EE 443 Design and Application of Digital Signal Processors
Homework Assignment #2, Spring 2016

Report Due: April 14, 2016

Demo: afternoon of April 14, 2016

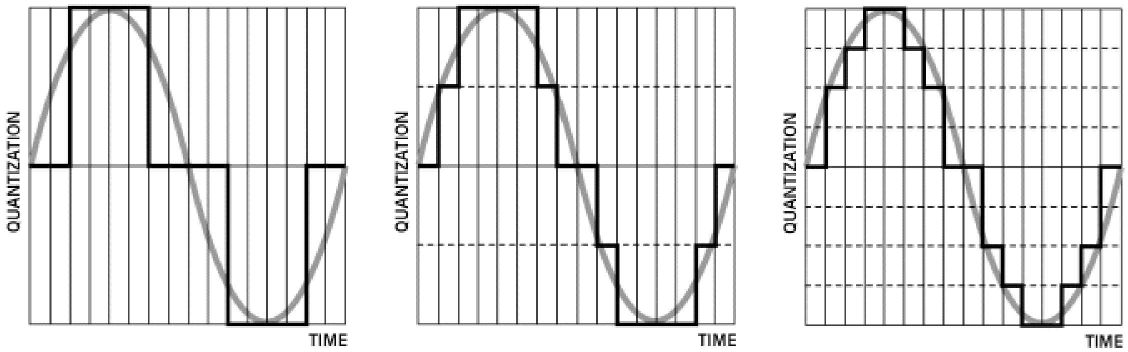
Home Assignment Grading Policy
- Assignments are graded on a group basis.
- Grades are based on both written reports and demos.
- Demos have to be presented to TA on the signed up slot. No late demo

Submission Instructions
- Use Windows Snip Tool to capture the image from Windows
- The report cover page should have: homework number, student name, and student number.

Problem 1:

Write a C program to quantize the MIC input (positive) unsigned integer signal of your own speech sampled at 8KHz based on the following quantization table (you can pre-define the maximum and minimum inputs empirically). The numbers in the table are the new values to be replaced on each sample for each level. For example, if we choose the 3-levels, then output can be 1*(Maximum Input), 0, or -1*(Maximum Input). Send the quantized speech values to the left headphone out and compare it with the original input signal, which is sent to right headphone. Try all 3-different quantization levels.

<table>
<tr><th>Quantization levels</th><th>3-levels</th><th>5-levels</th><th>7-levels</th></tr>
<tr><td rowspan="2">Max Input</td><td></td><td></td><td></td></tr>
<tr><td rowspan="4">1</td><td rowspan="2">1</td><td rowspan="2">1</td></tr>
<tr><td rowspan="2">(7*Max+Min)/8</td></tr>
<tr><td rowspan="4">0.5</td><td rowspan="2">0.75</td></tr>
<tr><td rowspan="2">(6*Max+2*Min)/8</td></tr>
<tr><td rowspan="8">0</td><td rowspan="2">0.5</td></tr>
<tr><td rowspan="2">(5*Max+3*Min)/8</td></tr>
<tr><td rowspan="4">0</td><td rowspan="4">0</td></tr>
<tr><td rowspan="2">(4*Max+4*Min)/8</td></tr>
<tr></tr>
<tr><td rowspan="2">(3*Max+5*Min)/8</td></tr>
<tr><td rowspan="4">-0.5</td><td rowspan="2">-0.5</td></tr>
<tr><td rowspan="2">(2*Max+6*Min)/8</td></tr>
<tr><td rowspan="4">-1</td><td rowspan="2">-0.75</td></tr>
<tr><td rowspan="2">(Max+7*Min)/8</td></tr>
<tr><td rowspan="2">-1</td><td rowspan="2">-1</td></tr>
<tr><td rowspan="2">Min Input</td></tr>
<tr><td></td><td></td><td></td></tr>
</table>



Tips :

