International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal | www.ijtsrd.com
ISSN No: 2456 - 6470 | Volume - 3 | Issue – 1 | Nov – Dec 2018

FPGA Based Hybrid LMS Algorithm Design on Distributed Arithmetic

Yamuna P[1], B K Venu Gopal[2]
[1]PG.Scholar, [2]Associate Professor
Department of Electronics and Communication Engineering,
University Visvesvaraya College of Engineering, Bengaluru, Karnataka, India

ABSTRACT
Filter plays a major role for removal of unwanted signal/noise from the original signal, especially Adaptive FIR filter is easy to attract for many applications, where it is need to minimize computational requirement. This paper shows a novel pipelined design for low-power, high-throughput, and low-area implementation of adaptive filter based on distributed arithmetic (DA). The DA formulation utilized for two separate blocks weight update block and filtering operations requires larger area and is n't suited for higher order filters therefore causes reduction in the throughput. These issues have been overcome by efficient distributed formulation of Adaptive filters. LMS adaptation performed on a sample by sample basis is replaced by a dynamic LUT update by the weight update scheme. Adder based shift accumulation for inner product computation replaced by conditional signed carry-save accumulation to reduces the sampling period and area density.

KEY WORDS: *Distributed Arithmetic, LMS Adaptive Filter, VHDL, DA, Adaptive Filter, LMS algorithm, Carrysave adder, Noise Cancellation, Digital Signal Processing*

1. INTRODUCTION
Adaptive digital filters find wide application in few advanced digital signal processing (DSP) areas, e.g., *noise and echo cancellation, system identification, channel estimation, channel equalization and so forth* [1]. The tapped-delay-line finite-impulse-response(FIR) filter whose weights are updated by the celebrated "*Widrow Hoff*" least-mean-square (LMS) algorithm [2] might be considered as the most straight forward known adaptive filter. The LMS adaptive filter is prominent because of its low-complexity, as well as due to its stability and attractive convergence performance [3]. because of its few essential use of current relevance and expanding limitations on area, time, and power complexity, efficient implementation of the LMS adaptive filter is still very vital. To implement the LMS algorithm, one needs to update the filter weights during each sampling period using the estimated error, which equals the difference between the current filter output and the desired response. The new weights of the LMS adaptive filter are refreshed. Then µ convergence factor or step-size, which is usually assumed to be a positive number and Nis the quantity of weights utilized in the LMS adaptive filter. The pipelining structure is proposed over the time consuming combinational blocks of the structure. The normal LMS algorithm does not support pipelined implementation due to its recursive behavior. It is adjusted to a form called the delayed LMS (DLMS) algorithm. This DLMS utilize an altered systolic architecture to reduce the adaptation delay where they have utilized moderately extensive processing elements (PEs) for accomplishing a lower adaptation delay with the basic way of one MAC task. The direct form configuration on the forward path of the FIR filter subsequently prompts a long critical path because an inner product computation to acquire a filter output. subsequently, when the input signal is having a high sampling rate, it becomes crucial to reduce the critical path of the structure with the goal that it could not surpass the sampling period [4]. Distributed Arithmetic based technique comprises of a structure which is stay away from of multiplier that expands the throughput.

2. FIR FILTER
The FIR filter is a circuit that filters a digital signal (samples of numbers) and provides output that is

another digital signal with attributes that are reliant on the response of the filter. The FIR filter is non – recursive. It utilizes a finite duration of none zero input values and produces a finite duration of output values which are non-zero.

These days, Field Programmable Gate Array (FPGA) technology is widely utilized in digital signal processing area of the fact that FPGA-based solution can accomplish high speeds owing to its parallel structure and configurable logic, thus providing great amount of flexibility and high reliability throughout design process and later maintenance. A number of architectures have been reported in the writing for memory-based implementation of DSP algorithms which includes orthogonal transforms and digital filters [6].

FIR filters use addition to calculate their outputs simply like averaging does. The primitive elements utilized in the outline of a FIR filter are delays, multipliers and adders. The FIR filter comprises of a series of delays, multiplications and additions to create the time domain output response. The impulse response of the FIR filter is the multiplication coefficients utilized. The phase of a FIR filter is linear which is its dominant feature. Filter is implemented sequentially utilizing a multiplier and an adder with a feedback which is outlined in the high level schematic in Fig. 1[11].

