FPGA Implementation of LMS Adaptive Filter using Distributed Arithmetic

[1]Dr.N.N.Kasat,[2]Ms.Priti.H.Gupta
[1]Professor in Electronics & Telecommunication, [2]PG Scholar
[1]Electronics & Telecommunication,
[1]Sipna College of engineering & technology, Amravati, India

Abstract: Digital filters are used to modify the characteristic of signals in time and frequency domains hence have been recognized as primary digital signal processing element. Adaptive filters are widely used in many signal processing applications, such as echo cancellation, where time varying noise needs to be removed from desired signals. The conventional techniques like pipelining and parallel processing can be employed to increase the throughput. Amongst the existing adaptive filters, least mean square (LMS) based finite impulse response (FIR) adaptive filter is the most popular one due to its inherent simplicity and satisfactory convergence performance It is comprised of a finite-impulse-response (FIR) filter, coefficients of which are updated based on the input samples using least mean square (LMS) Algorithm .

This paper delegates a novel pipelined architecture implementation of adaptive filter based on distributed arithmetic (DA) for typical noise cancellation applications low-power, high-throughput, and low-area. The throughput rate of the design is increased by update of parallel lookup table (LUT) and implementation of filtering and weight-update operations concurrently. The conditional signed carry-save accumulation is used in order to reduce the sampling period and area complexity for DA-based inner-product computation. Reduction of power consumption is achieved in the proposed design by utilizing a fast bit clock for carry-save accumulation but a much slower clock for all other operations. It consists of multiplexors of same number, LUT of small size, and nearly adders of half number compared to the existing design based on DA. From synthesis results & comparison table, it is clear that the throughput has increased & time delay has reduced than previous DA-based adaptive filter for filter lengths N = 2. Designing of FIR filter is done using VHDL and synthesized using Quartus II synthesis tool and Modelsim SE 6.2 simulator**.**

***Index Terms* - Distributed arithmetic, Adaptive FIR, LMS, LUT, FPGA, MAC**

I. INTRODUCTION

Adaptive filters have attracted the attention of many researchers during the last decades, due to their property of self-designing. They are used in numerous applications that include acoustic echo cancellation, noise cancellation, channel equalization and many other adaptive signal processing applications. An adaptive filter is a time-variant filter whose coefficients are adjusted in such a way so as to optimize a cost function or to satisfy some predetermined optimization criterion. They can automatically adapt (self-optimize/self-adjust) regardless of varying environments and changing system requirements. They can be suitably trained to perform specific type of filtering and decision-making tasks according to some updating equations which behave as training rules for the adaptive filter

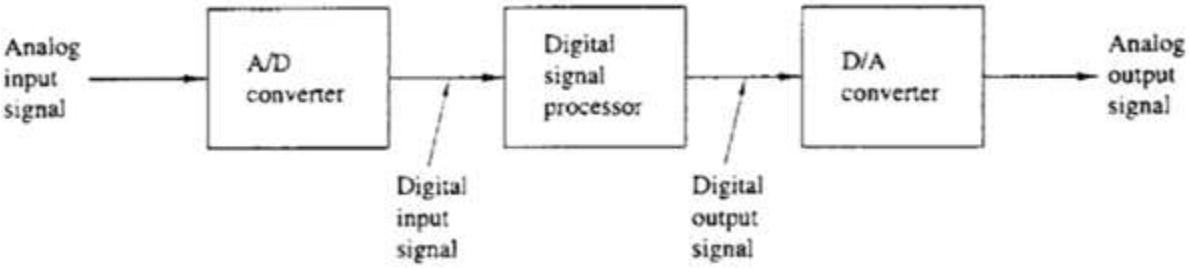


Figure 1.1 Block diagram of a digital signal processing system.

