FINAL YEAR PROJECT REPORT

Noise Cancellation Controller On DSP KIT Using LMS Algorithm

Project Advisor

(Mr.Aqeel Arshad)

Submitted by

Hassan Raza 071020-013

Muhammad Arif 071020-021

Ali Anees 071020-027

Department of Electrical Engineering

School of Science and Technology

University of Management and Technology

Noise Cancellation Controller On DSP KIT Using LMS Algorithm

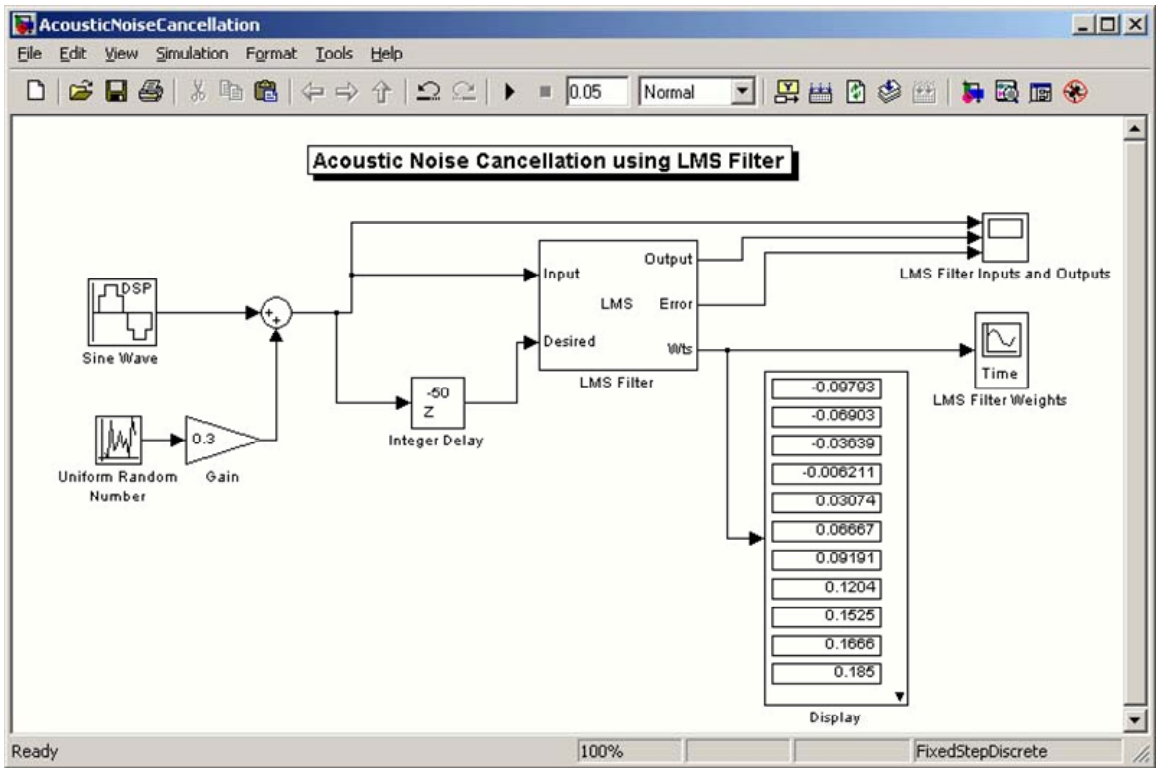
Project Report submitted to the
Department of Electrical Engineering, University of Management and Technology
in partial fulfillment of the requirements for the degree of
Bachelor of Science
in
Electrical Engineering

