IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

VLSI Design and Implementation of Multipliers for DSP Applications

Iqshan Ahamed A[1], Karthikeyan E[2], Naveen Baabu S[3], Sundhar P[4], Dr. S. C. Prasanna[5]
Students, Department of Electronics and Communication Engineering[1,2,3,4]
Professor, Department of Electronics and Communication Engineering[5]
SRM Valliammai Engineering College, Kattankulathur, India

Abstract: *A multiplier is a critical building block found in processors, embedded systems, VLSI applications, application specific integrated circuits, and most DSP applications. Speed, area, and power are the three primary thrust characteristics in VLSI design. Low power and high speed is the desirable characteristic in many applications that can extend the battery's life expectancy and increase the frequency of operation. The goal of this project is to design, implement and analyse the performance of array multipliers, booth multipliers, Wallace tree multipliers, and modified booth multipliers. In this work multipliers with different bit widths are implemented on Spartan 3E FPGA and their performances are analysed. Among these architectures Wallace tree multiplier provides higher speed of operation and consumes lesser power.*

Keywords: Wallace tree, Array, Booth algorithm, Spartan 3E FPGA, Xilinx ISE 14.7 Design Suite, Speed, delay, power

I. INTRODUCTION

Multiplication performance is critical for digital signal processing (DSP) tasks such as filtering, correlation, convolution, and Fourier transforms, which involves repeated multiply-accumulate operations. For example, multi-media applications like 3D graphics systems that require the execution of a large number of multiplications. So the digital signal processors are built around high-speed multipliers. The size, power consumption, and silicon area are all heavily influenced by the word length of multiplication. In VLSI, the three main thrust performance parameters are area, speed and power consumption. It is critical to have a design that is efficient in terms of area, speed and power. Therefore several researchers proposed several architectures for efficient implementation of multipliers. It is suggested in [1] that Wallace tree multiplier is capable of doing high speed operation than Braun array multipliers. Array multiplier is the conventional multiplier used to perform parallel multiplication. This multiplier is suitable for lesser multiplier widths, because it will occupy larger area, consumes more power and also give more delay as width increases [VLSI design- SIA publications]. In this project multiplier architectures such as Wallace tree multiplier, and multipliers based on the Booth algorithm are designed, implemented and compared its performance with conventional multipliers. It is seen that Wallace tree multiplier performs well in terms of speed of operation as well as power consumption. The rest of the paper is organised as follows. Section II describes the architectural details of array multiplier, Wallace tree multiplier and Booth algorithm based multiplier. The implementation results and its analysis is presented in Section III and IV respectively. Finally the hardware implementation and conclusion is described in section V and VI respectively.

II. MULTIPLIER ARCHITECTURES

A binary multiplier is a combinational circuit that multiplies two binary integers in digital electronics, such as computers. The architectural details of array multiplier, Wallace tree multiplier and Booth algorithm based multiplier is discussed below.

2.1 Array Multiplier

The multiplication process in array multiplier is demonstrated in (Figure 1) using 4x4 array multiplier. It show partial products can be generated. That is it requires AND gate for each bit position to generate partial products. Here 4 AND gates are needed in each stage for generating each partial product (partial product 1 - b0a3 b0a2 b0a1 b0a0). Then these partial products are added using half adders and full adders at each stage. Then the product can be obtained from the last stage of

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

adders. The architecture for 4x4 array multiplier using AND gates, half adders and full adders is shown in (Figure 2). Table 1.1 Device utilization for MxN multiplier. The number of AND gates, half adders and full adders needed for MxN multiplier is tabulated in Table 1

Table 1: Device Utilisation of Array Multiplier

No. of AND	No. of HAs	No. of FAs	Total adders
M*N	N	(M-2) *N	(M-1) *N

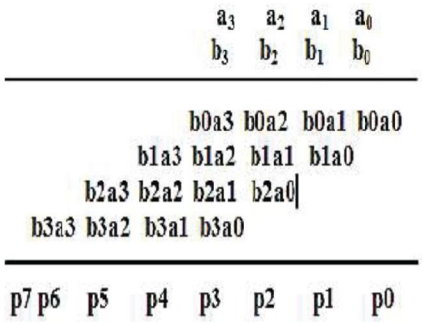

Figure 1: 4x4 Array Multiplier Methodology

The number of stages increases with multiplier width, which will also increase the complexity and also delay and power consumption. So for larger width multiplication where speed of operation and power consumption is critical array multipliers are not suitable.

