Circuits, Systems, and Signal Processing (2021) 40:1458–1478
https://doi.org/10.1007/s00034-020-01539-y

Efficient Architecture for the Realization of 2-D Adaptive FIR Filter Using Distributed Arithmetic

Prabhat Chandra Shrivastava[1] · **Prashant Kumar**[1] · **Manish Tiwari**[1] · **Amit Dhawan**[1]

Received: 3 October 2019 / Revised: 17 August 2020 / Accepted: 4 September 2020 /
Published online: 18 September 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

This paper presents an efficient architecture for two-dimensional (2-D) adaptive FIR filter architecture using the distributed arithmetic (DA) algorithm. DA-based filter architectures essentially require look-up tables (LUT). In the proposed filter architecture, RAM- or ROM-based LUT is replaced by adders- and logic gates-based structure that generates the LUT value corresponding to the input. Therefore, the MAC unit requires fewer logic gates and adders in DA-based realization. In addition, the memory sharing concept in architecture reduces the delay elements. Moreover, the complexity of the LUT hardware of higher-order filters is reduced by parallelly dividing the internal MAC block for the DA decomposition which offers a higher degree of modularity and parallelism in the proposed architecture. Further, 2-D delayed LMS algorithm is used for the updation of the filter coefficient weights. Furthermore, two-stage pipelining is used to reduce the critical path of the architecture and it also makes critical path delay independent of the order of the filter. ASIC synthesis results reveal the advantages of the proposed structure by reducing the area, power, ADP and EDP by 54%, 48.19%, 55% and 49%, respectively, as compared with the existing architecture for filter size 8×8.

Keywords 2-D adaptive filter · Distributed arithmetic · Hardware-based LUT · Multiplier-less filters · Raster scanning · 2-D delayed LMS algorithm

✉ Prabhat Chandra Shrivastava
prabhatphd@gmail.com

Prashant Kumar
pkumar.mnnit@gmail.com

Manish Tiwari
mtiwari@mnnit.ac.in

Amit Dhawan
dhawan@mnnit.ac.in

[1] Department of Electronics and Communication Engineering, MNNIT-Allahabad, Prayagraj, UP 211004, India

1 Introduction

In recent years, the two-dimensional (2-D) digital systems have attracted increasing attention due to their theoretical as well as application importance in digital filtering. The main feature of a 2-D filter is that the system dynamics is a function of two independent variables as a result of information propagation in two independent directions [31]. The 2-D adaptive digital filters have a wide range of communication and DSP applications such as adaptive equalizer, image enhancement, image compression and so on [7, 14, 27]. The 2-D adaptive digital filters can conveniently be simulated on general-purpose computers but the process may not be suitable for real-time applications. Hence, applications such as real-time image processing that involve a high data rate needs dedicated hardware structures to meet the high throughput demand. Hardware structures for the adaptive filter require weight updation algorithm to compute the filter coefficient in each iteration.

The LMS algorithm is the most widely used algorithm for the design of adaptive digital filters, not only due to simplicity in the calculation but also due to its good convergence rate. But still, LMS-based conventional structures for the 2-D adaptive filters require a large number of components in implementation and have very long critical path. Many algorithms and architectures for the 2-D adaptive filters have been reported in the literature [1, 3, 5, 6, 10, 15, 16, 19, 20, 24, 28, 29, 33, 35] to improve the convergence rate and to reduce the hardware and time complexities.

Hadhoud and Thomas [5] have presented the 2-D adaptive filter algorithm and derived 2-D LMS (TDLMS) algorithm for weight updation. Image enhancement using the TDLMS algorithm is described by Soni et al. [28]. Youlal et al. [35], has proposed a 2-D joint process lattice algorithm to tackle the convergence rate problem in 2-D adaptive digital filters, and it has been shown that this algorithm has a higher convergence rate than the TDLMS algorithm. Cho et al. [1]. has suggested an automatic step size adjustment method for the TDLMS algorithm to improve the convergence rate as well as estimation error in 2-D adaptive digital filters. Further, Ohki [24] has suggested a new 2-D LMS algorithm that can converge both the vertical and horizontal directions of a 2-D plane. The 2-D block adaptive filtering algorithms are presented by Mikhael [19, 20], Wang [33] and Tan [29] for IIR filtering. The block adaptive filtering algorithms tend to decrease the computational time with good convergence.

The direct form implementations of 2-D adaptive FIR filters have a very long critical path due to inner product computation for the weight updation and the computing output. This long critical path problem may be tackled by the pipelined-based architecture as suggested by [3, 6, 10, 15, 16, 32]. The pipelined architecture not only increases the latency in output computation but also the throughput of the architecture. Harada et al. [6] has suggested a pipelined architecture for the 2-D normalized LMS (TDNLMS) adaptive digital filters. In this architecture, it is shown that a constant and short critical path can be achieved without producing output latency. Systolic pipelined architectures of 2-D adaptive FIR filters are suggested by Van in [32]. In this suggested architectures, they have reduced the critical path delay using the multistage pipelining.

In the VLSI design power, area and throughput are the main constraints. The 2-D adaptive filter structure discussed above requires large numbers of multipliers that acquire a very large area in the chip for implementation. In the last two decades, multiplier-less technique, i.e., distributed arithmetic (DA)-based techniques, has become very popular in VLSI design due to its efficient area and high throughput capabilities of the multiply and accumulate (MAC) design [4, 8, 9, 11, 12, 17, 18, 22, 23, 25, 26, 30, 34]. Using this technique, Peled and Liu [26] have realized a 1-D digital filter which has far better performance than the multiplier-based structures. A ROM-based MAC has been suggested by White [34] for the designing of the fixed coefficient digital filters. Later, Grover [4] has suggested the efficient DA-based structure for the higher-order MACs. In the past decade, many 1-D digital filters using DA have been suggested in the literature [4, 8, 9, 17, 18, 22, 23, 25, 30]. Recently, researchers have suggested the efficient architectures for 1-D adaptive digital filters [8, 9, 18, 22, 30] based on DA algorithm, and in these reported architectures, researchers have shown a drastic reduction in power, area and critical path delay over the multiplier-based designs. Here, the authors feel that if we design the 2-D adaptive filter using DA techniques, then, we may expect to achieve the significant improvement as compared with existing designs. To the best of the author's knowledge, such problem has been not reported in the literature so far.

In this paper, we have proposed a multiplier-less 2-D adaptive FIR filter structure using a distributed arithmetic technique. The most important aspect of the proposed structure is that it has reduced the hardware complexity to a great extent, since multipliers are not present in the proposed structure. Using DA techniques, MAC is designed using the few gates and adders. Time complexities in the structure are reduced by using the parallelism and the pipelining approach. 2-D delayed LMS (TDDLMS) algorithm is used for the adaptive weight updation.

In the next section, we have discussed the preliminaries of the 2-D LMS algorithm and TDDLMS algorithm. DA decomposition of the 2-D adaptive FIR filter is discussed in Sect. 2. In Sect. 3, the proposed architecture is presented and the detailed working of the proposed structure is also discussed. In Sect. 4, hardware and time complexities of proposed structures are discussed and performance comparison of the proposed structures and the existing structures is presented. Finally, in Sect. 5, conclusions are given.

