Design and Implementation of Adaptive filtering algorithm for Noise Cancellation in speech signal on FPGA

(Mrs. A. B. Diggikar)
Electronics and Telecommunication Engg. Department,
SPWEC
Aurangabad, India
anujakulkarni50@gmail.com

(Mrs. S. S. Ardhapurkar), Member, IEEE
Electronics and Telecommunication Engg. Department,
ICEEM
Aurangabad, India
vaidyashubha21@gmail.com

Abstract— **In recent years FPGA systems are replacing dedicated Programmable Digital Signal Processor (PDSP) systems due to their greater flexibility and higher bandwidth, resulting from their parallel architecture. This paper presents the applicability of a FPGA system for speech processing. Here adaptive filtering technique is used for noise cancellation in speech signal. Least Mean Squares (LMS) , one of the widely used algorithm in many signal processing environment , is implemented for adaption of the filter coefficients. The cancellation system is implemented in VHDL and tested for noise cancellation in speech signal. The simulation of VHDL design of adaptive filter is performed and analyzed on the basis of Signal to Noise ratio (SNR) and Mean Square Error (MSE).**

Keywords- Adaptive Filter, LMS Algorithm , Active Noise cancellation, VHDL Design, SNR, MSE

I. INTRODUCTION

The digital signal processing applications impose considerable constrains on area, power dissipation, speed and cost. Thus the design tool should be carefully chosen. The most common tools for the design of such application are ASIC, DSP and FPGA . The DSP used for extremely complex math-intensive tasks but can't process high sampling rate applications due to its serial architecture. Whereas ASIC faces lack of flexibility and require long design cycle. The FPGA (Field programmable Gate Array) can make up disadvantages of ASIC and DSP. Hence FPGA has become the best choice for the design of signal processing system due to their greater flexibility and higher bandwidth, resulting from their parallel architecture.

This paper investigates the applicability of a FPGA system for real time audio processing systems. In recent years , acoustic noises become more evident due to wide spread use of industrial equipments. An Active (also called as Adaptive) noise cancellation (ANC) is a technique that effectively attenuates low frequencies unwanted noise where as passive methods are either ineffective or tends to be very expensive or bulky. An ANC system is based on a destructive interference of an anti-noise, which have equal amplitude and opposite phase replica of primary unwanted noise. Following the superposition principle, the result is noise free original sound [1,6].

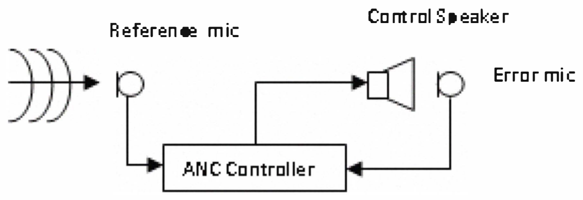


Figure 1. Basic concept of ANC

ANC systems are distinguished by their different goals that lead to different architectures. If all ambient sound shall be reduced, a feedback system with its simpler architecture may be used. If, as in our case, single sources of unwanted sound shall be compensated, a feed forward system is required. A feed forward system as shown in fig. 1 is characterized by two audio inputs per channel: one reference signal input for the sound to be removed, and second error input for the sound after the compensation.

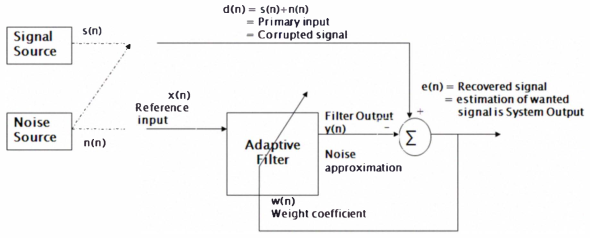


Figure 2. Adaptive Noise Canceller

An adaptive FIR feed forward system is shown in simple way in fig 2. for the selective cancellation of disturbing noise without affecting other sounds. [2] It is dual input system. The first inputs is primary signal d(n) which is wanted signal (say s(n)) corrupted by noise (say n(n)). The second input is reference signal x(n) can be interfacing noise supposed to be uncorrelated with the wanted signal but correlated with noise affecting original signal in an unknown way. The filter output signal y(n) is an estimate of the noise signal with inverted sign. This signal and the primary signal are superposed, so that the noise signal is cancelled and error signal e(n) is the result of this superposition which constitutes the overall system output. The adaptive filtering operation achieved the best results when system output is noise free. This goal is achieved by minimizing the mean square of the error signal [3]. The widely preferred LMS algorithm is used for the adaption of the filter coefficients.

