High Performance Complex Number Multiplier Using Booth-Wallace Algorithm

Rizalafande Che Ismail and Razaidi Hussin

School of Microelectronic Engineering
Kolej Universiti Kejuruteraan Utara Malaysia
P.O Box 77, d/a Pejabat Pos Besar
01007 Kangar, Perlis, Malaysia
rizalafande@kukum.edu.my

Abstract **This paper presents the methods required to implement a high speed and high performance parallel complex number multiplier. The designs are structured using Radix-4 Modified Booth Algorithm and Wallace tree. These two techniques are employed to speed up the multiplication process as their capability to reduce partial products generation to η/2 and compress partial product term by a ratio of 3:2. Despite that, carry save-adders (CSA) is used to enhance the speed of addition process for the system. The system has been designed efficiently using VHDL codes for 16x16-bit signed numbers and successfully simulated and synthesized using ModelSim XE II 5.8c and Xilinx ÌSE 6.1i. As a proof of concept, the system is implemented on Xilinx Virtex-II Pro FPGA board.**

I. INTRODUCTION

Complex number operations are the backbone of many Digital Signal Processing (DSP) algorithms especially for multimedia applications such as 3D graphics which mostly depend on extensive numbers of multiplications. Besides that, they are time critical components for radar, satellite and digital modulation applications too.

Complex number multiplication needs to be done using four real number multiplications and two additions. In real number processing, carry needs to propagate from the least significant bit (LSB) to the most significant bit (MSB) when binary partial products are added. Therefore, the addition and subtraction after binary multiplications limit the overall speed, although many techniques have been proposed to overcome the issue.

This paper presents the design method and efficient implementation of complex number multiplier by integrating Radix-4 Modified Booth algorithm [1],[2],[3] and Wallace tree structure [4],[5],[6],[7] for generating the partial product rows as well as for performing the addition process respectively. Furthermore, in enhancing the speed of the addition process, carry save addition adders are used for the implementation [8],[9].

The paper is divided into 5 sections. In the following section (Section 2), we present the architecture of the system. In Section 3, the complex number multiplier algorithm is discussed. Simulation and synthesis details are presented in Section 4 and conclusions in Section 5.

II. ARCHITECTURE

The complex number multiplication system can be divided into two main components known as real part (R) and imaginary part (I).

$$R + jI = (A + jB)(C + jD) \quad \ldots\ldots\ldots (1)$$

Based on equation (1), the real part is the output for (AC - BD) and the imaginary part is the output for (BC + AD). Each of these two main components required sub-components called modified Booth encoding (MBE), partial product generator and adders/subtractor. Fig. 1 shows the block diagram of the complex number multiplier employed in this study.

Based on conventional methods and observation of the equation (1), four separate multiplications are required to produce the real part as well as imaginary part numbers which lead to the implementation of four different MBE. However, in this study, only two MBEs

