ISSN: 0975-766X
CODEN: IJPTFI
Available Online through *Research Article*
www.ijptonline.com

REAL TIME AUDIO DENOISING USING DIGITAL FIR FILTERS WITH FPGA IMPLEMENTATION

[1]V Elamaran, [2]Har Narayan Upadhyay, [1]N Raju, [1]K Narasimhan
[1]Assistant Professor, Dept., of ECE, SEEE, SASTRA University, Thanjavur, Tamil Nadu, India.
[2]Associate Dean, Dept., of ECE, SEEE, SASTRA University, Thanjavur, Tamil Nadu, India.
Email: elamaran@ece.sastra.edu

Received on 05-11-2015 *Accepted on 23-12-2015*

Abstract

The need for audio signal processing arises from the fact that all the systems that we deal in real are not ideal, as a result the signal gets corrupted and may loss the information embedded in it. Signals have to be processed so that the information that they contain can be displayed, analyzed, or converted to another type of signal that may be of use. The complexity of analog filter implementation can now be overcome by employing digital filters. In this study, a real time implementation of FIR digital filters on Altera DE1 FPGA Kit is done using Verilog Hardware Description Language to analyze the results of audio enhancement. The audio signal is sampled using audio codec and then fed to the 2,4,8,16 and 32 tap FIR digital filters which are implemented on FPGA for audio enhancement. A noise generator is executed on FPGA to corrupt the signal and it is filtered to recover the audio signal. We simulate the code using Altera Quartus II synthesis software and the compilation reports are generated to know the FPGA resource utilization summary.

Keywords: Altera DE1, FIR filters, FPGA, Quartus II, Verilog HDL.

Introduction

Audio enhancement is becoming an important application with consumer electronics market. The high frequency components are considered as noise and can be eliminated most of the time in real time by designing suitable low pass digital filters. In technology like FPGAs, an implementation of digital FIR filters would be easy rather than the Infinite Impulse Response (IIR) Filters. Since there is a feedback from the output to the input in the case IIR filters, the implementation would be tedious. Also adaptive filters can be implemented to remove the noise using algorithms like least mean square (LMS) [1]. If the noise pattern follows a random pattern which is added with the signal, then adaptive

filters can be used. If the environment is totally a stochastic one, then Kalman filters would be a best choice to enhance the original audio signal [1] [2].

Filters are termed as a system which select a particular band of frequency from the available pool of frequencies. Now-a-days digital filters have wide applications in fields of communication, RADAR, aerospace, medicine, voice, image and so on. This popularity of digital filters over analog filters is because of its higher precision, better stability and reliability [3]. Matching problems does not occur in digital filters. Digital filters are implemented by software programming which offers flexibility. Digital filters are classified into two categories based on the impulse response. The one which has a finite impulse response is referred as FIR and the one which is having an infinite impulse response is referred as IIR. IIR filter has recursive structure which makes it capable of getting better frequency selection characteristics by use of less orders but it should be compensated for the nonlinear phase characteristics [4]. The presence of feedback makes the system more vulnerable to instability. The advantage of FIR filter over IIR is that it meets the strict requirements of amplitude and phase characteristics. FIR filters can be easily implemented, have precise linear phase characteristics and offers high stability [5].

With the advent of VLSI technology FPGAs (Field Programmable Gate Array) are being increasingly used for a wide range of computational applications. The popularity is because of its programmability, high reliability and low cost. FPGA has large number of logic elements and interconnects [6]. These logic elements are programmed to implement the required function by connecting appropriate routing switches. This study conveys the resource utilization summary about the logic interconnects are programmed to implement the logic functions. At the end the resource utilization summary is obtained for the Altera DE1 Cyclone II EP2C20F484C7 FPGA [7].

Materials and Methods

Digital fir Filter Design

The general time domain difference equation for the FIR filter is as follows in Equation (1),

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \qquad (1)$$

Where y(n) is the output, x(n-k) is the input sample delayed by k units, h(k) is the impulse response coefficients, and M is the number of filter tap [8]. The above equation involves M multiplications and M+1 additions per sample to compute

the result. This delay is implemented by having M-1 FIFO where each buffer acts as a unique delay implement. The multiplication of delayed sample with filter coefficient is achieved by shifting the bits in the buffer. These FIR filters have only zeros. They are physically stable since all implicit poles are located on zero. The way zero is placed on the z-plane diagram determines the frequency response characteristic of a filter. If a zero is located at –0.5, the distance from the zero to the lower frequency components is high. So it act as a low-pass filter [9]. If there is more distance, there is more gain for those frequency components. So, it is a low-pass filter in this case. Similarly, if a zero is located on +0.5, the distance from the zero to the higher frequency components would be very high. So, it can act as a high-pass filter.

