International Journal of Advanced Research in

Electronics, Communication & Instrumentation Engineering and Development

Volume: 1 Issue: 2 08-Feb-2014,ISSN_NO: 2347 -7210

Adaptive FIR Filter Using LMS Algorithm for an Area Efficient Design

R.Ranjitha[1] and R.Rajeswari[2]

PG Scholar, Dept of ECE, Dhaanish Ahmed college of Engineering, Tamil Nadu, India[1]

Associate Professor, Dept of ECE, Dhaanish Ahmed college of Engineering, Tamil Nadu, India[2]

krvranjitha@gmail.com[1], rrajeswari782003@gmail.com[2]

ABSTRACT— *This briefly presents a novel pipelined architecture for low-power and low-area implementation of Adaptive filter based on distributed arithmetic (DA). 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 multiplexers, smaller LUT, and nearly half the number of adders compared to the existing DA-based design. By changing the inner block we are going to achieve low area and low power .So our previous DA-based adaptive filter in average for filter lengths N=4and N=16 Implemented.*

Keywords— Adaptive filter, Distributed arithmetic, carry save adder.

I. INTRODUCTION

Adaptive filters are widely used in several digital signal processing applications. The tapped-delay line finite-impulse response (FIR) filter whose weights are updates by famous Widrow _Hoff least mean square (LMS) algorithm 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 forward path of the FIR filter results in a long critical path due to an inner-product computation to obtain a filter output [1]. Therefore when the input signal has a high sampling rate, hence it is used to reduce the critical path of the structure so that the critical path could not tolerate the sampling period. In the multiplier-less distributed arithmetic (DA)-based technique has gained substantial popularity for its high-throughput processing, which result in less cost and area efficient computing structure. This brief proposes a novel DA-based architecture for low-power, low-area and high-throughput pipelined implementation of adaptive filter with very low adaptation delay conventional adder-based shift accumulation is replaced by a conditional carry-save accumulation of signed partial linear products to reduce the sampling period. The bit-cycle period amounts to memory access time plus 1-bitfull-adder time (instead of ripple carry addition time) by carry-save accumulation [2]. The use of the proposed signal carry-save accumulation also helps to

reduce the area complexity and reduction of power consumption is achieved by using a fast bit clock for carry-save accumulation but a much slower clock for all other operations. The contributions of this brief are as follows reduction of power consumption is achieved by using a fast bit clock for carry-save accumulation but a much slower clock for all other operations. The use of the proposed signed carry-save accumulation also helps to reduce the area complexity of the proposed design. Cost - effective and area–time efficient computing structures. The existing design require a control unit for address generation and it is compared with proposed structure [3].Adaptive Signal Processing is concerned with the design analysis of the systems whose structure changes with respect to response of the incoming data. Application areas are similar to those of optimal signal processing but now the environment is changing, the signals are non stationary and or the parameters to be estimated are time-varying.

II. LMS ADAPTIVE ALGORITHMS

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 [3]. The estimated error is then used to update the filter weights in every cycle. The weights of LMS adaptive filter during the *n*th iteration are updated according to the following equations

$$\mathrm{w}(n+1) = \mathrm{w}(n) + \mu\, .e(n).\, \mathrm{x}(n) \qquad (1a)$$

Where

$$e(n) = d(n) - y(n) \qquad (1b)$$

$$y(n) = w_q \mathrm{T}(n).\, \mathrm{x}(n) \qquad (1c)$$

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

$$\mathrm{x}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]\mathrm{T} \qquad (2a)$$

$$\mathrm{w}(n) = [w0(n), w1(n), \ldots, wN-1(n)]\mathrm{T} \qquad (2b)$$

$d(n)$ 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, where μ is the convergence factor, and N is the filter length[5]. In this case pipelined designs, the feedback error $e(n)$ is available after certain number of cycles called as the “adaptation delay.” The pipelined architectures used for the delay error $e(n - m)$ for updating the current weight

instead of the most recent error, where m is the adaptation delay. Thus the weight-update equation of delayed LMS adaptive filter is given by

