FPGA Implementation of Booth's and Baugh-Wooley Multiplier using Verilog

Manish Chaudhary, Mandeep Singh Narula

Abstract: - Here, in this paper we have designed and implemented a Signed-Unsigned Booth's Multiplier and a Signed-Unsigned Baugh-Wooley Multiplier for 32-bits multiplication. The designing and verification is done through verilog on Xilinx 12.4. In this paper we tried to explain the step by step process that was adopted for Signed-Unsigned Booth's Multiplier. Also, two different approaches for implementing the Signed Baugh-Wooley multiplier in Singed-Unsigned Baugh-Wooley multiplier and after, the implementation we could see the differences in certain parameters. The array structure of Signed-Unsigned Booth's Multiplier and Signed-Unsigned Baugh-Wooley Multiplier is obtained from RTL synthesis are shown. Different parameters like power, CPU usage, CPU time, memory usage etc. have been compared.

Key Words: - array, booth, baugh-wooley, signed, unsigned, verilog,

I. INTRODUCTION

As known, multiplication plays as one of the most important function and is also the most important integral function in arithmetic operations [1]. In some applications of digital signal processing (DSP) like convolution and in FFT some of the computational intensive arithmetic function are applied regularly (e.g. Multiply and Accumulate, inner products) [2]. The multiplication can be defined when the process of addition is repeated multiple number of times, we understand from the multiplicand and the multiplier is that the number to be added is defined by multiplicand and the number of times it has to be added is given by the multiplier [3]. Optimization of the multiplier is a must for achieving maximum speed in the multiplier hence, generation of partial products and adding of partial product must be optimized [3]. Now, there are two types of binary numbers on which multiplication can be performed in digital electronics and digital computing they are i) usigned binary numbers and ii) signed binary numbers [4-5]. Signed multiplication is defined as the multiplication performed on signed binary numbers and Unsigned multiplication is performed on unsigned binary numbers [7].

At the LSB position of each partial product row due to the extra partial product bit an irregular partial product array is produced in modified booth's algorithm.
The multiplier which have high speed use this modified booth's algorithm in order to improve the speed of the multipliers by reducing the partial products array multipliers were used earlier [1]. For performing signed multiplication the baugh-wooley algorithm is relatively suited.
There are three steps involved in a N-bit wide multiplier for the reorganization of the partial product array. 1) for the N-1 partial product row the MSB and except the MSB of the last partial product row are inverted ii) in the Nth column 1 is added iii) of the final result the MSB is inverted[6].

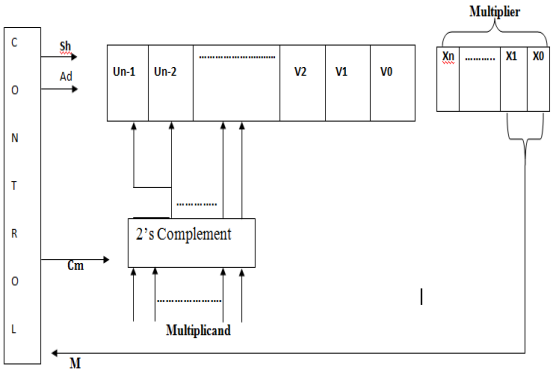

Figure 1: Booth's multiplier

In this paper we have presented a signed-unsigned booth's multiplier and a signed-unsigned baugh-wooley multiplier and implemented the design on FPGA's LCD display. Here, we used two different methods of implementing the signed baugh-wooley multiplier in order to reduce the delay.

II. METHODOLOGY

Here, rather than implementing and designing a booth's encoder and decoder for the multiplier [1]. We have implemented the booth's multiplier according to the steps in the algorithm.
Booth's Algorithm for signed multiplication:

1. Let, A, B and P be the predetermined values of x+y+1 length.
   a. A: the binary value of c represent the MSB(most significant bit)position and y+1 number of zeros be appended in the LSB(least significant bit)position.
   b. B: the two's compliment of c represent the MSB(most significant bit)position and y+1 number of zeros be appended in the LSB(least significant bit)position
   c. P: here, we append x bits of zero in the MSB(most significant bit)position then substitute the binary value of d and the LSB(least significant bit)position be represented by zero.
2. Now, considering some of the conditions for addition and arithmetic shift.
   a. In P if 01 represents the two positions of the LSB then A is added into P. Hence, P=P+A.
   b. In P if 10 represent the two positions of the LSB then B is added into P. Hence, P=P+B.
   c. In P if 11 represent the two positions of the LSB then there is no change in value of P
   d. In P if 00 represent the two positions of the LSB then there is no change in value of P

Revised Manuscript Received on June 06, 2013.
Manish Chaudhary, ECE, ITM University,Gurgaon, India.
Astt. Prof. Mandeep Singh Narula , ECE,ITM University,Gurgaon, India.

