Connor Aksama
EE 371
February 23, 2023
Lab 5 Report

Procedure

This lab is comprised of two tasks. The first task involved completing the implementation of a circuit that reflects audio data read from an on-board microphone back to the on-board speaker. The second task consisted of implementing a tone generator using a preprogrammed ROM module, and being able to switch between the task 1 data and this ROM data.

Task 1

I approached this task by first understanding the interface of the given CODEC, and the logic of when data read from the microphone should be written back to the speakers.

After implementing this logic, I properly configured the environment in the virtual lab – uploading the relevant Verilog files and MP3 data, then testing the functionality of the design.

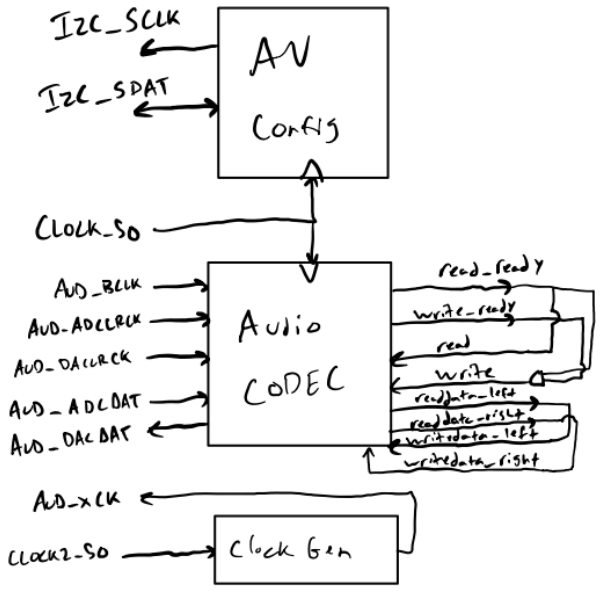


Figure 1. Top-Level Block Diagram for Lab 5 Task 1.

Task 2

I approached this task by first designing the tone generator module. I used the provided Python code to generate the MIF file, then implemented the logic to cycle and read through these values.

I then connected this module to the top-level module from Task 1, then created logic to switch between Task 1 data and data from this tone generator module based on input from an on-board switch. After implementing this logic, I tested and verified the design in ModelSim, then I properly configured the environment in the virtual lab – uploading the relevant Verilog files and MP3 data, then testing the functionality of the design.

