DESIGN OF POWER AND AREA EFFICIENT CSLA BASED MULTIPLIER FOR GAUSSIAN FILTER

Dr Ravi Sravanthi [1] **Shaik Reshma[2]**

[1]Associate Professor, Department of Electronics and Communication Engineering, PBR Visvodaya Institute of Technology & Science, Kavali, Nellore (D.t), Andhra Pradesh, India.

[2]PG Scholar, Department of Electronics and Communication Engineering, PBR Visvodaya Institute of Technology & Science, Kavali, Nellore (D.t), Andhra Pradesh, India.

Email id: reshmashaik1995@gmail.com

Abstract: Approximate computing device in a digital system reduces the design complexity and provide the high efficient solutions in order to increases its performance. The exact computing device is not always necessary for multimedia signal processing and data mining, which are capable of tolerate error. The proposed design of the approximate multiplier can able to reduce the design complexity and reduces its area in order to increases its performance. In this paper approximate multiplier, Truncation concept is used in order to reduces its area as well as power consumption when compared to normal approximate multiplier which is used in the existing system. This multiplier technique is fully focused on partial products accumulation which is very important in terms of power consumption. Therefore, the proposed truncation multiplier saves few adders in partial products, and evaluated with a image processing application. The existing system focused only on luminance based application, but the proposed work focused on both luminance and chrominance based application. Finally implement this proposed work in VHDL and synthesized in the XILINX-S6LX9 FPGA and compared in terms of area, power and delay reports.

Keywords: FPGA, Ripple Carry Adder, Multipliers, Gaussian Filter.

I. INTRODUCTION

The invention of transistor by William B. Shockley, Walter H. Brattain and John Bardeen of bell telephone laboratories drastically change the electronics industry and paved the way for the development of the Integrated Circuit (IC) technology. The first IC was designed by jack Kilby at Texas instruments at the beginning of 1960 and since that time there have already been four generations of ICs. Viz SSI (Small Scale integration), MSI (Medium Scale integration), LSI (Large Scale integration), and VLSI (Very Large Scale integration). Now we are already to see the emergence of the fifth generation, ULSI (Ultra Large Scale integration) which is characterized by complexities in excess of 3 million devices on a single IC chip. Further miniaturization is still to come and more revolutionary advances in the application of this technology must inevitably occur. Over the past several years, silicon CMOS technology has become the dominant fabrication process for relatively high performance and cost effective VLSI circuits. The revolutionary nature of this development is understood by the rapid growth in which the number of transistors integrated in circuits on a single chip.

Very-large scale integration (VLSI) is the process of creating an integrated circuit(IC) by combining thousands of transistors into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed.

INTRODUCTION TO VERILOG:

Verilog is a HARDWARE DESCRIPTION LANGUAGE (HDL). It is a language used for describing system like a network switch or a microprocessor or a memory or a flip-flop. It means, by using a

HDL we can describe any digital hardware at any level. Designs, which are described in HDL are independent of technology, very easy for designing and debugging, and normally more useful than schematics, particularly for large circuits.
Verilog supports a design at many levels of abstraction. The major three are:

1. Behavioral level
2. Register-transfer level
3. Gate level

FPGA INTRODUCTION:

The full form of FPGA is "Field Programmable Gate Array". It contains ten thousand to more than a million logic gates with programmable interconnection. Programmable interconnections are available for users or designers to perform given functions easily. A typical model FPGA chip is shown in the figure. There are I/O blocks, which are designed and numbered according to function. For each module of logic level composition, there are CLB's (Configurable Logic Blocks).

Field Programmable Gate Array (FPGA) devices were introduced by Xilinx in the mid-1980s. They differ from CPLDs in architecture, storage technology, number of built-in features, and cost, and are aimed at the implementation of high performance large-size circuits

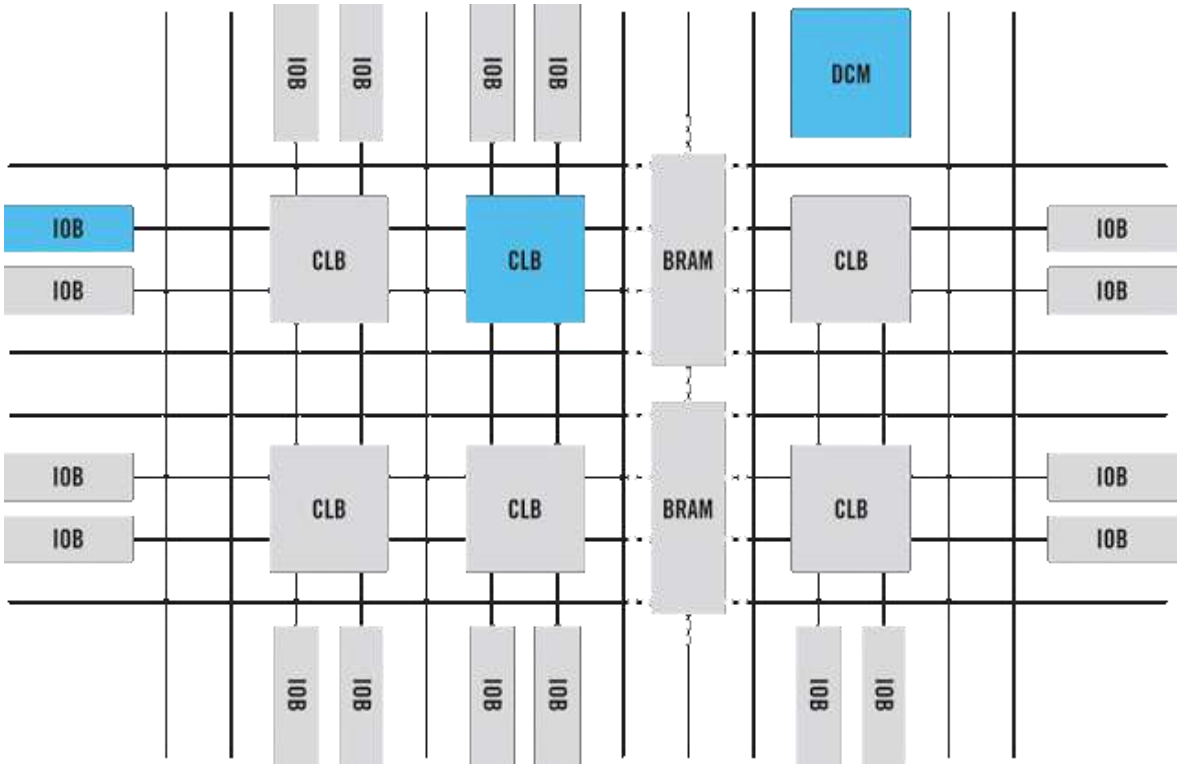


Fig.1.1. FPGA Architecture

LITERATURE SERVAY:

ADDER

An adder is a digital circuit that performs addition of numbers. In many computers and kinds of processors adders are used in the arithmetic logic units or ALU. They are also utilized in other parts of the processors, where they are used to calculate addresses, table indices, increment and decrement operators, and similar operations.

