Design and Implementation of 256*256 Booth Multiplier and its Applications

G S Surabhi
Department of Electronics and Communication,
K.S. Institute of Technology,
Bangalore

Bhavana. J
Department of Electronics and Communication,
K.S. Institute of Technology,
Bangalore

Madhu. S
Department of Electronics and Communication,
K.S. Institute of Technology,
Bangalore

ArunaRao. B. P
Department of Electronics and Communication,
K.S. Institute of Technology,
Bangalore

***Abstract*-This paper presents design of a booth multiplier which performs both signed and unsigned multiplication. This implementation describes in the form of RTL schematic. In the field of digital signal processing and graphics applications, multiplication is an important and is an application that demands a lot of computations. In this paper, we describe the implementation of 256-bit booth multiplier by comparing it with 64-bit and 128-bit booth multipliers. Booth algorithm employs both addition and subtraction and it also treats positive and negative operands uniformly. The proposed multiplier will be designed and verified using modalism with Verilog HDL, Xilinx is used for synthesis. This paper gives a complete idea of radix-64, radix-128, radix-256form booth algorithm.**

Keywords-Booth Multiplier, Xilinx, readix-256,radix-128, radix-64, multiplication.

I.INTRODUCTION

The multiplication of two binary inputs leads to manynumbers of gate count which occupies a large chip area on the digital system. The algorithm of booth multiplier furnishes a level to formulate a multiplier with greater effectiveness and speed. This algorithm is a better level of encoding.

The Booth multiplier makes use of addition and shifting algorithm. As compare to adder and subtractor multiplier are more complex. Multipliers play aimportant role in digital signal processing and other various applications. In this algorithm a partial product is generated by the multiplication of multiplicand with each bit of the multiplier.

A shift register is a very important digital building block. It has a large amount of applications. Registers are often used to store binary information appearing at the output. Shift registers are the logic types which are used for the storage and transfer of digital data. Multipliers are most commonly used in various electronic applications.

II. LITERATURE SURVEY

This paper briefly describes the method of implementing a digital multiplier called modified booth multiplier. From past one decade a number of efforts have been made to reduce dynamic power consumption in partial product generation and the power adder circuits about 40% of the research is on techniques for reducing partial products by changing the design of the modified booth multiplier.60% is on implementing low power adder circuits, error correction methods and circuit level power optimization techniques. A review from the earlier literature in this area is discussed in this report.

D.Govekar.al develop high speed modified both multipliers using hybrid adder. By employing modified booth multiplier design using hybrid adder shows better performance compare to conventional method of using carry look ahead adder.

Nagarjuna et.al has represented FIR Filter using the idea of multipliers with the reduction of bit size an elements source. FIR framework is applied using the enhanced edition unit multiplier. In proposed technique a booth multiplier is applied in this multiplier signed multiplication is an added advantage.

Honglan Jiang Fie has represented two signed 16*16-bit approximate radix-8 booth multipliers are designed using approximate recording adder with and without truncation of several less significant bits in the partial products. Multiplier are faster and more power efficient.

Prasanna Raj P Ravi has published a paper on analysis of multipliers for low power in, September 2009. In this paper they said that low power multipliers with high clock frequencies play an important role in today's digital signal processing.

Shwetha Khatri et.al, presents the FPGA implementation of a 64-bit fast multiplier using barrel shifter.in this research they have described the implementation of a 64-bit Vedic multiplier which is enhanced in terms of propogation delay. When it is compared with conventional multiplier like modified booth multiplier. Wallace tree multiplier, Braun multiplier, Vedic multiplication techniques for arithmetic operator. Synthesis report and static timing report used for the comparison of propogation delay. The design uses barrel shifter in base selection module and multiplier which achieves propogation delay of 6.78 Ins

III. ADVANTAGES

1. Booth multipliers are easily extensible.
2. These multipliers are easily pipelined.
3. Booth multiplier minimizes complexity.

