Project Knock Knock

Huanzhen Zhu, Saarth Mehrotra, Doyung Lee, Hayley Hansson

May 7, 2013

1 Introduction

A device has been created that can analyze the location and the pattern of a knocking frequency to determine whether the input knock matches a predetermined password knock. First we validated our proof of concept through a one-dimensional model. Our current two-dimensional device consists of four vibrational sensors placed on the corners of a surface, a circuit that involves either a comparator or an operational amplifier for processing, and an Arduino that calculates the results.

2 Construction

2.1 Materials

- 24 x 24 x 0.5 in^3 Wooden Board
- 4 Piezo Vibration Sensors (Measurement Specialties SEN-09199)
- Arduino Uno
- Computer

2.2 Diagrams

Following is a block diagram that demonstrates how our circuit operates:

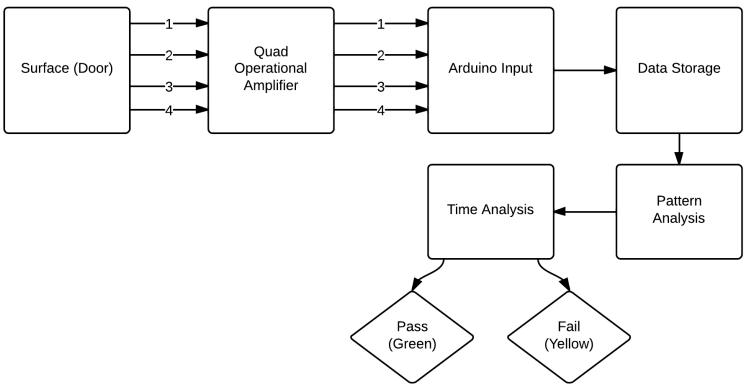

Figure 1: Block diagram of the process the Knock Knock system takes.

2.3 Process

Knock signals are first detected on a 24 x 24 x 0.5 in^3 wooden board propped by four wooden stomps. The board is marked into four quadrants, labeled 1-4 clockwise. The surface contains four corner piezoelectric vibrational sensors. The sensors that we chose to use were the minisense 100 made by Measurement Specialties (SEN-09199). They are a canti-lever instrument that creates a large voltage as the film moves from its natural position. The Arduino detects these signals and stores them. From there, the code determines which of the four quadrants were hit by measuring which piezo sensor was triggered first. The code then checks at which time steps the triggers arrived and determines if these ratios match the password timing (this is to check for the timing of each successive knock.)

2.4 Code

Our code is optimized for capturing data as quickly as possible. We have set up a timer, which is initiated after recording the first knock. We then store the knock data as an array of which quadrant the knock was recorded in and what the time step difference was between each knock.

In order to determine whether the knock data matches the knock password, a series of for loops and nested if statements were used with the arrays of the knock data and the password data. Each element of the knock and the password arrays were compared, followed by nested if statements to determine each subsequent element. There were two arrays for the data: one for the location verification and one for the timing. The timing array was used to determine the ratio between each knock and compared to a ratio key.

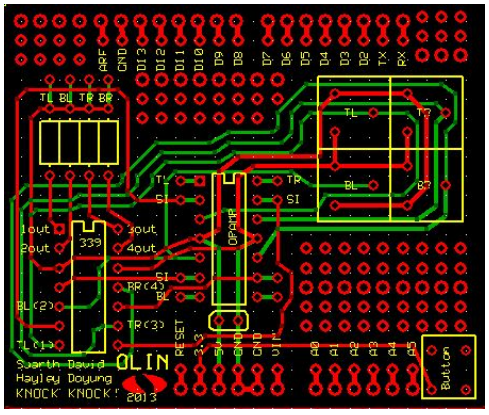

Figure 2: PCB Layout for both an Op-Amp and LM339 comparator circuit. The PCB was designed such that either the Op-Amp or the comparator could be used.

Figure 3: CAD representation of our system setup. This is effectively a mock door with four quadrants (1-4 clockwise). Each corner has a vibration sensor.

3 Aspects of the Project

3.1 Operational Amplifier Comparator

The piezoelectric sensor sends the voltage difference to the negative input of the operational amplifier. The positive input is connected to a 10k potentiometer. The potentiometer is used to determine what voltage the "trigger mark" is. In other words, if the sensor inputs a voltage higher than the trigger mark, the op-amp outputs a 5 volt signal. These railing signals are the knocks collected by the Arduino. Therefore, the sensitivities of the signals can be modified by tuning the potentiometers.

