

Critical-Path Analysis and Low-Complexity Implementation of the LMS Adaptive Algorithm

T.Reddysekhar¹, Dr.Atanu Das²M.Tech Scholar, Dept. of ECE, MITS, Madanapalle, Andhra Pradesh, India¹Assistant Professor, Dept. of ECE, MITS, Madanapalle, Andhra Pradesh, India²

ABSTRACT: This paper presents a precise analysis of the critical path of the Least - Mean - Square (LMS) adaptive filter for deriving its architectures for high - speed and low-complexity implementation. It is shown that the direct-form LMS adaptive filter has nearly the same critical path as its transpose-form counterpart, but provides much faster convergence and lower register complexity. From the critical-path evaluation, it is further shown that no pipelining is required for implementing a direct-form LMS adaptive filter for most practical cases, and can be realized with a very small adaptation delay in cases where a very high sampling rate is required. Based on these findings, this paper proposes three structures of the LMS adaptive filter: (i) Design 1 having no adaptation delays, (ii) Design 2 with only one adaptation delay, and (iii) Design 3 with two adaptation delays. For comparison of proposed systems its verilog coded and synthesized in Xilinx.

KEYWORDS: Adaptive filters, critical-path optimization, least mean square algorithms, LMS adaptive filter

I. INTRODUCTION

Filters of some sort are essential to the operation of most electronic circuits. It is therefore in the interest of anyone involved in electronic circuit design to have the ability to develop filter circuits capable of meeting a given set of specifications. In circuit theory, a filter is an electrical network that alters the amplitude and/or phase characteristics of a signal with respect to frequency. Ideally, a filter will not add new frequencies to the input signal, nor will it change the component frequencies of that signal, but it will change the relative amplitudes of the various frequency components and/or their phase relationships. Filters are often used in electronic systems to emphasize signals in certain frequency ranges and reject signals in other frequency ranges. Such a filter has a gain which is dependent on signal frequency.

Filters used for direct filtering can be either Fixed or Adaptive.

1. Fixed filters - The design of fixed filters requires a priori knowledge of both the signal and the noise, i.e. if we know the signal and noise beforehand; we can design a filter that passes frequencies contained in the signal and rejects the frequency band occupied by the noise.

2. Adaptive filters - Adaptive filters, on the other hand, have the ability to adjust their impulse response to filter out the correlated signal in the input. They require little or no a priori knowledge of the signal and noise characteristics (they require a signal (desired response) that is correlated in some sense to the signal to be estimated). Moreover adaptive filters have the capability of adaptively tracking the signal under non-stationary conditions.

The Least Mean Square (LMS) adaptive filter is the most popular and most widely used adaptive filter, not only because of its simplicity but also because of its satisfactory convergence performance. 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

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

modified to a form called the delayed LMS (DLMS) algorithm, which allows pipelined implementation of the filter. 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. has proposed a modified systolic architecture to reduce the adaptation delay. A transpose-form LMS adaptive filter is suggested in, 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 .

II. ADAPTATION ALGORITHM

The basic configuration of an adaptive filter, operating in the discrete time domain n , is illustrated in Figure 1. In such a scheme, the input signal is denoted by $x(n)$, the reference signal $d(n)$ represents the desired output signal (that usually includes some noise component), $y(n)$ is the output of the adaptive filter, and the error signal is defined as $e(n) = d(n) - y(n)$. The error signal is used by the adaptation algorithm to update the adaptive filter coefficient vector $w(n)$ according to some performance criterion. In general, the whole adaptation process aims at minimizing some metric of the error signal, forcing the adaptive filter output signal to approximate the reference signal in a statistical sense. There are several adaptation algorithms with different performance criterion. Due to its low complexity and proven robustness, Least Mean Square (LMS) algorithm is used here.

