*IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)*
*Volume 6, Issue 6, Ver. III (Nov. - Dec. 2016), PP 37-44*
*e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197*
*www.iosrjournals.org*

# Efficient VLSI Architectures for FIR Filters

O. Venkata Krishna[1], Dr. C.Venkata Narasimhulu [2], Dr. K. Satya Prasad[3]
[1](ECE, CVR College of Engineering, Hyderabad, India)
[2] (ECE, Geethanjali College of Engineering and Technology, Hyderabad, India)
[3] (ECE, University College of Engineering - JNTUK, Kakinada, India)

***Abstract*:** *The Finite Impulse Response (FIR) filters are widely used in many Digital Signal Processing (DSP) applications. For these applications, the low power, less area, high speed and low complexity FIR filter architectures are required. The researchers have proposed many FIR filters to meet the above design specifications. This paper is focused on the some efficient reconfigurable FIR filter architectures. The author Mohanthy et. al introduced a FIR filter architecture implemented for higher order fixed and reconfigurable applications. This filter is a block FIR filter, which realized in transpose-form configuration with less area, low power and less delay for large order filters. The second filter architecture in this paper is a custom reconfigurable power efficient FIR filter using multiplier less (Reduced Adder Graph) RAG-N Algorithm. In this method, the multiplier is realized using adders and shifters. This architecture is easy to implement, symmetrical and stable system. The next approach in this paper is implementation energy efficient FIR filter using alternative adders. In this, the filters are optimized for low power using multiplier less Multiple Constant Multiplication (MCM) algorithm. The alternative adder circuits are truncated and approximated to reduce the power consumption. Another efficient programmable FIR filter is implemented using Karatsuba Multiplication Algorithm. In this architecture, a parallel, modified booth pre-encoded, carry save Wallace tree multiplier is used for the multiplication of large numbers. This architecture is more efficient in terms of power, area and speed comparatively than other FIR filters. Finally, the comparison is taken place among the four efficient Very Large Scale Integration (VLSI) FIR filter architectures in terms of power, area and speed.*

***Keywords:*** *FIR, Low power, Booth encoder, Wallace tree multiplier, VLSI, MCM and RAG-N.*

## I. Introduction

Finite-Impulse Response filters are important building blocks in many DSP systems such as portable wireless systems, mobile phones and battery operated multimedia devices. The design metrics power, area and speed are considered as important parameters for VLSI architectures. The design methodology for high speed, low power and low area is essential in the implementation of all DSP (Integrated Circuit) ICs. The minimization of power dissipation and area depends on the selection of appropriate algorithms and mapping on to suitable architecture. A great extent of power can be reduced by the elimination of redundant and irrelevant computations in the particular system. Based on the above aspects, this paper reviewed different high performance VLSI FIR architectures for DSP applications. These architectures are implemented to reduce power dissipation, increasing the speed of operation and minimizing the area of the chip.

The total power consumption of Complementary Metal Oxide Semiconductor (CMOS) circuit is a combination of static power and dynamic power. Static power is defined as the power consumed when the input signal is a constant value. The dynamic power consumption is the power consumed when the transistors are active and the input signal change the state of the transistors frequently. In the existing methods of FIR filters, lot of power is consumed because of the switching activities in the circuit. The dynamic power can be reduced by proper identification and suppressing the unnecessary activities in the circuit. The speed of the any architecture depends on the longest path delay of the combinational circuit, setup time of the sequential circuits and clock skew. The power reduction and speed enhancement are open problems in the VLSI design.The previous works presented reconfigurable FIR filter architectures which were independent of the number of taps and non-zero digits in each tap were arbitrarily assigned [1]. The intention of the authors was to reduce the precision of coefficients and thus the filter complexity without affecting the filter performance. But some of the architectures demanded huge hardware resources and this makes the method infeasible for power constrained receiver applications. High speed programmable based reconfigurable FIR filter was proposed by different authors. This paper presents some efficient VLSI architectures for the solution of the above problems.

