International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 11, November-2014

Implementation of LMS Adaptive Filter using High Speed Vedic Multiplier

Akanksha Pawar
ME research Scholar
SSGI(FET).
Bhilai, India

Anil Kumar Sahu
Assistant Professor
SSGI(FET).
Bhilai, India

Dr. G. R. Sinha
Professor
(Electronics and Telecommunication Engg.)
& Associate Director
SSGI(FET),SSTC.
Bhilai, India

Abstract: **The main purpose of every filter is to acquire useful information from a signal with noise. A fixed normal filter is designed knowing the statistics of the two signals: the information one and the noise one. An adaptive filter auto sets continuously to the environment changes through the using of recursive algorithms, and is used when statistics and temporal changes of the signal are preliminarily unknown.**

An adaptive discrete filter accepts en enter *x*(n) and produces an output y(n) due a convolution with the filter weight *w*(k). A reference wished signal *d*(n), is compared at the output to obtain an error estimate *e*(n). This error signal is used to increasingly adjust the filter weight for the next interval of time. There are some algorithms used to adjust the weight among them are the LMS (Least Mean Square) and the RLS (Recursive Least Square).

This Project introduces the LMS (least mean squares) algorithm for the design of adaptive traversal filters with the use of a newly proposed high speed vedic multiplier.The LMS adaptive filter is coded in verilog HDL ,the design is synthesized using Xilinx XST and simulation is done using the tool Modelsim 6.5. The application side of this project includes identification of an unknown system in many communication and digital signal processing applications like Channel Equalization, Adaptive Noise Cancellation, Adaptive Echo cancellation.

***Key words: FIR filters ; Transversal filter; design of Vedic multiplier*.**

I. INTRODUCTION

Discrete-time (or digital) filters are ubiquitous in today's signal processing applications. Filters are used to achieve desired spectral characteristics of a signal, to reject unwanted signals, like noise or interferers, to reduce the bit rate in signal transmission, etc. The notion of making filters adaptive, i.e., to alter parameters (coefficients) of a filter according to some algorithm, tackles the problems that we might not in advance know, e.g., the characteristics of the signal, or of the unwanted signal, or of a systems influence on the signal that we like to compensate. Adaptive filters can adjust to unknown environment, and even track signal or system characteristics varying over time.

The earliest work on adaptive filters may be traced back to the late 1950s, during which time a number of researchers were working independently on theories and applications of such filters. From this early work, the least-mean-square LMS algorithm emerged as a simple, yet effective, algorithm for the design of adaptive transversal (tapped-delay-line) filters.

The LMS algorithm was devised by Widrow and Hoff in 1959 in their study of a pattern-recognition machine known as the adaptive linear element, commonly referred to as the Adaline [1, 2]. The LMS algorithm is a stochastic gradient algorithm in that it iterates each tap weight of the transversal filter in the direction of the instantaneous gradient of the squared error signal with respect to the tap weight in question.

Let w^(n) denote the tap-weight vector of the LMS filter, computed at iteration (time step) n. The adaptive operation of the filter is completely described by the recursive equation (assuming complex data)

$$w^{(n+1)} = w^{(n)} + x(n)*e(n)*m$$

where x(n) is the tap-input vector, d(n) is the desired response, e(n)=d(n)-x(n) and m is the step-size parameter. The quantity enclosed in square brackets is the error signal. The asterisk denotes complex conjugation, and the superscript H denotes Hermitian transposition (i.e., ordinary transposition combined with complex conjugation).

