Fixed Point Lms Adaptive Filter Using Partial Product Generator

Vidyamol S
M.Tech Vlsi And Embedded System
Ma College Of Engineering,
Kothamangalam,India
vidyas.saji@gmail.com

Abstract— The area and power efficiency is very important for every circuit and its applications. Here I present an efficient architecture of adaptive filter with partial product generator and delayed lms algorithm is also used. By using improved adder structure the area power efficiency can be increased. An adaptive algorithm is a procedure for adjusting the parameters of an adaptive filter to minimize a cost function chosen for the task at hand. We use Xilinx 14.5 to provide VHDL coding for our architecture. Two main part in the adaptive filter system are error computation block and weight update block. The error computation block will calculate the new error values, function weight update block is calculate new weight from the error value computed from the error update block.

Keywords— Partial Product Generator, Weight update block, Modified DLMS adaptive filter

I. INTRODUCTION

The adaptive filter is a self modifying filter, it will adjust the coefficient thus it will reduce the error value. The distance between desired signal and reference signal is considered as the output of the adaptive filter. There are wide variety of adaptive algorithms are used, the following aspects are used
a)filter structure b) Rate of convergence, mis adjustment, and tracking c) Computational aspects. LEAST MEAN SQUARE adaptive algorithm is most commonly used algorithm. The least mean square adaptive filter have some difference from common digital filter, that is common digital filter have only one input but an adaptive filter have two input desired signal and input signal and only one output signal. the basic configuration of adaptive filter is shown in figure 2.1operate in the discrete time domain n,input signal is x(n) output signal is y(n) reference signal is d(n) and updated weight is w(n),the error signal is the difference between d(n) and y(n).

\[ e(n) = y(n) - d(n) \]

The error signal is used by the adaptation algorithm to update the adaptive filter coefficient vector w(k) according to some performance criterion. In general, the whole adaptation process aims at minimizing some metric of the error signal, forcing the adaptive filter output signal to approximate the reference signal in a statistical sense. It is interesting to notice how this basic configuration fits perfectly in several practical applications such as system identification, interference canceling, channel equalization, and signal prediction. In communication systems, useful information is transmitted from one point to another across a medium such as an

Electrical wire, an optical fiber, or a wireless radio link. Non idealities of the transmission medium or channel distort the fidelity of the transmitted signals, making the deciphering of the received information difficult. In cases where the effects of the distortion can be modeled as a linear filter, the resulting “smearing” of the transmitted symbols is known as inter symbol interference (ISI). In such cases, an adaptive filter can be used to model the effects of the channel ISI for purposes of Deciphering the received information in an optimal manner. In this problem scenario, the transmitter sends to the receiver a sample sequence x(n) that is known to both the transmitter and receiver. The receiver then attempts to model the received signal d(n) using an adaptive filter whose input is the known transmitted sequence x(n).After a suitable period of adaptation, the parameters of the adaptive filter in w(n) are fixed and then used in a procedure to decode future signals transmitted across the channel.

II. RELATED WORKS

The LMS algorithm is very popular and has been widely used due to its extreme simplicity. Its convergence speed, however, is highly dependent on the condition number of the input-signal autocorrelation matrix defined as the ratio between the maximum and minimum Eigen values
of this matrix, it is used to finding the filter coefficient with minimum error.[1] One approach that has been successfully employed in situations where signal statistics are unknown is the on-line calculation of the convergence factor which takes part in updating the filter coefficients, with respect to the tap weights [2]. The LMS algorithm is but one of an entire family of algorithms that are based on instantaneous approximations to steepest descent procedures. Such algorithms are known as stochastic gradient algorithms because they use a stochastic version of the gradient of a particular cost function’s error surface to adjust the parameters of the filter [3]. An efficient architecture with dlms algorithm is developed [4]. The dlms adaptive algorithm have lower adaption delay compared to the lams algorithm and it contains large pipelined architecture [5]. Typical DSP Programs with highly real-time, design hardware and or software to meet the application speed constraint. It also deals with 3-Dimensional Optimization (Area, Speed, and Power) to achieve required speed, area-power tradeoffs and power consumption [6]. An efficient scheme is presented for implementing the LMS-based transversal adaptive filter in block floating-point (BFP) format, which permits processing of data over a wide dynamic range, at temporal and hardware complexities significantly less than that of a floating-point processor [7].

III. DELAYED LMS ADAPTIVE FILTER

Consider lms algorithm it will find the output signal from the input sample. we can propose the simplified cost function given by

\[ J_{\text{LMS}} = \frac{1}{2} e^2(n) \]  

This cost function can be thought of as an instantaneous estimate of the MSE cost function, as

\[ J_{\text{MSE}} = \mathbb{E}\{J_{\text{LMS}}\} \]  

Although it might not appear to be useful, the resulting algorithm obtained when \( J_{\text{LMS}} \) is used for \( J_{\text{MSE}} \) is extremely useful for practical applications. Taking derivatives of \( J_{\text{LMS}}(n) \) with respect to the elements of \( W(n) \), we obtain the LMS adaptive algorithm given by

\[ W_{n+1} = W_n + \mu \cdot e(n) \cdot x(n) \]  

Here,

- \( x(n) \) is the input vector
- \( w(n) \) is the weight vector of an Nth order LMS adaptive filter at the nth iteration.
- \( d(n) \) is the desired response and \( y(n) \) is the filter output of the nth iteration.
- \( e(n) \) denotes the error computed in the nth iteration which is used to update the weights.
- \( \mu \) is the convergence-factor.

