ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Design and Analysis of High Performance Fir Filter and Low Memory Adder

Bandaru Rajasekhar
PG Scholar,
Dept of ECE,
Intell Engineering College,
Anantapuramu, AP, India.

S.Ismail Saheb
Assistant Professor,
Dept of ECE,
Intell Engineering College,
Anantapuramu, AP, India.

Abstract:

Adaptive filtering constitutes an important class of DSP algorithms employed in several hand held mobile devices for applications such as echo cancellation, signal de-noising, and channel equalization. This brief presents a novel pipelined architecture or low-power, high-throughput, and low-area implementation of adaptive filter based on distributed arithmetic (DA). The throughput rate of the proposed design is significantly increased by parallel lookup table (LUT) update and concurrent implementation of filtering and weight-update operations. The conventional adder-based shift accumulation for DA-based inner-product computation is replaced by conditional signed carry-save accumulation in order to reduce the sampling period and area complexity.

Reduction of power consumption is achieved in the proposed design by using a fast bit clock for carry-save accumulation but a much slower clock for all other operations. It involves the same number of multiplexors, smaller LUT, and nearly half the number of adders compared to the existing DA-based design. From synthesis results, it is found that the proposed design consumes13% less power and 29% less area-delay product (ADP) over our previous DA-based adaptive filter in average for filter lengths N=16and32. Compared to the best of other existing designs, our proposed architecture provides 9.5 times less power and 4.6 times less ADP.

Keywords:
Adaptive Filter, Circuit Optimization, Distributed Arithmetic (DA), Least Mean Square (LMS) Algorithm.

I. INTRODUCTION:

Adaptive filters find extensive use in many signal processing applications such as channel equalization, echo cancellation, noise cancellation.

The finite impulse response (FIR) filters whose weights are updated by the famous Widrow-Hoff least mean square (LMS) algorithm is the most popularly used adaptive filter not only due to its simplicity but also due to its satisfactory convergence performance. The direct form configuration on the onward path of the FIR filter results in along critical path due to an inner product computation to obtain a filter output. Consequently, it is required to reduce the critical path of the structure if the input signal has high sampling rate. By reducing the critical path of the structure, thereby, the critical path could not exceed the sampling period. Distributed arithmetic (DA) is so named because it performed arithmetic operation. DA is bit serial computation in nature and it eliminates the need for hardware multipliers and is capable of implementing large order filters with very high throughput.

A lot of study has been done to implement the DA based adaptive FIR filter for area efficient design, the multiplier-less distributed arithmetic (DA) based technique has achieved plenteous popularity for its high throughput, but it results are increased in cost-effective, area and time efficient computing structures. DA based hardware efficient adaptive FIR filter inner product has been suggested by Allred et al. using two separate lookup tables (LUTs) Filtering lookup table and Auxiliary lookup table for filtering and weight updating module. Later, Guo and DeBrunner, have improved the design structure in by using only one lookup table instead of two LUTs for both filter and weight updating module. On the other hand, the design process in require more cycles for lookup table (LUT) update for each new sample, hence it do not support high sampling rate. Meher and Park have improved the design with low adaptation delay for high speed DA based adaptive filter. In a recent paper, Meher and Park proposed a new DA based adaptive filter architecture for low power, low area and high throughput with very low adaptation delay. This brief proposes an adaptive FIR filter using distributed arithmetic for area efficient design.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

High Throughput is achieved by using a parallel lookup table update and equivalent implementation of filtering and weight-updating operations. The conditional signed carry saved accumulation for DA-based inner product computation is designed by using 10 transistor full adder based carry saved accumulation of shift accumulation as shown in Fig.1. The use of the proposed design helps to reduce the area complexity and power consumption. During each cycle, the LMS algorithm computes a filter output and an error value that is equal to the difference between the current filter output and the desired response. The estimated error is then used to update the filter weights in every training cycle. The weights of LMS adaptive filter during the nth iteration are updated according to the following equations:

$$w(n+1) = w(n) + \mu \cdot e(n) \cdot x(n) \quad (1)$$

Where,

$e(n) = d(n) - y(n)$

$y(n) = w^{qT}(n) \cdot x(n)$

The input vector x (n) and the weight vector w (n) at the nth training iteration are respectively given by

