A NOVEL APPROACH OF HIGH SPEED BOOTH ENCODER MULTIPLIER FOR INTEGERS

K.Hareesh Chakravarthy 1*, N.Satyanarayana 2*

1. II.M.Tech (VLSI), Dept of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.

2. Asst.Prof, Dept. of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.

ABSTRACT

In this propose paper we have designed a signed booths multiplier as well as an signed and unsigned booth's multiplier for 4 bit and 8 bit and 16 bits performing the multiplication on both signed and unsigned numbers. Propose work is done through verilog on xiling 12.4 platform which provides calculated diversity by using various parameters. The unsigned booth multiplication is implemented when we done some modifications in the booths multiplication method. In this paper we have to propose booths algorithm for both signed and unsigned numbers divided into five steps.

Keywords: *Verilog, booth, signed and unsigned multiplier.*

I. INTRODUCTION

In arithmetic operation multiplication is an essential one and its applications are dated several decades back in time. Earlier ALU's adders were used to perform the multiplication originally. As the applications of Array multipliers were introduced the clock rates increased as well as timing constrains became austere. Ever since then methods to implement multiplication are proposed which are more sophisticated. As known the use of multiplication operation in digital computing and digital electronics is very intense especially in the field of multimedia and digital signal processing (DSP) applications. There are mainly three stages to perform multiplication: The first stage mainly consists of generating the partial products which are generated through an array of AND gates; Second stage consist of reducing the partial products by the use of partial product reduction schemes; and finally the product is obtained by adding the partial products.

The multiplication can be performed on: 1) Signed Numbers; 2) Unsigned Numbers. Signed multiplication a binary number of either sign (two numbers whose sign may are not necessarily positive) may be multiplied. But, in signed multiplication the sign-extension for negative multiplicands is not usable for negative multipliers and there are large numbers of summands due to the large sequence of 1's in multiplier. Unsigned multiplication binary number (whose sign is positive) is multiplied.

The Booth's algorithm is powerful for signed number multiplication (larger multiplier, lager number of multiplicands to be added). It performs multiplication by performing 2'compliment and the regular shift and adds process. As due to the extra partial bit at the least significant bit position there is irregularity produced in the array. It is seen that in booth technique if the operands are large (bit numbers) and there is a long sequence of 1's it is advantageous.

The booth's algorithm for multiplication can be modified to perform unsigned multiplication along with signed multiplication. We have done some basic changes in the algorithm to obtain the result of signed-unsigned multipliers. After implementing the signed booth multiplier and unsigned booth multiplier for 4, 8 and 16 bits binary numbers we have compared their parameters (power usage, current leakage, CPU usage, and memory usage) results with each other.

II. METHODOLOGY OF BOOTHS MULTIPLICATION

The algorithm is based on checking the bits of multiplier Y in two's compliment, which also include the implicit bit below the LSB,*Y-1=0.* Here we consider the addition of multiplicand 2^i to the product accumulator, considering y_i =0 and y_{i-1}=0 and 2^i multiplicand is subtracted from the product accumulator. Hence, the final product is achieved.

A. Steps of Booth's Algorithm implementation of (Signed)

Booth's algorithm is executed by repeatedly adding one of the two multiplicands and multipliers and then performing the rightward arithmetic shift.

Consider e and f a multiplicand and multiplier respectively these two values which are going to be multiplied and give product. Let, the bits of e and f be represented by x and y.

- ✓ Firstly we arbitrate the values of two predetermined A and S to obtain the product P. length of all these numbers should be equal(x+y+1)

a) Substitute the value of **e**(binary) in the MSB(most significant bit) and append the remaining bits with y+1 zeros.

b) S: Substitute the value of **-e**(two's compliment notation) in the MSB (most significant bit) and append the remaining bits with y+1 zeros.

c) P: Substitute the MSB with x bits of zeros. Then to the right of this insert the value of **f** and append the LSB (least significant bit) bits with zero.

- ✓ Now, consider the two least significant bits of **P.**

a) If the two least significant bits are 01, then the values of P becomes P+A. ignoring overflow

b) If the two least significant bits are 10, then the values of P becomes P+S. ignoring overflow

c) If the two least significant bits are 00, then the values of P is used as it is.

d) If the two least significant bits are 11, then the values of P is used as it is.

- ✓ After the completion of the 2nd step we will do the Arithmetic shift by single place towards the right. Let the new value be equal to P now.
- ✓ For y number of times repeat the 2nd and 3rd step.
- ✓ To obtain the final product result of e and f we have to drop the least significant bit from P.

