REALTIME IMPLEMENTATION OF DSP BASED AUDIO EQUALIZER

[1]Dr. Paresh M Dholakia, [2]Dr. Alpesh S Adeshara

[1]Associate Professor, [2]Associate Professor
[1] Department of Electronics and Communication Engineering, [2] Department of Electrical Engineering
[1,2] VVP Engineering College , Rajkot , India

Abstract : In many fields of sound processing like sound recording studios, or sound transmission to the listener during a concert, equalizers are applied for sound control and enhancement. Although initially developed for sound recording studios, equalizers have rapidly gained acceptance and can actually be found in almost every consumer product like car radios, hifi-amplifiers, or MP3 players. Their market increases every year worldwide. Audio equalizers are divided into parametric and graphic equalizers, latter are further subdivided into one octave and 1/3-Octave equalizer. In this paper, we present a real-time implementation of a Matlab/Simulink model of audio equalize using FDA tools and Simulink.. We will design our audio equalizer based on frequency sampling of a desired magnitude response and under linear phase constraints. This project was developed in order to deliver an insight on how audio equalization can easily be implemented with Matlab/Simulink for educational purposes. We will finally explain how a real-time audio equalizer based on a Simulink model can be implemented using the TMS320C6713 DSP board.

Index Terms **- TMS320C6713** DSP, FDA tools, and MP3 players

I. INTRODUCTION

Audio equalization is a technique which consists of boosting or cutting certain frequency components of a given signal for sound quality enhancement. Audio equalizers can be classified into two main groups: graphic and parametric equalizers. While parametric equalizers have their name from the fact that **parameters** of the filters are fully adjustable, graphic equalizers get their name from the fact that the relative positions of the sliders build a **graphic picture** of the desired magnitude response [1].

Parametric equalizers are standard low-/high-pass filters, low-/high-frequency shelving filters and peak filters. Shelving filters are not actually removing a portion of spectrum like standard low-/high-pass filters do. They only boost or cut parts of the audio spectrum and leave the rest unchanged [2]. Parametric equalizers can be controlled by changing several specific parameters, such as cutoff frequency, center or middle frequency, gain factor, bandwidth or Q-factor (see Fig. 1). The Quality Factor (Q-factor) of a filter is a synonym for bandwidth. It is defined to be the center frequency divided by the bandwidth in Hertz. Although offering certain flexibility, parametric equalizers remain complex, since three parameters per band must be controlled.

For graphic equalizers we only control with sliders the boost or cut by keeping the center or middle frequency unchanged (see Fig. 2). A graphic equalizer is a set of band-pass filters each with fixed center or middle frequency and a certain bandwidth. Where the first and last bands are respectively low-/highpass filter. Given **ISO** prescribed center frequencies and their respective bandwidth, the complexity previously observed by parametric equalizers is totally reduced. We only now control every band with one parameter (gain factor). Two important classes of graphic equalizers are the octave and 1/3-octave equalizers

In this paper, we will focus on a audio equalizer design, which gives a better frequency resolution [3]. Our model will cover a frequency range from 100 Hz up to 9 kHz and a boost/cut within the range of ± 20 dB. The remainder of this paper is organized as follows. In Section 2, we will introduce the design of a linear-phase audio equalizer. In Section 3, we will explain our Simulink model of a fast convolution based algorithm and demonstrate the possibility to run the Simulink model in real-time using a TMS320C6713 DSP board.