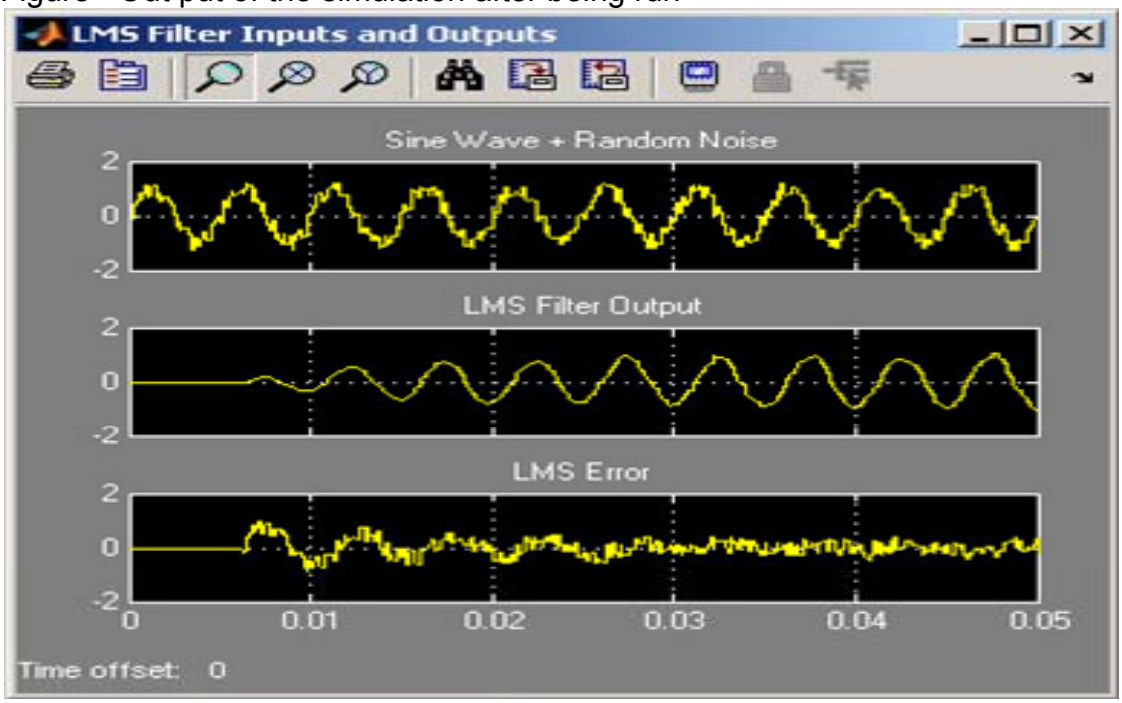
Hassan Raza	071020-013
Muhammad Arif	071020-021
Ali Anees	071020-027

Table of contents

List of Figures

List of figures:





Figure= LMS Filter Weights (Coefficients)