A. Mathematical treatment

Consider the transversal filter with input x(n) i.e. vector of the M(filter length) most recent input samples at sampling point n.

$$x(n) = [x(n), x(n-1), \ldots x(n-M+1)] \tag{1}$$

and w(n) i.e. vector of filter coefficients as

$$w(n) = [w_0(n), w_1(n), \ldots w_{M-1}(n)] \tag{2}$$

At some discrete time n, the filter produces an output y(n) which is linear convolution sum given by

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

Also can be represent in vector form as

$$y(n) = w^T(n)\,u(n) \tag{3}$$

The error signal is difference of this output with the primary signal d(n) given by ,

$$e(n) = d(n) - y(n) \tag{4}$$

And by squaring error we get

$$e^2(n) = d^2(n) - 2d(n)x^T(n)w(n) + w^T(n)x(n)x^T(n)w(n) \tag{5}$$

To optimize the filter design, we choose to minimize the mean-square value of e(n). Thus the cost function is defined as the MSE denoted by J

$$\begin{aligned} J &= E[e(n)e^*(n)] \\ &= E[|e(n)|^2] \end{aligned} \tag{6}$$

Where E denotes the statistical expectation operator.

Applying the operator ∇ to the cost function J, a gradient vector ∇ J obtain as

$$\begin{aligned} \nabla J(n) &= -2P + 2Rw(n) \\ &= -2x(n)d(n) + 2x(n)x^T(n)w(n) \end{aligned} \tag{7}$$

where,

R is the autocorrelation matrix of x(n), and P is the cross correlation matrix of d(n) and x(n).

The LMS algorithm is based on steepest-descent method. To formulate steepest-descent method, consider a cost function J(w) i.e. a continuously differentiable function of some unknown weight vector w.

To find an optimal solution w_0 (initial guess) that satisfies the condition

$$J(w_0) \le J(w) \quad \text{for all } w \tag{8}$$

Which is a mathematical statement of unconstrained optimization.

Starting with w(0), generate a sequence of weight vector w(1), w(2),….,such that the cost function J(w) is reduced at each iteration of the algorithm. therefore

$$J(w(n+1)) < J(w(n)) \tag{9}$$

Where w(n) is the old value of the weight vector and w(n+1) is its updated value

Substituting the estimate of equation 7 for the gradient vector $\nabla J(n)$ in the steepest-descent algorithm, a new recursive relation obtain for updating the weight vector as.

$$w(n+1) = w(n) + \mu x(n)e(n) \tag{10}$$

A scaling factor μ introduced here is step size parameter used to control the step width of the iteration and thus the stability and convergence speed of the algorithm[4,5].

The LMS algorithm is convergent in mean square if and only if μ satisfies the condition.

$$0 < \mu < \frac{1}{(M x \bar{P})} \tag{11}$$

where, $\bar{P}$ is the average power of tap inputs and M is Filter length.

II. VHDL IMPLEMENTATION OF SYSTEM

The VHDL design of the system is as shown in Fig. 3. Arithmetic is modeled with Q format number representation which provides for each pipeline stage an appropriate number of guard bits for representing the integer part and avoiding overflow effects.

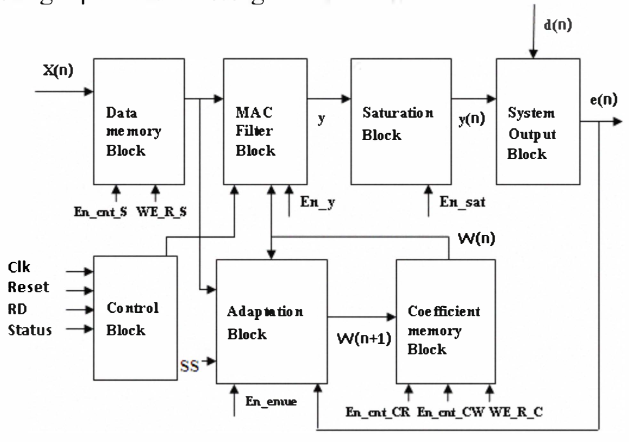


Figure 3. VLSI Design of the adaptive filter

The design splits into seven blocks as follows :

