International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 05 | May -2017 www.irjet.net p-ISSN: 2395-0072

Efficient FPGA Implementation of Radix 8 Partial Product Generator for FIR Filter and MAC Applications

Kalyani H N[1], H Umadevi[2]

[1] *M. Tech Student , Department of Electronics and Communication Engineering, Dr. Ambedkar institute of Technology,Bengaluru-560056*
[2] *Associate Professor, Department of Electronics and Communication Engineering, Dr. Ambedkar institute of Technology,Bengaluru-560056*

Abstract—*In this paper, we present a technique to reduce the maximum height of the partial product rows generated by a Radix-8 Modified Booth Encoded multiplier, without any rise in the area and delay of the partial product creation Block. This technique is of particular interest in all multiplier designs, but especially in Arithmetic multipliers for high-performance ALU designs and processors. The proposed method is generalized and can be applicable to higher radix encoding. The Proposed partial product generator is used in FIR Filter Design and MAC Architecture. We examine the proposed design by comparison with Normal Booth Multiplier; the results based on a theoretical analysis and on synthesis result shows its efficiency in terms of area, delay and power. Simulation results shows that the proposed partial product generator based designs significantly reduces the area, delay and power consumption when the word size of each operand in the multiplier is 16 bits; The Proposed multipliers is done by Verilog HDL and Simulated by ModelSim 6.4 c and Synthesized by Xilinx tool.*

Keywords-Radix-8 Modified booth encoded multiplier, FIR Filter, MAC, Digital Signal Processing (DSP), Wallace tree.

1. INTRODUCTION

In applications like arithmetic units of microprocessors, Digital_Signal_Processing (DSP) and multimedia, computer arithmetic is widely used. Compare to adder and sub tractors, multipliers are more complex. The operating speed of DSP is determined by the speed of the multipliers. Multipliers are used in implementations of filters, discrete Fourier transforms, correlations and range measurement. To design high speed, low power and compact multipliers many algorithms and architecture have been proposed. There are three steps in normal binary (NB) multiplication by digital circuit. Partial products are generated on the first . In the second step, all partial products are added until two rows of partial products are remain. In the last , the 2 partial product rows are added by carry propagation adder. The performance of multipliers can be increased by decreasing the number of partial product rows and decreasing the delay in the adder part.

In this paper, the number of partial product rows is reduced by using Modified Booth Encoding method, and radix method is for further reduction. For both positive and negative binary number multiplication booth algorithm is extensively used. Here 16*16 bit radix-8 partial product generator is proposed using modified booth algorithm. Radix-8 results in reduced power dissipation and less area compare to radix-4 and by using Wallace tree adder we obtain even lesser delay. Finally the output of proposed multiplier is applied to FIR filter and MAC to show proposed partial product generator is outperform than other.

This paper is arranged as follow. The brief explanation about Modified booth encoding is presented in section 2. Section 3 presents proposed work. The simulation results and comparison table are presented in section 4. The proposed design is applied to a MAC unit and FIR filter to realize the applicability of proposed design. Section 6 concludes the paper.

2. MODIFIED BOOTH ENCODING SCHEME

Modified booth algorithm is the fastest signed multiplication algorithm. It is also called as bit pair recoding. Number of addition and subtraction is more in booth algorithm that is time consuming so modified booth recoding is preferred. It consists of following steps,

1. Consider 2 inputs X and Y.
2. Append a zero to the lsb of multiplier Y and group the bits according to the radix methods.
3. Denote each group as partial products and to complete the group add necessary bits to Y. (example of grouping is shown in figure1)
4. Denote partial products groups according to the radix-8 booth encoding table shown in bellow table1.
5. By applying radix-8 encoding on multiplicands, obtain partial products.
6. Arrange the partial products such that pp2 is placed under pp1 after leaving 3 places from lsb of pp1, pp3 is placed under pp1 after leaving 6 places from lsb of pp1 and so on. Remaining locations are filled with 0's.
7. Extend sign bits of all the partial products according to msb bits
8. Finally add all partial products by using high performance adder.

