POWER EFFICIENT IMPLEMENTATION OF LEAST MEAN SQUARE ALGORITHM BASED FIR FILTER DESIGN USING FPGA

By

M. DEVIPRIYA * V. SARAVANAN ** N. SANTHIYAKUMARI ***

* M.E Student, Knowledge Institute of Technology, Anna University, Salem, Tamilnadu, India.
** PhD Schlor, Knowledge Institute of Technology, Anna University, Salem, Tamilnadu, India.
*** Professor & Head, Department of ECE, Knowledge Institute of Technology, Anna University, Salem, Tamilnadu, India.

ABSTRACT

This paper describes a high-speed and low-complexity implementation of FIR Filter using Least Mean Square Technique. Multiplex-Based Zero-Adaptation-Delay Structure and Two Adaptation Delay Structure for a direct LMS adaptive FIR filter is proposed. This paper describes that the proposed adder technique provides much faster convergence and lower complexity for obtaining lower area, power dissipation, high speed and lower propagation delay. The multiplexer circuits were schematized using the DSCH2 schematic design tool, and their layouts were generated with the Micro wind to VLSI layout CAD tool. The parameter analyses were performed with a BSIM4 analyzer. The proposed multiplex -based filters are used to carry save adder as well as other existing adder circuit in terms of power dissipation, propagation delay, latency, and throughput. Our proposed structure involves minimum power. Finally the simulations are done using Xilinx ISE design suite to get power and implemented on Spartan 3E FPGA kit.

Keywords: Adaptive Filters, Critical-Path Optimization, Least Mean Square Algorithms, LMS Adaptive Filter.

INTRODUCTION

Finite Impulse Response (FIR) filters are the most popular filters which are implemented in software. FIR filter is usually implemented by using a series of delays, multiplexer, and adders to create the filter's output [13]. An adaptive FIR filter with Least Mean Square (LMS) algorithm was developed to reduce the power. *Mux* based formulation of LMS algorithm is used to reduce the area where both, convolution operation to compute filter output and correlation operation to compute weight-increment term could be performed by using the same LUT. Thus a mux-based implementation of adaptive FIR filter is highly efficient [1][4]. The Mux based Least Mean Square adaptive FIR filter is used for implementing the noise cancellation because of its simplicity and also it is suitable for real-time applications.

1. Prior Work

Adaptive digital filters are widely used in many Digital Signal Processing (DSP) application areas, e.g., noise cancellation, channel estimation, channel equalization, system identification and echo cancellation etc. The tapped-delay-line, Finite-Impulse-Response (FIR), adaptive filter weights are updated by using Least-Mean-Square (LMS) algorithm [12].The LMS algorithm may also be called as steepest decent algorithm. The LMS adaptive filter is most popular not only due to its high performance and low-complexity, but also due to its more stable and satisfactory convergence performance [9]. Due to very important applications of increasing constraints on area, speed, time, and low power complexity, it is efficient to implement the LMS adaptive filter and is still quite very important. To implement the LMS algorithm, one has to update the filter weights during each sampling period using the estimated error, which is equal to the difference between the current filter output and the desired response.

The weights of the LMS adaptive filter during the nth iteration

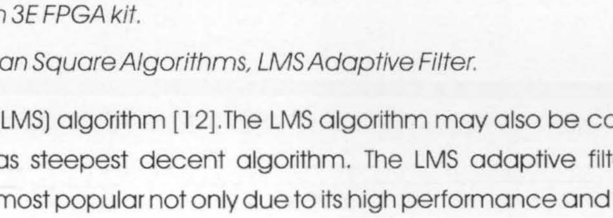


Figure 1. Block diagram of Least Mean Square Filter

are updated equations which are

$W(n+1) = w(n) + u\,e(n).x(n)$ (1)

Where

$e(n) = D(n) - Y(n)$ (2)

$Y(n) = W^T . X(n)$ (3)

Here the input vector x (n) and Weight vector W (n) at the nth iteration are given by,

$X(n) = [x(n, x(n-1) \ldots x(n-N-1)]$ (4)

$W(n) = [w(n0), w(n1) \ldots w(n-N-1)]$ (5)

Where

D (n) = desired response

Y (n) = Filter output of the nth iteration,

E (n) = the error computed during the nth iteration, which is used to update the weights, and is the convergence factor or step-size. Assume the step size to be a positive number and the number of weights N is used in the LMS adaptive FIR filter. The structure of a conventional LMS adaptive filter is shown in Figure 1. All weights are updated synchronously in each cycle, which are used to compute the output according to (1), direct-form realization of the FIR filter for implementation. However, the direct-form LMS adaptive filter is still to have a long critical path due to an inner product terms computation to obtain the filter output [12]. This is mainly based on the assumption of arithmetic operation that starts only after the complete input operands are generated.

