IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 5, Issue 5, Ver. I (Sep - Oct. 2015), PP 21-29
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Design of a New Fused Add-Multiply Operator Using Modified Booth Recoder

R. Krishna Chaitanya[1], Dr. R. Ramana Reddy[2]
[1] *Student, Department of ECE, MVGR College of Engineering, India*
[2] *Professor, Department of ECE, MVGR College of Engineering, India*

Abstract: *In many Digital Signal Processing (DSP) applications arithmetic operations are widely used and the speed of the applications mainly depends on these operations.Multiply-Accumulate unit(MAC) and Add-Multiply (AM) operator are the widely used operators for this purpose.In this paper, a new design of Fused Add-Multiply (FAM) operator is proposed which uses a newly designed recoding technique for Modified Booth Recodingandimplements the direct recoding of the multiplier in its Sum to Modified Booth (S-MB) form. It is simple, structured, and can be easily modified in order to apply either in signed or unsigned numbers and uses Dadda Carry Save Adder (CSA) for the reduction of partial products. Comparing with the existing recoding FAM designs, the proposed technique yields considerable reductions in terms of critical delay and hardware complexity of the newly designed FAM unit.*

Index Terms: *Add-Multiply operation, Arithmetic Circuits, Dadda tree, Look-up Tables (LUTs), Modified Booth Recoding, VLSI Design.*

I. Introduction

High speed arithmetic operations is a primary concern of high performance digital systems. Recent research activities in the field of arithmetic optimization have observed that the design of arithmetic components which share data can lead to significant performance improvements. In most DSP methods, the nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transform(DWT) are used based on the observation that an addition can often be subsequent to a multiplication (e.g., in symmetric FIR filters).Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition determines the execution speed proceedings and performance of the entire calculation. Because the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined by the multiplier, in general. So, for most arithmetic operations the Multiply-Accumulator (MAC) unit is used. Apart from MAC, the Add-Multiply (AM) unit is also used for many DSP applications. The straight forward design of the AM unit is by first allocating an adder and then driving its output to the input of a multiplier. This design significantly increases both area and critical path delay of the circuit. Targeting an optimized design of AM operators, new recoding schemes [1] are introduced for direct shaping of the sum of two numbers into Modified Booth (MB) form. For the multiplication of two numbers radix-2 Booth Algorithm is used and for high-speed multiplication, the radix-4 Modified Booth Algorithm (MBA) is commonly used [2-3]. Earlier fusion encoding techniques [4-6] are employed based on the direct recoding of the sum of two numbers in its MB form but the new recoding schemes focuses on optimized and efficient implementation of AM operator than the existing recoding schemes. Lyu and Matula [4] presented a signed-bit MB recoder which transforms redundant binary inputs to their MB recoding form. Thus, the carry-propagate (or carry-look-ahead) adder [7, 8] of the conventional AM design is eliminated, resulting in considerable gain in performance.

This work focuses on the efficient design of FAM operators, targeting the efficient addition of partial products. More specifically, the Dadda Carry Save Adder is used over Wallace Carry Save Adder as the former is slightly faster and requires fewer gates and less expensive compared to that of Wallace tree in order to reduce the critical path delay and area [9, 10]. The performance of FAM operator using both Wallace CSA Tree and Dadda CSA Tree is evaluated and is implemented using structured Verilog HDL. It is shown that the area and delay of the proposed method is reduced as compared to the existing method.

This paper is organized as follows: In Section II, the existing design of FAM operator is discussed. In Section III, the proposed design of FAM operator is presented. In Section IV, the experimental analysis showing the advantages of proposed method with respect to area and delay is reported. Finally, Section V concludes the work.

II. Existing Design

A. *Conventional AM Operator*

In the Conventional AM operator, first the inputs A and B are fed to the adder and the output of the adder Y is given as input to the multiplier where X and Y gets multiplied using the MB encoder and partial product generator. These partial products are added with the help of Wallace Carry Save Adder(CSA) and Carry-Look-Ahead (CLA) adder at the final stage. This technique adds significant delay as the carry has to propagate inside the adder and the critical delay depends on the bit width of the adder and also occupies significant area. To speed up the operation, CLA can be used but it increases the area. Fig. 1 shows the block diagram of conventional AM operator.

