EE 371 Lab5 Report: Digital Signal processing

Ziyi Huang, Shu Xu

ECE, Embedded System

University of Washington

May 17, 2019

Introduction

This lab is an exercise in using the audio coder/decoder (CODEC) on the DE1-SoC board. We are given some basic code before start. The lab involves connecting a microphone to the audio CODEC to provide input sound, altering the received sound by filtering out noise, and then playing the resulting sound through speakers/headphones. There are three tasks for this lab:
In task 1, we will make some simple modifications and pass the the input from microphone to the speakers.
In task 2, we will design a logic circuit which works as an averaging FIR filter.
In task 3, we will add fifo and accumulator to this design and make it a N-sample averaging FIR filter.
The rest of the report will describe the procedures, results, and the problems/feedbacks in sections, respectively.

Procedure

Task 1

In this task, we enable the microphone and speaker connection by simply passing the input signal from microphone to the Codec unit, which writes it to the speaker.

Task 2

In this task, we use Verilog logic circuit to implement an averaging Finite Impulse Response(FIR) filter. The schematic diagram of this filter is shown in Figure 1. The D-flip-flops work as delays to store the last 8 inputs at the last 8 clock edges. The outputs are divided by 8 and then summed, which becomes the final result. This procedure is exactly the averaging process.

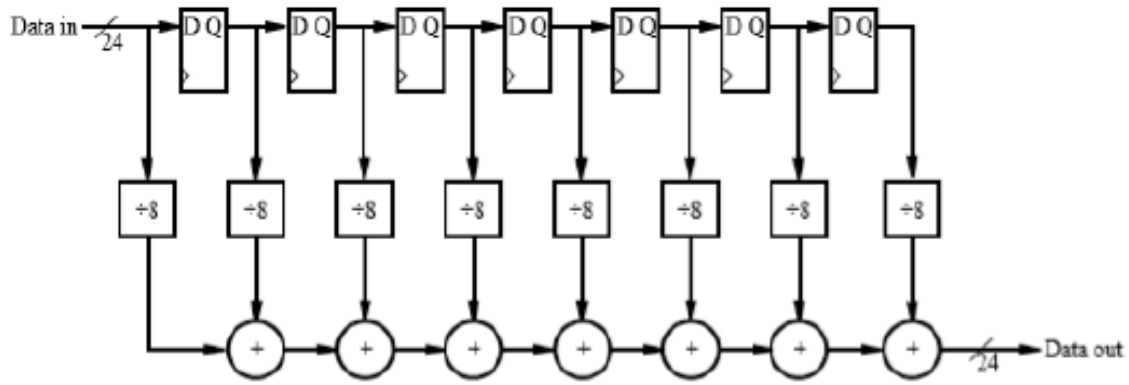


Figure 5. A simple averaging FIR filter

Figure 1: task2 schematic diagram

At the highest level, we build a filter module who accepts a 24-bit signal input and output its filtered signal. At the lower level, we built a 24-bit D-flip-flop module and connect 7 instances of it in series to implement the DFF group in the diagram. To script the numbering process, we use a 24x8 Verilog array buffer to store the DFF outputs. We then use a for loop to iterate over each item, divide its value by 8, and then add to the data output.

Task3

In this task, we improve the function of filter by using a parameterized implementation so that we can control the number of samples to average. To achieve this purpose, we add a FIFO and an accumulator to the design. The schematic diagram of task 3 design is shown in Figure 2. The FIFO outputs the sample that came in at (n+1) cycles ago, which then times -1 and sums with the new sample divided by N. This is to subtract the (n+1) cycle sample from the accumulator while adding the new sample to it, so that the accumulator only holds the average of the last n samples.