0-7803-9731-2/06/$20.00 ©2006 IEEE

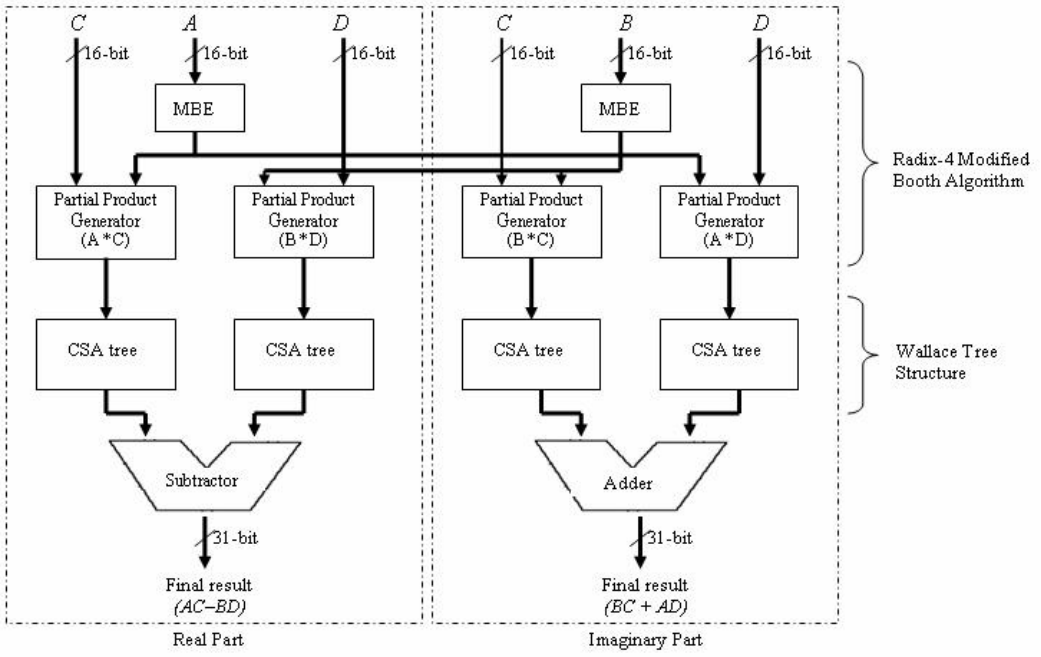


Fig. 1 Block diagram of the complex number multiplier.

are required to execute the multiplication process. It has been done in such a way that input A and B are always set to the multipliers and input C and D are always set to the multiplicand. By implementing these techniques, several logic gates can be removed.

III. Complex Number Multiplier Algorithm

For any complex number multiplier design, the most critical part is the multiplication process. In any multiplication operation, there are three major steps. For the first step, the partial products are generated. For the second step, the partial products are reduced to one row of final sums and carries. For the third step, the final sums and carries are added to generate the result.

In this paper, the first step is done using Radix-4 modified Booth algorithm because of its ability to cut the number of partial product rows by half. Then, Wallace tree structure is used for the second step to rapidly reduce the number of partial product rows to the final two (sums and carries). Finally, in a third step, carry propagate adders (CPA) is employed to add the final two rows resulting in final product.

A. Radix-4 Modified Booth Algorithm

The multiplication design based on Radix-4 modified Booth algorithm consists of two main blocks known as MBE and partial product generator as shown in Fig. 1.

MBE is an efficient encoding technique when it comes to reducing the partial product rows. The method of grouping the multiplier bits when using MBE technique is shown in Fig. 2. It is based on a window size of three bits and a stride of two [10]. The multiplier A and B are segmented into groups of three bits and each of the groups will produce the expected outputs as described in Table 1. The outputs are then connected to the partial product generator. As a result of using 16x16-bit signed numbers for this project, MBE will then generate eight partial product rows instead of sixteen rows traditionally.

There are many methods that have been discussed for generating partial product rows as reported in [5],[10],[11],[12]. The aim is to optimize the number of partial product rows as it will give significant effect to the speed, performance as well as area of the system. In this study, the proposed design methodology in generating partial product rows presented in [12] has been implemented. By using the technique, the partial product rows will no longer required two's complement error correction circuits and last negation signal as shown in Fig. 3. Hence, the multiplication can have a smaller critical path thus directly influences the speed of the multiplication process as well as the area of the circuit.

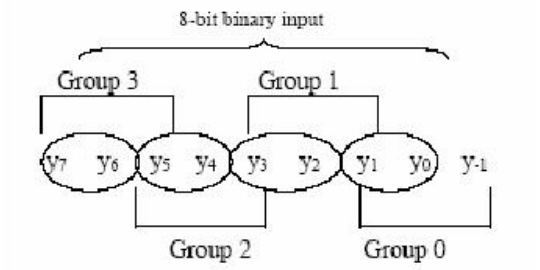


Fig. 2 Multiplier bits grouping according to Booth recording for 8-bit input [10].

y_{2i+1}	y_{2i}	y_{2i-1}	Generated Partial Products
0	0	0	0 * Multiplicand
0	0	1	1 * Multiplicand
0	1	0	1 * Multiplicand
0	1	1	2 * Multiplicand
1	0	0	-2 * Multiplicand
1	0	1	-1 * Multiplicand
1	1	0	-1 * Multiplicand
1	1	1	0 * Multiplicand

Table 1 Radix-4 Modified Booth Recording [13].

Fig. 3 The addition architecture of 16x16-bit signed multiplication [12].