LMS algorithm is a noisy approximation of steepest descent algorithm. It is a gradient type algorithm that updates the coefficient vector by taking a step in the direction of the negative gradient of the objective function.

$$w(n+1) = w(k) - \frac{\mu}{2} \frac{\delta J_w}{\delta w(n)} \dots\dots\dots (1)$$

LMS Algorithm: For each n

$$w_{n+1} = w_n + \mu e_n x_n \dots\dots\dots (2)$$

Where,

$$e_n = d_n - y_n, y_n = w_n^T x_n \dots\dots\dots (3)$$

Where, the input vector x_n , and the weight vector at the w_n Th iteration are, respectively, given by

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

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

d_n is the desired response y_n is the filter output, and e_n denotes the error computed during the n th iteration. μ is the step-size, and N is the number of weights used in the LMS adaptive filter.

2.1 Implementation of Direct - Form Delayed LMS Algorithm

In the case of pipelined designs 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 updating the current weight instead of the recent-most error. The weight-update equation of DLMS adaptive filter is given by

$$w_{n+1} = w_n + \mu e(n-n_1) \cdot x(n-n_1) \dots\dots\dots (4)$$

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

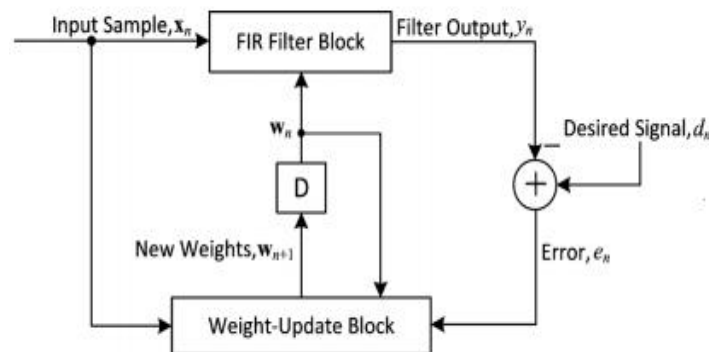


Fig 1: Structure of the conventional LMS adaptive filter

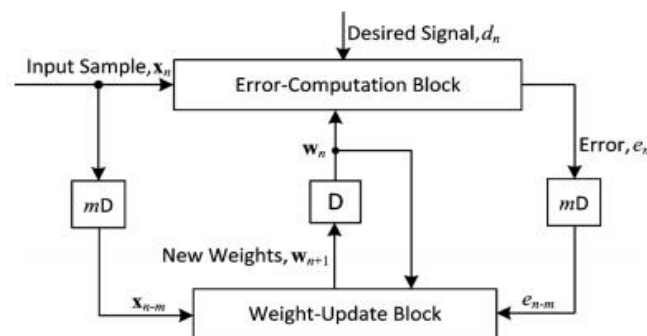


Fig 2: Generalized block diagram of direct form DLMS adaptive filter

Since all weights are updated concurrently in every cycle to compute the output according to (1), direct-form realization of the FIR filter is a natural candidate for implementation. However, the direct-form LMS adaptive filter is often believed to have a long critical path due to an inner product computation to obtain the filter output. This is mainly based on the assumption that an arithmetic operation starts only after the complete input operand words are available/generated. For example, in the existing literature on implementation of LMS adaptive filters, it is assumed that

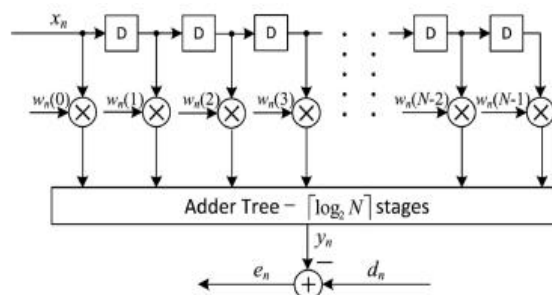


Fig 3: Error - computation block of Fig 2

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

The addition in a multiply-add operation can proceed only after completion of the multiplication, and with this assumption, the critical path of the multiply-add operation (T_{MA}) becomes ($T_{MULT} + T_{ADD}$), where T_{MULT} and T_{ADD} are the time required for a multiplication and an addition, respectively. Under such assumption, the critical path of the direct-form LMS adaptive filter (without pipelining) can be estimated as