3. When, the above conditions are verified and executed then arithmetic shift is done.
4. Repeat the second and third step for the number of bits in the multiplier.
5. This step is the most crucial step of all as here we get the final result. To get the final product we have to drop the LSB from [7].

After, these steps are implemented we have successfully designed a signed booth's multiplier. Now, for unsigned booth's multiplier the algorithm is similar to as that of signed booth's multiplier with just one simple modification

1. Let, A, B and P be the predetermined values of x+y+1 length.
   a. A: the binary value of c represent the MSB(most significant bit)position and y+1 number of zeros be appended in the LSB(least significant bit)position.
   b. P: here, we append x bits of zero in the MSB(most significant bit)position then substitute the binary value of d and the LSB(least significant bit)position be represented by zero.
2. Now, considering some of the conditions for addition and arithmetic shift.
   a. In P if 01 represents the two positions of the LSB then A is added into P. Hence, P=P+A.
   b. In P if 10 represent the two positions of the LSB then B is added into P. Hence, P=P-A.
   c. In P if 11 represent the two positions of the LSB then there is no change in value of P
   d. In P if 00 represent the two positions of the LSB then there is no change in value of P
3. When, the above conditions are verified and executed then arithmetic shift is done.
4. Repeat the second and third step for the number of bits in the multiplier.

This step is the most crucial step of all as here we get the final result. To get the final product we have to drop the LSB from P.

After implementing signed booth's multiplier and unsigned booth's multiplier we combine the two together in a single multiplier by using a select/control line.

In order to implement the signed baugh-wooley multiplier the array structure must be known

$$P = a[m-1]b[n-1]2^{m+n-2} + \sum_{i=0}^{i=n-2} NOT(a[m-1b[i])\,2^{i+m-1} + \sum_{j=0}^{j=m-2} NOT(a[j]b[n-1])\,2^{i+m-1} + \sum_{i=0,j=0}^{i=m-2,j=n-2} (a[i]b[j])2^{i+j} + 2^{(n-1)} + 2^{(m-1)} - 2^{(m+n-1)} \quad \ldots\ldots (1)\ [8]$$

Hence, the above equation will tell us about the array structure by placing m=n=32 in order to implement the 32 bit signed baugh-wooley multiplier.

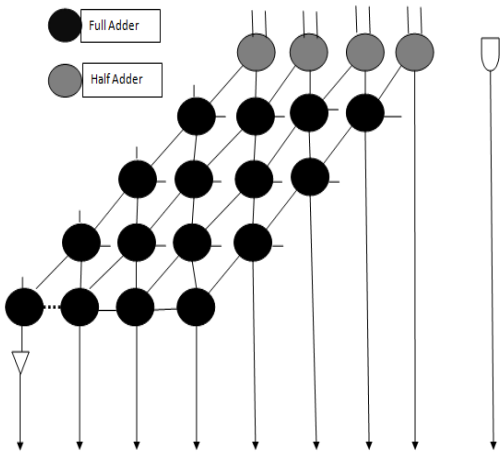

Figure 2: Baugh-Wooley multiplier using full adders and half adders

The figure 2 shows one of the methods that was adopted to implement the signed baugh-wooley multiplier with the use of full adders and half adders. While the second method comprises of carry save adder(CSA) which replaces all the full adders and half adders.

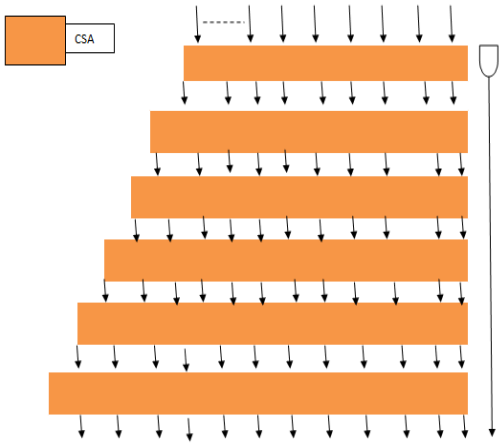

Figure 3 : Baugh-Wooley multiplier using CSA

Figure 3 shows the second method that is adopted to implement the signed baugh-wooley multiplier.

