Review on Implementation of Fir Adaptive Filter Using Distributed Arithmatic and Block Lms Algorithm

Miss. Rima P. Deshmukh[1], Prof. S. A. Koti[2], Prof. V. B. Baru[3]

[1, 2, 3](Department of Electronics and Telecommunication, Sinhgad College of Engineering Vadgaon, Pune, India)

Abstract: *Adaptive filters play very important role in signal processing application. There are several algorithms for implementation of filters such as Least mean square (LMS), Recursive least square (RLS), etc. The LMS algorithm is the most efficient algorithm for implementation of FIR adaptive filters. RLS algorithm gives faster convergence as compared to LMS but the computational complexity is high in case of RLS. An effective distributed arithmetic can be used to implement the block least mean square algorithm (BLMS). The DA based structure uses a LUT sharing scheme to calculate the filter output and weight increment terms of BLMS algorithm. The structure can save a number of adders. This paper presents a literature review on the different algorithms used for implementation of FIR adaptive filters and implementation of filters using distributed arithmetic and block LMS algorithm.*

Keywords: *LMS, BLMS, DA, RLS, ADF.*

I. Introduction

Adaptive filters are the core contributor in the digital signal processing applications. The most popular adaptive algorithms are Least Mean Square (LMS) and Recursive Least Square (RLS) algorithm. Because of simplicity of LMS algorithm it has been productively applied in many areas. Least mean square (LMS) based finite impulse response (FIR) adaptive filter is the most prevalent because it is simple and provides satisfactory convergence performance. FIR filters have many advantages such as FIR filters are linear in phase, coefficients are easy and simple to calculate, the design methods are generally linear, are always stable, and can be realized efficiently in hardware. The adaptive algorithm for FIR filters is broadly used in different applications such as biomedical, communication and control. Because of its simplicity it has been used in many applications where to minimize computational requirements is essential. It is rational to choose the adaptive algorithm in the design of adaptive filter and LMS algorithm can be selected for the designing purpose. Least mean squares algorithms are a class of adaptive filter that find the filter coefficients and relate that coefficients to produce the least mean squares of the error signal to provide the desired filter.

In Recursive least squares algorithm filter coefficients are found recursively. The filter coefficients minimize a weighted linear least squares cost function involving the input signals. The LMS algorithm aims to reduce the mean square error. In the beginning of RLS algorithm, the input signals are considered deterministic. In LMS the input signals are considered stochastic. As compared to other algorithms RLS exhibits faster convergence performance. However the fast convergence is achieved by giving high computational complexity. A distributed arithmetic (DA) can be formed for the application of block least mean square (BLMS) algorithm. DA forms an inner product or dot product of a pair of vectors in a one step by bit serial computation operation. Efficiency of mechanization is the main advantage of DA. DA based techniques uses look up table (LUT) sharing method for the for the calculation of filter outputs and weight-increment terms in BLMS algorithm. Using this technique significant saving of adders is achieved which constitute a main component of DA based structures. DA based BLMS algorithm performs convolution and correlation operation using same LUT. This will reduce the number of LUT words required to be updated per output. Hence saves power consumption and the external logic required.

II. Literature Review

Adaptive digital filters have tremendous applications in signal processing. According to the LMS algorithm, there is a delay in the feedback error for updating the weights which does not favor the pipeline implementation, when the sampling rate is high. [2] have proposed the delayed LMS algorithm for pipeline application of LMS based ADF. In LMS algorithm, the adaptation step can be performed after a fixed delay only, in some practical situations. In such a cases the implemented algorithm is a modified version of LMS algorithm known as the delayed LMS (DLMS) algorithm. In DLMS the coefficient adaptation is performed after a delay. The work shows the conditions for convergence and estimates of convergence rate, both for the mean of the DLMS filter coefficients and for its excess mean square error. The only difference between the LMS and DLMS algorithm is that the correction term for updating the filter weights of the current iteration are computed from error corresponding to the past iteration. Many methods have been proposed to implement BLMS based adaptive digital filters efficiently in systolic VLSI with minimum adaptation delay [2] [17] [11] [12]. In order to avoid adaptation delay [3] has proposed a modified DLMS algorithm. In some of the applications of the adaptive finite impulse response filtering, the adaptation algorithm can be applied with a delay only in the coefficient update. This has a dissimilar effect on the convergence behavior of the algorithm. In this work it is shown how the delayed LMS algorithm can be converted into the typical LMS algorithm at simply slight increase in the computational expense. The modified DLMS is used by [4] to derive a systolic architecture but it requires large amount of hardware resources as compared to the earlier one. BLMS is useful for fast and computationally-efficient implementation of adaptive digital filters. The convergence performance of BLMS ADFs and LMS ADFs are similar, but for block length *L* BLMS ADFs offers *L* fold higher throughput. Considering this , many BLMS algorithms like time and frequency domain block filtered-X LMS (BFXLMS) has been proposed by [19] for specific applications. Computationally more efficient BFXLMS using FFT and fast Hartley transform (FHT). A delayed block LMS algorithm and a concurrent multiplier-based design for high throughput pipeline execution of BLMS ADFs have been proposed [13].