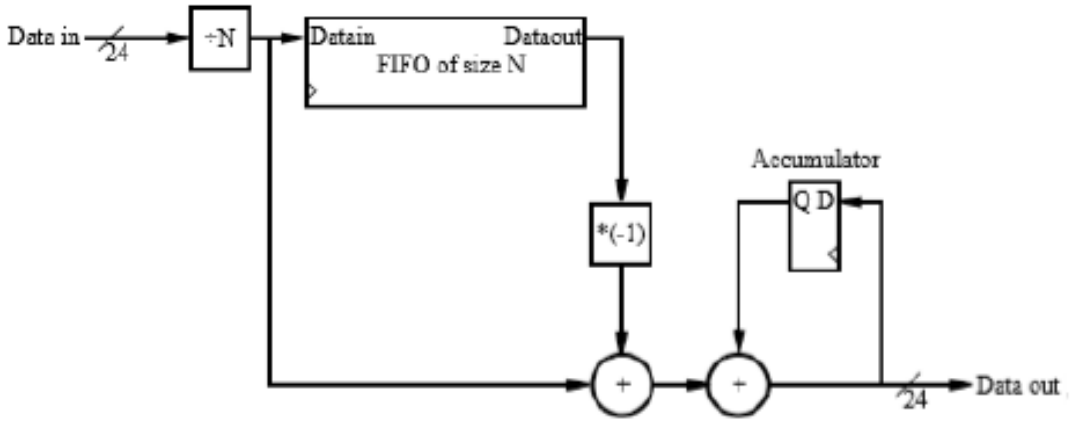


Figure 7. N-sample averaging FIR filter.

Figure 2: task3 schematic diagram

A 24xN FIFO is implemented in Verilog. The accumulator is implemented as a D-flip-flop that adds the subtracted input to the old accumulator value.

Results

Task 1

We were able to speak into the microphone and heard our amplified voice from the speaker.

