FOLDED ARCHITECTURE FOR NON CANONICAL LEAST MEAN SQUARE ADAPTIVE DIGITAL FILTER USED IN ECHO CANCELLATION

Pradnya Zode[1] and Dr.A.Y.Deshmukh[2]

[1]Research Scholar, Department of Electronics Engineering
G.H.Raisoni College of Engineering, Nagpur
[2]Professor, Department of Electronics Engineering
G.H.Raisoni College of Engineering, Nagpur

ABSTRACT

Power consumption reduction is transpiring drift in area of VLSI digital signal processing. This gives rise to need of minimization of silicon area which is done by folding algorithm. As silicon area decreases power consumption of a circuit decreases. Folding is an algorithm which reduces silicon chip area by combining various arithmetic operations into one operation by time scheduling technique. It is applied on iterative data flow graph with appropriate folding set. Least mean square algorithm alters coefficients of Adaptive filter in order to achieve desired output. Proposed work is focused on design of efficient VLSI architecture for LMS adaptive filter aims at reducing mainly area which results in power consumption reduction and hardware complexity. LMS filter structure used here is called non-canonical as transpose FIR structure is used. Results show that numbers of adders are reduced by 37.5 % and multipliers by 33.33% without changing characteristics of filter.

KEYWORDS

VLSI signal processing, folding, cutset retiming, iterative DFG, non-canonical.

1. INTRODUCTION

Today, every circuit has to face the power consumption issue, for both portable devices aiming at longer battery life and high-end circuits avoiding cooling packages and reliability issues that are too complex [1]. As a result, for any chip design, power consumption has to be taken into account very seriously. Earlier, only speed and silicon area were important in the design of integrated circuits, and power consumption was not an issue. Later, it was recognized that power consumption must be taken into account as a main design parameter. The increasing trend towards low power systems and high performance has forced researchers to come up with innovative design techniques which can achieve these objectives and meet the constricted system requirements. Again energy has always been a precious commodity. Smaller the size of gadgets, the lower is the energy requirements; this leads to the rise of minimization of the silicon area of the integrated circuits such as adders and multipliers. The performances and cost of any digital circuit depend on circuit design style. Therefore a careful choice of circuit design style is to be done for developing an architecture which has to establish optimal area-time-power trade-off.

DOI : 10.5121/vlsic.2016.7302

Due to the rapid growth of communication systems, Digital Signal Processing (DSP) is one of the fastest growing fields in electronics industry [7]. DSP plays a critical role in modern computing, which is used in numerous applications such as video compression, digital set-top box, cable modems, digital versatile disk, data communications, wireless communications, telecommunications, image processing, voice recognition systems, portable systems/computers, multimedia, speech processing, ,digital radio, digital still and network cameras, speech processing, transmission systems, radar imaging, acoustic beam formers, global positioning systems, and biomedical signal processing. The term "digital signal processing" refers to continuous mathematical manipulation on data applied in real time. These algorithms include digital filters, adaptive filters, discrete cosine transform, inverse discrete cosine transform, Fast Fourier transform, convolution, correlation, and several others [3].In Digital Signal processing architectures, the folding algorithm is used to systematically determine the control circuits where multiple algorithm operations are time multiplexed to a single functional unit. This folding technique can be used for synthesis of Digital Signal processing architectures that can be operated using single or multiple clocks. In synthesizing DSP architectures, it is important to minimize the silicon area of the integrated circuits, which is achieved by reducing the number of functional units (such as multipliers and adders), registers, multiplexers, and interconnection wires. For example, executing N addition operations using a single adder [2]. This means executing multiple operations on a single functional unit and the number of functional units in the implementation is reduced, thus resulting with low silicon area in integrated circuits [2].In this paper folding algorithm is applied to a least mean square (LMS) adaptive filter. The digital filter is one of the fundamental processing elements in any digital signal processing (DSP) system. Digital filters are used in DSP applications that range from video and image processing to wireless communications. Filters are used to achieve desired spectral characteristics of a signal, to reject unwanted signals, like noise or interferers, to reduce the bit rate in signal transmission, etc. The idea behind filters adaptive is that the filer should change of alter filter coefficients to some algorithm or real time. Adaptive filter is a computational device that attempts to model the relationship between two signals in real time in an iterative manner [4]. Adaptive filters, because of their ability to operate satisfactorily in non-stationary environments, have become an important part of DSP applications where the statistics of the incoming signals are unknown or changing. Some examples are in channel equalization, echo cancellation or adaptive beamforming. Adaptive filters can adjust to unknown environment, and even track signal or system characteristics varying over time. The least mean squared error (LMS) algorithm is a most popular choice for the adaptive filter implementation as the LMS algorithm is very simple and effective [4]. In this paper, we proposed an efficient low-area, low power folded VLSI architecture for LMS adaptive filter. Folding is a technique to reduce the silicon area by time multiplexing many algorithm operations into single functional units, such as adders and multipliers [2, 5]. The LMS filter structure used in this paper is referred as non canonical LMS filter as the FIR filter block in the LMS filter structure is the transpose FIR structure. It is advantages to use the transpose FIR structure as it has less critical path compared to the standard DFG (Data Flow Graph).

This paper is organized as follows. Review of LMS Adaptive Filter is given in section-2. Section-3 addresses the importance of Data Flow Graph Model. Basic essentials of folding technique are described in section-4. Section-5 presents the folding transformation technique applied on LMS adaptive filter. Section-6 presents the implementations of original LMS adaptive filter and folded non canonical LMS adaptive filter and their results with respect to power and area. Finally some conclusions are given in Section-7.