## II. Block FIR Filter in Transpose Form Configuration

In this section, the block FIR filter architecture is presented using MCM scheme, which is suitable for fixed filters as well as for reconfigurable filters.The realization of block FIR filter in the form of transpose configuration, which includes the MCM scheme and inherent pipelining. This realization gives low complexity and area-delay efficient architecture for large order FIR filters. This efficient FIR filter architecture is suitable

for both fixed and reconfigurable applications. Basically, the block processing technique is used to provide high throughput and to improve area- delay efficiency. The block based FIR filter can be realized easily for direct form configuration, where as the transpose form does not supports the block design directly. But, to get computational improvement in the filter, the MCM scheme with transpose form is needed. The transpose form has the inherent advantage of pipelining, and supports high sampling frequencies. Hence, a transpose form block FIR filter architecture is presented in this section.

**2.1 Transpose Form Block FIR Filter for Reconfigurable Applications**

The mathematical formula for the transpose form configuration FIR filter is given by;

$$Y(z) = S^0(z)[(z^{-1}(\ldots(z^{-1}(z^{-1}c_{M-1} + c_{M-2}) + c_{M-3}) + \ldots) + c_1) + c_0] \quad \ldots\ldots\ldots\ldots(1)$$

where $\mathbf{S}^0(z)$ and $\mathbf{Y}(z)$ are the *z*-domain representation of $\mathbf{S}^0_k$ and $\mathbf{y}_k$, respectively. The block FIR filter architecture implemented according to the above recurrence relation (1) of transpose form configuration is shown in Fig.1. The block architecture implemented by the sub blocks are Register Unit (RU), Coefficient Selection Unit (CSU), M-number of Inner Product Units (IPU) and Pipeline adder Unit (PAU).

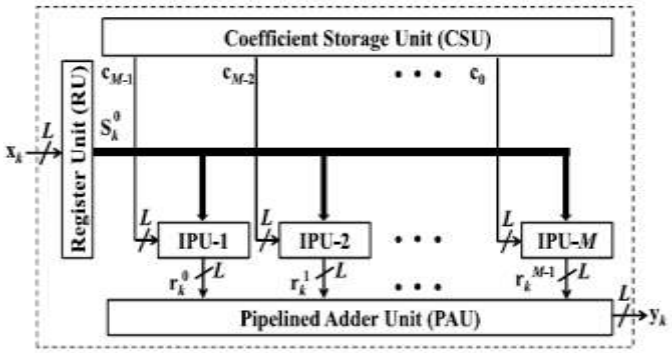


**Fig. 1.** Block FIR filter for reconfigurable applications.

The CSU block in the block FIR filter is used to store the coefficients of the all possible reconfigure applications. This block is constructed by N number of Read Only Memory (ROM) Look Up Tables (LUT), where N is the length of the filter. This block gives the information of the filter coefficients of particular channel filter. The RU block in the architecture receives the input samples $x_k$ during the clock cycle of k. This block produces L rows of $S^o_k$ in parallel form. These 'L' rows are applied to M number of IPUs of the block FIR architecture as shown in the Fig.2.The CSU also produces M short-weight vectors and applied for the M number of IPUs during the $k^{th}$ cycle. The inputs for the $(m+1)^{th}$ IPU are $C_{M-m-1}$ weight vector and L rows of $S^0_k$ from RU. Every IPU block performing the product on matrix vectors produces the partial outputs of the filter i.e $r^m_{k..}$ The Fig.3 represents the internal structure of the $(m+1)^{th}$ block. This IPU block also consists of L number of Inner Product Cells (IPC). The internal structure of the $(l+1)^{th}$ IPC also shown in Fig.4 for block size L=4. It receives the coefficient vector $c_m$ from the CSU block and $(l+1)^{th}$ row of $S^0_k$, which computes the result as *r(kL-l)* for $0 \le l \le L-1$. Following the same procedure, all the M IPU blocks produce M block of result. Finally, the partial inner products are added using PAU. The internal structure of the PAU is shown in Fig.5. The PAU gives the final L block size filter outputs. For every clock cycle, the structure receives the L filter inputs and produces a block of L filter outputs. The time duration for this architecture is given by the equation (2);

$$T = TM + TA + TFA \log_2 L \quad \ldots\ldots.(2)$$

where, $T_M$ is delay of one multiplier, $T_A$ is the delay of one adder, and $T_{FA}$ is one Full adder delay.

