JOURNAL OF CRITICAL REVIEWS
ISSN- 2394-5125 VOL 7, ISSUE 15, 2020

AREA-DELAY-POWER EFFICIENT FIXED-POINT LMS ADAPTIVE FILTER WITH LOW ADAPTATION-DELAY

Mrs.Nidhi Chahal[1], Samandeep Singh[2], Dr. Parveen Singla[3]

[1]Assistant professor, Department of Electronics and Communication, CEC Landran, Mohali (Pb), India
nidhi.leo08@gmail.com
[2]Research scholar, Department of Electronics and Communication ,CEC Landran, Mohali (Pb),India
samandhillon17@gmail.com
[3]Professor, Department of Electronics and Communication, CEC Landran, Mohali (Pb), India

Received: 16 March 2020 Revised and Accepted: 17 June 2020

ABSTRACT: We present an efficient architecture for the implementation of a delayed least mean square adaptive filter. For achieving lower adaptation-delay and area-delay-power efficient implementation, we use a novel partial product generator and propose a strategy for optimized balanced pipelining across the time-consuming combinational blocks of the structure. From synthesis results, we find that the proposed design offers nearly 17% less area-delay product (ADP) and nearly 14% less energy-delay product (EDP) than the best of the existing systolic structures, on average, for filter lengths N = 8, 16, and 32. We propose an efficient fixed-point implementation scheme of the proposed architecture, and derive the expression for steady-state error. We show that the steady-state mean squared error obtained from the analytical result matches with the simulation result. Moreover, we have proposed a bit-level pruning of the proposed architecture, which provides nearly 20% saving in ADP and 9% saving in EDP over the proposed structure before pruning without noticeable degradation of steady-state-error performance

KEYWORD: Adaptive filters, circuit optimization, fixed-point arithmetic, least mean square (LMS) algorithms

I. INTRODUCTION

The filter is an important component in the communication world. It can eliminate unwanted signals from useful information. However, to obtain an optimal filtering performance, it requires 'a priori' knowledge of both the signal and its embedded noise statistical information. The classical approach to this problem is to design frequency selective filters, which approximate the frequency band of the signal of interest and reject those signals outside this frequency band. The removal of unwanted signals through the use of optimization theory is becoming popular, particularly in the area of adaptive filtering. Adaptive digital filters have been applied to a wide variety of important problems in recent years. Perhaps one of the most well-known adaptive algorithms is the least mean squares (LMS) algorithm, which updates the weights of a transversal filter using an approximate technique of steepest descent. Due to its simplicity, the LMS algorithm has received a great deal of attention, and has been successfully applied in a number of areas including channel equalization, noise and echo cancellation and many others.However, the feedback of the error signal needed to update filter weights in the LMS algorithm imposes a critical limitation on the throughput of possible implementations. In particular, the prediction error feedback requirement makes extensive pipelining of filter computations impossible. As a result of this, the design of high speed adaptive filters has been predominantly based on the adaptive lattice filter, which lends itself easily to pipelining. Recently it has been shown that it is possible to introduce some delay in the weight adaptation of the LMS algorithm. The resulting delayed least mean squares (DLMS) algorithm, which uses a delayed prediction error signal to update the filter weights, has been shown to guarantee stable convergence characteristics provided an appropriate adaptation step size is chosen.

II. RELATED WORK

A lot of work has been done to implement the DLMS algorithm in systolic architectures to increase the maximum usable frequency but, they involve an adaptation delay of $\sim N$ cycles for filter length N, which is quite high for large order filters. Since the convergence performance degrades considerably for a large adaptation delay, Visvanathan *et al*. have proposed a modified systolic architecture to reduce the adaptation delay. A transpose-form LMS adaptive filter was suggested, where the filter output at any instant depends on the delayed versions of weights and the number of delays in weights varies from 1 to N. Van and Feng have

proposed a systolic architecture, where they have used relatively large processing elements (PEs) for achieving a lower adaptation delay with the critical path of one MAC operation. Ting *et al*. have proposed a fine-grained pipelined design to limit the critical path to the maximum of one addition time, which supports high sampling frequency, but involves a lot of area overhead for pipelining and higher power consumption, due to its large number of pipeline latches. Further effort has been made by Meher and Maheshwari to reduce the number of adaptation delays. Meher and Park have proposed a 2-bit multiplication cell, and used that with an efficient adder tree for pipelined inner-product computation to minimize the critical path and silicon area without increasing the number of adaptation delays.

