A 3D-Audio Reconfigurable Processor

Dimitris Theodoropoulos
D.Theodoropoulos@tudelft.nl

Georgi Kuzmanov
G.K.Kuzmanov@tudelft.nl

Georgi Gaydadjiev
g.n.gaydadjiev@tudelft.nl

Computer Engineering Laboratory, Electrical Engineering Dept.
Delft University of Technology
Postbus 5031, 2600 GA, Delft
The Netherlands

ABSTRACT

Various multimedia communication systems based on 3D-Audio algorithms have been proposed by researchers from the acoustic data processing domain. However, all systems reported in the literature follow a PC-based approach that introduces processing bottlenecks and excessive power consumption. In order to alleviate these problems, we propose a reconfigurable 3D-Audio processor that can record and render sound sources concurrently. Audio recording and rendering are performed by two hardware accelerators exploiting the beamforming and the Wave Field Synthesis algorithms. The theoretical scalability of the proposed processor is explored with respect to systems consisting of different microphone and loudspeaker arrays configurations. A working FPGA prototype is compared against a software implementation on a Core2 Duo system. Results suggest that the proposed reconfigurable hardware solution can process data up to 2.4x faster than the software approach, while power consumption is approximately 7 Watts according to the Xilinx XPower report.

Categories and Subject Descriptors: B.7.1 Hardware: Algorithms implemented in hardware

General Terms: Design, Performance

Keywords: 3D-Audio, Wave Field Synthesis, Beamforming, Reconfigurable Computing, Communication Systems

1. INTRODUCTION

Over the last decade, many companies and researchers have developed multimedia communication systems tailored to teleconferencing and online education. However, commercial products use stereophonic systems that inherent all the problems of stereophony [1], thus providing an average audio quality. Based on this fact, many researchers from the acoustic domain have developed systems that exploit true 3D-Audio algorithms, e.g. the Wave Field Synthesis (WFS) [2], to render the sound environment. The main advantage of such systems over the commercial ones is that they provide an immersive audio experience to the listener, thus significantly improving interaction among all meeting participants. However, these systems are implemented using standard desktop PCs. This approach introduces performance bottlenecks and excessive power consumption. In order to alleviate these problems, we propose a 3D-Audio hardware reconfigurable processor that provides an improved performance and lower power consumption compared to a standard PC-based approach. The main contributions of this paper are the following:

- A reconfigurable 3D-Audio processor is proposed that can efficiently record and render audio sources concurrently;
- A study based on the number of input channels and the available hardware resources is reported. Results for 3D-Audio systems, ranging from simple implementations with few microphones and loudspeakers to complex setups consisting of tens of input and output channels are presented;
- A performance and power consumption evaluation of the proposed design is presented. We compare our working hardware prototype against software implementations on a Core2 Duo at 3.0 GHz. Speedup of 2.4x compared to the software approach are reported with an order of magnitude lower power consumption.

The rest of the paper is organized as follows. Section 2 provides a brief theoretical background of the beamforming and WFS algorithms, while also describes a few commercial and experimental communication systems. Section 3 describes the proposed audio processor. In Section 4, our performance evaluation and resource utilization analysis are presented. Finally, Section 5 concludes the discussion.

2. BACKGROUND AND RELATED WORK

The Beamforming algorithm: Figure 1 illustrates a filter-and-sum beamformer [3]. The microphones array samples the propagating wavefronts and each microphone is connected to a FIR filter. All filtered signals are summed up to extract the desired audio source, in order to strengthen the primary source and attenuate any ambient noise. However, in order to effectively extract a moving source, it is mandatory to reconfigure the FIR filter coefficients according to the source location. For example, as is illustrated in Figure 1, a source tracking device updates the filter coefficients

FPGA'10, February 21–23, 2010, Monterey, California, USA.

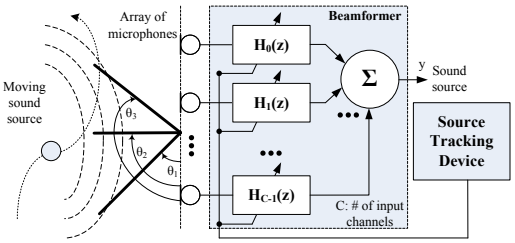

Figure 1: Filter-and-sum beamformer.

