Journal of Electrical and Electronics Engineering Research Vol. 1(1) pp. 023-029, November, 2009
Available online http://www.academicjournals.org/JEEER

Full Length Research Paper

Implementation of DSP algorithms on reconfigurable embedded platform

J. S. Parab*, R. S. Gad and G. M. Naik

Electronics Section, Department of Physics, Goa University, Goa, India.

Accepted 9 September, 2009

Field programmable PLD's are becoming a standard in hardware technologies, as application demands have out placed the conventional processor's ability to deliver. The right combination of price, performance, ease-of use, along with significant power savings, can be achieved by using a Field Programmable Gate Array (FPGA). The versatility of these devices with the EDA tools such as Quartus, ISE, and Mentor Graphics integrated with MATLAB-Simulink gives upper hand for designer in any complex Digital signal processing (DSP) designs. Altera and Xilinx has DSP generator and System generator as target specific tools which support various IP cores libraries for such designs. We have designed an embedded system with Xilinx Spartan III based FPGA for real time audio processing. Various audio effects such as Echo, Reverberation, Fading, Flanging etc. can be demonstrated for real time performance. The designed system has 12-bit ADC tuned for the base-band signal unto 500 KHz. Also, the system can be configured as a DSP processor with IP cores like FFT, convolution etc. The reconstruction of analog signal is achieved with the help of 12-bit DAC converter module. Designed board has many applications in the field of biomedical, consumer, industrial and military. The same audio effects were tested on Altera CYCLONE II based DSP development kit.

Key words: DSP, ASP, ECHO, embedded system, Spartan -III, FPGA, altera.

INTRODUCTION

Signals play an important role in our daily life. Signals that we encounter frequently are speech, music, picture and video signals (Richard, 2004). Often, signal is spatio-temporal in nature. Speech and music signals represent air pressure as functions of time at a point in space. Most signals we encounters are generated naturally, however, signal can be generated synthetically or by computer. The objective of signal processing is to extract the information carried by the signal and this extraction of information depends on the type and nature of signal carried by it. There are four methods of digital sound synthesis such as wavetable, spectral, non-linear and synthesis by physical modeling. Wavetable synthesis produces recorded or synthesized musical events stored in the digital memory and played back on demand. Spectral synthesis produce sound from frequency domain model basically by superposition of basis functions with time-varying amplitudes. Non-linear synthesis is a frequency modulation technique of time dependent phase term in the sinusoidal basis function. Physical modeling models the sound production methods of the vibrating physical structures by partial differential equations (Alles, 1980). We have implemented wavetable analysis in our design. The highest sampling frequency reported presently is around 1GHz. Such high frequencies are not usually used in practice since the achievable resolution of the A/D converter given by the word length decreases with an increase in the speed of the converters. For example, the reported resolution of an A/D converter operating at 1GHz is 6 bits (Poulton et al., 1987). On the other hand in most applications, the required resolution of an A/D converter is from 12 - 16 bits. Consequently, the sampling frequency of at most 10 MHz is presently a practical upper limit. Upper limit is becoming larger and larger with advances in technology. We have designed system having 12-bit ADC, having 1MHz sampling frequency.

*Corresponding author. E-mail. jsparab@unigoa.ac.in. Tel: 0832-6519342. Fax: +091-0832-2451184.