Note that this algorithm is of the general form. It also requires only multiplications and additions to implement. In fact, the number and type of operations needed for the LMS algorithm is nearly the same as that of the FIR filter structure with fixed coefficient values, which is one of the reasons for the algorithm’s popularity. Some modification is made on lms algorithm we get delayed lms algorithm. The equation of delayed lms algorithm is shown in below

\[ W_{n+1} = W_n + e(n - m) \cdot x(n - m) \]  

where, \( m \) is the adaptation-delay.

![Fig 2: Structure of delayed LMS adaptive filter](image)

The fig2 show the architecture of dlms adaptive filter. It is clear from the figure that the output is obtained only after m cycle of delay.

IV. PROPOSED SYSTEM

In the proposed architecture a new delay is introduced. The total delay is delay due to error computation block and weight update block. The error computation block consist filtering mechanism and error calculation process.
The proposed structure is shown in Fig 3. The output of weight update block is available only after n2 cycles. Filter output is available after n1 delay and input is available only n1 delay. So the modified equation is as below

\[ w_{n+1} = w_n + \mu \cdot e(n - n1) \cdot x(n-n1) \]  
\[ e(n-n1) = d(n-n1) - y(n-n1) \]  
\[ y(n) = w^Tn-n2 \cdot x(n) \]

4.1 ERROR COMPUTATION BLOCK

Error computation block of figure 3 consisting of partial product generator and adder tree and AOC(and/or cell). The function of each block is described in detail.

1) STRUCTURE OF PPG:

The structure of PPG is shown in Fig. 5 partial product generator is used for multiplication. A multiplier essentially consists of two operands, a multiplicand ‘Y’ and a multiplier ‘X’, and produces a product ‘P’. In a conventional multiplier, a number of partial products are formed first by multiplying the multiplicand with each bit of the multiplier. In the figure it consists of number of decoders, where l is number of bits. The decoder takes 2-bit input and produces output b0 b1 b2. If both input are 00 then b0=b1=b2=0. If input is 01 then b0=1, b1=b2=0. If input is 10 then b0=0, b1=1, b2=0. If input is 11 then b0=b1=0, b2=1.

The single bit data is multiplied with scaled coefficient. The coefficient is w, 2w, 3w and negative value represented as -w is multiplied with single bit input data. AOC contains 3 and gate and 2 or gate. The output of AOC is applied into adder network.

2) STRUCTURE OF AOC:

The single bit data is multiplied with scaled coefficient. The coefficient is w, 2w, 3w and negative value represented as -w is multiplied with single bit input data. AOC contains 3 and gate and 2 or gate. The output of AOC is applied into adder network.

3) STRUCTURE OF ADDER TREE:

Shift and add method is used to find the inner partial product. This method reduces the overall system delay. The structure of adder tree is shown in Fig 7. The figure shows the adder structure of 4 tap filter. The number of stages are determined by the number of bits in the input and number of tap. The equation of number of stages of adder is \( \log_2 N \) where N is the number of tap. The equation of number of stages of shift and adder is \( \log_2 L - 1 \) where L is the number of bits in the input.
4.2 WEIGHT UPDATE BLOCK
The function of weight-update block is shown in Fig.8. The convergence factor is taken to be a negative power of two to realize the corresponding multiplication of (3a) by a shift operation. The weight-update block consists of N carry-save units to update N weights. Each of those carry-save units performs the multiplication of shifted error values with the delayed input samples along with the addition with the old weights. Note that the addition of old weight with weight increment term is merged with multiplication pertaining to the calculation of weight-increment term.

V. FINITE-PRECISION EFFECTS AND OTHER IMPLEMENTATION ISSUES
In all digital hardware and software implementations of the LMS algorithm, the quantities e(n), d(n), and {x(n−i)} are represented by finite-precision quantities with a certain number of bits. Small numerical errors are introduced in each of the calculations within the coefficient updates in these situations. The effects of these numerical errors are usually less severe in systems that employ floating-point arithmetic, in which a mantissa-only numerical representation is used as compared to systems that employ fixed-point arithmetic, in which a mantissa-only numerical representation is used.

In all digital hardware and software implementations of the LMS algorithm, the quantities E(n), d.(n) and x(n−i) are This section evaluates the performance of the proposed modified least mean square (LMS) algorithm and shows the simulation results. The ISIM is the tool used here to check the performance of LMS adaptive filter. It is a complete HDL simulation environment that enables to verify the source code and functional and timing models using test bench. Input signal produce an output with amplification as shown figure. Arithmetic, in which a mantissa-only numer

CONCLUSION
In this paper, we proposed an efficient architecture for the design of a modified delayed LMS adaptive filter. By using a Partial Product Generator (PPG), the combinational blocks can achieve efficient area-delay product and energy-delay product. The proposed design gives the large efficient output comprises the existing output with large complexities. In future we propose the pipelining implementation with Partial Product Generator (PPG) across the time consuming combinational blocks of the delayed LMS adaptive filter structure. This is useful for the reduction of adaptation delay. And replacing adders in the adder circuit to check and compare the area, power and delay efficiency of the adaptive filter. we have proposed an efficient addition scheme for inner-product computation to reduce the adaptation delay significantly in order to achieve faster convergence performance and to reduce the critical path to support high input-sampling rates.

REFERENCES


