SRTTU

Journal of Electrical and Computer Engineering Innovations

JECEI, Vol. 7, No. 1, 2019

Low Computational Complexity and High Computational Speed in Leading DCD ERLS Algorithm

Fatemeh Abdi[1], Parviz Amiri[1,*], and Mohammad Hossein Refan[1]

[1]Faculty of Electrical Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran.

[*]Corresponding Author's Information: pamiri@sru.ac.ir

ARTICLE INFO

ARTICLE HISTORY:
Received 14 July 2019
Revised 7 September 2019
Accepted 15 September 2019

KEYWORDS:
Exponentially weighted Recursive least squares (ERLS)
Dichotomous coordinate descent (DCD)
Variable forgetting factor (VFF)
Field-programmable gate array (FPGA)

ABSTRACT

Adaptive algorithm adjusts the system coefficients based on the measured data. This paper presents a dichotomous coordinate descent method to reduce the computational complexity and to improve the tracking ability based on the variable forgetting factor. Vedic mathematics is used to implement the multiplier and the divider operations in the VFF equations. The proposed method decreases the area and increases the computation speed. The linear exponentially weighted recursive least squares as the main algorithm is implemented in many applications such as the adaptive controller, the system identification, active noise cancellation techniques, and etc. The DCD method calculates the inverse matrix in the ERLS algorithm and decreases the resources used in the field-programmable gate array, also the designer can use the cheaper FPGA board to implement the adaptive algorithm because the method doesn't need lots of resources. The proposed method leads to implementing complex algorithms with simple structures and high technology. The proposed method is implemented with ISE software on the Spartan 6 Xilinx board. The proposed algorithm calculates the multiplication result with less than 15ns time and reduces the used FPGA resources to lower than 20% as compared with the classic RLS.

1. Introduction

A linear least-squares (LS) is one of the popular methods in digital signal processing (DSP) applications. The LS applications include adaptive antenna arrays [1], active noise cancelation [2], multiuser detection [3], system identification [4], [5], micro-phonics effect, particle accelerator [6] and the controllers of power converter [7]. The system parameters and ambient conditions will be change during signal processing, and the adaptive algorithm adapts the system coefficients based on the changes which means a compensation on the system output. Adaptive algorithms have a wide range of applications at industries such as control, communications [8], radar and sonar signal processing [9], interference cancellation [10], and biomedical engineering [11]. The main specification of application is a filtering process that leads to creating a matching between the input signal and the desired response. The filter coefficients are updated based on the measured data in the system and they are applied as an input signal to the adaptive algorithm. By this way, the difference between the filter output and the desired response will be minimized in an either statistical or deterministic sense.

Least mean squares (LMS) and recursive least squares (RLS) algorithms are common algorithms which adjust and measure the system parameters according to the changes which can happen in the system. Important feathers of the LMS algorithm are simplicity and robustness in digital signal processing.

The RLS filter as a powerful algorithm can be implemented in applications such as the adaptive filters, prediction algorithm, controller, and system identification. Despite the convergence rate of the RLS algorithm is faster than the LMS algorithm, the computational complexity of the RLS algorithm is higher than the LMS algorithm because the RLS algorithm must calculate an inverse matrix [12], [13].

The new methods provide an optimal performance in solving the system equations. But some methods include complicated operations, so designers try to suggest different sub-optimal algorithms. There are methods such as the direct and iteration method which calculate the inverse matrix. These methods include complex operations such as multiplication, division, and square-rooting, which lead to high computational complexity and hence, the designer needs expensive hardware resources [14].

Using the direct methods, such as Gaussian elimination, Cholesky decomposition and QRD, one can obtain a high accuracy in the solutions. The main idea of the direct method is to reduce the system equations to an upper triangular or a lower triangular form. But these methods have complex operations such as multiplication, division, and square-rooting. These operations lead to high computational complexity and therefore, the system may need expensive hardware resources [15]. The iterative methods are suitable for a large or sparse system. These methods provide higher convergence rate and optimal performance and have some advantages compared with the direct methods. As they need less memory, they are faster and provide simpler solutions for complex structures [16].

There are two types of iterative methods: non-stationary iterative methods, such as the steepest descent and the CG and the stationary iterative methods, such as the Jacobi, Gauss-Seidel and coordinated descent (CD) algorithms. The stationary iterative methods have less computational complexity than non-stationary iterative methods [17].

The iterative methods provide higher convergence rate than the direct methods, also, they need less memory and simpler design than direct methods [18].

The dichotomous coordinate descent (DCD) algorithm is designed according to the CD iterative method. It requires no multiplication, division or square rooting operations, just uses additions and subtractions. Therefore, the hardware implementation of the DCD algorithm is optimal in real-time applications [16].

