DESIGN AND IMPLEMENTATION OF 64TAP IIR ADAPTIVE LMS FILTER WITH IMPROVED PERFORMANCE

Suma C J[1]

Dr.ambedkar institute of technology
Near Jnana Bharathi Campus,
Bengaluru, Karnataka
560056

Akalpita L kulkarni[2]

Associat Professor, Dept. of E&C
Dr.ambedkar institute of technology
Near Jnana Bharathi Campus,
Bengaluru, Karnataka -560056

***Abstract*—**
The gradual rise of multi feature moveable devices with high speed processors and forceful rate of growth in element density changes the designer attention towards power aware designer system. Adaptive filter will decrease in terms of space and consumption of power in low power VLSI styles. The smallest amount Mean sq. (LMS) filter could be a category of the adaptive filters kind that is often used thanks to its simplicity and satisfactory convergence performance. To decrease area-delay product and energy-delay product, this IIR adaptive filter uses LMS. For reducing this delay, implementation of filter in pipelined structure is done. By victimization two main blocks, the IIR adaptive filter structure is done: IIR block and new coefficients block (weights block). Series of partial product generators and shift/add tree area unit combined to create Weights block. XILINX 14.7 is used for simulation and MATLAB 2014Ra is used to check the result how much effectively it is working.

Index Terms* — *Adaptive filter, Least Mean Square (LMS) algorithm, pipeline architecture, very-large-scale integration (VLSI), Weights block.

I. INTRODUCTION

In low power VLSI designs an adaptive filter can obtain a reduction in terms of area and power consumption. The evolution of multi-feature portable devices with high speed processors and with drastic growth in component density turns the designer attention towards power aware design schemes. A system with a linear transfer function controlled by variable parameters and a means to adjust those parameters according to an optimization algorithm is called adaptive filter.

Least mean squares (LMS) algorithms are a class of adaptive filter used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean square of the error signal (difference between the desired and the actual signal). It is a stochastic gradient descent method in that the filter is only adapted based on the error at the current time.

Simulation is carried out by using XILINX (ver14.7) and its constraints are verified using the same. Checking the result how much effectively working using MATLAB R2014a.

II. EXISTING SYSTEM

The current IIR adaptive filter uses LMS to decrease area-delay product and energy-delay product. To minimize this delay, one can implement filter in pipelined structure. Shift-add tree correctly minimizes the vital path and silicon area without growing the range of adaptation delays. The structure of IIR adaptive filter designing is finished by means of using two primary blocks: IIR block and new coefficients block (weights block). Weights block consists of sequence of partial product mills and shift/add tree. Partial product mills have 2 to 3 decoders and AND/OR cells. Weights block performs multiply accumulate operations. Filter block relies upon upon on the new filter coefficients acquiring from weights block. The filter is designed in MATLAB (2013a) for its performance characteristics and its constraints are demonstrated using XILINX (verl4.7). The designs are carried out on the subject programmable array platform of XILINX units for filter lengths N = 8, 16, 32 with enter word size L = 8.

III. PROPOSED ALGORITHM

There are many algorithms used to adjust the coefficients of the digital filter to suit the desired response as nicely as possible. The LMS Algorithm is the greater successful of the algorithms due to the fact it is the most environment friendly in terms of storage requirement and indeed computational complexity, the basic LMS algorithm updates the filter coefficients after each sample. This layout is synthesized and analyzed with the assist of Xilinx ISE 14.7. For the filter length N= 64. Checking the end result how plenty effectively working the use of MATLAB R2014a.

LMS Filter Algorithm

Least mean squares (LMS) algorithms are a classification of adaptive filter used to mimic a favored filter by finding the filter coefficients that relate to producing the least suggest rectangular of the error signal (difference between the desired and the actual signal). The adaptation of the filter parameters is based on minimizing the mean squared error between the filter output and a desired signal. Due to the computational simplicity, the LMS algorithm is most in many instances used in the graph and implementation of integrated adaptive filters. In the signal processing there is large variety of stochastic gradient algorithm in that the LMS algorithm is an imperative aspect of the family. The LMS algorithm can be differentiated from the steepest descent approach by means of term end chiastic gradient for which deterministic gradient works typically in recursive computation of filter for inputs is used which is having the noteworthy function of simplicity for which it is made well-known over other linear adaptive filtering algorithms. Moreover, it does no longer require matrix inversion. LMS algorithm is most famous because of this convergence velocity but choice of step dimension is very necessary in the case of success of algorithm.

