ISSN 23499842(Online), Volume 1, Special Issue 2(ICITET 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

VLSI IMPLEMENTATION OF HIGH PERFORMANCE DISTRIBUTED ARITHMETIC (DA) BASED ADAPTIVE FILTER WITH FAST CONVERGENCE FACTOR

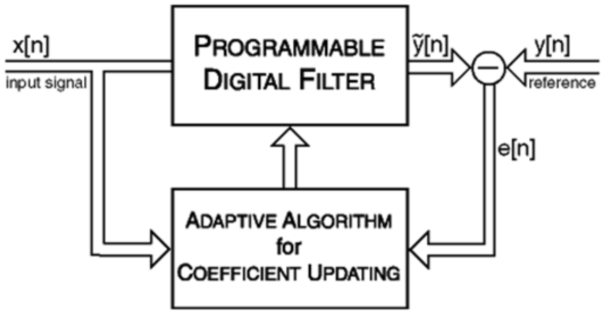
G. PARTHIBAN[1], P.SATHIYA[2]
PG Student, VLSI Design, Department of ECE, Surya Group Of Institutions, India[1].
Assistant Professor, Department of ECE, Surya Group of Institutions, India[2]

Abstract:
The key objective of this paper is to provide an idea for VLSI Implementation of RLS algorithm for noise cancellation with real time analog inputs. In this paper, we present an efficient architecture for the implementation of distributed arithmetic based multiplier less adaptive filter. The throughput rate of update and concurrent implementation of filtering and weight- update operations. The conventional adder-based shift accumulation for DA-based inner product reduce the sampling period and area complexity. The proposed implementation significantly outperforms the existing implementations in terms of three important key metrics. The least mean squares (LMS) algorithms adjust the filter coefficients to minimize the cost function. Compared to least mean squares (LMS) algorithms adjust the filter coefficients to minimize the cost function. Compared to least mean squares algorithms, the RLS algorithms achieve faster Convergence by variable step size. Proposed RLS algorithm require fewer computational resources and memory than the LMS algorithms. The implementation of the algorithms is less complicated due to DA based approach than the all other existing algorithms. Through MATLAB simulation experiments efficiency of RLS over LMS will be proved. The implementation results show that the proposed algorithm as superior performance in fast convergence rate, low complexity, and has superior performance in noise cancellation.

I. INTRODUCTION

Adaptive filters are often realized either as a set of program instructions running on an arithmetical processing device such as a microprocessor or DSP chip, or as a set of logic operations implemented in a field-programmable gate array (FPGA) or in a semicustom or custom VLSI integrated circuit. For this reason, we shall focus on the mathematical forms of adaptive filters as opposed to their specific realizations in software or hardware [1]. In this section, we present the general adaptive filtering problem and introduce the mathematical notation for representing the form and operation of the adaptive filter [2]. We provide a overview of the many and varied applications in which adaptive filters have been successfully used finally, we give a simple derivation of the least-mean-square (LMS) algorithm, which is perhaps the most popular method for adjusting the coefficients of an adaptive filter, and we discuss some of this algorithm's properties. All quantities are assumed to be real-valued. Scalar and vector quantities shall be indicated by lowercase (e.g., x) and uppercase-bold (e.g., X) letters, respectively. We represent scalar and vector sequences or signals as x(n) and X(n), respectively, where n denotes Here w represents the coefficients of the FIR filter tap weight vector, x(n) is the input vector samples, z-1 is a delay of one sample periods [3], y(n) is the adaptive filter output, d(n) is the desired echoed signal and e(n) is the estimation error at time. The aim of an adaptive filter is to calculate the difference between the desired signal and the adaptive filter output, e(n). The novel contributions of this paper are as follows:

1) A novel design strategy for reducing the adaptation delay, which very much impacts the convergence speed.
2) Bit-level optimization of computing blocks without adversely affecting the convergence performance



