International Journal of Scientific and Technical Advancements

ISSN: 2454-1532

FPGA Implementation and Hardware Analysis of LMS Algorithm Derivatives: A Case Study on Performance Evaluation

Aditya Bali[1#], Rasmeet kour [2], Sumreti Gupta[3], Sameru Sharma[4]
[1]Department of Electronics & Communication, GCET Jammu, J&K, India- 181122
[#] adityabali19@gmail.com:

Abstract— This paper presents hardware utilization, maximum operating frequency, delay and power of different variants of Least Mean Square (LMS) based adaptive filter. Two different structures or architectures are used for the implementation adaptive filter. First one is conventional serial structure and second one is tapped delay line serial structure. It has been found that the tapped delay line serial structure is better than conventional serial structure in terms of clock speed, hardware utilization and delay. Further, different variants of LMS algorithm viz. LMS, Sign-Data (SD), Sign-Error (SE) and Sign-Sign (SS) are implemented using aforesaid two structures and their performances are compared. The behavioral modelling of the architectures was done using Verilog HDL. After that the design was synthesized using Xilinx ISE 14.5 and further implemented on Xilinx Spartan3E XC3S500E Field Programmable Gate Arrays (FPGA) having package type FG320, speed grade -4 and onboard clock of frequency 50 MHz.

Keywords— Adaptive filters; sign-data; sign-error; sign-sign; LMS; FPGA; HDL

I. INTRODUCTION

Adaptive Filtering is used primarily in real time applications where the statistics of the signal changes with time or not known. In these (real time) applications adaptive filters changes its coefficients according to an adaptation algorithm to estimate the signal statistics iteratively. The importance of the adaptive filtering lies because of its simplicity and high performance capability. The performance of the adaptive filter depends upon the step size, type of adaptive algorithm and the structure or architecture used for the implementation of the adaptive filter. The step size of an adaptive filter is not easy to choose. It should be selected in a way that it is neither too small nor too large, as it affects the performance of the filtering algorithm. Proper choice of adaptation algorithm is important as it also affects the filter performance by changing the hardware utilization, convergence rate and MSE which in turn affects the filter performance. A fixed step size and self correcting adaptive filter (SCAF) (Nekouei, Talebi, Kavian, & Mahani, 2012) [1] was implemented of on a Spartan 3 XC3S400 FPGA. An IIR adaptive filter with particle swarm optimization (PSO) (Gao, Zeng, Wang, & Liu, 2008) [2] was implemented on FPGA. An adaptive impulsive noise filter (Rosado-Muñoz, Bataller-Mompeán, Soria-Olivas, Scarante, & Guerrero-Martínez, 2011) [3] was implemented on a Xilinx FPGA and compared its performance in terms of accuracy, performance and logic operation. A 64-tap 9-bit LMS adaptive FIR filter for active noise control (ANC) (Mustafa, Ali, Umat, & Al-Asady, 2009) [4] was implemented on Altera Cyclone II FPGA considering a 24 KHZ uniform random noise signal. An LMS adaptive filter designed by three different architectures (Elhossini, Areibi, & Dony, 2006) [5] was implemented on a Virtex II FPGA and it is then compared in terms of chip area and performance. High speed adaptive filter (Vella & Debono, 2006) [6] for echo cancellation was illustrated using different architectures. NLMS algorithm based adaptive filters (Mollaei, 2009) [7] was designed on SIMULINK and then implemented on a TMS320C6711 DSP board to investigate hardware and software system performance for different high and low frequency noises. A low power (0.156W) adaptive noise canceller (Ramakrishna & Kumar, 2013) [8] based on LMS algorithm was implemented in on Xilinx XC3s200 FPGA. The High Performance Computing (HPC-I) payload integrated into the Australian scientific mission satellite FedSat is designed and implemented (Visser, Dawood, & Williams, 2002) [9] on FPGA for space applications. Transposed form LMS adaptive filter architecture is proposed (Jones, 1992) [10] and its behaviour is compared with the standard LMS adaptive filter.

