High Performance of LMS Adaptive Filter Using LUT

VutlaVujjaini Devi[1], V.Pushpa[2], D.Lakshmi[3]
Research Scholar
Sakthi Mariamman Engineering College, TN, India

Abstract: The presents an efficient implementation of adaptive filter by minimizing the area and the power consumed with the use of least mean square algorithm. The memory based structures are replaced with the MAC units in order to achieve the optimized area and power. The LUTs are memory based units used for the design of the filter.

Keywords- MAC, LUT, memory, adaptive filter.

INTRODUCTION

The term 'filter' is often applied to any device or system that processes incoming signals or other data in such a way as to eliminate noise, or predict the next input signal from moment to moment[1] . During the last decades the adaptive filters have attracted many researches due to the property of their self-designing. Hence, an adaptive filter is a computational device that attempts to model the relationship between two signals in an iterative manner. Adaptive filters are often realized either with a set of logic operations implemented on a field programmable gate array (FPGA) or with a set of programming instructions running on a microprocessor or on a digital signal processor.

The adaptive filter is defined by 4 main aspects:
1. The signal processed by a filter.
2. The structure defines how the output of the filter is computed from its input signal.
3. The parameters within this structure that can be iteratively changed to alter the filter's input-output relationship.
4. The algorithm that describes how the parameters are adjusted from one time instant to next.

The adaptive filtering consists of 2 basic operations: the filtering process and adaptation process.

In filtering process, an output signal is generated from an input data signal using a digital filter. In the adaptation process, consists of an adaptation algorithm which adjusts the coefficients of the filter to minimize the desired function.

ICGPC 2014
St.Peter's University, TN, India.

AN ADAPTIVE FILTER STRUCTURE

The adaptive filter structure generally consists of tapped delay line, variable weights, whose inputs signals are the signals at the delay line taps, a summer to add weighted signals and machinery to adjust the weights automatically. The delay components allow the access to the past and future values in sequence. The adaptation process automatically seeks an optimal filter impulse response by adjusting weights.

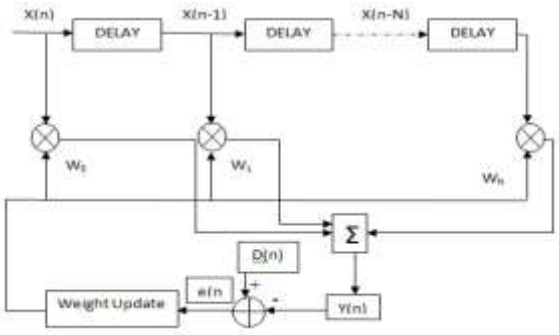

Figure 1: Adaptive Filter Block

In the above figure X(n) is the input sequence and set of W are the weights. The desired signal is given as D(n) and the filter output is the Y(n).The difference between the filter output and desired signal is given as the error out e(n), which updates the weight for optimization. The general form is given as shown below:

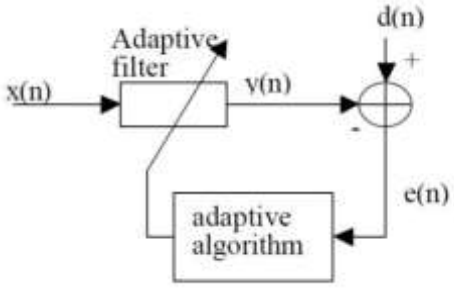

Figure 2: General Adaptive Filter Block

The adaptive filter has two distinct parts: one is the filter structure which is designed to perform the desired output, y(n) processing and second is the adaptive algorithm for altering the filter co-efficient in order to improve the performance. The input signal x(n) is weighted in a digital filter to produce output y(n). The adaptive algorithm is used to minimize the difference between the output [y(n)] and desired response [d(n)] that is, the error signal [e(n)].

