Turkish Journal of Computer and Mathematics Education Vol.12 No. 7 (2021), 1735-1750

Research Article

A Design of Low Power and Area efficient FIR Filter using Modified Carry save Accumulator Method

L Mohana Kannan[1] , D Deepa[2]

Assitant professor ,Department of Electronics and Communication Engineering,
RVS college of engineering and technolgy ,Coimbatore, India.mohancalls2000@gmail.com
professor ,Department of Electronics and Communication Engineering,Bannari Amman Institute of Technology Sathyamangalam - 638 401, Erode District, Tamil Nadu, India, ddeepa@bitsathy.ac.in

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 16 April 2021

Abstract : In VLSI circuit designs, the reduction of power, area, and delay parameters are increasing as the range of sophistication of applications. Nowadays, there are many real-time applications needs low power with high throughput than never before getting the range. Here the FIR filter is used to design an efficient digital signal processing system. In FIR filter, the adders and multipliers are the important components to reduce the delay and area. Therefore, this proposed method faced with more constraints: high speed, high throughput, and at the same time, consumes as minimal power as possible. The Finite Impulse Response Filter is widely used in Digital Signal Processing Applications, such as speech processing, loudspeaker equalization, echo cancellation, noise cancellation, arithmetic computations and, image-processing applications. This work proposes a design of low power and area efficient Finite Impulse Response Filter using Modified Carry save adder. In FIR Filter design, the power determination is by the configuration, which selected by the ripples present in the execution of filter. The filtering process carried out by eliminating the ripples presented in the pass band and stop band of designed architecture. The FIR architecture is reframed by adders and multipliers used in the design. The proposed digital FIR filter design reduces the uses of adders and multiplier blocks for its efficiency in area and delay. Thus, the multipliers interchanged as adders for its area efficiency. The input data bits multiplied by a filter coefficient and then the ripple carry adder selects the particular adder cell to perform addition, which depends on carry input (Ci). This input bit performs at the last of carry save accumulation process. Similarly, next adder's cell performs the same operations. The filtered output coefficient is getting from Carry save accumulation process. Thus, the proposed method can minimize the area, delay and power consumption of FIR filter design. Here with the use of Carry save adder with ripple carry adder gives the best optimization result. The Number of bonded inputs and outputs reduces the area of 24%. The experimental result achieves 24% of area reduction, 2% of delay and 3% of power consumption compared to the existing method. The Xilinx ISE 14.5 and Modelsim simulator is used to synthesis and simulation of the proposed architecture. The power consumption result analyzed by the Xilinx Power Estimator tool. This experimental result gives better robustness than most recent related literature.

Keywords: Finite Impulse Response Filter, Carry save accumulation process, Ripple carry adder, Area, Delay and, Power consumption

I. INTRODUCTION

Nowadays the VLSI design process used for both analog and digital circuit design. The VLSI digital chips increased device pressure and speed and decreased power dissipation. The electronic market is given to the importance of low power electronic product. Since the choice of logic chip is a very important constraint at the circuit level. It differs in energy, area, delay and, throughput. Therefore, the low power and area efficient VLSI designs selected for its robustness. The digital filters are widely used fundamental devices in digital signal processing applications. The Digital filters, classified into two types, which is Infinite Impulse Response filter and Finite Impulse Response filter. Due to the linear phase response and inherent stability, the FIR filters used most than the IIR filter. The Finite Impulse Response Filters are used in speech processing, noise cancellation, computer graphics, image processing and, telecommunications. In FIR filter, the impulse response is finite value, because this will settles to zero value in finite time duration. This filter requires no feedback because the rounding errors are not merged with summed iteration. The same error occurs when the signals are merged. This makes the result simpler. The importance of the low power utilization is required for the vast majority in recent methods on FIR filter channel. The power utilization of adders and multiplier decides the reduction in total area/power, and it improves the speed for whole filter design. FIR Filter channels are executed continuous/discrete manner. The major challenges are a simultaneous approximation in both magnitude and phase response. However, the various features in the expansion of FIR filter design have some challenges are faced by design engineers. Therefore, this research paper focused on developing low power and area efficient architecture design and further it increases the speed by reducing delay. The number of taps in the FIR Filter is directly proportional to the stop-band attenuation of the FIR Filter. Thus, as the number of taps increases, the filter attenuates unwanted signals better. However, increases in the number of taps also increase the hardware complexity. Therefore, maintaining certain levels of taps reduces power dissipation on the filter. In FIR Filter architecture, the impulse response is set as the filter coefficients. The impulse response is denoted as h(m) which determines the filter operations, that follows the Kronecker delta function. The finite impulse response arises

because the filter output is computed as a weighted, finite term sum, of past, present, perhaps future values of filter input i.e.,

$$y(t) = \sum_{t=0}^{N-1} h(m) \cdot x(m-t) \quad \text{…………………..(1)}$$

where,
x(t) is input signal
y(t) is output signal
h(m) is impulse response
The weight vector h is order of,

$$H=[h(0),h(1),\ldots h(N-1)]^T \quad \text{……………..(1a)}$$

Input matrix X_t denoted as,

$$X_t=[X_t^0 \quad X_t^1 \;\ldots\; X_t^4 \;\ldots\; X_t^{N-1}] \quad \text{…………(1b)}$$

Where X_t^m is the (m+1)th column of X_t is defined as,

$$X_t^m = [x(nd-m)\; x(nd-m-1)\; x(nd-m-d+1)]^T \quad \text{………(2)}$$

The FIR filter has two stages namely, approximation and realization. In approximation stage based on the specification and transfer function obtained through the following steps,

i. Desired frequency response is select in the frequency domain
ii. Length of the FIR filter is chosen
iii. Measure the quality of approximation value
iv. The Algorithm is used to find the best transfer function of the filter

The realization stage deals with choosing the structure to implement transfer function that is in the form of circuitry function. The unit impulse response of FIR filter is in the sequence of having non-zero values one only when its argument is equal to zero, i.e., t=0

$$\partial(t) = \begin{cases} 1, & t = 0 \\ 0, & t \neq 0 \end{cases}$$

The unit im[illegible]e shifted right or left by integer n_0 by writing,

$$\partial(t-t_0) = \begin{cases} 1, & t \neq t_0 \\ 0, & t = t_0 \end{cases}$$

The digital FI[illegible]ficients are multiply by input data bits and then the ripple carry adder takes the particular adder cel[illegible] addition, which depends on carry input (Ci). These input bits are sends to carry save accumulation at the last stage of adder unit. Similarly, each data bits are performed by same way with delay coefficient. The Carry Save Adder eliminates carry chaining process by saving the carry and provides the carry to the next higher bit adder. It consists of multiple one-bit full adders without carry chaining. Each CSA block having 2bit coefficients with its carry input. Sum of the partial bit is calculated, and the carry values given to the next higher bit. By the use of Carry save adder with Ripple Carry adder, the delay and, area minimization is possible.

