How to Cite:
Shobana, G., Chithiraimuthu, R., & Adhithyavel, A. (2022). Performance analysis and implementation of approximate multipliers on spartan 6 FPGA. *International Journal of Health Sciences*, *6*(S1), 10633–10652. https://doi.org/10.53730/ijhs.v6nS1.7546

Performance analysis and implementation of approximate multipliers on spartan 6 FPGA

Mrs G Shobana
Assistant Professor (Sr. Grade), Department of Electronics and Communication Engineering, Mepco Schlenk Engineering College (Autonomous), Sivakasi, Virudhunagar, Tamilnadu, India

Mr Chithiraimuthu R
UG Student, Department of Electronics and Communication Engineering, Mepco Schlenk Engineering College (Autonomous), Sivakasi, Virudhunagar, Tamilnadu, India

Mr Adhithyavel A
UG Student, Department of Electronics and Communication Engineering, Mepco Schlenk Engineering College (Autonomous), Sivakasi, Virudhunagar, Tamilnadu, India

***Abstract*---**Multiplication is a frequently utilized mathematical operation in numerous applications, including signal processing, image processing, and multimedia applications, where multipliers are key components. The speed and large area typically limit their performance. Approximate computing has been critical. These programs are unable to take precise values in specific methods. In them, they like approximate values. Approximate computing methods are used to solve this problem. Approximate multipliers and compressors are constructed by utilizing approximate computation approaches. Because more complicated circuitry is used in many applications, such as DSP, these approximate multipliers and approximate compressors are useful. The suggested work is simulated with Verilog and synthesized with Xilinx ISE 14.6 before being implemented in Spartan – 6. The hardware, size, and speed performance of the suggested multipliers. Both signed and unsigned multipliers are subjected to the multipliers.

***Keywords*---**Approximate Computing, Approximate Multipliers, Approximate Compressors, FPGA.

Introduction

The result of a computer processor is not precise. For instance, adding 1 to 1 can result in 2.01 or 1.98, but not 2. "Almost right" is good enough for many applications, including imaging and artificial intelligence. These chips have fewer circuits and use a lot less energy. Approximate computing is a new paradigm for designing systems that are both energy-efficient and high-performing. Approximate computing is a potential technique for designing energy-efficient VLSI systems and the best-suited DSP and image processing applications. In recent years, approximation computing has mostly been used in the field of VLSI to create DSP applications (i.e., especially in the field of VLSI DSP). Actual computing units do not appear to be required in applications such as multimedia signal processing and digital signal processing that can tolerate mistake. They'll be replaced with their near-identical counterparts. Approximate computing research for error-tolerant applications is becoming more popular. These applications rely heavily on adders and multipliers. The implementations in signed and unsigned multipliers, particularly for signed multipliers, are very useful in VLSI DSP applications. The primary goal of this study is to analyze and compare the performance of the proposed approximate multipliers. In the realm of VLSI DSP, this can help acquire values with greater accuracy and performance. The approximate signed and unsigned multipliers are designed in Verilog, simulated in ModelSim 14.0, and implemented on Spartan - 6 FPGA using Xilinx ISE 14.6

Literature Survey

Approximate Multipliers are based on the technique to building a high-speed multiplier that is both efficient and fast. LUTs are used to create the approximate Baugh – Wooley multiplier, which includes Type – A and Type – B LUTs. Applying the approximate computing method to high–performance approximate multipliers for hardware accelerators compares the critical–path delays (CPD) and Look-Up Table (LUT) that is obtained as a result [1]. These approximate multipliers are utilized in the image blending process based on the results obtained. Approximation adders are employed in the design of approximate multipliers as well. LUTs are used to create approximation adders with low error rates [2]. In the VLSI sector, an adder is a computational circuit for design. These approximate adders are used to improve the design metrics for picture applications. This approximation adder employs a basic half-adder (HA) and full-adder (FA) technique to create new approximate multipliers for image processing applications such as error tolerance [3]. For accelerator architectures, a new design methodology is being used to investigate state-of-the-art approximation adders. In approximation adders, it infers low-power approaches, high-performance analysis, and energy-efficient analysis [4]. Approximate Adders and Multipliers with a Majority of Logic. It is concerned with enhancing. During analysis, performance and consumption reduction are both important [5]. Compressors are used to construct approximation multipliers or just to design an efficient multiplier. As a result, in the multiplication process, approximation compressors are used to partial products to construct an approximate multiplier [6]. In [7], 4:2 approximate compressors are used to create approximate compressors for low-power approximate multipliers. It's made for low-order values

acquired from low-order multiplication. The numerous varieties of approximation compressors, such as 2:1, 3:2, 4:2, 5:3, and 6:3, are designed and explored in [8]. They're made up entirely of simple AND and OR gates. For the higher-order multiplication procedure in [8,] the 6:3 approximate compressors are employed. For both unsigned and signed multipliers, approximation compressors are used in [8]. The suggested approximate adder [9] is used to design unsigned approximate multipliers. [10] describes signed parallel multipliers that use two's complement and use the multiplier's multiplication mechanism, eliminating the need for sign extension in the approach of signed multiplication. The design of an approximate Wallace – Tree multiplier is covered in [11], as well as its comparison to their error-resilient systems. The separate design for DSP is detailed in [12] for implementing their approximation computing in the DSP application by employing approximate multipliers. The new proposed design of the new approximate multiplier[13] is used to provide low-power approximation multipliers for DSP applications. The Design of Vedic multipliers is done by using fault-tolerant is done in[14]. By using reversible logic un Vedic multiplier it is useful in cellular automation in them[14]. The conditional probability is applied to booth multipliers and also it is relevant to approximate multipliers[15]. Approximate Computing is applied to Booth multiplier of Radix-4 in [16] and in [17] error tolerant applications approximate computing applied to both multipliers same implemented in radix-4.

