FPGA based Architectures for High Performance Adaptive FIR Filter Systems

Sufeng Niu[1], Semih Aslan[2] and Jafar Saniie[1]

[1]*Department of Electrical and Computer Engineering*
Illinois Institute of Technology,
Chicago, IL, 60616, U.S.A.

[2]*SMART Lab, Ingram School of Engineering*
Texas State University
San Marcos, TX, 78666, USA

Abstract — **In this paper, we present a high performance adaptive FIR filter hardware architecture. In particular, the RLS (Recursive Least Square) algorithm for adaptive signal processing is explored based on QR decomposition, which is accomplished by using the Givens Rotation algorithm. The Givens Rotation algorithm is implemented using a systolic array and LUT-based Newton's method. This architecture is suitable for high-speed FPGAs or ASIC designs. It also solves the tradeoff between throughput and latency issues. As a case study, this QR design is tested using Xilinx XC5VLX110T FPGA. The findings show that the system is capable of running the QR decomposition at up to 200MHz with 56 clock cycles latency.**

I. INTRODUCTION

Adaptive filtering techniques have applications in different fields, such as wireless communication, noise cancelling techniques, channel estimation and medical signal processing [1]-[3]. However, the implementation of adaptive filtering is very complex [3]. The standard RLS algorithm requires a direct matrix inversion operation which may cause numeric stability problems, and QR decomposition with back substitution method is able to solve this issue. QR Decomposition Recursive Least Square (QRD-RLS) offers the most robust numerical properties and hardware specific accelerator architecture [4]-[9]. For these reasons, QRD-RLS is the implementation that will be discussed later.

In this paper, our goal is to realize a QRD-RLS algorithm using highly efficient hardware architecture. Hardware optimization is discussed for real-time signal processing purposes. Furthermore, we will show the performance and error analysis for the adaptive filter algorithms. The organization of the paper is as follows: Section II provides the background and design issues based on our previous work on fast QR decomposition. Section III proposes a new hardware design and its optimization using a Look Up Table based Newton (LUT-N) method. In Section IV, results and error margins are discussed.

II. QR DECOMPOSITION AND GIVENS ROTATION

A. QR Decomposition and RLS algorithm

The structure of a classical adaptive filtering system is shown in Figure 1. The filter is aimed at reducing noise or any other undesired signals from the input signal. By minimizing the error function, the adaptive algorithm estimates weight adjustments to obtain the FIR filter coefficients. The popular methods for estimation of filter coefficients are the RLS and LMS algorithms. Adaptive filtering using the RLS algorithm performs better compared with the Normalized Least Mean Square (NLMS) algorithm due to faster convergence and smaller final error [2] [3]. However, the complexity of a standard RLS algorithm [4] requires $O(N^2)$ operations per sampling period, including multiplications and divisions, where N is the number of taps in the adaptive filter. Also, numerical overflow limits RLS applications.

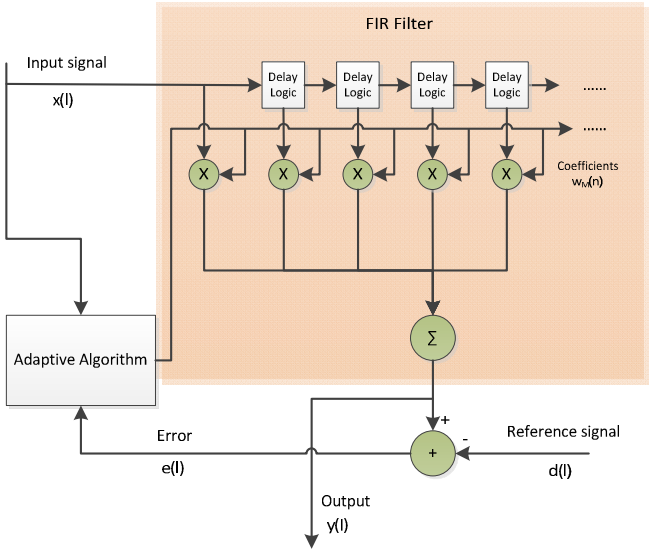

Figure 1. Adaptive filter structure

QR decomposition has better and more robust numeric properties than other methods. Based on our previous work [5], we can map the QR decomposition algorithm to the VLSI and FPGA platform with good performance compared with conventional designs.

