IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 6, Issue 3, Ver. II (May. -Jun. 2016), PP 32-40
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Critical-Path Realization and Implementation of the LMS Adaptive Algorithm Using Verilog-HDL and Cadence-Tool

Shashirekha C[1], Deepthi Dayanand[2]
ECE Department , Sahyadri College of Engineering and Management, Mangaluru, India
Assistant Professor, ECE Department , Sahyadri College of Engineering and Management, Mangaluru, India

Abstract: *This paper introduces a brief description about Critical path realization of the Least Mean Square (LMS) adaptive filter and modified Delayed Least Mean Square (DLMS) adaptive filter to achieve a lower adaptation delay. To achieve lower adaptation delay, it is shown that the direct form LMS filter has longest critical path delay, to reduce the critical path delay pipelining implementation is required but conventional LMS adaptive filter does not support when it exceeds the desired sample period .So modified DLMS adaptive filter is used. This paper propose two structures of the LMS adaptive filter (i) Structure 1 having no adaptation delay, (ii) Structure 2 with only one adaptation delay .Proposed structure is coded using verilog and synthesized in Xilinx 10.1. and Cadence tool.*

Keywords: *Adaptation Delay, Adaptive filter, Critical path, Delayed LMS adaptive filter, LMS adaptive filter.*

I. Introduction

The adaptive filters are utilized for several applications in Digital Signal Processing (DSP) domains such as system detection, channel estimation, channel equalization etc . RLS algorithms has high convergence performance compared to LMS algorithms but computation complexity is more in RLS algorithm, so LMS algorithms are widely used in adaptive filter. Widrow Hoff Least Mean Square(LMS) algorithm [1] is used for update weights of the tapped delay line Finite-Impulse response(FIR) filter and it is considered as the simplest adaptive filter. To implement LMS algorithm, during each sampling period one has to update the filter weights using the estimated error e(n) which equals the difference between the current filter output y(n) and desired response d(n).

For nth iteration weights of the LMS adaptive filter are updated by following equations

$$W(n+1) = W(n) + \mu e(n)X(n) \quad (1)$$

where $$e(n) = d(n) - e(n) \quad (2)$$

and $$y(n) = W(n)^T X(n) \quad (3)$$

where y(n) is the filter output of the nth iteration, d(n) is the desired response, e(n) is the error computed during the nth iteration, μ is the convergence factor and N is the number of weights used in the LMS adaptive filter is shown in Fig.1However the direct-form LMS adaptive filter has a long critical path delay due to an inner product computation to obtain the filter output[2]. To reduce the critical path delay pipelining implementation is required but conventional LMS adaptive filter does not support. Therefore, it is modified to a form called Delayed LMS(DLMS) algorithm which allows pipelined implementation where m is the number of adaptation delay [4]. It consists of two computational block that is Error computational block and Weight update block . Assume that the error-computation path is implemented in m pipelined stages, the computation of error provides a latency of m cycles, so that the error computed by the structure at the nth cycle is e(n-m) and it is used with the input samples delayed by m cycles to generate the weight increment term. Fig.2. shows the generalized block diagram of direct-form DLMS adaptive filter.

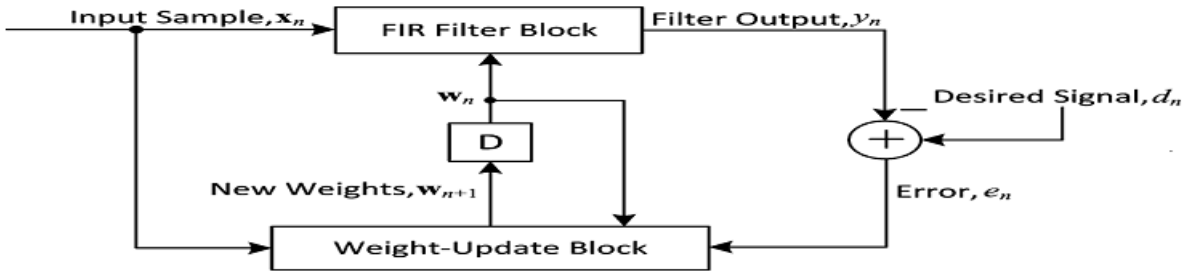

Fig.1. Generalized block diagram of conventional LMS adaptive filter.