Digital Fir Filters – 2, 4, 8, AND 16 TAP

The 2-tap digital filter is first implemented with simple impulse response coefficients as {1/2,1/2}. This is shown in Figure 1. By changing the alternate coefficient with minus sign, a 2-tap high pass filter can be implemented. Similarly by analyzing pole-zero pattern, band-pass, band-reject and notch filters can be designed easily. Simulation software tools like Matlab/Simulink can be used to identify the filter coefficients. Filter Design Analysis (FDA) tool from Matlab is an easy approachable method to determine the filter coefficients for the given specifications. This tool helps to view the impulse response plot, pole-zero diagram, impulse response coefficients, phase and group delay response, etc [10].

The FPGA audio codec samples the audio analog input with 24 bits per sample. Schematic designs for 2-tap, 4-tap, 8-tap and 16-tap digital FIR filters are shown in Figure 1, 2, 3, and 4 respectively.

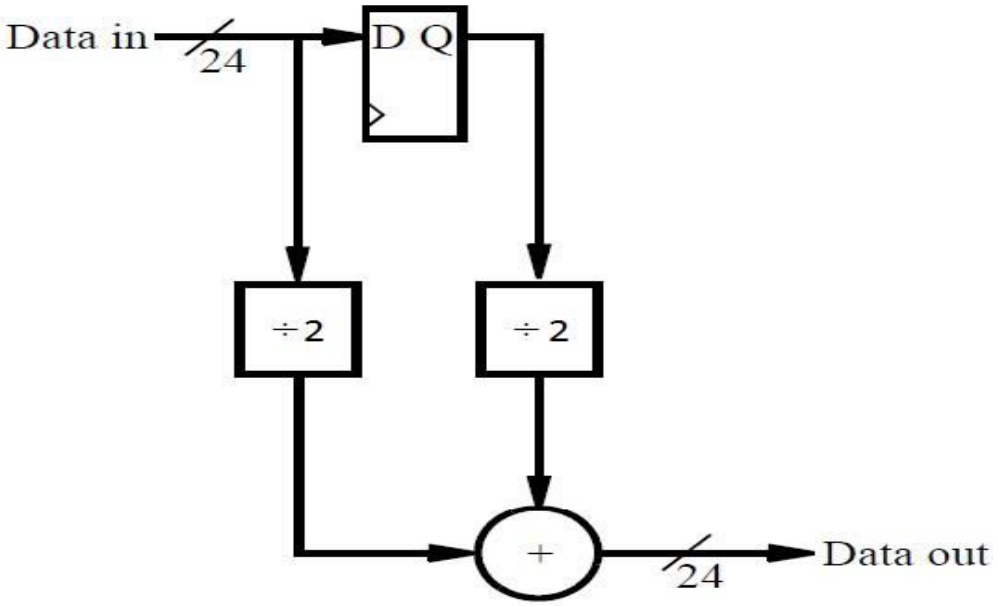

Figure 1: A 2-Tap Digital FIR filter.

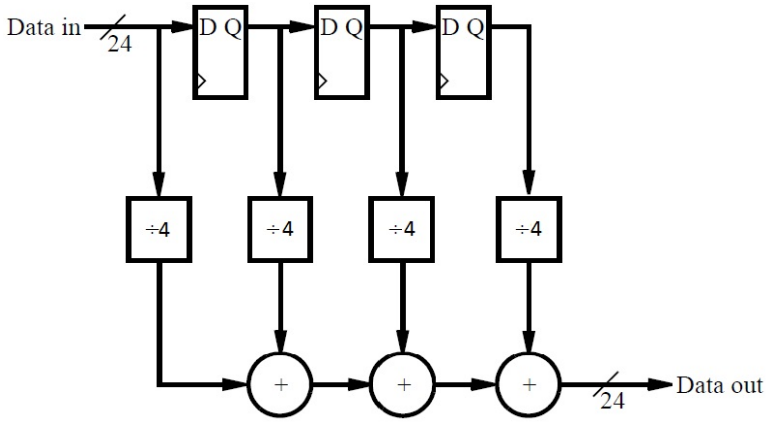

Figure 2: A 4-Tap Digital FIR filter.