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

$$w(n) = [\omega_0(n), \omega_1(n), \ldots, \omega_{N-1}(n)]^T \quad (2)$$

d (n) is the desired response, and y(n) is the filter output of the nth iteration. e (n) denotes the error computed during the nth iteration, which is used to update the weights, μ is the convergence factor, and N is the filter length. In the case of pipelined designs, the feedback error e (n) becomes available after certain number of cycles, called the adaptation delay.‖ The pipelined architectures therefore use the delayed error e (n−m) for updating the current weight instead of the most recent error, where mis the adaptation delays. The weight-update equation of such delayed LMS adaptive filter is given by

$$w(n+1) = w(n) + \mu \cdot e(n-m) \cdot x(n-m) \quad (3)$$

II.PROPOSED DA-BASEDAPPROACH FOR INNER-PRODUCT COMPUTATION

The LMS adaptive filter, in each cycle, needs to perform an inner-product computation which contributes to the most of the critical path. For simplicity of presentation, let the inner product of be given by

$$y = \sum_{k=0}^{N-1} \omega_k \cdot x_k \quad (4)$$

Where wk and x_k for $0 \le k \le N-1$ form the N-point vectors corresponding the current weights and most recent N−1input, respectively assuming L to be the bit width of the weight, each component of the weight vector may be expressed in two's complement representation

$$\omega_k = -\omega_{k0} + \sum_{l=1}^{L-1} \omega_{kl} \cdot 2^{-l}$$

Where w_{kl} denotes the l^{th} bit of w_k. substituting, we can write in an expanded form

$$y = -\sum_{k=0}^{N-1} x_k \cdot \omega_{k0} + \sum_{k=0}^{N-1} x_k \cdot [\sum_{l=0}^{L-1} \omega_{kl} \cdot 2^{-l}]$$

To convert the sum-of-products form of into a distributed form, the order of summations over the indices k and l in can be interchanged to have

$$y = [\sum_{l=1}^{L-1} 2^{-l} \cdot y_l] - y_0 \quad \text{where } y_l = \sum_{k=0}^{N-1} x_k \cdot \omega_{kl} \quad (7)$$

and the inner product given by can be computed as

slices of vector ware fed one after the next in the least significant bit (LSB) to the most significant bit (MSB) order

$$y = -\sum_{b=0}^{N-1} x_b \cdot \omega_{b0} + \sum_{l=1}^{L-1} 2^{-l} \cdot [\sum_{b=0}^{N-1} x_b \cdot \omega_{bl}] \quad (8)$$

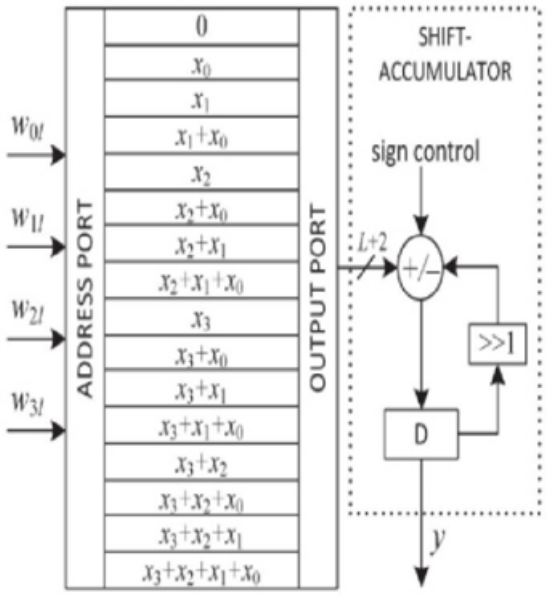

Fig.1. Conventional DA-based implementation of four-point inner product.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