In [13] a block LMS algorithm with delayed weight adaptation for hardware execution of FIR adaptive filters has been proposed. The delayed block least mean square algorithm take a block of *L* input samples and produces block of *L* output, in every training cycle. The simulation result in [13] shows that the DBLMS algorithm has convergence performance same as that of the DLMS algorithm. A highly synchronized systolic architecture for FIR adaptive filters has been derived. The suggested architecture can support *L* time higher sampling rate when compared with the other pipelined designs and hence include less samples of adaptation delays and would provide a more effective execution of LMS-based adaptive filters. [9], [10] have suggested structure for FPGA implementation of BLMS ADFs based on distributed arithmetic. [9] derived a design and implement a high throughput ADF using Fast Block Least Mean Squares (FBLMS) adaptive algorithm. The structure of filter is built on Distributed Arithmetic. The structure calculate the inner product as: (i) shifting (ii) accumulating (iii) storing in look-up table. The desired adaptive digital filter obtained will be multiplier less. Hence a DA based execution of adaptive filter is area efficient. FPGA implementation results imitates that the proposed DA based adaptive filter in [9] can implement with meaningfully smaller area usage, (about 45%) less than that of the remaining FBLMS algorithm based adaptive filter. The structure in [10] replaces multiply-and accumulates operations with a series of look-up-tables (LUT). FPGA implementation results in [10] conforms that the suggested DA based adaptive filter can implement with expressively smaller area usage about 52% less than that of the existing FBLMS algorithm based adaptive filter applications. The structure in [8] for block LMS ADFs supports a very low sampling rate because it uses single multiply-accumulate cell for the computation of filter output and the weight increment term.

DA in [18] uses bit-serial operations and LUTs to implement high throughput filters which uses simply about one cycle per bit of resolution irrespective of filter length. Though, building adaptive DA filters requires recalculation of the LUTs for every adaptation which can deny any performance advantages of DA filtering. With the help of an auxiliary LUT with distinctive addressing, the efficiency and throughput of DA adaptive filters can be made as same order as fixed DA filters. In this paper, a new hardware adaptive filter structure has been suggested for very high throughput LMS adaptive filters. [18] have described the development of DA adaptive filters and showed that practical executions of DA adaptive filters have very high throughput comparative to multiply and accumulate architectures. [18] also showed that DA adaptive filters have a potential area. The power consumption advantage over digital signal processing microprocessor architectures is also achieved.

III. Review On Fir Filters Implementation Using BLMS And DA

3. 1. THE LMS ALGORITHM

The least-mean-square (LMS) is a search algorithm in which a generalization of the gradient vector calculation is made possible by properly modifying the objective function. in order to establish a range for the convergence factor that will guarantee stability, the convergence characteristics of the LMS algorithm are inspected.

The LMS algorithm can be calculated as:

$$w(k+1) = w(k) + \mu e(k)x(k) \tag{1}$$

$$e(n) = d(n)-y(n) \tag{2}$$

$$y(n) = w^T(n)x(n) \tag{3}$$

x(n) and w(n) can be defined as

$$x(n)=[x(n), x(n-1), \ldots, x(n-M+1)]^T \tag{4}$$

$$w(n)=[w(0), w(1), \ldots, w(M-1)]^T \tag{5}$$

where,

x(n) is input vector.
w(n) is the weight vector .
y(n) is the filter output.
d(n) is the desired response.
μ is the convergence factor.

