Modified Booth Dadda Multiplier using Carry Look ahead adder Design and Implementation

Shiwani Dod
M.Tech Research Scholar
Department of Electronics and Communication Engineering
Rayat and Bahra Institute of Engineering, Kharar, Punjab, India
Email: shiwithakur@gmail.com

***Abstract*—This paper proposes a novel 8X8 bit Modified Booth Dadda Multiplier architecture which is an improved version of Modified Booth Wallace Multiplier. The idea involves the generation of partial products using Modified Booth Algorithm. The addition of these partial products is done using Dadda Tree which is hierarchically divided into two levels. The proposed Modified Booth Dadda Multiplier provides significant reduction in area and complexity in comparison to Modified Booth Wallace Multiplier as Dadda Tree requires less number of half adders and full adders as compared to Wallace Tree. The proposed multiplier has lower ratio of power to area as when size of a multiplier decreases, the ratio of power to area also decreases, due to shorter interconnect lines and decreased glitching. Furthermore to improve the speed of addition at the third level of computation 4-bit Carry Look-Ahead Adder is used which provides better efficiency in terms of area/speed.**

Keywords: Wallace Tree, Dadda Tree, Carry Look-Ahead Adder, Modified Booth Dadda Multiplier, Modified Booth Wallace Multiplier, Modified Booth Encoder.

I. INTRODUCTION

Any multiplier can be divided into three stages i.e. partial products generation stage, partial products reduction stage and the final addition stage. In the first stage, the multiplicand and the multiplier are multiplied bit by bit to generate the partial products. In this stage, a Modified Booth Encoder Algorithm has been used to reduce the number of partial products to half and generate all the partial products in parallel. The second stage is the most important, as it is the most complicated and determines the speed of the overall multiplier. If speed is not an issue, the partial products can be added serially, reducing the design complexity. For high-speed design, the Wallace Tree construction method is usually used to add the partial products in a tree-like fashion in order to produce two rows of partial products that can be added in the last stage. Although fast, the Wallace Tree introduces other problems such as wasted layout area and increased complexity. In this research paper the Wallace Multiplier has been replaced with Dadda Multiplier which provides almost the same speed with reduction in size. In the last stage, the two-row outputs of the tree are added using high-speed cascaded 4-bit Carry Look-Ahead Adders to generate the product.

II. RESEARCH METHODOLOGY

Modified Radix-4 Booth Algorithm process three bits at a time during recoding. Recoding the multiplier in higher radix is a powerful way to speed up standard Booth multiplication algorithm. In each cycle a greater number of bits can be inspected and eliminated therefore, total number of cycles required to obtain products gets reduced. Number of bits inspected in radix r is given by $n = 1 + \log_2 r$ [1]. In case of Modified Booth Multiplier the number of partial products is reduced to half of the number of bits. This algorithm groups the original multiplier into groups of three consecutive digits where the outermost digit in each group is shared with the outermost digit of the adjacent group. Before grouping a zero is added to the LSB of the 8 bit multiplier. Each of these groups of three binary digits then corresponds to one of the numbers from the set {2, 1, 0, -1, -1}. Each encoded digit in the multiplier performs a certain operation on the multiplicand, X, as illustrated in Table 1.

Partial Product Generator (PPG) is the combination circuit of the product generator and the 5:1 MUX circuit. Product generator is designed to produce the product by multiplying the multiplicand X by 0, 1, -1, 2 or -2. A 5:1 MUX is designed to determine which product is chosen depending on the re-coded digits [2]. A PPG takes an 8-bit input and generates a 9-bit output. Sign Extension Correctors are designed to enhance the ability of the Booth Multiplier to multiply not only unsigned numbers but as well as signed numbers.

