Performance Analysis of Various Multipliers for Digital Signal Processing Applications

C. Saranya[1], Dr. M. Balasubramaniam[2], Dr. R. Naveen[3]

Assistant Professor, Department of ECE, INFO Institute of Engineering, Coimbatore [1, 2]

Associate Professor, Department of ECE, INFO Institute of Engineering, Coimbatore [3]

Abstract: Multipliers have great importance in both digital signal processors and microprocessors. So designing a reliable multiplier is the most important factor in designing a signal processor. Fast multiplication process is very important in DSPs for convolution, Fourier transforms etc. Power, speed and area are the prime design constraints of a multiplier for signal processing applications. Here a comparative analysis is made on different multiplier architectures which have been used for various signal processors. The different multipliers architectures chosen are array multiplier, a column bypass multiplier, row bypass multiplier, Vedic multiplier and Booth Wallace multiplier. The multiplier architectures simulated using tanner EDA tool and the results are compared in terms of delay, power consumption and area.

Keywords: Multipliers, power consumption, delay, area

I. INTRODUCTION

The demand for very high speed processing has been increasing as a result of expanding computer and signal processing applications. Digital signal processing is an important unit in any electronic communication devices. Digital Signal Processors are used to perform the common operations such as video processing; filtering and Fast Fourier transform (FFT). Multiplication is most fundamental operation in digital systems and digital signal processors [1]. In many digital signal processor algorithms, the multiplier lies in the critical delay path and ultimately determines the performance of algorithm. The speed of multiplication operation is of great importance in digital signal processor as well as in general processor. In the past multiplication was implemented generally with a sequence of addition, subtraction and shift operations. This was a tedious and time consuming process .In recent years, many researchers have tried and are trying to design multipliers which offer either of the following design targets; high speed, low power consumption, regularity of layout and hence less area or even combination of them in one for various signal processing applications. Multiplier consumed 35% of power and also occupied 45% of chip area in 64 point radix-4 pipelined FFT processor. Therefore, multiplier is most critical, power hungry arithmetic unit that requires more area and computational time in any signal processor [2]. So by designing low power and high speed multiplier the overall performance of any digital signal processor in terms of power, delay and area can be improved. Here a comparison is made on existing multipliers which are used in current signal processing applications such as array, column bypass, row bypass, Vedic and Booth Wallace multipliers. These multipliers were simulated using tanner electronic design automation tool and results are compared in terms of power, delay and area.

II. ARRAY MULTIPLIER

In this array multiplier, partial product is generated by taking the multiplicand and one bit of multiplier each time. The addition is carried out by high speed carry save adder and the final product is obtained by employing parallel adder. The number of partial products depends upon the number of multiplier bits used in the design. A 4x4 array multiplier is shown in Fig. 1. Each products terms are generated in parallel with the help of AND gates. Each partial product can be added with the sum of partial product which has previously generated by using the row of adders. The left shifted carry out is added with the sum generated by the first adder and the newly generated partial products. The shifting process is done by the Carry Save Adder (CSA) and the Ripple carry adder or any fast adder can be used for the final stage addition [3].

			a_3	a_2	a_1	a_0
			b_3	b_2	b_1	b_0
			a_3b_0	a_2b_0	a_1b_0	a_0b_0
		a_3b_1	a_2b_1	a_1b_1	a_0b_1	
	a_3b_2	a_2b_2	$a_1 b_2$	a_0b_2		
a_3b_3	a_2b_3	a_1b_3	a_0b_3			
			. . . .		$a_1b_{0+}a_0b_1$	a_0b_0

Fig. 1. Structure of Array multiplier

III. COLUMN BYPASS MULTIPLIER

All Multiplier in which columns of adder are simply bypassed is called column bypass multiplier. In this multiplier, the multiplication operations in a column can be disabled if the corresponding bit in the multiplicand is zero as shown in Fig.2. In this design the addition and multiplication operations in the $(i+1)_{th}$ column can be bypassed if the bit, a_i, in the multiplicand is zero, that is all the partial products a_i b_j, $0 \leq j \leq n-1$, are zero.

Here two tri state buffers and one 2:1 multiplexer is used to attach each adder in the Carry Save Adder array. As the bit, a_i, in the multiplicand is zero, their inputs in the $(i+1)_{th}$ column will be disabled and the carry output in the column must be set to be zero to produce the correct output. Hence, the modification process can be achieved by adding an AND gate to the outputs of the last row [4].

