International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.3 Issue No.4, pp : 422-425 1April 2014

VLSI Implementation of Fixed-Point LMS Adaptive Filter with Low Adaptation Delay

[1]C.Preethi, [2]Ms.M.Praveena,
[1]PG scholar (M.E., VLSI design) , [2]Assistant professor/ ECE
[1,2]PSNACollege of Engineering and Technology, Dindigul
[1]preethiclement21@gmail.com, [2]praveenabe88@gmail.com

Abstract— This paper presents the modified delayed LMS adaptive filter consists of Weight update block with Partial Product Generator (PPG) to achieve a lower adaptation delay and efficient area, power, delay. To achieve lower adaptation delay, initially the transpose form LMS adaptive filter is designed but the output contains large delay due to its inner product process. Here, the pipelining structure is proposed across the time consuming combinational blocks of the structure to reduce the critical path. From the simulation results, we find that the proposed design offers large efficient output comprises the existing output with large complexities.

Keywords--**Partial Product Generator, Weight update block, Modified DLMS adaptive filter.**

Overview of proposed Method: The direct form LMS Adaptive filter used in this paper, involves long critical path due to its inner product process. So, the pipelining is implemented but it increases the sample period. After that, the transpose form LMS Adaptive filter was designed, but the output contains large delay. Next the processing elements are used for achieving low adaptation delay but there exists a critical path. So the proposed design of delayed LMS adaptive filter is used to achieve lower adaptation delay, area, power efficiently.

I.INTRODUCTION

The Least Mean Square (LMS) adaptive filter is the widely used filter because of its simplicity and performance. Least mean squares (LMS) algorithms are a class of adaptive filter used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal (difference between the desired and the actual signal). It is stochastic gradient method in that the filter is only adapted based on the error at the current time. The LMS algorithm is the most popular method for adapting a filter, which have made it widely adopted in many applications. Applications include adaptive channel equalization, adaptive predictive speech coding, Noise Suppression and on-line system identification. Recently, because of the progress of digital signal processors, a variety of selective coefficient update of gradient-based adaptive algorithms could be implemented in practice. The Least Mean Square adaptive filter is used here because it differs from a traditional digital filter in the following ways: A traditional digital filter has only one input signal $x(n)$ and one output signal $y(n)$. An adaptive filter requires an additional input signal $d(n)$ and returns an additional output signal $e(n)$. The filter coefficients of a traditional digital filter do not change over time. The coefficients of an adaptive filter change over time. Therefore, adaptive filters have a self-learning ability that traditional digital filters do not have. The filter is an important component in the communication world. It can eliminate unwanted signals from useful information. However, to obtain an optimal filtering performance, it requires 'a priori' knowledge of both the signal and its embedded noise statistical information. The classical approach to this problem is to design frequency selective filters, which approximate the frequency band of the signal of interest and reject those signals outside this frequency band.

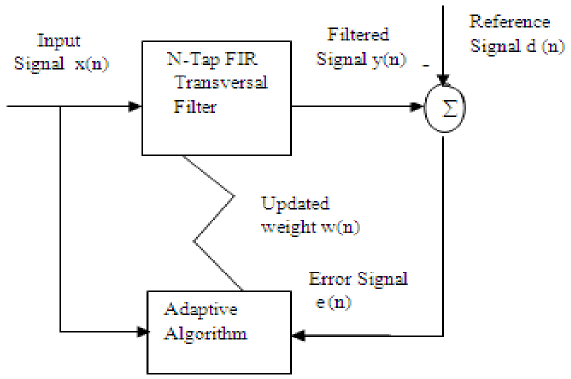

Fig.1: Adaptive filter system

The removal of unwanted signals through the use of optimization theory is becoming popular, particularly in the area of adaptive filtering. These filters minimize the mean square of the error signal, which is the difference between the reference signal and the estimated filter output, by removing unwanted signals according to statistical parameters.

II. RELATED WORKS

This algorithm is a class of adaptive filter used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal [1]. The LMS algorithm was devised for the study of a pattern-recognition machine known as the adaptive linear element. 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 weights [2]. The existing systolic architectures for the LMS algorithm with delayed coefficient adaptation have large adaptation delay and hence degraded convergence behaviour. The proposed system gives the systolic architecture with minimal adaptation delay and input/output latency, thereby improving the convergence behaviour to near that of the original LMS algorithm. [3]. An efficient systolic architecture for the DLMS adaptive filter is based on a new tree-systolic processing element (PE) and an optimized tree-level rule. Applying tree-systolic, a higher convergence rate than that of the conventional DLMS structures can be obtained without the properties of the systolic-array architecture [4]. The DLMS adaptive algorithm is introduced to achieve lower adaptation-delay. It can be implemented using pipelining. But it can be used only for large order adaptive filters [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]. The implementation of adaptive filters with fixed-point arithmetic requires to evaluate the computation quality. The accuracy may be determined by calculating the global quantization noise power in the system output [8]. The LMS algorithm is the most popular method for adapting a filter, which is used in many applications such as adaptive channel equalization, adaptive predictive speech coding, Noise Suppression and on-line system identification.