TABLE 1: RE-CODED DIGIT [4]

Block	Re-coded digit	Multiplication of X with
000	0	0
001	+1	+1
010	+1	+1
011	+2	+2
100	-2	-2
101	-1	-1
110	-1	-1
111	0	0

TABLE 2: OUTPUT OF PPG [5]

					1BE	1E	1E	1P8	1P7	1P6	1P5	1P4	1P3	1P2	1P1	1P0
				1	2BE	2P8	2P7	2P6	2P5	2P4	2P3	2P2	2P1	2P0		
		1	3BE	3P8	3P7	3P6	3P5	3P4	3P3	3P2	3P1	3P0				
1	4BE	4P8	4P7	4P6	4P5	4P4	4P3	4P2	4P1	4P0						

For 8X8 bit multiplier four partial products are generated. The output of PPG is illustrated in Table 2. In Table 2 each row represents a partial product generated from 5:1 MUX and 1E, 2E, 3E and 4E are sign extension bits with 1BE being complement of 1E.

For partial product reduction Wallace Tree or Dadda Tree Multipliers can be used. With Wallace Tree Multipliers, rows are grouped into sets of three during each reduction stage [3]. Within each three row set, full adders take in the three bits from a column and give two bits as output. The two bit output of a full adder is the sum and the carry bit. The sum bit is stored in the same column while the carry bit is stored in the column located on the immediate left of the full adder column. Half adders take in two bits from a column and generate a sum and a carry. The carry is stored in the column lying to the immediate left of the half adder column and the sum is stored in the same column. Rows that are not part of a three row set are transferred to the next stage without modification. The height of the matrix in the jth reduction stage, wj, is defined by the following recursive equations:

$$wo=N \text{ where N is the number of partial products}$$

$$wj+1=2.\ [wj\ /3]+wj \bmod 3 \text{ where } [\ ] \text{ represents the floor function}$$

For example, a 32X32 bit Wallace Multiplier uses eight reduction stages with matrix heights of 22, 15, 10, 7, 5, 4, 3 and 2 and for four rows the reduction stages would be with matrix heights 3 and 2. Figure 1 shows the Wallace Tree structure for 8X8 bit Modified Booth Wallace Multiplier. In Figure 1(c) and Figure 1(e) the tip of the arrow represents the sum generated from the full adder or half adder while the arrow points to the carry generated from the full adder or half adder. From Figure 1 it is evident that the number of full adders used in stage (1) is 9 and the number of half adders used is 4 while in stage (2) the number of full adders used is 6 and half adders is 7. The total number of full adders used in both the stages is 15 and the total number of half adders used in both the stages is 11.

In 1965, Dadda refined Wallace's method by proposing a unique placement strategy for the reduction stage full adders and half adders. Using Dadda's technique, the number of full adders and half adders is minimized, but the fast carry propagate adder is larger. For the reduction of the NXN partial product matrix, Dadda proposes a sequence of matrix heights that are determined by working back from the final two-row matrix. In order to implement the minimum number of reduction stages, the heights of each intermediate matrix is limited to the largest integer that is no more than 1.5 times the height of its successor. The number of reduction stages for Dadda and Wallace Multiplier is same. For example, a 32 X32 bit Dadda Multiplier uses 8 reduction stages with matrix heights of 28, 19, 13, 9, 6, 4, 3 and finally 2.

TABLE 3: NUMBER OF COMPONENTS REQUIRED IN NXN WALLACE AND DADDA MULTIPLIER [6]

Multiplier	No. of Full	No. of Half	CLA Length
DADDA	N2-4.N+3	N-1	2.N-2
WALLACE	N2-4.N+2+S	>N	2.N-1-S

Table 3 illustrates the number of full adders and half adders required in Dadda and Wallace Multipliers and the CLA length of NXN Dadda and Wallace Multipliers. In Table 3, N refers to the bits of the multiplier and S refers to the number of stages of the Wallace multiplier. The length of the Carry Look-Ahead Adder of Wallace Multiplier is less than the length of the Carry Look-Ahead Adder of Dadda Multiplier for S>1. However, the number of full adders required in Wallace Multiplier are more than those required in Dadda Multiplier for S>1. The number of half adders required in Dadda Multipliers is always less than those required in Wallace Multiplier irrespective of the value of S.