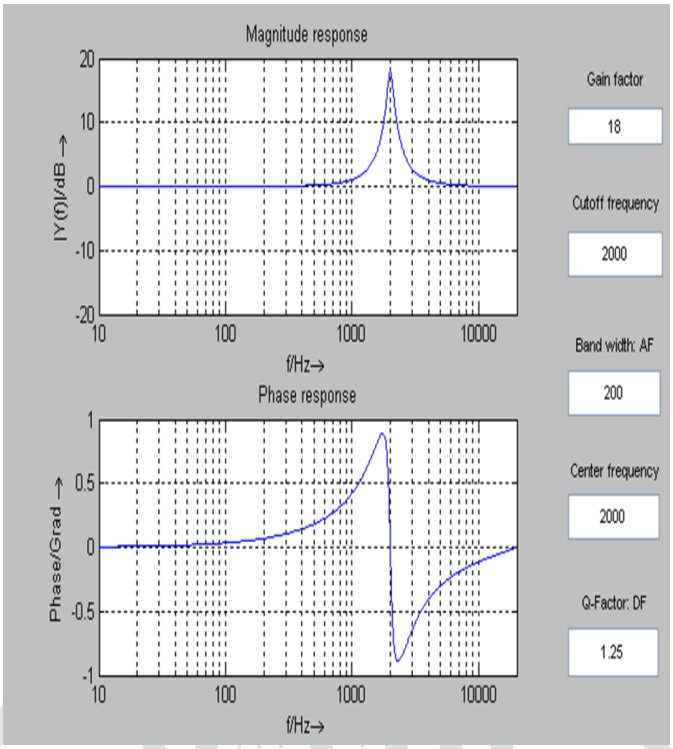

Figure 1: Parametric equalization: Peak filter boost

II. AUDIO EQUALIZER DESIGN:

First a desired frequency response is specified by positioning sliders in the model, where the relative position of a slider corresponds to the magnitude of the related 1/3-octave band. We then perform non-recursive filter design by fast convolution using a frequency sampling method in order to derive our impulse response [4]. The spectrum of a linear phase system can be obtained as explained in [4], and given by:

$$H(e^{j\Omega}) = |H(e^{j\Omega})| \cdot e^{-j\frac{N_F-1}{2}\Omega}, \quad \Omega = \frac{2\pi k}{N_F}, \tag{1}$$

where $k = 0, 1, ...,NF - 1$, with NF as the length of The impulse response and $|H(e^{j\Omega})|$ corresponding to our desired magnitude response. The sampling in the frequency domain is done at equidistant steps.

Our design model fulfills conditions for filter type II: real-valued impulse response with even length and even symmetry. Therefore the real part of the Spectrum will have even symmetry, while the imaginary part of the spectrum will have odd symmetry [4]. We will derive the real and imaginary part of the spectrum by decomposing the linear phase as:

$$e^{-j\frac{N_F-1}{2}\Omega} = \cos(\frac{2\pi k}{N_F}\frac{N_F-1}{2}) - j\sin(\frac{2\pi k}{N_F}\frac{N_F-1}{2}) \tag{2}$$

with $k = 0, 1, ..., N_F/2 - 1$. Since our transfer function H(z) is real and of even length, the following condition should be fulfilled:

$$H(k = \frac{N_F}{2}) = 0 \tag{3}$$

We derive our impulse response h(n) through an NF -point IDFT of H(k), then perform zero-padding of h(n) up to the length of the signal block to be filtered and finally transform the new h(n) by an N-point DFT to obtain H(k) of the filter as shown in [4][5].

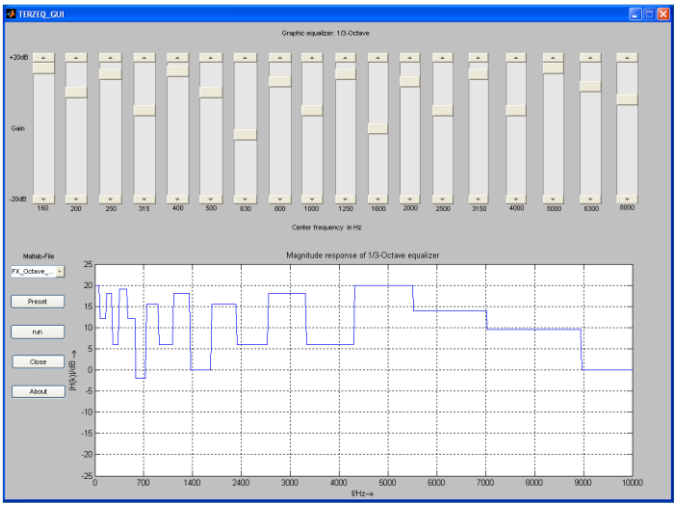

Figure 2: Graphic equalization.

III. AUDIO EQUALIZER SIMULINK MODEL:

Our first model is implemented with a Matlab filter design and analysis (FDA) tool, where we are using to control boost and cut cases of the desired frequency response. A Simulink implementation of the filter design by FDA tools shown in Fig. 3. First we generate the desired frequency response in every band by using different filters. We then concatenate the contribution of every band to derive the magnitude response. Regarding (Eq. (2)), we generate two arrays of length $N_F/2$ for the real and imaginary part of the spectrum. Since the spectrum has to be periodical after $k = N_F/2$, we therefore build the second part of the spectrum using the symmetry properties. We finally derive our impulse response using an IFFT block.

