LABVIEW FPGA IMPLEMENTATION OF DIGITAL REVERBERATORS

Ioan CATUNA Erwin SZOPOS Marina D. TOPA
Technical University of Cluj-Napoca, Bases of Electronics Department
Gh. Baritiu Str., 26, 400027 Cluj-Napoca, ROMANIA, Tel: 0264401243
ionutcp@student.utcluj.ro, erwin.szopos@bel.utcluj.ro, marina.topa@bel.utcluj.ro

Abstract: The goal of this paper is to present a way in which digital reverberation algorithms can be implemented on FPGA. Schroeder's late reverberator was chosen to be analyzed, implemented as an online algorithm in LabVIEW FPGA and then deployed on a National Instruments CompactRIO system. The hardware implementation is tested with three different voice sequences, and its parameters are adjusted so that the reverberation effect improves the quality of the sound, without inserting distortions or unwanted noise.

Keywords: *Comb and all-pass filters, Schroeder's reverberator, loop and reverberation times, LabVIEW FPGA.*

I. INTRODUCTION

We spend most of our lives in reverberant environments, such as concert halls, offices, city streets etc. The presence of reverberation in auditioned music or voice is, in most cases, preferred by listeners [1]. Music recorded with very little or no reverberation is of very poor quality, a lot thinner and weaker than even outdoor symphonic music [2]. However, there is a fine balance to be achieved between little reverberation (dry, lifeless music) and too much reverberation (unintelligible or even muddy music) [1].

There is a distinction between *early* reverberation and *late* reverberation. The former term refers to reflections of a sound, due to nearby surfaces, which occur during the first few tens of milliseconds after the original sound was applied. The latter term refers to effects which occur after the first few milliseconds from signal application, when the number of reflected waves increases significantly, while their amplitude decreases in time. During late reverberation, the echoes move in all directions, their intensity being independent from the receiver's location inside the room [4]. This is why artificial reverberation algorithms represent important tools for recording studios. From steel and spring plates devices, technology has evolved towards digital reverberators, based on linear discrete-time filters [1]. The first and most well known digital reverberator is Schroeder's late reverberator, having a structure based on comb and all-pass filters [4].

One of the technologies which are heavily involved today in digital signal processing is the field programmable gate array (FPGA), a type of digital programmable device. There are many ways to create hardware designs intended for FPGAs:

- writing code in a hardware description language (such as VHDL or Verilog) [5];
- writing a MatLab script (translated into VHDL code by a program like System Generator from Xilinx);
- creating a Virtual Instrument (VI), using LabVIEW FPGA from National Instruments. While hardware description languages are tedious and time consuming, using a tool such as LabVIEW FPGA represents an intuitive way to develop FPGA code, while significantly shortening development and debugging times [7].

This paper presents the FPGA implementation of Schroeder's late reverberator using LabVIEW FPGA, as a rather general method of implementing in hardware reverberation algorithms. Our approach is cell-based, so that the cells composing Schroeder's reverberator (implemented as VIs) could be reused for other reverberators, with minimal effort.

II. THEORETICAL OVERVIEW

A. The Comb Filter

One of the basic cells in Schroeder's late reverberator is the comb filter, used to create replicas of the direct path signal, which are time-delayed and reduced in intensity. The replicas are delayed by the same amount of time [6].

The comb filter consists of a delay, its output multiplied by a gain and fed back to the input [4]. Its structure is presented in Figure 1.

In a reverberator, multiple comb filters are used, arranged in a parallel structure, in order to obtain a long reverberant decay [4]. Because of the identical spacing of the replicas, the sensation of a pitched tone superimposed on the signal is created. This is why, in a reverberator, the comb filter is used in conjunction with at least one all-pass filter [6].

B. The All-Pass Filter

The all-pass filter is the other basic cell in Schroeder's reverberator, obtained by modifying a comb filter and used

Manuscript received September 4, 2010; revised November 03, 2010

for multiplying the number of echoes obtained at the output of the parallel comb structure [3, 4].

The structure of the all-pass filter is presented in Figure 2.

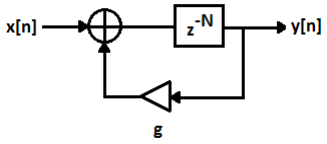

Figure 1. Block diagram of the comb filter.

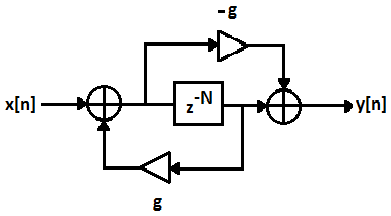

Figure 2. Block diagram of the all-pass filter.

