International Journal of Electronics Engineering (ISSN: 0973-7383)
Volume 10 • Issue 2 pp. 597-603 June 2018-Dec 2018 www.csjournals.com

ASIC Design Analysis of FIR Filter Using DistributedArithmeticAlgorithm

[1]Rashmi Bawankar, [2]Rajesh Mehra
[1]ME Scholar, [2] Associate Professor,
[12]ECE, NITTTR, Chandigarh.

Abstract -Thepaper offerseffective distributed arithmetic algorithm (DA) method basedmeant for high rate throughput reconfigurable implementation of FIR digital filters whose filter coefficients always changesin the course of runtime. Conservatively, for reconfigurable Distributed Arithmetic based execution of FIR digital filter, the Look Up Tables are mandatory to be realized in RAM which are priceymeant for ASIC execution. As a consequence, a shared LUT scheme is anticipated to apply the DA calculation. The proposed design has been worked out on 90nm and 180nm technology, for the ASIC implementation. The area as well as power consumption has been decreased whereas the speed has been improved. A 32, 64, 128 tap finite-impulse-response filter with the projectedapplicationyields 2 times more throughput and consumes almost 5 times less area when equated to the best of existing architectures.
Keywords -Finite Impulse Response (FIR) filter,Distributed arithmetic (DA) Algorithm, Optimization.

Introduction

The reconfigurable FIR filters plays a substantial role in several electronics application in today's modern world. The dynamically changing filter coefficients of these reconfigurable FIR filter during the runtime helps to run the devices speedily with high throughput processing abilities. With increase in throughput processing and regularity, there is substantial increase in hardware requirement that is multipliers. Which thus affects cost-effectiveness and area-time efficiency of computation. Distributed Arithmetic algorithm (DA) is a method which is used all over the world so that one can avoid using multipliers to implement sum-of-products computations. It has acquired even more admiration for the large throughput capability as well as regularity which has further resulted into area-time efficient and the cost-effective for computation [1]. DA is frequently used to form efficiently the Multiply-Accumulate Computation circuit (MAC) for various FIR filters and various Digital Signal Processing (DSP) applications. The high computational efficiency is the key benefit of DA [2]. The conservative multipliers are not required as DA allocatesmultiplication and accumulation operations through shifters, lookup-tables (LUT), and the adders. The DA code generation supports for the fixed-point filters designs only [3]. While the data path when using HDL coding, produced for the structural design of DA, it is boosted for the full length precision calculations, even various algorithms are implicated like LMS, Delayed LMS, Block LMS [4,5]. The taps of the values with zero valued coefficients are ignored and size is reduced of the DA-based LUT consequently, which could have increased exponentially with length L+1 for fundamental DA [6].

Distributed Arithmetic

In DA-based realization of a FIR finite impulse response filter, a stream of data input is given over a parallel-serial shift-register which produces a sequential bits. This data is further fed to bit-wide shift-register which aids as a delay, loading the bit serial data sequences. The delay-line is tapped to form the address which directories into a lookup-table (LUT) [7]. The look-up (LUT) saves the partial products sums over the filter coefficients space. A shift and adder follow the LUT. This logic sequentially do the addition of the values obtained from look-up table (LUT). A lookup table is performed sequentially for each and every bit and on all clock cycle, LUT outcome is added to the shifted and accumulated result from the previous cycle [8].
This is the simple form of Distributed Arithmetic which is serial in nature, functioning one bit at a time only. The filter clock cycle number to generate the output relays upon the bitwise length of input stream data. If data input stream is X bits in length, then the FIR Filter takes in sequence X clock cycles to calculate the output. FIR Filters having the Symmetric and asymmetric structures are exceptional cases, as it requires X+1 clock cycles, one extra clock cycle is essential to route carry bit of pre-adders [9].
The fundamentally bit serial environment of DA limits the throughput. In a way to improve throughput, the simple DA algorithm is modified and it computes additional one bitwise sum at a time. The amount of instantaneously calculated bit

wise sums is stated as a twos power termed the Distributed Arithmetic radix. As an example, a DA-based radix of 8(2^3) designates that a three bit wise sum is calculated at the similar time, and carry forwarded, for examples one dimensional and two dimensional array structures [10]. To compute additional one bit wise sum at the equivalent time, the coder replicates the lookup tables. The DA to perform on two bits consequently (radix 4), the bits which are odd are given to LUT and the bits which are even are at the equivalent time fed to an alike LUT. The LUT results of bits which are odd are left-shifted beforehand to sum up to the lookup table results of bits which are even. This output is further fed to an accumulator which shifts the feedback data by two places. Handling additional one bit at same time acquaint with parallelism into the action, which improves the performance [11].