This paper gives emphasis mainly on the structure or the architecture used for the implementation of the adaptive filter. The adaptive filter was designed and simulated in SIMULINK and its hardware implementation was done using verilog HDL on the Xilinx XC3S500E Field programmable gate arrays (FPGA) [11]. The hardware implementation of adaptive filters is not easy as it requires dedicated hardware to meet the demanding time requirement and hence FPGA's are preferred over Microprocessors, microcontrollers and digital signal processors (DSP) chips because FPGA chips are reconfigurable and can process data and information simultaneously for different processing applications. This paper is organized into the following four sections. The first section briefly describes adaptive filtering. Next section is adaptive algorithm which includes description of filtering algorithms such as LMS, Sign-based algorithms. Next section is on tapped delay line. Following this section is design and implementation section. Simulation results, hardware analysis and discussion have been presented in the subsequent section and finally conclusions drawn from this comparative study have been summarized.

II. Adaptive Filtering

An adaptive filter consists of a digital filter and the adaption algorithm for updating the weights or coefficients. Figure 1 depicts the basic structure of adaptive filter [11]. The first block is a digital filter which can be an FIR or IIR filter. However, in general we use FIR filter because it is less complicated and highly stable [12]. The second block is a weight adaptation block which updates the filter weights or coefficients according to the adaptive algorithm to reduce the error signal. The coefficients of the adaptive filter are determined during a training sequence where a known data pattern is transmitted. The adaptive algorithm adjusts the filter coefficients to force the received data to match the training sequence data. After the training sequence is completed, the switches are put in the other position, and the actual data is transmitted. During this time, the error signal is generated by subtracting the output signal from the reference signal. The input signal is filtered to produce an output that is typically passed on for subsequent processing. This, signal is then passed on to a circuit to modify the parameters of the filter until the quality of the filter output is as good as possible.

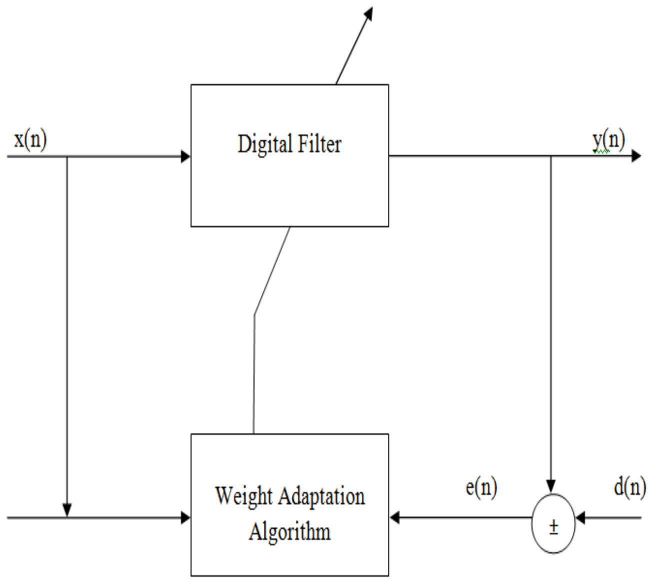

Fig. 1. Block diagram of an adaptive filter showing all the basic details and signals.

III. Adaptive ALGORITHMS

Adaptive algorithms are based on the minimization of a certain cost function (optimization problem). The most widely used cost function is the quadratic error. This cost function is optimum when errors follow a Gaussian distribution. There are a large number of adaptive algorithms such as LMS, NLMS, DLMS, BLMS, RLS, Leaky LMS, Sign-Data (SD), Sign-Error (SE) and Sign-Sign (SS) etc. The selection of an adaptive algorithm depends upon various factors such as filtering performance, application, hardware utilization, delay, clock speed and power. This paper concentrates mainly on four algorithms viz. LMS, SD, SE and SS.