2. Hardware Description of the ERLS Algorithm

The RLS classic algorithm requires about $4N^2$ multiplications and $3N^2$ additions/subtractions. An ERLS algorithm can identify the system parameters in dynamic systems [1]. The ERLS algorithm is shown in Table 1, the numbers of the system coefficients are shown with the symbol N. Pm is the number of multipliers, and Pa is the number of adders in the DCD algorithm which are employed to calculate the inverse matrix. The algorithm consists of an N×N symmetric positive matrix (R), and N×1 two vectors ($\hat{h}$ and β0). The R matrix is defined as the correlation matrix of the reference signal X, and the β0 vector is the cross-correlation vector between the reference signal and the desired response [19]-[22].

An adaptive algorithm should estimate and identify an optimal vector ($\hat{h}(i)$) that leads to the error signal goes to zero and the zero value is the optimal value in the error signal [8] a regularization matrix is defined with Π symbol and a forgetting factor can be selected between zero and unity $0 < \lambda < 1$.

TABLE 1
ERLS ALGORITHM [19]

Step	Equation	×	+
	Initialization: $\hat{h}(0)=0$, $r(0)=0$, $R(0)=\Pi$		
	For i=1, 2…		
1	$\mathbf{R}(i) = \lambda\mathbf{R}(i-1) + \mathbf{X}(i)X^T(i)$	N(N+1)/2	N(N+1)
2	$y(i) = X^T(i)\hat{h}(i-1)$	N	N-1
3	$e(i) = d(i) - y(i)$	0	1
4	$\beta_0(i) = \lambda r(i-1) + e(i)\mathbf{X}(i)$	N	2N
5	$R(i)\Delta\hat{h}(i)=\beta_0(i) \rightarrow \Delta\hat{h}(i), r(i)$	Pm	Pa
6	$\hat{h}(i) = \hat{h}(i-1) + \Delta\hat{h}(i)$	0	N
Total	Multiplies = $(N^2+5N)/2+Pm$; Adds $\leq N^2+4N+Pa$		

The regularization matrix is a diagonal matrix $\Pi=\eta I$, a small positive number is assigned as the regularization parameter $\eta > 0$ and I is the N×N identity matrix [1], [8].

$$\mathbf{R}(i)\hat{h}(i) = \boldsymbol{\beta}(i) \tag{1}$$

The **R** matrix is an auto-correlation matrix with size N×N, and the β vector is the cross-correlation vector with length N. The coefficient vector h(i) can be calculated according to the normal Equation (1), and the algorithm identifies and estimates the upper triangle part in **R**(i) and this method leads to reducing the computational complexity. The forgetting factor has effects on calculating the **R** and β elements.

The proposed method has been implemented to calculate the linear equations in the system based on leading DCD of serial FPGA, on a Xilinx Spartan 6. The λ value is a positive constant factor known as the forgetting factor. The $\mathbf{X}(i)$ vector is the input signal with adaptive filer and the d(i) vector is the desired value in the adaptive algorithm [23]-[25].

$$\mathbf{R}(i) = \lambda\mathbf{R}(i-1) + \mathbf{X}(i)\mathbf{X}^{T}(i) \tag{2}$$

$$\boldsymbol{\beta}(i) = \lambda\boldsymbol{\beta}(i-1) + d(i)\mathbf{X}(i) \tag{3}$$

From (2-3) equations, we obtain:

$$\Delta\mathbf{R}(i) = \mathbf{R}(i) - \mathbf{R}(i-1) \tag{4}$$

$$\Delta\boldsymbol{\beta}(i) = \boldsymbol{\beta}(i) - \boldsymbol{\beta}(i-1) \tag{5}$$

$$\Delta\mathbf{R}(i) = (\lambda-1)\mathbf{R}(i-1) + \mathbf{X}(i)\mathbf{X}^{T}(i) \tag{6}$$

$$\Delta\boldsymbol{\beta}(i) = (\lambda-1)\boldsymbol{\beta}(i-1) + d(i)\mathbf{X}(i) \tag{7}$$

By using Equations (4) and (5), we obtain:

$$\Delta R(i)\hat{h}(i-1) = (\lambda-1)[\beta(i-1) - r(i-1)] + X(i)y(i) \tag{8}$$

The adaptive filter output is defined with y(i),

$$y(i) = X^{T}(i)\hat{h}(i-1) \tag{9}$$

$$\beta_0(i) = \lambda r(i-1) + e(i)\mathbf{X}(i) \tag{10}$$

The error signal (e(i)) is calculated according to below equation:

$$e(i) = d(i) - y(i) \tag{11}$$

The $\mathbf{X}(i)$ data is an input signal to the FPGA board and the d(i) is the desired data. R RAM saves the values of R matrix, β RAM and the x register save the values of the β and x vectors respectively. The $\hat{h}$ RAM stores the values of the coefficients. According to the steps 1 to 4 in Table 1, The $\mathbf{R}(i)$ matrix is updated and the $\boldsymbol{\beta}_0(i)$ vector is calculated in the ERLS block. Step 2 computes the output of the filter (y(i)) [16], [20].