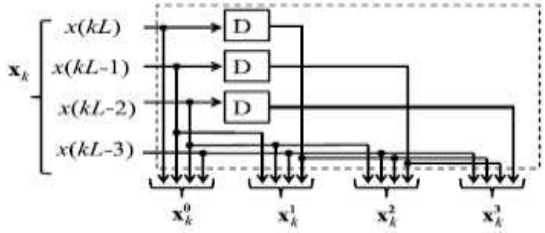


**Fig. 2.** Register Unit (RU) internal structure for block size L=4.

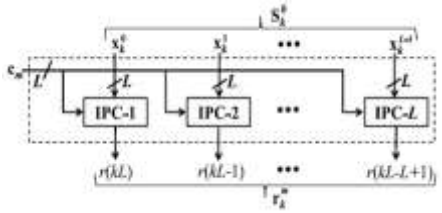


**Fig. 3.** $(m+1)^{th}$ internal structure of IPU.

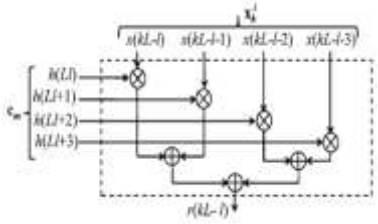

**Fig. 4.** IPC internal structure for $L = 4$.

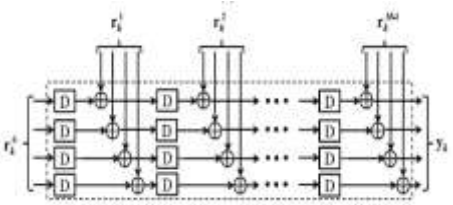

**Fig. 5**. PAU internal structure for block size L=4.

**2.2 FIR Filter Architecture Based On MCM For Fixed- Coefficients**

The MCM based FIR filter architecture does not require CSU and IPU blocks for fixed coefficient applications. This structure is fixed for one type of filter only. In this architecture, the multiplication process is performed in horizontal and vertical common sub-expression elimination method. Hence number of shift–add operations are reduced in MCM block. The Fig.6 represents the architecture of MCM based FIR filter.

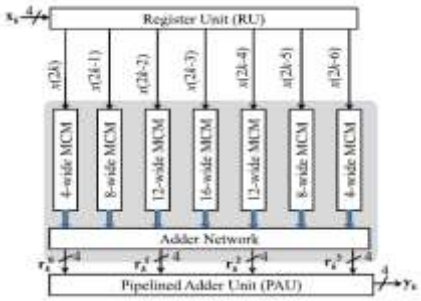


**Fig. 6.** The MCM based FIR filter architecture for block size $L = 4$ and filter length $N = 16$.

The MCM based FIR filter architecture consists of six MCM blocks with respect to six input samples. These MCM blocks produce product terms using shift and adding method by sub-expression of MCM blocks. The inner products are finally added and output is given by PAU as shown in Fig.5.This reconfigurable FIR filter architecture contains CSU block, RU block, M number of IPUs with internal product cells and PAU. The CSU consists of *N* ROM units of *P* words each, where *P* is the number of FIR filters to be implemented for reconfigurable structure. Each IPU block consists of L number of IP cells, where each IP cell involves *L* multipliers and *(L-1)* adders. The RU involves *(L-1)* registers of *B*-bit width. The PAU involves *L(M-1)* adders and the same number of registers. Each register has a width of *(B+ B`)*, where B is the bit width of input sample and B` is bit width of filter coefficients respectively. Finally, the FIR filter structure realized by *LN* multipliers, L(N-1) adders, and [B(N-1) + B`( N-L) ] flip flops. This process takes place for L samples in one clock cycle. There is no multiplier based direct form reconfigurable FIR filter architecture previously. The authors Mohanthy et.al [2] proposed a first time multiplier based direct form FIR filter architecture. These two designs are synthesized using design compiler and the power, area and minimum clock periods are estimated and corresponding reports are generated.