2 Preliminaries

Let us consider an image matrix D of dimension $M \times M$. We have also assumed that the image D is corrupted by the noise, and the corrupted image is represented by the matrix X. Let matrix $[W]_{N\times N}$ is the weight of the filter coefficients of 2-D adaptive

FIR filter. Image matrix D, corrupted image matrix X and weight of filter coefficient W may be defined as

$$D = \begin{bmatrix} d(m,n) & d(m,n-1) & \ldots\ d(m,n-M+2) & d(m,n-M+1) \\ d(m-1,n) & d(m-1,n-1) & \ldots\ d(m-1,n-M+2) & d(m-1,n-M+1) \\ d(m-2,n) & & & \\ . & & & \\ . & & & \\ d(m-M+1,n) & d(m-M+1,n-1) & \ldots\ \ldots & d(m-M+1,n-M+1) \end{bmatrix}_{M\times M} \tag{1a}$$

$$X = \begin{bmatrix} x(m,n) & x(m,n-1) & \ldots\ x(m,n-M+2) & x(m,n-M+1) \\ x(m-1,n) & x(m-1,n-1) & \ldots\ x(m-1,n-M+2) & x(m-1,n-M+1) \\ x(m-2,n) & x(m-2,n-1) & \ldots\ x(m-2,n-M+2) & x(m-2,n-M+1) \\ . & & & \\ & . & \ldots & \\ x(m-M+1,n) & x(m-M+1,n-1) & \ldots\ \ldots & x(m-M+1,n-M+1) \end{bmatrix}_{M\times M} \tag{1b}$$

and

$$W = \begin{bmatrix} w(0,0) & w(0,1) & w(0,2)\ \ldots\ w(0,N-2) & w(0,N-1) \\ w(1,0) & w(1,1) & w(1,2)\ \ldots\ w(1,N-2) & w(1,N-1) \\ w(2,0) & w(2,1) & w(2,2)\ \ldots\ w(2,N-2) & w(2,N-1) \\ . & & & \\ . & & & \\ w(N-1,0) & w(N-1,1)\ \ldots & \ldots\ldots & w(N-1,N-1) \end{bmatrix}_{N\times N} \tag{1c}$$

The output equation for 2-D adaptive FIR filter can be given as

$$y(m,n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x(m-i,n-j)\, w_k(i,j) \tag{2}$$

where $x(m,n)$, $w_k(i,j)$ and $y(m,n)$ are the current processed corrupted input pixel, weight of the filter coefficient at kth iteration and the output pixel, respectively. Iteration number k can be given as $k = mM + n$, where $0 \le m \le M-1$ and $0 \le n \le M-1$. The weight of the filter coefficient w_k in each iteration may be updated by the TDLMS algorithm. If $e(k)$ represents the error in the output pixel $y(m,n)$ with respect to the desired pixel $d(m,n)$, then, the error signal $e(k)$ at kth iteration can be given as

$$e(k) = d(m,n) - \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x(m-i,n-j)\, w_k(i,j) \tag{3}$$

mean square error (MSE) of $e(k)$ can be calculated as

$$\mathrm{MSE} = E\{e(k)^2\} \tag{4}$$

by putting $e(k)$ from (3) into (4)

$$E\{e(k)^2\} = E\{d^2(m,n) + \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\sum_{p=0}^{N-1}\sum_{q=0}^{N-1} x(m-i,n-j)x(m-p,n-q) \times w_k(i,j)w_k(p,q) - 2\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} d(m,n)\,x(m-i,n-j)w_k(i,j)\} \tag{5}$$

weight of the filter coefficient corresponding to the least value of $E\{e(k)^2\}$ may be calculated using the maxima–minima theorem. After simplification, the adaptive weight updation algorithm from [5] can be expressed as

$$w_{k+1}(i,j) = w_k(i,j) + 2\mu\, e(k)\, x(m-i,n-j) \tag{6}$$

where μ is a convergence factor. Realization of 2-D adaptive FIR filters based on (6) may have the very long critical path due to the feedback loop. Here, delay register can be used in the critical path to reduce delay which modifies the weight updation algorithm (6) as follows,

$$w_{k+1}(i,j) = w_k(i,j) + 2\mu\, e(k-D)\, x(m_D-i,n_D-j) \tag{7a}$$

where D is delay used in the critical path and this algorithm is named as 2-D delayed LMS (TDDLMS) algorithm. In realization of TDDLMS, the degree of the pipeline process is determined by the amount of delay D. By inserting the delay register after each computation unit, we can minimize the critical path delay [6, 10], but on the other hand, the convergence property of this algorithm becomes poor as the amount of delay registers increase in the critical path and it also increases the output latency. Therefore, we cannot use the higher order of pipelining in the 2-D adaptive FIR filter. Poor convergence in DLMS algorithm can be improved by the proper selection of the convergence factor μ. For the proper convergence of (5), the expression for μ is derived in [24] and from (16) of [24], it can be expressed as

$$0 < \mu < \frac{1}{\lambda\text{max}} \le \frac{1}{\lambda} \tag{7b}$$

where

$$\lambda = E\|\, x(m,n)x(m-i,n-i)\|\,. \tag{7c}$$

The output Eq. (2) of the 2-D adaptive FIR filter can be decomposed by DA algorithm similarly to the DA-based decomposition given in [17, 18, 22, 23, 25, 30]. To get DA expression for the 2-D adaptive FIR filter, Eq. (2) can be further expressed as

$$y(m,n) = \sum_{i=0}^{N-1} v^i \tag{8}$$

where

$$v^i = \sum_{j=0}^{N-1} x(m-i,n-j)\, w_k(i,j) \tag{9}$$

Let us consider filter weight coefficient $w_k(i,j)$ in (9) is the normalized value $|w_k(i,j)| < 1$ and represented in 2's complement form. Thus, the weighted 2's complement representation for $w_k(i,j)$ may be given as

$$w_k(i,j) = -w^{ij}_{k[0]} + \sum_{b=1}^{L-1} w^{ij}_{k[b]} 2^{-b} \tag{10}$$

By using (10), Eq. (9) can be written as

$$v^i = \sum_{j=0}^{N-1} x^{ij} \left[-w^{ij}_{k[0]} + \sum_{b=1}^{L-1} w^{ij}_{k[b]} 2^{-b} \right] \tag{11}$$

where $x^{ij} = x(m-i,n-j)$. After rearrangement of Eq. (11), it may be expressed as

$$v^i = \sum_{b=0}^{L-1} \left[\sum_{j=0}^{N-1} x^{ij}\, \hat{w}^{ij}_{k[b]} \right] 2^{-b} \tag{12}$$

where

$$\hat{w}^{ij}_{k[b]} = \begin{cases} -w^{ij}_{k[b]} & \text{for } b = 0 \\ w^{ij}_{k[b]} & \text{otherwise} \end{cases}$$

Equation (12) can be further written as

$$v^i = \sum_{b=0}^{L-1} v^{ib} 2^{-b} \tag{13a}$$

where

$$v^{ib} = \sum_{j=0}^{N-1} x^{ij}\, \hat{w}^{ij}_{k[b]} \tag{13b}$$

In (13b), v^{ib} represents the matrix multiplication of weight vector $\hat{w}_{k[b]}^{ij}$ and bit vector x^{ij} and mathematically it can be represented as

$$v^{ib} = \left[x^{i0}\ x^{i1}\ x^{i2}\ x^{i3} \ldots x^{iN-2}\ x^{iN-1} \right] \begin{bmatrix} \hat{w}_{k[b]}^{i0} \\ \hat{w}_{k[b]}^{i1} \\ . \\ . \\ \hat{w}_{k[b]}^{iN-2} \\ \hat{w}_{k[b]}^{iN-1} \end{bmatrix} \tag{14}$$