III. DELAYED LMS ADAPTIVE FILTER

For every input sample, the LMS algorithm calculates the filter output and finds the difference between the computed output and the desired response. Using this difference the filter weights are updated in every cycle. During the n-th iteration, LMS algorithm updates the weights as follows:

$$W_{n+1} = w_n + \mu \cdot e(n) \cdot x(n) \qquad (1a)$$

Where,

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

$$y(n) = wT_n \cdot x(n) \qquad (1b)$$

Here,

x(n) is the input vector
w(n) is the weight vector of an Nth order LMS adaptive filter at the nth iteration, respectively, given by,

$$x(n) = [x(n),\ x(n-1),\ \cdot\cdot\cdot,\ x(n-N+1)]^T$$
$$w_n = [w_n(0),\ w_n(1),\ \cdot\cdot\cdot,\ w_n(N-1)]^T$$

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.
μ is the convergence-factor.

The DLMS algorithm, instead of using the recent-most feedback-error e(n) corresponding to the n-th iteration for updating the filter weights, it uses the delayed error e(n−m), (i.e.) the error corresponding to (n−m)-th iteration for updating the current weight. The weight-update equation of DLMS algorithm is given by,

$$W_{n+1} = w_n + \mu \cdot e(n-m) \cdot x(n-m) \qquad (2)$$

where, m is the adaptation-delay.

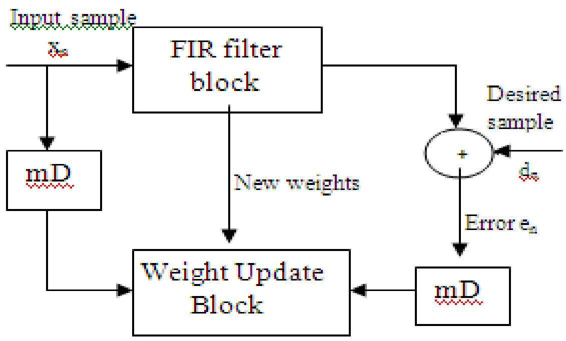

Fig 2: Structure of delayed LMS adaptive filter

The structure of conventional delayed LMS adaptive filter is shown in Fig. 1. It can be seen that the adaptation-delay m is the number of cycles required for the error corresponding to any given sampling instant to become available to the weight adaptation circuit.

IV.PROPOSED SYSTEM

In the conventional DLMS algorithm (Fig.1) the adaptation delay of m cycles amounts to the delay introduced by the whole of adaptive filter structure consisting of FIR filtering and weight adaptation process. But instead, this adaptation delay could be decomposed into two parts. One part is the delay introduced due to the FIR filtering and the other part is due to the delay involved in weight adaptation.

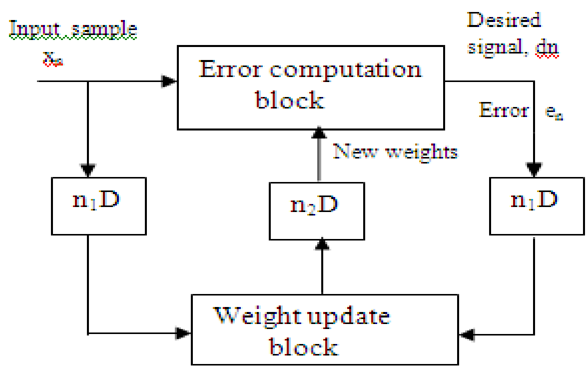

Fig 3: Structure of modified DLMS adaptive filter

Based on such decomposition of delay, the proposed structure of DLMS adaptive filter is shown in Fig.3. The proposed adaptive filter architecture consists of two main computing blocks, namely the error computation block and weight-update block. The computation of filter output and the final subtraction to compute the feedback error are merged in the error computation unit to reduce the latency of error computation path. If the latency of computation of error is n1 cycles, the error computed by the structure at the nth cycle is $e(n - n1)$, which is used with the input samples delayed by n1 cycles to generate the weight-increment term. The weight-update equation of the proposed delayed LMS algorithm is, therefore, given by,

$$w_{n+1} = w_n + \mu \cdot e(n - n1) \cdot x(n - n1) \qquad (3a)$$

Where,

$$e(n - n1) = d(n - n1) - y(n - n1) \qquad (3b)$$

and

$$y(n) = w^T_{n-n2} \cdot x(n) \qquad (3c)$$

We can notice that during weight adaptation, the error with n1 delays is used while the filtering unit uses the weights delayed by n2 cycles. By this approach the adaptation-delay is effectively reduced by n2 cycles. The proposed algorithm can be implemented efficiently with very low adaptation-delay which is not effected substantially by the increase in filter order.