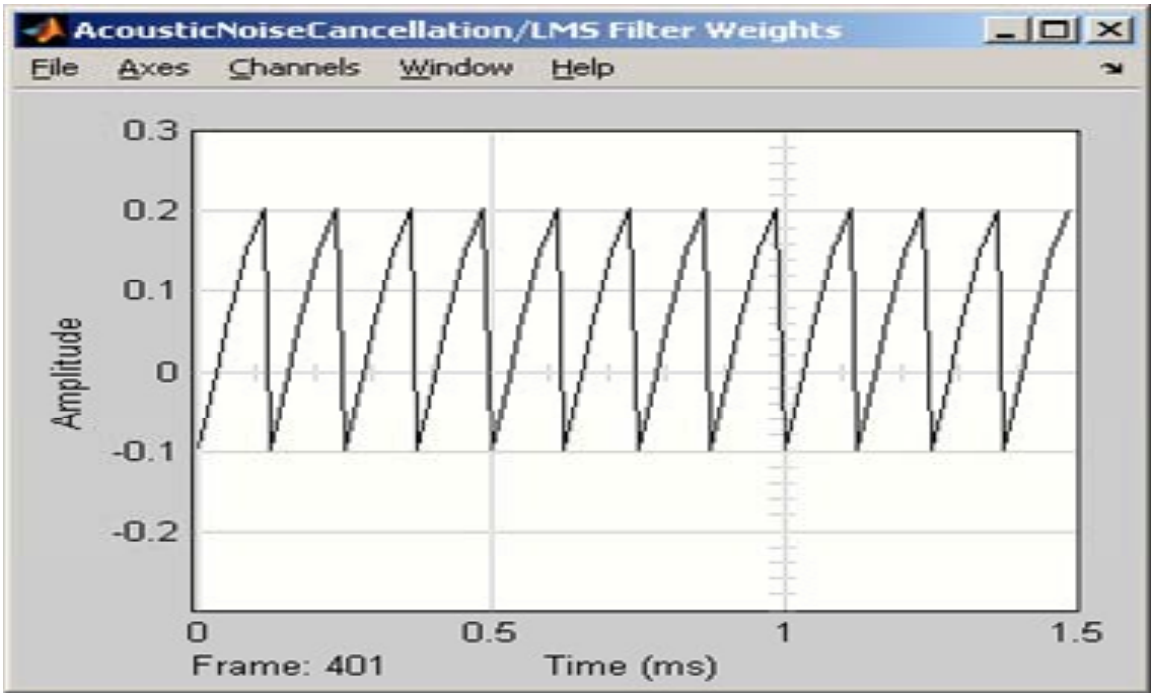
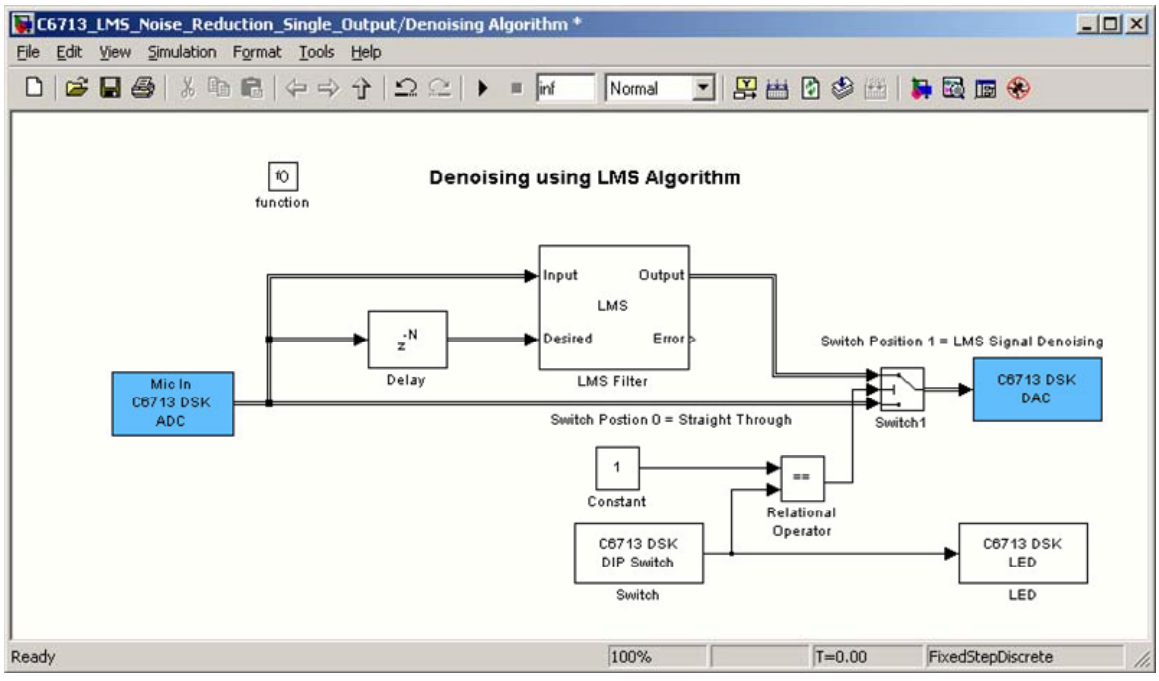
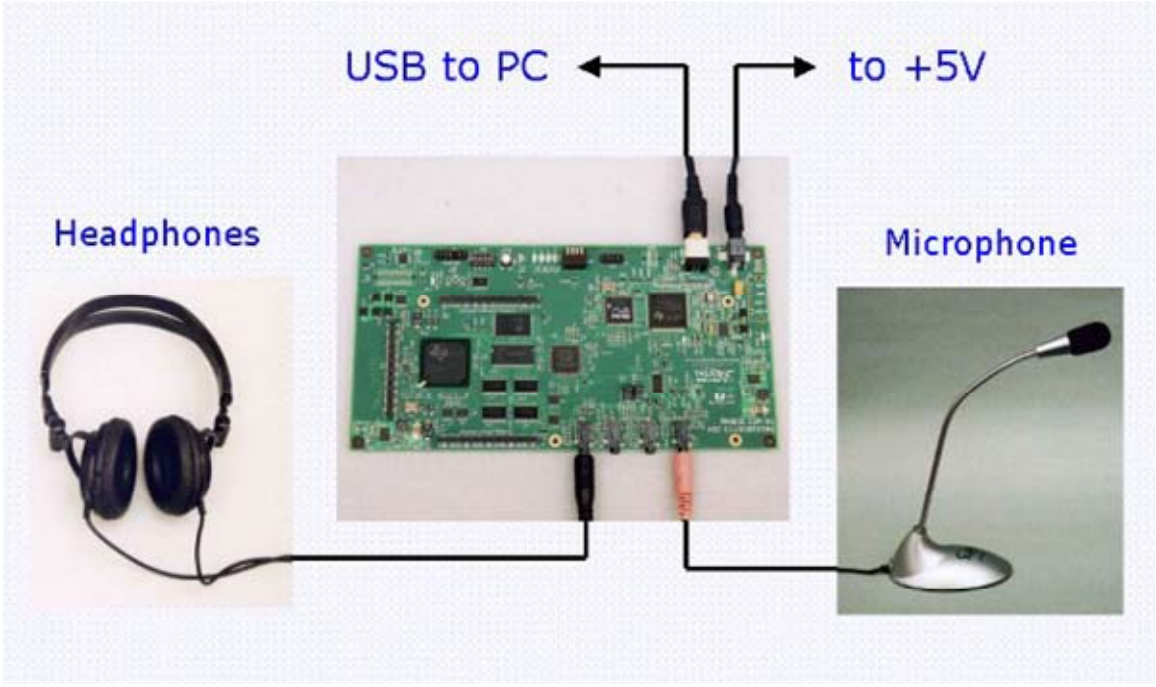