A. LMS Algorithm

The LMS algorithm is similar to the steepest-descent method in which it adapts the weights by iteratively approaching the MSE minimum. During the n^{th} iteration, LMS algorithm updates the coefficients as follows:

$$w_{n+1} = w_n + \mu.\, e(n).\, x(n) \quad (1)$$

where, μ is the step size, $e(n) = d(n) - y(n)$ is the error signal, d(n) is the desired response, $y(n) = w_n^T.x(n)$ is the output, $x(n) = [x(n), x(n-1), \dots\dots, x(n-N+1)]^T$ is the input vector and $w_n = [w_n(0), w_n(1), \dots\dots, w_n(N-1)]^T$ is the weight vector of an n^{th} order LMS adaptive filter at the n^{th} iteration, respectively.

B. Sign Based LMS Algorithms

The importance of sign family LMS algorithm lies due to the fact that their implementation cost is much lesser as compared to their other counterpart algorithms. In this we replace the input regressor vector and the estimation error components of the update term by their signs that reduce computing time and dynamic range requirements by turning multiplications into bit shifts. Applying the sign function to the standard LMS algorithm returns the following three types of sign LMS algorithms namely, the sign-data, the sign-error and the sign-sign algorithms.

1. *Sign-Data Algorithm*
It is a version of the LMS algorithm in which the sign function to the error signal x(n). This algorithm updates the coefficients of an adaptive filter using the following equation:

$$w_{n+1} = w_n + \mu.\, e(n) \mathrm{sign}(x(n)) \quad (2)$$

2. *Sign-Error Algorithm*

In this algorithm the sign function to the input signal vector e(n). This algorithm updates the coefficients of an adaptive filter using the following equation:

$$w_{n+1} = w_n + \mu \mathrm{sign}(e(n)) x(n) \quad (3)$$

3. *Sign-Sign Algorithm*
Here the sign function is applied to both the error signal e(n) and the input signal vector x(n). This algorithm updates the coefficients of an adaptive filter using the following equation:

$$w_{n+1} = w_n + \mu \mathrm{sign}(e(n)) \mathrm{sign}(x(n)) \quad (4)$$

IV. Tapped Delay Line

The Tapped Delay line delays an input by the specified number of sample periods and outputs all the delayed versions. It is used to discretize a signal in time or resample a signal at a different rate. There is a predefined block for the tapped delay line in simulink library which is used in designing adaptive filter. The block has five parameters viz.

initial condition, sample time, number of delays, order output vector starting with and which can change its operation. The block accepts one scalar input and generates an output vector that contains each delay. For example, if we want to design a fifth order adaptive filter then the number of delays used must be four and it generates an output vector that contain each delay.

V. Design and Implementation

The present work has been carried out on SIMULINK and Xilinx ISE 14.5. The adaptive filter was designed and implemented in SIMULINK using various toolboxes and blocksets available in its library. Figure 2-3 shows the implementation of LMS adaptive filter having filter length 'm' by both conventional serial structure and tapped delay line serial structure. The other variants of LMS adaptive filter (Sign-Data, Sign-Error and Sign-Sign) are also implemented using the structures shown in the Figure 2-3. For the implementation of Sign- Data adaptive filter, we need to apply signum function so that it can approximate the input signal. For, Sign-Error adaptive filter signum function needs to be applied at the error signal and for Sign-Sign adaptive filter, signum function needs to be applied at both the input and error signal. After, the implementation of the adaptive filters using these algorithms by both the structures, the behavioral model of these adaptive filters was written using verilog HDL. The behavioral model was then simulated and synthesized using Xilinx ISE 14.5. After, that it was successfully downloaded to FPGA kit and converted into hardware.