4.1 Error computation block

The error computation block is implemented by a pipelined inner-product computation module (Fig.3) and the weight-update block is implemented by N number of pipelined multiply-accumulate units. The multiplications of input samples with corresponding weights followed by their summation in the filter computation block have been realized by a carry-save chain and the error computation is merged with the filter output computation. In the weight update block, the multiplication of the convergence-factor with the error and input values are combined with the addition with old weights according to (3a) to obtain the final updated weights in N parallel carry-save chains.

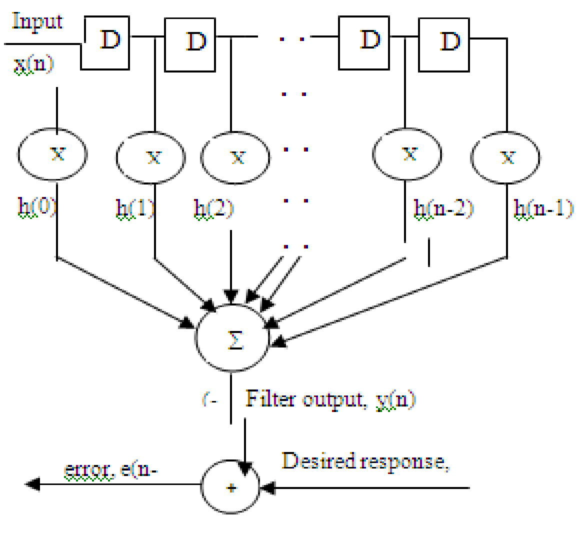

Fig 4: Structure of Error-computation block

4.2 Weight update block

The function of weight-update block is shown in Fig.4. 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.

The final outputs of the carry-save units constitute the updated weights which serve as an input to the error computation block as well as the weight-update block for the next iteration. A pair of delays are introduced before and after the final addition of the weight-update block to keep the critical-path equal to one addition time. The shaded region in Fig.6 indicates the carry- save unit corresponding to the first weight which takes the delayed input samples and shifted from of delayed error value (to take care of the multiplication by convergence-factor) and the old weights as input. Thus the weight-update block takes a total of two delays, i.e., n2 = 2.

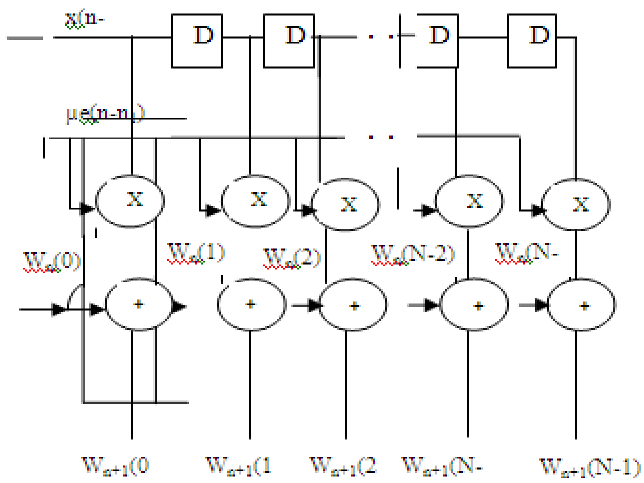

Fig 5: Structure of Weight update block

V.PERFOMANCE RESULTS

This section evaluates the performance of the proposed modified least mean square (LMS) algorithm and shows the simulation results. The first result declares about the output of LMS adaptive filter with delay. It is having some delay in the output of Least Mean Square adaptive filter. And the second result declares about the output of LMS adaptive filter without delay. After the clock input has given the output of the adaptive filter is achieved without delay. The ModelSIM 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.

