Design of an Active Noise Control System Using Combinations of DSP and FPGAs

Reza Hashemian, Senior Member IEEE
Associate Professor, Northern Illinois University

Field Programmable Gate Arrays (FPGAs) offer a quick and cost effective implementation for medium to large size digital designs that traditionally have been carried out mainly by ASIC and DSP implementation. However, there are still a number of processing-intensive applications that no single FPGA chip, developed today, can handle the entire design.

In this paper, an experimental design of an active noise control system is introduced that combines both DSP and FPGA in a task oriented structure. In this design, data received by the local microphone and the output signals to the secondary speakers in the noise field are handled by the FPGA. This include I/O buffers, data paths, memory access, FIFO, bit-wise manipulations (data shift and logical operations), and the control block (FSM). Data crunching such as high speed manipulations and additions for updating the coefficients in the Signal Processing Block is handled by the DSP. There are two major design criteria considered here: 1) split the task between the FPGA and the DSP in order to reduce the number of DSP instructions as much as possible, and 2) make the devices to work simultaneously and with minimum dependency.

Introduction

As an alternative to passive noise cancellation techniques and often in combination with them, active noise cancellation (ANC) offers an effective solution in certain applications. Although still in the development stage, ANC is receiving considerable attention for applications involving industrial apparatus, dynamic systems, and domestic appliances. In contrast to passive techniques, ANC systems are small, portable, adjustable to different environments, and less costly. An ANC system can be effective across the entire noise spectrum, but it is particularly appropriate at low frequencies of up to 300 Hz, where passive systems are less effective.

The success of an ANC system depends mainly on fulfilling two criteria: first, the anti-noise waveform must closely match the shape and frequency of the noise waveform; and second, the anti-noise wave must be precisely 180 degrees out of phase with respect to the original noise waveform, when reached to the target area.

Failure to fulfill one or both of these criteria may cause the ANC system to generate a second acoustic noise rather than cancel the original one. These criteria apparently impose some restrictions on the canceling system and somewhat limit its application. First, for a highly effective canceling system the noise source must be nearly stationary in relation to the speaker emitting the anti-noise waveform. Second, the noise source should be located in close proximity to the ANC system and, for the best results, the target noise must be dominantly propagating in one direction. This suggests more of an effective "zone silencing" rather than an open area cancellation.

Acoustic delay is another important issue that must be dealt with in a noise canceling system. Physically there are always distances between the source, the anti-noise generator (speaker) and the residue noise detector (microphone). These physical distances provide noise propagation delays which in turn cause different phase shifts, depending on the relative location of objects. In a general case of a non-periodic noise, prediction techniques and adaptive systems are used to deal with the problem. In this case, an ANC system is dependent on its ability to predict noise from its memory of the past, or adapt the response to the incoming signal as closely as possible. In a periodic noise system, however, prediction is simply done by storing one or more cycles of noise. This makes the periodic noise cancelers much simpler and more effective. And, as a matter of fact, periodic noise is one of the most common and dominated source of noise in industrial and even domestic environment today.

Traditionally, two basic methods are used in ANC systems. In the first method, known as adaptive cancellation, noise is detected by one or more microphones. The system then adapts itself to generate anti-noise waveform which minimizes the residue (mic.) noise. Adaptive cancellation can be used for both periodic and non-periodic noise. However, when used with non-periodic noise the adaptive method is limited because it usually involves a prediction or an estimation process. Feed forwarding is often used to predict the noise before it reaches the target. As for the periodic noise, the adaptation and estimation is basically reduced to utilizing the past noise cycle for the generation of the present anti-noise waveform.

The second type of active noise cancellation system is based on the synthesis method. This involves sampling and storing one or more noise cycles and, based on this information

Paper published in the 1995 PLD Conference Proceedings

received an actual anti-noise waveform is generated to suppress the noise. This method assumes that the current noise cycle will be close to, if not the same as, the past noise cycle. In other words, this method is more appropriate to a periodic noise environment, as mentioned earlier. Anti-noise waveform, in this case, is generated by associating a digital pulse train with the noise cycle and using this pulse train to synchronize the anti-noise waveform with the emitted noise. Typically, this pulse train can be derived from a non-acoustic source, such as an engine's tachometer or odometer.