$$w(n+1) = w(n) + \mu.\, en - m) .x(n - m). \quad (3)$$

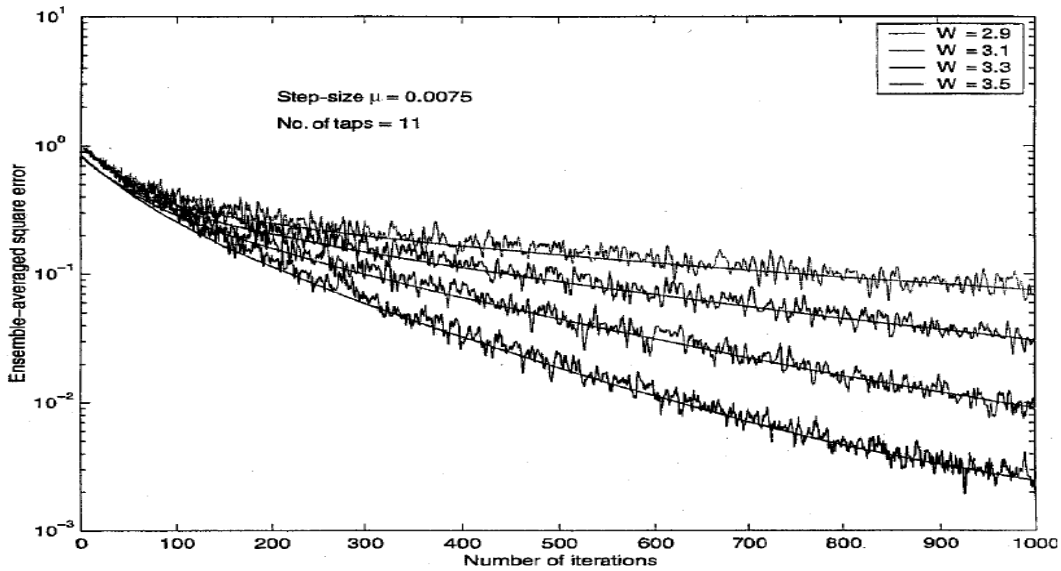

Fig 1.LMS adaptive algorithm

III. PROPOSED DA-BASED APPROACH FOR INNER PRODUCT COMPUTATION

The LMS adaptive filter, in every cycle, needs to perform an inner-product computation which is used in the most of the critical path. For simplicity the inner product of (1c) be given by

$$y = \sum_{k=0}^{n-1} w_{\,k}\, N_K \quad (4)$$

Where w_k and X_K or $0 \le k \le N - 1$ form the N-point vectors corresponding the current weights and most recent N − 1 input respectively [6]. Assuming L to be the width of the weight, the each component of the weight vector may be expressed in 2's Complement representation as

$$W_{\,k=} - W_{ko} + \sum_{l=1}^{L-1} W_{kl}\, 2^{-l} \quad (5)$$

Where w_{kl} denotes the l th bit of w_k Substituting (5), we can write (4) in an expanded form

Volume: 1 Issue: 2 08-Feb-2014,ISSN_NO: 2347 -7210

$$y = -\sum_{k=0}^{N-1} x_k\, w_{ko} + \sum_{k=0}^{N-1} x_k \,[\sum_{l-1}^{L-1} w_{kl}\, 2^{-l} \qquad (6)$$

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 $\{w_k\}$ or $0 \leq l \trianglelefteq L - 1$ as shown in Fig. 2 we perform the shift accumulation using carry-save accumulator, the carry save adder will avoid the unwanted addition and thus minimize the switching power dissipation and minimize area complexity.