Here, the value of $\hat{w}_{k[b]}^{ij}$ may be either 0 or 1, so v^{ib} can be represented as the summations of weight vector $\hat{w}_{k[b]}^{i0}\hat{w}_{k[b]}^{i1} \ \ldots \ \hat{w}_{k[b]}^{iN-1}$. For bit vector, x^{i0}, x^{i1}, x^{i2}, x^{i3}, … $x^{i(N-2)}$, $x^{i(N-1)}$ 2^N bit combinations are possible. Many methods have been proposed by researchers [4, 17, 18, 23, 25, 30, 34] to precompute all the possible values of the inner product summation expressions (IPSE). In [34], the author has suggested the ROM-based architecture in which all possible values of IPSEs are precomputed and stored in ROM. Further, MUX is used to access the IPSE value corresponding to the bit vectors. But in the case of the adaptive filter, the value of filter coefficients changes adaptively which prohibits the use of ROM-based architecture. For such filters, RAM-based architecture has been suggested by Meher [4] but RAM-based architectures require a large number of registers in realization. In the proposed architecture, we have used the basic combinational circuits-based structure to generate the IPSE as depicted in Fig. 1. In SIPSE's architecture (Fig. 1), AND gate is used to select the desired input samples corresponding to the bit vector, thereafter adders are used to generate the corresponding IPSE. In Fig. 1, at the last stage, addition is performed using the carry-lookahead adder (CLA) to reduce the path delay. Further, from (13b), it can be observed that to compute the value of v^{ib}, accessed IPSE is required to be shifted and added to the previous result. Carry-save adder (CSA) based structure is used for shifting and adding the IPSE as shown in Fig. 2. In this structure, the previous output result of full adder is fed to the input to the next right full adder whose results provide the desired shifting along with accumulation.

As the order of filter increases, the complexity of the structure (Fig. 1) proposed for the (14) also increases. It adds $(\log_2 N)T_{\text{adder}}$ delay in the critical path. So, to reduce the critical path delay, higher-order matrix multiplication of weight vector and bit vector can be subdivided into the small order of matrix multiplication and can be expressed as

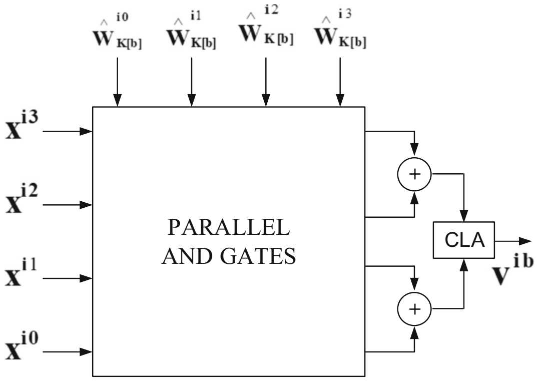


Fig. 1 Structure for generation of IPSE (SIPSE)

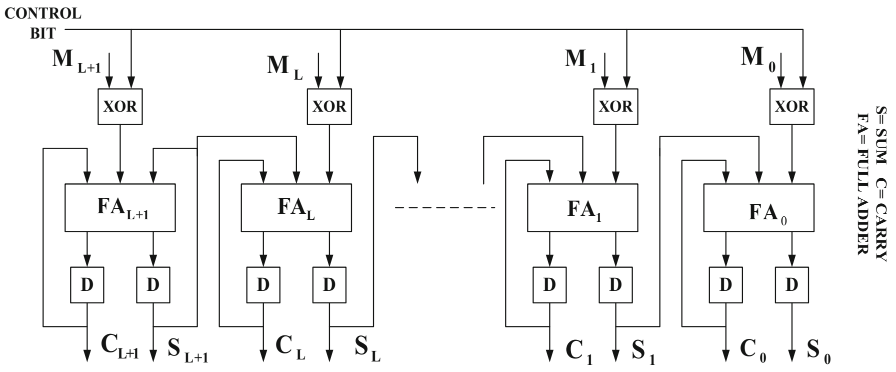


Fig. 2 Shift Accumulation unit (SAU)

$$v^{ib} = \left[x^{i0}\ x^{i1}\ \ldots\ x^{ip-1}\right] \begin{bmatrix} \hat{w}^{i0}_{k[b]} \\ \hat{w}^{i1}_{k[b]} \\ . \\ . \\ \hat{w}^{ip-1}_{k[b]} \end{bmatrix} + \left[x^{ip}\ x^{i(p+1)}\ \ldots\ x^{i2p-1}\right] \begin{bmatrix} \hat{w}^{i\,p}_{k[b]} \\ \hat{w}^{i(p+1)}_{k[b]} \\ . \\ . \\ \hat{w}^{i2p-1}_{k[b]} \end{bmatrix}$$

$$+ \cdots + \left[x^{i(Q-1)p}\ \ldots\ x^{iQp-1}\right] \begin{bmatrix} \hat{w}^{i(Q-1)p}_{k[b]} \\ \hat{w}^{i1}_{k[b]} \\ . \\ . \\ \hat{w}^{iQp-1}_{k[b]} \end{bmatrix} \quad (15)$$

where $Q = N\big/P$ and P is small-order matrix multiplication. After using this decomposition, (9) can be rewritten as

$$v^i = \sum_{b=0}^{L-1} v_0^{ib} 2^{-b} + \sum_{b=0}^{L-1} v_1^{ib} 2^{-b} + \cdots + \sum_{b=0}^{L-1} v_{Q-2}^{ib} 2^{-b} + \sum_{b=0}^{L-1} v_{Q-1}^{ib} 2^{-b} \tag{16}$$

where $v_0^{ib}, v_1^{ib}, \ldots, v_{Q-2}^{ib}, v_{Q-1}^{ib}$ represent internal expressions of (15) and each expression $v_0^{ib}, v_1^{ib} \ldots, v_{Q-2}^{ib}, v_{Q-1}^{ib}$ is similar to the (14). Therefore, similar to the structure of (14), a dedicated structure can be designed for each term of $v_0^{ib}, v_1^{ib}, \ldots, v_{Q-2}^{ib}, v_{Q-1}^{ib}$. Further, shift accumulator unit (SAU) can be used for shift and addition with each individual structure of $v_0^{ib} v_1^{ib}, \ldots, v_{Q-2}^{ib} v_{Q-1}^{ib}$ to calculate the internal summation expressions of (16). The generalized SAU is depicted in Fig. 2. Output of internal summations of (16) can be pipelined and after pipeline can be added using the adder to calculate v^i as shown in Fig. 3.

3 Proposed Architecture for the 2-D Adaptive FIR Filter

The adaptive filter structure is a unified structure of filtering and weight updating block as depicted in Fig. 4. The filtering block and weight updating block update their filter coefficients according to which types of adaptive algorithms are used. In this proposed architecture, we have used 2-D FIR filter. Equations (2) and (7a) represent the mathematical expression of 2-D FIR filter unit and weight updation unit, respectively. For systolic structure realization, the z-domain representation of (2) may be expressed as