2. Methodology

Using least mean square algorithm, is more stabile and simplie. Figure 2 shows the multiplex based direct form of least mean square technique for weight updating for which

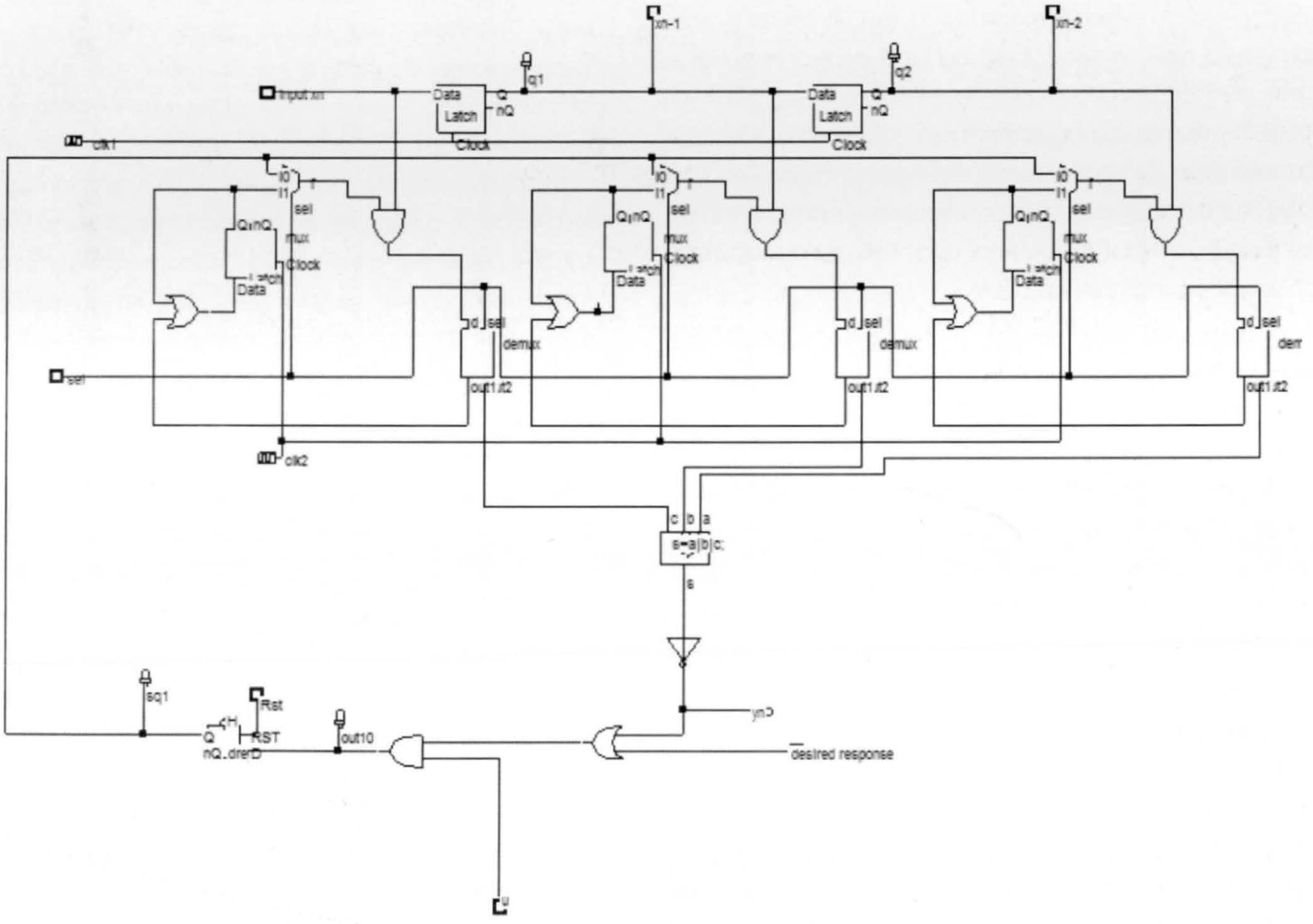


Figure 2. Mux based 3 Tap LMS Adaptive FIR Filter

error estimation is also very less [1],[3]. Compared to the existing design, the proposed structure is more efficient.

There are two main computing blocks in LMS adaptive FIR filter, namely,

- The error-computation block to compute the filter output
- The weight-update block to compute the weight increment terms.

Both blocks are most common in the error-computation and weight-update blocks for area-intensive components: the multipliers, weight registers, and tapped-delay line. The adder tree is used for weight updating, which constitutes only a small part of the circuit that vary to two computing blocks. For the zero adaptation-delay implementation, the computation complexity for both weight updated and error computational blocks is required to be performed in the same period of the cycle. Weight updating is performed in every iteration since the structure is non-pipelined type, weight updating and error computation cannot occur simultaneously. Therefore, the multiplications of both these blocks could be multiplexed by the same set of multipliers, while the same registers could be used for both the blocks. If error computation is performed in the first period of the half cycle, weight update is performed in the second period of the half cycle.