The Table 4 and Table 5 give the area of Wallace and Dadda Multipliers in 0.25 and 0.18 micron CMOS technology [1]. As shown in Table 4 and Table 5, Wallace Multipliers require from 4% to 7% more area than Dadda Multipliers for operand sizes from 8 to 64 bits. The increase in area of Wallace Multipliers is primarily due to the large number of half adders they require. As the operand size becomes larger, the percent increase in area of the Wallace Multipliers to the Dadda Multipliers also grows.

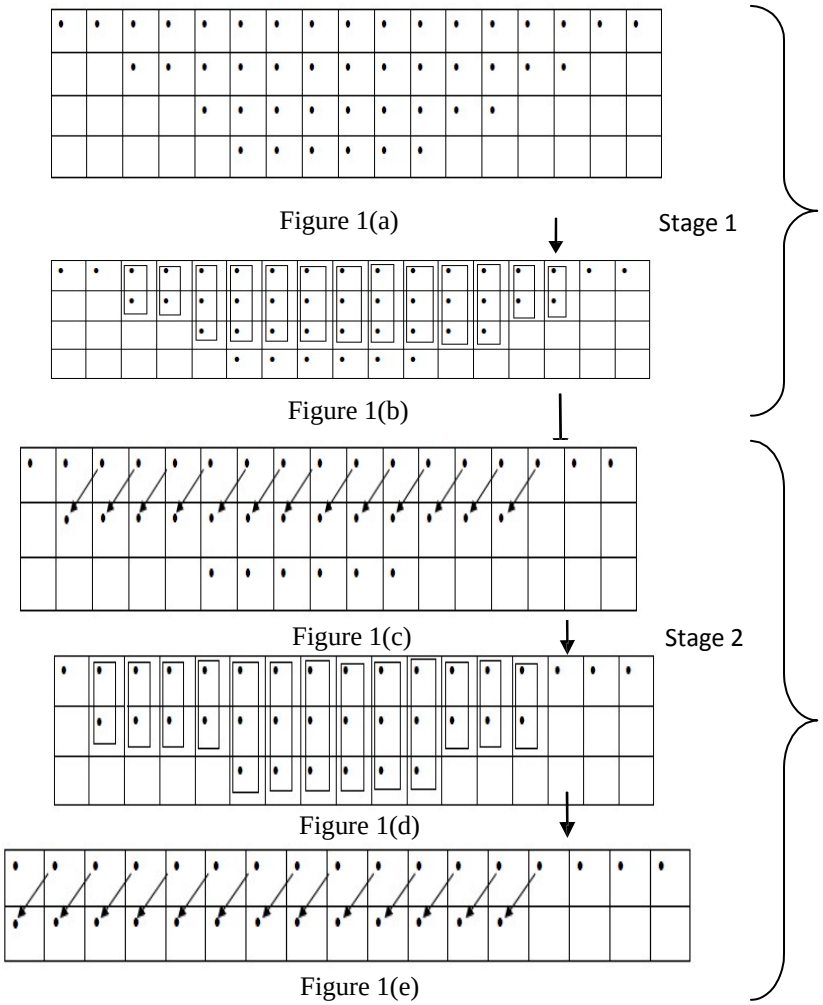

Figure 1: Dot diagram for partial product reduction using Wallace Tree (R. C. Ismail, et al., 2006), 1(a) Initial dot matrix 1(b) Grouping of rows 1(c) Generated 3 row matrix 1(d) Grouping of rows 1(e) Generated 2 row matrix

TABLE 4: AREA OF DADDA AND WALLACE MULTIPLIERS FOR 0.25 MICRON TECHNOLOGY [11]

Multiplier NXN	**Area(µm2)**		**Percent Increase**
	Dadda	***Wallace***	
8X8	19,252	20,087	4.3
16X16	67,133	70,703	5.3
32X32	243,088	257,673	6.0
64X64	973,399	1,038617	6.7

