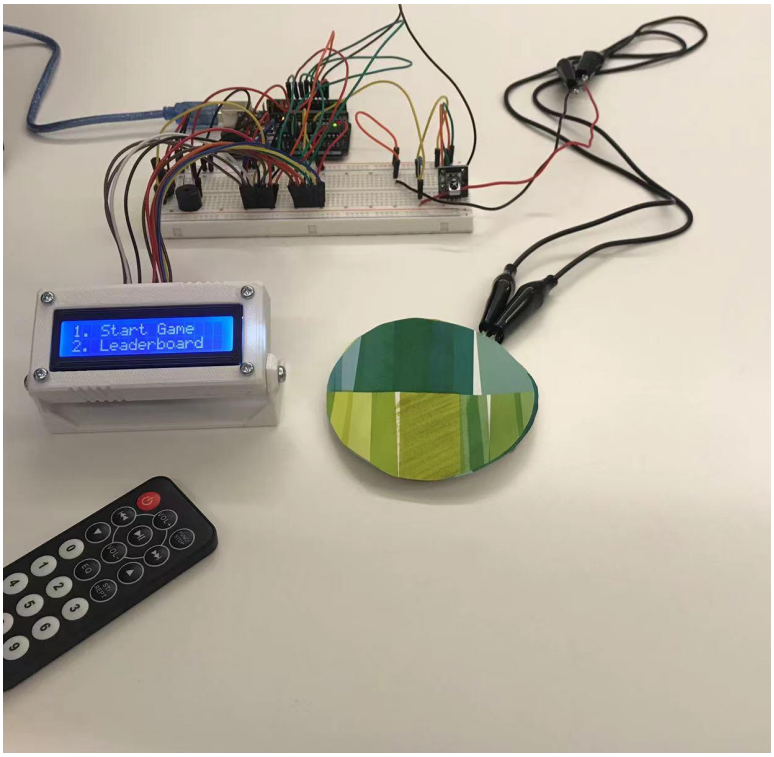
“How Fast Can You Chug!” Project Report



Prepared for:

ECE 5

Prepared By:

Samvel Manukyan, Bo Zhao, Ethan Brooker

March 11, 2022

Table of Contents

DEVICE FUNCTIONALITY

Our device allows one to test the amount of time that it takes for them to chug a drink and then allows them to compare their score to the scores of their friends. It does this using a force sensor that is able to register when the drink is picked up and will start a timer. This timer will continue until the drink is placed back on the sensor. Once the timer has stopped, the player can compare their score to other competitors using the leaderboard. In order to scroll through the leaderboard and reset the game, the player uses a remote that is registered through an IR receiver. Furthermore, our device can also be used as a Rubix cube timer in that instead you would be using a Rubix cube instead of a drink.

Our future plans for the project are to implement another LCD display and force sensor that would allow multiple players to go against each other in a head to head competition.. Displaying tunes after a player starts the challenge, and after the challenge if a score threshold is met using “multithreading” with the Arduino.

HARDWARE COMPONENTS

Main hardware needed for the project:

- Arduino UNO R3 Controller Board
- Breadboard
- LCD1602 Module
- Remote Control
- IR Receiver Module
- Force Sensing Resistor (0.5in ,circle, 1oz-22LB,Flexible) - $13.80 - Amazon
- 10k Potentiometer
- Alligator Clips (10 pack) - $5.70 - ECE Shop

DESIGN TIMELINE

Lab 1 (February 16th):

In this lab, our group was able to fully assemble the hardware for the force sensor and the LCD display. We were also able to figure out how the force sensor was able to register values and printed those values on the LCD display. We also noticed that the force sensor wasn't able to register a force reliably if it was placed on a surface area larger than the sensor itself. This led us to test different versions of coasters and situations in which the sensor would be able to register larger objects, like the bottom of a cup. We weren't able to get as much work as we would have liked to have done in this lab because the internet stopped working for a large portion of it. This made it very difficult to research some of the issues we were facing in implementing the code.

Lab 2 (February 23rd):

In this lab we were able to get the LCD display to actually count upwards. However, we did not use the millis function. Instead, we were using the delay function inorder to increment the time after a weight is lifted from the force sensor. This method meant that we were not getting exact values for the actual time, and overtime the number of displayed seconds and milliseconds would deviate from the true amount of time. We also spent some time beginning to setup and troubleshoot the piezo buzzer, although we were not able to fully get it working.