Task 2

The block diagram is shown in Figure 3

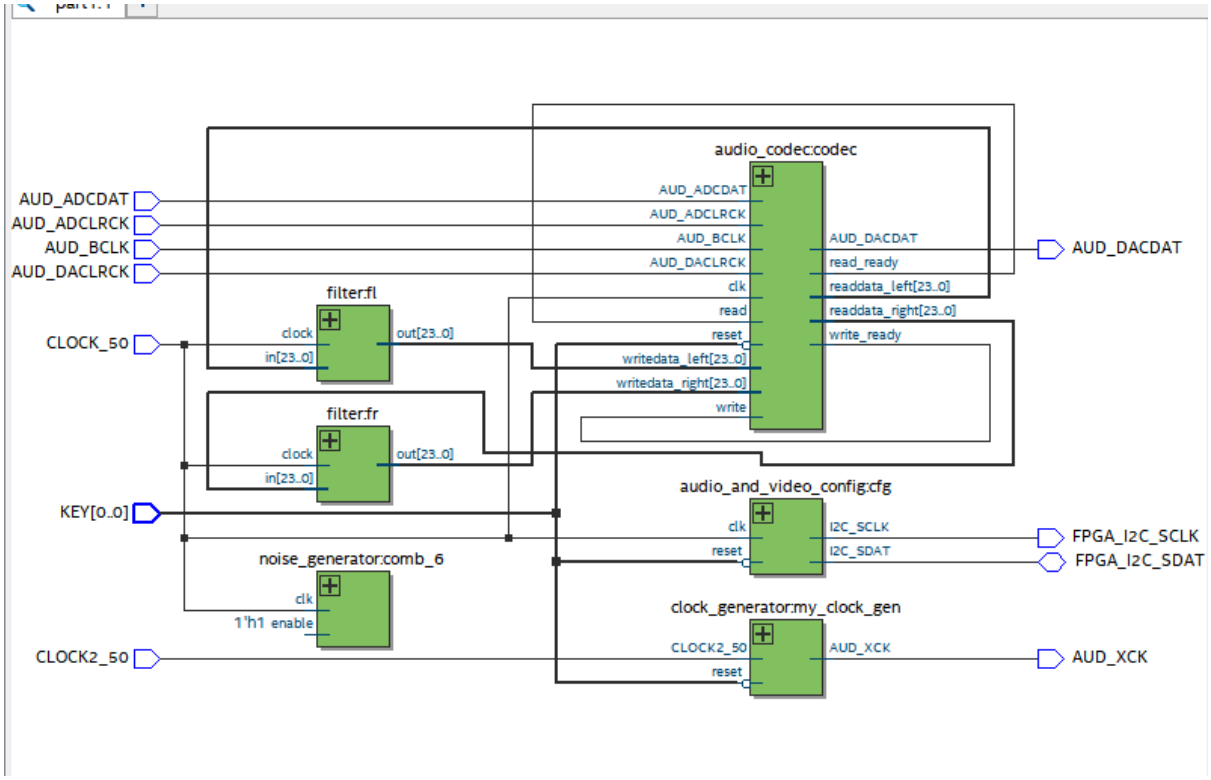


Figure 3: task2 block diagram

We design a testbench