Now, to implement the unsigned baugh-wooley multiplier the steps involved are similar to that of normal multiplication that is shift and add [8].

```
                    a3   a2   a1   a0
X                   b3   b2   b1   b0
-------------------------------------------
                  a3b0 a2b0 a1b0 a0b0
             a3b1 a2b1 a1b1 a0b1
        a3b2 a2b2 a1b2 a0b2
   a3b3 a2b3 a1b3 a0b3                    Product P,
-------------------------------------------
                                a0b0  ---> p0
                           a1b0+a0b1  ---> p1
                      a2b0+a1b1+a0b2+carry  ---> p2
                 a3b0+a2b1+a1b2+a0b3+carry  ---> p3
            a3b1+a3b2+a1b3+carry  ---> p4
       a3b2+a2b3+carry  ---> p5
  a3b3+carry  ---> p6
carry  ---> p7
```

Figure 4: Unsigned baugh-wooley multiplier

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

After implementing signed baugh-wooley multiplier and unsigned baugh-wooley multiplier we combine the two together in a single multiplier by using a select/control line

After the successful implementation of signed-unsigned booth's multiplier and baugh-wooley multiplier they are implemented on FPGA and their results are displayed on the 2x16 LCD display present.

III. RESULTS AND WAVEFORM

When the verilog code is executed and verified through the help of test bench the RTL's and waveforms are obtained and the waveforms are analysed for error if there are no errors found then we have successfully implemented the multipliers. The waveforms and the arrangement of array structure for the multipliers are shown below.

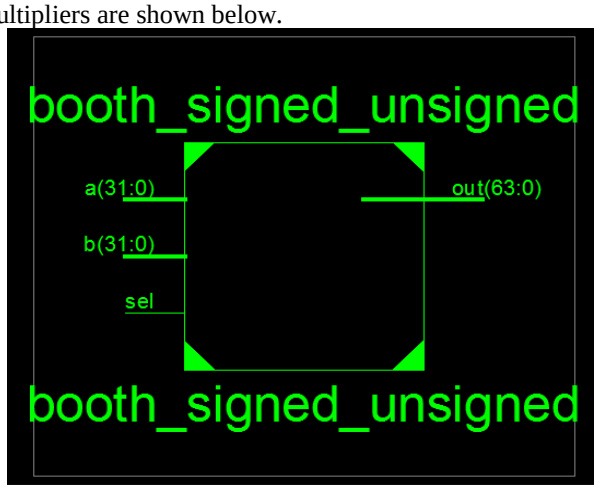

Figure 5:RTL Of Booth Multiplier

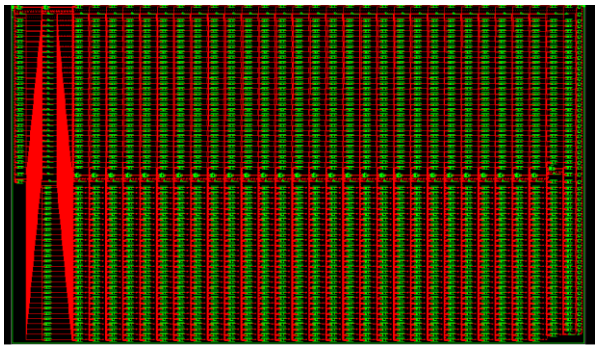

Figure 6: Array Structure Of Booth's Signed-Unsigned Multiplier

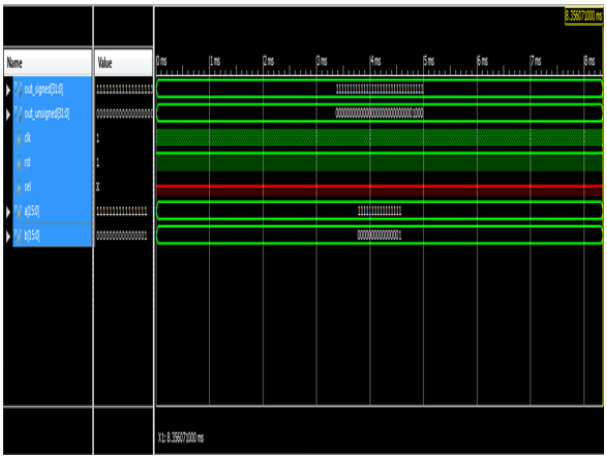

Figure 7:For signed-unsigned Booth Multiplier where a=-1 and b=1 sel=1(signed multiplier is choosen)

