Microprocessors and Microsystems 71 (2019) 102884

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Design and implementation of Least Mean Square adaptive FIR filter using offset binary coding based Distributed Arithmetic

D. Kalaiyarasi[a,*], T. Kalpalatha Reddy[b]

[a] Sathyabama Institute of Science and Technology, Chennai, TamilNadu, India
[b] Sri Venkateswara College of Engineering, for Women, Tirupati, India

ARTICLE INFO

Article history:
Received 15 May 2019
Revised 25 July 2019
Accepted 29 August 2019
Available online 30 August 2019

Keywords:
Least Mean Square
Adaptive filter
Distributed Arithmetic
Carry Save Accumulator
Shift accumulator
Offset binary coding

ABSTRACT

Distributed Arithmetic (DA) based architecture is an efficient technique to attain high throughput without hardware multiplier and also it is essential for bit serial operation. The DA based Finite Impulse Response (FIR) adaptive filter is well suited for hardware implementation in Field Programmable Gate Array (FPGA) device. In conventional DA the partial products of the filter coefficients have been pre-calculated and stored in Look up Table (LUT) which in turn will increase the logic elements and power. To overcome this problem DA based Least Mean Square (LMS) adaptive filter using offset binary coding (OBC) without LUT is proposed. The proposed method will reduce the logic elements by half when compared to the conventional DA based OBC filter. The Carry Save Accumulator (CSA) is used to carry out the operation of shift and accumulation. The proposed architecture is implemented in Quartus II 9.1 with the device as Stratix-EP2S15F484C3 which offers 13.72% high throughput, 56.92% reduction in logic elements, 42.84% reduction in power, 57.74% reduction in logical registers for $N=16$ and for $N=32$ the number of logical element is reduced to 80.87%, 66.66% reduction in power and 24.12% high throughput.

1. Introduction

In signal processing adaptive filters are broadly used for wireless and wired applications. In digital signal processing (DSP), multiplier plays an important role, but it is the primary source of power dissipation in DSP devices. Recently the multiply accumulate (MAC) unit are replaced by DA technique for hardware efficient and low power utilization. The main function of the filter such as bit level multiplication by convolution is carried out using DA.

The previous researches mainly focus on updating weight operation and LUT with concurrent filter operation. The resource utilization by this technique is good but does not satisfy the low hardware complexity. Our primary objectives are to minimize the hardware complexity, area, power and achieving the high throughput rate. The LMS adaptive filter based on DA using OBC with CSA for addition operation is proposed in this paper. Further, the section are organized with the literature study in Section 2, the proposed method in Section 3, result and discussions in Section 4 and the conclusion in Section 5.

2. Literature study

Haykin and Widrow [1] explained a complicated equation of LMS based adaptive filter. White [2] described a biquadrate digital filter with the help of DA by vector dot product as well as vector matrix product mechanization. Through this DA method, an efficient computation mechanism involved. Also, this method has been applied to the transforms and nonlinear processing.

Daniel et al. [3] described an LMS adaptive filter design by LUT update with the help of matched auxiliary LUT which is realized in FPGA and enables rapid prototyping of digital circuits. This method results in a high-speed LMS adaptive filter. Daniel et al. [4] proposed architecture based on hardware adaptive filter of LMS using DA with bit following function as well as LUT to reach elevated throughput from one cycle per bit resolution of filter length along with increased speed. This method was realized using FPGA.

Guo et al. [5] explained adaptive filter using DA which utilizes coefficients as addresses to access the LUT which is stored with sum of delayed input samples. This method results in high speed, low area, and less complexity. NagaJyothi and Sridevi [6] described in detail about the comparative outcomes between FIR filter and an adaptive filter using the architecture of LUT-less DA and LUT based DA.

* Corresponding author.
E-mail address: kalaivlsi2018@gmail.com (D. Kalaiyarasi).

https://doi.org/10.1016/j.micpro.2019.102884
0141-9331/

Ruth Joanna et al. [7] explained DA based FIR filter by using Carry Select Adder (CSLA) for the summation of inner product to reduce time complexity, area and delay. Khan and Ahamed [8] described architecture for LMS adaptive filter based on DA using OBC technique. This reduces the size of lookup tables by half. This method achieves significant reduction in power and area with less hardware complexity.

Guo and DeBruner [9] described a scheme involved the use of address as coefficients for accessing the LUTs. To update the weight of the filter double smart LUT updating technique is used. This method results in high speed and reduced area. Chandekar and Pawar [10] explained a Block LMS (BLMS) based DA adaptive filter algorithm for computing their inner product via shift and accumulate method. They also discussed a parallel architecture to improve the speed, reduce the area and power of the filter.