## III. Multiplier Less FIR Filter Configuration Using RAG-N Algorithm

In this section, a customized reconfigurable power efficient FIR filter architecture is introduced [3]. The FIR filter is realized using RAG-N algorithm without any direct multiplier. Generally, multiplier only occupies more area and it increases power consumption in FIR filters. Hence the multiplication process is done by only adders and shifters using RAG-N algorithm. The general equation of any FIR filter is given by,

$$y(n) = b_0 x(n) + b_1 x(n-1) + \ldots + b_{N-1} x(n-N+1) \quad \ldots\ldots..(3)$$

The above FIR filter equation (3) can be constructed by Adder/subtractor (A/S) unit and Right-Left (R/L) shifter. The RAG-N algorithm is used to realize the coefficients of the filter. The accumulation block is different to multiplier, which occupies less area. Generally, the reconfigurable FIR filter architecture consists of accumulation and multiplication blocks. The accumulation block is constructed by adder/subtractor block and D flip flop.This block can give more flexibility for the reconfiguration without altering the inputs. The reconfigurable architecture is made up of multiplier and accumulation unit. It is constructed by using specific number of vertices. Each vertex has programmable addition and subtraction unit and shifter (left/right) unit. Shifter is attached before the adder shifter unit as shown in Fig.7.


**Fig.7.** Architecture for vertex realization.

There is flexibility of interconnections between the adjacent vertices so that we can select any one input from adjacent vertices or input. Similarly other input of the adder shifter unit is selected in the same way but without shifting. Shifter provides the power of two multiplication or division. Division is important because the coefficients must be realized in order. The output of the adder shifter unit is given to the register and stored in that. Therefore high frequency clocked design is created. In the architecture of Fig.8, left programmable shifter is connected to each end of multiplier stage to obtain even coefficients. This stage also realizes negative coefficients.

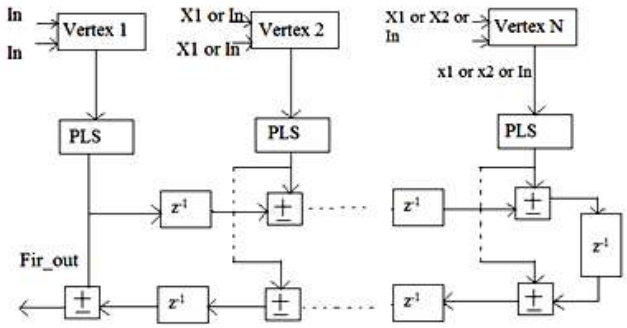


**Fig.8.** Reconfigurable Hardware.

The RAG-N algorithm optimizes and minimizes the cost of coefficients. If the coefficients generated through multiplier stage is negative then it is rectified by accumulation stage which perform subtraction in place of addition in A/S unit. This accumulation stage is different from other stage because it is based on pipelining method with the ability to pass the inputs to all stage without change. At first time, coefficients are not known. So two of the A/S units have to be made as dashed line in this stage and hence this stage is useful. The realization of the FIR filter coefficients using adders and shifters is shown in Fig.9. For example, the impulse function with 8993 coefficients is considered as input for the filter. It provides the shifted value of it as a second input and further using the shifter and adder and a delay element realize the specific coefficient. A set of three 16 bit adders only is required to realize the coefficients. Hence, the number of adders are reduced for this design comparatively multiplier architecture.

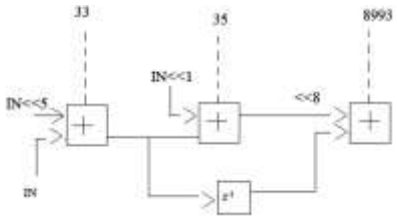


**Fig. 9.** Generation of coefficient

**3.1 RAG-N Algorithm**

The RAG-N algorithm optimizes the coefficients in terms of less number of adders and also optimizes the multiplier using look tables. The following steps are involved in the algorithm.