for the filter module. The simulation is shown in Figure 4

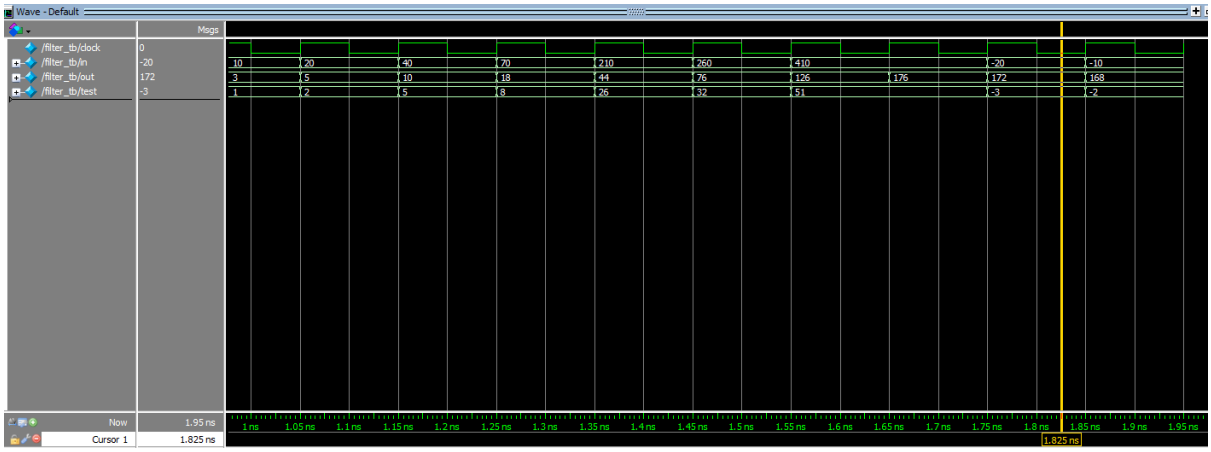
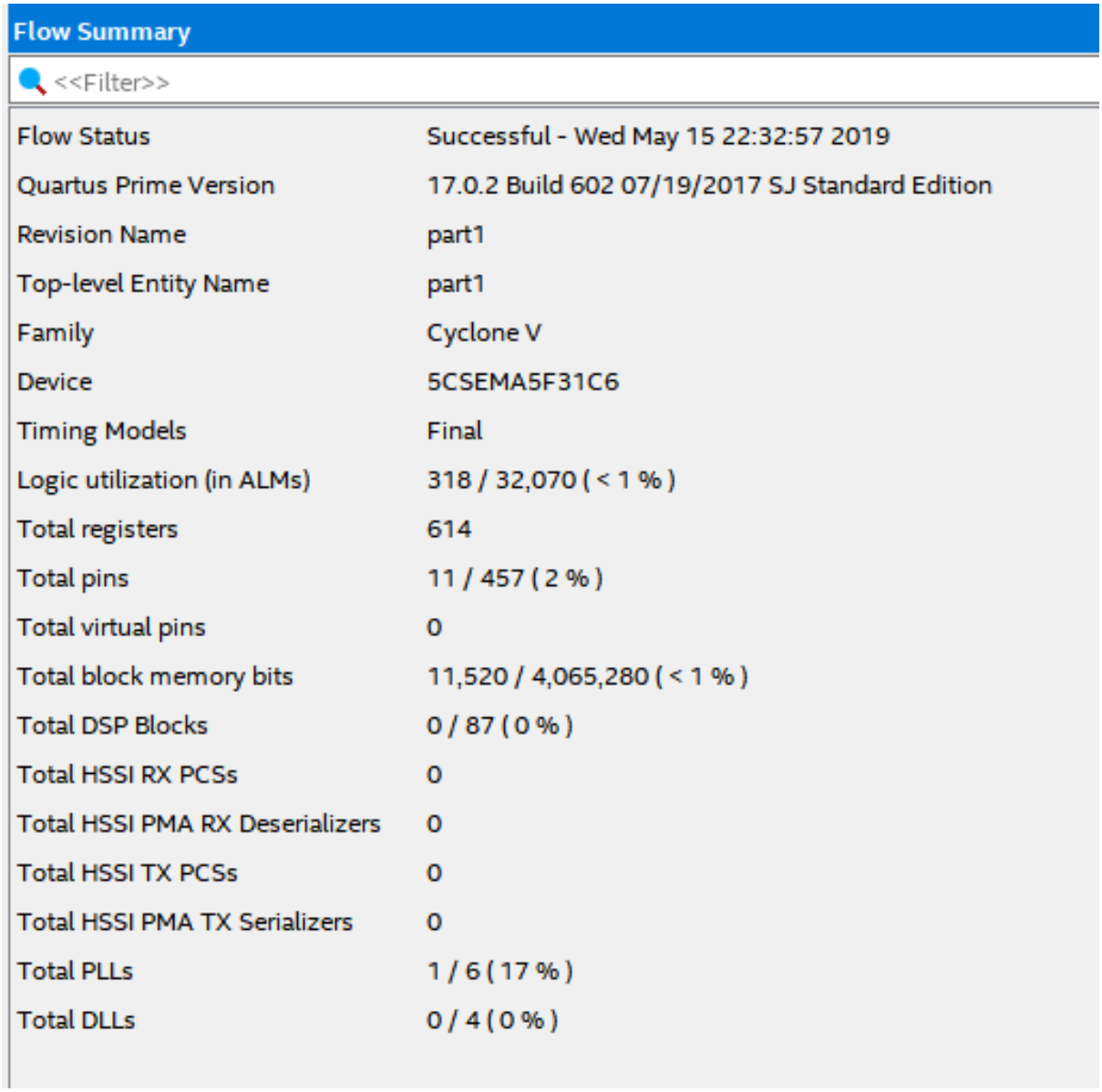