The ERLS algorithm is shown in Fig.1. Firstly, the new values of the x vector will be read from the x register. The $\mathbf{X}(i)$ elements are read from the transpose block which generate $\mathbf{X}^{T}(i)$, the multiplier block identifies upper triangular elements of the vector $\mathbf{X}(i)\mathbf{X}^{T}(i)$. The R block reads the $\mathbf{R}(i-1)$ elements from the R RAM memory and the new value will be shifted based on the λ value which is a positive integer number, also, the multiplier block is replaced with the shifter block and Vedic operations. The R calculator block computes the $\mathbf{R}(i)$ elements and the new value will be written into the R RAM memory [13], [5].

The proposed method reduces the computational complexity because it needs no multiplier and divider operations. The elements addresses of the r(i−1) vector are written into the β RAM, those addresses are calculated based on the RAM reader block. The x RAM reads the $\mathbf{X}(i)$ elements and the calculated value is multiplied by the error signal. The β block computes the new β value and the new value is then written into the β RAM based on the addresses which are determined with the RAM reader block [16], [21].

3. Hardware Description of Leading DCD Algorithm on FPGA

In many applications, a hardware implementation is one of the significant requirements in real-time. But the computational complexity increases the execution time in microprocessors [22]. For solving the mentioned problems, a method is presented with less computational complexity, based on a leading DCD-exponentially weighted recursive least squares algorithm.

The DCD algorithm is an effective and optimal method [21], [23] and has no division nor multiplication operations [18]. The proposed method estimates the parameters faster and more accurate than other existing methods. It offers an efficient hardware implementation [24].

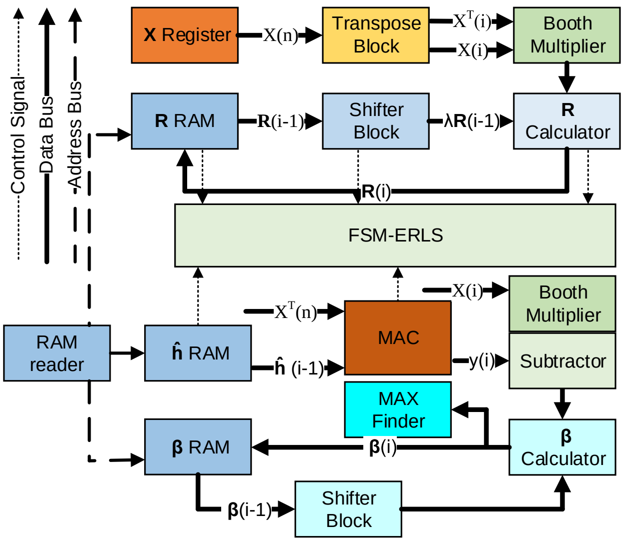

Figure 1: The main diagram of the ERLS algorithm.

The DCD block estimates the Equation (1) and calculates the r(i) residual vector based on the equations in step 5 (Table 1). Elements of the $\mathbf{R}(i)$ matrix and r(i), $\boldsymbol{\beta}_0(i)$, and $\hat{\mathbf{h}}(i)$ vectors have been determined by 32-bit fixed-point accuracy.

The leading DCD serial algorithm requires the smallest FPGA resources. This method is suitable for applications which has many parallel steps. It provides the fastest convergence rate in the design with parallel structures.

The DCD hardware structure is shown in Fig. 2. The $\hat{h}(i)$ elements are updated in the h updater block according to step 5 (Table 2). The β Updater updates the r(i) residual vector. The "leading" component will be computed with the Max finder block according to step 1 in Table 2.

The C block computes the C value according to the third and fourth steps in Table 2. The Finite State Machine (FSM DCD) leads to pipeline-based running all of the blocks. The Max finder block and the r block will execute all steps synchronously [21]. The C block read the $r_n(i)$ values from the β RAM memory and the $\mathbf{R}_{n,n}(i)$ value from the R RAM memory. The structure of the r Updater and the ĥ updater blocks are shown in Figs. 3 and 4 [24], [25].

Fig. 4 calculates the new value of the **r** vector based on step 5 in Table 2. This figure needs two addresses to read the **R**:,n value and the previous value of the **r** vector. Those addresses will be read through RAM reader block. After reading the vectors, the previous value of **r** vector should be as an input signal to the shifter block. This vector is multiplied in $2^{\Delta m}$, and the shifter block will generate the result without the multiplier operation. The next block will add or subtract new vectors according to the sign of (rn) value.

