ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Innovative Approach Architecture Designed For Realizing Fixed Point Least Mean Square Adaptive Filter with Less Adaptation Delay

D.Durgaprasad
Department of ECE, Swarnandhra College of Engineering & Technology, A.P, India.

J.E.N.Abhilash
Department of ECE, Swarnandhra College of Engineering & Technology, A.P, India.

S. Srilali
Department of ECE, Swarnandhra College of Engineering & Technology, A.P, India.

Abstract:
In this work, we propose a new Architecture for Delayed Least Mean Square Adaptive filter. In this proposed architecture, we used a novel partial product generator in place of multipliers to compute the inner product by sharing a common sub expression. The adaptation delay is significantly reduced by using an efficient addition scheme in computation of inner product. The adaptation delay and power dissipations are further reduced by introducing optimized balanced pipelining across the time consuming blocks of the structure. The fixed point representation is used for signals that can be involved in proposed architecture. From the synthesis results, the proposed design yields less area delay product and power delay product when compared with existing designs.

Keywords: ***Adaptive filter (AF), Adder tree optimization, Fixed point representation, Least mean square (LMS) algorithm.***

1. INTRODUCTION

An adaptive filter is a computational device. The adaptive filters are used to made the relationship between two signals in an iterative manner. The adaptive filters can be implemented as a set of program instructions running on arithmetical devices such as digital signal processing chip or microprocessor. The adaptive filters are also realized as a set of logic operations in FPGA (or) in custom very large scale integrated circuit. The errors due to the numerical precision effects are ignoring in these implementations, but adaptive filter fundamental operation can be characterized separately in specific physical realization. Because of this specific realization of adaptive filter in hardware (or) software, we can concentrate on the mathematical forms of adaptive filter.

The adaptive filter has several adaption algorithms. In which the Least mean square algorithm (LMS) is used in different applications of adaptive filter because of its computational simplicity. In order to obtain the stability, the range of the convergence factor must be decided by observing the convergence characteristics of the least mean square algorithm.

The least mean square algorithm contains two processes. They are filtering process and adaptive process.
1. Filtering process: In this process, an input signal is applied to the filter block in the adaptive filter structure that generates corresponding output signal. Calculating the error signal by comparing the calculated output signal with applied desired signal. The calculated error signal is applied to the weight update block in adaptive filter.

2.Adaptive process: In this process, the weight coefficients of the weight update block in adaptive filter structure are calculate using input signal and error signal. The calculated weight coefficients are adapted to the filter block in adaptive structure.

The general structure of the adaptive filter is shown in Fig.1

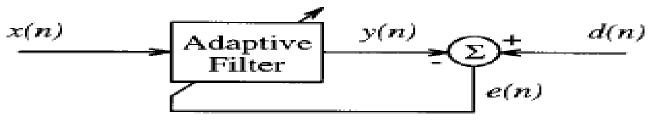


Fig.1: General Structure of the Adaptive filter

In the Fig.1 x(n) is input signal, y(n) is output signal, d(n) is desired response signal and e(n) is error signal. The input signal is applied to the adaptive filter structure that calculates the corresponding filter output signal. The error signal is computed by comparing the calculated output signal with applied desired response signal. The error signal equation is given as

$$e(n) = d(n) - y(n)$$

The calculated error signal is considered into adaptation procedure, which adjusts the filter parameters from time instant n to time instant (n+1). This adaptation procedure is represented with oblique arrow in Fig.1.The output of the adaptive filter is better and better match to the applied desired response signal when the time instant n is raised in adaptation process. Due to the increment of time index n the magnitude of e(n) decreases over time. In this “better” means the parameters of the adaptive filters are varied in well defined manner with popular adaptation algorithms.

In adaptive filtering problem, the adaptation means the parameters of the system are adjusted from time instant n to time instant (n+1). The structure of the adaptive filter decides the number and types of parameters of the adaptive filter.

2. RELATED WORKS

Many works has been done to implement the delayed least mean square (DLMS) algorithm in systolic architecture to increase the usable frequency [4],[7],[8].The existing structures of adaptive filters have N cycles adaptation delay for filter length N, this is high for higher order filters. The convergence performance decreases because of large adaptation delay, so Visvanathan et al (9) modifies the existing systolic architecture to maximize the convergence performance by reducing the adaptation delay. L.K .Ting and C.F.N. Cowan have proposed transpose form least mean square adaptive filter, in which the output of the filter depends on delayed weight coefficients. In order to reduce the adaptation delay ,Van and Feng[11] introduce large processing elements(PEs) in systolic architecture .Ting et al [12] have introduce fine grained pipelined design in adaptive filter structure to reduce the critical path to one addition, this can supports higher sampling frequency. This design [12] have large area for pipelining and large power dissipation than in [11] because of using large number of pipeline latches. Meher and Maheshwari [13] have been put effort to reduce the adaptation delay. Meher and Park introduce 2-bit multiplication cell and efficient adder tree in adaptive filter structure to reduce the critical path and area without increasing the adaptation delay [14],[15].