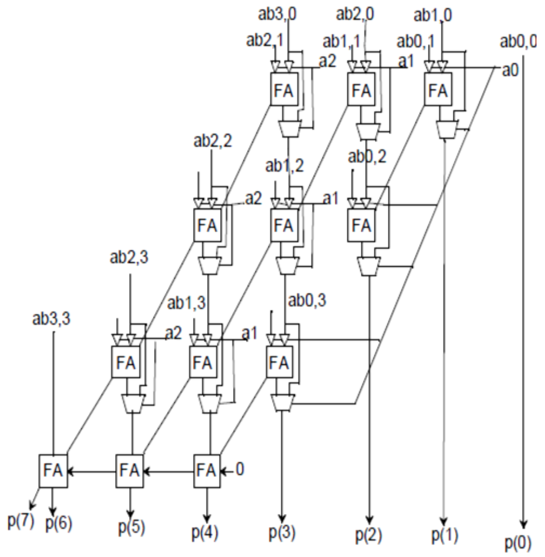

Fig. 2. Structure of Column Bypass multiplier

IV. ROW BYPASS MULTIPLIER

In this multiplier, the multiplication process is optimized by bypassing row of adders which are redundant. Here partial products of multiplicand bits and multiplier bits are generated with the help of AND gates. It is always possible that a bit in the multiplier is a zero, so if one input to the AND gate is zero then the output of that gate will be zero. By this process the complete row of partial product becomes zero.

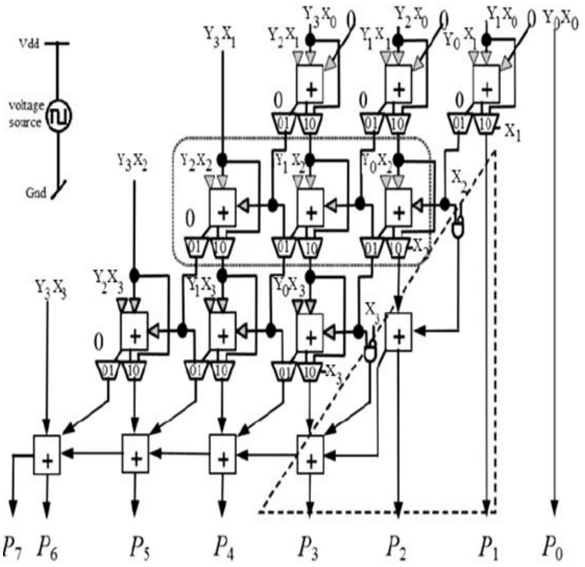

Fig. 3. Structure of Row Bypass multiplier

Now by adding the elements of this row results in zero. So the complete row is bypassed which reduces the signal transition without affecting the result. Fig.3 shows the structure of row bypass multiplier. Here selection lines of 2:1 multiplexers and Tri-state buffers should have the same bit value when compared with the multiplier row elements in which adder circuit lies.

Whenever the select line is zero, the output from the previous adder is passed through the multiplexer enter into the input of the next adder. Similarly the partial product is allowed to enter the adder with the help of Tri-state buffer present in the input side. Thereby results in bypassing the complete row of adders [4].

V. VEDIC MULTIPLIER

The hardware architecture of 4×4 bit Vedic multiplier module are shown in Fig.4. Here Urdhva Tiryagbhyam (vertically and crosswise) sutra is used to propose such architecture for the multiplication of two binary numbers. The beauty of Vedic multiplier is that here partial product generation and additions are done concurrently. Hence it is well adapted to parallel processing. The feature makes it more attractive for binary multiplications. This in turn reduces power which is the primary motivation behind this work.

The 4×4 bit Vedic multiplier module is implemented using four 2x2 bit Vedic multiplier modules. The inputs of the module are A= A3 A2 A1 A0, B= B3 B2 B1 B0 and output is S7 S6 S5 S4 S3 S2 S1 S0. Here A and B are divided into two parts that is A3A2 and A1 A0 for A and B3 B2 and B1B0 for B. Each block as shown Fig.4 is 2×2 bit Vedic multiplier. First 2×2 bit multiplier inputs are A1A0 and B1B0. The last block is 2×2 bit multiplier with inputs A3 A2 and B3 B2. The middle one shows two 2×2 bit multiplier with inputs A3 A2 & B1B0 and A1A0 & B3 B2.