B. Steps of Booth's Algorithm implementation of (Unsigned)

The booth's algorithm for unsigned multiplication is almost same but the only difference is the along with repeatedly adding the two multiplicands and multipliers they are also repeatedly subtracted and then the arithmetic shift is performed. Secondly two's compliment is not done.

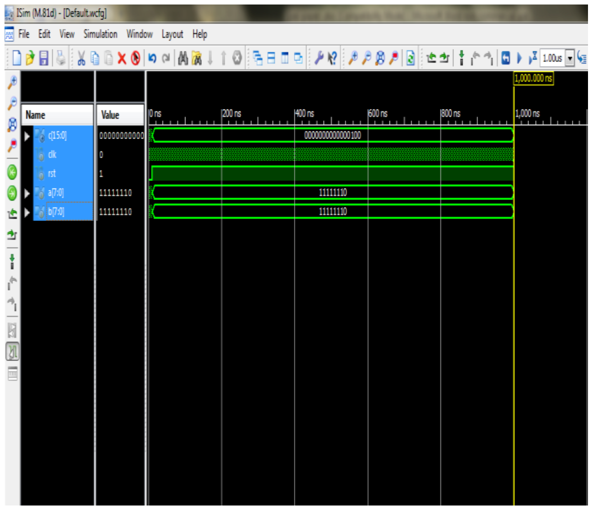
Consider e and f a multiplicand and multiplier respectively these two values which are going to be multiplied and give product. Let, the bits of e and f be represented by x and y.

- ✓ Firstly we arbitrate the values of two predetermined A and S to obtain the product P. length of all these numbers should be equal(x+y+1)
  a) A: Substitute the value of e(binary) in the MSB(most significant bit) and append the remaining bits with y+1 zeros.
  b) S: Substitute the value of e in the MSB (most significant bit) and append the remaining bits with y+1 zeros.
  c) P: Substitute the MSB with x bits of zeros. Then to the right of this insert the value of f and append the LSB (least significant bit) bits with zero.
- ✓ Now, consider the two least significant bits of P.
  a) If the two least significant bits are 01, then the values of P becomes P+A. ignoring overflow
  b) If the two least significant bits are 10, then the values of P becomes P-S. ignoring overflow
  c) If the two least significant bits are 00, then the values of P is used as it is.
  d) If the two least significant bits are 11, then the values of P is used as it is.

- ✓ After the completion of the 2nd step we will do the Arithmetic shift by single place towards the right. Let the new value be equal to P now.
- ✓ For y number of times repeat the 2nd and 3rd step.
- ✓ To obtain the final product result of e and f we have to drop the least significant bit from P.

III. SIMULATION RESULT

In this section we have written and simulated the Verilog code on xilings of the booth's multipliers (signed and unsigned) for 4-bit and 8-bit and have verified it through a test bench, which has generated a waveform representing the output function. The name of simulator is ISim. In this simulator we observe test benches. In test bench we get waveforms representation of the given parameters and response of the system according to given parameters.


ISim (M.81d) - [Default.wcfg]
File Edit View Simulation Window Layout Help
1.00us
1,000.000 ns
0 ns
200 ns
400 ns
600 ns
800 ns
1,000 ns

Name	Value	
c[15:0]	00000000000	0000000000000100
clk	0	
rst	1	
a[7:0]	11111110	11111110
b[7:0]	11111110	11111110

Figure 1: Waveform Of The Product (8-bit signed)

Figure1 shows the output waveform. Here, the input given is 8 bit value a = 8'b11111110; b = 8'b11111110. After the executing the test bench according to the verilog code the output waveform is achieved which is c=0000000000000100

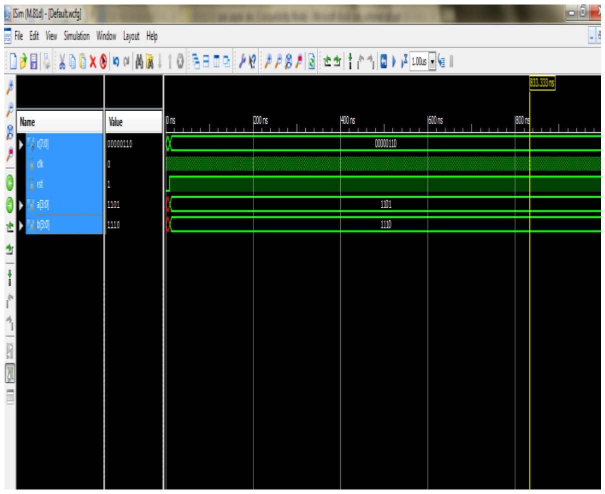


Figure 2: Waveform of The Product (4-bit signed)

Figure2 shows the output waveform. Here, the input given is 8 bit value a = 4'b1101; b = 4'b1110;. After the executing the test bench according to the Verilog code the output waveform is achieved which is c=00000110.