IV. APPLICATIONS

1. Booth multiplier is arithmetic operator for DSP applications, such as filtering and for Fourier transforms.
2. Booth multiplier is used to achieve high execution speed.
3. These multipliers tend to consume most of power in DSP computation.

V. PROCEDURE AND METHODOLOGY

Here we are using XILINX ISE design suite 13.1 version for Implementing booth multipliers. Device name is XC6XLX4, package is TQG144, synthesis tool is XST and simulation is done by ISIM.Language preferred is VERILOG. This tool is used for implementing design, synthesis RTL view and examine behavioural simulation. This software is easy to use and implement complex designs using VERILOG HDL.

VI. BOOTH MULTIPLIER

Booth's multiplication is meant for multiplying two's compliment representation of signed binary numbers. The algorithm was invented by Andrew Donald Booth in 1950 while doing study on crystallography atBIRKBECKcollege in Bloomsbury, London. Booth algorithm is used for simulation and development of digital multiplier. Booth algorithm uses a small number of additions and shift operations to do the work of multiplications. Booth algorithm is a method that will reduce the number of multiplicand and multipliers.

Booth algorithm converts the multiplier Y in two's compliment form and implicitly appends a bit Y1=0 below the least significant bit. After every multiplication partial product thus generated is shifted according to its bit order and then all the partial products are added to obtain the finial product.

The following are the steps that involves in execution:

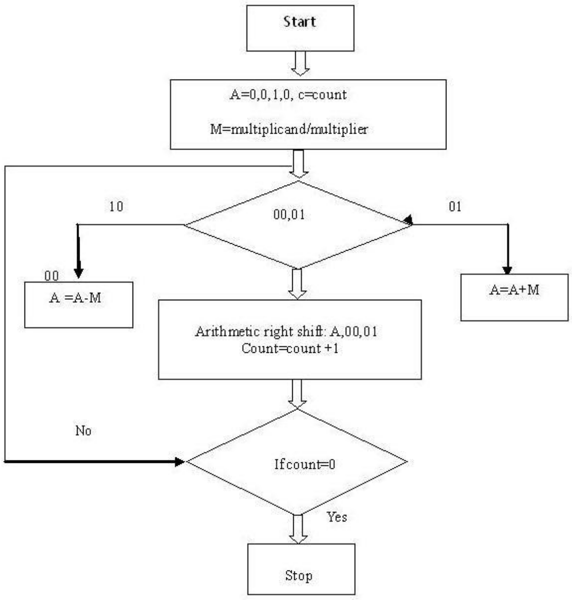
1. The product is in accumulator Z is left unchanged if the compared bits are equal.
2. The multiplicand times 2^i is added to Z when Y=0 and Y1=1.
3. The multiplicand times 2^i is subtracted to Z when Y=1 and Y1=0.

Finally, a signed product Z is obtained.

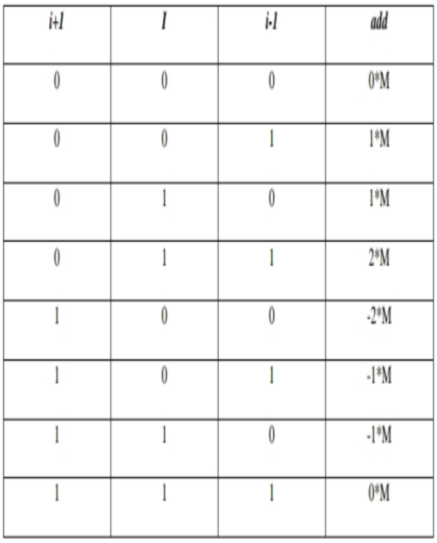
Similar to the multiplier both the multiplicand and product are also in two's compliment representation. The algorithm can be generalized to any number system which can support addition and subtraction.

The ordering of iterations is from LSB to MSB starting from I=0. Accumulator Z is shifted to1 right for multiplication by 2 and LSB is shifted out in the case. Subsequent computation of addition and subtraction is done executed on the resulting N bits of Z.