B. *Fused AM Operator*

To overcome the drawbacks of conventional AM operator design which adds significant delay because of the adder, the fusion techniques [4-6] are used where the inputs A and B is directly recoded into S-MB form. This decreases the critical path delay and reduces area occupation. For the efficient design of FAM operator, new recoding techniques [1] were introduced which reduces the significant critical path delay and area as compared to present recoding schemes. Fig. 2 shows the block diagram of the FAM operator where the adder block is fused into MB encoder block.

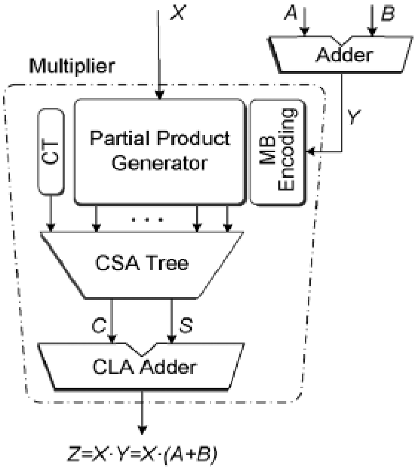


Fig. 1. AM operator based on the conventional design.

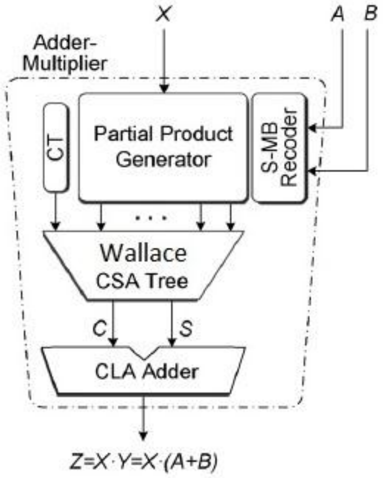


Fig. 2. AM operator based on the fused design with direct recoding of the sum and in its MB representation.

C. *Sum to Modified Booth Recoding Technique(S-MB)*

To Booth recode the multiplier term using radix-4 representation, the bits in blocks of three are considered, such that each block overlaps the previous block by one bit [2]. Grouping starts from the LSB, and the first block only uses two bits of the multiplier since there is no previous block to overlap. The overlap is necessary in order to know what happened in the last block, as the MSB of the block acts like a sign bit. Since the LSB of each block is used to know what the sign bit was in the previous block, the least significant block never have any negative products as the LSB of the first block is always assumed to be 0. In the case where there are not enough bits to obtain a MSB of the last block, the multiplier is sign extended by one bit. In S-MB recoding technique, the sum of two consecutive bits of input $A(a_{2j},a_{2j+1})$ with two consecutive bits of input $B(b_{2j},b_{2j+1})$ is recoded into one MB digit $\mathbf{y}_j^{MB}$. In order to transform inputs into MB form the use of signed-bit arithmetic is needed. For this purpose, the two new types of signed Half Adders (HA) and Full Adders (FA) are used which are referred as HA*, HA**, FA*, and FA** respectively considering their inputs and outputs to be signed. Fig. 3 shows the signed HAs and FAs and their respective truth tables. Considering p, q as binary inputs and c, s are the outputs (carry and sum respectively) of HA* and HA**. Signed HA* implements the relation $2^{\cdot}c - s = p + q$ where the sum is considered negatively signed and the output takes one of the values {0, +1, +2}. With p as a negative input and q as a positive input, HA** implements the relation $2^{\cdot}c - s = - p + q$ resulting in the output values {-1, 0, +1}. For FA* and FA** adders p, q and c_i are assumed as inputs and s, c_o are the output sum and carry respectively. In FA* adder both s and q are considered negatively signed and implements the relation $2^{\cdot}c_o - s = p - q + c_i$. The output values are {-1, 0, +1, +2}. In the case of FA** adder, the two inputs p and q are negatively signed and it implements the relation $-2^{\cdot}c_o + s = - p - q + c_i$ and the output values become {-2, -1, 0, +1}. Here, both the conventional and signed HAs and FAs are used for new alternative recoding schemes[1].