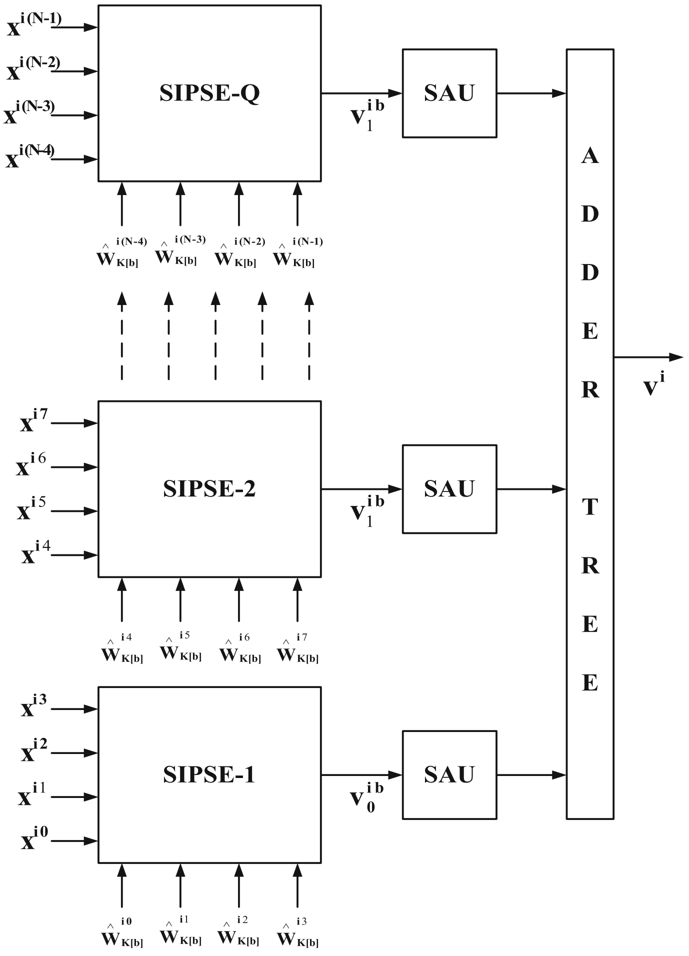
$$\begin{aligned} Y(z_1, z_2) = (((\ldots(X_N(z_1, z_2)z_1^{-1} + X_{N-1}(z_1, z_2))z_1^{-1} + \cdots + X_2(z_1, z_2))z_1^{-1} \\ + X_1(z_1, z_2))z_1^{-1} + X_0(z_1, z_2)), \end{aligned} \tag{17}$$

where

$$X_r(z_1, z_2) = \sum_{i=0}^{N-1} X(z_1, z_2) z_2^{-i} w_k(r, i) \tag{18}$$

Here, input samples are processed in the row-wise scanning mode (i.e., Raster Scanning). In the raster scanning mode, the input samples appear as $\{x(0, 0), x(0, 1), x(0, 2), \ldots x(0, M-2), x(0, M-1), x(1, 0), x(1, 1), \ldots x(1, M-1), \ldots x(M-1, 0), x(M-1, 1), \ldots x(M-1, M-1)$, where M represents the width of the image input sample. To generate the desired delayed input samples in each iteration, $M-1$ delay is required to add at each systolic stage in (13) and hence $z_1^{-1} = z^{-M}$. In order to map (17) into the desired systolic structure, M length shift register block (SRB) is needed at each systolic stage. The SRB blocks are depicted in Fig. 5.

After analyzing the content of the shift registers (which are used for the raster scanning at each systolic stage), we find that each systolic stage's SRB contains all



$x^{i(N-1)}$
$x^{i(N-2)}$
$x^{i(N-3)}$
$x^{i(N-4)}$
SIPSE-Q
v_1^{ib}
SAU
$\hat{W}_{K[b]}^{i(N-4)}$ $\hat{W}_{K[b]}^{i(N-3)}$ $\hat{W}_{K[b]}^{i(N-2)}$ $\hat{W}_{K[b]}^{i(N-1)}$
x^{i7}
x^{i6}
x^{i5}
x^{i4}
SIPSE-2
v_1^{ib}
SAU
$\hat{W}_{K[b]}^{i4}$ $\hat{W}_{K[b]}^{i5}$ $\hat{W}_{K[b]}^{i6}$ $\hat{W}_{K[b]}^{i7}$
x^{i3}
x^{i2}
x^{i1}
x^{i0}
SIPSE-1
v_0^{ib}
SAU
$\hat{W}_{K[b]}^{i0}$ $\hat{W}_{K[b]}^{i1}$ $\hat{W}_{K[b]}^{i2}$ $\hat{W}_{K[b]}^{i3}$
ADDER TREE
v^i