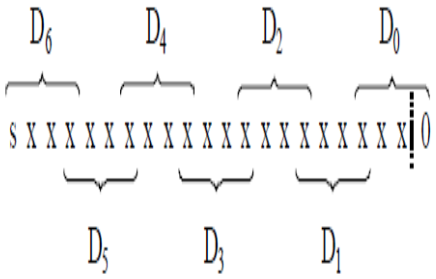

Figure 1 radix-8 grouping

Group of Multiplier bits	Operation to be perform on Multiplicand(A)
0000	0*A
0001	1*A
0010	1*A
0011	2*A
0100	2*A
0101	3*A
0110	3*A
0111	4*A
1000	-4*A
1001	-3*A
1010	-3*A
1011	-2*A
1100	-2*A
1101	-1*A
1110	-1*A
1111	0*A

Table 1 Radix 8 booth_encoding table

3. PROPOSED PARTIAL PRODUCT GENERATOR

The proposed system technique is a new ECW based modified partial product generator. The proposed Radix-8 partial product generator consists of four major blocks. They are Radix-8 booth encoding and decoding, partial product generator, ECW and adder. Radix-8 encoding block generates the encoding table shown in table1. Partial product generator block generates partial products and arrange them as shown in figure 3.Finally these partial products are added by adder. The proposed system is shown in figure 3

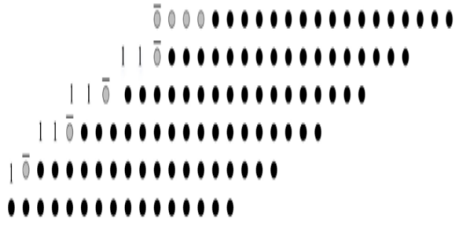

Figure 2 Partial product arrangements

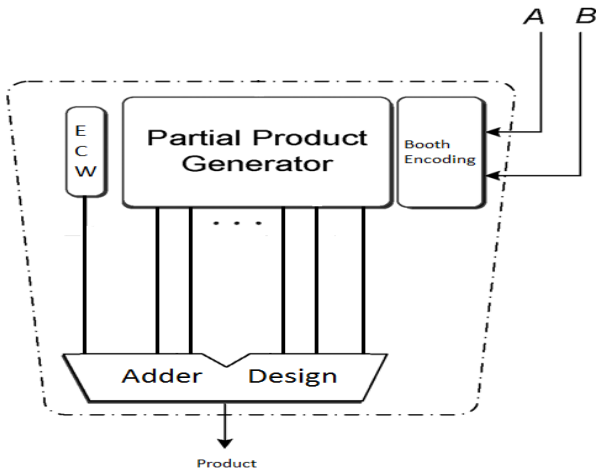

Figure 3 Proposed Radix-8 partial product generator

4. SIMULATION RESULTS AND ANALYSIS

The implementation of radix-8 partial product generator using Verilog code is done. The output of radix-8 partial product generator is shown in figure3.the simulation results contain the inputs A=0000000001010101=85, B=0000000011010011=211 and output P=17935.

The synthesis output of radix-8 partial product generator is shown in Table 2 and it is compare with existing design.

METHOD NAME	AREA			DELAY		
	LUT	SLICES	GATE	Over all Delay	Gate Delay	Path Delay
Approximate Booth Radix 8 16 Bit Multiplier	1167	608	29489	93.224ns	35.780ns 38.4% logic	57.444ns 61.6% route
Modified Proposed Radix 8 16 Bit Multiplier	1122	588	29171	87.451ns	33.935ns logic	53.516ns route