The proposed structure of a multiplex based zero-adaptation-delay structure for a direct form of tapped LMS adaptive filter, consists of multipliers. The input samples are fed to the multipliers from a common tapped delay line. The weight values (which are used for storage purpose in registers) and the estimated error value (after right-shifting by a fixed number of locations to realize multiplication by the step size μ) are given to the multipliers as the other input through a 2:1 multiplexer. The proposed structure requires adders for the updating of weights and adder tree to add the multipliers output for computation of the filter output [11]. Also, it requires a subtraction operation to compute the error value and 2:1 de-multiplexors to move the product values either towards the adder tree or to move to the weight-update circuit. All the multiplexors and de-multiplexors are required to be controlled by a clock signal.

The weight value of stored registers in the delay line are clocked at the rising edge of the clock pulse and remain unchanged for a full clock period since the structure is required to take one new sample in every clock cycle. During the first half period of each clock cycle, weight values are stored in different registers and fed to the multiplier through the multiplexors which is required to compute the filter output. The product words are fed to the adder tree though the de-multiplexors. The filter output is computed by the adder tree and the error value is computed by a subtraction operation. Then the estimated error value is shifted right to obtain μ.e(n) and broadcasted to all N multipliers in the weight-update circuits. Note that the LMS adaptive filter requires at least one delay at a suitable location to break the recursive loop [4], [13]. A delay is inserted either after the adder tree, before the computation, or after the computation. If the delay is placed just after the adder tree, then the critical path shifts to the weight-updating circuit and gets increased by time delay. Therefore, we should place the delay after computation of e (n) or μ.e (n), but preferably after μ.e (n) computation to reduce the register width. The first half-cycle of each clock period ends with the computation of μ.e (n), and during the second half cycle, the value is fed to the multipliers though the multiplexors to calculate the de-multiplexed output to be added to the stored weight values to produce the new weights. The computation during the second half of a clock period is completed once a new set of weight values is computed. The updated weight values are used in the first half-cycle of the next clock cycle for computation of the filter output and for subsequent error estimation. When the next cycle begins, the weight registers are also updated by the new weight values.

Therefore, the weight registers are also clocked at the rising edge of each clock pulse. The time required for error computation is more than weight updating. The system clock period could be less if we just perform these operations one after the other in every cycle. This is possible since all the register contents can change once at the beginning of a clock cycle, but we cannot exactly determine when the error computation is over and when weight updating is completed. Therefore, the authors need to perform the error computation during the first half-cycle and the weight updating during the second half-cycle.

Accordingly, the clock period of the proposed structure is twice the critical-path delay for the error-computation block.

2.1. Two Adaptation Delays

The proposed structure for a two-adaptation-delay LMS adaptive filter is shown in Figure 3, which consists of three pipeline stages, where the first stage ends after the first level, the adder tree in the error-computation unit, and the rest of the error-computation block comprises the next pipeline stage[6][7][8]. Figures 4 to 7 show the structure of LMS Adaptive FIR Filter.

The weight-update block comprises of the third pipeline stage. The two-adaptation-delay structure involves N/2 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 [2], [10] given by