Active Periodic Noise Cancellation

Periodic noise such as that emitted by industrial equipment and even domestic appliances is quite common in our highly industrialized society. A number of techniques are used in active periodic noise cancellation (APNC) systems. Some APNC systems address the sound cancellation needs of multi-dimensional environments such as a working factory. Others are more limited in scope and function as zone silencing systems. For example, a zone silencing system might be used to cancel the noise around the head of a driver in a vehicle.

Here, in this article, we use the synthesis method to generate the anti-noise waveform, in an APNC system. Contrary to the traditional approach, the signal processing is totally done in the time domain. This not only adds to more real and physical understanding of the processes it also saves time avoiding unnecessary switching domains between time and frequency. The second major change from the traditional approach is the separation of signal processing and signal flows, used in this method. We use both a DSP and a FPGA to share the processes needed in an APNC system, as will be discussed shortly.

In a shared processing environment where different processing types are being used simultaneously it is important to appropriately distinguish between different processes and data flows in order to place them in the right type of environment with the right timing. In our design case, the process sharing between a DSP and a FPGA is aimed at: i) reduction of both the memory requirement and the programming instruction for the DSP; and ii) leaving all the data flows and interfacing to be handled by the FPGA. One may be facing with several problems in this kind of task oriented environment. First, how the data is handled between the two units (the DSP and the FPGA)? Apparently the clocking for the two units can not be the same, because the response times are different. Then, an asynchronous communication link must be set up between the two units. The second problem is to separate the tasks in such a manner that optimization of both response time and hardware are fulfilled. In other words, one must keep both units operational in most of the time, and occupied with the most suitable operations for that unit.

Figure 1 conceptually shows how the system is constructed. A FPGA (TI's TPC12 Series) and a TMS320C2x DSP are the two major units used in our Noise Control System. As shown, the residue (error) noise is received by the FPGA after being digitized. The data is pre-processed (logical shift, inversion, and addition) by the FPGA and stored into a Data Buffer. The DSP then picks up the data from the Buffer and goes into further processing (multiplication, summation, and signal adaptation) and delivers the data back to the FPGA. Finally, the FPGA performs some post-processing (data shifting and sequencing) on the data and generates the anti-noise waveform in digital samples. As illustrated, the process requires "interrupt request" by the FPGA to send the data to the DSP, and "data ready" signal from the DSP to place it into the Buffer. This communication is naturally asynchronous because the clocking for the two units (CLK1 and CLK2) are different.

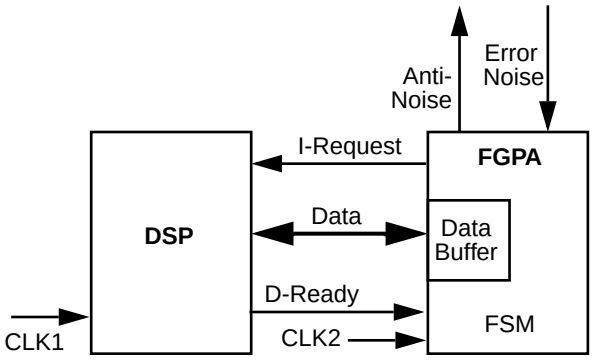

***Figure 1** • Acoustic Noise Cancellation with DSP and FPGA*

Figure 2 shows the overall structure of the APNC system. It consists of three basic parts: i) the noise channel, where the source produces the noise, the anti-noise speaker generates the opposing noise signal, and the microphone which picks up the residue noise. ii) The second part is the data flow section. This part consists of ADC and DAC (signal filtering is also included), clocking (pulse train), and signal pre-processing (coefficient μ and subtraction) operations. iii) The third part is the Anti-noise Generating Block producing the desired waveform for the noise cancellation. First, we discuss the noise channel. Consider x(t) and y(t) to be the noise and the anti-noise waveforms at the target location (microphone, in our case), respectively. The residue noise, e(t), picked up by the microphone is expressed by