2. REVIEW OF LMS ADAPTIVE FILTER

Adaptive filter is a computational device that attempts to model the relationship between two signals in real time in an iterative manner [3]. The computational skill within adaptive filtering

are the algorithms used to calculate the updated filter weights. An adaptive filter is defined by four aspects: the signals being processed by the filter and the structure that defines how the output signal of the filter is computed from its input signal, the parameters within its structure that can be iteratively changed to alter the filter's input-output relationship and the adaptive algorithm that describes the way how parameters are adjusted from one time, instant to the next. The general adaptive filter system is shown in figure 1.

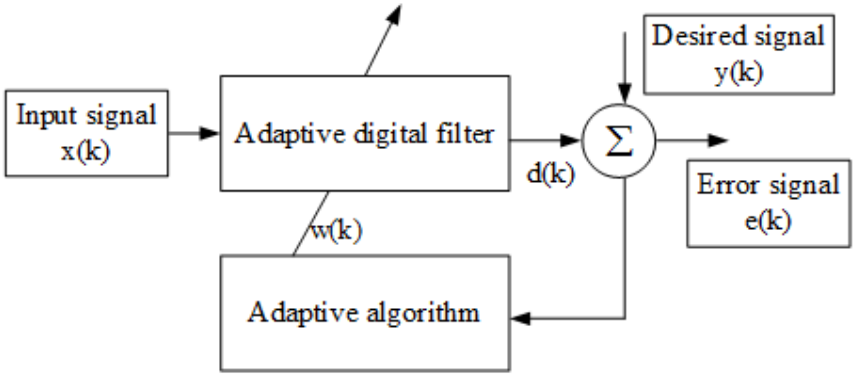


Figure 1. General adaptive filter system

The LMS algorithm offers a very simplistic yet powerful approach; giving good performance under the right conditions [4].The LMS algorithm is a stochastic gradient algorithm which uses a fixed step-size parameter to control the updates to the tap weights of a transversal filter. The algorithm aims to minimize the mean-square error, the error being the difference in y (k) and d (k). It has been used in a wide range of applications because of its robustness and low computational complexity. It is iterative procedure involves computing the output of a Finite Impulse Response (FIR) filter produced by a set of filter coefficients and followed by the generation of an estimated error by comparing the filters output to a desired response and finally adjusting the filter coefficients based on the estimation error. An adaptive filtering is fundamentally based on discrete mathematics of least square minimization theory and least square is of course widely used in statistical analysis and virtually every branch of science and engineering. Digital filters weights are updated as some function of the error signal. [4]

Aim of the Adaptive filter is to minimize the error signal and this depends on nature of input, length of Adaptive filter [4]. Adaptive algorithm

x (k)=input signal
y (k) = desired signal
e (k)=Error signal
d (k)=w(k)x(k)
e (k) = d(k) – y(k)
Weight update algorithm, $w(k+1) = w(k) + 2\mu\, e(k)\, x(k)$. (1)
μ = step size

3. DATA-FLOW GRAPH MODEL

Data flow graph is a type of graphical representation of digital signal processing algorithms. Graphical representations are efficient for finding out the data flow properties of DSP algorithms and the execution of different subtasks. It bridges the gap between algorithmic description and structural implementation of a DSP algorithm as a result it is easy to map DSP algorithms to hardware implementations. DFG is a conceptual view for expressing the function of a DSP system. DFGs are used for structural properties analysis and to find out architectural alternatives

using high level transformations like pipelining, parallel processing ,retiming ,folding, unfolding. In DFG representations the nodes represent computations (or functions or subtasks), while the directed edges represent data paths (data communications between nodes) and each edge has a nonnegative number of delays associated with it [2]. DSP algorithms are described by non-terminating programs which executes the same code repetitively. The DFG describes the data flow among subtasks or basic computations shown by nodes in a signal processing algorithm. The DFG for a 3-tap FIR filter is shown figure 2.

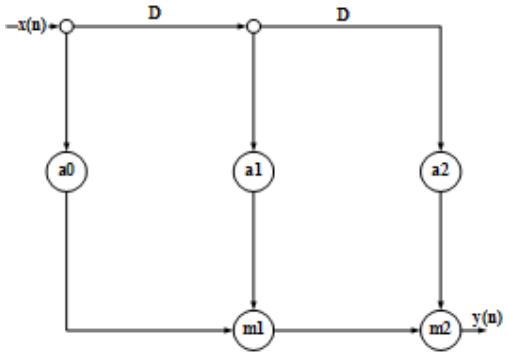


Figure 2. DFG of a 3-tap FIR filter

The DFG for LMS adaptive filter is shown in figure 3 which is used for further operations in this paper. The FIR filter structure block in the LMS filter structure is the transpose FIR structure. It is advantages to use the transpose FIR structure as it has less critical path compared to the standard DFG.

