Survey on DLMS Adaptive Filter With Low Delay

T.J.MILNA (M.E.,) #1, S.K MYTHILI (Ph.D) #2

ECE Department, SVS COLLEGE OF ENGINEERING, COIMBATORE

1milnachithu@gmail.com
2mydhili2@gmail.com

Abstract-In practical applications of the LMS adaptive transversal filtering algorithm, a delay in the coefficient is updated. This paper discusses the behavior of the delayed LMS algorithm. This paper presents an efficient architecture for the implementation of a delayed least mean square adaptive filter. In order to achieve lower adaptation delay, a novel partial product generator is used. The convergence and steady state behaviors of the adaptive filter are compare and analyze.

Keywords- Area Delay Product (ADP), Energy Delay (EDP), systolic architecture, adaptive delay, steady state behavior, convergence.

I. INTRODUCTION

Adaptive digital filters have been applied to a variety of important problems in recent years. Perhaps one of the most well known adaptive algorithms is the least mean squares (LMS) algorithm, which updates the weights of a transversal filter using an approximate technique of steepest descent. Due to its simplicity, the LMS algorithm has received a great deal of attention, and has been successfully applied in a number of areas including channel equalization, noise and echo cancellation and many others.

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 signal and the actual signal). It is a stochastic gradient descent method in which the filter is adapted based on the current time error.

The basic idea behind LMS filter is to update the filter weights to converge to the optimum filter weight. The algorithm starts by assuming a small weights (zero in most cases), and at each step, where the gradient of the mean square error, the weights are found and updated. If the MSE-gradient is positive, the error increases positively, else the same weight is used for further iterations, which means we need to reduce the weights.

If the gradient is negative, weight need to be increased. Hence, basic weight update equation during the nth iteration:

\[ w_{n+1} = W_n - \mu \Delta \varepsilon(n) \]  

where \( \varepsilon \) represents the mean-square error, \( \mu \) is the step size, \( W_n \) is the weight vector. The negative sign indicates that, need to change the weights in a direction opposite to that of the gradient slope. The mean-square error which is a function of filter weights is a quadratic function which says that it has only one extreme, which minimizes the mean-square error. is the optimal weight. The LMS thus, approaches towards this optimal weight by ascending/descending down the mean-square-error verses filter weight curve.

II. OVERVIEW OF LITERATURE SURVEY

The LEAST MEAN SQUARE (LMS) adaptive filter is the most popular and most widely used adaptive filter, not only because of its simplicity but also because of its satisfactory convergence performance [1] [2]. 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 nature, it is modified to a form called the Delayed Least Mean Square (DLMS) algorithm [3][5], which allows pipelined implementation of the filter.

A lot of work has been done to implement the DLMS algorithm in systolic architectures to increase the maximum usable frequency [3] [6] [7] but, they involve an adaptation delay of ~ N cycles for filter length N, which is quite high for large order filters. Since the convergence performance degrades considerably for a large adaptation delay, Visvanathan et al.[8] have proposed a modified systolic architecture to reduce the adaptation delay. A transpose-form LMS adaptive filter is suggested in [9], where the filter output at any instant depends on the delayed versions of weights and the number of delays in weights varies from 1 to N. Van and Feng [10] have proposed a systolic architecture, where they have used relatively large processing elements (PEs) for achieving a lower adaptation delay with the critical path of one MAC operation. Ting et al.[11] have proposed a fine-grained pipelined design to limit the critical path to the maximum of one addition time, which supports high sampling frequency, but involves a lot of area overhead for pipelining and higher power consumption than in [10], due to its large number of pipeline latches. Further effort has been made by Meher and Maheshwari [12] to reduce the number of adaptation delays. Meher and Park have proposed a 2-bit multiplication cell, and used that with an efficient adder tree for pipelined inner-product computation to minimize the critical path and silicon area without increasing the number of adaptation delays [13][14].