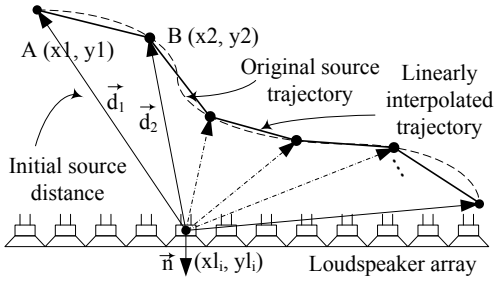

Figure 2: Linear interpolation of a sound source.

with a different set when a moving source is inside the aperture defined by the $\theta_2 - \theta_1$ angle and when it is inside the aperture defined by the $\theta_3 - \theta_2$ angle (beamsteering).

The WFS algorithm: The WFS acoustic algorithm was initially proposed by Berkhout [2]. Assuming the loudspeaker setup illustrated in Figure 2, in order to drive a loudspeaker with $(xl_i,\ yl_i)$ coordinates so as the rendered sound source has a perceivable distance $|\overrightarrow{d_1}|$ from it, the so called *Rayleigh 2.5D operator* [4] has to be calculated:

$$Q_m(\omega, |\overrightarrow{d_1}|) = S(\omega)\sqrt{\frac{jk}{2\pi}}\sqrt{\frac{Dz}{z+Dz}}\frac{z}{|\overrightarrow{d_1}|}\frac{exp(-jk|\overrightarrow{d_1}|)}{\sqrt{|\overrightarrow{d_1}|}} \quad (1)$$

where k=ω/c is the wave number, c is the sound velocity, z is the inner product between $\overrightarrow{n}$ and $\overrightarrow{d_1}$, Dz is reference distance, i.e. the distance where the Rayleigh 2.5D operator can give sources with correct amplitude, $S(\omega)$ is the source term, $\sqrt{\frac{jk}{2\pi}}$ is a 3dB/octave correction filter, $\sqrt{\frac{Dz}{z+Dz}}$ is an amplitude decay and exp(-jkr) is a time delay. When a sound source moves from A(x1, y1) to B(x2, y2), the traversed trajectory is linearly interpolated, as illustrated in Figure 2. Further details can be found in [5], [6] and [4].

Related work: There are various commercial products ([7], [8]) regarding multimedia communication systems that provide excellent video quality. Audio is captured by a microphones array and is delivered to the remote location using two stereo channels or high definition audio. However, researchers from the acoustic data processing domain have proposed various systems that alleviate the problems of stereo-based systems [1]. For example, in [9] the authors describe a 3D-Audio system consisting of 12 linearly placed microphones connected to a standard PC. The sound source is tracked through audio and video tracking algorithms, while the beamformer is steered accordingly. The audio signal is extracted through beamforming and encoded using the MPEG2-AAC or G722 encoders. The encoded signal is received from a second remote PC and the audio signal is rendered using the WFS technology through a 10 loudspeakers array. Another similar system is presented in [10]. The authors describe a real-time audio system that exploits the beamforming technique and the WFS technology. The system consists of 4 PCs and performs sound recording from a remote location A, transmits it to another one B, and renders it through a speaker array utilizing WFS.

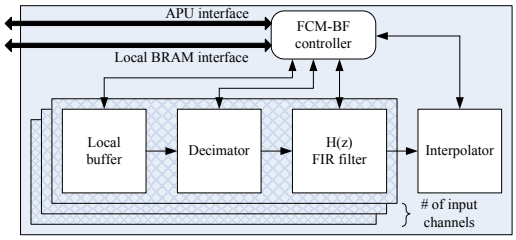

Figure 3: The beamforming hardware accelerator.

3. PROPOSED DESIGN

Sound acquisition hardware accelerator: Figure 3 depicts the FCM-BF[1] [11], which receives 16-bit signed audio samples and all calculations are performed in a fixed-point arithmetic format. The accelerator can be customized according to the number of microphones (input channels). The gridded area in Figure 3 shows the modules that have to be replicated for every microphone. A *local buffer* is used to temporarily store 1024 16-bit samples before they are processed, which are fetched through the *local BRAM interface*. As soon as all samples from all microphones are stored to their corresponding local buffers, the *FCM-BF controller* initiates the samples processing. All input channels are processed concurrently since there is no dependency among them. The *decimator* is used to downsample the input data. Every *H(z) FIR filter* includes a specific set of FIR filter beamsteering coefficients banks. The output signals of all filters are accumulated and the beamformed signal is upsampled by the interpolator to the initial sampling frequency. The current implementation utilizes the Auxiliary Processor Unit (APU) interface to communicate with the host processor.