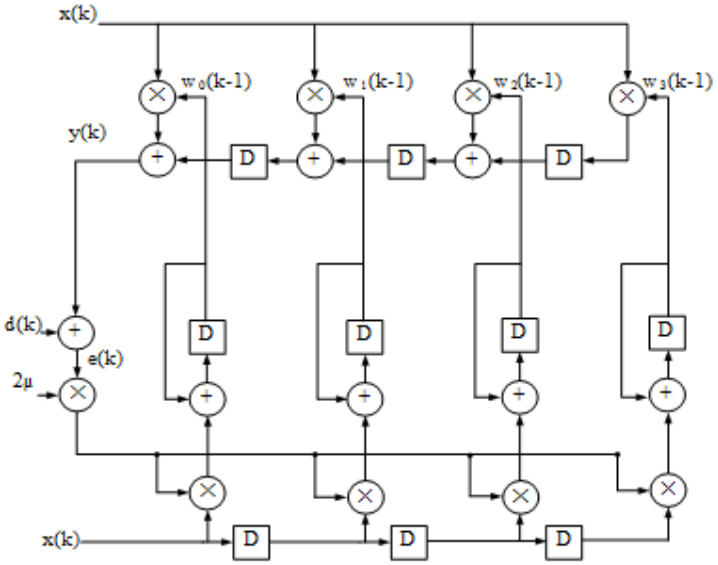


Figure 3. LMS Adaptive filter DFG

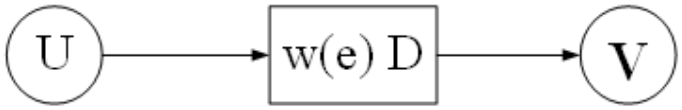
4. BASIC ESSENTIALS OF FOLDING TECHNIQUE

The folding transformation technique is introduced by K.K. Parhi which is described in [2]. A brief review of folding transformation is given here before applying it to LMS adaptive filter. The folding transformation is explained in Figure 4. a) And Figure 4. b). Figure 4 a) shows an edge from node U to node V with D weight i.e. the edge have w (e) delay value on it. Figure 4 b)

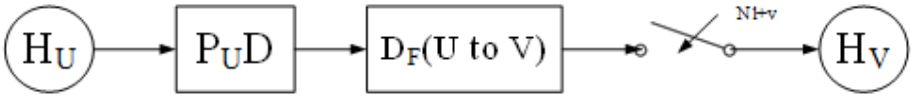
shows the corresponding folded edge. The functional source unit of node U is H_U which is pipelined by P_U stages must pass through delays given by

$$D_F(U \rightarrow V) = Nw(e) - P_U + v - u \quad (2)$$

The edge switched at destination functional unit H_V of node V at instance Nl+v, where N is a folding factor. It is defined as the number of operations folded to a single functional unit while u and v are the folding orders of nodes U and V respectively. The folding order of a node the time partition to which the node is scheduled to execute in hardware. A folding set, S, is defined as an ordered set of operations, which contains N entries, executed by the same functional unit. For a folded system to be possible, $D_F(U \rightarrow V) \geq 0$ must hold for all of the edges in the DFG. Once valid folding sets have been assigned, retiming can be used to satisfy this property or determine that the folding sets are not feasible [1].



(a)



(b)

Figure 4 (a) and (b)

The main objective of this paper is to design LMS adaptive filter data flow graph shown in figure 2 using Folding Algorithm. This leads to reduction in execution time, silicon area and power consumption. Folding is a technique to reduce the silicon area by time multiplexing many operations into single functional units (such as adder and multiplier). The LMS adaptive data flow graph consists of adders, multipliers and delays.

5. FOLDING TRANSFORMATION TECHNIQUE

The complete procedure for folding transformation technique is described stepwise below [2].

1. Selection of appropriate folding order and accordingly folding set
2. Write the folding equation using DFG
3. If needed perform retiming for folding
4. Rewrite folding Equation
5. Using folding equation construct Lifetime table
6. Find out Lifetime chart and minimum number of registers
7. Carry out data allocation using forward and backward register allocation technique
8. Draw folded architecture

In this paper, the folding factor N=4 is selected for performing folding on LMS adaptive filter. For reducing the complexity here, we have divided the LMS DFG of figure 2 in two parts first the upper part is the FIR filter block shown in figure 5 and the below part is the filter weight adaptation block shown in figure 11.

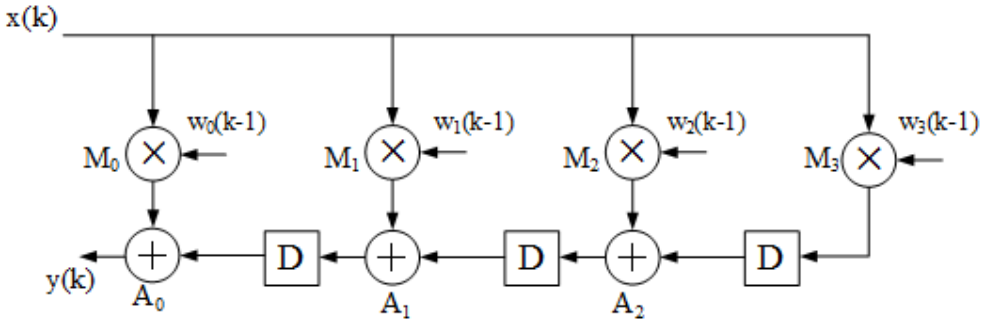


Figure 5. FIR filter block

5.1. Stepwise Folding of upper part FIR filter block

Step 1: Select Folding order N=4 and folding sets M for multipliers and A for adders as

$M = \{M_0, M_1, M_2, M_3\}$
$A = \{A_0, A_1, A_2, Ø\}$ where Ø is the null operation.

The folding set M for multipliers means a single multiplier in the folded architecture will perform 4 different operations depends on the order specified in folding set.