Figure 2 illustrates the hardware structure for QR decomposition which contains a triangular array organized by Boundary cells (marked as 00, 11, 22, 33) and Internal cells. Nevertheless, the Boundary cell consists of the most critical computation components: division and square root. This makes the Boundary cell computation heavy and the calculation load imbalance with a long feedback loop. The Internal cell only contains multipliers and adders. Therefore, the Boundary cell is the key component to optimize hardware timing and utilization.

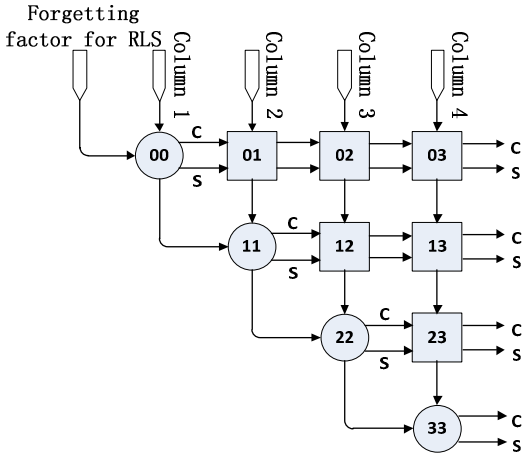

Figure 2. Systolic array for Givens Rotation [5]

III. IMPROVED DESIGN

A. LUT-N method for Boundary cell architecture

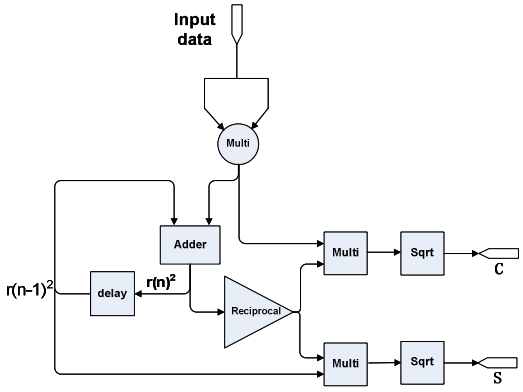

Figure 3. CORDIC-based Boundary cell algorithm [5]

The Boundary cell structure [5] is shown in Figure 3, which states,

$$c = \frac{a}{\sqrt{r_{n-1}^2 + a^2}},\ s = \frac{r_n}{\sqrt{r_{n-1}^2 + a^2}} \tag{3}$$

The term a is the input data, c is cosine term and s is the sine term, r_n and r_{n-1} are the hardware internal nodes. Figure 3 is based on the CORDIC algorithm for square root operation [3]. The reciprocal and square root operations are very expensive in terms of resources. Furthermore, these operations create long latency and, consequently, increase buffer size and latency for timing synchronization.

Our new design for the Boundary cell based on Newton's method and Look Up Table (LUT) ensures lower latency and fewer logic resources. As shown in Figure 4, the new Boundary cell integrates square root and division by using LUT-based Newton's (LUT-N) method. Thus, the design of a new architecture for Newton's method implementation is the key objective of this computation.