Block diagram of LMS adaptive filter:

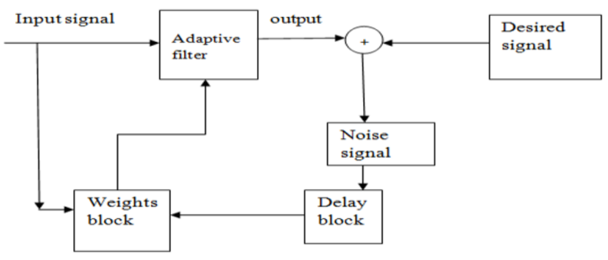


Figure 1: block diagram of LMS adaptive filter.

The block diagram of LMS adaptive filter is shown in figure 1 where the delay block represents the amount of delays introduced by the whole adaptive filter block. LMS adaptive filter has two important blocks in the design:

1. Adaptive filter block
2. Weights block

The Least-Mean-Square algorithm in words**:**

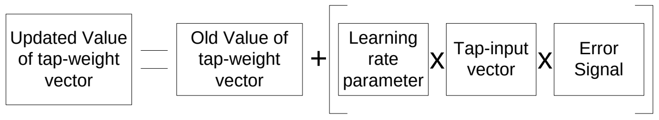

Updated Value of tap-weight vector
=
Old Value of tap-weight vector
+
Learning rate parameter
X
Tap-input vector
X
Error Signal

Figure 2: The LMS Algorithm in words

Figure 2 shows the block diagram of LMS adaptive filter. The simplicity of the LMS algorithm and ease of implementation means that it is the best choice for many real-time systems.

The implementation steps for the LMS algorithm

1. Define the desired response. Set each coefficient weight to zero.

$$w(n)_{=0}, n = 1,2,3,\ldots,N \ldots\ldots\ldots\ldots\ldots\ldots\ldots 1$$

For each sampling instant (n) carry out steps (2) to (4):

2. Move all the samples in the input array one position to the right, now load the current data sample n into the first position in the array. Calculate the output of the adaptive filter by multiplying each element in the array of filter coefficients by the corresponding element in the input array and all the results are summed to give the output corresponding to that data that was earlier loaded into the input array.

$$y(n) = \sum_{n=0}^{N-1} w(n)x(n) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 2$$

3. Before the filter coefficients can be updated the error must be calculated, simply find the difference between the desired response and the output of the adaptive filter.

$$e(n) = y(n) - d(n) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 3$$

4. To update the filter coefficients, multiply the error by μ, the learning rate parameter and then multiply the result by the filter input and add this result to the values of the previous filter coefficients.

$$\vec{w}(n+1) = \vec{w}(n) + \mu . e(n).\vec{x}(n) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 4$$

Where

μ: is the step size of the adaptive filter

$\vec{w}(n)$: Is the filter coefficients vector

$\vec{x}(n)$: Is the filter input vector

Then LMS algorithm calculates the cost function J (*n*) by using the following equation:

$J(n) = e^2(n)$ ……………………………………………..5

Where e2 (n) is the square of the error signal at time *n*

We may write the result in the form of three basic relations as follows:

1. Filter output:

$Y(n)=x(n)*w^T(n)$……………………………………..……6

2. Estimation error or error signal:

$e(n)=d_n - y_n$…………………………………………...…7

3. Tap weight adaptation:

$W_{n+1} = w_n+\mu*e(n)*x(n)$……………………………….……8

Equations 7 and 8 define the estimation error e(n), the computation of which is based on the current estimate of the tap weight vector w(n). Note that the second term, μ * e(n) *x(n) on the right-hand side of equation 8 represents the adjustments that are applied to the current estimate of the tap weight vector w(n). The iterative procedure is started with an initial guess w(0). The algorithm described by equations 7 and 8 is the complex form of the adaptive IIR based least mean square (LMS) algorithm. At each iteration or time update, this algorithm requires knowledge of the most recent values u(n), d(n) w(n). The LMS algorithm is a member of the family of stochastic gradient algorithms. When the LMS algorithm operates on stochastic inputs, the allowed set of directions along which we "step" from one iteration to the next is quite random and therefore cannot be thought of as consisting of true gradient directions.