$$e(t) = x(t) + y(t) \quad (1)$$

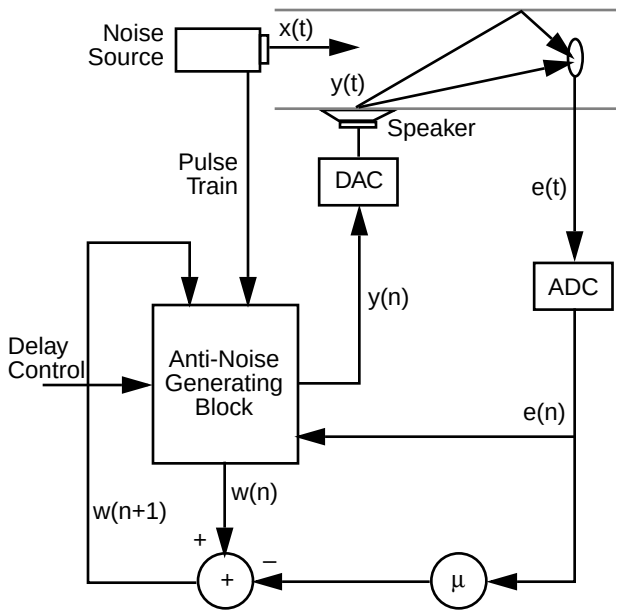

Figure 2 • *APNC System with Noise Channel*

This indicates that the residue noise is the algebraic sum of the original noise, x(t), and the anti-noise waveform, y(t), in the vicinity of the microphone. For the sake of simplicity, we assume that both the noise and the anti-noise waves propagate within a noise channel with limited number of reflections and with negligible distortion along the channel. After digitizing the signal and assuming that data conversions are lossless (except for the processing delays) we may write Eq. (1) in its digital form as

$$e_j(n) = x_j(n) + y_j(n) \qquad (2)$$

Where, n is the noise cycle count and j stands for the position of the sample within the cycle. No signal delay is assumed in Eq. (2). However, with acoustic delays and multiple reflections of the anti-noise waveform within the channel we can substitute Eq. (2) by Eq. (3).

$$e(n) = x(n) + \sum_i a_i \bullet y(n - v_i) \qquad (3)$$

Where, a_i and v_i are the attenuation factor and the acoustic delay of the ith reflection of the anti-noise within the channel, respectively. Note that the sampling indicator, j, is dropped for simplicity.

The residue noise leaving the channel is multiplied by a converging coefficient $0.0 < \mu < 1.0$ and $\mu \bullet e(n)$ is used to update the anti-noise samples stored in the FIFO registers, as shown in Eq. (4).

$$w(n + M) = w(n) - \mu \bullet e(n) \qquad (4)$$

Where, M is an integer 1 or more. The multiplication $\mu \bullet e(n)$ is simply performed by an appropriate right shift applied to e(n). For values of μ=l, 0.5, 0.25, or 0.125 we apply 0, 1, 2, or 3 right shifts, respectively. Shifted data ($\mu \bullet e(n)$) is then subtracted from a previous noise sample stored in the FIFO, inside the FPGA, and the result is directed back into the FIFO which contains M•N registers in sequence; where, N indicates the number of samples per noise cycle.

The next step is to synthesize the anti-noise waveform from the samples stored in the FIFO. This is done by the Signal Processing Block (SPB), shown in Figure 3. The operation works as follows: A set of noise samples stored in the FIFO are accessed by the SPB through a multi-bus line v_0, v_1, …. Each of these samples, taken from different locations (registers) within the FIFO, represents a sample of noise with a particular delay (v_i) attached to it. It is the SPB to use one or more of the samples, scale and add them together to produce a sample of the anti-noise waveform to best suppress the noise. The performance is simply measured by residue noise, e(n), which is also used to further adapt and modify the anti-noise waveform, as given in Eq. (4). The anti-noise is then generated using Eq. (5).

$$y(n) = \sum_i b_i \bullet w(n + v_i) \qquad (5)$$

Where, b_i is a multiplier constant assigned by the SPB. While the FIFO resides in the FPGA the SPB is handled by the DSP chip, as shown in Figure 1. To better understand the process in the SPB we consider the following cases.