The size of lookup table grows with increasing filter order. For the high-order filters, size of lookup table (LUT) is decreased to considerable amount. In order to decrease the size, the LUT is split up into several LUTs, known as LUT partitions or the slices [12]. These each of the lookup table partition functions on a diverse taps set and the output result acquired through the partitions are added.

For the linear time-invariant networks the sum-of-products states the output response as:

$$y(n) = {}_{i=1}^{i} A_i x_i\,(n) \qquad (1)$$

where:
y(n) is the network-response at a time 'n',
$x_i(n)$ is the ith input at a time 'n',
A_i is the weight factor of ith input-variable which remains time-invariant [13,14].

In the applications where filter is used the constants, A_i, are the coefficients of filter, while x_i, are variables which are previous sample of data from the source. The severe multiplication nature of above equation 1 is valued by observing that an output response of single data requires accretion of product terms. While in Distributed Arithmetic the process of addition of the product terms is carried through look-up tables which is implemented in configurable logic blocks [15].
The constant factors, Ai, are not constrained they don't even requires to be matched with the data word length, on the additional hand the constants might be having a mixed integer or even fractional set-up. The variable, xi, can be written in fractional form as:

$$x_i = -x_{i0} + {}_{b=1}^{B-1} x_{ib}\, 2^{-b} (2)$$

Where,
x_{ib} is binary variable and takes the value of 0 or 1.
A x_{i0} indicates signed value of -1. Substituting equation (2) into equation (1) the result obtained is-

$$y = {}_{i=1}^{I} A_i\, [-x_{i0} + {}_{b=1}^{B-1} x_{ib}\, 2^{-b}] \,/\, A_i 2^{-b}$$

$$y = {}_{i=1}^{I} x_{i0} A_i + {}_{i=1}^{I} {}_{b=1}^{(B-1)} x_{ib} (3)$$

The word in the square bracket indicates a multiplying operation including a bit of input variable as well as all other bits of constant. The '+' sign signify summation. Scaled contributions of square bracket pairs to total sum is signified by exponential factors [16]. Distributed Arithmetic FIR filter computes the filter action in H clock cycles irrespective of the filter tap size, K. So, the large throughput rate is acquired using the Distributed Arithmetic execution, especially if K >> B, where K is tap of filter. If the filter tap increases the memory need increases exponentially, at the rate of 2^K [18].

The same Distributed Arithmetic Table (DA-LUT) may be shared in time form in series computation or may be imitated several intervals for parallel calculation pattern. If consecutive summation of all the DA-LUTs outcomes is essential then the array of parallel-adders is required. While the multiple parallel addition operation of DA-LUT outputs given as in equation (4) is only performed with help of single adder that also if the adder is time-shared [17].
Henceforth the DA lookup table or DALUT is shown as:

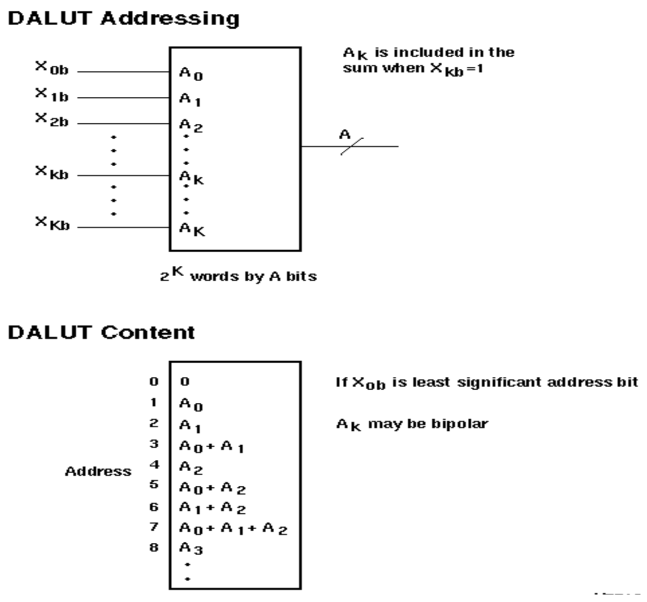

Figure 1.Look-up Table of Distributed Arithmetic [3]

As presented in figure 2, the structure depicts the input data samples x are given to the serial-in-parallel-out shift register at every instant of clock cycle. The size of this serial-in-parallel-out shift register is supposed to be N, it thus decomposes N newest samples to P which is of interval M, where p = 0, 1,2,3,4,……….., P-1. This is then fed to reconfigurable partial pro-generators which computes the partial products. These computed partial products is added through pipeline adder tree and further shifted as well as added to get the final outcome y [18].