$$T = 2 T_{MULT} + (N+1) T_{ADD}$$

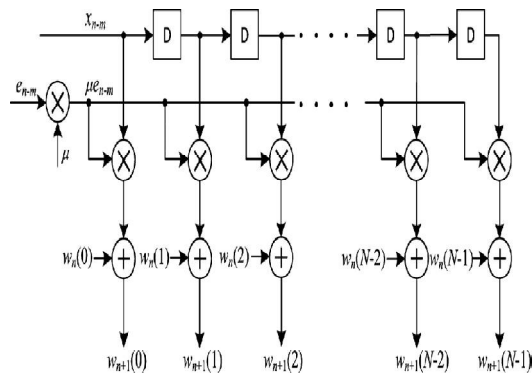


Fig 4: Weight - update block of Fig. 3

A generalized block diagram of direct-form DLMS adaptive filter is shown in Fig.2. It consists of an error-computation block (shown in Fig. 3) and a weight-update block (shown in Fig. 4). The number of delays shown in Fig. 2 corresponds to the pipeline delays introduced due to pipelining of the error-computation block

2.2 Implementation of Transpose-Form Delayed LMS Algorithm

The transpose-form FIR structure cannot be used to implement the LMS algorithm given by (1), since the filter output at any instant of time has contributions from filter weights updated at different iterations, where the adaptation delay of the weights could vary from 1 to N . It could, however, be implemented by a different set of equations as follows

$$y_n = \sum_{k=0}^{N-1} x_{n-k} w_{n-k}$$

$$w_{n-1}(k) = w_n(k) + \mu e_n x_{n-k}$$

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

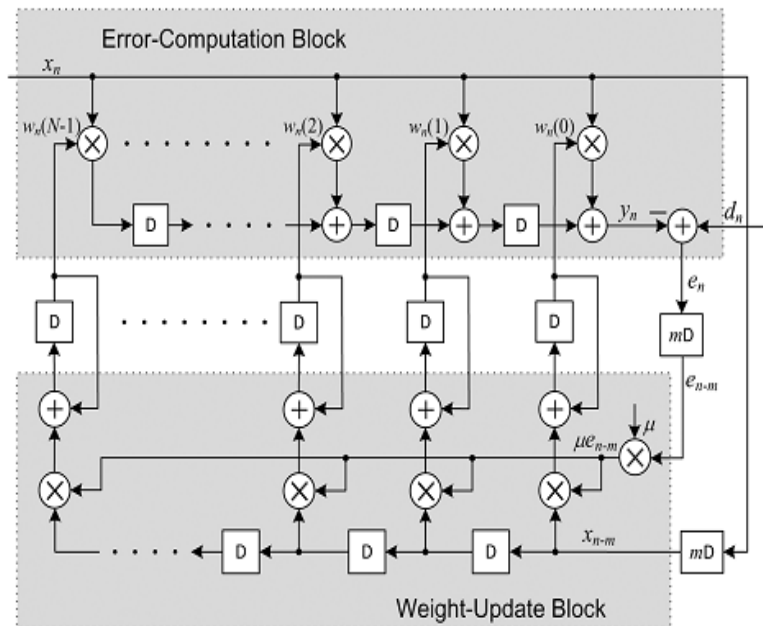


Fig 5: Structure of transpose-form DLMS adaptive filter

The additional adaptation delay m could be at most 2 if no more delays are incorporated within the multiplication unit or between the multipliers and adders. If one delay could be placed after the computation of y_n and another after the computation of e_n , then $m=2$.

III. PROPOSED DESIGN

We find that the direct-form FIR structure not only is the natural candidate for implementation of the LMS algorithm in its original form, but also provides better convergence speed with the same residual MSE. It also involves less register complexity and nearly the same critical path as the transpose-form structure. Therefore, we have preferred to design a low-complexity direct-form structure for implementation of the LMS adaptive filter.