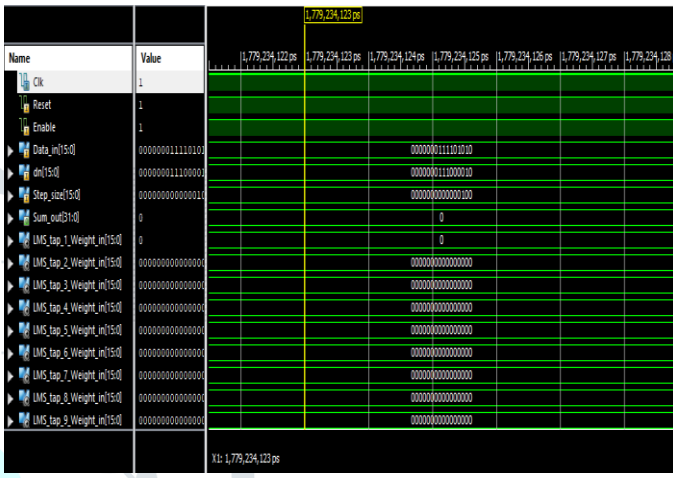
IV. IMPLEMENTAION RESULTS, ANALYSIS AND COMPARISIONS

Simulation result of LMS adaptive filter

Figure 3 simulation result when Reset=1 here the inputs are: Clk, Reset Enable, Data_in, dn, Step_size, Sum_out is the output of the system Duty cycle=50%, Period is taken as1000.Data_in is the input signal which consists noise. "dn" is the reference (desired signal) signal in which noise is not present. dn is less than the Data_in value.

Initially by setting the Reset as 1, it gives all LMS tap output and Sum_out as zero. Next by setting Reset as 0, the adaption of weights occurs. Finally, we are getting Sum_out which is the total output after passing the signal 63 times as filter length is chosen as N=64. Sum_out is the total output value which is nearby dn value in which noise is reduced

and obtaining an output signal as free from noise. Sum_out = LMS_tap_63_out;



Name	Value
Clk	1
Reset	1
Enable	1
Data_in[15:0]	000000011110101
dn[15:0]	000000011100001
Step_size[15:0]	00000000000001(
Sum_out[31:0]	0
LMS_tap_1_Weight_in[15:0]	0
LMS_tap_2_Weight_in[15:0]	000000000000000
LMS_tap_3_Weight_in[15:0]	000000000000000
LMS_tap_4_Weight_in[15:0]	000000000000000
LMS_tap_5_Weight_in[15:0]	000000000000000
LMS_tap_6_Weight_in[15:0]	000000000000000
LMS_tap_7_Weight_in[15:0]	000000000000000
LMS_tap_8_Weight_in[15:0]	000000000000000
LMS_tap_9_Weight_in[15:0]	000000000000000

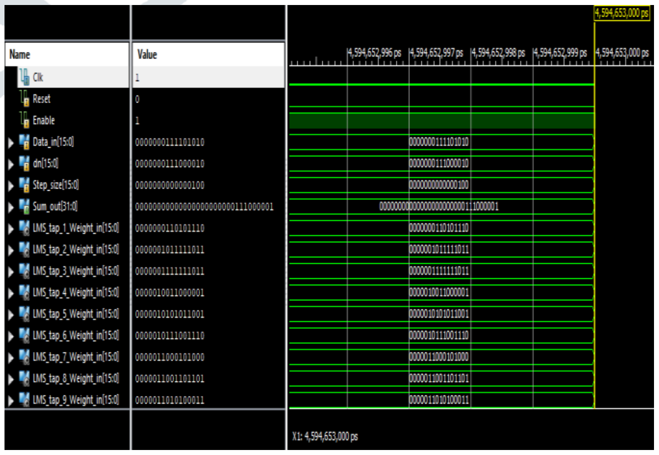
Figure 3: simulation result when Reset=1