Lab 3 (March 2nd):

In this week's lab, our group changed our previous method of using delays for the timer, to a method that involved the millis() command. This command involved instantiating a value as the time passed until the Arduino started up. The value would then be compared to a new value set to zero and taking the difference would yield the exact time that has passed since the new value was instantiated. The value of the time passed was then displayed on the LCD monitor, updated in a loop. This method fixed the issue above that caused deviation from the real time after a long period. In this lab we also were able to get the buzzer to play either Happy Birthday or Star Wars tune, however we struggled to implement a system where they would play when a score was achieved. Our group also spent time beginning to troubleshoot and test out using the EEPROM library so that we could permanently save the scores that people achieved. This would mean that everytime we turned on the device people would be able to see the top scores of all time. It was also in this lab where we decided that we were going to add in the remote control to be able to scroll through the leaderboard and to reset the game.

Lab 4 (March 9th):

This was the last lab before the science fair, and in this lab our group completed the code for the leaderboard as well as for the IR Receiver and remote. For the leaderboard, we were now able to store the top 5 scores that were recorded from when the Arduino was last turned on. We didn't end up using the EEPROM library due to time constraints and issues involving implementing that code, therefore the leaderboard was reset every time the device was reset. We also were able to implement the code for the IR receiver and remote using different cases. We set it up so that there were different functions for the remote including resetting the leaderboard, resetting the game, viewing the leaderboard, and scrolling up and down through the 5 top scores on the leaderboard. We also spent more time figuring out how to play different songs from the Arduino, and we found a lot of songs that would be fun to play out of the buzzer during and after the timer. However, we realized that due to the timer in the Arduino, we couldn't have any other functions being completed while the song was playing. This meant that other aspects of our device would fail, and therefore we chose to remove the buzzer. In this lab, our force sensor also broke, which forced us to attach alligator clips to the strips of the sensor in order to register values. Finally, we were able to create a coaster that would allow the force sensor to register a cup that was bigger than the sizer of the force sensor. In order to do this, we used a stack of coins that were placed under the force sensor, and then a piece of cardboard on top of the sensor. This meant that now the force was not equally distributed across the entire coaster, but was concentrated on the sensor itself.