Figure 4: task2 filter simulation

The flow summary of task 2 design is shown in Figure 5.



Flow Summary

<<Filter>>

Flow Status	Successful - Wed May 15 22:32:57 2019
Quartus Prime Version	17.0.2 Build 602 07/19/2017 SJ Standard Edition
Revision Name	part1
Top-level Entity Name	part1
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	318 / 32,070 (< 1 %)
Total registers	614
Total pins	11 / 457 (2 %)
Total virtual pins	0
Total block memory bits	11,520 / 4,065,280 (< 1 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	1 / 6 (17 %)
Total DLLs	0 / 4 (0 %)

Figure 5: task2 design flow summary

Task3

The block diagram of task3 is shown in Figure 6.

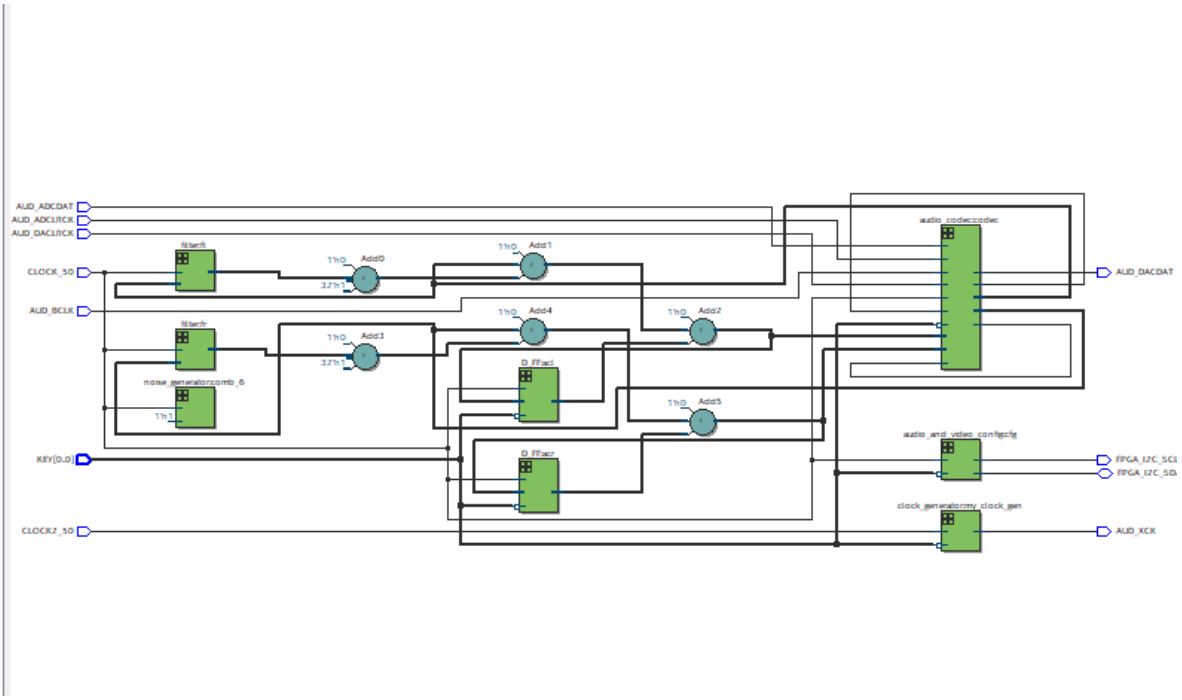


Figure 6: task3 block diagram

The flow summary of task 3 design is shown as Figure 7.

Flow Summary

<<Filter>>

Flow Status	Successful - Thu May 16 15:35:30 2019
Quartus Prime Version	17.0.2 Build 602 07/19/2017 SJ Standard Edition
Revision Name	part1
Top-level Entity Name	part1
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	878 / 32,070 (3 %)
Total registers	2260
Total pins	11 / 457 (2 %)
Total virtual pins	0
Total block memory bits	33,696 / 4,065,280 (< 1 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	1 / 6 (17 %)
Total DLLs	0 / 4 (0 %)

Figure 7: task3 design flow summary

The screenshot of signalTap is shown as Figure 8

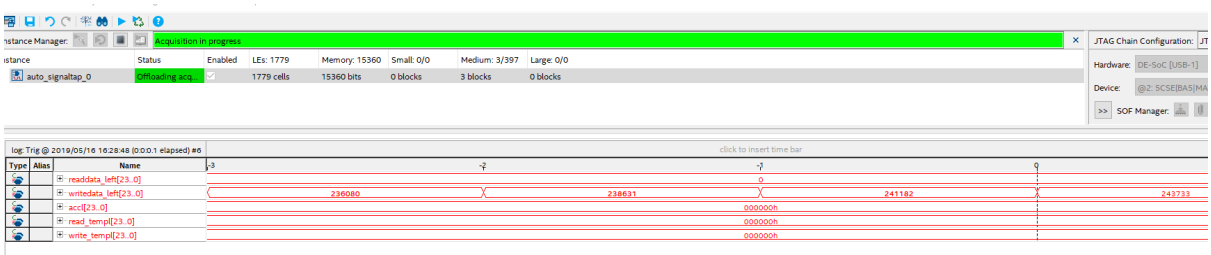


Figure 8: task3 signalTap screenshot

We design a testbench for the top level driver. The simulation is shown in Figure 9 and Figure 10