FIR Filter plays an important role in digital signal processing which could be used to as extraction and Interpolation. It is usually implemented by using a series of delays, multiplexer, and adders to create the filter's output. Generally, these filters consist of several multiply-and-accumulate (MAC) unit depending upon tap-size. In order to operate the filter system at high sampling rate, the critical path must be reduced. The conventional techniques like pipelining and parallel processing can be employed to increase the throughput. But, it comes at the cost of increased logic complexity, chip area usage, and power consumption

Adaptive filters are time-varying since their parameters are continually changing in order to meet a performance requirement. The hardware implementation requires several of performances such as high speed, low power dissipation, small chip area and good convergence characteristics. Among many adaptive filter algorithms, the least-mean-square (LMS) algorithms have been used because of their relatively small computational complexity of 2L, where L is the filter length. The algorithm uses a gradient descent to estimate a time varying signal. In this algorithm the filter weights are updated for tracking the desired filter output using the error information. Distributed arithmetic is a powerful technique for reducing the size of a parallel hardware multiply-accumulate that is well suited to FPGA designs. The DA targets the products of sums which cover all filtering application and frequency transfer functions. It uses Look- up Table (LUT) which stores the constant coefficients of FIR Filter. When number of taps increases, LUT grows exponentially. Unlike MAC-based adaptive filter, it is more effective technique for realizing large tap-sized filters. This is due to pre-computation of filter partial products and storing them in LUT. As a result, it occupies relatively smaller area as compared with MAC- based design. When tap-size of filter increases, the time required to update LUT contents grows rapidly. Thus a DA based implementation of adaptive filter is highly efficient.

The FPGA platform provides high performance and flexibility with the option to reconfigure. It allow applications to run in parallel, so that filtering, correlation, and many other applications can all run simultaneously. It can offer 10 to 1000 times the performance of the most advanced digital signal processor at similar or even lower costs. Nowadays, the use of FPGAs is increasing. Moreover, an FPGA is more efficient in power consumption, an advantage for battery-operated systems, and, for the same application, requires less clock system speed compared to a DSP or a general-purpose processor, offering better electromagnetic compatibility properties. In this paper, FPGAs are the target hardware used.

II. LITERATURE REVIEW

Adaptive filters play a very crucial role in signal processing. LMS algorithm is a broadly used adaptive algorithm for its robustness and low hardware complexity .this chapter reviews the different implementations of distributed arithmetic based adaptive digital FIR filter design using various LMS algorithms.

Mohd Tasleem Khan et. al. [1] presented technique to store possible filter partial products in a look-up table (LUT) followed by a shift- accumulation (SA) unit. Usually, all the address location of LUT needs to be re-calculated in every iteration. In this paper, author proposed a new strategy for updating the LUT contents without rotation of addresses in successive iterations. This results in a low complexity implementation with high speed. The proposed technique employs random-access memory (RAM) based LUT for storing offset binary coding (OBC) combinations of input samples and filter weights. Neil Woolfries et. al. [2] presented the technique for the use of dynamically reconfigurable FPGAs for the implementation of real-time, adaptive stack filter algorithms. The approach adopted in their paper is based on the use of stack filters that avoid these difficulties by employing logical algorithms that do not rely on any arithmetic functions. Stack filters for adaptive applications are suited to a hardware implementation on reconfigurable FF'GAs.

.Devipriya et. al. [3] presented technique to explore the power consumption for the architecture of Finite Impulse Response (FIR) adaptive filter. An adaptive FIR filter with Block Least Mean Square (BLMS) algorithm was developed to reduce the power. Distributed arithmetic (DA) based formulation of BLMS 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. GAO Jinding et. al. [4] presented the direct form FIR linear-phase low-pass filter using Kaiser Window function was designed out based on DSP Builder system modelling approach. FIR filters implemented on FPGA are used to achieve by encoding the underlying hardware description language, its development efficiency of this method is very low. In this paper, Kaiser Window function designing method based on DSP Builder system modelling was used to develop a direct linear phase FIR low-pass filter implemented on FPGA,.

Jyothirmayi Alahari et. al. [5] delegated a novel pipelined architecture implementation of adaptive filter based on distributed arithmetic (DA) for low-power, high-throughput, and low-area. Compared to the best of other existing designs their proposed design is better for area and power consumption. Offset binary coding is popularly used to reduce the LUT size to half for area-efficient implementation of DA. G.Selvapriya et. al. [6] given solution to the problem DA formulation employed for two separate blocks weight update block and filtering operations requires larger area and is not suited for higher order filters therefore causes reduction in the throughput. The direct form configuration on the forward path of the FIR filter results in a long critical path due to an inner-product computation to obtain a filter output. Therefore, when the input signal has a high sampling rate, it is necessary to reduce the critical path of the structure so that the critical path could not exceed the sampling period. These pipeline implementation of LMS-based ADF uses correction terms for updating the filter weights of the current iteration calculated from the error corresponding to a past iteration this briefs Delayed LMS (DLMS) algorithm.