TABLE 5: AREA OF DADDA AND WALLACE MULTIPLIERS FOR 0.18 MICRON TECHNOLOGY [11]

Multiplier NXN	Area(μm2)		Percent Increase
	Dadda	*Wallace*	
8X8	9,017	9,377	4.0
16X16	31,980	33,574	5.0
32X32	121,313	129,132	6.4
64X64	472,973	506,081	7.0

Table 6 and Table 7 give the time delay estimates of Wallace and Dadda Tree Multipliers for 0.25 and 0.18 micron CMOS technology. Table 6 and Table 7 show that the delay estimates of Dadda and Wallace Tree Multiplier are approximately the same for lower values of N and for higher bits Dadda Multiplier takes marginally higher delay as compared to Wallace Multiplier and hence, Dadda Multiplier has the benefit of decrease in area for approximately same value of delay as compared to the Wallace Multiplier.

TABLE 6: DELAY ESTIMATE OF WALLACE AND DADDA MULTIPLIER FOR 0.25 MICRON TECHNOLOGY [11]

Multiplier	Delay (ns)	
NXN	*Dadda*	*Wallace*
8X8	1.8	1.8
16X16	2.7	2.8
32X32	3.5	3.5
64X64	5.1	5.0

TABLE 7: DELAY ESTIMATE OF WALLACE AND DADDA MULTIPLIER FOR 0.18 MICRON TECHNOLOGY [11]

Multiplier NXN	Delay(ns)	
	Dadda	*Wallace*
8X8	1.6	1.6
16X16	1.9	2.0
32X32	2.4	2.4
64X64	2.9	2.8

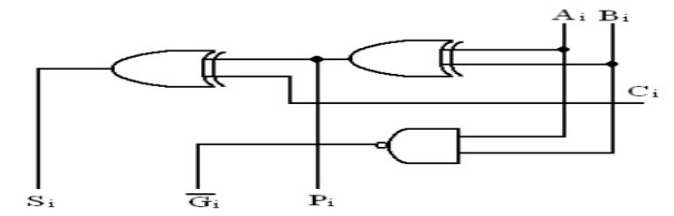

Figure 2(a)

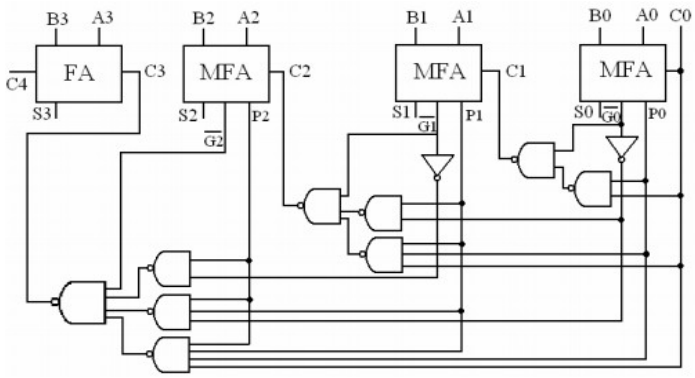

Figure 2(b)

Figure 2: 4-bit CLA used in Modified Booth Dadda Multiplier [2], 2(a) Modified Full Adder (MFA) 2(b) 4-bit CLA

For the final addition stage 4- bit Carry Look-Ahead Adder was used [3]. The structure of the 4-bit Carry Look-Ahead Adder is illustrated in Figure 2(a) and Figure 2(b). In the 4-bit Carry Look-Ahead Adder shown in Figure 2(b), the fourth MFA is replaced by a full adder to reduce the area (number of gates) without sacrificing

the speed improvement of the Modified Carry Look-Ahead Adder. The 4-bit CLA which is illustrated in Figure 2(b) is cascaded in series to design the higher bit CLA for the final addition stage of the Modified Booth Dadda Multiplier. The cascaded CLA is shown in Figure 3.