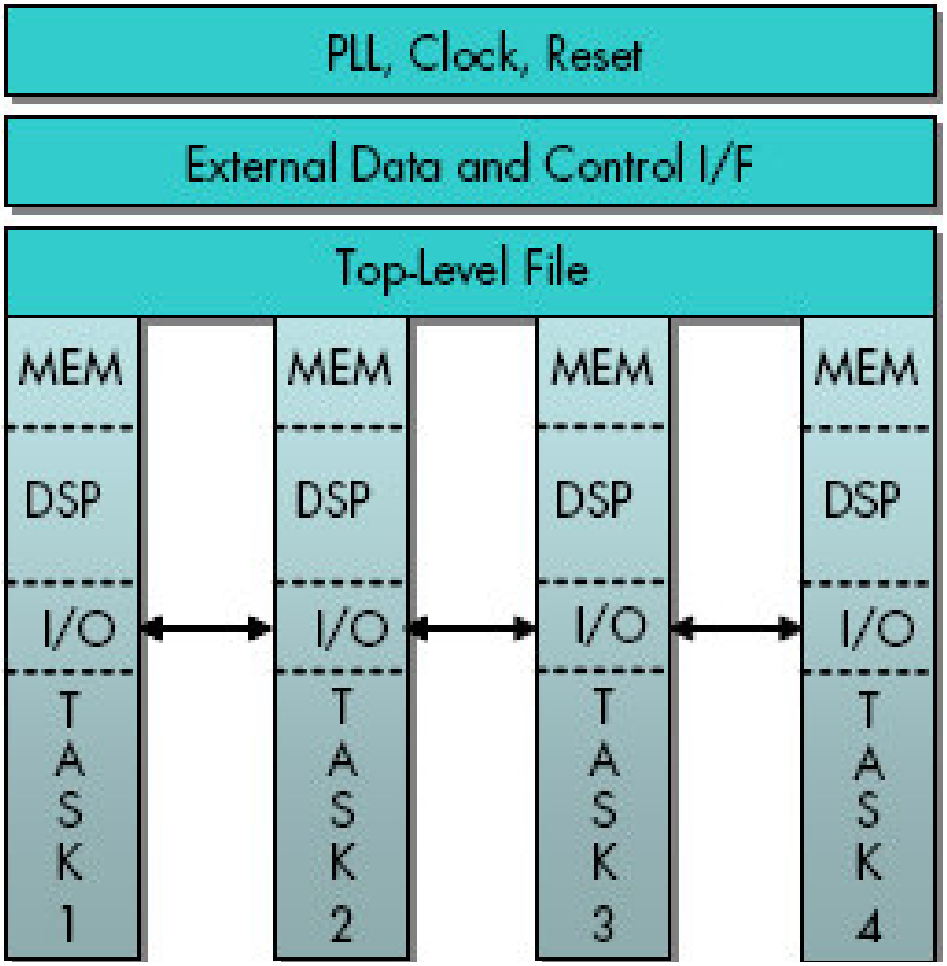

Figure 1a. FPGA block diagram.

PROCESSORS PLATFORMS: FPGA AND DSP

Two types of programmable platforms are used in the Embedded and VLSI design application domain that is DSP and FPGAs.

DSP are a specialized form of microprocessor, while FPGAs are a form of highly configurable hardware. DSP processors have conventionally moved to higher levels of performance through a combination techniques such as increasing clock cycle speeds, increasing the number of operations performed per clock cycle, adding optimized hardware co-processing functionality (such as a Viterbi decoder), implementing more complex instruction sets, minimizing sequential loop cycle counts, adding high performance memory resources, implementing modifications including deeper pipelines and superscalar architectural elements. However, ultimately, each of these design enhancements seeks to increase the parallel processing capability of an inherently serial process.

The following factors make FPGAs promising, particularly for high performance computing applications: (i) The potential for thousand-fold parallelism (ii) The embedding of control logic (iii) Presence of on-board memory in FPGA also has significant performance benefits. For one, having memory on-chip means that the processor logic's memory access bandwidth is not constrained by the number of I/O pins on device (iv) FPGA with greater capacity can occupy the same board footprint as an older device, allowing performance upgrades without board changes. Advantages of FPGAs for high performance computing are discussed in depth in one of the latest references (Kamat et al., 2009).

Diagram in the Figure 1a shows the how FPGA resources assigned can be tailored to the task requirements, which can be broken up along logical partitions. This makes a well-defined interface between tasks and largely eliminates unexpected interaction between tasks, because each task runs continuously, much less memory is required than in the digital signal processor, which must buffer the data and process in batches. As FPGAs distribute memory throughout the device, each task is permanently allocated the dedicated