REVIEW OF LMS ALGORITHM

The adaptive filter algorithm followed is the LMS algorithm. The LMS algorithm is defined as the least mean squares (LMS) algorithms adjust the filter coefficients to minimize the cost function. The adaptive algorithm operates with the arbitrary initial conditions: each time new samples are received for the input signal and the desired response, appropriate corrections are made to the previous of the filter co-efficient. The adaptation takes place until error is minimized completely so that efficiency is achieved to the maximum. Considering the figure 2, the signals are given as: Let X(k) be the input vector of the time delayed input sequence x(n) and x(k) is the input at time k. It is given as:

$X(k)=\{x(k), x(k-1), x(k-2),\ldots..x(k-N)\}^T$

Similarly W(k) be the weights applied to the filter coefficients at time k, is given as:

$W(k)=\{w(k), w(k-1), w(k-2),\ldots..w(k-N)\}^T$

The LMS algorithm performs the following operations to update the filter coefficients:

1. Calculate the output signal y(k) of the filter. The output of the filter y(k) represents an estimate of the desired response. y(k) is calculated as the convolution of the weight of the vector and input vector.

$$y(k) = \sum_{n=0}^{N-1} W_n(k)x(k-n) = W^T(k)x(k)$$

2. The error signal e(k) is the estimated error, is defined as the difference between estimated response and desired response.

$$e(k)= d(k)-y(k)$$

The error signal and the input signal are applied to the weight update algorithm to updates the filter coefficients. The LMS algorithm updates its coefficients through the minimization of the mean of the instantaneous squared error denoted by $[E(e^2(k))]$. The LMS algorithm assume that x(k) and d(k) are wide-sense stationary erotic processes, therefore, their means and variances are constant. X(k) and W(k) are assumed to be independent. The LMS iterative weight update algorithm follows the following equation:

$$W(k+1)=W(k)+2\mu e(k)X(k)$$

The step size parameter μ is a small positive constant. The selection of the value of the step size influences the filter co-efficient of the filter.

LITERATURE SURVEY

Sang Yoon Park et al. [2013][2] The LMS algorithm is a practical scheme for realizing Wiener filters. 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 modified to a form called the delayed LMS (DLMS) algorithm.

The fixed-point implementation issues of LMS algorithm are taken into consideration introduce the number of pipeline delays, the area (number of logical elements), sampling period, and energy consumption. The proposed structure consists of *N*numberof2-bitpartial product generators (PPG) corresponding to *N* multipliers and a cluster of *L*/2 binary adder trees, followed by a single shift–addoperation.MAC block is used for the multiplication operation. The area achieved is 26.660m and power dissipation is 70.3. The software used for simulation is Xilinx.

Pramod Kumar Meher et al.,[2010]proposed Distributed arithmetic (DA)based computation is popular for its efficient and optimal memory-based implementation of finite impulse response(FIR)filter where the filter outputs are computed as inner – product of input-sample vector sand filter-coefficient vector. In this paper, however, we show that the look-up-table (LUT)-multiplier-based approach, where the memory elementsstoreallthepossiblevaluesofproductsofthefiltercoefficients could be an area-efficient alternative to DA-based design of FIR filter with the same through put of implementation. By operand and inner –product decompositions, respectively, we have designed the conventional LUT-multiplier-based and DA-based structures for FIR filter of equivalent throughput, where the LUT-multiplier-based design involves nearly the same memory and the same number of adders, and less number of input register at the cost of slightly higher adder –widths than the other.

Megha Maheshwari et al.,[2011] has presented a modified delayed least means square (DLMS) adaptive algorithm to achieve lower adaptation-delay and proposed an efficient pipelined architecture for the implementation of this adaptive filter. It has been shown that the proposed DLMS adaptive filter can be implemented by a pipelined inner-product computation unit for calculation of feedback error, and a pipelined weight-update unit consisting of N parallel multiply accumulators, for filter order N. From the synthesis results, the existing direct-form structure involves nearly 50% more area-delay product (ADP) and nearly 74% more energy per sample (EPS) than the proposed one, in average, for filter orders N = 8, 16 and 32.

PROPOSEDSYSTEM

In general, digital filters are divided into two categories, including Finite Impulse Response (FIR) and Infinite Impulse Response(IIR).Adaptive filter is a type of FIR filters that are widely applied to a variety of digital signal processing areas for the virtues of providing linear phase and system stability.

The FPGA-based adaptive filter use direct arithmetic blocks that are said as multiply-and-accumulate (MAC) blocks with the augment of the filter order. However, according to Distributed Arithmetic, we can make a Look-Up-Table (LUT) to conserve the MAC values and callout the values according to the input data if necessary. Therefore, LUT can be created to take the place of MAC units so as to save the hardware resources.