Addition is one of the most basic and primary arithmetic operation which adds the two binary (1, 0) digits. A circuit is said to be combinational circuit when it adds the two binary bits, according to the example demonstrated below, is called a half adder. A circuit that sums three bits, the third bit generated from the summation of previous least significant bit(LSB), these type of circuits are known as full adder circuits.

APPROXIMATION OF PARTIAL PRODUCTS:

The accumulation of other partial products with probability 1/4for am, n and 7/16 for pm, n uses approximate circuits. Approximate half-adder, full-adder, and 4-2 compressor are proposed for their accumulation. Carry and Sum are two outputs of these approximate circuits. Since Carry has higher weight of binary bit, error in Carry bit will contribute more by producing error difference of two in the output. Approximation is handled in such a way that the absolute difference between actual output and approximate output is always maintained as one. Hence Carry outputs are approximated only for the cases, where Sum is approximated. In adders and compressors, XOR gates tend to contribute to high

area and delay. For approximating half-adder, XOR gate of Sum is replaced with OR gate as given in (2). This results in one error in the Sum computation as seen in the truth table of approximate half-adder in Table II. A tick mark denotes that approximate output matches with correct output and cross mark denotes mismatch

$$Sum = x1 + x2$$
$$Carry = x1 \cdot x2 \text{..... (3)}$$

In the approximation of full-adder, one of the two XOR gates is replaced with OR gate in Sum calculation. This results in error in last two cases out of eight cases. Carry is modified as in (3) introducing one error. This provides more simplification, while maintaining the difference between original and approximate value as one. The truth table of approximate

$$W = (x1 + x2)$$
$$Sum = W \oplus x3$$
$$Carry = W \cdot x3 \text{.....} (4)$$

3 REDUCTION OF ALTERED PARTIAL PRODUCTS:

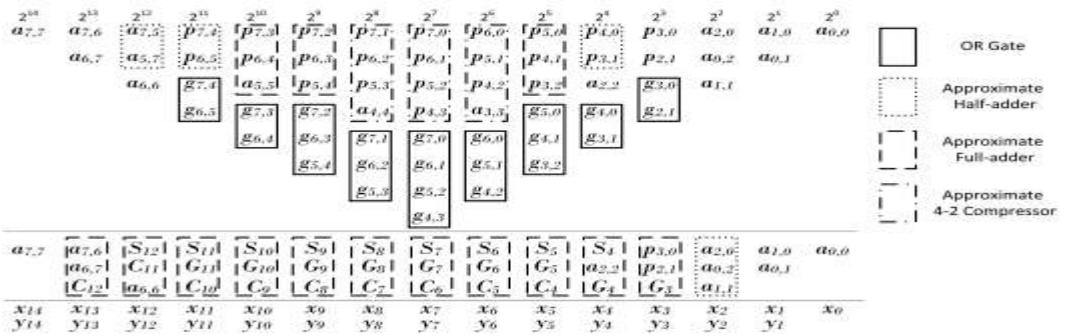


Fig 1.2: Reduction of altered Partial Products

The accumulation of generate signals is done column wise. As each
Element has a probability of 1/16 of being one, two elements being 1in the same column even decreases. For example, in a column with 4 generate signals, probability of all numbers being 0 is(1−pr)4, only one element being one is 4pr(1−pr)3, the probability of two elements being one in the column is 6pr2(1−pr)2, three ones is4pr3(1−pr) and probability of all elements being 1 ispr4,wherepris 1/16. The probability statistics for a number of generate elements

RIPPLE CARRY ADDER:

It is possible to create a logical circuit using multiple full adders to add N-bit numbers. Each full adder inputs a Cin, which is the Cout of the previous adder. This kind of adder is called a ripple-carry adder, since each carry bit "ripples" to the next full adder. Note that the first (and the only the first) full adder may be replaced by a half adder (under assumption that Cin=0).

The layout of a ripple-carry adder is simple, which allows fast design time; however, the ripple-carry adder is relatively slow, since each full adder wait
for the carry bit to be calculated from the previous full adder. The gate delay can easily be calculated by inspection of the full adder circuit. Each full adder requires three levels of logic. In a 32-bit ripple-carry adder, there are 32 full adders, so critical path (worst case) delay is 3x31 (from input to later adder) +2 (in later adder) =95 gate delays. The general equations for the worst-case delay for a n-bit carry-ripple adder is

$$\mathbf{T}\text{cra}(n)=(n-1)*\mathbf{T}c+\mathbf{T}s=(n-1)*3D+2D=(3n-1)D$$

Multiple full adder circuits can be cascaded in parallel to add an N-bit number. For an N-bit parallel adder, there must be N number of full adder circuits. A ripple carry adder is a logic circuit in which carry-out of each full adder is the carry in of the succeeding next most significant full adder. It is called a1 ripple carry adder because each carry bit gets rippled into the next stage.

MULTIPLIERS:

To reduce the number of partial products to be added, Modified Booth algorithm is one of the most popular algorithms. To achieve speed improvements Wallace Tree algorithm can be used to reduce the number of sequential adding stages. Further by combining both Modified Booth algorithm and Wallace Tree technique we can see advantage of both algorithms in one multiplier

MULTIPLICATION ALGORITHM

1. If the LSB of Multiplier is '1', then add the multiplicand into an accumulator.
2. Shift the multiplier one bit to the right and multiplicand one bit to the left.
3. Stop when all bits of the multiplier are zero.