1. Make all even coefficients to odd and store them in fundamental set which are not completed.
2. Evaluation of cost of coefficients using table of lookup.
3. Removal from all 0-cost of set of fundamental to reduce the costs.
4. Make a set for storing of all fundamental depending upon their cost sets.
5. Pair the set with multiplication of power-of-2 with same Set in order not to add fundamentals. So repetition of step 4.

This architecture design is implemented by Sakthivel et.al in [3] using Cadence tools. This Filter is designed for the frequency of 140 MHz. The power consumption for 20 tap filter is 0.0166mW/MHz. The architecture occupies less hardware due to multiplier-less, which consumes less power for less number of coefficients.

## IV. Energy-Efficient Digital CMOS FIR Filters With Approximate Adder Circuits

This section presents the synthesis of energy efficient FIR filter architecture improvement which were already optimized for area and power using state of the art MCM technique. The approximation of adders and multipliers evaluation is also presented [4].

**4.1 Approximate Adders Evaluation**

The carry chain only determines the critical path of the N bit adders. Here, the approach is the breaking of the carry chain into different blocks such as precise and approximate blocks. All the blocks are parallel computed and gives faster results. The speculation of the carry in of each block alters the magnitude of the accuracy and error. In the Fig.10, the speculation of carry-in is statistically set to '0' and '1'( i.e. set to GND and VDD) for (Error Tolerant Adders) ETA-I, ETA-II and ETA-IV.The ETAI adder in the Fig. 10 is divided as precise and approximate blocks. The precise block consists of a conventional Ripple Carry Adder (RCA) and the carry-in is statistically set to '0'resulting in 50% probability of a correct carry-in speculation. The approximate block consists of half adders and the generate operator for each bit position i.e. AND gate. In this block no carry chain in propagation. The sum is computed in the direction of MSB to the LSB. Each bit is computed by half adders until the first carry generate is met (i.e. AND operation equal to '1'). When this condition occurs all the remaining bits are statically set to '1'.

In the ETA-II, the original precise block is divided into more precise blocks. Each block is a RCA block and consists of k-bits and no dependency between them. The k-bit Carry Look Ahead (CLA) topology is used to calculate the carry-in speculation in each RCA block. Then the critical path is k bit RCA and k-bit CLA. The ETA-IV block topology is same as the previous one ETA-II block. The main difference is that carry-in chain which is not presented longer, but in ETA-IV the carry-in chain presented for long time. The ETA-IV is combination of CLA and Carry Select Logic (CSL). The CLA selects the multiplexer from the CSL block. The CSL block gives the carry-in speculation for the next RCA block. Then the critical path is k-bit RCA+ Kbit CLA + k-bit CSL.

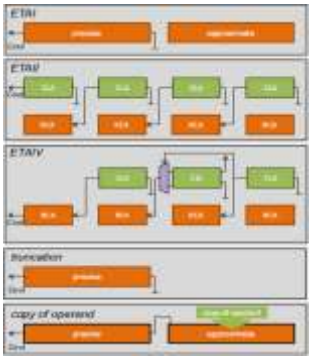


**Fig. 10.** Approximate N-bit adders under evaluation

**4.2 FIR Filter Implementation using MCM Technique**

In the parallel digital FIR filters implementation in the transposed architecture, as in Fig.11, the same input is multiplied by a set of constant coefficients, whose operation is referred to as MCM. The constant multiplications are generally realized in parallel using additions/subtractions and shifting operations. Since the full implementation of multipliers is costly in hardware, the MCM problem is formulated as an optimization problem to find both: (i) the minimum number of addition/subtraction operations which implement the constant multiplications, since word-shifts are free in the hardware parallel implementation of the FIR filters, and (ii) the shortest word-length for each adder/subtractor.

