ZULFAQAR Journal of Defence Science, Engineering & Technology
e-ISSN: 2773-5281
Vol. 5, Issue 2 (2022)
DOI: https://doi.org/10.58247/jdset-2022-0502-10
Journal homepage: https://zulfaqarjdset.upnm.edu.my

IMPLEMENTATION OF MODIFIED BOOTH-WALLACE TREE MULTIPLIER IN FPGA

Anis Shahida Niza Mokhtar[a,b*], Nurlisa Zahari[c], Chew Sue Ping[a], Norlaili Ismail[a], Ahmad Sani Ismail[a]

[a] Department of Electrical & Electronic Engineering, Faculty of Engineering, National Defence University of Malaysia, Sungai Besi Camp, 57000 Kuala Lumpur
[b] Centre for Tropicalization, National Defence University of Malaysia, Sungai Besi Camp, 57000 Kuala Lumpur, Malaysia
[c] Maxis Tower, Kuala Lumpur City Centre, 50088 Kuala Lumpur, Malaysia

ARTICLE INFO

ARTICLE HISTORY
Received: 15-05-2022
Revised: 01-07-2022
Accepted: 01-10-2022
Published: 31-12-2022

KEYWORDS
Booth multiplier
FPGA
Modified booth multiplier
Wallace tree

ABSTRACT

The main purpose of this paper is to present the design and implementation of the Modified Booth Algorithm introduced by Andrew Donald Booth in 1950 and the Wallace tree structure is by an Australian Computer Scientist Chris Wallace in 1964. Combination of both algorithms is to implement a versatile algorithm widely used for digital signal processing application. Due to the highly demand on the fast microprocessors, designers come out with multiple techniques to produce a high-speed multiplier. Modified Booth's Algorithm have the advantages on faster multiplication process by reducing the generation of the partial products half of the number of bits of the multiplicand. The Wallace-tree multiplier itself giving a speed up in additional stage by reducing the adding partial products using the half adder and full adder instead of the long AND gate thus minimize the complexity of circuit. Combination of these two algorithms producing a new architecture of a high speed and low implementation area in one multiplier. This fulfils the requirement of high-speed computer system nowadays. The algorithm was developed using Verilog HDL in Quartus II software and the result obtained from Modelsim-Altera then the design is implemented in FPGA DE2 Cyclone II to verify the result.

1.0 INTRODUCTION

Multipliers arithmetic algorithm plays important role in the performance of digital signal processing algorithm [1]. Many types of multipliers designed to match the requirement of high-speed data processing [2]. Multiplication is basically an addition of the multiplicand itself to number of times of the multiplier generating levels of partial products. The critical path is determined more by the multiplier. Next, product is formed by the additional of the partial products. All multipliers would be having these three operation stages which is the generation of partial products, the additional of partial products and the final addition stage [3]. For this project, Modified Radix-4 Booth Algorithm as the multiplier giving full advantages in reducing the multiplication into half. A simple encoder is reduced from many operations in conventional multipliers compared to 4 operations of Radix-4 [4]. The speed of multiplication can be increased by reducing the number of partial products and accelerating the accumulation of partial products. From many multipliers design of implementing high speed parallel multipliers, Booth Algorithm and Wallace-tree structure are an efficient implementation of a high-speed parallel multiplier [5-7].

2.0 MODIFIED BOOTH ALGORITHM

Following the standard add-shift operation of multiplier, adding the multiplicand number of times to multiplier in partial products. Big number of multiplicands increase the total partial products to be added for those large multipliers. Booth Algorithm design will be reducing the number of multiplicand multiples.