Figure = Complete Simulation Model Of Noise Cancellation In Matlab



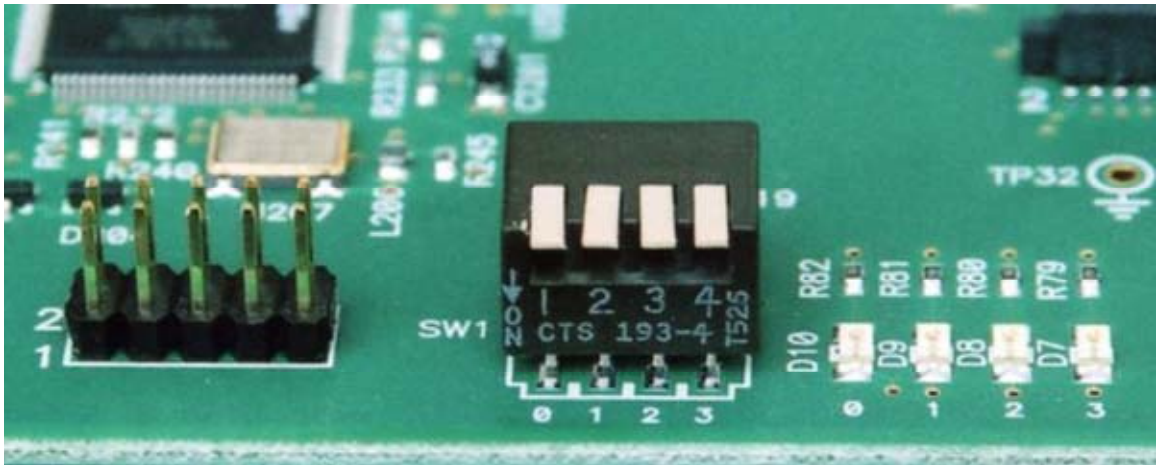
Figure=Texas Instruments DSK6713 Setup

Alternatively, we can use computer loudspeakers.



Figure= Microphone Straight Through to Loudspeakers

To check out the microphone and loudspeakers, set the DIP switches on the DSK6713 as follows:



Figure= Connecting the DSK6713

Start Code Composer Studio for DSK6713 and use Debug

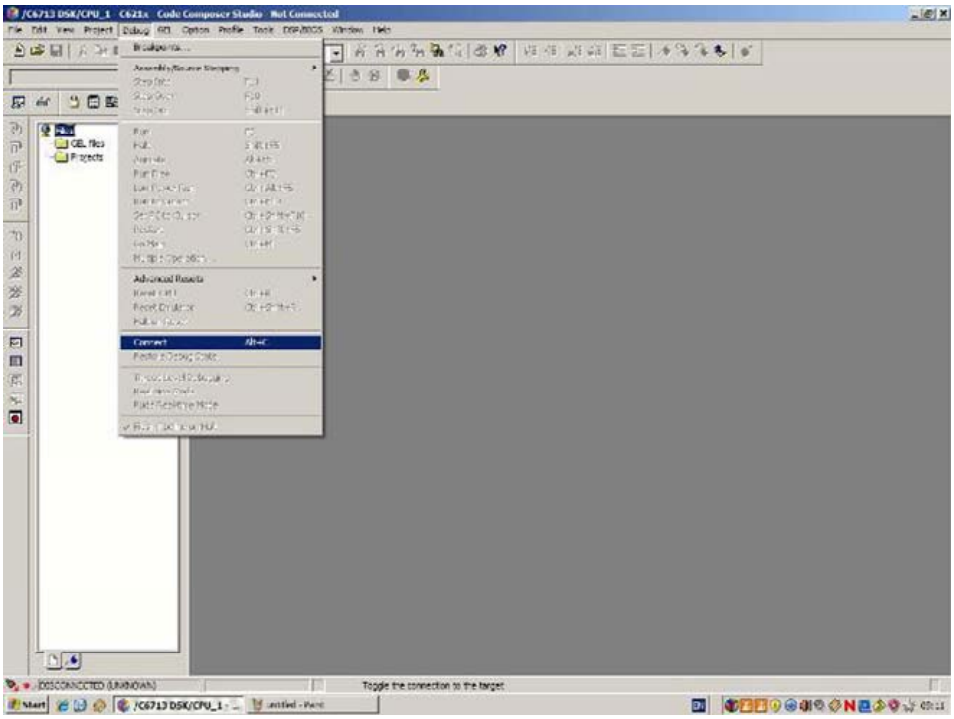
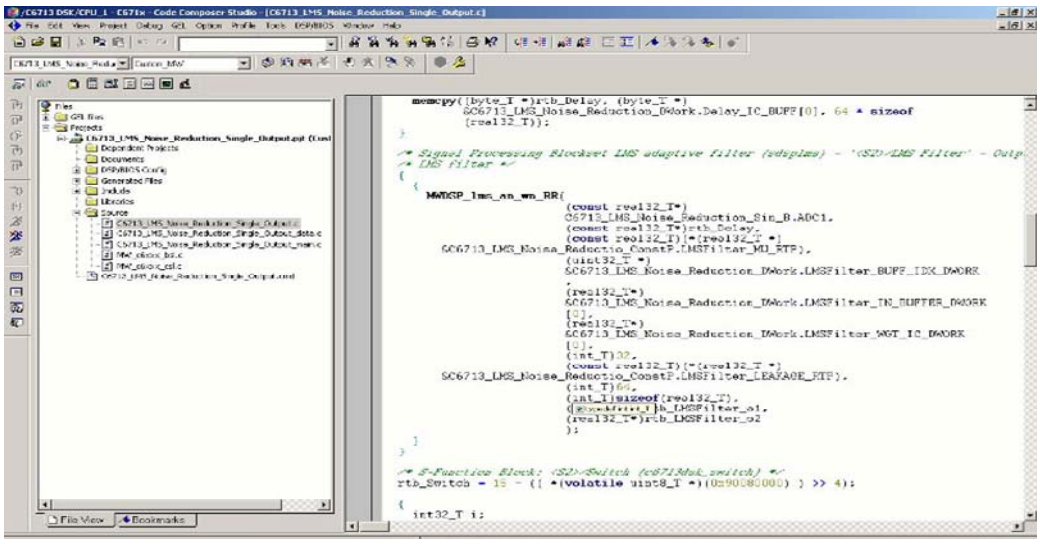