3. DELAYED LMS ADAPTIVE FILTER

The Fig.2 shows the structure of a direct form least mean square adaptive filter

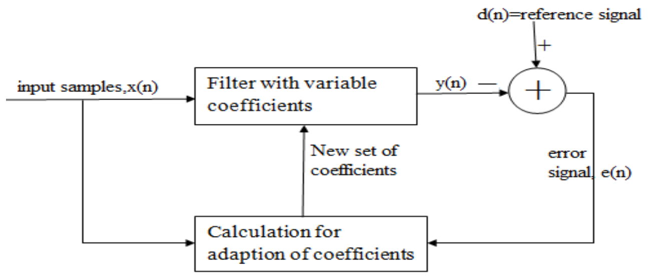


Fig.2: Structure of a direct form LMS Adaptive filter

In the Fig.2 x(n) is input signal, y(n) is output signal, d(n) is desired response signal and e(n) is error signal. The LMS algorithm generates the output signal for applied input signal and compares the calculated output signal with desired response signal then the resulting signal is called error signal. For every cycle the filter weights are updated with the help of obtained error signal. At n^{th} iteration the weight coefficients of LMS adaptive filter are updated based on the following equations

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \cdot e_n \cdot \mathbf{x}_n$$

Where

$$e_n = d_n - y_n \quad y_n = \mathbf{w}_n^T \cdot \mathbf{x}_n$$

In the above expressions x_n and w_n represents the input vector and weight vector. At nth iteration x_n and w_n is given as

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

d_n represents the desired response signal, y_n represents filter output signal ,e_n represents the error signal at nth iteration ,the number of weight coefficients are represented with N and µ represents the step size.

The computation of inner product with the direct form least mean square adaptive filter requires long critical path. The pipelined implementation is used to reduce the critical path size, but the pipelined implementation is not support in direct form least mean square adaptive filter, so that we can introduce delay in the adaptive filter structure.

The conventional structure of delayed least mean square adaptive filter structure is shown in Fig.3.

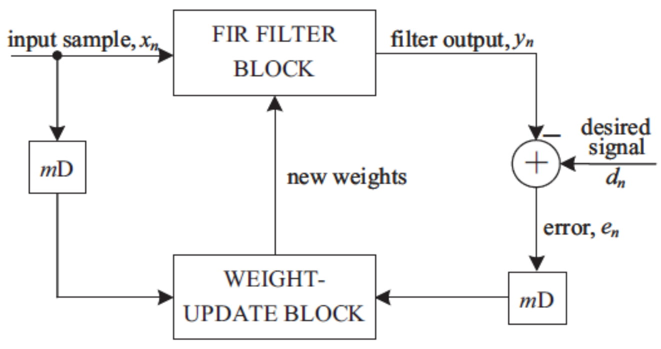


Fig.3: The conventional structure of DLMS adaptive filter

The Fig.3 uses m pipeline stages, so the error signal e_n is available after m cycles. Here m is called the adaptation delay. The delayed error signal e_{n-m} is used in delayed least mean square adaptive filter. The weight coefficients update equation for DLMS adaptive filter at (n-m)th iteration is given as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \cdot e_{n-m} \cdot \mathbf{x}_{n-m}$$

Where the adaptation delay is m cycles.

4. PROPOSED ARCHITECTURE

The conventional structure of delayed LMS adaptive filter consists m cycles adaptation delay that amounts to the total delay introduced by the entire adaptive filter structure, i.e delay introduced by FIR filtering block and the weight update block. So that the adaptation delay is divided into two parts. One part is the pipeline stages generated delay in finite impulse response (FIR) filter block and the other part is pipelining generated delay in weight update block.