II. VERILOG HDL

Verilog is used to model the design in this project. It is a hardware description language (HDL) used to model electronic systems. It is sometimes called Verilog HDL, which supports the design, verification, and implementation of analog, digital, and mixed-signal circuits at various levels of abstraction. Verilog was originally developed and owned by Gate Way design in 1984. After this, Cadence Design Systems purchased Gate Way and continued selling Verilog-XL as a Verilog-HDL simulator with PLI support in 1990. In 1995 Cadence released the specs for Verilog-HDL and they were accepted as IEEE -1364 standard which included the PLI1.0 (TF/ACC) routines as a standard for all Verilog Simulators. In 1993 PLI2.0 (VPI) routines were released as a standard by OVI and in 1999 IEEE will vote on updating the 1364 standard to include PLI2.0. In 2001 IEEE accepted the updated Verilog standard commonly known as Verilog

2001 and today, there are a dozen simulators that simulate Verilog HDL. 29 Verilog was generated as a language for the industry rather than academia. It is very C like programming style that closely represents hardware. VHDL supports 9 values logic, where as Verilog supports 7 strengths on 3 values. Compared to VHDL, VHDL offers more programming constructs where as Verilog is closer to hardware. Basically, there are various reasons for converting the IOP VHDL design to Verilog. Verilog HDL is a general purpose hardware description language that is easy to learn and easy to use. It is similar to C programming language. Designers with C programming experience will find it easy to learn Verilog HDL, and will be comfortable with its syntax. Besides, Verilog allows different levels of abstraction to be mixed in the same models. Thus, a designer can define a hardware model in terms of switches, gates, RTL, or behavioral code. Also, a designer needs to learn only one language for stimulus and hierarchical design. On top of that, most popular logic synthesis tools support Verilog HDL.

III. FILTER DESCRIPTION

Digital filters can be divided into two categories: finite impulse response (FIR) filters and infinite impulse response (IIR) filters. Although FIR filters, in general, require higher taps than IIR filters to obtain similar frequency characteristics, FIR filters are widely used because they have linear phase characteristics, guarantee stability and are easy to implement with multipliers, adders and delay elements [1,2]. The number of taps in digital filters varies according to applications. In commercial filter chips with the fixed number of taps [3], zero coefficients are loaded to registers for unused taps and unnecessary calculations have to be performed. To alleviate this problem, the FIR filter chips providing variable-length taps have been widely used in many application fields. However, these FIR filter chips use memory, an address generation unit, and a modulo unit to access memory in a circular manner. The paper proposes two special features called a data reuse structure and a recurrent-coefficient scheme to provide variable-length taps efficiently. Since the proposed architecture only requires several MUXs, registers, and a feedback-loop, the number of gates can be reduced over 20 % than existing chips.

In, general, FIR filtering is described by a simple convolution operation as expressed in
the equation (1)

$$y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n-k] \qquad (1)$$

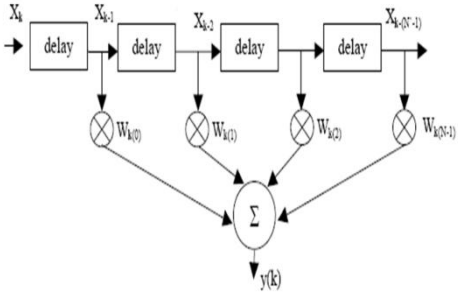

Figure1: Block diagram of transverse filter

IV. THE LMS ADAPTATION ALGORITHM

The LMS algorithm was created by Widrow and Hoff in 1960 to be used in the training of neural network. It uses a rough gradient approximation, and seeks the wished weight vector.

This process is used to find the weight vectors for training the ALC (Adaline). The learning rules can be incorporate to the same device that therefore can be auto adapted as there are presented the wished inputs and outputs. The weight vectors values are changed as every combination input-output is processed. This goes on until the ALC gives the correct outputs. This is a truly training process since there is not necessary to clearly calculate the weight vector value.

Where μ is a convergence factor, that defines how big or small is the signal passing thru the filter; its value will be very important for the convergence speed and for the LMS algorithm stability. It takes values between 0 and $2/MS_{máx}$, where M is the number of filter steps and $S_{máx}$ is the highest value of power spectral density of the input *u* step.