Kalaiarashi.K et. al. [7] presented a novel pipelined architecture for low-power, high-throughput, and low-area implementation of adaptive filter based on distributed arithmetic (DA). The throughput rate of the proposed design is significantly increased by parallel lookup table (LUT) update and concurrent implementation of filtering and weight-update operations. Reduction of power consumption is achieved in the proposed design by using a fast bit clock for carry-save accumulation but a much slower clock for all other operations. It involves the same number of multiplexors, smaller LUT, and nearly half the number of adders compared to the existing DA-based design.

III. PROPOSED WORK

It is comprised of a finite-impulse-response (FIR) filter. The FIR architecture used as the unknown system and as the filter in adaptive algorithm is designed and implemented on FPGA. The weights of FIR filter are updated using least mean square (LMS) algorithm due to its simplicity and satisfactory convergence. Generally, these filters consist of several multiply-and-accumulate (MAC) units depending upon tap-size. The direct form configuration of FIR filter occupies more area and provides fewer throughputs due to physical multipliers involved in MAC units. In order to operate the filter system at high sampling rate, the critical path must be reduced. The conventional techniques like pipelining and parallel processing can be employed to increase the throughput. Figure shows the block diagram of the FIR architecture for LMS adaptive filter. During the process of LMS filter verification, the value of step size parameter, µ is varied to measure the convergence speed with different step size (µ is varied from 1/2 to 1/248). The process of adaptive algorithm and the function of each block are summarized as follows:

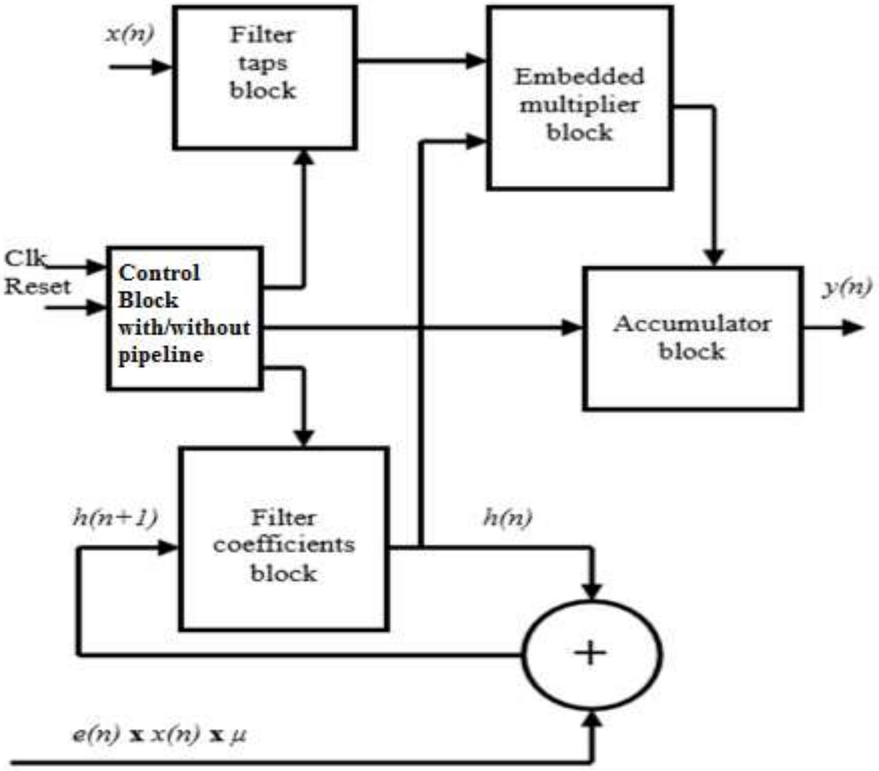

x(n)
Filter taps block
Embedded multiplier block
Clk
Reset
Control Block with/without pipeline
Accumulator block
y(n)
h(n+1)
Filter coefficients block
h(n)
+
e(n) x x(n) x µ

Fig 3.1 Block diagram Proposed of Logic

a) Filter taps block: provide the necessary delay (filter taps) for the input test signal, x (n) before being multiplied with filter coefficients, h (n).