Mohanty et al. [11] described Intra iteration LUT sharing and LUT using BLMS DA based adaptive filter to decrease the power, area as well as delay. Through this technique the hardware complexity is reduced along with the filter length of blocks size. Park and Meher [12] described a parallel architecture of DA based Adaptive Filter (ADF) by using fast bit clock for CSA. Through this method, the power is reduced with increased speed and low area when compared with the existing technique. Park and Meher [13] explained DA based FIR reconfigurable filter whose coefficients change during runtime with the help of shared LUT. This design reduces 50–70% of the area-delay and energy using Xilinx Virtex-5FPGA device (XC5VSX95T-1FF1136) parallelized. Terrell, [14] described microprocessors based realization using signal filtering with simple low pass filters. Padmapriya and Prabha [15] proposed reconfigurable architectures using dual mode operation. One mode is multiplierless and other mode is testable reversible mode. This method reduces the area and power

3. Related work

3.1. Distributed arithmetic (DA) algorithm

DA is the method which is bit serial in nature and it is quite fast. Also it is an efficient procedure for calculating the inner products as well as variable data vectors. The explicit multiplications are done by the ROM LUTs. In DA, multiplications are mixed and re-ordered so that the arithmetic becomes distributed through the structure rather than lumped. By adopting the DA technique the area is saved upto80%in the DSP applications. Fig. 1 shows the DA based N-tap FIR filter [16]. In the above figure $x(n)$, $x(n-1)$, …$x(n-N)$ represent the input signal which will be applied to LUT and the output is given to the adder. The inputs to the adder are the sign control, output from LUT and output delay feedback data. The adder output is given to the accumulator and finally the output data is attained.

The FIR filter can be represented as,

$$o = \sum_{i=0}^{N-1} x_i g_i \tag{1}$$

where o is the output, x_i is the input and be the N-bit scaled 2's complement number g_i is the filter coefficients

The scaled 2's complement number for i-bits can be expressed as

$$x_i = -x_{i0} + \sum_{j=1}^{k-1} x_{ij} 2^{-j} \tag{2}$$

Substituting (2) in Eq. (1) and can be written as

$$O = \sum_{i=0}^{N-1} g_i x_{i0} + \sum_{i=0}^{N-1} \left[\sum_{j=1}^{k-1} g_i x_{ij} 2^{-j} \right] \tag{3}$$

To convert the sum of product forms into distributed form, the order of the summation over i and j indices are interchanged as,

$$o = \sum_{j=0}^{k-1} g_j 2^{-j} \text{ and } g_j = \sum_{i=0}^{N-1} g_i x_{ij} \tag{4}$$

The partial sum, g_j depends on the x_{ij} values for $j = 0,1,2\ldots k-1$ can have 2^N possible values which are precomputed and stored in LUT.

4. Proposed method

One of the advanced techniques used over DA architecture is offset binary coding (OBC) scheme to reduce the ROM size. OBC is also termed as excess code, excess-k, excess-N or biased representation. In a coding system of digital, all zero's matches to the least amount negative value and all one's matches to the greatest positive value. OBC can be applied to decrease the size of sampled input by a factor of 2 to 2^{L-1}. It can be derived as follows.

$$g_i = \frac{1}{2}\{g_i - (-g_i)\} \tag{5}$$

$$g_i = -g_0 + \sum_{j=1}^{L-1} g_{ij} 2^{-j} \tag{6}$$

$$-g_i = -\overline{g_{i0}} + \sum_{j=1}^{L-1} g_{ij} 2^{-j} + 2^{-(L-1)} \tag{7}$$

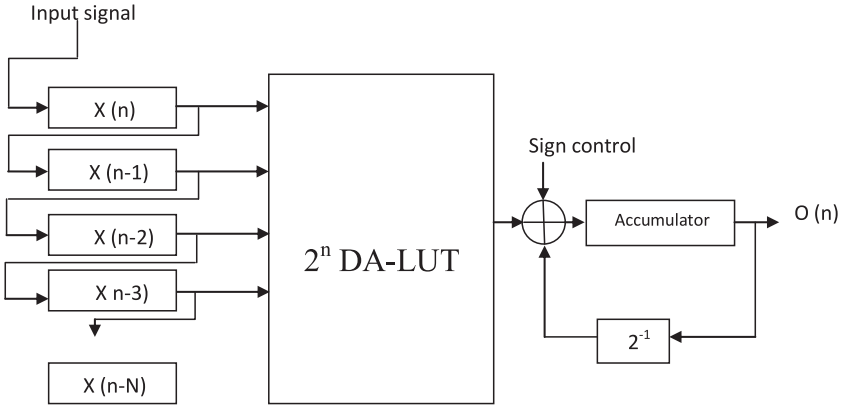

Fig. 1. DA based N-tap FIR filter.