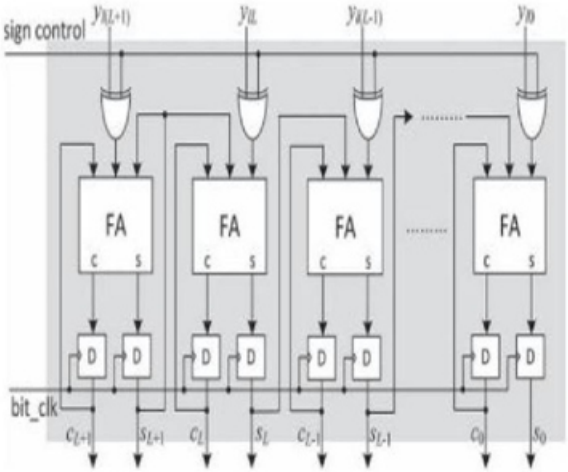

Fig.2. Carry-save implementation of shift accumulation.

Since any element of the N-point bit sequence {wkl for 0≤k≤N−1}can either be zero or one, the partial sum yl for l=0,1,...,L−1can have 2N possible values. If all the 2N sequence {wkl}as address bits for computing the inner product. The inner product of can therefore be calculated in L cycles of shift accumulation, followed by LUT-read operations corresponding to L number of bit slices{wkl} for 0≤ l ≤ L−1. Since the shift accumulation in Fig.2 involves significant critical path, we perform the shift accumulation using carry-save accumulator, as shown in Fig2. The bit to the carry-save accumulator. However, the negative (two's complement) of the LUT output needs to be accumulated in case of MSB slices. Therefore, all the bits of LUT output are passed through XOR gates with a sign-control input which is set to one only when the MSB slice appears as address. The XOR gates thus produce the one's complement of the LUT output corresponding to the MSB slice but do not affect the output for other bit slices. Finally, the sum and carry words obtained after L clock cycles are required to be added by a final adder (not shown in the fig.3), and the input carry of the final adder is required to be set to one to account for the two's complement operation of the LUT output corresponding to the MSB slice. The content of the kth LUT location can be expressed as

$$c_k = \sum_{j=0}^{N-1} x_j \cdot k_j \qquad (9)$$

Where kj is the (j+1) th bit of N-bit binary representation of integer k for 0 ≤ k ≤ 2 N−1. Note that ck for 0 ≤ k ≤ 2N−1 can be pre computed and stored in RAM-based LUT of 2N words. However, instead of storing 2N words in LUT, we store (2N−1) word s in a DA table of 2N−1 registers. An example of such a DA table for N=4is shown in Fig.4 It contains only 15 registers to store the pre computed sums of input words. Seven new values of ck are computed by seven adders in parallel.

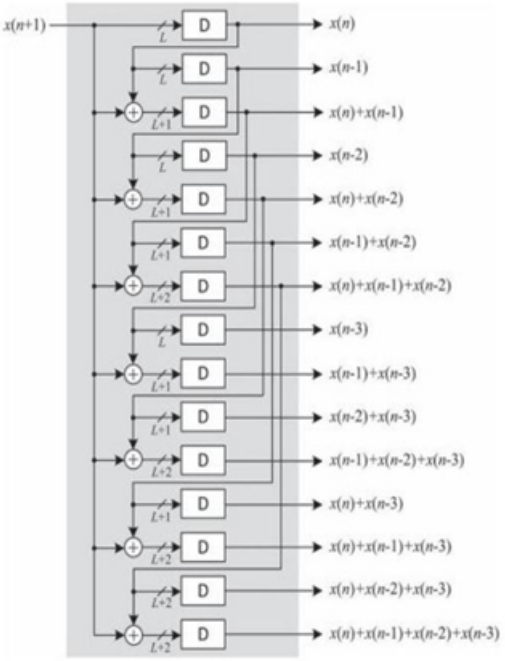

Fig.3. DA table for generation of possible sums of input samples. (9)

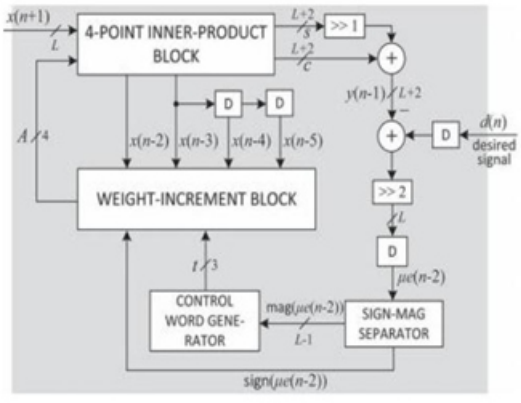

Fig.4. Proposed structure of DA-based LMS adaptive filter of filter length N=4.