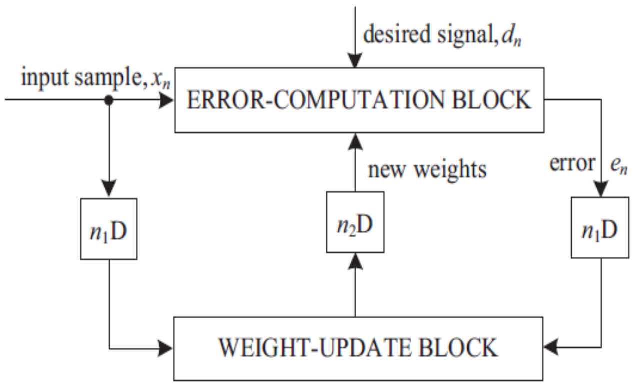


Fig.4: Modified delayed least mean square adaptive filter structure

Depending on the decomposition of delay, the conventional delayed least mean square adaptive filter structure is modified and that can be shown in Fig.4. The latency of computation of error signal is assumed to be n_1 cycles. The structure calculated the error at nth cycle is denoted with $e_{(n-n1)}$. The weight increment terms are generated by using n_1 cycles delayed input samples along with calculated error signal. The modified DLMS adaptive filter weight update equations are given by

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \cdot e_{n-n_1} \cdot \mathbf{x}_{n-n_1}$$

Where

$$e_{n-n_1} = d_{n-n_1} - y_{n-n_1}$$

$$y_n = \mathbf{w}^T_{n-n_2} \cdot \mathbf{x}_n.$$

From the above equation, In weight coefficients update process n_1 cycles delayed error and input signals are used. The updated weights are used in filtering block after n_2 cycles delay. The modified DLMS adaptive filter structure contains two main blocks. They are

- Error computation block
- Weight update block

Now, we can discuss the design strategy of the existing and proposed structures of error computation block and weight update block.

A. Proposed Structure of the Error Computation Block

The Fig.5 shows the Proposed Error Computation Block structure of the N-tap delayed least mean square adaptive filter. The Proposed structure consists the N number of partial product generators (PPG) corresponding to N multipliers and L/2 number of binary adder trees, shift add tree with $\log_2$ L-1 stages.

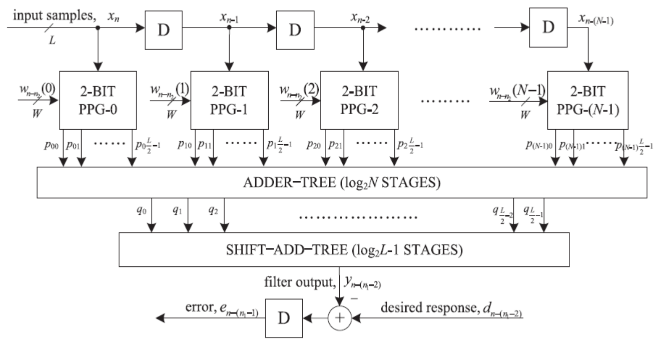


Fig.5: Proposed Structure of the Error Computation Block

i) Structure of Partial Product Generator (PPG)

The Fig.6 shows the proposed structure of each Partial product generator (PPG). The proposed structure of PPG contains the L/2 number of 2 to 3 decoders and L/2 number of AND/OR cells (AOC). The 2 bit digit (u_1u_0) is applied as input to the 2 to 3 decoder and it can generate three outputs. The generated outputs are $b0=u_0\cdot(u_1bar),b_1=(u_0\ bar)\cdot u_1$ and $b_2=u_0\ \cdot u_1$. The output $b_0=1$ for 2 bit digit $u_1u_0=1$, the output $b_1=1$ for 2 bit digit $u_1u_0=2$ and the output $b_2=1$ for 2 bit digit $u_1u_0=3$. The outputs b_0,b_1 and b_3 are applied to the AND/OR cell along with w,2w and 3w. Here w,2w and 3w are represented in 2's complemented form and sign extended to (W+2) bits each. The sign of the input samples considered carefully while calculating the partial product of the most significant digit (MSD) ,such as $u_{L-1}u_{L-2}$ digits of the input sample. So the w,-2w, and –w are applied to the $(L/2-1)^{th}$ AND/OR cell (AOC) because the digits $u_{L-1}u_{L-2}$ have four possible values 0,1,-2,-1.

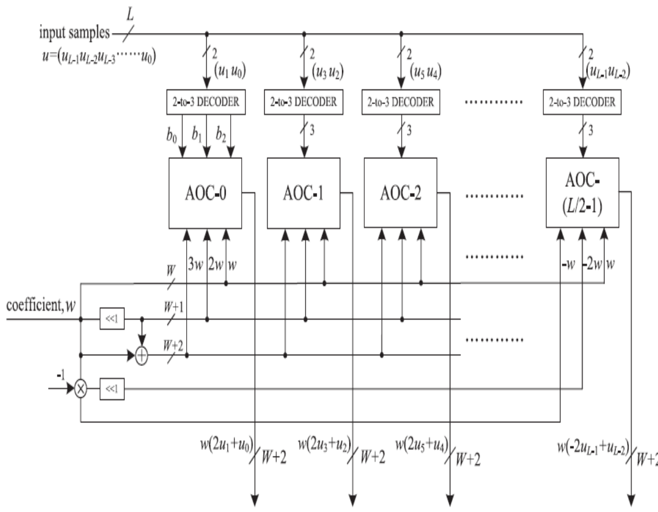


Fig.6: Proposed structure of PPG

A) Structure of AND/OR cells (AOCs):

