ECE 4514 Fall 2019: Homework 3

Assignment posted on February 8
Assignment due on February 16 11:59 AM

Homework weight: 100 points

- Homework 3 is a board-level assignment on Finite State Machine Design. Refer to Homework 2, the I2C standard specification, the DE1SoC schematics/ user manual, the WM 8732 datasheet.

Tone Generation for the Audio Codec on the DE1-SoC board

In this homework, you have to design a digital interface for the audio CODEC on your DE1SoC kit. The interface consists of two components:

- An I2C master interface, which will be used to program settings into the audio CODEC.
- An Audio interface, which will be used to send digital audio to the audio CODEC.

The interface you will design in the FPGA has three modes. Each mode is activated by pressing one of the buttons on the DE1SoC:

- KEY[1] enables the 'loopback' mode. In the loopback, the audio CODEC feeds the line-in directly to the line-out.
- KEY[2] enables the '1 KHz square wave' mode. In that mode, the FPGA will send a digital stereo audio stream of 16-bit samples and approx. 48KHz sample rate to the audio CODEC.
- KEY[3] enables the 'silence' mode. In the silence mode, the audio CODEC is powered off, and no audio appears on the line-out output.

This homework requires you to build familiarity with the following two components:

- The Wolfson WM8273 CODEC: Programming of the chip, digital audio format
- The DE1-SoC board: Connectivity of the FPGA to the CODEC and the audio connectors.

You are encouraged to use your solution for homework 2 to complete this homework. A sample homework 2 solution will be made available after the submission deadline for homework 2.

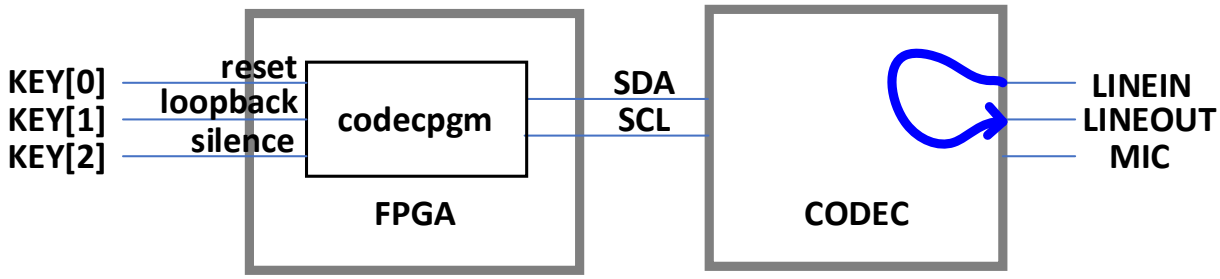
Step 1: Read the Wolfson CODEC datasheet

Unfortunately, there is no quick way around it without breaking the honor code. If you want to send sound to the CODEC, you will have to program it. And to do that successfully, you will need to understand how that chip works. Read the Wolfson CODEC datasheet, paying special attention to 'DEVICE OPERATION'. Answer the following questions:

- How do you set the chip in 'bypass' mode using I2C programming?
- How do you power-down and power-up the chip using I2C programming?
- What is the meaning of each of the following audio interface signals
    - DACDAT
    - DACLRC
    - BCLK
    - ADCLRC
    - ADCDAT
- How do you select the following audio settings using I2C programming
    - master-mode (Figure 31)
    - left-justified audio data (Figure 26)
    - 48 KHz sample rate (Table 18)
    - 16-bit sample precision

Step 2: Programming silence and loopback mode

We will do this homework in two steps. The first is to program an application called 'loopback'. It is the same as the complete homework three without the 1KHz mode. You only need a working I2C master interface for it.



