Retiming of Delayed Least Mean Square Algorithm for Adaptive Filter: A Review

B.V. Hood¹ Prof. R.N. Mandavgane² Prof. J.D. Dhande³
¹M.Tech. Student ²³Professor
1.2.3Department of Electronics & Telecommunication Engineering
¹, 2, 3Bapurao Deshmukh College of Engineering, Sewagram (Wardha) Maharashtra

Abstract—Least mean square (LMS) algorithm is a broadly used adaptive algorithm for its robustness and low hardware complexity. Though in practical applications, many modified LMS algorithm had been proposed. Delayed LMS algorithm is suited in hardware implementation with performance degradation. When performance degradation is not acceptable, a correction term can be added to avoid this degradation. But adding a correction term will increase power consumption. To make this more achievable we are retiming this DLMS architecture to reduce the critical path and making it more responsive and less degradable. We are indulge in this paper to understand the throughput changes that will empower whenever we are going to change realization of LMS from LMS_DLMS_RDLMS.

Key words: Adaptive Filtering, Delayed LMS Algorithm, Retiming Technique, Pipelining

I. INTRODUCTION

Least Mean Squares (LMS) algorithm is one of the class of adaptive filter which is used to imitate a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal. Error signal is a difference between the desired and the actual signal. It is stochastic gradient descent method in that the filter is only adapted based on the error at the current time.

A variation of the least mean square (LMS) algorithm, called the delayed LMS (DLMS) algorithm is mainly suited for highly pipelined, adaptive digital filter implementations. Delay is an inherent problem. The conventional delayed least mean square (LMS) algorithm converges more slowly than the LMS algorithm due to the delay introduced in coefficient adaptation.

Retiming is a transformation technique used to change the position of delay elements in a circuit. The purposes of retiming are to facilitate to minimize the clock period and to reduce number of registers needed. While pipelining produces a mechanism to reduce the critical paths and improve the speed performance, pipelining of the adaptive LMS filter is difficult as it introduces delay in the error feedback loop. During the past two decades, many researchers have proposed several algorithms in order to overcome this difficulty. These approaches involve the modification of the adaptation algorithm to allow pipelining to be effectively implemented. These have resulted in the development of look-ahead and pipelined DLMS based algorithms that act to ensure stable convergence behaviour in filters that have been pipelined. However, the current DLMS-based architectures exhibit either a short critical path but a slow convergence and poor tracking, or fast convergence and good tracking but a long critical path.

II. LMS, DLMS AND RDLMS ALGORITHM

A. Least Mean Square Algorithm

1) Introduction:

The LMS algorithm is the most widely used adaptive filtering algorithm in practice. The wide spectrum of applications of the LMS algorithm can be attributed to its simplicity and robustness to signal statistic. The key feature of the LMS algorithm is its simplicity. It requires neither measurement of the correlation function, nor matrix inversion. It uses Mean Square Error (MSE) as a criterion. LMS uses a step size parameter, input signal and the difference of desired signal and filter output signal to frequently calculate the update of the filter coefficients set.

Fig. 1: Schematic of LMS algorithm

2) Summary:

LMS stands for Least Mean Square, this is an algorithm which reduces error by updating weight at each tap interval change. This all efforts are on account of making error as low as possible. Constrain in this LMS are the feedback of the error signal needed to update filter weights in the LMS algorithm imposes a critical limitation on the throughput of possible implementations. This issue was addressed in DLMS algorithm.

B. Delayed Least Mean Square Algorithm:

1) Introduction:

This issue was researched by scientist who conducted a study of the Delayed LMS (DLMS) algorithm and showed
that the delay in the coefficient adaptation has only a slight influence on the steady-state behavior of the LMS algorithm if the step size is within certain bounds. Fig. 2 shows a block diagram of an adaptive DLMS model. Usually, a smaller step size compared to the standard LMS algorithm has to be applied. The major penalties with the DLMS algorithm are a reduced convergence speed for stationary signals and a poorer tracking performance for non-stationary signals.

2) Challenges:
The main challenge now is to use these delays as a means of pipelining the LMS filter. This process firstly involves the determination of the number of delays needed to achieve a fully-pipelined version of the circuit. This is important as a value that is too small will lead to a low-speed circuit, while too large a value will produce slower convergence rate and poor tracking capacity. Once this has been determined, a suitable retiming technique is then used to re-distribute these delays throughout the circuit architecture in order to achieve a fully pipelined implementation.

