UNIVERSITY OF CALIFORNIA, DAVIS
Department of Electrical and Computer Engineering
EEC 172 Spring 2023

LAB 4 Verification

Team Member 1: Bradley Manzo

Team Member 2: Thomas Ke

Section Number/TA: A05 / Begum Kasap

Demonstrate the ability to distinguish between buttons using the microphone (via DTMF) to your TA

Date	TA Signature	Notes
5/19	RS	Well done! 1 lab late.

Demonstrate multi-tap texting via DTMF between two Launchpad boards to your TA

Date	TA Signature	Notes
5/22	RS	Well done! 1 lab late.

Implementation	**/70**
- Functionality	/55
- Robustness	/10
- Presentation	/5

Report	**/30**
- Thoroughness	/10
- Clarity	/10
- Presentation	/5
- Code Clarity/Commentary	/5

EEC 172 Lab Four

Bradley Manzo
Computer Engineering B.A.
916953788
brmanzo@ucdavis.edu

Thomas Ke
Computer Engineering B.A.
917513004
thke@ucdavis.edu

1. INTRODUCTION

This lab builds upon Lab 3 as it involves decoding an input signal from a peripheral device, however the signal is no longer digital and discrete as in the case of the AT&T remote and IR receiver. Instead, this lab requires us to decode a continuous and analog signal from an audio input using the Goertzel Algorithm.

2. GOALS AND TASKS

Completing the first part of the lab requires the construction of the circuit operating the microphone amplifier and analog-to-digital converter to collect a set number of samples and identify the associated DTMF value. These values are collected in an array after 410 samples and then averaged to identify the associated DTMF character.

The second part of the lab requires us to decode these incoming DTMF values into letters using the multi-tap scheme from Lab 3 and display an outgoing text onto the OLED screen.

3. CONCEPTS AND UNDERSTANDING

3.1 Part I

Understanding how a continuous stream of sampled data is converted into a digital signal requires a fundamental understanding of the circuit elements involved as well as the underlying Goertzel algorithm. When the microphone amplifier is powered, it translates the frequency of sound waves into the frequency of an analog signal which is then converted to a digital signal using the ADC. Once enough samples have been collected, the average is subtracted to remove the DC bias and is then then trimmed and converted from big endian to little endian and is then passed to the Goertzel algorithm to complete processing. Big-endian describes the order format where the largest value is stored first; vice versa for Little-endian. The Goertzel algorithm is a form of Discrete Fourier Transform (DFT) that is useful for identifying the presence of a single frequency in a sample. The Goertzel algorithm is fed corresponding coefficients related to the target frequencies, which in this case are the characteristic frequencies of the row and column for our Dual Tone Multi Frequency (DTMF) matrix of outputs. We calculated these and initialized them manually by plugging the desired frequencies into the provided function and enclosing them in a separate array. By multiplying all samples with each of the eight coefficients we receive an array of eight powers. The greatest of the first four identifies the row, and the greatest of the second four identify the column. The decoded character is then the character located at that position in the matrix. DTMF is historically associated with the dial tones on telephones. Each telephone button generates two superimposed tones from two frequency groups: the low frequency group corresponding to the rows, and the high frequency group corresponding to the columns. The frequencies were chosen in a way that all tones are within the audible frequency range, non are multiples of each other, and the sum or difference of any two will not result in another. Finally, these samples are collected using an interrupt handler triggered by a timer that increments up to a set number of samples, and only then setting a flag high to process the samples. By initializing the timer with a strategic period, we can accurately measure how much time has passed between samples and between consecutive interrupts.