3. 2. Block LMS algorithm

It uses block processing technique in which block of output is calculated from a block of input samples during each iteration. The processing of the LMS ADFs can be increased *L* fold using BLMS algorithm, where *L* is the block size. For (*k*+1)-th input block, BLMS algorithm for updating the filter weights can be given by:

$$w_{k+1}(n) = w_k(n)+\mu\sum_{l=0}^{L-1} e(kL+l)x(kL+l-n) \tag{6}$$

$$y(kL+l)=w_k^{T}(n)x_k(n+l) \tag{7}$$

$$w_k(n)=[w_k(0), w_k(1), \ldots, w_k(N-1)]^T \tag{8}$$

input vector $x_k(n+l)$ consists of *N* input samples

$$x_k(n+l)=[x(kL+l), x(kL+l-1), \ldots x(kL+l-N+1)]^T \tag{9}$$

The *k*-th block of errors can be calculated using the relation

$$e_k(l)=d_k(l)-y_k(l) \tag{10}$$

where

$$y_k(l)=[y(kL), y(kL+1), \ldots, y((k+1)L-1)]^T \tag{11}$$

$$e_k(l)=[e(kL), e(kL+1), \ldots, e((k+1)L-1)]^T \tag{12}$$

$$d_k(l)=[d(kL), d(kL+1), \ldots, d((k+1)L-1)]^T \tag{13}$$

for $l = 0, 1, \ldots, L-1$ and $n = 0, 1, \ldots, N-1$

3.3. Distributed Arithmetic (DA):

Distributed arithmetic plays key role in digital signal processing functions. DA is an efficient technique for calculation of sum of product or vector dot product or multiply and accumulate. The basic DA technique is bit serial in nature. It is basically a bit level rearrangement of MAC operation. It efficiently implements MAC using basic building blocks i.e. look up tables in FPGA. Area saving by using DA is 50-80%. DA can be applied to BLMS and also to the other algorithms to reduce computations and also the area usage.

The DA-BLMS structure consist of one error bit-slice generator (EBSG), one DA-module and one weight-update cum bit-slice generator (WBSG). WBSG updates the filter weights and produces the essential bit-vectors in accordance with the DA-formulation. EBSG computes the error block according to (10) and generates its bit-vectors. The DA-module updates the LUTs and makes use of the bit-vectors created by WBSG and EBSG to compute the filter output and weight-increment terms according to (19) and (20).

Let X_k is the input matrix having size (LxN), L is input block size, N is a filter length and it is decayed into M square matrices S^j_k having size (LxL). The weight vector is decomposed into M short weight vectors C^j_k. the output y_k can be written as

$$y_k = \sum_{j=0}^{M-1} S^j_k . C^j_k \tag{14}$$

Let $c_k^j(r)$ and $e_k(r)$, be the (r + 1)-th components of the L-point vectors c_k^j and e_k, and assumed to be B-bit numbers in 2's complement representation:

$$c_k^j(r) = (c_k^j(r))_0 + \sum_{l=1}^{B-1} 2^{-l} (c_k^j(r))_l \tag{15}$$

$$e_k(r) = (e_k(r))_0 + \sum_{l=1}^{B-1} 2^{-l} (e_k(r))_l \tag{16}$$

$(c_k^j(r))_l$ and $(e_k(r))_l$ are the *l*-th bit of $c_k^j(r)$ and $e_k(r)$. Substituting (15) in (14), we have

$$u(i,j)=\sum_{r=0}^{L-1}[\sum_{l=1}^{B-1}[x(Lj'-i-r)(c_k^j(r))_l]2^{-l} + x(Lj'-i-r)(c_k^j(r))_0] \quad (17)$$

Reorganizing the order of summation, (13) may otherwise be expressed as:

$$u(i,j)=\sum_{r=0}^{L-1} signl\,[\sum_{l=1}^{B-1}[x(Lj'-i-r)(c_k^j(r))_l]\,2^{-l} \quad (18)$$

where $j' = k - j$, $\text{sign}_l = 1$ for $1 \le l \le B\text{-}1$, and $\text{sign}_l = -1$ for $l = 0$. where $j' = k - j$, $\text{sign}_l = 1$ for $1 \le l \le B\text{-}1$, and $\text{sign}_l = -1$ for $l = 0$. Every term in (17) represents the inner-product of s_k^{ij}. All possible inner-products are pre-computed and stored in an LUT and when the l-th bit-vector of weight vector $(c_k^j)_l=[w_k(jL)_l \;\; w_k(jL+1)_l \, .. \, w_k(jL+L\text{-}1)_l]$ for $l = 0, 1, \ldots, B\text{-}1$, is fed to the LUT as address, its inner-product is read from the LUT with s_k^{ij}. The computation of inner sum of (18) in the form of memory read operation could be expressed as:

$$u(i,j)=\sum_{l=0}^{B-1} \text{sign}_l\,[\,F((c_k^j)_l)]\,2^{-l} \quad (19)$$

where F(.) is a memory-read operation, $(c_k^j)_l$ for $l = 0,1, \ldots, B - 1$, is used as LUT-address. The inner-product may be expressed in the form of memory-read operation as

$$v(i,j)=\sum_{l=0}^{B-1} \text{sign}_l\,[\,F((e_k)_l)]\,2^{-l} \quad (20)$$

where e_k is the l-th bit-vector of error-vector e_k defined as:$(e_k)_l = [e(kL), e(kL\text{-}1), \ldots, e(kL\text{-}L\text{-}1)_l]$, which is used as address of an LUT to read its inner-products with s_k^{ij}. LUT contents for the computation of $u(i,j)$ and $v(i,j)$ are just the same. When the bit-vector $(c_k^j)_l$ is used as address, the partial results of $u(i,j)$ are read from the LUT, and when $(e_k)_l$ is used as address, then partial results of $v(i,j)$ are read from the same LUT. Therefore, by using the suggested scheme, a common set of LUTs could be used for the calculation weight-increment terms and filter output. Since, a block of input samples changes after each iteration, the LUTs are necessary to be updated to accommodate the new input-block in each iteration.

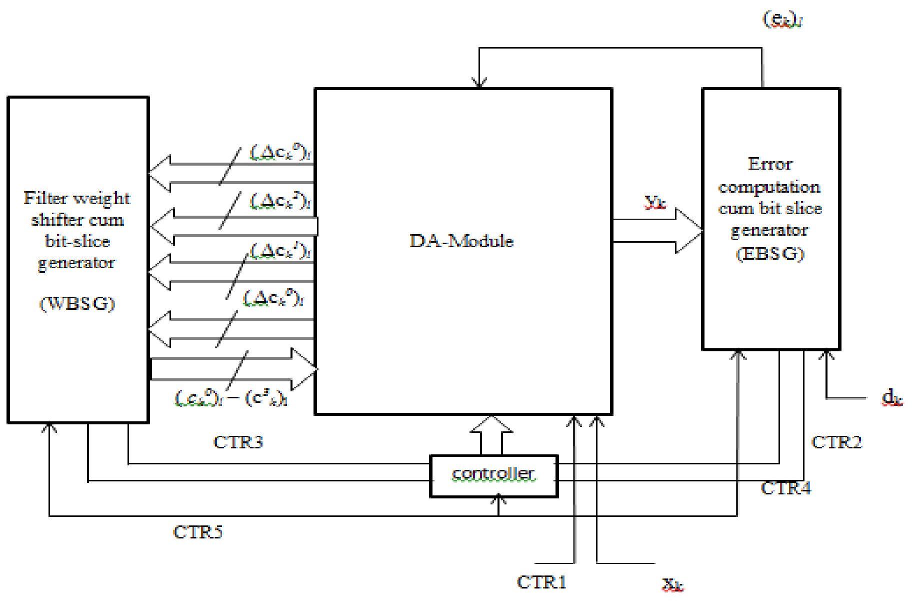

Fig. 2. DA based structure for implementation of BLMS adaptive FIR filters for $N = 16$ and $L = 4$. The subscript l varies from 0 to B-1 in B cycles.

IV. Conclusion

In this paper, the review of FIR Adaptive Filter Using Distributed Arithmetic and Block LMS Algorithm is presented. In this, the different algorithms and the use of distributed arithmetic is discussed. In section II, the work done by different researcher in implementation of adaptive filters is summarized. In section III, the LMS algorithm, BLMS algorithm and FIR filter implementation with the help of BLMS and distributed arithmetic is discussed in brief.