Methodology

Design of Unsigned Multipliers

The type multipliers are unsigned multipliers. Unsigned multipliers, in simple terms, are multipliers in which the multiplication method is the simple method. Array multiplier, Wallace – Tree multiplier, Dada Multipliers, Booth Multipliers, and others are examples of unsigned multipliers. Unsigned multiplier operation applies to these multipliers. The multipliers mentioned above are examples of parallel multipliers. However, each parallel multiplier has its own specialty in comparison to the others. In comparison to typical serial multipliers, the Array multiplier is a parallel multiplier that employs only a few gates. When compared to array and conventional multipliers, the Wallace tree multiplier requires less logic and fewer gates. In the realm of VLSI DSP, this dada multiplier is the principal multiplier utilized in DSP applications.

Array Multiplier

An array multiplier is a digital combinational circuit that uses or is designed to use an array of full adders and half adders to multiply binary values. This array is used to add all of the product keywords in a near-simultaneous fashion. Simple AND and OR gates are used to create this array multiplier. Partially multiplying the multiplied bits and checking them one by one. An array multiplier, on the other hand, has a larger number of gates, making it uneconomical until integrated circuits were introduced. Array means that it is arranged in the means of parallel type multipliers in which this will be the main reason for less delay and area usage compared to conventional multipliers.

Wallace - Tree Multiplier

Because its height is logarithmic in word size rather than linear, the Wallace tree multiplier is much faster than a standard array multiplier. Wallace tree multiplier gate count is lower than array multiplier gate count, and the area occupied is likewise lower. The partial product passing and procedures, however, are identical to those used by array and traditional multipliers. The key difference is that they employ fewer gates and have a smaller coverage area. As a result, some designers use the Wallace tree multiplier, although in most cases, designers avoid Wallace tree multipliers because of their design complexity. The partial products obtained in a Wallace – Tree multiplier are vertically propagated, and the carry is propagated to the following stage in a vertical manner as well, which is the main reason for its complexity, but on the other hand, it covers a smaller number of gates.

Design of Signed Multipliers

The process of multiplying two numbers is known as multiplication. In binary multiplication, there are two types of multiplication. There are two types of multiplication: signed and unsigned. Unsigned multiplication is a common method of multiplication that is used all throughout the world. Negative number multiplication, on the other hand, uses signed multiplication. This digitally signed multiplication employs the 2's complement method.

The extension method is utilised for this signed multiplication sign. Sign extension is a computer arithmetic procedure that increases the amount of bits in a binary integer while keeping the sign and value of the number. This is accomplished by appending digits to the number's most significant side, according to a technique that varies depending on the signed number representation utilised. If the positive number 5 is represented in binary as 0101, for example, A -5 negative number, on the other hand, will be represented in binary as 1011That is, normal positive only uses all of the bits represented by the numbers 8 4 2 1.

The Most Significant Bit (MSB) is taken as negative (i.e. -8 4 2 1) in negative. For binary multiplication, this is the approach used in signed multiplication. However, because of the sign extension for signed multiplication, the binary bits will be extended, requiring more logic to be employed, which will increase implementation time and gate-level. As result, the Baugh – Wooley multiplier is used to solve this problem.

Baugh - Wooley Multiplier

There is no need for a sign extension method in this Baugh – Wooley multiplier because it uses 2's complement. The Baugh – Wooley multiplication approach is introduced to address the limitations of the sign extension method. It is employed in the multiplication of signed numbers. To construct direct multiplication of signed numbers, the Baugh Wooley technique was devised. Each of the partial products to be added when multiplying two's complement integers directly is a signed number.

$$A = -a_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-1}(a_i 2^i) \qquad (1)$$

$$B = -b_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-1}(b_i 2^i) \quad (2)$$

$$P = A \times B \quad (3)$$

$$P = (-a_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-1}(a_i 2^i)) \times (-b_{n-1}\ 2^{n-1} + \sum_{i=0}^{n-1}(b_i 2^i)) \quad (4)$$

This multiplier was first introduced by Baugh and Wooley. The main advantage of the Baugh-Wooley multiplier is that it may be used for both signed and unsigned multiplication. This is a parallel multiplier as well. It is also quick and compactible as compared to other parallel multipliers. It has a smaller zone, less duration, and minimal power scattering. The Baugh Wooley multiplier's architecture relies heavily on carry save calculation.

								a7	a6	a5	a4	a3	a2	a1	a0
								b7	b6	b5	b4	b3	b2	b1	b0
							1	$\overline{a7b0}$	a6b0	a5b0	a4b0	a3b0	a2b0	a1b0	a0b0
							$\overline{a7b1}$	a6b1	a5b1	a4b1	a3b1	a2b1	a1b1	a0b1	
						$\overline{a7b2}$	a6b2	a5b2	a4b2	a3b2	a2b2	a1b2	a0b2		
					$\overline{a7b2}$	a6b3	a5b3	a4b3	a3b3	a2b3	a1b3	a0b3			
				$\overline{a7b4}$	a6b4	a5b4	a4b4	a3b4	a2b4	a1b4	a0b4				
			$\overline{a7b5}$	a6b5	a5b5	a4b5	a3b5	a2b5	a1b5	a0b5					
		$\overline{a7b6}$	a6b6	a5b6	a4b6	a3b6	a2b6	a1b6	a0b6						
1	a7b7	$\overline{a7b6}$	$\overline{a7b5}$	$\overline{a7b4}$	$\overline{a7b3}$	$\overline{a7b2}$	$\overline{a7b1}$	$\overline{a7b0}$							
P15	**P14**	**P13**	**P12**	**P11**	**P10**	**P9**	**P8**	**P7**	**P6**	**P5**	**P4**	**P3**	**P2**	**P1**	**P0**

Figure 1 Method in Baugh – Wooley Multiplier

Design of 4:2 Approximate Compressors

Compressors are single-bit adder circuits with more than three inputs (as in a full adder) but fewer outputs than a full adder. It's also utilized in high-speed multipliers, and it just requires a small number of gates to create. Compressors are being improved further, with 4:2 compressors being created for higher-order compressors. XOR – XNOR is used to design a simple 4:2 compressor, but the design is adjusted for higher-order 4:2 compressors. Low power consumption is observed in higher-order compressors.