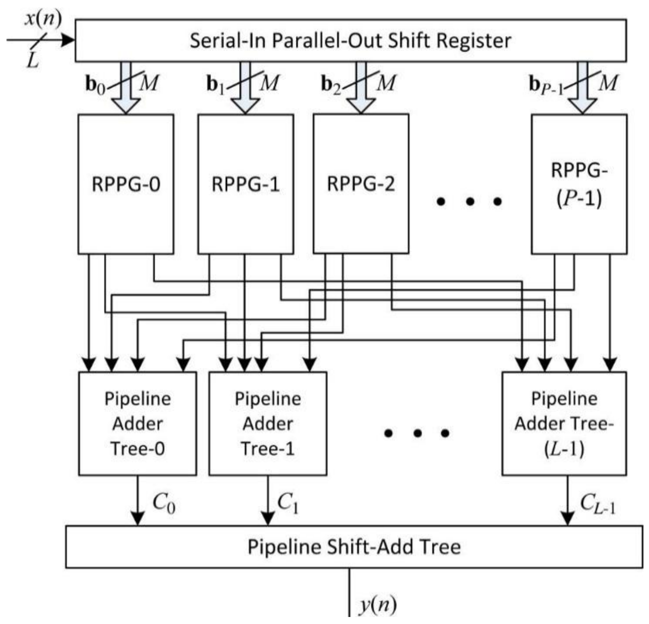

Figure 2. Reconfigurable FIR Filter ASIC Design using DA [1].

The reconfigurable partial pro generator blocks consists of parallel LUTs to implement the FIR filter. The LUTs are shared through L bit slices as well as array register is preferred over memory based LUT so that they can be accessed simultaneously, thus they can be updated in less cycles [19]. The inputs of MUX as shown figure 3 are given as 0, h(2p), h(2p+1) and h(2p) + h(2p+1), while 2-bit digit bi, is given to MUX l for 0 ≤ l ≤ L-1 as the control word. This MUX then gives the partial product Si, for 0 ≤ l ≤ L-1.

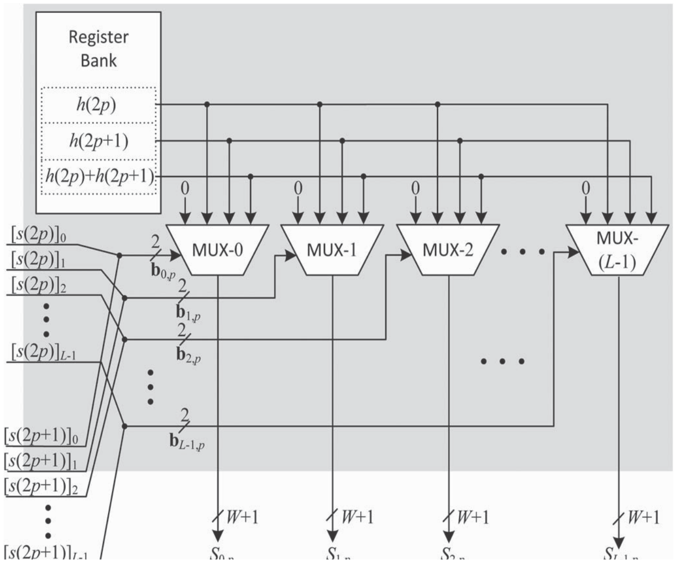

Figure3.Reconfigurable partial product generator (RPPG) block [1].

Hence, partial product generator is decomposed into U parallel slices as well as each slice has its own V time multiplexed functions [20, 21, 22]. Where L is given by L = UV, U and V being positive integers.

Proposed DA Based Filter

The anticipatedconfiguration of the Distributed Arithmetic constituted FIR digital filter meant for ASIC application is as depicted in Fig. 2. The ingoingtasters {x(n)} arriving at each sampling instant are provided to a serial-in–parallel-out shift register having size N. The SIPOSR crumbles the N latest most samples to P vectors bp of length M for p = 0,1,2,3,4,……....,P−1 and provides these to P reconfigurable partial product generators to analyze the partial products conferring to [23, 24]. The arrangement of the projected RPPG is depicted in Fig. 3 for M = 2. For high-throughput execution, the RPPG generates L partial products conforming to L bit slices in parallel by using the LUT poised of a very single register bank of 2M − 1 registers and L number of 2M : 1 MUXes [25].

