# COMPARATIVE STUDY OF WIDEBAND SINGLE REFERENCE ACTIVE NOISE CANCELLATION ALGORITHMS ON A FIXED-POINT DSP

*Bharath Siravara, Mohamed Mansour, Randy Cole, and Neeraj Magotra*

Texas Instruments Inc.,
Dallas, TX, 75243
{bsiravara, mfmansour, rcole, nmagotra}@ti.com

## ABSTRACT

In this paper, we present a comparative study of fixed-point implementations of various feedback Active Noise Cancellation (ANC) algorithms for headset applications. Different adaptive algorithms for wideband ANC were implemented on a TMS320C54x fixed-point DSP with single-precision words. The performance of each of them was tested and analyzed using different types of noise signals.

## 1. INTRODUCTION

Over the past decade, environmental noise issues have gained attention due to the tremendous growth of technology that has led to noisy engines, heavy machinery, pumps, high speed wind buffeting and a myriad of noise sources. Exposure to high decibels of sound proves damaging to human beings from both a physical and a psychological aspect. The problem of controlling the noise level in the environment has been a focus of research over the years. The steady increase in the performance of DSPs coupled with the decrease in their power consumption has enabled the use of DSPs in a variety of portable hearing enhancement devices such as hearing aids, headsets, hearing protectors, etc. For the large part, these applications tend to be battery powered and hence the energy consumption of the DSP is often the biggest constraint. Fixed-point DSPs tend to be much more energy efficient than their floating point counterparts. Hence, these ANC applications would be implemented on a fixed-point DSP. The paper presents a detailed study of fixed-point implementations of the various flavors of the feedback ANC algorithms for headset applications. To our knowledge, it's the first work that compares performance using fixed-point DSP.

The paper is organized as follows. In section 2, we describe the experimental setup. In section 3, we describe the implementation details of the feedback FXLMS algorithm and its variants. In Section 4, we discuss the coefficient quantization procedure. In section 5, we present the results of the simulations in terms of computational complexity and algorithm performance.

## 2. EXPERIMENTAL SETUP

ANC systems are broadly classified as *feedforward* systems where a coherent reference noise input is sensed or *feedback* systems [1] where the controller does not have the benefit of a reference signal. In the case of headsets and hearing protectors, it is better not to have a reference sensor on the outer casing of the headphone. The configuration for the feedback ANC system is as shown below.

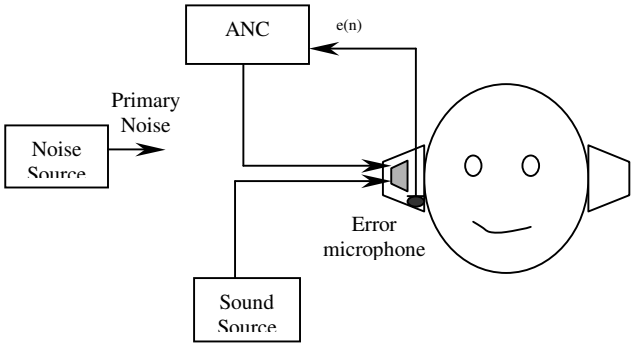



*Figure* 1. *Feedback ANC system for a headset*

The most common algorithm used for feedback ANC is the feedback FXLMS algorithm and its variations [1]. The system model for the algorithm is shown in Figure 2. In this model, W(z) is the adaptive filter, S(z) is the secondary path transfer function, and Š(z) is the estimate of the secondary path transfer function S(z). We assume that there is no error in the estimation of S(z) i.e., Š(z)=S(z).

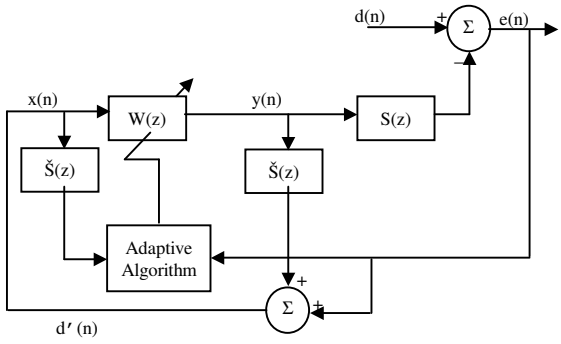



*Figure* 2. *Feedback FXLMS algorithm.*

In this paper, we consider different adaptive algorithms namely the Least Mean Square (LMS), Normalized LMS (NLMS), adaptive IIR, and subband adaptive filtering. These algorithms

