Implementation of Modified Booth Multiplier using Pipeline Technique on FPGA

Navdeep Kaur
Post Graduate Student
Department of Electronics and communication
Lovely Professional University Punjab, India

Rajeev Kumar Patial
Assistant Professor VLSI Design
Department Of Electronics and communication
Lovely Professional University Punjab, India

ABSTRACT
This paper presents 16×16 bit Radix-4 Modified Booth's Multiplier (MBM) optimized for high speed multiplication by using pipeline Technique. This paper aims at reduction of hardware utilization. This is accomplished by the use of 3:2 compressor adders. An efficient VHDL code has been written, successfully simulated on Modelsim 10.2 simulator and Xilinx 12.4 navigator is used for synthesizing the code. Simulation result shows the clock period of 2.689ns. The selected device to synthesize the code is xc3s500e-4pq208 of Sartan-3E family. The area utilization is shown as 222 numbers of slices and 383 numbers of LUTs.

General Terms: Multiplier, Pipelining Technique

Keywords: Compressor, Modified Booth Recoding, Pipelining, Radix-4, Xilinx navigator

1. INTRODUCTION
The multiplier is one of the key hardware blocks in most of the digital and high performance systems such as digital signal processors and microprocessors [1]. An increasing number of high-speed DSP applications have need of a high precision fixed- or floating-point multiplier suitable for VLSI implementation [2]. The multipliers are the better option for high-speed data processing. Various algorithms and architectures have been proposed to accelerate multiplication are Booth Algorithm [3], Modified Booth Algorithm [4], Braun and Baugh-Wooley. This paper enhances the performance of pipelined MBM. The computation of the multipliers manipulates two input data to generate many partial products for subsequent addition operations [5].

The multiplier can be divided into three stages: Partial products generation stage, partial products reduction stage, and the final addition stage. First stage comprises Radix-4 Booth encoder logic to generate partial products. To determine the speed of the overall multiplier, the second stage is the most important, as it is the most complicated stage. Here, Wallace tree which is composed of 3:2 compressor is used for partial products reduction which rapidly reduce the number of partial product rows to the final two (sums and carries). (3:2) compressor is the leaf cell of hierarchy. The last stage comprises of a simple adder, which adds the sum and carry output of the previous stage to provide the final result.

The pipelining is a popular technique to increase throughput of a high speed system, which divides total system into several small cascade stages and adds some registers to synchronize outputs of each stage [6]. As the number of stages increases, the power consumption and area gets increased. Thus, pipeline technique can be introduced in Wallace Tree in order to improve the performance. The block diagram of MBM using pipelining is shown in Figure 1.

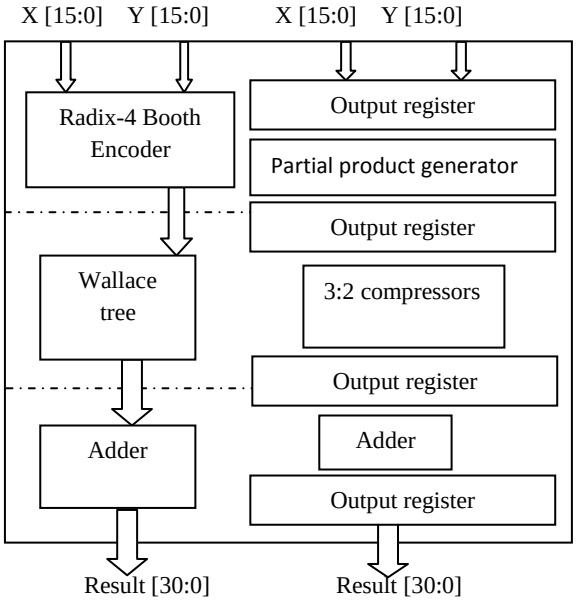

Fig 1: Block diagram of Proposed Pipelined Modified Booth Multiplier

The remainder of this paper is organized as follows: Section 2 deals with Partial Product Generation, Section 3 deals with 3:2 compressors used in Wallace tree structure, final adder that is for addition of last sum and carry bits is discussed in section 4, Section 5 gives the result analysis of the simulated modules and demonstrates the efficiency of the designed multiplier in terms of delay and area. Finally a concluding remark is given in Section 6.