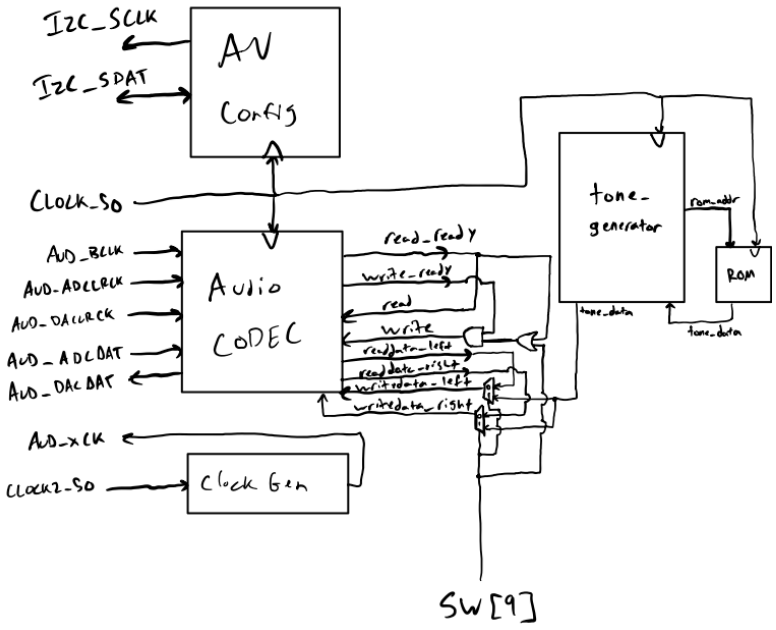


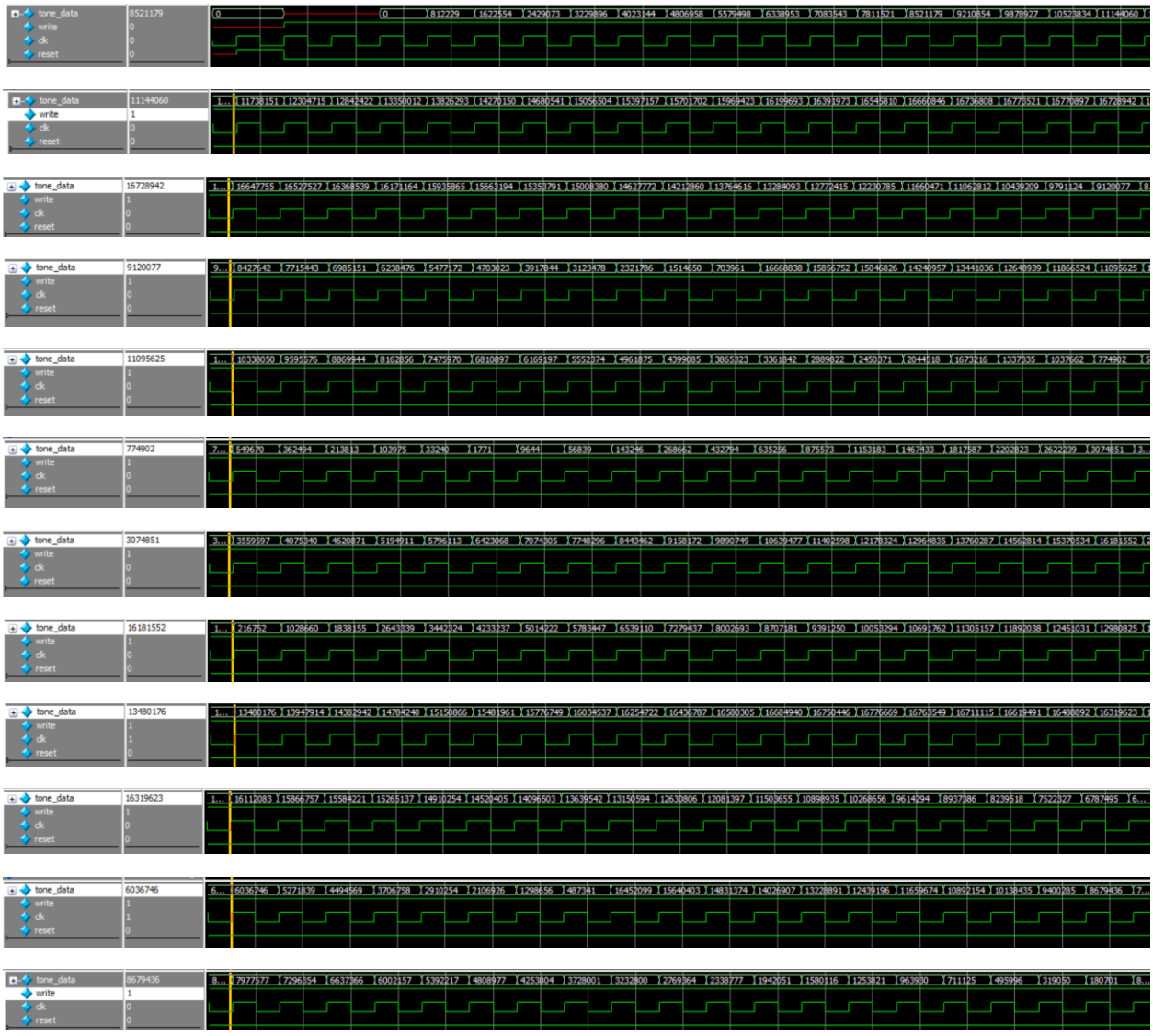
Figure 2. Top-Level Block Diagram for Lab 5 Task 2.

Results

Task 1

No simulations were required for Task 1 of this lab.