. Fig. 1 Block diagram of an adaptive filter

II. RELATED WORK & CONTRIBUTIONS

The problem of the efficient realization of a Delayed Least Mean Squares (DLMS) transversal adaptive filter is investigated. A time-shifted version of the DLMS algorithm is derived. The restructured algorithm, due to its order recursive nature, is well suited to parallel implementation. In addition, a pipelined systolic-type architecture which implements the algorithm is presented. The performance of the pipelined system is analyzed, and equations for speedup are derived. The pipelined system described in this paper is capable of much greater throughput than existing conventional LMS implementations, making it a good candidate for real-time applications where high sampling rates are a requirement. Also, due to its highly modular nature, the system is easily expandable. The problem of implementing a high sampling rate transversal form adaptive filter is investigated. A highly pipelined systolic-type alternative to the conventional LMS adaptive filter is presented. The proposed system consists of a linear array of identical processing modules specify suited to the computational requirements of the delayed LMS algorithm. The resulting adaptive filter structure can accommodate very high sampling rates, which are independent of the filter order. the proposed DLMS adaptive filter can be implemented by a pipelined inner-product computation unit for calculation of feedback error, and a pipelined weight-update unit consisting of N parallel multiply accumulators, for filter order N. From the synthesis results we find that the existing direct-form structure of involves nearly 50% more area-delay product (ADP) and nearly 74% more energy per sample (EPS) than the proposed one, in average, for filter orders N = 8, 16 and 32. To have satisfactory convergence performance of DLMS algorithm, we have reduced the adaptation delay and at the same time we have also reduced the critical path to support high input-sampling rate. For achieving lower adaptation-delay and to have area-delay-power efficient implementation, we have proposed a novel multiplication cell and we have optimized the number of pipeline latches across the time-consuming combinational blocks of the structure. delayed least mean square (DLMS) adaptive filter with low-adaptation delay to have better convergence performance and to maintain small critical path to support high input sampling rate. Besides, we have proposed a novel multiplication cell for efficient implementation of error estimation block and weight update block of the adaptive filter. The implementation of adaptive filters with fixed-point arithmetic requires to evaluate the computation quality. The accuracy may be determined by calculating the global quantization noise power in the system output. In this paper, a new model for evaluating analytically the global noise power in the LMS algorithm and in the NLMS algorithm is developed. Two existing models are presented, then the model is detailed and compared with the ones before. The accuracy of our model is analyzed by simulations This approach has for main advantage to be more tractable than the models and to be valid for all types of quantization. A High-Speed FIR Adaptive Filter Architecture using a Modified Delayed LMS Algorithm. A modular pipelined implementation of a Delayed LMS transversal adaptive filter. The LMS Algorithm with Delayed Coefficient updating..High Sampling Rate Delayed LMS Filter Architecture. A Systolic Architecture for LMS Adaptive Filtering with Minimal Adaptation Delay. Low Adaptation-Delay LMS Adaptive Filter Part-I: Introducing a Novel Multiplication Cell. Low Adaptation-Delay LMS Adaptive Filter Part-II: An Optimized Architecture. Accuracy Evaluation of Fixed - Point LMS Algorithm.

III. PROPOSED METHOD

a) *LMS algorithm using distributed arithmetic*

The LMS algorithm is a type of adaptive filter known as stochastic gradient-based algorithms as it utilizes the gradient vector of the filter tap weights to converge on the optimal wiener solution [6]. It is well known and widely used due to its computational simplicity. It is this simplicity that has made it the benchmark against which all other adaptive filtering algorithms are judged

w(n + 1) = w(n) + 2ue(n)x(n) (1)

x(n) = [x(n) x(n-1) x(n-2) …. x(n-N+1)]T (2)

w(n) = [w0(n) w1(n) w2(n) … wN-1(n)]T (3)

Distributed Arithmetic was introduced into FPGA design to save MAC blocks with the development of FPGA technology.

b) *Design of the Distributed Arithmetic algorithm*

The N-length FIR filter can be described as:

x(n) =2 B XB(n) + u2 B XB(n) (4)

Where h[n] is the filter coefficient and x[n] is the input sequence to be processed. The FIR structure consists of a series of multiplication and addition units, and consume N MAC blocks of FPGA, which are expensive in high speed system. Compared with traditional direct arithmetic, Distributed Arithmetic can save considerable hardware resources through using LUT to take the place of MAC units. If we construct a LUT which can store all the possible combination of values, we can calculate the value of 2M in advance and store them in the LUT. Using LUT address signal, the shifting and adding operation are carried out on the output of the LUT. It can be realized through N-1 cycles and the result of multiplication accumulation can be achieved directly. So the complicated multiplication-accumulation operation is converted to the shifting and adding operation.