3.1 Zero Adaptation Delay

As shown in Fig. 2, there are two main computing blocks in the direct-form LMS adaptive filter, namely, i) the error-computation block (shown in Fig. 3) and ii) the weight-update block (shown in Fig. 4). It can be observed in Figs. 3 and 4 that most of the area-intensive components are common in the error-computation and weight-update blocks: the multipliers, weight registers, and tapped-delay line. The adder tree and subtractor in Fig-4 and the adders for weight updating in Fig-5, which constitute only a small part of the circuit, are different in these two computing blocks. For the zero-adaptation-delay implementation, the computation of both these blocks is required to be performed in the same cycle. Moreover, since the structure is of the non-pipelined type, weight updating and error computation cannot occur concurrently. Therefore, the multiplications of both these phases could be multiplexed by the same set of multipliers, while the same registers could be used for both these phases if error computation is performed in the first half cycle while weight update is performed in the second-half cycle.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

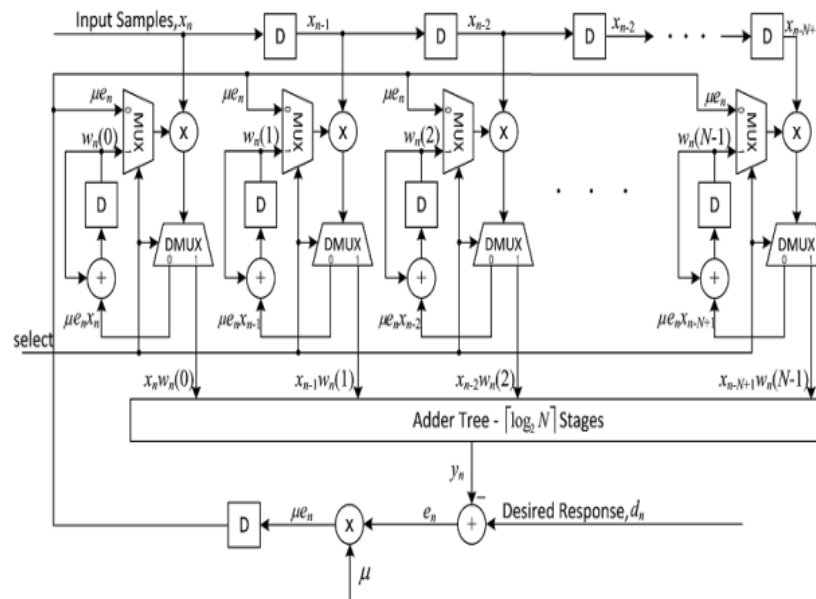


Fig 6: Structure for Zero-Adaptation-Delay Time- Multiplexed Direct-Form LMS Adaptive Filter

The registers in the delay line are clocked at the rising edge of the clock pulse and remain unchanged for a complete clock period since the structure is required to take one new sample in every clock cycle. During the first half of each clock period, the weight values stored in different registers are fed to the multiplier through the multiplexers to compute the filter output. The product words are then fed to the adder tree through the de-multiplexers.

3.1.1 One Adaptation Delay:

The proposed structure for a one-adaptation-delay LMS adaptive filter consists of one error-computation unit as shown in Fig1.5 and one weight-update unit as shown in Fig1.6. A pipeline latch is introduced after computation of μe_n . The multiplication with μ requires only a hardwired shift, since μ is assumed to be a power of 2 fraction. So there is no register overhead in pipelining. Also, the registers in the tapped delay line and filter weights can be shared by the error-computation unit and weight-updating unit. The critical path of this structure is the same as C_{ERROR} [derived in (11)], given by

$$T = T_{\text{MULT}} - ([\log_2 N] + 1)\Delta$$

3.1.2 Two Adaptation Delays:

The proposed structure for a two-adaptation-delay LMS adaptive filter is shown in Fig3.2, which consists of three pipeline stages, where the first stage ends after the first level of the adder tree in the error-computation unit, and the rest of the error-computation block comprises the next pipeline stage.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