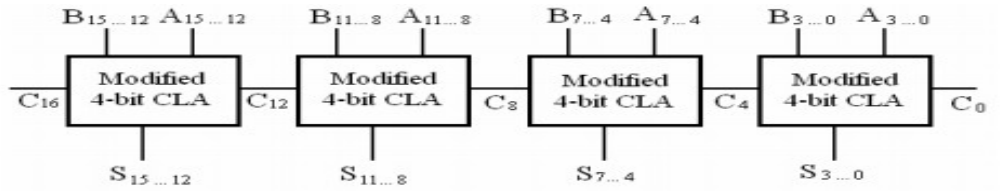

Figure 3: Cascading of the 4-bit CLA [12]

III. PROPOSED WORK

The dot diagram for the Dadda Multiplier used with Modified Booth Multiplier is shown in Figure 4. In Figure 4(c) and Figure 4(e) the tip of the arrow represents the sum generated from the full adder or half adder while the arrow points to the carry generated from the full adder or half adder. For 8X8 bit Modified Booth Multiplier the number of partial products generated is four which gives two stages with the first stage giving a three row matrix and final giving a two row matrix.

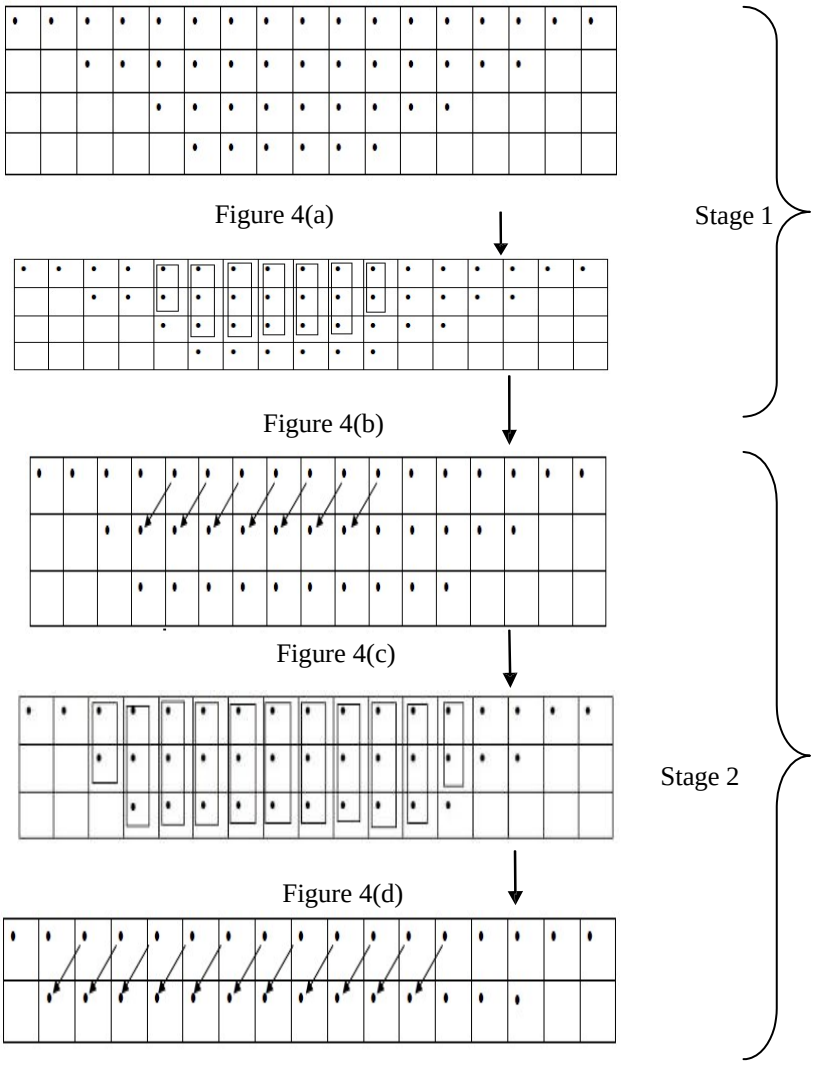

Figure 4: Dot diagram for partial product reduction using Dadda Tree

From Figure 4 it is evident that the number of full adders used in stage (1) is 5 and the number of half adders used is 2 while in stage (2) the number of full adders used is 9 and the number of half adders used is 2. The total number of half adders used in the two stages is 4 while the total number of full adders used is 14. In the proposed

Modified Booth Dadda Multiplier, the numbers of half adders have been reduced by 63.6% and the number of full adders has been reduced by 6.7% which leads to significant reduction in area.