were implemented in a mixture of fixed-point C and assembly on the TMS320C54x DSP. All data and filter coefficients were implemented using 16-bit word length. The sampling frequency is 24kHz. A variety of input signals were used to evaluate the performance of each of these algorithms including real-life noise recorded by the authors at car and plane noise environments. These are the most commonly used environments for noise canceling headsets. A commercially available headset, with a microphone mounted inside serving as an error sensor, was used in the simulations. The secondary path transfer function was calculated on the physical system using the LMS algorithm with white noise excitation. The impulse response and the transfer function of the secondary path are shown in Fig. 3. Note that, the impulse response as derived from real hardware components is far from ideal.

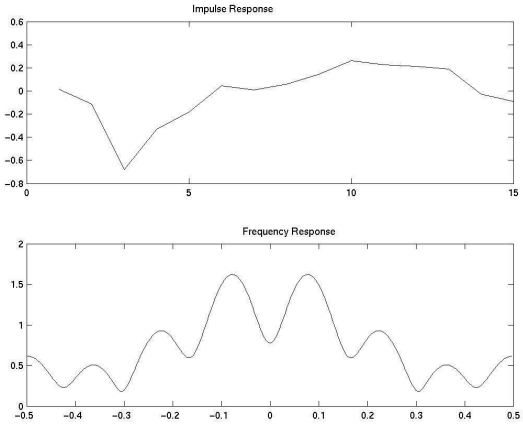



*Figure* 3. *Impulse Response and Transfer Function of the*

## 3. ANC ALGORITHMS

The basic model used for all our simulations is as shown in Fig. 2. The different variations of the FXLMS algorithm were used for the adaptive filtering block. In the basic FXLMS algorithm the adaptive filter $W(z)$ is an FIR filter and the filter coefficients are adapted using the LMS algorithm. The filter adaptation equations are [2, ch. 5]:

$$e(n)=d(n) - \underline{w}^T(n).\underline{x}(n)$$
$$\underline{w}(n+1) = \underline{w}(n)+\mu.e(n).\underline{x}'(n) \qquad (1)$$

Note that the input samples for adaptation $\underline{x}(n)$ is a processed version of the input to the adaptive filter $\underline{x}(n)$. This is necessary to ensure the alignment with the noise samples. In the following subsections we will introduce different variations of the FXLMS algorithm. These variations will be assessed in section 4.

### 3.1 FXNLMS Algorithm

The adaptation equation is quite similar to the basic FXLMS algorithm. However, the adaptation step is variable and inversely proportional to the second norm. of the input i.e.,

$$\mu(n)= \alpha / \|\underline{x}'(n)\|^2 \qquad (2)$$

In our simulation we set $\alpha = 0.2$. Note that, $\|\underline{x}'(n)\|^2$ can be efficiently calculated using only two multiplication and two additions as follows

$$\|\underline{x}'(n+1)\|^2 = \|\underline{x}'(n)\|^2 + |\underline{x}_N'(n+1)|^2 - |\underline{x}_1'(n)|^2$$

where $\underline{x}_i'(n)$ is the $i^{th}$ component of the input vector at time $n$.

The division in fixed-point arithmetic is not trivial. We used a Taylor-based division algorithm that has good convergence characteristics. The idea of the algorithm was to convert the division operation to multiplications and additions using the well-known formula:

$$(1-x)^{-1} = 1 + x + x^2 + x^3 + x^4 + \ldots\ldots..$$

Convergence is guaranteed if $|x|<1$. Here, we needed to evaluate $x/y$. Without loss of generality, we will assume that $x < y$, and $x$ and $y$ are unsigned with the same Q-value. If $y = a - b$ (with b < a), then

$$\frac{x}{y} = \frac{x}{a} \cdot \frac{1}{1-\frac{b}{a}}$$

$$= \frac{x}{a}.(1 + \frac{b}{a} + (\frac{b}{a})^2 + (\frac{b}{a})^3 + \ldots\ldots\ldots + (\frac{b}{a})^n + R_{n+1})$$

Now, the crucial step is to choose $a$, so as to simplify the above calculation. Our choice of $a$ was the smallest power of 2 greater than y. For example if y= 1101b, then $a$ = 10000b. In this case dividing by $a$ is equivalent to right shifting. If $a = 2^r$, then $b/a = b<< (15-r)$, where "<<" denotes the right shifting. Note that, for this particular choice of $a$, we have $b/a \le 1/2$, and hence convergence is guaranteed. Typically 3 or 4 terms were enough to get good approximation. If $n$ terms are used in the Taylor expansion, then the division will need $(n-1)$ multiplications, and $(n-1)$ additions.