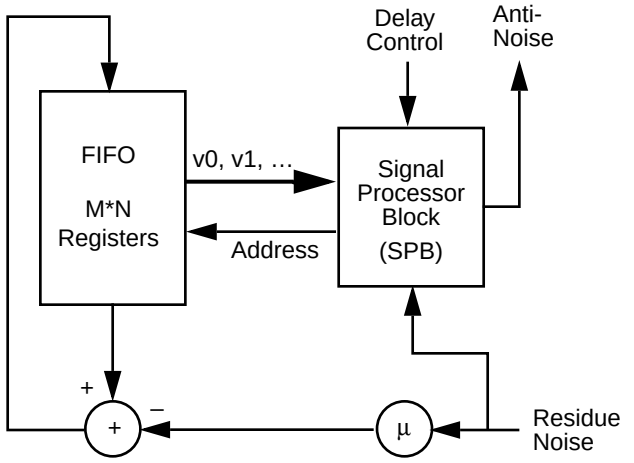

Figure 3 • *Anti-noise Generator Block*

Case 1

We ignore any wall reflections in this case and assume a direct acoustic path from the speaker to the microphone, as shown in Figure 4. We assume an acoustic delay of v_0 (including data processing delays, as well) in this case. Now,

for $a_0 = b_0 = 1.0$, and M=1 the noise equations (3), (4), and (5) are reduced to:

$$e(n) = x(n) + y(n - v_0)$$
$$w(n+l) = w(n) + \mu \bullet e(n) \quad (6)$$
$$y(n) = w(n+v_0)$$

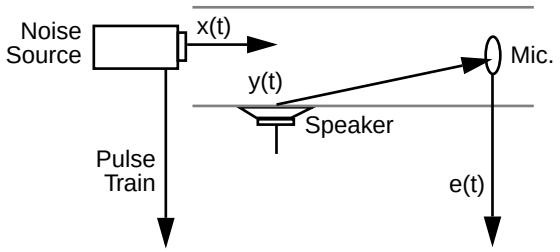

Figure 4 • Channel Signal Path for Case 1

From Eq. (6) it is easy to calculate the next cycle residue noise, e(n+1), which is given by expression

$$e(n + 1) = (1 - \mu) \bullet e(n) \quad (7)$$

It is evident from Eq. (7) that for any value of $0.0 < \mu < 1.0$ the residue noise will progressively vanish as long as the periodicity of the noise is not grossly violated.

Case 2

In this case we include all single reflections from the walls of the noise channel, as shown in Figure 5. We assume a constant attenuation factor a_1 for the reflected anti-noise signal. The modified noise equations for this case become as

$$e(n) = x(n) + y(n - v_0) + a_1 \bullet y(n - v_1)$$
$$w(n + l) = w(n) - \mu \bullet e(n) \quad (8)$$
$$y(n) = w(n + v_0) + b_1 \bullet w(n+v_l)$$

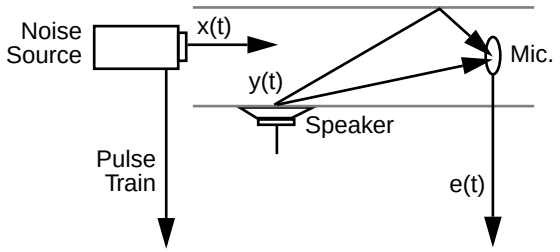

Figure 5 • Channel Reflection for Case 2

Where, v_1 is the acoustic delay for the reflected waveform, and v_0 is still is the acoustic delay for the direct path. Assuming $v = v_1 - v_0$ as the "difference delay" between the reflected path and the direct path, we can calculate the next cycle residue noise as

$$e(n + 1) = e(n) - \mu(1 + a_1 b_1)e(n)$$
$$-\mu[b_1 e(n + v) + a_1 e(n - v)] \quad (9)$$

It is evident from Eq. (9) that, with a right selection of the convergence factor μ, and the fact that a_1 and b_1 are much smaller than 1.0, the residue noise e(n) theoretically converges to zero. However, channel distortion, higher reflections, and other factors may prevent e(n) to vanish completely.

Our analysis may further continue for multiple wall reflections, although, in practical sense, this may only have minor effect on the overall performance of the system.

Shared Processing with DSP and FPGA Combined

In our design we limit our case to Case 2, i.e., direct acoustic path between the anti-noise speaker and the target area (microphone) with single hit wall reflections included, as shown in Figure 5.

There are three parameters b_1, v_0, and v_1 to take care of by the SPB which basically resides in the DSP. Here is how it works: the SPB searches for the best location in the FIFO to pick up the sampled data. The location is specified by a 3-bit control signal (or address) and a set of 8 to 1 MUXs deliver the data to the SPB. The data is then multiplied by a coefficient b_1 and is prepared for summation and generation of the anti-noise sample waveform.