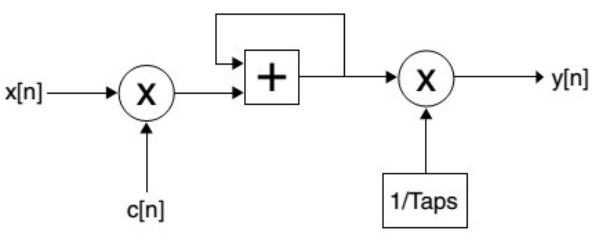

Fig.1: FIR Filter

The LMS algorithm updates the coefficient in an iterative way and feeds to the FIR filter. The FIR filter at that point utilizes this coefficient c(n) in addition to the input reference signal x(n) to generate the output response y(n). The output y(n) is then subtracted from the desired signal d(n) to produce an error signal e(n), or, in other words by the LMS algorithm to compute the next set of coefficients.

3. ADAPTIVE FIR FILTER

The idea behind a closed loop adaptive filter is that a variable filter is balanced until the error (the difference between the filter output and the desired signal) is minimized. Most adaptive filters are dynamic filters which change their attributes to accomplish desired output. Adaptive filter has various parameters which should be changed in order to maintain optimal filtering. Adaptive filtering has two types of algorithms one is Least *Mean0Square* (LMS) and the other *Recursive0Least Squares* (RLS) optimal filtering. one of the applications of adaptive filter is noise cancellation.

Adaptive filter has two vital properties: first, it can work successfully in unknown environment; second, it is used to track the input signal of time-varying characteristics [7]. Due to DSP has serial architecture, it can't process high sampling rate applications and used only for amazingly complex math-intensive tasks. An example of adaptive filter system is shown in Fig. 3. The objective of the adaptive algorithm is to modify the coefficient of the adaptive filter, w[n], with the end goal to limit the error term e[n] in the mean-squared sense. The adaptive algorithm basically distinguishes a vector of coefficient w[n] that minimizes the following quadratic equation,

$$\varepsilon[n] = E\{|e[n]|2 \qquad (3.1)$$

where e[n] = d[n] □y[n], d[n] is the output from the unknown system (desired signal), y[n] is the output from the adaptive filter, the expected value, and _[n] is the mean squared error (MSE). Many methods exist to solve Eqn. 3.1,-0- the most widely recognized being the strategy for steepest descent. This procedure proceeds until the adaptive algorithm achieves steady state, where the concession between the current solution vector and the optimal solution, or MSE, is at its minimum. In general, adaptive algorithms can be separated into four steps that are performed consecutively: filtering, computing the error, calculating the coefficient updates, and updating the coefficients. When split into this form, the essential deference prediction and interference canceling.

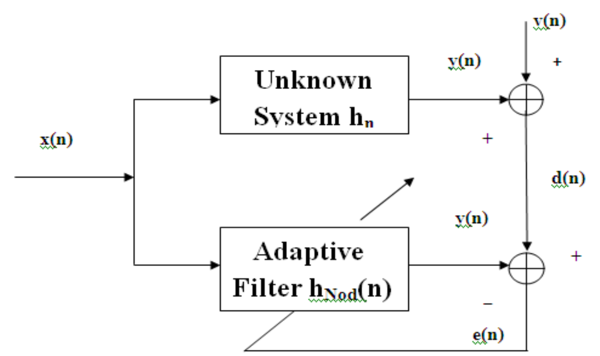

Fig 3 Unknown and adaptive filter have different length

Adaptive filtering has two basic operations: the *filtering process* and the *adaptation process*. A digital filter used to generate output signal from an input signal by the filtering process, whereas the adaptation process consists of an algorithm which minimize a desired cost function by adjusting the coefficients of the filter.

4. LMS ADAPTIVE FILTER (EXISTING DESIGN)

LMS algorithm is introduced. LMS remains for *Least-Mean-Square*. This algorithm was created by “*Bernard Widrow*” during the 1960’s, and is the first0generally used adaptive algorithm. It is still generally utilized in adaptive digital signal processing and adaptive antenna arrays, primarily because of its simplicity, ease of implementation and great convergence properties.