Figure = Complete Model Running In Code Composer



Abstract

This paper shows the implementation of a low cost active noise control (ANC) on the DSP Starter Kit TM6713 . The system consists of an error microphone plus a signal amplifier, two powered speakers (primary and secondary source) and the controller (the DSP Starter Kit). The total performance is limited by the DSK features but it is possible to reach 30 dB of acoustic attenuation in narrowband acoustic noises. The fixed-point DSP controller has been programmed with an assembler and a debugger supplied by *Texas Instruments,* and the final application is built on the *Windows* operating system environment where the user can easily modify the parameters of the active noise controller (filters taps, convergence factors, leaky factors) with a friendly control panel.

Dedication

With the blessings of our parents and hard work of our teachers we are able to come to this stage to make this project .It was a hard tough time which we put it to our project with our team work we are today able to present this project. So we would like to dedicate this project to our beloved parents and respected teachers and to our own team ,without our teachers co-operation we can't be able to do this project . May God bless them and may God give us a good reward of our hardwowk.

Chapter I. Introduction

Introduction:

What is active noise control? Active control is sound field modification, particularly sound field cancellation, by electro-acoustical means. In its simplest form, a control system drives a speaker to produce a sound field that is an exact mirror-image the offending sound (the "disturbance"). The speaker thus "cancels" the disturbance, and the net result is no sound at all. In practice, of course, active control is somewhat more complicated. The name differentiates "active control" from traditional "passive" methods for controlling unwanted sound and vibration. Passive noise control treatments include "insulation", silencers, vibration mounts, damping treatments, absorptive treatments such as ceiling tiles, and conventional mufflers like the ones used on today's automobiles. Passive techniques work best at middle and high frequencies, and are important to nearly all products in today's increasingly noise-sensitive world. But passive treatments can be bulky and heavy when used for low frequencies. The size and mass of passive treatments usually depend on the acoustic wavelength, making them thicker and more massive for lower frequencies. The light weight and small size of active systems can be a critically important benefit. In control systems parlance, the main four parts of an active control system are:

The *plant* is the physical system to be controlled; typical examples are a headphone and the air inside it, or air traveling through an air-conditioning duct.