b) Filter coefficients block: store and provide the updated filter coefficients at every completion of filter taps.

$$h(n+1) = h(n) + \mu . e(n) . x(n) \qquad (1)$$

c) Embedded multiplier block: multiply the filter taps output signal with the new updated filter coefficients.

d) Accumulator block: the products of the multiplication is then accumulate at this stage to produce the filtered signal, y(n). The accumulator is then cleared at every completion of filter taps and ready for the new input signal, x(n).

$$y(n) = \sum_{k=0}^{N-1} h_k x(n-k) . \qquad \textbf{(2)}$$

$$e(n) = d(n) - y(n) \qquad (3)$$

e) Control block: Synchronized the process of filter taps output signal with the new updated filter coefficients and provide the timing signal to clear and start the process in accumulator block.

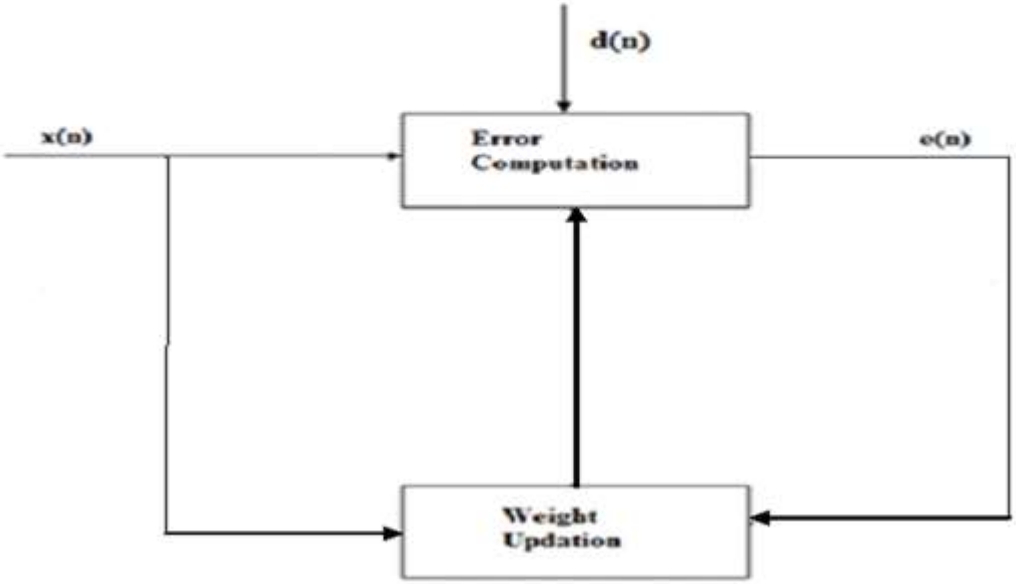


Fig 3.2 Block diagram of LMS Adaptive Filter

Figure shown below is the implemented block diagram. It results in improved throughput due to addition of three pipeline register which delay & stores intermediate values of updated weight, error signal respectively. The input signal is delayed by pipeline register 1. Input signal is provided to error computation block whose output is given to pipeline register 2. Output of the PR3 stage provided to weight updating unit whose output is given to PR3 stage. All the PR stages improve throughput. Below shows the 3 stage pipeline diagram.