$$\sum_{b=0}^{B-1} h[n] \sum_{n=0}^{N-1} 2^b x_b[n] = \sum_{b=0}^{B-1} 2^b \sum_{n=0}^{N-1} h[n] x_b[n]$$

ISSN 23499842(Online), Volume 1, Special Issue 2(ICITET 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

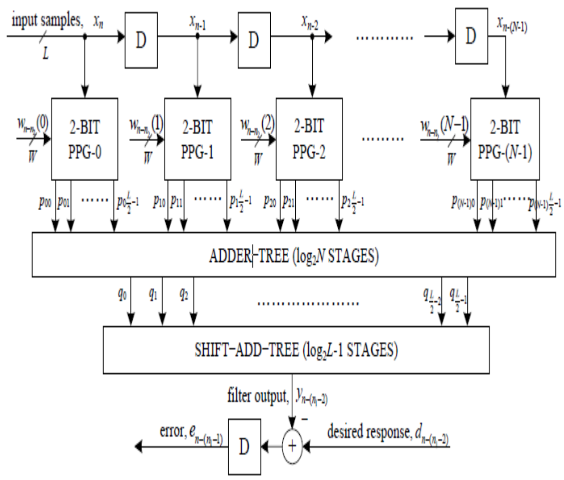


Fig3.1 Proposed structure of error computational block

c) *Proposed LUT Optimization Scheme*

In the new approach to LUT design, where only the odd multiples of the fixed coefficient are required to be stored, which we have referred to as the odd-multiple-storage (OMS) scheme in this brief. In addition, we have shown that, by the anti symmetric product coding (APC) approach, the LUT size can also be reduced to half, where the product words are recoded as anti symmetric pairs. The APC approach, although providing a reduction in LUT size by a factor of two, incorporates substantial overhead of area and time to perform the two's complement operation of LUT output for sign modification and that of the input operand for input mapping**.** The OMS scheme in does not provide an efficient implementation when combined with the APC technique.

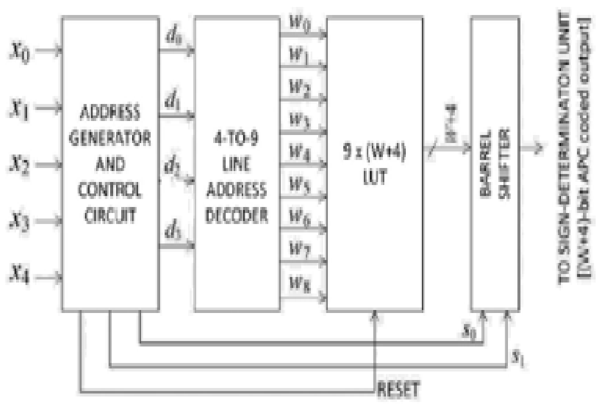


Fig 3.2 Proposed APC–OMS combined LUT design

d) Modified OMS for LUT Optimization

The LUT for the multiplication of an L-bit input with a W-bit coefficient could be designed by the following strategy.

1) A memory unit of [(2L/2) + 1] words of (W + L)-bit width is used to store the product values, where the first (2L/2) words are odd multiples of A, and the last word is zero.

2) A barrel shifter for producing a maximum of (L − 1) left shifts is used to derive all the even multiples of A.

3) The L-bit input word is mapped to the (L − 1)-bit address of the LUT by an address encoder, and control bits for the barrel shifter are derived by a control circuit.

TABLE I

a) APC words for different input

Input, X	product values	Input, X	product values	address $x'_3x'_2x'_1x'_0$	APC words
0 0 0 0 1	A	1 1 1 1 1	$31A$	1 1 1 1	$15A$
0 0 0 1 0	$2A$	1 1 1 1 0	$30A$	1 1 1 0	$14A$
0 0 0 1 1	$3A$	1 1 1 0 1	$29A$	1 1 0 1	$13A$
0 0 1 0 0	$4A$	1 1 1 0 0	$28A$	1 1 0 0	$12A$
0 0 1 0 1	$5A$	1 1 0 1 1	$27A$	1 0 1 1	$11A$
0 0 1 1 0	$6A$	1 1 0 1 0	$26A$	1 0 1 0	$10A$
0 0 1 1 1	$7A$	1 1 0 0 1	$25A$	1 0 0 1	$9A$
0 1 0 0 0	$8A$	1 1 0 0 0	$24A$	1 0 0 0	$8A$
0 1 0 0 1	$9A$	1 0 1 1 1	$23A$	0 1 1 1	$7A$
0 1 0 1 0	$10A$	1 0 1 1 0	$22A$	0 1 1 0	$6A$
0 1 0 1 1	$11A$	1 0 1 0 1	$21A$	0 1 0 1	$5A$
0 1 1 0 0	$12A$	1 0 1 0 0	$20A$	0 1 0 0	$4A$
0 1 1 0 1	$13A$	1 0 0 1 1	$19A$	0 0 1 1	$3A$
0 1 1 1 0	$14A$	1 0 0 1 0	$18A$	0 0 1 0	$2A$
0 1 1 1 1	$15A$	1 0 0 0 1	$17A$	0 0 0 1	A
1 0 0 0 0	$16A$	1 0 0 0 0	$16A$	0 0 0 0	0

