EE 451 – LAB 4

Sound Processing

In this laboratory you will implement a program to generate special audio effects using the DSK board. See Section 15.5 of Mitra for musical sound processing techniques.

Introduction

Music generated in a studio does not sound natural compared to the music performed inside a concert hall. In the concert hall, the sound waves propagate in all direction and reach the listener from various directions and at various times. The listener hears the early reflections first, followed by densely packed echoes referred to as reverberation. The amplitude of the echoes decays exponentially every time the sound waves bounce off a wall (or anything that is not soundproof). Since the reflection properties of different materials are not the same at different frequencies, the reverberation time varies.

Delay systems with adjustable delay factors are employed to artificially create the early reflections. Electronically generated reverberation combined with artificial echo reflections are usually added to the recordings made in a studio.

Echoes are generated by delay units. An infinite number of echoes spaced D sampling periods apart with exponentially decaying amplitudes can be created by a transfer function of the form

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-\alpha\, z^{-D}}, \quad |\alpha| < 1 \qquad (1$$

The impulse response of such a transfer function for α = 0.8 and D = 4 is depicted in Figure 1. The parameter D should be large enough to produce delays of, say, 50 ms to 500 ms.

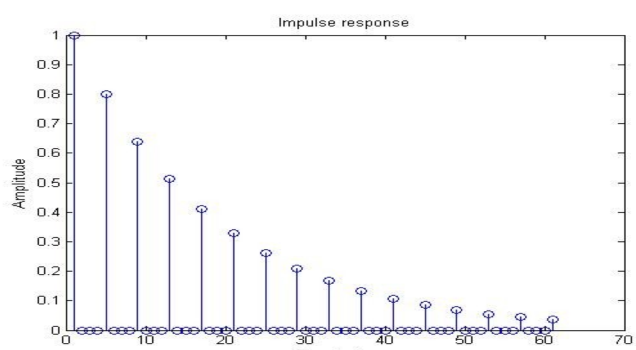

Figure 1. Impulse response of an echo generator transfer function.

The sound reaching a listener in a closed space consists of several components: direct sound, early reflections, and reverberations. The reverberation is composed of densely packed echoes. A realistic reverberation has a transfer function give by

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\alpha + z^{-D}}{1 + \alpha z^{-D}} \quad \text{(2}$$

Another interesting effect is a chorus effect, which would be generated by several instruments playing the same notes at about the same times, but with small changes in amplitude and timing. A simple chorus effect can be generated with the equation:

$$y(n) = x(n) + \alpha_1 x(n - D_1) + \alpha_2 x(n - D_2) + \alpha_3 x(n - D_3) \quad \text{(3}$$

where α_i and D_i are constant and different. The delays D_i could be randomly varied with very slow variations; in such case these would be $D_i(n)$ to account for variation in time.

The Prelab

1. Get the magnitude response of the echo generator using Equation 1 and plot the magnitude response. Assume a value of D to generate a delay of 300 ms using a sampling frequency rate of 48 kHz. What should you use for α?

2. Get the frequency response of the echo generator using Equation 2 and plot the magnitude response. What is the difference between this response and the one from Equation 1?

The Lab

1. Create a program that implements an echo effect. We will use a HWI to implement the echo: (i) read a new input sample from the codec, (ii) read a delayed value from a buffer, (iii) output the sum of input and delayed value, and (iv) store new input and fraction of delayed value. A template for this program is shown in Figure 1. Try different values of where α and D.

2. Create a program that implements a chorus effect. Experiment with different values of α_i and D_i

```
//========= Lab4.c =========
//  This program is used to generate an echo and chorus effects
//

#include "dsk6713.h"
#include "dsk6713_aic23.h"                              // codec support
#include "dsk6713config.h"

//======= Variable definition =============

// define the gain Alfa, Buffer size, the Buffer, delay D

//==========================================

Uint32 fs = DSK6713_AIC23_FREQ_48KHZ;                  // set sampling rate
#define DSK6713_AIC23_INPUT_MIC 0x0015
#define DSK6713_AIC23_INPUT_LINE 0x0011
Uint16 inputsource=DSK6713_AIC23_INPUT_MIC;            // select LINE IN input

void generate_echo(void)
{

// Get a sample x(n), generate the output y(n), update buffer and
// time index n

  return;

}

void main()
{

  // Set up needed to for interrupts
  IRQ_globalDisable();                                 //disable interrupts
  DSK6713_init();                                      // call BSL to init DSK-EMIF,PLL)
  hAIC23_handle=DSK6713_AIC23_openCodec(0, &config);// handle(pointer) to codec
  DSK6713_AIC23_setFreq(hAIC23_handle, fs);            // set sample rate
  DSK6713_AIC23_rset(hAIC23_handle, 0x0004, inputsource);  // choose mic or line in
  MCBSP_config(DSK6713_AIC23_DATAHANDLE,&AIC23CfgData);// interface 32 bits to AIC23
  MCBSP_start(DSK6713_AIC23_DATAHANDLE, MCBSP_XMIT_START | MCBSP_RCV_START |
  MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC, 220);      // start data channel
  CODECEventId=MCBSP_getXmtEventId(DSK6713_AIC23_codecdatahandle);//McBSP1 Xmit
  IRQ_map(CODECEventId, 11);                           //map McBSP1 Xmit to INT11
  IRQ_reset(CODECEventId);                             //reset codec INT 11
  IRQ_globalEnable();                                  //globally enable interrupts
  IRQ_nmiEnable();                                     //enable NMI interrupt
  IRQ_enable(CODECEventId);                            //enable CODEC eventXmit INT11
  MCBSP_write(DSK6713_AIC23_DATAHANDLE,0);             //start McBSP interrupt outputting a sample
  for(n=0; n<Buf_size; n++) buffer[n]=0;               //initiallize buffer for the echo

}
```

Figure 1. Program template for Lab4.