Study of Various High Speed Multipliers

Sumod Abraham
Student Dept. of ECE
M.R.C.E. Faridabad
sumod11abraham@gmail.com

Sukhmeet Kaur
Asst. Professor Dept. of ECE
M.R.C.E. Faridabad
sukhmeetkaur.mrce@mrei.ac.in

Shivani Singh
Asst. Professor Dept. of ECE
M.R.C.E. Faridabad
shivanisingh.mrce@mrei.ac.in

***Abstract*- Multiplication or repeated addition is the basic operation used in both Mathematics and Science. The speed of multiplier determines the speed of all Digital Signal Processors. This paper describes four multipliers that is Modified Booth Multiplier, Vedic Multiplier (Urdhva Tiryakbhyam Sutra), Wallace Multiplier and Dadda Multiplier. In Modified Booth Multiplier Algorithm '0' is appended to the right of LSB and then three bits starting from the LSB are grouped as a set to decide the partial product. So as to increase its speed, desk calculators are used that perform the operation of shifting faster. Urdhva Tiryakbhyam Sutra is performed by two multiplication techniques that is straight above multiplication and diagonal multiplication. Then finally, their sum is taken. Here reduction of multi bit multiplication to single bit multiplication takes place followed by the process of addition. Since there is only one step generation of partial product, carry propagation from LSB to MSB is reduced. There are three steps in which both Wallace and Dadda Multiplier work. They are partial product formation, reduction of partial products formed to two rows and addition of these two rows using carry propagation adder. Wallace Multiplier reduces all possible partial products and so has a smaller carry propagating adder whereas Dadda Multiplier performs only minimum necessary reduction and thus has a larger carry propagating adder. But as far as speed is concerned Dadda Multiplier is faster than Wallace Multiplier.**

Keywords- Modified Booth Multiplier, Vedic Multiplier, Wallace Multiplier, Dadda Multiplier, partial products.

I. INTRODUCTION

Addition, subtraction, multiplication and division are the basic arithmetic operations that are performed in any system. Out of these, it is the multiplication operation that scales one number by another[1], slows down any system. [2] The performance of many computational problems are often dominated by the speed at which a multiplication operation can be executed. A multiplier is one of the key hardware blocks in most digital signal processing (DSP) systems. The important operations in digital signal processing are filtering, convolution, and inner products [3]. Typical DSP applications where a multiplier plays an important role include digital filtering, digital communication and spectral analysis. There are different types of multipliers out of which Booth Multiplier, Vedic Multiplier (Urdhva Tiryakbyam Sutra), Wallace Multiplier and Dadda Multiplier will be discussed in this paper.

II. MODIFIED BOOTH MULTIPLIER

In Booth Multiplier, we consider set of two bits each, from the LSB of the multiplier and perform the associated operations to form partial products (PP). Here, the number of PP formed is equivalent to the number of bits of the multiplier. That is, for 'n' bit multiplier, 'n' PP are formed. So, cost of the multiplier is high and its speed is slow. In Modified Booth multiplier, for 'n' bit multiplier we have 'n/2' PP (if n is even) and '(n+1)/2' PP (if n is odd) which lowers the cost and improves the speed of the multiplier.

The steps given below are to be followed so as to perform multiplication using Modified Booth Multiplier:

1. Append a '0' to the right side of the LSB. Then, group the multiplier bits in sets of three starting from the LSB.

2. Compare the bit set with the one shown in the table [4] drawn below and perform the operation accordingly to form partial products.

M(i+1)	M(i)	M(i-1)	OPERATION
0	0	0	0
0	0	1	+1
0	1	0	+1
0	1	1	+2
1	0	0	-2
1	0	1	-1
1	1	0	-1
1	1	1	0

Fig 1: Modified Booth Multiplier Operation Table

In the table drawn, '+1' means multiplicand bits be placed as it is. '+2' means double the multiplicand bits (i.e. multiplicand bits plus multiplicand bits). '-1' means two's complement of multiplicand bits and '-2' means two's complement of the doubled multiplicand bits.

3. To each partial product formed append a '0' to the left if the product is positive or a '1'if the product is negative. Then add.