Related Works:

1. "Accuracy evaluation of fixed-point LMS algorithm",By R. Rocher, D. Menard, O. Sentieys, and P. Scalart, in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2004, pp. 237–240.

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.

In this paper, a new model for evaluating the noise power in a fixed-point implementation of the LMS algorithm is presented. This approach has for main advantage to be more tractable to be valid for all types of quantization. This model can be improved through the determination of $E(\rho n)$ since the two methods (equations 26 and 27) can be improved. A global model must be developed for this term. Nevertheless, further studies have to be carried out in order to develop this methodology for all types of systems and particularly, non-linear systems.

2. "Architecture Design for an Adaptive Equalizer using LMS 2Tap filters",By *P.S. Radhika, N.Porutchelvam – in SSRG*
International Journal of VLSI & Signal Processing (SSRG-IJVSP) – volume 2 Issue 1 Jan-Feb 2014

In this paper, an adaptive equalizer using LMS algorithm has been simulated using Modelsim software. Also the variants of
LMS algorithm sign- sign, sign-data and sign-error are simulated for low complexity adaptive equalizer. Analysis of mean squared errors for LMS and its variants have been done. From the results of Modelsim, input data is taken for VHDL code simulation of LMS adaptive equalizer. The output from VHDL simulation is plotted using Modelsim. The two tap process method has improves the Architecture and the LMS adaptive equalizer is discussed here and simulation results have been obtained. The Xilinx simulation used for to finding the error data.

The LMS algorithm and the variants of LMS algorithm sign-sign LMS, sign-error LMS and sign-data LMS are verified and their feasibility is tested for eleven tap adaptive equalizer. VHDL simulation of two tap adaptive equalizer is tested for LMS algorithm. Multiplexed multiplier architecture is proposed for LMS adaptive equalizer in order to reduce the number of multipliers . To reduce the hardware complexity and feedback latency still more ,sign-sign LMS algorithm can be used for VHDL simulation of adaptive channel equalizer which results in area and power savings. On the other hand the converging speed is low compared to normal LMS algorithm.

3."An Efficient Adaptive Fir Filter Based On Distributed Arithmetic",By M.Usha , 2,R.Ramadoss - in *International Journal of Engineering Science Invention ISSN (Online, www.ijesi.org Volume 3 Issue 4, April 2014 , PP.15-20*

Adaptive filtering constitutes an important class of DSP algorithms employed in several hand held mobile devices for applications such as echo cancellation, signal de-noising, and channel equalization. The throughput of the proposed design is increased by parallel lookup table (LUT) update.The 16:1 multiplexer is replace by a 8:1 and 2:1 MUX. The conventional adder-based shift accumulation for DA-based inner-product computation is replaced by conditional signed carry-save accumulation in order to reduce the area complexity; the power consumption of the proposed design is reduced by using a fast bit clock for all operations. It involves the same

number of multiplexors, smaller LUT, and nearly half the number of adders compared to the existing DA-based design. The proposed architecture is found to involve significantly 29% less area-13% less power and throughput compared with the existing DA-based implementations of FIR filter and a increase in operating frequency of 12MHZ is achieved.

We have suggested an efficient pipelined architecture for low-power, high-throughput, and low-area implementation of DA-based adaptive filter. Throughput rate is significantly enhanced by parallel LUT update and concurrent processing of filtering operation and weight-update operation. We have also proposed a carry-save accumulation scheme of signed partial inner products for the computation of filter output. From the synthesis results, we find that the proposed design consumes 13% less power and 29% less ADP over our previous DA-based FIR adaptive filter in average for filter lengths N=16 Compared to the best of other existing designs, our proposed architecture provides 9.5 times less power and 4.6 times less ADP. Offset binary coding is popularly used to reduce the LUT size to half for area-efficient implementation of DA which can be applied to our design as well.