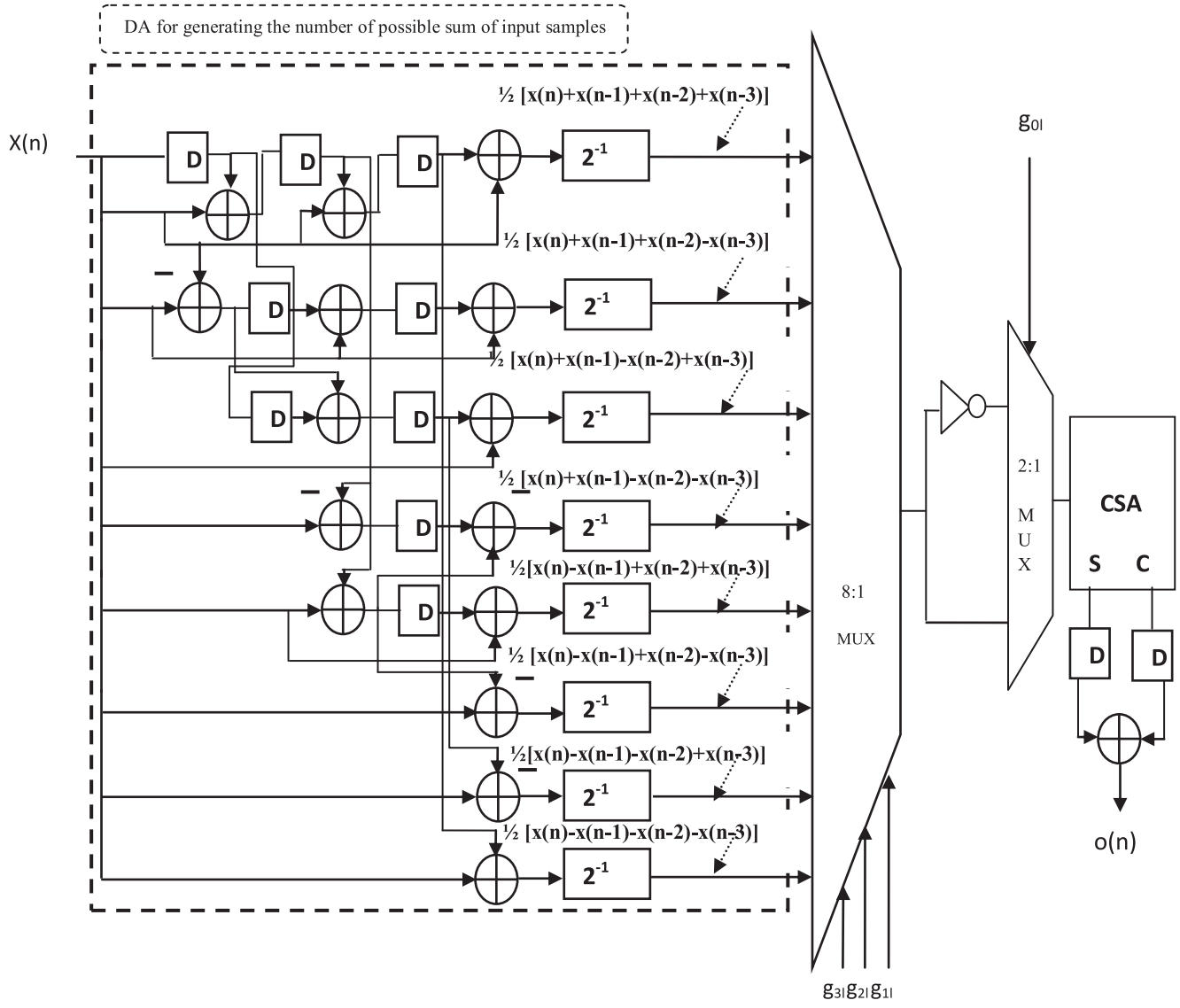

Fig. 2. Proposed OBC based DA filter without LUT for 4- tap FIR filter.