Step 2: Write the Folding Equation for DFG in figure 4 using folding equation

$$D_F\,(U \rightarrow V) = N\,w\,(e) - P_U + v - u$$

- $D_F\,(M_0 \rightarrow A_0) = 4\,(1) - 2 + 0 - 0 = -2$
- $D_F\,(M_1 \rightarrow A_1) = 4\,(1) - 2 + 1 - 1 = -2$
- $D_F\,(M_2 \rightarrow A_2) = 4\,(1) - 2 + 2 - 2 = -2$
- $D_F\,(M_3 \rightarrow A_2) = 4\,(2) - 2 + 2 - 3 = 1$
- $D_F\,(A_2 \rightarrow A_1) = 4\,(1) - 1 + 1 - 2 = 2$
- $D_F\,(A_1 \rightarrow A_0) = 4\,(1) - 1 + 0 - 1 = 2$

Step 3: If needed perform Retiming for folding

For proper folding none of the edge of the DFG should have negative delay value. As there are negative delays on some edges so have to perform retiming for folding on the same structure so that all edge delay should have positive value. Retiming is a transformation technique used to change the locations of delay elements in a circuit without affecting the input/output characteristics of the circuit [2]. Here cutest retiming is used which is a special case of retiming. DFG with cutest is shown in figure 6. The retimed DFG is shown in figure 7.

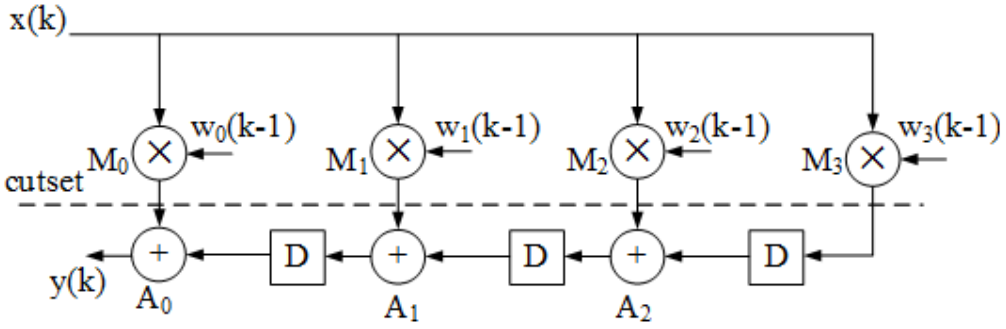


Figure 6. FIR filter DFG with cutset

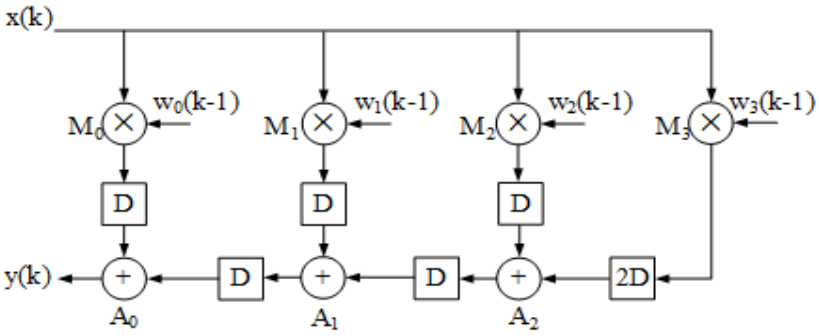


Figure 7. Retimed FIR filter DFG

Step4: Rewrite Folding Equation

- $D_F(M_0 \rightarrow A_0) = 4(1) - 2 + 0 - 0 = 2$
- $D_F(M_1 \rightarrow A_1) = 4(1) - 2 + 1 - 1 = 2$
- $D_F(M_2 \rightarrow A_2) = 4(1) - 2 + 2 - 2 = 2$
- $D_F(M_3 \rightarrow A_2) = 4(2) - 2 + 2 - 3 = 1$
- $D_F(A_2 \rightarrow A_1) = 4(1) - 1 + 1 - 2 = 2$
- $D_F(A_1 \rightarrow A_0) = 4(1) - 1 + 0 - 1 = 2$

From the above folding equations it is clear that now all the edges in the DFG have positive value of delay.

5.2. Register minimization

The folded structure may contain a more number of registers for storing the intermediate results. Therefore there is a need to minimize the number of registers used in the FIR filter architecture so that the silicon area occupied by the registers is to be minimized [2]. The procedure for computing the minimum number of registers and for allocating the data to these registers is as follows:

Step5: Using folding equation construct Lifetime table

The life period for each node is shown in Table 1, which is calculated using (3) and (4).

$$T_{input} = u + P_u \quad (3)$$

$$T_{output} = u + P_u + \max v\{D_F (U \rightarrow V)\} \quad (4)$$

Table 1: Life time table for each node of the FIR filter

Node	T_{input}	T_{output}	$T_{input} \rightarrow T_{output}$
M_0	0+2=2	2+2=4	2→4
M_1	1+2=3	3+2=5	3→5
M_2	2+2=4	4+2=6	4→6
M_3	3+2=5	5+5=10	5→10
A_0	--	--	--
A_1	1+1=2	2+2=4	2→4
A_2	2+1=3	3+2=5	3→5

Step 6: Find out Lifetime chart and minimum number of registers

A linear lifetime chart for the FIR is shown in Fig.8, which graphically represent the lifetime of each variable in a linear fashion. A variable is said to be alive from the time it is produced until it is consumed [6]. After the sample is consumed, it is dead. A variable takes up one register when it is alive. In lifetime analysis, the number of live variables at any time unit is determined, which gives the minimum number of registers required to implement the FIR architecture.The linear lifetime chart shown in Figure 7 is based on [2], where each horizontal line represents a clock cycle and each vertical line represents the lifetime of a variable. From the lifetime chart, the minimum numbers of registers that can be used to implement the architecture of a FIR filter is the maximum number of live variables at any time step [6]. The minimum number of registers to implement the FIR is maximum of live variables =max {0, 1, 2, 3, 4} = 4.