4."Enhanced-Convergence Normalized LMS Algorithm", By Maurice Givens, *Erik G. Larsson* – in IEEE SIGNAL PROCESSING MAGAZINE MAY 2009, *Digital Object Identifier 10.1109/MSP.2009.932168*

Least mean square (LMS) algorithms have found great utility in many adaptive filtering applications. This article shows how the traditional constraints placed on the update gain of normalized LMS algorithms are overly restrictive. We present relaxed update gain constraints that significantly improve normalized LMS algorithm convergence speed.The goal of this lecture has been to provide an overview of approaches to (2), in the communications receiver context. Which method is the best in practice? This depends much on the purpose of solving (2): what error rate can be tolerated, what is the ultimate measure of performance (e.g., frame-error-rate, worst-case complexity, or average complexity),and what computational platform is used. Additionally, the bits in **s** may be part of a larger code word and different **s** vectors in that code word may either see the same ***H*** (slow fading) or many different realizations of ***H*** (fast fading). This complicates the picture, because notions that are important in slow fading (such as spatial diversity) are less important in fast fading, where diversity is provided anyway by time variations. Detection for MIMO has been an active field for more than ten years, and this research will probably continue for some time.

5." A Multiple Error LMS Algorithm and Its Application to the Active Control of Sound and Vibration", By STEPHEN J. ELLIOTT, IAN M. STOTHERS, AND PHILIP A. NELSON – in IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-35, NO. 10, OCTOBER 1987

An algorithm is presented to adapt the coefficients of an array of FIR filters, whose outputs are linearly coupled to another array of error detection points, so that the sum of all the mean square error signals is minimized. The algorithm uses the instantaneous gradient of the total error, and for a single filter and error reduces to the "filtered x LMS" algorithm. The application of this algorithm to active sound and vibration control is discussed, by which suitably driven secondary sources are used to reduce the levels of acoustic or vibrational fields by minimizing the sum of the squares of a number of error sensor signals. A practical implementation of the algorithm is presented for the active control of sound at a single frequency. The algorithm converges on a timescale comparable to the response time of the system to be controlled, and is found to be very robust. If the pure tone reference signal is synchronously sampled, it is found that the behavior of the adaptive system can be completely described by a matrix of linear, time invariant, transfer functions. This is used to explain the behavior observed in simulations of a simplified single input, single output adaptive system, which retains many of the properties of the multichannel algorithm. A generalization of the filtered *x* LMS algorithm has been presented which minimizes the sum of the mean square outputs of a number of errors, each linearly related to the outputs of a number of adaptive filters. The derivation of the algorithm involved the assumption that the adaptive filters were only varying slowly compared to the timescale of the response of the system to be controlled. However, simulations of the algorithm using a sinusoidal reference, and a practical implementation in an active sound control application, have shown that the algorithm is able to converge in a time comparable to the response time of the system to be controlled. The simulations of the algorithm also indicate that the total error converged to a value close to theo ptimum least mean sum of squares solution, and that it was robust to errors made in the assumed response of the system to be controlled and to uncorrelated measurement noise.Similar behavior is also shown by a simplified, single input, single output version of the algorithm, which corresponds to the filtered *x* LMS algorithm. The pole

positions of the equivalent transfer function, derived using the approach of Glover, can, however, be easily evaluated in this case. These can be used to analytically derive an expression for the optimum convergence coefficient, which, in this case, agrees well with computer simulations, and is approximately equal to the reciprocal of the delay in the system to be controlled, measured in samples. The equivalent transfer function can also be used to analytically demonstrate that there is a f90" phase condition on the estimate of the system response in the limit of slow adaption. Two modifications to the multichannel algorithm are presented. The first which penalizes effort as well as error, and the second which minimizes the sum of a higher order function of the errors. The latter algorithm tends to a minimax solution as the power to which the individual errors are raised is increased.

III. PROPOSED SYSTEM

3.1 Objective:

To present an efficient architecture of a modified delayed least mean square adaptive filter for achieving lower area,delay and power.

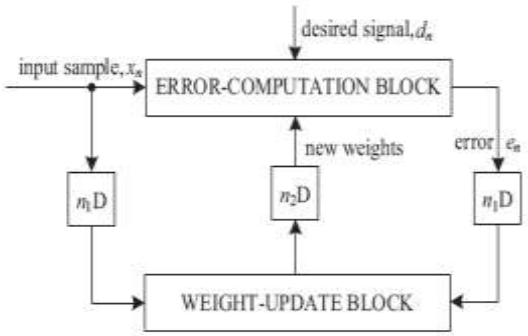

Fig.3.1 Block Diagram of the modified delayed LMS adaptive filter.