So the final result of multiplication, which is of 8 bit, S7 S6 S5 S4 S3 S2 S1 S0. To get final product (S7 S6 S5 S4 S3 S2 S1 S0), four 2×2 bit Vedic multiplier and three 4-bit Ripple-Carry (RC) Adders are required [5]. This Vedic multiplier can be used to reduce delay. Early literature speaks about Vedic multipliers based on array multiplier structures. But this architecture is efficient in terms of speed and parallel processing.

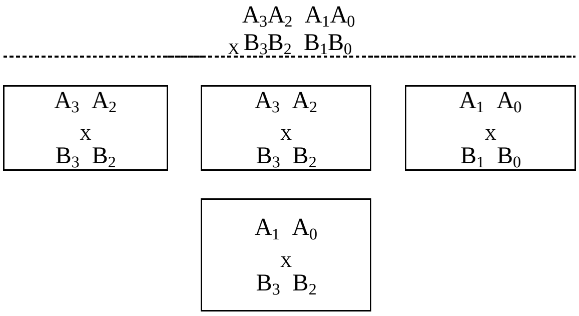
A_3A_2 A_1A_0
x B_3B_2 B_1B_0
A_3 A_2 x B_3 B_2
A_3 A_2 x B_3 B_2
A_1 A_0 x B_1 B_0
A_1 A_0 x B_3 B_2

Fig. 4. Structure of Vedic multiplier

VI. BOOTH WALLACE MULTIPLIER

Booth Wallace multiplier includes three modules [6]. In first module Booth encoding and partial product generation is done, in second module Wallace tree is formed for partial product reduction, in third module a carry look ahead adder is used to perform final addition.

In order to overcome the disadvantages of radix 2 booth algorithm modified booth algorithm is used, where more number of bits is considered at a time. Radix4 scans three bits at a time and radix8 scans four bits at a time.

In the booth algorithm, multiplier is grouped in an overlapped manner initially, means the MSB of one group will become LSB of next group. Also before grouping add a zero to the LSB of the multiplier.

Then each group is encoded to the range of {-2,-1,0,1,2} in the case of radix 4 and to the range of {-4,-3,-2,-1,0,1,2,3,4} in the case of radix 8. By using radix 8 only three partial products are generated and speed can be increased.

Multiplication flow diagram of booth Wallace multiplier is shown in Fig.5. After generating the partial products the reduction is done with help Wallace tree. Here compressors can be used for the faster reduction. A 3 bit full adder is called 3:2 compressors. It reduces 3 inputs to 2 inputs. Higher order compressors [7] are also available like 4:2, 5:2 etc.4:2 compressors is just a cascade of two full adders. It actually compresses 5 inputs to 3 inputs.