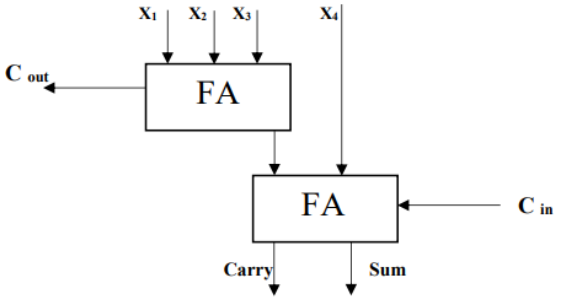

Figure 2 Conventional 4:2 Compressor

A 4:2 compressor, as shown in Figure 2, is used to create higher-order compressors. This compressor is also used to create multipliers. The 4:2 compressor, as shown in Figure 2, is made up of two complete adders. This specific compressor design has full adders in them.

$$Sum = x1 \oplus x2 \oplus x3 \oplus x4 \oplus Cin \quad (5)$$

$$C_{out} = (x1 \oplus x2)\, x3 + (x1 \oplus x2)\, \grave{}x1 \quad (6)$$

$$Carry = (x1 \oplus x2 \oplus x3 \oplus x4)\, Cin + (x1 \oplus x2 \oplus x3 \oplus x4)\, \grave{}x4 \quad (7)$$

The error rate for 4:2 compressors is around 53%, as seen above. In compressors, approximate computation is used to circumvent this constraint in the field of DSP applications. This approximation compressor has only unneeded hardware circuitry and uses less energy than 4:2 exact compressors. The mistake rate is cut in half when accurate compressors are used. These are three designs for approximation compressors based on their hardware circuits. The equations shown in 5, 6, 7 explains the sum, C_{out} and carry in them. Nowadays, the use of compressors for design of low power design of compressor sin them. By using these compressors like these methods applied we can design some efficient compressors. It reduces impact of carry propagation of large data matrix. The compressor designed by own necessary parameters as available.

Table-1
Truth Table for 4:2 Conventional Compressor

C_{in}	X_4	X_3	X_2	X_1	C_{out}	Carry	Sum
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	1	1	0	0
0	0	1	0	0	0	0	1
0	0	1	0	1	1	0	0
0	0	1	1	0	1	0	0
0	0	1	1	1	1	0	1
0	1	0	0	0	0	0	1
0	1	0	0	1	0	1	0
0	1	0	1	0	0	1	0
0	1	0	1	1	1	0	1
0	1	1	0	0	0	1	0
0	1	1	0	1	1	0	1
0	1	1	1	0	1	0	1
0	1	1	1	1	1	1	0

(a) Table-1 Contd

C_{in}	X_4	X_3	X_2	X_1	C_{out}	Carry	Sum
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	1	0
1	0	0	1	1	1	0	1
1	0	1	0	0	0	1	0
1	0	1	0	1	1	0	1
1	0	1	1	0	1	0	1
1	0	1	1	1	1	1	0
1	1	0	0	0	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	0	1	0	1	1
1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1

Approximate Compressor Design - I

NAND and XNOR gates are used in Design 1, which demand less supply voltage and so utilise less power. NAND and XNOR gates take up less space and burn less power because they are implemented in CMOS technology.

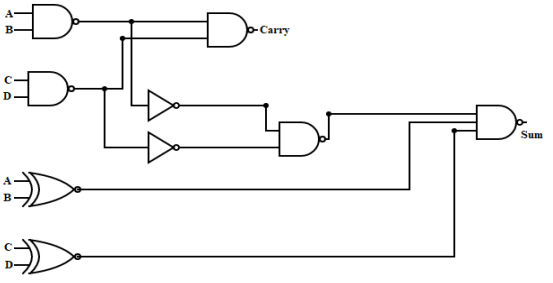

Figure-3 Approximate Compressor Design-I

Carry = ~{[~(AB)].[~(CD)]} (8)

Sum = ~{$(A \otimes B)(C \otimes D)${~[$(AB)(CD)$]}} (9)

Table-2
Truth Table for 4:2 Approximate Design-1 Compressor

A	B	C	D	Carry	Sum
0	0	0	0	0	0
0	0	0	1	0	1

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

Approximate Compressor Design - II

In contrast to design #1, the suggested approximation compressor uses two AND gates and two 3-input OR gates, counting the number of 1s in the value rather than the value itself. The output equations for the estimated design #2 compressor are as follows:

$$Carry = (CD) + A + B \quad (10)$$

$$Sum = (AB) + C + D \quad (11)$$

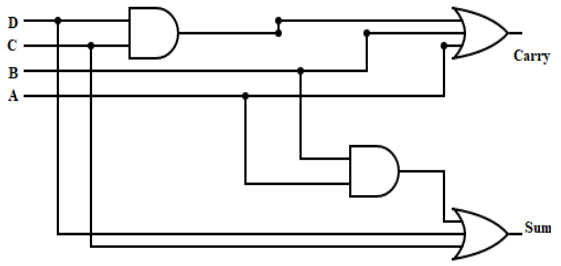

Figure 4 Approximate Compressor Design – II

Table-3
Truth Table for 4:2 Approximate Design-2 Compressor

A	B	C	D	Carry	Sum
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	1
0	0	1	1	1	1
0	1	0	0	1	0
0	1	0	1	1	1
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

Approximate Compressor Design- III

With only two 2-input OR gates, the suggested approximation 4-2 compressor is implemented with minimal hardware circuitry. Figure depicts the proposed 4-2 compressor. When compared to prior approximate compressors, this approximately design requires less hardware circuitry and has a lower gate count.