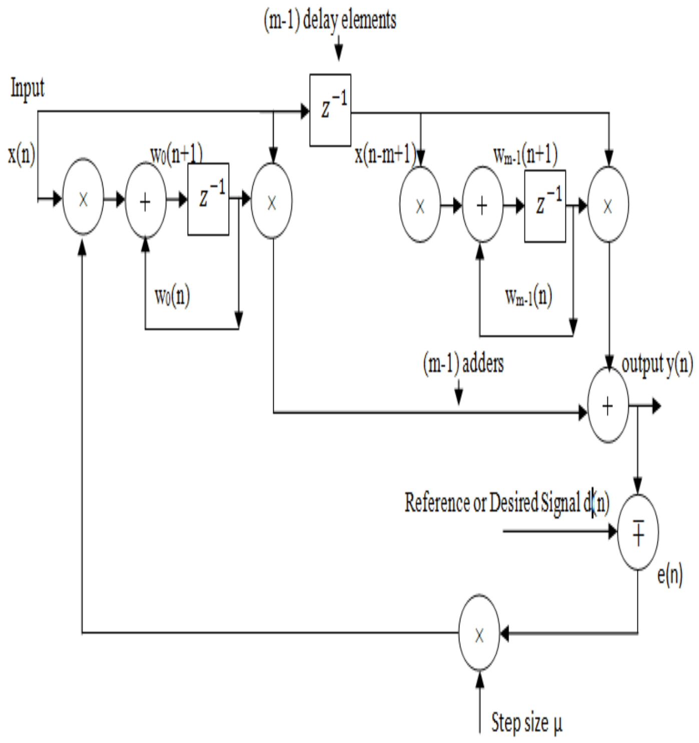

Fig. 2. Conventional Serial Structure of LMS Adaptive filter of filter length 'm'

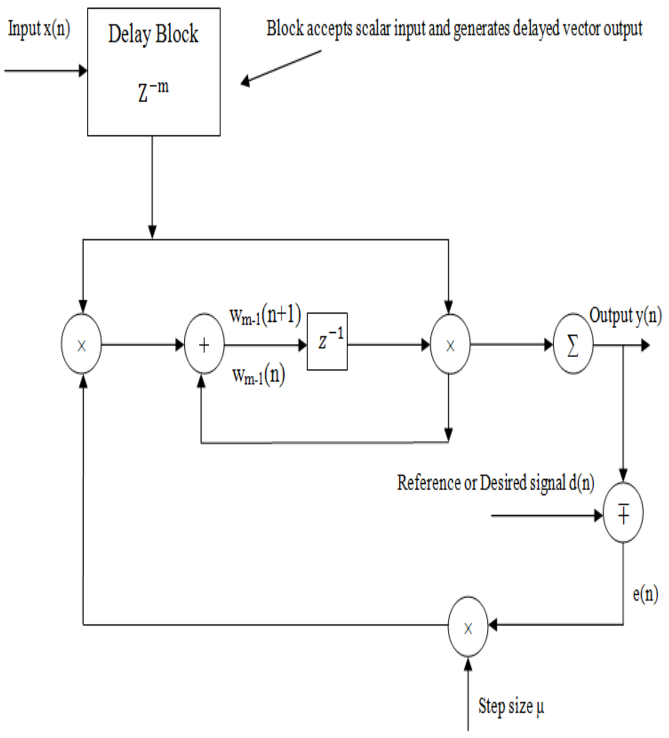
Input x(n)
Delay Block
z^-m
Block accepts scalar input and generates delayed vector output
Wm-1(n+1)
Wm-1(n)
z^-1
Σ
Output y(n)
Reference or Desired signal d(n)
e(n)
Step size μ

Fig. 3. Tapped Delay line Serial Structure of LMS Adaptive Filter of filter length 'm'

VI. Simulation Results and Hardware Analysis

From the hardware analysis it has been observed that tapped delay line serial filter implementation is better than conventional serial filter implementation in terms of delay (clock frequency), number of multiplexers, number of comparators and number of registers as shown in Table 1-4 and Figure 4-7. Since, there are less number of registers and multiplexers in tapped delay line serial implementation as compared to conventional serial implementation, there is less combinational delay between input and output in tapped delay line structures. Thus, for high speed applications tapped delay line filter implementation is preferred over conventional serial filter implementation. Further, on comparing all the algorithms (LMS, SD SE and SS), it has been found that all these algorithms contains the same number of multipliers, subtractor and registers but, the number of adders, multiplexer and comparators are different in each algorithm. The number of adders are same in both the implemented structures but there is a corresponding change in the number of adders when there is a change in adaptation algorithm. This result can be seen from Table 1 and Table 4. Also, it is found that the number of multiplexers and number of comparators are different in both the implemented structures and they are further changed when the algorithm for the implementation is changed. This result can also be seen from Table 1 and Table 4. Thus, by comparing the hardware utilization of both the structures it had been found that Sign-Sign algorithm is best in terms of hardware utilization as it contains least number of adders.