ADVANTAGES:

- Low delay
- Noise removable capacity is improved.
- Low power consumption

APPLICATIONS:

- Audio and speech applications

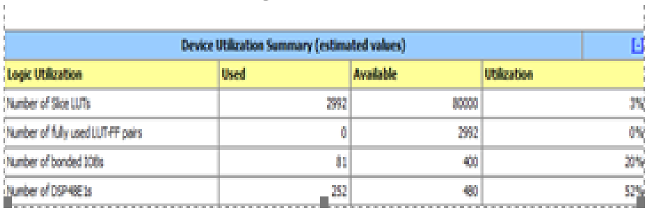


Name	Value
Clk	1
Reset	0
Enable	1
Data_in[15:0]	0000000111101010
dn[15:0]	0000000111000010
Step_size[15:0]	0000000000000100
Sum_out[31:0]	00000000000000000000000111000001
LMS_tap_1_Weight_in[15:0]	0000000110101110
LMS_tap_2_Weight_in[15:0]	0000001011111011
LMS_tap_3_Weight_in[15:0]	0000001111111011
LMS_tap_4_Weight_in[15:0]	0000010011000001
LMS_tap_5_Weight_in[15:0]	0000010101011001
LMS_tap_6_Weight_in[15:0]	0000010111001110
LMS_tap_7_Weight_in[15:0]	0000011000101000
LMS_tap_8_Weight_in[15:0]	0000011001101101
LMS_tap_9_Weight_in[15:0]	0000011010100011

Figure 4: Simulation result when Reset=0

Figure 4 shows the simulation result of 64 TAP LMS adaptive filter when Reset is given as zero. By setting Reset as zero the adaption of weights is taking place and giving final output Sum_out in which noise is reduced.

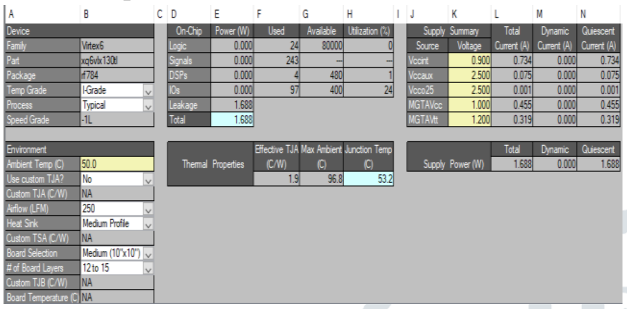
Table 1: Total area of design



Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	2992	80000	3%
Number of fully used LUT-FF pairs	0	2992	0%
Number of bonded IOBs	81	400	20%
Number of DSP48E1s	252	480	52%

Table 2 shows the total area occupied by the design. The use of LMS adaptive filter occupies the less area.

Table 2: Total power.



Device	
Family	Virtex6
Part	xq6vlx130tl
Package	rf784
Temp Grade	I-Grade
Process	Typical
Speed Grade	-1L

Environment	
Ambient Temp (C)	50.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	250
Heat Sink	Medium Profile
Custom TSA (C/W)	NA
Board Selection	Medium (10"x10")
# of Board Layers	12 to 15
Custom TJB (C/W)	NA
Board Temperature (C)	NA

On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	24	80000	0
Signals	0.000	243	---	---
DSPs	0.000	4	480	1
IOs	0.000	97	400	24
Leakage	1.688			
Total	1.688			

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	1.9	96.8	53.2

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	0.900	0.734	0.000	0.734
Vccaux	2.500	0.075	0.000	0.075
Vcco25	2.500	0.001	0.000	0.001
MGTAVcc	1.000	0.455	0.000	0.455
MGTAVtt	1.200	0.319	0.000	0.319

	Total	Dynamic	Quiescent
Supply Power (W)	1.688	0.000	1.688

Table 2shows the total power consumption by the design.