Carry = A + C (12)

Sum = B + D (13)

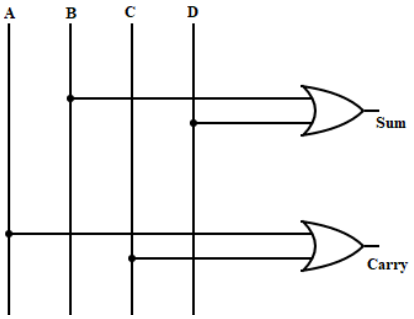

Figure-5 Approximate Compressor Design – III

Table-4
Truth Table Approximate Design 3 Compressor

A	B	C	D	Carry	Sum
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

Design of Approximate Multipliers

Approximate computation is used to produce approximate values. Designing approximate multipliers can be done in a variety of ways. The approximate multipliers are developed utilising the own method's designing LUTs. The suggested method involves designing approximation multipliers by using approximate compressors in the partial products of multipliers during the multiplication process. It is depicted in the diagram. The partial products are compressed, or in other words, decreased, as indicated in Figure 3.5.1, by utilizing 4:2 approximation compressors. The 4:2 Approximate compressors are applied to the LSB side (i.e. the Least Significant Bit side) to approximate it. However, on the MSB side (i.e., the Most Significant Bit side), it should not be approximated or lowered. This is because the MSB side contains larger bits, and if they are approximated, the overall result will not be as expected.

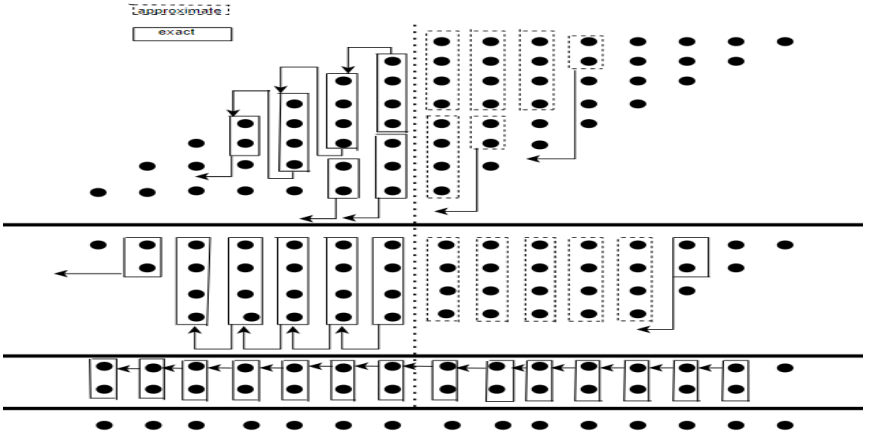

Figure-6 Approximate Multiplier Design

The LSB is simply approximated, as seen in Figure 7. It is applied to all multipliers that are calculated and approximated. The Baugh – Wooley multiplier, the Array multiplier, and the Wallace – Tree multiplier are all estimated. Design 3 approximate compressors are more efficient than other designs.

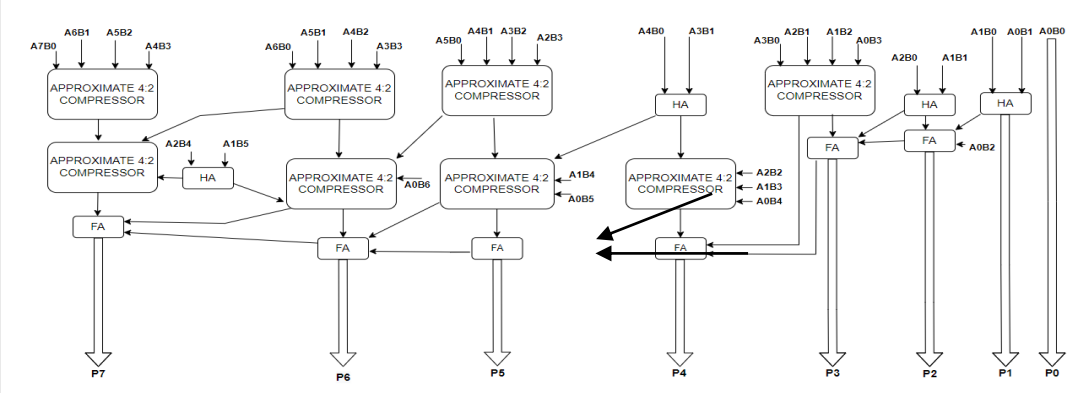

Figure-7 Approximate 4:2 Multiplier Design on LSB side

Results and discussions

ModelSim Altera 14.0 is used to model the proposed approximation multiplier work, which is then synthesised by Xilinx 14.6 and implemented on a Spartan – 6 FPGA. It is designed as 8x8 multipliers in simulation outcomes, and the input is delivered as 8 – input into them.

Simulation Results

Simulation results for

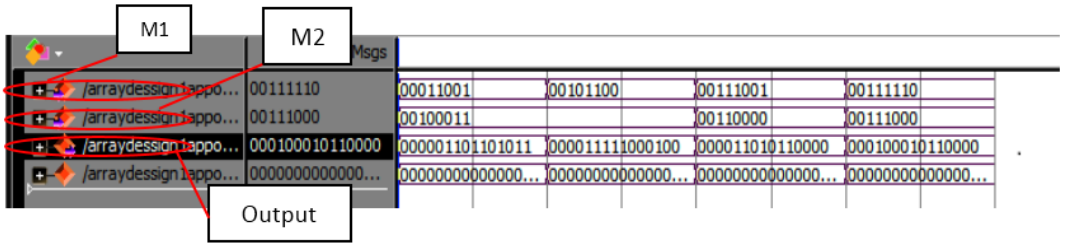

Figure-8 Simulation result for Approximate array multiplier Using Design 3 Compressor

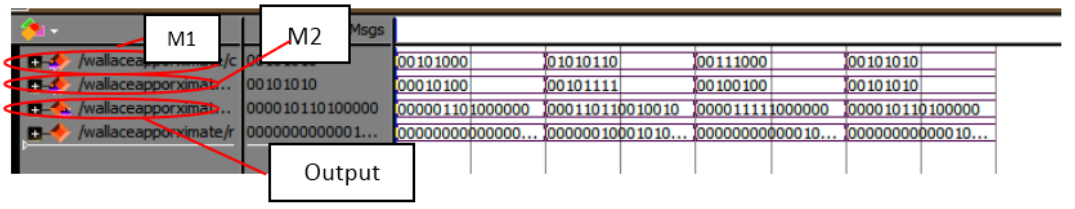

Figure-9 Simulation result for Approximate Wallace-Tree multiplier Using Design 2 Compressor