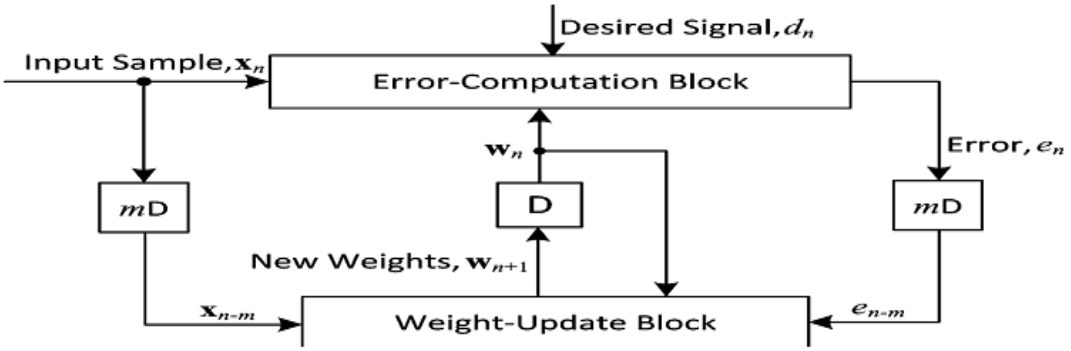

Fig.2. Basic block diagram of Delayed LMS adaptive filter

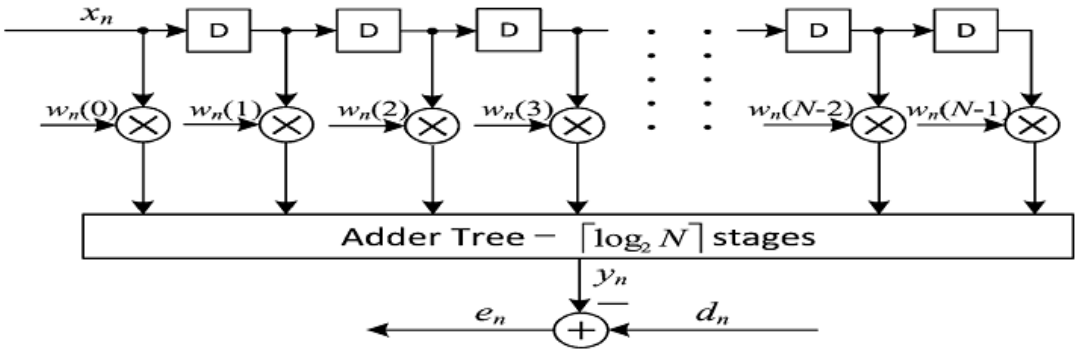

Fig.3. Error-computation block of Delayed LMS adaptive filter.

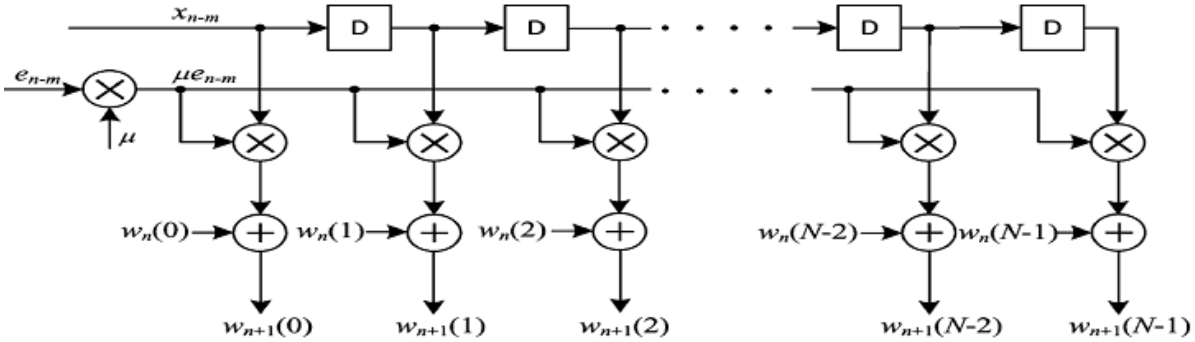

Fig.4. Weight-update block of Delayed LMS adaptive filter.

II. Cr □t □cal Path Analys □s Of D □red form Lms Adapt □ve F □lter

To find out the Critical Path of Direct form LMS adaptive filter, consider the implementation of an inner product x(0)w(0)+x(1)w(1)+x(2)w(2)+x(3)w(3) of length 4. The Computations of the first level adders are completed in time Tmult+ Tfas + Tfac, where Tfas is the delay due to the 3-input XOR operation and Tfac=Tand + Txor are the propagation delays of AND and XOR. Take Δ = Tfac + Tfas. The second level adder are completed in time Tmult+2 Δ.In general, inner product of length N involves delay

$$\text{Tip} = \text{Tmult} + \lceil \log_2 N \rceil \Delta \quad (4)$$

Cerror is time involved in error computation and Cupdate is time involved in weight updating computation. The critical path of the error computation block Cerror is given by