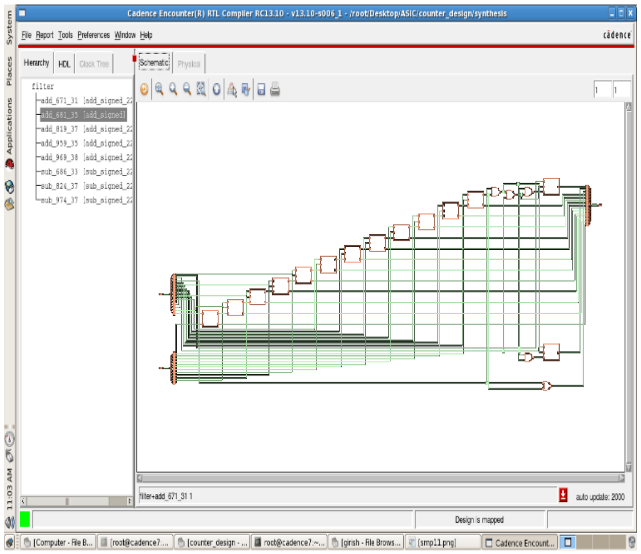
Cadence Encounter(R) RTL Compiler RC13.10 - v13.10-s006_1 - /root/Desktop/ASIC/counter_design/synthesis
File Report Tools Preferences Window Help
Hierarchy
HDL
Clock Tree
Schematic
Physical
filter
Design is mapped

Figure 4.Proposed structure for FIR filter based on DA scheme

In the projectedassembly, the reduced storingingestionby means of involving each and every Look up table across L bit slices is given. The register array is being chosen for this veryperseverance rather than memory-based look up table so that to access the look up table contents at once. In addition, the contents in the register-based look up table is able to be update in parallel in lesser cycles than the memory-based look up table to executeanticipated FIR digital filter [26, 27, 28]. The width of each and every register with the look up table is $(W + \lceil \log_2 M \rceil)$ bits, where W is the word-length of the filter coefficient. The input of the MUXes are 0, $h(2p)$, $h(2p + 1)$, and $h(2p) + h(2p + 1)$; and the two-bit digit $\mathbf{b}_{l,p}$ is fed to MUX l for $0 \leq l \leq L - 1$ as a control word. We can find that MUX l provides the partial product $S_{l,p}$ for $0 \leq l \leq L - 1$ given by [24].

Results And Discussions

The projected structure is shown for ASIC implementation assures a very high throughput compared to the other structures for $R < L$ as well few adders and small LUT that to the systolic structure. The structure presented involves lesser adders as well as registers, but slightly larger look up table, compared tovery other structure. However, on behalf ofASICexecution, the DRAM-based look up tableneeds fewquantity of slices (QOS) as compared to the register-based look up table of the identical size. It have to be illustrious that, as M increases, the size of the look up table increases exponentially, however the total number of adders declines. The silicon areas of the assemblies in Fig.5 are proportional to the value of P, and the filters will have much benefit due to more partaking of the LUT when L has a huge value.

TABLE 1 :Performance Comparison For L=W =32, M =64

Parameter	**90nm**		**180nm**	
Filter Length	**32**	**64**	**32**	**64**
Area (sq.um)	12756	13563	12418	13225
Power (nW)	11.33	15.71	15.51	23.52
Speed (ns)	1.202	1.1995	1.2008	1.1994