978-1-4799-6805-3/15/$31.00 ©2015 IEEE

III. VEDIC MULTIPLIER

[5] The term 'Vedic' is derived from the word 'Veda' which means the store-house of all knowledge. The mathematics based on Veda is called Vedic Mathematics. There are sixteen sutras in vedic mathematics each dealing with geometry, algebra etc. Urdhva Tiryakbyham Sutra (meaning vertically and clockwise) is one among them. Here since the generation of partial products and their sum are calculated in parallel (that is together), the multiplier is independent of clock frequency. Thus Urdhva Tiryakbyham Sutra doesn't require microprocessor to operate at high clock frequency. Following is the Urdhva Tiryakbyham Sutra algorithm with an example:

1. Consider two binary numbers A and B, each with say 4 bits.

2. Number each bit of A (from LSB) as A(0), A(1), A(2) and A(3). Similarly B will be numbered as B(0), B(1), B(2) and B(3).

3. Multiply A(0) and B(0). The obtained result will be S(0).

4. Multiply A(0) and B(1) and add the result to the product of A(1) and B(0). The obtained sum is S(1) and carry is C(1).

5. Add the results obtained by multiplying A(2) with B(0), A(1) with B(1) and A(0) with B(2), with C(1) to get the sum as S(2) and carry as C(2).

6. Add the results obtained by multiplying A(3) with B(0), A(2) with B(1), A(1) with B(2) and A(0) with B(3), with C(2) to get the sum as S(3) and carry as C(3).

7. Add the results obtained by multiplying A(3) with B(1), A(2) with B(2) and A(1) with B(3), with C(3) to get the sum as S(4) and carry as C(4).

8. Add the results obtained by multiplying A(3) with B(2) and A(2) with B(3) with C(4) to get the sum as S(5) and carry as C(5).

9. Multiply A(3) and B(3) and add the result with C(5) to obtain the sum as S(6) and carry as C(6).

10. Finally, the product is C(6)S(6)S(5)S(4)S(3)S(2)S(1)S(0).

The same vertical and clockwise format discussed before may be used to multiply two binary numbers each having any number of bits.

IV. WALLACE MULTIPLIER

Wallace multiplier has three stages in which it works [6]. In the first stage we form a matrix of partial products. In the second stage, we compress the matrix into height of two. That is with two rows. Finally in the third stage, we add the two rows together using carry propagating adder to get the desired result. Here, the partial products are reduced as soon as possible.

Wallace algorithm is explained below with an example:

1. Consider two decimal numbers 15 and 51. Convert both these numbers to its equivalent binary that is 00001111 for 15 and 00110011 for 51.

2. Multiply both the numbers bit by bit as shown below:

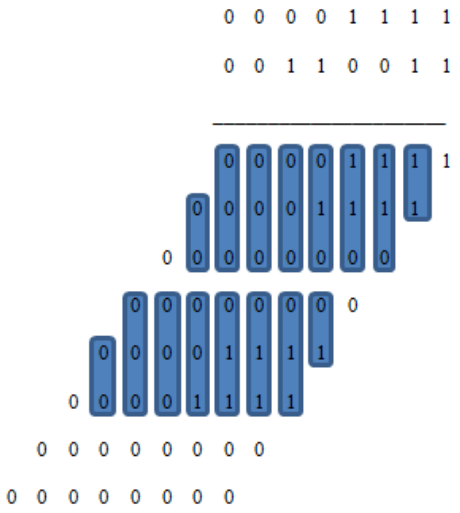

Fig 2: Partial product matrix

3. Then compress the above matrix to 6, 4, 3 and 2 rows using (2,2) or (3,2) counter [4]. As shown in figure 2, consider each column and check if two or three bits can be combined together by (2,2) or (3,2) counters. Here (2,2) means two inputs (marked in shade) and two outputs (sum and carry). Similarly (3,2) means three inputs and two outputs. Based on the ouputs received figure 2 is drawn as shown.

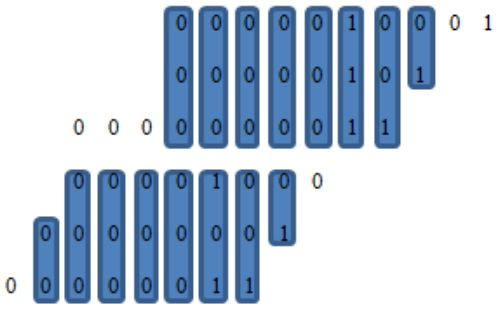

Fig 3: Matrix reduced to height of 6 by Wallace Multiplier

4. To draw figure 3, the bits which are not shaded i.e where counters are not used are placed as it is. The place (column)

where these counters are used (in figure 2), there first place the unshaded bits (like in seventh column from right, 0 is unused so it will be placed first in column seven of figure 3 and likewise for all other columns). Then those bits where counters are used (shaded), there the sum of the paired bits(from figure 2) are placed (in figure 3) and the next bit (in figure 3) will have the carry of the previous column (figure 2).