Fig. 3 DA decomposition structure for higher-order MAC

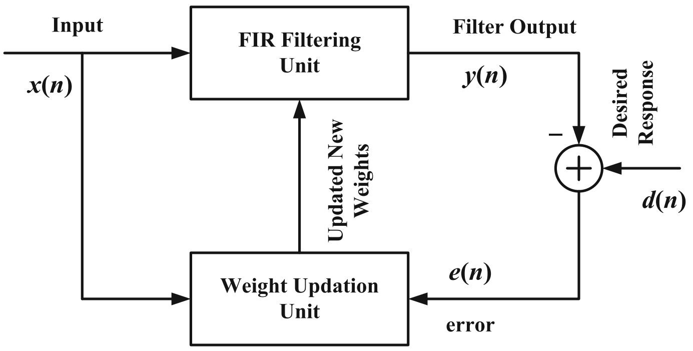


Fig. 4 Block diagram of adaptive filter

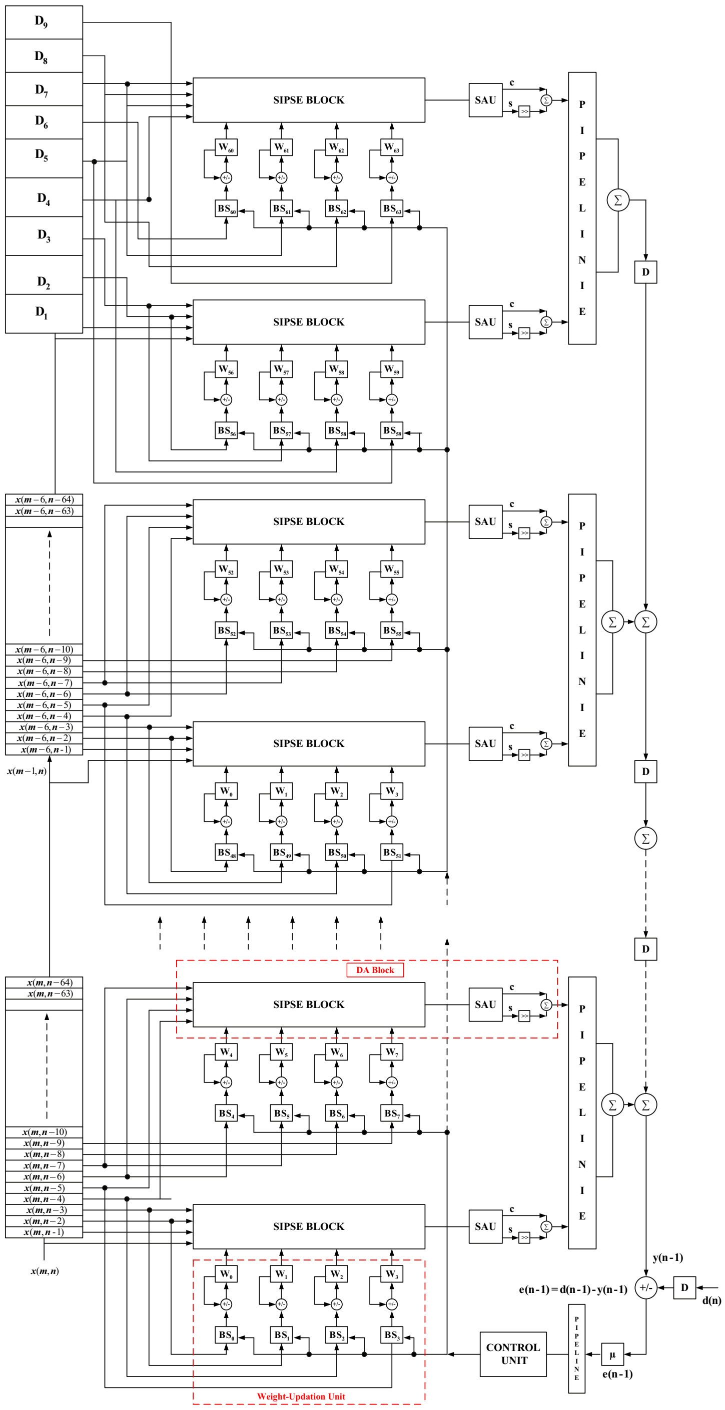


Fig. 5 Proposed architecture of 8 × 8 size 2-D adaptive FIR filter

the required delayed input samples. Therefore, unlike the suggested structures of [3, 6, 10, 16], internal-systolic structures are not required. It can be directly fetched from the SRB at each stage. But for the last systolic stage, we need a dedicated systolic structure to generate the required delayed input samples. This sharing methodology saves $(N-1)^2$ registers where N indicates the order of the filters. Furthermore, the critical path delay in the proposed architecture can be reduced by a novel hybrid systolic rearrangements of delay elements. To get such systolic rearrangements, Eq. (17) can be further expressed as

$$\begin{aligned} Y(z_1, z_2) = (z^{-(M-c)}(z^{-(M-c)}(\ldots z^{-(M-c)}(z^{-(M-c)}(X_N(z_1, z_2)z^{-c} + X_{N-1}(z_1, z_2))z^{-c} \\ + \cdots + X_2(z_1, z_2))z^{M} + X_1(z_1, z_2))z^{-c} + X_0(z_1, z_2)) \end{aligned} \tag{19}$$

To map (19) into the structural form, the M length SRB proposed at each systolic stage can be divided into two parts of length $M-c$ and c each. For proper operation of the systolic structure, the $M-c$ length SRB can be connected to the input side and the remaining c lengths SRB can be connected at output side at each systolic stage to get the desired shifting. The overall proposed architecture for 8×8 order adaptive filters is shown in Fig. 5. In the proposed architecture, we have considered $c = 1$ so, $M-1$ length SRB is connected to each systolic stage at the input side. Since SRB contains all required delayed inputs, all required inputs are directly fetched from the SRB blocks. At the output side, outputs of internal blocks are delayed and added in a systolic manner and each systolic stage contains one delay element since $c = 1$. For simplicity in structure, fetched input samples are subdivided into the group of four input samples and fed to the corresponding SIPSE blocks. Similarly, the filter coefficients are also divided into the group of four and the least significant bit (LSB) of each filter coefficient is fed to the corresponding SIPSE block. Each SIPSE block is followed by one SAU block. To access the value of SIPSE, bits of each filter coefficients are shifted one bit right with each clock in a circular-shift manner. Since bit-width of filter coefficients is L, output computation of each subblock requires L clock cycles. Moreover, SAU is followed by one-stage pipelining. After the pipelining stage, outputs of each internal block are added together. Further, the computed output of each internal block is delayed and added as per (19) to calculate the final output as depicted in Fig. 5.

In the proposed architecture (Fig. 5), the TDDLMS algorithm is used for the weight updation of the filter. As per the TDDLMS algorithm (7a), an adder/subtractor is required to compute the error signal. Further, two multipliers are required to perform multiplication operations $2\mu e(k-2)$ and $2\mu\, e(k-2)\, x(m_2-i, n_2-j)$. In the proposed design, for the convenience in multiplication with 2μ, the convergence factor μ is selected nearest to 2^{-v} where v is an arbitrary number so, the multiplication can be performed just by shifting the bit of error signal $e(k-2)$. In the proposed design, we have taken the convergence factor $\mu = 1/64$. Further, the barrel shifter-based multiplication is proposed for multiplication of error function $2\mu e(k-2)$ and input signal $x(m_2-i, n_2-j)$, i.e., $2\mu\, e(k-2)\, x(m_2-i, n_2-j)$. To perform the barrel shifter-based multiplications, the sign and magnitude of $2\mu e(k-2)$ are separated and the magnitude part is fed to the control logic unit that generates the control word X for barrel shifters. The logic for the control unit is shown in Fig. 6. Since multiplicand (magnitude part of $2\mu\, e(k-2)$) of all weight updation units is same, control word X for

Fig. 6 Logic for control word

```
if $D_7$=1 then Control X='000'
else if $D_6$=1 then Control X='001'
else if $D_5$=1 then Control X='010'
else if $D_4$=1 then Control X='011'
else if $D_3$=1 then Control X='100'
else if $D_2$=1 then Control X='101'
else if $D_1$=1 then Control X='110'
otherwise  Control X='111'
```

all barrel shifters is same. Furthermore, two-stage pipelining is used in the architecture which generated the error in each iteration is delayed by two units than the current input signal. For the weight updation as per (7a), the input samples which are fed to the barrel shifters are also delayed by two units. These delayed input samples already exist in SRB block, so, it can be directly fetched from the SRB block.

4 Hardware Time Complexity and Performance Comparison

The proposed architecture (Fig. 5) is unified structure of SRB block, DA block and weight updation unit. For designing of $N \times N$ size filter, N SRB blocks are required to generate delayed input samples. Each SRB contains the $M - 1$ registers. For the generation of the desired input samples at last systolic stage, a dedicated systolic structure of $N - 1$ register is required. SRB blocks are followed by the DA block. In each DA block, N size of MAC is subdivided into the S parallel small block required for the DA decomposition where $S = N/4$. In the proposed structure, the size of filter, i.e., $N = 8$. Each small block consists of one SIPSE block and one SAU block. The SIPSE block is comprised of the four AND gates, two OR gates and two adders. Further, one-stage pipelining is used which requires $N^2/4$ registers in the overall architecture. After pipelining, output of SAUs is added using the $\log_2(N/4)$ stage tree adder which requires the $(N/4) - 1$ adders. For the weight updation, overall structure requires N^2 barrel shifter. Theoretical hardware and time complexities details are given in Table 1.

4.1 Time Complexity

The proposed architecture contains one control register for the synchronization. It generates the one synchronization bit to enable the filter coefficient register and SAU block with each clock and rest part of the architecture after L clock cycle. Therefore, in each iteration, the SIPSE and SAU blocks are processed for the L times. Since two-stage pipelines are used in the architecture, the critical path of the architecture may be divided

Table 1 Theoretical comparison of hardware and time complexities of the proposed architecture and the conventional architecture and the earlier reported architectures [5, 10, 15, 32]

Design	Throughput	Multipliers	Adders	Registers
Conventational LMS-based	$\frac{1}{3T_M+[3+2\log_2 N]T_{PA}}$	$2N^2+1$	$2N^2+N+2$	$M(N-1)+N(2N-1)$
Hadhoud [5]	$\frac{1}{3T_M+(2N)T_{PA}}$	$2N^2+1$	$2N^2+N+2$	$M(N-1)+N(2N-1)$
Kimijima [10] (LDLMS-based)	$\frac{1}{T_M+2T_{PA}}$	$2N^2+1+3\delta$	$2N^2+N+3\delta$	$(3/2)N(N+1)+MN$
Matsubara [15] (DLMS-based)	$\frac{1}{T_M+(N+1)T_{PA}}$	$2N^2+1$	$2N^2+N+2$	$(N-1)(M+N)+N^2+2$
Van [32] (TDDLMS-based)	$\frac{1}{T_M+T_{PA}}$	$2N^2+1$	$N(N+2)+2$	$(N-1)(M+2N)+N^2+D$
Proposed (TDDLMS-based)	$\frac{1}{L(T_{CLA}+T_{PA}+T_{AND}+T_{SAU})}$	0	$\frac{5}{2}N^2$	$N(M+N/4)+1$