$$T = MAX\{T_{MULT} + \Delta, T_{AB}\} \quad (6)$$

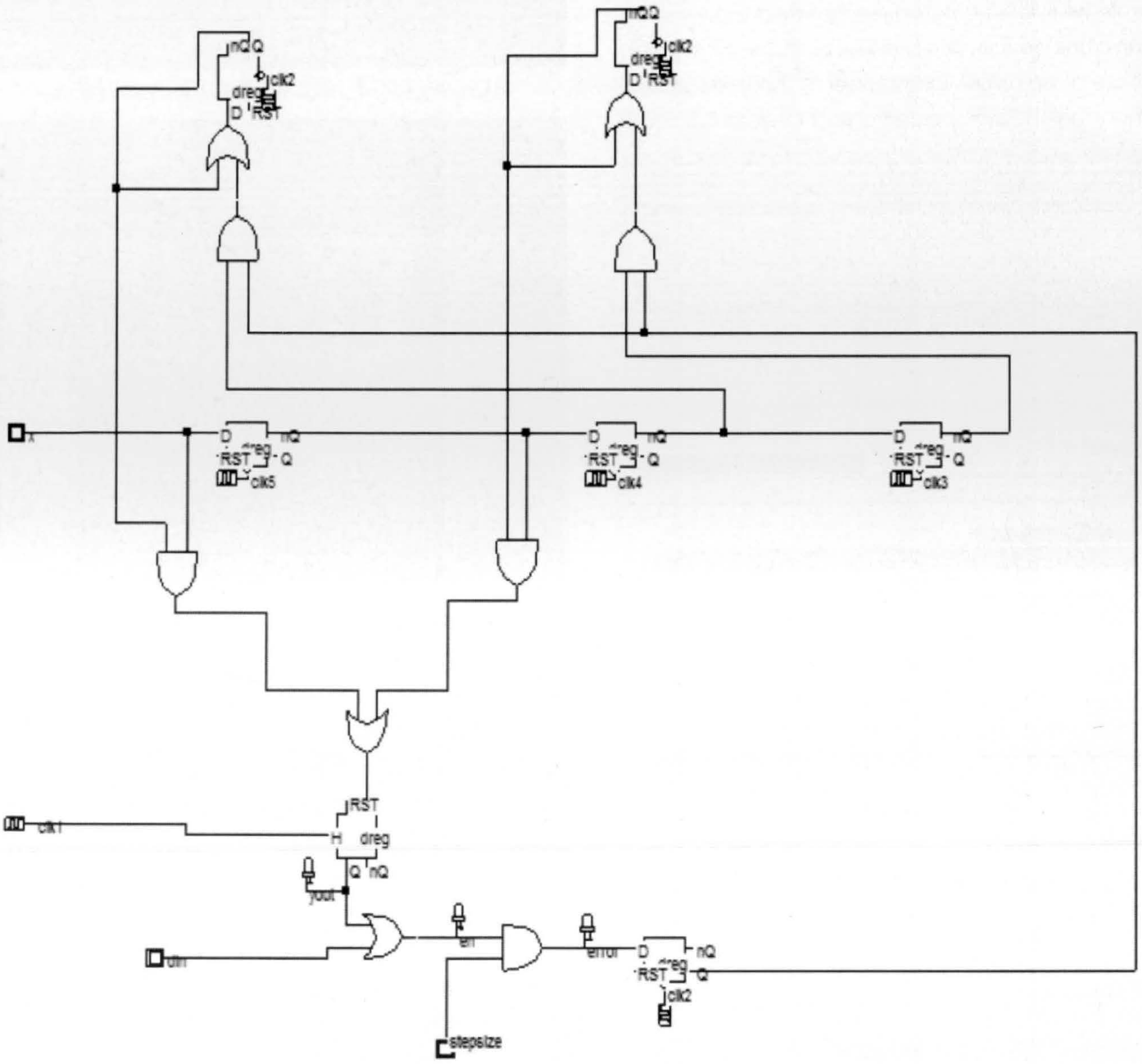


Figure 3. Two Adaptation Delay LMS Adaptive FIR Filter

Where T_{AB} refers to the adder-tree delay of stages to add words along with the time required for subtraction in the error computation

3. Simulation Result

The simulation was done by using the DSCH2 schematic design tool, and their layouts were generated with the Micro wind 2 VLSI layout CAD tool and Xilinx ISE 12.1 tool. The parameter analyses were performed with a BSIM4 analyzer.

Resource Utilisation

Table1 and 2. Shows the Device Utilization summary which represents the implementation result mentioning resource utilization of the device Xilinx SPARTAN 3E kit. The synthesized outputs are given below. The number of gates required to implement the utilization percentage of the total hardware