From above it is clear that the multiplication has been changed to addition of numbers. If the Partial Products are added serially then a serial adder is used with least hardware. It is possible to add all the partial products with one combinational circuit using a parallel multiplier. However it is possible also, to use compression technique.

EXISTING SYSTEM

In applications like multimedia signal processing and data mining which can tolerate error, exact computing units are not always necessary. They can be replaced with their approximate counterparts. Research on approximate computing for error tolerant applications is on the rise. Adders and multipliers form the key components in these applications. Approximate full adders are proposed at

transistor level and they are utilized in digital signal processing applications. Their proposed full adders are used in accumulation of partial products in multiplier

To reduce hardware complexity of multipliers, truncation is widely employed in fixed-width multiplier designs. Then a constant or variable correction term is added to compensate for the quantization error introduced by the truncated part. Approximation techniques in multipliers focus on accumulation of partial products, which is crucial in terms of power consumption. Broken array multiplier is implemented, where the least significant bits of inputs are truncated, while forming partial products to reduce hardware complexity. The proposed multiplier in saves few adder circuits in partial product accumulation.

APPROXIMATE HALF ADDER

The half adder produces the outputs which is a sum of the two input bits along with its carry if generated. A half adder circuit contains two bits as an input which are usually denoted as A and B and two bits as output known as sum and carry which are usually denoted by S and C respectively. The output Sum(S) is a two bit XOR of input bit A and input bit B. The output Carry(C) is a AND of input bit A and input bit B. Necessarily the half adder produces an output which is the sum of the two 1-bit umbers and the carry(C) is obtained at the most significant bit(MSB) if the sum of each individual bit produces a carry. Half adder circuits are fairly very easy and simple to construct but it has its limitations, the drawback is cannot hold on or to include a carry when a multi-bit operation needs to performed. The Boolean equations for half adder are as given below bit A and input bit B. The output Carry(C) is a AND of input bit A and input bit B. Necessarily the half adder produces an output which is the sum of the two 1-bit numbers and the carry(C) is obtained at the most significant bit(MSB) if the sum of each individual bit produces a carry. Half adder circuits are fairly very easy and simple to construct but it has its limitations, the drawback is cannot hold on or to include a carry when a multi-bit operation needs to performed. The Boolean equations for half adder are as given below.

In adders and compressors, XOR gates tend to contribute to high area and delay. For approximating half-adder, XOR gate of *Sum* is replaced with OR gate as given in (2). This results in one error in the *Sum* computation as seen in the truth table of approximate half-adder in Table II. A tick mark denotes that approximate output matches with correct output and cross mark denotes mismatch

$$Sum = x1 + x2$$

$$Carry = x1 \cdot x2$$

As can be seen, the probability of misprediction is very low. As the number of *generate* signals increases, the error probability increases linearly. However, the value of error also rises.

Inputs		Exact Outputs		Approximate Outputs		Absolute Difference
$x1$	$x2$	$Carry$	Sum	$Carry$	Sum	
0	0	0	0	0 ✔	0 ✔	0
0	1	0	1	0 ✔	1 ✔	0
1	0	0	1	0 ✔	1 ✔	0
1	1	1	0	1 ✔	1 ✗	1

Table 1:Truth table of approximate half adder

APPROXIMATE FULL ADDER

The full adder circuits are employed to overcome the drawback associated with half adder circuit. The full adder circuit is a combinational arithmetic circuit which is used to carry out the summation of three bits (A, B and CIN). The full adder generated the corresponding output as

Sum(S) and Carry(COUT) which is as similar to the half adder circuit. The full adder may be constructed by cascading two half adders which is as shown in Fig.5. The summation of input bits A and B are give directly to the second half adder, where first summation output adds a carry CIN if a carry is generated from the previous summation of LSB operation and then finally its generates a output sum value. The carry-out is obtained from the results of an OR operation derived from the carry output from the both half adders. Both at the gate level and the transistor level there are plenty of adders architecture each one exhibits different parameters.

In the approximation of full-adder, one of the two XOR gates is replaced with OR gate in *Sum* calculation. This results in error in last two cases out of eight cases. *Carr y* is modified as i introducing one error. This provides more simplification, while maintaining the difference between original and approximate value as one. The truth table of approximate full-adder is given in Table III

$$W = (x1 + x2)$$
$$Sum = W \oplus x3$$
$$Carr\ y = W \cdot x3.$$

Inputs			Exact Outputs		Approximate Outputs		Absolute Difference
$x1$	$x2$	$x3$	$Carry$	Sum	$Carry$	Sum	
0	0	0	0	0	0 ✔	0 ✔	0
0	0	1	0	1	0 ✔	1 ✔	0
0	1	0	0	1	0 ✔	1 ✔	0
0	1	1	1	0	1 ✔	0 ✔	0
1	0	0	0	1	0 ✔	1 ✔	0
1	0	1	1	0	1 ✔	0 ✔	0
1	1	0	1	0	0 ✗	1 ✗	1
1	1	1	1	1	1 ✔	0 ✗	1

Table 2: Truth table of approximate full adder

APPROXIMATE 4-2 COMPRESSOR

Two approximate 4-2 compressors in [5] produce nonzero output even for the cases where all inputs are zero. This results in high E D and high degree of precision loss especially in cases of zeros in all bits or in most significant parts of the reduction tree. The proposed4-2 compressor overcomes this drawback.

Inputs				Approximate outputs		Absolute Difference
$x1$	$x2$	$x3$	$x4$	$Carry$	Sum	
0	0	0	0	0 ✔	0 ✔	0
0	0	0	1	0 ✔	1 ✔	0
0	0	1	0	0 ✔	1 ✔	0
0	0	1	1	1 ✔	0 ✔	0
0	1	0	0	0 ✔	1 ✔	0
0	1	0	1	0 ✗	1 ✗	1
0	1	1	0	0 ✗	1 ✗	1
0	1	1	1	1 ✔	1 ✔	0
1	0	0	0	0 ✔	1 ✔	0
1	0	0	1	0 ✗	1 ✗	1
1	0	1	0	0 ✗	1 ✗	1
1	0	1	1	1 ✔	1 ✔	0
1	1	0	0	1 ✔	0 ✔	0
1	1	0	1	1 ✔	1 ✔	0
1	1	1	0	1 ✔	1 ✔	0
1	1	1	1	1 ✗	1 ✗	1

Table 3: truth table of approximate4-2 compressor

4-2 compressor. In 4-2 compressor, three bits are required for the output only when all the four inputs are 1, which happens only once out of 16 cases.