Calculator block C gets the index of arg maxp=1... N {|rp|} from max finder block and computes the C value. The ĥn value is read from ĥ RAM memory based on the ĥ update block which computes the new value and writes it in the ĥ RAM. The residual vector r read from β RAM, is shifted, added and subtracted each element of R parameters. These steps are pipelined to achieve an effective time in updating.

TABLE 2
REAL-VALUED LEADING DCD ALGORITHM FOR SERIAL FPGA IMPLEMENTATION [20]

State	Operation
0	Initialization: $\hat{\mathbf{h}} = 0$, $\mathbf{r} = \beta$, m = Mb, k = 0, Δm = 0
1	n = arg maxp=1,…,N {| rp |}
2	If m = 0, algorithm stops Else, m = m-1 , α = 2m , Δm = Δm+1
3	$c = R_{n,n} - \| r_n \| 2^{\Delta m+1}$
4	If c < 0, go to state 5 Else, go to state 2
5	ĥn = ĥn +sign(rn)α $\mathbf{r} = \mathbf{r} \times 2^{\Delta m}$ – sign (rn) **R**:,n n = arg max p=1,…,N {| rp |} Δm = 0, k = k+1 If k = Nu , algorithm stops; else, go to state 3

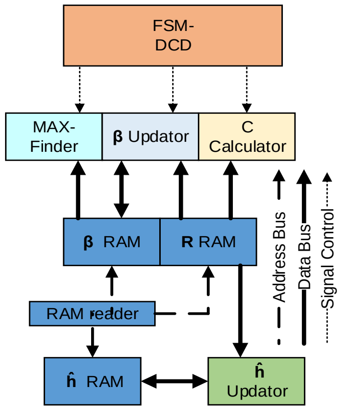

Figure 2: The DCD algorithm architecture.

4. IMPROVED VARIABLE FORGETTING FACTOR USING THE VEDIC MATHEMATICS

The proposed method is based on an improved Variable forgetting factor and DCD-Exponentially Weighted RLS algorithm (IVFF-Leading DCD-ERLS) and it is an optimal method in system identification and the controllers. The forgetting factor λ has an important role in the features of the LS algorithms such as convergence rate, tracking ability, and stability. If the λ value is near to one, the algorithm has proper stability and fast convergence rate while has low tracking ability. The improved IVFF-DCD ERLS algorithms have been developed to find the desired performance and a good tradeoff between parameters.

The variable forgetting factor is calculated according to the below equation:

$$\lambda_n = \min\left\{ \frac{\sigma_{qn}\sigma_{vn}}{\varepsilon + |\sigma_{en} - \sigma_{vn}|}, \lambda_{max} \right\} \qquad (12)$$

The λ_n value will be replaced by the λ value in Table 1, the ERLS algorithm will be executed based on the variable forgetting factor. The new structure will lead to speeding the tracking ability. Where $E\{e^2_n\}=\sigma^2_{en}$ is the power of the prior error, the power of the system noise is defined with σ^2_{vn}, which is the PRBS signal that is the system noise in this application:

$$q_n = X_n^H R_{n-1}^{-1} X_n \text{ and } E\{q_n^2\} = \sigma^2_{eqn}$$

The ε > 0 value is a very small constant to prevent the denominator to become zero and λ_{max} number is smaller than one. $E\{q_n^2\}$ denotes the statistical expectation operator [26]. The multiplier and divider are main operations in digital signal processing. Their parameters such as area and delay have important role in the design. Equation (12) needs the multiplier and the divider operations to calculate the optimal forgetting factor. For this reason, the Vedic multiplier (32-bits) is designed using the 16 bits Vedic multiplier and high-speed carry-save adder. The carry-save adder is used to improve the delay and operation frequency [27].

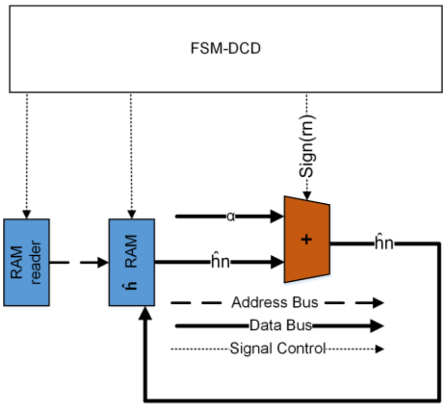

Figure 3: The ĥ updater block.

