International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 2, Issue. 5, Sep.-Oct. 2012 pp-3479-3472 ISSN: 2249-6645

FPGA Implementation of the Block RlS Algorithm

S.VijayaLakshmi[1], K.Raghuram[2]
[1.] *M.Tech Scholar, E.C.E, Pragati Engineering College, Surampalem, JNTUK, India*
[2.] *Associate professor, E.C.E, JNTUK*

ABSTRACT: *Recursive Least Square (RLS) algorithm which recursively find the filter coefficients that minimises a weighted linear least square cost function. Recent digital transmission systems impose the application of channel equalizers with short training time and high tracking rate. These requirements turn our attention to adaptive algorithms, which converge rapidly. The demand for fast convergence and less MSE level cannot be met by conventional adaptive filtering algorithms such as LMS. The best choice is the block recursive least squares (RLS) algorithm. Block Recursive Least Squares algorithms are known to exhibit better performances. In this paper we present a simple architecture for the implementation of a variant of Block RLS algorithm where the weight updation and error calculation are both calculated block wise. The performance of the Simplified BRLS and LMS algorithms are compared in MATLAB simulations and the hardware outputs from the FPGA are verified with the simulations.*

Keywords: *RLS, ALTERA, Modelsim, Quartus, Matlab.*

I. INTRODUCTION

In spite of the computation efficiency of the algorithm, additional simplifications may be necessary in some applications. There are several approaches that may be used to reduce the computational requirements of the RLS algorithm. One of these is the block RLS algorithm [2]. Block digital filtering has been extensively discussed by Burrus, Mitra and others. The technique involves calculation of a block of finite set of output values from a block of input values. Block implementations of digital filters allow efficient use of parallel processors which can result in speed gains [1]

RLS is one of the adaptive filtering algorithms derived from steepest descent algorithm used in wide variety of applications including System Identification, Channel Equalization, Adaptive antennas, and Noise cancellation. There are several variants of RLS which are selected depending on the system constraints like speed, hardware efficiency, accuracy of measurements etc. Block RLS is one of the variants in which the weights are updated once per every block data instead of updating on every clock cycle of input data. In block RLS, though the updation is done at block level the error values are calculated every clock cycle and are averaged over the block of the data. In this paper we further simplified these calculations and performed the error calculation also, once per block of data, making it a block implementation in the true sense and making it affable for hardware implementation.

The trade off for this approximation is the increased delay for convergence of weights and increase in oscillations of the error square plot. But with this approximation the algorithm could be easily implemented on FPGA using minimal hardware. The rest of the paper is Organized as follows. Section III describes the architecture of implementation. Section IV briefs on simulation results. Section V shows the hardware utilization summary; Section VI contains the conclusion of the paper.

II. ADAPTIVE SYSTEM IDENTIFICATION

The adaptive system identification is primarily responsible for determining a discrete estimation of the transfer function for an unknown digital or analog system. The same input x (n) is applied to both the adaptive filter and the unknown system from which the outputs are compared (see figure 1). The output of the adaptive filter y (n) is subtracted from the output of the unknown system resulting in a desired signal d(n). The resulting difference is an error signal e(n) used to manipulate the filter coefficients of the adaptive system trending towards an error signal of zero. After a number of iterations of this process are performed, and if the system is designed correctly, the adaptive filter's transfer function will converge to, or near to, the unknown system's transfer function. For this configuration, the error signal does not have to go to zero, although convergence to zero is the ideal situation, to closely approximate the given system. There will, however, be a difference between adaptive filter transfer function and the unknown system transfer function if the error is nonzero and the magnitude of that difference will be directly related to the magnitude of the error signal.

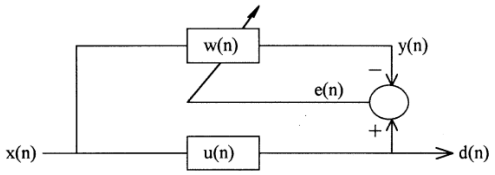


Figure1. Adaptive filter.