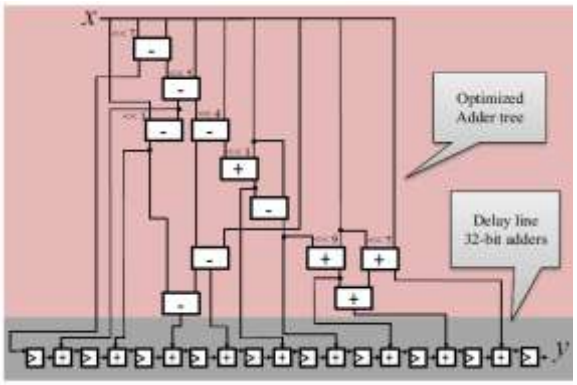


**Fig. 11.** A ten taps transposed FIR filter implemented by MCM algorithm.

The approximate strategy for MCM instances, in order to verify the impact on power and area in the digital FIR filters, when the precisions of the operations are relaxed.The used MCM instances were obtained from the exact depth-first search algorithm. This algorithm finds the minimum solution through the use of lower and upper bound values of the search space for the MCM problem instances. Fig.11 depicts a hypothetical example of an optimized ten taps digital FIR filter regarding the MCM operation of a transposed form filter whose taps are represented by 127x, 95x, -93x, -15x, 78x, 81x, 80x, 592x, 721x, and 129x.The input sample is denoted by x and the output by y, and word registers and adders are represented as boxes. In this example the parallel filter is designed with left shifters (≪), addition (+) and subtraction (-) operations. At the bottom of the figure the filter delay line is depicted, which realizes the sum process that outputs one filtered sample (y) per clock cycle, while the remaining part of the architecture is known as the partial terms computation or the adder tree hardware. The adder tree is previously optimized by the solver technique.

## V. FIR Filter Based on Karatsuba Algorithm

In this section, a new algorithm based FIR filter is introduced for efficient applications. The algorithm is Karatsuba Multiplication Algorithm, which reduces the area occupied by the filter and power consumption of the filter [5]. The Karatsuba algorithm reduces the 2N×2N bit multiplication into a set of (N+1)×(N+1) or N×N bit multiplications and additions.The multiplication is very important operation in FIR filters. In this design, a modified Booth encoder carry Save (CS) Wallace tree multiplier is implemented using Karatsuba algorithm [6]. The architecture of the Multiplication Unit (MU) presented in the Fig.12. is used for FIR filter implementation .The coefficients of the FIR filter are pre-encoded and stored in the Modified Booth (MB) form to increase the efficiency. It consists of a Wallace tree for adding the partial products, one Partial Product Generator (PPG) and one correction term to obtain the right results of the multiplication.

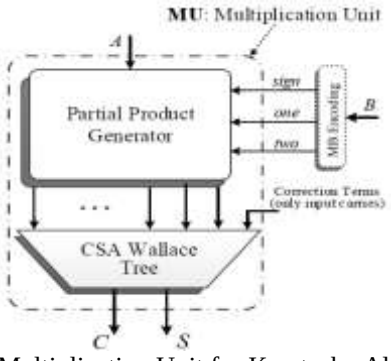


**Fig. 12.** Multiplication Unit for Karatsuba Algorithm.

The general transfer function of FIR filter in time domain is given in the equation (4) ,

$$y(n) = \sum_{k=0}^{T-1} h(k).x(n-k) \qquad \text{……(4)}$$

Where ,T is filter order, h(k) is filter coefficients, x(n) is input and y(n) is outputs. The conventional architecture for FIR filter for order T=4 is shown in the Fig.13. It consists of MU (shown in Fig.12), basic delay unit Multiplexers, Carry Look-Ahead Adder (CLA) and final correction term.

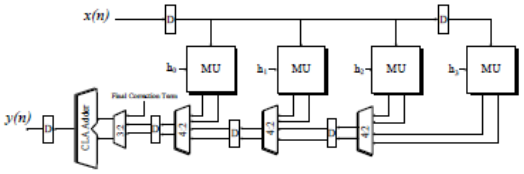


**Fig. 13.** The transposed form Conventional architecture for FIR filter.

**5.1 Karatsuba Algorithm**

Let us consider two numbers: *a* and *b* both of 2*N* bits. Each number can be divided to two sub-words of *N* bits, as follows: $a = a_H.2^N + a_L$ and $b = b_H.2^N + b_L$

The product *P* of these numbers is given by the next relation:

$$p = a.b = a_H.b_H.2^{2N} + (a_L.b_H + a_H.b_L).2^N + a_L.b_L \qquad \text{……(5)}$$

Where, $P_H = a_H.b_H$, $P_M = a_L.b_H + a_H.b_L$, $P_L = a_L.b_L$. We compute also the next quantity:

$$dP = da.db = (a_H - a_L).(b_H - b_L) = P_H + P_L - P_M \qquad \text{……..(6)}$$

Consequently the term $P_M$ is given by the following relation: According to (5) and (6) for the product P holds that: $P_M = P_H + P_L - dP$ and $P = (P_H.2^N + P_L).(2^N + 1) - dP.2^N$.