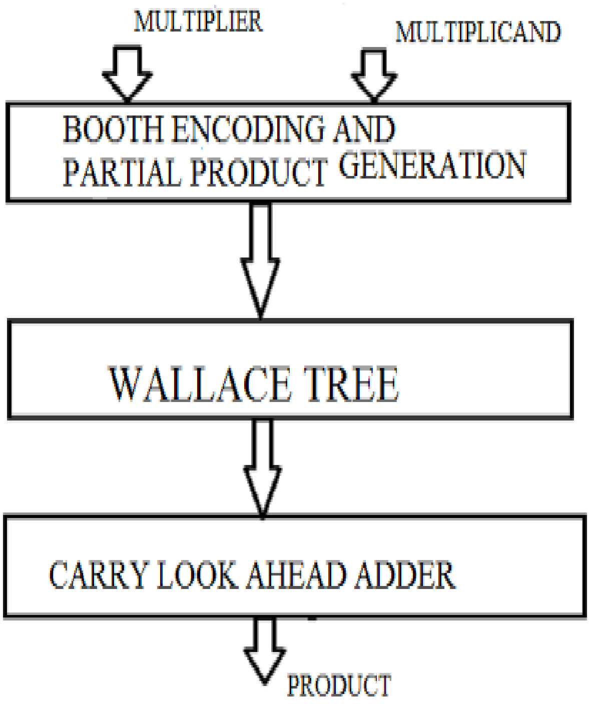

Fig. 5. Structure Booth Wallace Multiplication flow

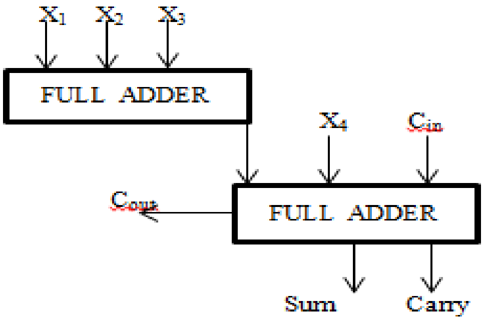

Fig. 6. 4:2 Compressors

4:2 compressors is shown in Fig.6. Here sum will go to the same position whereas the Carry and C_{out} goes to the next higher position and final addition is done by Carry look ahead adder [8]. This adder overcomes the problem of ripple carry adder. It includes a generator Gi = X_iY_i and propagate the term $P_i = X_i$ xor Y_i for finding carry. C_i+1 = G_i + P_iC_i.

VII. CONCLUSION

These four multipliers are simulated using Tanner Electronic Design Automation tool and obtained results are tabulated in Table I and Table II.The simulated results shows that Booth Wallace multiplier has less transistor count and power consumption when compared with other three multipliers.

TABLE I

4*4 Multipliers(500 MHZ)	Transistor count	Average Power Consumption (µW)	Delay (ps)	Power delay product (aJ)	Area (µm^2)
Array	366	38.06	40.74	1387.60	18.75
Column Bypass	300	35.12	40.15	1312.29	15.20
Row Bypass	320	36.79	41.28	1410.66	17.5
Vedic	588	40.06	41.64	1374.81	19.37
Booth Wallace	280	34.14	40.27	1251.69	11.87

TABLE III

4*4 Multipliers (1GHZ)	Transistor count	Average Power Consumption (µW)	Delay (ps)	Power delay product (aJ)	Area (µm2)
Array	366	52.06	39.55	1396.23	18.75
Column Bypass	300	50.12	39.15	1330.60	15.20
Row Bypass	320	51.29	40.85	1421.50	17.5
Vedic	588	54.32	40.64	13782.45	19.37
Booth Wallace	280	49.28	39.80	1276.10	11.87

DOI 10.17148/IJARCCE.2016.5954

Based on the analysis it is concluded that Booth Wallace multiplier shows better performance in terms of transistor count and power consumption. But Column Bypass multiplier shows better performance in terms of delay when compared with other multipliers. So these two multipliers are worth enough to make the performance of digital signal processor more efficient.

ACKNOWLEDGMENT

We thank the management and Head of the research centre of INFO Institute of Engineering for their support and encouragement towards this research work.

REFERENCES

[1] Z. Abid, H. El-Razouk, and D. El-Dib,” Low power multipliers based on new hybrid full adders, “Microelectronics Journal. vol.39, pp. 1509-1515, 2008.

[2] W. C. Yeh and C.-W. Jen,” High-speed and low-power split-radix FFT,” IEEE Transactions on Signal Processing. vol. 51, pp. 864-874, 2003.

[3] J.T. Yan and Z.W. Chen, “Low-power multiplier design with row and column bypassing,”IEEE International SOC Conference.pp.227-230, 2009.

[4] Pardhi, Sonam, and Neetesh Dodkey, “A Survey on Different Multiplier Architectures,” IJCER. vol.4,no.6, pp.135-138, 2016.

[5] Anju Agarwal, “FPGA Implementation of Low Power and High Speed Vedic Multiplier using Vedic Mathematics,” IOSR Journal of VLSI and Signal Processing. vol. 2, no.5, pp.51-57. 2013.

[6] Juny Mary Jose and Reen Paul, “A Novel Booth Wallace Multiplier for DSP Applications,” IJCER, vol.6,no.3, pp.32-36, 2016.

[7] Jagadeshwar Rao, M. and Sanjay Dubey,”A high speed Wallace tree multiplier using modified booth algorithm for fast arithmetic circuits,” IOSR Journal of Electronics and Communication Engineering, vol. 3,no.1, pp. 07-11,2012.

[8] Sharma, Neeta, and Ravi Sindal,”Modified Booth Multiplier using Wallace Structure and Efficient Carry Select Adder,” International Journal of Computer Applications, vol.68, no.13, pp. 39-42, 2012.

DOI 10.17148/IJARCCE.2016.5954