A. Adaptive Filters in System Identification Mode:

Since there are many applications for adaptive filters such as those just described, we often alter the structure of the filter used to suit the application. In this project we are interested in the use of adaptive filters for the purposes of system identification in which case a block diagram of the filter will be of the form shown in Figure 2. Notice that the error signal in this case is the difference between the filter output and the desired response, the output of the unknown system.

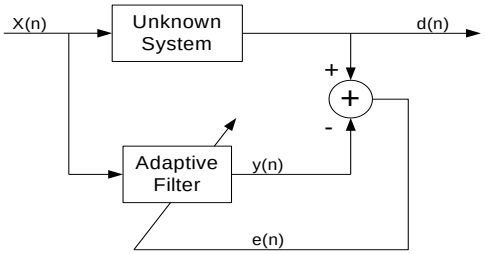


Figure2. An adaptive filter in system identification mode

B. Convolution:

In this sub-section, we will define the above term for the purposes of this project. Let us first define Correlation, in essence the Cross Correlation of two waveforms is a method for comparing the waveforms. Consider two waveforms, both sampled at the same rate. The sum of the products of the corresponding pairs of points is represented as a measure of the correlation of the two waveforms. Convolution can now be introduced as the cross correlation of two signals with one of them reversed [4]. Equation 1.0 defines the process for continuous time signals.

$$y(t) = x(t) * h(t) \qquad \textbf{1.0}$$

where * denotes convolution, when ***x(t) and h(t)*** are two finite digital signals this Equation can be represented as

$$y(n) = \sum_{k=0}^{N-i} h(k) - x(n-k) \qquad \textbf{1.1}$$

Notice that this is the same as Equation 1.1, which describes the operation of digital filters. In this case one of the sequences to be convolved is the impulse Response of the digital filter. Therefore it can be recognized that digital filtering is an application of convolution. Convolution can be viewed as a description of how the input to a given system reacts with the system's impulse response to produce an output. Convolution in the time domain is very computationally intensive and the fact that the convolution of two sequences can be calculated more efficiently if the signals are transformed from the time domain to the frequency domain is exploited in following chapters in an attempt to reduce the computation necessary for digital filters.

C. Recursive and non Recursive Filters:

Non-recursive filters are filters where the output depends only on current and previous input samples as depicted in Figure 3. An important property of non-recursive filters is they will always be stable. FIR filters are in general non-recursive, which in turn means that they are also always stable [2].

A recursive filter on the other hand is a filter whose output samples may depend on previous output samples as well as the current and previous input samples.

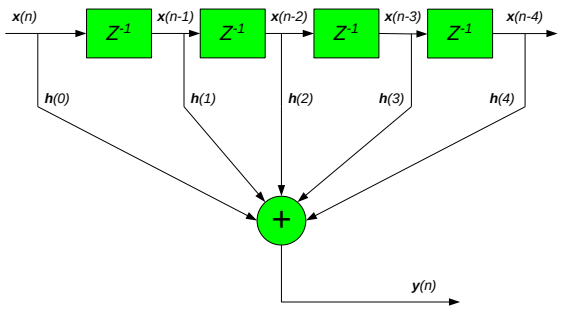


Figure 3. A Non-Recursive filter structure

Having introduced the concept of digital filters and the fact that the basis for digital filtering is time domain convolution, in particular stable FIR filters.

D. Digital filtering (Echo cancellation):

The hybrid is basically a bridge circuit with three ports and it is here that impedance mismatches can occur if the bridge is not perfectly balanced [1]. The adaptive filter is in essence the echo canceller. This filter operates in system identification mode with the echo path representing the unknown system the filter is to identify. Notice the inputs to the adaptive filter, speech from the transmitter constitutes the desired response and Equation 1.2 represents the error signal.

$$e(n) = s(n) - y(n) \qquad \textbf{1.2}$$

where ***y(n)*** is the output of the adaptive filter and ***s(n)*** is composed of the other speaker's speech and the echo of the speaker's own speech. For satisfactory operation of the echo canceller, the length of the filter's impulse response must be greater than the length of the longest echo path.

Having introduced the project in the previous chapter it is time to introduce digital filtering. Adaptive filters are mainly implemented with digital filters, digital filtering therefore has a significant role to play in this project.