HA*

s= p ⊕ q

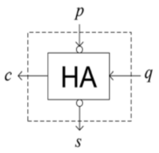
c = p ∨ q

Inputs		Output Value	Outputs	
p(+)	q(+)		c(+)	s(-)
0	0	0	0	0
0	1	+1	1	1
1	0	+1	1	1
1	1	+2	1	0

Output Value = $2^{\cdot} c - s = p + q$

(a)

HA**



s= p ⊕ q

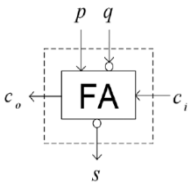
c = ~p ∧ q

Inputs		Output Value	Outputs	
p(-)	q(+)		c(+)	s(-)
0	0	0	0	0
0	1	+1	1	1
1	0	-1	0	1
1	1	0	0	0

Output Value = $2^{\cdot} c - s = - p + q$

(b)

FA*



Inputs			Output Value	Outputs	
p(-)	q(+)	c_i (+)		c(+)	s(-)
0	0	0	0	0	0
0	0	1	+1	1	1
0	1	0	-1	0	1
0	1	1	0	0	0
1	0	0	+1	1	1
1	0	1	+2	1	0
1	1	0	0	0	0
1	1	1	-1	1	1

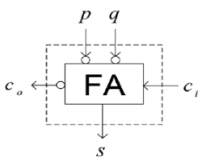
s= p ⊕ q ⊕ c_i

c_o = ((p ∨ ~q) ∧c_i) ∨(p ∧ ~ q)

Output Value = $2^{\cdot} c_o - s = p - q + c_i$

(c)

FA**



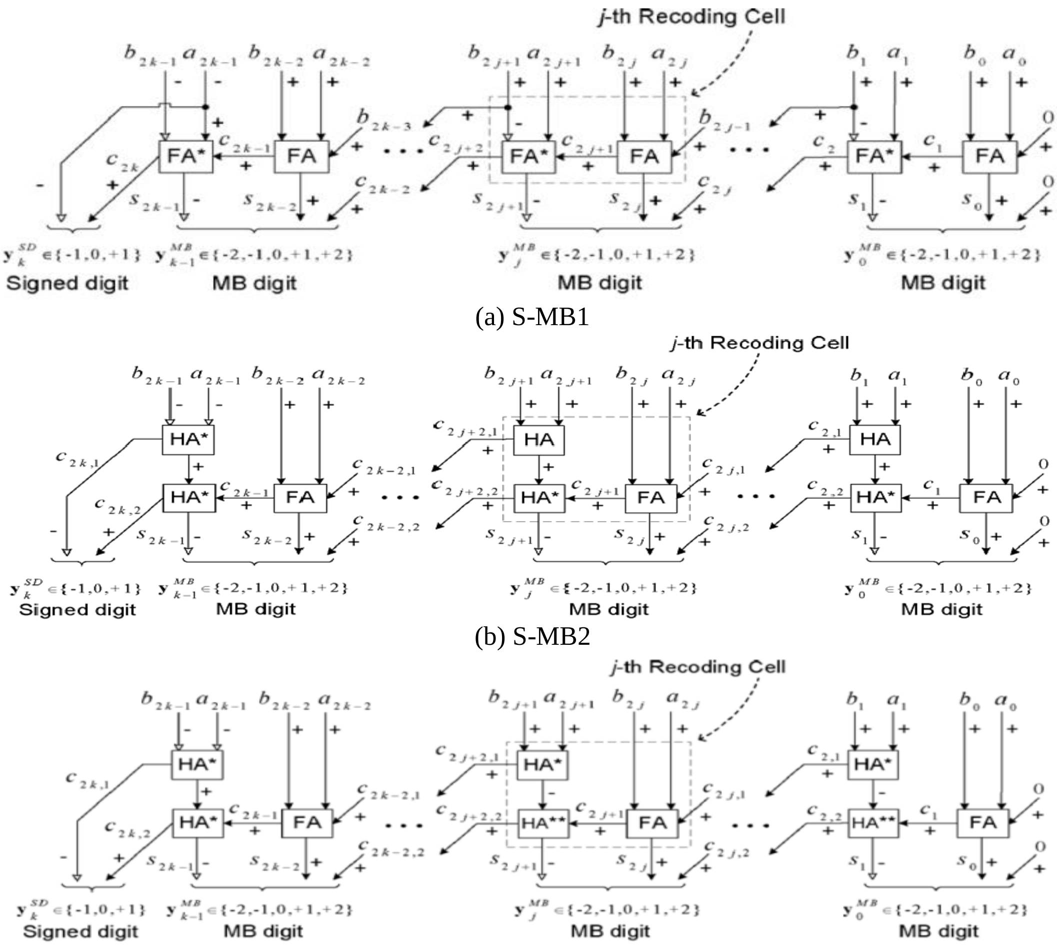
Inputs			Output Value	Outputs	
p(-)	q(-)	c_i (+)		c(-)	s(+)
0	0	0	0	0	0
0	0	1	+1	0	1
0	1	0	-1	1	1
0	1	1	0	0	0
1	0	0	-1	1	1
1	0	1	0	0	0
1	1	0	-2	1	0
1	1	1	-1	1	1