Only 4 labs were included because all labs were delayed 2 weeks

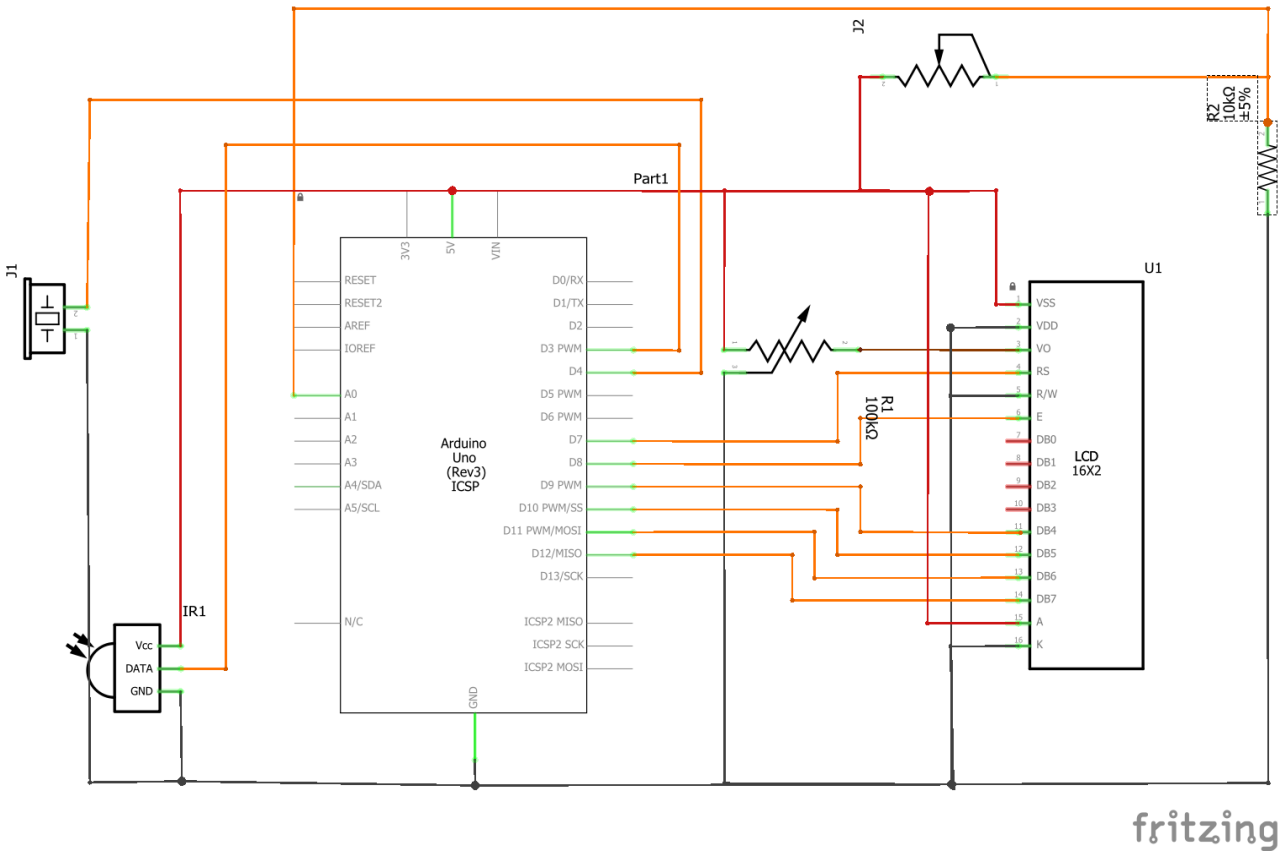
SOFTWARE DESIGN

Our function included 3 main libraries: Liquid Crystal, IRremote, and KickSort library, each responsible for the LCD, the remote controller and the leaderboard function. The setup function defined all the pins and set the appropriate ones to send and receive signals. The sort speeds function utilized the KickSort library to sort an array from low to high. The tone and pitch function were also present but not used in the project because it caused conflicts with the internal Arduino timer and created bugs in our project.

The general pseudo code in loop function is presented:

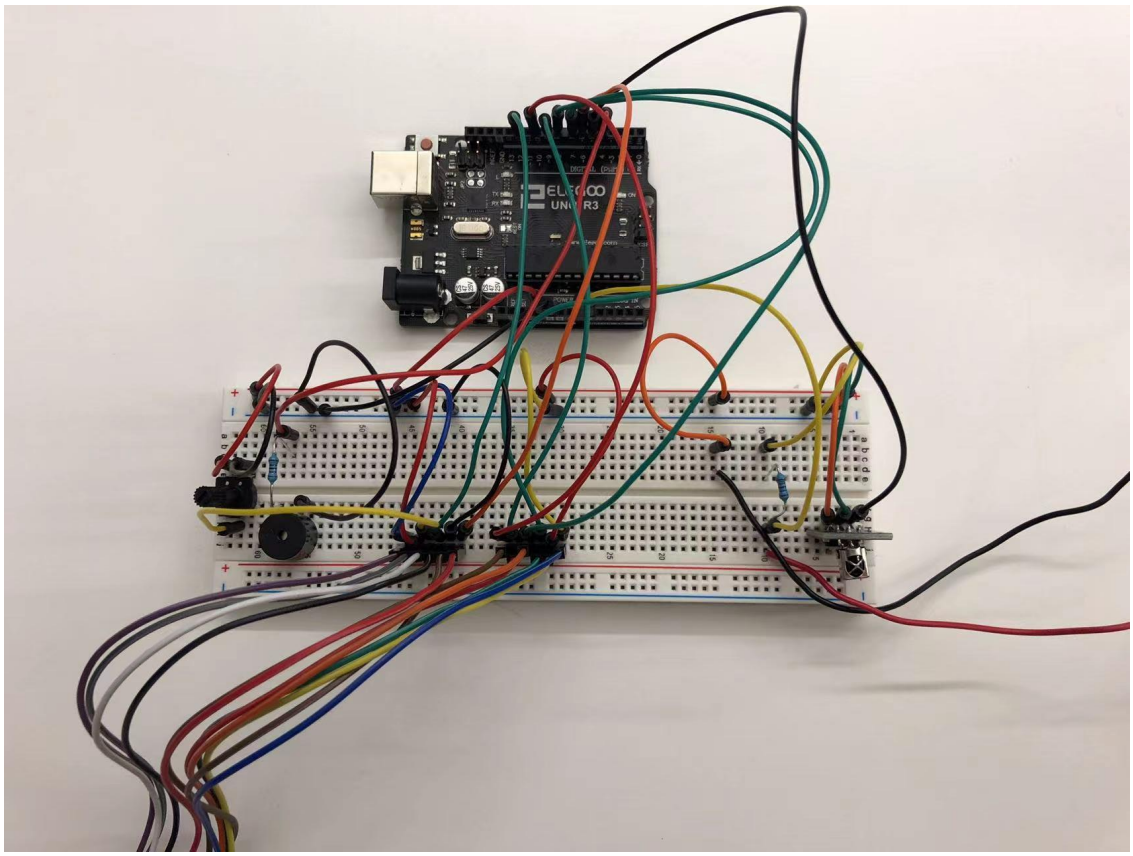
- If the IR receiver decodes something
    - Continue to receive signal
    - Switch case for the signals decoded
    - If 1 is pressed on the controller
        - Goes into the timer function
            - If a force is detected, let the timer be ready to start
            - If a force disappears, start the timer
                - Save the score in scores variable
            - If a force is again detected, stop the timer and print the score
                - Sorts the array in which scores are stored
        - Stop the case
    - If 2 is pressed
        - Enter the leaderboard
            - Display the leaderboard and the first speed.
        - If the down button is pressed
            - Print the next two scores (2, 3) on the LCD
        - If << button is pressed
            - Print the next two scores (4, 5) on the LCD
        - If the up button is pressed
            - Print the scores before (2, 3) on the LCD
        - If the >> button is pressed
            - Print the first page of the leaderboard
        - Stop the case
    - If the power button is pressed
        - Return the main menu of the game
        - Continue listening for IR signal
    - If the reset button is pressed
        - Resets all scores in the leaderboard