$$\text{Cerror} = \text{Tmult} + (\lceil \log_2 N \rceil + 1) \Delta \quad (5)$$

The critical path of the weight updating block Cupdate is given by

Cupdate = Tmult+ Δ (6

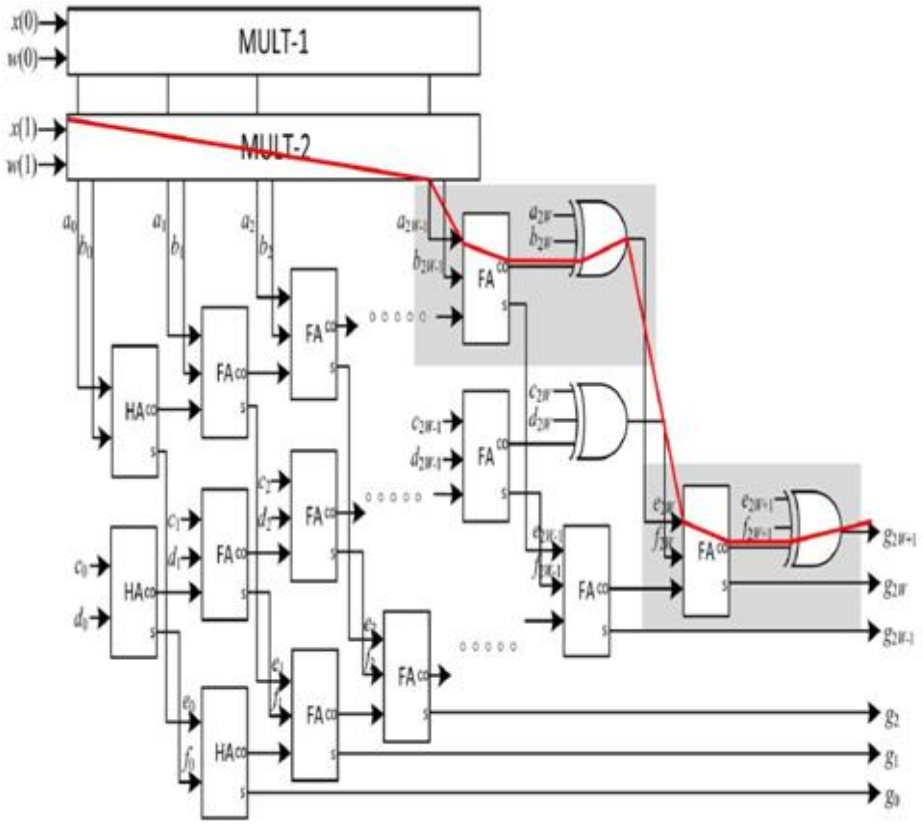

Fig. 5. Shows the Critical path of an inner product computation of length 4.

III. Proposed Structure

3.1. Zero Adaption Delay

For Zero adaptation delay it requires two computing blocks of Delayed LMS adaptive filter i)The error computation block ii)The weight update block shown in Fig.3 and Fig.4. respectively. In error computation and weight updated block most of the area intensive components are common i.e. weight register, multiplier, tapped delay line. In Fig.3 the adder tree and substractor and in Fig.4 adder for weight updating are different in these two computing blocks. The proposed structure consists of N multiplier ,a common tapped delay line, N 2:1 multiplexers ,N 1:2 de-multiplexers, N adder for modification N weights and an adder tree is used to add the output of N multipliers for computation of the filter output . Also it requires a substractor to compute the error value.

During the first half of the cycle, the input samples are fed to the multiplier from common tapped delay line. The N 2:1 multiplexers select one of the input among from N weight of values and it also requires estimated error values depending on the select line .In the first half the cycle N multiplexer selects, N weight values are fed to the other input to the N multiplier .The product words then fed to the N 1:2 de-multiplexers and N 1:2 de-multiplexers move these product words either towards the adder tree or weight update circuit depending on the select line. In first half of the cycle product words are must fed to the adder tree through the de-multiplexer. The filter output is computed by the adder tree and error value is also computed by the substractor. Then the computed error value is right shifted to obtain and it is broadcasted to all the N multipliers

During the second half of the clock period the input samples are fed to the multiplier from common tapped delay line .The N 2:1 multiplexer select estimated error as the other input to the multiplier. The product words are then fed to the N 1:2 de-multiplexer .In second half of the cycle product words are must fed to the weight updated circuit and new set of weight value is computed .In the next cycle begins ,the weight registers are also updated by the new weight register.