II. LITERATURE SURVEY

Basant Kumar Mohanty and Pramod Kumar Meher (2016) proposed the structure for reconfigurable applications. This includes Multiple Constant Multiplication methods for transpose form of FIR filter structure [1]. The transpose form of filter configuration does not support for the block processing, unlike direct form filter configurations. Each channel takes the 4 bits input from the register unit and sends to the pipelined adder block then performs the operation. The Distributed arithmetic structure used to design an FIR filter. Constant shift method and programmable shift method has been used for the same design. Area Delay product, Power Delay product, and Energy per sample is estimated.

Jongsun Park (2004) proposes the programmable digital FIR filter with computational sharing multiplier. Here, CMOS technology used for floor planning and FIR die implementation [2]. The sharing multiplier reduces the number of multipliers unit but increases the adder block by sharing the input coefficients. This method analyzes the power, clock timing, die area and, voltage results. CMOS logic also finds the result of power delay product and layout area.

AP Vinod and Edumund M-K Lai (2006) proposed the design and Implementation of FIR filter for software-defined radio receivers [3]. Here, IF processing, Pseudo floating-point representation and software-defined radio proposed. The coefficient is taken by PFP coding algorithm and the filter designed with its impulse response. The input coefficients shifted at each stage of multiplication, later it performs addition. After taking the result of the pass band, the impulse response of filters produces the stop band signals by the pseudo coefficient algorithm.

N Sameeksha Rai (2018) describes the design of FIR filter for DSP applications. Each input channel takes 4-bit of inputs, it sent to the partial product generator [5]. Next, the block of Vedic multiplier (uradhva triyakbhyam sutra) performing the filter coefficients. The multiplied outputs are sends to the adder block with carry input. Each 2-bit input cell takes 4-bit adder unit. The number of adders and multipliers were high since the availability of adders is increased. Hence, it takes more area.

. S Madhavi (2018) proposed the design of FIR filter using Dadda Multiplier with parallel prefix adder [6]. The bit arrangement of the input coefficient depends on Dadda algorithm. Last two stage of row matrix bits taken for the accumulation process. This method increases the delay while simulating each input coefficients. Parallel prefix adder selects the carry values. Each cells of an adder having two bits of input with selected carry input. The parallel structure consumed much area and power. To compensate this, new parallel adders designed with symmetry coefficient of the filter. This is for preprocessing of input signals through the filter unit. Here the number of adders not minimized for best constraints.

Nisha Chaudhary, Shweta Meena (2017) proposed the design of FIR filter using Booth multiplier with structural adder [7]. The length of the bits reduced, by adding the number of adders in the circuit unit. Each coefficient bits having separate MCM block with adder tree, it performs the shift-add method. Two 4bit input coefficients take the performance of MCM block shifter-I and next two 4bit coefficients performed with carry bit. Booth multiplier reduces the use of the number of adders but using the structural adder with shift-add method causes design complexity. It consumes less area but it increases delay.

Sumalatha Madugula (2017) proposes the method of using shift - add method for FIR filter design without the multiplier [8]. Filter coefficients shifted at each stage of addition and add the result. It causes slow performance and occupies many areas. Without the multiplier, input coefficients not aligned properly. It was having more ripples in the pass band and stop band. Adder block takes much time to produce output.

S Kiruthika and A Vimala starbino (2017) proposes the design of FIR filter using Full Adder cell [9]. The multiplied coefficients are taken as an input of full adder block and it having half adder circuitry in it. Each bit performed by full adder circuit and filtered coefficient bit stored in flip-flop block. The number of availability of adders increased while getting an output. It takes more area and power consumption while simulating through tanner tool.

L Durga Prasad (2017) describes the method of using EDBNS multiplier for FIR filter design [10]. The functions of Discrete Fourier Transforms used for transpose form of FIR filter structure. By using Extended Double-Base Number System multiplier, it saves the filter coefficients at each stage of simulation; hence, it extends the area for adder block. Each coefficient shifted and multiplied then the shifted coefficients send to accumulation unit, for performing addition. Overall the performance is slow, while using EDBNS multiplier.

R Geetha and R Bavya (2017) proposed method is used Bi-Recoder multiplier with Vedic multiplier and Wallace tree multiplier [11]. The Bi-Recoder multiplier generates partial products, which contains 8 bits of four MUX block. Therefore, it takes the number of adders much. The impulse response of the FIR filter is the product value of Bi-Recoder. The performance speed is high, but it occupies much area for their multiplication process. In Bi-recoder techniques, booth multiplier should perform the rules of recoder binary value. The coefficient bit is same; it performs the operation of shifting or if it changes the coefficient bit value, based on low/high bits addition and subtraction performed. It takes number of stages for the process.

R Gopalana and A Parameswari (2016) Designing the MCM technique for FIR filter architecture [12]. Input coefficients stored in the register unit and it split to MCM blocks, it performs multiplications based on a higher order of coefficient bits. The output of multiplier is taken as an input of accumulator unit, through adder network block. Then the accumulator performed. Hence, it takes the minimal delay, but power consumption is high. The power dissipation increased when the shifter gets overloaded at the stage of previous performance. Every time it shifts the value and adds the coefficients makes irregular outputs.

R.K Mohana Lakshmi, P Saravanakumar (2015) proposed the method of using Multiple Constant Multiplier with Wallace tree adder [13]. Here, the Pipelined shift-add method reduces the multiplier block; but the use of Wallace tree adder, the number of adder cells and bit alignment channels increased. After getting the simulation result, the number of LUTs and IOBs are increased. Because the pipelining process takes, adders block much for separating the input coefficients. Wallace reduction process takes the three groups of data, if it is two groups; it sends a single bit to the next block. Therefore, the number of stages increased, since the delay is increased.

Vadapalli Siddhartha (2015) proposed the iterative method of Remez exchange algorithm using Booth multiplier and Wallace tree multiplier [14]. The filter coefficients can extract by Remez exchange algorithm. This process minimizes the frequency response of the filter. The Area is reduced. Ripples in the pass band and stop bands equalized by this method. The output coefficients are not an exact value, because of rounding of input coefficients performed by Remez algorithm. However, it has a very high design complexity.

T Radha and M Velmurugan (2015) describe the design of FIR filter uses Hybrid adder and Vedic multiplier [15]. The odd input coefficients performed by carry select adder and booth multiplier functions. Then Vedic multiplier multiplies the coefficient bit. The accumulator block performs at last. The resultant coefficient

bits stored in the flip-flop register. It achieves better throughput, but it increases power consumption. The limitations of this method only focused on reducing the number of non-zero coefficients.

E Chitra and T Vigneswaran (2015) describe the functions of the Distributed Arithmetic algorithm [20]. It provides the multiplication free method for calculating inner products of fixed-point data, based on LUTs of pre-calculated partial products. The output coefficients are not proper since it takes much time to analyze the data bits. Therefore, the time delay affected real-time performance.

