Fixed-Point LMS Adaptive Filter with Low Adaptation Delay

K.L.Sowmya[1], M.Pradeep[2]
[1]PG scholar (M.Tech., VLSI Design), [2]Assistant Professor
Shri Vishnu Engineering College for Women, Bhimavaram, India

***Abstract*—This paper presents an efficient architecture for the implementation of a Delayed Least Mean Square (DLMS) adaptive filter consists of Error computation block and weight update block. For achieving lower adaptation delay, we use novel partial product generator and proposed an optimized balanced pipelining across the time consuming blocks of the structure. We propose an efficient fixed-point implementation scheme in the proposed architecture. We present here the optimization of the design to reduce the number of pipeline delays along with area, sampling period and energy consumption. Moreover we proposed a bit-level pruning of the proposed architecture, which provides saving in delay without noticeable degradation in the performance. The proposed system is precise and easy to configure and update the filter coefficient in an efficient manner to achieve the faster convergence.**

Key words : Least Mean Square(LMS) algorithm, Modified Delayed adaptive filter, Partial Product Generator, Weight Update Block.

I. INTRODUCTION

Digital Signal Processing (DSP) made significant contribution in the communication world to increase the speed and reduce the power consumption of the digital circuit. One of the examples of the DSP is filter. 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 removal of unwanted signals through the use of optimization theory is becoming popular, particularly in the area of adaptive filtering. Recently, because of the progress of digital signal processors, a variety of selective coefficient update of gradient-based adaptive algorithms could be implemented in practice. These algorithms minimize the mean square of the error signal, which is the difference between the reference signal and the estimated filter output, by removing unwanted signals according to statistical parameters. The LMS algorithm is the most popular method for adapting a filtering, which have made it widely adopted in many applications. It is used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal (difference between the desired and the actual signal).Applications include adaptive channel equalization, adaptive predictive speech coding, Noise Suppression and on-line system identification. The direct-form LMS adaptive filter involves a long critical path due to an inner-product computation to obtain the filter output. The critical path is required to be reduced by pipelined implementation when it exceeds the desired sample period. Since the conventional LMS algorithm does not support pipelined implementation because of its recursive behavior, it is modified to a form called the delayed LMS(DLMS) algorithm with small modifications to suit VLSI technology[1-2] .

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 ~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, The LMS algorithm used with pipelined and parallel FIR (finite impulse response) filter architectures produces delays in the co-efficient or increases the hardware requirement, decreased by adaptation delays, high performance rate without the dependency on FIR filter length. Hybrid architecture is proposed by combining a transpose – form architecture. Compute the correction term and calculates the delays LMS error [3].The existing system used to modify version of the DLMS algorithm without degrading the convergence characteristics. It results in higher throughput maintenance by using a new look ahead transformation. It results in improving the convergence characteristics though the algorithm suffers large hardware complexity with more delays [4].Lan *et al.*,[5] proposed an efficient systolic and suitable for a single chip realization, which furnish the lowest critical period in the word – level and better convergence without sacrificing finite – driving, area, cost, regularity & local connection characteristics. Higher speed FPGA combined with DLMS is implementation to maintain low latency output. In addition DLMS, and "fine grained" pipelining is implemented with direct form and transpose form it results direct form has high speed, low latency design compared to transpose form [6].Yi *et al.* [7] Proposed fine grained pipelined design of an adaptive filter that supports high sampling frequency but with pipeline depth. The adverse effects of the architecture is that power dissipation increases, adaptation delay increases and convergence performance degrades. Meher *et al.* [8] has been made further effort to reduce the number of adaptation delay. Sang *et al.* [9] proposed a novel pipelined architecture for low power, high throughput and low area adaptive filter based on distributed arithmetic (DA) which gives enhanced throughput by parallel LUT table & concurrent process of filter operation and weight updating. To decreases area complexity, carry – save accumulation is used. The reduction in power consumption is achieved by using fast - bit clock [9].

The existing work on the DLMS adaptive filter does not discuss the fixed-point implementation issues, e.g., location of radix point, choice of word length, and quantization at various stages of computation, although they directly affect the convergence

performance, particularly due to the recursive behavior of the LMS algorithm. Besides, we present here the optimization of the design to reduce the pipeline delays along with area and sampling period and energy consumption. The proposed design is found to be more efficient in terms of the power-delay product (PDP) and energy-delay product (EDP) compared to the existing structures.