5. Then similarly compress the above matrix (figure 3) to 4 rows to form the matrix shown in figure 4.

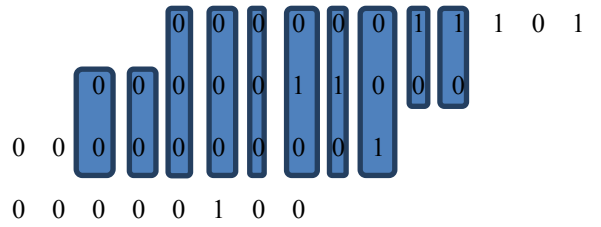

Fig 4: Matrix reduced to height of 4 by Wallace Multiplier

5. Then similarly compress the above matrix (figure 4) to 4 rows to form the matrix shown in figure 5.

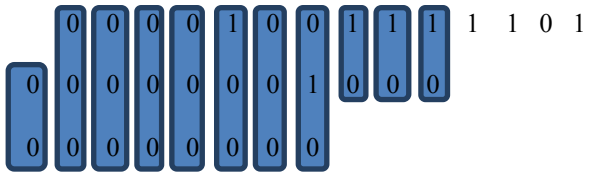

Fig 5: Matrix reduced to height of 3 by Wallace Multiplier

6. Then, compress the above matrix to height of 2 as shown.

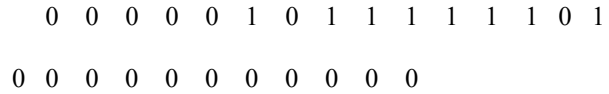

Fig 6: Matrix reduced to height of 2 by Wallace Multiplier

7. Then, finally add the two rows of the matrix (figure 6) to get the result 0 0 0 0 0 0 1 0 1 1 1 1 1 1 0 1 which is the equivalent of 765_d (15 X 51).

V. DADDA MULTIPLIER

There are three stages in which a Dadda Multiplier works [6]. First is the formation of partial product matrix. Then this matrix in reduced to two rows (that is height is kept two). Finally using carry propagating adder, these two rows are added. Dadda algorithm performs minimum reduction necessary at each level. [7] In order to realize the minimum number of reduction stages, the height of each intermediate matrix is limited to the least integer that is no more than 1.5 times the height of its successor. For the purpose of reducing partial product matrix, counters like (2,2) and (3,2) are used.

The algorithm of Dadda Multiplier is explained below with an example:

1. To multiply 54 with 74, first convert these decimal numbers to their binary equivalent.

2. Then perform multiplication, to get partial products. But do not add the partial products. This step is shown below:

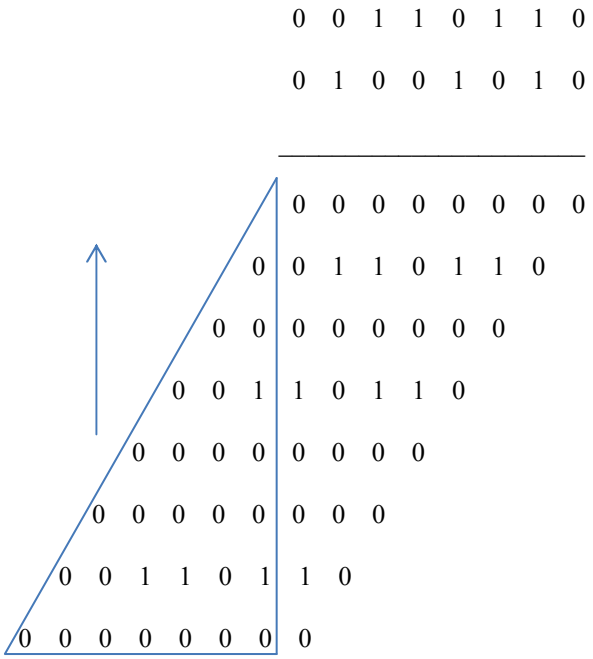

Fig 7: Partial product matrix

3. Then the bits under the triangle are shifted upwards to form the pattern as shown below in figure 8.

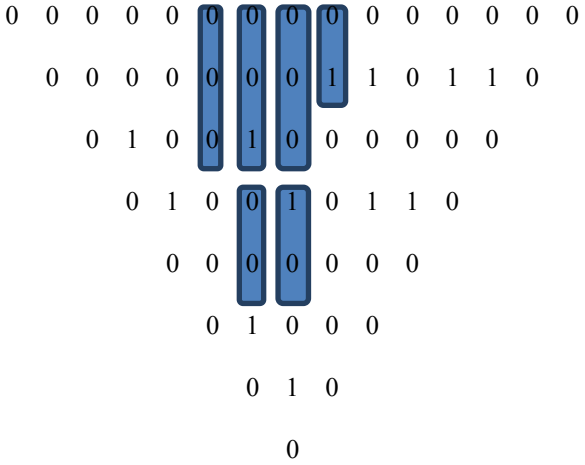

Fig 8: Pattern formed using Dadda Multiplier

4. Now the matrix is compressed to 6,4,3 and 2 rows (1.5 times the height of its successor) [7]. Starting with compression to 6 rows, using counters as shown in figure 9:

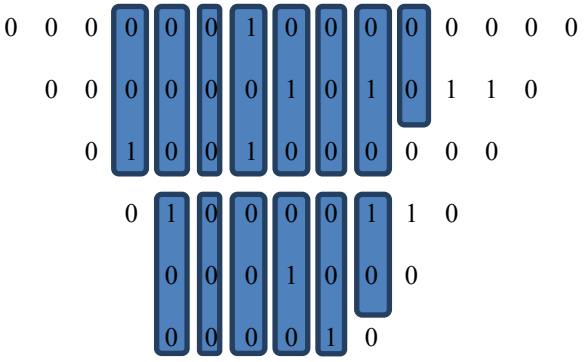

Fig 9: Matrix reduced to height of 6 by Dadda Multiplier

5. Now matrix is reduced to 4 rows in figure 10:

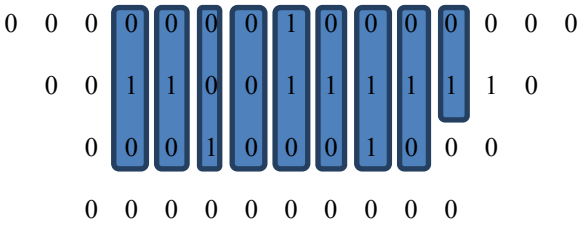

Fig 10: Matrix reduced to height of 4 by Dadda Multiplier

6. Further matrix is reduced to 3 rows, as in figure 11:

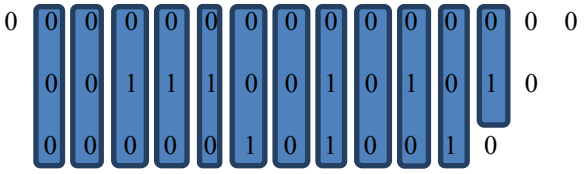

Fig 11: Matrix reduced to height of 3 by Dadda Multiplier

7. Then reducing the above matrix to 2 rows, we get the following format as shown in figure 12.

0 0 0 1 1 1 1 0 0 0 1 1 0 0 0

0 0 0 0 0 0 1 0 0 0 0 1 0

Fig 12: Matrix reduced to height of 2 by Dadda Multiplier

8. Finally we add the two rows (of figure 12) together to get the final output 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 which is the binary equivalent of 3996_d (54 X 74).

VI. PROPOSED ARCHITECTURE

As the demand for higher performance arithmetic units such as high speed multipliers, efforts have been focused on performing new algorithms and improving functionality of their constitutive elements [8]. Modified Booth Multiplier (radix 4) reduces the number of partial products formed [9] when compared to ordinary booth multiplier. Also Dadda Multiplier uses less number of adders when compared with Wallace Multiplier. So if both these algorithms (Modified Booth and Dadda) are used partially to make a new algorithm Booth Dadda Algorithm, then it will be efficient than the earlier algorithms. That is, in Booth Dadda algorithm partial products will be formed using modified Booth algorithm and the process of addition will be performed by Dadda algorithm. The proposed architecture is drawn below in figure 13.