P C Franklin (2014) describes the method of using Wallace tree multiplier with binary to excess -1 converter [21]. The multiplied outputs added using carry select accumulator, then the coefficient bits converted for system performance. It increases the speed, but it consumes much memory. The Carry Select Adder block having D latch for registry of coefficients and the impulse response of filters analyzed by Wallace tree multiplier block, at the last the coefficient bits converted by Binary to Excess-1 converter. It makes the output complicate.

Dhivya V M and Sridevi (2014) proposed the floating point DADDA multiplier with Radix 2 method. It comprises recoder techniques and shifting of adder unit [22]. Each 4-bit input takes separate impulse coefficients and it performs addition by radix 2 operation then the coefficient bits are multiplied by floating point DADDA multiplier. The interchangeable bits are aligned by Radix 2 methods. It consumes less power, but the delay increased by using recoder technique. Due to the serial multiplication process, the performance of DADDA multiplier was slow.

Mahatha Nayak Bhukya (2012) describes the performance of FIR filter design using Distributed Arithmetic algorithm [23]. The advantage of this method is to determine an appropriate filter order automatically. This will improve the performance of an FIR filter design. The input coefficient has bit alignment process based on performance. Separate values are given to the multiplier block and it performs the addition with each carry input. It achieves better throughput but it involves complexity in designing when compared to our proposed method.

C Uthayakumar, B Justus Rabi (2014) proposes the design of FIR filter using the windowing method [24]. Rectangular windowing method used with finite array values. The input coefficients are a matrix form of an array, and then it selects the input coefficients by row matrix. They have analyzes the performance for low pass, high pass and, band pass filter is designed using MAT lab simulator. The ripples present in pass band and stop bands reduced by adder circuit in FIR filter.

S Satheeskumaran and, K Sasikala (2017) proposed the paper, modified DA algorithm with SoC Encounter for FIR filter design [28]. This module created with control signal variables. The input data shifted to cause delay using D flip-flop. When the reset control signal is active, the memory, multiplier and adder blocks are initiating the value. When the configure signal is active, the coefficients are configured and stored in the memory unit. When the configure signal is passive, the delayed inputs are multiplied with the coefficients and resulting products are added to generate filtered output. For optimizing the filter function, they adopted certain algorithms and techniques.

III. EXISTING METHOD

Multiple Constant Multiplication methods used in previous systems it takes more areas for the weight of the multiplier, and it increases the availability of adder blocks [1]. FIR filter used in DSP applications with the use of more number of multipliers and adders in each block cause availability of LUT and FF blocks, since it takes more area. The modified transpose FIR filter structure contains more delay block. Then the number of adder cells is increased. Using of Vedic multiplier with carry increment adder, the area utility is more [5]. The uses of Wallace tree multiplier with Multiple Constant Multiplication processes, each bit are performing shift, addition, and multiplication with the delay of a single bit on each block [13]. It causes delays in performance. The conventional method uses the DA algorithm, it achieves better throughput, but it involves the complexity of the design [23]. Conventional method uses DADDA multiplier, the input coefficients selected by Matrix form of an array. Directly, it multiplied the coefficients on an array matrix [22]. It reduces the adder unit, but the system takes much time to produce the output. The Recoder methods used with booth multiplier [18], this multiplier depends on the process of repeating addition and shifting the coefficients. According to the multiplication procedure, if the coefficient bits are same, it performs shifting operations. On the other hand, the coefficient bits varied, it performs addition/ subtraction based on coefficient values. When implemented in the digital signal processor for higher order coefficients hardware requirements, arithmetic operations, area usage, and power consumption increase drastically. To minimize these factors an algorithm by which FIR filter frequency and, phase response can represent with a minimum number of non-zero coefficients.

A. Booth multiplier with Recoding method

The multiplier circuit is based on repeated addition and shifting procedure. Booth algorithm provides a procedure for multiplying binary integers in signed-2's complement representation [18].

Table1: Recoded rules

Q_n	Q_{n+1}	Recoded bits	Operation performed
0	0	0	Shift
0	1	+1	Add M
1	0	-1	Subtraction M
1	1	0	Shift

According to the multiplication procedure, strings of 0's in the multiplier require no addition but just shifting and a string of 1's in the multiplier from bit weight 2m to weight 2d can be treated as 2m+1–2d. Booth algorithm involves recoding the multiplier first. In the recoded format, each bit in the multiplier can take any of the three values: 0, 1 and -1.

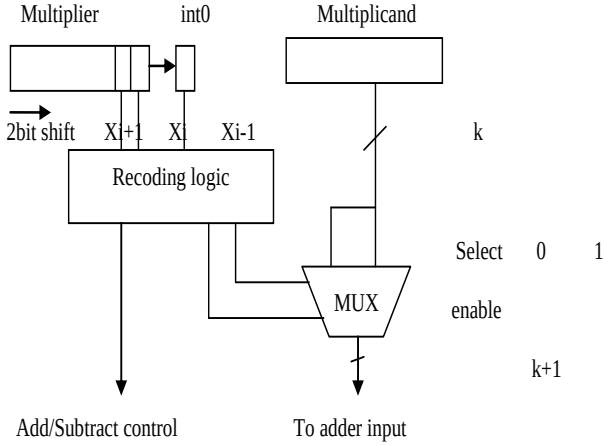


Fig1. Block diagram for Booth multiplier with recoding method

This block diagram shows the performance of Booth multiplier with recoding technique. Based on multiplier and multiplicand bits the recoding operations performed by recoding rule. Adder's carry out and adder's sum bit given to the recoding logic unit and it performs the operation. This output is taken to the adder control block. The unused part of the multiplier, if it having sampled bit, it directly sends to the MUX control block. By using booth multiplier, number of adders and the number of logic blocks are increased, when the process of recoding technique.

B. Floating point multiplier using DADDA algorithm

Multipliers are the major components of high performance system used extensively in digital signal processing applications. The floating-point arithmetic involves manipulating components and shifting fractions. The multiplication process takes the input coefficients and it performs partial product. The sum of partial product decomposes the binary multiplier unit. The floating-point number represented in the form:

Significant digits * base ^ (exponent)

This representation depends on radix point and this represented real numbers are placed anywhere on the block. This floating-point representation values supports, much range of input values.

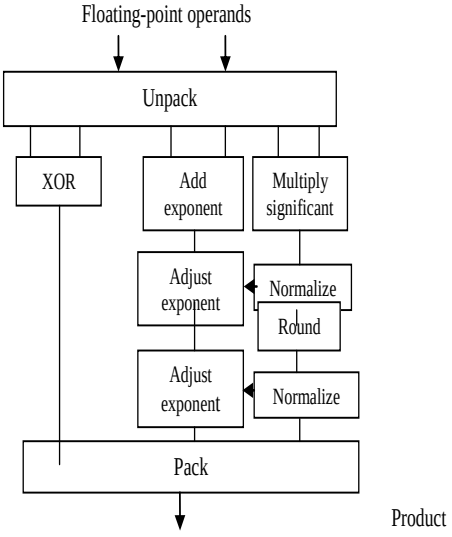
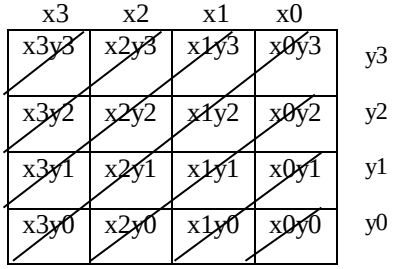


Fig2. Block diagram for Floating-point multiplier

Initially, the input values are unpacked into sign bit exponent and a significant bit then it applied to the sub-modules. The sign of multiplication results obtained from XOR block. Normalize the product using normalize block. Find, whether the product is overflow or underflow and rounding the coefficient sign bit separately then it is normalized. Finally, set the sign bit coefficient based on the user-constrained filter.