The search process is split into two steps. In the first step we neglect any reflection and assume $a_1 = b_1 = 0.0$, and then search for v_0. In this search the SPB tries for the best location inside the FIFO, as discussed earlier, and receives $w(n+v_0)$ from the FIFO. With this information the SPB can generate the anti-noise waveform for Case 1, direct path, as given in Eq. (6). In the second step, with v_0 being fixed (unless the "Delay Control" line is high indicating that the target has relatively moved), the SPB looks for the best values for the parameters b_1 and v_1. This is done by measuring the residue noise after the first step and trying to select and modify the best values for b_1 and v_1 such that the residue noise is minimized.

Hardware Implementation

We have used an ACT 2 (TI's TPC1240 84-Pin PLCC) FPGA for the hardware implementation. This type of FPGA has the advantage of providing a large degree of functionality in a relatively small gate area, and being multiplexer based makes it suitable for our proposed architecture. Some of the design criteria of the chip are as follows:

- Data is represented by 8-bit words.

- The number of samples per cycle is N = 12, and M = 1 and expandable to 3.
- An external pulse train is used to generate the internal clock.
- In addition to the input and output ports eight different locations along the FIFO registers are also accessed for signal delay purposes. The selection of the location is done through multiplexing.
- For the noise fundamental frequency about 300 Hz the system is capable of adjusting delays from zero to 10 mSecs.
- The system is built with a test mode feature. In its test mode the actual noise is picked up by the microphone. The anti-noise is then added to the noise internally, very much similar to that happens in the noise channel, and the residue noise in subsequently measured for the system performance. While in this mode acoustic delays could also be added to the operation.
- A control block (FSM) is included in the design to provide clocking and all other control signals necessary for the operation. This block also controls the communication between the FPGA and the DSP.

Figure 6 shows a block diagram of the FPGA design in its simplest form. The control signals and the communication links to the DSP are not present. As mentioned earlier, the design is using an ACT 2 (TI's TPC1240 84-Pin PLCC) chip with 4,000 usable gate modules.

Conclusions

In recent years, governmental bodies, industrial concerns and even increasing numbers of individual consumers have realized that noise is a pollutant of our industrialized society and that noise pollution can be just as harmful, if not more so, to our psychological and physiological well being as other pollutants that are commonly classified as toxic. Noise is just as toxic as chemicals and other forms of pollution.

Both active and passive noise control methods will be used in the years ahead to combat noise pollution. Several automobile manufacturers have already implemented active noise control systems in their products and manufacturers of domestic appliances are studying the most cost effective way of actively canceling the noise produced by motors in dish and clothes washers, and other household appliances. Many of the commercial airlines are asking the aerospace industry to incorporate active noise cancellation systems to decrease the noise created by jet engines, In the workplace, enlightened manufacturing concerns are considering active and passive noise control techniques to reduce the noise levels in factories and in the operating enclosures of large equipment such as bulldozers and heavy cranes.

Field Programmable Gate Arrays (FPGAs) in combine with DSPs will play an important role in the further development and implementation of active noise control systems. The ease with which FPGAs can be used to prototype active noise cancellation systems will facilitate more effective experimentation which will bring actual active noise cancellation products to market faster.

References:

1. E. Ziegler Jr. "Selective Active Cancellation System for Repetitive Phenomena." United States Patent No. 4,878,188, Oct. 31, 1989.
2. N.R. Shanbhag, and K.K. Paraki. "A Pipelined LSM Adaptive Filter Architecture." proc. Asilomar Conf. on Sig., Syst. and Comput., Pacific Grove, CA, Nov. 1991.
3. S.D. Brown, R.J. Fransis, J. Rose, and Z.G. Vranesic, "Field Programmable Gate Arrays", KLUWER Aca Pub, 1992.
4. R. Hashemian, K. Golla, S.M. Kuo, and A. Joshi, "Design and Construction of an Active Periodic Noise Canceling System Using FPGAs," 36th Midwest Symposium on Circuits and Systems, Detroit, MI, 1993.
5. S.J. Elliott and P.A. Nelson, "Active Noise Control," IEEE Signal Processing Magazine, October 1993.