For the final stage of 8X8 bit Modified Booth Dadda Multiplier 15-bit CLA is required while for Modified Booth Wallace Multiplier 13-bit CLA is required. There is an improvement in speed of the 15-bit CLA when implemented in the form of a 12- bit CLA cascaded with two half adders and one full adder as compared to the implementation of the 15-bit CLA in the form of four cascaded blocks of 4-bit CLA .The simulation results for the two Carry Look-Ahead Adders are shown in Figure 5 and Figure 6. The 15-bit CLA implemented in the form of a 12-bit CLA cascaded with two half adders and one full adder is illustrated in Figure 7. In Figure 7, ha stands for the column being compressed using half adders while fa stands for the column being compressed using full adders. The 12-bit CLA is made by cascading three 4-bit CLAs which are illustrated in Figure 2.

Figure 5: Simulation Results for 15-bit CLA implemented as shown in figure 7

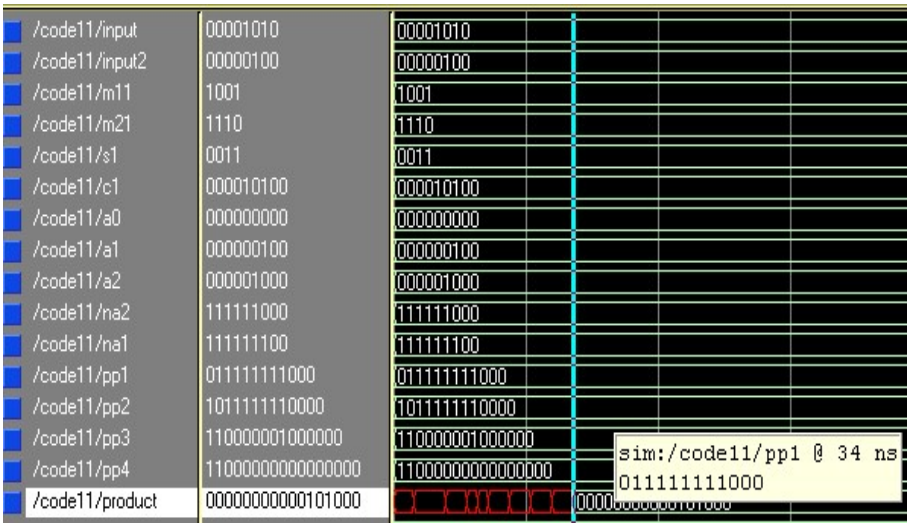

Figure 6: Simulation Results for 15-bit CLA implemented using 4-bit CLA

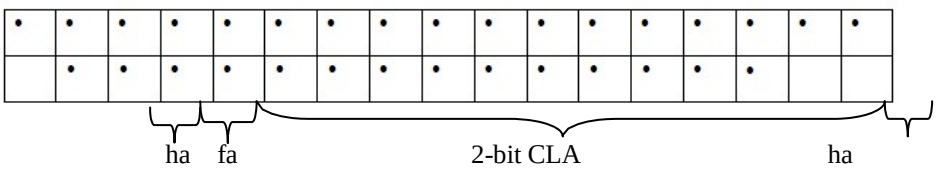

Figure 7: 15-bit CLA implemented as a 12-bit CLA cascaded with two half adders and one full adder