1) Introduction to Digital Filters:

The system or network in the case of digital filters are mathematical algorithms, these algorithms operate on digital signals and attempt to modify these signals in the same way analogue filters modify analogue signals. Equation 1.3 defines the operation of linear digital filters [2].

$$y(n) = \sum_{k=0}^{N-1} h(k) - x(n-k) \qquad \textbf{1.3}$$

where ***h(k), k = 0,1…N-1*** are the filter coefficients, ***x(n)*** is the filter input and ***y(n)*** is the filter output. We note at this point that the above Equation represents the convolution of the input signal ***x(n)*** with the filter's impulse response ***h(k)*** to give the filter output ***y(n)***. There will be further discussion of convolution in Section 2.2.

- Automatic updating of frequency Response if a programmable processor is used, which means that adaptive filtering can be implemented more easily.
- The filter output and input can be stored for further use.
- Some characteristics are not possible with analogue filters.
- Can easily take advantage of advances in VLSI technology.
- Performance is not affected by temperature variations or by slight differences in components making it possible to repeat the characteristics from one filter to the next.
- More than one signal can be filtered at a time.
- Precision is only limited by word length.
- It is possible to use digital filters at lower frequencies, which are often found in biomedical applications.

2) Disadvantages of Digital Filtering:

- The speed at which the filter operates may be limited by the speed of the processor or by the number of arithmetic operations involved. This number increases

as the specifications of the filter are tightened.

- Digital filters are subject to round-off noise encountered in computation and if the input signal was sampled to Analogue to Digital Conversion noise.
- The design of digital filters is a far more complex task than designing an analogue filter. Computer aided design techniques in the right hands however do help to overcome this problem. Also once designed the system is usable with little or no modification for other different digital signal processing tasks. [2]

III. ARCHITECTURE DESCRIPTION

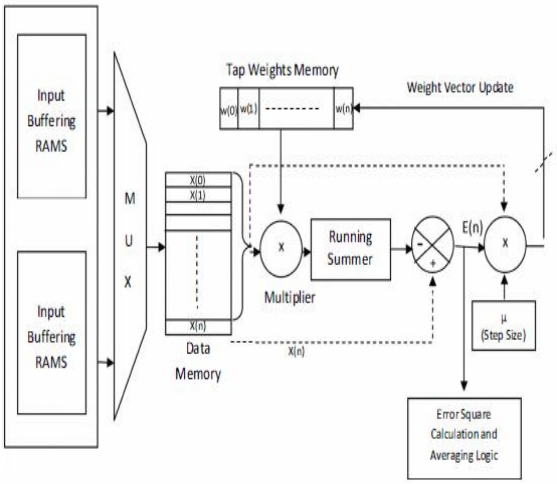


Figure 4: Block Diagram of the block LMS Architecture

A. The Weakness of the LMS Algorithm

The LMS Algorithm has one profound weakness, and it is that computational cost increases to an undesirable level as the length of the impulse response increases. This is mainly due to the fact that the algorithm lies in the time domain, leaving the algorithm at an obvious disadvantage when an impulse response is very long. The computational power required simply becomes too high for efficiency of use. There are block LMS versions of the LMS algorithm, which attempt to compensate for this problem in the time domain, however Frequency domain adaptive filtering holds the key to the solution of the very long impulse response problem.

Disadvantages:

- Convergence rate of LMS is very poor.
- BLMS algorithm provides the fastest convergence rate but highly computational expensive.
- MSE error is very high for any given SNR ratio.

The recent digital transmission systems impose the application of channel equalizers with short training time and high tracking rate. These requirements turn our attention to adaptive algorithms, which converge rapidly. The demand for fast convergence and less MSE level cannot be met by conventional adaptive filtering algorithms such as LMS. The best choice is the Block recursive least squares (BRLS) algorithm. Block Recursive Least Squares algorithms are known to exhibit better performances.