To maintain minimal error difference as one, the output "100" (the value of 4) for four inputs being one has to be replaced with outputs "11" (the value of 3). For Sum computation, one out of three XOR gates is replaced with OR gate. Also, to make the Sum corresponding to the case where all inputs are ones as one, an additional circuitx1·x2·x3·x4 is added to the Sum expression. This results in error in five out of 16 cases. Carry is simplified as in (4). The corresponding truth table is given in Table III.

$$W1 = x1 \cdot x2$$
$$W2 = x3 \cdot x4$$
$$Sum = (x1 \oplus x2) + (x3 \oplus x4) + W1 \cdot W2$$
$$Carry = W1 + W2. \text{.....(5)}$$

arithmetic units (half-adder, full-adder, and 4-2 compressor) are proposed for approximation. The arithmetic units are not only reduced in complexity, but care is also taken that error value is maintained low. While systemic approximation helps in achieving better accuracy, reduced logic complexity of approximate arithmetic units consumes less power and area. The proposed multipliers outperforms the existing multiplier designs in terms of area, power, and error, and achieves better peak signal to noise ratio (PSNR) values in image processing application.

APPROXIMATE MULTIPLIERS

Approximate multipliers are widely being advocated for energy-efficiient computing In applications that to in exhibit an inherent tolerarance to inaccuracy. However, a key inclusion of as key design parameter, besides, the performance, area and power makes the identification of the most suitable approximate multipliers quite challenging. In this paper, we identify three major decision making factors for the selection of an approximate multipliers circuit: the type of approximate full adder(FA) used to construct the multiplier the architecture array or tree, of the multiplier in the main multiplier module. Based on the factors,, we exposed the design space for circuit level implementation of approximate multipliers. We used circuit level implementation of some of the most widely used approximate full adders, cells are then used to develop circuits for the implementation of higher bit all levels. The reports also prevents of validation of our results using an available application blending. In this DSP application, the multiplier is the main priority to reduce signal noise, fluctuation in all type of gadgets. In this method of truncated architecture is fully designed based on full adders, here carry operation is followed as per the same operation of sum, based upon this architecture it will take more critical path delay, propagation delay, and its perform slowest operation of arithmetic functions.

A goal of this truncated multiplier is to reduce the large area in the internal and external architecture using rounded based technique, which computed the truncation multiplier will have summing the two n-bit partial products, this operation of two n-bits, the MSB of most significant rows and columns with truncated, deleted and rounding to correction in variable method. A normal multiplier of n x n bit computes and get the weighted sum of output of 2n bits. A multiplier in signal processing the output represented the MSB part of n bits is useful, because it's signed oriented outputs, example of this design such as digital signal based application. A truncated multiplier is a hardware efficient multiplier, it will useful to increases the tradeoff accuracy and reduced the hardware cost, since this truncated multiplier will help to produce the output of n-bits form n x n bits of multiplication, it will take less significant, and some of the partial products are removed and also replace using the technique of deletion, reduction and truncation. In the partial products of this multiplier more number of columns are eliminated regarding the area and power consumption, in case the delay also decreases with compare to the normal operation of 2n outputs of

n x n multiplier, but some drawbacks will have on this truncated multiplication, because this multiplier is not concentrated on carry operation,

Multimedia and image processing applications, may tolerate errors in calculations but still generate meaningful and beneficial results. This work deals with a high speed approximate multiplier with TDM tree and carry prediction circuit. The modified multiplier utilizes an optimized TDM carry save tree which reduces the device utilization on FPGA as well as the combinational path delay and power consumption. The proposed design is analyzed using the simulation and implementation results on Xilinx Spartan 3E family.

IMPLEMENTATION OF PROPOSED TRUNCATION MULTIPLIER

PROPOSED ADDER DESIGN

A Modified Structure of Proposed SQRT Carry select adder design will have a HSCG Unit (Half Sum Carry Generation) and SCS Unit (Sum carry generation), it will help to process N number of addition, using XOR, AND, OR Gate's. These design will reduction of number of logic gate's will have compared to Ripple carry addition, and existing SQRT CSLA Design. Here HSG will provide the three outputs, such as operation of XOR, AND, OR Gates, the input CIN will select the Sum and Carry using Multiplexer. Here CIN=0 the multiplexer will select Sum of XOR outputs, and Carry of AND outputs, if CIN=1 the multiplexer will select Sum of Inverted XOR outputs, and Carry of OR Gate outputs. The architecture of Proposed SQRT based Carry select adder design will shown in Fig.8.

CIN	A	B	XOR	~XOR	OR	AND	SUM	CARRY
0	0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	1	0
0	1	0	1	0	1	0	1	0
0	1	1	0	1	1	1	0	1
1	0	0	0	1	0	0	1	0
1	0	1	1	0	1	0	0	1
1	1	0	1	0	1	0	0	1
1	1	1	0	1	1	1	1	1

Table 4:Truth Table of Proposed SQRT

Table 4, will have shown input and output values of SQRT CSLA using HSCG – SCG unit adder design. Here the Multiplier selection will have happened from input of CIN, If CIN = 0, the SUM Operation will get the input from XOR Gate, and Carry operation will get the input from AND Gate, Once the CIN will goes high, the Multiplexer will switch the inputs, the SUM Operation will get the input from inverted XOR Gate, and Carry operation will get the input from OR Gate, here sharing process will have happen based upon CIN Inputs. This Corresponding architecture gate count will take only Four logic gates and two multiplexers.