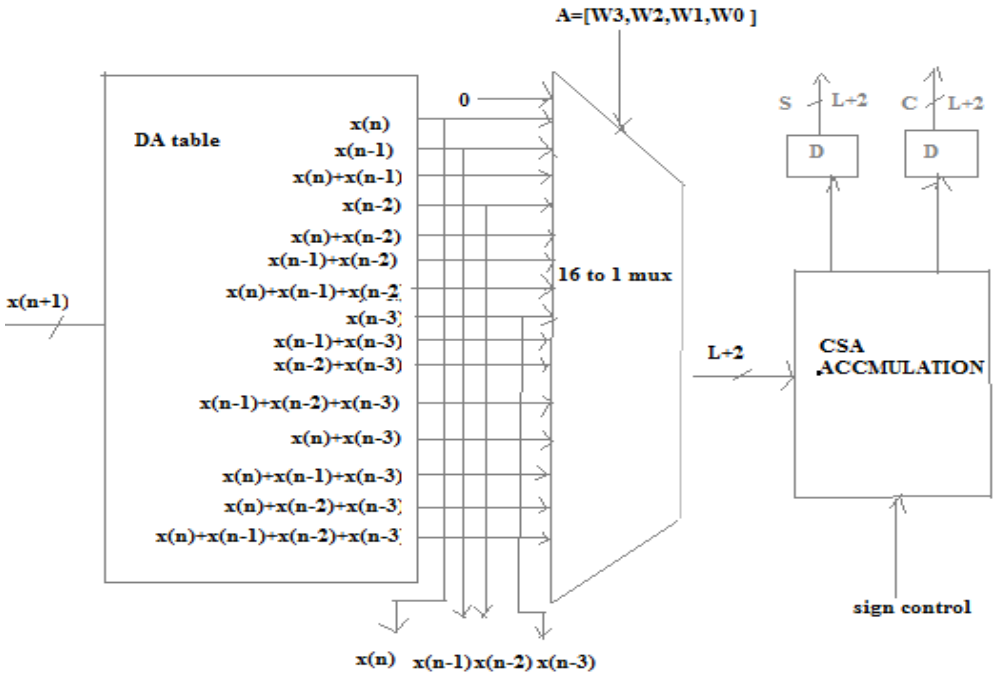

Fig 2. Structure of the four-point inner-product block.

The full adder using half adder is better because minimize the gate count and avoid the unwanted Carry bits. The Fig 3shows each of these columns, there is a possibility that we have a carry coming in the preceding column. Hence, we want to build a circuit with three input bits and produce the sum of these inputs as its output. We can use a and b to represent the bit from each of the numbers in the column but there will be a 3rd input which we can call c_{in} corresponding to the potential carry from the preceding column..The content of the k th LUT location can be expressed as

$$c_k = \sum_{j=0}^{n-1} x_j \,.\, k_j \qquad (7)$$

Where k_j is the (j + 1) th bit of N-bit binary representation of integer k for $0 \leq k \leq 2N - 1$. Note that ck for $0 \leq k \leq 2N - 1$ can be pre computed and stored in RAM-based LUT of $2N$ words. Hence, instead of storing $2N$ words in LUT, we store ($2N - 1$) words in a DA table of $2N - 1$ registers.

.

Fig 3.Full adder using half adder

.

.

IV.PROPOSED DA-BASED ADAPTIVE FILTER STRUCTURE

The computation of adaptive filters of large orders needs to be decomposed into small adaptive filtering blocks since DA based implementation of inner product of long vectors requires a very large LUT [3]. Therefore, we describe here the proposed DA-based structures of small- and large-order LMS adaptive 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 $N = 4$ is shown in Fig. 4. It contain 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.