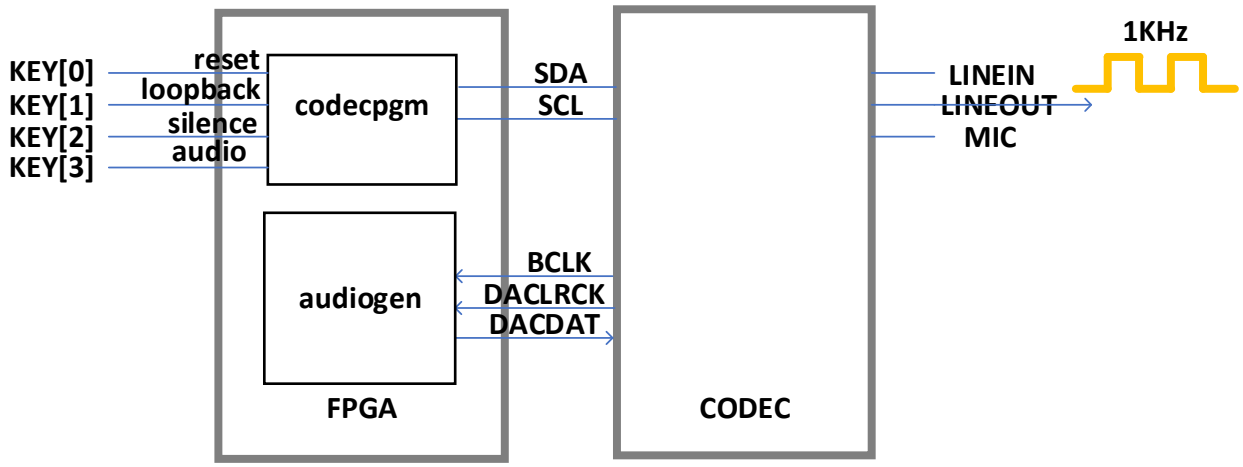
You are designing a single module, codecpgm (that could include multiple other modules), that generates a programming sequence for the CODEC. Pressing 'silence' will silence the CODE. Pressing 'loopback' will put the CODEC in bypass mode.

You can test this by feeding an audio signal into the LINEIN of the DE1SoC board, using a cable with a 3.5 mm plug. The other side goes into an audio source, such as the headphone output of your smartphone or your laptop.

Important: You will need to generate a system clock and feed it to the audio codec through the AUD_XCK output. According to the data sheet, the recommended clock frequency is 12MHz. To keep things simple, you may use ¼ of the 50MHz system clock for this, or 12.5 MHz.

Step 3: Programming a 1KHz square wave

The second step is to generate a digital audio stream with a square wave audio signal. Note that you cannot simply generate a digital signal; you have to generate a sampled-data version of it, and feed this through the CODEC audio interface (BCLK, DACLRCK, DACDAT). The operation of this interface is explained in the Wolfson data sheet.



Here is an example of how such a 1KHz signal looks like (some details, which you have to figure out, are left out). It consists of 24 samples of a high DAC value followed by 24 samples of a zero DAC value. At 48 KHz sample rate, we then get a sampled-audio square wave of 1KHz.

In the FPGA top-level, there are two modules. The first one, codecpgm, is an extended version of the one you made for step 2. This extended version can also program the CODEC into audio output mode. The second module is audiogen. That is the module that will generate the DAC sample stream.

Important: You must operate the CODEC in master mode, so that it generates its own BCLK and DACLRCK signals. Both clocks are much slower than 50MHz, so the design of the audiogen module can treat them as data clocks. Study the 'pattern' example of Lecture 6 to learn how you can develop an FSM with a data clock.

What to turn in

You need to design all of the FPGA modules, making use of finite state machines. You can use the solution of homework 2, either your own or the class-provided solution. Your code must be synthesiable in Quartus and it must run on the DE1 SoC kit.

Push your solution before the deadline to github.

Coding Guidelines

Please refer to Chapter "RTL Coding Guidelines " of the Reuse Methodology Manual (accessible for free on the Tech network: https://link.springer.com/book/10.1007%2Fb116360). These are generic guidelines for HDL coding; I expect you to aim to follow at least Chapter 5.2 and Chapter 5.5.

I value code clarity but I do not intend to penalize you if you don't follow these guidelines to the letter. In the other hand, if you turn in messy code with obvious violations against the principles and main directives in this guideline, you will receive a penalty of up to 25% of the points (even if everything works perfectly).