$s = p \oplus q \oplus c_i$ Output Value = $-2 \cdot c_o + s = -p - q + c_i$

$c_o = ((p \vee q) \wedge \sim c_i) \vee (p \wedge q)$

(d)

Fig. 3. Signed Adders and their truth tables (a) HA* (b) HA** (c) FA* (d) FA**.

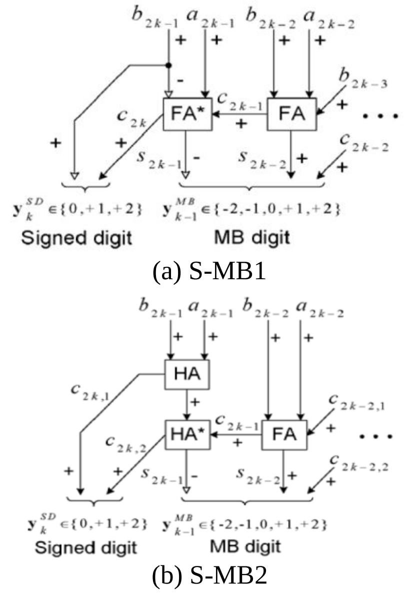

j-th Recoding Cell
FA*
FA
HA
HA*
HA**
b_{2k-1} a_{2k-1} b_{2k-2} a_{2k-2}
b_{2j+1} a_{2j+1} b_{2j} a_{2j}
b_1 a_1 b_0 a_0
c_{2k} c_{2k-1} c_{2j+2} c_{2j+1} c_{2j} c_2 c_1
s_{2k-1} s_{2k-2} s_{2j+1} s_{2j} s_1 s_0
$\mathbf{y}_k^{SD} \in \{-1,0,+1\}$
Signed digit
$\mathbf{y}_{k-1}^{MB} \in \{-2,-1,0,+1,+2\}$
$\mathbf{y}_j^{MB} \in \{-2,-1,0,+1,+2\}$
$\mathbf{y}_0^{MB} \in \{-2,-1,0,+1,+2\}$
MB digit

(a) S-MB1

(b) S-MB2

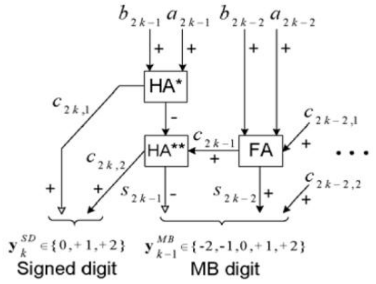
(c) S-MB3

Fig. 4. S-MB Recoding schemes for signed numbers.


FA*
FA
HA
HA*
b_{2k-1} a_{2k-1} b_{2k-2} a_{2k-2}
b_{2k-3}
c_{2k} c_{2k-1} c_{2k-2}
$c_{2k,1}$ $c_{2k,2}$ $c_{2k-2,1}$ $c_{2k-2,2}$
s_{2k-1} s_{2k-2}
$\mathbf{y}_k^{SD} \in \{0,+1,+2\}$
$\mathbf{y}_{k-1}^{MB} \in \{-2,-1,0,+1,+2\}$
Signed digit
MB digit

(a) S-MB1

(b) S-MB2



(c) S-MB3

Fig. 5. S-MB Recoding schemes for unsigned numbers.