This paper provide the principles of Distributed Arithmetic, and introduce it into the filter design, that use Distributed Arithmetic, which saves considerable MAC blocks to decrease the circuit scale, meanwhile, divided LUT method is used to decrease the required memory units and pipeline structure is also used to increase the system speed.

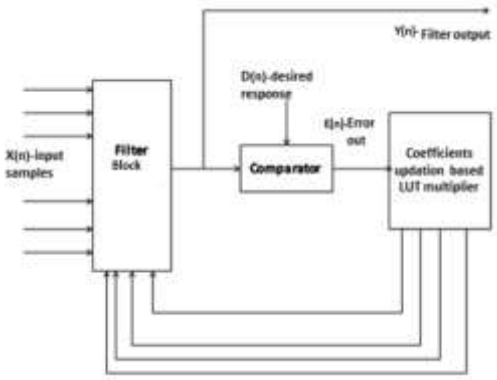

Figure 3: Proposed Adaptive Filter

A fixed coefficient filter can be implemented using DA efficiently regardless of the filter length. However, problem arises when implementing an adaptive filter whose coefficients are to be adapted which requires that the contents of the LUT are to be updated. In-case of a Least Mean Square (LMS) adaptive filter, the weights are updated to minimize the mean squared error e according to the update equation.

$$wi[n+1]=wi[n]+\mu x[n-i]e[n]$$

$d[n]$ being the desired signal and μ being the step size.

EXPERIMENTAL ANALYSIS

The simplest technique is known as window technique, which is based on designing a filter using well known frequency domain transition functions called "windows". The experimental analysis includes two divisions one is the filter co-efficient generation and other is the partial product method, which is the adaption process.

FILTER COEFFICIENT GENERATION

The parameter □ is an important coefficient of Kaiser Window which involves the windows types. We can get a variety of windows like Rectangular window, Hanning window, Hamming window, and Blackman window with the adjustment. A 31-order FIR low-pass filter is designed using Kaiser Window, and the parameter is as follows: □=3.39, w=0.18. We can obtain the filter coefficients using Matlab and few are as follows:

h(0)=h(31)=0.0019;h(1)=h(30)=0.0043;
h(2)=h(29)=0.0062;h(3)=h(28)=0.0061;
h(4)=h(27)=0.0025;h(5)=h(26)=-0.0050;
h(6)=h(25)=-0.0148; the output obtained is shown below:

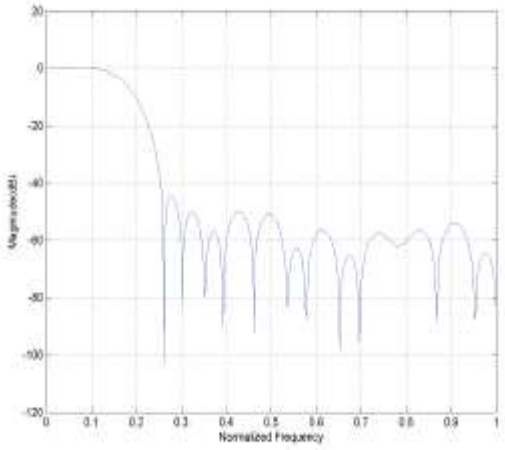

Figure 4: Frequency–Amplitude of filter.

REVIEW OF PARTIAL PRODUCT METHOD

The parallel computing is adopted to improve the speed of calculation. The complicated multiplication-accumulation operation is converted to the shifting and adding operation when the DA algorithm is directly applied to realize linear time invariant system. However, the scale of the LUT will increase exponentially with the coefficient. If the coefficient is small, it is very convenient to realize through the rich structure of FPGA LUT; while the coefficient is large, it will take up a lot of storage resources of FPGA and reduce the calculation speed.

Meanwhile, the N-1 cycles also result in the too long LUT time and the low computing speed. The improvement and optimization of the DA algorithm aiming at the problems of the configuration in the coefficient of adaptive filter, the storage resource and the calculating speed, which make the memory size smaller and the operation speed faster to improve the computational performance.