Algo. Name	Multiplier			Adder				Subtractor	Muxes	Registers	Comparators		
	16 x 16 bit	32 x 16 bit	Total	16 bit	32 bit	33 bit	Total	17 bit	32 bit 4:1	16 bit	>16 bit	<16 bit	Total
LMS	6	5	11	1	21	4	26	1	10	13	0	0	0
SD	6	5	11	1	11	4	16	1	15	13	5	5	10
SE	6	5	11	1	20	4	25	1	11	13	1	1	2
SS	6	5	11	1	10	4	15	1	16	13	6	6	12

Table 1. Hardware Utilization of Conventional Serial Implementation (order5)

Algo. Name	Clk. Freq (Max.) (MHz)	Period (min.) (ns)			Min. I/P arrival time before clk. (ns)			Max. O/P required time after clock (ns)			Max. Combinational Path Delay (ns)		
		Logic	Route	Total	Logic	Route	Total	Logic	Route	Total	Logic	Route	Total
LMS	14.386	51.167	18.346	69.513	24.52	7.493	32.013	3.863	0.420	4.283	4.490	1.313	5.803
SD	15.324	46.714	18.544	65.258	22.58	6.844	29.425	3.863	0.420	4.283	4.490	1.295	5.785
SE	14.778	48.649	19.018	67.667	20.08	6.818	26.906	3.863	1.017	4.880	4.490	1.309	5.799
SS	15.779	44.373	19.003	63.376	18.149	6.168	24.317	3.863	0.420	4.283	4.490	1.295	5.785

Table 2. Timing Analysis of Conventional Serial Implementation (order5)

Algo. Name	Multiplier			Adder						Subtractor	Muxes	Registers	Comparators		
	16 x 16 bit	32 x 16 bit	Total	16 bit	32 bit	33 bit	34 bit	35 bit	Total	17 bit	32 bit 4:1	16 bit	>16 bit	<16 bit	Total
LMS	6	5	11	1	21	2	1	1	26	1	6	12	0	0	0
SD	6	5	11	1	11	2	1	1	16	1	11	12	1	1	2
SE	6	5	11	1	20	2	1	1	25	1	7	12	1	1	2
SS	6	5	11	1	10	2	1	1	15	1	12	12	2	2	4

Table 3. Hardware Utilization of Tapped Delay Line Serial Implementation (order 5)

Algo. Name	Clk. Freq (Max.) (MHz)	Period (min.) (ns)			Min. I/P arrival time before clk. (ns)			Max. O/P required time after clock (ns)			Max. Combinational Path Delay (ns)		
		Logic	Route	Total	Logic	Route	Total	Logic	Route	Total	Logic	Route	Total
LMS	16.586	44.294	16.070	60.364	25.520	7.493	32.013	3.863	0.420	4.283	4.490	1.313	5.803
SD	18.882	38.041	14.921	52.962	22.581	6.684	29.425	3.863	0.420	4.283	4.490	1.292	5.782
SE	17.102	41.953	16.518	58.471	20.088	6.818	26.906	3.863	1.017	4.880	4.490	1.309	5.799
SS	19.581	35.770	15.369	51.069	18.149	6.168	24.317	3.863	1.017	4.880	4.490	1.287	5.777

Table 4. Timing Analysis of Tapped Delay Line Serial Implementation (order 5)

Further, from the timing analysis of both the implemented structures as shown in Table 2 and Table 4 it had been found that clock speed of tapped delay line structures are on higher side as compared to conventional serial implementation as shown in Figure 4. Also, clock speed gets changed when the algorithm used for the implementation of adaptive filter is changed (in both the implemented structures). The complete comparisons of all the algorithms in both the implementations are shown in Table 1-4