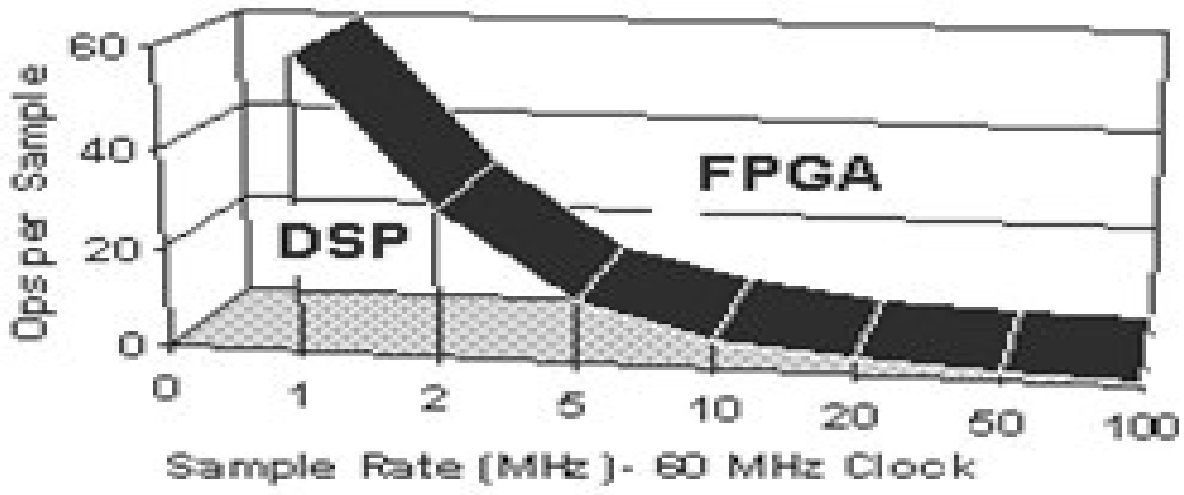

Figure 1b. FPGAs are a better solution in the region above the curve.

memory it needs. This provides a high degree of isolation between tasks and results in modification of one task being unlikely to cause unexpected behavior in another task. This, in turn, allows developers to easily isolate and fix bugs in a logical and predictable fashion. In the past, the use of digital signal processors was nearly ubiquitous, but with the needs of many applications outstripping the processing capabilities of digital signal processors (measured in millions of instructions per second (MIPS)), the use of FPGAs is growing rapidly (Figure 1b). The comparison between digital signal processors and FPGAs focuses on MIPS comparison, which, while certainly important, is not the only advantage of an FPGA. Equally important, and often overlooked, is the FPGA's inherent advantage in product reliability and maintainability.

Higher performance implementations of specific DSP algorithms are increasingly available through implementation within FPGAs (http://www.andraka.com /dsp.htm.). Ongoing architectural enhancements, advancement in development tool, speed increases and cost reductions are making FPGA implementation attractive for an increasing range of DSP-dependent applications. FPGA technology advances have increased clock speeds and available logic resources and beyond the range required to implement many DSP algorithms effectively at an attractive price. FPGA implementation provides the added benefits of reducing costs along with design flexibility and future design modification options.

EMBEDDED SYSTEM IMPLEMENTATION FOR DSP USING XILINX SYSGEN

Objectives were to design a low cost, high speed and reliable DSP system which includes a digital signal processing with real time I/O control. FPGA is used to process the digitized data sampled by the ADC and then output the same to the DAC module.

The aims and objectives of the design are summarized as follows:

(1) To design a general purpose embedded system using FPGA for real time digital signal processing.
(2) Configuring the FPGA as a customized digital signal processor.
(3) Port the soft processor IP core like NIOS II interface with the above design as a co-processor.

The diagram in Figure 2 shows the detailed implementation of the hardware using Spartan III FPGA. A digital signal is applied as an input to the operational amplifier LM324 which acts as a buffer and is then given to the 12-bit ADC AD7891 module. This ADC converts the analog signal into the digital domain. The digitized data is then fed to the FPGA XC3S200 Spartan-III for further processing. The FPGA is programmed using Xilinx ISE webpack over JTAG interface. The FPGA is programmed through IDE Xilinx ISE Webpack 6.3i. The processed data from FPGA is then converted back to analog form using a 12-bit DAC AD7541. The reconstructed analog signal is then given to LM324 for amplification.

The design hardware has several handshaking signals as shown in the Figure 3 for external on board ADC and DAC interface. The data from the ADC after conversion is routed over 12-bit data bus known as 'db'. The processed data over a user's logic is output to DAC over 'dac_out' (12-bit). The DAC output is reconstructed to analog form for representation to external world. The logic in the FPGA can be manually written using HDL for any computation in digital signal domain. With increasing complexities of systems and varied levels of expertise of embedded designers, there is an increasing need for tools that provide a higher level of abstraction, empowering the