TABLE II

b) OMS Based design of the LUT of APC Words

input X' $x'_3x'_2x'_1x'_0$	product value	# of shifts	shifted input, X''	stored APC word	address $d_3d_2d_1d_0$
0 0 0 1	A	0	0 0 0 1	$P0 = A$	0 0 0 0
0 0 1 0	$2 \times A$	1			
0 1 0 0	$4 \times A$	2			
1 0 0 0	$8 \times A$	3			
0 0 1 1	$3A$	0	0 0 1 1	$P1 = 3A$	0 0 0 1
0 1 1 0	$2 \times 3A$	1			
1 1 0 0	$4 \times 3A$	2			
0 1 0 1	$5A$	0	0 1 0 1	$P2 = 5A$	0 0 1 0
1 0 1 0	$2 \times 5A$	1			
0 1 1 1	$7A$	0	0 1 1 1	$P3 = 7A$	0 0 1 1
1 1 1 0	$2 \times 7A$	1			
1 0 0 1	$9A$	0	1 0 0 1	$P4 = 9A$	0 1 0 0
1 0 1 1	$11A$	0	1 0 1 1	$P5 = 11A$	0 1 0 1
1 1 0 1	$13A$	0	1 1 0 1	$P6 = 13A$	0 1 1 0
1 1 1 1	$15A$	0	1 1 1 1	$P7 = 15A$	0 1 1 1

The proposed APC–OMS combined design of the LUT for $L = 5$ and for any coefficient width W is shown in Fig. 3. It consists of an LUT of nine words of $(W + 4)$-bit width, a four-to-nine-line address decoder, a barrel shifter, an address generation circuit, and a control circuit for generating the RESET signal and control word ($s1s0$) for the barrel shifter. The pre computed values of $A \times (2i + 1)$ are stored as Pi, for $i = 0, 1, 2, \ldots, 7$, at the eight consecutive locations of the memory array, as specified in Table II, while $2A$ is stored for input $X = (00000)$ at LUT address "1000," as specified in Table III. The decoder takes the 4-bit address from the address generator and generates nine word-select signals, i.e., $\{wi$, for $0 \le i \le 8\}$, to select the referenced word from the LUT. The 4-to-9-line decoder is a

ISSN 23499842(Online), Volume 1, Special Issue 2(ICITET 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

simple modification of 3-to- 8-line decoder, as shown in Fig.