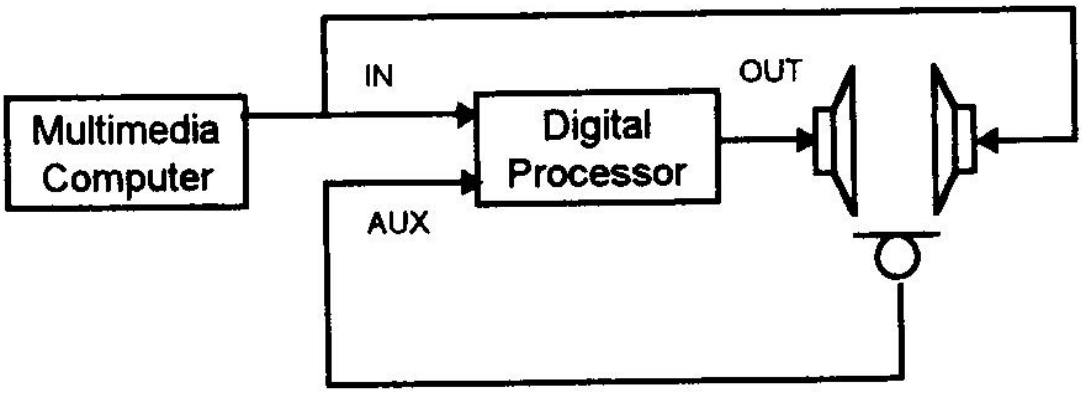
Sensors are the microphones. accelerometers, or other devices that sense the disturbance and monitor how well the control system is performing.

Actuators are the devices that physically do the work of altering the plant response; usually they are electromechanical devices such as speakers or vibration generators.
The *controller* is a signal processor (usually digital) that tells the actuators what to do, the controller bases its commands on sensor signals and, usually, on some knowledge of how the plant responds to the actuators. Analog controllers may also be used, although they are somewhat less flexible and thus more difficult to use.

Active noise control configuration:

The ANC developer kit to implement is working in an acoustic configuration as shown in figure 1.

Figure 1. Block diagram of the basic active noise controller



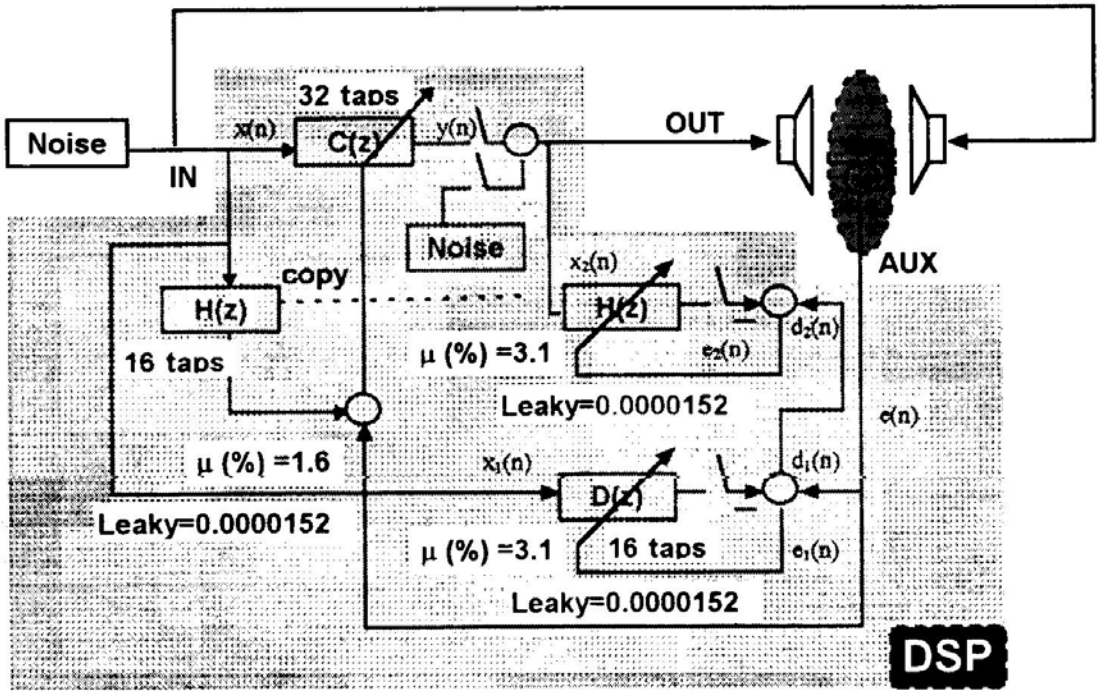
A personal computer, equipped with a low cost soundcard like any *Soundblaster* or compatible card, and with an audio software like *Cool-Edit* ,can generate any kind of noise signal in a loop mode. The *Cool-Edit* software generates many types of periodic signals and three different random signals. The maximum frequency of these signals is 500 Hz. The attenuation of higher frequencies is irrelevant because the silent zone is only one tenth of the acoustic wavelength .For an acoustic wave of 500 Hz the effective silent zone is only 7 cm.

One of the two channels of a stereo file, noise generated by the soundcard, is applied to one powered speaker to produce an undesired acoustic noise to cancel (figure 1). The other channel is applied to one input of the DSP starter kit, the active noise controller. The DSP produces an output signal that is also applied to another powered speaker, trying to cancel the noise. A microphone catches both added noises. This signal makes the DSP to work properly in order to reach the maximum attenuation.