2. PARTIAL PRODUCT GENERATION
Radix-2 Booth algorithm does not work well when the multiplier has isolated ones. In such case the recorded multiplier has more number of one's when compared to the actual multiplier. So we group 3 bits for finding the recorded multiplier which will help to overcome the above said disadvantage. To multiply X by Y, the Radix 4 Booth algorithm starts from grouping Y by three bits and encoding into one of {-2, -1, 0, 1, 2} [7]. Block Diagram of Modified Booth Encoder as shown in Figure 2. The Table 1 shows rules to generate the encoded signals by Modified Booth recoding scheme [8].

In radix-4 Booth Algorithm, multiplier operand Y is partitioned into 8 groups having each group of 3 bits. In first group, first bit is taken zero and other bits are least Significant two bit of multiplier operand. In second group, first bit is most significant bit of first group and other bits are next two bit of multiplier operand. In third group, first bit is most significant bit of second group and other bits are next two bits of multiplier operand. This process is carried on. For each group,

Partial product is generated using multiplicand operand X. For n-bit multiplier, there are n/2 or n/2+1 groups are formed [9], which generate the partial products as PP [n+m-1], where n, m are the number of bits in multiplier and multiplicand respectively. It is based on encoding the two's complement multiplier where Y can be represented as:

$$Y = \sum (-2Y_{2i+1}+Y_{2i}+ Y_{2i-1})2^{2i}; \; 0 \le i \le n-2 \qquad (2.1)$$

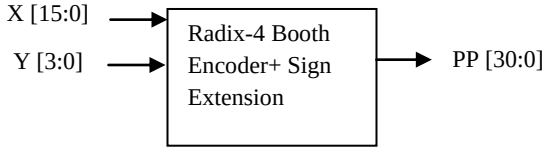

Fig 2: Block Diagram of Modified Booth Encoder

Table1: Radix-4 Modified Booth recoding table

Y_{2i+1}	Y_{2i}	Y_{2i-1}	Recorded Digits	Operand Multiplication
0	0	0	0	0×Multiplicand
0	0	1	+1	+1×Multiplicand
0	1	0	+1	+1×Multiplicand
0	1	1	+2	+2×Multiplicand
1	0	0	-2	-2×Multiplicand
1	0	1	-1	+1×Multiplicand
1	1	0	-1	-1×Multiplicand
1	1	1	0	0×Multiplicand

3. WALLACE TREE STRUCTURE

The Wallace tree shows a good performance by using the carry save adders instead of the ripple carry adders. It is, however, still the most critical part of the multiplier because it is responsible for the largest amount of computation [10]. In the proposed architecture, 3:2 compressors are used for realizing the reduction in the number of partial product addition stages.

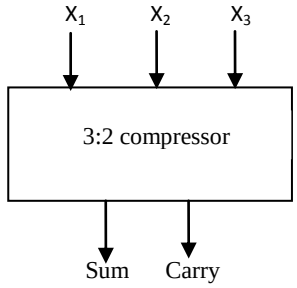

Figure 3: A 3:2 Compressor

A 3-2 compressor takes 3 inputs X1, X2, X3 and generates 2 outputs, the sum bit 's', and the carry bit 'c' as shown in Figure 3 [11]. In this A+B+C+D= (A+B) + (C+D). A, B, C and D are added in parallel and then they are added together. They require only two full adder delays where as A+B+C+D requires three full adder delays. This is shown in Figure 4 [12].

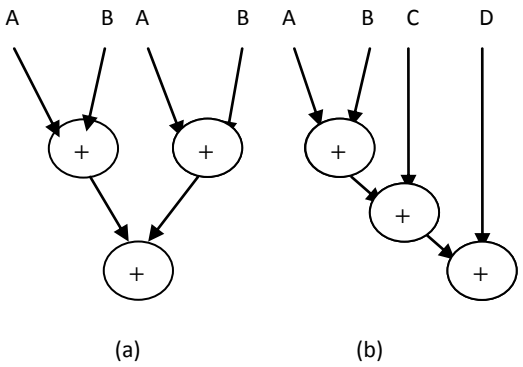

Fig 4: (a) Two adder delay level and (b) Three adder delay level

The 3-2 compressor can be employed as a full adder cell when the third input is considered as the Carry input from the previous compressor block or X3 = C_{in}.