VII. BOOTH MULTIPLIER FLOW CHART



VIII. BOOTH MULTIPLIER TRUTH TABLE



i+1	I	i-1	add
0	0	0	0*M
0	0	1	1*M
0	1	0	1*M
0	1	1	2*M
1	0	0	-2*M
1	0	1	-1*M
1	1	0	-1*M
1	1	1	0*M

IX. RESULT

Design and simulation of 64-bit 128-bit and 256-bit booth multiplier is implemented. We have implemented 128- bit and 256-bit multipliers which is not been found in most of the papers. The simulation output of the booth multipliers until 256-bits are displayed. We have used radix-128 and radix-256 which is complexed and increases the speed and reliability.

RTL view for 64-bit booth multiplier:

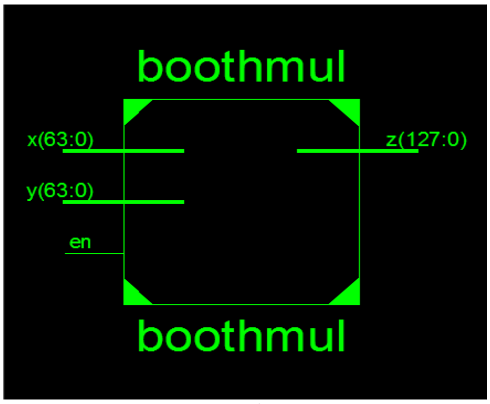
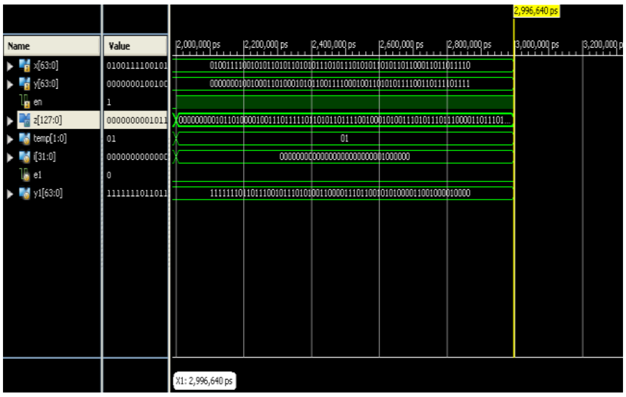


Fig1: RTL view for 64-bit Multiplier

Viewing an RTL schematic opens anNGR filethat can be viewed as a gate level schematic. This schematic is generated after the HDL synthesis phase of the synthesis process. The above figure shows the RTL schematic of the 64-bit booth multiplier

Result and Simulations of 64-bit booth multiplier:



```
Name        Value            2,000,000 ps  2,200,000 ps  2,400,000 ps  2,600,000 ps  2,800,000 ps  3,000,000 ps  3,200,000 ps
x[63:0]     0100111100101    0100111100101011010110101011101011101010110101101100011011011110
y[63:0]     0000000100100    0000000100100011010001010110011110001001101010111100110111101111
en          1
z[127:0]    0000000001011    00000000010110100001001110111110110101101111001000101001110101110110000110111101...
temp[1:0]   01               01
i[31:0]     0000000000000    00000000000000000000000001000000
e1          0
y1[63:0]    1111111011011    1111111011011100101110101001100001110110010101000011001000010000
2,996,640 ps
X1: 2,996,640 ps
```

Fig2: output waveform for 64-bit multiplier

Here the Verilog code for 64-bit multiplier is being synthesized using the Xilinx version of 13.1 on the device XC6XLX4 and the results for the synthesis of 64-bit multiplier is as shown in the fig.2

