Lab1: Arduino Mega Familiarization Report

Kevin Lu 2167804 30-June-2023
Assignment: ECE474, Lab 1

A. **Introduction:**

The lab learning objectives are very simple for this lab. The main objectives are to understand how to set up Arduino on your computer. Then work through basic Arduino circuits and code that allow a LED to blink on the Arduino board as well as an external LED. Then, again, through basic Arduino circuits and code, understand and implement functionality for a speaker that lets to beep as well as continuously play a sound at 250hz.

Within these learning objectives, there are individual key points one should understand. For example, understanding how delay, with adjusting the frequency of the speaker, works as well as how to use the millis() function to layer functions on top of each other to run in parallel. Finally, learning how to use an oscilloscope to measure the frequency of a speaker and the basic functions and buttons of an oscilloscope.

B. **Methods and Techniques:**

The activities included setting up Arduino on a computer and building basic circuits around the board and external LEDs, and an external speaker. In addition, you would need to implement code that makes such hardware work and interact with each other.

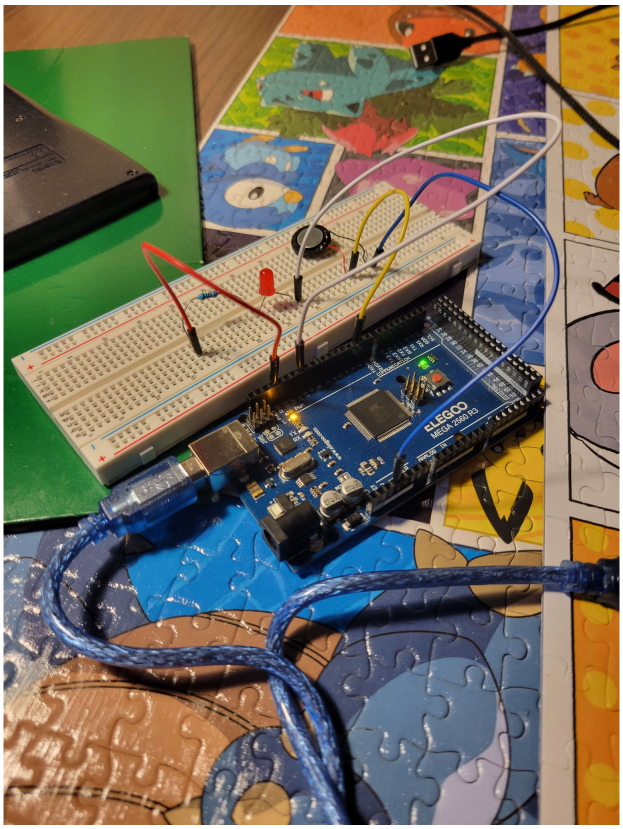
As such, the skills required to finish these activities are an understanding of C and Arduino toolkits and the ability to build basic circuits. Another activity we did was using the oscilloscope to measure the hertz of the speaker. The skillset here is knowing how to operate an oscilloscope: the functions of the knobs and buttons as well as its setup.

The instruments required included an Arduino mega set, a computer with Arduino installed, and an oscilloscope.

There were no code tests but rather tested through trial and error. Since we have a physical product we can flash code onto, we just needed to observe the Arduino Mega to test our implementation. In addition, we tested the frequency of the speaker using the oscilloscope.

C. **Experimental Results**:

As I mentioned earlier there were no code tests but rather we tested our implementation through the physical product such as the LEDs and speaker. The results we looked for are those that matched the specification. Whether the LEDs were flashing, alternative flashing, whether there was a speaker that beats on LED alternation, and a constant speaker at 250hz with LED alternating flashing. In order to confirm that our code and hardware worked together as intended we had to look for these results where applicable. The only part where we couldn't rely on the physical product was the hertz of the speaker. We needed to achieve a hertz of 250, and we validated this through our oscilloscope findings. Aftering hooking up the oscilloscope to our speaker, we measured the current to find a square wave and a reading of 250hz which validates our implementation.



D. **Code Documentation:**

For my code documentation, all tasks have the same initialization process which was to initialize the digital pin LED_BUILTIN, PIN_10, and PIN_2 as an output. These were simply just variable names for the values of 13, 10, and 2 respectively. For 5.1, however, I needed another variable called previousTime that supported keeping track of time intervals. This section was lines 24-35.

For task 2.2, the lines ranged from 39-45. This section was to blink the onboard LED which was to use delays and digitalWrite functions within a loop function.

For task 3.3, the lines ranged from 49-5.5. This section was to blink the external LED which required a change of ports.

For task 4.2, the lines ranged from 61-75. This section was to blink the onboard and off-board LEDs with speaker clicks. This required a lot more code and required alternating voltage requests for the external and onboard LEDs. Making the speaker click used a similar method as well.

For task 5.1, the lines ranged from 82-102. This section differed from the rest by using millis() instead of the delay() function to implement a timer. I initialized an unsigned long (since time can get very large) variable, currentTime, that kept track of the currentTime. Then finding the difference between the currentTime and the previousTime variable I was able to create intervals to flash one of the two LEDs. The speaker would continuously run, utilizing delays() to change the hz, until 2000ms where it would stop sounding. Lastly, I had to set previousTime to currentTime after 400ms which marked the end of a cycle.

E. **Overall performance summary**:

Two failures of my demo session were getting the oscilloscope to work and generating the right hertz. First off, the oscilloscope was in a different mode and generated a static-like wave instead of a square wave. Even after this fix, the oscilloscope was glitching out and produced random readings for the hertz of the speaker. Sometimes it would say 50hz other times it would say 500hz or infinity. This was another issue I had to overcome with the help of the staff. After everything with the oscilloscope was figured out, I was generating 200hz, and this is where I was directed to add another delay between sending a low voltage and high voltage to the speaker. After this quick change, my demo was a success and I produced an output of 250hz. Initially, I was still utilizing the “delay()” function which was not the right path as it completely halts the program. After finding out about the “millis()” function, I updated my code which greeted a successful demo.

I believe I accomplished the learning objectives very well. Although there were hiccups along the way, I understood the main ideas surrounding this lab: creating a basic circuit with LEDs and speakers and its interaction with the software. The only issue that I had was the oscilloscope issue which I overcame through a bit more online research and help from staff.

F. **Teamwork Breakdown**:

This was a solo lab so I did all the sections and components of this lab.

G. **Discussion and conclusions**:

The most challenging part of this lab was understanding how an oscilloscope works and how I can use one to measure the current of a speaker. Although we were given a simple guide and introduction to how the main three knobs work (vertical, horizontal, trigger), I was not given enough knowledge to fix issues with the oscilloscope. Initially, my speaker did not produce a square wave and had a static-like wave. I did not know how to fix it, however, the solution was simply just pressing a single button. It was moments like this that made the oscilloscope portion of this lab the most challenging for me.

The part that I am especially proud of would be utilizing the “millis()” function to measure the current time the program is at. Utilizing this information, I was able to layer multiple functions on top of each either depending on what interval of time the currentTime vs the previousTime was at. This allowed me to finish task 5.1 of the lab, which was the most difficult, in a few lines of code.

Something that I learned in addition to the official learning objectives was how to use a breadboard. Initially, I was connecting the resistor and LED by twisting them together which is not a good idea. My partner taught me how to use a breadboard which has been very helpful since it links hardware together for you.