Low Power Area Efficient High Speed Implementation of LMS Adaptive Filter using Distributed Arithmetic

T Pitchaiah
Dept Of ECE
Vignan's Foundation for Science, Technology and Research (VFSTR) University, Guntur-522213, Andhra Pradesh, India.
tp1977_ece@yahoo.com

P V Sridevi
Dept of ECE
A U College of Engg.(A)
Andhra University
Vishakapatnam-530003, India.
Pvs6_5@yahoo.co.in

Abstract-- **In this paper, we present a hardware efficient architecture for the implementation of Least Mean Square (LMS) Adaptive Finite Impulse Response (FIR) Filter. The direct implementation of LMS Adaptive FIR filter involves more multipliers and adders (MAC's) units. Multipliers occupy much of the silicon area and consume more power. The Distributed Arithmetic (DA) replaces the multiplier with lookup table shift and add operations with bit serial nature. Due to multiplier-less implementation of DA based Adaptive filters, the architecture becomes Area efficient, low power and high speed. To reduce the area and sampling period the general conventional adder based shift accumulation for DA based inner product computation is replaced by conditional signed carry save accumulation. From ASIC synthesis results, it is found that the proposed design consumes 39.72% ,44.95% less dely and 52.5%, 87.16% less area-delay product compared to direct multiplier based implementation of Adaptive FIR filter for filter lengths N=16 and 32 respectively and by FPGA implementation it is found that the proposed design consumes 53%, 69% less dely and 42%, 83% less area-dealy product for filter lengths of N=16 and 32 respectively.**

Index Terms- **Adaptive Filters, ASIC, Distributed Arithmetic (DA), Field Programmable Gate Arrays (FPGA), Finite Impulse Response (FIR), Least Mean Square (LMS).**

1. I. INTRODUCTION

Adaptive Filters are widely used in many signal processing applications such as system identification and modeling, equalization, interference and echo cancellation [1]. The tapped delay line finite impulse response (FIR) filter whose weights are updated by the famous Widrow-Hoff least mean square algorithm is the most popularly used adaptive filter not only due to its simplicity but also due to its satisfactory convergence performance [2]. In the last two decades, the multiplier-less distributed arithmetic (DA)-based technique [3] has increased its importance in the implementation of filters, due to its high throughput processing capability and regularity, which results in cost effective and area-time efficient computing structures [4]. In the literature DA based design of Adaptive filter has been suggested by [5]-[7]. A fixed-coefficient filter can be easily realized using DA by storing the partial products of filter coefficients in the LUT[11]. But in adaptive filter implementation, there is some difficulty in partial products of the filter coefficient stored in the LUTs. These are to be calculated every time before filtering.

This paper implements the DA based architecture for low power, low area and delay efficient adaptive filter for high speed applications. These parameters are achieved by modifications in the conventional adder based shift accumulation with conditional signed carry save accumulation and by concurrent implementation of filtering and weight updation.

This paper is organized as follows. Section II describes the brief review of LMS algorithm followed by the proposed DA based LMS algorithm in section III. ASIC & FPGA implementation and synthesis results are given in IV followed by conclusion is mentioned in V.

II. REVIEW OF LMS ALGORITHM

The basic structure of a conventional LMS adaptive filter is shown in Figure.1

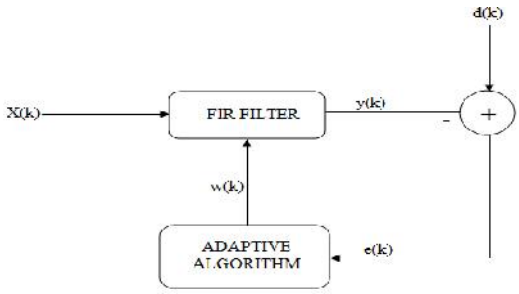

Figure.1 Structure of conventional LMS adaptive filter

Consider an adaptive filter that process an input signal x(k) and generates the output signal y(k) as per

$$y(k) = WT\ X \qquad (1a)$$

where WT =[w0(k), w1(k), w2(k),…, wN-1(k)] is the filter coefficient vector and XT=[x(k), x(k-1), …… ,x(k-N+1)] is the input sample vector. Where N represents the number of filter coefficients. The weights of the LMS adaptive filter during the kth iteration are updated according to the following equations:

$$w(k+1)=w(k)+\mu.e(k).x(k) \qquad (1b)$$

with

$$e(k)=d(k)-y(k) \qquad (1c)$$

Here d(k) is the desired response, e(k) is the error computation during kth iteration and μ is the convergence factor or step size. The basic working of the adaptive system is , in each cycle LMS algorithm computes a filter output and an error using eqn(1c). By using estimated error the filter weights are updated in every training cycle to reach the minimum error value.