Figure-10 Simulation result for Approximate array multiplier Using Design III Compressor

As shown in Figures 8, 9, 10 are the simulation results for approximate multipliers design-3 for all types of multipliers. The reason is that they have low gate count in them and also they have better efficiency in them so Design -3 has better efficiency. Where M1, M2 are inputs. *4.2 Performance measures:*
Error Analysis between three approximate design multipliers can be determined by two parameters namely Error Percentage (EP), Average Error Percentage (AEP), Maximum Error Possibility (MEP). The formulae for these two are given below as, where N is total number of samples

Difference = Original Value – Obtained Value (14)
Error Possibility = Difference ÷ Original Value (15)

EP = Error Possibility x 100 (16)

$$AEP = \sum_{i=1}^{N}\left(\frac{Epi}{N}\right) \quad (17)$$

Test case results:

Table 5, Table 6 and Table 7 shows the Test cases for designed approximate Array multipliers, approximate Wallace-tree multipliers and for approximate Baugh-Wooley multiplers. Multiplication of two random numbers between 0 to 255 is done. Error percentage for the same has been calculated using the equation (14), (15) and finally with (16). The error analysis is performed for almost different possible input vectors for 8- bit multiplication. In this it represents which has low error percentage values they are the best of those cases. Table 8 and Table 9 shows the Comparison results for AEP (i.e) Average Error Percentage and the comparison results for MPE (i.e) Maximum Percentage Error as shown. In this AEP is calculated by the equation (17).

(i) Test case results for Approximate Array Multiplier for three designs

Table-5
Test cases for Approximate Array Multipliers for Design-I, II, III

S.no	M1	M2	Original Value	AD I	AD II	AD III	EP (AD I)	EP (AD II)	EP (AD III)
1.	42	42	1764	1556	1780	1700	11.79	-0.90702	3.628
2.	45	36	1620	1972	1668	1652	-21.72	-2.962	-1.975
3.	60	40	2400	2208	2560	2464	8	-6.667	-2.667
4.	65	49	3185	3121	3121	3249	2.0094	2.0094	-2.0094
5.	70	52	3640	3128	4040	3808	14.065	-10.989	-4.615
6.	73	56	4088	4536	4344	4280	-10.958	-6.262	-4.696
7.	79	60	4740	5060	4844	4588	-6.751	-2.194	3.206
8.	82	65	5330	5778	5794	5538	-8.705	-8.705	-3.902
9.	84	68	5712	5648	5856	5776	1.120	-2.521	-1.1204
10.	91	70	6370	6082	7090	6170	4.521	-11.302	3.139

(ii) Test case results for Approximate Wallace-Tree Multiplier

Table-6
Test cases for Approximate Wallace-Tree Multipliers for Design-I, II, III

S.no	M1	M2	Original Value	AD I	AD II	AD III	EP (AD I)	EP (AD II)	EP (AD III)
1.	42	42	1764	1360	1776	1440	22.902	-0.68027	18.36
2.	45	36	1620	1456	1664	1648	10.12	-2.716	-1.728
3.	60	40	2400	2016	2496	2240	16	-4	6.667
4.	65	49	3185	3185	3377	3249	0	-6.0282	-2.0094
5.	70	52	3640	3644	3532	3812	-0.1098	2.96	-4.725
6.	73	56	4088	4092	4028	4028	-0.09784	1.467	1.467

7.	79	60	4740	4488	4832	4848	5.316	-1.9409	-2.27
8.	82	65	5330	5330	5538	5538	0	-3.902	-3.902
9.	84	68	5712	5456	5536	5520	4.481	3.081	3.361
10.	91	70	6370	6142	6062	6454	3.579	4.835	-1.318

(iii) Test case results for Approximate Baugh-Wooley Multiplier

Table-7
Test cases for Approximate Baugh-Wooley Multipliers for Design-I, II, III

S.no	M1	M2	Original Value	AD I	AD II	AD III	EP (AD I)	EP (AD II)	EP (AD III)
1.	42	-42	-1764	-6388	-3468	-18740	-262.13	-96.59	-962
2.	-45	36	-1620	-5982	-108	-18996	-269.25	93.33	-1072
3.	60	40	2400	11808	7808	20928	-392	-225.33	-772
4.	-65	-49	3185	185	10745	32263	50	-237.36	-912
5.	70	52	3640	1592	5064	23005	56.26	-39.120	-532
6.	-73	56	-4088	-6888	-664	-17432	-68.49	83.757	-326.4
7.	79	-60	-4740	-12020	-10068	-19556	-153	-112.4	-312.57
8.	-82	-65	5330	2930	11250	27602	45.028	-111	-417.86
9.	84	68	5712	3632	7200	20880	36.4	-26.050	-265.54
10.	-91	-70	6370	4106	12362	28992	35.54	-94.065	-355.13

Comparison of AEP and MPE

AEP – Average Error Percentage MPE – Maximum Percentage Error

Table-8
Comparison of AEP for Approximate Multipliers

MULTIPLIERS	AEP For AD I	AEP For AD II	AEP For AD III
Array Multiplier	6.3	5.04	0.89
Wallace-Tree Multiplier	6.2	2.7	1.39
Baugh-Wooley Multiplier	92.22	76.4	83.5