Acknowledgement

I wish to express my sincere thanks and deep sense of gratitude to respected mentor and guide Prof.S.A. Koti, Assistant professor and Prof. V.B. Baru, Associate Professor in Department of Electronics and Telecommunication Engineering of Sinhgad college of Engineering, Vadgaon (BK), Pune 41, for the technical advice, encouragement and constructive criticism, which motivated to strive harder for excellence.

REFERENCES

[1] S.Haykin and B.Widrow, Least-Mean-Square Adaptive Filters. Hoboken, NJ: Wiley- Interscience, 2003.

[2] R.Haimi-Cohen, H.Herzberg, and Y.Beery, "Delayed adaptive LMS filtering: Current results," in Proc.IEEE Int. Conf. Acoust., Speech, Signal Process, Albuquerque, NM, Apr. 1990, pp. 1273–1276.

[3] R.D.Poltmann, "Conversion of the delayed LMS algorithm into the LMS algorithm," IEEE Signal Process. Lett., vol. 2, p. 223, Dec. 1995.

[4] S.C.Douglas, Q. Zhu, and K. F. Smith, "A pipelined LMS adaptive FIR filter architecture without adaptive delay," IEEE Trans. Signal Process, vol. 46, pp. 775–779, Mar. 1998.

[5] S.A.White, "Applications of distributed arithmetic to digital signal processing: A tutorial review," IEEE ASSP Mag., vol. 6, pp. 4–19, Jul. 1989.

[6] D.J.Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst., vol. 52, no. 7, pp. 1327–1337, Jul. 2005.

[7] R.Guo and L.S.DeBrunner, "Two high performance adaptive filter implementation schemes using distributed arithmetic," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 9, pp. 600–604, Sep. 2011.S.

[8] R.Jayashri, H.Chitra, H.Kusuma, A. V. Pavitra, and V. Chandrakanth, "Memory based architecture to implement simplified block LMS algorithm on FPGA," in Proc. Int. Conf. Commun. Signal Process. (ICCSP), Feb. 10–12, 2011, pp. 179–183.

[9] S.Baghel and R.Shaik, "FPGA implementation of fast block LMS adaptive filter using distributed arithmetic for high-throughput," in Proc. Int. Conf. Commun. Signal Process. (ICCSP), Feb. 10–12, 2011, pp. 443–447.

[10] S.Baghel and R.Shaik, "Low power and less complex implementation of fast block LMS adaptive filter using distributed arithmetic," in Proc. IEEE Students Technol. Symp., Jan. 14–16, 2011, pp. 214–219.

[11] L.D.Van and W.S.Feng, "Efficient systolic Architectures for 1-D and 2-D DLMS adaptive digital filters," in Proc. IEEE Asia Pacific Conf. Circuits Syst., Tianjin, China, Dec. 2000, pp. 399–402.

[12] L.D.Van and W.S. Feng, "An efficient architecture for the DLMS adaptive filters and its applications," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 48, no. 4, pp. 359–366, Apr. 2001.

[13] B.K.Mohanty and P.K.Meher, "Delayed block LMS algorithm and concurrent architecture for high-speed implementation of adaptive FIR filters," presented at the IEEE Region 10 TENCON2008 Conf., Hyderabad, India, Nov. 2008.

[14] D.P.Das, G.Panda, and S.M.Kuo, "New block filtered-X LMS algorithms for active noise control systems," IEE Signal Procesd., vol. 1, no. 2, pp. 73–81, Jun. 2007.

[15] D.J.Allred, H.Yoo,V. Krishnan, W. Huang, and D.V.Anderson, "A novel high performance distributed arithmetic adaptive filter implementation on an FPGA," in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), 2004, vol. 5, p. V-161-4.

[16] Basant K. Mohanty, "A High-Performance Energy-Efficient Architecture for FIR Adaptive Filter Based on New Distributed Arithmetic Formulation of Block LMS Algorithm" IEEE Transactions On Signal Processing, Vol. 61, No. 4, February 15, 6–2013.

[17] V.Visvnathan and S.Ramanathan, "A modular systolic architecture for delayed least mean square adaptive filtering," in Proc. IEEE Int. Conf. VLSI Des., Bangalore, 1995, pp. 332–337.

[18] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst., vol. 52, no. 7, pp. 1327–1337, Jul. 2005.

[19] Q.Shen and A.S.Spanias, "Time and frequency domain X block LMS algorithm for single channel active noise control," Control Eng. J., vol. 44, no. 6, pp. 281–293, 1996.