Sound Rendering hardware accelerator: Figure 4 illustrates the FCM-WFS [6]. As input, it receives 16-bit signed audio samples, while all computations are performed in a fixed-point arithmetic format. The accelerator can be parameterized according to the number of loudspeakers (output channels) and is connected to the host processor utilizing also the APU interface. As soon as the host processor invokes the WFS hardware accelerator, the *FCM-WFS Controller* accesses a local BRAM to fetch 1024 samples. These samples are processed from a 3dB/octave frequency correction *FIR filter* and stored to the *Samples buffer* inside the *WFS engine* module. The WFS hardware accelerator distributes the required processing tasks for a certain set of loudspeakers to different RUs (Rendering Units). The gridded area in Figure 4 illustrates the RU internal modules. The *Loudspeakers coordinates* buffer keeps the coordinates of all loudspeakers that will be processed from a particular RU. Within each RU, all loudspeakers are processed iteratively, whereas in every iteration 1024 audio samples are computed. The *Preprocessor* module calculates the initial and final source distance from a particular loudspeaker

[1]The Xilinx terminology is used, thus FCM is an abbreviation for Fabric Coprocessor Module. BF is an abbreviation for BeamForming.

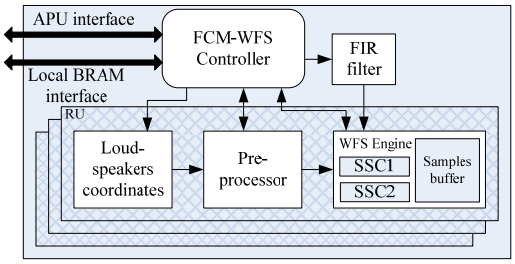

Figure 4: The WFS hardware accelerator.

for every 1024-samples time frame. Furthermore, it calculates the source velocity and amplitude decay, which are forwarded to the *WFS Engine* module to process all 1024 samples. In order to reduce the processing time, within the *WFS Engine* two Samples Selection Cores (SSCs) submodules are employed and process concurrently two samples per clock cycle. All calculated samples are stored back to the local BRAM.

The 3D-Audio reconfigurable processor: Figure 5 depicts the complete proposed design that is based on the beamforming and WFS hardware accelerators mapped onto a Virtex4 FX60. This device integrates two PowerPC general purpose processors. The 3D-Audio processor is separated into two parts, the *Sound Acquisition* and the *Sound Rendering* ones. The *Sound Acquisition* part utilizes the *PowerPC_0* as host processor and the beamforming hardware accelerator, illustrated by the shaded *FCM-BF* box in Figure 5. As it was mentioned earlier, the *FCM-BF* is connected to the *PowerPC_0* using the APU interface, and on-chip *PLB BRAM 0* through its PORTB. This shared memory implementation allows the FCM-BF to access memory more efficiently compared to accessing it through the Processor Local Bus 0 (PLB0). *PLB BRAM 0* is also connected through its PORTA to PLB, in order to be able to exchange data between the *PowerPC_0* and the *FCM-BF*. A second PLB (PLB1) is employed to connect a 16 Kbytes on-chip BRAM to host the software that is executed from the *PowerPC_0*. This approach reduces considerably the workload and the congestion of the PLB0, since all program instructions are fetched using the dedicated PLB1, while all data are fetched through the PLB0. A DDR SDRAM external memory is also connected to PLB0 to store audio samples recorded from a microphones array. In order to reduce the impact of memory data transfers to the total execution time, *DMA controller 0* and *Interrupt controller 0* are employed and connected to PLB0. Finally, in order to provide debugging capabilities, an RS232 module is also connected to PLB0.