TABLE I

LOCATION OF PIPELINE LATCHES FOR $L = 8$ AND $N = 8$, 16, AND 32

N	Error-Computation Block		Weight-Update Block
	Adder Tree	Shift–add Tree	Shift–add Tree
8	Stage-2	Stage-1 and 2	Stage-1
16	Stage-3	Stage-1 and 2	Stage-1
32	Stage-3	Stage-1 and 2	Stage-2

Fig 3.1 Shows the Block Diagram of the Modified Delayed LMS Adaptive Filter.There are two main computing blocks in the adaptive filter architecture:

1) the error-computation block, and

2) weight-update block.

In this Section, we discuss the design strategy of the proposed structure to minimize the adaptation delay in the error-computation block, followed by the weight-update block.

3.2 Block Diagram

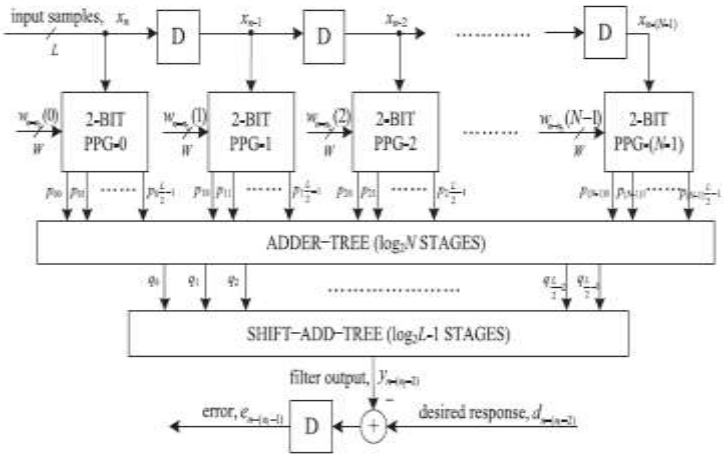

Fig.3.2. Proposed structure of the error-computation block.

3.3 Pipelined Structure of the Error-Computation Block

The proposed structure for error-computation unit of an *N*-tap DLMS adaptive filter as shown in Fig3.2.It consists of *N* number of 2-b partial product generators (PPG) corresponding to *N* multipliers and a cluster of *L*/2 binary adder trees, followed by a single shift–add tree. Implementation, we use a novel partial product generator and propose a strategy for optimized balanced pipelining across the time consuming
combinational blocks of the structure. In this project, we find that the proposed design offers less area-delay product and less energy-delay product than the best of the existing systolic structures, on average, for filter lengths N = 8. But We propose an efficient implementation scheme of the proposed architecture for steady-state error. Moreover, we have proposed a bit-level pruning of the proposed architecture, which provides saving in ADP and saving in EDP over the proposed structure.