Substituting Eqs. (6) and (7) in Eq. (5),

$$g_i = \frac{1}{2}\left[-g_{i0} + \sum_{j=1}^{L-1} g_{ij}2^{-j} + \overline{g_{i0}} - \sum_{j=1}^{L-1}\overline{g_{ij}}2^{-j} - 2^{-(L-1)}\right] \quad (8)$$

$$= 1/2\{-(g_{i0} - \overline{g_{i0}}) + \sum_{j=1}^{L-1}(g_{ij} - \overline{g_{ij}})2^{-j} - 2^{-(L-1)}) \quad (9)$$

$$= -\sum_{i=0}^{N-1}\frac{1}{2}x_i(g_{i0} - \overline{g_{i0}}) + \sum_{i=0}^{N-1}\sum_{j=1}^{L-1}\frac{1}{2}(g_{ij} - \overline{g_{ij}})2^{-j} - \sum_{i=0}^{N-1}(x_i/2)2^{-(L-1)} \quad (10)$$

By interchanging indices, i and j Eq. (10) can be written as

$$o = -\sum_{i=0}^{N-1}\frac{1}{2}x_i(g_{i0} - \overline{g_{i0}}) + \left[\sum_{j=1}^{L-1}\sum_{i=0}^{N-1}\frac{1}{2}x_i(g_{ij} - \overline{g_{ij}})\right]2^{-j} - \sum_{i=0}^{N-1}\frac{x_i}{2}2^{-(L-1)} \quad (11)$$

The inner product Eq. (11) can have 2^N possible values for any value of N-bit sequence. For each new sample, it is necessary to update the filter as well as the update the filter coefficients, these should be performed concurrently. Otherwise latency will increases as in the case of conventional DA. In the DA based OBC approach, memory may raise exponentially in size from 2^k to 2^{k+1} word which is the major factor for the increase of the ROM size. Due to this technique the hardware complexity to implement LUT memory also increases. Therefore, the DA based OBC technique without LUT is proposed to overcome this disadvantage.

The Fig. 2 shows the design of proposed OBC based DA filter without LUT for 4-tap FIR filter. In this method the number of possible sum of input samples are generated from the Eq. (11).This values are given to the 8:1 multiplexer and one of those input samples are selected with filter coefficients g_{ij} serves as selection line and its output is given to the 2:1 multiplexer.

Then the output of 2:1 MUX is given to the CSA. The output S and C gets delayed and added. Finally the output $o(n)$ value is produced.