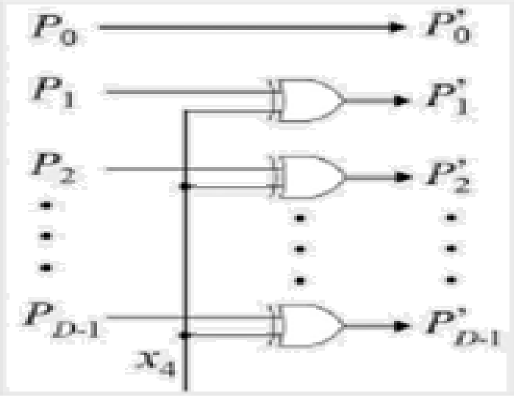


Fig 4.3Sign modification LUT output

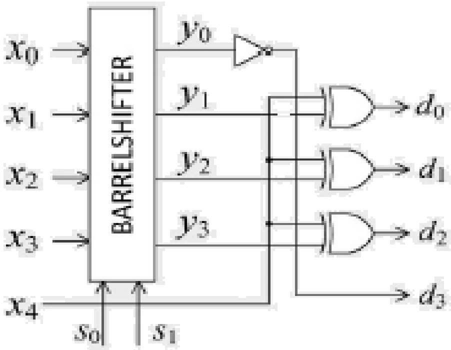


Fig 4.4 Address-generation circuit

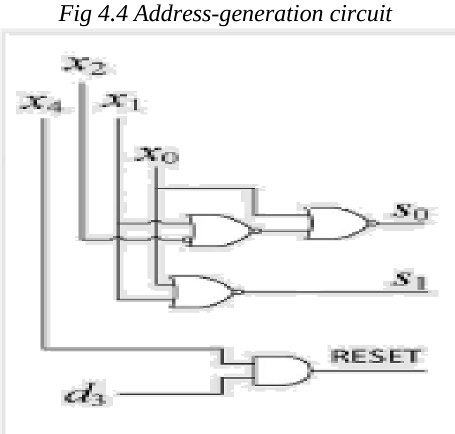


Fig 4.5 Control Circuit For Generation of S0, S1, and RESET

IV. SIMULATION RESULT& VERIFICATIONS

a) Weightage updation in LMS

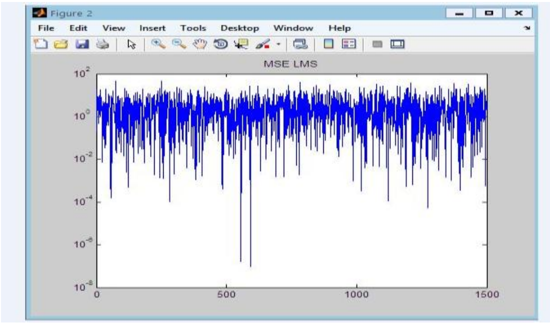


Fig 4.1 We*ightage updation in LMS*

b) We*ightage updation in RLS*

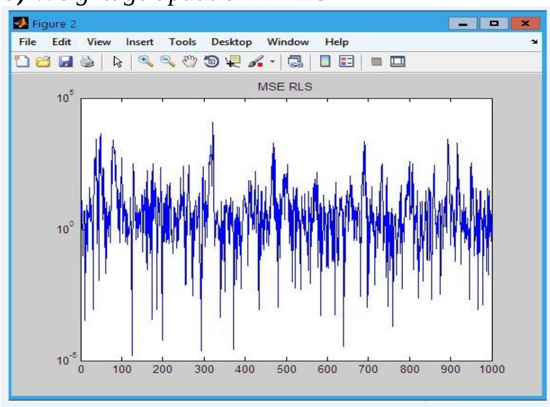


Fig 4.2 Weightage updation in RLS

c) *LMS Waveform*

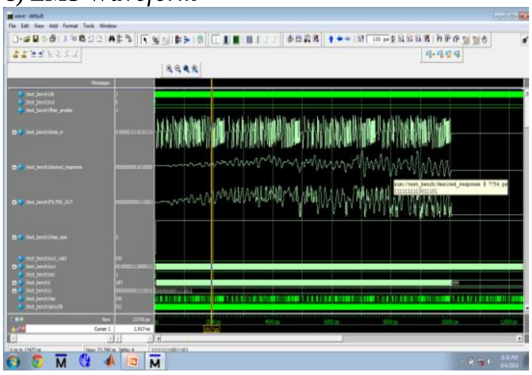


Fig 4.3 *LMS Waveform*

d) Area without DA analysis

PowerPlay Power Analyzer Status	Successful - Fri Sep 05 10:32:18 2014
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	nivipathi
Top-level Entity Name	adoptive_filter
Family	Cyclone III
Device	EP3C16F256C6
Power Models	Final
Total Thermal Power Dissipation	70.69 mW
Core Dynamic Thermal Power Dissipation	2.40 mW
Core Static Thermal Power Dissipation	51.79 mW
I/O Thermal Power Dissipation	16.50 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