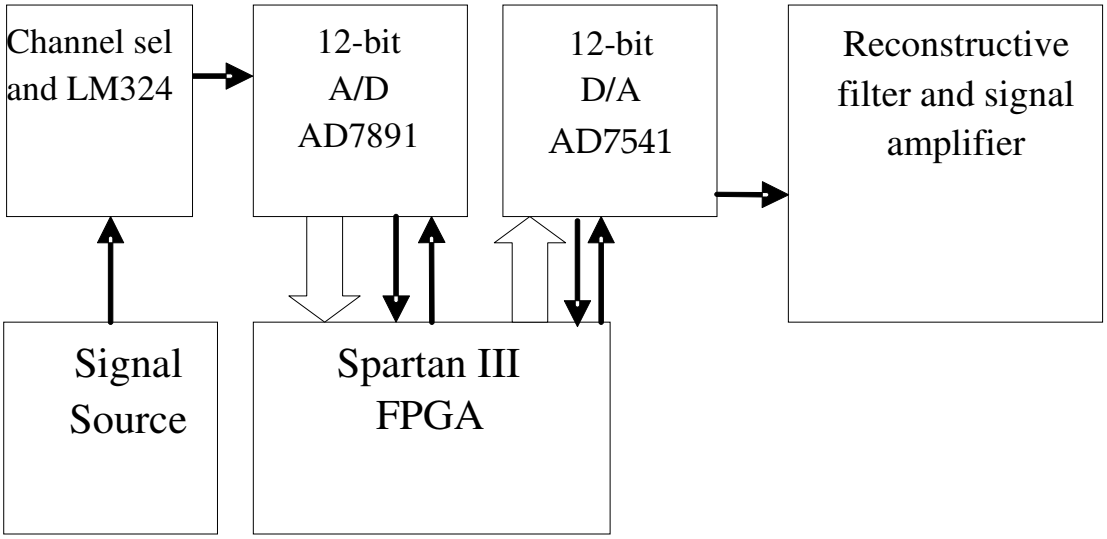

Figure 2. Block diagram of embedded system.

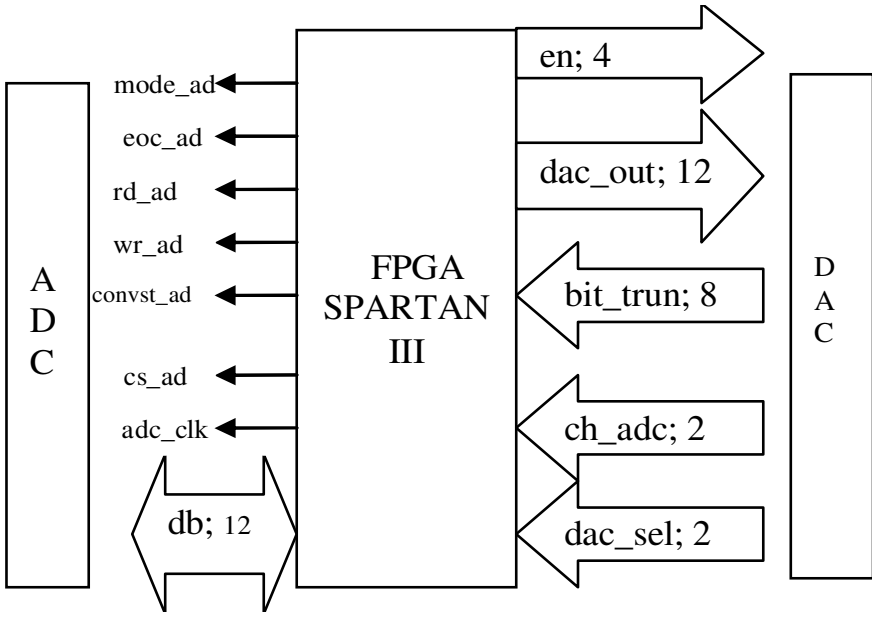

Figure 3. Higher level schematic.

empowering the domain experts to use DSPs in building embedded systems rather than spending precious time at the prototyping stage. Graphical programming paradigms have continually evolved to address this problem. There are number of tools that allow a mixture of visual and textual programming such as Signal (Benveniste and Guernic, 1990), Lustre (Halbwachs et al., 1991) and Silage (Hilfinger, 1985).

Completely graphical programming environments such as Ptolemy (http://ptolemy.eecs.berkeley.edu/.) from University of California, Berkeley and National Instruments LabVIEW have evolved to encompass different application domains.

System generator (Xilinx) (http://xilinx.com/products /design_resources/dsp_central/grouping/index.htm) or DSP generator (ALTERA) (http://altera.com/technology /dsp/dsp-index.jsp.) provides libraries with Simulink customized for respective vendor's targets which generates the HDL code for variety of DSP functionality. The proper EDA interface of the automatic code generator is shown in Figure 4.