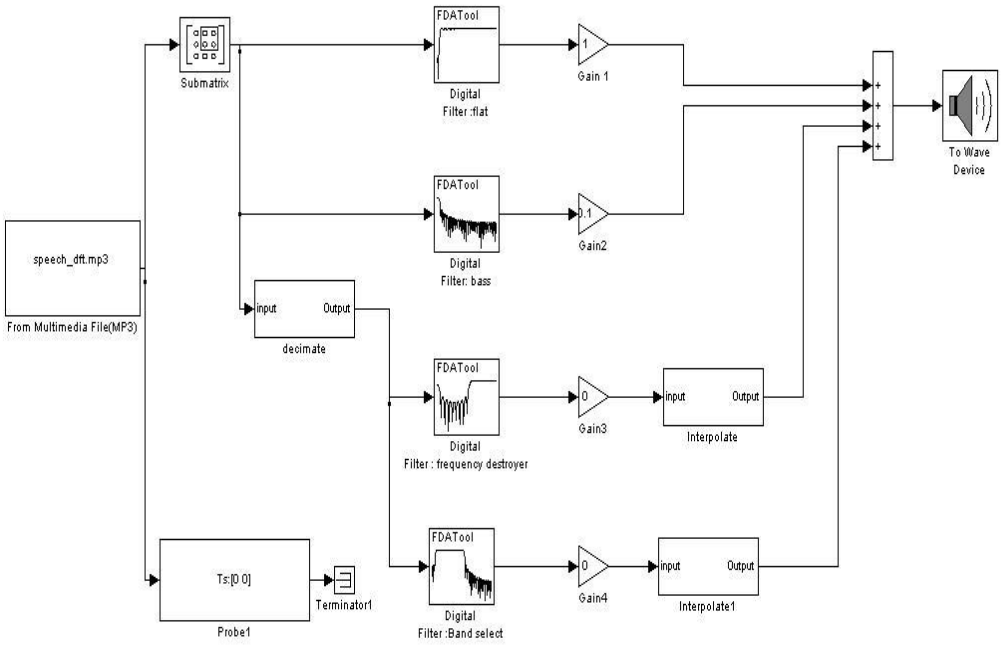

Figure 3: Audio equalization using FDA tool and Simulink

Specification of the sample time is very important for a good communication between Simulink blocks. Where sample time propagation is not possible, sample time conflicts can be avoided by inheriting sample time from the fastest sample time available in the model or the sample time of the first input.

To prepare our Simulink model for a real-time implementation, we slightly modify the model with new blocks in order to communicate with the TMS320C6713 DSP board. We first include to the Simulink model a 6713DSK block from the C6000 target preferences group in the embedded target for TI C6000 DSP folder. We then replace our input signal generator and the signal

output from the previous model by C6713 DSK ADC and C6416 DSK DAC blocks from the library browser in the C6713 DSK Board Support group. The model requires a software version of 6713 DSK Code Composer Studio (CCS) to be installed on the PC.

When data type conflicts between Simulink blocks occur, we solve the conflicts by inserting data type conversion blocks with the desired specification. To run the model in real-time, we first ensure our PC is connected to the TMS320C6713 DSP board via a USB-Input and then start the C64xx Code composer studio for C codes generation. To start the implementation on the DSP board, we first need to click on tools in the Simulink editor menu and choose build model from real-time workshop submenu. The C64xx Code composer studio will then generate the C-code for the model. In the Matlab command window, the C code generation and the debugging procedure from the CCS can be viewed. To listen to the result of equalization, we connect the Line In of the DSP board with the PC-headphone output. From the headphone output of the DSP board, we connect our external headphones.

IV. MAGNITUDE RESPONSES OF DIFFERENT FILTERS USED FOR AUDIO EQUALIZER

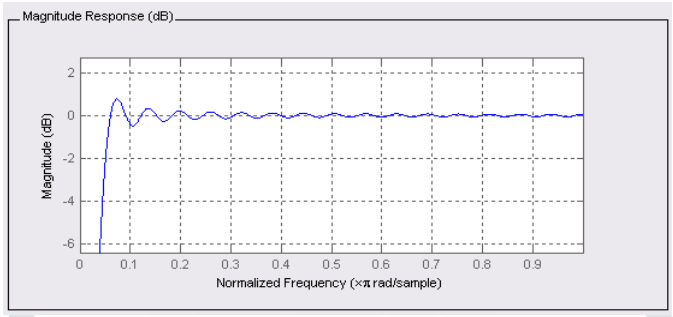

Figure 4 : Magnitude Response of 1st filter (Flat)