*Corresponding Author | Mokhtar, A. S. N. | anis@upnm.edu.my

```
Multiplier                             0 1 0 1 0 1
Multiplicand                           0 0 1 0 1 0
                           ----------------------
                                       0 0 0 0 0 0
                                     0 1 0 1 0 1
                                   0 0 0 0 0 0
                                 0 1 0 1 0 1
                               0 0 0 0 0 0
                             0 0 0 0 0 0
                           ----------------------
                             0 0 0 1 1 0 1 0 0 1 0
```

Figure 1. Conventional multiplier operation

```
Multiplier                   0 1 0 1 0 1
Multiplicand                 0 0 1 0 1 0
                             1   1   1
               ------------------------
               0 0 0 0 0 0 0 0 1 0 1 0
               0 0 0 0 0 0 1 0 1 0
               0 0 0 0 1 0 1 0
               ------------------------
               0 0 0 0 1 1 0 1 0 0 1 0
```

Figure 2. Modified booth algorithm operation

A modification is made in Figure 2 from the Conventional Booth Multiplier from Figure 1. For the purpose in this paper focusing Modified Radix-4 Booth Algorithm by taking groups of three bits at a time. It starts with LSB. The first block comprises only two bits of the multiplier and it assumes zero for the third bit. For multiplier to compare with the rules of encoded signal in Table 1 to generate the partial products [8-9]. Booth re-coding encodes multiplier bits into [-2, 2].

Table 1. Modified Radix-4 encoding table

Multiplicand Bits, Y			Recorded Operation on Multiplier, X
Yi-1	Yi	Yi+1	
0	0	0	0X
0	0	1	+X
0	1	0	+X
0	1	1	+2X
1	0	0	-2X
1	0	1	-X
1	1	0	-X
1	1	1	0X

Radix-4 Booth algorithm is given below:

1. Extend the sign bit 1 position if needed to confirm that n is even.
2. Add on a 0 to the right of the LSB of the multiplier.
3. According to the value of each vector, each Partial Product will be 0, +X, –X, +2X or –2X.
4. The negative values of X are made by taking the 2's complement.

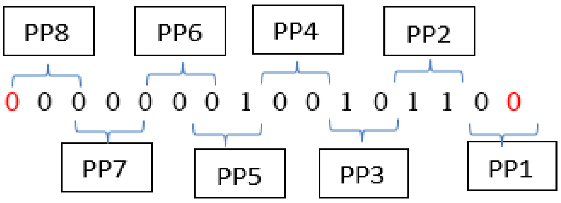


Figure 3. Radix-4 re-coding scheme

For Example:

Multiplier, X = 0000000000111100 (60)
Multiplicand, Y = 0000000100101100 (150)

PP1, PP4 = 100 encode -2X which equal to multiplier bits' times -2. This is obtained by shifting 1 bit of the multiplier bits and 2's compliment multiplier bits.

X = 0000000000111100
2X = 00000000001111000
-2X = 11111111110001000

PP2 = 011 +2X which equal to multiplier bits times 2. This is obtained by shifting 1 bit of the multiplier bits.

X = 0000000000111100
2X = 00000000001111000

PP3, PP5 = 010 encode +X which equal to multiplier bits times by 1. For this encode just take the value of the multiplier.

X = 0000000000111100

PP6, PP7, PP8 = 000 encode 0X which equal to multiplier bits times 0.

X = 0000000000111100
0X = 0000000000000000

After doing the operations and obtaining the partial products, the additions of partial products are done to obtain the answer.

```
Multiplier                   0000000000111100
Multiplicand                 0000000010010110
PP1                         11111111110001000
PP2                       00000000001111000
PP3                        0000000000111100
PP4                    11111111110001000
PP5                     0000000000111100
PP6                   0000000000000000
PP7                 0000000000000000
PP8               0000000000000000
Result           0000000000000000100001100101000
```

Figure 4. Modified Radix-4 calculation

In designing Modified Booth multiplier in this project are using 16 bits of multiplier and multiplicand. Which resulting in 16 partial products for conventional multiplier but reduced to half in this Modified Booth Algorithm leaving only 8 partial products to be added later.