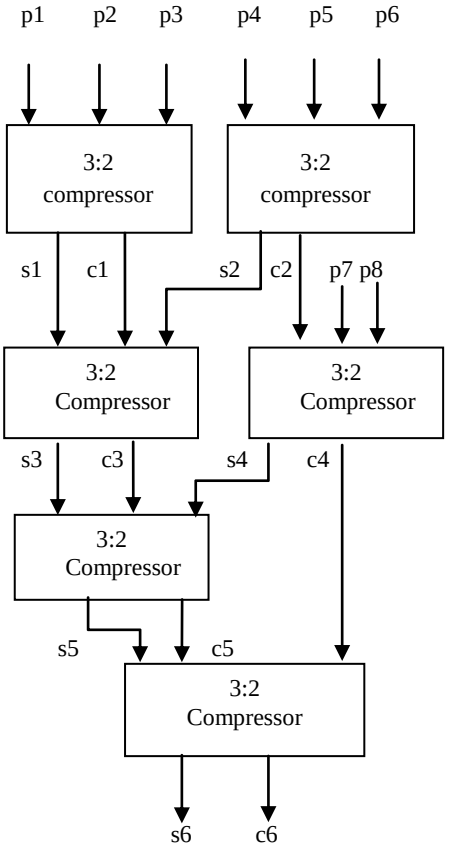

Fig 5: Wallace tree for eight partial products

3:2 compressors are the key microcell used in the arithmetic multiplication [13]. The equations governing the existing 3-2 compressor outputs are shown below

$$\text{Sum} = x1 \oplus x2 \oplus x3 \qquad (3.1)$$

$$\text{Carry} = (x1 \oplus x2)\, x3 + (\overline{x1 \oplus x2})\, x1 \qquad (3.2)$$

The 3:2 compressor structures for eight partial products are shown in Figure 5. The first row compresses partial products as p1, p2, p3 p4, p5 and p6 by the use of two compressor blocks to produce sum (s1, s2) and carry (c1,c2). The second row adds the s1, c1 and s2 also the remaining partial products p7, p8 get adds with c2. At last number of partial product

reduce to final sum and carry. This approach is much faster approach used for Wallace multiplier. The delay of the basic pipelined stages of multiplier is reduced by applying 3-stage pipelines in the Wallace tree. The delay of the critical path is again reduced. The 3:2 compressors make use of a carry save adder. In carry save adder, the carry digit is taken from the right and passed to the left, just as in conventional addition; but the digit carry is passed to the left which is the result of the previous calculation and not the current one [14]. The number of levels in the Wallace tree using 3:2 compressors can be approximately given as

$$\text{Number of levels} = \frac{\text{Log } (k/2)}{\text{Log } (3/2)} \quad (3)$$

Where, k is the number of partial products

4. FINAL STAGE ADDITION

The final stage in the Wallace tree multiplier for addition of partial products can be further reduced by the use of tree adders. This stage is also crucial for any multiplier because in this stage addition of large size operands is performed so in this stage fast carry propagate adders like Carry-look Ahead Adder or Carry Skip Adder or Carry Select Adder can be used as per requirement [15]. Instead of using carry select adder which reduce the delay but increase the complexity, in present architecture, the 32 bits of the sum and carry operands with sign extension get added with simple addition operation for binary arithmetic. For implementation of simple addition operation need to declare library during coding.

5. IMPLEMENTATION RESULT

This section describes the simulation results of Modified Booth Multiplier circuit in VHDL using Wallace tree approach and pipelining technique are shown in figure 6 and 7. The 12.4 navigator is used for synthesizing the code. The code is synthesized using Spartan3E family with device selected as xc3s500e-4pq208. As compressors form the essential requirement of high speed multipliers. Table 2 shows the utilization of hardware by the given MBM with pipelining.