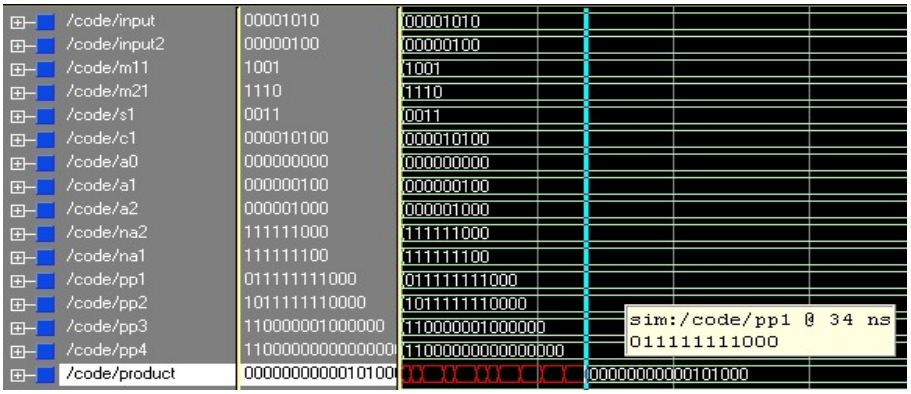

Figure 8: Simulation Results for Modified Booth Wallace Multiplier

The simulation results for Modified Booth Wallace Multiplier whose 13-bit CLA was implemented as a 12-bit CLA cascaded with one half adder is shown in Figure 8. The 13-bit CLA of Modified Booth Wallace Multiplier is illustrated in Figure 9. On analysis of the simulation results obtained for Modified Booth Wallace Multiplier and Modified Booth Dadda Multiplier it is observed that the Modified Dadda Multiplier provides same delay as Modified Booth Wallace Multiplier up to the first two stages. The difference in delay of the two multipliers comes up in the final accumulation stage. When the CLA of the Modified Booth Dadda Multiplier is implemented as shown in Figure 7, the delay of the proposed multiplier is lesser than the Modified Booth Wallace Multiplier. The delay is same when the 15-bit CLA of Modified Booth Dadda Multiplier is implemented using four cascaded blocks of the 4-bit CLA.

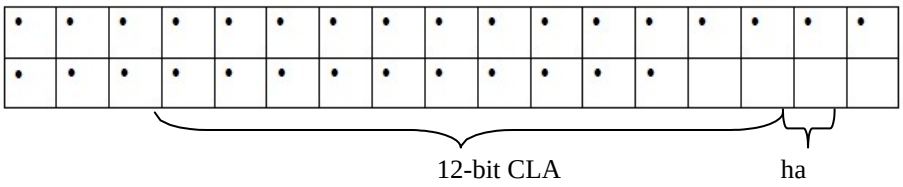

Figure 9: 13-bit CLA of Modified Booth Wallace Multiplier

CONCLUSION

In the second stage of Modified Booth Wallace Multiplier, the Wallace approach uses several stages of full and half adders as carry save adders that are arranged to maximize the reduction at each stage. Full adders take in three bits from a column and give two bits as output i.e. one sum and one carry for a net reduction of one bit per full adder. On the other hand, half adders take in two bits and give one sum and carry as output. Half adders don't reduce the bits of the partial product matrix but reduce the size of the Carry Look-Ahead Adder. Dadda Multiplier uses minimum number of full adders and half adders for each stage there by reducing the area and complexity of the multiplier. The delay for the proposed 8X8 bit Modified Booth Dadda Multiplier was found to be less than that of the Modified Booth Wallace Multiplier when the 15-bit CLA for Modified Booth Dadda Multiplier was implemented as shown in Figure 7. The total half adders used in the Modified Booth Dadda Multiplier were 50% less (including the half adders used in CLA) than those used in Modified Booth Wallace Multiplier and the full adders used in both the multipliers were same. Hence, by using the proposed Modified Booth Dadda Multiplier significant decrease in area and time delay has been obtained over Modified Booth Wallace Multiplier.

REFERENCES

[1] Andrea C. Bickerstaff, Earl E. Swartzlander, Michael J. Schulte, "Analysis of Column Compression Multipliers",15th IEEE Symposium on Computer Arithmetic Proceedings, pp 33-39, 2001.