The objective of the LMS algorithm is to deliver the *mean square error0* for the given condition and weights that minimize the mean-squared error between a desired signal and the arrays output, loosely speaking, it attempts to maximize reception towards the desired signal (who or what the array is trying to communicate with) and limit reception from the interfering or undesirable signals in the Fig 4

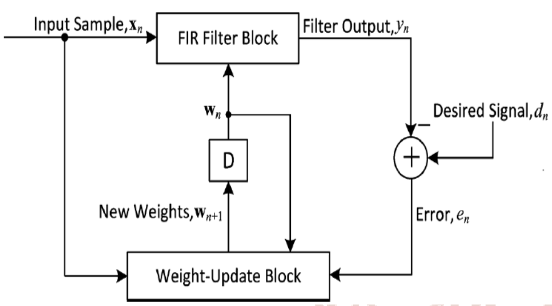

Fig 4 Conventional LMS adaptive filter

some data is required before optimal weights can be resolved. The weights of LMS adaptive filter during the nth emphasis are updated according to the following equation. The updated weight is

wn+1= wn+ µenxn0

en = dn – yn0

yn = wTnx0

Where en is the error, dn is the desired response during nth iteration calculated, µis convergence factor or step size. The weight vector Wn and input vector xn is given by

$$\mathbf{x}_n = [x_n, x_{n-1}, \ldots, x_{n-N+1}]^T$$
$$\mathbf{w}_n = [w_n(0), w_n(1), \ldots, w_n(N-1)]^T$$

The number of weights used in the LMS adaptive filter is denoted as N. It is outstanding that the LMS algorithm has a moderate convergence for correlated inputs. Another circumstance that can show up in identification applications is when the coefficients of the model are time-varying. The adaptive algorithm should give a mechanism to track the changes of the model. Fig 4 in the LMS algorithm described has a very low computational0complexity (number of additions, subtractions, divisions, multiplications per iteration) and memory load, which makes it exceptionally interesting for practical implementations. It is notable that the step-size µ influences the performances of the adaptive filter. In spite of its low complexity, the LMS has additionally some drawback.

5. DISTRIBUTED ARITHMETIC (DA)

Distributed arithmetic (DA) is ordinarily utilized for signal processing algorithms where figuring the inner product of two vectors involves majority of the computational work load. This sort of computing profile depicts a huge part of signal processing algorithms, so the potential usage of distributed arithmetic is huge. The inner product is usually processed utilizing multipliers and adders. Distributed0Arithmetic, alongside Modulo Arithmetic, are computation algorithms that perform multiplication with look-up table based plans. Both blended some enthusiasm more than two decades prior however have mulled from that point onward. In fact, DA particularly focuses on the sum of products (sometimes referred to as the vector dot product) computation that spread a large number of the vital DSP filtering and frequency transforming functions. XILINX FPGA look up table are not familiar for DSP users and DA algorithm popularly used to produce filter designs. The introduction of the DA algorithm is to a great degree straightforward yet its applications are to a great degree wide. The science incorporates a blend of Boolean and ordinary algebra and require no prior preparation - even for the logic designer.

DA fundamentally diminishes the quantity of additions required for filtering. This decrease is especially detectable for filters with high piece accuracy. This reduction in the computational remaining burden is a result of storing the pre-computed partial sums of the filter co-efficient in the memory table. Distributed arithmetic needs fewer arithmetic computing resources and no0multipliers, when compared with other alternatives.

This part of distributed arithmetic is a good one for computing environments with limited computational resources, especially multipliers. These kind of computing environments can be found on older field-programmable gate arrays (FPGAs) and low-end, ease FPGAs.