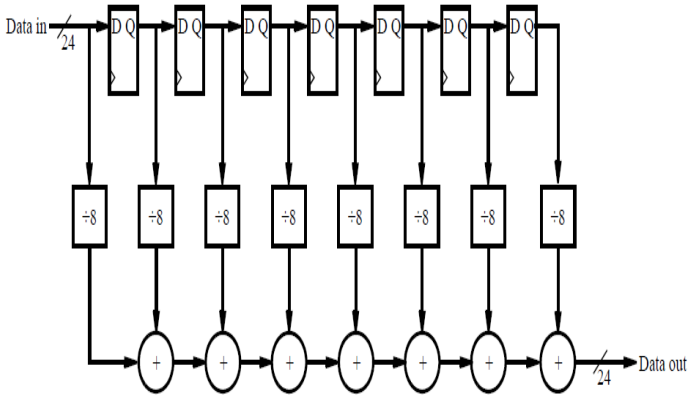

Figure 3: An 8-Tap Digital FIR filter.

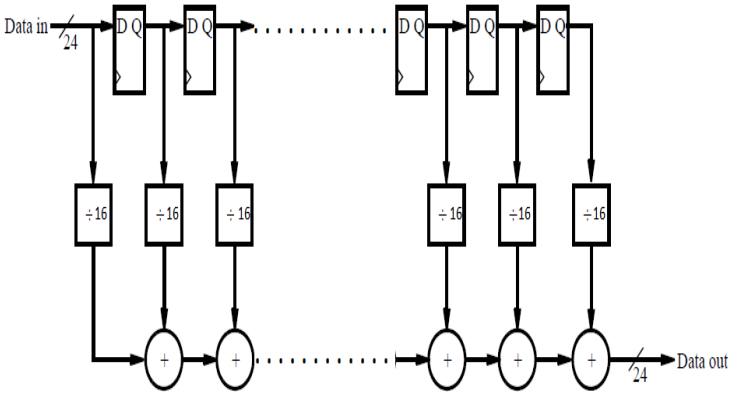

Figure 4: A 16-Tap Digital FIR filter.

Noisy Data Generator

To test these low-pass FIR filters, a noisy signal is being added along with the original input audio. This noisy signal is generated using Verilog Hardware Description Language (HDL) as a subprogram of the main module [11] [12]. The Verilog HDL code for noisy data is written as follows:

```
module noisy_wave(clock, en, q);
        input clock, en;
```

```
output [23:0] q;
reg [2:0] count;
        always@(posedge clock)
                if (en)
                        count = count + 1'b1;
        assign q = {{10{count[2]}},count, 11'd0};
endmodule
```

The result of RTL viewer of this noisy wave module is shown in Figure 5.

This noisy generator module can be used to verify the filtering operation of these low-pass digital FIR filters. This noise effect should be minimized with the help of filters. If the noise environment follows a stochastic process, then this kind of conventional FIR filters do not fit for filtering. In such a scenario, adaptive FIR filters can be implemented.

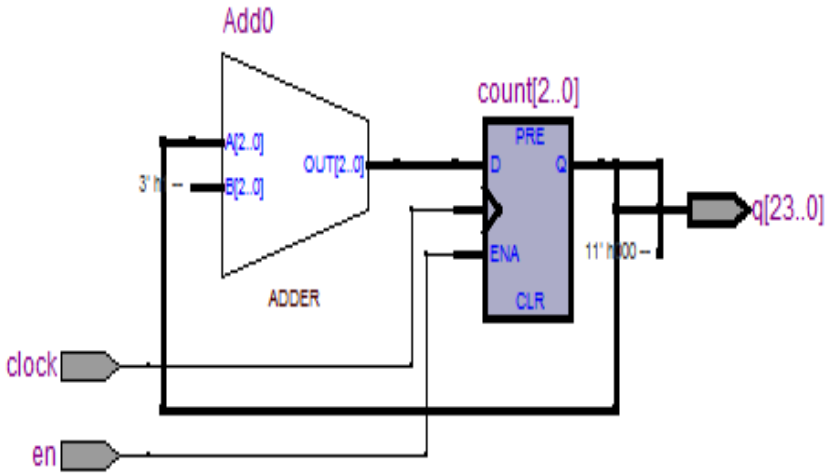

Figure 5: RTL view of noise generator.

Results

All the module results are presented along with each resource utilization summary report. Module 1 capture the sampled analog audio input and does no processing i.e., filtering. Here the quantized output audio is produced. Module 2 again does no filtering but the noisy data is added. Module 3 implements 8-tap low-pass filter to remove the noise. Module 4 and 5 implement the 16-tap and 32-tap low-pass FIR filters. These all kind of filters are nothing simple moving average type.