By increasing the number of echoes obtained from the comb filter, the all-pass filter emulates the effect of natural reverberation [6]. By cascading two such filters, the overall echo density is highly improved [1].

C. Schroeder's Late Reverberator

Schroeder was the first to construct artificial reverberators based on digital signal processing, during the early 1960's. His implementation of a late reverberator consists of four comb filters, connected in parallel, and two all-pass filters, cascaded with the parallel comb structure. The design is visible in Figure 3.

D. Loop Time and Reverberation Time

These two terms are very important in characterizing and evaluating the performances of reverberators. The loop time, denoted by τ, refers to the time required for a comb or an all-pass filter to loop back a signal to its input [6]. It is described by (1), where N represents the order of the delay

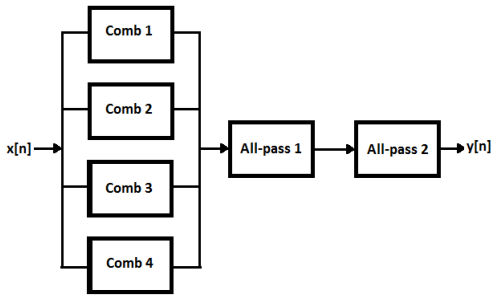

Figure 3. Block diagram of Schroeder's late reverberator.

cell in the comb filter, and f_S represents the sampling frequency of the input signal.

$$\tau = N / f_S. \tag{1}$$

The reverberation time is defined as the time needed for the pressure level of a sound inside a room to decay by 60dB from its initial value [4]. There is a clear link between loop time, reverberation time and gain for each comb and all-pass filter inside the reverberator:

$$g = 10^{\frac{-3\tau}{T_r}}, \tag{2}$$

where g represents the gain of a comb or all-pass filter, τ represents its loop time, while T_r represents its reverberation time.

III. HARDWARE IMPLEMENTATION

The reverberator was implemented in LabVIEW FPGA 8.6.1. The resulting VI was run on a NI CompactRIO-9014 system, with a 3 million gate FPGA backplane, the NI-9104. The implementation was conceived as having a main VI running on the FPGA, containing all the comb and all-pass filters which form the reverberator. The block diagram of this VI is presented in Figure 4. Comb and all-pass filters

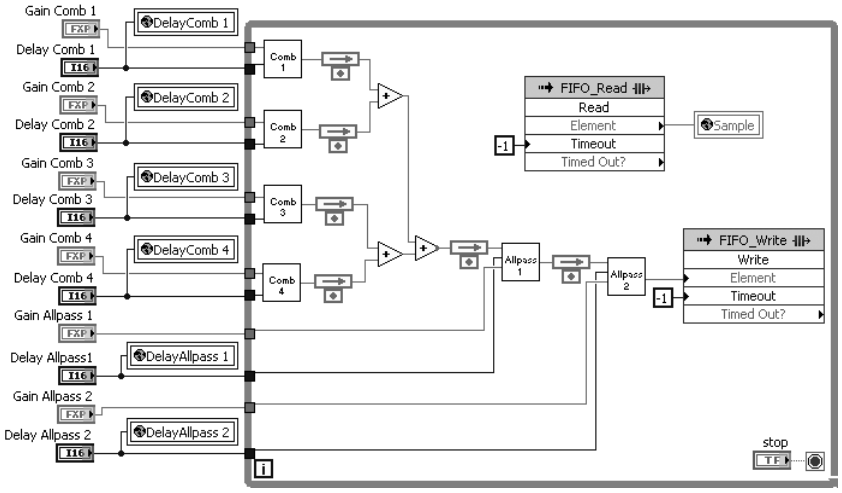

Figure 4. LabVIEW FPGA architecture of Schroeder's late reverberator.

are implemented as subVIs.

The main VI running on FPGA is controlled by a VI running on a PC, which feeds the audio samples to the FPGA through a FIFO, and receives the processed audio samples, through another FIFO. This VI also receives from the user the desired loop and reverberation times, and from them it extracts the actual delays and gains needed by the reverberator. These values are sent to the FPGA by writing the values of some LabVIEW controls. While the audio samples are sent from the PC to the FPGA and the FPGA processes them one by one, the design can also work with live audio data, by feeding data to the FPGA from a sampled analog input instead of a FIFO.

Each comb filter in the implemented reverberator is represented as a subVI on the FPGA; the block diagram of such a subVI is presented in Figure 5. The all-pass filters are also represented as subVIs on the FPGA (Figure 6.).