In the pipelined designs, the feed- back error can be available after certain number of cycles, which is called as adaption delay[13]. The pipelined design uses delayed error e(k-a) for updating the current weight. The weight updation equation can be resembles as

$$w(k+1) = w(k) + \mu.e(k-a).x(k-a). \qquad (2)$$

Here, e(k-a) is the delayed error and a is the adaption delay.

III PROPOSED DA BASED LMS ADAPTIVE FILTER.

The DA based LMS adaptive consists of two blocks, namely Weight increment and N-Point Inner product computation block. These two performs N-multiplication-accumulation and error calculations. For the simplicity presentation, let the inner product of (1e) given by

$$y = \sum_{l=0}^{N-1} w_l . x_l \qquad (3)$$

Where wl and xl for 0 < l < N-1 form the N-point vectors corresponding to the current weights and most-recent N-1 input respectively. Assuming M to be the bit-width, and each component of the weight vector can be expressed in two's complement representation is given by

$$w_l = -w_{l0} + \sum_{m=1}^{M-1} w_{lm} . 2^{-m} \qquad (4)$$

Where wlm denotes the mth bit wl. Substituting (4), we can write (3) in expanded form

$$y = -\sum_{l=0}^{N-1} x_l . w_{l0} + \sum_{l=0}^{N-1} x_l . \left[\sum_{m=1}^{M-1} w_{lm} . 2^{-m}\right] \qquad (5)$$

To convert the sum of products form of (3) into a distributed form, the order of summation in (5) can be interchanged to have

$$y = -\sum_{l=0}^{N-1} x_l . w_{l0} + \sum_{m=0}^{M-1} 2^{-m} . \left[\sum_{l=0}^{N-1} x_l . w_{lm}\right] \qquad (6)$$

and the inner product of equation (6) can be computed as given by

$$y = \left[\sum_{m=0}^{M-1} 2^{-m} . y_m\right] - y_0$$

where

$$y_m = \left[\sum_{l=0}^{N-1} x_l \,.\, w_{lm}\right] \qquad (7)$$

Any element of the N-point bit sequence {wlm } for 0 < l < N-1 can be either zero or one and ym can be stored all the 2N possible values in look up table. The bit sequence {wlm} is used as address bits for computing the DA-based inner product. Therefore the inner product of equation (7) can be calculated in M cycles of shift accumulation followed by lookup table read operations are corresponding to M number of bit sequences {wlm for 0 < m < M-1} as shown in figure 1.Since the partial inner product ym bit slices are added while subtracting y0.Therefore, the sign of partial sums are need to be modified. In order to solve this problem we are using carry save accumulator [7].It can also solved a critical path problem in shift accumulation block.

According to the Reference [5], we can update the content of the Look up table(LUT). The location of partial inner product can be followed as

$$c_l = \sum_{i=0}^{N-1} x_i \,.\, k_i \qquad (8)$$

Here, ki is the (i+1)th bit of N-bit binary representation of integer i for (0< l < 2N-1).The cl sum of products can be pre-computed and stored in RAM lookup table of 2N words. However, by using DA table we can store (2N-1) words in (2N-1) registers.

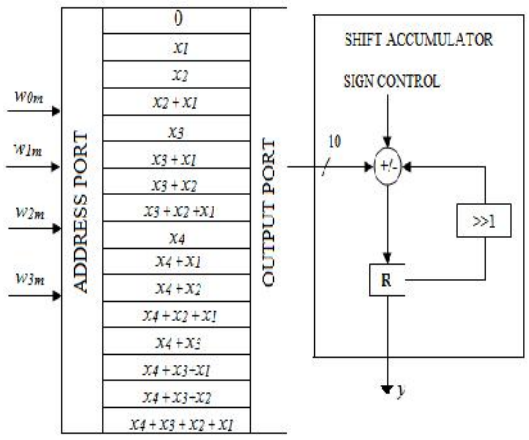

Figure.2 .Distributed Arithmetic technique based implementation of four point inner product block diagram.