C. Retiming Concept and Techniques:
1) Introduction:
Retiming is a transformation technique used to change the location of delay elements in a circuit without affecting the input/output characteristics of the circuit. Retiming has many applications in synchronous designs including the reduction of the clock period of the circuit, reduction of the power consumption of the circuit, and logical synthesis. In the clock period and the number of registers were reduced for a signal flow graph (SFG) by redistributing the delays. Retiming can be used to reduce the power consumption of a circuit by reducing switching, which can lead to dynamic power dissipation in static CMOS circuits. Unlike pipelining, retiming does not increase the circuit latency.

2) Summary:
Retiming is changing delay position from path so that critical path can be reduced to iteration bound which can be done. By changing position of delays it makes us achieve to have critical path to iteration bound it is being explained in RDLMS more practically. Over that retiming have more advantages it slows circuit operation but computation time is less.

III. LITERATURE REVIEW
Literature review is the most important methodology as it provides the guidelines and evident supports for the research work, by extracting information on the relevant topic and work done on existing project.

With reference to paper by Ying Yi and Roger Woods et.al [1] presents an efficient system level design flow for implementing FPGA hardware using IRIS and System Generator. They found two issues, first is circuit timing issue and second is latency issue. To overcome these issues, they used IRIS synthesis tool (developed for VLSI chip design) and for latency issue, retiming of the SFG architecture was used. They designed 8-bit/8-tap TF-FPDLMS and TF-RDLMS predictor systems and they achieved a smaller area and higher speed in TF-RDLMS filter than TF-FPDLMS filter. The whole system is simulated using VHDL netlist and implemented using the Xilinx Virtex-II FPGA technology resulting in a throughput rate of 170 Msample/s.

<table>
<thead>
<tr>
<th>Circuit Name</th>
<th>Sampling Rate(MHz)</th>
<th>Critical Paths(ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-FPDLMS1/2</td>
<td>169.588</td>
<td>5.923</td>
</tr>
<tr>
<td>TF-RDLMS1/2_SRL</td>
<td>169.722</td>
<td>5.873</td>
</tr>
</tbody>
</table>

Table 1: Virtex-II implementation (post-layout) results for sampling rate and area [1]

Figure above shows Virtex-II implementation (post-layout) results for sampling rate and area. Here comparative parameters of both TF-FPDLMS and TF-RDLMS predictor systems are given which shows for TF-RDLMS predictor system, sampling rate is 169.722MHz and critical path is 5.173ns which is less as compared to TF-FPDLMS predictor systems.

<table>
<thead>
<tr>
<th>Circuit Name</th>
<th>Sampling Rate(MHz)</th>
<th>Clock rate(MHz)</th>
<th>Critical Paths(ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-RDLMS1/2</td>
<td>80.65</td>
<td>161.3</td>
<td>6.2</td>
</tr>
<tr>
<td>TF-RDLMS1/2_SRL</td>
<td>81.95</td>
<td>163.9</td>
<td>6.1</td>
</tr>
<tr>
<td>TF-RDLMS1/2.NEW</td>
<td>86.2</td>
<td>172.4</td>
<td>5.8</td>
</tr>
<tr>
<td>TF-RDLMS1/4</td>
<td>41.675</td>
<td>166.7</td>
<td>6.0</td>
</tr>
<tr>
<td>TF-RDLMS1/4_SRL</td>
<td>42.375</td>
<td>169.5</td>
<td>5.9</td>
</tr>
<tr>
<td>TF-RDLMS1/4_NEW</td>
<td>46.725</td>
<td>186.9</td>
<td>5.4</td>
</tr>
<tr>
<td>TF-RDLMS1/8</td>
<td>21.925</td>
<td>175.4</td>
<td>5.7</td>
</tr>
<tr>
<td>TF-RDLMS1/8_SRL</td>
<td>22.725</td>
<td>181.8</td>
<td>5.5</td>
</tr>
<tr>
<td>TF-RDLMS1/8_NEW</td>
<td>24.613</td>
<td>196.9</td>
<td>5.1</td>
</tr>
</tbody>
</table>

Table 2: Hardware sharing (post-layout) results for sampling rate and area [2]
The paper of Asit Kumar Subudhi, Biswajit Mishra, Mihir Narayan Mohanty et.al [3] presented a low cost and high performance programmable digital finite impulse response (FIR) filter and designed a hybrid adaptive filter with a direct-form FIR filter. The proposed system was Coded in VHDL, executed on the Xilinx and output is simulated on MATLAB.