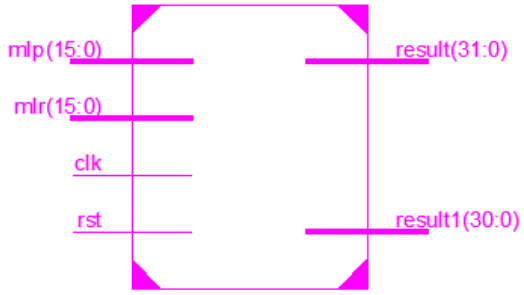

Fig 6: RTL view of pipelined Radix-4 MBM

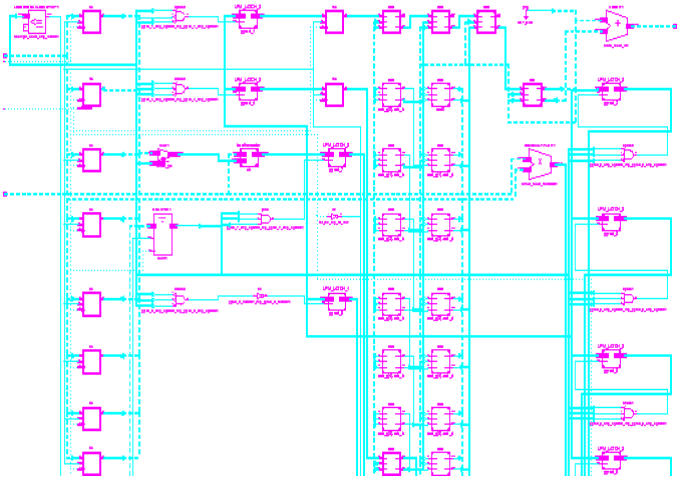

Fig 7: Internal Schematic view of MBM

The results are for the signed and unsigned multiplication of 16-bit multiplicand and 16-bit multiplier depending upon partial product generation by Radix-4 Booth encoder Logic, partial product reduction by 3:2 compressors and final adder addition. The choice of optimum multiplier involves three key factors: Area, propagation delay, reconfiguration time. The MBM reduce the partial products to half to provide the speed advantage. The primary source of propagation delay in circuit is the adder, so the 3:2 compressor used for Wallace tree to add the partial products, the layers of 3:2 compressors is used to decrease the propagation delay is formulated as Wallace Tree [10].The area utilization depends upon the number of LUT's (Look Up Table) and SLICES used for synthesizing the code. The minimum clock period of 2.689ns is produced and the clock frequency is 371.8MHz. The pipelined organization requires sophisticated complication techniques for modem processors [16].

Table 2: Hardware Utilization by MBM

Logic Utilization	Used	Utilization
Number of Slices	222	4%
Number of Slice Flip Flops	306	3%
Number of 4 input LUTs	383	4%
Number of bonded IOBs	97	41%
Number of GCLKs	1	4%
Number of MULT18X18SIOs	1	5%
Clock period	2.689ns	-
Clock frequency	371.8MHz	-

The given MBM is able to test the pipelining approach in order to reduce the critical path. In order to test the performance of MBM a comparative analysis is done as described in Table 3.

Table 3: Comparison

Logic Utilization	MBM using CLA [12]	MBM using CSA [12]	Proposed MBM	% reduction	
Number of slices	394	377	222	- 43.6%	-41%
Number of LUTs	794	718	383	-51.7%	-46.6%
Delay calculation	51.92 ns	22.38 ns	26.89ns	- 48%	+20%

It is clear from above comparative analysis that hardware utilization in terms of number of slices of proposed MBM is 43.6% less as compare to MBM using CLA and 41% less as compare to MBM using CSA, similarly in terms of LUTs 51.7% and 46.6% less. The calculated delay is as 26.89ns which is 48% less as compare to MBM with CLA and 20% more when compared with CSA by introducing pipelining in the Booth Multiplier and Wallace Tree.