Table-9
Comparison of MPE for Approximate Multipliers

MULTIPLIERS	MPE For AD I	MPE For AD II	MPE For AD III
Array Multiplier	~11	~2	~4

Wallace-Tree Multiplier	~23	~5	~18
Baugh-Wooley Multiplier	~50	~93	~26

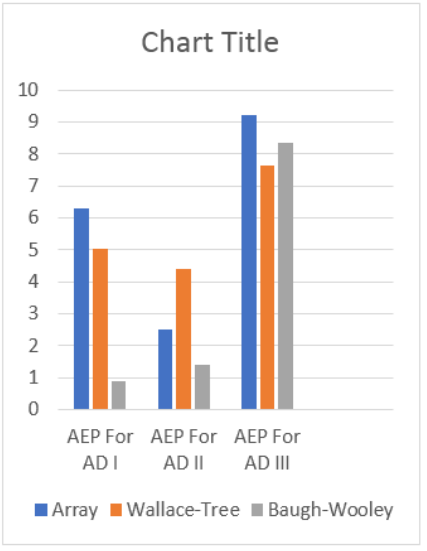

Figure-15 Graphical representation for AEP of Approximate Multipliers

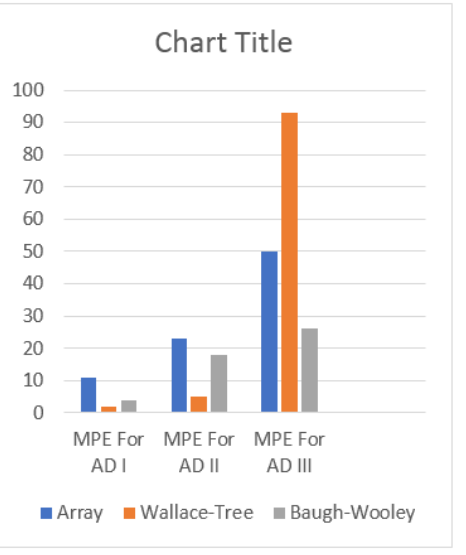

Figure-16 Graphical representation for MPE of Approximate Multipliers

Implementation results

The designed multiplier is designed in Verilog and simulated by using Modelsim Altera 10.e and it is implemented in Spartan-6 FPGA successfully. The implementation result is shown in the format of RTL Schematic representation as shown in figure 15. It shows the representation for Approximate multiplier using Design-3 Approximate Compressor. The Figure 15 and Figure 16 shows the graphical representation of Average Error Percentage (AEP) and also for Maximum Percentage Error (MPE).

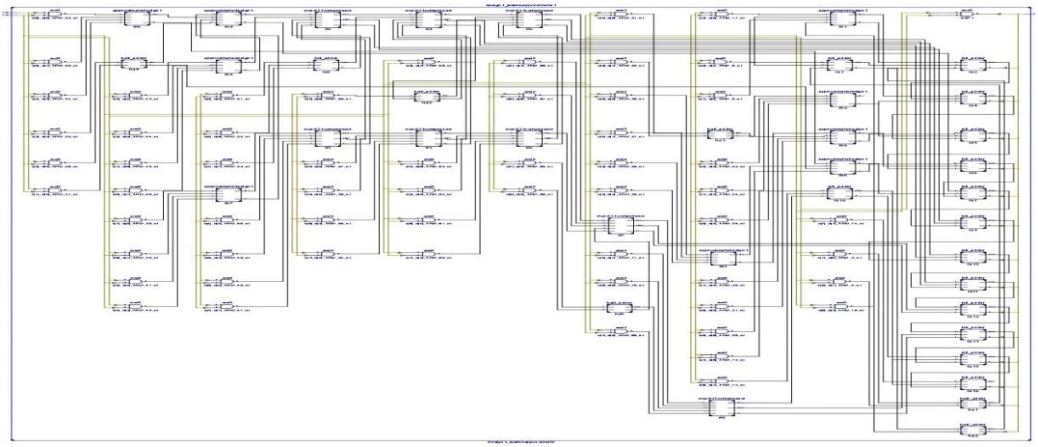

Figure-17. RTL Schematic Representation for Approximate Array Multiplier Design – III

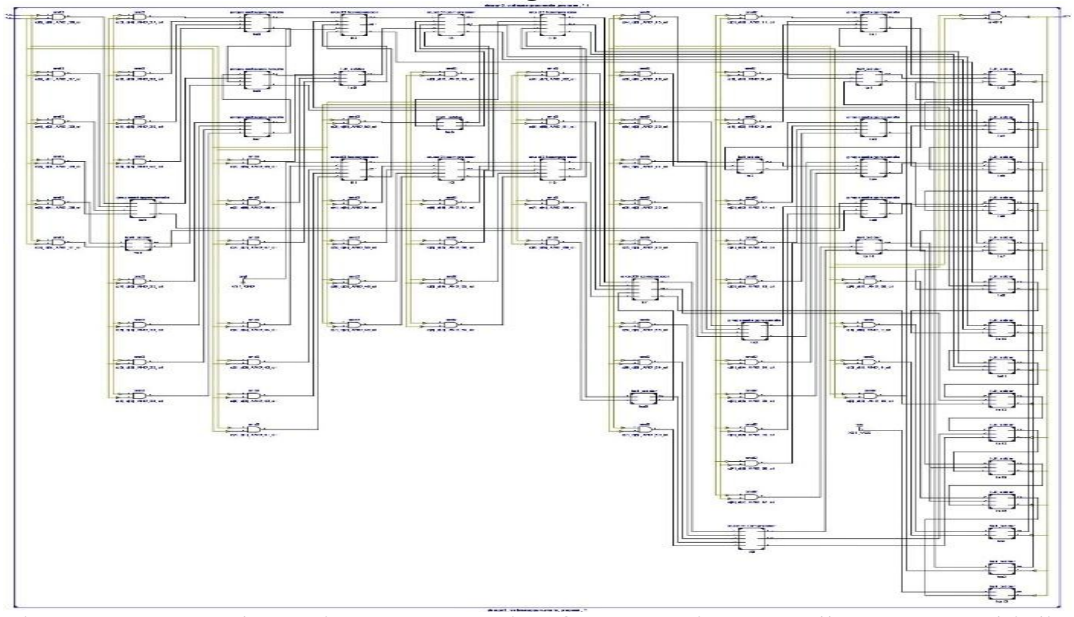

Figure-18. RTL Schematic Representation for Approximate Wallace-Tree Multiplier Design – III