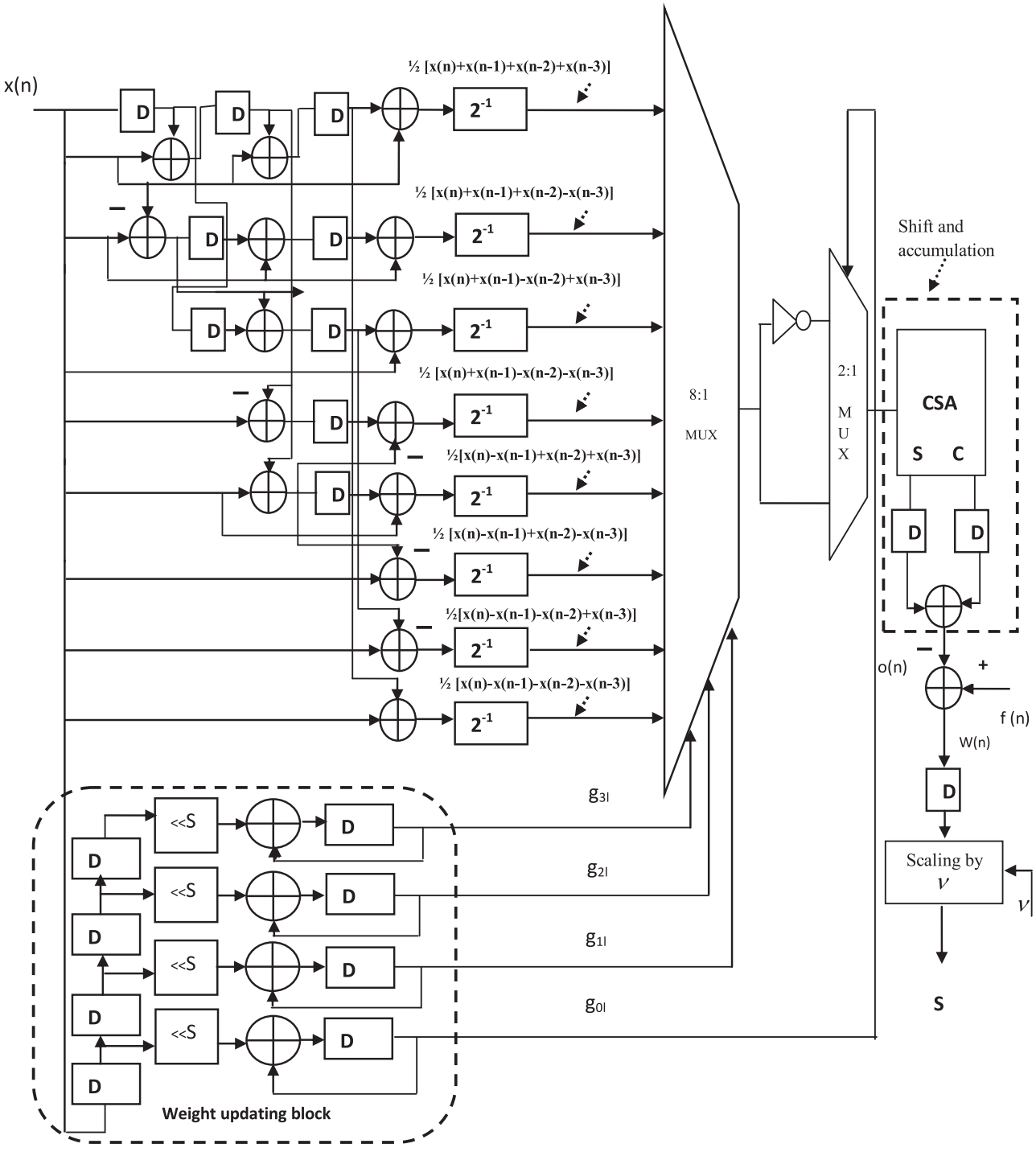

Fig. 3. Proposed LMS adaptive filter design based on DA using OBC without LUT.

4.1. LMS adaptive filter algorithm

The LMS adaptive filter is used to find the coefficients of FIR filter. In the proposed method, LMS act as a weight updating block to calculate the filter coefficients.

In the LMS adaptive filters the weights of the nth iteration are updated by subsequent expressions:

$$g(n+1) = g(n) + \nu.w(n).x(n) \tag{13}$$

where $w(n) = f(n) - o(n)$

$$o(n) = g^{qT}(n).x(n) \tag{14}$$

Here $w(n)$ is the error signalwhere $x(n)$ is the input and $g(n)$ is weight at the nth iteration are as follows:

$$x(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^T \tag{15}$$

$$g(n) = [g_0(n), g_1(n), \ldots, g_{N-1}(n)] \tag{16}$$

$f(n)$ is desired response, filter length is N, $o(n)$ represents output at the nth iteration, $w(n)$ is error signal calculated at the iteration of nth stage, and $w(n)$ is employed to weight update, ν is a factor of convergence. The equation of weight-update filter of