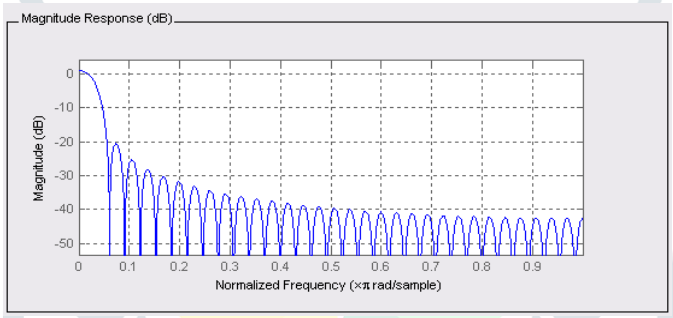

Figure 5 : Magnitude Response of 2nd filter (Bass Boosting)

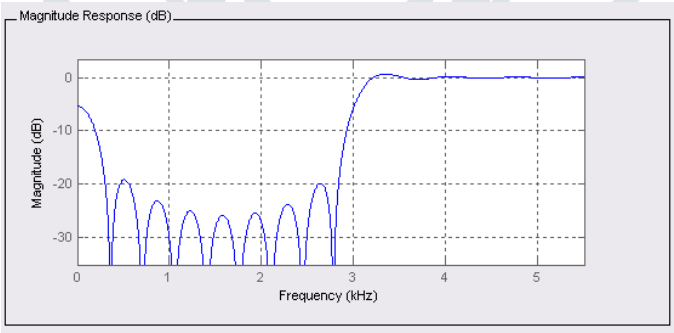

Figure 6 : Magnitude Response of 3rd filter (Frequency Destroyer).

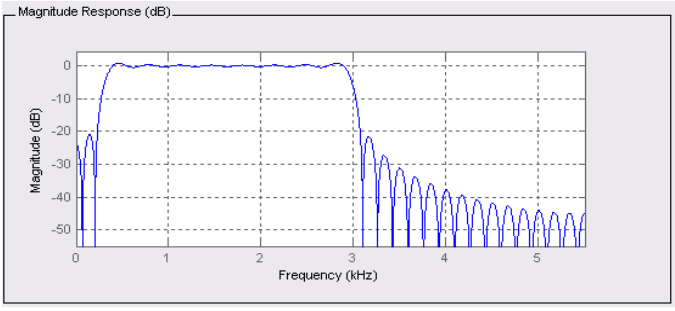

Figure 7: Magnitude Response of 4th filter (Band Select).

V. CONCLUSIONS :

In this contribution we show how audio equalizers based on FDA tools can easily be implemented with Matlab/Simulink for educational purposes. We have explained the design and the implementation of a audio equalizer for a real-time evaluation with a TI DSP board. Since Simulink is an intuitive environment to quickly test algorithms, we hope that with these example novices in this field will be encouraged to quickly start their own implementations. Our objective is also to give to students in the Digital Signal Processing (DSP) lecture a sense of fun for modeling and simulating, through an environment that encourages them to ask questions, perform model debugging, and test their implementations by listening to the audio results. In future works, we will implement with the same tools specific parametric equalizers like shelving and peak filters.

REFERENCES:

[1] Dennis A. Bohn. Constant-q graphic equalizers. *Audio Eng Soc.,Vol.34,no.9,* 1986.

[2] Udo Zoelzer (Ed). *DAFX:Digital Audio Effects*. J. Wiley & Sons, Chichester, 2002.

[3] Udo Zoelzer. *Digitale Audiosignalverabeitung*. B.G. Teubner, 2005.

[4] Udo Zoelzer. *Digital Audio Signal Processing*. J. Wiley & Sons, Chichester, 1997.

[5] Recursive and Non-reeursive Realizations of Digital Filters Designed by Frequency SamplingTechniques : *IEEE TRANSACTIONS ON AUDIO AND ELECTROACOUSTICS VOL. Au-19, NO. 3 SEPTEMBER 1971LAWRENCE R. RABINER, Member, IEEE RONALD W. SCHAFER, Member, IEEE Bell Telephone Laboratories, Inc. Murray Hill, N. J. 07974*