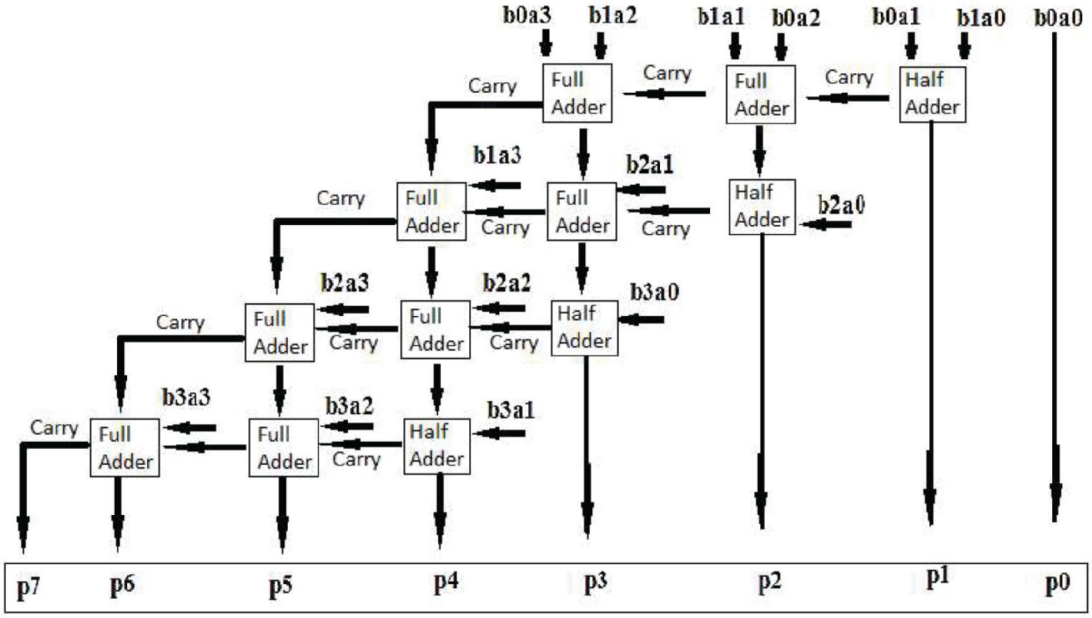

Figure 2: 4x4 Array Multiplier Implementation

2.2 Wallace Tree Multiplier

The final outputs of Wallace tree multiplier will be generated by a collection of adders. Because of the lengthy wires required to propagate carries from low order bits to high order bits, carry propagate addition are rather sluggish.

Wallace's use of carry save adders (CSAs, also known as mixture of full adders and half adders or 3-2 counters) to add three or more numbers in a redundant and carry propagation freeway is perhaps the single most important development in boosting the speed of multipliers.(Figure. 3) illustrates the procedure.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

Any number of partial products can be added and reduced to two values without using a carry propagate adder by recursively using the basic three input adder. To reduce the two numbers to a single end product, just a single carry propagate addition is required.

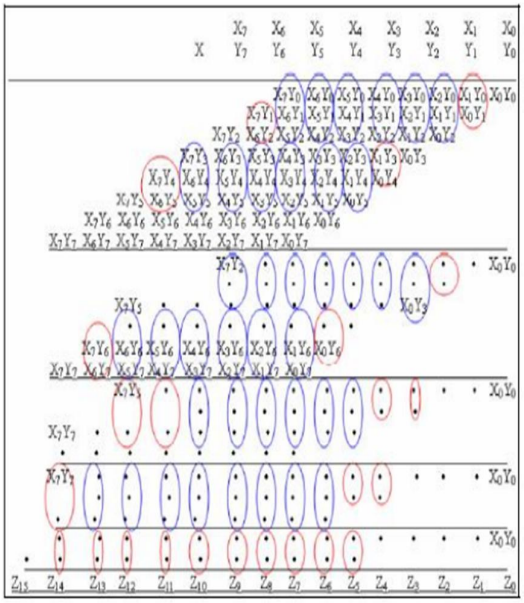

Figure 3: 8*8-bit Wallace tree multiplier

There are three stages in the Wallace tree:

1. Multiply each multiplier bit by the multiplicand bit position. The weights of created partial products vary depending on the location of the multiplier bits.
2. Using layers of full and half adders, reduce the number of partial products to two.
3. We now have two rows of sum and carry, which we may combine using traditional adders.