1) ***S-MB1 Recoding Scheme***: In this scheme, conventional FA and signed FA* are used. In order to obtain the encoded MB digits $\mathbf{y}_j^{MB}$, S_{2j}, S_{2j+1},$C_{2j}$are used. A conventional FA with inputs a_{2j}, b_{2j} and b_{2j-1} produces the carry C_{2j+1} and the sum S_{2j}, while the FA* with inputs a_{2j+1}, b_{2j+1} and C_{2j+1} produces the carry C_{2j+2} and the sum $S_{2j+1.}$ Fig. 4(a) showsthe S-MB1 recoding scheme for signed numbers and Fig. 5(a) for unsigned numbers.
2) ***S-MB2 Recoding Scheme***: The second recoding scheme S-MB2 for signed and unsigned numbers are shown in Fig. 4(b) & Fig. 5(b) respectively. As in the S-MB1 scheme, a conventional full adder is used to produce the carry C_{2j+1} and

the sum S_{2j} but here in place of FA*, conventional HA and signed HA* is used to produce the carry C_{2j+2} and the sum $S_{2j+1.}$

3) ***S-MB3 Recoding Scheme***: The third recoding scheme S-MB3 uses conventional FA to produce the carry C_{2j+1} and the sum $S_{2j}$and to produce the carry C_{2j+2} and the sum S_{2j+1} it uses signed HA* and HA**. The S-MB3 recoding scheme for signed and unsigned numbers is shown in Fig. 4(c) and Fig. 5(c) respectively.

D. *Wallace and Dadda Tree Multiplier*

Wallace Tree multiplier design has been an essential multiplier in low-power VLSI design [6]. In high-speed designs, 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.Dadda multipliers perform few reductions only when compared to Wallace multiplier. Because of this, Dadda multipliers have less expensive reduction phase, but the numbers may be a few bits longer, thus requiring slightly bigger adders. So, Dadda tree provide less area as compared to Wallace tree[10].

The reduction method of partial products using Dadda tree is as follows:

a. Let d_j=2 and compute d_{j+1}=floor(3*d_j/2)
b. Find the largest d_j that is less than the maximum number of bits in any column.
c. Now, check for every column if the number of bits in the column is less than the d_j. If greater, use HAs and FAs to ensure that the number of elements in the column is ≤d_j.
d. Repeat the same above steps until only two rows are left.

The dot diagramshown in Fig. 6(a) implements the above algorithm for 8*8 multiplier. Four levels are required for the reduction of the matrix with the heights of 6, 4, 3, and 2. In the figure, the diagonal line joining two dots indicates that these two dots are output of a FA and two dots joined by a crossed diagonal line indicates the outputs from a HA. Similarly, the dot diagram for 8*8 multiplier using Wallace reduction is shown in Fig. 6(b). It also requires four reduction levels with matrix heights of 6, 4, 3, and 2. The closer examination between the two reduction methods shows that the despite the presence of longer final adder, Dadda multiplier is faster and smaller over Wallace multiplier [10].

III. Proposed Design

In this section, design of a new fused add multiply operator is proposed and its block diagram is shown in Fig. 7. The proposed design consists of direct S-MB recoder, partial product generator (PPG), Dadda CSA, CLA and Correction Term (CT).

Let us consider the multiplication of 2's complement numbers X and Y with each consisting of n=2k bits. The multiplicand Y can be represented in MB form as:

$$Y = \langle y_{n-1} y_{n-2} \ldots y_1 y_0 \rangle_{2's} = -\, y_{2k-1} \,.\, 2^{2k-1} + \sum_{i=0}^{2k-1} y_i \;.\, 2^i$$

$$= \langle y_{k-1}^{MB} y_{k-2}^{MB} \ldots y_1^{MB} y_0^{MB} \rangle_{MB} = \textstyle\sum_{j=0}^{k-1} y_j^{MB} \;.\, 2^{2j} \quad (1)$$