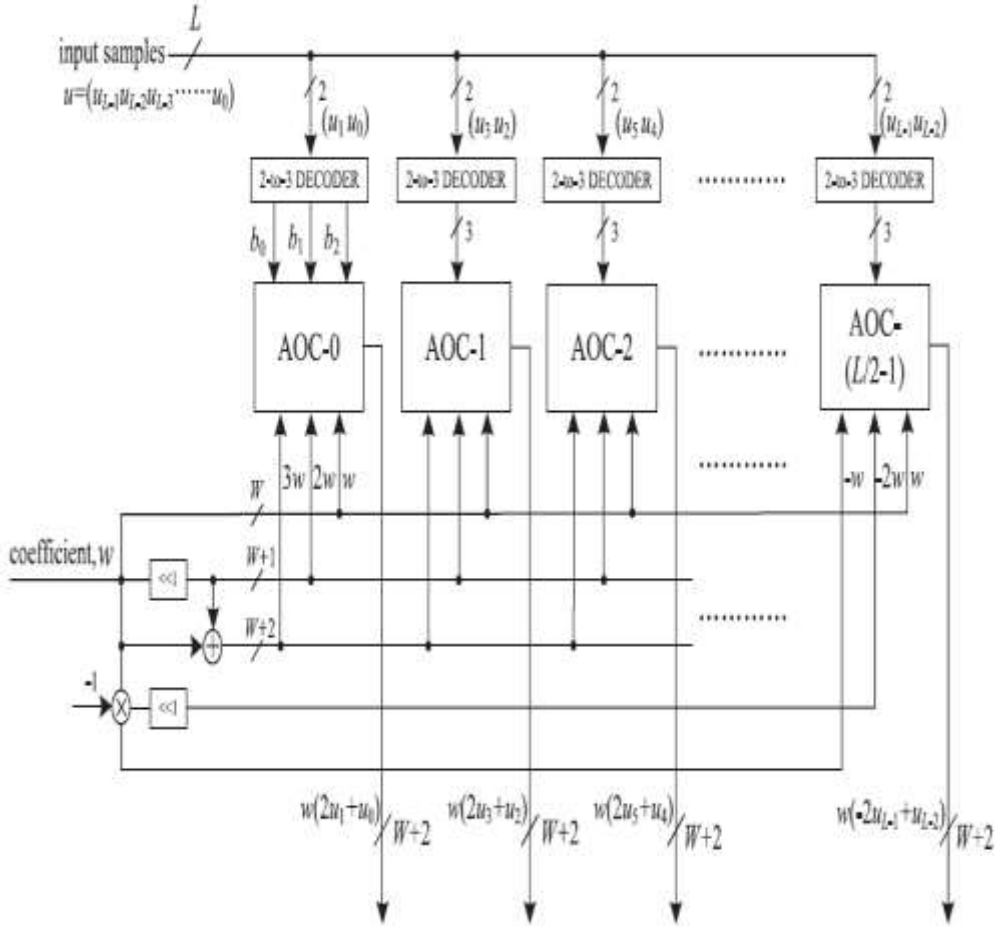

Fig.3.3.Proposed structure of PPG. AOC stands for AND/OR cell.

3.3.1 Partial Product Generation:

The structure of each PPG as shown in Fig 3.3,consists of 4 numbers of 2-to-3 decoders and the same number of AND/OR cells.

i)2-3 Decoders:

Each of the 2-to-3 decoders takes a 2-b digit as input and produces three outputs bo,b1,b2 formula for decoder:b0 = u0 . u1, b1 =. u0 · u1, and b2 = u0 · u1. The decoder output b0, b1 and b2 along with w, 2w, and 3w are fed to an AOC, where w, 2w, and 3w are in 2's complement representation and sign-extended to have (W + 2) bits each. To take care of the sign of the input samples while computing the partial product corresponding to the most significant digit (MSD), i.e., (uL−1uL−2) of the input sample, the AOC (L/2 − 1) is fed with w, −2w, and −w as input since (uL−1uL−2) can have four possible values 0, 1, −2, and −1.

3.3.2 AOCs:

Each AOC consists of three AND cells and two OR cells. Each AND cell takes an n-bit input D and a single bit input b, and consists on AND gates. It distributes all the n bits of input D to its n AND gates as one of the inputs. The other inputs of all the n AND gates are fed with the single-bit input b. Each OR cell similarly takes a pair of n-bit input words and has n OR gates. A pair of bits in the same bit position in B and D is fed to the same OR gate. The output of an AOC is w, 2w, and 3w corresponding to the decimal values 1, 2, and 3 of the 2-b input (u1u0), respectively. The decoder along with the AOC performs a multiplication of input operand w with a 2-b digit (u1u0), such that the PPG performs L/2 parallel multiplications of input word w with a 2-b digit to produce L/2 partial products of the product word wu.