Explanation of the second step: Add a following layer if there are three or more rows with the same weight:

1. Input any three rows with the identical weights into a complete adder. For each of the three input wires, the outcome will be an output row with the same weight, i.e., sum, and an output row with a greater weight, i.e., carry.
2. If there are two rows of the same weight left, use a half adder to combine them.
3. Connect the last row to the next layer if there is just one remaining. The Wallace tree has the benefit of having just O (log n) reduction layers (levels), each with an O(1) propagation time. Multiplication is just O (log n), not much slower than addition, because partial products are O(1) while final addition is O(log n) (however, gate count increases progressively). Regular adders would take O(log n2) time to combine incomplete products.

	STAGE-1 OUTPUT	STAGE-2 ACTION	OUTPUT	SATGE-3 ACTION	OUTPUT
Weight-1	1	PT	1	PT	1
Weight-2	2	HA	1	PT	1
Weight-4	3	FA	2	HA	1
Weight-8	4	FA	3	FA	2
Weight-16	3	FA	2	HA	2
Weight-32	2	HA	2	HA	2
Weight-64	1	PT	2	HA	2
Weight-128	0				1

Figure 4: Action in constructing 4-bit Wallace tree multiplier

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

Multiply each Xi with each Yi in the first stage (AND gates) to generate a total of n*n intermediate wires. Each wire is assigned a weight, such as X0. 1(20.20 =1) is the weight of Y0. Y3 has the same weight as X3. Using extra stages made up of full adders and half adders, you may reduce the number of intermediary wires. Using a complete adder, combine any three wires of the same weight; the outcome in the following step is one wire of the same weight (sum) and one wire of a greater weight (i.e., carry). In the last phase, all weights have only one or two wires. To make two 2n-bit values, connect the wires together.

2.3 Radix 2 Booth Multipliers

This method quickly and efficiently calculates the multiplication of two signed binary numbers in two's complement notation. This approach evaluates adjacent pairs of bits in the 'N'-bit multiplier Q in signed two's complement format, including an implied bit below the least significant bit, N1 = 0. When these two bits are equal, the product accumulator P remains unchanged. The multiplicand times 2i is added to P where Qi = 0 and Qi1 = 1, and the multiplicand times 2i is removed from P when Qi = 1 and Qi1 = 0. P's final value is the signed product.

The steps are not in any specific order in this case. It generally proceeds from LSB to MSB starting at I = 0; the multiplication by 2i is then replaced by slow shifting of the P accumulator to the right between stages; low bits can be pushed out, and subsequent additions and subtractions can only be done on the higher N bits of P.

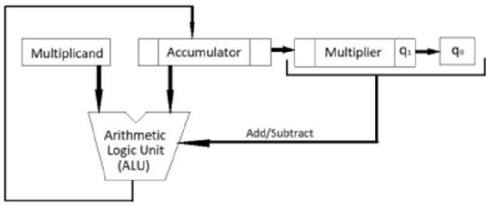

Figure 5: Booth's Algorithm Architecture

The radix-2 Booth's algorithm is demonstrated in Figure (ex. Fig. 6) as a flowchart.

To further understand Booth's technique, consider the following scenario. Take the multiplier Q = +3 and the multiplicand A = -6. The functioning of this method may be characterised as a trace table that displays the state of each processing step. In this case, input A is a negative number, and the 2's complement counterpart is required to finish the calculation. A= $(-6)_{10} = (1010)_2$ while (-A) = $(0110)_2$