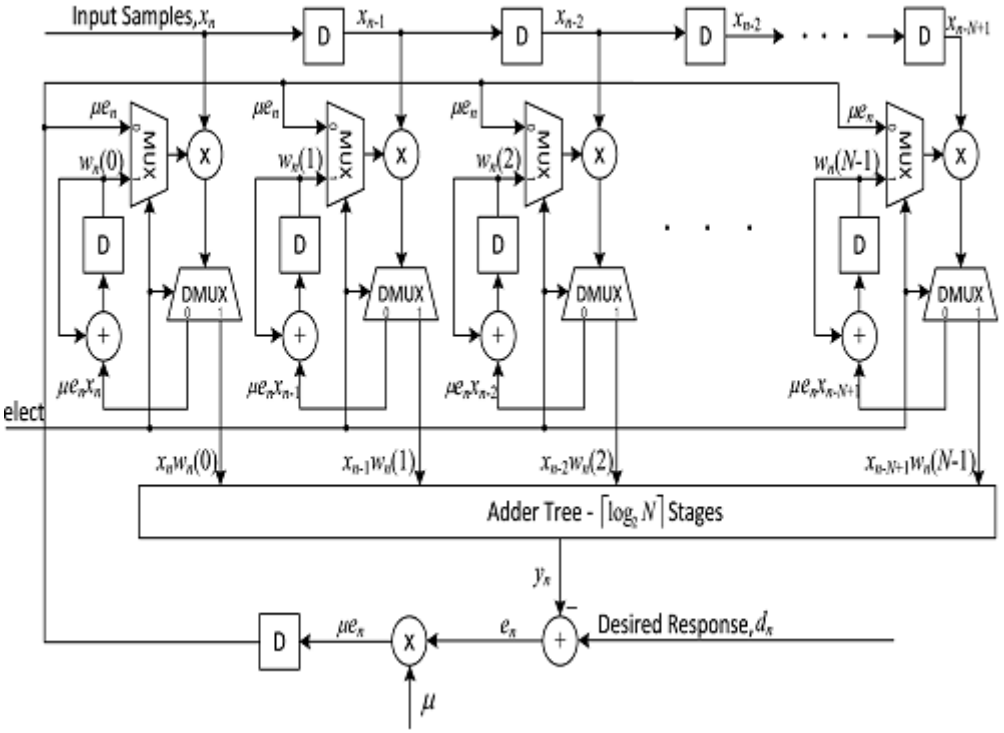

Fig .6.Structure for Zero - adaptation delay

3.2. One Adaptation Delay

The proposed structure mainly consists of error computation unit and one weight updated unit which are shown in the fig.3 and Fig.4 respectively It consists of three pipeline stages in that first pipeline stage ends after the first level of the adder tree in error computation unit, the second pipeline stage is comprised in error computation block and third pipeline stage is comprised in weight update block and it also contains the tapped delay line, adder, multiplier, adder tree, input samples and one subtraction block .In the first half of the clock period it will process Error-computation value and in second half of the clock period it will compute Weight-updating value for one adaptation delay.

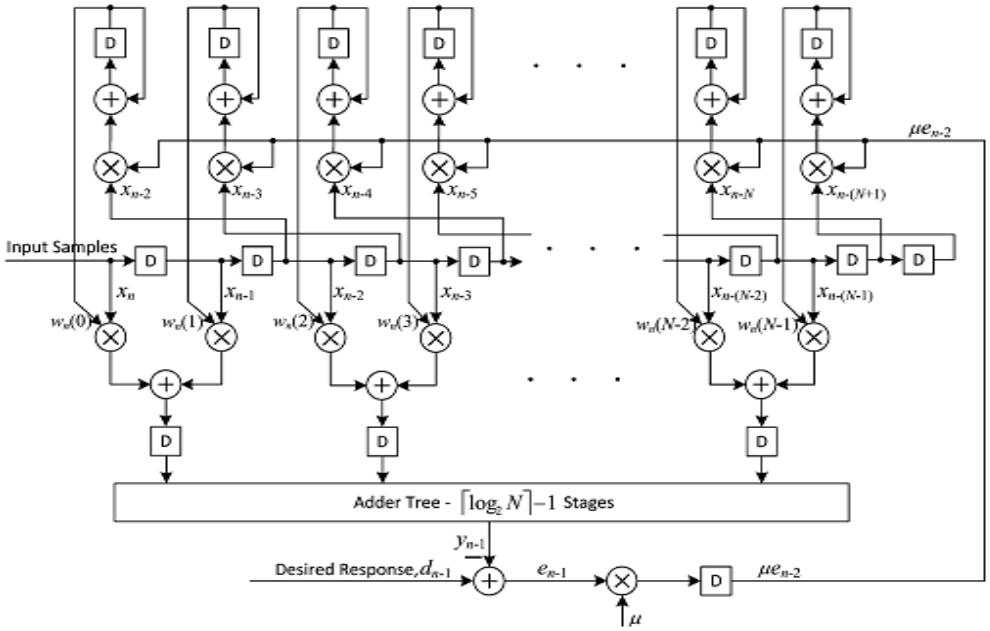

Fig. 7.Structure for One-adaptation-delay

IV. Results And Discussion

4.1 Implementation results using the Xilinx.10.1 tool.

For Direct-form LMS adaptive filter of inner product computation of direct-form LMS adaptive filter , Zero and One adaptation delay structure verilog code is written and syntax error is also checked using Xilinx 10.1 tool and simulation result is shown in Fig.8, Fig.9, and Fig.10 respectively.