Table 2 Comparison

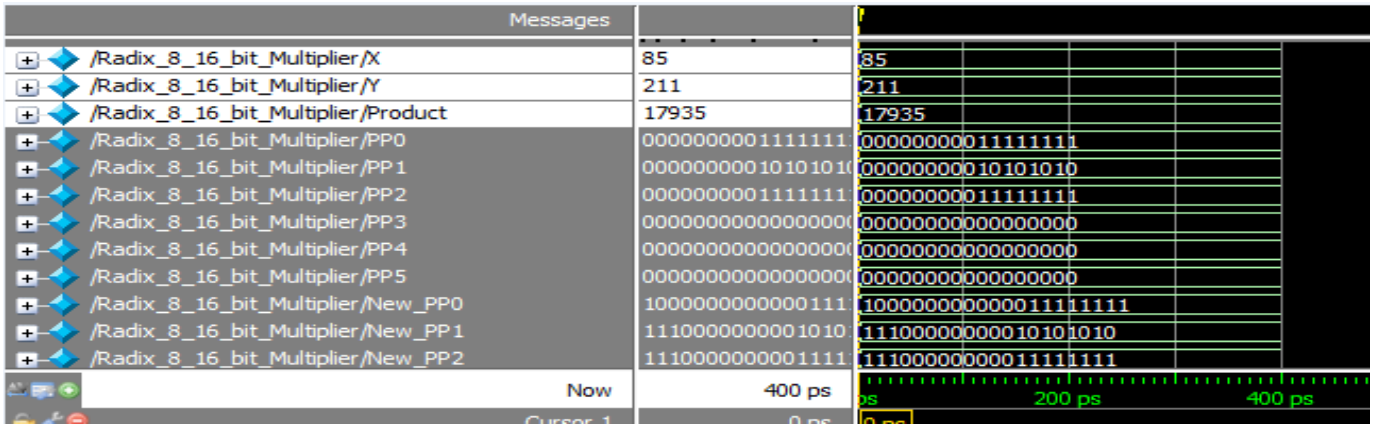

Figure 4 Proposed 16 bit partial product generator output

5. FIR FILTER AND MAC APPLICATION

In this section proposed radix-8 partial product generator is applied to a low_pass FIR filter and MAC uni to cheek the visibility of proposed design. FIR filter and MAC unit are designed using verilog code in Xilinx14.7 and simulated using ModelSim6.5c. The simulation results are shown in figure 4 and 5.