[2] Himanshu Thapliyal, Neela Gopi, K.K. Praveen Kumar, M.B. Srinivas, "Low Power Hierarchical Multiplier and Carry Look-Ahead Architecture", IEEE International Conference on Computer Systems and Applications, pp. 88-92, 2006.

[3] Kiat Seng Yeo, Kaushik Roy, "Low-Voltage, Low Power VLSI Subsystems", Mc Graw Hill, 2005.

[4] Lakshmanan, Masuri Othman, Mohamad Alauddin Mohd.Ali, "High Performance Parallel Multiplier using Wallace-Booth Algorithm", Proceedings of the IEEE International Conference on Semiconductor Electronics, pp. 433 – 436, 2002.

[5] Liu zhizhe, Zhong Shunan,"Full-custom Design of a 16-bit Multiplier for 0.5um Processing", 1st International Conference on Information Science and Engineering (ICISE), pp. 230-233, 2009.

[6] M.V.Praveen Kumar, S. Sivanantham , S.Balamurugan, P.S.Mallick, "Low Power Reconfigurable Multiplier with Reordering of partial Products", International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), pp. 532-536, 2011.

[7] Na TANG, Jian-Hui JIANG, Kenneth LIN, "A High-Performance 32-bit Parallel Multiplier Using Modified Booth's Algorithm and Sign-Deduction Algorithm", Proceedings of the 5th International Conference on ASIC, volume 2, pp. 1281-1284, 2003.

[8] Rizalafande Che Ismail, Razaidi Hussin, "High Performance Complex Number Multiplier Using Booth-Wallace Algorithm", IEEE International Conference on Semiconductor Electronics, pp. 786-790, 2006.

[9] Shivani Parmar, Kirat Pal Singh, “Design of High Speed Hybrid Carry Select Adder”, International Advance Computing Conference, IEEE, pp. 1654-1660, 2013.
[10] Pallavi Saxena, Urvashi Purohit, Priyanka Joshi, “Analysis of Low Power, Area- Efficient and High Speed Fast Adder”, International Journal of Advanced Research in Computer and Communication Engineering, pp. 3705-3710, 2013.
[11] Maroju SaiKumar,Dr. P. Samundiswary, “Design and Performance Analysis of Various Adders using Verilog”, International Journal of Computer Science and Mobile Computing, pp. 128-138, 2013.
[12] Neeraj Jain,Puran Gour,Brahmi Shrman, “A High Speed Low Power Adder in Multi Output Domino Logic”, International Journal of Computer Applications, pp.30-33, 2014.
[13] Arun Kumar P., “Reusable High Speed Architecture Design for Video Transmission in Satellite Applications”, Recent Researches in Electrical Engineering, pp. 316-324, 2014.
[14] Pallavi Saxena, “Design of Low Power and High Speed Carry Select Adder Using Brent Kung Adder”, International Conference on VLSI Systems, Architecture, Technology and Applications,IEEE, pp. 1-6, 2015.
[15] Kirat Pal Singh, “Performance analysis of Dual gate MOSFET Arithmetic Logic Unit”, International Conference on Recent Trends of Computer Technology in Academia, pp. 96-101, 2012.
[16] Priya Meshram, Mithilesh Mahendra, Parag Jawarkar, “Designed Implementation of Modified Area Efficient Enhanced Square Root Carry Select Adder”, International Journal for Research in Emerging Science and Technology, pp. 96-99, 2015.
[17] Shruti Murgai, Ashutosh Gupta, Gayathri Muthukrishnan, “Energy Efficient and High Performance 64-bit Arithmetic Logic Unit using 28nm Technology”, International Conference on Advances in computing, Communications and Informatics,IEEE, pp. 453-456, 2015.
[18] Atef Ibrahim, Fayez Gebali, “Optimized structures of hybrid ripple carry and hierarchical carry lookahead adders”, Microelectronics Journal, Elsevier, pp. 783-794, 2015.