B. WALLACE TREE STRUCTURE

The Wallace tree structure is one of various reduction tree design techniques available for fully parallel multiplier architectures. It is a method of combining either 3:2 or/and 2:2 CSA adders to add together seven or more numbers of size *k*-bit. These two adders basically serve as a normal full adder and half adder as shown in Fig. 4. Furthermore, by applying Wallace tree concept, the number of partial product rows can be decreased by a factor of 1.5. In this case, due to 16-bit signed numbers used in this project, there will be five levels of CSA trees required to complete the addition operation.

In executing the addition process using this methodology, the initial step is to rearrange the partial product bits to become like a diamond shape. This is to ensure that it can be easily grouped into either three or two bits. Based on Fig. 3, the partial product bits have been repositioned as shown in Fig. 5. The dot notations are used to represent the partial product bits. Fig. 6 shows steps by steps how the addition operation is performed using CSA adders according to Wallace tree structure.

The realisation of the final results for producing the real and imaginary part have been depicted using carry propagate adders.

IV. SIMULATION AND SYNTHESIS

The VHDL codes have been successfully simulated using ModelSim XE II (ver. 5.8c) and have been synthesized using Xilinx ÌSE 6.1i. As a proof of design concept, the complex number multiplier architecture has been implemented on Xilinx Virtex-II Pro FPGA development board and shown working properly. The target chip used is Xilinx Virtex-II Pro device number 2VP7FF672 with speed grade set to -6. The maximum operating frequency of the system is capable to achieve up to 648MHz and on its own consumed only 26% of total areas on that FPGA board.

V. CONCLUSION

In this paper, a high performance complex number multiplier for 16x16-bit signed numbers has been presented. It can be concluded that the use of Radix-4 modified Booth algorithm and Wallace tree structure lead to a better performance in throughput

speed and area. The utilisations of CSA adders in performing addition process for partial product rows also have been shown to greatly influence the speed of the system.

REFERENCES

[1] A. D. Booth, "A Signed Binary Multiplication Technique," *Quartely J. Mechanical and Applied Math.*, pp. 236-240, 1951.

[2] S. Sunder, "A Fast Multiplier Based On Modified-Booth Algortihm," *International Journal of Electronics*, vol. 75(22), pp. 199-208, 1993.

[3] J. H. J. Kenneth Lin, Na Tang, "A High-Performance 32-bit Parallel Multiplier Using Modified Booth Algorithm and Sign Deduction Algorithm," *5th International Conference on ASIC*, pp. 1281-1284, 2003.

[4] C. S. Wallace, "A Suggestion for a Fast Multiplier," *IEEE Transactions on Computers*, pp. 14-17, 1964.

[5] M. O. Lakshmanan, Alauddin Mohd Ali, "High Performance Parallel Multiplier Using Wallace-Booth Algorithm," *IEEE International Conference on Semiconductor Electronics*, pp. 433-436, 2002.

[6] K. F. Pang, "Architectures for Pipelined Wallace Tree Multiplier-Accumulators," *IEEE International Conference on Computer Design: VLSI in Computers and Processors*, pp. 247-250, 1990.

[7] Y. H. Niichi Itoh, "A 600 MHz 54x54-bit Multiplier with Rectangluar Styled Wallace Tree," *IEEE Journal of Solid State Circuits*, vol. 36, No. 2, pp. 249-257, 2001.

[8] T. K. William Jao, S. Tjiang, "Circuit Optimization Using Carry Save-Adder Cells," *IEEE Transactions on Computers-Aided Design of Integrated Circuits and Systems*, vol. 17, pp. 974-984, 1998.

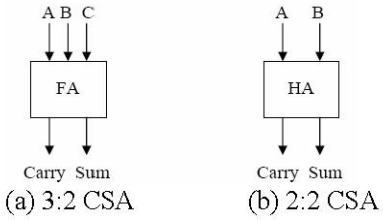
[9] W. J. T. Kim, Steve Tjiang, "Arithmetic Optimization using Carry Save Adders," *Proceeding of ACM*, pp. 433-438, 1998.

[10] J. L. G. J.Y. Kang, "A Fast and Well-Structured Multiplier," *EUROMICRO Systems on Digital System Design*, pp. 692-701, 2004.