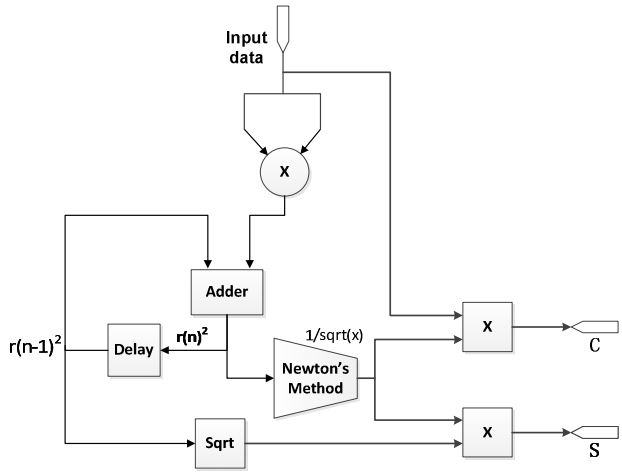

Figure 4. LUT-N Boundary cell algorithm

The Newton's approximation (also known as Newton-Raphson method) uses an iteration approach to find the solution for $y = \frac{1}{\sqrt{x}}$, but two issues limit its hardware implementation: initial guess value for starting the algorithm and uncertain number of iterations for reaching the desired accuracy. The initial guess affects the convergence speed and the number of iterations, while the number of iterations has an effect on the estimation precision and computation time. Our objective is to design a fast and inexpensive approach to approximate the estimates, and then to increase the estimation accuracy iteratively using Newton's method. Thus, we apply LUT to obtain the initial and approximated values for y. We define a function $f(y)$:

$$f(y) = y^2 - \frac{1}{x} \tag{4}$$

And the objective is to obtain the value of y when $f(y) = 0$. This can be done by Newton's iterative method:

$$y_{n+1} = y_n - \frac{f(y_n)}{f'(y_n)} \tag{5}$$

We derive a solution based on Equation 4 and 5:

$$y_{n+1} = y_n(1.5 - \frac{xy_n^2}{2}) \tag{6}$$

where y_1 is the initial guess value and x is the input data. The initial value for y can be approximated using LUT. Using LUT with a uniform interval may cause a large error between the initial value and the actual value. Therefore, a non-uniform LUT is introduced to generate the initial value table. Based on Equation 5, the smaller x value corresponds with the larger y value. Our MATLAB code for table generation is shown below:

```
for i=1:loop_time
    data1=3/4*data_start;
    data2=5/8*data_start;
    data3=data_start/2;

    table=[table,data1,data2,data3];
    data_start=data3;
end
```

The code generates the initial value table. In each loop, the table's step becomes smaller than the last step to achieve more sampled value with larger derivative value of the function. As shown in Figure 5 (discrete samples in blue superimposed on the actual function in red), we keep the approximation table values with higher sampling rate for smaller x values.

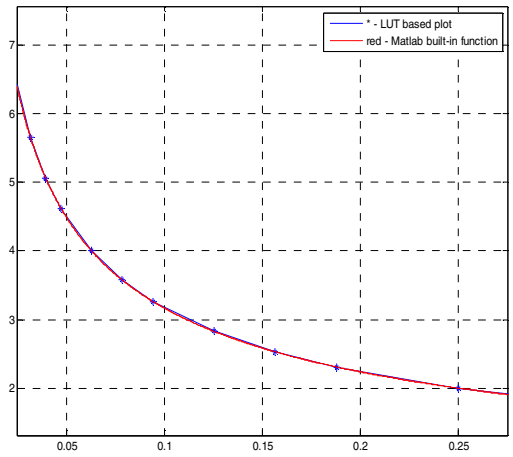

Figure 5. Plot of Initial values for the Look-up table

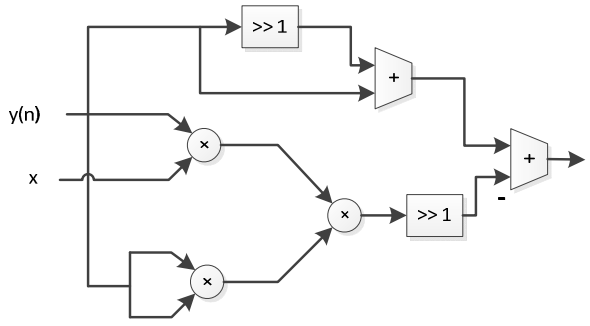

Figure 6. Unfolded hardware architecture for Equation 6

Based on this initial value table, the first iteration has already ensured the error to be under 10^{-3}. Detailed error analysis is discussed in the next section. The corresponding circuit structure for Equation 6 is described in Figure 6. In Figure 6, three multipliers are still required. The folded structure can also be used by time multiplexing. In our implementation, the hardware throughput is accelerated by pipelining to increase clock frequency. This design is implemented on Xilinx Virtex 5 XC5VLX110T FPGA to verify the algorithm and performance. The XC5VLX110T [6] FPGA offers 64 DSP48 hard cores for DSP applications and 69,120 logic cells. Hardware simulation and performance is discussed in the next Section.