The LMS (least mean squares) algorithm is an approximation of the steepest descent algorithm which uses an instantaneous estimate of the gradient vector of a cost function. The estimate of the gradient is based on sample values of the tap-input vector and an error signal. The algorithm iterates over each coefficient in the filter, moving it in the direction of the approximated gradient [1]. For the LMS algorithm it is necessary to have a reference signal d[n] representing the desired filter output. The difference between the reference signal and the actual output of the transversal filter is the error signal

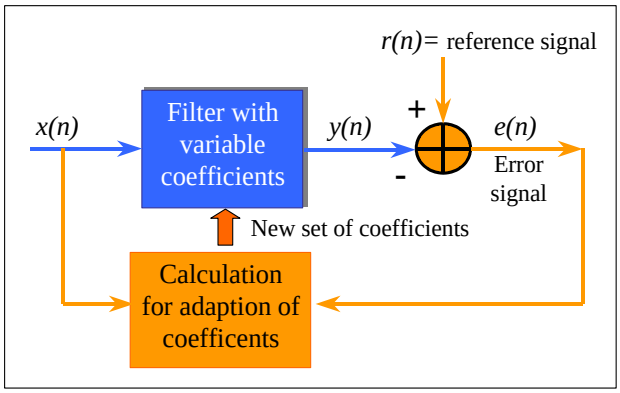

Figure 2: Block diagram of LMS filter

LMS Algorithm Steps

- **Filter output** $y[n] = \sum_{k=0}^{M-1} u[n-k]w_k^*[n]$
- **Estimation error** $e[n] = d[n] - y[n]$
- **Tap-weight adaptation** $w_k[n+1] = w_k[n] + \mu u[n-k]e^*[n]$

$$\begin{pmatrix}\text{update value}\\ \text{of tap - weigth}\\ \text{vector}\end{pmatrix} = \begin{pmatrix}\text{old value}\\ \text{of tap - weight}\\ \text{vector}\end{pmatrix} + \begin{pmatrix}\text{learning-}\\ \text{rate}\\ \text{parameter}\end{pmatrix}\begin{pmatrix}\text{tap}-\\ \text{input}\\ \text{vector}\end{pmatrix}\begin{pmatrix}\text{error}\\ \text{signal}\end{pmatrix}$$

V. SIMULATION RESULT

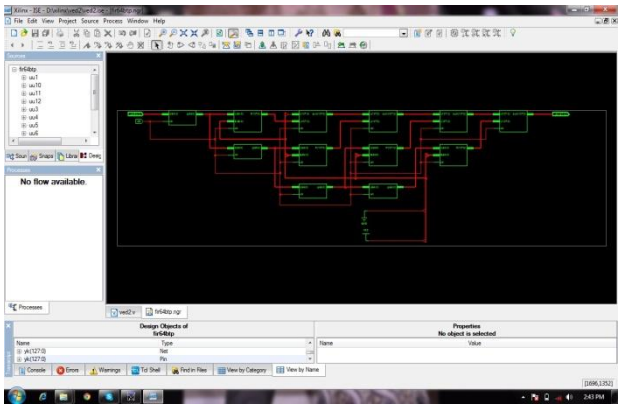

Figure 3: Schemetic of LMS filter

A. HDL Synthesis Report

Macro Statistics

- # Adders/Subtractors : 1
- 128-bit adder : 1
- # Registers : 8199
- 128-bit register : 1
- 4-bit register : 8192
- 64-bit register : 6
- # Xors : 24582
- 1-bit xor2 : 16384
- 128-bit xor2 : 12
- 16-bit xor2 : 768
- 32-bit xor2 : 192
- 4-bit xor2 : 4096
- 64-bit xor2 : 58
- 8-bit xor2 : 3072