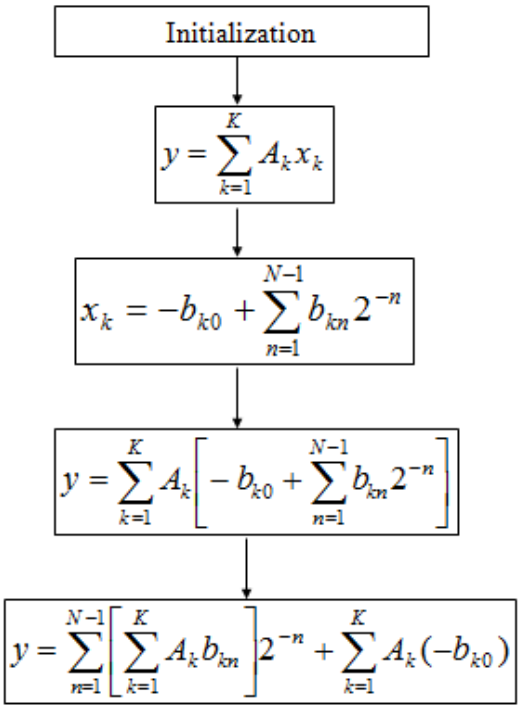

Figure 5. Flow Chart Of Distributed Airthematic (DA)

RESULTS

The Adaptive filter is simulated in HDL language using Xilinx and in order to analyze the power, area and frequency consumed Quartus 2 is made use. On executing the below shown RTL is achieved for the adaptive filter.

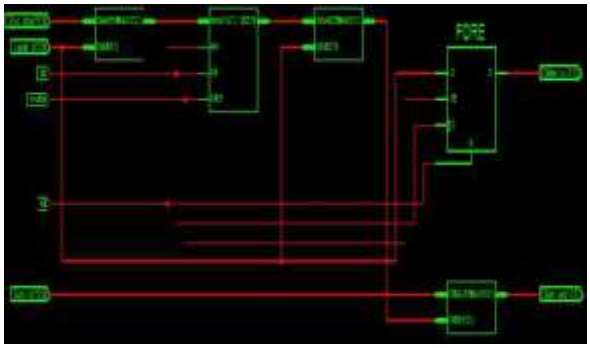

Figure 6: RTL Schematic of Adaptive Filter.

The simulation waveform for the above RTL schematic is obtained as shown below:

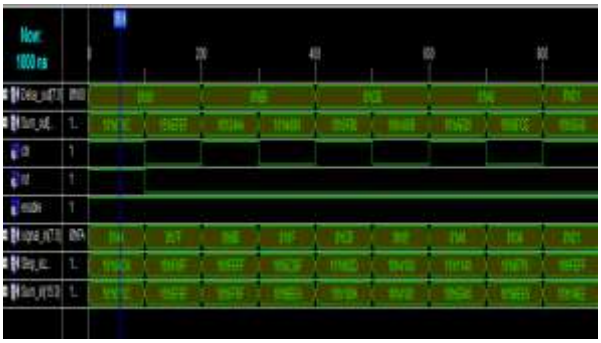

Figure 7: Simulation Waveform.

The above results are simulated using Xilinx and in order to obtain the analysis report we simulate using the quarters II software. So, the following results have obtained as shown below:

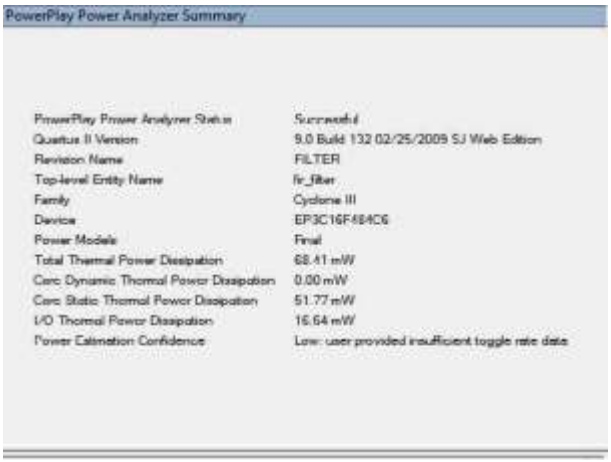

Figure 8: Power Analysis.