Task 2



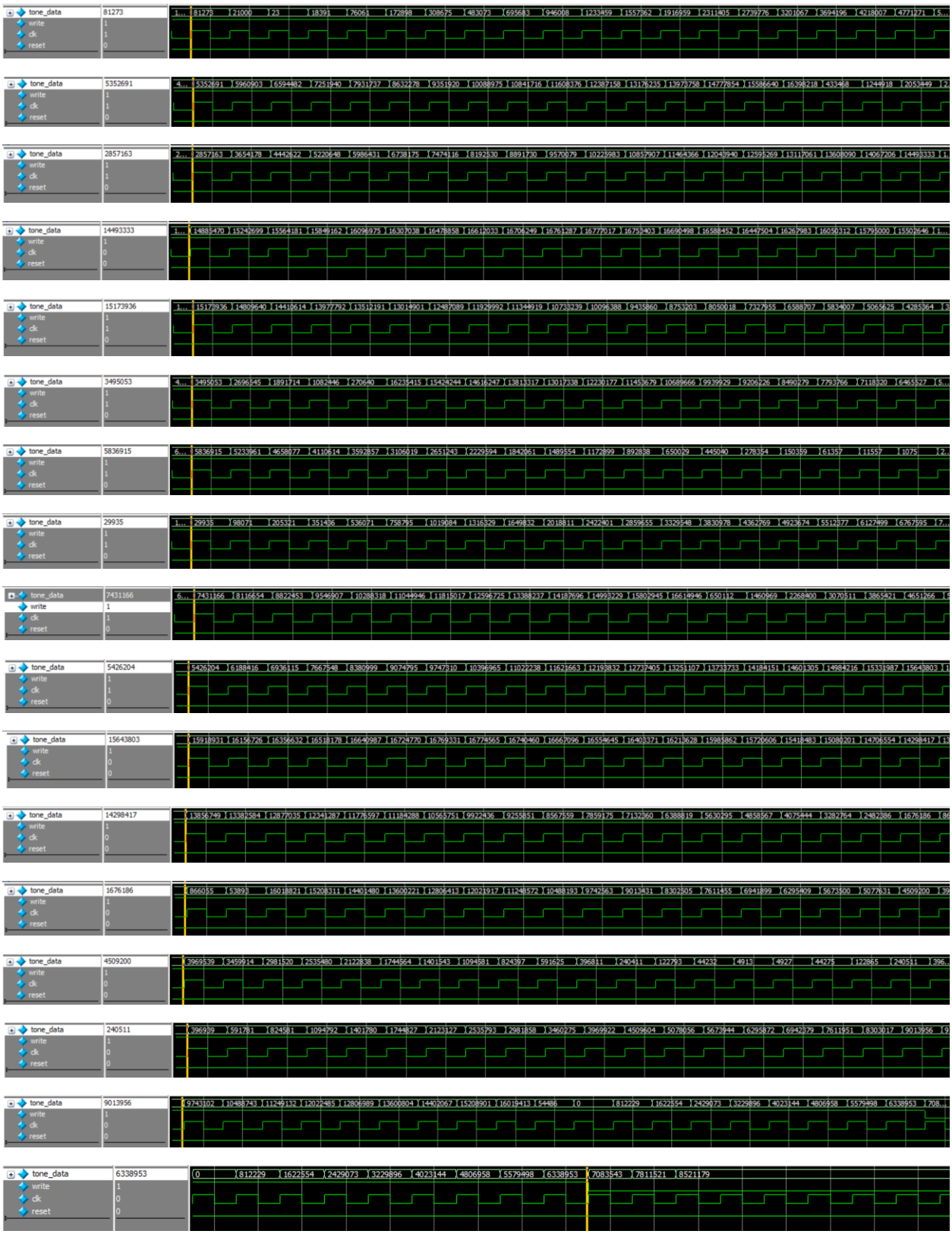


Figure 3. ModelSim Waveforms for the tone generator module.

This simulation shows the output sequence from the tone generator module. This simulation shows that the output values follow the same sequence as specified by the ROM's MIF file, and that the values cycle back to the beginning after reaching the end of the sequence, and that the output data does not change while write is low.

Final Product

The overarching goal of this lab was to gain experience interfacing with an audio CODEC – producing and reading audio data and only writing/reading while the CODEC is ready. In Task 1, I completed the implementation of a system that would read and playback audio played through a speaker adjacent to the board. In Task 2, I added to the Task 1 system a module that would produce a sine wave audio data, as well as the ability to toggle between this sine wave data and data read from the Task 1 system. (Completed as specified by the lab documents.)

Appendix: SystemVerilog Code

(See next page)