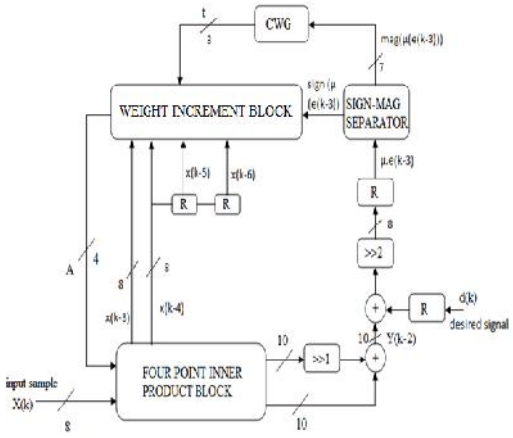

Figure.3.The block diagram of fourth order DA-technique based LMS adaptive filter.

These registers can be presented in 4-point inner product table, it is also contains 16X1 MUX and carry save accumulation block in existing [4] one. Instead of four stages of 2by1 multiplexor (MUX) is replaced by multi bit input and output of 16by1 multiplexor (MUX) is used to reduce the program complexity. The bit slices of weights A={w0m w1m w2m w3m} for 0 < m< M-1 are used as selection lines in LSB to MSB order. The output of multiplexer passed through the carry save accumulation (CSA) block. The error value can be obtained from the difference between desired output d(k) and right shift of the final adder sum of CSA output y(k-2). All the bits of error except the MSB bits are ignored. The multiplication of error and input xl are implemented by a right shift through the number of locations followed by number of leading zero's in the magnitude of the error. The magnitude and sign values are separated from the SIGN-MAG block. The magnitude of the computed error value is decoded through the control word generator (CWG) as t for the 3-stage barrel shifter [4]. we have taken as $\mu=1/N$. Here N is the filter length. The weight-update block consists of four barrel shifter cells, four adder/subtractor cells and parallel to serial converter. Here, the sign computation error value is giving as carry in for the adder/subtractor block [4].

The proposed structure design involves eight 4-point inner product blocks, eight weight update blocks, four binary adder trees are used for eight sum's ,which are coming from the eight 4-point inner product block.

Similarly, eight ripple carry adders are used for the carries of 4-point inner product blocks

Similarly, eight ripple carry adders are used for the carries of 4-point inner product blocks. Compare to the sum words, carry words are used for the double the weight, four cain1 bits are taken as input carry in bits for the ripple carry adder. Similarly, cain2 is the carry bit for the next stages. It can be shown in Figure.4. The lower five LSB's of e(k) is truncated for N=32 to make the word length of eight bits. Since, the design needs the most significant one of μe(k) and the truncation does not degrade the more performance of the adaptive filter. The RTL schematic was shown in Figure.7.

IV. ASIC & FPGA IMPLEMENTATION AND SYNTHESIS RESULTS

The designed filters were coded in Verilog HDL. The ASIC implementation was done using Cadence TSMC45nm Technology and physical design was also performed. The simulation results are shown in TABLE I and physical design was shown in Figures 5 & 6 for DLMS 16 and 32 order filters. The same filters are also implemented on FPGA, synthesis results of 32 order LMS filter and DA based LMS Adaptive filter results are shown in TABLES II, III and VI. FPGA Synthesis was done by XILINX-13.4v,ISE Simulator. The hardware used for testing is GENESYS board with Xilinx VIRTEX-5 series XC5VLX50T-1FF1136 FPGA. The hardware co-simulation setup was shown in Figure.8.

This board can be can be compatible with Xilinx's Micro Blaze, CAD tools, chip scope, EDK, free web pack. So designs can be completed at no extra cost. For High speed performances it offers, 12 digital clock managers, 48 DSP slices and up to 500MHz clock speeds. The simulation results of LMS adaptive filter and DA based LMS adaptive filter are shown in Figures.7 and 8.