3.0 WALLACE TREE STRUCTURE

The Wallace tree structure proposed by Wallace in 1964 [10-12]. The idea to speed up the additional stage by reducing the partial products to be added by taking a group of three rows of partial products. The additional of partial products generated from the Modified Booth Algorithm [1] can be added in sequence for standard additional operation [1]. This logic is easy to implement but it this method causing large delay of time. Thus, for this algorithm, the partial products are solved by using the Wallace tree structure method. The Wallace tree speed up the additional stage by reduced the level or partial product leaving two rows to be added on the last stage. The results of multiplications are from the addition of the last two rows of partial products.

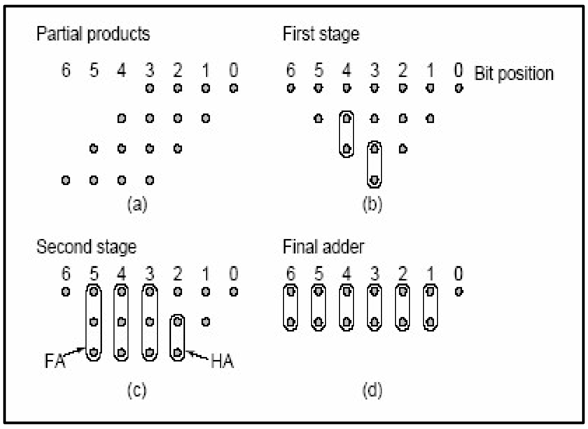


Figure 5. Additional steps of partial products in Wallace Tree

Figure 5 shows steps to implement Wallace tree structure for additional of partial products. First is the generation of the partial products which is from the Modified Booth Algorithm multiplier. Next the bits of partial products are reduced to rows by taking group of three rows to produce next rows of sum and products. Three bits' signal are passed to a one-bit full adder input Wallace Tree circuit, and the output signal is supplied to the next stage full adder of the same bit position producing sum and carry [13]. The new rows of sums and carries will be repeated until the last two rows of sum and carry left. Last step is the architecture of adders to sum up the last two rows for the final products [14].

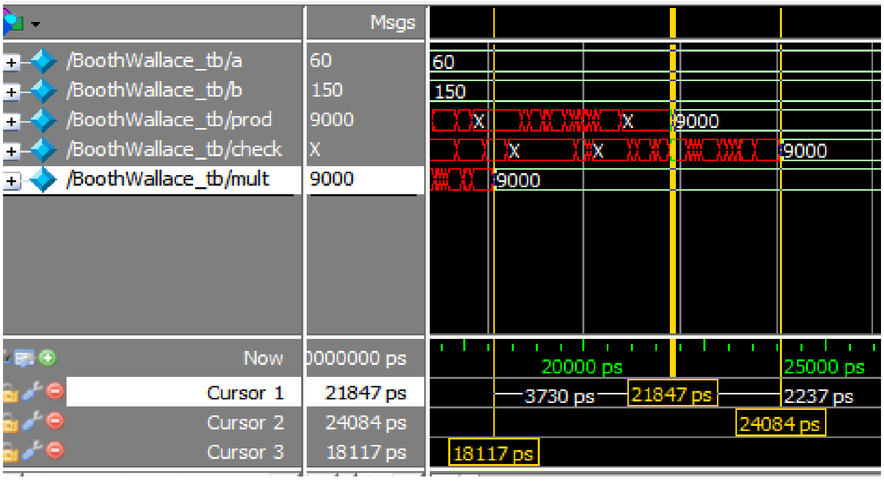
4.0 RESULTS AND ANALYSIS

The algorithm was developed using Verilog HDL in Quartus II and simulated in Modelsim-Altera.