CIRCUIT SCHEMATIC



The top left most component is the buzzer connected to pin 4 and ground.
The bottom left component is the IR receiver accompanied by the remote controller.
Middle right component is the LCD connected to pins 7-12.
Top right is the force sensor connected to a 10k Ohms resistor and pin A0.
Between the Arduino and the LCD is the potentiometer connected to the LCD.

CIRCUIT PROTOTYPE



From left to right on the breadboard:

- Potentiometer - Connected to the LCD, VCC, and GND to control the brightness of the LCD
- Buzzer - implemented but not used because of conflict with arduino's internal timer
- LCD (not shown) - The LCD is connected through the jumper wires in order to fit our 3-D printed case. The pins are connected to Arduino pins 7-12
- Force sensor (not shown) - The force sensor is connected to pin A0 of the Arduino in order to read the different weights placed on it.
- IR receiver - used in conjunction with the remote controller. Connected to Arduino pin 3, VCC, and GND.
- Black wires used for GND, red wires for VCC, other wires are used for distinguishing each pin connection.

TESTING

In terms of testing, one of the main things we wanted to ensure was that the sensor would be able to register an empty cup on the sensor. This is important because if the sensor didn't register the empty cup, the timer wouldn't stop and the score wouldn't be recorded. In order to test this we tried placing an empty plastic cup on the sensor with different amounts of force. The force sensor was able to register every single case except for if it was put down as softly as possible. We also tested it by allowing people to play our game, and we found that it was able to register the empty cup every time. This is because when people were being timed they always placed the cup down with more force than the minimum threshold.

We performed tests for the speed of the remote controller and how fast the IR receiver can receive a signal for the best results. However, the IR receiver still had bugs in which it would receive random signals for the up and down buttons, causing the scrolling function of our leader board a bit buggy. We tried a lot of different approaches and the bug was left unresolved as the signal it was receiving had different hexadecimal values causing delays in scrolling. This was not a huge problem because the function would still work. However, multiple clicks were required.

Furthermore, the timer involved a lot of testing because of the various extraneous behaviors that it produces. There were issues involving the timer being setup completely backwards where it would start when the force sensor detects the weight. There is the issue of the timer going straight to the scores page, not showing the incremented time. There were also issues involving the integration of the EEprom library into the timer which caused integer overflow and weird behaviors to occur in the leaderboard. However, after various testing and debugging by our team and the assistance of the TAs, we were able to fix all the issues with the timer and successfully run our game.