T_{PA} Delay of parallel adder, T_{AND} delay of AND gate, T_{SAU} delay of SAU block, T_M delay of multiplier in three time zones namely first stage, second stage and third stage. The first stage contains the SIPSE and SAU blocks and its delay is given as $L(T_{\text{SIPSE}} + T_{\text{SAU}})$ where T_{SIPSE} and T_{SAU} are the delays of SIPSE and SAU blocks, respectively. second stage delay is combinational path delay between the first stage pipelining and second stage pipelining and its delay will be $(\log_2(N/4) + 2)T_{\text{PA}}$ where T_{PA} is delay of the parallel adder. The last stage, i.e., third stage, contains the weight updation architecture and its delay is $T_{\text{control\_unit}} + T_{\text{Barrel-shifter}} + T_{\text{adder}}$ where $T_{\text{control\_unit}}$, $T_{\text{Barrel - shifter}}$ and T_{adder} are the delays of control unit, barrel shifter and an adder, respectively. The first stage offers the maximum delays, therefore, first stage is the critical path of the architecture.

4.2 Performance Comparison

Hardware and time complexities of the proposed and the earlier reported architecture are given in Table 2. The reported architectures [5, 10, 15, 32] have required multipliers for the multiplication operation. The multiplier acquires most of the area in VLSI design while no multiplier is required in the proposed architecture. The graphical comparison is depicted in Fig. 7a. The proposed architecture requires the slightly greater number of adders than architecture as reported above, and it can be observed from Fig. 7b. Further, the proposed architecture requires the lesser number of registers than the pipelined architecture of 2-D adaptive FIR filter reported above depicted in Fig. 7c. The details of other comparison parameters such as data arrival time (DAT) area, power, throughput, area-delay product (ADP) and energy-delay product (EDP) are discussed in Table 2.

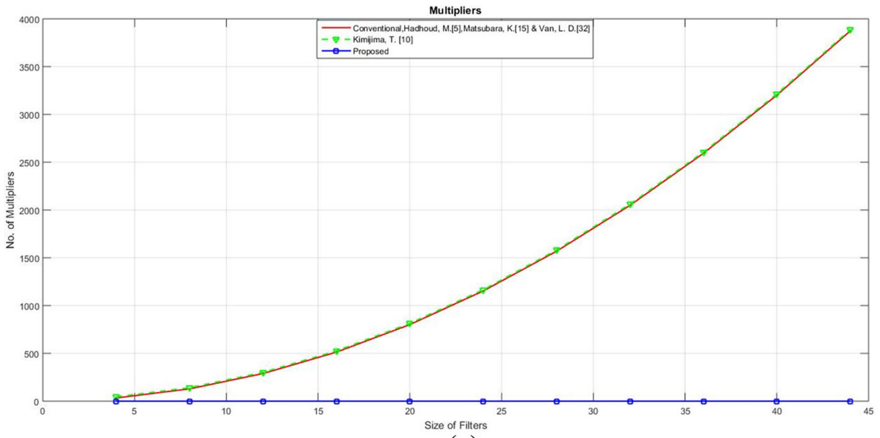
We have coded the architecture as reported above for the filter size 4×4 and 8×8 and the proposed architecture for same filter size, respectively, in Verilog-HDL. The coded designs are synthesized with the help of Synopsys Design Compiler using the 90-nm TSMC standard CMOS library. The results (DAT (Data arrival time), area and power) obtained from the synthesis are reported in Table 2. The comparative graphs are plotted in Fig. 8 which shows the improvement in terms of area, power, ADP and EDP of the proposed architecture than the earlier reported architectures. The proposed architecture for filter size 4×4 requires 59%, 55.87%, 53% and 41% lesser area, power, ADP and EDP, respectively, and for filter size 8×8 requires 54%, 48.19%, 55% and 50% lesser area, power, ADP and EDP, respectively, than the architecture in [32] as shown in Fig. 8. It has 59.51% lesser ADP and 53.15% lesser EDP than the architecture of [15] for filter size 8×8.

Remark 1 In the application domain, measurement of real-world parameters and real-time operations is big challenge for any practical system. The best example of such type of systems is power system where we remotely measure the system parameters. The remote measurement devices introduce delays, usually modeled as memory [31], into the control loop and are thus potential threats to the power system stability [13, 21, 31]. Moreover, when we model the real-world systems with mathematical approximations [2, 31], then, due to approximation errors, uncertainty occurs in the system which further degrades the performance and affects the system stability severely [31]. We find that the adaptive filters may turn out to be more suitable solutions for such type

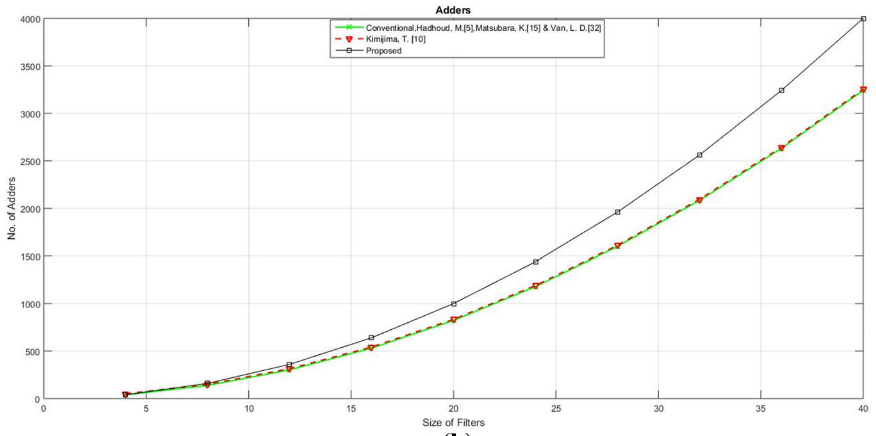
Table 2 Comparison of synthesis results of the proposed architecture and architectures reported in [10, 15, 32] for the filter size $4 \times 4$and 8×8 with the help of Synopsys Design Complier using the 90-nm standard cell library. Bit-width of the filter coefficients and input samples is taken 8

Designs	Filter size (N)	DAT (ns)	Area (μm^2)	Power consumption (mw)	EDP (mw $\times$ ns)	ADP ($\mu m^2 \times$ ns)
Kimijima [10] (LDLMS-based)	4	3.07	142,278.003	3.71	11.3897	436,793.469
	8	3.21	476,631.329	12.79	41.0559	1,529,986.57
Matsubara [15] (DLMS-based)	4	3.59	108,967.753	2.89	10.3751	391,194.233
	8	3.97	411,197.48	10.99	43.6303	1,632,454
Van [32] (TDDLMS-based)	4	2.71	127,197.03	3.47	9.4037	344,703.951
	8	3.11	469,714.778	12.99	40.3989	1,460,812.96
Proposed (TDDLMS-based)	4	3.09	51,971.11	1.771	5.47,239	160,590.73
	8	3.13	211,128.196	6.53	20.4389	660,831.253