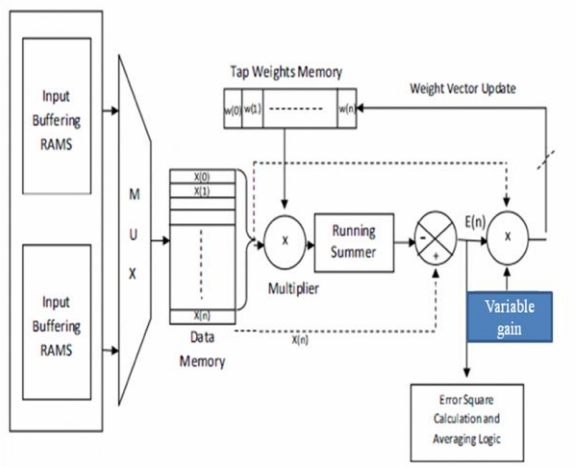


Figure 5. Block Diagram of the Proposed Architecture

The block diagram implementation is as shown in the figure5 The proposed architecture is designed to perform standalone implementation and by adding appropriate buffering blocks the architecture is made capable for real time implementation. The incoming continuous data is stored in input buffering RAMS to provide for the calculations involved until the weight updation. The data is read out alternatively from each of the input RAM blocks and is written into input data memory. Initial tap weights are Stored in tap weight memory block. Both the data and weights are readout of the memory simultaneously and passed to the multiplier which multiplies the data vector and weight vector and passes the data on to the running summer block which adds up the multiplier output .And then the running summer output, equivalent to the FIR filter output, (which ideally requires N multipliers depending on the number of taps which here is reduced to one) is then subtracted from next input sample to estimate the error. The error value obtained is then multiplied with variable gain factor and data vector to calculate the weight updates required reducing the error from each calculation. The weight updates are then added to the previous weight values and written back into the weight vector memory. The updated weights are now ready for next set of data. As mentioned earlier ideally FIR filter structure requires N multiplier depending on the number of weights considered for implementation, but with the architecture proposed. the number of multipliers is reduced to one with the pipelining of the data. The architecture presented consumes minimal hardware making it suitable for FPGA implementation.

Advantages:

- Extremely fast convergence.
- RLS algorithm to achieve the steady state error in very less no of iteration along with mean square error is also less compare to RLS algorithm for different signal to noise ratio.
- Power consumption is reduced along with hardware reduction by reducing number of iteration.

.

B. Performance Measures in Adaptive Systems:

1. Convergence Rate

The convergence rate determines the rate at which the filter converges to its resultant state. Usually a faster

convergence rate is a desired characteristic of an adaptive system. Convergence rate is not, however, independent of all of the other performance characteristics. There will be a tradeoff, in other performance criteria, for an improved convergence rate and there will be a decreased convergence performance for an increase in other performance. For example, if the convergence rate is increased, the stability characteristics will decrease, making the system more likely to diverge instead of converge to the proper solution. Likewise, a decrease in convergence rate can cause the system to become more stable. This shows that the convergence rate can only be considered in relation to the other performance metrics, not by itself with no regards to the rest of the system.

2. Minimum Mean Square Error

The minimum mean square error (MSE) is a metric indicating how well a system can adapt to a given solution. A small minimum MSE is an indication that the adaptive system has accurately modeled, predicted, adapted and/or converged to a solution for the system. A very large MSE usually indicates that the adaptive filter cannot accurately model the given system or the initial state of the adaptive filter is an inadequate starting point to cause the adaptive filter to converge. There are a number of factors which will help to determine the minimum MSE including, but not limited to quantization noise, order of the adaptive system, measurement noise, and error of the gradient due to the finite step size.

3. Computational Complexity

Computational complexity is particularly important in real time adaptive filter applications. When a real time system is being implemented, there are hardware limitations that may affect the performance of the system. A highly complex algorithm will require much greater hardware resources than a simplistic algorithm.

4. Stability

Stability is probably the most important performance measure for the adaptive system. By the nature of the adaptive system, there are very few completely asymptotically stable systems that can be realized. In most cases the systems that are implemented are marginally stable, with the stability determined by the initial conditions, transfer function of the system and the step size of the input.