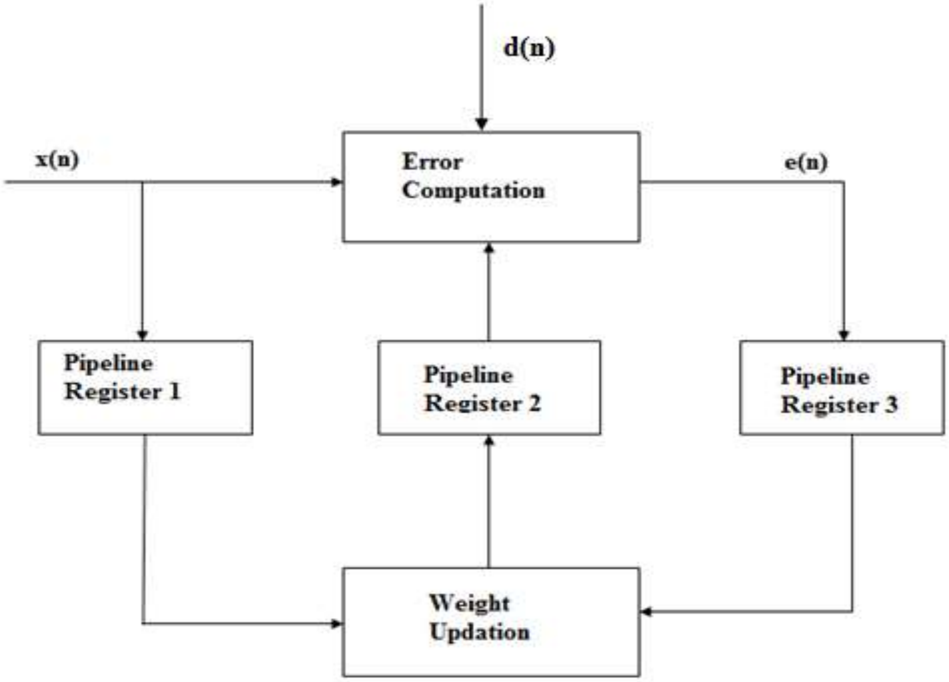


Fig 3.3 Block diagram of Pipeline LMS adaptive filter

IV. RESULTS AND DISCUSSION

Proposed logic has been developed using VHDL, simulation is done using Modelsim SE 6.3f software & synthesized using ALTERA Quartus II on Cyclone® IV EP4CE22F17C6N FPGA

4.1 Flow Chart It gives the flow of the project such as steps which is been followed to design the project. Below gives the flowchart for the project.

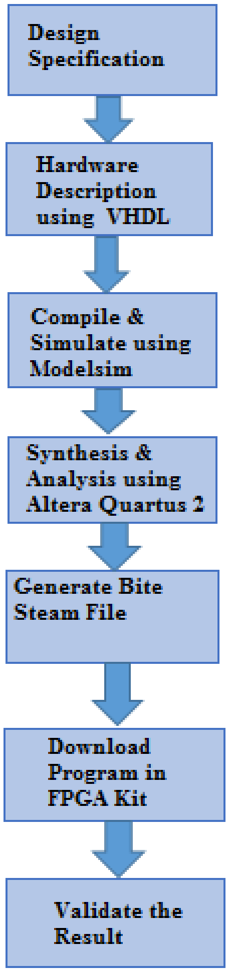
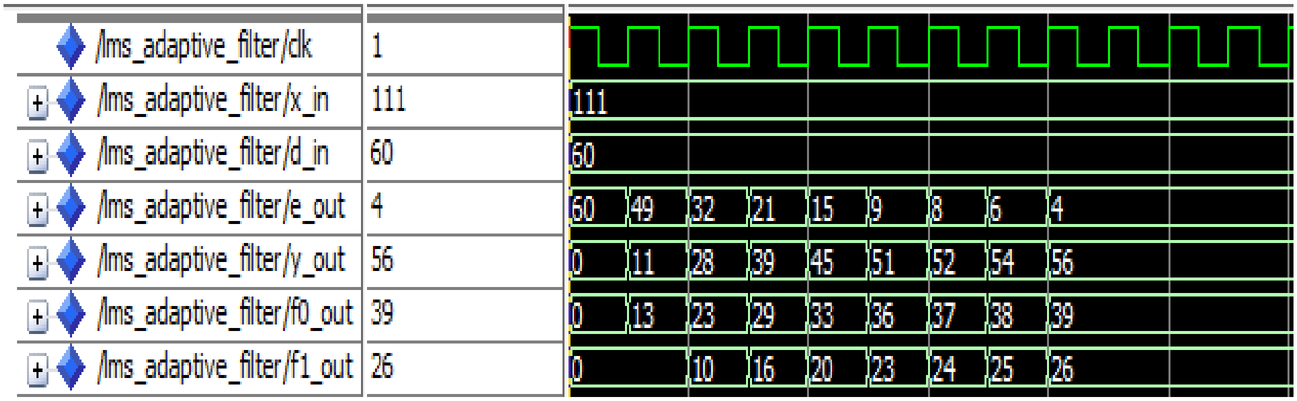