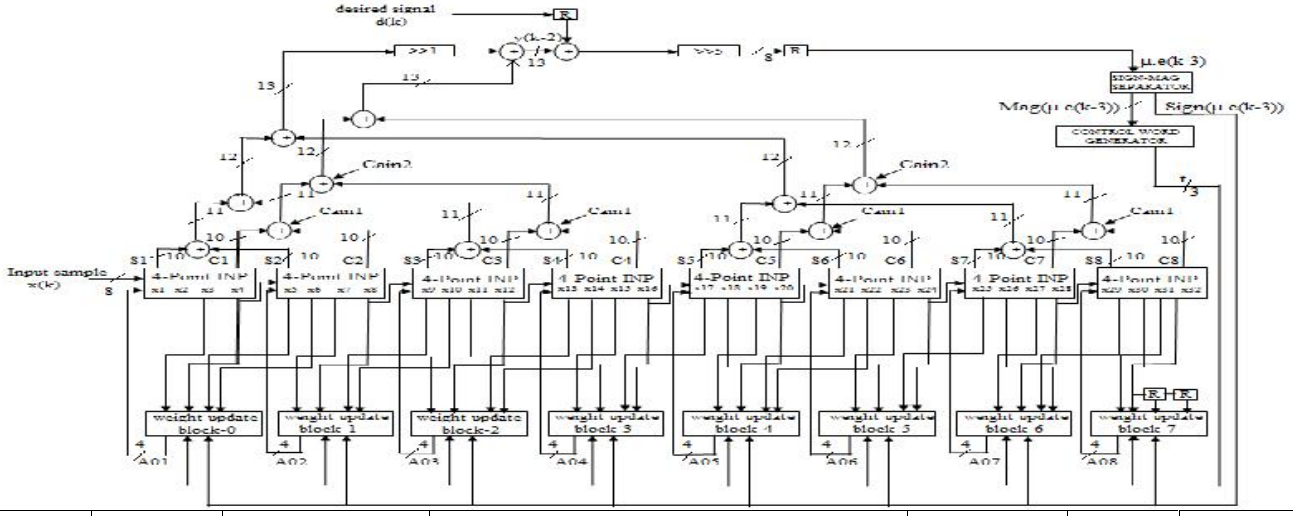

Figure.4.The structure of DA-based LMS Adaptive filter of order 32

TABLE I. ASIC SIMULATION RESULTS FOR LMS FILTER AND DA BASED LMS ADAPTIVE FILTERS

FILTER DESIGN METHOD	FILTER ORDER	AREA(Sq.uM)		POWER(mW)			DELAY (ns)	MSF (MHz)	ADP (Sq.uM*ns)	EPS (mW*ns)
		CELLS	AREA	LEAKAGE POWER	DYNAMIC POWER	TOTAL POWER				
LMS	16	2300	5924	0.116620	106.959439	107.076059	6.077	164.55	36000.148	650.70
DLMS	16	1750	5515	0.113873	84.978760	85.092633	3.593	278.31	19815.395	305.73
LMS	32	15296	38949	0.707777	531.716442	532.424220	8.624	115.95	335896.176	4591.62
DLMS	32	3290	10526	0.214993	159.866166	160.081160	4.096	244.14	43114.496	655.69

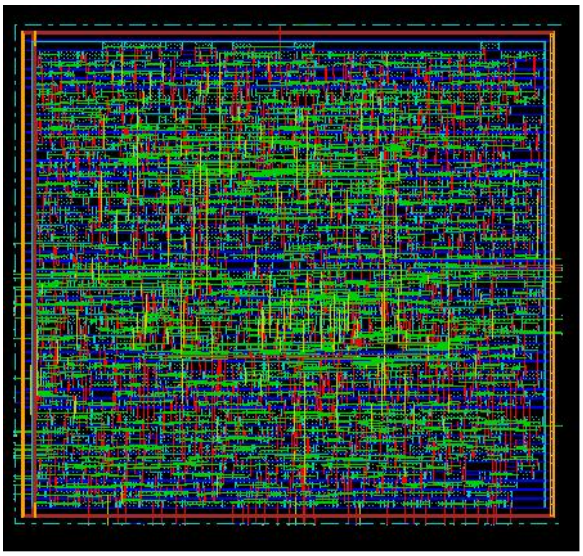

Figure.5. The Physical Design of DA based LMS Adaptive filter of length- 16.

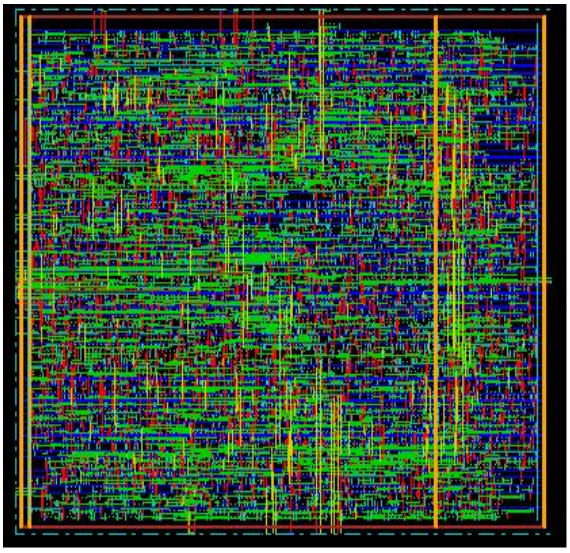

Figure.6. The Physical Design of DA based LMS Adaptive filter of length- 32.