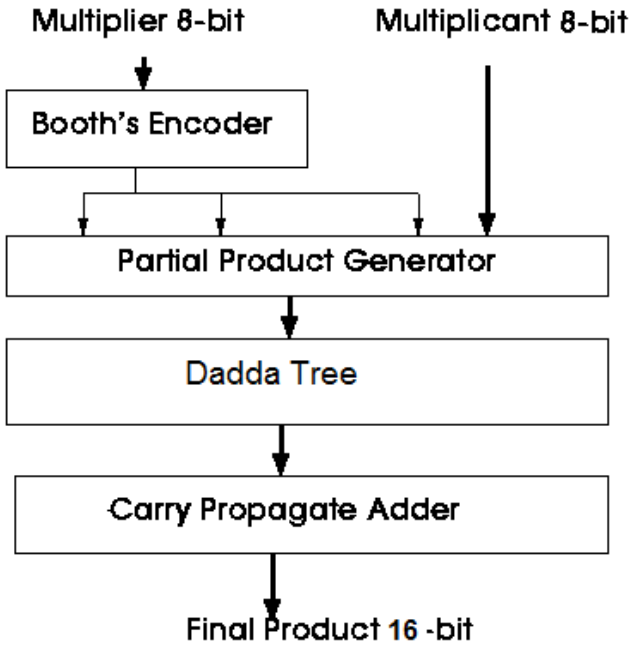

Fig 13: Proposed architecture

This proposed architecture will reduce the size of the multiplier (less number of adders) and will increase its speed (reduced number of partial products formed).

VII. CONCLUSION

It is the multiplier, which determines the speed of any Digital Signal Processing system. So, various algorithms have been developed in order to perform multiplication faster and also to reduce the size of the multiplier. Various multipliers like Modified Booth multiplier, Vedic multiplier (Urdhva Tiryakbyham Sutra), Wallace multiplier and Dadda multipliers are discussed here. All these multipliers perform the task of multiplication correctly but their algorithms are different. The number of adders used in Dadda multiplier is less than that in case of Wallace. Also by changing the

compressors, that is by using 5:2 or 7:2, delay time can be reduced [10]. Thus, increasing the speed of the multiplier.

REFERENCES

[1] "VHDL Modeling of Booth Radix-4 Floating Point Multiplier for VLSI Designer's Library" by Wai-Leong Pang, Kah-Yoong Chan, Sew-Kin Wong and Choon-Siang Tan in Wseas Transactions On Systems.

[2] "VLSI design of low power digital FIR filter using PSPICE and VLSI design of high speed digital FIR filter using VERILOG HDL." (2013). Chapter 5 by Vigneswaran, T.

[3] *"Efficient Implementation of 16-bit Multiplie-Accumulator Using Radix-2 Modified Booth Algorithm and SPST Adder Using Verilog"* by Addanki Purna Ramesh, Dr. A.V.N. Tilak and Dr. A.M. Prasad in International Journal of VLSI design & Communication Systems (VLSICS) Vol.3, No.3, June 2012

[4] *"Review Article: Efficient Multiplier Architecture In VLSI Design" by* M. Jeevitha, R.Muthaiah and P.Swaminathanin in Journal of Theoretical and Applied Information Technology 2012. Vol. 38 No.2.

[5] "*Review on floating point multiplier using ancient techniques"* by Sushma S. Mahakalkar and Dr.S.L.Haridas in IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE) e-ISSN: 2278-1676, p-ISSN: 2320-3331 PP 01-04.

[6] "*A comparison of Dadda and Wallace multiplier delays*" by Townsend, Whitney J., Earl E. Swartzlander Jr, and Jacob A. Abraham," in Optical Science and Technology, SPIE's 48th Annual Meeting on International Society for Optics and Photonics, 2003.

[7] "*A high speed binary floating point multiplier using Dadda algorithm*." by Jeevan, B., Narender S., Dr. C.V. Krishna Reddy and Dr. K. Sivani in International Multi-Conference IEEE, 2013 on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013.

[8] "*A New High, Area Efficient 7-2 Compressor for Fast Arithmetic Operations"* by Mohammad Tohidi, Yasin Amani, Mostafa Amirpour and Saeid Karamzadeh at Internatıonal Journal of Natural and Engineering Sciences 7 (2): 72-77, 2013 ISSN: 1307-1149, E-ISSN: 2146-0086.

[9] *"Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix-2)"* by Sukhmeet Kaur, Suman and Manpreet Signh Manna in Advance in Electronic and Electric Engineering. ISSN 2231-1297, Volume 3, Number 6 (2013), pp. 683-690.

[10] "*Implementation of Wallace Tree Multiplier Using Compressor*" by Naveen Kr. Gahlan, Prabhat Shukla and Jasbir Kaur in Int. J. Computer Technology & Applications, Vol 3 (3), 1194-1199 ISSN:2229-6093.