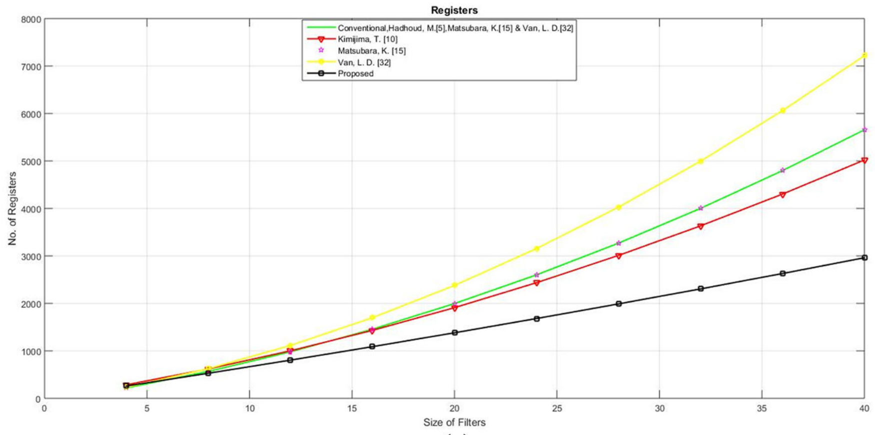
Power is estimated at the 20 MHz frequency. In [10, 15, 32], for multiplication Wallace tree multiplier used



(a)



(b)



(c)

Fig. 7 Complexities comparison of the proposed architecture and conventional architecture and earlier reported architectures [5, 10, 15, 32] with respect to size of the filters. **a** Multipliers requirement, **b** adders requirement, **c** register requirements

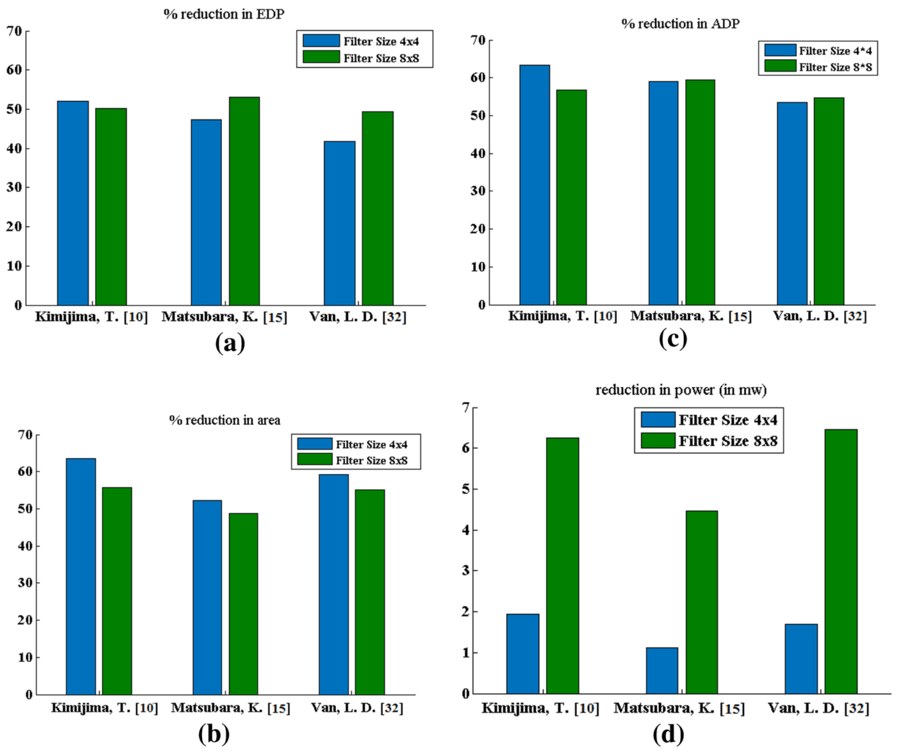


Fig. 8 Reduction in EDP, area, ADP and power requirement of the proposed architecture in comparison with the architectures proposed in [10, 15, 32]. **a** Percentage reduction in EDP, **b** percentage reduction in area, **c** percentage reduction in ADP, **d** reduction in power requirement (in mw)

of applications. Readers are encouraged to consider the applications of 2-D adaptive filtering in power systems which may be an interesting problem for future work.

5 Conclusions

In this paper, we have proposed an efficient architecture for 2-D adaptive FIR filter using DA algorithm. Hardware-based architecture has been designed for the computation of DA-LUT expressions. A MAC unit is designed using the fewer adders and logic gates. Memory sharing has been proposed in the architecture that reduces the memory requirement in a large number. Further, two-stage pipelining is used in the architecture to limit the critical path delay. Furthermore, for the higher-order filters, MAC is subdivided into the small-order MAC for DA decomposition that reduces the hardware complexities for the higher-order filter and limits the critical path equivalent to the small-order filter. The convergence factor μ is selected nearest to 2^{-v} where v is an arbitrary number, therefore, by LSB truncation method, it can be multiplied with the error. Moreover, for weight updation, barrel shifter-based structure has been used.

Since in all weight updation block multiplicands are common, i.e., $2\mu e(k-2)$ only one control unit is required for all barrel shifters.

Theoretical and ASIC synthesis comparison shows that the proposed architecture is area, power and throughput efficient than the earlier reported architectures. It also requires the lesser number of registers. Multipliers that acquire large are in the VLSI design are required in large numbers in case of the architecture [5, 10, 15, 32], while the proposed architecture is multiplier-less. The ASIC synthesis shows that the proposed architecture reduces the area, power, ADP and EDP by 54%, 48.19%, 55% and 49%, respectively, as compared with the existing architecture for filter as discussed [10] for filter size 8×8. The ADP and EDP of the proposed architecture also require 59.51% and 53.15% lesser than the architecture [15].

Acknowledgements The authors would like to give their sincere thanks to the Editor-in-Chief and to the anonymous reviewers for their valuable comments and suggestions to improve the quality of this paper. In this paper, all data are provided in full in the results section.

References

1. H. Cho, R. Priemer. Automatic step size adjustment of the two-dimensional LMS algorithm. in *Proceedings of 1994 37th Midwest Symposium on Circuits and Systems, Lafayette*, LA, USA, vol. 2, pp. 864–867 (1994). https://doi.org/10.1109/mwscas.1994.518950
2. I. Dassios, A practical formula of solutions for a family of linear non-autonomous fractional nabla difference equations. J. Comput. Appl. Math. **339**, 317–328 (2018). https://doi.org/10.1016/j.cam.2020.113032
3. S.C. Douglas, Q. Zhu, K.F. Smith, A pipelined LMS adaptive FIR filter architecture without adaptation delay, in *IEEE Transactions on Signal Processing*, vol. 46, no. 3, pp. 775–779 (1998). https://doi.org/10.1109/78.661345
4. R.S. Grover, W. Shang, Q. Li, A faster distributed arithmetic architecture for FPGAs. in *International Symposium on FPGAs* (FPGA'02) Monterey, CA, USA, pp. 31–39 (2002). https://doi.org/10.1145/503048.503054
5. M.M. Hadhoud, D.W. Thomas, The two-dimensional adaptive LMS (TDLMS) algorithm. IEEE Trans. Circuits Syst. **35**(5), 485–494 (1988). https://doi.org/10.1109/31.1775
6. A. Harada, K. Nishikawa, H. Kiya, A pipelined architecture for the normalized LMS adaptive digital filters. in *1998 IEEE Asia-Pacific Conference on Circuits and Systems. Microelectronics and Integrating Systems. Proceedings* (Cat. No.98EX242), Chiangmai, Thailand, pp. 73-76 (1998). https://doi.org/10.1109/apccas.1998.743661
7. S. Haykin, *Adaptive Filter Theory* (Prentice-Hall, Englewood Cliffs, 1986)
8. H. Jiang, L. Liu, P.P. Jonker, D.G. Elliott, F. Lombardi, J. Han, A high-performance and energy-efficient FIR adaptive filter using approximate distributed arithmetic circuits. IEEE Trans. Circuits Syst. I Regul. Pap. **66**(1), 313–326 (2019). https://doi.org/10.1109/TCSI.2018.2856513
9. M.T. Khan, R.A. Shaik, Optimal complexity architectures for pipelined distributed arithmetic-based LMS adaptive filter. IEEE Trans. Circuits Syst. I Regul. Pap. **66**(2), 630–642 (2019). https://doi.org/10.1109/TCSI.2018.2867291
10. T. Kimijima, K. Nishikawa, H. Kiya, Pipelining of 2-dimensional adaptive filters based on the LDLMS algorithm. in *Proceedings of the 1998 IEEE International Symposium on Circuits and Systems*, *ISCAS '98* (Cat. No.98CH36187), Monterey, CA, vol. 5, pp. 190–193 (1998). https://doi.org/10.1109/iscas.1998.694440
11. P. Kumar, P.C. Shrivastava, M. Tiwari, A. Dhawan, ASIC implementation of area-efficient, high-throughput 2-D IIR filter using distributed arithmetic. Circuits Syst Signal Process **37**, 2934–2957 (2018). https://doi.org/10.1007/s00034-017-0698-z
12. P. Kumar, P.C. Shrivastava, M. Tiwari, G.R. Mishra, High-throughput, area-efficient architecture of 2-D block FIR filter using distributed arithmetic algorithm. Circuits Syst. Signal Process. **38**, 1099–1113 (2019). https://doi.org/10.1007/s00034-018-0897-2