Here, the required logic code of user is generated by

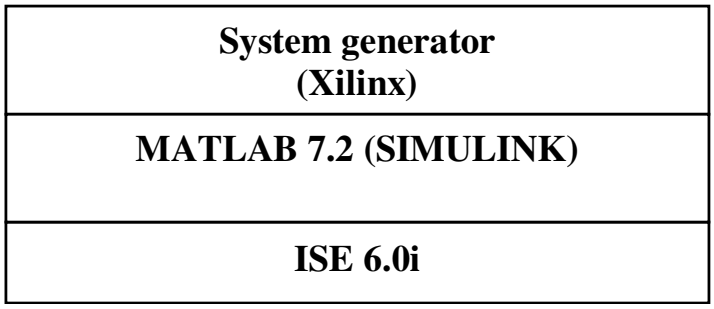

Figure 4. Xilinx EDA interface for DSP application.

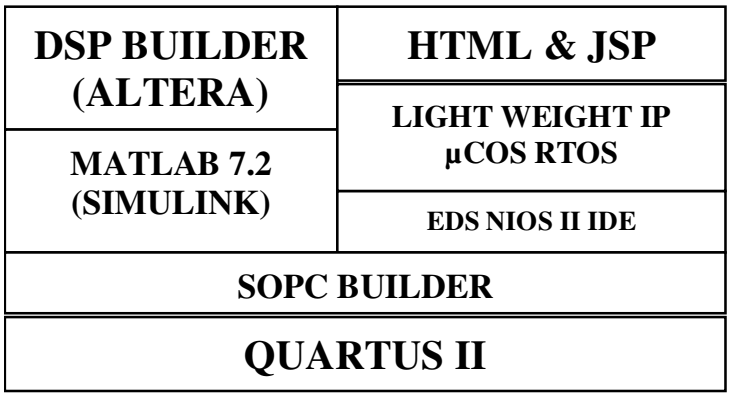

Figure 5. ALTERA EDA interface.

the system generator. The generated system generator compiles the intermediate code in form of HDL and the same is available as component of HDL. The component is then mapped with the entity of HDL code of the ADC and DAC interface as shown in figure 3. The 'bit' file after compilation is programmed over FPGA for required performance (http://www.edu.org/comp.lang.vhdl/). The VHDL architecture of the above described mapping required major four processes which will monitor the overall handshake that is conversion of data over ADC, reading ADC, writing ADC data to DAC, selection of DAC channel.

EMBEDDED SYSTEM IMPLEMENTATION FOR DSP USING ALTERA DSP GENERATOR

The Altera EDA interface of the automatic code generator for DSP application is shown in Figure 5. ALTERA DSP generator is more users friendly, as the input and output port ADC/ DAC are available in the block form compared to XILINX HDL code. The audio signal is given to 12- bit ADC ADS55001, digitized data is then passed through the different audio synthesis blocks like Echo, Revberation, Flaging etc. This effect are selected based on Logic at DIP switches. The synthesized output is then passed through 14-bit D/A converter DAC904. The analog output of a D/A converter is connected to speakers to hear desired audio effect.

This parallelism can be extended to complex form, to other effect like chorus as shown in the Figure 6.

AUDIO SYNTHESIS CASE STUDY

The system has been demonstrated to process the audio signal. Various effects like echo, chorus, reverberation, flanging, fading and equalization can be implemented. Echo effect (If the delay is more than 50 - 70 ms, it is perceptible to human ear as an echo.) was tested for satisfactory performance and the effect of change in 50 - 70 ms delay over superimposed ensemble of signal was studied.

Echo effect

Echo is the repetition of a sound by reflection of sound waves from a surface. It arises in communication systems, when signals encounter a mismatch in impedance (Messerschmitt, 1984). The same has been modeled using Simulink Signal generator as shown in Figure 6.