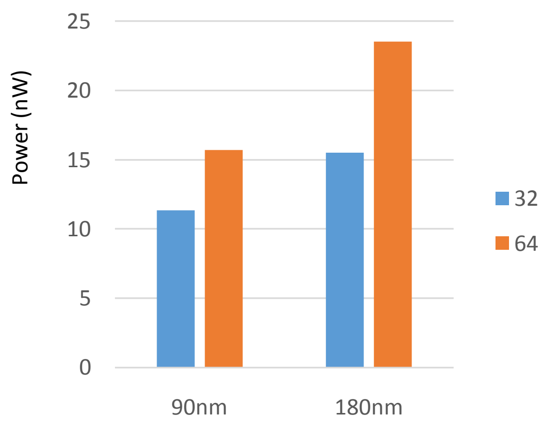

Figure 5. Area Comparison for 90 and 180nm technology

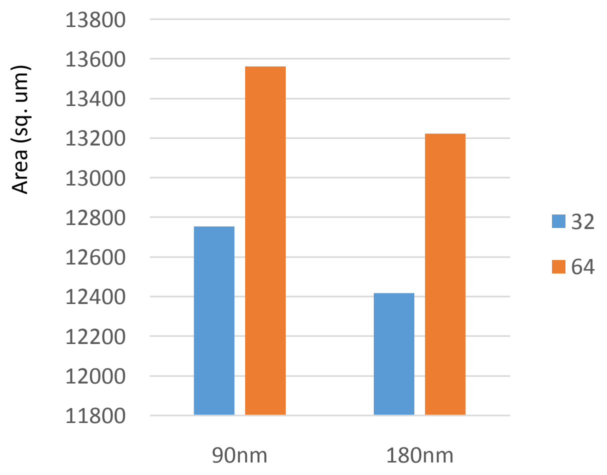

Figure 6.Power Comparison for 90 and 180nm technology

Conclusion

In this brief, an effectual structureaimed at high-throughput reconfigurable Distributed Arithmetic based ASICapplication of FIR digital filtersis presented. It has predicted that the structure hardware cost possibly will be concentrated by partaking the similar registers by the Distributed Arithmetic Algorithm units for various bit slices. The projectedplan has almostless area along with power consumption for the ASIC implementation. The design will able to be adopted to any order filters and is predominantlybeneficial for filters of higher order as well as large base units. The percentage improvement in speed is 2 times, while power consumption is 5.4 times less and even area is decreased.

References

[1] S. Y. Park, P. K. Meher, (2014) "Efficient FPGA and ASIC Realizations of a DA-Based Reconfigurable FIR Digital Filter", IEEE Transactions on Circuits and Systems-II, Express Briefs, Vol. 61, No. 7, pp. 511-515.
[2] M. Surya Prakash, R. A. Shaik, (2013) " Low-Area and High-Throughput Architecture for an Adaptive Filter Using Distributed Arithmetic," IEEE Transactions on Circuits and Systems-II, Express Briefs, Vol. 60, No. 11, pp. 781-785.
[3] S. Y. Park, P. K. Meher, (2013) "Low Power, High-throughput, and Low Area Adaptive FIR Filter Based on Distributed Arithmetic", IEEE Transaction on Circuits and Systems-II, Express Briefs, Vol. 60, No. 6, pp. 346-350.
[4] B. K. Mohanty, P. K. Meher, (2013) "A High-Performance Energy-Efficient Architecture for FIR Adaptive Filter Based on New Distributed Arithmetic Formulation of Block LMS Algorithm," IEEE Transactions on Signal Processing, Vol. 61, No. 4, pp. 921-932.
[5] Mohd. Tasleem Khan, and Rafi Ahamed Shaik, (2018) "Optimal Complexity Architectures for Pipelined Distributed Arithmetic-Based LMS Adaptive Filter,"IEEE Transactions On Circuits And Systems–I: Regular Papers, pp. 1-13.
[6] R. Guo and L. S. DeBrunner, (2011) "Two High-Performance Adaptive Filter Implementation Schemes using Distributed Arithmetic," IEEE Transaction Circuits System II, Express Briefs, Vol. 58, No. 9, pp. 600-604.
[7] Priyanka Nain, Dr. G.S. Virdi, (2017) " Look-Up Tablle based ScalliingAccumullatoriin MAC usiing VHDL," International Journal of Electronics Engineering, Vol. 9, No. 2, pp. 91-99.
[8] Honglan Jiang, Leibo Liu, Pieter P. Jonker, Duncan G. Elliott, Fabrizio Lombardi, and Jie Han, (2018) "A High-Performance and Energy-Efficient FIR Adaptive Filter Using Approximate Distributed Arithmetic Circuits,"IEEE Transactions On Circuits And Systems–I: Regular Papers, pp. 1-13.
[9] S. Baghel and R. Shaik, (2011) "FPGA Implementation of Fast Block LMS Adaptive Filter using Distributed Arithmetic for High-Throughput," IEEE International Conference Communication Signal Processing, pp. 443–447.
[10] P. K. Meher, S. Chandrasekaran, and A. Amira, (2008) "FPGA Realization of FIR Filters by Efficient and Flexible Systolization using Distributed Arithmetic," IEEE Transaction Signal Processing, Vol. 56, No. 7, pp. 3009–3017.
[11] S. Baghel and R. Shaik, (2011) "Low Power and Less Complex Implementation of Fast Block LMS Adaptive Filter using Distributed Arithmetic," IEEE Students Technology Symposium, pp. 214–219.
[12] D. J. Allred, H. Yoo, V. Krishnan, W. Huang and D. V. Anderson, (2005) "LMS Adaptive Filters using Distributed Arithmetic for High Throughput," IEEE Transaction Circuits System, Vol. 52, No. 7, pp. 1327–1337.
[13] M. S. Prakash and R. A. Shaik, (2016) "DA Based Approach for the Implementation of Block Adaptive Decision Feedback Equalizer," IET Signal Processing, Vol.10, No.6, pp. 676-684.