RTL Netlist and Compilation Report Summary

All the modules are implemented with Altera FPGA EP2C20F484C7 using Verilog HDL. The RTL view netlists for the modules 1, 2, and 3 are shown in Figure 6, 7, and 8 respectively. Modelsim tool is used for simulation and Quartus II software is used for synthesis and real-time implementation in FPGA chip [13] [14]. Only few module RTL view netlists

are presented here. To obtain noise as output, a high-pass filter is designed and the effect is tested. By connecting these low-pass and high-pass filters appropriately, a band pass and/or band-reject filters can be designed [15] [16].

The resource utilization summary report for no filter, noisy data with no filter, 8-tap, 16-tap and 32-tap FIR filters are shown in Tables 1, 2, 3, 4 and 5 respectively. The total number combinational logic functions are the same as the total number of logic elements for all the modules.

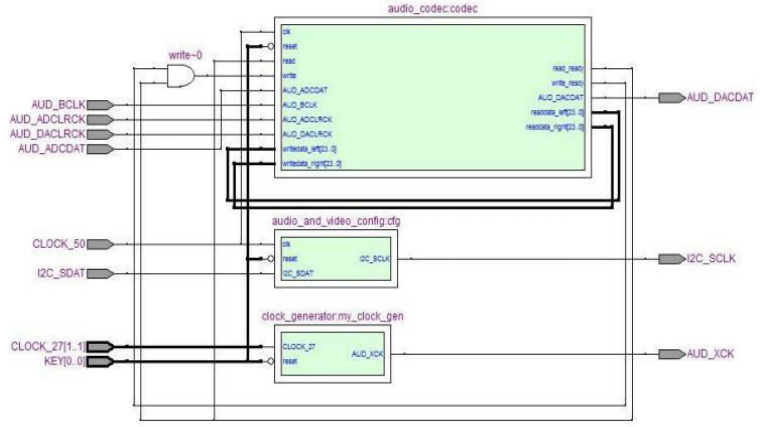

Figure 6: RTL view of the design without filter.

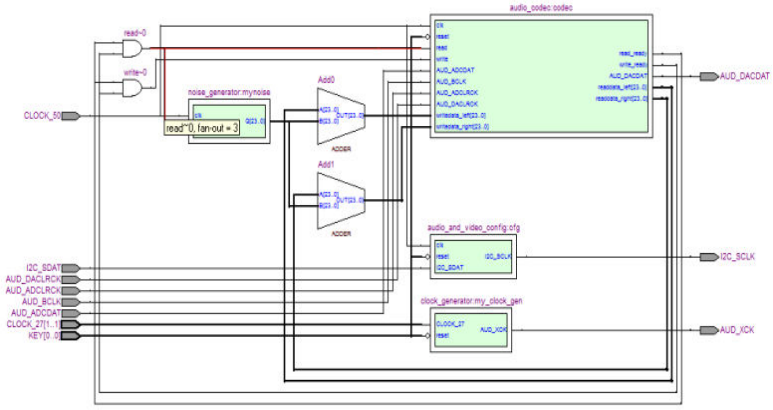

Figure 7: RTL view of noisy data with no filter.

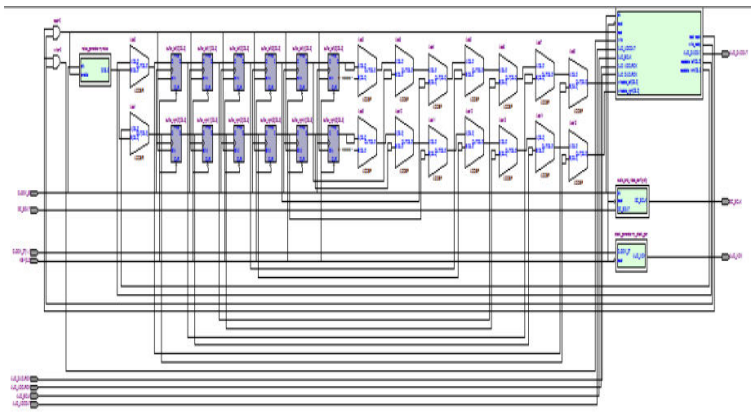

Figure 8: RTL view of 8-tap FIR filter.

Signal-to-Noise Ratio Improvement Results

The input and filter output audio samples are recorded and analyzed with signal-to-noise ratio (SNR) using Matlab programs. Table 6 shows the SNR improvement which is calculated by subtracting the input SNR from the output SNR for the modules 3, 4, and 5.

Table-1: Compilation report of Module 1 (No Filter).

Description	Available resources	Used	Utilization
Pins	315	11	3 %
Logic Elements	18752	517	3 %
Registers	18752	293	2 %
Memory Bits	239616	12288	5 %

Table-2: Compilation report of Module 2 (Noisy data, no filter).