III.PROPOSED DA BASED ADAPTIVE FILTERSTRUCTURE:

The computation of adaptive filters of large orders needs to be decomposed into small adaptive filtering blocks since DA based

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

implementation of inner product of long vectors requires a very large LUT. Therefore, we describe here the proposed DA-based structures of small- and large-order LMS adaptive m filters separately in the next two sections.

A. Proposed Structure of Small-Order Adaptive Filter:

The proposed structure of DA-based adaptive filter of length N=4is shown in Fig. 4. It consists of a four-point inner product block and a weight-increment block along with additional circuits for the computation of error value e(n) and control word t for the barrel shifters.

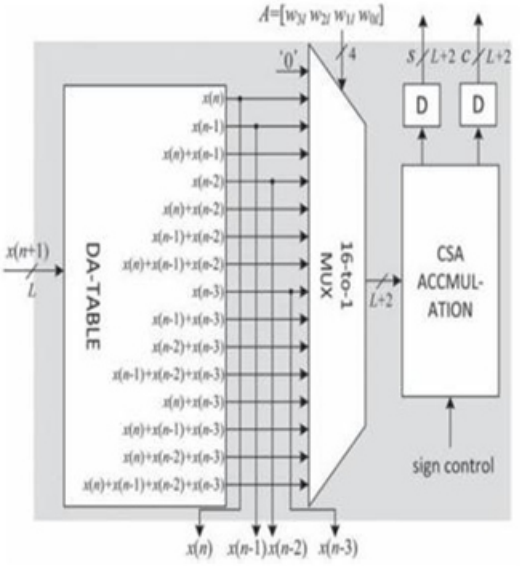

Fig.5. Structure of the four-point inner-product block.

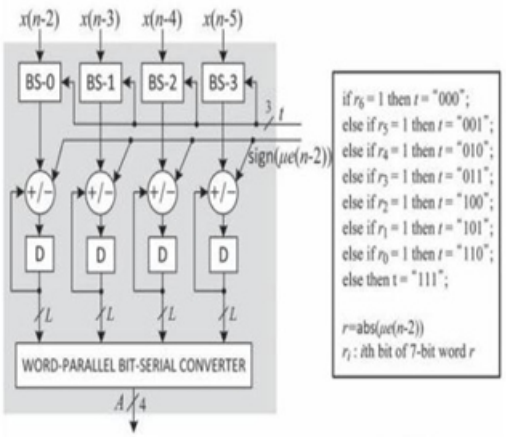

Fig.6. Structure of the weight-increment block for N=4. And Logic used for generation of control word t for the barrel shifter for L=8.

Fig.6. Structure of the weight-increment block for N=4. And Logic used for generation of control word t for the barrel shifter for L=8.

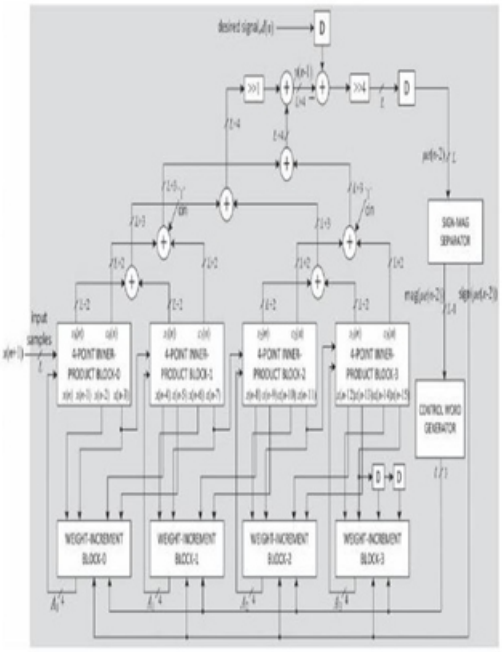

Fig.7. Proposed structure of DA-based LMS adaptive filter of length N=16andP=4.