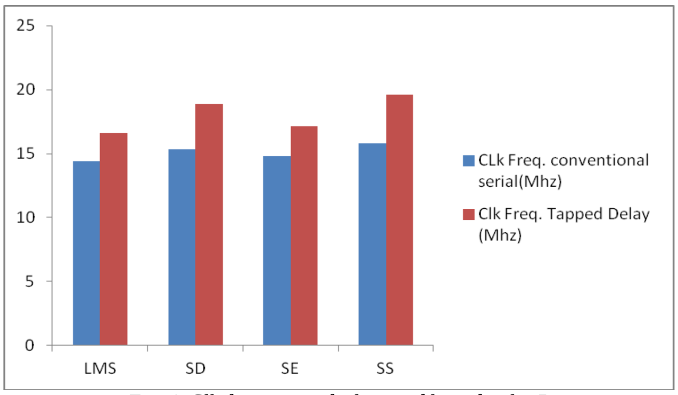

Fig. 4. Clk frequency of adaptive filter of order 5

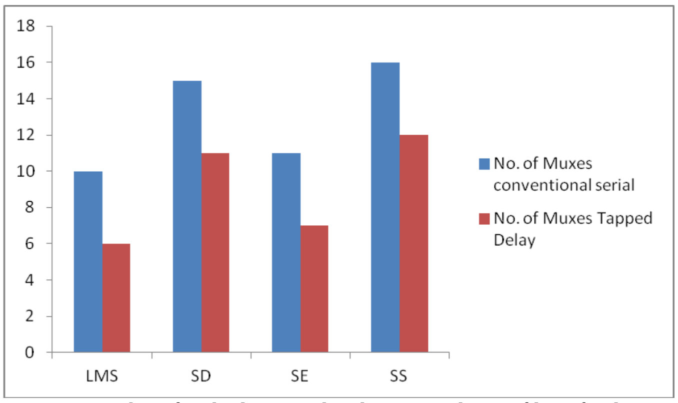

Fig. 5. Number of multiplexer used in designing adaptive filter of order 5

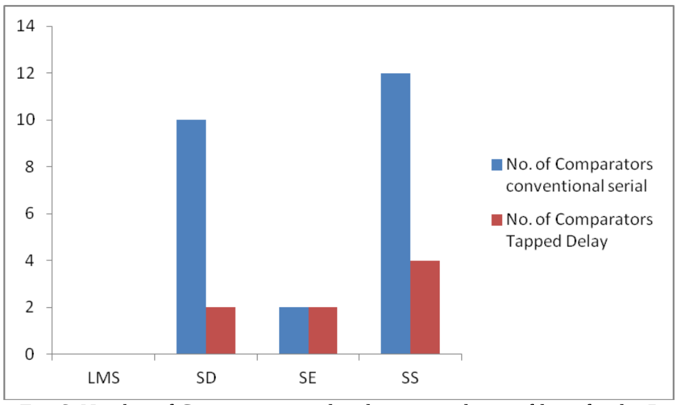

Fig. 6. Number of Comparators used in designing adaptive filter of order 5

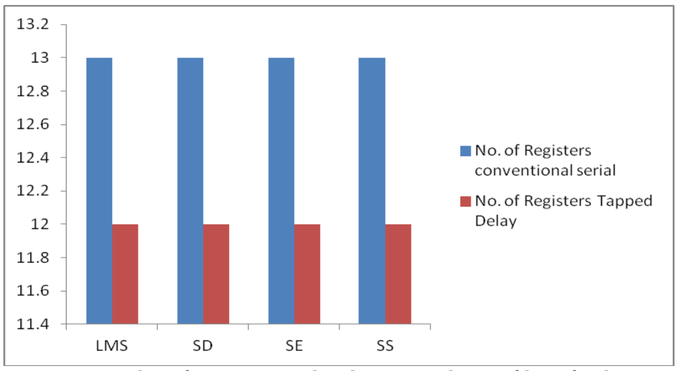

Fig. 7. Number of Registers used in designing adaptive filter of order 5

VII. Conclusion

Adaptive filters using all the four algorithms are implemented by both the structures i.e. conventional serial structure and tapped delay line serial structure. It has been found that tapped delay line filter is better in terms of throughput, clock speed, delay and hardware utilization. Thus, tapped delay line filter structures are better suited for high speed applications. Further, by comparing all the four algorithms it is found that sign-sign algorithm is best in terms of hardware utilization and clock speed as compared to other algorithms but, at the cost of slight degradation in the filtering performance. This is because of the presence of signum function which truncates both the input signal and the error signal either to value +1 or−1

References

[1] Nekouei.F, Talebi.N.Z, Kavian.Y.S, Mahani.A, “FPGA Implementation of LMS Self Correcting Adaptive Filter (SCAF) and Hardware Analysis,” in *Proc. 8th IEEE, IET International Symposium on Communication Systems, Networks and Digital Signal Processing*, 2012, pp. 1-5..

[2] Z. Gao, X. Zeng, J. Wang, J. Liu, "FPGA implementation of adaptive IIR filters with particle swarm optimization algorithm," in *Proc. 11th IEEE International Conference on Communication Systems (ICCS)*, Singapore, 2008, pp. 1364-1367.

[3] Rosado-Muñoz A, Bataller-Mompeán M, Soria-Olivas E, Scarante C, Guerrero-Martínez, JF, “FPGA Implementation of an Adaptive Filter Robust to Impulsive Noise: Two Approaches,” *IEEE Transactions on Industrial Electronics*, Vol. 58, No. 3, pp. 860-870, March 2011.