Figure.7. The RTL Schematic of DA based LMS Adaptive filter of length- 32.

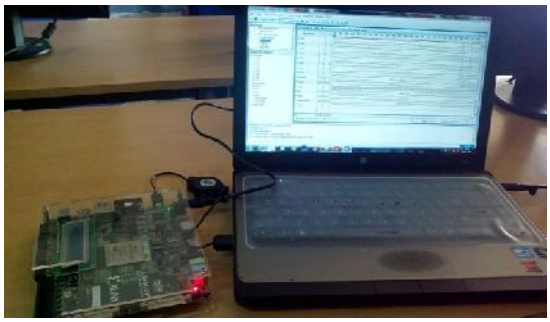

Figure.8.The Hardware co-simulation setup.

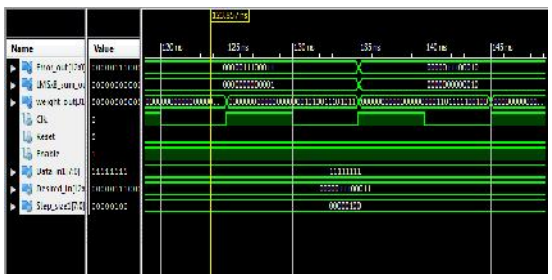

Figure.9. Simulation for LMS adaptive filter of length- 32

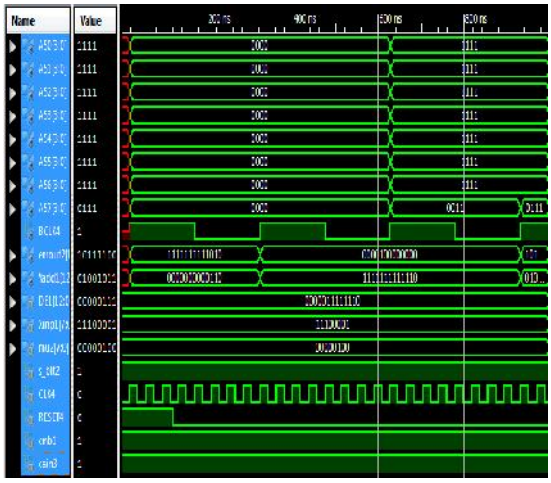

Figure.10. Simulation for DA based LMS Adaptive filter of length -32

TABLE II. SYNTHESIS RESULTS FOR LMS FILTER AND DA BASED LMS ADAPTIVE FILTER OF LENGTH 32

Parameter	LMS filter	DA Based LMS Adaptive filter
multipliers	65	2
Adders/subtractors	64	28
MAC's	30	-
Multiplexers	-	64
Registers	745	869
Flip-Flops	745	869
IO's	90	93
BEL's	6456	1920
LUT1	117	9
LUT2	1517	173
LUT3	404	298
LUT4	249	317
LUT5	136	215
LUT6	223	362
DSP48E	47/48	-
No. of slice registers	732/28800	888/28800
No. of slice LUT's	2714/28800	1376/28800
Number used as logic	2714/28800	1376/28800
No.of LUT Flip-Flop pairs used	2997	1583
Number with an unused Flip-Flop	2265/2997	695/1583
Number with an unused LUT	283/2997	207/1583
No. of fully used LUT-Flip flop pairs	449/2997	681/1583

TABLE III. TIMING SUMMARY ANALYSIS

Designs	Filter Length	Min Period (ns)	Input Arrival Time (ns)	Output Required Time(ns)	Max Frequency (MHz)
LMS filter	16	32.237	11.224	19.813	31.020
	32	64.445	13.350	30.642	15.517
DA based LMS Adaptive Filter	16	6.501	6.915	9.294	153.822
	32	8.393	8.877	11.186	119.147

TABLE IV. DELAY AND ADP ANALYSIS

Designs	Filter Length(N)	Delay (ns)	ADP(sq.µm)*ns
LMS filter	16	44.599	398179.872
	32	78.260	4062633.12
DA based LMS Adaptive Filter	16	9.708	185461.632
	32	11.670	385390.018