DADDA multiplier is extracted form of the parallel multiplier. The algorithm of DADDA multiplier is based on below matrix form. The partial product matrix arranged in the first stage. It reduces the row matrix of adders [22].



x3y3 x3y2 x3y1 x3y0 x2y0 x1y0 x0y0
x2y3 x2y2 x2y1 x1y1 x0y1
x1y3 x1y2 x0y2
x0y3

Fig3. 4x4 matrix DADDA Algorithm

In DADDA multiplier, multiply each bit of one of the arguments, by each bit of the other arguments, yielding the result. Depending on the position of the multiplied bit, the wires carry different weights. Then, it reduces the number of partial product of full adder. Finally, it groups the wire in two numbers and adds them with conventional adder's coefficients. In floating point multiplier with DADDA algorithm, the two input bits performs significant multiplication and the result after adding two exponents is not true exponent and obtained by subtracting bias value.

$$E_A = E_{A\text{-true}} + \text{bias}$$
$$E_B = E_{B\text{-true}} + \text{bias}$$
$$E_A + E_B = E_{A\text{-true}} + E_{B\text{-true}} - 2\text{xbias}$$

Therefore,

$$E_{true} = E_A + E_B - \text{bias}$$

From the above analysis, the bias added twice so, bias has to be subtracted once from the result. If the resultant bit is more than 5 bits, want to normalize result value.

IV. PROPOSED METHOD

The method proposes design of low power and area efficient FIR filter structure with Carry save accumulation process. FIR filters realized using different combinations of adders and multiplier's block. The FIR filters having input coefficient with finite duration; this coefficient multiplied by input data bits and performs ripple carry addition with carry input (Ci). At the last stage carry save addition is performed and the output values are taken from the function Y(n). Similarly, each input bits multiplied and accumulated by same manner with single delay function. By the use of Carry Save Adder with Ripple carry adders, the delay and power reduction is possible. The filtered output is realized through the series of delay, multiplier and, adder.

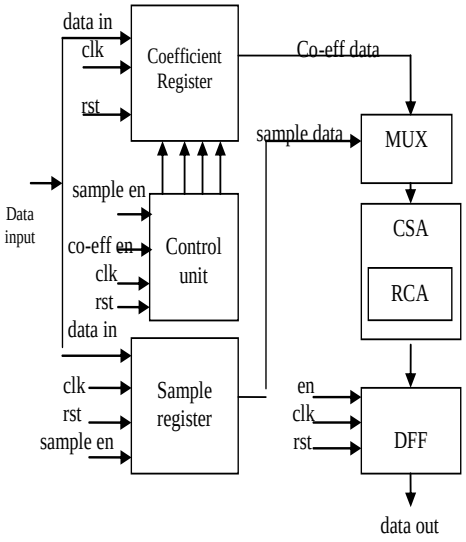


Fig4. Block Diagram for proposed method

The block diagram of proposed architecture shows that the data inputs are sent as data in or/and sample in for their registry. Both inputs enabled by control unit block of architecture. If the coefficient data enable is high, it takes the value of the input coefficient to the register. If the sample input enables is high, it takes the sample input values. Both the functions based on data inputs. The coefficient data and sample data are the input of the multiplier block. It performs the multiplication of coefficients and sends the data to carry save accumulation process. These processes having the blocks of ripple carry adders. It takes the input from the multiplier block and performs the operation. The carry save accumulator add the input coefficient with carry input (Ci). At the last, D flip-flop stores the output data and send the result.

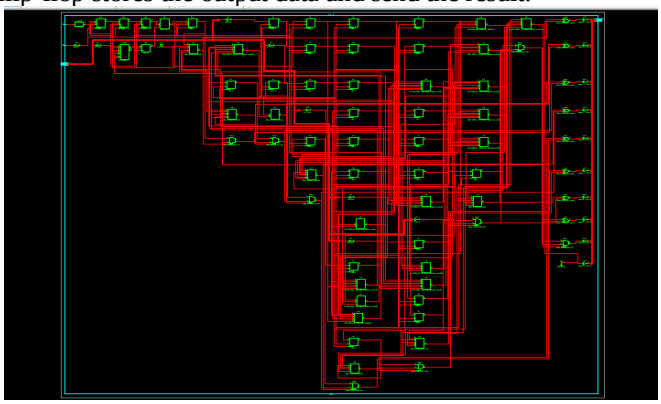


Fig5. RTL Schematic view for proposed method

Most of the multipliers like booth multiplier, Wallace tree multiplier, parallel multipliers etc., performs with various types of adders like full adder, carry look ahead adder, carry select adder out of all the adders is concluding, the carry save accumulation process is best one in terms of speed.

The power delay product is the product of power dissipation and total delay. The unit is Joule and it denoted as PDP.

Power delay product is,

$PDP = P_{dis} \times D_{tp}$

$= 14mW \times 1.436ns$

PDP = 201 Joule.

The proposed system achieves the best result of power delay product when compared to the conventional method.

A. Finite Impulse Response

Finite Impulse Response is widely used in Digital Signal Processing Applications such as speech processing, echo cancellation, adaptive noise cancellation and wireless communication.

The inverse discrete time Fourier transforms to find the corresponding time domain representation $h_d(n)$, as given by

$$h_d(m) = 1/2\pi \int_{-\pi}^{\pi} H_d(\omega)\, e^{j\phi(\omega)}\, d(\omega) \qquad \text{……..(1)}$$

Generally, $h_d(n)$ is not of finite length and since we were looking for a finite impulse response. Since we can truncate $h_d(n)$ and achieve an approximation of desired response. FIR Filter's impulse response is of finite duration because this will settles to zero value at a finite time interval. Design of FIR filter considers their performance and speed. The fast and efficient FIR filter design is considering the use of adders and multiplier blocks in the architecture.