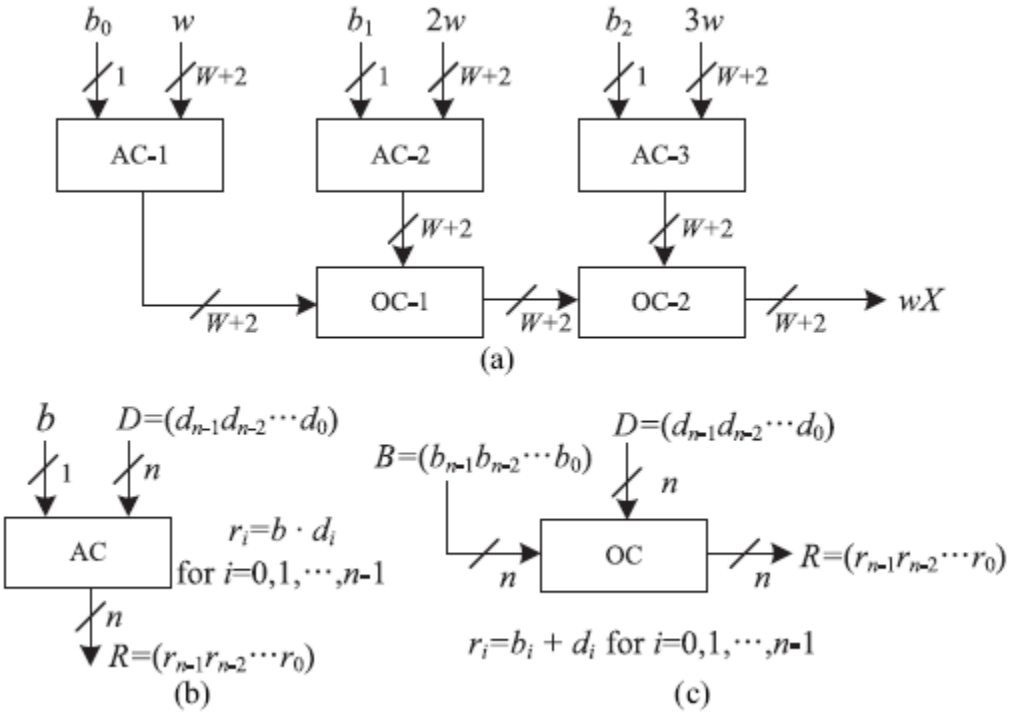

Fig.3.4.Structure and function of AND/OR cell. Binary operators · and + in (b) and (c) are implemented using AND and OR gates, respectively.

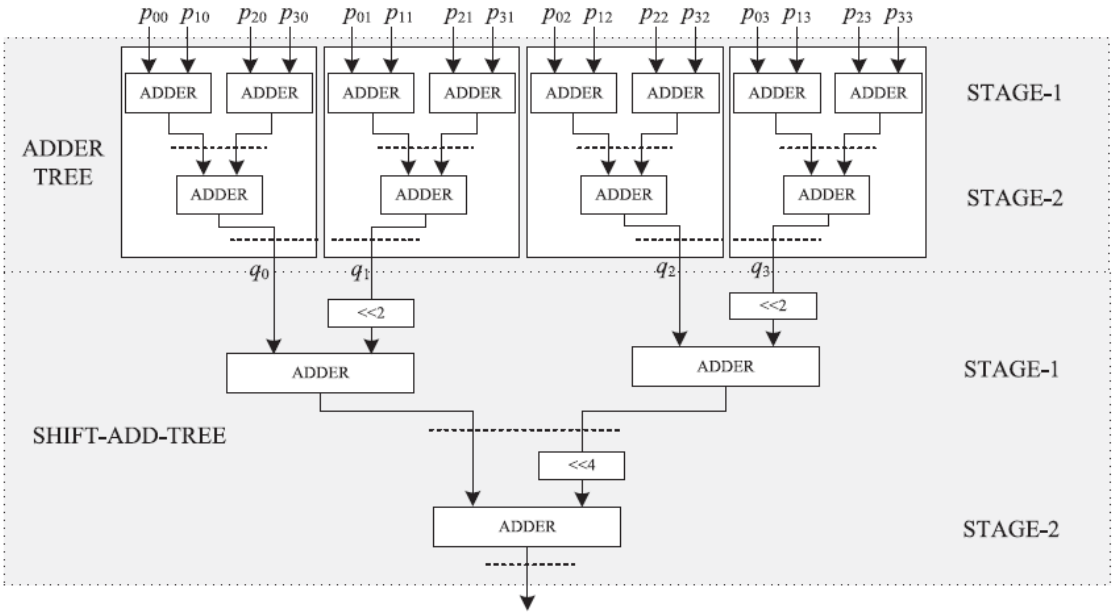

Fig.3.5. Adder-structure of the filtering unit for *N* = 4 and *L* = 8.

3.3.3 Adder Tree:

Conventionally, we should have performed the shift-add operation on the partial products of each PPG separately to obtain the product value and then added all the four product values to compute the desired inner product. However, the shift-adds operation to obtain the product value increases the word length, and consequently increases the adder size of three additions of the product values. To avoid such increase in word size of the adders, we add all the four partial products of the same place value from all the four PPGs by Ripple carry Adder tree.

i)Shift-add tree:

All the FOUR partial products generated by each of the four PPGs are thus added by four binary adder trees. The outputs of the four adder trees (RCA) are then added by a shift-add tree according to their place values.