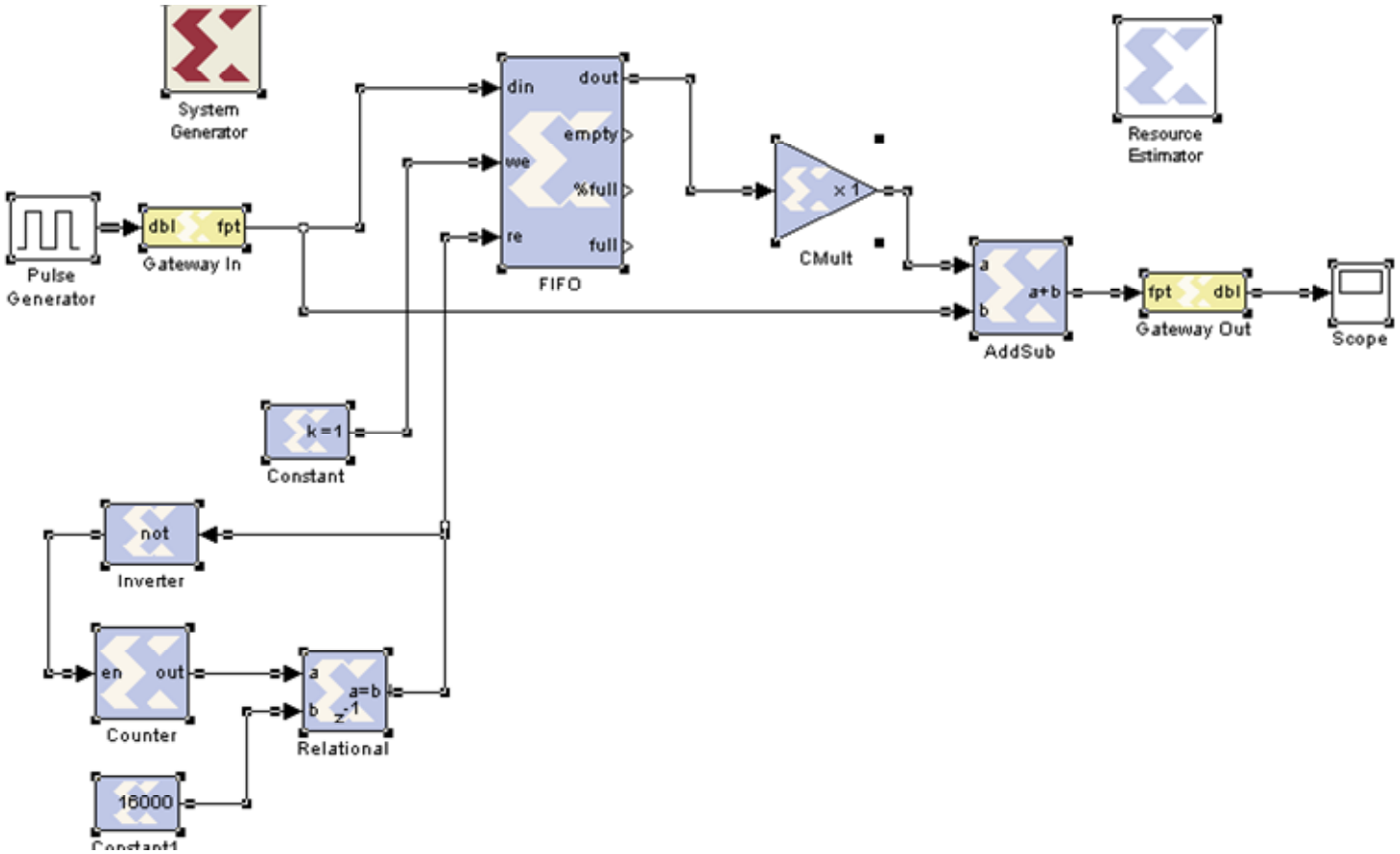

Figure 6. Block diagram of the echo in the DSP generator of Xilinx IDE.

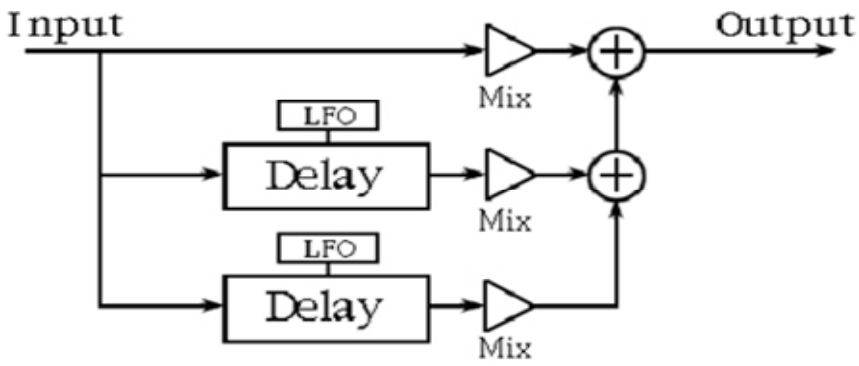

Figure 7. Block diagram of the chorus effect.