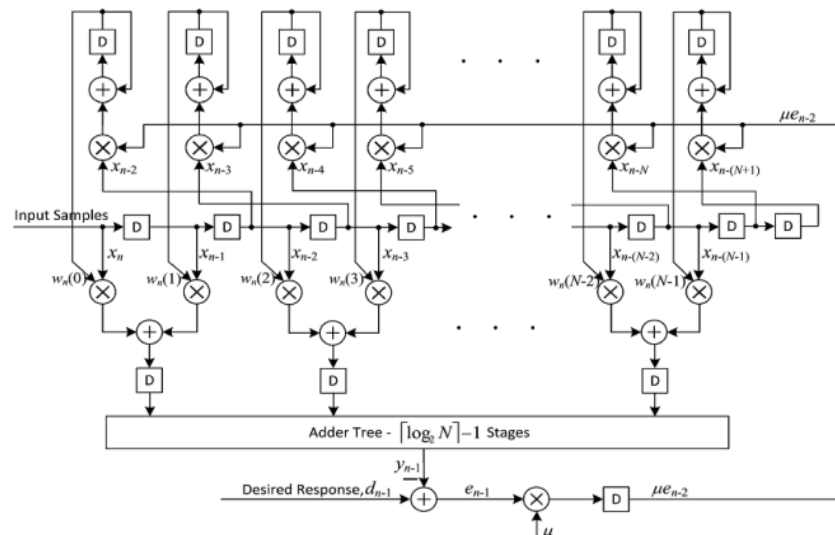


Fig 7: Proposed Structure for Two-Adaptation-Delay Direct-Form LMS Adaptive Filter

The weight-update block comprises the third pipeline stage. The two adaptation delay structure involves additional registers over the one-adaptation-delay structure. The critical path of this structure is the same as either that of the weight-update unit C_{UPDATE} or the second pipeline stage.

IV. SIMULATION RESULT

Low complexity adaptive filter is verilog coded and simulated on Xilinx to check the desired functionality. The critical path of direct form of FIR filter is also analysed. The filter structured in Verilog is synthesized on Xilinx ISE. Fig 8.shows simulation result of conventional adaptive filter and fig-9 shows simulation result of proposed structure.

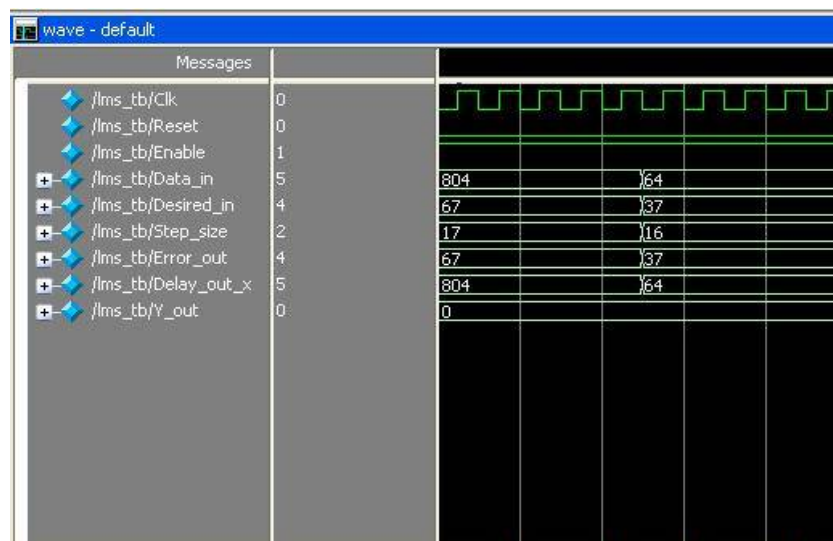


Fig 8: Simulation result of conventional adaptive filter

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

The above waveform shows the conventional adaptive filter depending up on the clock it will generate the outputs when depending up on the enable it will gives the y_out for given desired inputs