The Fig.7 shows the structure and function of the AND/OR cells. The structure of each AND/OR cell contains the 3 AND cells and 2 OR cells. The Fig.7 (b) shows the structure and functions of AND cell. Each AND cell contains n number of AND gates and takes single bit input b, n-bit input D as inputs. Apply the n bit input D to the one of the input of n number of AND gates and the single-bit input b is applied to the other input of all the n AND gates. The Fig.7 (c) shows the structure and functions of OR cell. Each OR cell contains n number of OR gates and takes two n-bit inputs B and D. The bits in the same position of B and D is applied to the same OR gate.

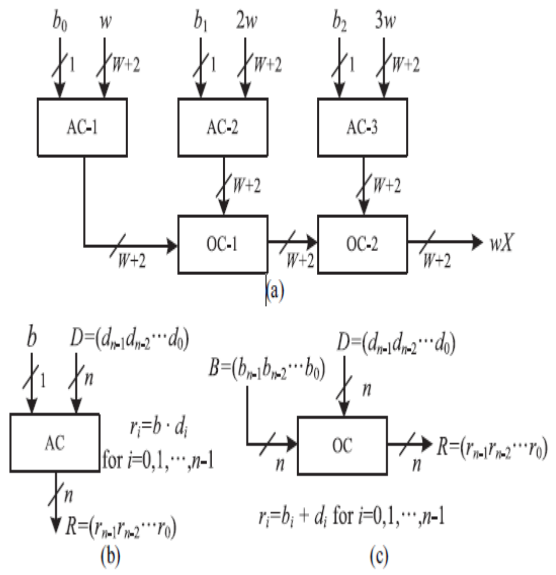

b_0
w
b_1
$2w$
b_2
$3w$
1
$W+2$
AC-1
AC-2
AC-3
OC-1
OC-2
wX
(a)
b
$D=(d_{n-1}d_{n-2}\cdots d_0)$
$B=(b_{n-1}b_{n-2}\cdots b_0)$
n
AC
$r_i=b\cdot d_i$
for $i=0,1,\cdots,n-1$
OC
$R=(r_{n-1}r_{n-2}\cdots r_0)$
$r_i=b_i+d_i$ for $i=0,1,\cdots,n-1$
(b)
(c)

Fig. 7: Structure and function of AND/OR cells.

The w, 2w and 3w are the outputs of the AND/OR cell corresponding to the 2-bit digit input (u_1u_0) decimal values 1,2 and 3. The multiplication of input operand w with a 2-bit digit (u_1u_0) is performed by AOC along with decoder. Such that, the L/2 parallel multiplications of input operand w with 2 bit digit is performed using partial product generator (PPG) and produce L/2 number of partial products of the product word wu.

ii) Structure of Adder Tree:

Generally, the shift add operation is performed on PPG generated partial product values to get the product value and calculate the desired inner product by adding all the N product values. The word length of the product value increases because of the shift add operation and this can also increases the size of adder that can perform N-1 additions of the product values. To reduce the size of the adder ,we can perform addition operation on all the N partial products of the same place value from all the partial product generators with one adder tree.