5. Filter Length

The filter length of the adaptive system is inherently tied to many of the other performance measures. The length of the filter specifies how accurately a given system can be modeled by the adaptive filter. In addition, the filter length affects the convergence rate, by increasing or decreasing computation time, it can affect the stability of the system

IV. SIMULATION REPORTS

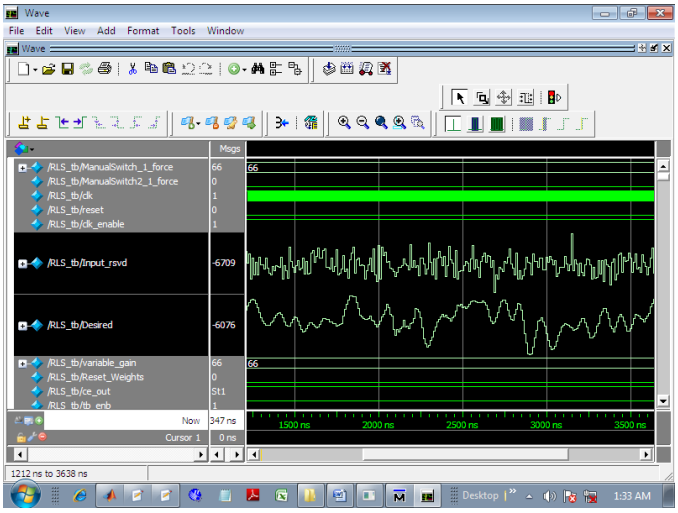


Figure 6. Simulation results

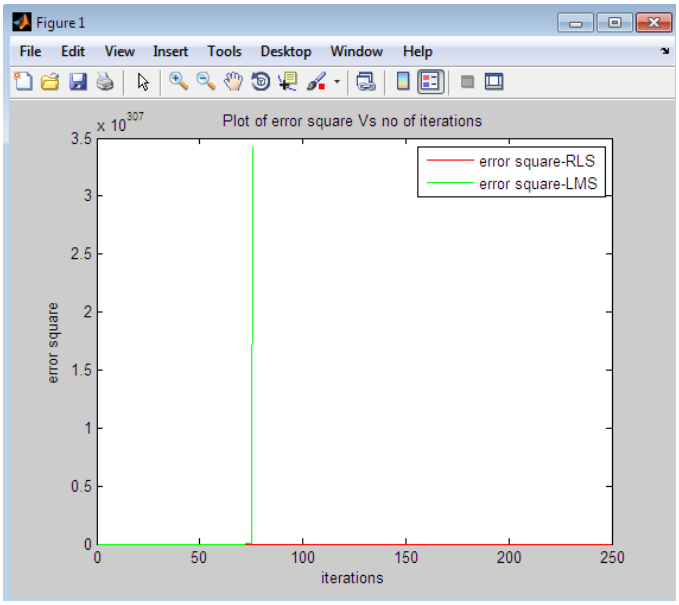


Figure 7. Variation gain Vs step size

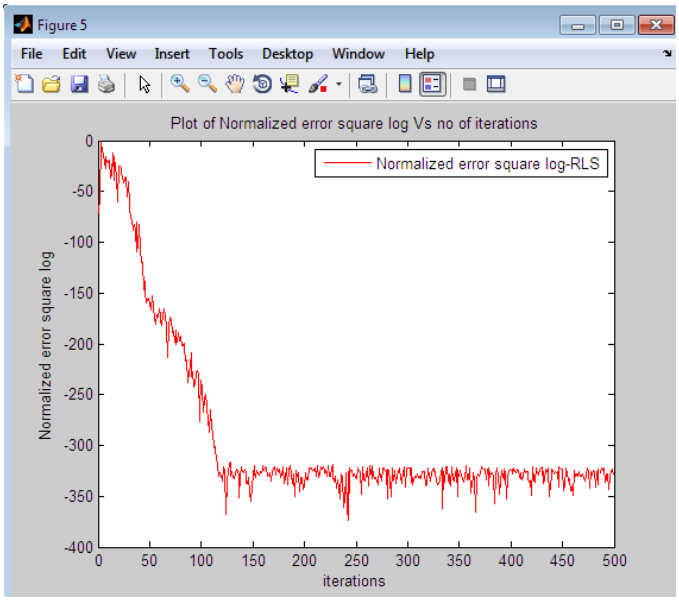
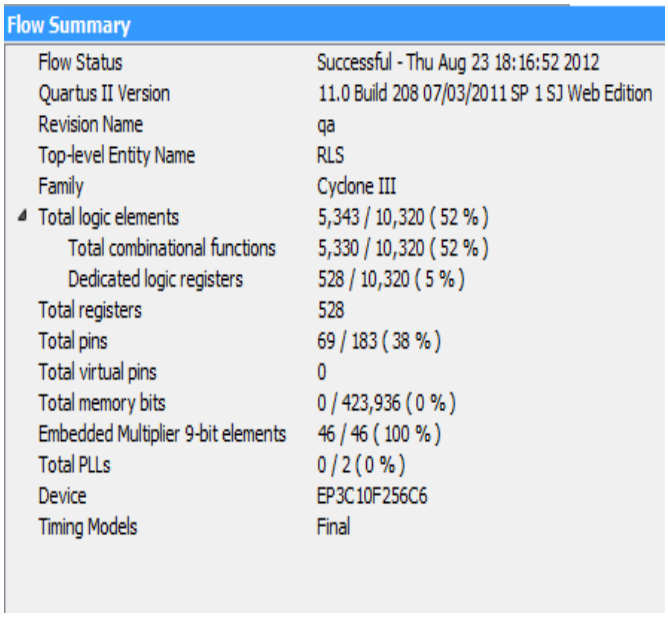


Figure 8.Learning Curves for RLS

V. DEVICE UTILISATION SUMMERY



Flow Summary	
Flow Status	Successful - Thu Aug 23 18:16:52 2012
Quartus II Version	11.0 Build 208 07/03/2011 SP 1 SJ Web Edition
Revision Name	qa
Top-level Entity Name	RLS
Family	Cyclone III
Total logic elements	5,343 / 10,320 (52 %)
Total combinational functions	5,330 / 10,320 (52 %)
Dedicated logic registers	528 / 10,320 (5 %)
Total registers	528
Total pins	69 / 183 (38 %)
Total virtual pins	0
Total memory bits	0 / 423,936 (0 %)
Embedded Multiplier 9-bit elements	46 / 46 (100 %)
Total PLLs	0 / 2 (0 %)
Device	EP3C10F256C6
Timing Models	Final

Figure9. Device utilization report

VI. CONCLUSION