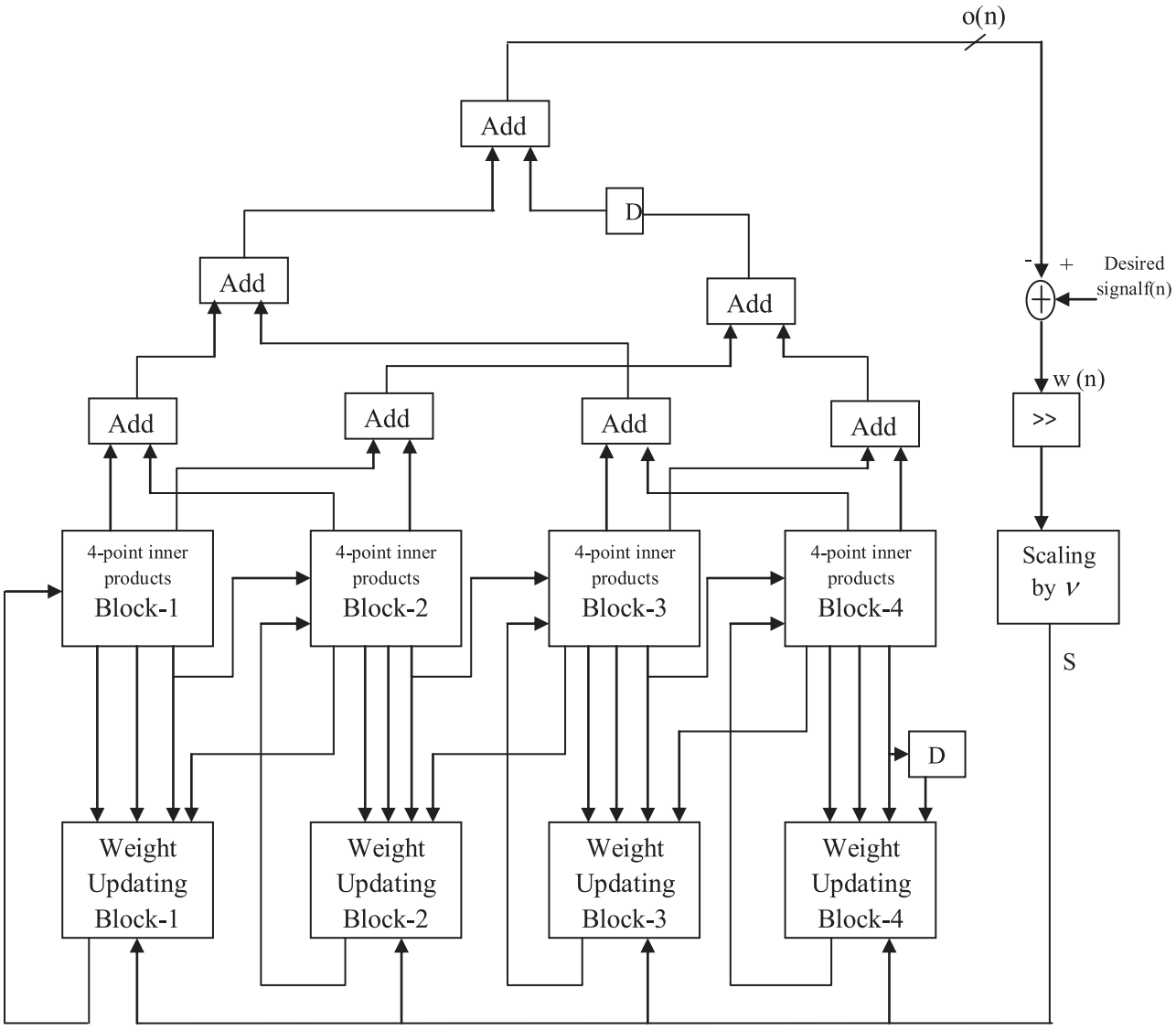

Fig. 4. DA filter of adaptive for $N = 16$.

"LMS" is as follows:

$$g(n+1) = g(n) + \nu.w(n-m).x(n-m) \tag{17}$$

To compute an error and the filter output in each cycle is equal to the difference between the desired response and the current filter output. This process was repeated continuously. Fig. 3 shows the design of proposed LMS Adaptive filter based on DA using OBC without LUT.The figure consists of two main blocks namely filter block and weight updating block. The filter block includes inner products performed with delays and adders without LUT. The output of 8:1MUX is given to the input of 2:1 MUX. The second is the weight updation block which consists of delay, shifter and adder. The product term in LMS Adaptive algorithm is implemented by the shifter which is used to shift the input samples by appropriate number of location i.e. determined by the value of *v.w(n)* and then added with current weights that gives new weights which serves as the address for 8:1 MUX.

The weight updating block output is given to the 8:1 MUX as a selection line input. In parallel, the inner products is updated for every new input samples. This parallel functioning of filtering and weight updation enhances the throughput rate significantly and also the proposed design does not require the extra circuit for address generation which decrease the hardware complexity. After L clock cycles, the filter output $o(n)$ has been obtained by shift and accumulation. To find the error value $w(n)$, the output is subtracted from the desired value $f(n)$ i.e., $w(n) = f(n) - (n)$. The $w(n)$ has been scaled by using convergence factor ν to produce the output as ν w(n). In each iteration, the LMS algorithm computes the value of error and output. Fig. 4 shows $N = 16$ and the inner product $P = 4$ for the DA based adaptive filter.

Similarly, the proposed structure for $N = 32$ is shown in Fig. 5.

5. Results and discussions

The architecture of LMS adaptive filter based on DA using CSA was simulated and synthesized using Quartus II 9.1 with the device as Stratix-EP2S15F484C3 for both proposed and conventional method. Here the input is considered as 8-bit binary data according to that the outputs are obtained and parameter values are calculated.

Table 1 illustrates that the proposed method has better performances than the conventional method in terms of the less delay,

Table 1
Comparison of conventional and proposed architecture.

Design	Filter length N	DAT (ns)	MUF (MHz)	No. of logic elements	No. of registers	Power (mW)	EPS (Energy per sample) (mW•ns)
Conventional design [13]	16	6.542	209.16	903	1013	2.677	17.4744
	32	7.615	249.81	2092	2054	5.043	38.3766
Proposed design	16	5.646	331.36	389	428	1.530	8.6383
	32	5.779	326.07	400	442	1.681	9.7144