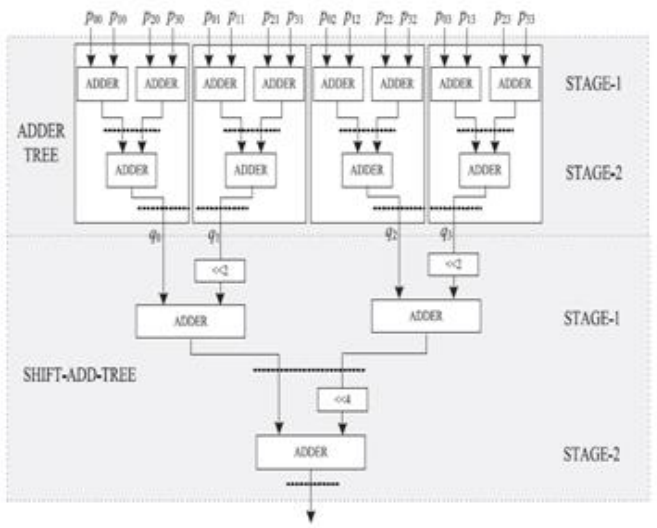


Fig.8: Structure of the Adder Tree

Each of the N partial product generators generates L/2 number of partial products and these can be added by L/2 number of adder trees. The shift add tree perform the addition operation on the outputs of the L/2 binary adder trees according to their place values. The $\log_2$ N stages of adders are required by adder tree to add N partial products. The $\log_2$ L-1 stages of adders are required by shift add tree to add the outputs of the binary adder trees. The Fig.8 shows the error computation block addition system for 4 tap filter and input word length 8,i.e N=4 and L=8.The addition scheme with N=4 and L=8 requires 4 binary adder trees of two stages each and shift add tree with two stages. The pipeline latches are used to reduce the critical path to one addition, so the pipeline latches possible locations are shown with dashed lines in Fig.8. After every addition, we can introduce pipeline latches, so $\log_2$ N+$\log_2$L-1 stages requires L(N-1)/2+L/2-1 latches. For higher values of N and L, these latches introduce high adaptation delay and large area, more power dissipation. In some cases the Pipeline latches are not needed to maintain the critical path to one addition time. Depending on these observations, we can remove the pipeline latches at unwanted locations without increasing of critical path. The Table I shows the possible locations of pipeline latches for filter lengths N=4, 8 and 16, input word size L=8.

Table I: Locations of Pipeline latches for filter lengths N=4, 8, 16 and input sample L=8

N	Error-computation Block		Weight-update Block
	Adder Tree	Shift-add Tree	Shift-add Tree
4	Stage-2	Stage-1 and 2	Stage-1
8	Stage-2	Stage-1 and 2	Stage-1
16	Stage-3	Stage-1 and 2	Stage-1

B. Proposed Structure of the Weight Update Block

The Fig.9 shows the proposed structure of the Weight Update Block. In order to update N filter weights, the N number of multiply accumulate operations of the form $(\mu \times e) \times x_i + w_i$ are performed with weight update block structure. The convergence factor μ is considered as negative power of 2. The multiplication of step size with recently available error is performed by a shift operation. Now each of the MAC units performs the multiplication of delayed input samples with shifted value of error and add previous weights to the multiplication result. The N number of PPGs and shift adder trees performs the N multiplications of the MAC. The L/2 partial products are generated by each of the partial product generator that corresponding to the 2-bit digits of the input samples x_i and the product of the recently shifted error value μe. Here with in the multiplier the sub expression 3μe is shared because in weight update block the scaled error is multiplied with the entire N delayed input values. This sub expression is shared across all the multipliers, this can reduces the complexity of the adder. The outputs of the weight update block are the desired updated weights that can be used as inputs to the error computation block and weight update block for next iteration.

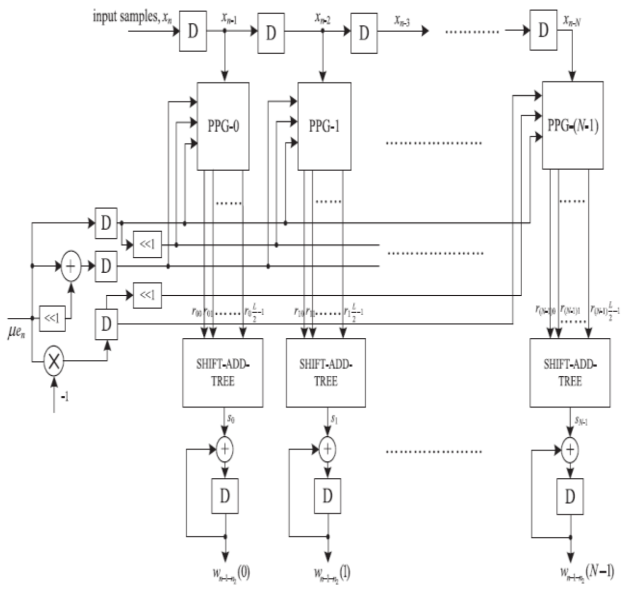


Fig.9: Proposed structure of the weight update block

C. Adaptation Delay

The adaptation delay is divided into n_1 and n_2 as shown in Fig.4. The delayed error generated by error computation block in Fig.5 takes n_1-1cycles. After scaling the delayed error by μ is applied to the weight update block in Fig.9. The input samples are delayed by one cycle before the PPG, so the FIR filtering introduces the delay n_1. The proposed weight update block obtains w_{n-1-n2}, so the weights are delayed by n_2+1 cycle. But the latch before the PPG introduces one cycle delay that can be considered in the delay of the error computation block. So the that delay introduced by weight update block is n_2.The pipeline latches locations are decided based on Table II, The n_1 value is 5. Where the error computation block as three latches, one latch is before the Partial product generator in Fig.9 and other latch is after the subtraction in Fig.5. The value of n_2 is 1, the latch in the shift add tree of the weight update block.