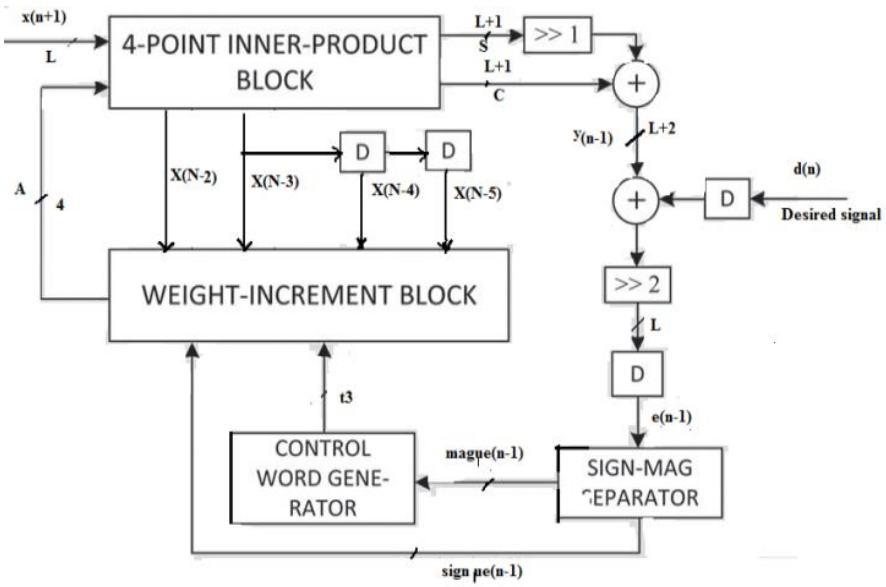

Fig 4. Proposed structure of DA-based LMS adaptive filter length $N = 4$.

The adaptive filter has two inputs the primary input *d(n),* and reference signal *x(n)*. The goal of adaptive filtering systems is used to reduce the noise and obtaining the uncorrupted signal. Hence to achieve this task, the noise signal is needed .That signal is fed to the system, and it is called a reference signal *x(n)*. However, the reference signal is not the same signal as the noise portion of the primary signal. It can be changed in amplitude, phase or time delay. Therefore the reference signal cannot be simply subtracted from the primary signal to obtain the desired portion at the output. This filtered noise is the system's prediction of the noise portion *and,* the resulting signal is called error signal *e(n),* and it presents the output of the system.

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 of filter length P as

$$y = \sum_{k=0}^{p-1} w_k . x_k + \sum_{k=p}^{2p-1} w_k . x_k + \sum_{k=n-p}^{n-1} w_k . x_k \qquad (8)$$

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. 5. It consists of four inner-product blocks of length $P = 4$. The $(L + 2)$-bit sums and carry produced by the four blocks are added by two separate binary adder trees [9]. Four carry-in bits should be added to sum words which are output of four 4-point inner-product blocks.

Volume: 1 Issue: 2 08-Feb-2014,ISSN_NO: 2347 -7210

Hence the carry words are double the weight which is compared to the sum words, 2 carry-in bits are set as input carry at the first level binary adder tree of carry words, which is equivalent to four carry-in bits to the sum words.