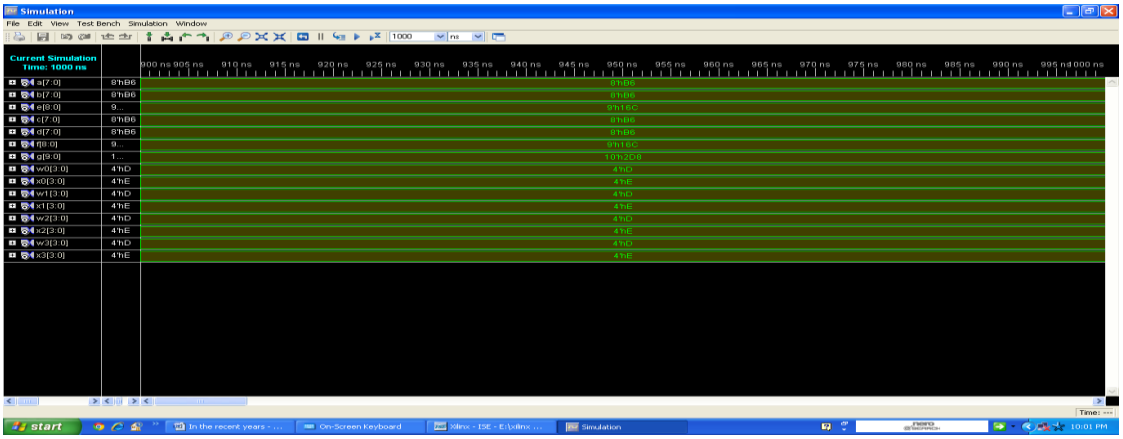

Fig.8. Simulation result for inner product computation of Direct-form LMS adaptive filter.

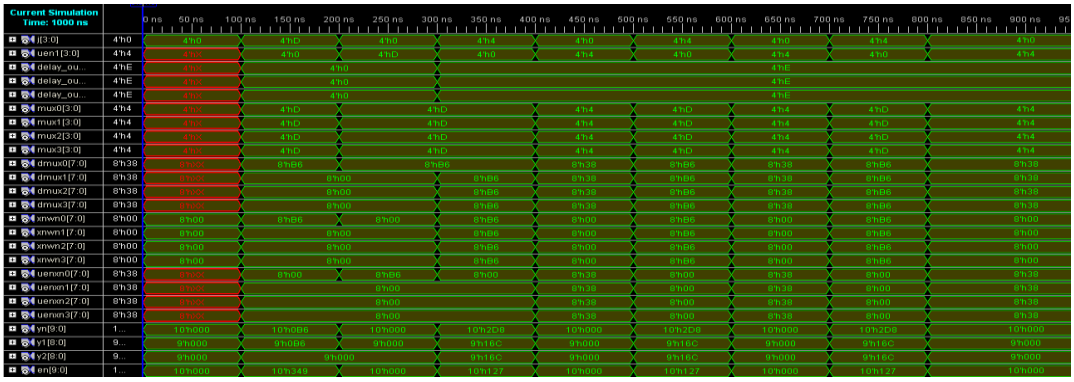

Fig.9.Simulation result for zero adaptation delay.

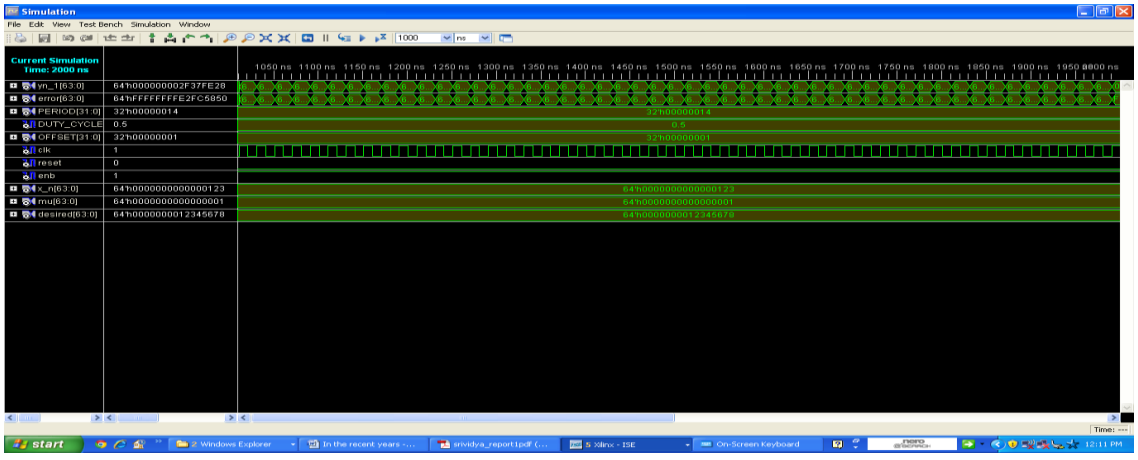

Fig.10. Simulation result for one adaptation delay

4.1 Implementation results using the Cadence- tool.

For Direct-form LMS adaptive filter of inner product computation length 4 , Zero and One adaptation delay structure simulation are shown in Fig.11, Fig.12, and Fig.13 respectively