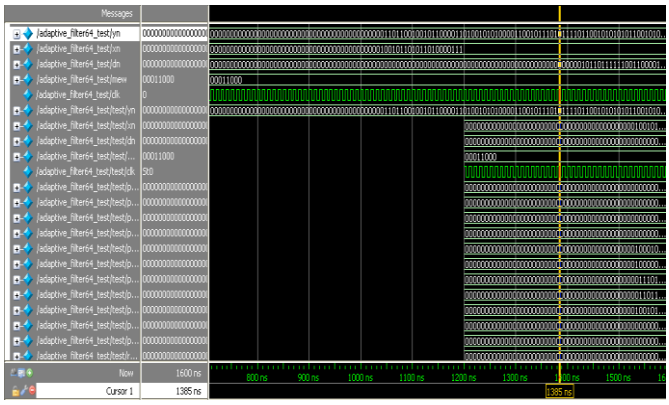

Figure 4: simulation of LMS filter

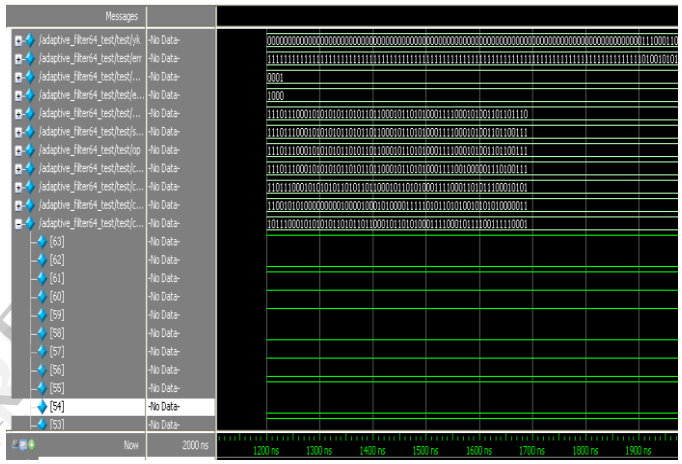

Figure 5: simulation of FIR filter

VI. CONCLUSION

This paper presents an Adaptive FIR filter system using Least-Mean-Square Algorithm which has flexibility, power-efficiency and configuration time advantages. This project includes a newly proposed Vedic multiplier concept published from IETE journals which is highly efficient in terms of speed and due to its regular and parallel structure it can be realized easily on silicon as well leads to a reduction in hardware cost. The Adaptive Least Mean Square FIR filter is coded using Verilog HDL and further it is Synthesized using the tool Xilinx7.1i and simulation is done using Modelsim 6.3f.The presented Adaptive Least Mean Square FIR filter system is responsible for providing the best solution for realization and autonomous adaptation of FIR filters, and can be used in many communication and Digital signal processing applications like Channel Equalization, Adaptive noise cancellation, Adaptive echo cancellation, System identification and many others.

REFERENCES

1. Haykin, Simon, Adaptive Filter Theory, Prentice Hall, Upper Saddle River, New Jersey, 1996.
2. Bernard Widrow and Samuel D. Stearns: "Adaptive Signal Processing", Prentice-Hall, Inc., UpperSaddle River, NJ, 1985.
3. Honig, Michael L., Messerschmitt, David G., Adaptive Filters, Structures, Algorithms and Applications, Kluwer Academic Publishers, Boston, 1984.
4. Jenkins, W. Kenneth, Hull, Andrew W.,Strait, Jeffrey C., Schnaufer, Bernard A., Li, Xiaohui, Advanced Concepts in Adaptive Signal Processing, Kluwer Academic Publishers, Boston, 1996.
5. Purushottam D. Chidgupkar and Mangesh T. Karad, "The Implementation of Vedic Algorithms in Digital Signal Processing", Global J. of Engng. Educ., Vol.8, No.2 © 2004 UICEE Published in Australia.
6. Himanshu Thapliyal and Hamid R. Arabnia, "A Time-Area- Power Efficient Multiplier and Square Architecture Based On Ancient Indian Vedic Mathematics", Department of Computer Science, The University of Georgia, 415 Graduate Studies Research Center Athens, Georgia 30602-7404, U.S.A.
7. E. Abu-Shama, M. B. Maaz, M. A. Bayoumi, "A Fast and Low Power Multiplier Architecture", The Center for Advanced Computer Studies, The University of Southwestern Louisiana Lafayette, LA 70504.