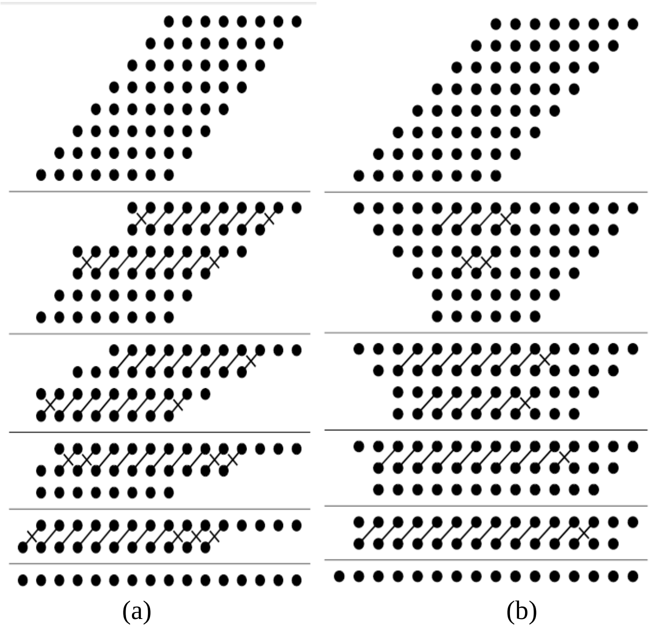


Fig. 6. Dot diagram for 8*8 multiplier using (a) Wallace tree (b)Dadda tree

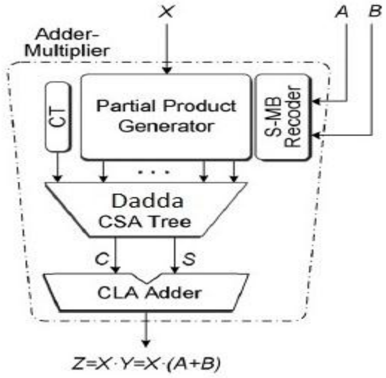


Fig. 7. Proposed efficient FAM operator.

where $$y_j^{MB} = -2y_{2j+1} + y_{2j} + y_{2j-1} \quad (2)$$

Digits $\mathbf{y}_j^{MB}$Ɛ {-2, -1, 0 , +1, +2}, 0≤ j ≤ k-1, corresponds to three consecutive bits of y_{2j+1}, y_{2j} and y_{2j-1} with MSB bit overlapped in order to prevent sign bit and consideringy$_{-1}$=0. In MB encoding technique each digit is represented by three bits named *s*, *one* and*two*. The sign bit shows if the digit is negative (s=1) or positive(s=0). Signal *one* shows if the absolute value of a digit is equal to 1 (one=1) or not (one=0). Signal *two* showsif the absolute value of a digit is equal to 2 (two=1) or not (two=0). Using these three bits the MB digits$\mathbf{y}_j^{MB}$can be calculated by the relation given in equation 3:

$$y_j^{MB} = (-1)^{s_j}.[one_j + 2.two_j] \quad (3)$$

The gate level schematic for the implementation of MB encoding signals is shown in Fig.8. In the FAM design presented here, the multiplier is the parallel one based on MB algorithm. In order to implement the operation of Z= X.Y=X.(A+B), where all the inputs consists of n=2k bits and are in 2's complement form. Firstly the inputs A and B are fed to one of the S-MB recoding scheme and those outputs are fed to the MB encoder. Now, the outputs s_j, one$_j$, two$_j$ are fed to the PPG alongwith input X to generate k partial products.Fig. 9 shows the PPG unit for the generation of the i-th bit $p_{j,i}$ of the partial product PP$_j$.For the computation of least and the most significant bits of partial products x_{-1}=0 and x_n=x_{n-1} are considered respectively. For n number of input bits, k partial products are generated.

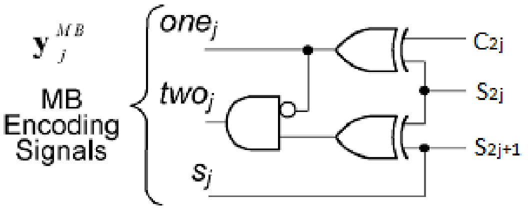


Fig. 8. Gate level schematic of MB encoding signals.