The four-point inner-product block [shown in Fig.5 includes a DA table consisting of an array of 15 registers which stores the partial inner products yl for $0 < l \leq 15$ and a 16: 1 multiplexor (MUX) to select the content of one of those registers. Bit slices of weights A= {w3l w2l w1l w0l} for $0 \leq l \leq L-1$ are fed to the MUX as control in LSB-to MSB order, and the output of the MUX is fed to the carry-save accumulator (After L bit cycles, the carry-save accumulator shift accumulates all the partial inner products and generates a sum word and a carry word of size (L+2) bit each. The carry and sum words are shifted added with an input carry —1‖ to generate filter output which is subsequently subtracted from the desired output d (n) to obtain the error e (n). As in the case, all the bits of the error except the most significant one are ignored, such that multiplication of input xk by the error is implemented by a right shift through the number of locations given by the number of leading zeros in the magnitude of the error. The magnitude of the computed error is decoded to generate the control word t for the barrel shifter. The logic used for the generation of control word t to be used for the barrel shifter is shown in Fig 6. The convergence factor μ is usually taken to be O (1/N). We have taken μ= 1/N. However, one can take μ as 2−i/N,

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

where i is a small integer. The number of shifts t in that case is increased by i, and the input to the barrel shifters is pre shifted by I locations accordingly to reduce the hardware complexity. The weight-increment unit [shown in Fig 7] for N=4 consists of four barrel shifters and four adder/subtract or cells. The barrel shifter shifts the different input values xk for k=0,1,..,N−1 by appropriate number of locations (determined by the location of the most significant one in the estimated error). The barrel shifter yields the desired increments to be added with or subtracted from the current weights. The sign bit of the error is used as the control for adder/sub tractor cells such that, when sign bit is zero or one, the barrel-shifter output is respectively added with or subtracted from the content of the corresponding current value in the weight register.

B.Proposed Structure of Large-Order Adaptive Filter:

The inner-product computation of (4) can be decomposed-into N/P (assuming that N=PQ) small adaptive filtering blocks 1 of filter length P as

$$y = \sum_{k=0}^{P-1} \omega_k \cdot x_k + \sum_{k=P}^{2P-1} \omega_k \cdot x_k \cdots + \sum_{k=N-P}^{N-1} \omega_k \cdot x_k \quad (10)$$

Each of these P-point inner-product computation blocks will accordingly have a weight-increment unit to update P weights. The proposed structure for N=16 and P=4 is shown in Fig 7. It consists of four inner-product blocks of length P=4, which is shown in Fig 7. The (L+2)-bit sums and carry produced by the four blocks are added by two separate binary adder trees. Four carry-in bits should be added to sum words which are output of four 4-point inner-product blocks. Since the carry words are of double the weight compared to the sum words, two carry-in bits are set as input carry at the first level binary adder tree of carry words, which is equivalent to inclusion of four carry-in bits to the sum words. Assuming that μ=1/N, we truncate the four LSBs of e(n) for N=16 to make the word length of sign-magnitude separator be L bit. It should be noted that the truncation does not affect the Performance of the adaptive filter very much since the proposed design needs the location of the most significant one of μ e (n).

IV. COMPLEXITY AND SYNTHESIS RESULTS

We have estimated the hardware and time complexities of the proposed design and the existing DA-based adaptive filters .

The proposed design uses two clocks, namely, the bit clock and the byte clock. The duration of the byte clock is the same as the sampling period. The bit clock is used in carry-save accumulation units and word parallel bit-serial converters, while the byte clock is used in the rest of the circuit. The duration of bit clock is given byTBC=4TM+TFA+TXOR+TD, where TM, TFA, TXOR, and TD are the delays of a 2 : 1 MUX, a full adder, an XOR gate, and aD flip-flop, respectively, and L is the bit width of inputs as well as coefficients. Note that4TMis the delay of a 16: 1 MUX, since it consists of four stages of 2: 1 MUXes. The duration of the sample period (byte clock) of the proposed design is L×TBC. The sample period of the design C1=2P + max (L,2P−1)+log2Q clock periods, where the duration of each clock period amounts to the sum of the input LUT update time (TR), the delay of three-stage barrel shifter (3TM) and (L+2)-bit addition time (TA) for weight update.Similarly, the sample period of the design C2=2P−1+log2Q+L+1 clock periods.