IV. Error Analysis and Simulation Results

In the LUT-N method design, a 16-bit fixed-point number representation is used. The input data allows a 1 bit integer part, 14-bit fractional part and 1 bit for sign digit. The output data format is a 1 bit sign digit, 7-bit integer digit and 8-bit fractional digit. Thus, the input data ranges from $6.1035*10^{-5}$ to 4 and the output data ranges from 0.0039 to 129.

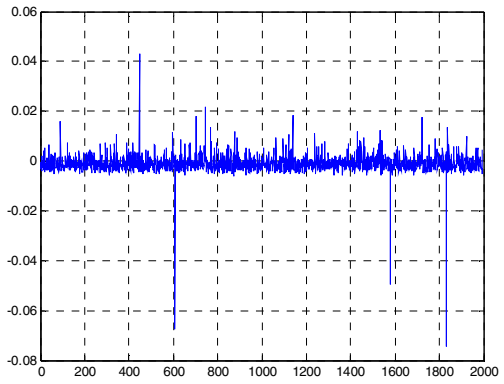

Figure 7. Error between hardware results and MATLAB built-in function

We generated 2,000 random values as the test vector from MATLAB. Figure 7 is the error between hardware results from Modelsim and the MATLAB built-in function using only one iteration for the Newton estimation method. Based on the result shown in Figure 7, the average error for hardware implementation is $9.6042*10^{-4}$. The maximum ratio between error and the actual result is 0.98%, which is acceptable for fast hardware implementation. The differences between MATLAB simulation results and RTL simulation results is due to the fact that the MATLAB code runs in floating-point number representation, while the RTL code runs in fixed-point number representation. If the value of the test vector is too small, the initial guess value is not close to the final solution. Hence, only one iteration in the estimation may cause a large error. That is why some spike values exist in Figure 7. Hence, it is preferable to use multiple iterations for the Newton method, which needs more logic resources and causes higher latency.

The computation time for 4x4 QR Decomposition (QRD) is 0.218ms for PC Xeon 3.30GHz multi-core processor based

on floating-point representation by Matlab tic/toc function, and the FPGA fixed-point implementation needs only 0.858µs for the same 4x4 matrix size with Xilinx Virtex5 device. The LUT-N method's hardware achieves 10 clock cycles latency with 263MHz clock speed running on Xilinx XC5VLX110T FPGA, and Xilinx IP cores can operate at 45 clock cycles with 245MHz clock frequency. Based on the proposed design, the 4 by 4 QRD is able to run up to 200MHz with only 56 clock cycles. The fast inverse square root operation synthesis resources by LUT-N method and CORDIC-based method are compared in Table 1 and Table 2.

TABLE I. CORDIC-BASED SYNTHESIS RESULTS (XUPV5)

Number of Slice Registers	2191 out of 69,120 3.17%
Number of Slice LUTs	997 out of 69,120 1.44%
Number used as Logic	700 out of 17,280 4.05%
Number of DSP48Es	0 out of 64 0%

TABLE II. LUT-N BASED SYNTHESIS RESULTS (XUPV5)

Number of Slice Registers	197 out of 69,120 0.29%
Number of Slice LUTs	189 out of 69,120 0.27%
Number used as Logic	99 out of 17,280 0.57%
Number of DSP48Es	3 out of 64 4.68%

Moreover, Xilinx IP core implementation of the CORDIC method requires 10 times more resources compared to our proposed improved design. The usage of three DSP48 slices in LUT-N design method is not a critical issue since the latest FPGA devices provide a very large number of built-in DSP slices.