Table 2. Calculation of modified Booth Wallace multiplier

Multiplier		0000000000111100
Multiplicand		0000000010010110
PP1		11111111110001000
PP2	1	00000000001111000
PP3		0000000000111100
PP4		11111111110001000
PP5	2	0000000000111100
PP6		0000000000000000
PP7	3	0000000000000000
PP8		0000000000000000
sum1		00011111110110101000
carry1		000000000111100 0000
sum2		00011111111101111000
carry2		0000000000010000000
sum3		00000000000000000
carry3		00000000000000000
sum4		00011111100010010000101000
carry4		000000001110111111000 0000
sum5		00000000000000010000000
carry5		0000000000000000000000
sum6		00011111011101101100101000
carry6		0000000010001001000000 0000
carry5		0000000000000000000000
prod_sum		000000111100110010011001 01000
prod_carry		0000000000100010010000000 000
result		0000000000000000100011001 01000

Figure 6 shows the simulation result for 60 x 150 by taking three types of multipliers using Modelsim-Altera to analyse the delays of the output.



Msgs

/BoothWallace_tb/a 60
/BoothWallace_tb/b 150
/BoothWallace_tb/prod 9000
/BoothWallace_tb/check X
/BoothWallace_tb/mult 9000

Now)000000 ps
Cursor 1 21847 ps
Cursor 2 24084 ps
Cursor 3 18117 ps

Figure 6. Simulation result from Modelsim

The result for simulation were recorded in Table 3 to compare the difference delay from the designed Modified Booth-Wallace with the conventional multiplier.

Table 3. Simulation result

Type of Multiplier	Delay (ps)	Difference delay (ps)
Conventional Multiplier *(mult)*	18117	-
Booth Multiplier *(check)*	24084	5697
Modified Booth Multiplier *(prod)*	21847	3730

Figure 6 and Table 3 showed the result when Modified Booth Wallace multiplier implement into FPGA DE2 board. 16 bits input binary used onto the switches and product of 32 bits' binary shown on 7 segments in Hexadecimal. Modified Booth-Wallace multiplier algorithm run using Quartus II software then simulated on Modelsim-Altera to see the delay. The delay results are taken to analyse the speed of the multiplier to generate output of the multiplication as shown in Figure 6.

Table 4. Comparison types of multipliers

	Conventional Multiplier [15]	Modified Booth Algorithm [16]	Wallace tree Structure [15]	Modified Booth-Wallace
Delay (ns)	61.39	24.23	36.350	18.23
No of LUTs	844	1292	1000	713

From the results of the simulation on Table 4, Modified Booth-Wallace multiplier had faster processing time compared to conventional multiplier, Modified Booth and Wallace tree structure itself. Modified Booth Algorithm multiplier reduced the number of partial products contributes to the less area of logic gates implement in circuit while Wallace structure lessen the circuitry structure of conventional adders thus both algorithm when combined producing one fast multiplier because of the less logic gates used in the multiplier to be implement in circuits [17-18].

5.0 CONCLUSION

The Modified Booth-Wallace multiplier generates n/2 partial products hence decreasing the number of partial products generated with Wallace structure reducing the additional process of partial products. Combination of both algorithms producing a high-speed multiplier with less area implementation on circuit. The simulation results and analysis are implemented using Altera Quartus II and Modelsim. The hardware implementation of the multiplier is implemented using FPGA DE2 Altera Cyclone II board. Taking the interest in a high speed and reduced area with power consumption multiplier, Modified Booth-Wallace algorithm was chosen as the fastest multiplier for Digital Signal Processing.

6.0 ACKNOWLEDGMENT

The authors fully acknowledged Ministry of Higher Education (MOHE) and National Defence University of Malaysia (NDUM) for this research feasible. The authors also would like to thank the technical staff in every institution involved for their assistant.

List of Reference

[1] Othman, M., & Ali, M. M. (2002, December). High performance parallel multiplier using Wallace-Booth algorithm. In ICONIP'02. Proceedings of the 9th International Conference on Neural Information Processing. Computational Intelligence for the E-Age (IEEE Cat. No. 02EX575) (pp. 433-436). IEEE.

[2] Chandrakasan, A., Bowhill, W. J., & Fox, F. (2001). HighSpeed VLSI Arithmetic Units: Adders and Multipliers.