Adaptive algorithms:

The active controller is based on the classical filtered-x LMS adaptive algorithm, working with other adaptive FIR algorithms that keep a robust convergence to the minimum residual signal at the error microphone. The complete block diagram is shown in figure 2.

Figure 2. Adaptive algorithms implemented in the DSP



As shown in the figure, we can choose any of the typical configurations used in a *single input single output* active noise canceller. The algorithm LMS applied to the FIR H(z) estimates the electrical-acoustic error path, the transfer function, and this estimate is used by the filtered-X LMS adaptive algorithm applied to C(z). The estimation is necessary to keep a working stability in the whole process .Such estimates can be configured to be on/off line with the cancelling process made by the adaptive C(z) FIR filter. If off line estimation is chosen, an auxiliary random noise is generated by the DSP before C(z) begins to have the optimum coefficients. Once H(z) has its minimum residual, the *unbiased* H(z) coefficients are copied to the filtered-X LMS algorithm. Then, the active control attenuation begins to work.

On the other hand, if on line estimation is chosen, the LMS algorithm applied to H(z) works together with the filtered-X LMS algorithm applied to C(z). In this case, the *biased* coefficients are updated and copied to the filtered-X LMS algorithm in each iteration. There is a third LMS algorithm applied to another FIR filter, D(z). This algorithm tries to remove any bias in the H(z) estimation . making the total performance more robust to instabilities.

In every adaptive filter the filter taps, the percentage of the maximum convergence factor and a noisy leaky factor are previously chosen by the user. The maximum convergence factor is calculated and updated each iteration. First, the average power of the input signal to any adaptive FIR filter, p(n), is calculated with a simple recursive IIR algorithm, and then factor is also calculated:

$$p(n) = \alpha x^2(n) + (1-\alpha)p(n-1)$$

$$\mu = \frac{1}{taps \cdot p(n)} \cdot \frac{\mu(\%)}{100}$$

Instabilities can occur when the error signal, caught by the microphone, is higher than the reference signal input. In this case, the algorithm multiplies the error signal by a multiplier to make both RMS values similar at the beginning of the adaptation process.

2. ALL-ZERO FILTER USING MSE

In order to find the best coefficients to use in our filter, we need to define what best means. Since the original speech is unknown (we are trying to find it), and the transfer functions are unknown, we need to estimate the transfer function. In order to do this effectively, we need a way of determining if our estimate is good or not. Since we are trying to find the filter that matches the reference signal to the part of the primary signal contributed by the noise source, it would be natural to look at the resulting difference (primary minus filtered reference) and try to minimize its energy.

Mathematically, minimizing this result gives the optimal filter if the speech and the reference are uncorrelated. If however, some of the speech signal
leaks into the reference signal, then this method would try to minimize the speech part of the signal as well. This would result in poorer noise cancellation, and speech attenuation (bad).

3. OPTIMAL FILTER SOLUTION

In order to implement the optimal solution, a filter size must be chosen. A filter order of 60, seems to be about the maximum needed, based on trail and error. Because the Wiener-Hopf solution is deterministic, choosing a larger filter size will not decrease performance (as is the case for LMS).

The optimal solution does a poor job of removing the noise from the signal. This makes sense, because the optimal filter assumes that the signal is stationary, but we know from the problem that this is not the case. But we do know that the signal is piecewise stationary. Thus, if we can determine the time at which the signal changes, we can use the optimal filtering method on each of these sections separately to come up with independent weight values.

To find the time when the filter coefficients change, a spectrogram was used to see if any visible drastic changes occur in the signal. It turns out that the change occurs precisely half-way through the signal at sample 23,000 (out of 46,000 total samples). When we use the Wiener-Hopf solution on each of these sections separately, we are able to hear the original speech, *logic clearly dictates, that the needs of the many, outweigh the needs of the few*, spoken by Leonard Nimoy, without any interference. Even the time when the transfer function changes, is hardly detectable from the speech.

The minimum of the cost function, J_{min}, is simply the expected value of the energy in the error (which in this case is the expected value of the energy in the speech). J_{min} was found to be 0.00018 for the first half, and 0.00013 for the second half. If we do not split the signal in half, we get a much larger J_{min} of 0.0022, which is approximately 12 times larger than the split solution.

In examining the coefficients of the transfer function, it becomes obvious that they are sums of decaying complex exponentials. This signifies that out all-zero filter is probably better modeled as an all-pole filter. Finding in the all-pole equivalent of the filter we get the following two transfer functions (one for each half of the signal in time, respectively):

$G_1(z) = 1 / (1 + 0.5\, z^{-1}\ \ 0.2\, z^{-2} + 0.2\, z^{-6})$

And

$G_2(z) = 1 / (1 + 0.5\, z^{-2})$

The first half of the signal requires a much larger filter order to accurately model the transfer function (~60), since the coefficients do not die out rapidly. The second half of the filter requires a much smaller order (~10) to model the changed transfer function.