PROPOSED TRUNCATION MULTIPLIER

A Multiplier is more important in today technology such a application of digital signal processing, image processing and cryptography application method, since this all application is a most high

priority in today technology such as 3G, LTE, Telecommunication, audio and video processing and so on. In this DSP application, the multiplier is the main priority to reduce signal noise, fluctuation in all type of gadgets. In this method of truncated architecture is fully designed based on full adders, here carry operation is followed as per the same operation of sum, based upon this architecture it will take more critical path delay, propagation delay, and its perform slowest operation of arithmetic functions. A goal of this truncated multiplier is to reduce the large area in the internal and external architecture using rounded based technique, which computed the truncation multiplier will have summing the two n-bit partial products, this operation of two n-bits, the MSB of most significant rows and columns with truncated, deleted and rounding to correction in variable method.

A normal multiplier of n x n bit computes and get the weighted sum of output of 2n bits. A multiplier in signal processing the output represented the MSB part of n bits is useful, because it's signed oriented outputs, example of this design such as digital signal based application. A truncated multiplier is a hardware efficient multiplier, it will useful to increases the tradeoff accuracy and reduced the hardware cost, since this truncated multiplier will help to produce the output of n-bits form n x n bits of multiplication.

It will take less significant, and some of the partial products are removed and also replace using the technique of deletion, reduction and truncation. In the partial products of this multiplier more number of columns are eliminated regarding the area and power consumption, in case the delay also decreases with compare to the normal operation of 2n outputs of n x n multiplier, but some drawbacks will have on this truncated multiplication, because this multiplier is not concentrated on carry operation, such as carry addition and carry skip operation, here used number of full adders for addition, but not implemented the simple and efficient gate level implementation with carry operation to significantly reduced the area, power and delay.

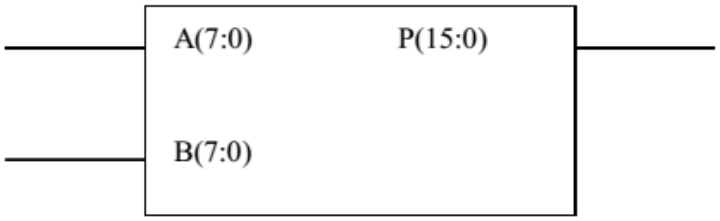


Fig 1.3: Block Diagram of Standard 8x8 Multiplier

An FIR filter it not required a feedback based inputs, which means, this filters is not computed any rounding errors in summing and multiplication. An FIR filter is inherently stable to produce output values and it can be no maximum value impulse response Nth order times, it can easily design and also easily configure sequence of linear phase coefficient, it will also applicable to detect the phase sensitive applications such as crossover filter design, mastering, seismology and data communications.

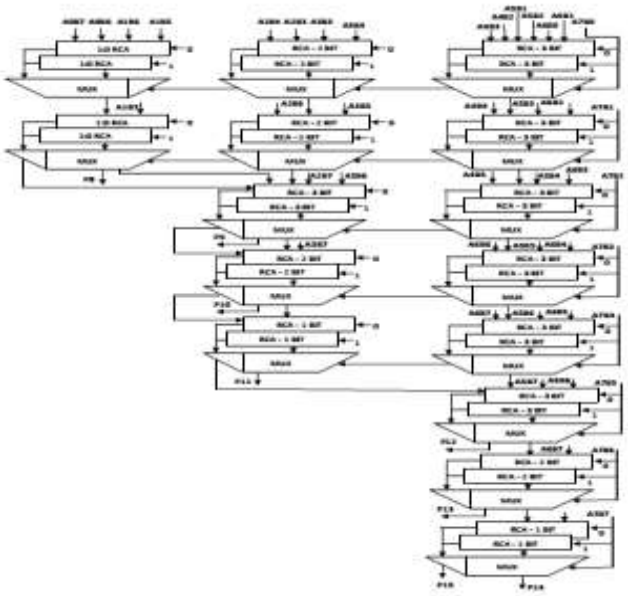


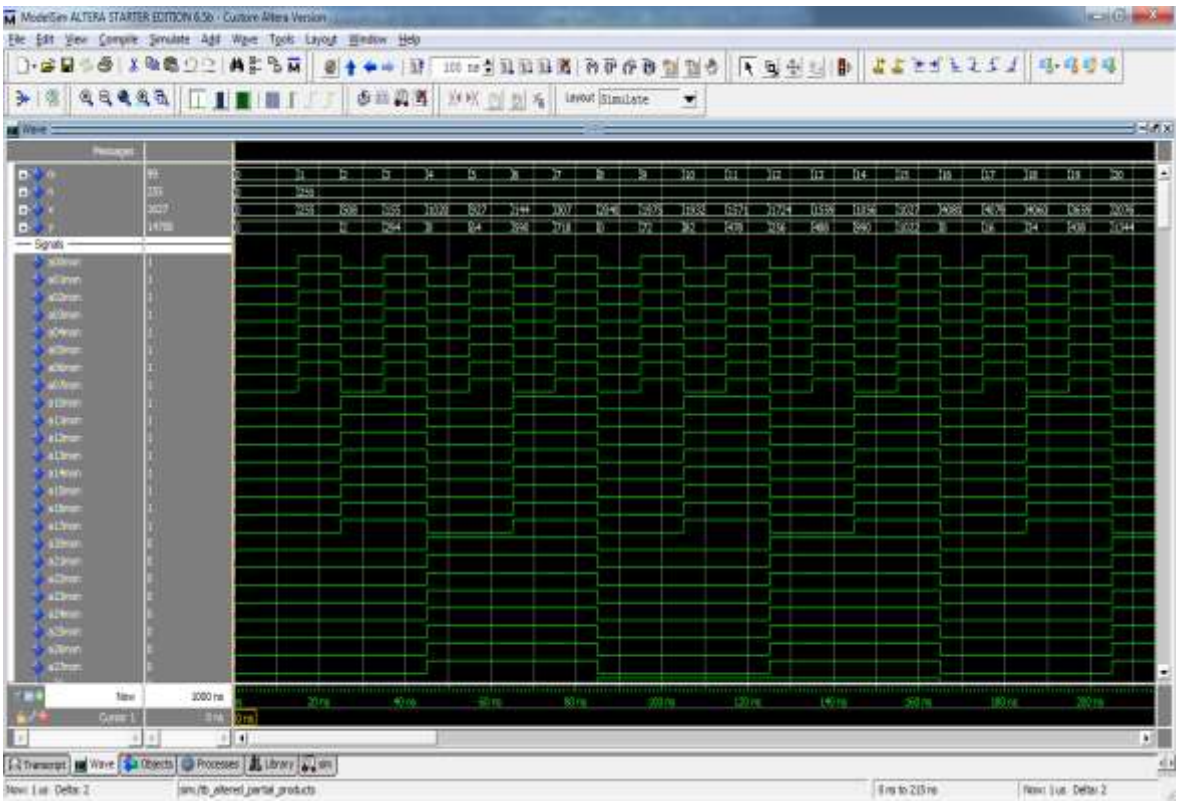
Fig 1.4:Truncation Multiplier architecture using HSCG-SCG RCA