Design Specification
Hardware Description using VHDL
Compile & Simulate using Modelsim
Synthesis & Analysis using Altera Quartus 2
Generate Bite Steam File
Download Program in FPGA Kit
Validate the Result

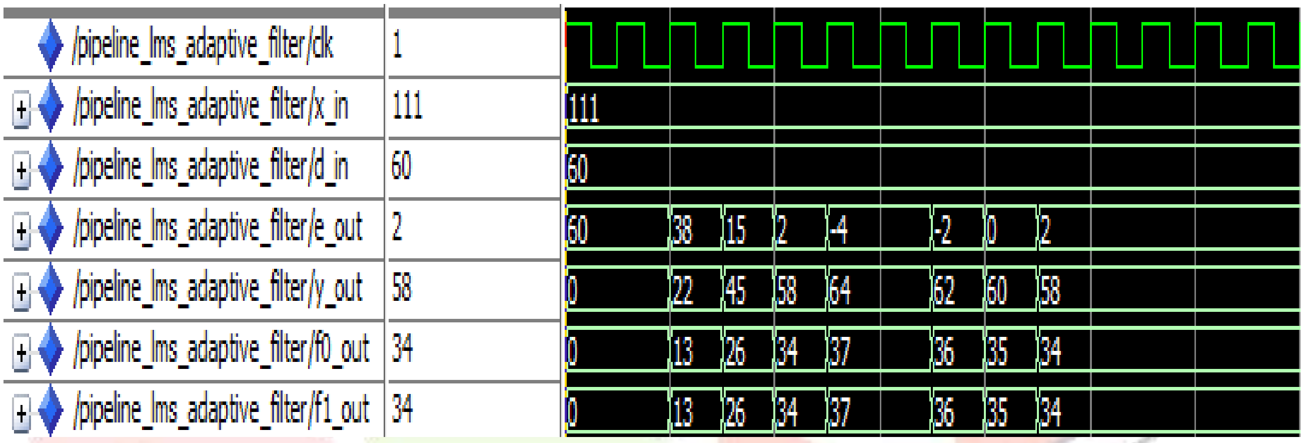
Fig 4.1 Flow chart

At every rising edge of clock x_in=111 & d_in=60 is applied at the input stage because of which y_out = 0 initially. At the same timef0_out & f1_out also be zero. According to LMS Algorithm error will be reduced & y_out will be closer to the desired output. Finally we will get error e_out=4 & y_out=56 for fig 4 & Similarly for Fig 5 because of the pipeline architecture for the same set of input error will be minimized & we will get e_out=2 and y_out=58 .Simulation results shows that the pipeline architecture is more efficient as compare to previous one.



```
/lms_adaptive_filter/clk      1
/lms_adaptive_filter/x_in     111   111
/lms_adaptive_filter/d_in     60    60
/lms_adaptive_filter/e_out    4     60  49  32  21  15  9   8   6   4
/lms_adaptive_filter/y_out    56    0   11  28  39  45  51  52  54  56
/lms_adaptive_filter/f0_out   39    0   13  23  29  33  36  37  38  39
/lms_adaptive_filter/f1_out   26    0       10  16  20  23  24  25  26
```

Fig 4.2: LMS Adaptive Filter



```
/pipeline_lms_adaptive_filter/clk      1
/pipeline_lms_adaptive_filter/x_in     111   111
/pipeline_lms_adaptive_filter/d_in     60    60
/pipeline_lms_adaptive_filter/e_out    2     60  38  15  2   -4  -2  0   2
/pipeline_lms_adaptive_filter/y_out    58    0   22  45  58  64  62  60  58
/pipeline_lms_adaptive_filter/f0_out   34    0   13  26  34  37  36  35  34
/pipeline_lms_adaptive_filter/f1_out   34    0   13  26  34  37  36  35  34
```

Fig 4.3: Pipeline LMS Adaptive Filter

The table shows comparison of LMS adaptive filter with & without pipeline. It compares different parameters such as Propagation delay, Maximum Frequency & throughput

Parameter/Method	**Without Pipeline**	**With Pipeline**
Propagation Delay	10.997 ns	10.637 ns
Maximum Frequency	90.993 MHz	94.011 MHz
Throughput	727.47 Mbps	752.09 Mbps

V. CONCLUSION & FUTURE SCOPE

A new high-performance implementation for DA based LMS adaptive filter has been presented. It is based on storing possible filter partial products in a look-up table (LUT) followed by a pipeline register (PR) unit. Throughput rate is significantly enhanced by LUT update and concurrent processing of filtering operation and weight-update operation. The simulation result shows that the proposed method gives better performance than the existing methods in terms of throughput. This improved performance is further accurately achieved at the cost of a slightly increased computational complexity.