6. CONCLUSION

The pipelining is the most widely used technique to improve the performance of digital circuits. In this paper, a 16 × 16 bit Radix-4 Modified Booth Multiplier with pipelining technique is designed. Both the delay time and area of MBM is found to be 26.89ns and 222 slices, 383 LUTs respectively. The delay of proposed MBM is 48% less as in case of MBM with CLA and 20% more than MBM with CSA. There is a tradeoff existing between proposed and other two in term of delay. The present code is synthesizable on Spartan-3E efficiently. The simulation results prove that code is efficient in terms of delay and area. This approach will be well suited for multiplication of numbers with more than 16-bit size for high speed applications. Modified Booth multiplier using Radix-4 Modified Booth algorithm and Wallace tree using compressors is very a good technique for high speed applications and its implementation with different logics in VLSI. Further work can be extended for optimization of the given multiplier in terms of both factors as area and delay.

7. ACKNOWLEDGMENTS

The authors would like to thank the generous support of the faculty of Electronics Department of Lovely Professional University. The authors are grateful to the university to provide great opportunity to work in VLSI design field and permission to publish this paper.

8. REFERENCES

[1] Wen-Chang Yeh and Chein-Wei Jen, "High-speed Booth encoded parallel multiplier design", IEEE Transaction on Computers, vol. 49, pp. 692-701, July 2000.

[2] A.R. Cooper , " Parallel architecture modified Booth multiplier" IEEE Proceedings, Vol.135, Pt. G, No. 3, June 1988.

[3] A. D. Booth, "A signed binary multiplication technique", Quarterly J. Mechanical and Applied Math, vol. 4, pp. 236-240, 1951.

[4] O. L. Mac-Sorley, "High Speed Arithmetic in Binary Computers", Proceedings of IRE, Vol.49, No. 1, January, 1961.

[5] D. Jackuline Moni, P. Eben Sophia, "Design of low power and high speed Configurable Booth Multiplier" IEEE Transaction, 2011, 978-1-4244-8679.

[6] Hwang-Cherng Chow and I-Chyn Wey, "A 3.3V 1GHz high speed pipelined Booth multiplier", Proceedings of IEEE ISCAS, vol. 1, pp. 457-460.,May 2002 .

[7] Soojin Kim and Kyeongsoon Cho., "Design of High speed Modified Booth Multipliers Operating at GHz Ranges", World Academy of Science, Engineering and Technology, 2010.

[8] Chetan Gupta, "Design and implementation of a 32 bit MAC unit with pipelined variable stage Carry Select Adder" Electronics Dept., Thapar University, June 2012.

[9] 'Modified Booth Multiplier' Digital Electronics Project 2 in 2008.

[10] J. Fadavi-Ardekani, "M × N booth encoded multiplier generator using optimized Wallace trees", IEEE Transaction on Very Large Scale Integration (VLSI) System, vol. 1, pp. 120–125, 1993.

[11] S. F. Hsiao, M. R. Jiang, and J. S. Yeh, "Design of high speed low-power 3-2 counter and 4-2 compressor for fast multipliers," Electron. Lett, vol. 34, no. 4, pp. 341–343, 1998

[12] Kulvir Singh Research Scholar, Dilip Kumar, "Modified Booth Multiplier with Carry Select Adder using 3-stage Pipelining Technique" International Journal of Computer Applications (0975 – 8887) Volume 44– No14, April 2012.

[13] P. J.; De Michelli, G., "Circuit and Architecture Trade for High-Speed Multiplication", IEEE Journal Solid State Circuits, vol. 26, pp. 1184-1198, Sept. 1991.

[14] C. S.Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron Computers*, vol. EC-13, pp. 14–17, 1964.

[15] V. Oklobdzija, "High-Speed VLSI Arithmetic Units: Adders and Multipliers in Design of High-Performance Microprocessor Circuits", Book Chapter, Book edited by A Chandrakasan, IEEE Press, 2000.

[16] S. B. Tatapudi and J. G. Delgado-Frias, "Designing pipelined systems with a clock period Approaching pipeline register delay," Proceedings of IEEE MWSCAS, vol. 1, pp. 871-874, Aug. 2005.