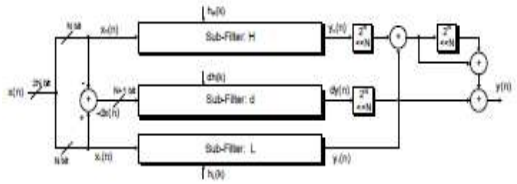


**Fig. 14.** Karatsuba FIR Filter architecture

The Fig.14 shows the architecture of the Karatsuba FIR filter. According to the above equation, the computation of the product *P* can be simplified in the calculation of three smaller products plus some operations of shifts, additions and substitutions. We applied this formula in order to split the original filter into three filters of reduced dynamic range, working in parallel. The relations given in the equations (7), (8) and (9) describe these sub-filters.

$$y_H(n) = \sum_{k=0}^{T-1} h_H(k).x_H(n-k) \qquad \text{…….(7)}$$

$$y_L(n) = \sum_{k=0}^{T-1} h_L(k).x_L(n-k) \qquad \text{…… (8)}$$

$$dy(n) = \sum_{k=0}^{T-1} dh(k).dx(n-k) \qquad \text{……...(9)}$$

The filter output becomes: $y(n) = (y_H(n).2^N + y_L(n).(2^N + 1) - dy(n).2^N$ ……(10)

The architecture of the three sub-filters is identical with that shown in Fig.13, except the output which is in CS representation. For the output conversion three hardwired shifters and two CLAs have been used. The conventional and Karatsuba architecture implementations are compared in terms of power, area and delay. In particular, the number of registers used for the Karatsuba implementation is $2T(6N+2+3log2T)$, while for the Conventional implementation is $2T(4N+log2T)$. The number of registers is doubled because CS arithmetic is used in both FIR filter implementations. There are significant savings of the proposed Karatsuba based FIR filters in terms of *Area*, *Delay* and *Power* compared to the conventional ones.Specifically an average reduction of about 15% in the delay, 9% in area and 17% in power have been obtained.

**5.2 Performance Comparison for Different FIR Architectures.**

The performance comparison between the Efficient VLSI FIR filter Architectures in terms of the area, delay and power consumption for different filter lengths presented in table. I. Performance comparison shows that the block based transposed FIR filter architecture involves significantly less Area-Delay Product (ADP) and less energy Per Sample (EPS) than the existing block direct-form structure for medium or large filter lengths. Application-specific integrated circuit synthesis [7] result shows that the block structure for block size 4 and filter length 64 involve 42% less ADP and 40% less EPS than the best available FIR filter structure of [8][9] for reconfigurable applications. The RAG-N algorithm based architecture is operating fast for larger coefficients rather than other algorithm as the rate of growth is linear for the algorithm. The power measured is 0.0166mW/MHz, then the total power for the frequency 140 MHz is 2.33 mW.

**Table. I.** Comparison between the Different FIR filter Architectures.

| Reviewed papers-Authors | Filter length (N) | Delay | Area | Power (mW) |
|---|---|---|---|---|
| Mohanthy [2] | 16 | 1.40 ns | 232089$\mu m^2$ | 100.9 |
| | 32 | 1.40 ns | 476503 $\mu m^2$ | 186.4 |
| | 64 | 1.40 ns | 957186 $\mu m^2$ | 366.2 |
| Sakthival [3] | 20 | 117 ps | 43606 $\mu m^2$ | 2.33 |
| L.B.Soares [4] | 40 | 1.03 ns | 57760 $\mu m^2$ | 9.60 |
| E.Kyritsis [5] | 16 | 0.82 ns | 0.145$mm^2$ | 106.2 |
| | 32 | 0.90 ns | 0.283$mm^2$ | 197.1 |