5. Fixed Point Representation

The choice of radix point and word lengths of the input samples, weights and internal signals required to be decided in fixed point implementations. In Fig.10 the binary number is represented in fixed point format.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

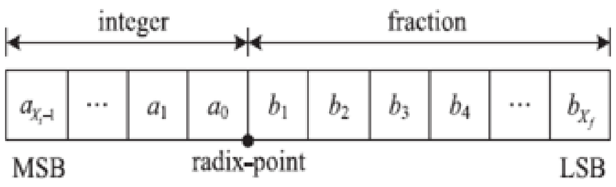


Fig.10: The Fixed Point Representation of a binary number

Let us consider (X, X_i) is the fixed point representation of a binary number, here X represents the word length and X_i represents the integer length. The design constraints desired accuracy and hardware complexity is considered by the hardware designer to decide the location of the radix point and word length of x_n and w_n in Fig.9. Let us assume (L, L_i) and (W, W_i) are the fixed point representation of the input signals and filter weights. The fixed point representation of all the other signals in Fig.5 and Fig.9 can be decided as shown in Table II.

The output of the PPG block is signal p_{ij} its value is three times the value of input coefficients. So we can avoid overflow by adding two more bits to the integer length and word length of the coefficients. In Fig.8 each stage output size in adder tree is one bit more than the input signal size. So the output of the adder tree in fixed point representation is given as (W+$\log_2$N+2, W_i+$\log_2$+2). The fixed point representations of the output of shift add tree is of the form (W+L+$\log_2$, W_i+L_i+$\log_2$N), Assume the least significant bits in the shift add tree or in the adder tree are not truncated. The shift add tree output has W number of bits only, so the W most significant bits required to be considered out of (W+L+$\log_2$N) bits. So that the fixed point representation for output signal y is given as (W, W_i+L_i+$\log_2$). The fixed point representation of desired signal is same as the output signal representation, usually its quantization is given as input. Because of this reason truncation/zero padding and scaling/sign extension are needed. The LMS algorithm is used, so that the sign of output signal y and desired signal d is same. The fixed point representation of the error signal e_n is also same as the representation of output signal y.

Table II: Fixed point representation of the signals of the proposed DLMS adaptive filter

Signal Name	Fixed-Point Representation
X	(L,L_i)
W	(W,W_i)
P	(W+2,W_i+2)
Q	(W+2+$\log_2$N,W_i+2+$\log_2$N)
y,d,e	(W,W_i+L_i+$\log_2$N)
μe	(W,W_i)
R	(W+2,W_i+2)
S	(W,W_i)

The DLMS adaptive filter convergence with adaptation delay of n_1 cycles will be make sure if

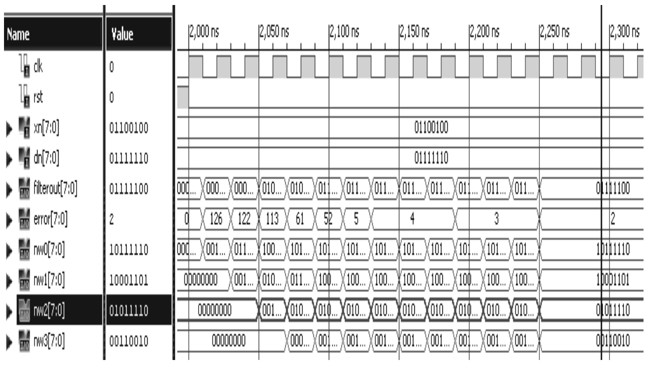
$$0 < \mu < \frac{2}{(\sigma_x^2(N-2)+2n_1-2)\sigma_x^2}$$