Description	Available resources	Used	Utilization
Pins	315	11	3 %
Logic Elements	18752	546	3 %
Registers	18752	296	2 %
Memory Bits	239616	12288	5 %

Table-3: Compilation report of Module 3 (8-Tap FIR filter).

Description	Available resources	Used	Utilization
Pins	315	11	3 %
Logic Elements	18752	901	3 %
Registers	18752	821	4 %
Memory Bits	239616	11520	5 %

Table-4: Compilation report of Module 4 (16-Tap FIR filter).

Description	Available resources	Used	Utilization
Pins	315	11	3 %
Logic Elements	18752	1381	7 %
Registers	18752	856	2 %
Memory Bits	239616	11264	5 %

Table-5: Compilation report of Module 5 (32-Tap FIR filter).

Description	Available resources	Used	Utilization
Pins	315	11	3 %
Logic Elements	18752	1582	3 %
Registers	18752	912	2 %
Memory Bits	239616	12287	5 %

Table-6: SNR improvement comparison.

Module	Input SNR (dB)	Ouput SNR (dB)	SNR improvement (dB)
Module 3	–0.5379	–0.2304	0.3075
Module 4	–0.5379	–0.1437	0.3942
Registers	–0.5379	–0.0457	0.4922

Results and Discussions

Simulation results show that the better results are produced with higher-order FIR filters. This is obvious but with more computational resources. All modules have the impulse response coefficients as 1/N, if the N is the number of taps for a filter. These coefficients can be modified and tested with the help of FDA tool and will be implemented in FPGA. This work can be further extended to implement an adaptive FIR filter for the stochastic noise environment. An existing noise-generator module may also be modified to generate a different random pattern of samples like a white noise. To improvise the design in a better way with shorter time, advanced tools like DSP Builder can be used.

Acknowledgement

We thank SASTRA University for providing financial support for this research work under the Research and Modernization fund – R&M/0034/SEEE-014/2013-14.

References:

1. Li Tan, Digital Signal Processing: Fundamentals and Applications, Academic Press, USA, 2007.
2. Sanjit K.Mitra, Digital Signal Processing – A Computer Based Approach, Mc-Graw Hill. 2007.
3. Edumand Lai, Practical Digital Signal Processing for Engineers and Technicians, Newnes, 2004.
4. John G.Proakis, Dimitris G.Manolakis, Digital Signal Processing – Principles, Algorithms, and Applications, Pearson Education, 2006.
5. T.J.Terrel, S.Lik-Swan, Digital Signal Processing – A Student Guide, Macmillan, 1996.
6. V.Elamaran and G.Rajkumar, FPGA Implementation of Point Processes using Xilinx System Generator, Journal of Applied Theoretical and Applied Information Technology, 2012, Vol 41, No 2, pp.201-206.
7. Chao Cheng, Keshab K.Parhi, High Speed VLSI Implementation of 2-D Discrete Wavelet Transform, IEEE Transactions on Signal Processing, 2008, Vol 56, No 1, pp.393-403.
8. Bernard Mulgrew, Peter Grant, John Thompson, Digital Signal Processing: Concepts and Applications, Palgrave Macmillan, 2003.

9. Roberto Cristi, Modern Digital Signal Processing, Brooks&Cole publishers, 2004.

10. Emmanuel C.Ifeachor, Barrie W.Jervis, Digital Signal Processing – A Practical Approach. Prentice Hall of India, 2002.

11. Samir Palnitkar, Verilog HDL – A Guide to Digital Design and Synthesis, Pearson Education, 2003.

12. B.Stephen Brown, V.Zvonko, Fundamentals of Digital Logic with VHDL Design, Mc-Graw Hill, 2005.

13. S Saadi, M Touzia, Guessoum, FPGA Implementation of 2D signals encoder using QMF based dyadic DWT: Application to neutron tomography projections, Journal of Applied Sciences, 2007, Vol. 7, No. 11, pp. 1528-1533.

14. Yazhini and R.Ramesh, FIR Filter Implementation using Modified Distributed Arithmetic Architecture, Indian Journal of Science and Technology, 2013, Vol. 5, No. 6, pp.4485-4491.

15. V.Elamaran, A.Aswini, V,Niraimathi and D.Kokilavani, FPGA Implementation of Audio Enhancement using Adaptive LMS Filters, Journal of Artificial Intelligence, 2012, Vol 5, No 4, pp.221-226.

16. Uwe Meyer-baese, Digital Signal GProcessing using Field Programmable Gate Arrays, Springer, 2001.

Corresponding Author:

V Elamaran*,

Email: *elamaran@ece.sastra.edu*