The Vedic multiplier leads to fast performance in the current processors. The proposed multiplier will start to calculate the result with small size from the input number (multiplicand size of (2 x 2)). Large bits (N x N) break into smaller bits (N/2 = n). In this method, the input bits will be divided to n/2 and this method will be repeated until the number bits reach to 2 x 2. The 24 bit or 30 bit Vedic multiplier using Urdhva-Tiryakabhyam method is implemented to multiply the mantissa part in the float point using the IEEE754 standard. The Vedic multiplier improves some parameters such as memory and area in comparing with conventional multiplier [28], [29]. When designers need more accuracy, they can increase the bits in mantissa or fractional numbers. Fig.5 shows the 32 bit Vedic multiplier.

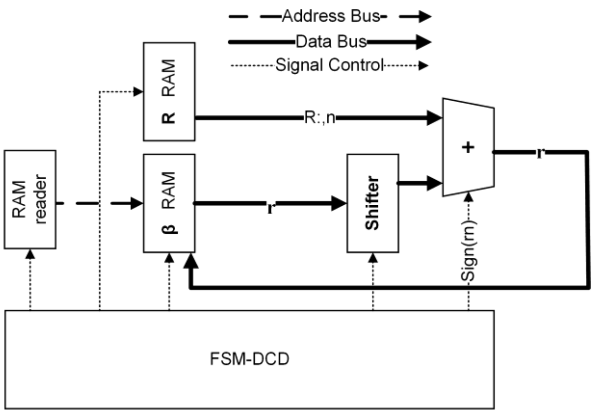

Figure 4: The r updater block.

The structure below shows how the Vedic multiplier operates:

A2 A1 A0

B2 B1 B0

M0 = A0B0

K0M1 = A1B0 + A0B1

K2K3M2 = A2B0 + A1B1 + A0B2 + K0

K4K5M3 = A1B2 + A2B1 + K2 + K3

M5M4 = A2B2

This method needs logical gates and Vedic mathematics to calculate the M0-M5 intermediate signals. A2A1A0 and B2B1B0 are inputs and M0-M5 is the multiplication results. The Vedic multiplier with 3-bits is explained to design 16 bits Vedic multiplier in the figure below [25]. The proposed algorithm will identify and calculate the system coefficients and forgetting factor based on 1 and 4 steps in Table 1 and the variable forgetting factor traces the system changes better with the Vedic structure [30-32].

5. Results

Here, the results are presented using the computer simulation in the ISE software. FSM DCD block generates the control signals and applies them to other blocks. This block has some input signals such as Done-C and stop-xXT. When Done-C is equal to 1, it means that the new value of C signal is calculated and the algorithm will be transferred to a new step. The En-Write signals will enable the RAM memories to write the new value according to the addresses which RAM-reader block generates them. When the algorithm calculates the xXT result, the stop-xXT signal goes to unity. The output signals are defined to enable the write-signal in RAMs and some blocks such as C calculator and MAX finder [33]. Fig. 6 shows the obtained result in the FSM DCD block. Fig. 7 shows the Max Finder result. Four numbers are defined in the residual vector and this block will execute step 1 in Table 2. Table 3 shows the used FPGA resources for MAX finder block. The Max finder will find the maximum value and its index between the values in the residual vector r.

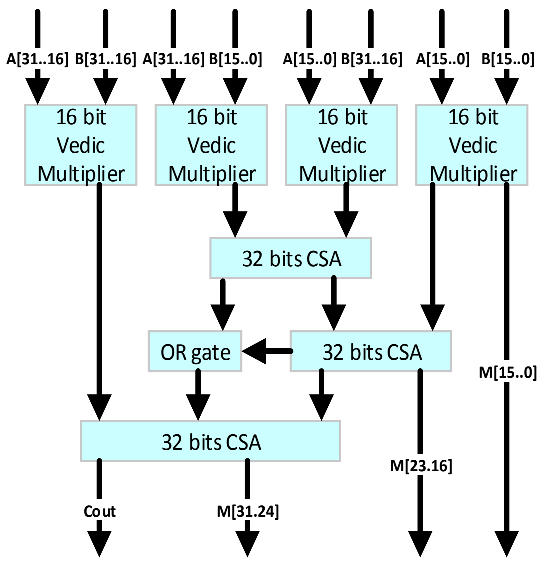

Figure 5: The 32 bits Vedic multiplier.

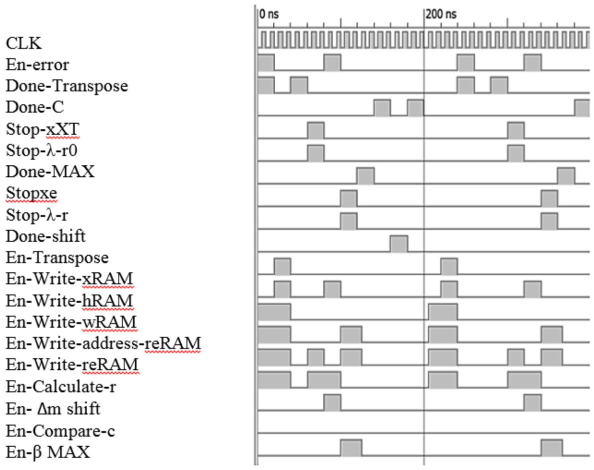

Figure 6: The FSM DCD result.