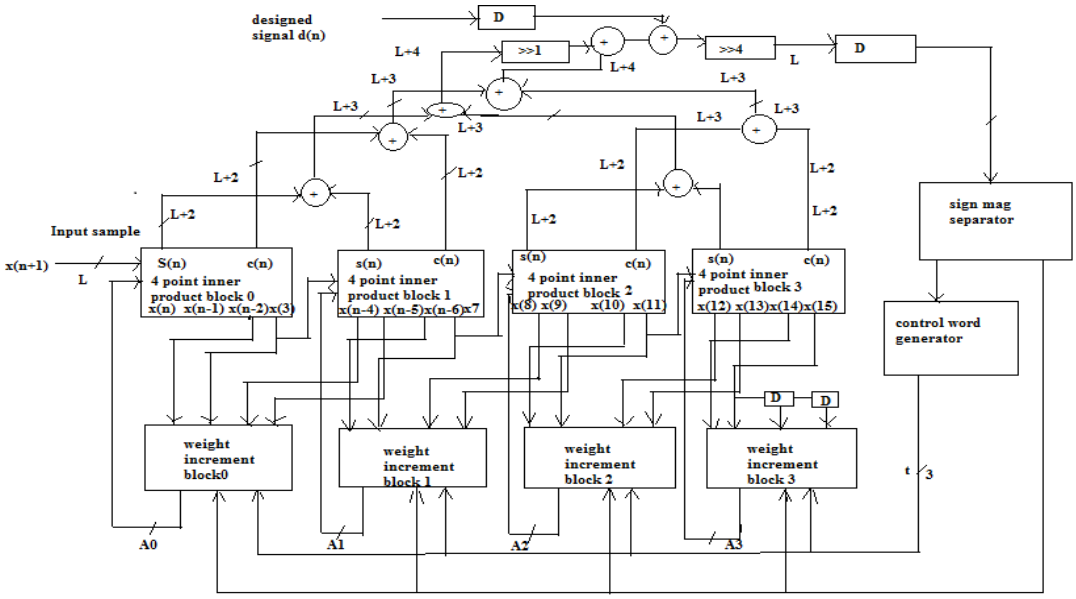

Fig 5 Proposed structure of DA based LMS adaptive filter of length N=16 and p=4

V. SYNTHESIS RESULTS

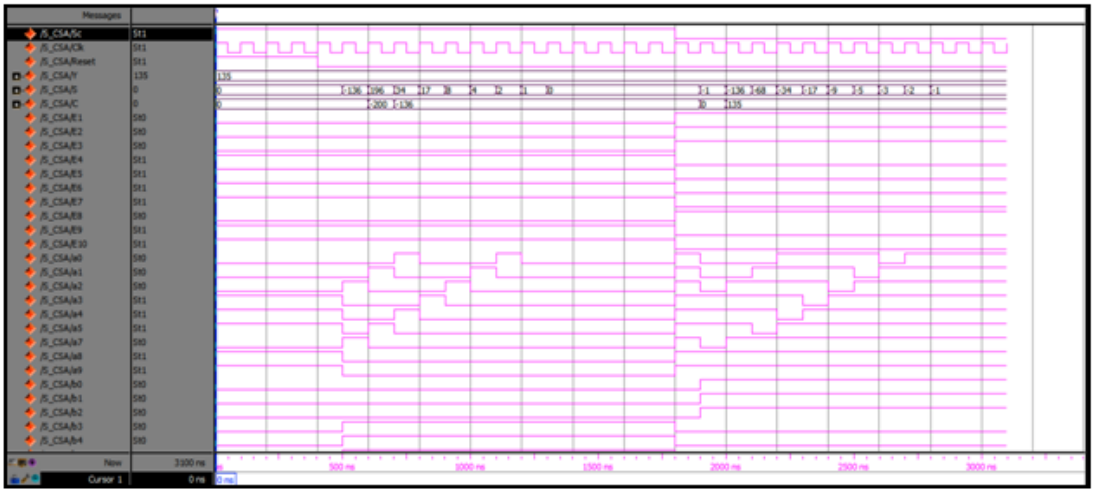

Fig 6 CARRIES SAVE ADDER

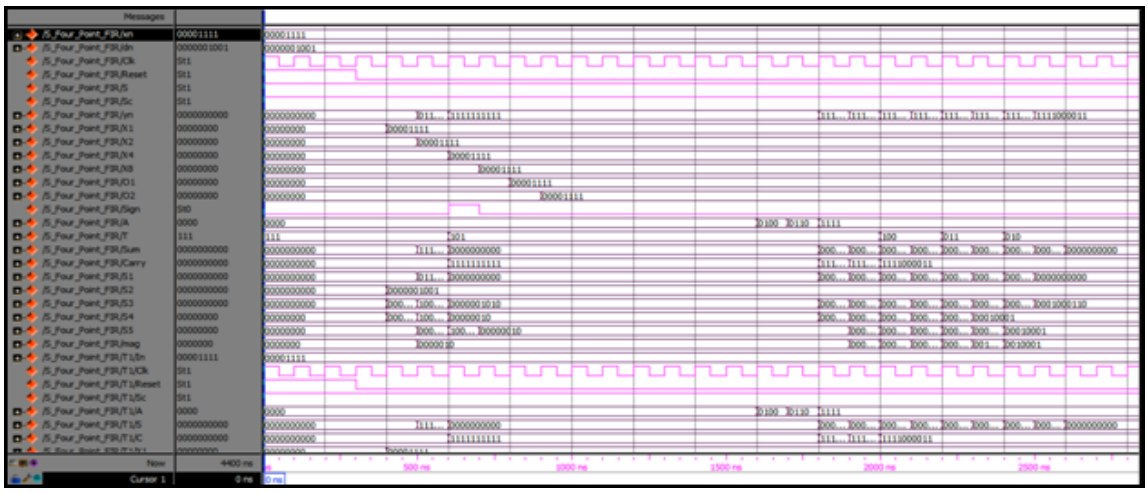

Fig 7 FOUR POINT INNER PRODUCT

TABLE I

COMPARISON OF AREA AND POWER

FILTER TAP	AREA (Sq.μm)	POWER (mw)
(N=4) PROPOSED	10630	35mw
MODIFICATION	10518	29mw
(N=16) PROPOSED	21261	70mw
MODIFICATION	21036	57mw