ADVANTAGES

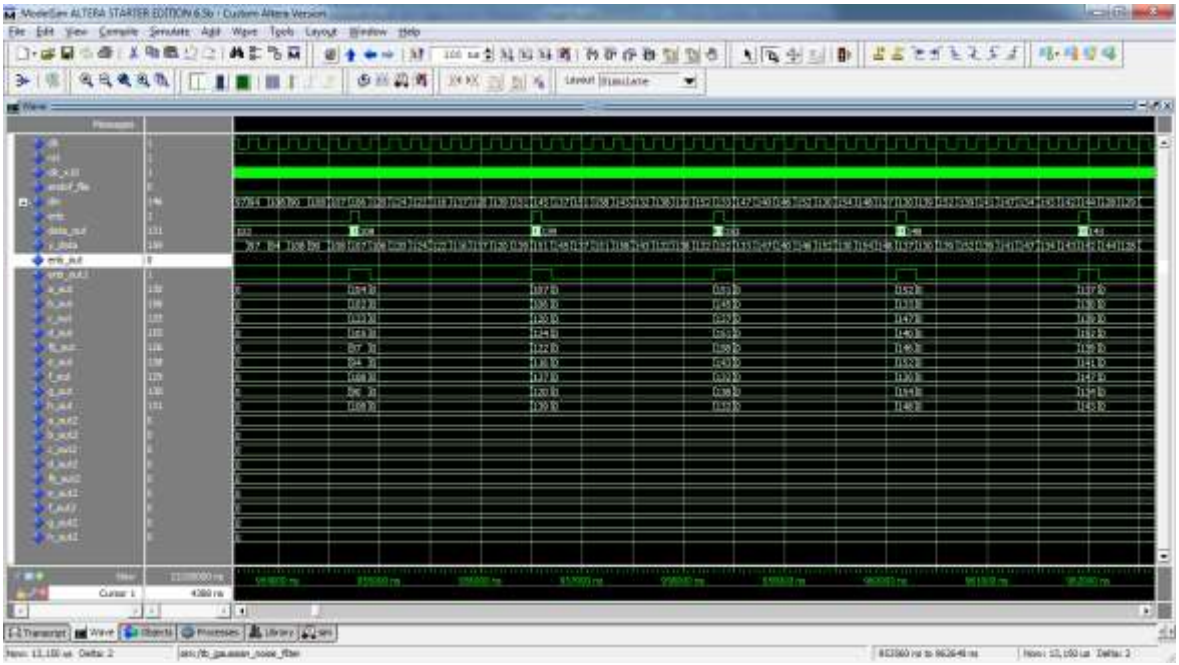
- Provide the output with luminance and chrominance
- Less Area and Power
- High Performance

RESULTS AND DISCUSSION:

1.1 EXISTING OUTPUT FOR MULTIPLIER

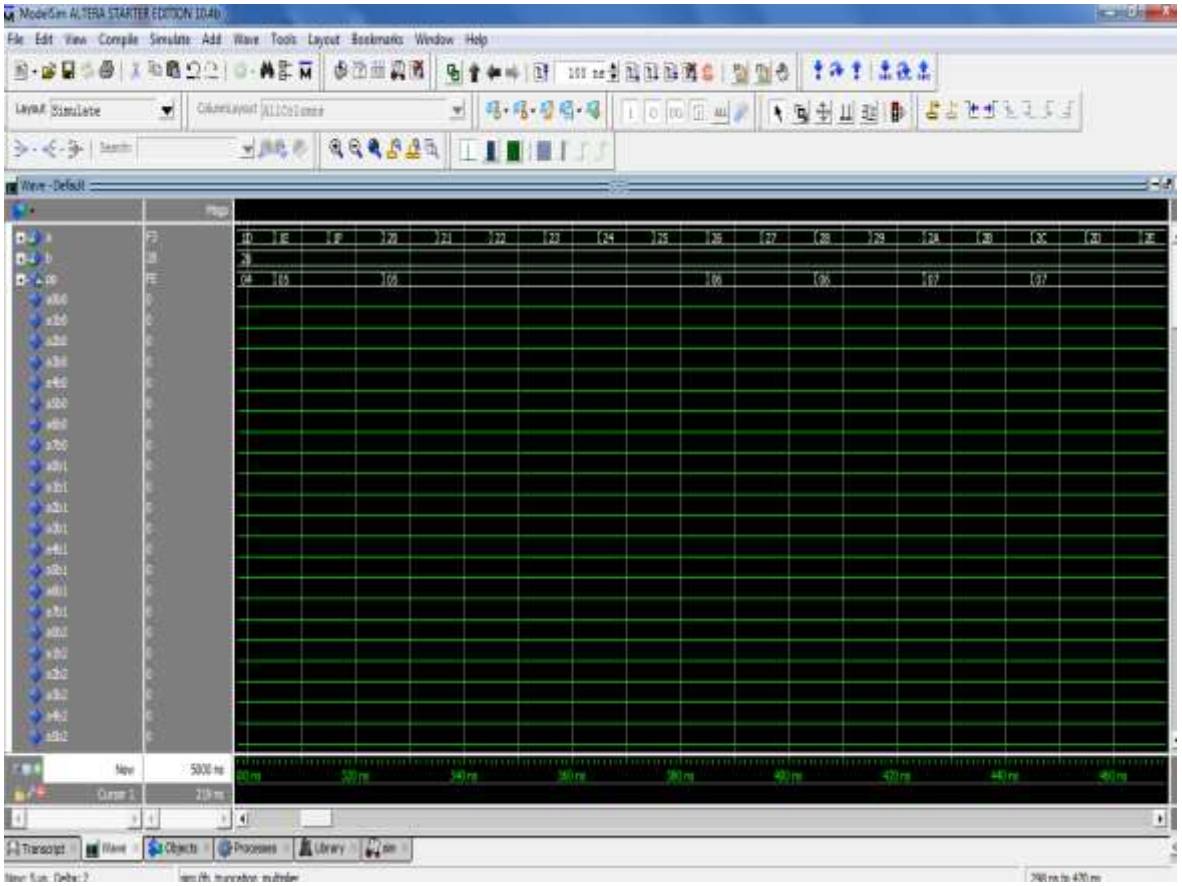


7.1.3 EXISTING OUTPUT FOR GAUSSIAN FILTER

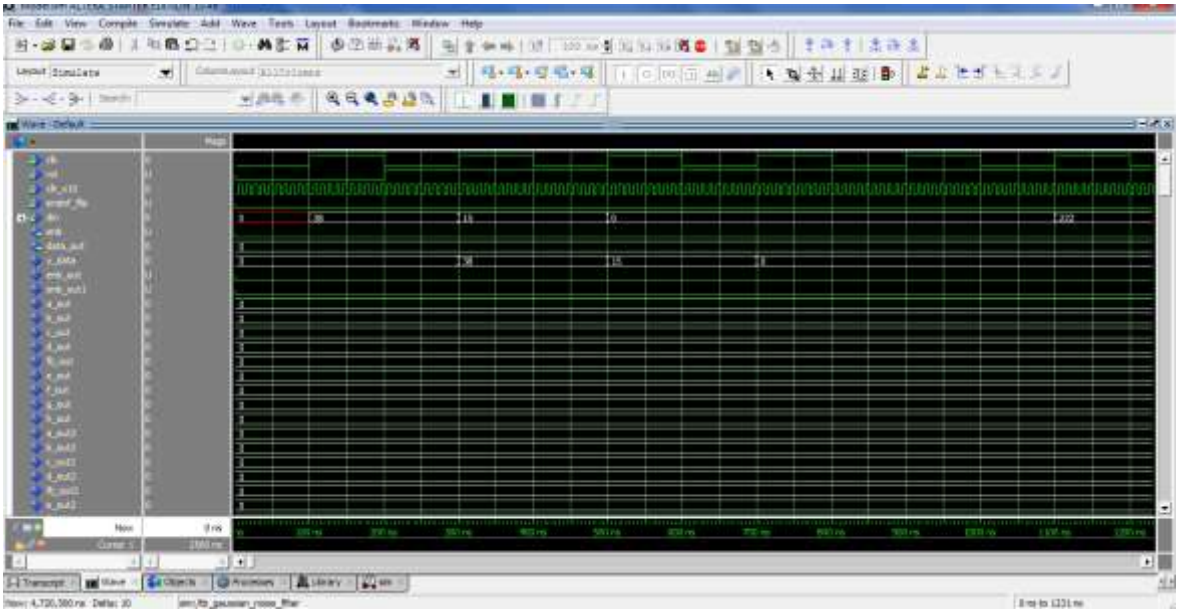


PROPOSED OUTPUT

PROPOSED OUTPUT FOR MULTIPLIER



PROPOSED OUTPUT FOR GUASSIAN FILTER



COMPARISON OF MULTIPLIERS

An approximate multiplier is design for n=8 bits. The multiplier is implemented in VHDL and synthesized in Xilinx S6LX9 FPGA, from the Xilinx synthesizes report, we get number of slice registers, number of slice LUT, number of occupied slice, number of IOB, and total power. The efficacy of approximate multiplier is given better compare to existing normal multiplier. In this approximate multiplier, the partial products generation reduction based on half-adder, full-adder, and 4-2 compressor to generate the final partial products. The comparison of approximate multiplier with normal truncation multiplier shown in table.

Multiplier	Number of Slice LUT	Number of Occupied Slice	Number of IOB	Total Power (mW)
Approximate Multiplier	102	53	46	14
Truncation Multiplier	143	56	24	14

Table 5: Comparison Table 1

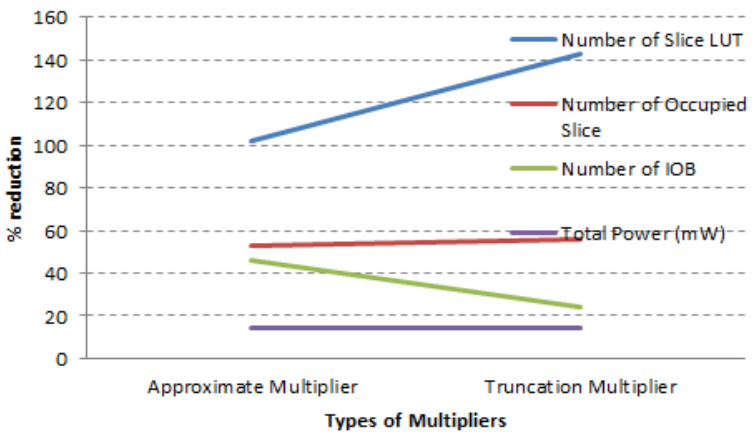


Fig 1.5: Percentage of reduction no of logic gates

The table comparison clearly shoes that approximate multiplier has better in number of slice LUT, occupied slices, and number of IOB, but same total power for both multipliers. The comparison graph shown in figure. A modified design truncation multiplier has been proposed, in which the adder deign has been replaced by new proposed SQRT CSLA HSCG-SCG adder, which is already proved compare to all CSLA based adders. This adder has used in implementation of truncation multiplier, this multiplier provide the better performance compare to approximate multiplier in terms of number of slice LUT, occupied slices, number of IOB, and power is same for both multipliers. The comparison of two multipliers shown in table and Figure.