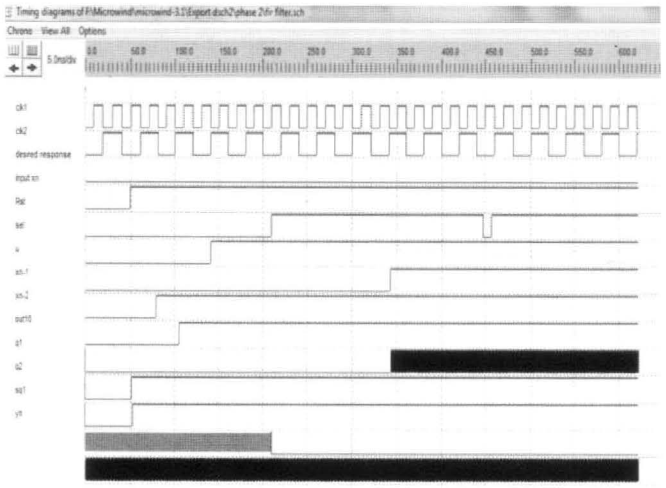


Figure 4.Timing Diagram for Mux Based Direct form LMS Adaptive Filter

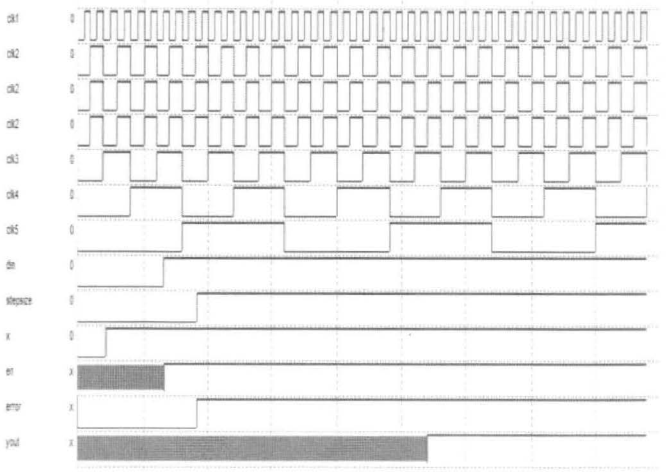


Figure 5.Timing Diagram for Two Adaptation Delay LMS Adaptive FIR Filter

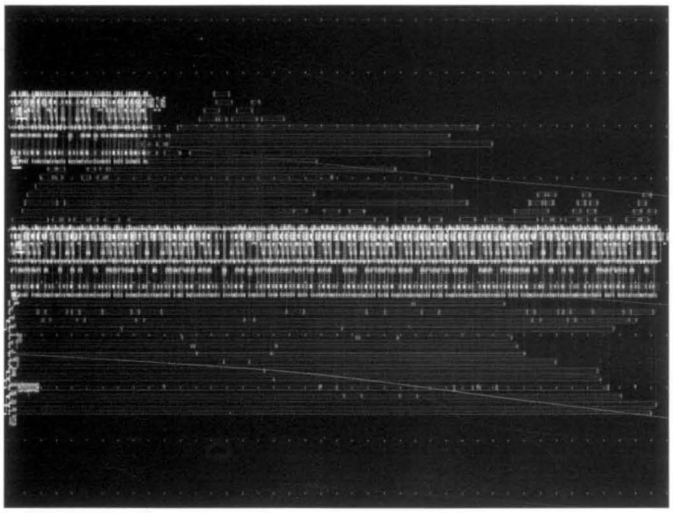


Figure 6. Layout for Mux Based LMS Adaptive FIR Filter

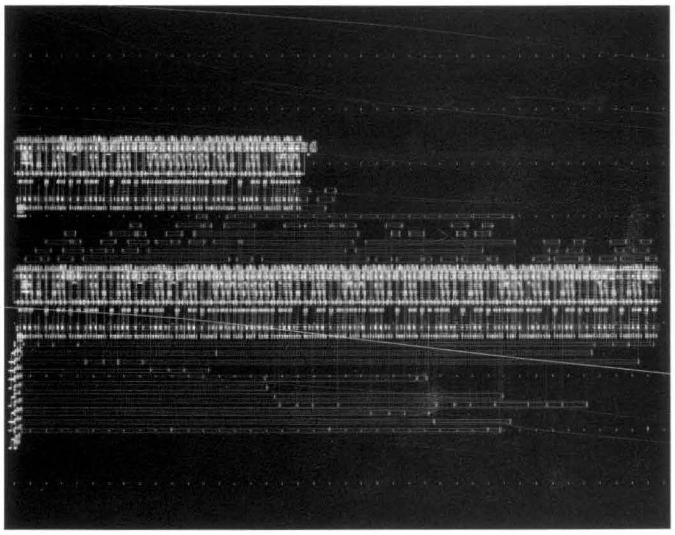


Figure 7. Layout for Two Adaptation Delay LMS Adaptive FIR Filter

in the FPGA kit is summarized in Table1.The number of Slice flip-flops, Look-up table, and number of bonde IOBs, number of GCLKs are summarized in Table 1 and 2.

The results prove that the Mux based LMS adaptive filter implemented for the Adaptive Noise Cancellation satisfy the desired demand by providing stability of Convergence in a non-stationary environment. Efficient implementation is justified by the resource utilization efficiency as shown in the Table1 and 2. Figures 8 to 11 show the simulation results of LMS adaptive filter.

The experimental setup for the implementation of mux based LMS adaptive filter and two adaptation delays is shown in Figure 12 and 13. The experimental set up is done