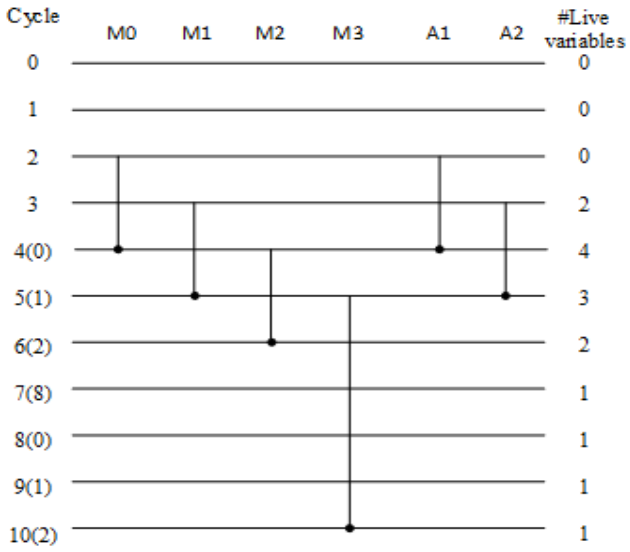


Figure 8. Lifetime chart for FIR filter

Step 7: Data allocation using forward backward register allocation technique

The registers are denoted as R1, R2, R3, and R4 in the allocation table shown in Figure 9. The variables are allocated forward first and then to an appropriate backward register, thus the name "forward-backward" [6]. In this allocation technique, the registers are reused and dead variables are not stored.

Cycle	Input	R1	R2	R3	R4	Output
0						
1						
2	A_1,M_0					
3	A_2,M_1	M_0		A_1		
(0) 4	M_2	M_1	M_0	A_2	A_1	A_1,M_0
(1) 5	M_3	M_2	M_1		A_2	A_2,M_1
(2) 6		M_3	M_2			M_2
(3) 7			M_3			
(0) 8				M_3		
(1) 9					M_3	
(2)10					M_3	M_3

Figure 9. Register allocation table for FIR filter

Step 8: Folded Architecture

The architecture has drawn with the help of all above 7 steps results in a folded structure for the FIR filter is shown in figure 10.

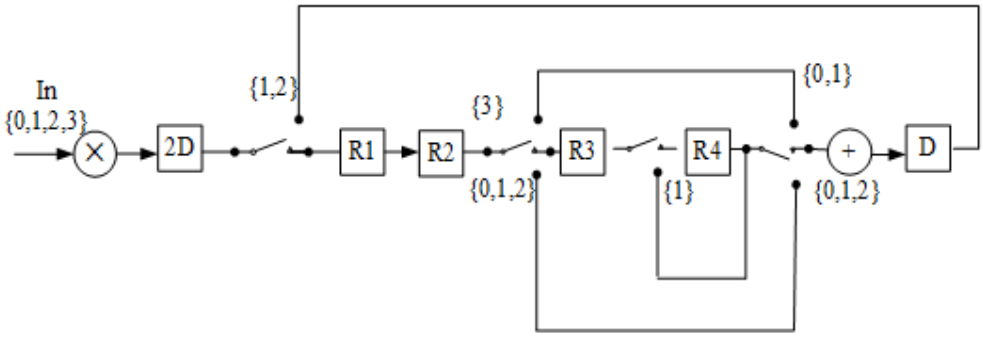


Figure 10. Folded Architecture of FIR filter

5.3. Stepwise folding of lower part of DFG i.e. weight update block

Figure 10 shows the folded architecture of the upper part of the LMS adaptive filter. The final folded structure will be the combined folded structure of upper and lower structure of LMS filter.

The below process is folding process for the lower part of LMS i.e. weight update block DFG is shown in figure 11.

Step1: Selection of appropriate Folding order and accordingly Folding Set

Again Folding order, N=4 and Folding sets for multiplier and adder are
$M = \{M_0, M_1, M_2, M_3\}$
$A = \{A_0, A_1, A_2, A_3\}$