3.2 Part II

Part 2 mainly builds upon concepts explored in previous labs; however, we will briefly summarize them here. Once the characters are decoded into the corresponding button that was pressed, we refer to the number of times the button was repeatedly pressed within a certain timeframe to identify the proper letter to be printed to the screen. These delays will be elaborated on within our methods as they are design considerations rather than concepts. These characters are then printed to the OLED in the same way as labs two and three. The main difference between past labs is that we now have two peripherals attached to our SPI module: The ADC as a MISO, and the OLED as a MOSI. While the CC3200 only has two pins, since there is only unidirectional communication between the two, there is no collision or interference of data. The only step in assuring static operation is properly asserting and deasserting the proper chip select GPIOs during operation, as well as disabling the associated timers to prevent interrupts from interrupting out of these stages guaranteeing atomicity within our program.

4. METHODS

4.1 Part I

Beginning the lab, we constructed the circuit according to the provided diagram. While we produced several iterations, the one that provided the best results was having the microphone amplifier and ADC on a separate breadboard and grounded and powered by 3.3 VCC from a separate CC3200 board. The other CC3200 board was programmed and powered the OLED and voltage regulator with 5V. By increasing the distance between these components with long wires and separate grounds, we are doing our best to prevent crosstalk and noise on our signals. When we confirmed we were receiving data through our ADC and into our samples array, we developed a function to trim away the first three and last three of bits of garbage data from our 16-bit samples and then convert from big-endian to little-endian by reversing the order using bitwise operations. These samples are then collected into an array of 410 samples and passed to the Goertzel function. The sampled data originates from a range of time that is 0.0625ms long and is controlled by a timer of 5000 clock cycles. Once the 410th sample is taken, the ADC interrupt handler is disabled, and the sample array is passed into the Goertzel algorithm. The Goertzel function operates as explained in *3.1 Part I*, and produces the row and column values corresponding to the DTMF matrix. To prevent every character from being passed, we

programmed a confidence interval of four identical consecutive inputs which helps to prevent noise from affecting our outputs.

4.2 Part II

Much like with Lab 3, the second part required us to translate the DTMF characters into letters in accordance with the multi-tap phone scheme. To achieve the same result as lab 3 with a continuous input, we haveto add additional arbitration. We programmed an additional timer to increment a counter variable on a period of 1 millisecond. Because this count variable is global, it can be accessed from anywhere in the program. We can effectively measure delay by recording the previous ending count and subtracting the current count. We designed a lower bound delay between receiving confident outputs of 80ms to effectively debounce the input, preventing repeated inputs from being generated by accidently holding the button down too long. The second delay lasts for one second and determines when to stop listening for a repeat signal and print the current chosen character. We then print to the OLED using the drawchar() from the AdafruitGfx libraries in a similar manner to lab 3. We implemented the deletion by drawing a black rectangle and decrementing the position to write the next character, however we did not pursue the other features of lab 3 such as changing the font color and sending/receiving texts through UART.

5. Discussion of Challenges

This lab proved very challenging for us as we struggled to successfully build the circuit and read values from our ADC. Through the combination of iterating the circuit, debugging with our Saleae Logic Analyzer, and refactoring our interrupt handlers, we were successfully and reliably sampling data. Our original design for converting from big to little endian had to be revised because the number of operations required wasn't feasible for the short sampling duration. When debugging our OLED, we believe we accidentally short circuited our 5V pins resulting in an misfunctioning SPI module. In addition to this, our Saleae went missing from lab in the first week of this lab and it has not been returned. These two challenges proved to significantly delay our lab.

6. Contribution Breakdown

Bradley constructed the circuit, researched the Goertzel algorithm, implemented the interrupt handlers and received the signal from the microphone. Bradley then designed the decoding function and multi-tap logic and printed the text to the OLED.

7. Conclusion

This lab was much harder than we anticipated, and the several setbacks we experienced out of our control resulted in a decreased sense of optimism and motivation for learning. Despite these setbacks, it was interesting to apply concepts such as DFT to sample a continuous input, converting audio into digital values. Additionally, we feel like we have improved upon our previous decoding program for Lab3, debugging some of the minor errors as well as ironing out some major timing inconsistencies.