1. Data Memory block : The single port RAM is designed for storage of the audio samples.

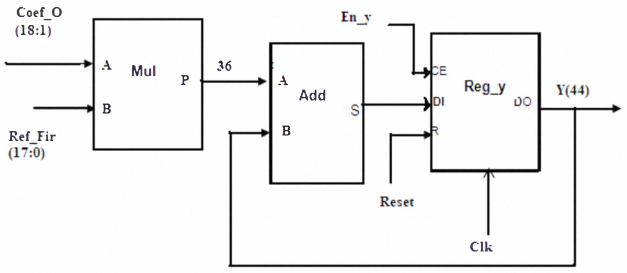


Figure 4. FIR Filter Block

2. MAC Filter Block : The FIR filter design is based on the transposed direct form in order to keep the maximum data path length short, The filter is implemented as a sequential MAC unit which performs M accumulations of products during every sample period so that a resource sharing can be utilized. since the audio sample period f_S provides a large amount of available clock cycles per audio sample, no parallel structure with M multipliers and M-1 adders is necessary.

   This block is designed as three-stage pipeline for the filtering cycle The input samples read from the data RAM block are multiplied with their corresponding filter coefficient taken from the dual-ported Coefficient RAM block and stored in the accumulator.

3. Saturation Block : The filter output signal is fed to the saturation block, which prevents the filter output from overflow and inverts the sign of the output signal to provide the phase shift for the compensation step.
4. System output block : The Adder unit is used to implement equation of error signal e(n) from saturation block output y(n) and primary signal d(n). This output is the required system output.
5. Adaption Block algorithm : A four-stage pipeline structure designed for the adaptation of the coefficients. The coefficient is calculated by a product of the input sample (Ref_Fir) , the error signal (Err) and Step size parameter (SS). A register is inserted to this path that splits the arithmetic chain for achieving a shorter signal delay so that a clock frequency of f_{CLK} = 50MHz can be met.
6. Coefficients Memory block : This block designed for storage of the current filter coefficients. The dual port RAM is chosen to support a parallel processing of the coefficient update block and the FIR Filter block. With two address inputs the reading address of the coefficients and the address for writing back the updated coefficients can be incremented within two interleaved clock periods.
7. Control Block : The Control path functionality is implemented as the Finite state machine (FSM). The FSM controls the processing of the two parallel pipelined data paths. The state diagram of the FSM shown in fig. 6 describes a sequence which is started with each new input sample pair Err (error signal) and XN (reference signal). Total seven states are specified.

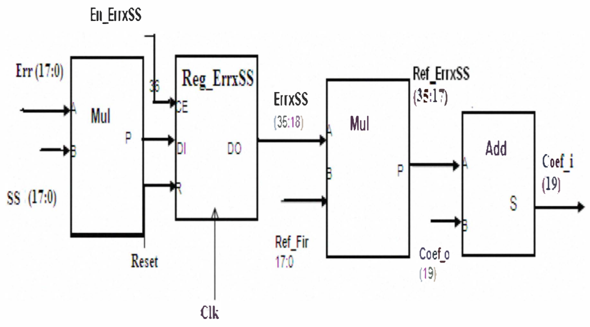


Figure 5. Coefficient Adaptation Unit

The FSM will go through the following sequence:

1. START : This is default state in which all registers clears results from the previous calculation cycle. Through an input RD , a new sample XN is stored in RAM by the signal WE_S.
2. ERRXSS: performs calculation of the product of error signal Err and step size factor SS and stored in the register Reg_ErrxSS by the signal En_ErrxSS.

3/4 FILTER / ADAPT : There is alternating sequence

of two pipeline operations runs in parallel. The filter block performs operations of updating address for reading input sample and coefficient, outputting memory and accumulating product of sample and Coefficient and saving in Register Reg_Y by the signal En_Y. The pipeline for the adaptation of the coefficients performs operations of updating address for reading input sample and coefficient, outputting memory, updating address for writing, Accumulation of a product from Ref_Fir and ErrxSS on the current coefficient and storage of the adapted coefficients.