### 3.2 Adaptive IIR Filtering

Usually a large filter order is needed for the adaptive filter $W(z)$. This can be compensated by using adaptive IIR filters. However, a stability test is needed after each adaptation iteration.

The choices for IIR adaptive filtering are either the output error method or the equation error method [2, chapter 15] with stabilization procedure. The complexity and memory requirement of both algorithms restrict their use for online ANC. However, we developed a simpler structure whose performance is very comparable to the above algorithms.

The basic idea of the algorithm is to use a cascade of second order IIR blocks, and test the stability of each block per se. The stability test for a second order IIR filter is straightforward. If the IIR filter has the form $1/(1+a_1z^{-1}+ a_2z^{-2})$, then the stability test is [4, chapter 3]:

$$|a_2| < 1, \text{ and } \quad |a_1| < 1+ a_2 \qquad (3)$$

The structure of the ANC algorithm is shown in fig. 4. Each $B_i(z)$ is a second order IIR filter. Define $y^{(i)}(n)$ as the output to $B_i(z)$. Denote $a_1^{(i)}$ and $a_2^{(i)}$ as the coefficients $B_i(z)$, then we have,

$$y^{(i)}(n) = y^{(i-1)}(n) - a_1^{(i)}.y^{(i)}(n-1) - a_2^{(i)}.y^{(i)}(n-2) \qquad (4)$$
with $y^{(0)}(n) = \Sigma_k w(k)x(n-k)$ , and $y(n) = y^{(L)}(n)$

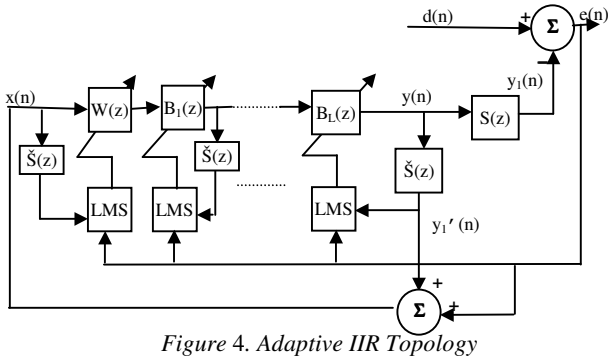



*Figure* 4. *Adaptive IIR Topology*

The adaptation of the FIR filter coefficients is exactly as before. However, the adaptation of the IIR filter coefficients is a little more complicated. Each filter output $y^{(i)}(n)$ is convolved with Š(z) to produce $y^{(i)\prime}(n)$, then the filter coefficients are adapted according to

$$a_k^{(i)}(n+1) = a_k^{(i)}(n) + \mu . e(n) . y^{(i)\prime}(n-k) \qquad (5)$$

where $i$ = 1,2,…,L, and $k$ = 1,2. The simple stability test (3) is applied after each adaptation. If the test fails, then no adaptation takes place for any filter (including the FIR filter). In this case, $\mu$ is lowered (up to a certain value) by multiplying it by a factor less than 1.

### 3.3 Subband Adaptive Filtering

We used a single-stage of dyadic orthogonal wavelet decomposition for subband adaptive filtering implementation. Note that, the subband ANC algorithms proposed in the literature (e.g., [6]) are not suitable for real-time implementation because of the excessive computational requirements. Instead, we used the configuration shown in Fig. 5. To avoid the excessive delay with subband decomposition, we used the simple Haar wavelet [3, chapter 5] in our simulation.