We have tried to review the implementations of distributed arithmetic based adaptive digital FIR filter design using various LMS algorithms. Near to the desired value of output can be achieved by introducing more taps these will lead to increase in complexity, more area consumption & increased cost.

REFERENCES

[1] Mohd Tasleem Khan, Shaikh Rafi Ahamed "A New High Performance VLSI Architecture for LMS Adaptive Filter using Distributed Arithmetic" IEEE Computer Society Annual Symposium on VLSI, 2017

[2] Neil Woolfries, Patrick Lysaght, Stephen Marshall, Gordon McGregor and David Robinson "Fast Implementation of Non Linear Filters Using FPGA" Dept. of Electronic and Electrical Engineering, University of Strathclycle

[3] M.Devipriya, V.Saravanan, N.Santhiyakumari "power efficient and high throughput of fir filter using block least mean square algorithm in FPGA" IJRET, Volume: 03 ,Special Issue: 02 ,Mar-2014

[4] GAO Jinding, HOU Yubao SU Long "Design and FPGA Implementation of Linear FIR Low-pass Filter Based on Kaiser Window Function "978-0-7695-4353-6/11 © 2011 IEEE DOI 10.1109/ICICTA.2011

[5] Jyothirmayi Alahari, M.Valarmathi "optimized adaptive fir filter based on distributed arithmetic "IJSETR, Volume 3, Issue 4, April 2014

[6] G.Selvapriya, M.Mano, K.RekhaSwathiSri, Mr. S.Karthick "High Throughput, Low Area, Low Power Distributed Arithmetic Formulation for Adaptive Filter" IJIRCCE, Vol.2, Special Issue 1, March 2014

[7] Kalaiarashi.K, Mr.Santhakumar.K "Optimization of Adaptive Fir Filter for High Throughput, Low Power & Area Using Distributed Arithmetic" IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 4, Issue 1, Ver. II Jan. 2014

[8] Basant Kumar Mohantya, Pramod Kumar, Meher.b, Subodh Kumar Singala, M.N.S. Swamy "A high-performance VLSI architecture for reconfigurable FIR using distributed arithmetic", INTEGRATION, the VLSI journal (2016),

[9] Mr. Wasim Maroofi, Prof. Lalit Jain , Prof. Sanjay Ganar "Distributed Arithmetic Based FIR Adaptive Digital Filter Design Using LMS Algorithm", International Journal on Recent and Innovation Trends in Computing and ISSN: 2321-8169 Volume 3 ,Issue 2

[10] M. Backia Lakshmi, D. Sellathambi "Reconfigurable FPGA Implementation of FIR Filter using Modified DA Method "Volume 4 Issue 5, May 2015

[11] Miss. Rima P. Deshmukh, Prof. S. A. Koti, Prof. V. B. Baru "Review on Implementation of Fir Adaptive Filter Using Distributed Arithmetic and Block Lms Algorithm " , IJMER, Vol. 4, Issue. 6, June. 2014

[12] Naveen Shankar Naik, Dr. Kiran Gupta, IEEE "An Efficient Reconfigurable FIR Digital Filter Using Modified Distribute Arithmetic Technique" Volume 5, Issue 6, June 2015

[13] R. Mustafa1, M. A. Mohd Ali, C. Umat and D.A. Al-Asady "Design and Implementation of LMS Adaptive Filter on Altera Cyclone II FPGA for Active noise Control "

[14] Shu-Shin Chin, Wei Wu and Sangjiu Hong "Rapidly Reconfigurable Coarse-Grained FPGA Architecture for Digital Filtering Applications "

[15] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 52, no. 7, pp. 1327–1337, July 2005

[16]Adaptive filter From Wikipedia, the free encyclopaedia

[17]Least mean squares filter From Wikipedia, the free encyclopedia

[18] S. Haykin and B. Widrow, Least-mean-square adaptive filters. John Wiley & Sons, 2003, vol. 31.