Muthulakshmi.G, Revathi.S et.al [4] presented an efficient architecture for the implementation of a delayed least mean square Adaptive filter. They achieved the proposed design with less area-delay product (ADP) and less energy-delay product (EDP). For this they used a novel partial product generator i.e. PPG. They also presented the optimization of design to reduce the number of pipeline delays along with the area, sampling period, and energy consumption. The simulation is carried out by the Modelsim 6.3f as a simulator tool. Here, the future work involves that to reduce the adder complexity by replacing the various adders in adder blocks to achieve the better performance of area and power. Here, they used a adder tree optimization concept for optimization of area, delay and power complexity.

C.Preethi, Ms.M.Praveena et.al [5] presented the modified delayed LMS adaptive filter consist of Weight update block with Partial Product Generator (PPG). They achieved efficient area-delay product, energy-delay product and lower adaptation delay. For this they used ModelISIM tool. In future, they will propose the pipelining implementation with PPG across the time consuming combinational blocks of the delayed LMS adaptive filter structure.

Shashikala Prakash, Renjith Kumar T.G, Subramani H et.al [6] implemented on Xilinx Virtex–4 FPGA as part of realization of an Active Vibration Control system. In this paper, they considered both fixed point and floating point data representations and their comparative results showed that a convergence rate of fixed point and floating point implementation with buffer size16 for mu values 0.1 and 0.05 respectively. Floating point resulted in greater accuracy, faster convergence compared to fixed point. Residual error with floating point implementation was almost negligible. The proposed system implemented in VHDL using Xilinx ISE tool 12.4 Version and simulated on Xilinx ISIM (ISE Simulator).

DevendraGoyal, Manish Singhal et.al [7] developed various algorithms for active interference cancellation to obtain adaptive filter mainly LMS, NLMS and RLS algorithm. They found NLMS has a better learning rate than LMS based adaptive filter. The synthesize tool QUARTUS-II presented that the DLMS algorithm has a faster pipeline architecture than LMS algorithm in the cost of using more chip area due to use of extra registers. To design of digital signal processing application, FPGA was used. They found that now days FPGA systems are replacing dedicated PDSP systems due to their greater tractability and eminent bandwidth.

IV. COMPARATIVE STUDY OF EARLIER WORK

In a first paper, the author Ying Yi and Roger Woods used a Platform/Software of VHDL netlist and Xilinx Virtex–II FPGA. They achieved a speed of 170 Msample/s and smaller area. In a second paper, the author Y. YI AND, R. WOODS, L.K.TING, C.F.N.COWAN used a Platform/Software of Matlab and Xilinx Virtex–II. They achieved a
speed of 182 Msamples/s and reduction in area. In a third paper, the author Asit Kumar Subudhi, Biswajit Mishra, Mihir Narayan Mohanty suggested a Platform/Software of VHDL, Xilinx and MATLAB. They achieved a power consume with less adaptation delay and low cost with high performance. In a fourth paper, the author Muthulakshmi.G, Revathi.S used a Platform/Software of ModelSim 6.3f. They achieved a Less ADP and EDP, less area, less power consumption and lower delay. In a fifth paper, the author C.Preethi, Ms.M.Praveena used a Platform/Software of ModelSIM. They achieved a Lower adaptation delay. In a sixth paper, the author Shashikala Prakash, Renjith Kumar T.G, SubramaniH used a Platform/Software of VHDL, Xilinx ISE 12.4 Version and Xilinx ISIM. They achieved a 90% reduction in error. In a seventh paper, the author Devendra Goyal, Manish Singhal suggested a Platform/Software of QUARTUS-II. They achieved a faster speed and less area.

V. CONCLUSION

As per discussion in a literature review Retimed DLMS reduces critical path upto 5.1ns and increases its speed upto 182 Msamples/s. We propose a retiming of delayed least mean square algorithm for adaptive filter to increase the speed of the devices. The main target is to increase the speed of these devices by reducing the critical path.

REFERENCES