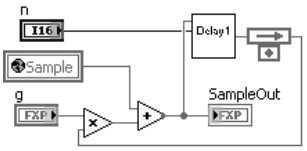

Figure 5. LabVIEW FPGA architecture of a comb filter.

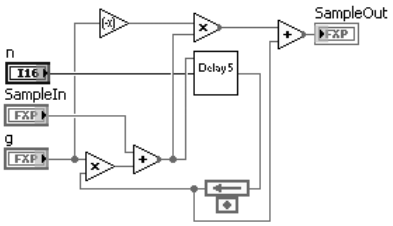

Figure 6. LabVIEW FPGA architecture of an all-pass filter.

In both Figures 5 and 6, the use of a subVI specialized in obtaining delays is visible. The block diagram of this type of subVI is shown in Figure 7. Practically, there is a state machine with three states, "Idle", "PutNew" and "ReadElement". During "Idle" state, the subVI does not perform any action. However, it exits this state whenever the requested delay is different from zero, entering "PutNew" state. During "PutNew" state, the new sample that was received is written to a location in the FPGA block RAM. The current address is incremented, so that the following sample will be written in a memory cell immediately after the one where the current sample is written. If the end of the available memory area is reached, the address is looped back to the beginning. During "ReadElement" state, the address is decremented by the value of the required delay N, so that the sample that was written in the RAM N sampling intervals ago can be read. If the beginning of the available memory area is reached, the address is looped back to the end.

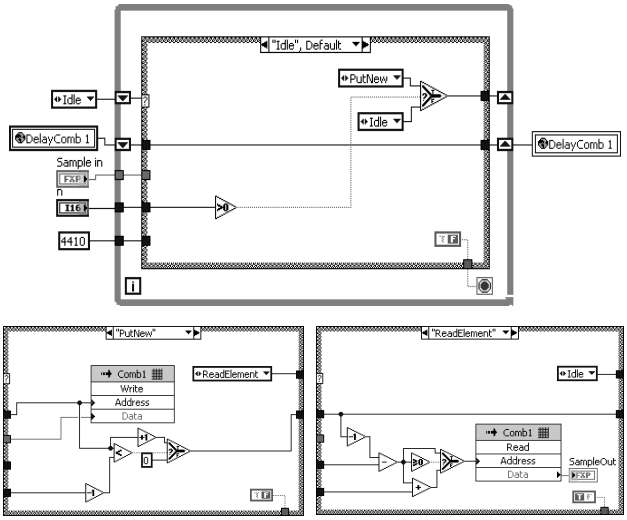

Figure 7. LabVIEW FPGA architecture of a delay cell and the corresponding case loop.

For the FPGA implementation, the numbers were represented as 19-bit fixed-point (FXP) values, with 1 sign bit, 2 bits for the integer part and 16 bits for the decimal part.

The input and output audio sequences were all sampled at 44.1 kHz, for maximum quality and compatibility with existing digital audio standards. However, the LabVIEW FPGA implementation was optimized so that every input sample is processed much faster, in around half a sampling period. This means that much higher sampling frequencies could be used, up to around 90 kHz, while keeping the reverberator working in real time.

IV. EXPERIMENTAL RESULTS

The loop times that were adopted for the comb and all-pass filters are based on values recommended by [6]. These values are presented in Table 1. The values of the reverberation times for the all-pass filters that were employed are 96.83 ms and 32.92 ms, respectively, based on recommendations made by [6]. With these values, three different high-quality voice recordings were tested: recording no. 1 belonged to a man, while recordings no. 2 and no.3 belonged to women. The reverberation time of the comb filters was adjusted so that the output voice is enhanced, compared to the input.

To avoid obtaining too much reverberation, a comparatively auditioning decision between the input and output of the reverberator was taken, for each of the three recordings. The results of this test are shown in Table 2. One can notice that, in the case of the female voices, significantly less reverberation time (0.4 to 0.48 seconds) was needed in order to improve the quality of the recordings. For the male voice, 0.6 seconds of reverberation time was required for a significant improvement in recording quality.

Filter type	Loop time [ms]
Comb filter no. 1	29.7
Comb filter no. 2	37.1
Comb filter no. 3	41.1
Comb filter no. 4	43.7
All-pass filter no. 1	5
All-pass filter no. 2	1.7

Table 1. Loop times of the comb and all-pass filters.

Parameter	Voice sequence no.		
	1	2	3
Comb filters reverberation time [ms]	600	400	480
Allpass filter no. 1 reverberation time [ms]	96.83	96.83	96.83
Allpass filter no. 2 reverberation time [ms]	32.92	32.92	32.92

Table 2. Experimentally obtained reverberation times.