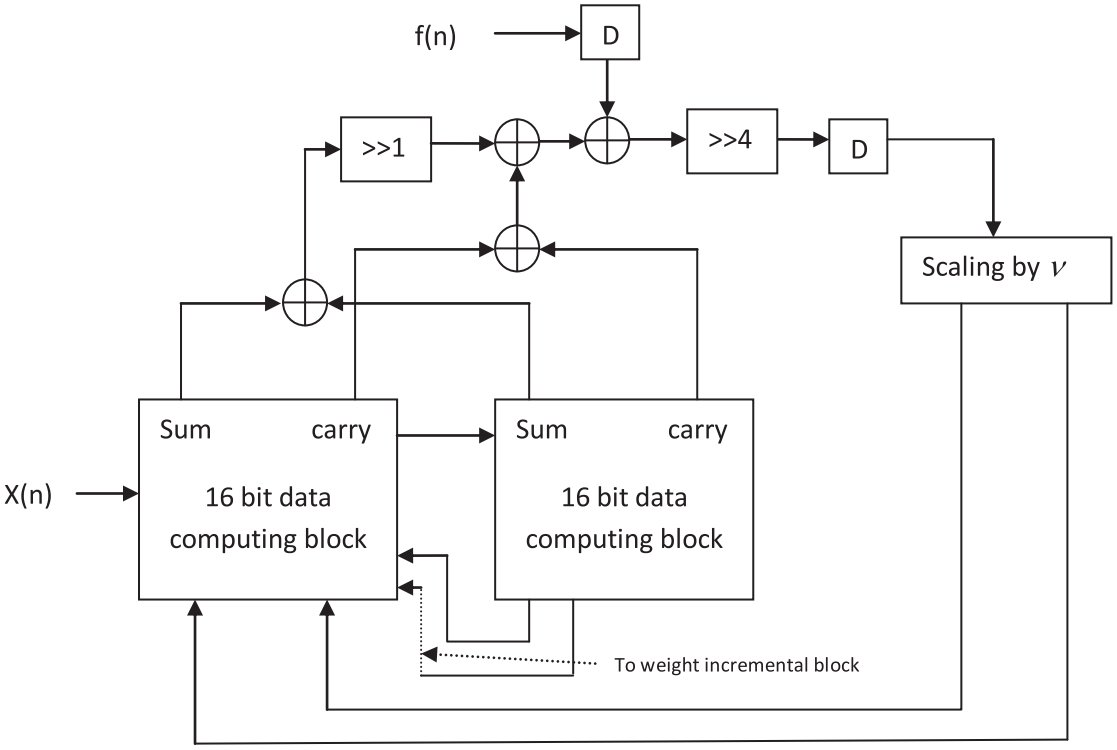

Fig. 5. DA filter of adaptive for $N = 32$.

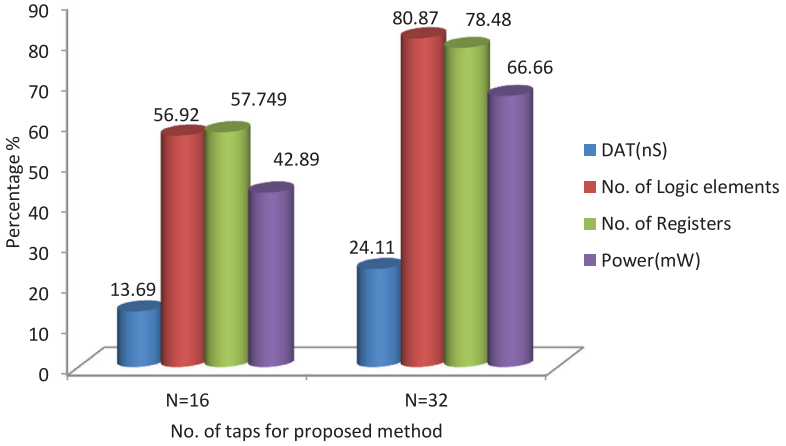

Fig. 6. Efficiency analysis of proposed architecture for $N = 16$ and 32.

less logical elements, less number of registers and less power with the higher throughput rate.

Fig. 6 shows the efficiency analysis of the proposed method for $N = 16$ and $N = 32$.From this figure it clearly shows that the proposed method performances are high compared to the conventional method.

6. Conclusion

In this work, the LMS adaptive FIR filter using DA based OBC without LUT for filter length $N = 16$ and $N = 32$ have been proposed to increase the performance of the filter and the design is synthesized using Quartus II 9.1 StratixEP2S15F484C3. When compared

to the DA based LMS adaptive FIR filter with LUT, the proposed method achieves better performance. The functioning of filtering and weight updation parallelly enhances the throughput rate significantly and also the proposed design does not require any extra circuit for address generation which minimizes the hardware complexity. The proposed method offers 13.72% high throughput,56.92% reduction in logic elements, 42.84% reduction in power, 57.74% reduction in logical registers for $N = 16$ and for $N = 32$ the number of logical element is reduced to 80.87%, 66.66% reduction in power and 24.12% high throughput. From the Table 1 it is observed that the performance of the proposed method is found to be very effective. In future, it can be extended to $N = 64$ and $N = 128$ length with pipeline and parallel processing.

Declaration of Competing Interest

None.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.micpro.2019.102884.