Where σ_x^2 represents the average power of input samples. The step size value μ is defined as negative power of 2(2^{-n}) , where n ≤ W_i+L_i+$\log_2$N . The radix point location is changed by performing the multiplication with μ. Since the arithmetic operations are not needed for the multiplication with μ and the truncation error is not introduced. We try to use minimum step size, i.e. n> W_i+L_i+$\log_2$N, truncate the some of the least significant bits of the error signal e_n. Let us assume that n=L_i+$\log_2$N, i.e,$\mu=2^{-(L_i+\log_2 N)}$.The fixed point representation of μe_n is (W, W_i). The weight increment term s is equal to $\mu e_n x_n$, its fixed point representation is (W+L, W_i+L_i). But the shift add tree in weight update block consider only W_i most significant bits in computations and discard the rest of the most significant bits. There is an assumption that weights converge toward the optimum value, decreasing the weight increment terms and more number of zeros are contained in MSB end of error signal. We can also perform truncation on L-L_i least significant bits of the weight increment terms, so that the weight values have the same fixed point representation. We can also assuming no overflow occurs in weight update during addition operation. Otherwise, at ever iteration the word length of the weights are increased. This assumptions are valid under the weight increment terms are minimum and weights are converged. Although in training period overflow occurs, so weight update is not correct and will lead to extra iterations to achieve convergence. So that the updated weights can be calculated in truncated form (W, W_i). The calculated weights are applied to the error computation block.

6. RESULTS AND PERFORMANCE ANALYSIS

We have verified the proposed design and existing designs by writing the VHDL code, simulated and synthesized. The word size of the input signal and weight coefficients are considered as 8,i.e.,L=W=8 and filter length is chosen to be 4,i.e.,N=4.

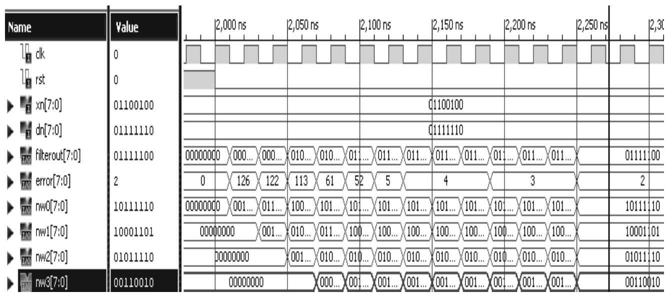
The simulation result of the LMS adaptive filter with Modified Booth Multiplier is shown in Fig.11.



Name	Value
clk	0
rst	0
xn[7:0]	01100100
dn[7:0]	01111110
filterout[7:0]	01111100
error[7:0]	2
nw0[7:0]	10111110
nw1[7:0]	10001101
nw2[7:0]	01011110
nw3[7:0]	00110010

Fig.11.The LMS adaptive filter with Modified Booth Multiplier

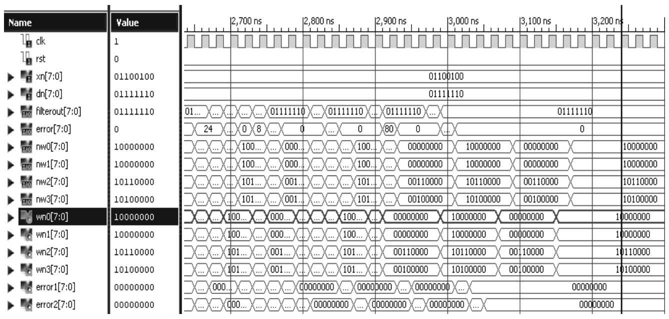
The simulation result of the LMS adaptive filter with Baugh-Wooley Multiplier is shown in Fig.12.



Name	Value
clk	0
rst	0
xn[7:0]	01100100
dn[7:0]	01111110
filterout[7:0]	01111100
error[7:0]	2
nw0[7:0]	10111110
nw1[7:0]	10001101
nw2[7:0]	01011110
nw3[7:0]	00110010

Fig.12. The LMS adaptive filter with Baugh Wooley Multiplier

The simulation result of the LMS adaptive filter with Proposed Architecture is shown in below Fig.13.



Name	Value
clk	1
rst	0
xn[7:0]	01100100
dn[7:0]	01111110
filterout[7:0]	01111110
error[7:0]	0
nw0[7:0]	10000000
nw1[7:0]	10000000
nw2[7:0]	10110000
nw3[7:0]	10100000
wn0[7:0]	10000000
wn1[7:0]	10000000
wn2[7:0]	10110000
wn3[7:0]	10100000
error1[7:0]	00000000
error2[7:0]	00000000

Fig.13. The LMS adaptive filter with Proposed architecture

The Table III and Fig.14 shows the comparison between the LMS adaptive filter designed with multipliers and the LMS adaptive filter with Proposed architecture in terms of Delay, Power Delay Product and Area Delay Product. The proposed design could achieve less delay, Area delay product and power delay product compared with existing structure.