III.DELAYED LMS ALGORITHM

The weights of LMS adaptive filter during the nth cycle are updated according to the following equations[2]:

$$W_{n+1}=W_n+\mu e_n X_n \tag{1a}$$

Where

$$e_n=d_n-y_n \quad yn=W^T_n \,.\, X_n \tag{1b}$$

where the input vector X_n and the weight vector W_n at the nth iteration are ,respectively given by

$$X_n=[x_n,x_{n-1},\ldots,x_{n-N+1}]^T$$

$$W_n=[w_n(0),w_n(1),\ldots w_n(N-1)]^T,$$

d_n is the desired response, yn is the filter output, e_n denotes the error computed during the nth iteration. μ is the step-size,and N is thye number of weights used in the LMS adaptive filter.
The LMS algorithm involves a long critical path due to its inner product computation and also due to its recursive nature it does not support the pipelined implementation.So it is modified to form the Delayed LMS adaptive filter.Fig. 1 shows the structure of conventional Delayed LMS adaptive filter.

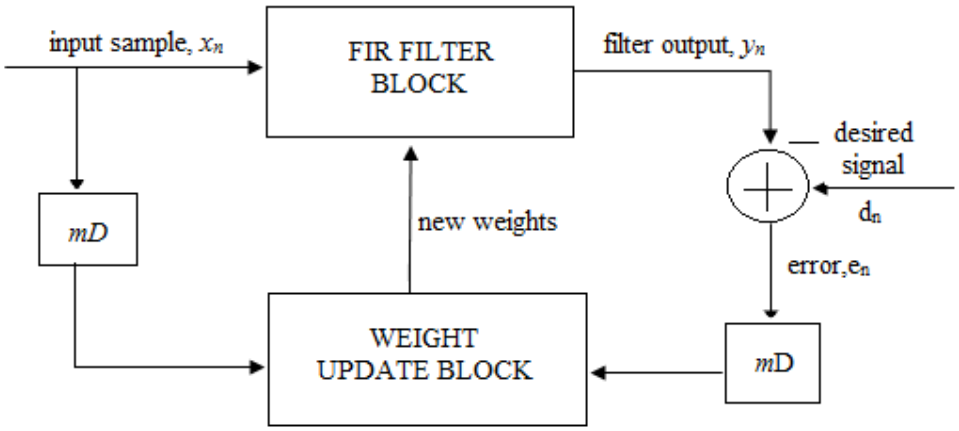

Fig. 1.Structure of Conventional Delayed LMS adaptive Filter.

In case of pipeline design with m pipeline stages ,the error e_n becomes available after m cycles,where m is called the adaptation delay.The DLMS algorithm therefore uses the delayed error e_{n-m},i.e., the error corresponding to (n-m)th iteration for upadating the weights instead of recent most error.The weight update equation of DLMS adaptive filter is given by

$$W_{n+1}= W_n+\mu \,.\, e_{n-m} \,.\, x_{n-m} \,. \tag{2}$$

W_{n+1} is updated weight,W_n is previous weight, μ is step-size, . e_{n-m} is error at the (n-m)th cycle.The adaptation delay of m cycles amounts to the delay introduced by the whole adaptive filter structure consisting of FIR filtering and the weight-update process.It is shown in[12] that the adaptation delay of conventional LMS can be decomposed into two parts:one part is the dealy introduced by pipeline stages in FIR filtering, and the other is due to the delay involved in pipelining the weight-update process.Based on such decomposition of delay ,the DLMS adaptive filter can be implemented by a structure shown in Fig. 2.Assuming the latency of computation of error is n1 cycles,which is used with the input samples delayed by n1 cycles to generate the weight increment term.
The weight update equation of the modified DLMS algorithm is given by

$$W_{n+1}=W_n+\mu \,.\, e_{n-n1} \,.\, X_{n-n1} \tag{3a}$$

Where

$$e_{n-n1} =d_{n-n1}-y_{n-n1} \tag{3b}$$

and

$$y_n=W_{n-n2}{}^T \,.\, X_n \tag{3c}$$

We noticed that ,during weight update, the error with n_1 delays is used, while the filtering unit uses the weight delayed by n_2 cycles.The modified DLMS algorithm decouples computation of the error computation block and weight update block allows us to perform optimal pipeling by feed forward cut set retiming of both these sections to minimize the number of pipeline stages and adaptation delay.