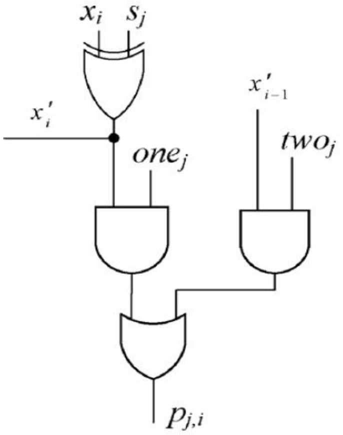


Fig. 9. Generation of i-th bit $p_{j,I}$ of the partial product PP_j.

After the partial products are generated, they are added, properly weighted, through a Dadda CSA tree alongwith the correction term (CT) which is given by the following equation:

$$Z = X.Y = CT + \sum_{j=0}^{k-1} PP_j \,.\, 2^{2j} \tag{4}$$

$$\begin{aligned} CT &= CT(low) + CT(high) = \\ &= \sum_{j=0}^{k-1} c_{in,j} \,.\, 2^{2j} + 2^n\left(1 + \sum_{j=0}^{k-1} 2^{2j+1}\right) \end{aligned} \tag{5}$$

where $c_{in,j} = (one_j \vee two_j) \wedge s_j$.
Finally, the output of the Dadda CSA tree is fed to the final adder to obtain the final result Z=X.Y.

IV. Experimental Analysis

The proposed design of a new FAM operator is implemented in structured Verilog HDL using Xilinx ISE 14.1 and simulated using ISim simulator for all the three S-MB recoding schemes for both Wallace CSA tree and Dadda CSA tree. The results shows that use of Dadda CSA tree for partial products reduction yield considerable performance in terms of area and delay over Wallace tree reduction. Among three recoding schemes, S-MB2 has better performance over S-MB1 and S-MB3. Apart from the recoding schemes, various types of adders like Ripple Carry Adder, CLA and Carry Select Adder are also implemented. Area required for implementing recoding schemes and adders are presented in table 1. Delay for 8 bit FAM and for 16 bit FAM are presented in Fig. 10 and Fig. 11 respectively.

V. Conclusion

The design of a new FAM operator using three new recoding schemes and three types of adders is implemented. The partial products are reduced using both Wallace CSA tree and Dadda CSA tree. The comparison results show that among all the recoding schemes and adders, S-MB2 and CSLA respectively, is having lesser delay for 8-bit operation and 16-bit operation when implemented using Dadda CSA tree over Wallace CSA tree. The area comparison shows that the number of LUTs and Slices used by RCA is less as compared to others adders and SMB-1 among recoding schemes. So, in overall performance CSLA is preferred among adders and S-MB2 is preferred among the recoding schemes.

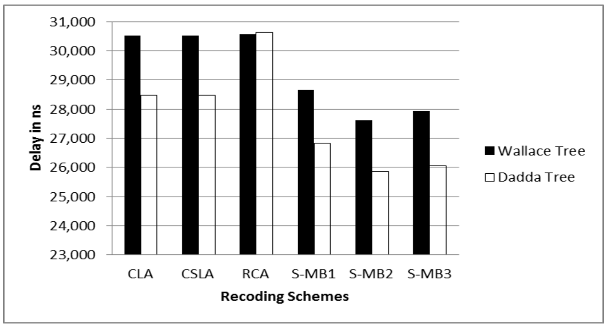


Fig. 10. Comparison of delay for 8 bit New FAM operator.

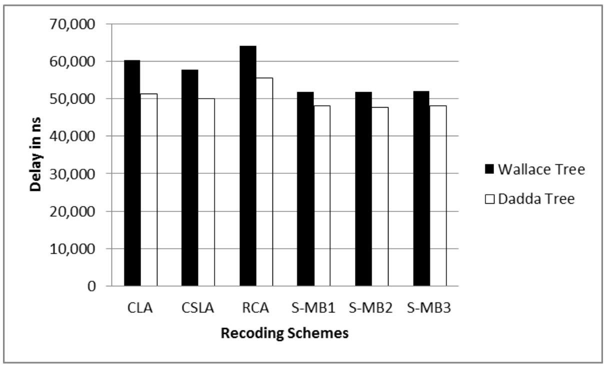


Fig. 11. Comparison of delay for 16 bit New FAM operator.