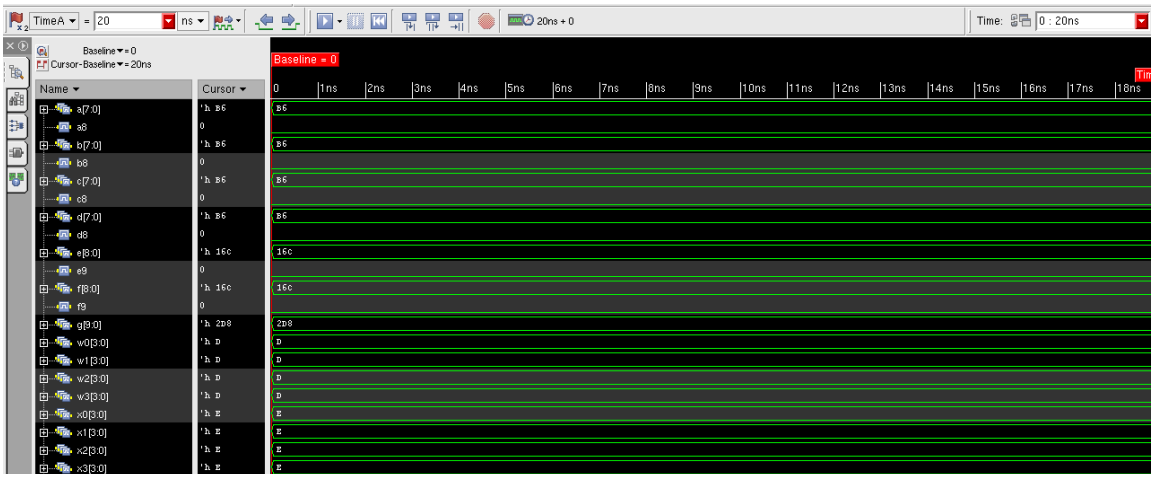

Fig.11 Simulation result for inner product computation of direct-form LMS adaptive filter

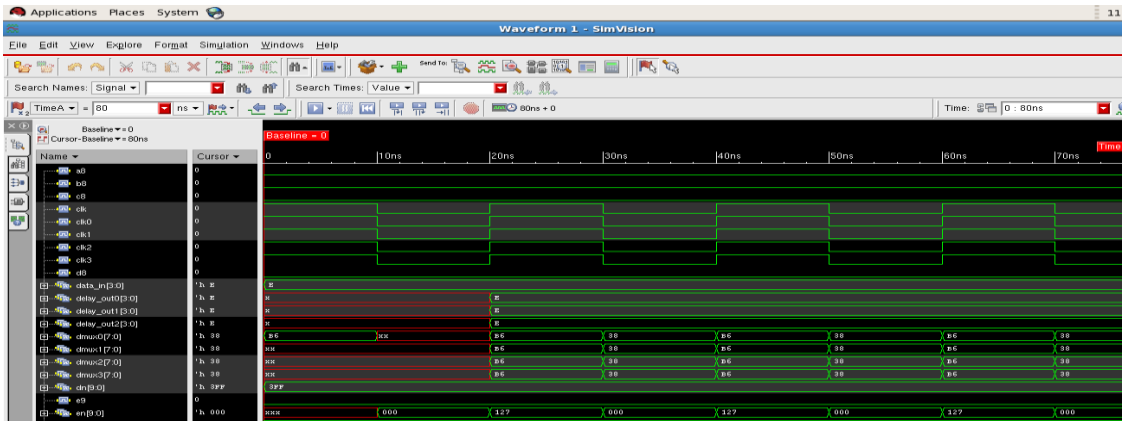

Fig.12. Simulation result of Zero adaptation delay

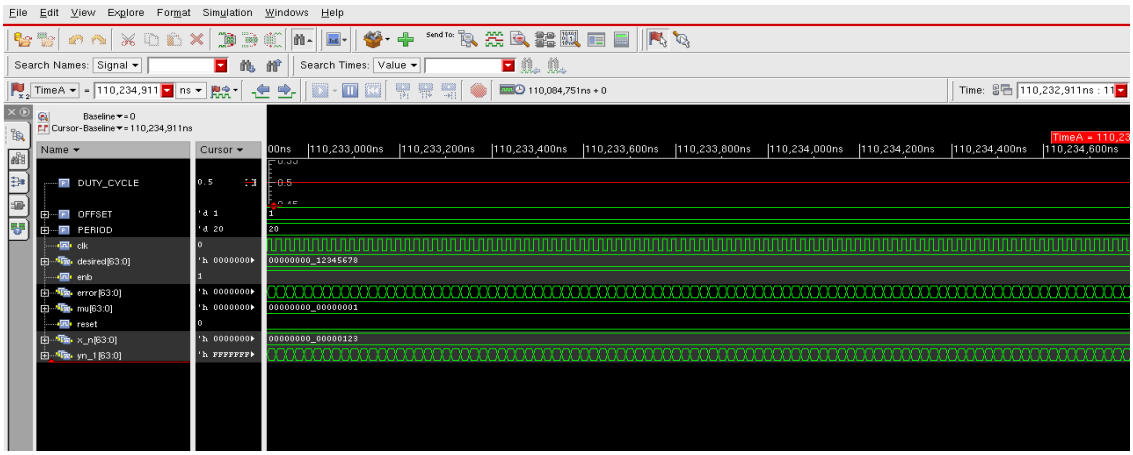

Fig.13. Simulation result for One adaptation delay

For Direct-form LMS adaptive filter of inner product computation length 4 , Zero and One adaptation delay structure physical design result are shown in Fig.14, Fig.15 and Fig.16 respectively