13. M. Liu, I. Dassios, G. Tzounas, F. Milano, Stability analysis of power systems with inclusion of realistic-modeling WAMS delays. IEEE Trans. Power Syst. **34**(1), 627–636 (2019). https://doi.org/10.1109/TPWRS.2018.2865559
14. W.S. Lu, A. Antoniou, *Two-Dimensional Digital Filters* (Marcel Dekker, New York, 1992)
15. K. Matsubara, K. Nishikawa, H. Kiya, 2-D pipelined adaptive filters based on 2-D delayed LMS algorithm (special section of papers selected from ITC-CSCC'96). IEICE Trans. Fund. Electron. Commun. Comput. Sci. **80**, 1009–1014 (1999)
16. K. Matsubara, K. Nishikawa, H. Kiya, Pipelined LMS adaptive filter using a new look-ahead transformation. IEEE Trans. Circuits Syst. II: Analog Digit. Signal Process. **46**(1), 51–55 (1999). https://doi.org/10.1109/82.749082
17. P.K. Meher, S.Y. Park, Critical-path analysis and low-complexity implementation of the LMS adaptive algorithm. IEEE Trans. Circuits Syst. I Regul. Pap. **61**(3), 778–788 (2014). https://doi.org/10.1109/TCSI.2013.2284173
18. P.K. Meher, S. Y. Park, High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic. in *2011 IEEE/IFIP 19th International Conference on VLSI and System-on-Chip*, Hong Kong, pp. 428–433 (2011). https://doi.org/10.1109/vlsisoc.2011.6081621
19. W.B. Mikhael, S.M. Ghosh, Two-dimensional block adaptive filtering algorithms. in *[Proceedings] 1992 IEEE International Symposium on Circuits and Systems*, San Diego, CA, USA, vol. 3, pp. 1219–1222 (1992). doi: 10.1109/ISCAS.1992.230305
20. W.B. Mikhael, S.M. Ghosh, Two-dimensional block adaptive filtering algorithms with optimum convergence factors. IEEE Trans. Circuits Syst. II: Analog Digit. Signal Process. **42**(8), 505–515 (1995). https://doi.org/10.1109/82.404072
21. F. Milano, I. Dassios, Small-signal stability analysis for non-index 1 Hessenberg form systems of delay differential-algebraic equations. IEEE Trans. Circuits Syst. I Regul. Pap. **63**(9), 1521–1530 (2016). https://doi.org/10.1109/TCSI.2016.2570944
22. B.K. Mohanty, P.K. Meher, S.K. Patel, LUT optimization for distributed arithmetic-based block least mean square adaptive filter. in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 5, pp. 1926–1935 (2016). https://doi.org/10.1109/tvlsi.2015.2472964
23. B.K. Mohanty, P.K. Meher, S.K. Singhal, M.N. Swamy, A high performance VLSI architecture for reconfigurable FIR using distributive arithmetic. J. Integr. VLSI J. **54**, 37–46 (2016). https://doi.org/10.1016/j.vlsi.2016.01.006
24. M. Ohki, S. Hashiguchi, Two-dimensional LMS adaptive filters. IEEE Trans. Consum. Electron. **37**(1), 66–73 (1991). https://doi.org/10.1109/30.73648
25. S.Y. Park, P.K. Meher, Efficient FPGA and ASIC Realizations of a DA-Based Reconfigurable FIR Digital Filter. IEEE Trans. Circuits Syst. II Express Briefs **61**(7), 511–515 (2014). https://doi.org/10.1109/TCSII.2014.2324418
26. A. Peled, B. Liu, A new hardware realization of digital filters. IEEE Trans. Acoust. Speech Signal Processing **22**(6), 456–462 (1974). https://doi.org/10.1109/TASSP.1974.1162619
27. M.A. Sid-Ahmed, *Image Processing: Theory, Algorithms and Architectures* (McGraw-Hill, New York, 1995)
28. T. Soni, J.R. Zeidler, W.H. Ku, Performance evaluation of 2-D adaptive prediction filters for detection of small objects in image data. IEEE Trans. Image Process. **2**(3), 327–340 (1993). https://doi.org/10.1109/83.236534
29. A. Tan., S. T. Chen, Image enhancement with 2-D adaptive LMS-based recursive filters. in *Proceedings of International Conference on Signal Processing, Applications and Technology*, vol. 2. pp. 1068–1072 (1995)
30. A. Tiwari., P. Kumar., M. Tiwari, High throughput adaptive block FIR filter using distributed arithmetic. in *2016 1st India International Conference on Information Processing (IICIP)*, Delhi, pp. 1–6 (2016). https://doi.org/10.1109/iicip.2016.7975385
31. M. Tiwari, A. Dhawan, An LMI approach to optimal guaranteed cost control of uncertain 2-D discrete shift-delayed systems via memory state feedback. Circuits Syst. Signal Process. **31**, 1745–1764 (2012). https://doi.org/10.1007/s00034-012-9410-5
32. L.-D. Van., W.-S. Feng, Efficient systolic architectures for 1-D and 2-D DLMS adaptive digital filters. in *2000 IEEE Asia-Pacific Conference on Circuits and Systems. Electronic Communication Systems*. (Cat. No.00EX394), Tianjin, China, pp. 399–402 (2000). https://doi.org/10.1109/apccas.2000.913519

33. T. Wang, C.-L. Wang, A new two-dimensional block adaptive FIR filtering algorithm and its application to image restoration. IEEE Trans. Image Process. **7**(2), 238–246 (1998). https://doi.org/10.1109/83.661002
34. S.A. White, Applications of distributed arithmetic to digital signal processing: a tutorial review. IEEE ASSP Mag. **6**(3), 4–19 (1989). https://doi.org/10.1109/53.29648
35. H. Youlal, M. Janati-I, M. Najim, Two-dimensional joint process lattice for adaptive restoration of images. IEEE Trans. Image Process. **1**(3), 366–378 (1992). https://doi.org/10.1109/83.148609

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.