OUTPUT OF LMS ADAPTIVE FILTER WITH DELAY

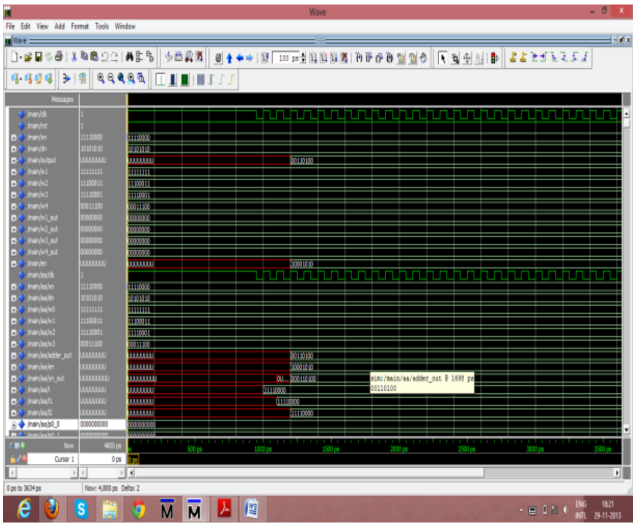

Fig 6: Output with delay

OUTPUT OF LMS ADAPTIVE FILTER WITHOUT DELAY

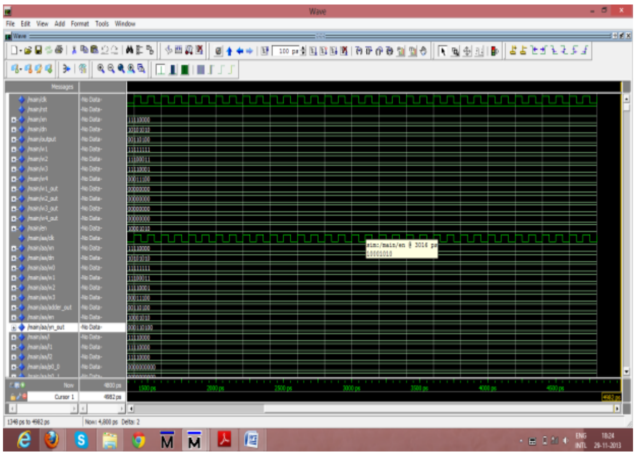

Fig 7: Output without delay

VI. 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.

References

i. *B.Widrow and S. D. Stearns, Adaptive Signal Processing. Englewood Cliffs, NJ, USA: Prentice-Hall, 1985.*

ii. *S. Haykin and B. Widrow, Least-Mean-Square Adaptive Filters. Hobo-ken, NJ, USA: Wiley, 2003.*

iii. *S. Ramanathan and V. Visvanathan, "A systolic architecture for LMS adaptive filering with minimal adaptation delay," in Proc. Int Conf Very Large Scale Integr,(VLSI) Design,Jan. 1996,pp. 286-289.*

iv. *Y. Yi, R. Woods, L.-K. Ting, and C. F. N. Cowan, "High speed FPGA-based implementations of delayed-LMS filters," J. Very Large Scale Integr. (VLSI) Signal Process., vol. 39, nos. 1–2, pp. 113–131, Jan. 2005.*

v. *L. D. Van and W. S. Feng, "An efficient systolic architecture for the DLMS adaptive filter and its applications," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process., vol. 48, no. 4, pp. 359–366, Apr. 2001.*

vi. *P.K.Mehar and M.Maheshwari, "A high speed FIR adaptive filter architecture using a modified delayed LMS algorithm," in Proc. IEEE Int Symp. Circuits Syst, May 2011,*

vii. *K. K. Parhi, VLSI Digital Signal Procesing Systems: Design and Implementation. New York, USA: Wiley, 1999.*

viii. *C. Caraiscos and B. Liu, "A roundoff error analysis of the LMS adaptive algorithm," IEEE Trans. Acoust., Speech, Signal Process., vol. 32, no. 1, pp 34–41, Feb. 1984.*

ix. *R. Rocher, D. Menard, O. Sentieys, and P. Scalart, "Accuracy evaluation of fixed-point LMS algorithm," in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., May 2004, pp. 237–240.*

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

xi. *G. Long, F. Ling, and J. G. Proakis, "The LMS algorithm with delayed coefficient adaptation," IEEE Trans. Acoust., Speech, Signal Process., vol. 37, no. 9, pp. 1397–1405, Sep. 1989.*

xii. *G. Long, F. Ling, and J. G. Proakis, "Corrections to 'The LMS algorithm with delayed coefficient adaptation'," IEEE Trans. Signal Process., vol. 40, no. 1, pp. 230–232, Jan. 1992.*

xiii. *H.Herzberg and R. Haimi-Cohen, "A systolic array realization of an LMS adaptive filter and the effects of delayed adaptation," IEEE Trans. Signal Process., vol. 40, no. 11, pp. 2799–2803, Nov. 1992.*

xiv. *M. D. Meyer and D. P. Agrawal, "A high sampling rate delayed LMS filter architecture," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process., vol. 40, no. 11, pp. 727–729, Nov. 1993.*

xv. *L.-K. Ting, R. Woods, and C. F. N. Cowan, "Virtex FPGA imple-mentation of a pipelined adaptive LMS predictor for electronic support measures receivers," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 1, pp. 86–99, Jan. 2005.*