The next experiment on the implemented reverberator involved the application of an impulse to its input, and the recording of its output, thus obtaining the reverberator's impulse response. This experiment was performed for each of the three reverberation times obtained during the previous experiment.

Figures 8, 9 and 10 show the plots of the impulse response in these three cases. It can be observed that the impulse response has a longer decay time in Figure 8, corresponding to the longer reverberation time of 0.6 seconds. The plots in Figures 9 and 10 are rather similar, due to the comparable reverberation times employed for the comb filters (0.4 seconds and 0.48 seconds, respectively).

V. CONCLUSIONS

FPGAs represent a viable and accessible way of implementing digital signal processing algorithms, from rather simple to very complex ones.

In our paper, we proved that through its intuitive nature, LabVIEW FPGA can be successfully used for developing artificial reverberation algorithms running on FPGAs, while reducing development and debugging times compared to hardware description languages.

ACKNOWLEDGEMENTS

The authors would like to thank National Instruments Romania, for their continued support during the development and testing of the project. The Romanian National University Research Council under Grant ID 1057 entitled "2.5D Modeling of Sound Propagation in Rooms and Improvement of Room Acoustical Properties using Digital Implementations" sponsored this work.

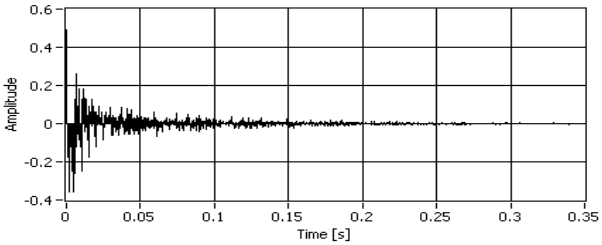

Figure 8. Impulse response of the implemented Schroeder reverberator, with the reverb times obtained for voice sequence no. 1.

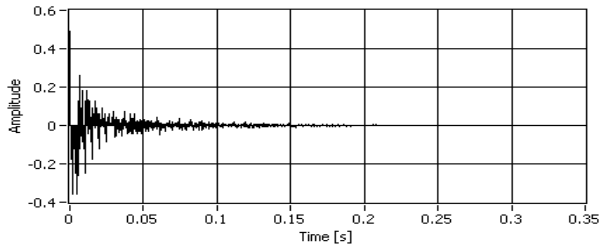

Figure 9. Impulse response of the implemented Schroeder reverberator, with the reverb times obtained for voice sequence no. 2.

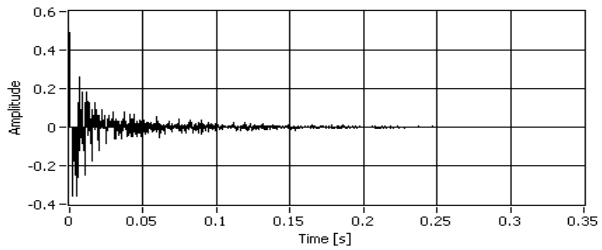

Figure 10. Impulse response of the implemented Schroeder reverberator, with the reverb times obtained for voice sequence no. 3.

REFERENCES

[1] M. Kahrs and K. Brandenburg, *"Applications of Digital Signal Processing to Audio and Acoustics",* Kluwer Academic Publishers, 2002, pp. 85-131.

[2] F. Alton Everest, *"The Master Handbook of Acoustics"*, 4th ed., McGraw-Hill, 2001, pp.129-164.

[3] U. Zölzer, X. Amatriain, D. Arfib, J. Bonada, G. De Poli et al., *"DAFX - Digital Audio Effects"*, John Wiley & Sons, 2002, pp. 170-186.

[4] N. Toma, M. D. Ţopa, E. Szopos, "Reverberation Algorithms", *Acta Technica Napocensis, Electronics and Telecommunications*, vol. 46, Number 2/2005, pp. 27-34.

[5] I. Dornean, M. D. Ţopa, B. S. Kirei, "Digital Implementation of Artificial Reverberation Algorithms", *Acta Technica Napocensis, Electronics and Telecommunications*, vol. 49, Number 4/2008, pp. 1-4.

[6] E. Doering, "Schroeder Reverberator", *Connexions*, March 18, 2008. Available: http://cnx.org/content/m15491/1.2/.

[7] "Creating Custom Hardware with LabVIEW. NI LabVIEW FPGA Module", *National Instruments*, 2005. Available: http://www.ni.com/pdf/products/us/labview_fpga_module.pdf