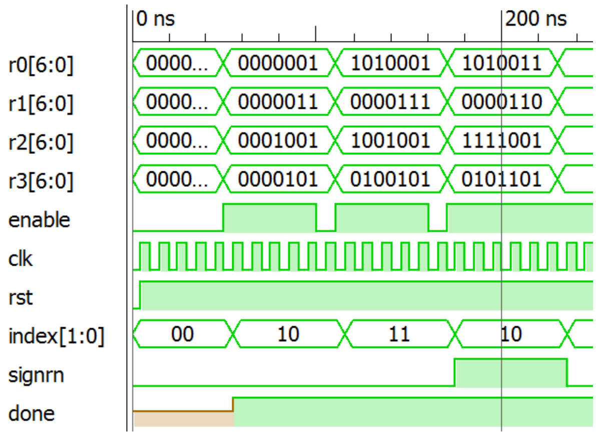

Figure 7: The result of Max finder block.

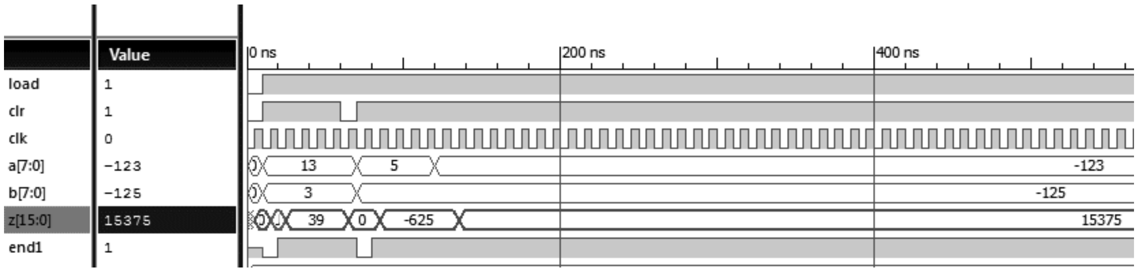

Figure 8: The result of Vedic multiplier.

TABLE 3
FPGA RESOURCES FOR MAX FINDER BLOCK

Resource	Utilization
Number of Slice Registers	1%
Number of Slice LUTs	1%
Number of BUFG/BUFGMUXs	6%
Number of OLOGIC2/OSERDES2s	5%

TABLE 4
USED FPGA RESOURCES AT Spartan 6

Algorithm	IVFF-Leading DCD-ERLS	Classic RLS
Resource	Utilization	
Slice Register	15%	38%
Slice LUTs	54%	62%
Fully used LUT-FF pairs	18%	20%
BUFG/BUFGCT RLS	18%	38%
DSP48A1S	37%	52%

The index of the maximum value is used in the next step to calculate the coefficient and the residual vectors. The Max finder should find the maximum value without any attention to sign bit. In Fig. 7, the index is the location of the biggest value and the signrn is the sign of the biggest value.

Fig. 8 shows the result of the Vedic multiplier. A and b input values are two signed numbers, z number is the result of multiplier and end1 signal shows the time when the output is valid and stable. Clr signal is a clear signal and low active. When Clr signal goes to zero, the algorithm deletes the previous value. The multiplier ensures the output value at a high speed. The used resources in FPGA are shown in Table 4. The proposed algorithm has fast tracing ability and reduces the used resources in FPGA. Fully used LUT-FF (Look up table-Flip Flop) pairs mean that how well your design uses the slice components. We have also used the adjacent flip-flop within that slice for every LUT in the design. Normally the designs have some logics that only use the LUT pair. The proposed method (Improved IVFF leading -DCD-ERLS) includes more operations because it has faster convergence rate and good tracing ability [34].

The LUTs are organized in Slices which mean that those elements share connections in order to utilize fast carry chain. LUT is the truth table and this truth table effectively defines how your combinatorial logic behaves. DSP48A1S is the digital signal processing unit in FPGA and includes the adder, multiplier, register, logic, and ALU. The proposed method needs FPGA with less resources, such as DSP48A1S, slice LUT and etc.

6. Conclusion

The leading DCD-ERLS algorithm is improved based on variable forgetting factor with Vedic mathematics to calculate the multiplier and divider operations in the relevant equations. This structure leads to good tracking ability, less delay and area, and faster convergence rate. The IVFF-Leading DCD ERLS algorithm calculates the inverse matrix according to the leading DCD algorithm and it does not need multiplier and divider operations. This structure just uses simple operations and reduces the used FPGA resources such as register less than 15% and the DSP48A1S more than 37%. The Vedic multiplier is done with a lower time about 15 ns. The proposed method reduces the cost of the final product because the producer can implement the proposed method with cheaper FPGA.

