (IJIRAE), ISSN: 2278-2311, Volume-1, Issue -1, April 2014, pp.98-102.

[10] Anju, "Performance Comparison of Vedic Multiplier and Booth Multiplier", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958, Volume-2, Issue -5, June 2013, pp.336-339.

[11] Sneha Manohar, Ramteke, Yogeshwar, Khandagre and Alok Dubey, "Implementation of Low Power Booth's Multiplier by Utilizing Ripple Carry Adder", International Journal of Scientific and Engineering Research (IJSER), ISSN: 2229–5518, Volume-5, Issue-5, May 2014, pp.145-149.

[12] Garima Tiwari, "Analysis, Verification and FPGA Implementation of Low Power Multiplier", International Journal of Scientific and Engineering Research (IJSER), ISSN: 2319 –1163, Volume-2, Issue-3, May 2013, pp. 220-224.