The existing work on the DLMS adaptive filter does not discuss the fixed-point implementation issues, e.g., location of radix point, choice of word length, and quantization at various stages of computation, even though they directly affect the performance of the convergence, particularly due to the recursive behavior of the LMS algorithm. we present here the optimization of our previously reported design [13], [14] to reduce the number of pipeline delays with the area, sampling period, and energy consumption. The proposed design is found to be more efficient in terms of the power-delay product (PDP) and energy-delay product (EDP) compared to the existing structure.

III. DLMS ADAPTIVE FILTER MODULAR PIPELINED IMPLEMENTATION

A modular pipelined filter architecture [3] based on a time-shifted version of the DLMS algorithm is discusses here. This pipelined architecture displays the most desirable features of both lattice and transversal form adaptive filters. As in an adaptive lattice filter, the computations are structured to be order recursive, resulting in a highly pipelined implementation. Also, the weights are updated locally within each stage. However, the equations being implemented actually correspond to a true transversal adaptive filter, and hence desirable properties of this structure are preserved. The modular pipeline consists of a linear array of identical processing elements (PES) which are linked together using both local and feedback connections. Each PE performs all the computations associated with a single coefficient of the filter.

A significant advantage of the modular structure of the pipelined DLMS filter is that, unlike conventional transversal filters, the order of the filter can be increased by simply adding more PE modules to the end of the pipeline. The performance of the system is computed using speed up over single processor system. The other advantages of this structure are High throughput. Useful for real time applications and its easily expandable.

IV. VIRTEX FPGA IMPLEMENTATION OF PIPELINED ADAPTIVE LMS PREDICTOR

FPGAs provide a good combination of high-speed implementation features with the flexibility of a COTS platform [11]. FPGA’s have grown over the past decade to the point where there is no wan assortment of adaptive algorithms which can be implemented on a single FPGA device. However, the direct implementation of an adaptive filter on an FPGA often proves to be slow due to the error feedback signal in the recursive structure. Typically, the system throughput rate of many DSP algorithms can be improved by exploiting concurrency in the form of parallelism and pipelining. Since virtex is used very limited area is increased. The pipelined architecture shows better interconnect delays.

V. SYSTOLIC ARCHITECTURE OF DLMS ADAPTIVE FILTER AND APPLICATION
LMS adaptive algorithm reduces approximately the mean-square error by recursively altering the weight vector at each sampling instance.

\[
y_{n} = W_{n}^{T} x_{n}
\]

(2)

\[
e_{n} = d_{n} - y_{n}
\]

(3)

\[
W_{n+1} = W_{n} + \mu e_{n} x_{n}
\]

(4)

Where \(d_{n}\) is the desired signal and \(y_{n}\) is the output signal. The step-size is used for adaptation of the weight vector, and \(e_{n}\) is the feedback error. The work is focused on reducing delay and critical path at the same time satisfying the requirements of a systolic array. It is known that the tree method enhances the performance of adaptive FIR digital systems. However, the tree structure lacks driving-consideration, modularity, and local-connection[10]. While the number of tree levels increases, the critical period would be sacrificed since the pipeline is not sufficiently full. Here, the tree concept is applied to devise a new generalized tree-systolic processing element. The design parameters involve the desired critical period, operating voltage, aspect ratio, and logic style. Let the maximum number of tap-connections of the feedback error signal be just larger than or equal to the value to achieve a high degree of reliability and convenient processing. The processing element operates at high throughput and with local connection unlike the earlier structures. The two convergence parameters delay (D) and required number of different kinds of PEs (Np), affects the system more than the number of kinds of PEs. Hence optimum value of delay and Np has to be chosen. Here the main aim is to reduce the value of d hence the value of p is chosen such that it gives minimum delay. If there are more than one value of p that gives same delay, then the value of Np is taken into account. The value of p with low Np is considered. The system proposed is implemented in two various applications namely, the system identification and adaptive equalization.