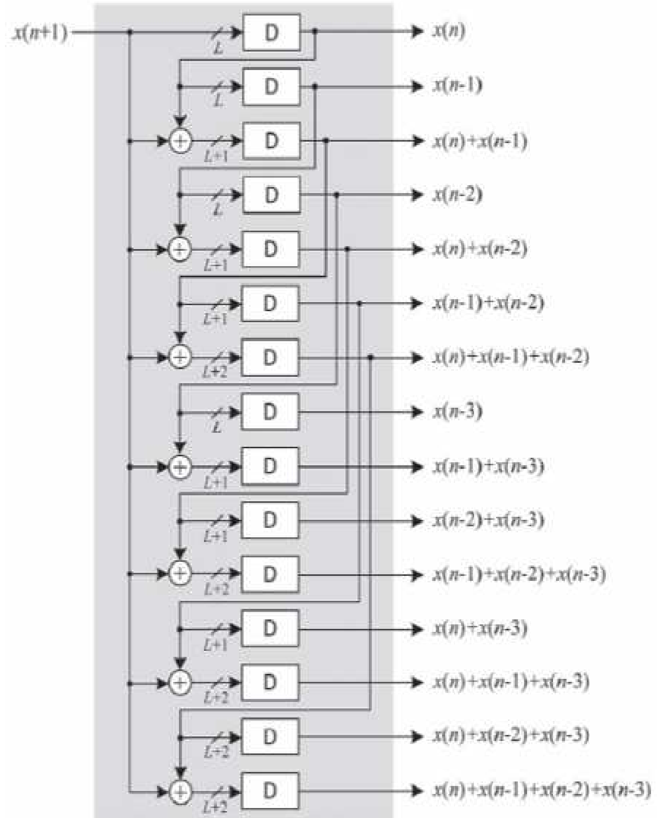

Fig 5 Distributed arithmetic

"Note that cK for{0 ≤ k ≤ 2N – 1} can be pre-registered and put away in RAM-based LUT of 2N words. Notwithstanding, rather than putting away 2N words in LUT, store (2N − 1) words in a DA table of 2N − 1 registers. A case of such a DA table for N = 4 is appeared in Fig 5. It contains just 15 registers to store the pre computed sums of input words. Seven new estimations of ck0 are computed by seven adders in parallel Fig 5[11]. AK is the incorporate into Sum when Xkb =1 that is the least significant bit. The substance of the kth LUT area can be identified as kj. where kj is the (j + 1)th bit of N-bit binary representation of integer k for 0 ≤ k ≤ 2N − 1. A 4-tap (K = 4) execution of the DA filter is appeared in Figure 3.1. The contains all 16 possible combination sums of the filter weights w0;w1;w2 and w3.The bank of shift registers in Fig5[11] stores 4 consecutive input samples (x[n - i]; i =0…0……..3). The connection of rightmost bits of the shift registers becomes the address of the memory table". At every clock cycle the shift register is shifted right.

6. DISTRIBUTED ARITHMETIC BASED APPROACH (PROPOSED DESIGN)

The essential block diagram for the proposed system for the design of an Adaptive FIR Filter using Distributive Arithmetic is proposed *DA-Based LMS Adaptive Filter* The proposed structure of DA-based adaptive filter of length N = 4 is appeared in Fig 6.1[11]. It comprises of a four-point inner product block and a weight-increment block along with additional circuits for the calculation of error value e(n) and control word t for the barrel shifters. On relating with existing DA based adaptive filter0implementations the proposed architecture has achieved low complexity.

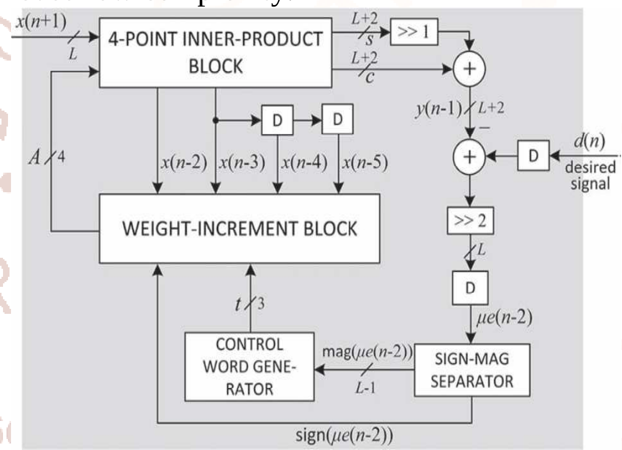

Fig 6.1 DA-based LMS adaptive filter of filter length N

For accomplishing this preferred standpoint Offset Binary Coding (OBC) has been utilized in this filter design -based LMS adaptive filter design requires 16 delay element DA-table structure.

6.1. FOUR POINT INNER-PRODUCT BLOCK

The underlying part of the filtering process of the LMS adaptive filter, for each cycle, is the need to perform an inner product calculation. Distributed Arithmetic (DA) procedure is bit-sequential in nature. It is entirely a bit-level modification of the multiply and accumulation operation. The fundamental DA is a computational algorithm that affords effective usage of the weighted sum of0products, or dot product. The four-point inner-product block shown in Fig.6.2 incorporates a DA table comprising of an array of 15 registers which stores the partial inner products yl for {0 < l ≤ 15} and a 16: 1 multiplexor (MUX) to choose the substance of one of those registers. Bit cuts of weights A = {w3l w2l w1l w0l} for 0 ≤ l ≤ L − 1 are fed to the MUX as control in LSB-to- MSB order, and the carry Save accumulator is fed by the output of the MUX.

After L bit cycles, the carry-save accumulator shift accumulates all the partial inner products and produces a sum word and a carry word of size (L + 2) bit each. The carry and sum words are shifted added with an input carry "1" to generate filter output which is consequently subtracted from the desired output d(n) to acquire the error e(n).