CRediT authorship contribution statement

D. Kalaiyarasi: Conceptualization, Data curation, Validation, Writing - original draft, Writing - review & editing. **T. Kalpalatha Reddy:** Conceptualization, Writing - review & editing.

References

[1] S. Haykin, B. Widrow (Eds.), Least-Mean-Square Adaptive Filters, 31, John Wiley & Sons, 2003.
[2] S.A. White, Applications of the distributed arithmetic to digital signal processing: a tutorial review, IEEE ASSP Mag. 6 (July (3)) (1989) 4–19.
[3] D.-T. Allred, W. Huang, V. Krishnan, H. Yooand, D.V. Anderson, An FPGA implementation for a high throughput adaptive filter using distributed arithmetic, in: Field-Programmable Custom Computing Machines, 12th Annual IEEE Symposium on, April, 2004, pp. 324–325.
[4] D.-J.A. Yoo, V. Krishnan, W. Huang, D.-V. Anderson, LMS adaptive filters using distributed arithmetic for high throughput, IEEE Trans. Circuits Syst. I 52 (7) (2005) 1327–1337.
[5] R. Guo, L.-S. DeBrunner, A novel adaptive filter implementation scheme using distributed arithmetic, in: Signals, Systems and Computers (ASILOMAR), Conference Record of the Forty Fifth Asilomar Conference on, November, IEEE, 2011, pp. 160–164.
[6] G. NagaJyothi, S. Sridevi, Distributed arithmetic architectures for FIR filters–a comparative review, in: Wireless Communications, Signal Processing and Networking (WiSPNET), International Conference on, March, IEEE, 2017, pp. 2684–2690.
[7] S. Ruth Joanna, A.V. Anathalakshmi, Implementation of efficient adaptive FIR filter based on distributed arithmetic, in: IEEE International Conference on Innovations in Information Embedded and Communication Systems, 2015, pp. 1–4.
[8] M.T. Khan, S.R. Ahamed, Area and power efficient VLSI architecture of distributed arithmetic based LMS adaptive filter, in: 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID), January, IEEE, 2018, pp. 283–288.
[9] R. Guo, L.S. DeBruner, Two high-performance adaptive filter implementation schemes using distributed arithmetic, IEEE Trans. Circuits Syst.-II 58 (September (9)) (2011).
[10] A.A. Chandekar, M. Pawar, Delay and power optimized adaptive filter using distributed arithmetic, in: International Conference on Electronics, Communication and Aerospace Technology, 2017, pp. 256–261.
[11] B.K. Mohanty, P.K. Meher, S.K. Patel, LUT optimization for distributed arithmetic-based block least mean square adaptive filter, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 24 (May (5)) (2016) 1–10.
[12] S.Y. Park, P.K. Meher, Low power, high throughput and low area adaptive FIR filter based on distributed arithmetic, IEEE Trans. Circuits Syst.-II 60 (June (6)) (2013).
[13] S. Yoon Park, P.K. Meher, Efficient FPGA and ASIC realizations of DA-based reconfigurable fir digital filter, IEEE Trans. Circuits Syst.-II 61 (July (7)) (2014) 1–5.
[14] T.J. Terrell, Microprocessor-based signal filters, Microprocess. Microsyst. 11 (3) (1987) 131–140.
[15] S. Padmapriya, V.L. Prabha, Design of an efficient dual mode reconfigurable FIR filter architecture in speech signal processing, Microprocess. Microsyst. 39 (7) (2015) 521–528.
[16] D. Kalaiyarasi, T.K. Reddy, Area efficient implementation of FIR filter using distributed arithmetic with offset binary coding, IOSR J. VLSI Signal Process. 4 (3) (2014) 01–09.

D. Kalaiyarasi received the Bachelor of Engineering in-Electronics and Communication Engineering from University of Madras and Master of Engineering in Power Electronics and Drives system from Anna University Chennai, Tamilnadu India. She is working as an Associate Professor in Department of Electronics and Communication Engineering at Panimalar Engineering College, Chennai. Currently, She is prusing her PhD in Department of Electrical and Electronics Engineering from Sathyabama Institute of Science and Technology Chennai, India. She published many papers in International/National journals and conferences. Her areas of interests are Signal processing, Image processing and VLSI. She is a member of IEEE.

T. Kalpalatha Reddy has graduated in Electronics and Communication Engineering from Sri Venkateswara University, Tirupati, India. She has done Master degree in Applied Electronics and Ph.D. in the faculty of Electronics and Communications Engineering, Anna University, Chennai, India. She has over 25 years of Teaching Experience. She is a member of many professional body including IET, IETE and ISTE. She has published many papers in peer referredInternational and National journals and conferences.