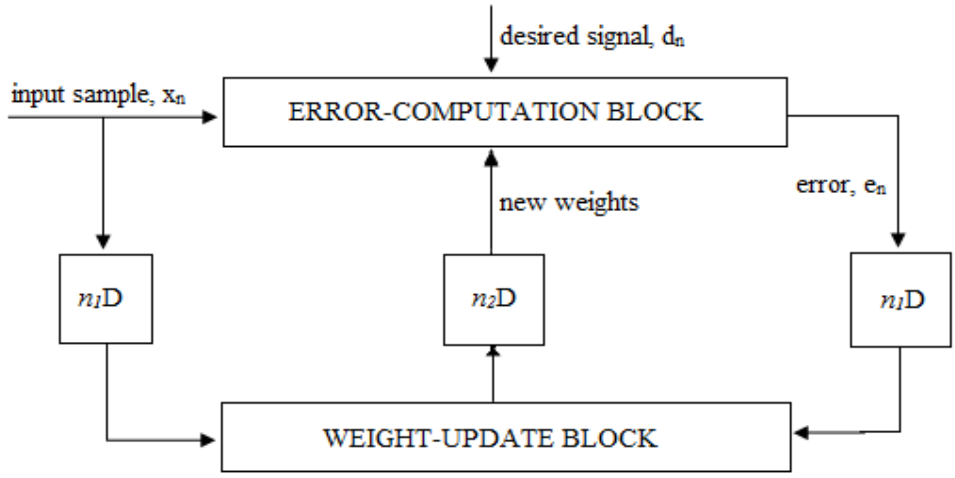

Fig 2. Structure of the modified delayed LMS adaptive Filter.

III.PROPOSED ARCHITECTURE

As shown in Fig. 2, there are two main computing blocks in the adaptive filter architecture : 1)the error-computation blocks, 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.

A. Error-Computation Block

The proposed structure for error-computation unit of an N-tap DLMS adaptive filter is shown in Fig. 3. It consists of N number of 2-bit partial product generators (PPG) corresponding to N multipliers and a cluster of L/2 binary adder trees, followed by a single shift–add tree. Each sub block is described in detail.

1) *Structure of PPG* : The structure of each partial product generator (PPG) is shown in Fig.4.It consists of L/2 number of 2-to -3 decoders and the equal number of AND/OR cells (AOC) Each of the 2-to-3 decoders takes a 2-bit digit (u1u0) as input and produces three outputs b0 = u0. u1, b1 = u0 · u1, and b2 = u0 · u1, such that b0 = 1 for (u1u0) = 1, b1 = 1 for (u1u0) = 2, and b2 = 1 for (u1u0) = 3. The decoder output b0, b1 and b2 along with w, 2w, and 3w are given to an AOC, where w, 2w, and 3w are in 2's complement representation and sign-extended to have (W + 2) bits each.

2) *Structure of AOCs:* The structures of an AOC, as shown in Fig 5, consists of three AND cells and two OR cells. Each AND cell takes an n-bit input and a single bit input b, also consists of n 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. 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). The decoder along with the AOC performs 2-bit multiplication and L/2 parallel multiplications with a 2-bit digit to produce L/2 partial products of the product word.