The status Cnt_st indicates the highest reading at the address is coefficient present at dual port RAM.

5. STOP: It is the last multiplication of sample and coefficient and accumulation of the result by the signal En_Y. Then read address of coefficient from memory for the transition to a next state.
6. UPDATE: The accumulation result (filter output) stored in the output register by the signal

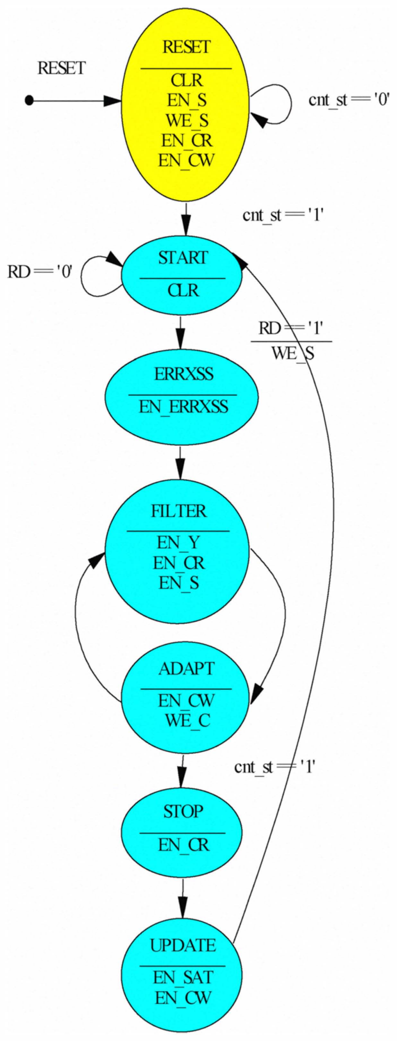


Figure 6. State diagram of the FSM of the adaptive filter

En_sat and fed to the saturation block, which holds this value for one sample period and performs the adjustment of the RAM address counters for the next sequence.

7. RESET : When system is Reset, state becomes Reset to reset all registers and content of RAMs.

III. SIMULATION AND RESULTS

Xilinx ISE 9.li development environment was used for implementation of above VLSI design. The design has been transferred to VHDL code and its the hardware simulation done with the Xilinx ISE simulator and also implemented on Spartan-3 FPGA XC3s400pq208-5 board.

The sound is taken as wave file for testing of system. A linear combination of the generated noise and the original signal is used as the primary input for the filter. The ref. signal is noise which is correlated version of noise added with original signal . Filter output is compensation signal denoted by y(t) which is noise approximation. The system output is error signal denoted by e(t) which is recovered signal. The Q17 format is used for arithmetic computations.

The figure 7 and 8 shows synthesis report and RTL (Register-transfer level) schematic of designed system respectively. The FPGA utilization reported is below 66 % for all resource categories for filter order of 256 (see fig. 14). Only 3 of the 16 available embedded multipliers were used and the transposed direct form FIR filter contains only one multiplier and one adder regardless of the filter order. The longest data path in the system is given by the multiply-add operations of the coefficient adaption and limits the clock frequency got up to 80MHz. The figure from 9 and 10 shows VHDL simulation for Adaptation block , FIR filter block respectively. The synthesis report and RTL schematic and simulation results of FSM control block are shown in figure 11 to 13 respectively.
The system tested for different sound signals. Here results with three different sound signals are presented (see fig. 15). Original and Recovered Signal obtains are same. The output Signal to noise ratio (SNR) and Mean square error (MSE) of the same signal was measured at fixed step size of 0.001 for analysis and is shown in table 1 given below . SNR of denoised signal is enhanced with minimum MSE.