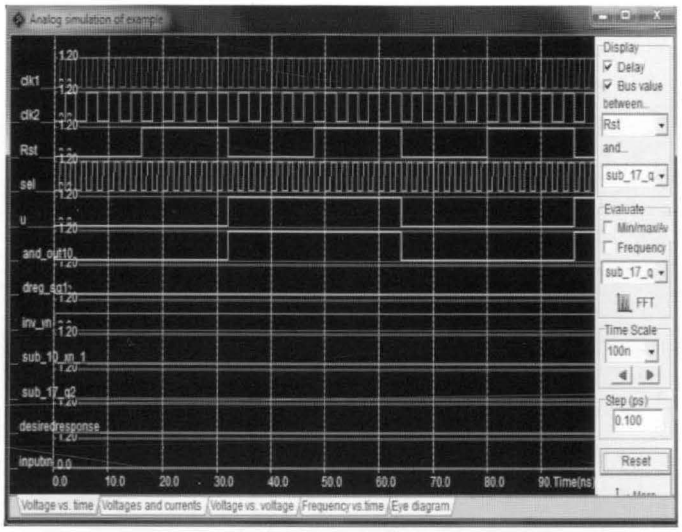


Figure 8. Simulation result for mux based 3 tap LMS adaptive FIR filter

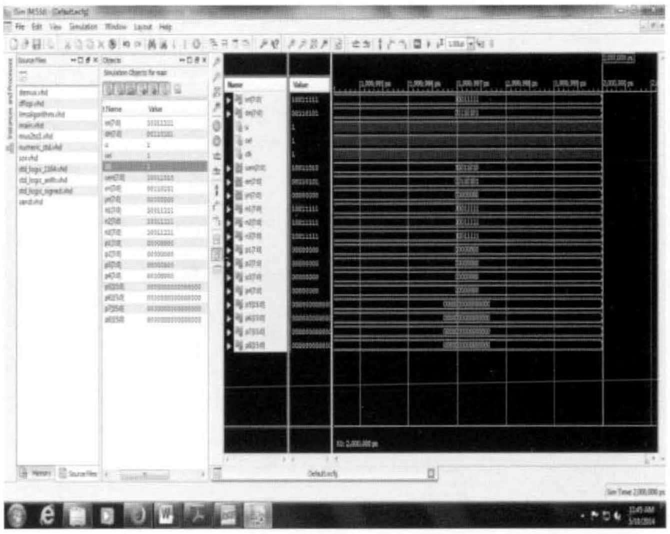


Figure 10. Simulation results for Mux based LMS adaptive FIR filter using Xilinx ISE v12.1

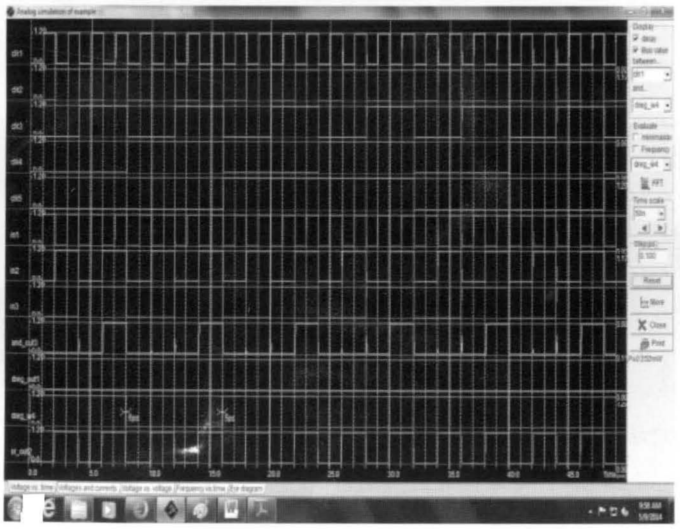


Figure 9. Simulation result for two adaptation delay LMS adaptive FIR filter

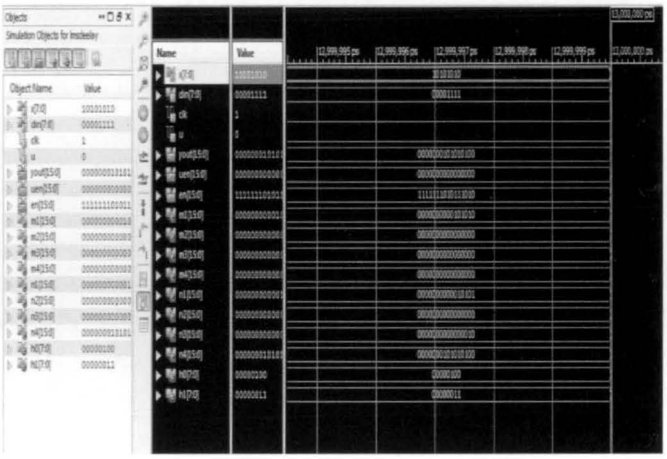
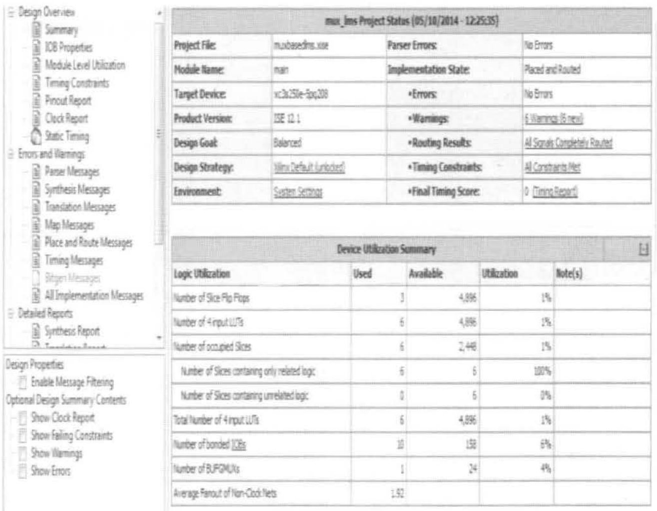


Figure 11. Simulation results for two adaptation delay LMS adaptive FIR filter using Xilinx ISE v12.1



mux_lms Project Status (05/10/2014 - 12:25:35)			
Project File:	muxbasedlms.xise	Parser Errors:	No Errors
Module Name:	main	Implementation State:	Placed and Routed
Target Device:	xc3s250e-5pq208	•Errors:	No Errors
Product Version:	ISE 12.1	•Warnings:	6 Warnings (6 new)
Design Goal:	Balanced	•Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	All Constraints Met
Environment:	System Settings	•Final Timing Score:	0 (Timing Report)