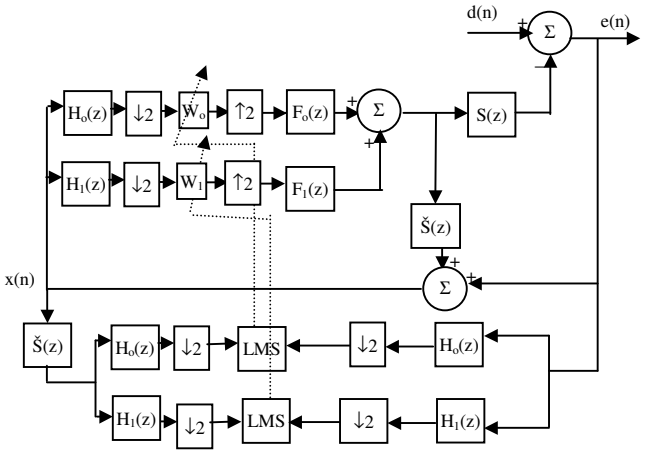



*Figure 5. Topology of Subband ANC*

## 4. QUANTIZATION ANALYSIS

The main step in converting any algorithm from its floating point to its fixed-point version is to identify fixed-point parameters to represent each of the critical variables and coefficients. Thus each of these variables can be represented in the finite word length available so as to minimize the error due to quantization. The quantization analysis of the algorithms described in the previous section was done using a tool developed at TI called Fast Quantization Tool (FaQT) [7]. The FaQT tool is a PC based tool that tracks the values of a large number of variables and calculates the appropriate quantization formats for each of these variables. Each of the algorithms described in the previous section were implemented in floating point using C and the appropriate calls to the FaQT tool were inserted into the floating point simulation. The simulations using the FaQT tool were then run for a number of different input signals to determine the optimum quantization formats for each of the coefficients and variables. A fixed quantization format was used in the implementation wherein the representation of each of the variables was fixed beforehand and the fixed-point implementation did not have to track and adjust the format of any of the variables. Furthermore, the formats were chosen so as to completely avoid saturation. The fixed-point implementation was thus tweaked to ensure that the quantization effects due to the finite word length were minimal with a significant savings in the MIPS required to implement the algorithm on the DSP. The FaQT tool greatly helped in shortening the development time of the fixed-point implementation of the algorithms.

## 5. SIMULATION RESULTS

### 5.1 Complexity

In any fixed-point real time implementation resources are usually limited. Therefore simplicity is one of the main factors in deciding which algorithm to use. In what follows we describe the computation and memory requirements for each of the algorithms. The following notations are used: $N$ is the FIR adaptive filter order, $K$ is the order of Š(z), $M$ is the IIR order. Note that, each filtering operation of order $q$ will need in general $q$ multiplications and $q$ additions. In our implementation, we used circular indexing mode [5, chapter 5], hence the overhead in memory movements of all algorithms is minor and can be neglected. In table 1, we give the detailed MIPS requirement for each algorithm (normalized by the sampling frequency). Typically, for our application we have $K \le 30$, $N \le 50$, and $M \le 8$.

| *Algorithm* | *Multiplications* | *Additions* | *Memory* |
|---|---|---|---|
| FXLMS | $2N + 2K + 1$ | $2N + 2K + 1$ | $2N + K$ |
| FXNLMS | $2N + 2K + 6$ | $2N + 2K + 6$ | $2N + K$ |
| IIR | $2N + 2M + (1+M/2)K$ | $2N + 2M + (1+M/2)K$ | $2N + M + KM/2$ |
| Subband | $4N + 2K + 28$ | $4N + 2K + 30$ | $4N + 2K + 18$ |

Table 1. Complexity of ANC Algorithms

The algorithms were profiled using the profile tool available with Code Composer Studio (CCS) development software to determine the MIPS requirements of each of the algorithms. The appropriate profile points were inserted and the CCS cycle count profiler was invoked to provide an accurate picture of the MIPS requirement of each of the algorithms. This also enabled us to ensure that the implementations met their real time deadlines. The complexity of each of the algorithms with different filter lengths is described in figure 6. In this figure, we plots the number of CPU cycles for each new data sample, i.e., the number of cycles for the interrupt routine that handles the new data sample. For subband filtering the number of cycles is not the same for each sample, hence the average is calculated.