The BRLS algorithm is introduced as the main adaptive algorithm in the time domain and its operation is examined. An alternative representation of signals in the frequency domain is then introduced, which allows the convolution of two signals to be calculated in a much more efficient manner. The cost of transforming the signals to and from the frequency domain must be accounted for however and for short filter impulse responses it is too high to allow frequency domain filtering replace time domain filtering. Substantial savings can be made however as the impulse response increases, an approximate crossover point was portrayed. In this report a possible implementation of the Block RLS in the Verilog HDL programming language is presented and a possible application as an adaptive equalizer explored.

REFERENCES

[1] ”Memory Based Architecture To Implement Simplified Block LMS Algorithm On FPGA”IEEE Transaction by Jayashri R, Chitra H, Kusuma S
[2] “Communication Systems” 4th edition,Simon Haykin
[3] IEEE Communications Magazine, March 1982, Adaptive Equalization
[4] “Digital Signal Processing: A practical Approach” 2nd edition, E.Ifeachor and B.Jervis, Prentice Hall.
[5] T.Yoshida, Y.Liguni, H.maeda "Systolic array implementation of block LMS Algorithm", Proc of lEEE Electronic letters 1998.
[6] IEEE Transactions on Signal Processing, Vol. 39, No. 10, October 1991. On the Complexity of Frequency Domain Adaptive Filtering, Neil K. Jablon
[7] “Adaptive Filter Theory”, Simon Haykin
[8] https://www.ee.calpoly.edu/~jbreiten/C/
[9] http://www.ee.nuigalway.ie/tech/misc.html, Department of Electronic Engineering, NUI, Galway