A similar approach is followed regarding the *Sound Rendering* part of the audio processor. In this case, *PowerPC_1* is used as the host processor. Another 16 Kbytes of on-chip BRAM are available to store the software, connected through the dedicated PLB1. The WFS hardware accelerator, illustrated by the *FCM-WFS* shaded box in Figure 5, is also connected to the host processor through the APU interface. Furthermore it is also connected to the PORTB of *PLB BRAM 1* in order to efficiently exchange data with *PowerPC_1*. The same memory provides PORTA, which is connected to PLB0. Again the PLB0 is mainly used for data transfers, while PLB1 is explicitly used to fetch instructions from the instructions memory. Similarly to the approach followed by the *Sound Acquisition* part, *DMA Controller 1* and

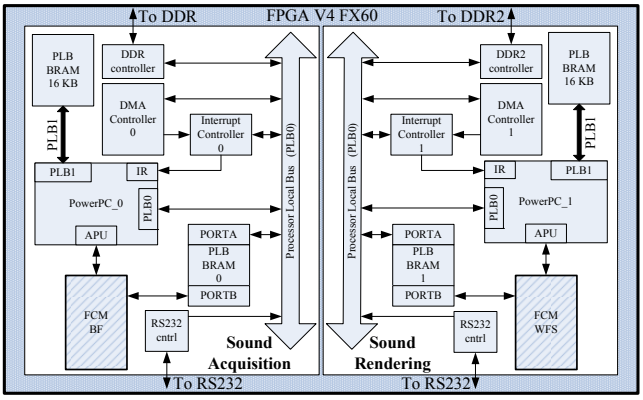

Figure 5: The 3D-Audio reconfigurable processor.

Table 1: FPGA resource utilization.

	Slices	DSP Slices	BRAM (bytes)
BF (other resources)	282	2	4608
WFS (other resources)	6356	0	0
BF (1 channel)	275	2	20480
WFS (1 RU)	3298	16	41984
System infrastructure	9062	0	87552

Interrupt controller 1 are employed to accelerate data transfers between the DDR2 SDRAM and the on-chip BRAM. Finally, again another RS232 unit is also connected to PLB0, in order to provide debugging functionality to the *Sound Rendering* part.

4. EXPERIMENTAL RESULTS

Hardware prototype: In order to evaluate the proposed design, a hardware prototype was developed with Xilinx ISE 9.2i and EDK 9.2i CAD tools and mapped onto a Xilinx ML410 FPGA board with a Virtex4 FX60 FPGA. The current prototype consists of 16 input channels and 1 RU, thus up to 32 output channels. Furthermore, it occupies 19483 slices (78%), 50 DSP slices (39%) and 488448 bytes (91%) of the FPGA resources. The two PowerPCs run at 200 MHz, while the rest of the system functions at 100 MHz.

Design scalability: Table 1 presents the resource distribution among the processor modules. The first two rows contain the resources required to control the FCM-BF and FCM-WFS designs respectively. The third and the fourth row contain all required resources to instantiate a new input channel and RU to the FCM-BF and FCM-WFS respectively. The fifth row presents all resources occupied by the system infrastructure, like PLBs, on-chip memories, and the external interrupt and DMA controllers. Two Ethernet MAC IPs are also considered in the infrastructure resources reported in the fifth row, since they will be included to the design to perform further experiments by connecting two FPGA boards.

We investigated various system setups that could be mapped on medium and large-sized FPGAs. Our study considers the Virtex4 FX60, FX100 and FX140 FPGAs that contain two PowerPC processors. Figure 6 depicts 14 systems with different number of input and output channels. The number of input channels was selected to range from 4 up to 56 to investigate the complexity of various system setups that each FPGA could accommodate. FX60 FPGAs can be used to accommodate a system with up to 16 input channels and 32 loudspeakers. More complex systems can be built using

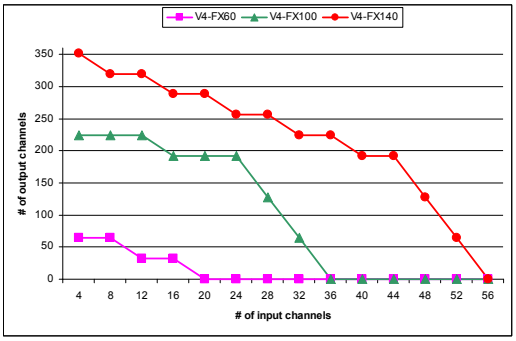

Figure 6: Input and output channels tradeoff.

Table 2: SW and HW implementations timings.

Sources	Audio time (msec)	SW (msec)	HW (msec)	SpeedUp
1	43683	10782	2221	**4.8**
2	21842	12577	4442	**2.8**
3	14561	19361	6663	**2.9**
4	10921	21987	8884	**2.4**

FX100 FPGAs, which could accommodate setups with 32 microphones and 64 loudspeakers. Finally, FX140 FPGAs can be used for systems with up to 52 microphones and 64 loudspeakers.