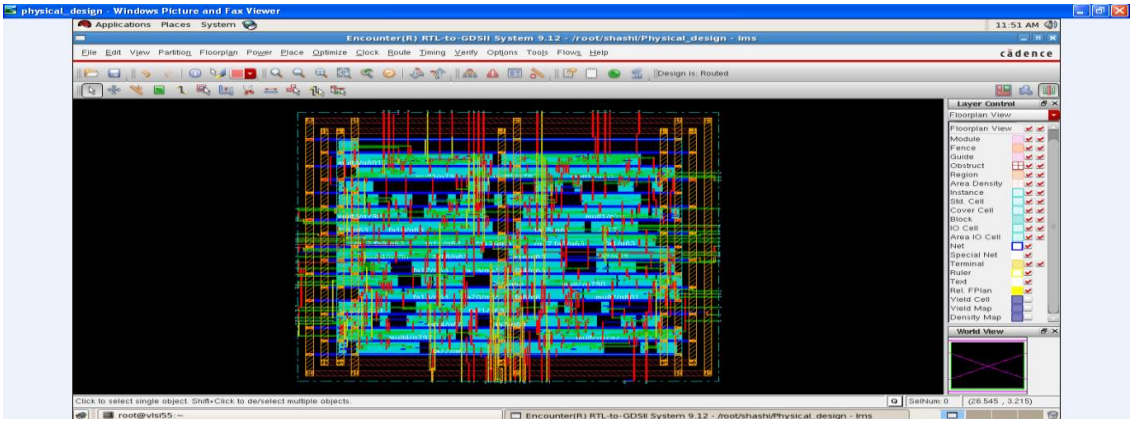

Fig.14 Physical design of direct-form LMS adaptive filter.

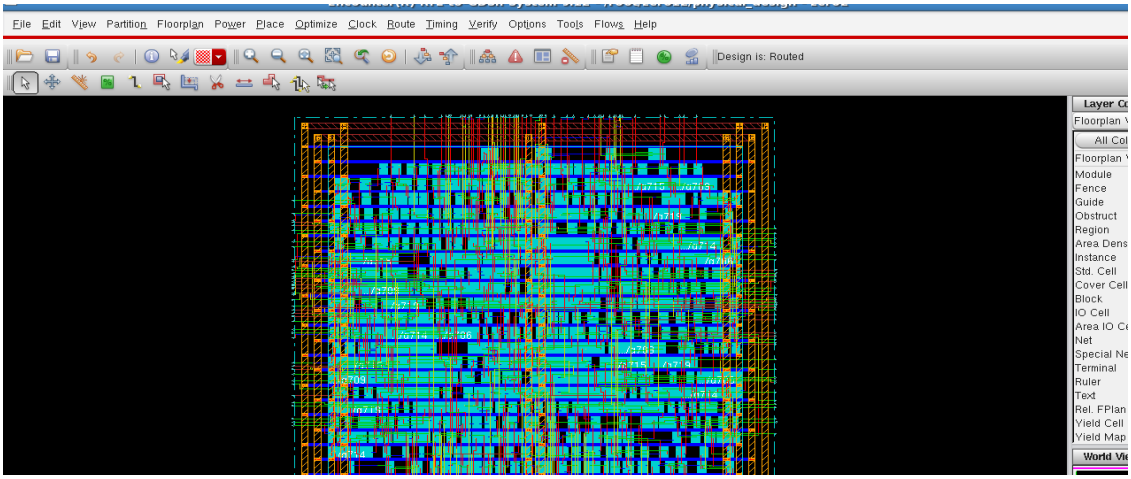

Fig.15. Physical design of Zero adaptation delay.