Fig 4.4 *Area without DA analysis*

ISSN 23499842(Online), Volume 1, Special Issue 2(ICITET 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

e) *Power Analyzer without DA analysis*

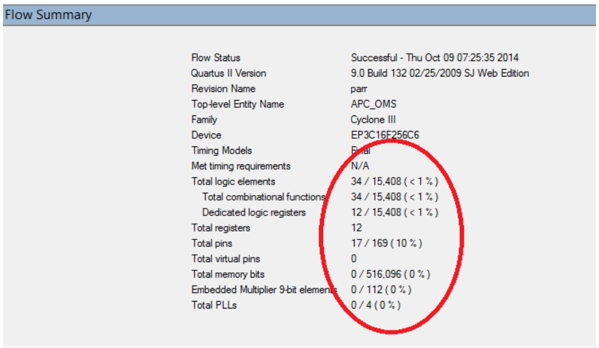


Fig 4.5 *Power Analyzer without DA analysis*

f) Power Analyzer with DA analysis

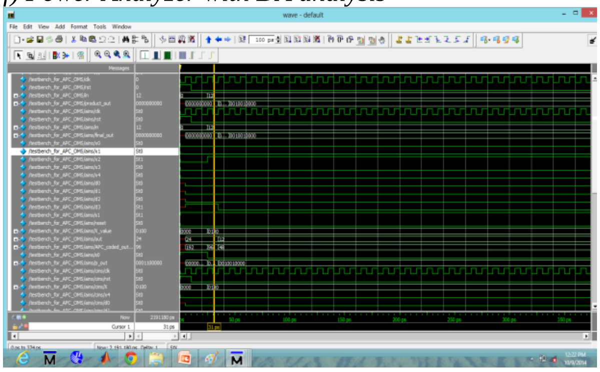


Fig 4.6 *Power Analyzer with DA analysis*

g) APC & OMS Waveform

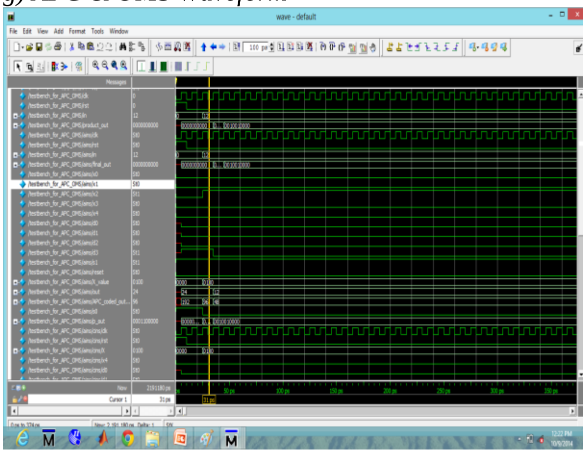


Fig 4.6 *APC & OMS Waveform*

h) Comparison for multipliers & DA

TYPE	MULTIPLIER S USED	LOGIC ELEMENT S USED
MULTIPLIER S BASED	33	297
DISTRIBUTE D ARITHMETIC BASED	00	34

V.CONCULSION

An area-delay-power efficient low adaptation-delay architecture for fixed-point implementation of LMS adaptive filter. We have used a novel partial-product generator for efficient implementation of general multiplications and inner-product computation by Distributed Arithmetic. We have proposed an efficient addition-scheme for inner-product computation to reduce the adaptation-delay significantly in order to achieve faster convergence performance, and to reduce the critical path to support high input sampling-rate. The proposed structure involves significantly less adaptation-delay and provides significant saving of area-delay-product and energy-delay-product compared with the existing structures. We have reduced the area complexity by introducing APC and OMS. We have proposed an fixed-point implementation of the proposed architecture, and derived the expression for steady-state error. We find that the steady-state MSE obtained from the analytical result matches with the simulation result.

REFERENCES

[1] G. Long, F. Ling, and J. G. Proakis, “Corrections to ‘The LMS algorithm with delayed coefficient adaptation’,” *IEEE Trans. Signal Process.*, vol. 40, no. 1, pp. 230–232, Jan. 1992.

[2] M. D. Meyer and D. P. Agrawal, “A modular pipelined implementation of a delayed LMS transversal adaptive filter,” in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1990, pp. 1943–1946

[3] M. D. Meyer and D. P. Agrawal, “A high sampling rate delayed LMS filter architecture,” *IEEE Trans. Circuits Syst. II, Analog Digital Signal Process.*, vol. 40, no. 11, pp. 727–729, Nov. 1993.

[4] P. K. Meher and M. Maheshwari, “A high-speed FIR adaptive filter architecture using a modified delayed LMS algorithm,” in Proc. IEEE International Symposium on Circuits and Systems (ISCAS), Rio de Janeiro, Brazil, May 2011.