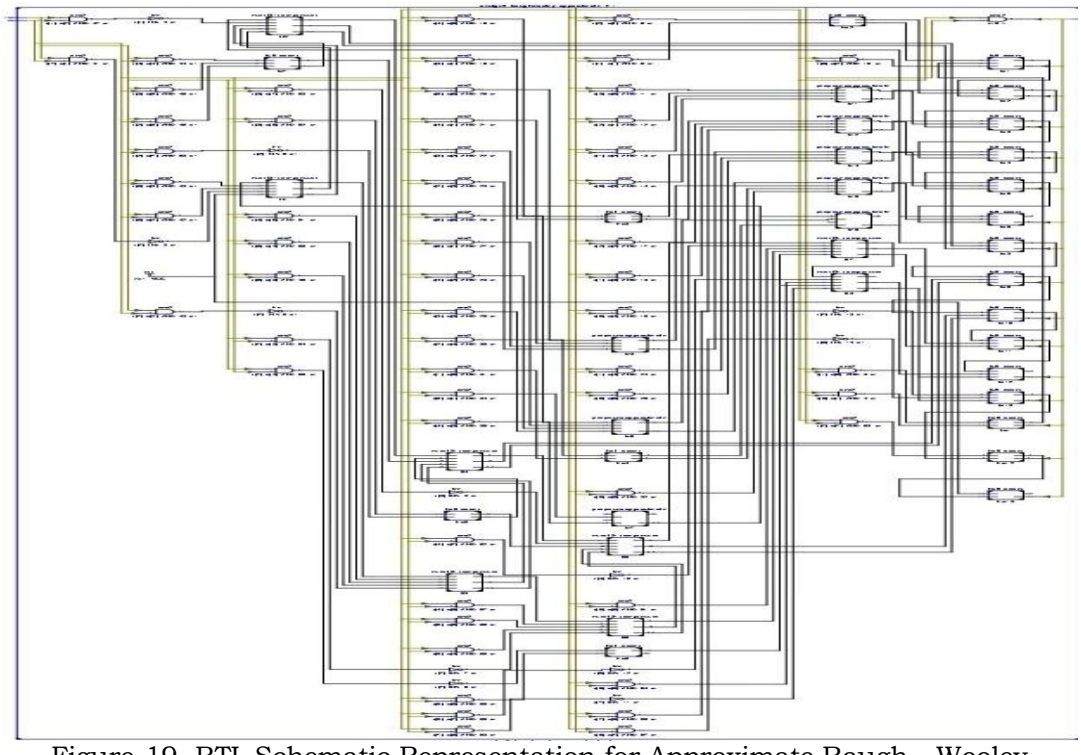

Figure-19. RTL Schematic Representation for Approximate Baugh - Wooley Multiplier

Design-III

Device utilisation:

The simulated results are taken in ModelSim Altera 10.e and then synthesized Xilinx ISE 14.6 and results such as delay, and resources utilized are tabulated. It is tabulated in Table-10. For device utilization it compared Number of LUTs, Slices, delays, logic, bonded IOBs and number of XORs.

Table-10
Device Utilization for Multipliers

Multipliers	Number of Occupied Slices	Number of Slice LUTs	Delay (ns)	Number of used Logic	Bonded IOBs	XORS
Array Multiplier	34	84	29.703ns	84	32	104
Wallace-Tree Multiplier	34	82	23.949ns	84	32	104
Baugh-Wooley Multiplier	39	95	20.396ns	95	32	127
Approximate Array Multiplier Design-I	30	69	18.475ns	69	32	85
Approximate Array	25	64	18.781n	64	32	67

Multiplier Design-II			s			
Approximate Array Multiplier Design-III	21	51	17.388n s	51	32	67
Approximate Wallace-Tree Multiplier Design-I	24	62	18.080n s	69	32	86
Approximate Wallace-Tree Multiplier Design-II	27	57	18.781n s	64	32	65
Approximate Wallace-Tree Multiplier Design-III	21	51	16.110n s	51	32	65
Approximate Baugh-Wooley Multiplier Design-I	28	68	18.30ns	68	32	87
Approximate Baugh-Wooley Multiplier Design-II	27	63	18.795n s	63	32	69
Approximate Baugh-Wooley Multiplier Design-III	25	56	16.986n s	56	32	69

As above Table 10 shows the device utilized by Array Multiplier, Wallace-Tree Multiplier, Baugh-Wooley Multiplier and also its Approximate Multipliers for the Designs-1,2,3. As shown in the Tabulations Design-3 Approximate Design is in efficient way.

Table-11
Comparison of Parameters between 4:2 Approximate Compressors for Design – I, II, III

Contents	Exact 4:2 Compressor	Approximate 4:2 Compressor Design-I	Approximate 4:2 Compressor Design-II	Approximate 4:2 Compressor Design-III
Number of Occupied Slices	2	1	1	1
Number of Slice LUTs	2	1	1	1
Delay (ns)	6.331ns	6.177ns	6.177ns	5.961ns
Number of used Logic	2	1	1	1
Bonded IOBs	8	6	6	6

As above Table shows the comparison of parameters for 4:2 Exact Compressor and also with approximate compressors for all Designs-I, II, III. From these its determines that Design-III is efficient.

Conclusion and future scope

The Approximate Multipliers for both unsigned multipliers and signed multipliers like Array multiplier, Wallace-Tree multipliers, and Baugh-Wooley multiplier are designed and simulated using Modelsim Altera 10. e followed by synthesis done in Xilinx ISE 14.6 and is implemented in Spartan-6 device in Xilinx Plan Ahead Tool. The method to design approximate multipliers by using approximate 4:2 compressors in them. The test cases and error analysis are performed for almost 20 different input vectors for all possible input vectors for 8-bit multiplication. The efficiency is improved for Design – III. It is evaluated in two parameters such Average Error Percentage (AEP) and Maximum Possible Error (MPE). It is inferred that approximate multipliers have very perfect approximate values in them. Hence in the field of Image Processing filtering methods like Median filtering, Mean filtering methods can be designed. Hence the designed multiplier can be utilized for FIR, IIR filter DSP applications.