References

[1] S. Haykin, *Adaptive Filter Theory*, 4nd ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[2] M. Ned, T. M. Undeland, and W. P. Robbins, *Power Electronics: Converters, Applications, and Design*, 3rd ed., John Wiley & Sons, Inc., 2003.

[3] F. Abdi and P. Amiri, “Design and implementation of adaptive FxLMS on FPGA for online active noise cancellation,” *Journal of the Chinese Institute of Engineers*, vol. 41, no. 2, pp. 132-140, 2018.

[4] M. Algreer, M. Armstrong, and D. Giaouris, “Active on-line system identification of switch mode dc-dc power converter based on efficient recursive DCD-IIR adaptive filter,” *IEEE Transactions on Power Electronics*, vol. 27, pp. 4425-4435, Nov. 2012.

[5] P. G. Vasundhara, and N. Puhan, “An improved block adaptive system for effective feedback cancellation in hearing aids,” *Digital Signal Processing*, vol. 48, pp. 216–225, 2015.

[6] R. Rybaniec et al., “FPGA based RF and piezo controllers for SRF cavities in CW mode,” in *Proc. 2016 IEEE-NPSS Real Time Conference (RT)*, pp. 1-2, 2016.

[7] M. Shirazi, R. Zane, and D. Maksimovic, “An auto-tuning digital controller for DC-DC power converters based on online frequency-response measurement,” *IEEE Trans. Power Electron.*, vol. 24, no. 11, pp. 2578–2588, Nov. 2009.

[8] P. B. Bhat, V. K. Prasanna, and C. S. Raghavendra, “Adaptive communication algorithms for distributed heterogeneous systems,” in *Proc. The Seventh International Symposium on High Performance Distributed Computing*, pp. 310-321, 1998.

[9] Y. Guo, H. Xiao, and Qiang Fu, “Least square support vector data description for HRRP-based radar target recognition,” *Applied Intelligence*, vol. 46, pp. 365-372, 2016.

[10] P. K. Sethy and S. Bhattacharya, “Interference cancellation in adaptive filtering through LMS algorithm using TMS320C6713DSK,” *International Journal of Electronics and Communication Engineering*, vol. 5, no. 2, pp. 113-124, 2012.

[11] R. Qureshi, S. A. R. Rizvi, S. H. Musavi, S. Khan, and K. Khurshid, “Performance analysis of adaptive algorithms for removal of low frequency noise from ECG signal,” presented at the 2017 International Conference on Innovations in Electrical Engineering and Computational Technologies (ICIEECT), Karachi, Pakistan, 2017.

[12] M. M. Peretz and S. Ben-Yaakov, “Time-domain identification of PWM converters for digital controllers design,” in *Proc. IEEE Power Electronics Specialists Conference,* pp. 809–813, 2007.

[13] Y. V. Zakharov, B. Weaver, and T. C. Tozer, “Novel signal processing technique for real-time solution of the least squares problem," in *Proc. 2nd International Workshop on Signal Processing for Wireless Communications*, pp. 155-159, 2004.

[14] G. H. Golub and C. F. Van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore, 3rd edition, 1996.

[15] S. D. Muruganathan and A. B. Sesay, “A QRD-RLS-based predistortion scheme for high-power amplifier linearization,” *IEEE Trans. Circuits and Systems - II: Express Briefs*, vol. 53, no. 10, pp. 1108–1112, 2006.

[16] Y. V. Zakharov and T. C. Tozer, “Multiplication-free iterative algorithm for LS problem,” *Electronics Letters*, vol. 40, no. 9, pp. 567–569, April, 2004.

[17] Z. Liu, J. V. McCanny, G. Lightbody, and R. L. Walke, “Generic SoC QR array processor for adaptive beam forning,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 50, no 4, pp. 169–175, April 2003.

[18] D. S. Watkins, Fundamentals of matrix computations, Hoboken, N. J., Wiley, 2002.

[19] L. Jie, "DCD algorithm: architectures, FPGA implementations and applications" Ph.D. dissertation, University of York Nov 2008.

[20] Y. Zakharov, G. White, and L. Jie, “Fast RLS algorithm using dichotomous coordinate descent iterations,” presented at the 2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 2007.

[21] G. K. Boray and M. D. Srinath, “Conjugate gradient techniques for adaptive filtering,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 39, no. 1, pp. 1–10, 1992.

[22] J. Morroni, R. Zane, and D. Maksimovic, “An online stability margin monitor for digitally controlled switched-mode power supplies,” *IEEE Transactions on Power Electronics*, vol. 24, no. 11, pp. 2639-2648, 2009.