[4] R. Mustafa, M. A. Mohd Ali, C. Umat, and D. A. Al-Asady, "Design and implementation of least mean square adaptive filter on altera cyclone II field programmable gate array for active noise control," in *Proc. IEEE Symposium on Industrial Electronics and Applications (ISIEA)*, Kuala Lumpur, Malaysia, October 2009, pp. 479-484.

[5] A. Elhossini, S. Areibi, and R. Dony, "An FPGA implementation of the LMS adaptive filter for audio processing," in *Proc. IEEE International Conference on Reconfigurable Computing and FPGA's*, September 2006, pp. 1-8.

[6] M. Vella, and C.J. Debono, "The implementation of a high speed adaptive FIR filter on a field programmable gate array," in *Proc. IEEE Mediterranean Electrotechnical Conference (MELECON)*,Benalmadena, Spain, May 2006, pp. 113-116.

[7] Y. Mollaei, "Hardware implementation of adaptive filters," in *Proc. IEEE Student Conference on Research and Development (SCOReD)*, Malaysia, November 2009, pp. 45-48.

[8] V. Ramakrishna and T.A. Kumar, “Low Power VLSI Implementation of Adaptive Noise Canceller Based on Least Mean Square Algorithm,” in *Proc. 4th IEEE*

International Conference on Intelligent Systems, Modelling and Simulation, 2013, pp. 276-279.

[9] S. J. Visser, A. S. Dawood, and J. A. Williams, "FPGA Based Real-time Adaptive Filtering for Space Applications," in *Proc. IEEE International Conference on Field-Programmable Technology*, 2002, pp. 322-326.

[10] D. L. Jones, "Learning Characteristics of Transpose-Form LMS Adaptive Filters," *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 39, No.10, pp. 745-749, October 1992.

[11] Aditya Bali and Ravi Kumar, "FPGA Implementation of Variable Step Size Adaptive Filters for Signal De-Noising", *in Proc. Journal of Semiconductors Devices and Circuits*, Vol. 1, Issue 1, 2015, pp. 1-13.

[12] Aditya Bali and Ravi Kumar, "Performance of LMS Algorithm Derivatives in De-Noising: A Comparative Study", *in Proc. International Conference on Innovations in Electrical, Electronics and Computer Science Engineering (IEECSE-2014)*, Chandigarh, June 2014, pp. 6-12.