TABLE 1 Performance analysis on the basis of SNR and MSE

Sr. No.	**SNR (dB) of Original Signal**	**SNR (dB) of denoised signal**	**MSE**
1	4.1731	23.9733	2.7684e-005
2	13.0461	34.1360	2.0341e-004
3	13.4070	28.9671	1.9350e-004

Since filtering and adaption are implemented as parallel processes, the number of cycles N required by the FSM is

$$N = 2M + 5 \tag{12}$$

The maximum realizable filter order depends on the available clock cycles within a sample period and the number of clock cycles , which are necessary to process one FSM cycle.

```
Macro Statistics
# FSMs                                            : 1
# RAMs                                            : 2
 16x18-bit single-port distributed RAM            : 1
 16x19-bit dual-port block RAM                    : 1
# Multipliers                                     : 3
 18x18-bit multiplier                             : 3
# Adders/Subtractors                              : 3
 18-bit adder                                     : 2
 19-bit adder                                     : 1
# Counters                                        : 3
 4-bit down counter                               : 1
 4-bit up counter                                 : 2
# Accumulators                                    : 1
 44-bit up accumulator                            : 1
# Registers                                       : 57
 Flip-Flops                                       : 57

=============================================================
```

Figure 7. Synthesis Report of VLSI design of the adaptive filter

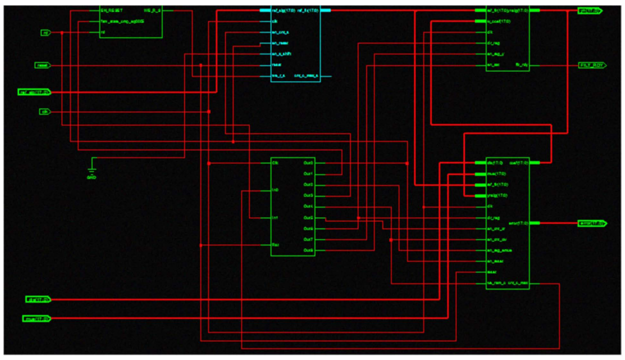


Figure 8. RTL Schematic in detail of design

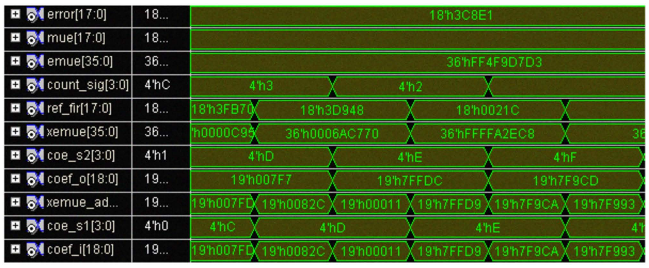

error[17:0] 18... 18'h3C8E1
mue[17:0] 18...
emue[35:0] 36... 36'hFF4F9D7D3
count_sig[3:0] 4'hC 4'h3 4'h2
ref_fir[17:0] 18... 18'h3FB70 18'h3D948 18'h0021C
xemue[35:0] 36... h0000C95 36'h0006AC770 36'hFFFFA2EC8
coe_s2[3:0] 4'h1 4'hD 4'hE 4'hF
coef_o[18:0] 19... 19'h007F7 19'h7FFDC 19'h7F9CD
xemue_ad... 19... 19'h007FD 19'h0082C 19'h00011 19'h7FFD9 19'h7F9CA 19'h7F993
coe_s1[3:0] 4'h0 4'hC 4'hD 4'hE
coef_i[18:0] 19... 19'h007FD 19'h0082C 19'h00011 19'h7FFD9 19'h7F9CA 19'h7F993

Figure 9. VHDL Simulation of the Coefficient Update Unit

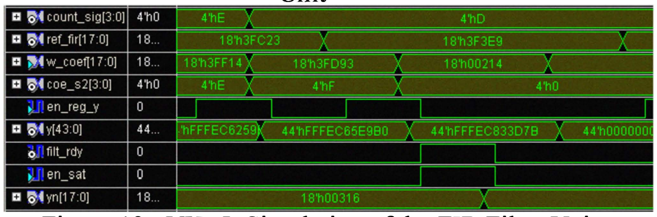

count_sig[3:0] 4'h0 4'hE 4'hD
ref_fir[17:0] 18... 18'h3FC23 18'h3F3E9
w_coef[17:0] 18... 18'h3FF14 18'h3FD93 18'h00214
coe_s2[3:0] 4'h0 4'hE 4'hF 4'h0
en_reg_y 0
y[43:0] 44... 'hFFFEC6259 44'hFFFEC65E9B0 44'hFFFEC833D7B 44'h0000000
filt_rdy 0
en_sat 0
yn[17:0] 18... 18'h00316