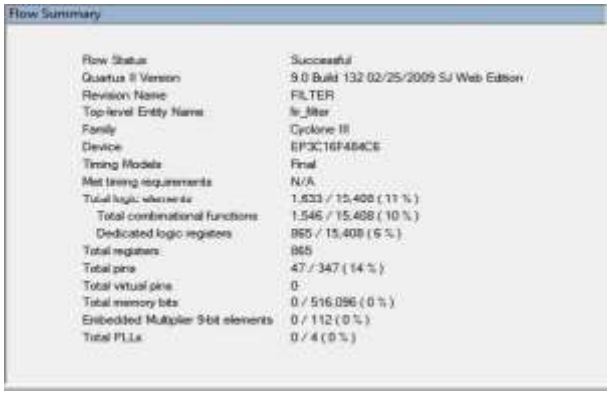

Figure 8: Area Analysis.

APPLICATION

The Adaptive filter is used in various fields and is mainly used for system identification, echo cancellation, noise cancellation etc.. The block shown below is for the noise cancellation using adaptive filter.

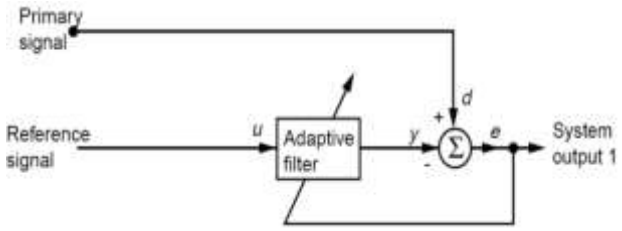

Figure 9: Noise Cancellation Block.

On executing with the adaptive filter in MATLAB, the below result is obtained.

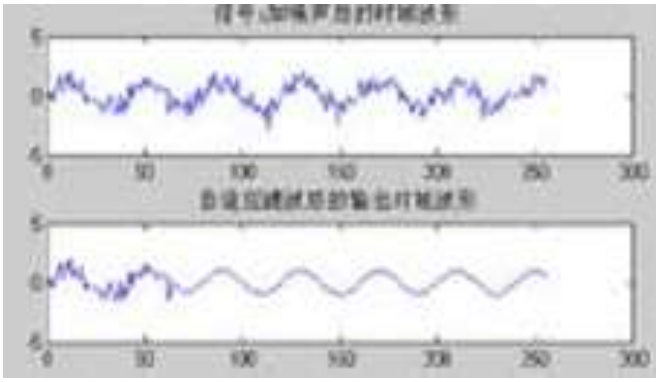

Figure 10: Noise Cancellation

CONCLUSION

An efficient adaptive filter has being designed and simulated using Xilinx so comparing with the literature survey [2], the area and the power has been minimized

.REFERENCES

[1] Introduction to adaptive filter by Douglas B.Williams, 1999.

[2] On Adaptive Least Mean Square FIR filter: New implementations and applications, Tampere, 2004.

[3] Adaptive Filters by Bernard Widrow, 1971.

[4] Adaptive filter architecture for FPGA architecture by Joseph G.Petrone, 2004.

[5] S. Haykin and B. Widrow, 2003, Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley.

[6] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, Jul. 2005, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 7, pp. 1327–1337.

[7] Y. Voronenko and M. Puschel, May 2007l, "Multiplier less multiple constant multiplication," ACM Trans. Algorithms, vol. 3, no. 2, pp. 1–38.

[8] P. K. Meher, S. Candrasekaran, and A. Amira, Jul. 2008, "FPGA realization of FIR

filters by efficient and flexible systolization using distributed arithmetic," IEEE Trans. Signal Process., vol. 56, no. 7, pp. 3009–3017.

[9] P. K. Meher, Jul. 2008, "New approach to look-up-table design and memory-based realization of FIR digital filter," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 3, pp. 592–603.

[10] P. K. Meher, April 2010, 'LUT Optimization for Memory-Based Computation,' *IEEE Trans on Circuits & Systems-II*, pp.285-289.

[11] P. K. Meher, September 2010, 'Novel Input Coding Technique for High-Precision LUT Based Multiplication for DSP Applications' *The18th IEEE/IFIP International Conference on VLSI and System-on-Chip* (VLSI-SoC 2010), pp. 201-206, Madrid, Spain.

[12] P. K. Meher, September 2010, 'An Optimized Lookup-Table for the Evaluation of Sigmoid Function for Artificial Neural Networks' *The18th IEEE/IFIP International Conference on VLSI and System-on-Chip* (VLSI-SoC 2010), pp. 91-95, Madrid, Spain,