Comparison against software implementation: In order to evaluate the performance of our prototype, 64 MB of data of a single audio source were tested and processed. Audio samples are signed 16-bit and the sampling frequency is 44.1 KHz. In order to test the FCM-BF module, audio data were stored in the DDR memory. In each iteration, 1024 samples=2 Kbytes per channel are processed. Consequently, since the current prototype has 16 input channels, in each iteration 16*2=32 Kbytes are processed, requiring 2048 iterations in total to process all data stored in the DDR. The extracted audio source from FCM-BF was stored again back to the DDR. In order to test the FCM-WFS module, the extracted audio source from FCM-BF was used as input already available to the DDR2 SDRAM. Again, in each iteration, 1024 audio samples are processed, requiring in total 2048 iterations to complete the entire audio source rendering through 32 loudspeakers. The results are stored back to DDR2.

Table 2 shows the timing results when executing the beamforming and the WFS algorithms in OpenMP-annotated software and hardware, and the obtained speedup. When there are up to two sources, both implementations can process data in real-time, since the execution times are shorter than the actual audio time. However, the software implementation fails to meet the timing limits with three or four sources, since in both cases the execution time is longer than the audio one. In contrast, the proposed hardware approach, can process all of them in real-time. It should be noted that these time measurements include the data transfers between the external memory and the on-chip one. Finally, desktop PCs with high-end CPUs, consume tens of Watts. In contrast, the proposed FPGA solution, according to Xilinx Xpower, requires approximately 7 Watts, which is an order of magnitude lower than the PC-based approach.

5. CONCLUSIONS

This paper presented a 3D-Audio reconfigurable processor that can record and render audio sources concurrently using the beamforming and WFS algorithms. A working hardware prototype was implemented and mapped onto a Virtex4 FX60 FPGA. Furthermore, the design scalability was examined based on the number of input channels and available resources. In addition, conducted performance tests showed that the proposed design can achieve an execution speedup of 2.4x against the Core2 Duo solution. Power consumption is also decreased to approximately 7 Watts, which is at least an order of magnitude comparing to the software approach.

6. ACKNOWLEDGEMENTS

This work was partially sponsored by hArtes, a project (IST-035143) of the Sixth Framework Programme of the European Community under the thematic area "Embedded Systems"; and the Dutch Technology Foundation STW, applied science division of NWO and the Technology Program of the Dutch Ministry of Economic Affairs (project DCS.7533).

The authors would like to explicitly thank Lars Hörchens and Jasper van Dorp Schuitman from the Laboratory of Acoustical Imaging and Sound Control of the TU Delft University for their valuable contribution to accomplish this work.

7. REFERENCES

[1] G. Theile and H. Wittek, "Wave field synthesis: A promising spatial audio rendering concept," in *Acoustical Science and Technology*, 2004, pp. 393–399.

[2] A. Berkhout, et. al., "Acoustic Control by Wave Field Synthesis," in *Journal of the Acoustical Society of America*, vol. 93, May 1993, pp. 2764–2778.

[3] B. V. Veen and K. Buckley, "Beamforming: a versatile approach to spatial filtering," in *IEEE ASSP Magazine*, vol. 5, April 1988, pp. 4–24.

[4] J. van Dorp Schuitman, "The Rayleigh 2.5D Operator Explained," Laboratory of Acoustical Imaging and Sound Control, TU Delft, The Netherlands, Tech. Rep., June 2007.

[5] P. Vogel, "Application of Wave Field Synthesis in Room Acoustics," Ph.D. dissertation, TU Delft, The Netherlands, 1993.

[6] D. Theodoropoulos, et. al., "Reconfigurable Accelerator for WFS-Based 3D-Audio," in *IEEE RAW*, May 2009, pp. 1–8.

[7] LifeSize Inc., "LifeSize Conference 200 Series."

[8] Polycom Inc., "Polycom CX5000 Conference Station."

[9] J. Beracoechea, et. al., "On building Immersive Audio Applications Using Robust Adaptive Beamforming and Joint Audio-Video Source Localization," in *EURASIP Journal on Applied Signal Processing*, June 2006, pp. 1–12.

[10] H. Teutsch, et. al., "An Integrated Real-Time System For Immersive Audio Applications," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, October 2003, pp. 67–70.

[11] D. Theodoropoulos, et. al., "A Reconfigurable Beamformer for Audio Applications," in *IEEE SASP*, July 2009, pp. 80–87.