Figure 10. VHDL Simulation of the FIR Filter Unit

```
Found finite state machine <FSM_0> for signal <fsm_state>.
-----------------------------------------------------------------------
| States            | 7
| Transitions       | 10
| Inputs            | 2
| Outputs           | 9
| Clock             | clk (rising_edge)
| Reset             | reset (positive)
| Reset type        | synchronous
| Reset State       | z_reset
| Power Up State    | idle
| Encoding          | automatic
| Implementation    | LUT
-----------------------------------------------------------------------
```

Figure 11. FSM Synthesis Report

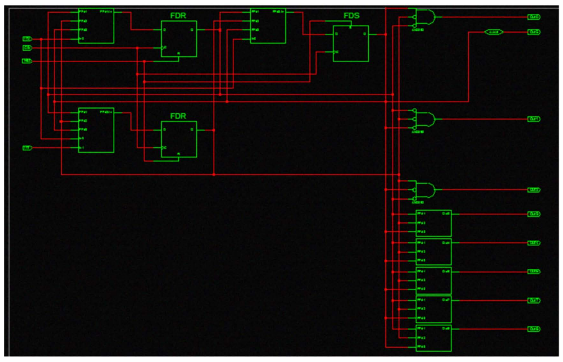
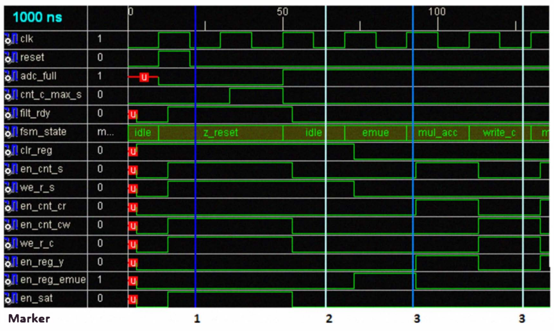


Figure 12. RTL Schematic and of FSM



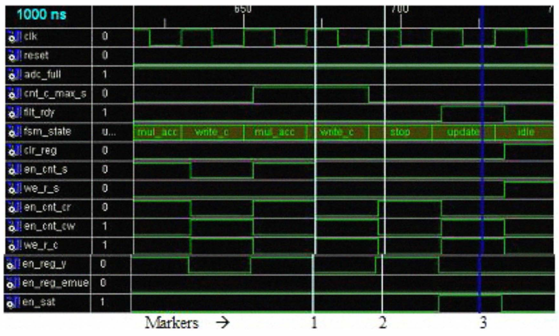
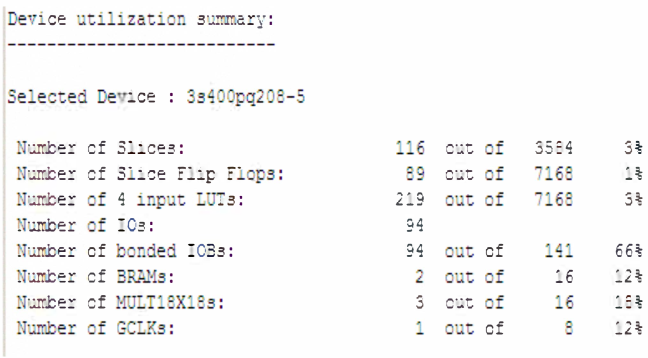


Figure 13. VHDL Simulation of the FSM Unit a) at the Start of the Machine Cycle b) at the End of the Machine Cycle.

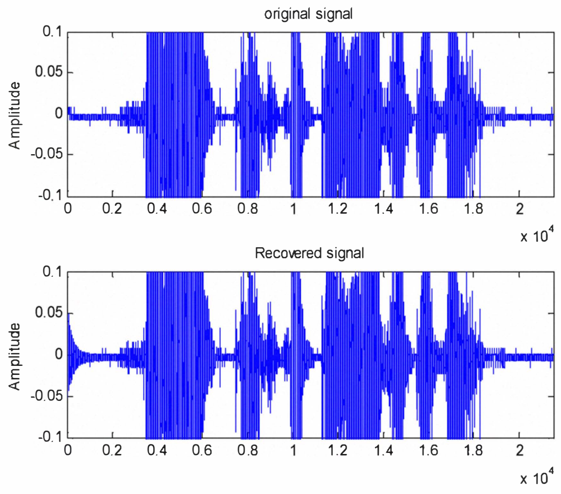


```
Device utilization summary:
---------------------------

Selected Device : 3s400pq208-5

 Number of Slices:                 116  out of   3584     3%
 Number of Slice Flip Flops:        89  out of   7168     1%
 Number of 4 input LUTs:           219  out of   7168     3%
 Number of IOs:                     94
 Number of bonded IOBs:             94  out of    141    66%
 Number of BRAMs:                    2  out of     16    12%
 Number of MULT18X18s:               3  out of     16    18%
 Number of GCLKs:                    1  out of      8    12%
```