Figure 6. DELAY, FIFO, ADDER, SUBTRATOR etc are the basic building block for a number of effects in audio signal processing. The running data signal is captured and stored in the FIFO block. Total 2k data samples are captured and stored at any given time. Past signals stored are fed one at a time (after required delay of 50 - 70 ms) to adder block along with running signal. The effects can be understood by varying the 'delay' and 'gain' of the system described.

Chorus effect

More complex example illustrating parallelism is the chorus effect, shown in Figure 7. The chorus effect is often used to alter the sound of an instrument to make it

Table 1. Altera cyclone II resources.

Resource	Available	Used
Logic elements	68416	2374 (3%)
Registers	1183	0
Pins	422	35 (1%)
Total memory bits	1,152,000	5,780 (1%)
Embedded multiplier 9-bit elements	300	1
PLL	4	0

Table 2. Xilinx Spartan III resources.

Resource	Available	Used
System gate	200 k	25 K
Logic cells	4320	3135
Dedicated multipliers	12	4
User I/O	173	35
Block RAM	216K	30K

sound as if multiple instruments are playing. If the instrument where a human voice, then this effect would tend to make the single voice sound like a choir. We perceive the multiple voices or instruments, since there is always imprecise synchronization and slight pitch variation when multiple voices or instruments are playing at the same time. These are the principal characteristics of a chorus effect.

Similar way can be used in implementing the parallel computing for complex block-like design (shown in Figure 7) for 32 or more channels. Such implementation can have potential application for the Audio analysis and synthesis Tomography, ECG, EEG etc.

RESOURCES USED BY ALTERA AND XILINX PLATFORM TO IMPLEMENT AUDIO SYNTHESIS ALGORITHMS

The compared resources used by Altera and Xilinx platform to implement audio synthesis algorithms are given in Table 1 and 2 respectively. After analyzing the compilation report of both platforms, it was found that Altera platform is more versatile and their designs are more optimized than the Xilinx.

DISCUSSION AND CONCLUSION

The system designed can be upgraded to incorporate Soft core processor like NIOS II (Altera), Micoblaze, Picoblaze (Xilinx) for improving the flexibility of the system. The real time operating system (RTOS) component can also be incorporate to involve the multitasking of the process for real time performance. We have tested audio effects on both Altera and Xilinx platform. We have concluded that implementing the audio synthesis algorithms like Echo, Reverberation, and Flanging etc. on Altera Platform is easier, user friendly and also provides design flexibility and less compilation and development time than implementing the same on Xilinx Platform.

REFERENCES

Alles HG (1980). Music synthesis using real time digital techniques. Proc. IEEE, 68: 436-499,

Benveniste A, Le Guernic (1990). P Hybrid Dynamical Systems Theory and the SIGNAL Language," IEEE Tr. Automatic Control, 35 (5): 525-546.

Halbwachs N, Caspi P, Raymond P, Pilaud D (1991). The Synchronous Data Flow Programming Language LUS-TRE,"Proc. IEEE, 79(9):1305-1319.

Hilfinger P (1985). A High-Level Language and Silicon Compiler for Digital Signal Processing", Proceedings of the Custom Integrated Circuits Conference, IEEE Computer Society Press, Los Alamitos, CA, pp. 213-216 .

http://altera.com/technology/dsp/dsp-index.jsp.

http://www.andraka.com/dsp.htm.

http://www.edu.org/comp.lang.vhdl/.

http://xilinx.com/products/design_resources/dsp_central/grouping/index.html

Messerschmitt DG (1984).Echo cancellation in speech and data transmission' IEEE, J. on selected areas in communication, SAC-2:283-297, March [6]

Poulton K, Corcoran JJ, Horna T (1987). A 1-GHz 6-bit ADC system', IEEE J. Solid-State Circuits, SC-22 : 962-970.

Rajanish KK, Santosh SA, Vinod SG (2009). Unleash the System On Chip using FPGAs and Handel C", Springer, , XXIV, Hardcover. 176p.

Richard GL (2004). Understanding Digital Signal Processing', Prentice Hall, ISBN 0131089897. pp.121-125.

The Ptolemy Project http://ptolemy.eecs.berkeley.edu/.