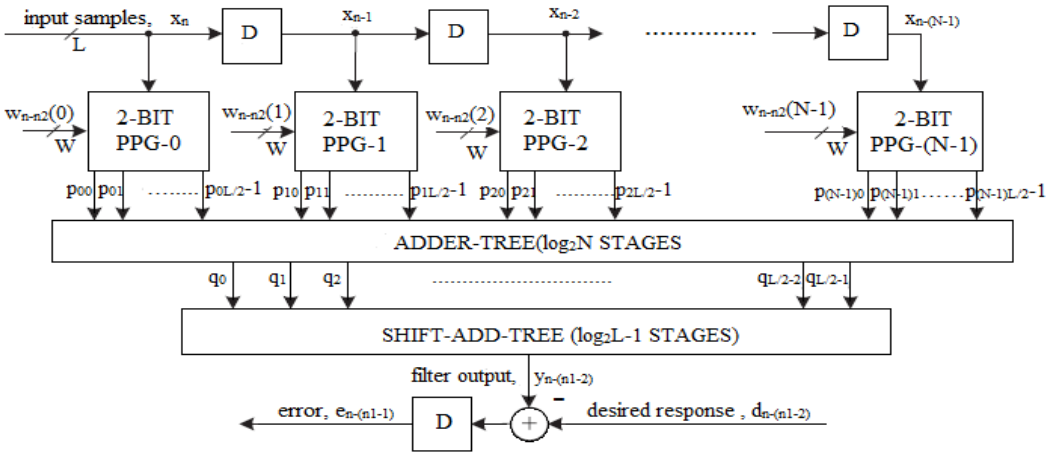

Fig 3.Proposed structure of the error-computation block

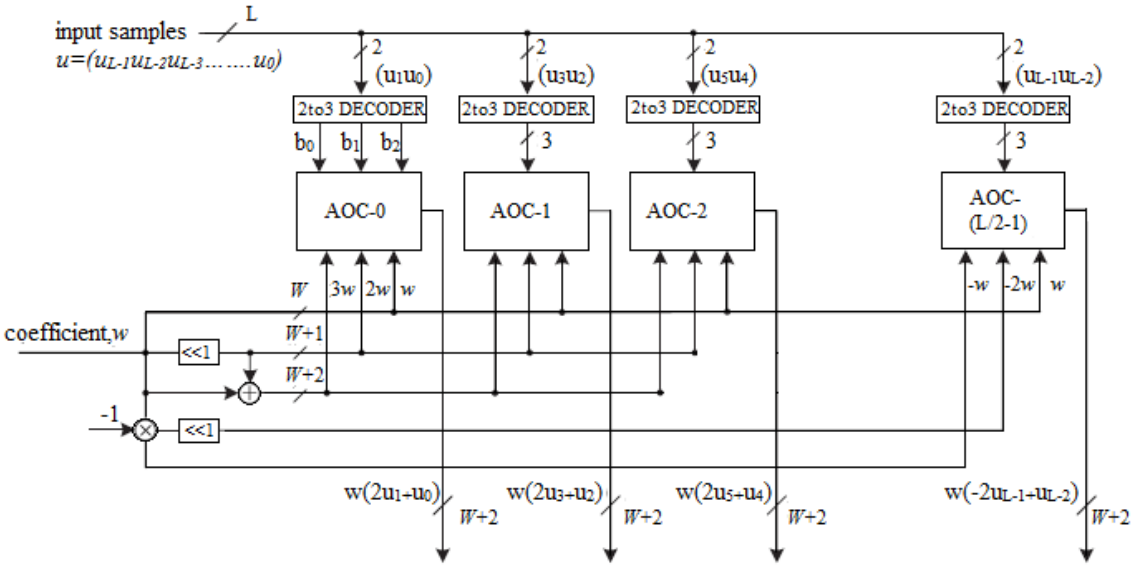

Fig 4.Partial Product Generator

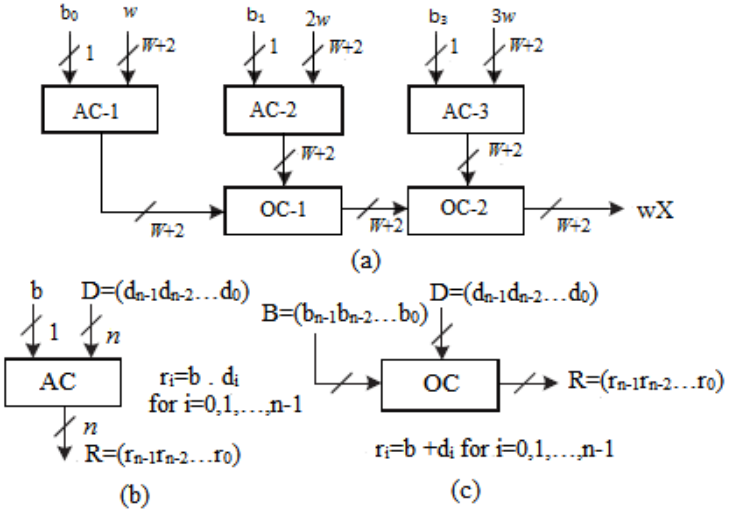

Fig 5.Structure and Function of AND/OR cell

3) *Structure of Adder Tree:* The shifts-add operation on the partial products of each PPG gives the product value and then added all the N product values to compute the inner product output. However, the shift-add operation obtains the product value

which increases the word length, and the adder size. To avoid increase in word size of the adders, we add all the N partial products of the same place value from all the 'N' PPGs by a single adder tree. All the partial products generated by all PPGs are thus added by binary adder trees. The outputs of adder trees are then added by shift-add tree according to their place values. The addition scheme for the error computation block for a four tap filter is shown in fig. 6.we have shown all possible locations of pipeline latches by dashed lines to reduce the critical path to one addition time. If we introduce pipeline latches after every addition it requires more number of latches which would lead to a adaptation delay. The final adder in the shift-add tree contributes to the maximum delay to the critical path. Based on that observation, Table I**,** shows the possible pipeline latches for various filter lengths.