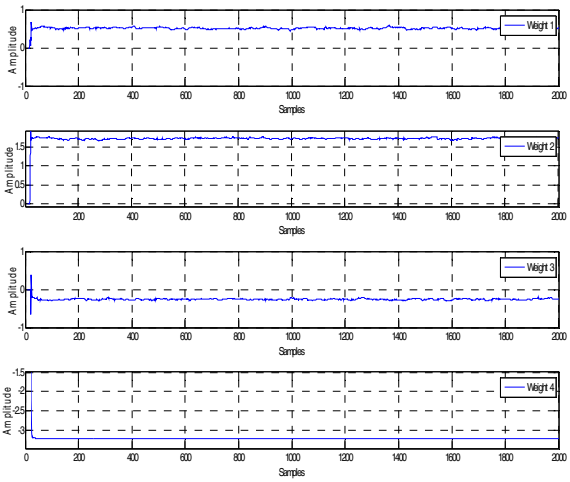

Figure 8. RLS weights adjustments based on QR factorization

We import the QR hardware module described above into the RLS algorithm implementation (see Figure 1) into a MATLAB Simulink tool. This system identification model is tested and verified using the following 4-tap FIR filter in Equation 7:

$$y(n) = 0.492x(n) + 1.6732x(n-1) - 0.242x(n-2) - 3.12x(n-3) \quad (7)$$

After 2000 clock periods, the weights extraction (i.e., the FIR coefficients) is achieved by the back substitution method. The estimated FIR filter coefficients are shown in Figure 8. Error output between the desired signal and the esimated signal is less than 0.025% as shown in Figure 9. The QRD-RLS simulation verifies the QR factorization implementation and also yields a good estimation performance.

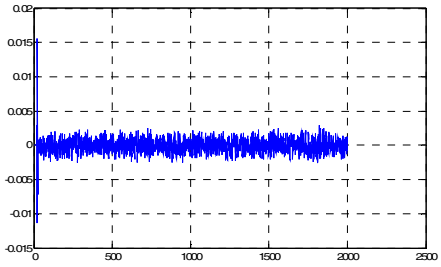

Figure 9. RLS error between desired signal and input signal

V. CONCLUSION AND FUTURE WORK

In this project, an improved Boundary cell architecture based on the LUT-N method is introduced for the QRD-RLS algorithm based on hardware implementation. The optimized hardware allows the LUT-N algorithm running at higher throughput with lower latency compared with the CORDIC based algorithm. The error of the LUT-N method is reasonable and acceptable for high-speed adaptive signal processing.

REFERENCES

[1] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice Hall, 1985.

[2] M. M. Sheikh Algunaidi, M. A. Mohd Ali, Md. Fokhrul Islam., " Comparative Analysis of Fetal Electrocardiogram (ECG) Extraction Techniques using System Simulation," *International Journal of the Physical Sciences*, vol.6(21), pp. 4952-4959, 2011.

[3] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall. 1985.

[4] Cioffi, J.M., "The Fast Adaptive ROTOR's RLS Algorithm," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.38, no.4, pp.631-653, Apr 1990.

[5] Aslan, S. Niu, S. Saniie, J. , "FPGA Implementation of Fast QR Decomposition Based on Givens Rotation," *Circuits and Systems (MWSCAS), 2012 IEEE 55th International Midwest Symposium on* , vol., no., pp.470-473, 5-8 Aug. 2012.

[6] Xilinx, "Virtex-5 FPGA Data Sheet," 2012. [Online]. Available: http://www.xilinx.com/.

[7] Dongdong Chen; Sima, M.; , "Fixed-Point CORDIC-Based QR Decomposition by Givens Rotations on FPGA," *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on* , vol., no., pp.327-332, Nov. 30 2011-Dec. 2 2011.

[8] Karkooti, M.; Cavallaro, J.R.; Dick, C.; , "FPGA Implementation of Matrix Inversion Using QRD-RLS Algorithm," *Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on* , vol., no., pp. 1625- 1629, October 28 - November 1, 2005.

[9] W. M. Gentleman and H. T. Kung, "Matrix Triangularization by Systolic Arrays," in *Proceedings of the Society of PhotoOptical Instrumentation Engineers Conference Series*, 1981, vol. 298, pp. 19-26.