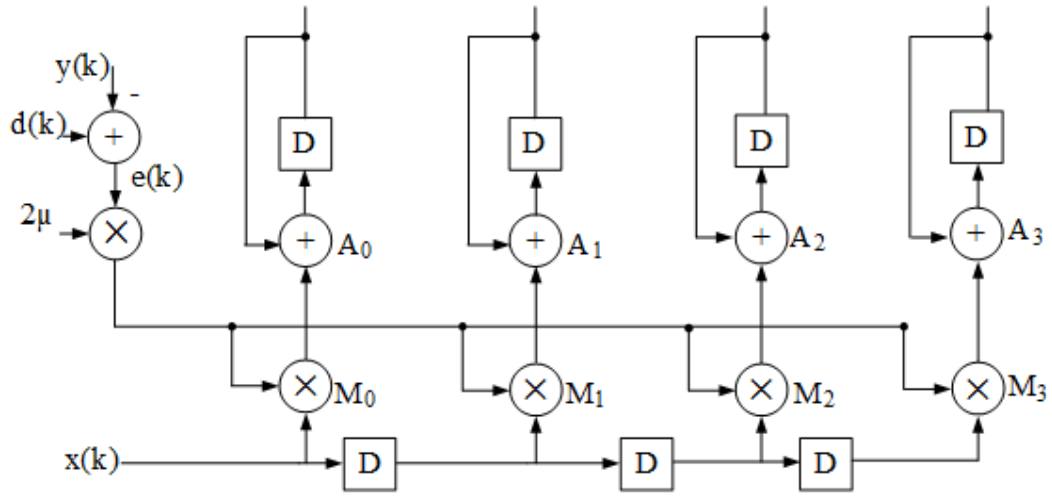


Figure 11. LMS weight update block DFG

Step 2: Write the Folding Equation using DFG

- $D_F (M_0 \rightarrow A_0) = 4 (0) - 2 + 0 - 0 = -2$
- $D_F (M_1 \rightarrow A_1) = 4 (0) - 2 + 1 - 1 = -2$
- $D_F (M_2 \rightarrow A_2) = 4 (0) - 2 + 2 - 2 = -2$
- $D_F (M_3 \rightarrow A_3) = 4 (1) - 2 + 2 - 3 = -2$

Step3: If needed perform Retiming for folding.

As all edges have negative value of delay so cutset is applied to the DFG as shown in figure 12.

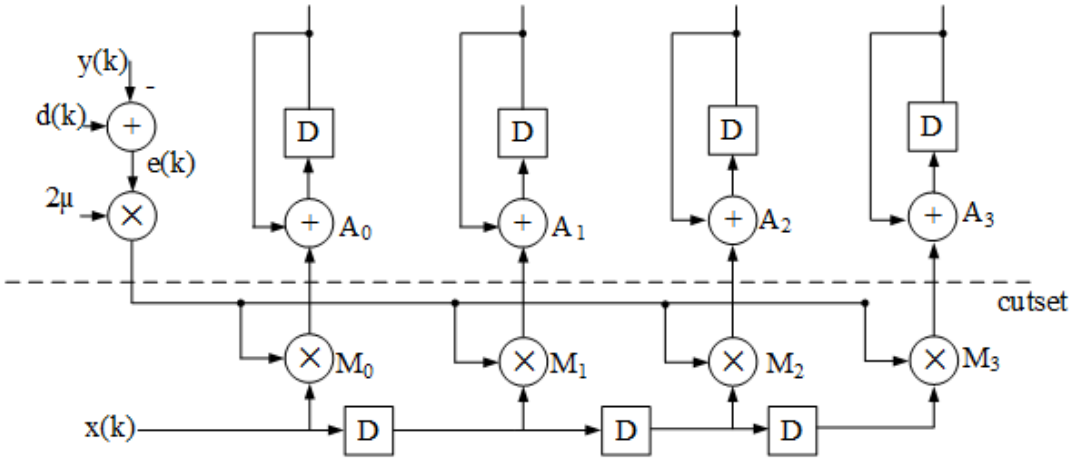


Figure 12. LMS weight update block DFG with cutest

Figure 13 shows the effect of cutest retiming on the above DFG.

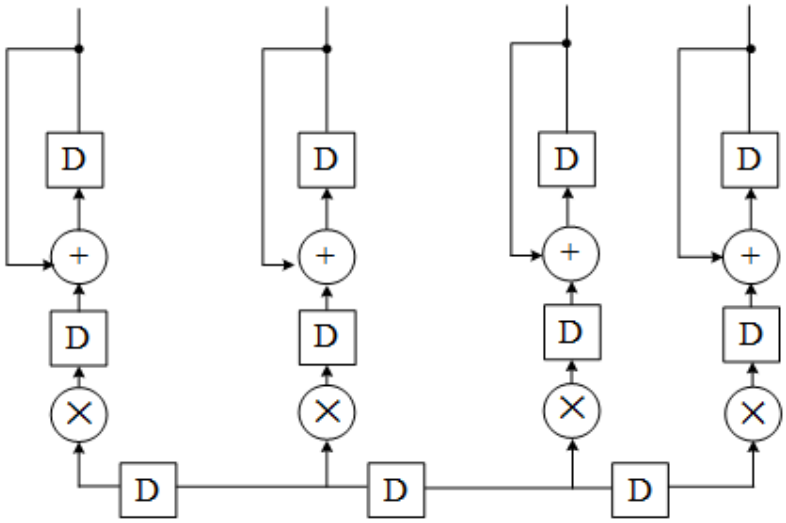


Fig.13.Retimed LMS weight update block DFG

Step4: Rewrite Folding Equation

- $D_F (M_0 \rightarrow A_0) = 4 (1) – 2 + 0 – 0 = 2$
- $D_F (M_1 \rightarrow A_1) = 4 (1) – 2 + 1 – 1 = 2$
- $D_F (M_2 \rightarrow A_2) = 4 (1) – 2 + 2 – 2 = 2$
- $D_F (M_3 \rightarrow A_3) = 4 (1) – 2 + 2 – 3 = 2$

Step 5: Using above folding equation constructed Lifetime table is shown in table 2

Table 2. Life time table of LMS weight update block

Node	T_{input}	T_{output}	$T_{input} \rightarrow T_{output}$
M_0	0+2=2	2+2=4	2→4
M_1	1+2=3	3+2=5	3→5
M_2	2+2=4	4+2=6	4→6
M_3	3+2=5	5+2=7	5→7

Step 6: Find out Lifetime chart and minimum number of registers

The lifetime chart for LMS weight update block is shown in figure 14 and the maximum number of registers needed for the implementation of folded architecture is maximum of live variables=max {0, 1, 2} = 2.