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 Stage-2	Stage-1
16	Stage-3	Stage-1 and Stage-2	Stage-1
32	Stage-3	Stage-1 and Stage-2	Stage-2

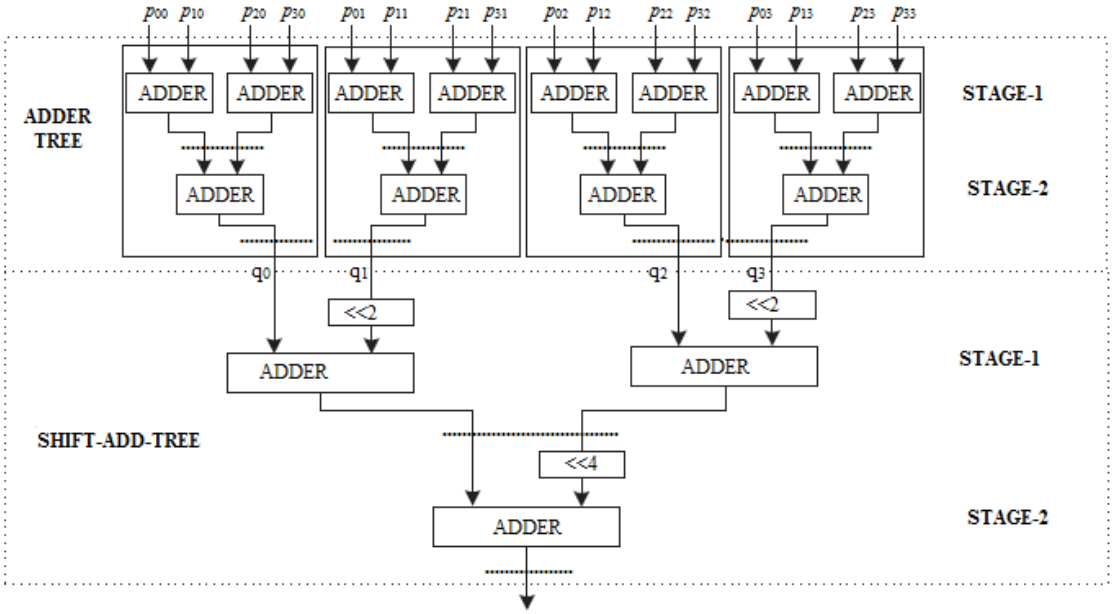

Fig 6.Adder Structure of the filter unit for N=4 and L=8

Adder-Tree Optimization

The adder tree for the computation of filter output can be pruned for further optimization of area delay and power complexity.To illustrate the proposed pruning optimization of adder tree and shift-add tree for the computation of filter output,we take the simple example of filter length N=4, considering the word lengths L and W to be 8.The dot diagram contains 10 dots which represents the partial products generated by the PPG unit.We have four sets of partial product corresponding to four partal products of each multiplier,Since L=8.Each set of partial product of the same weight values contain four terms,Since N=4.The fina;l sum without truncation should be 18 bits.however we use only 8 bits in final sum,and the rest 10 bits are finally discarded.To reduce the computational complexity,some of the LSBs of inputs of adder tree can be truncated,while some guard bits can be used to minimized the impact of truncation on the error performance of the adaptive filter

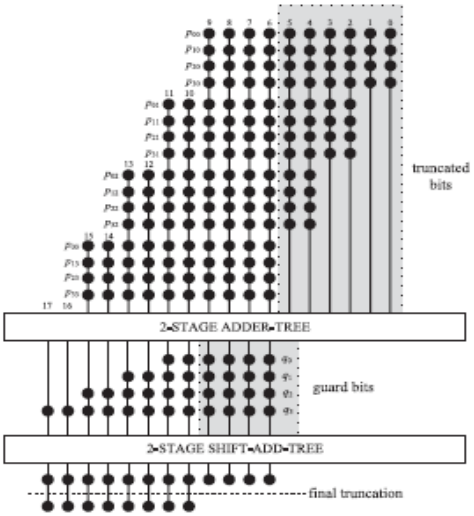

Fig. 7.Dot-diagram for optimization of adder tree in the case ofN=4,L=8,and W=8

B. Structure Of Weight-Update block

The proposed pipelined structure of weight-update block is shown in Fig.6. It performs 4 multiply-accumulate operations of the form $(\mu \times e) \times x_i + w_i$ 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 by the shift operation. Each MAC unit performs the multiplication of the shifted value of error with the delayed input samples x_i followed by the additions with the corresponding old weight values w_i. All 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 μ× e with the number of 2-bit digits of the input word xi. The sub expression can be shared across all the multipliers. This leads to a gradual reduction of adder in complexity. The final outputs of MAC units constitute updated weights to be used as inputs to the error -computation block and the weight-update block for the next iteration. Fig .7 shows the weight update block with N=4.It updates the filter coefficients and applied to the error computation block for generating next error.