References

[1] Salim Ullah; Semeen Rehman; Muhammad Shafique; Akash Kumar; (2021*).High-Performance Accurate and Approximate Multipliers for FPGA-Based Hardware Accelerators.* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, (), –doi:10.1109/tcad.2021.3056337

[2] Waqar Ahmad; Berke Ayrancioglu; Ilker Hamzaoglu; (2021). *Low Error Efficient Approximate Adders for FPGAs*. IEEE Access, (),doi:10.1109/access.2021.310737

[3] Anusha, G.; Deepa, P. (2020). *Design of approximate adders and multipliers for error tolerant image processing*. Microprocessors and Microsystems, 72(), 102940–.doi:10.1016/j.micpro.2019.102940

[4] Soares, Leonardo Bandeira; da Rosa, Morgana Macedo Azevedo; Diniz, Claudio Machado; da Costa, Eduardo Antonio Cesar; Bampi, Sergio (2019). *Design Methodology to Explore Hybrid Approximate Adders for Energy-Efficient Image and Video Processing Accelerators.*. IEEE Transactions on Circuits and Systems I: Regular Papers, (), 1–14.doi:10.1109/TCSI.2019.2892588

[5] Liu, Weiqiang; Zhang, Tingting; McLarnon, Emma; Oneill, Maire; Montuschi, Paolo; Lombardi, Fabrizio (2019). *Design and Analysis of Majority Logic Based Approximate Adders and Multipliers*. IEEE Transactions on Emerging Topics in Computing, (), 1–1. doi:10.1109/TETC.2019.2929100

[6] Chinthalgiri Jyothi;Kuruva Gayathri;Saranya Karunamurthi;Sreehari Veeramachaneni; Noor Mahammad S; (2020). *Area Efficient Approximate 4–2 Compressor for Multiplier Design* . 2020 IEEE India Council International Subsections Conference (INDISCON), (), –. doi:10.1109/indiscon50162.2020.00055

[7] Strollo, Antonio Giuseppe Maria; Naples, Ettore; De Caro, Davide; Petra, Nicola; Meo, Gennaro Di (2020). *Comparison and Extension of Approximate 4-2 Compressors for Low-Power Approximate Multipliers* . IEEE Transactions on Circuits and Systems I: Regular Papers, (), 1–14. doi: 10.1109 / TCSI.2020.2988353

[8] Esposito, Darjn; Strollo, Antonio Giuseppe Maria; Naples, Ettore; De Caro, Davide; Petra, Nicola (2018*). Approximate Multipliers Based on New*

Approximate Compressors. IEEE Transactions on Circuits and Systems I: Regular Papers, (), 1–14.doi: 10.1109 / TCSI.2018.2839266

[9] Jiang, Honglan; Liu, Cong; Lombardi, Fabrizio; Han, Jie (2018). *Low-Power Approximate Unsigned Multipliers With Configurable Error Recovery* . IEEE Transactions on Circuits and Systems I: Regular Papers, (), 1–14.doi:10.1109/TCSI.2018.2856245

[10] Baugh, C.R.; Wooley, B.A. (1973). *A Two's Complement Parallel Array Multiplication Algorithm*. ,C-22(12), 1045–1047.doi:10.1109/t-c.1973.223648

[11] Bhardwaj, Kartikeya; Mane, Pravin S.; Henkel, Jorg (2014). [IEEE 2014 15th International Symposium on Quality Electronic Design (ISQED) - Santa Clara, CA, USA (2014.03.3-2014.03.5)] Fifteenth International Symposium on Quality Electronic *Design - Power- and area-efficient Approximate Wallace Tree Multiplier for error-resilient systems.* , (), 263–269. doi:10.1109/ISQED.2014.6783335

[12] Zhao, Yue; Li, Tong; Dong, Feng; Wang, Qin; He, Weifeng; Jiang, Jianfei (2019). IEEE 2019 IEEE 13th International Conference on ASIC (ASICON) - Chongqing, China (2019.10.29-2019.11.1)] 2019 IEEE 13th International Conference on ASIC (ASICON) –*A New Approximate Multiplier Design for Digital Signal Processing.* , (), 1–4. doi:10.1109/asicon47005.2019.8983437

[13] Farshchi, Farzad; Abrishami, Muhammad Saeed; Fakhraie, Sied Mehdi (2013). IEEE 2013 17th CSI International Symposium on Computer Architecture and Digital Systems (CADS) - Tehran, Iran (2013.10.30-2013.10.31)] The 17th CSI International Symposium on Computer Architecture & Digital Systems (CADS 2013) – *New approximate multiplier for low power digital signal processing*. (), 25–30.doi:10.1109/cads.2013.6714233

[14] Dr.A.Kamaraj *Realisation of Vedic sutras for multiplication in Verilog* in SSRG International Journal of VLSI & Signal Processing Impact Factor=-1.0000; Paper Online Link: -1; April 2017, DOI: -1; Volume No: 4, Issue No: 2; PageNo: 18-22

[15] Mr.G.Kirubakaran, Dr.Seshasayanan *A precise positioning conditional probability estimator for fixed width booth multipliers* Proceedings of IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI) Vol.1, Issue.1, pp.1892-1896, June-2018 , DOI: 10.1109/ICPCSI.2017.8392044

[16] KimKibeom HwangSeokha LeeYoungjoo LeeSunggu *Approximate Radix-4 Booth Multiplication Circuit* JSTS(Journal of Semiconductor Technology and Science)IEIE Vol. 19, No. 5, p.435-445, 18 August 2019 DOI :https://doi.org/10.5573/JSTS.2019.19.5.435

[17] Liu W., Qian L., Wang C., Jiang H., Han J., Lombardi F., 2017*, Design of approximate radix-4 booth multipliers for error-tolerant computing*, IEEE Transactions on Computers, Vol. 66, No. 8, pp. 1435-1441