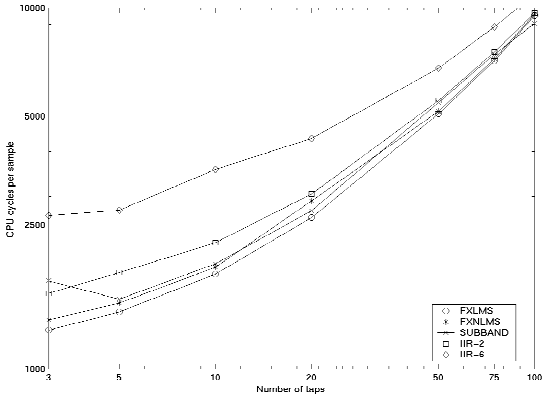



*Figure 6. Algorithms Computational Requirements*

## 5.2 Experimental Results

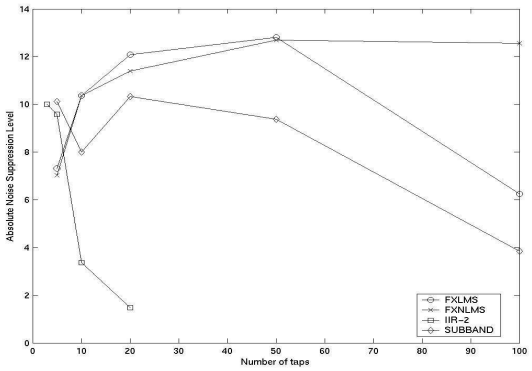
The algorithms described in section 3 were tested using plane and car noises recorded by the authors. The algorithms were tested versus different filter orders. For IIR adaptive filtering, the IIR order is fixed at 2, which is found to be sufficient for the given microphone transfer function. The results are illustrated in Figures 7. From the figure, we see that, the FXLMS algorithm outperforms the other algorithms especially for higher filter orders. The IIR implementation achieves around 10 dB improvement (for car noise) with filter order 3 for the MA part and 2 for the AR part.

It is important to mention that, the performance is very dependent on the noise type as shown in the large variation between the two noise types. For example the improvement using subband filtering is around 10 dB for car noise and no more than 4 dB for plane noise (under the same parameters setting).

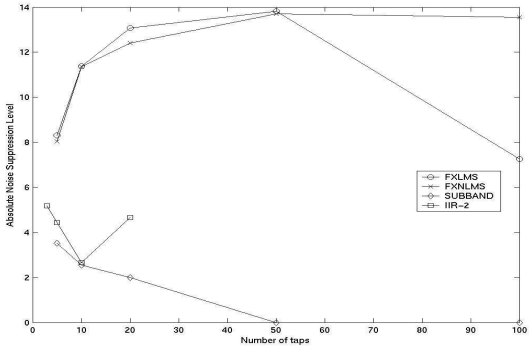
Also note that, high orders of the IIR adaptive implementation lead to instability and hence the filter does not adapt efficiently with the new data. This will be considered in the future work.

## 6. CONCLUSION

In this study, we reviewed the common implementations of the FXLMS algorithm for active noise cancellation on fixed-point DSP. In this study, we concentrated on algorithms of *practical* importance. For example, we didn't consider the FXRLS algorithm because of its excessive complexity (although its floating-point counterpart outperforms the algorithms that we have discussed in this paper). From the results, it is seen that, the conventional FXLMS algorithm and its NLMS implementation outperforms all the variations in the performance and the complexity. Compared with our floating-point implementations, the degradation after fixed-point implementation is around 2 dB.




a. Car Noise




b. Plane Noise

*Figure 7. Algorithms Performance*

## References


1. Kuo S., and Morgan D., "Active Noise Control: A Tutorial Review", Proceeding of the IEEE, pp. 943-973, June 1999.
2. Haykin S., "Adaptive Filter Theory", 4th edition, Prentice Hall, 2002.
3. Mallat S., " A Wavelet Tour of Signal Processing", 2nd edition, Academic Press, 1998.
4. Proakis J., and Manolakis D., "Digital Signal Processing", 3rd edition, Prentice Hall, 1996.
5. TMS320C54x DSP Reference Set, Volume 1: CPU and Peripherals.
6. Morgan D., and Thi J., "Delayless Subband Adaptive Filter Architecture", IEEE Trans. On Signal Processing, Vol. 43, No. 8, pp. 1819-1830, August 1995.
7. Kamath S., Magotra N., Shrivastava A., "Quantization Analysis Tool for Fixed Point Implementation of Real Time Algorithms on the TMS320C5000", ICASSP, 2002.