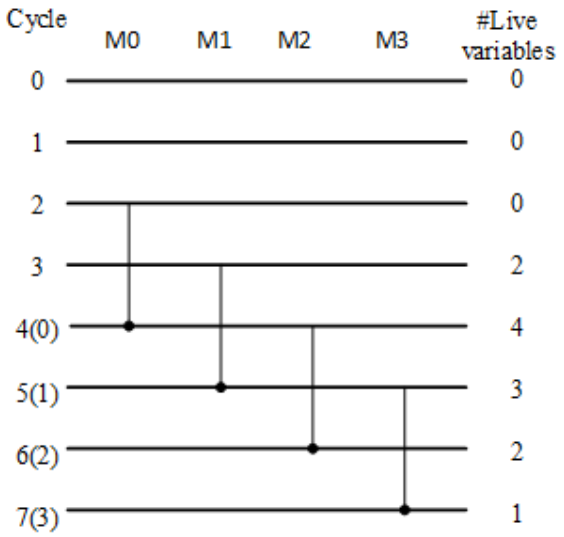


Figure 14. Lifetime chart for LMS weight update block

Step 7: Data allocation using the forward backward register allocation technique is shown in figure 15 [6].

Cycle	Input	R1	R2	Output
0				
1				
2	M_0			
3	M_1	M_0		
(0) 4	M_2	M_1	M_0	M_0
(1) 5	M_3	M_2	M_1	M_1
(2) 6		M_3	M_2	M_2
(3) 7			M_3	M_3

Figure 15. Allocation table for LMS weight update block

Step 8: Folded architecture for LMS weight update block is shown in figure 16.

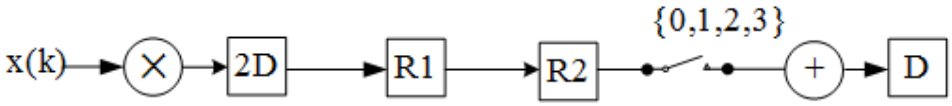


Figure 16.Folded Architecture for LMS weight update block

Folded structure of FIR filter block and folded structure of weight update block are joined to form the complete LMS adaptive filter folded structure shown in figure 17.

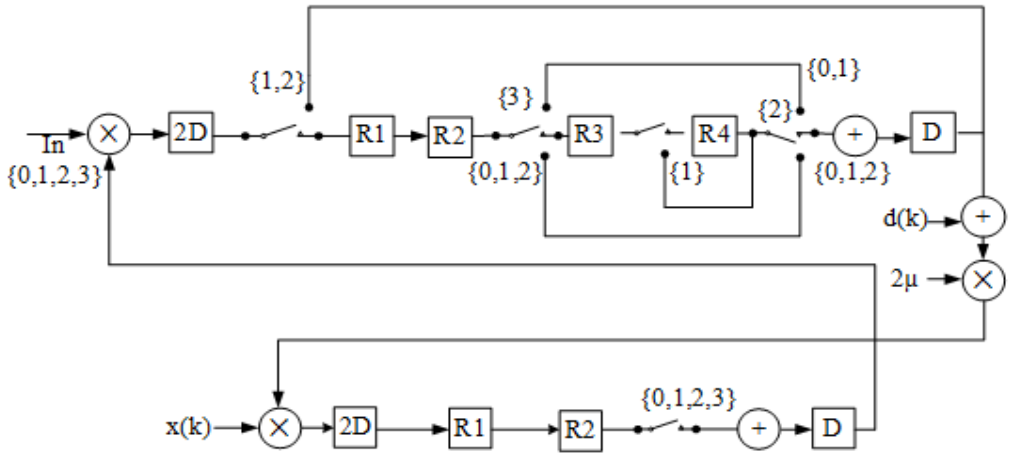


Figure 17. Final Folded Architecture of non canonical LMS adaptive filter

6. IMPLEMENTATION OF PROPOSED WORK AND RESULTS

Table 3 shows a comparison between the conventional and folded non canonical LMS adaptive filter in terms of the number of functional units i.e. adders & multipliers and the number of registers. As shown, the folding transformation results in a lower number of functional units at the cost of a slight increase in the number of registers.

Table 3. Comparison between the conventional and folded LMS adaptive filter

	No. of functional units		No. of registers
	Adders	Multipliers	
Conventional	8	9	10
Folded	3	3	13

It is seen that the number of adders are reduced by 37.5 % and multipliers by 33.33%. This result definitely shows the reduction in silicon area as per objective and hence the power consumption is also reduced.

6.1. Device utilization comparison of original DFG and folded DFG

The original and folded LMS adaptive filter is modelled in Verilog HDL and synthesized using ISE-Xilinx tool. Both the designs were synthesized using Spartan FPGA and the results are shown in the table 4.

Table 4. Synthesis result of conventional and proposed folded DFG architecture

Parameter	Conventional	Folded
Power	0.017W	0.011W
Delay	9.655ns	11.403ns
Power-delay product	0.164135	0.125433
frequency	103.572MHz	80.625MHz

Main focus of this paper is to design a filter which can be used in telecommunication for echo cancellation using LMS Adaptive algorithm. The echo cancellation period of unfolded LMS structure with folded one is compared and shown in fig.18.An audio dot wave file is called in software MATLAB and noise is added to this file as shown in fig.18. 2048 output samples are generated which are stored in a do file. This do file is given as an input the Verilog coding and simulated. Output text file is generated which is given to the MATLAB programming and the samples are plotted as shown in figure 18.

