Lab 3 –Tone Generator

1 Objectives

1. Create a module with a parameter to generate tones on the speaker.

2 Introduction

In this lab you will create a simple digital tone generator that plays the notes of the C major scale as the rotary encoder is turned.

3 Prelab exercises

You must design and implement two modules. A tonegen module that generates an output at a specified frequency, and an enc2freq module that maps the encoder output to a set of frequencies in order to play a scale. The tonegen module should have the following ports:

```
module tonegen
  #( parameter FCLK )              // clock frequency, Hz
   ( input logic [31:0] freq,      // frequency to output on speaker
     input logic onOff,            // 1 -> generate output, 0-> no output
     output logic spkr,            // speaker output
     input logic reset_n, clk); // reset and clock
```

The FCLK parameter will be used to indicate the input clock frequency (in this case it will be 50 MHz, but the module should work for any frequency).

To keep the circuitry small, we would like to avoid any multiplication or division operations when generating a frequency. One simple way to generate the desired frequency is to keep a count which increments by twice the desired frequency each clock cycle until it reaches the FCLK value. Once the count reaches the FCLK value, the output toggles and the count starts at 0 again. For example, if we have a 50 MHz clock (FCLK = 50,000,000) and we desire a 1 kHz output frequency, we need to toggle the clock every $\frac{50\ MHz}{1\ kHz} \cdot \frac{1}{2} = 25{,}000\ cycles$ (note the multiplication by ½ is necessary as the spkr output toggles twice per cycle). If the count is incremented from 0 by 2·1000 each clock cycle until the count of 50,000,000 is reached, that would be 25,000 clock cycles, hence generating a 1 kHz output. Note that the multiplication by 2 can be accomplished using a simple left shift.

Simulate your tonegen module using the simple testbench provided and observe the waveforms to ensure that your module works as expected.

Write the implementation of the enc2freq module as follows:

```
module enc2freq
   ( input logic cw, ccw,          // outputs from lab 2 encoder module
     output logic [31:0] freq,     // desired frequency
     input logic reset_n, clk); // reset and clock
```

The module should cycle through the notes in the C Major scale (262 Hz, 295 Hz, 328 Hz, 349 Hz, 393 Hz, 437 Hz, 491 Hz, 524 Hz), increasing the frequency every four pulses of cw, and decreasing the frequency every four pulses of ccw. Create a self-checking testbench to simulate and automatically verify (ie. check outputs and report Pass/Fail status) the functionality of your enc2freq module.

4 Lab Activities

Create a lab3 module based on your lab2 module that adds the `enc2freq` and `tonegen` modules. Note that the clock input to the `tonegen` module should be the 50 MHz input called `CLOCK_50`. You will need to include one additional output to the lab3 module called `spkr` which is a signal connected to the speaker on the BoosterPack, and two additional inputs call `s1` and `s2` which are connected to S1 and S2 active low pushbuttons.

Display the hexadecimal value of the lower 16 bits of `freq` on the 4-digit 7-segment display. Connect the `s1` input from the BoosterPack to the `reset_n` inputs of the `enc2freq` and `tonegen` modules, and the `s2` input to the `onOff` input of the `tonegen` module as a mute button.