RTL view for 128-bit booth multiplier

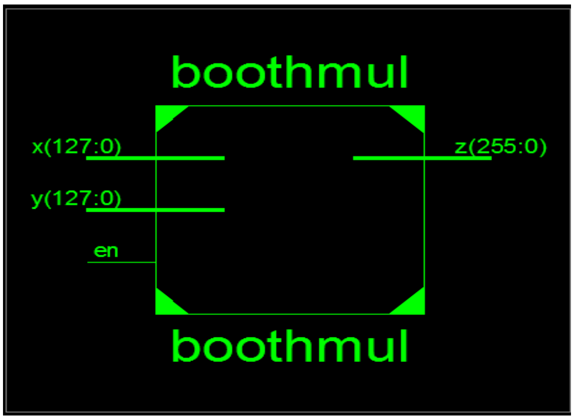
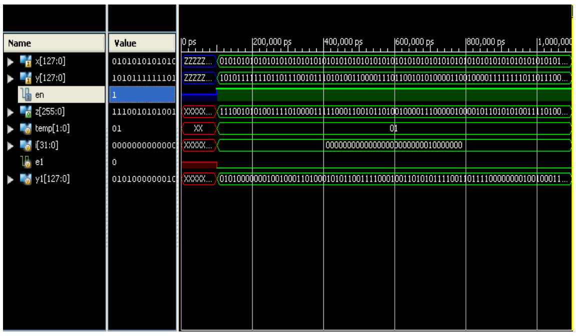


Fig3: RTL view of 128- bit booth multiplier

Here the fig:3 shows the RTL schematic of the design synthesized for 128-bit booth multiplier. In the above RTL schematic of the 128-bit booth multiplier there are two inputs X and Y where each one of them is 128bit and the output is Z which is of 256 –bits this is about the RTL schematic of the 128-bit booth multiplier.

Result and Stimulation for 128-bit booth multiplier



```
Name         Value            0 ps  200,000 ps  400,000 ps  600,000 ps  800,000 ps  1,000,000
x[127:0]     0101010101010    ZZZZZ...  0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101...
y[127:0]     1010111111101    ZZZZZ...  1010111111101101110010111010100110000111011001010100001100100001111111101101011100...
en           1
z[255:0]     1110010101001    XXXXX...  1110010101001111010000111110001100101101001000011100000100001011010101001111010100...
temp[1:0]    01               XX  01
i[31:0]      0000000000000    XXXXX...  00000000000000000000000010000000
e1           0
y1[127:0]    0101000000010    XXXXX...  0101000000010010001101000101011001111000100110101011110011011110000000010010001 1...
```

Fig4: output waveform of the 128-bit booth multiplier

The fig:4 shows the output waveform for the 128-bit booth multiplier designed. Verilog code for the 128-bit booth multipliers being synthesized using Xilinx ISE 13.1 version of the device XC6XLX4. Here the inputs are X and Y which are of 128- bits and Y1 is the 2's compliment input Y which is a signed input. Temp is used for naming a variable that stores a value temporarily in Program and I is an integer. Z is the output which is of 256-bits.

RTL view for 256-bit booth multiplier

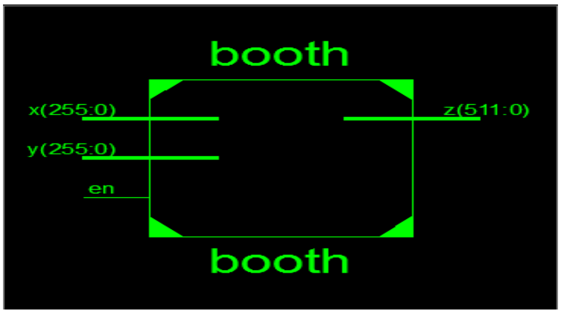


Fig5: RTL view of 128- bit booth multiplier

In this paper,64-bit, 128-bit and 256-bit booth multiplier were carried out for DSP and graphics applications.256-bit booth multiplier takes less delay but more area than that of 64-bit multiplier, Less power is consumed by 256-bit booth multiplier compared to 64-bit multiplier. And also from this it is clear that 64-bit booth multiplier less number of transistors compared to 256-bit multiplier. The selection of booth multiplier depends upon requirement applications. The results of booth multiplier vary according to the change in technique, methodology and hardware logics.