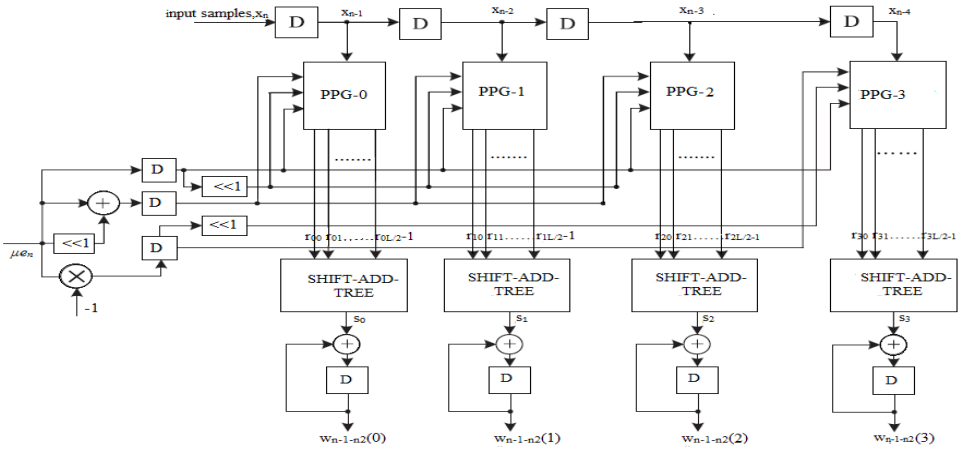

Fig 8.Structure of the weight-update block

IV.EXTENDED WEIGHT-UPDATE BLOCK

With the architecture details of 4-tap weight update block, to increase the performance of the adaptive filter, to a further extent, we have increased the taps of the filter.For this,we have added an another set of 4 taps which can accommodate four more MAC operations.We have added 4 more filter coefficients to peform more MAC operations. Fig. 8 shows the architecture of Extended weight update block. The performance of the 8tap adaptive filter depends on the speed of convergence.The speed of convergence is equal to the number of iterations it takes to reach finite solution.By performing weight updations with extended weight update block,the filter converges at faster rates.

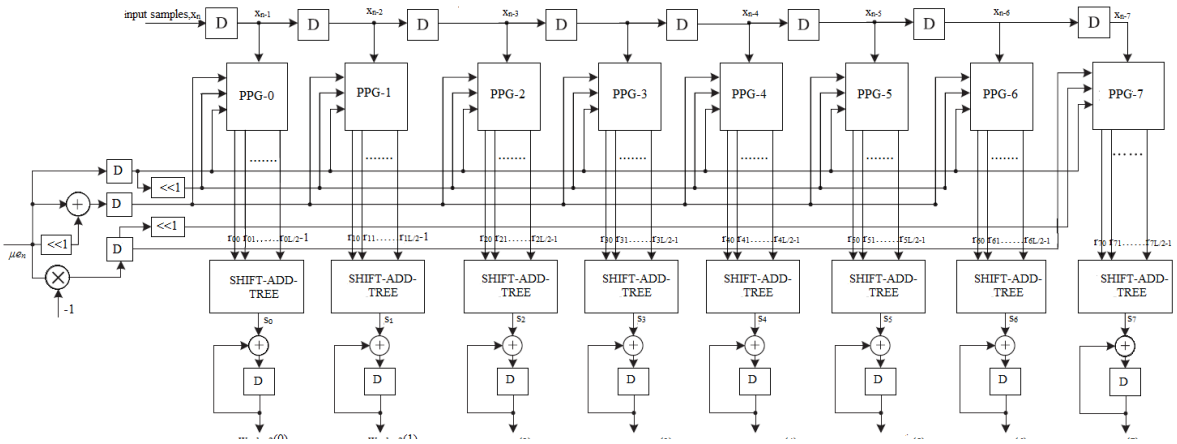

Fig. 9.Extended weight update block

V. Fixed-Point Design Considerations

For fixed -point implementation, the choice of word lengths and radix points for input samples, weights, and internal signals need to be decided. given as the input. For this purpose, the specific scaling/sign extension and truncation/zero padding are required. Since the LMS algorithm performs learning so that y has the same sign as d, the error signal e can also be set to have the same representation as y without overflow after the subtraction. LSBs of weight increment terms are truncated so that the terms have the same fixed-point representation as the weight values. We also assume that no overflow occurs during the addition for the weight update. Otherwise, the word length of the weights should be increased at every iteration, which is not desirable. The assumption is valid since the weight increment terms are small when the weights are converged. MSBs in the computation of the shift–add tree of the weight-update circuit are to be retained, while the rest of the more significant bits of MSBs need to be discarded. This is in accordance with the assumptions that, as the weights converge toward the optimal value, the weight increment terms become smaller, and the MSB end of error term contain more number of zeros.