Multiplier	Number of Slice LUT	Number of Occupied Slice	Number of IOB	Total Power (mW)
Approximate Multiplier	102	53	46	14
HSCG-SCG Based Truncation Multiplier	66	27	24	14

Table: 1.6: comparison of two multipliers

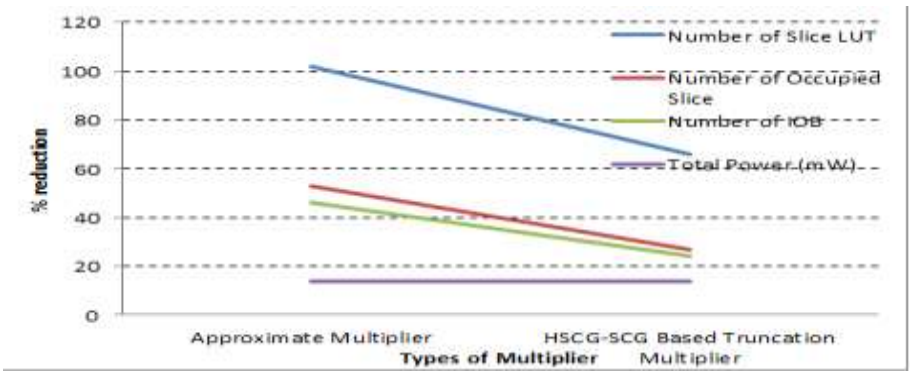


Fig 7.3(b): Percentage of reduction no of logic gates, multiplier, LUT, Occupied Slices, IOB

The percentage reduction in comparison of approximate and HSCG based truncation multiplier in terms of number of slice LUT, Occupied slice, and IOB plotted in Figure , shows that the HSCG-SCG based truncation multiplier reduce 35.29%, 49.05%, and, 47.82% respectively, the total power maintain both multipliers are same.

Finally these two multipliers like approximate and truncation based on HSCG-SCG multiplier implemented in Gaussian filter, and evaluated the performance in terms of number of slice register, number of slice LUT, number of occupied slice, number IOB, total power, dynamic power, and delay. The comparison of two Gaussian filters shown in table and Figure.

The comparison of two Gaussian filters shows in terms of slice register, slice LUT, occupied slice, IOB, total power, dynamic power, and delay, the HSCG-SCG truncation multiplier based Gaussian filer reduce the little bit of slice LUT, occupied slice, and delay , remain parameters maintain same.

Gaussian Filter	Number of Slice Registers	Number of Slice LUT	Number of Occupied Slice	Number of IOB	Total Power (mW)	Dynamic Power (nW)	Delay(ns)
Approximate Multiplier with Gaussian Filter	122	457	164	21	25	11	3.581
HSCG Adder & Truncation Multiplier with Gaussian Filter	122	444	161	21	25	11	3.523

Table 1.7: Comparison Table 3

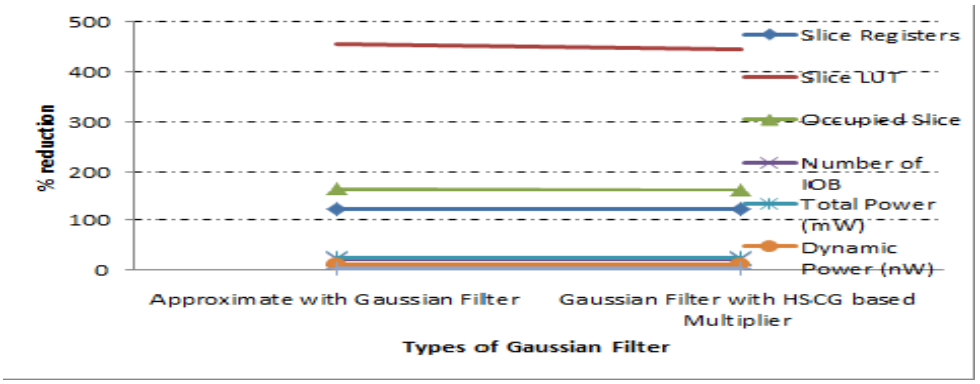


Fig 1.8 Percentage of reduction no of logic gates, multiplier, LUT, Occupied Slices, IOB

CONCLUSION:

In this brief, to propose efficient Truncation multiplier based on SQRT CSLA HSCG-SCG adder. The HSCG-SCG (half sum carry generation –sum carry generation) is used to reduce the number of logic gates to design the full adder circuit. This adder implemented in truncation multiplier which is reduce the number of logic gates, number of IOB, number of slice, and slice registers compare to existing approximate multiplier, and also implemented the truncation multiplier in Gaussian filter to reduce the noise of image, which is also perform the little bit higher than approximate multiplier based Gaussian filter. In this the parameters of multiplier comparison with truncation multiplier.

REFERENCES

[1] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput.- Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[2] E. J. King and E. E. Swartzlander, Jr., "Data-dependent truncation scheme for parallel multipliers," in Proc. 31st Asilomar Conf. Signals, Circuits Syst., Nov. 1998, pp. 1178–1182.

[3] K.-J. Cho, K.-C. Lee, J.-G. Chung, and K. K. Parhi, "Design of low-error fixed-width modified booth multiplier," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 12, no. 5, pp. 522–531, May 2004.

[4] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[5] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," IEEE Trans. Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.

[6] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," IEEE Trans.

Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.
[7] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 10, pp. 3105–3117, Oct. 2016.
[8] P. Kulkarni, P. Gupta, and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture," J. Low Power Electron., vol. 7, no. 4, pp. 490–501, 2011.
[9] C.-H. Lin and C. Lin, "High accuracy approximate multiplier with error correction," in Proc. IEEE 31st Int. Conf. Comput. Design, Sep. 2013, pp. 33–38.
[10] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in Proc. Conf. Exhibit. (DATE), 2014, pp. 1–4.
[11] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD), Oct. 2011, pp. 667–673.
[12] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," IEEE Trans. Comput., vol. 63, no. 9, pp. 1760–1771, Sep. 2013.
[13] S. Suman et al., "Image enhancement using geometric mean filter and gamma correction for WCE images," in Proc. 21st Int. Conf., Neural Inf. Process. (ICONIP), 2014, pp. 276–283.
[14] U Penchalaiah, VG Siva Kumar, "Survey: Performance Analysis of FIR Filter Design Using Modified Truncation Multiplier with SQRT based Carry Select Adder," International Journal of Engineering & Technology, spc, 7(2.32) (2018) 23-34.
[15] U. Penchalaiah and S. K. VG, "Design of High-Speed and Energy-Efficient Parallel Prefix Kogge Stone Adder," 2018 IEEE International Conference on System, Computation, Automation and Networking (ICSCA), Pondicherry, 2018, pp. 1-7.