V. CONCLUSION

Based on synthesis results it is observed that, the DA based LMS adaptive filter performance is better than the MAC based LMS adaptive filter. The ADP and Delay parameters are superior in the proposed design and these are less by 42% and 53% for 16th order DA based LMS Adaptive filter compared to MAC based LMS filter. Similarly, for the 32nd order DA based LMS Adaptive filter parameters are less by 83% and 69% respectively in FPGA implementation. The ASIC implementation shows that ADP and Delay are less by 44.95% & 39.72% for N=16 DA based LMS filter compared to direct LMS filter, and 87.16% & 52.5% less for N=32 DA based LMS filter compared to direct LMS filter.

REFERENCES

[1] Simon Haykin, Adaptive Filter Theory. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

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

[3] S.A. White. "Applications of the distributed arithmetic to digital signal processing:A tutorial review." IEEE ASSP Mag., vol.6, no.3, pp.4-19, july.1989.

[4] S Y Park and P K Meher "Low-Power, High-Throughput, and low-Area Adaptive FIR Filter based on Distributed Arithmetic" IEEE Trans. on circuits and systems II. Express Briefs, Vol.60, No.6, pp.346- 350, June 2013.

[5] D. J. Allred, H. Yoo, V.Krishna, W.Huan, 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.

[6] 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.
[7] R Guo and L.S. DeBrunner, "A Novel adaptive filter implementation scheme using distributed arithmetic." in proc. Asilomar conf.signals, syst, comput., pp. 160-164 , Nov. 2011.
[8] M. D. Meyer and P. Agarwal, "A modular pipelined implementation of a delayed LMS transversal adaptive filter." in proc.IEEE Int.symp.circuits syst, pp. 1943-1946, May 1990.
[9] M Surya Prakash and Rafi Ahamed Shaik, "Low area and high throughput architecture for an adaptive filter using distributed arithmetic." IEEE Trans.on circuits and syst.II, Express Briefs, vol.60, no.11, pp.781-785, Nov.2013.
[10] Shanthala S, and S.Y.Kulkarni , "High speed and low power FPGA implementation of FIR filters for DSP Applications" European Journal of Scientific Research, Vol 31, issue 1, May 2009.
[11] K Parhi, VLSI Digital Signal Processing systems: Design and Implementation. Wiley India Pvt. Ltd., New Delhi, 2007.
[12] Uwe Mayer-Baese. "Digital Signal Processing with field programmable Gate Arrays" springer, 2007.
[13] Pramod Kumar Meher and S.Y.Park "High Throughput pipelined realization of adaptive FIR filter based on distributed arithmetic", 19th International conference on VLSI and System on Chip(SoC), 2011 IEEE/IFIP, pp.428-433.
[14] T Pitchaiah, M D Lakshmi and P.V. Sreedevi, "FPGA implementation of low area and delay efficient adaptive filters using distributed arithmetic." in proc of Int.conf. on Adavances in Engineering & Technology Research(ICAETR), pp. 1-5, 2014.
[15] T Pitchaiah, P.V. Sridevi, and S Koteswara Rao, "Adaptive noise cancellation using LMS algorithm in monte carlo simulation" in proc. of 2nd Int.conf. on Electronics and Communication Systems (ICECS), pp.368-372, 2015.

ABOUT AUTHORS

Mr. T Pitchaiah received the B.Tech in Electronics and Communication Engineering from Acharya Nagarjuna University, Guntur in 1998. He has done his M.E in Electronic Instrumentation from Andhra University, Visakhapatnam in 2002. He has 14 years of teaching experience. He is currently working towards his Ph.D degree at Andhra University, Visakhapatnam. His research interest includes design of dedicated and reconfigurable architectures for signal processing algorithms and communication applications. He is member of IEEE.

Prof. P V Sridevi received the B.Tech in Electronics and Communication Engineering from Acharya Nagarjuna University, Guntur in 1986. She has done her M.E in Applied Electronics from PSG College of Technology, Coimbatore in 1988. She received Ph.D degree from Andhra University in 1997. She has 25 years of teaching and research experience. Currently she is working as Professor, Department of ECE, Andhra University, Visakhapatnam. Her research interest includes Antennas, VLSI etc. She has contributed more than 50 technical papers to various journals and conference proceedings. She is a recipient of The Sir Thomas Wond Memorial Gold Medal for 1988-89 (Institution of Engineers).