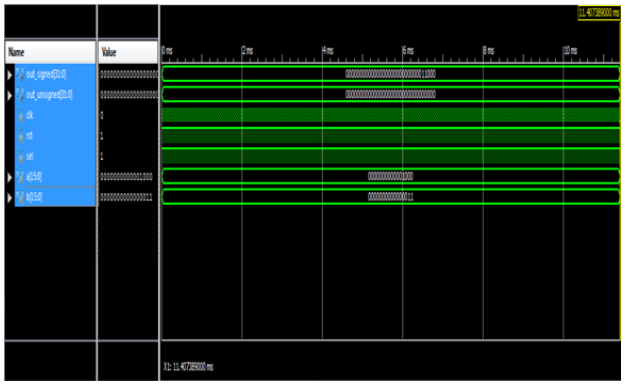

Figure 8: For signed-unsigned Booth Multiplier where a=8 and b=3 sel=0(unsigned multiplier is choosen)

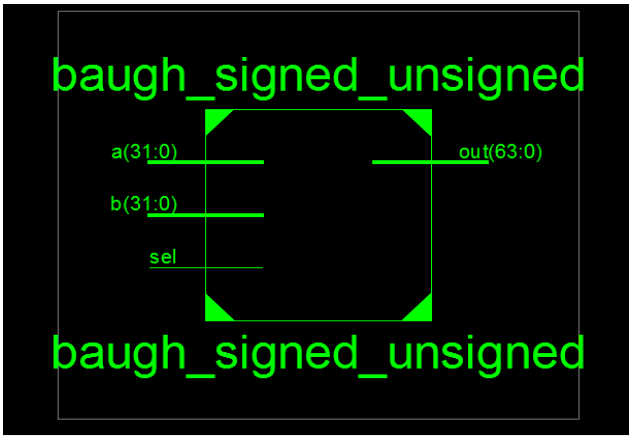

Figure 9:RTL Of Baugh-Wooley

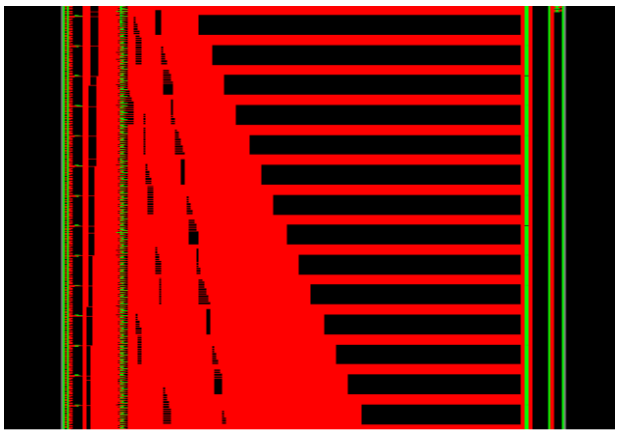

Figure 10: Array Structure Of Baugh-Wooley

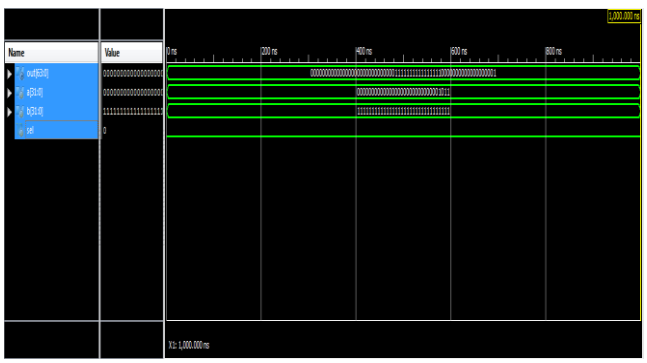

Figure 11: For Signed-Unsigned Baugh-Wooley Multiplier Where a=11 and b=515 sel=0(Unsigned Baugh-Wooley Multiplier Is Chosen)