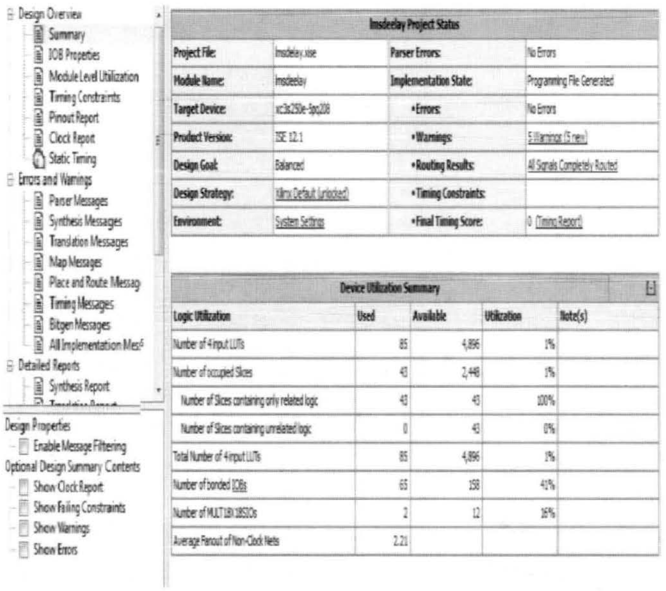
Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	3	4,896	1%	
Number of 4 input LUTs	6	4,896	1%	
Number of occupied Slices	6	2,448	1%	
Number of Slices containing only related logic	6	6	100%	
Number of Slices containing unrelated logic	0	6	0%	
Total Number of 4 input LUTs	6	4,896	1%	
Number of bonded IOBs	10	158	6%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	1.52			

Table1. Device utilization summary for Mux based LMS adaptive filter

using FPGA SPARTAN 3E kit along with Computer loaded with Xilinx System Generator Software as shown in figure 12 and 13.

3.1 The SPARTAN-3E FPGA

The FPGA implementation of Xilinx Spartan-3E XC3S250E is shown in Figure 6. Spartan-3E devices contain a two-dimensional row and column based architecture to implement custom logic. For implementing Mux based LMS filter and two adaptation delay LMS adaptive FIR filter, and the number of operations for an N-tap filter have been reduced to 2* N multiplications and N additions per coefficient update [5].



Imsdeelay Project Status			
Project File:	lmsdelay.xise	Parser Errors:	No Errors
Module Name:	lmsdelay	Implementation State:	Programming File Generated
Target Device:	xc3s250e-5pq208	•Errors:	No Errors
Product Version:	ISE 12.1	•Warnings:	5 Warnings (5 new)
Design Goal:	Balanced	•Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	
Environment:	System Settings	•Final Timing Score:	0 (Timing Report)

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	85	4,896	1%	
Number of occupied Slices	43	2,448	1%	
Number of Slices containing only related logic	43	43	100%	
Number of Slices containing unrelated logic	0	43	0%	
Total Number of 4 input LUTs	85	4,896	1%	
Number of bonded IOBs	65	158	41%	
Number of MULT18X18SIOs	2	12	16%	
Average Fanout of Non-Clock Nets	2.21			

Table 2. Device utilization summary for two adaptation delay LMS adaptive FIR filter

Conclusion

The FIR filter is designed with the direct form of least mean square technique and benefits the low complexity. The mux based FIR filter using LMS technique is implemented to sample the input data at both the error estimation and weight updating scheme. It is suitable for low power applications, and active noise cancellation applications. In a constant throughput rate system, it is possible to reduce half of the power by replacing the mux in multiplier. The proposed MUX based FIR filter consumes power of 0.236 mW and for two adaptation delay LNS adaptive filter it is

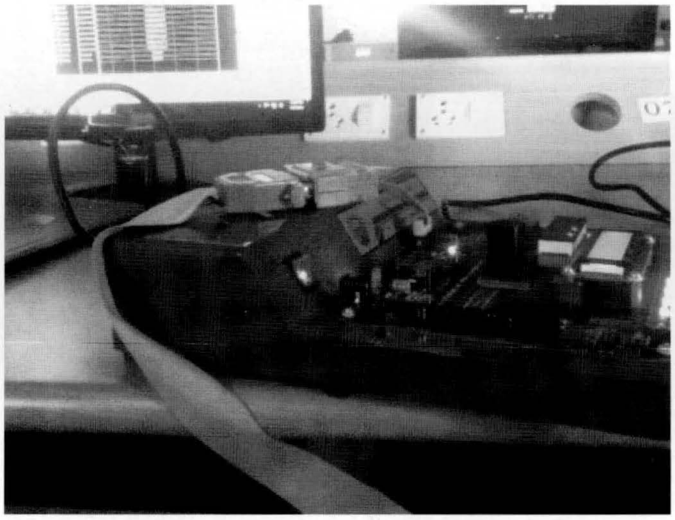


Figure.12. Experimental set up for implementation of Mux based LMS adaptive FIR filter using Xilinx ISE v12.1 in Spartan 3E