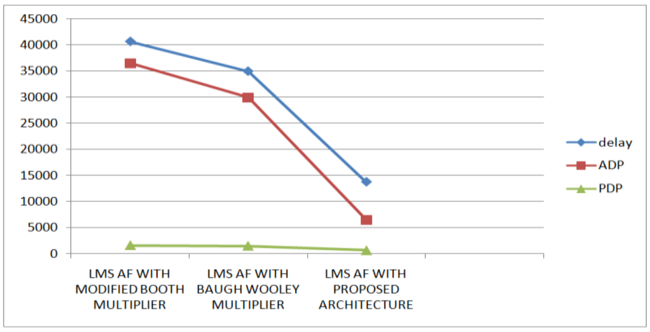


Fig.14.Comparision Graph of Adaptive Filter

TABLE-III

Design	Delay (ns)	Area-delay Product (sq.mmns)	Power-delay Product (w ns)
LMS adaptive filter with Modified Booth Multiplier	40.689	36498.033	1.62756
LMS adaptive filter with BaughWooley Multiplier	34.958	29924.048	1.468236
LMS adaptive filter with Proposed architecture	13.788	6507.936	0.675612

7. CONCLUSION

We proposed a new Architecture for Delayed Least Mean Square Adaptive filter. In this proposed architecture, we used a novel partial product generator in place of multipliers to compute the inner product by sharing a common sub expression. The adaptation delay is significantly reduced by using an efficient addition scheme in computation of inner product. The adaptation delay and power dissipations are further reduced by introducing optimized balanced pipelining across the time consuming blocks of the structure. The fixed point representation is used for signals that can be involved in proposed architecture. From the synthesis results, the proposed design yields less area delay product and power delay product when compared with existing designs.

REFERENCES

[1] P. K. Meher and S. Y. Park, "Area-Delay-Power Efficient Fixed-Point LMS Adaptive Filter With Low Adaptation-Delay", IEEE Transactions on Very Large Scale Integration (vlsi) Systems, vol. 22, no. 2, FEB 2014.

[2] B. Widrow and S. D. Stearns, Adaptive Signal Processing. Englewood Cliffs, NJ, USA: Prentice-Hall, 1985.

[3] S. Haykin and B. Widrow, Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley, 2003.

[4] 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.

[5] G. Long, F. Ling, and J. G. Proakis, "The LMS algorithm with delayed coefficient adaptation," IEEE Trans. Acoust., Speech, Signal Process.,vol. 37, no. 9, pp. 1397–1405, Sep. 1989.

[6] 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.

[7] H. Herzberg and R. Haimi-Cohen, "A systolic array realization of an LMS adaptive filter and the effects of delayed adaptation," IEEE Trans.Signal Process., vol. 40, no. 11, pp. 2799–2803, Nov. 1992.

[8] 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.

[9] S. Ramanathan and V. Visvanathan, "A systolic architecture for LMS adaptive filtering with minimal adaptation delay," in Proc.Int. Conf. Very Large Scale Integr. (VLSI) Design, Jan. 1996, pp. 286–289.

[10] Y. Yi, R. Woods, L.-K. Ting, and C. F. N. Cowan, "High speed FPGA-based implementations of delayed-LMS filter," J. Very Large Scale Integr. (VLSI) Signal Process., vol. 39, nos. 1–2, pp. 113–131,Jan. 2005.

[11] L. D. Van and W. S. Feng, "An efficient systolic architecture for the DLMS adaptive filter and its applications," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process., vol. 48, no. 4, pp. 359–366,Apr. 2001.

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

[13] P. K. Meher and M. Maheshwari, "A high-speed FIR adaptive filter architecture using a modified delayed LMS algorithm,"in Proc. IEEE Int. Symp. Circuits Syst., May 2011,pp. 121–124.

[14] P. K. Meher and S. Y. Park, "Low adaptation-delay LMS adaptive filter part-I: Introducing a novel multiplication cell," in Proc. IEEE Int.Midwest Symp. Circuits Syst., Aug. 2011, pp. 1–4.

[15] P. K. Meher and S. Y. Park, "Low adaptation-delay LMS adaptive filter part-II: An optimized architecture," in Proc. IEEE Int. Midwest Symp.Circuits Syst., Aug. 2011, pp. 1–4.

[16] Borth, D.E., Gerson, I.A., Haug, J.R., and Thompson, C.D., A flexible adaptive FIR filter VLSI IC, IEEE J. Sel. Areas Commun., 6(3), 494–503, April 1988.

[17] Ahmed Elhossini, Shawki Areibi, Robert Dony, “An FPGA Implementation of the LMS Adaptive Filter for Audio Processing”, IEEE International Conference on Reconfigurable Computing and FPGA's, ReConFig 2006.

[18] Reid M. Hewlitt, “Canonical Signed Digit Representation for Fir Digital Filters”, IEEE Workshop on Signal Processing Systems, 2000,pp.416-426.