1. To receive MIC input, change the array aic23_demo[] ,

```
aic23_demo[4] = 0x0014; //MIC input
```

2. Change the unsigned integer input to the signed integer before doing the quantization as,

```
leftChannel = unsigned2signed(IORD_ALTERA_AVALON_PIO_DATA(LEFTDATA_BASE));
```

Problem 2 :

Write a C program that reads MIC input and immediately outputs signal sampled at 8 KHz. Utilize the DIP switches to generate the echo effect. If SW_0 is switched OFF, it outputs the original sound, but if SW_0 is switched ON, enable channel inputs to be combined with delayed data (with gain of 0.6). The delay should be variable using keys to increase or decrease the delay. One example could be to use one KEY as an increase and another to decrease the delay amount. You should also have a method of changing the amount of the delay by 0 to maximum of 1000 samples with increments of 50 samples. This delay will be used to create an echo effect. Send the echo speech to both channels of the headphone.

Problem 3:

Write a C-callable assembly program to real-time sample your own speech input data using AIC with sampling frequency 8 KHz. For every 128 samples (a frame) captured, use the C program to call the assembly to flip the sampled data and output them in a reversed order for this frame. Use UART to send three frames of the original and reversed speech to Matlab for graphing.

Tips :

1. In yourISR.h, you may need two different buffers for capturing the samples and processing the data respectively, where `flip(data2, flip_data, 128)` is the assembly code function.

```
int data1[128], data2[128];
int flip_data[128];

static void handle_leftready_interrupt_test(void* context, alt_u32 id) {
volatile int* leftreadyptr = (volatile int *)context;
*leftreadyptr = IORD_ALTERA_AVALON_PIO_EDGE_CAP(LEFTREADY_BASE);
IOWR_ALTERA_AVALON_PIO_EDGE_CAP(LEFTREADY_BASE, 0);
/*******Read, playback, store data*******/
leftChannel = unsigned2signed(IORD_ALTERA_AVALON_PIO_DATA(LEFTDATA_BASE));

if(leftCount<128){

data1[leftCount] = leftChannel;
if(leftCount==0) flip(data2, flip_data, 128);
IOWR_ALTERA_AVALON_PIO_DATA(LEFTSENDDATA_BASE, flip_data[leftCount]);
}
else{
data2[leftCount%128] = leftChannel;
if(leftCount==128) flip(data1, flip_data, 128);
IOWR_ALTERA_AVALON_PIO_DATA(LEFTSENDDATA_BASE, flip_data[leftCount%128]);
}
leftCount = (leftCount+1)%256;

}
```

2. .s file must be in your project folder.

Problem 4:

Based on the SoP (MAC) assembly code given in Lecture note 3, please write a C program with reference to page 9 of the Lecture note 4 to call the SoP assembly code to implement an IIR filtering of your input speech (note that there are two SoPs in the finite difference equation of an IIR filtering. The IIR filter is a simple lowpass filter with coefficients given below. The input signal is an input mono channel music sampled at 8 KHz. Send the lowpass result to headphone.

$$H(z) = \frac{\sum_{m=0}^{M-1} b_m z^{-m}}{1+\sum_{l=1}^{N-1} a_l z^{-l}} = 4.29\times10^{-3}\frac{1+z^{-1}}{1-0.801z^{-1}}\frac{1+2z^{-1}+z^{-2}}{1-1.638z^{-1}+0.81z^{-2}}$$

$$y(n) = -\sum_{l=1}^{N-1} a_l y(n-l) + \sum_{m=0}^{M-1} b_m x(n-m)$$

Problem 5:

Write a C program to read in, with 32KHz sampling rate, and immediately output stereo music input (from Youtude or any kind of music source). Also program in the ability to create a stereo mix output with the DIP switches. To create a "stereo mix", simply **subtract** the left channel input from the right channel input and output to the **right channel**, then **add** the left channel input to the right channel to create the **left** channel output. Define the following functionality for the DIP switches:

1) When no DIP switch is pressed down, simply output the stereo signals with no processing.

2) If DIP #1 is down, generate a "stereo" mix and output to left and right channels