VI.SIMULATION AND RESULT

The proposed design was coded in Verilog and synthesized. The word-length of input samples and weights are chosen to be 16, with 16 bit internal representation. The convergence-factor was chosen to be a negative power of two to realize its multiplication through a simple shift operation. LSBs of the error and the weights are truncated to keep the word-length restricted to 16 bit. The structure of and the best of systolic structures were also coded in Verilog, similarly, and synthesized using the same tool.

A modified DLMS adaptive algorithm to achieve less adaptation-delay compared with the conventional DLMS algorithm, and shown that the proposed DLMS algorithm can be implemented efficiently by a pipelined inner product computation unit and parallel and pipelined weight update unit using carry-save reductions. Substantial reduction of adaptation-delay, ADP and EPS over the existing structures had been achieved by the proposed design. For 16-bit word length and filter order 8 the adaptation-delay of proposed structure is 3 cycles, which increases only by one cycle when the filter order increases by double times. Therefore, it can be used for large order adaptive filters as well. We are trying for further optimization of the structure by using suitable compressors.

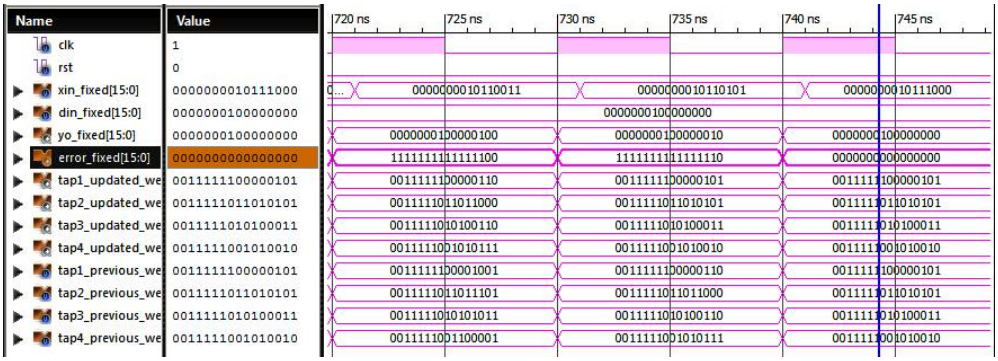

Fig 8.Simulation of modified adaptive filter with N=4

When16 bit input signal along with desired signal applied to an pipelined structured adaptive filter with N=4, initially the error will be more at initial clock cycles. but when we go on filtering, the error signal becomes completely zero and filteroutput converged at 74th clock cycle. Fig. 8 shows the simulation results with four taps.

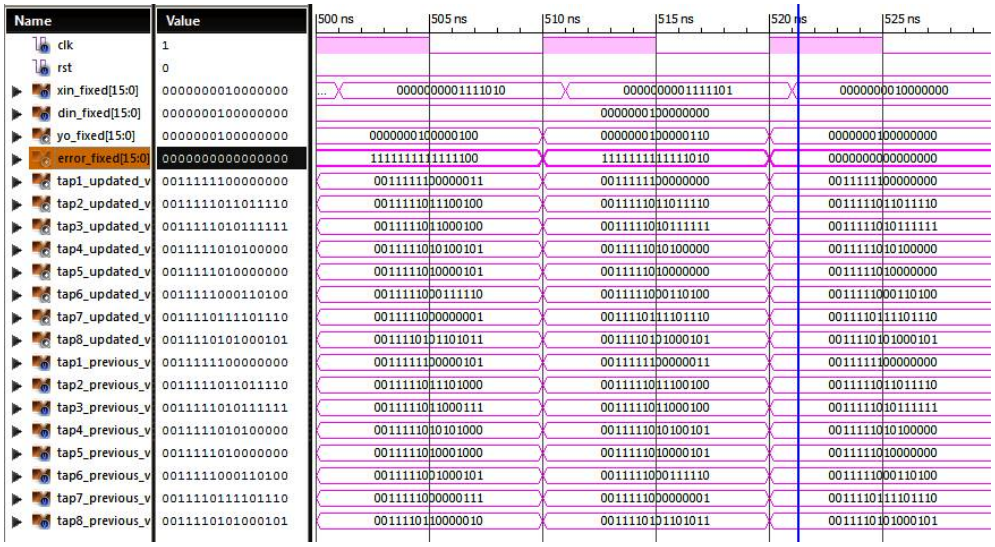

Fig. 9.Simulation of Modified adaptive filter with N=8

When16 bit input signal along with desired signal applied to an pipelined structured adaptive filter with N=8, initially the error will be more at initial clock cycles. but when we go on filtering, the error signal becomes completely zero and filter output converged at 52nd clock cycle. Fig 9.shows the simulation results with eight taps DLMS adaptive filter..we found that the adaptive filter with N=8 converges at higher rates without noticeable degradation in the adaptation delay.Table II shows the delay comparison of conventional with the proposed DLMS adaptive filter.