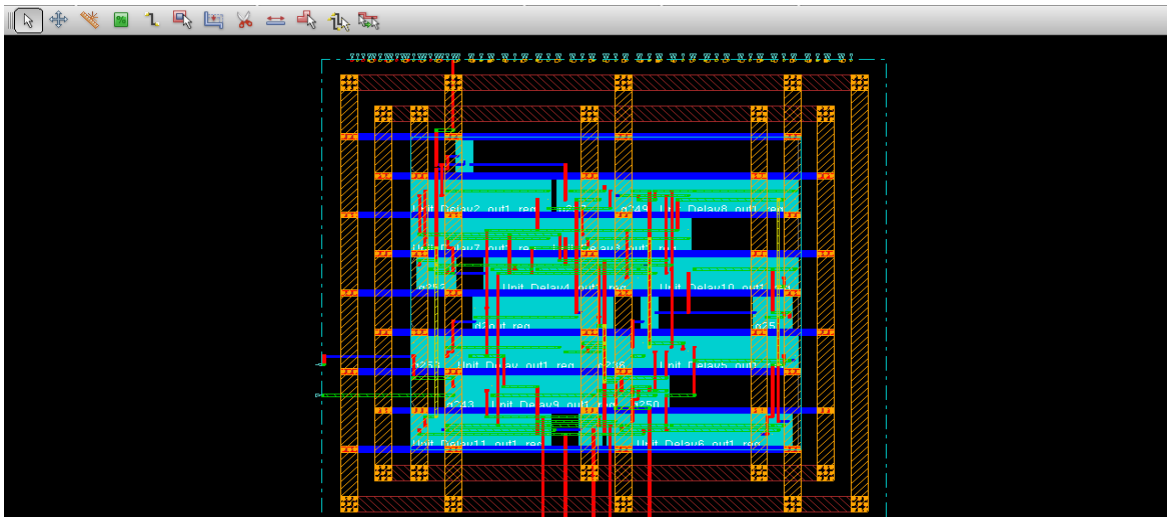

Fig.16. Physical design of an One adaptation delay.

Table 1: Hardware Utilization of inner product computation of direct-form LMS adaptive filter

Device Utilization Summary (estimated values)				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	
Number of Slices	28	3584		0%
Number of 4 input LUTs	50	7168		0%
Number of bonded IOBs	98	97	101%	
Number of MULT18X18s	4	16		25%

Table 2: Hardware utilization of Zero adaptation delay

Device Utilization Summary (estimated values)				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	
Number of Slices	99	960		10%
Number of Slice Flip Flops	12	1920		0%
Number of 4 input LUTs	174	1920		9%
Number of bonded IOBs	234	108	216%	
Number of MULT18X18SIOs	4	4		100%
Number of GCLKs	3	24		12%

Table. 3 Hardware utilization of One adaptation delay

Device Utilization Summary (estimated values)				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	
Number of Slices	2623	960	273%	
Number of Slice Flip Flops	541	1920		28%
Number of 4 input LUTs	4663	1920	242%	
Number of bonded IOBs	227	108	210%	
Number of MULT18X18SIOs	3	4		75%
Number of GCLKs	1	24		4%

Table.4 Comparison of Zero and One adaptation delay.

Device utilization			
		Zero adaptation delay	One adaptation delay
Logic utilization	Available	Used	Used
Number of Slices	960	99	2623
Number of Slice flip flops	541	12	541
Number of input LUTs	4663	174	4663
Number of bounded IOBs	227	234	227
Number of MULT18X18SIO	3	4	3
Number of GCLKs	1	3	1

V. Conclusion

For Direct-form LMS adaptive filter the critical path is evaluated. For critical path of an inner product computation is coded using verilog code and synthesize on Xilinx ISE 10.1 and using cadence tool simulation and physical layout generation is done. The proposed structure Zero and One adaptation delay are also coded using verilog and synthesized on Xilinx ISE ver.10.1and for this also simulation ,synthesis and physical design is obtained. By observing the above table, the zero adaptation delay is better than one adaptation delay structure.

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]. 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.
[4]. E. Mahfuz, C. Wang, and M. O. Ahmad, "A high-throughput DLMS adaptive algorithm," in Proc. IEEE Int. Symp. Circuits Syst., May 2005, pp. 3753–3756.
[5]. 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.

[6]. 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
[7]. Y. Yi, R. Woods, L.-K. Ting, and C. F. N. Cowan, "High speed FPGA-based implementations of delayed-LMS filters," Trans. Very Large Scale Integr. (VLSI) Signal Process., vol. 39, no. 1–2, pp.113–131, Jan. 2005.
[8]. 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.
[9]. D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in Proc. IEEE Southeastcon, Apr. 1993.
[10]. M. Mboup, M. Bonnet, and N. Bershad, "LMS coupled adaptive prediction and system identification: A statistical model and transient mean analysis," IEEE Trans. Signal Process., vol. 42, no. 10, pp. 2607–2615, Oct. 1994.
[11]. C. Breining, P. Dreiseitel, E. Hansler, A. Mader, B. Nitsch, H. Puder, T. Schertler, G. Schmidt, and J. Tilp, "Acoustic echo control," IEEE Signal Process. Mag., vol. 16, no. 4, pp. 42–69, Jul. 1999.
[12]. W. A. Harrison, J. S. Lim, and E. Singer, "A new application of adaptive noise cancellation," IEEE Trans. Acoust., Speech, Signal Process.,vol. 34, no. 1, pp. 21–27, Feb. 1986.
[13]. S. Coleri, M. Ergen, A. Puri, and A. Bahai, "A study of channel estimation in OFDM systems," in Proc. IEEE Veh. Technol. Conf., 2002,pp. 894–898.
[14]. J. C. Patra, R. N. Pal, R. Baliarsingh, and G. Panda, "Nonlinear channel equalization for QAM signal constellation using artificial neural networks,"IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 29, no. 2,pp. 262–271, Apr. 1999.
[15]. G. A. Clark, S. K. Mitra, and S. R. Parker, "Block implementation of adaptive digital filters," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-29, no. 3, pp. 744–752, Jun 1981.