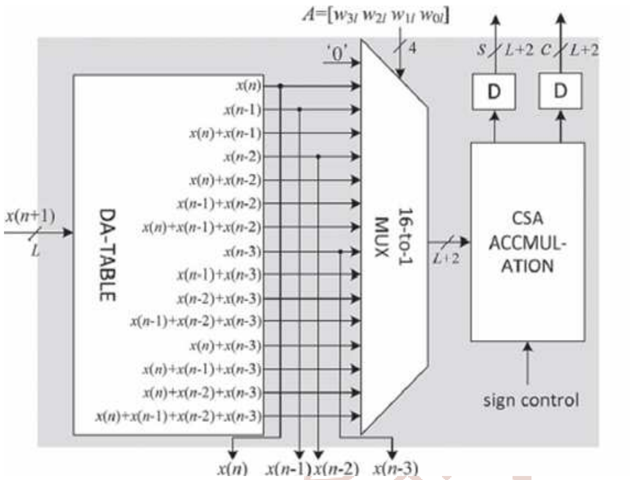

Fig 6.2 Four-point inner-product block

Each one the bits of the error except the MSB bit are ignored, such that multiplication of input *xk* by the error is executed by a right shift, in the magnitude of the error the number of locations are given by the number of leading zeros. "To generate the control word t for the barrel shifter the magnitude of the computed error is decoded by the logic. The convergence factor μ is taken as0 (1/N). also we use μ as 2−i/N, where i is a small integer". The number of shifts t in that case is increased by i, and the input to the barrel shifters is pre shifted by i locations accordingly to reduce the hardware complexity. "DA used to compute the inner0(dot) product of a constant coefficient vector and a variable input vector in a single bit serial operation". This is the task that contributes to most of the critical path. Let the inner product be thought to be

y(n)=WqTn X(n) , e(n)=d(n)-y(n)

where the weight vector w(n) and input vector x(n) at the nth iteration are respectively given by

*{X(n)=[x(n),x(n-1),…….x(n-N+W(n)=[w0 (n), w1 (n),….,}*the corresponding current value in the weight register.

6.2 THE WEIGHT INCREMENT UNIT

It comprises of four barrel shifters and four adder/sub tractor cells in the Fig.6.3[11]. "The barrel shifter shifts the distinctive input values xk for {k =0, 1, . . ., N − 1} determined by the location of the most significant one in the estimated error by appropriate number of locations. The barrel shifter decides, added with or subtracted from the current weights.

Realize the corresponding multiplication of a shift operation. To update n weights, the weight-update block consists of n carry-save units. Each carry-save units performs the multiplication of shifted error values with the delayed input samples along with the addition with the old weights. To the calculation of weight-increment term the addition of old weight with weight increment term is merged with multiplication.

To control for adder or subtract cells the sign bit of the error is used such that, when sign bit is zero the barrel-shifter output added and when one it is subtracted from the content of the corresponding current value in the weight register [11]. Multiplication is performed by each MAC unit, of the shifted value of error with the delayed input samples xn-m followed by the additions.

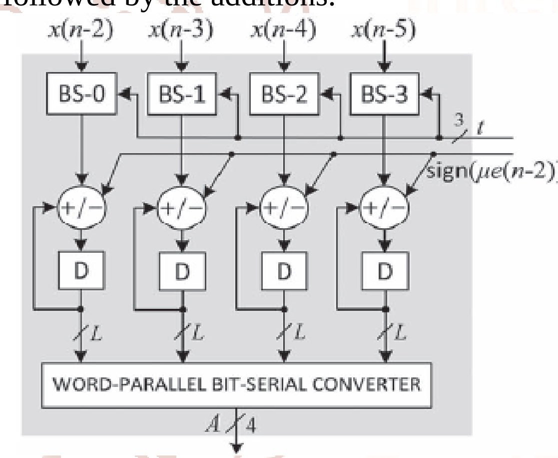

Fig 6.3 the weight increment unit

The final outputs of the carry-save units constitute the updated weights which serve as an input to the error computation block as well as the weight-update block for the next iterations. To keep the critical-path equal to one addition time a pair of delays are introduced before and after the final addition of the weight-update block.