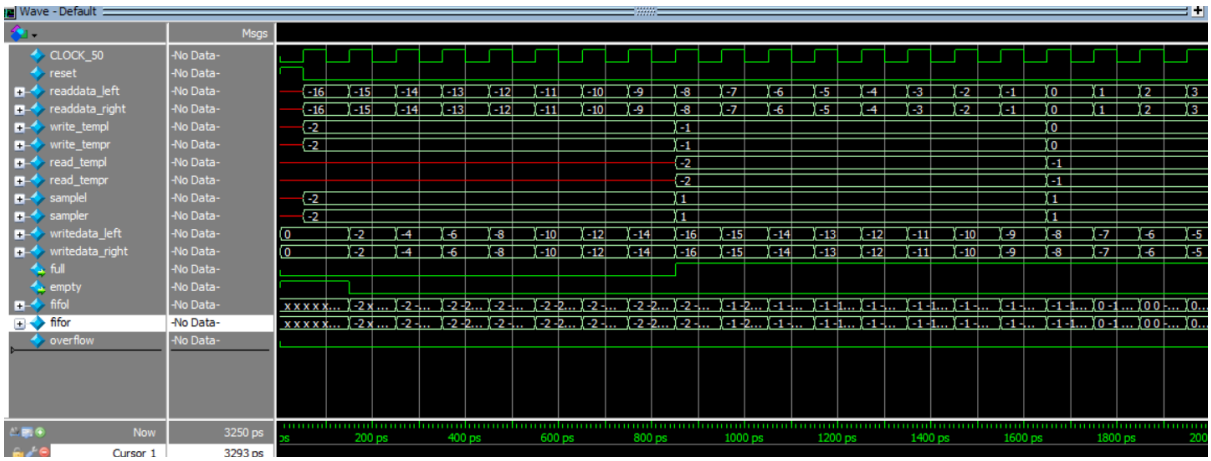


Figure 9: task3 simulation - negative input

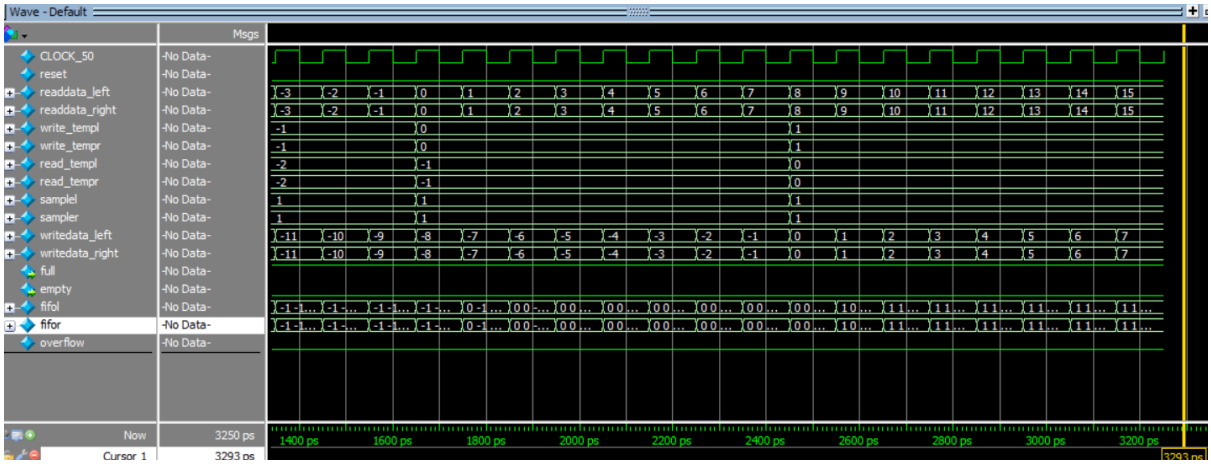


Figure 10: task3 simulation - positive input

The results of all testbenches agree with the expected values.

Problem Faced & Feedback

Approximatley 20 hrs were spent on this lab, consisting of 20% understanding the circuit and designing the circuit, 30% coding and simulating, 50% debugging for the hardware.

The first two tasks are easier parts.

The hardest part of task 3 is to understand the design and debug hardwares. We spent some hours to actually understand what happened in the this circuit. After we completed the circuit, the output sound always came with noise that we couldn't eliminate. After we change to a different location in the lab, the noise was gone. Maybe there were signal interruptions in the north corner of lab classroom.