The average energy and area reductions are shown for the approximate FIR filters composed by the truncation adder synthesized for 10 MHz frequency [10]. In the approximate adder approach ETA IV described in [11], states that each adder requires the transistor count 1444 and power consumption is 0.24 mW. The overall area and power consumption is calculated for filter length of 40 as 57760 $\mu m^2$ and 9.6 mW respectively. The Karatsuba based FIR filters also more efficient in terms of *Area*, *Delay* and *Power* compared to the conventional filters.Specifically an average reduction of about 15% in the delay, 9% in area and 17% in power have been obtained.

## VI. Conclusion

In this paper, different types of efficient VLSI architectures for FIR filters are presented with different implementation techniques. Multiple VLSI optimization techniques for the improvement of the FIR filter methods are presented. All these filters are compared for power, delay and area with other common designs and it demonstrated that every technique is most effective for the design constraints of low area, low power and low delay. Some of the filters are used for the reconfigurable applications and some of them are used for the long filter lengths with low complexity. In the future work, it may be proposed a block based FIR Filter architecture in transpose form configuration using RAG-N algorithm to improve the speed, and to minimize area and power. In this algorithm, multiplication process is carried out by adders and shifters instead of MCM technique. Further power and area may be reduced using ETA-IV type approximate adders in RAG-N algorithm rather than conventional adders.

## References


[1]. V. Sandhiya, S. Karthick & M.Valarmathy, "A survey of new reconfigurable Architectures for implementing fir filters With low complexity" *IEEE Int. Conf. on Computer Communication and Informatics (ICCCI -2014), Jan, 2014.*

[2]. B. K. Mohanty & P. K. Meher "A High-Performance FIR Filter Architecture for Fixed and Reconfigurable Applications" *IEEE Trans. VLSI Syst.,Vol. 24 issue.2 ,pp. 444-452, 2016.*

[3]. R. Sakthivel, I. Mishra, Vrushali Jalke & Asmitha Wachaspati " A custom Reconfigurable Power efficient FIR Filter" *IEEE Int. Conf. on circuit , power and Computing Technologies, pp. 1-4, 2016.*

[4]. L. B. Soares, E. A. Ce´sar da Costa & S. Bampi "Design of area and energy-efficient digital CMOS FIR filters with approximate adder circuits*" Analog Integr Circ Sig Process, Springer journal, 2016.*

[5]. E. Kyritsis & K. Pekmestzi "Hardware Efficient Fast FIR Filter Based on Karatsuba Algorithm" *IEEE Int. Conf. on Modern Circuits and systems technologies (MOCAST), pp. 1-4, 2016.*

[6]. C. Eyupoglu, "Performance Analysis of Karatsuba Multiplication Algorithm for Different Bit Lenghts," *Procedia – Social and Behavioral Sciences, vol. 195, pp. 1860-1864, Jul. 2015.*

[7]. R. Mahesh and A. P. Vinod, "New reconfigurable architectures for implementing FIR filters with low complexity," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 29, no. 2, pp. 275–288, Feb. 2010.*

[8]. S. Y. Park and P. K. Meher, "Efficient FPGA and ASIC realizations of a DA-based reconfigurable FIR digital filter," *IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 61, no. 7,* pp. 511–515, Jul. 2014.

[9]. I. Hatai, I. Chakrabarti & Swapna B "An Efficient VLSI Architecture of a Reconfigurable Pulse-Shaping FIR Interpolation Filter for Multistandard DUC" *IEEE Tran. on very large scale integration (VLSI) systems, vol. 23 Issue. 6,* pp. 1150-1154, 2014.

[10]. Gupta, V., Mohapatra, D., Raghunathan, A., & Roy, K. (2013). Low-power digital signal processing using approximate adders. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(1), 124–137.

[11]. Zhu, N., Goh, W. L., Wang, G., & Yeo, K. S. (2010).*Enhanced low-power high-speed adder for error-tolerant application. In SoC Design Conference (ISOCC), 2010 International, Seoul.*