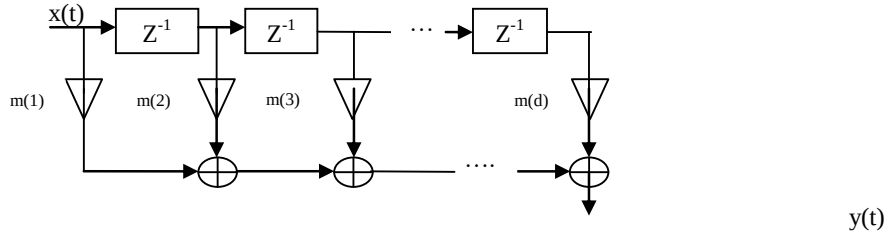


Fig6. FIR Filter Structure

The FIR Filter Direct form structure shows the delay of each data bit and it is multiply by MUX. The final added bits are taken from y(t) as output.

The order of filter coefficients are expressed as,
Zero order, $y_t = h_0 X_t$
First order, $y_t = h_0 X_t + h_1 X_{t-1}$
Second order, $y_t = h_0 X_t + h_1 X_{t-1} + h_2 X_{t-2}$
Third order, $y_t = h_0 X_t + h_1 X_{t-1} + h_2 X_{t-2} + h_3 X_{t-3}$
Fourth order, $y_t = h_0 X_t + h_1 X_{t-1} + h_2 X_{t-2} + h_3 X_{t-3} + h_4 X_{t-4}$

The frequency response of FIR filter is,

$$H(\exp(j\Omega)) = \sum_{t=0}^{N-1} h_m(\exp(-jt\Omega)) \qquad \text{……..(2)}$$

from equation (2),

$$H(\exp(j\Omega)) = h_0 \exp(-j0) + h_N \exp(-jN\Omega) + h_1 \exp(-j\Omega) + h_{N-1} \exp(-j\Omega) + \ldots. \qquad \text{……..(3)}$$

Taking the common factor, $\exp(-jN\Omega/2)$

$$H(\exp(j\Omega)) = \exp(-jN\Omega/2) * \{h_0 \exp(jN\Omega/2) + h(N) \exp(-jN\Omega/2) + h_1 \exp(j(N-2)\,\Omega/2) + h_{N-1} \exp(-j(N-2)\,\Omega/2) + \ldots\} \qquad \text{……..(4)}$$

If the filter length N+1 is odd, then the last term is single term of $h_{N/2,}$ which is the centre coefficient (tap) of the filter. The FIR Filters realized using different combinations of adders and multipliers blocks. Fast adders and fast multipliers results faster operations.

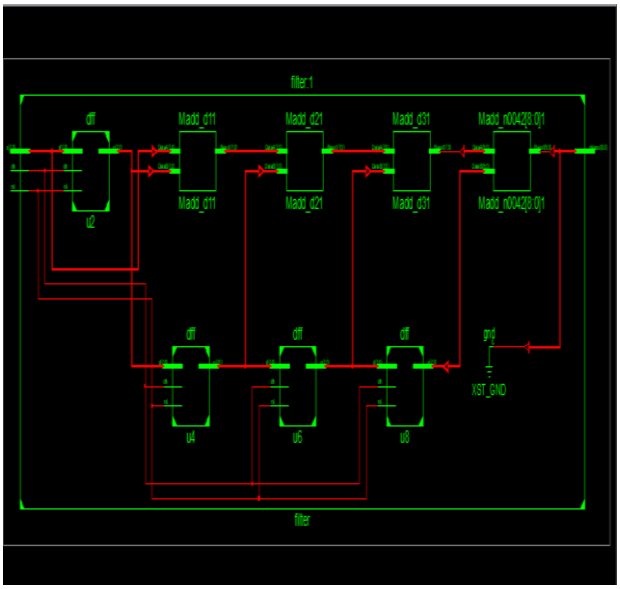


Fig7. RTL schematic view for FIR Filter Design

Filters are the most important part of Digital Signal Processing applications. Filters have two uses, one is signal separation and the other one is signal restoration. Signal separations used only when the signal contaminated with noise or other unwanted signals. The Signal is restoration used only when the signal has distorted. Implementing LTI form of FIR filter with its coefficient is,

$$y(t) = \sum_{t=0}^{N-1} h(t) \cdot x(m-t) \qquad \text{…….(5)}$$

The above equation can expressed in Z domain as,

$$Y(z) = X(z)\,H(z) \qquad \text{……(6)}$$

$$Y(z) = \sum_{t=0}^{N-1} h(t) z^{-t} \sum_{t=0}^{\infty} x(t)\, z^{-t} \qquad \text{……..(7)}$$

$$Y_t(z) = Y_{t0(z4)} + Z^{-1}Y_{t1(z4)} + Z^{-2}Y_{t2(z4)} + Z^{-3}Y_{t3(z4)} \qquad \text{……..(8)}$$

$$Y_t(z) = (H_{t0} + Z^{-1}H_{t1} + Z^{-2}H_{t2} + Z^{-3}H_{t4}) * (X_{t0} + Z^{-1}X_{t1} + Z^{-2}X_{t2} + Z^{-3}X_{t3}) \qquad \text{……(9)}$$

From equation (6),

Transfer function of the filter is,

$$H(z) = Y(z)/X(z) \qquad \text{…….(10)}$$

The Z transform of the filter is,

$$X(z) = \sum_{t=\infty}^{-\infty} x(t)\, Z^{-t} \qquad \text{……(11)}$$

The equation (11) becomes,

$$X(z) = a + 2aZ^{-1} + 3aZ^{-2} + 4aZ^{-3} + 5aZ^{-4} + 4aZ^{-5} + 3aZ^{-6} + 2aZ^{-7} + aZ^{-8}$$

$$X(z) = \frac{aZ^8 + 2aZ^7 + 3aZ^6 + 4aZ^5 + 5aZ^4 + 4aZ^3 + 3aZ^2 + 2aZ + 1}{aZ^8} \qquad \text{……(12)}$$

The linear phase response of FIR filter is,

$$H(\omega) = |H(\omega)|e^{j\phi(\omega)} \qquad \text{……(13)}$$

where, $\phi(\omega) = -\omega n_0$

The real and imaginary part of $H(\omega)$ is,

$$H(\omega) = R(\omega) + j\, I(\omega) \quad \ldots\ldots\ldots\ldots(14)$$

The magnitude and phase response defined as,

$$|H(\omega)| = \sqrt{R^2(\omega) + I^2(\omega)}$$
$$p(\omega) = \arctan\left(I(\omega)/R(\omega)\right)$$

Hence, the equation (12) obtained.
The frequency response of the filter is,

$$H(\omega) = h_0 + h_1 e^{-j\omega} + \ldots.. + h_{n-1}\, e^{-j(n-1)\omega} \ldots\ldots\ldots\ldots..(15)$$

$$H(\omega) = \sum_{t=0}^{n-1} h_t \cos t\omega - j \sum_{t=0}^{n-1} h_t \sin t\omega \quad \ldots\ldots\ldots\ldots..(16)$$

The transfer functions of amplitude response is,

$$H(\omega) = A(\omega) e^{j\phi(\omega)} \quad \ldots\ldots\ldots\ldots\ldots\ldots(17)$$

The impulse response of discrete time signal is derived from an inverse z transform of the transfer function and the frequency response of filter depends on coefficient of the impulse response.