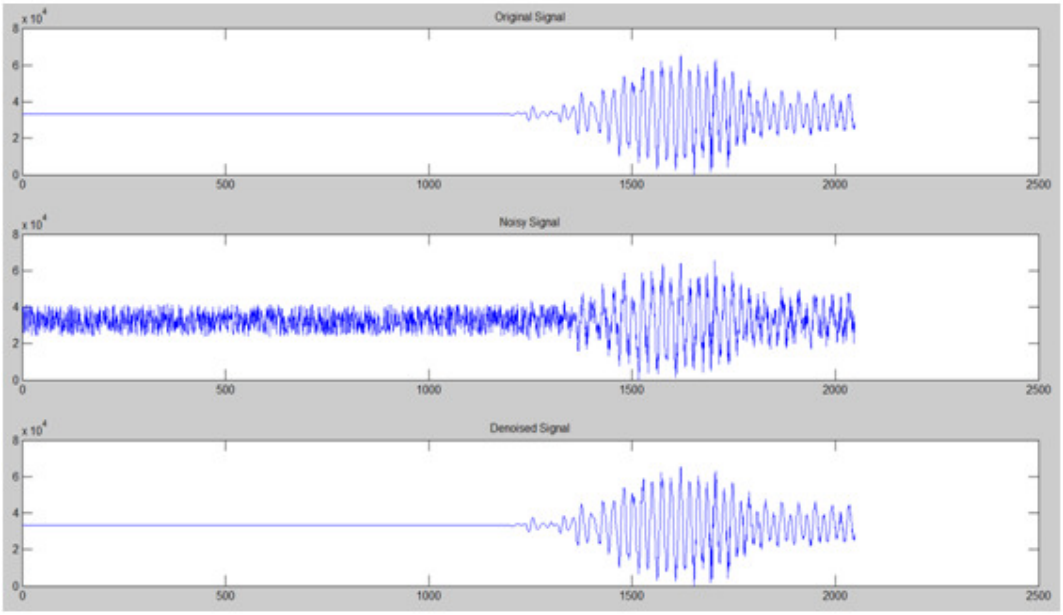


Figure 18. Echo cancellation output

7. CONCLUSION

In this work, a suitable non canonical LMS adaptive filer was realized in a digital environment using Xilinx ISE tool. This implemented digital filter can be used in telecommunication for echo cancellation. Our results show that the folded architecture of LMS adaptive filter significantly reduces the area of the filter, but there is a trade-off with the delay. Again the power consumption is reduced by 35%. Folding transformation reduces the number of hardware functional units by factor N at the expense of increasing the computation time by a factor of N. Here, folding factor = 4 means that the iteration period of folded hardware is 4 unit time means each node in filter is executed exactly once every 4unit time in folded structure.

REFERENCES

[1] Christian Piguet, History of Low-Power Electronics, CRC Press, 2005

[2] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. Hoboken, NJ, USA: Wiley, 1999.

[3] V. Oppenheim, R. W. Schafer, and J. R. Buck, Discrete-time signal processing. Prentice-hall Englewood Cliffs, 1989.

[4] S.Haykin, Adaptive Filter Theory. Prentice Hall, 1991

[5] K. K. Parhi, C. Y.Wang, and A. P. Brown, "Synthesis of control circuits in folded pipelined DSP architectures," IEEE J. Solid-State Circuits, vol. 27,no. 1, pp. 29–43, Jan. 1992.

[6] Rajalakshmi Karuppuswamy, Kandaswamy Arumugam, and Swathi Priya M." Folded Architecture for Digital Gammatone Filter Used in Speech Processor of Cochlear Implant "ETRI Journal, Volume 35, Number 4, August 2013

[7] Keshab K. Parhi, "Verifying Equivalence of Digital Signal Processing Circuits" Asilomar 2012.

AUTHORS

Pradnya Zode received B.E. (Electronics & Telecommunication) & M.Tech. (VLSI Design) Degrees from the RTM Nagpur University in 2001 and 2006 respectively. Currently working towards the Ph.D. from RTM Nagpur University, Nagpur She is currently working as Assistant Professor at Electronics Engineering Department of Yeshwantrao Chavan College of Engineering, Nagpur. She is Associate Member of Institution of Engineers (India) and Life Member of Indian Society for Technical Education. Her main interests include the design of Low-power VLSI Signal Processing.

Amol Deshmukh received M.E.(Electronics) from the SGGSIET, Nanded and PhD from VNIT, Nagpur in 2001 and 2010 respectively. He is currently Deputy Director and Dean Placements at G.H. Raisoni College of Engineering, Nagpur. He is Technical Committee Member of IEEE Soft Computing, USA & Senior Member-IEEE. His main interests include the Low Power VLSI Design, Embedded System Design and Low-power VLSI Signal Processing. He has also worked as Coordinator TEQIP-II (World Bank Assistance Project). He has to his credit around 100 plus International Conference and Journal Publications. He has also worked as International Co-Chair for ICETET-08, ICETET-09, ICETET-10, ICETET-11, ICETET-12, ICETET-13 (International Conference on Emerging Trends in Engineering & Technology). He has filed 02 Patents. He has received a research grant of Rs.7.37 Lacs from AICTE under RPS.