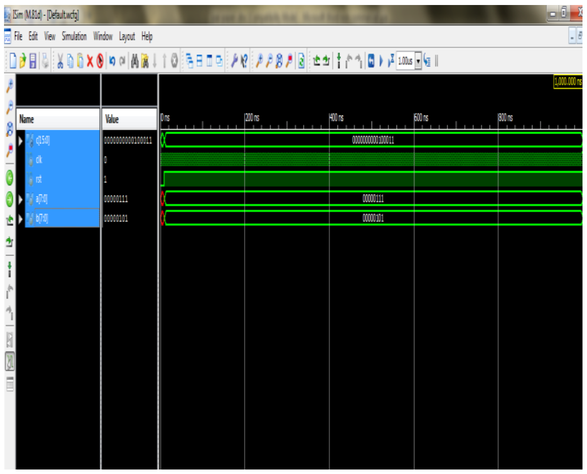


Figure 3:Waveform Of The Product(8-bit unsigned)

Figure3 shows the output waveform. Here, the input given is 8 bit value a = 8'b00000111; b = 8'b00000101. After the executing the test bench according to the verilog code the output waveform is achieved which is c=0000000000100011.

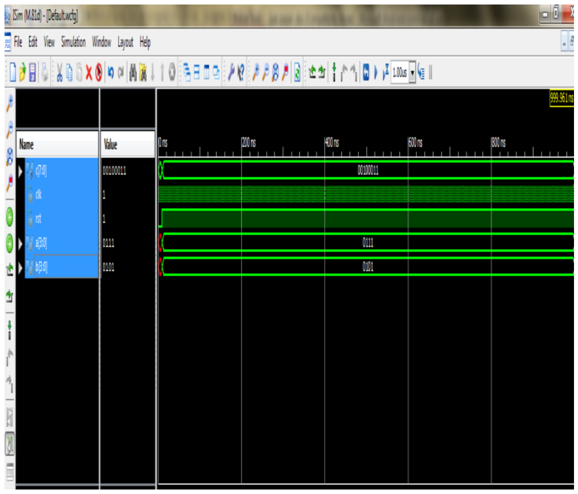


Figure4: Waveform of The Product (4-bit unsigned)

Figure4 shows the output waveform. Here, the input given is 4 bit value a = 4'b0111;b = 4'b0101;. After the executing the test bench according to the Verilog code the output waveform is achieved which is c=00100011

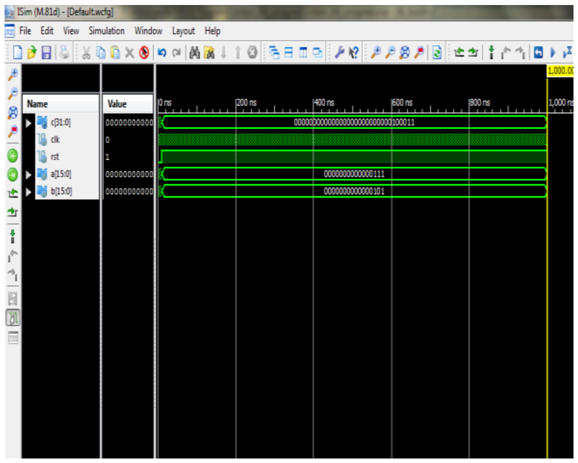


Figure 5: Waveform of The Product (16-bit unsigned)

Figure5 shows the output waveform. Here, the input given is 16bit valuea=16'b0000000000000111;b=16'b0000000000000101; After the executing the test bench according to the verilog code the output waveform is achieved which is c=0000000000000000000000000100011.

From the above simulated results we can see the difference in the various parameters of the multiplier of 4,8,16 bits. The 16-bit unsigned multiplier consumes the maximum power of 37.69 mw as compared to others the minimum power consumption is done by 4-bit singed and unsigned which is 36.87 mw. Also we can see the maximum current is consumed by the 16-bit unsigned multiplier 14.91 ma as compared to others the minimum current consumption is done by 4-bit singed and unsigned which is 14.23 ma.

IV. CONCLUSION

Modified Booth's multiplier has been successfully implemented for both signed and unsigned numbers using Xilinx 12.4 platform. Result obtained from simulation waveforms matches exactly with the desired value for 4 bit, 8 bit and 16 bit signed and unsigned multiplier. Different parameters like leakage power, dynamic power, total power I/O pads used, dynamic and quiescent current etc have been compared for different bits. As expected, total power dissipated for 16 bit multiplier exceeds power for 8 bit multiplier. However the difference is not large.

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.

[5] 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

[6] Ravindra P Rajput and 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 © 2012 IEEE.