[3] Asif, S., & Kong, Y. (2015, June). Performance analysis of Wallace and radix-4 Booth-Wallace multipliers. In 2015 Electronic System Level Synthesis Conference (ESLsyn) (pp. 17-22). IEEE.

[4] Kuang, S. R., Wang, J. P., & Guo, C. Y. (2009). Modified booth multipliers with a regular partial product array. IEEE Transactions on Circuits and Systems II: Express Briefs, 56(5), 404-408.

[5] Rao, M. J., & Dubey, S. (2012, December). A high speed and area efficient Booth recoded Wallace tree multiplier for fast arithmetic circuits. In 2012 Asia Pacific conference on postgraduate research in microelectronics and electronics (pp. 220-223). IEEE.

[6] Mokhtar, A. S. N., Reaz, M. B. I., Marufuzzaman, M., & Ali, M. A. M. (2013). Hardware implementation of a high speed inverse park transformation using CORDIC and PLL for FOC brushless servo drive. Elektronika ir Elektrotechnika, 19(3), 23-26.

[7] Mokhtar, A. S. N., Reaz, B. B. I., Maruffuzaman, M., & Ali, M. A. M. (2012). Inverse Park transformation using cordic and phase-locked loop. Rev. Roum. Sci. Techn.–Electrotechn. et Energ, 57(4), 422-431.

[8] Aparna, P. R., & Thomas, N. (2012, February). Design and implementation of a high performance multiplier using HDL. In 2012 International Conference on Computing, Communication and Applications (pp. 1-5). IEEE.

[9] MacSorley, O. L. (1961). High-speed arithmetic in binary computers. Proceedings of the IRE, 49(1), 67-91.

[10] Wallace, C. S. (1964). A suggestion for a fast multiplier. IEEE Transactions on electronic Computers, (1), 14-17.

[11] Mokhtar, A. S. N., Karim, S. A., Chew, S. P., Dardin, S. M. F. S. M., Supian, L. S., & Hashim, F. R. (2017). FPGA implementation of CORDIC algorithm in digital modulation. Journal of Fundamental and Applied Sciences, 9(3S), 279-293.

[12] Mohd Yusop, N. M., Samsudin, N., Mokhtar, A. S. N., Ahmad, S. R., Baharum, A, & Mohammad Amran, M. F., "Cube Polygon: A New Modified Euler Method to Improve Electric Circuit Efficiency," Zulfaqar Journal of Defence Science, Engineering & Technology, Vol. 3, No. 1, 2020.

[13] Cooper, A. R. (1988, June). Parallel architecture modified Booth multiplier. In IEE Proceedings G (Electronic Circuits and Systems) (Vol. 135, No. 3, pp. 125-128). IET Digital Library.

[14] Fadavi-Ardekani, J. (1993). M* N Booth encoded multiplier generator using optimized Wallace trees. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 1(2), 120-125.

[15] Ram, G. C., Rani, D. S., Balasaikesava, R., & Sindhuri, K. B. (2016, May). Design of delay efficient modified 16 bit Wallace multiplier. In 2016 IEEE international conference on recent trends in electronics, information & communication technology (RTEICT) (pp. 1887-1891). IEEE.

[16] Sharma, N., & Sindal, R. (2013). Modified booth multiplier using wallace structure and efficient carry select adder. International Journal of Computer Applications, 68(13).

[17] Daud, N. G. N., Hashim, F. R., Mustapha, M., & Badruddin, M. S. (2014, November). Hybrid modified booth encoded algorithm-carry save adder fast multiplier. In The 5th International Conference on Information and Communication Technology for The Muslim World (ICT4M) (pp. 1-6). IEEE.

[18] Shukri, F. A. A., Ali, F., Alias, A., & Nasir, N. A. A. M. (2021). Application of Fuzzy Analytical Hierarchy Process (FAHP) For Teaching Quality Evaluation at Defence Foundation Centre. ZULFAQAR Journal of Defence Science, Engineering & Technology, 4(2).