Note that if the transfer function between the reference and the noise component of the primary is an all-pole filter, the transfer function between the noise component of the primary and the reference will be an all-zero filter, and will fit our all-zero model better. If we do this however, the error of the filter will no longer be the speech signal desired, but instead it will be the speech filtered through our all-zero model. So if we reverse the roles of the primary and reference signals, we need to further add the inverse of the all-zero filter to the error signal to recover the original speech, as is shown in figure 3. By doing this, we will gain by requiring a much smaller filter order (seven for the first half of the signal and three for the second half).

4. LEAST-MEAN-SQUARES (LMS)

While the above Wiener-Hopf solution perfectly solves the problem, it required an intelligent system modification (breaking the problem into two parts) that will generally not work for an adaptive noise cancellation problem. Also, the optimal method cannot be performed in real-time, because the entire signal must be given in order for the solution to work. The LMS solution avoids both of these problems by adapting the filter weights as the speech is being played. The result however, is not close to the quality of the Wiener-Hopf solution.

In order to use the LMS algorithm, we must first determine a filter order and then a good step-size. This can be done by trail-and-error, or by finding the autocorrelation of the reference signal and the cross-correlation between the reference and primary signals. Since we already did this for the optimal filter, we know the filter order need not be any larger than 60. But where as increasing the filter order did not degrade the performance of the optimal filter, it will degrade the performance of LMS. This is because the larger filter order, the larger the eigenvalue spread of the autocorrelation matrix of the reference signal (we dont change the autocorrrelation function, but we include more terms from that function into the autocorrelation matrix). Increasing the eigenvalue spread, in turn decreases the rate of convergence of LMS to the optimal solution. This occurs because the maximum step-size for convergence is related to the maximum eigenvalue, but the rate of convergence is proportional to the step-size and the minimum eigenvalue, which is shown in the equation:

$1/\tau \approx 2\lambda_{min}$

where is the step-size, λ_{min} is the minimum eigenvalue, and τ is the overall time-constant (which is inversely related to the convergence rate).

As a compromise, we chose $M = 20$ for the order of the filter. If we use the rule of thumb, $= 0.2/\lambda_{maz}$, then the theoretical convergence time constant for the LMS rule is given by

$\tau \approx 0.4\ \lambda_{max} / \lambda_{min} \approx 4000$ samples

This gives the time constant for the convergence of the slowest converging weight. Figure 6 shows the weight tracks of different weights for a single step-size. Figure 7 shows the weight track for w_2, for different values of . From these weight tracks you can clearly see the transition at sample 23000. By trail-and-error, the maximum step-size was found to be 0.3 (for M=20). Using this value resulted in very rough sounding speech, however, because the weights were changing so rapidly. The best value seemed to be about 0.04. Figure 8 shows the learning curve (MSE of the output).
o improve the performance of LMS, we can swap the primary and reference signals as we did in the optimal filter case. This filter should be easier to train using LMS for two reasons. Firstly, the filter order now only needs to be seven for reasons previously mentioned. Secondly, most of the optimal weight values are zero, so the filter will initialized to the optimal value for those. Of course, the speech will be slightly muddled for reasons previously mentioned. Figure 9 shows the weight tracks for this setup.

4. LMS USING BATCH PROCESSING

Instead of using straight LMS,where we modify the weight values for every new datapoint, we can store the modifications that we would make for each data point, and then after a while apply all of the modifications at one time.For this project, we will use a batch size of 2000.

Since we are averaging the weight update over 2000 samples,we need a much larger value for step-size. By trail-and-error we found that a value of = 2 gives us the fastest convergence. This method performs poorly when compared to LMS. The reason for this is that there is not enough time for the LMS algorithm to adapt properly. It only has 11 iterations to converge before the transfer function changes. Even when the desired and reference signals are swapped, the speech is hardly recognizable. As the batch size is decreased, the algorithm performs better.

Chapter II. Results and discussion

Results and discussion:

The active noise canceller has been tested in a small room with different types of noisy signals produced by fans, engines, cars, helicopters, and so on. These signals have been recorded previously in the Windows audio file format *(noise.wav)* and then played in loop mode with the *Cool-Edit* audio software. Other typical signals like random and periodic noise, generated directly by the audio software, have also been tested. The values of the control parameters depends on the speakers separation, the microphone position and the input signal levels in the DSP.

One experiment tested had the next configuration: the two speakers were very close, about 25 cms, and the microphone was closer to the speaker tied to the DSP output than that tied to the soundcard output. The attenuation produced is shown in the next