VII. CONCLUSION AND FUTURE WORK

We have suggested an efficient pipelined architecture for low-power, and low area implementation of DA-based adaptive filter. We have implemented 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 less power and less ADP over our previous DA-based FIR adaptive filter in average for filter length. Compared to the best of other existing designs our proposed design is better for area and ADP consumption. In future work can be implemented on digital communication, signal processing application, digital radio receivers, down converts, software radio receivers and echo cancellation.

REFERENCES

[1] Abdul Qayyum and Moona Mazher,“ Design of Programmable Efficient Finite Impulse Response Filter Base On Distributive Arithmetic Algorithm", Volume 1,December 2012 .

[2] Basant K. Mohanty, Senior Member, IEEE, and Pramod Kuma Meher, Senior Member, IEEE Senior Member, IEEE ‘A High- Performance Energy-Efficient Architecture for FIR Adaptive Filter Based on New Distribute Arithmetic Formulation of Block LMS Algorithm’ VOL. 61, NO. 4, FEBRUARY 15, 2013

[3] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson “LMS adaptive filters using distributed arithmetic for high throughput IEEE Trascircuit Syst. I, Reg. Papers, vol.52, no. 7,pp. 1327–1337,Jul. 2005.

[4] M. D. Meyer and P. Agrawal, “A modular pipelined implementation of a delayed LMS transversal adaptive filter,” in Proc. IEEE In Symp Circuit Syst., May 1990, pp. 1943–1946.

[5] N.V.Sai Rupa, M.Ram Mohan Reddy, “Memory Efficient Architecture For High Speed Fir Filter Using Distribute Arithmetic Volume 4, Issue 9, September-2013".

[6] P. K. Meher and S. Y. Park, “High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic,” in VLSI Symp. Tech. Dig., Oct. 2011, pp. 428–433.

[7] P.Sravanthi, CH. Srinivasa Rao, S.Madhava Raeo ,”A Novel Approach of Area-Efficient FIR Filter Design Using Distributed Arithmetic with Decomposed LUT” Volume 7, Issue 2 (Jul. - Aug. 2013), PP 1318

[8] R. Guo and L.S. DeBrunner, “A novel adaptive filter Implementation using distributed arithmetic,” in Proc Asilomar Conf. Signals, Syst., Comput., Nov. 2011, pp.160–164.

[9] P.Sravanthi, CH.Srinivasa Rao, S.Madhava Rae ,”A Novel Approach of Area-Efficient FIR Filter Design Using Distributed Arithmetic with Decomposed LUT Volume 7, Issue 2 (Jul. - Aug. 2013), PP 1318

[10] Patrick Longa, Ali Miri, "Area-Efficient Fir Filter Design on FPGAs using Distributed Arithmetic" IEEE International Symposium on Signal Processing and Information Technology, pp:248-252,2006

[11] R. Guo and L. S. DeBrunner, “Two high-performance adaptive Filter ’' Syst. II, Exp. Briefs, vol. 58, no. 9, pp. 600–604, Sep. 2011.

[12] R. Guo and L. S. DeBrunner, "A novel adaptive filter Implementation using distributed arithmetic, "in Proc Asilomar Conf.Signals,Syst., Computer., Nov. 2011, pp.160–164.

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

[14] Sangyun Hwang, Gunhee Han,Sungho Kang, Jaeseok Kim, "New Distributed Arithmetic Algorithm for Low-Power FIR Filter Implementation", IEEE Signal Processing Letters, Vol.11, No5, pp:463-466,May, 2004

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