There is a continuous research undertaken on different radix multiplier, depending on the impacts the result to a larger extent. There is a scope of

Here the fig:5 shows the RTL schematic of the design synthesized for 256-bit booth multiplier. In the above RTL schematic of the 256-bit booth multiplier there are two inputs X and Y where each one of them is 256-bit and the output is Z which is of 512-bits this is about the RTL schematic of the 256-bit booth multiplier.

Result and Stimulation for 256-bit booth multiplier

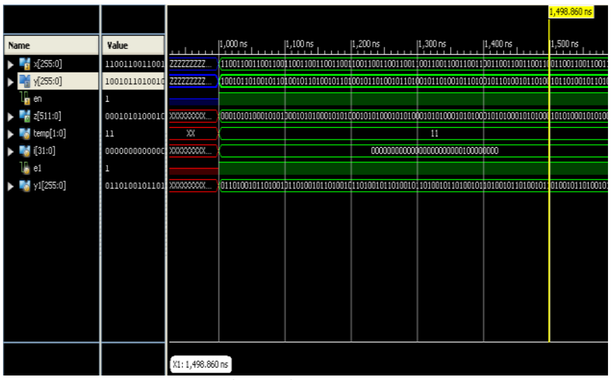


Fig6: output waveform of the 256-bit booth multiplier

The fig:6 shows the output waveform for the 256-bit booth multiplier designed. Verilog code for the 256-bit booth multipliers being synthesized using Xilinx ISE 13.1 version of the device XC6XLX4. Here the inputs are X and Y which are of 256- bits and Y1 is the 2's compliment input Y which is a signed input. Temp is used for naming a variable that stores a value temporarily in Program and I is an integer. Z is the output which is of 512-bits.

X. CONCLUSION

analysis to be carried out for the multiplication depending on the number of bits a number being multiplied or whether they are signed or unsigned numbers.

REFERENCES

[1] A B. PAWAR, "radix-2 vs radix-4 high speed multiplier", international journal of advanced research in computer science and software engineering, volume5, issue3, pp.329-333, March 2015.

[2] Jyothi kli et .al" A review of different methods of booth multiplier." International journal of engineering research and applications (IJERA), volume 7, 2017.

[3] Elisardoantelo et.al. Improved 64-bit radix -16 booth multiplier based on partial product array height reduction. "IEEE Transaction", 2016.

[4] Divyagovekr, AmeetAmonkar," Design and implementation of high-speed modified booth multiplier using Hybrid Adder", IEEE international conference on computing methodology andcommunication (ICCMC),2017.

[5] R. Balakumaran, E. Prabhu" Design of high-speed multiplier using modified booth algorithm with hybrid carry look-ahead adder", IEEE international conference on circuit, power and computing tehnologies,2016.

[6] Luo, Tao, et al."A racetrack memory based in memory booth multiplier for cryptography application. "Design Automation conference (ASP-DAC),2016 21st Asia and south pacific. IEEE,2016.

[7] G.Haridas, David et. al "Area Efficient Low Power Modified Booth Multiplier for FIR Filter." International Conference on emerging trends in engineering. Science and technology (ICETEST), volume 24,2016.

[8] BS, Diana et.al. "Modified Booth Multiplier with FIR Filter. "International Journal of Science and Research (IJSR)Volume-3,2014.

[9] Honglan Jiang, Fei Qiaoet.al" Approximate radix-8 Booth Multiplier for Low power and High-Performance Operation. IEEE Transaction",2015.

[10] Nagarjuna et.al." A Novel Architecture implementation of FIR Filter using booth Multiplier. "International journal of Industrial Electronics and Electrical Engineering, Volume-2,2014.

[11] Prasanna P Raj had published the paper on VLSI Design and analysis of multipliers for low power usage in September 2009.

[12] International Journal of VLSI design and communication system (VLSICS)volume.7,2016.

[13] International Journal of computer applications (0975-8887) Volume181-no.2, July 2018.

[14] International Journal of emerging technology and advanced engineering volume- 3, March 2013.

[15] International Journal for research in technological studies volume-5, February 2018.