[23] L. Jie, Y. V. Zakharov, and B. Weaver, “Architecture and FPGA design of dichotomous coordinate descent algorithms,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 11, pp. 2425-2438, 2009.

[24] P. S. Chang and A. N. Willson Jr., “Analysis of conjugate gradient algorithms for adaptive filtering,” *IEEE Transactions on Signal Processing*, vol. 48, no. 2, pp. 409–418, 2000.

[25] A. S. K. Vamsi and S. R. Ramesh, “An efficient design of 16 bit mac unit using vedic mathematics,” presented at the 2019 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2019.

[26] L. Zhang, Y. Cai, ChunguangLi-an, and R. C. Lamare, “Variable forgetting factor mechanisms for diffusion recursive least squares algorithm in sensor networks,” *EURASIP Journal on Advances in Signal Processing*, vol. 2017, no. 1, pp. 1-23, 2017.

[27] C. R. S. Hanuman and J. Kamala, “Hardware implementation of 24-bit vedic multiplier in 32-bit floating-point divider,” presented at the 2018 4th International Conference on Electrical, Electronics and System Engineering (ICEESE), Kuala Lumpur, Malaysia, 2018.

[28] A. Menon and R. J. Renjith, “Implementation of 24 Bit high speed floating point Vedic multiplier,” *International Journal of*

Advance Engineering and Research Development, vol. 4, no. 5, pp. 742-749, July 2017.

[29] M. lloyde George, "Novel mantissa similarity investigator for path delay reduction of product mantissa calculation," presented at the 25th International Conference, New York, USA, July 2018.

[30] M. Maraş, E. N. Ayvaz, and A. Özen, "A novel adaptive variable forgetting factor RLS algorithm," presented at the 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, 2018.

[31] S. C. Chan, H. J. Tan, and J.Q Lin, "A new local polynomial modeling based variable forgetting factor and variable regularized PAST algorithm for subspace tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54 , no. 3, pp. 1530-1544, June 2018.

[32] Y. Lu, Q. Li, Z. Pan, and Y. Liang, "Prognosis of bearing degradation using gradient variable forgetting factor RLS combined with time series model," IEEE Access, vol. 6, pp. 10986 – 10995, 2018.

[33] S. Asif Hossain, A. Mallik, and Md. Arman Arefin, "A signal processing approach to estimate underwater network cardinalities with lower complexity," *JECEI*, vol. 5, no. 2, pp. 131-138, 2017.

[34] J. Khosravi, M. Shams Esfandabadi, and R. Ebrahimpour, "Image registration based on sum of square difference cost function," *JECEI*, vol. 6, no. 2, pp. 263-271, 2018.

BIOGRAPHIES

Fatemeh Abdi was born in 1984 in Qom, Iran. She received her B.Sc. from Azad University of Saveh in 2008, M.Sc. from the University of Tabriz in 2012 and she is completing the Ph.D. thesis in Shahid Rajaee Teacher Training University (SRTTU), Tehran, Iran, all in the field of Electrical Engineering. Her interests include Hardware Design, Adaptive Filters, Analog Circuits and Systems, Power Amplifier, Voltage Converter and regulator.

Parviz Amiri was born in 1970. He received the B.Sc. degree from University of Mazandaran in 1994, M.Sc. from Khajeh Nasir Toosi University (KNTU Tehran, Iran) in 1997, and his Ph.D. from Tarbiat Modares University (TMU, Tehran, Iran) in 2010, all degrees in Electrical Engineering (Electronics). His main research interest includes electronic circuit design in industries. His primary research interest is in RF and power electronic circuits, with focus on high efficient and high linear power circuit design. He is currently with the Faculty of Electrical Engineering at Shahid Rajaee Teacher Training University (SRTTU), Tehran, Iran.

Mohammad Hossein Refan received his B.Sc. in Electronics Engineering from Iran University of Science and Technology (IUST), Tehran, Iran in 1972. After 12 years working and experience in industry, he started studying again in 1989 and received his M.Sc. and Ph.D. in the same field and the same University in 1992 and 1999, respectively. He is currently an Associate Professor of the Faculty of Electrical Engineering, Shahid Rajaee Teacher Training University (SRTTU), Tehran, Iran. He is the author of about 50 scientific publications on journals and international conferences. His research interests include GPS, DCS, and Automation System.

How to cite this paper:
F. Abdi, P. Amiri, and M. H. Refan, "Low computational complexity and high computational speed in leading DCD ERLS algorithm," *Journal of Electrical and Computer Engineering Innovations*, vol. 7, no. 1, pp. 19-26, 2019.
DOI: 10.22061/JECEI.2019.5666.243
URL: http://jecei.sru.ac.ir/article_1139.html