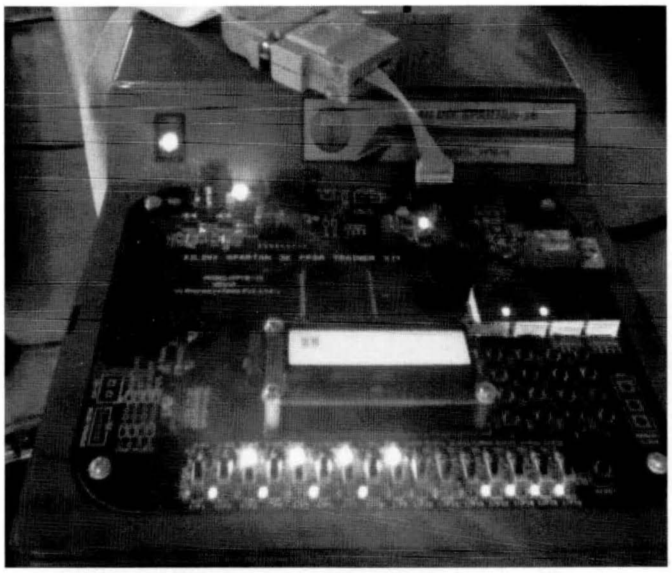


Figure13. Experimental set up for implementation of two adaptation delay LMS adaptive FIR filter using Xilinx ISE v12.1.

0.252mW. It consumes 8% ~52% less dynamic power .

References

[1]. **S. Y. Park and P. K. Meher (2013)** "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.

[2]. **Pramod Kumar Meher and Sang Yoon Park (2013)** "Critical-Path Analysis and Low-Complexity implementation of the LMS Adaptive Algorithm" in Proc. *IEEE transactions on circuits and systems.*

[3]. **P.K.Meher and S.Y.Park (2011)**, "Low adaptation-delay LMS adaptive filter Part-II: An optimized architecture," in Proc. *IEEE Int. Midwest Symp. Circuits Syst.*

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

[5]. **Xilinx (2009)**, "Spartan-3E FPGA Family Complete Data Sheet", *Xilinx,* Inc.

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

[7]. **Y.Yi,R.Woods, L.K.Ting, and C.F.N.Cowan (2005)**, "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.

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

[9]. **S. Haykin and B. Widrow (2003),** "Least-Mean-Square Adaptive Filters", Hoboken, NJ, USA: Wiley-Inter science.

[10]. **L.D.Van and W. S. Feng (2001),** "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.

[11]. **M. D. Meyer and D. P. Agrawal (1990),** "A modular pipelined implementation of a delayed LMS transversal adaptive filter," in Proc. *IEEE Int. Symp. Circuits Syst.*, pp. 1943–1946.

[12]. **G.Long, F.Ling, and J.G.Proakis (1989),** "The LMS algorithm with delayed coefficient adaptation" *IEEE Trans. Acoust., Speech, Signal Process.*, Vol. 37, No. 9, pp. 1397–1405.

[13]. **B. Widrow and S. D. Stearns (1985),** Adaptive Signal Processing. Engle-wood Cliffs, NJ,USA: Prentice-Hall.

ABOUT THE AUTHORS

M. Devipriya is presently doing ME degree in VLSI Design from Anna University, Chennai in Knowledge Institute of Technology, Salem. She received her B.E degree in Electronics and Communication Engineering from Tamilnadu College of Engineering Coimbatore in 2011. She is a member in IEEE and IETE. Her areas of interest include Low power VLSI Design and Digital system design.

V. Saravanan is presently pursuing Ph.D. in Adaptive signal from Anna University, Chennai. He received his B.E degree in Electronics and Communication Engineering from Anna University, Chennai and M.E in VLSI Design from the same University. His current research interest includes VLSI design and Digital signal processing.

Dr. N. Santhiyakumari is currently working as a Professor and Head in the ECE department, Knowledge Institute of Technology, Salem. She received her B.E degree in Electronics and Communication Engineering from Madras University, Chennai in Government College of Engineering, Salem in 1997 and M.Tech., degree in Advance Communication Systems from SASTRA University, Tanjavur in 2002. She has been awarded with Ph.D. degree for her work in the Biomedical Image Processing area by Anna University, Chennai in 2008. She has 17 years of teaching and 10 years of Research experience. She is guiding 8 Ph.D. research scholars registered under Anna University, Chennai in the field of Biomedical Image Processing, Networking, Communication and VLSI. She is an active reviewer for the peer reviewed International Journals like Journal of Medical Systems and Journal of Neuroradiology (Springer). She is a Life Member in ISTE, IEEE, and IETE. Her areas of interest include Biomedical Image Processing And Applications Of Soft Computing Techniques in Image Analysis.