[11] A. F. Sadiq M. Sait, Gerhard Beckoff, "A Novel Technique for Fast Multiplication," *IEEE Fourteenth Annual International Phoenix Conference on Computers and Communications*, vol. 7803-2492-7, pp. 109-114, 1995.

[12] B. P. R. Che Ismail, "Performance Enhancement and Reduced Area Parallel Multiplier," *IEEE National Symposium on Microelectronics (NSM2005)*, pp. 252 - 258, 2005.

[13] O. L. MacSorely, "High Speed Arithmetic in Binary Computation," *IEEE Proceedings*, vol. 49, pp. 67-91, 1990.



(a) 3:2 CSA (b) 2:2 CSA

Fig. 4 The block diagram of CSA.

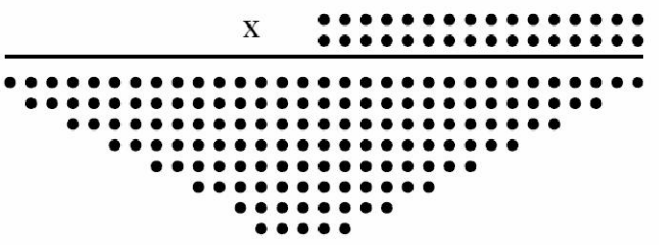
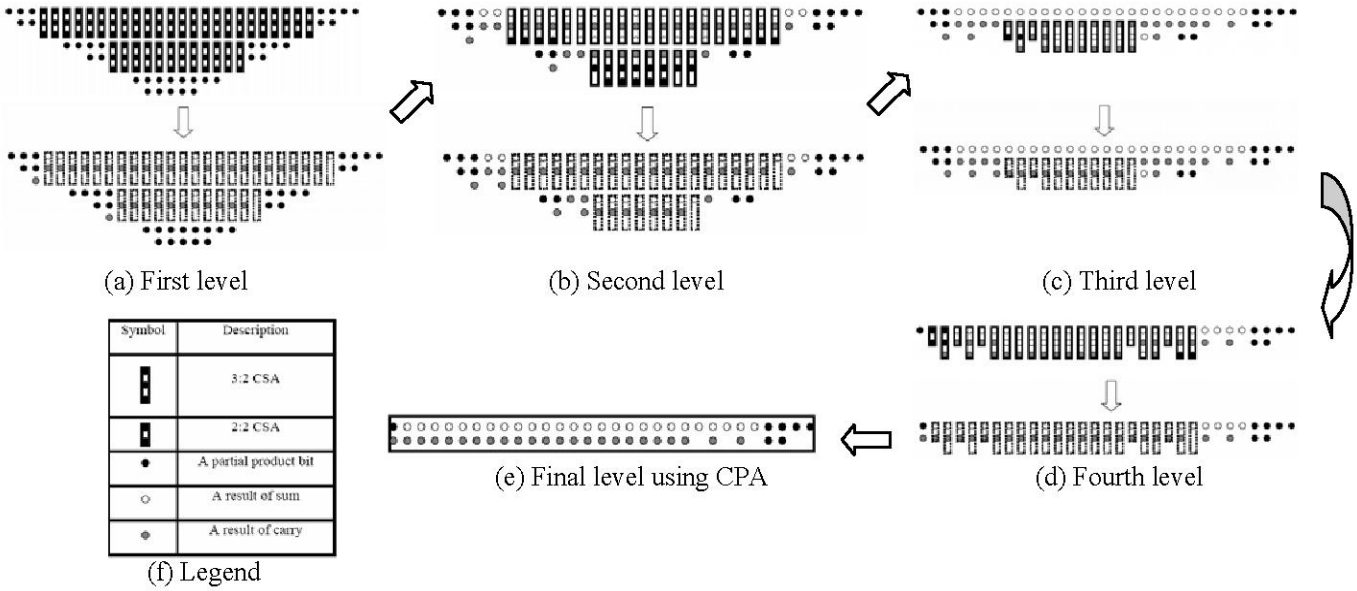


Fig. 5 The dot notations represent partial product bits.



(a) First level (b) Second level (c) Third level

(f) Legend (e) Final level using CPA (d) Fourth level

Fig. 6 Steps by steps the addition operation using CSA.