B. Modified Carry save Accumulation process

The Carry save adder is used to the computes the sum of three or more bit values in binary format and it differs from other adders because the outputs of two numbers are in same dimensions as the input. These output sequences are the sequence of partial sum bits, and another that is a sequence of carry bit.

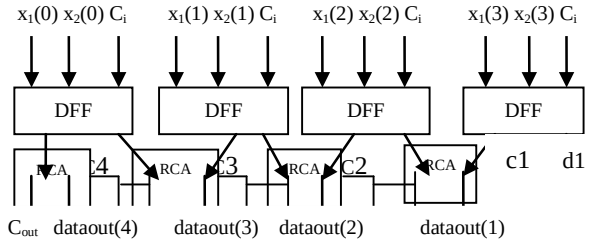


Fig8. Structures of Modified Carry save Accumulation process

In this carry save accumulator contains D flip flop for register and ripple carry adder with carry inputs. The input coefficients are multiplied and it is stored in D flip flop, it gives the input to RCA block it selects the enabled bits and is performed the addition. The CSA is performs with carry input from ripple carry adder. The use of carry save adder with ripple carry adder, delay reduction, and power reduction is possible.

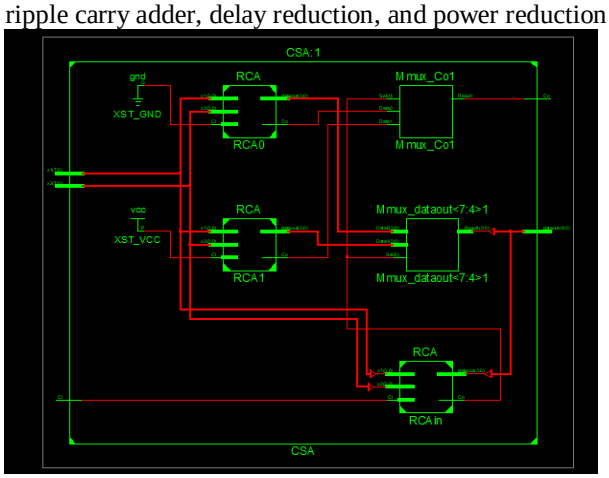


Fig9. RTL Schematic view for Modified Carry save accumulation process

The schematic diagram shows that the functions of the modified carry save accumulation process. Here, the input coefficients taken from ripple carry adder with its carry input. RCA0 and RCA1gets input from register unit and it perform the operation of ripple carry addition. After getting the result of RCA, sends the result to the MUX unit block. If the sample input coefficients presented, the RCA-in block takes the value and performs the operation. This output of RCA-in sends the result to data-out unit of MUX block. Moreover, carry output sends the result to the C_{out} unit of MUX block.

a. Carry save Adder

Carry save adder is high speed, multi-operand adder. It consists of the ladder of full adder block. It is a type of digital adders, but it is not a normal digital adder like others. Because the outputs of two bits are the same dimensions as the inputs bit. These two bits are called sequence of partial sum bits, and sequence of carry bits.

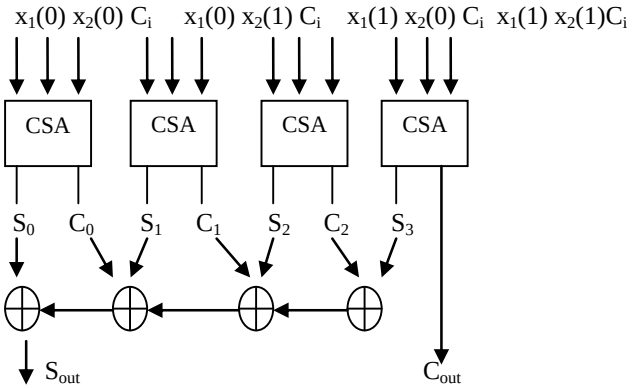


Fig10. Block diagram for Carry save adder

The Carry Save Adder eliminates carry chaining by saving the carry and provides the carry to the next higher bit adder. It consists of multiple one-bit full adders without carry chaining. Each CSA block having 2bit coefficients with its carry input. The sum of the partial bit is calculated, and the carry inputs are given to the next higher bit. Then it added to the sum of the partial bit in the next stage. The sum is available after two stages of successive additions.

b. Ripple carry adder

Ripple carry adder is a digital adder circuit that performs the operation of arithmetic sum of two binary bits. Ripple Carry Adder is designed by cascaded form of full adder circuit. The carry output of least significant full adder block is the input of next most significant full adder block. Each carry bit gets rippled into the next stage. The carry inputs are only valid when the sum and carry output bits have occurred in half adder block.

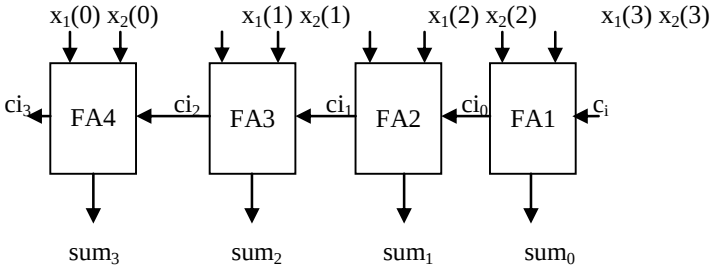


Fig11. Block diagram of Ripple carry adder

In ripple carry adder, the output is generated by the previous stage of carry output is produced. After the carry values are rippled by the block of Least Significant full adder to Most Significant full adder, the sum of the most significant bit is generated. The sum Sum_0 and carry output ci_3 of the full adder block-1 is used after getting the propagation delay of full adder block-1. In the same way, a sum out sum_3 of the full adder block-4 is generates after getting the result of joint propagation delays of full adder block-1 to full adder block-4. After generating the result from Ripple Carry Adder block, the coefficient sends to carry save adder unit. The output is stored in D flip flop of MUX unit. As a result, the final sum and carry bits will be generates after a considerable delay response.

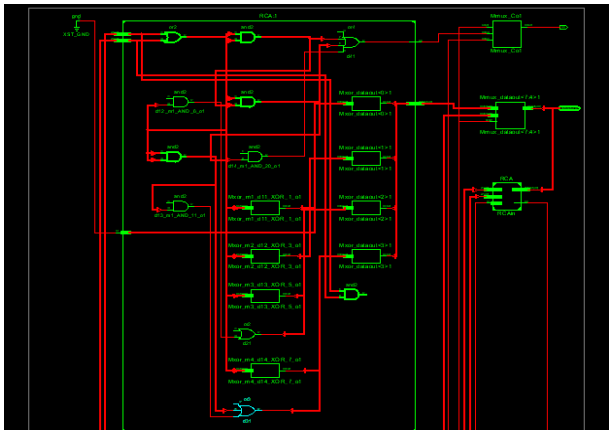


Fig12. RTL schematic view for filter coefficient allignment in RCA block

The diagram shows the alignment of filter coefficient bits in a ripple carry adder. The input data bits are stored in D flip-flops and it aligned based on low/high values of coefficients; the XOR operations used to generate the separate the coefficient to MUX unit. These values arranged in the RCA unit and send to the MUX block.