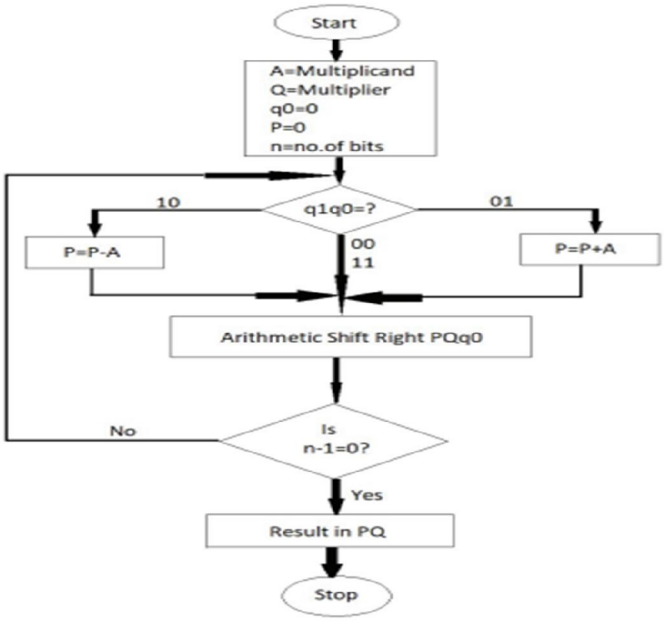

Figure 6: Flowchart of radix 2 Booth's algorithm

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

n	Accumulator, P	Multiplier, Q $q_4q_3q_2q_1$	q_0	Action
4	0000	0011	0	Initialization, Value of q_1q_0=10, P=P-A
	0111	0011	0	Arithmetic shift right PQq0
3	0011	1001	1	Value of q1q0=11, Arithmetic shift right PQq_0
2	0001	1100	1	Value of q1q0=01, P=P+A
	1010	1100	1	Arithmetic shift right PQq0
1	1101	0110	0	Value of q1q0=00, Arithmetic shift right PQq0
0	1110	1011	0	Value of n-1=0, process complete. Result in the PQ

Figure 7: Radix 2 booth's algorithm grouping table

At the point where n-1 Equals 0, the result in PQ = 11110110. The final product in base-10 must be its 2's complement equivalent since this is a negative integer. Booth's approach preserves the sign of the result. The result must be stated in negative notation when the signed bit in the value of PQ is 1. (00010010)2 = (-18)10 is the 2's complement of PQ.

2.4 Booth's Radix 4 Multiplication Algorithm

The Booth's Multiplication method has been significantly improved, resulting in an increment in the count of bits grouped and a reduction in the number of computing phases. These tactics have been shown to increase the multiplier's performance and, as a result, the efficiency of DSP applications. In the Radix-4, Radix-8, and Radix-16 type Booth's Multiplication method, Table 2 specifies the bit grouping and the related operation. Radix-32, Radix-128, Radix-256, and even radix-4096 kind multipliers were developed using a similar technique, with more study and implementation advised for best application design.

Table 2: Grouping Table of Radix 4

CODE	OPERATION
000,111	0
001	1* multiplicand
010	1* multiplicand
011	2* multiplicand
100	-2*multiplicand
101	-1*multiplicand
110	-1*multiplicand

III. Simulation Results

Array multiplier, Wallace tree multiplier, Radix 2 Booth's multiplier were implemented and simulated in Xilinx 14.7 ISE Design suite and generated output were given below.