7. SIMULATION AND RESULTS

Simulation and synthesis is performed by XILINX software using VHDL programming. Here it very well may be seen that the power consumption has reduced to just below half of that in the existing design. This has resulted because of a reduced switching activity of the design based on carry-save adder. An efficient pipelined architecture for, and low delay implementation of DA-based adaptive filter.

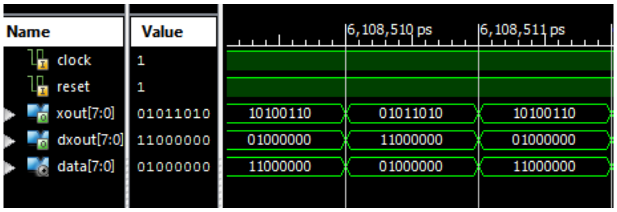

Fig 7.1 Simulation output for DA based weighted LMS algorithm

The simulation results in the fig.7.1 and fig.7.2 describes the DA based weighted LMS algorithm output and filtered, shifted outputs of adaptive LMS filter algorithm respectively.

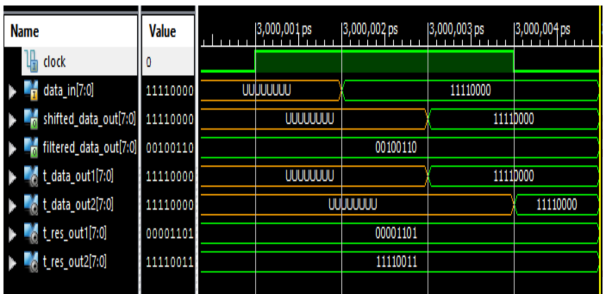

Fig 7.2 Simulation output for adaptive LMS based filter algorithm

A carry-save accumulation scheme of signed partial inner products for the computation of filter output has been implemented. From the synthesis results, it was found that the proposed design consumes less power over our previous DA based FIR adaptive filter. In future, work can be implemented on digital communication, signal processing application, digital radio receivers, software radio receivers and echo cancellation.

DESIGN	Gate Count	Delay (nS)	Power (mW)
EXISTING	31,500	2.3	91
PROPOSED	40,000	1.70	67

Table 1: Result Comparison of Existing and Proposed design in terms of Area, Delay and Power

8. CONCLUSION

This paper presents significantly less adaptation delay and provided significant saving of area and power. In proposed we are going to deal with direct form LMS algorithm along with transposed form of delayed least mean square algorithm (TDLMS), where the power is significantly reduced and the performance is increased along with decrease in area. From the synthesis results, it was found that the proposed design consumes less power over our previous DA-based FIR adaptive filter.

9. ACKNOWLEDGMENTS

Thanks for the base idea paper by Dr. S. A. Whité, "*Applications of the distributed arithmetic to digital signal processing: A tutorial review,*" IEEE ASSP Mag., vol. 6.

We would like to thank our mentor manikandan who have been greatly supporting and provided steady guidance and offered help to make this a success.

10. REFERENCES

1. Apolinário Jr, José A., and Sergio L. Netto. "Introduction to Adaptive Filters." In QRD-RLS Adaptive Filtering, pp. 1-27. Springer US, 2009.
2. B. Widrow and S. D. Steárns, Adaptive signal processing. Prentice Hall, Englewood Cliffs, NJ, 1985.
3. S. Háykin and B. Widrow, Least-mean-square adaptive filters. Wiley-Interscience, Hoboken, NJ, 2003.
4. Park, Sang Yoon, and Prámod Kumar Mehér. "Lowpower, high-throughput, and low-area adaptive FIR filter based on distributed arithmetic." Circuits and Systems II: Express Briefs, IEEE Transactions on 60, no. 6 (2013): 346-350.
5. D. J. Allréd, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 7, pp. 1327–1337, Jul. 2005.
6. P. K. Meher, 'LUT Optimization for Memory-Based Computation,' IEEE Trans on Circuits & Systems-II, pp.285-289, April 2010.
7. Háykin, Simon S. Adaptive filter theory. Pearson Education India, pp.18, 1996.
8. A. Croisiér, D. Estebán, M. Lévilion, and V. Rizo, "Digital filter for PCM encoded signals US Patent 3, 777, 130," 1973.
9. S. Zohar, "New Hardware Realizations of Nonrecursive Digital Filters," IEEE Transactions on Computers, vol. C-22, no. 4, pp. 328–338, 1973.
10. A. Peléd and B. Liu, "A New Hardware Realization of Digital Filters," IEEE Transactions on ASSP, vol. 22, no. 6, pp. 456–462, 1974.