Checking the result using MATLAB:

To check the result how effectively its working MATLAB is used. Plot input signal and desired signal with respect to time. The Sum_out is stored in output.txt file. Plotting the output.txt file in MATLAB we get the final output graph. The output graph represents that it consists only a few amounts of noise and remaining signal is free from noise.

MATLAB output: Input signal

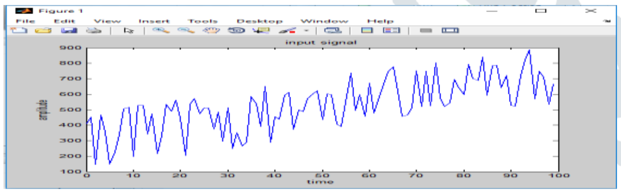


Figure 5: input signal

Figure 5 shows the input signal graph which consists of noise.

MATLAB output: desired signal

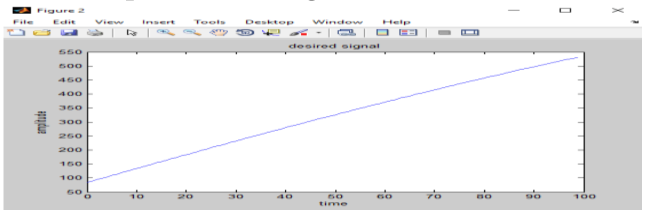


Figure 6: desired signal.

Figure 6 shows the desired signal graph which has no noise.

MATLAB output: output graph

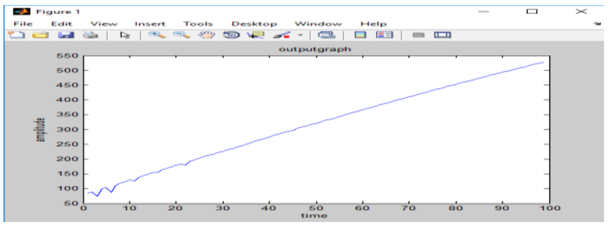


Figure 7: output graph

Figure 7 shows the output graph in which only few amounts of noise is present, remaining signal is free from noise.

V. CONCLUSION

This IIR adaptive filter using LMS reduces area delay product and energy delay product with pipelined structures. This design is performed for the filter length n=64. This design is synthesized and analyzed with the help of Xilinx ISE 14.7. Checking the result how much effectively working using MATLAB is done. Here only a few percent of noise are present, remaining signal is free from the noise. So, we can conclude that LMS adaptive filter gives the output which is almost free from the noise.

VI. REFERENCES

[1]Pramod Kumar Meher, Sang Yoon Park, Area-Delay-Power Efficient Fixed-Point LMS Adaptive Filter With Low Adaptation delay.

[2] R.B. Blackman, Linear Data Smoothing and Prediction in Theory and Practice, Addison-Wesley Pub. Co., 1965.

[3] Franklin F. Kuo, James F. Kaiser, System Analysis by Digital Computer, Wiley, 1967.

[4] G. Tufte, P. Haddow, "Envolving an Adaptive Digital Filter", EH'00, pp. 143-150.

[5] Knut Arne Vinger, Jim Torresen, Implementing Evolution of FIR Filters Efficiently in an FPGA.

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

[7] M. D. Meyer, D. P. Agrawal, "A modular pipelined implementation of a delayed LMS transversal adaptive filter", Proc. IEEE Int. Symp. Circuits Syst., pp. 1943-1946, May 1990.

[8] M. D. Meyer, D. P. Agrawal, "A high sampling rate delayed LMS filter architecture", IEEE Trans. Circuits Syst. II Analog Digital Signal Process., vol. 40, no. 11, pp. 727-729, Nov. 1993.

[9] K. Babulu, M. Kamaraju, P. Bujjibabu, K. Pradeep, "Design and implementation of H/W efficient Multiplier: Reversible logic gate approach", presented at the IEEE ICCSP 2015 conference, 2015.