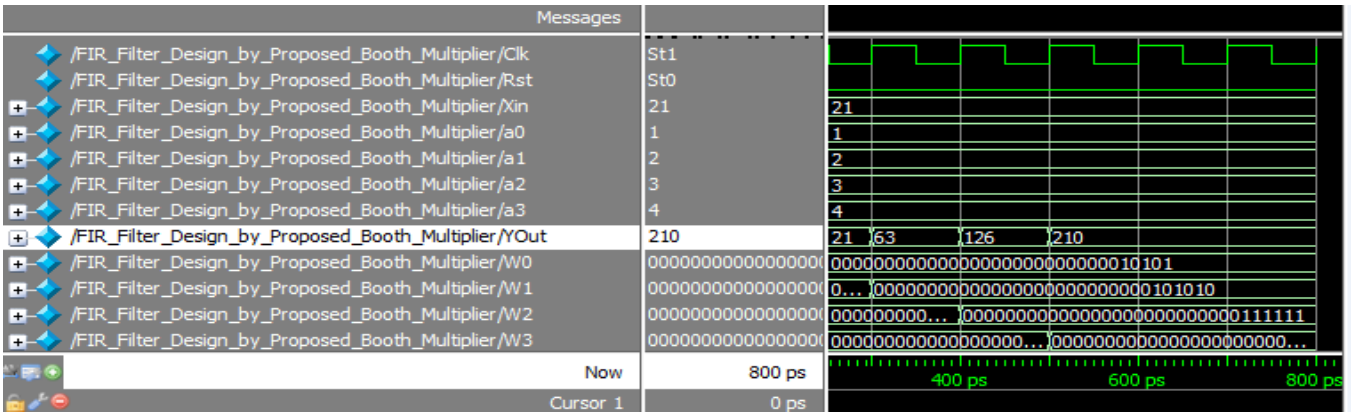

Figure 4 FIR filter output

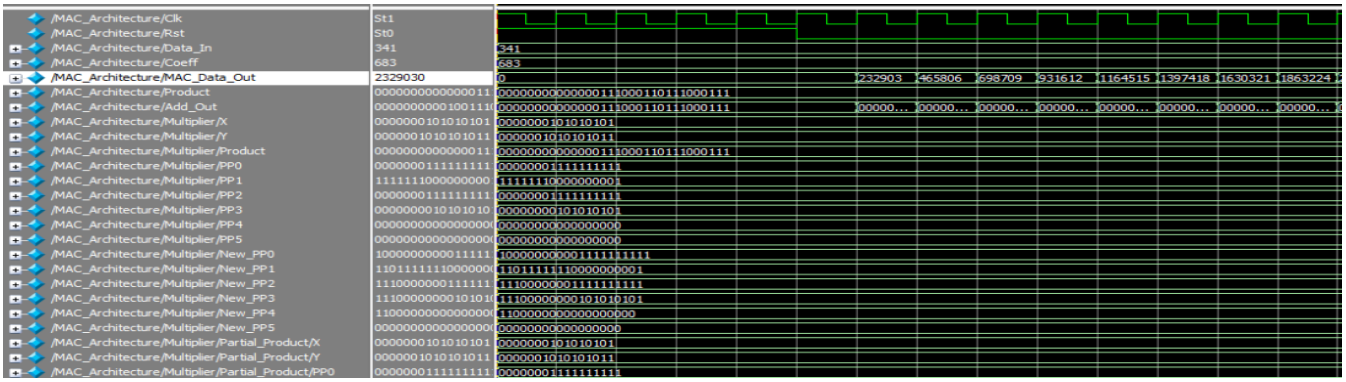

Figure 5 MAC unit output

6. CONCLUSION

In this paper, an efficient 8*8 and 16*16 bit radix8 partial product generator using modified_booth_algorithm is implemented and compare with existing normal booth multiplier which interns reduces both area and delay. The result shows the proposed design consumes less area delay.

In the proposed signed radix-8 partial product generator is implemented as FIR Filter and MAC Unit to examine applicability.

REFERENCES

[1] Y. Harata, Y. Nakamura, H. Nagase, M. Takigawa, and N. Takagi, "**A high speed multiplier using a redundant binary adder tree**," IEEE J. Solid-State Circuits, vol. SC-22, pp. 28-34, 1987.

[2] H. Edamatsu, T. Taniguchi, T. Nishiyama, and S. Kuninobu, "**A 33 MFLOPS floating point processor using redundant binary representation**," in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC), pp. 152–153, 1988.

[3]] H. Makino, Y. Nakase, H. Suzuki, H. Morinaka, H. Shinohara, and K. Makino, "**An 8.8-ns 54×54-bit multiplier with high speed redundant binary architecture,**" IEEE J. Solid-State Circuits, vol. 31, pp. 773-783, 1996.

[4]] Y. Kim, B. Song, J. Grosspietsch, and S. Gillig, "**A carry-free 54b×54b multiplier using equivalent bit conversion algorithm,**" IEEE J. Solid-State Circuits, vol. 36, pp. 1538–1545, 2001

[5] S. Kuang, J. Wang, and C. Guo, "**Modified Booth multiplier with a regular partial product array**," IEEE Trans. Circuits Syst. II, vol. 56, pp. 404-408, 2009.

[6] J. Kang and J. Gaudiot,"**A simple high-speed multiplier design**," IEEE Trans. Computers, vol. 55, pp.1253-1258, 2006.

[7] Y. He and C. Chang, "**A new redundant binary Booth encoding for fast 2 -bit multiplier design,**" IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, pp. 1192–1199, 2009.

[8] M. Ercegovac and T. Lang, "**Comments on 'a carry-free 54b×54b multiplier using equivalent bit conversion algorithm'**," IEEE J. Solid-State Circuits, vol. 38, pp. 160–161, 2003.