TABLE II
DELAY COMPARISION TABLE

S No	Method	Adaptation Delay
1.	Conventional DLMS Adaptive Filter	20.218ns
2.	DLMS Adaptive Filter	12.392ns
3.	Modified DLMS Adaptive Filter	12.430ns

VII.CONCLUSION

In this paper, we implemented an effiecient low adaptation delay architecture in fixed-point implementation .We used a novel partial product generator for efficient implementation of inner product computation.We have proposed an efficient scheme for inner product computation to reduce the adaptation dealy significantly to achive faster convergenceand reduce the critical path to support high sampling rates. Aside from this,we proposed a strategy for the optimized balanced pipelining across the time consuming blocks of the structure to reduce the adaption delay and power consumption as well. We were also implemented the proposed design in xilinx devices and simulated the adaptive filter with four taps and eight taps.We found that the adaptive filter with extended weight update block converges at faster rates without noticeble degradation in the adaptation delay and also achieved high performance.

VIII.REFERENCES

[1] G. Long, F. Ling and J.G. Proakis. 1989. “The LMS algorithm with delayed coefficient adaptation,” IEEE Trans. On Acoustic, speech & Signal processing. Vol. 37, pp. 1397-1405, September.

[2] Corrections to ‘The LMS algorithm with coefficient adaptation’,” IEEE Trans. On Signal Processing, vol. 40, pp 230- 232,

January 1992.

[3] Scott C.Douglass, Quanhong Zhu and Kent F. Smith. 1998. "A Pipelined LMS Adaptive FIR Filter Architecture without Adaptation Delay", IEEE Trans. On Signal Processing. Vol. 40, No.3, pp. 775 – 779, March.

[4] Katsushige Matsubara, Kiyoshi Nishikawa and Hitoshi Kiya. 1999. "Pipelined LMS Adaptive Filter Using a New Look – Ahead Transformation ", IEEE Trans. On Circuits and Systems-II, pp. 51-55. Vol.46, No.1, January.

[5] Lan and Feng. 2001. "An Efficient Systolic Architecture for the DLMS Adaptive Filter and Its Applications", IEEE Trans. On Circuit and Systems – ii:Analog and Digital signal Processing.vol.48,n0.4,April 2001

[6] L.-K. Ting, R.Woods and C.F.N. Cowan. 2005. " Virtex FPGA implementation of a pipelined adaptive LMS predictor for electronic support measures receivers", IEEE Trans. Very Large Scale Integrated (VLSI) Syst. Vol. 13, No.1, pp. 86 – 99, January.

[7] Y.Yi.R.Woods, L.-K. Ting, R.Woods and C.F.N. Cowan. 2005. "High speed FPGA based implementations of delayed – LMS filters," J. Very Large Scale Integr. (VLSI) Signal Process., vol. 39, No.1-2, pp. 113 –131, Jan 2005.

[8] P.K. Meher and M .Maheshwari. 2011. "A high – Speed FIR Adaptive filter architecture using a modified delayed LMS algorithm", in Proc. IEEE Int. Symposium Circuit Syst., May. pp. 121 – 124.

[9] Sang Yoon Park and Pramod Kumar Meher. 2013."Low – Power, High – Throughput and low – Area adaptive FIR Filter Based on Distributed Arithmetic" – IEEE Trans. On Circuit and Systems – ii: express briefs. Vol. 60, No.6, June.

[10] Pramod Kumar Meher and Sang Yoon. 2014. "Critical – Path Analysis and Low complexity Implementation of the LMS Adaptive Algorithm– IEEE Trans. On Circuits and Systems. Vol. 61, No.3, March.

[11] K. K. Parhi. 1999. VLSI Digital Signal Processing Systems: Design and Implementation. New York, USA: Wiley.

[12] S. Haykin. 1991. Adaptive Filter Theory. Prentice Hall.

[13] S. Haykin and B.Widrow. 2003. Least – Mean – Square Adaptive Filters. Hobokin, NJ, USA: Wiley.