Each of the binary adder trees require two stages of adders to add N partial product, and the shift–add tree to add four output of four binary adder trees.2 The addition scheme for the error computation block for a four-tap filter and input word size L = 8 is shown in Fig. 3.5. For N = 4 and L = 8, the adder network requires four binary adder trees of two stages each and a two-stage shift–add tree. In this figure, we have shown all possible locations of pipeline latches by dashed lines, to reduce the critical path to one addition time, which would lead to a high adaptation delay and introduce a large overhead of area and power consumption for large values of four and eight.

3.4 Pipelined Structure of the Weight-Update Block

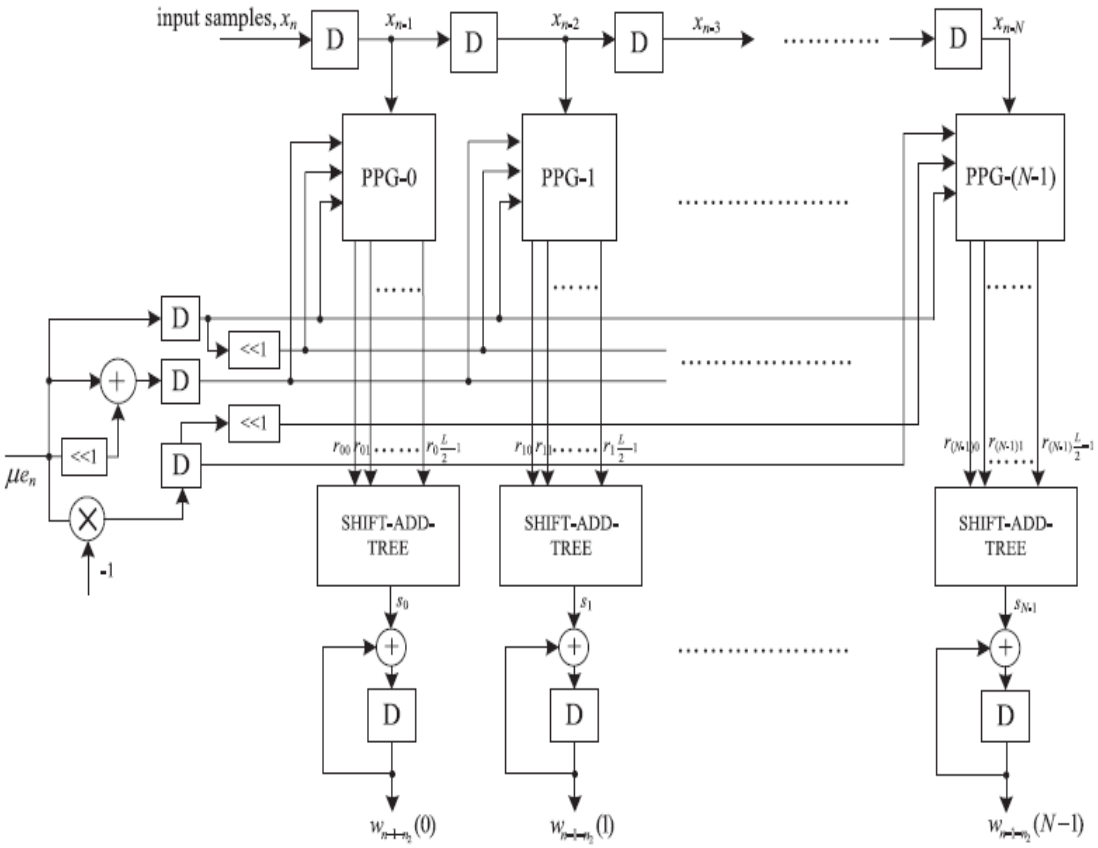

Fig.3.6.Proposed structure of the weight-update block.