V. RESULT

Thus the design of low power and area efficient FIR filter was designed and with the use of modified carry save accumulation process. The area, power and, delay parameters are analyzed. In this parameter, separate values analyzed such that gate delay, path delay and, net delay for delay analysis, voltage, and current for power analysis, LUTs, slices, and IOBs for area analysis. The power delay product was calculated. All the parameters compared with the conventional method, it achieves a better result.

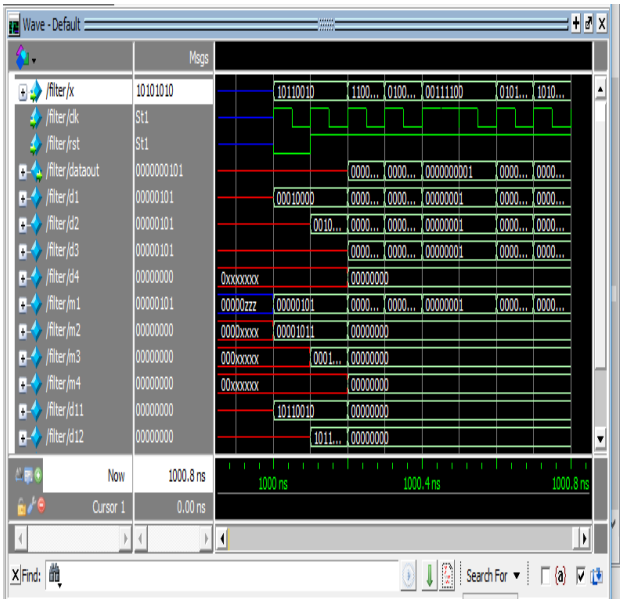


Fig. 13 Simulated output for proposed method

The simulated result shows the design of FIR filter using a modified carry save Accumulation process. An input of the filter is x1 and x2. The output is data out. Initially, we can give the input data by force constant and trigger the clock signal. The reset value is one, at the stage compilation. Triggering the clock cycle changes the data out. After getting simulation output, we can view each stage of the result.

A. Simulated output of Modified Carry save Accumulator

The diagram shows the simulated output of carry save adder with ripple carry adder. The inputs A, B and, Ci, the outputs So and Co. In Ripple Carry Adder block, the coefficient bits added with its carry inputs and sents the results to carry save adder. If the control unit enables the input bits and, produces the clock signal, the first two bits are sends to the RCA block with its carry input (Ci), it gives output S1 and C1. Next stage C1 is added with the next two input bits. Similarly, all the coefficients added the carry value and perform the operations. Finally, all the sum is added by Carry Save Adder. The result is stored in D flip flop.

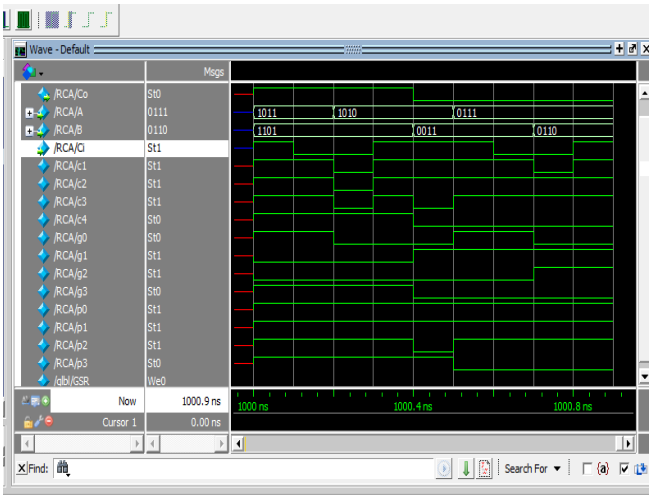


Fig. 14 Simulated output for Modified carry save accumulator

B. Device Utilization Summary

Slice Logic Utilization	Used	Available	Utilization
Number of Slice Registers	30	4,800	1%
Number used as Flip Flops	29		
Number used as AND/OR logics	1		
Number of Slice LUTs	30	2,400	1%
Number used as logic	21	2,400	1%
Number used as Memory	1	1,200	1%
Number of LUT-FF pairs	12	38	30%
Number of bonded input and outputs	19	102	18%
Number with an unused Flip Flop	18	39	46%
Number with an unused LUT	9	39	23%
Number of occupied Slices	13	600	2%
Number of MUXCYs used	8	1,200	1%
Number of BUFG/BUFGMUXs	1	16	6%

C. Performance analysisresult

Power Analyzes		
Voltage(V)	**Current(mA)**	**Power(mW)**
1.2	0.4	48
2.5	0.3	75
2.5	1	25
Delay Analyzes		
Gate delay(ns)	**Path delay(ns)**	**Net delay(ns)**
1.436	9.703	1.156
Area Analyzes in number of LUTs		
LUTs	**Slices**	**IOBs**

12	30	19
Timing Summary		
Minimum time delay achieved: 1.436ns Input arrival time delay before clock cycle: 2.886ns Output required time delay after clock cycle: 9.843ns Frequency : 696.452MHz		

D. Comparison Table

Methods	**Power (mW)**	**Area**		**Delay (ns)**	**PDP**
		LUTs	**Slices**		
Existing method uses MCM with Pipelined Adder unit [1]	90	82	44	1.690	1521
Existing method uses Booth multiplier with carry skip adder network [7] &[18]	59	72	44	15.83	9339
Existing method uses carry select adder with shifters used for each block of FIR coefficients [8]	27	18	45	7.819	1911
Existing FIR design uses Wallace tree and Vedic multipliers for the performance reduction of delay [11]&[16]	21	67	46	4.357	914
Proposed Method uses Modified Carry save Accumulation Process	14	12	30	1.436	201

VI. Conclusion

Thus, we conclude that the design and performance of the FIR filter is better robustness than the existing method by area, delay and, power parameters. The reduced use of adders and multiplier block gives systems speed higher. Due to the reduction of the number of bonded input and outputs (IOBs), slices and, LUTs, the area utilization is less. By using modified carry save accumulator method, the filter coefficients aligned properly, therefore it gives better throughput. In future work, the number of adder and multipliers reduce by using different algorithms.