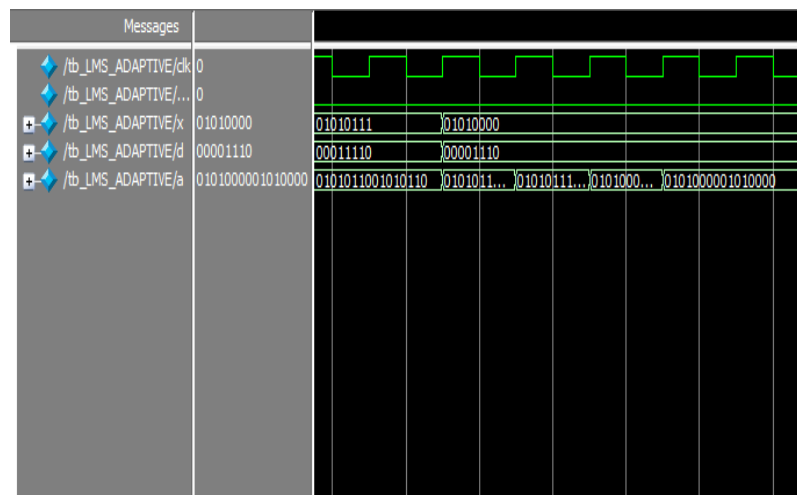


Fig 9: Simulation result of proposed LMS Adaptive Filter

The above waveform shows the when clock and enable is low we get the desired outputs will be obtained i.e the Concatenation of x and d

V. CONCLUSION

Low complexity LMS adaptive filter is proposed and critical path is also analyzed using direct form and transposed form of FIR filter .It is found that critical path of direct and transposed form is same but direct form is having less number of registers means low complexity than transposed .For direct form three systems are proposed. They are following:1)zero adaptation delay 2)one adaptation delay 3) two adaptation delay .It is found that for conventional adaptive filter delay is 19.732 ns and for proposed system delay is 4.182 ns.

Future Scope:

In this paper we modified adder tree by efficient CSA, CLA adder. The PPO unit generates the number of partial products then it will pass through adder tree instead of this adder tree we use the CSA, CLA adder.

REFERENCES

- [1] B.Widrow and S. D. Stearns, Adaptive Signal Processing. Englewood Cliffs, NJ, USA: Prentice-Hall, 1985.
- [2] S. Haykin and B.Widrow, Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley-Inter science, 2003.
- [3] G.Long,F.Ling,andJ.G.Proakis, "TheLMSalgorithmwithdelayedcoefficient adaptation,"IEEE Trans. Acoust., Speech, Signal Process.,vol. 37, no. 9, pp. 1397–1405, Sep. 1989.
- [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] L.D.Vanand W.S.Feng, "An efficient systolic architecture for the DLMS adaptive filter and its applications,"IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process, vol. 48, no. 4, pp. 359–366, Apr.2001.
- [6] 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.
- [7] E. Mahfouz, C. Wang, and M. O. Ahmad, "A high-throughput DLMS adaptive algorithm," in Proc. IEEE Int. Symp. Circuits Syst., May 2005, pp. 3753–3756.

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

- [8] P.K.Meherand ,S.Y.Park, “Low adaptation-delay LMS adaptive filter Part-II: An optimized architecture,” in Proc. IEEE Int. Midwest Symp. Circuits Syst., Aug.2011.
- [9] P. K. Meher and S. Y. Park, “Area-delay-power efficient fixed-point LMS adaptive filter with low adaptation-delay,” Trans. Very Large Scale Integr. (VLSI) Signal Process.[Online]. Available: <http://ieeexplore.ieee.org>
- [10] Y.i.R.Woods, L-K.Ting, and C.F.N.Cowan, “High speed FPGA-based implementations of delayed-LMS filter,” Trans. Very Large Scale Integr. (VLSI) Signal Process, vol. 39, no. 1–2, pp. 113–131, Jan. 2005.
- [11] S. Y. Park and P. K. Meher, “Low-power, high-throughput, and low area adaptive FIR filter based on distributed arithmetic,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 60, no. 6, pp. 346–350, Jun. 2013.
- [12] “TSMC 90 nm general-purpose CMOS standard cell libraries—tcbn90ghp” [Online]. Available: www.tsmc.com/
- [13] TSMC 0.13 m General-Purpose CMOS Standard Cell Libraries -tcb013ghp [Online]. Available: www.tsmc.com/ [14] 3GPP TS 36.211, Physical Channels and Modulation, ver. 10.0.0 Release 10, Jan. 2011.
- [15] 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.