3.1 Array Multiplier

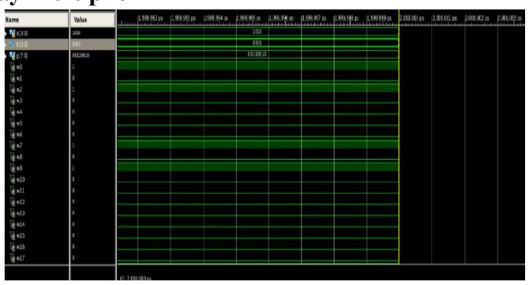

Figure 8: Simulation results of 4- bit array multiplier

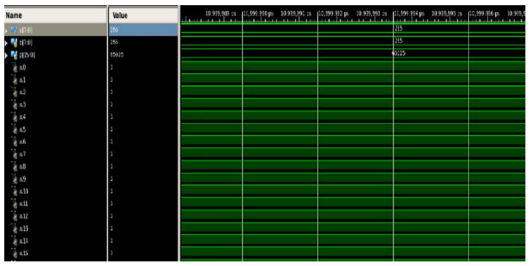

Figure 9: Simulation results of 8- bit array multiplier

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

3.2 Wallace Tree Multiplier

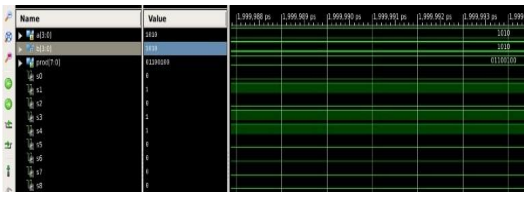

Fig. 10: Simulation results of 4- bit Wallace tree multiplier

Fig. 11: Simulation result of 8 -bit Wallace tree multiplier

3.3 Radix 2 Booth's Multiplier

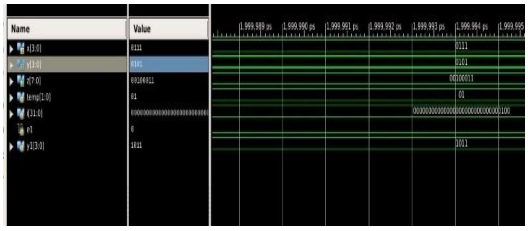

Fig. 12: Simulation results of 4- bit radix 2 Booth's algorithm

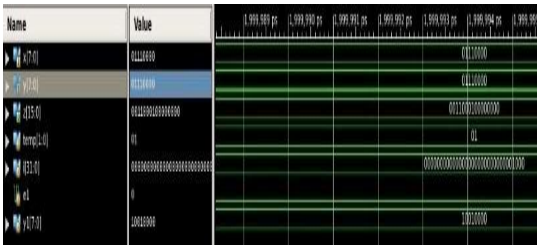

Fig. 13: simulation results of 8 -bit radix 2 Booth's algorithm

IV. PERFORMANCE AND REPORT ANALYSIS

4.1 Device Utilisation Summary

The device utilisation summary, time delay, power parameters of array multiplier, Wallace tree multiplier, Radix – 2 Booth's algorithm for 4 bit and 8- bit are tabulated below. The power consumption of these multipliers are calculated with the help of Xilinx Vivado Design suite. The tabular column 3 given below is device utilisation summary of 4- bit multipliers

Table 3: 4-bit Multipliers

PARAMETER	Array	Wallace	Radix 2 Booth's	Radix 4 Booth's
No. of slices	18/63400	22/63400	43/63400	55/64300
No. of bonded IO's	16/210	16/210	16/210	16/210
Total delay (nS)	3.345	3.172	5.740	5.295

The tabular column 4 given below is device utilisation summary of 8- bit multipliers.

Table 4: 8-Bit Multipliers

PARAMETER	Array	Wallace	Radix 2 Booth's	Radix 4 Booth's
No. of slices	99/63400	94/63400	216/63400	259/63400
No. of bonded IO's	32/210	32/210	32/210	32/210
Total delay (Ns)	7.847	7.153	12.57	10.972

The tabular column 5 given below is device utilisation summary of 4- bit multipliers.

Table 5: 16-Bit Multipliers

PARAMETER	Array	Wallace	Radix 2 Booth's	Radix 4 Booth's
No. of slices	351/63400	553/63400	818/63400	1086/63400
No. of bonded IO's	64/210	64/210	64/210	64/210
Total delay (nS)	15.017	12.948	25.059	18.49

B. POWER ANALYSIS:

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

The tabular column 6 given below is the power consumption of 4 -bit multipliers.

Table 6: 4-Bit Multipliers

Multipliers	Dynamic Power(W)	Static Power(W)	Total Power(W)
Array	3.942	0.143	4.085
Wallace	3.920	0.143	4.063
Radix 2 Booth's	3.776	0.143	3.918
Radix 4 Booth's	5.651	0.150	5.801

The tabular column 7 given below is the power consumption of 8 -bit multipliers.

Table 6: 8-Bit Multipliers

Multipliers	Dynamic Power(W)	Static Power(W)	Total Power(W)
Array	13.649	0.199	13.848
Wallace	13.849	0.202	14.10
Radix 2 Booth's	16.071	0.222	16.293
Radix 4 Booth's	18.096	0.244	18.341

The tabular column 8 given below is the power consumption of 16-bit multipliers.

Table 6: 16-Bit Multipliers

Multipliers	Dynamic Power(W)	Static Power(W)	Total Power(W)
Array	48.047	1.346	49.398
Wallace	37.58	0.726	38.318
Radix 2 Booth's	40.157	0.852	41.009
Radix 4 Booth's	45.539	1.166	46.705

V. HARDWARE IMPLEMENTATION

The hardware implementation of Wallace tree multipliers was done with the help of Spartan 3E FPGE kit available in the VLSI laboratory which is shown in the figure(Ex. Fig. 14,15 and 16).