The weight-update block performs N multiply-accumulate operations of the form *(μ ×e) ×xi + wi*to update *N* filter weights. The step size μ is taken as a negative power of 2 to realize the multiplication with recently available error only by a shift operation. Each of the MAC units therefore performs the multiplication of the shifted value of error with the delayed input samples *xi* followed by the additions with the corresponding old weight values *wi*. All the *N* multiplications for the MAC operations are performed by *N* PPGs, followed by *N* shift add trees. Each of the PPGs generates *L*/2 partial products corresponding to the product of the recently shifted error value $\mu \times e$ with *L*/2, the number of 2-b digits of the input word *xi*, where the sub expression $3\mu \times e$ is shared within the multiplier. Since the scaled error ($\mu \times e$) is multiplied with the entire*N* delayed input values in the weight-update block, this sub expression can be shared across all the multipliers as well. This leads to substantial reduction of the adder complexity. The final outputs of MAC units constitute the desired updated weights to be used as inputs to the error-computation block as well as the weight-update block for the next iteration.

3.5 Adaptation Delay

As shown in Fig. 3.1, the adaptation delay is decomposed into *n*1 and *n*2. The error-computation block generates the delayed error by *n*1 −1 cycles as shown in Fig. 3.2, which is fed to the weight-update block shown in Fig. 3.6 after scaling by μ; then the input is delayed by 1 cycle before the PPG to make the total delay introduced by FIR filtering be *n*1. In Fig. 3.6, the weight-update block generates **w***n*−1−*n*2, and the weights are delayed by *n*2+1 cycles. However, it should be noted that the delay by 1 cycle is due to the latch before the PPG, which is included in the delay of the error-computation block, i.e., *n*1. Therefore, the delay generated in the weight-

update block becomes *n*2. If the locations of pipeline latches are decided as in Table I, *n*1 becomes 5, where three latches are in the error-computation block, one latch is after the subtraction in Fig. 3.2, and the other latch is before PPG in Fig. 3.6. Also, *n*2 is set to 1 from a latch in the shift-add tree in the weight-update block.

IV. CONCLUSION

We proposed an area–delay-power efficient low adaptation delay architecture for fixed-point implementation of LMS adaptive filter. We used a novel PPG for efficient implementation of general multiplications and inner-product computation by common sub expression sharing. Besides, 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 rates. Aside from this, we proposed a strategy for optimized balanced pipelining across the time-consuming blocks of the structure to reduce the adaptation delay and power consumption, as well. The proposed structure involved significantly less adaptation delay and provided significant saving of ADP and EDP compared to the existing structures. We proposed a fixed-point implementation of the proposed architecture, and derived the expression for steady-state error. We found that the steady-state MSE obtained from the analytical result matched well with the simulation result. We also discussed a pruning scheme that provides nearly 20% saving in the ADP and 9% saving in
EDP over the proposed structure before pruning, without a noticeable degradation of steady-state error performance. The highest sampling rate that could be supported by the ASIC implementation of the proposed design ranged from about 870 to 1010 MHz for filter orders 8 to 32. When the adaptive filter is required to be operated at a lower sampling rate, one can use the proposed design with a clock slower than the maximum usable frequency and a lower operating voltage to reduce the power consumption further.

V. REFERENCES

[1] “Accuracy evaluation of fixed-point LMS algorithm”,By R. Rocher, D. Menard, O. Sentieys, and P. Scalart, in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., May 2004, pp. 237–240.

[2] 2.” Architecture Design for an Adaptive Equalizer using LMS 2Tap filters”,By P.S. Radhika, N.Porutchelvam – in SSRG International Journal of VLSI & Signal Processing (SSRG-IJVSP) – volume 2 Issue 1 Jan-Feb 2014

[3] 3.”An Efficient Adaptive Fir Filter Based On Distributed Arithmetic”,By M.Usha , 2,R.Ramadoss - in International Journal of Engineering Science Invention ISSN (Online, www.ijesi.org Volume 3 Issue 4, April 2014 , PP.15-20

[4] 4.”Enhanced-Convergence Normalized LMS Algorithm”, By Maurice Givens, Erik G. Larsson – in IEEE SIGNAL PROCESSING MAGAZINE MAY 2009, Digital Object Identifier 10.1109/MSP.2009.932168

[5] 5.” A Multiple Error LMS Algorithm and Its Application to the Active Control of Sound and Vibration”, By STEPHEN J. ELLIOTT, IAN M. STOTHERS, AND PHILIP A. NELSON – in IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-35, NO. 10, OCTOBER