Figure 14. Device Utilization summary



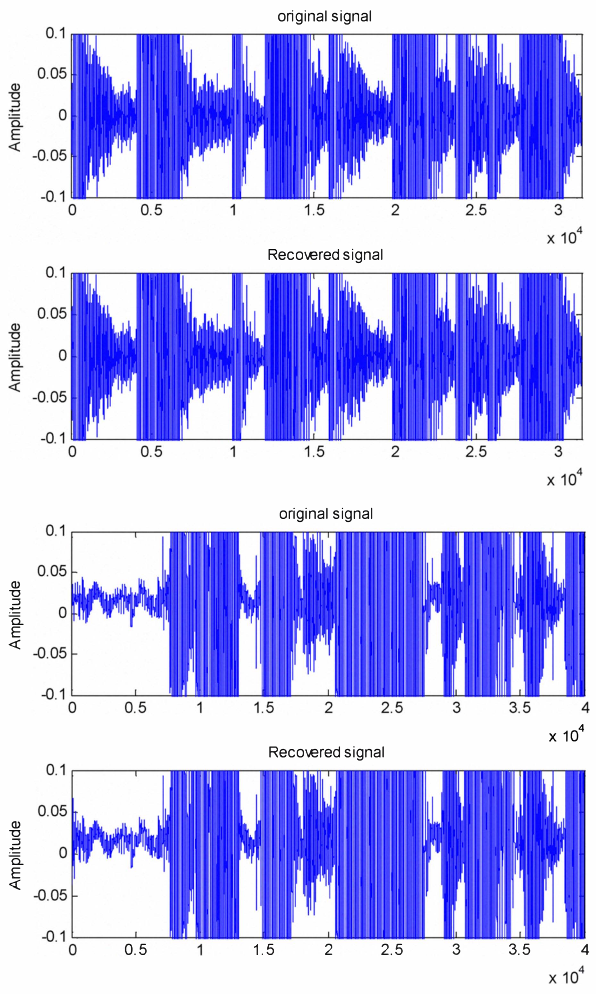


Figure 15. Original and Recovered Signal for three different inputs as in table 1 respectively

CONCLUSION

The FPGA platform is well suited for the complex real time audio processing. An adaptive noise cancellation process has successfully been implemented for filter order up to 256 using Spartan -3 FPGA XC3s400pq208-5 board.
When tested with different signals, the system showed an improved performance compared to the original signal. For future work, we planned to implement this system with AFA (Adaptive Filtering with Averaging) algorithm , which converge rapidly with lower complexity.

REFERENCES

[1] L. J. Eriksson, M. C. Allie, and C. D. Bremigan, “Active Noise Control using Adaptive digital Signal Processing “ in Proc. ICASSP , New York, 2004 pp. 2594-2597

[2] Dimitris G. Manolakis, Vinay K. Ingle, and Stephen M. Kogon, “Statistical and Adaptive Signal Processing”, McGraw- Hill, 2000.

[3] Simon Haykin. “Adaptive Filters Theory” Pearson Education, 2008.

[4] “Modified Adaptive Filtering Algorithm for Noise Cancellation in Speech Signals” -V. R. Vijaykumar, P. T. Vanathi & P. Kanagasapabathy ELEKTRONIKA IR ELEKTROTECHNIKA ISSN 1392 – 1215 2007. No. 2(74)

[5] C. Mosquera, J.A. Gomez “Adaptive Filters for Active Noise Control” , Sixth international congress on sound and vibration Copenhagen, Denmark

[6] Colin H. Hansen “ Understanding Active Noise Cancellation ” IOS Press – 2002