**Fig 1.** Overall architecture of systolic array

**Fig 2.** Block diagram of system identification

**Fig 3.** Block diagram of adaptive equalization

The advantage of this method is that it reduces delay and critical path with finite driving, local connections, and satisfactory convergence at no extra area cost.

**VI. LOW ADAPTATION DELAY LMS ADAPTIVE FILTER**

To implement the LMS algorithm, during each sampling period of training phase, one has to compute a filter output and an error value which equals to the difference between the current filter output and the desired response. The estimated error is used to update the filter weights in every cycle. In case of pipelined designs, the feedback-error \(e(n)\) corresponding to the \(n\)th iteration is not available for updating the filter weights in the same
iteration. It becomes available after certain number of cycles, called the adaptation delays. The DLMS algorithm therefore uses the delayed error. In this method the critical path is reduced by using a pipelined register structure.

Fig 4. Modified multiplier unit

A new multiplier block is utilized in the error computation multiplication like a conventional MAC unit. It consists of \( L/2 \) AND/OR cells (AOC), where \( L \) represents the length of input bits and 2-to-3 decoders. This modified structure is shown in figure 5.

The area delay product and energy delay product of the proposed multiplier cell are considerably less compared to other architectures.

VII. SUMMARY OF EXPERIMENTAL RESULTS

This table 1 shown the synthesis result of the proposed existing design in terms of area, leakage power, energy per sample (EPS) and ADP obtained for filter length \( N=8,16, \) and 32. The design explained in [18] could achieve less area and more power reduction compared with [11] by removing redundant pipeline latches, which are not required to maintain a critical path of one addition time. It is found that the proposed design involves 17% less ADP and 14% less EDP than the best previous work of (10), on average for filter length \( N=8,16 \) and 32. The proposed design was also implemented on the field table programmable gate array (FPGA) platform of Xilinx devices.

<table>
<thead>
<tr>
<th>Design</th>
<th>Filter length, N</th>
<th>Area</th>
<th>Leakage POWER</th>
<th>EPS</th>
<th>ADP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ting et al</td>
<td>8</td>
<td>24204</td>
<td>0.13</td>
<td>18.49</td>
<td>26867</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>48049</td>
<td>0.27</td>
<td>36.43</td>
<td>53591</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>95693</td>
<td>0.54</td>
<td>72.57</td>
<td>116745</td>
</tr>
<tr>
<td>Vam and</td>
<td>8</td>
<td>13796</td>
<td>0.08</td>
<td>7.29</td>
<td>81349</td>
</tr>
<tr>
<td>Feng[10]</td>
<td>16</td>
<td>27739</td>
<td>0.16</td>
<td>14.29</td>
<td>56893</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>556580</td>
<td>0.52</td>
<td>27.64</td>
<td>73998</td>
</tr>
<tr>
<td>Pramod</td>
<td>8</td>
<td>14659</td>
<td>0.07</td>
<td>8.33</td>
<td>15552</td>
</tr>
<tr>
<td>Kumar[18]</td>
<td>16</td>
<td>26660</td>
<td>0.14</td>
<td>14.58</td>
<td>31192</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>48317</td>
<td>0.27</td>
<td>21.00</td>
<td>38824</td>
</tr>
</tbody>
</table>

VIII. CONCLUSION

This paper presents a survey of the existing adaptive filter implementation with low adaptation delay. This survey briefly describes the principles behind the adaptive filter in order to understand different implementation styles in a better way and their structures. From the comparison of these techniques it is concluded that the partial product generator (PPG) architecture is the best for implementation of low power adaptive filter. This architecture used a novel PPG for efficient implementation of general multiplications and inner-product computation by common sub expression sharing. Besides, this implementation uses an efficient addition scheme for inner-product computation to reduce the adaptation delay thereby achieving fast convergence performance. Further it reduces the critical path to support high input-sampling rates.
REFERENCES


[18]. Pramod kumar Meher,Sang yoon park “Area delay power efficient fixed point LMS adaptive filter with low adaptation delay”IEEE trans on Very Large Scale Integration system, vol.22,No.2 feb.2014