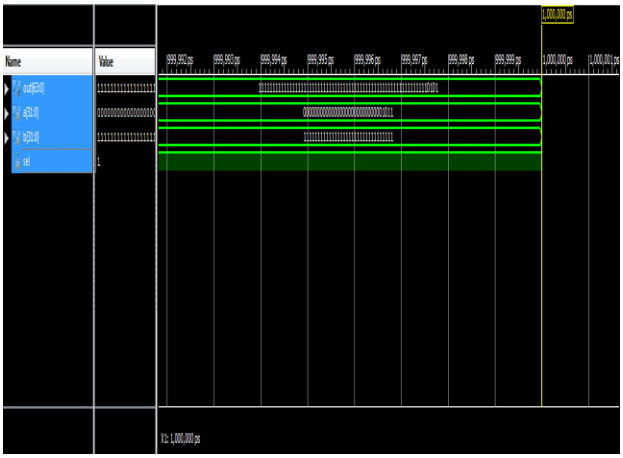

Figure 12: For Signed-Unsigned Baugh-Wooley Multiplier where a=11, -1 and b= 515 sel=1(signed Baugh-Wooley Multiplier Is chosen)

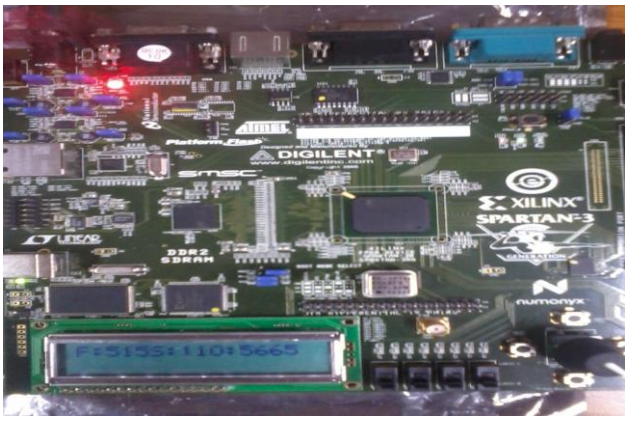

Figure 13:FPGA implementation of Booth's multiplier

Figure 14:FPGA implementation of Baugh-wooley multiplier

Parameters	Signed-unsigned Baugh-wooley Using CSA	Signed-unsigned Baugh-wooley Using full adder and half adder	Signed-unsigned Booth's multiplier
Voltage	0.076W	0.076W	0.076W
Current	0.26A	0.26A	0.26A
Logic	4066/9312 (43% utilization)	4037/9312 (43% utilization)	Na
I/O's	129/232 (56% utilization)	129/232 (56% utilization)	64/232 (28% utilization)
Maximum Combinational Path Delay	82.503ns	93.656ns	81.826ns

IV. CONCLUSION

We have successfully implemented the signed-unsigned booth's multiplier and signed-unsigned baugh-wooley multiplier on FPGA and it can seen from the above table that there is 11.90% improvement in the maximum combinational path delay for the signed-unsigned baugh-wooley multiplier using CSA as compared to the Signed-unsigned baugh-wooley multiplier using full adders and half adders.

REFERENCES

1. Ravindra P Rajput M. N Shanmukha Swamy" High speed Modified Booth Encoder multiplier for signed and unsigned numbers", 2012 14th International Conference on Modelling and Simulation, 978-0-7695-4682-7/12
2. Shiann-Rong Kuang, Member, IEEE, Jiun-Ping Wang, and Cang-Yuan Guo" Modified Booth Multipliers With a Regular Partial Product Array", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, VOL. 56, NO. 5, MAY 2009, 1549-7747
3. J.Yeo "Low Voltage,LowPower Vlsi"
4. Floyd, Floyd Thomas L. "Digital Fundamentals" ISBN-978-81-7758-763-0 ,2009
5. James E. Stine "Digital Computer Arithmetic Datapath Design Using Verilog Hdl", Volume 1, ISBN-1-4020-7710-6
6. Magnus Själander and Per Larsson-Edefors" High-Speed and Low-Power Multipliers Using the Baugh-Wooley Algorithm and HPM Reduction Tree", 978-1-4244-2182-4/08, 2008 IEEE
7. Andrew D. Booth. A signed binary Multiplication technique. The Quarterly Journal of Mechanics and Applied Mathematics, Volume IV, Pt. 2, 1951
8. Baugh-Wooly Data Sheet " High Performance Multipliers in Quick Logic FPGAs"

AUTHORS PROFILE

Manish Chaudhary. He completed the B.Tech in the field of Electronics and Communications from Greater Noida Institute Of Technology(G.B.T.U, University) from Greater Noida in the year of 2011. Now, pursuing Master Degree(m.tech) in the field of Electronics and Communications from ITM University Gurgaon, Haryana

Mandeep Singh Narula has completed M.Tech in Microelectronics and VLSI from IIT Kharagpur in the year 2008. His areas of interest are Low Power VLSI, RTL Verification, and Analog VLSI.