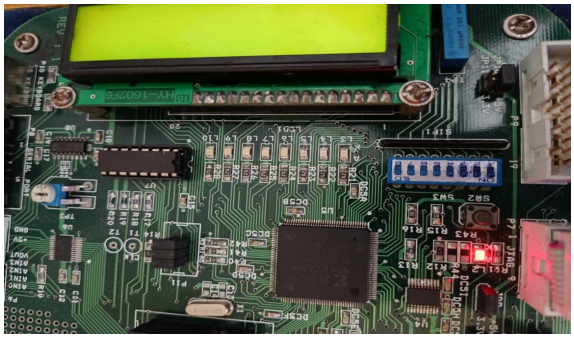

Figure 14

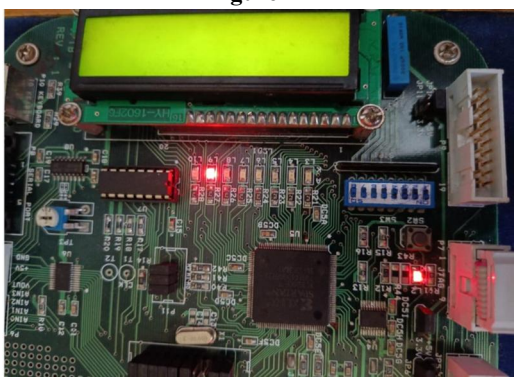

Figure 15

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 3, May 2022
Impact Factor: 6.252

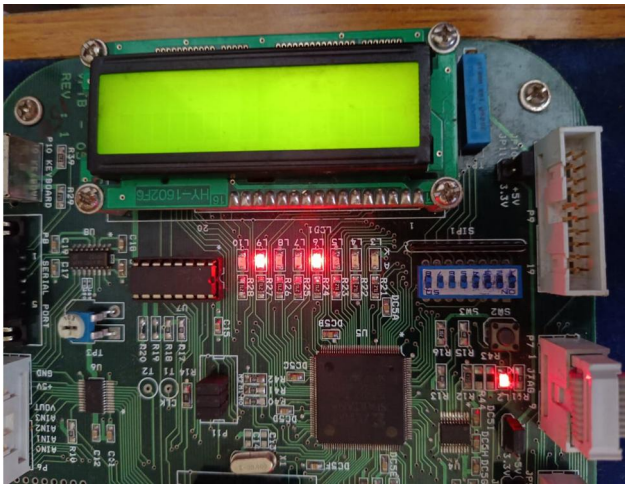

Figure 16

VI. CONCLUSION

The multipliers were implemented using Xilinx 14.7 ISE Design Suite and the performance of these multipliers were analysed successfully. From this performance analysis, the Wallace tree multipliers (i.e., 3.172nS for 4 bit, 7.152nS for 8 bit and 12.948nS for 16 bit) are having high speed and less delay compared with array and booth's algorithm. In terms of power consumption, Wallace consumed less power. So, for DSP application, it is better to use Wallace tree multiplier. Wallace tree multiplier was implemented in Spartan 3E FPGA kit due to its better performance.

ACKNOWLEDGMENT

We sincerely thanking our respected principal Dr. B. Chidhambararajan. We also expressing our profound thanks to the Head of the Department Dr. Komala James and to our supervisor Dr. S. C. Prasanna, Assistant professor for their constant support, guidance and motivation while facing tough time in our project and making this project a successful one.

REFERENCES

[1]. P.V. Rao, C Prasanna Raj, S. Ravi, "VLSI Design and Analysis of Multipliers for Low Power", Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing", 2009.PP-1354-1357.

[2]. R.Bajaj, S. Chhabra, S. Veeramachaneni and M B Srinivas, "A Novel, Low-Power Array Multiplier Architecture", International Institute of Information Technology-Hyderabad,2017.

[3]. Sangeetha P, Aijaz ali khan, "Comparison of Braun multiplier and Wallace multiplier techniques in VLSI" in IEEE Access, 2019.

[4]. Iffat Fatima, "Analysis of Multipliers in VLSI" Journal of Global Research in Computer Science,2014.

[5]. N. Honarmand, M.R.Javaheri, N.SedaghatiMokhtari and A. Afzali-Kusha "Power Efficient Sequential Multiplication Using Pre-computation" ISCAS 2006.

[6]. Anandha Gunduru, Youngstown state university, "Analysis of Booth's Multiplier Algorithm vs Array Multiplier Algorithm and their FPGA Implementation" in 2019.

[7]. https://www.researchgate.net/publication/2575879_Fast_Multiplication_Algorithms_And_Implementation

[8]. Laxman S, Darshan Prabhu R, Mahesh S Shetty, Mrs. Manjula BM, Dr. Chirag Sharma "FPGA Implementation of Different Multiplier Architectures" ISSN 2250- 2459, Volume 2, Issue 6, June 2012)