Table I Comparison of Area Using Different Recoding Schemes.

		CLA	CSLA	RCA	S-MB1	S-MB2	S-MB3
No. of Slices	Wallace Tree	323	326	322	332	333	334
	Dadda Tree	298	303	297	304	305	311
No. of LUTs	Wallace Tree	589	594	586	588	595	599
	Dadda Tree	539	547	536	538	546	548

References

[1]. Kostas Tsoumanis, Sotiris Xydis, Nikos Moschopoulos, and KiamalPekmestzi, "*An Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Operator*", IEEE Transactions On Circuits And Systems—I: Regular Papers, Vol. 61, No. 4, pp. 1133-1143, April 2014.

[2]. SukhmeetKaur, Suman and ManpreetSignh Manna, "*Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix-2)*," Advance in Electronic and Electric Engineering., Volume 3, Number 6 (2013), pp. 683-690.

[3]. Young-HoSeo and Dong-Wook Kim, "*A New VLSI Architecture of Parallel Multiplier–Accumulator Based on Radix-2 Modified Booth Algorithm*", IEEE Transactions On Very Large Scale Integration (Vlsi) Systems, Vol. 18, No. 2, pp. 201-208, February 2010.

[4]. C. N. Lyu and D. W. Matula, "*Redundant binary Booth recoding*," in Proc. 12th Symp. Comput. Arithmetic, 1995, pp. 50–57.

[5]. J. D. Bruguera and T. Lang, "*Implementation of the FFT butterfly with redundant arithmetic*," IEEE Trans. Circuits Syst. Il, Analog Digit. Signal Process., vol. 43, no. 10, pp. 717–723, Oct. 1996.

[6]. DeepikaPurohit and Himanshu Joshi, "*Comparative Study and Analysis of Fast Multipliers*," International Journal of Engineering and Technical Research (IJETR), Volume-2, Issue-7, July 2014.

[7]. O. L. Macsorley, "*High-speed arithmetic in binary computers*," Proc. IRE, vol. 49, no. 1, pp. 67–91, Jan. 1961.

[8]. N. H. E. Weste and D. M. Harris, "*Datapath subsystems*," in CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Readington: Addison-Wesley, 2010, ch. 11.

[9]. M. Daumas and D. W. Matula, "*A Booth multiplier accepting both a redundant or a non redundant input with no additional delay*," in Proc. IEEE Int. Conf. on Application-Specific Syst., Architectures, and Processors, 2000, pp. 205–214.

[10]. Whitney J. Townsend, Earl E. Swartzlander and Jacob A. Abraham, "*A Comparision of Dadda and Wallace multiplier Delays*," Computer Engineering Research Center, The University of Texas at Austin.

R. KRISHNA CHAITANYA received B.Tech degree from Thandra Paparaya Institute of Science and Technology, Bobbili in the year 2008 and presently pursuing M.Tech degree in VLSI Design in MVGR College of Engineering, Vizianagaram. His research interests include Low Power VLSI Design and Embedded Systems.

Dr. R. RAMANA REDDY did AMIE in ECE from The Institution of Engineers (India) in 2000, M.Tech (I&CS) from JNTU College of Engineering, Kakinada in 2002, MBA (HRM & Marketing) from Andhra University in 2007 and Ph.D in Antennas in 2008 from Andhra University. He is presently working as Professor & Head, Dept. of ECE in MVGR College of Engineering, Vizianagaram. Coordinator, Center of Excellence – Embedded Systems, Head, National Instruments Lab VIEW academy established in Department of ECE, MVGR College of Engineering. Convener of several national level conferences and workshops.Published about 50 technical papers in National/International Journals Conferences. He is a member of IETE, IEEE, ISTE, SEMCE (I), IE, and ISOI. His research interests include Phased Array Antennas, Slotted Waveguide Junctions, EMI/EMC, VLSI and Embedded Systems.