[14] SuganthiVenkatachalam and Seok-Bum Ko, (2018) "Approximate Sum-of-Products Designs Based on Distributed Arithmetic," IEEE Transactions On Very Large Scale Integration (VLSI) Systems, pp.1-5.
[15] A. Nanda, T. Vigneshwaran and Ashwani K. Rana, (2014) "DA-Based Efficient Testable FIR Filter Implementation on FPGA Using Reversible Logic," Springer- Circuit, Systems, and Signal Processing, Vol. 33, No. 3, pp. 863-884.
[16] P. K. Meher and S. Y. Park, (2011) "High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic," IEEE, 19th International Conference VLSI-SOC, pp. 428–433.
[17] R. Guo and L. S. DeBrunner, (2011) "A Novel Adaptive Filter Implementation Scheme using Distributed Arithmetic," Forty Fifth Asilomar Conference on Signals, System, Computers, pp. 160–164.
[18] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, (2004) "A Novel High Performance Distributed Arithmetic Adaptive Filter Implementation on an FPGA," IEEE International Conference Acoustic, Speech, Signal Processing, Vol. 5, pp. 161-164.
[19] S. A. White, (1989) "Applications of Distributed Arithmetic to Digital Signal Processing: A tutorial review," IEEE Magazine, Vol. 6, No. 3, pp. 4–19.
[20] T. Hentschel, M. Henker, and G. Fettweis, (1999) "The Digital Front-end of Software Radio Terminals," IEEE Personal Communication Magazine Vol. 6, No. 4, pp. 40–46.
[21] L. Ming and Y. Chao, (2012) "The Multiplexed Structure of Multi-channel FIR Filter and Its Resources Evaluation," International Conference on Computer Distributed Control and Intelligent Environment Monitoring, pp. 764–768.
[22] I. Hatai, I. Chakrabarti, and S. Banerjee, (2013) "Reconfigurable Architecture of a RRC FIR Interpolator for Multi-standard Digital Up Converter," IEEE 27th International Symposium on Parallel & Distributed Processing, Workshops and PhD Forum, pp. 247–251,.
[23] P. K. Meher, (2006) "Hardware-efficient Systolization of DA-based Calculation of Finite Digital Convolution," IEEE Transaction Circuits System II, Express Briefs, Vol. 53, No. 8, pp. 707–711.
[24] E. Ozalevli, W. Huang, P. E. Hasler, and D. V. Anderson, (2008) "A Reconfigurable Mixed-signal VLSI Implementation of Distributed Arithmetic used for Finite-impulse Response Filtering," IEEE Transaction Circuits System I, Registered Papers, Vol. 55, No. 2, pp. 510–521.
[25] Rajesh Mehra, Lajwanti Singh, (2013) "FPGA based Speed Efficient Decimator using Distributed Arithmetic Algorithm," International Journal of Computer Applications, Vol. 80, No. 11, pp. 37-40.
[26] Rajesh Mehra, Swapna Devi, (2010) "FPGA Based Design of High Performance Decimator using DALUT Algorithm," ACEEE International Journal on Signal and Image Processing, Vol. 1, No. 2, pp. 9-13.
[27] K. H. Chen and T.-D. Chiueh, (2006) "A Low-power Digit-based Reconfigurable FIR Filter," IEEE Transaction Circuits System II, Express Briefs, Vol. 53, No. 8, pp. 617–621.
[28] M. Tasleem Khan, Dr. Shaik Rafi Ahamed, (2018) "Area and Power Efficient VLSI Architecture of Distributed Arithmetic Based LMS Adaptive Filter," 31th International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems, IEEE, pp. 283-288.