References

1. Basant kumar Mohanty, Pramod kumar Meher, "A High performance FIR Filter Architecture for Fixed and Reconfigurable Applications," IEEE Trans. VLSI system, vol. 24, No. 2, pp. 444–452, February 2016.
2. JongsunPark, Woopyo Jeong, Hamid Mahmoodi-Meimand, "Computation Sharing Programmable FIR Filter for Low-Power and High Performance Applications," IEEE Inter. Jour. Of Solid State Circuits, vol. 39, No. 2, pp. 348-357, February 2004.
3. A P Vinod and Edmund M K Lai, "Low Power and High-Speed Implementation of FIR Filter for Software Defined Radio Receivers," IEEE Transactions on Wireless Communications, vol. 5, No. 7, pp. 1669-1675, July 2006.
4. Kuan-Hung Chen and Tzi-Dar Chiueh, "A Low Power Digit-Based Reconfigurable FIR Filter," IEEE Transactions on Circuits and Systems, vol. 53, No. 8, pp. 617-621, August 2006.
5. N Sameeksha Rai, Pannaga Shree B S, Meghana YP, Arunkumar P Chavan, H V Ravish Aradhya, "Design and Implementation of 16tap FIR filter for DSP Application," International coference on Advances in Electronics, Computer and, Communications, 2018.
6. S Madhavi, K Rasagna, N Kavya, M Sindhu, "Implementation of Programmable FIR filter using DADDA Multiplier and Parallel prefix Adder," International Conference on Inventive Research in Computing Applications, 2018.
7. Nisha Chaudhary, Shweta Meena, "Design and Implementation of FIR Filter with Modified Product Accumulation Block using Booth Multiplier," International Conference on Trends in Electronic Informatics, 2017.
8. Sumalatha Madugula, Panchala Venkata naganjaneyulu, Kodati Satya Prasad, "Implementation of FIR Filter for low power and Area Minimization using Shift-Add Method without Multipliers," Inter. Jour. Intelligent Engineering and Systems, Vol. 10, No. 6, pp. 250-262, July 2017.
9. Gopalakrishnan, R., Mohan, A., Sankar, L. P., & Vijayan, D. S. (2020). Characterisation On Toughness Property Of Self-Compacting Fibre Reinforced Concrete. In Journal of Environmental Protection and Ecology (Vol. 21, Issue 6, pp. 2153–2163).
10. L Durga Prasad, J E N Abhilash, B Subrahmaneswara Rao, "Design of Efficient FIR Filter with EDBNS multiplier using Transpose method for various Applications," Inter. Jour. Engineering Development and Research, vol. 5, No. 4, pp. 1182-1190, 2017.
11. RGeetha, R Bavya, "Design of FIR Filter using different multiplier Architecture for High Speed and Low Power Applications," Inter. Jour. Of MC square Acientific Research, vol. 9, No. 1, pp. 1-9, April 2017.
12. R Gopalana, A Parameswari, "MCM based FIR Filter Architecture for High performance," South Asian Journal of Engineering and Technology, vol. 2, No.21, pp. 83-88, March 2016.
13. R K Mohana Lakshmi, P Saravanakumar, "FIR Filter Implementation Based on MCM Technique using Wallace Multiplier and Carry save Adder," Inter. Jour. Of Advances in Engineering, vol. 1, No. 3, pp. 110-115, March 2015.
14. Vadapalli Siddhartha, "FIR Design and Implementation using Remez Exchange Algorithm with Multipliers and Adders," Inter. Jour. Of VLSI and Embedded Systems, vol. 6, pp. 1599-1606, October 2015.
15. T Radha, M Velmurugan, "Low Power Digital FIR Filter Design Using Optimized Adder and Multiplier," Inter. Jour. Of Advances in Engineering, vol. 1, No. 4, pp. 538-544, April 2015.
16. M Jayashree, "Design of High Speed and Area Efficient FIR filter Architecture using Modified Adder and Multiplier," Inter. Jour. Of Engineering and Techniques, vol. 4, No. 3, pp. 537- 543, May 2018.
17. A P Vinod, E M-K Lai and, Emmanuel, "Implementation of Low Power and High Speed Higher Order Channel Filter for Software Radio Receiver," IEEE Inter. Symposium on Personal, Indoor and Mobile radio Communications, 2006.
18. Shuchi Nagaria, Anushka Singh and, Vandana Niranjan, "Efficient FIR Design Using Booth Multiplier for VLSI Applications," Inter. Conference on Computing , Power and Communication Technologies, September 2018.
19. R Mahesh and A P Vinod, "New reconfigurable architecture for Implementing FIR filter with low Complexity," IEEE Trans. Computer Aided Design Integr. Circuits Syst. II, Exp. Briefs, vol. 53, No. 8, pp. 275-288, February 2010.

20. E Chitra, T Vigneswaran, "DA based Efficient Parallel Digital FIR Filter Implementation for DDC and ERT Applications," Inter. Jour. of Engineering and Technology, vol. 7, No. 2, pp. 727-733, May 2015.
21. P C Franklin, M Ramya, R Nagarajan, T M Mini Priya and, M Balamurugan , "Design of Resource Efficient FIR Filter Structure using Adders and Multiplier," Inter. Jour. of Advanced research in Computer and Communication Engineering, vol. 3, No. 2, pp. 5434-5437, February 2014.
22. Dhivya V M, Sridevi A, "A High Speed Transposed form FIR Filter using Floating Point DADDA Multiplier," Inter. Jour. of Research in Engineering and Science, vol. 2, No. 5, pp. 14-20, May 2014.
23. Tholkapiyan, A.Mohan, Vijayan.D.S, A survey of recent studies on chlorophyll variation in Indian coastal waters, IOP Conf. Series: Materials Science and Engineering 993 (2020) 012041, 1-6.
24. C Uthayakumar, B Justus Rabi, "Design of FIR Filter using Window Method," Inter. Jour. of Electronics and Communication, vol. 2, No. 10, pp. 21-25, October 2014.
25. Yu Pan and Pramod Kumar Meher, "Bit-Level Optimization of Full-Adder Trees for multiple constant multiplications for efficient FIR filter implementation," IEEE Trans. On Circuits and System I: Regular Papers, vol. 61, No. 2, pp. 455-462, February 2014.
26. T Ramesh Reddy and, K Soundara Rajan, "Low Power and Low Area Digital FIR Filter using Different Multipliers and Adders," Inter. Jour. of Engineering Research and Technology, vol. 1, No. 3, pp. 1-4, May 2012.
27. Levent Aksoy, Paulo Flores and, Jose Monteiro, "Exact and Approximate Algorithms for the Filter Design Optimization Problem," IEEE Trans. On Signal Processing, vol. 63, No. 1, pp. 142-154, January 2015.
28. Kundeti Nagarjuna and, R krishna, "Efficient Implementation of High Speed Low power Digital FIR Filter Architecture Design by Radix 4 Multiplier," Inter. Jour. of Science, Engineering and Technology, vol. 2, No. 4, pp. 254-272, May 2014.
29. S Satheeskumaran and, K Sasikala, "VLSI Implementation of Modified Distributed Arithmetic based Low Power and High Performance Digital FIR Filter," Inter. Jour. of Current Engineering and Scientific Research, vol. 4, No. 10, pp. 59-66, 2017.
30. B Jeevan, S Narender, C V Krishna reddy and, K Sivani, "A High speed Binary Floating Point Multiplier using Dadda Multiplier," IEEE Inter. Multi-Conference on Automation, Computing, Communication, Control and, compressed sensing, pp. 455-460, June 2013.