Sourindu Chatterjee

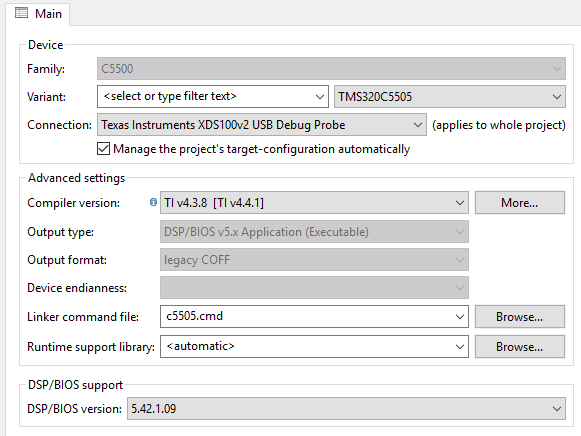
Real Time DSP

Graduate Project:

Parametric Equalizer

**BUILD**:

The bulk of the project was built by importing the project files from the Kuo textbook code, Exp 10.5. The initial target configuration based on the book code is as shown:



In order to add pushbutton and OLED functionality source files and libraries were incorporated to the overall book code design.

**OVERVIEW**:

The goal of this project was to use the C5515 eZdsp to implement a real time software based parametric equalizer using fixed point C code.

In general an equalizer is a circuit or software based filtering system that allows for signal equalization, where the user can dynamically control certain output characteristics of signals. In particular a parametric equalizer is an equalizer that has three primary “parameters” for controlling signal output. These parameters include the Q factor (i.e. the bandwidth), the center frequency, and the gain. The controls are also often split up it to low, middle, and high frequency bands. Using these variables as inputs to the control system the desired output can be achieved through several methods including classes of filters such as shelving, peak, and notch filters.

Although the book code uses second order shelving and peak filters as the second order sections of a larger overall system filter, in does not, strictly speaking, implement a parametric equalizer system. The element of variable control is essential.

This project therefore attempts to add control capabilities to the book code.

**DESIGN:**

There are two sides to the design approach in this project. There is control side and the filter design side.

Normally to implement a parametric equalizer in a digital system, the filter coefficients would be calculated at run time based on variable levels of the three mentioned parameters. However for real time processing, a fixed point implementation on this system would be complicated and take too long to run efficiently. So this design attempts to simulate this process by having predefined filter coefficients stored in header file lookup tables.

A pushbutton system of nested filter parameter menus using the pushbuttons to both cycle through and select values to determine the appropriate filter was desired. However, de-bounce timing issues made using a linear menu more practical.

In terms of specific parameter values used to determine filters, there are three filtering techniques used: low shelf filtering, peak filtering, and high shelf filtering. The center frequencies are: 20Hz, 100Hz, 500Hz, 2500Hz, 12500Hz. There are also 7 gain stages ranging from +9 dB to -9dB in steps of 3dB.

Some of the base filter designs are as shown:



Figure 1. High Shelf Filters



Figure 2. Peak Filters



Figure 3. Low Shelf Filter

The Filters are implemented using 5 center frequencies to generate a low and high shelving filter as well as a peak/notch filters. Each of these second order IIR filters are combined into a single filter as second order sections.

MATLAB is used to generate the filter coefficients for each of the second order sections using the functions PeakFilter() and ShelfFilter() Given in the Kuo text.

The filter arrays created by MATLAB are stored in header files with the CCS project.

freqs=[20,100,500,2500,12500];

fs=48000;

qfactor=1;

type='high';

gain=0;

ParamEQFilters(freqs,gain,qfactor,fs,type)

This yields:

/\*--------------------------------------------------------------------------------------------------------------------------------

0dB HIGH

--------------------------------------------------------------------------------------------------------------------------------\*/

Int16 high\_20Hz\_0dB[SECTIONS\*5]=

{

//Low Shelf Filter

(Int16)(-1.997378586642692\*UNIT-RDA),(Int16)(0.997385431570743\*UNIT+RDA),

(Int16)(0.997385431570743\*UNIT/SCALE+RDB),(Int16)(1.000000000000000\*UNIT/SCALE+RDB),(Int16)(-1.997378586642692\*UNIT/SCALE-RDB),

//Peak Filter

(Int16)(-1.994770854193204\*UNIT-RDA),(Int16)(0.994777690184672\*UNIT+RDA),

(Int16)(0.994777690184672\*UNIT/SCALE+RDB),(Int16)(1.000000000000000\*UNIT/SCALE+RDB),(Int16)(-1.994770854193204\*UNIT/SCALE-RDB),

//High Shelf Filter

(Int16)(-1.997378586642692\*UNIT-RDA),(Int16)(0.997385431570743\*UNIT+RDA),

(Int16)(0.997385431570743\*UNIT/SCALE+RDB),(Int16)(1.000000000000000\*UNIT/SCALE+RDB),(Int16)(-1.997378586642692\*UNIT/SCALE-RDB),

};

The main file selects the desired filter based on factors such as center frequency, gain, and quality factor ( i.e. bandwidth ). It uses switches and the OLED to convey information.

After selecting the filter, the array of filter coefficients are fed to an IIR assembly routine in order to carry out the real time filtering process.

The program as a whole reads in audio from the audio line-in and feeds the result to audio line-out via direct memory access (DMA) and memory mapping.

**RESULTS:**

Some of the methods initially attempted in this project were perhaps somewhat overly ambitious in scope. As a result neither the control menu nor the filter design function exactly as intended. The controls had to be reworked into a linear configuration. The filter design suffered from the ambiguity of some of the fixed point implementation mechanics from the original code as well as volume and noise issues due to imprecisely defined gain values. In the end it was often difficult to distinguish frequency content changes between different filter. However given the original goal of adding parametric control functionality to the book code the overall project was relatively successful