When P=Q=4 for filterlength 16 and L=8, C1 and C2 can be found to be 26 and 19, respectively. The designs need 26 and 19 clock cycles, respectively, to produce an output sample. However, since the proposed design gives one output per cycle, the throughput per unit time of the proposed design is found to be much higher than that of the designed.The clock period of the proposed design is slightly larger than that of the design in because the carry-save adder has additional delay of the XOR gate to deal with the signed input. However, the additional delay of an XOR gate marginally affects the overall clock period. In Table I, we have also listed the number of hardware components used by the proposed and existing designs.

The proposed design for N=4 involves 14 adders/sub tractors, 4 logarithmic barrel shifters (of three stages each), and 31 registers. The design in on the other hand involves 25 adders, 15 shifters, and 33 registers. The number of adders/shifters/registers in the weight-increment block is equal to N each in the proposed design, whereas that in the design in amounts to (2P −1) N/P, which increases rapidly with N. The proposed design has an adaptation delay of two clocks, one of which is after the addition of carry and sum words in the inner-product computation blocks and the other is after the error calculation, whereas the designed and have no such adaptation delay. The adaptation delay of two cycles, however, does not make noticeable degradation of the convergence performance.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

We have coded the proposed design and the existing designs in Verilog HDL and synthesized by Synopsys, Design Compiler using a 65-nm CMOS library for filter length N=16and32to find the area, time, and power complexities. The word length of input samples and weights are taken to be eight. We have shown the synthesis results in terms of area, data arrival time, minimum samplingperiod, throughput, power consumption, area-delay product (ADP), and energy per sample (EPS). From the synthesis results, we find that the proposed design has almost 6.3 times more throughput and 8.7 times less ADP over the design in average for filter lengths N=16and32. Moreover, the proposed design consumes 16.6 times less power than thedesigned. Since the design in generates the output sample every 3•2P−1+log2Qclock, it can be seen that its EPS is muchhigher than that of the proposed design. The proposed design also offers 4.2 times more throughput, 4.6 times less ADP, and 9.5 times less EPS than the design in [5]. Compared to the design in, the proposed design involves 13% less power consumption and 29% less ADP.

V. CONCLUSION:

We have suggested an efficient pipelined architecture for low-power, high-throughput, and low-area implementation of DA-based adaptive filter. Throughput rate is significantly enhanced by parallel LUT update and concurrent processing of filtering operation and weight-update operation. We have also proposed a carry-save accumulation scheme of signed partial inner products for the computation of filter output. From the synthesis results, we find that the proposed design consumes 13% less power and 29% less ADP over our previous DA-based FIR adaptive filter in average for filter lengths N=16and32. Compared to the best of other existing designs, our proposed architecture provides 9.5 times less power and 4.6 times less ADP. Offset binary coding is popularly used to reduce the LUT size to half for area-efficient implementation of DA, which can be applied to our design as well.

VI. REFERENCES:

[1]Sang Yoon Park, Member, IEEE, and Pramod Kumar Meher, Senior Member, IEEE, ―Low-Power, High-Throughput, and Low-Area Adaptive FIR Filter Based onDistributed Arithmetic‖, IEEE Transactions on Circuits and Systems—Ii: Express Briefs.Vol.60,No.6,June 2013.

[2]Basant K. Mohanty, Senior Member, IEEE, and Pramod Kumar Meher, Senior.Member, ―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, 2013.

[3]M.Vijaya Laxmi and P.Annapurna Bai, ―Design of 128-bit Kogge-Stone Low Power Parallel Prefix VLSI Adder forHigh Speed Arithmetic Circuits‖, International Journal of engineering and advanced technology, Issue , 2013.

[4]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. I, Reg. Papers, vol. 52, no. 7, pp. 1327–1337, Jul. 2005.

[5]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.

[6]R. Guo and L. S. DeBrunner, ―A novel adaptive filter implementation scheme using distributed arithmetic,