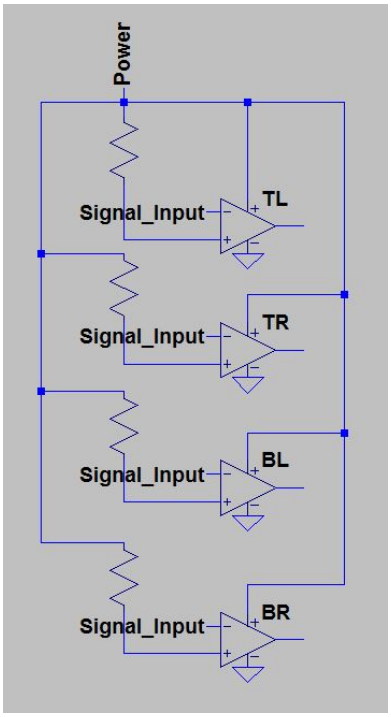

Figure 4: Circuit schematic for an Op-Amp powered comparator. TL, TR, BL and BR mean "top left," "top right," "bottom left," and "bottom right" respectively, describing the positions of the sensors on the mock door. The resistors are a representation of the 10k potentiometers.

3.2 Time Calculations and Clock Cycles

The speed of sound through a generic piece of wood is averaged at 3960 m/s. Due to the size of our board, knocks are usually detected around a span of 1 feet, or 1/3 meters. Therefore, each sensor receives the signal at a magnitude of microseconds. However, the difference of time between each knock would be in the magnitude of around one hundred nanoseconds. This time difference causes the experiment to be performed at the highest resolution that the Arduino can generate. A clock cycle reference is implemented such that once the first knock is detected, a counter is started. Every knock is referenced with its count number; because the count loop is going at a speed of 67.5 nanoseconds, knock position is able to be differentiated from each other. Also, digital reads were chosen instead of analog ports to further boost speed, as digital only requires a bit to store its positioning as either HIGH or LOW.

3.3 Logic

The password analysis of our device requires both position and timing to match in order to unlock. To accomplish both of these matches, position and timing are collected independently of each other. Currently, the position is tracked and abstracted based on which sensor receives the signal first. Therefore, without knowing the absolute positions of the knocks, we can accurately collect enough relevant data to make position important but not strict. This feature actually benefits the device because it provides the signal enough leeway without reducing excessive accuracy. Also, it increases the rate at which the Arduino loops to provide higher resolution.

4 Results

The constructed device behaves fairly well, although the reliability of the construction is questionable. While the oscilloscope receives clear and accurate readings with an ability to read positioning, the Arduino has a tendency to error if the knocks become too fast, and the counter may reset when higher numbers are reached. Below is an example of the oscilloscope's collected data.

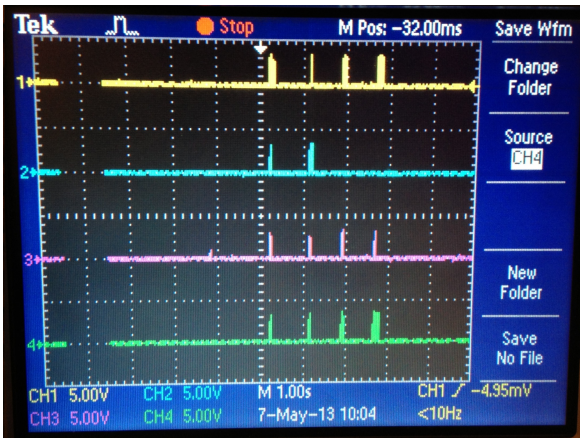

Figure 5: Knock detection on a four channel oscilloscope. A photo had to be taken due to a lack of data transfer methods from the oscilloscope to a computer (scope model is obsolete).

Each channel represents the signal from a piezoelectric sensor. Note that each knock appears to hit approximately the same time. At a higher resolution, the Arduino can detect which sensor hits first. Sometimes when the board is hit at an extreme location (the edge or the corner), some sensors do not receive any signal. Following are the results of the Arduino testing.

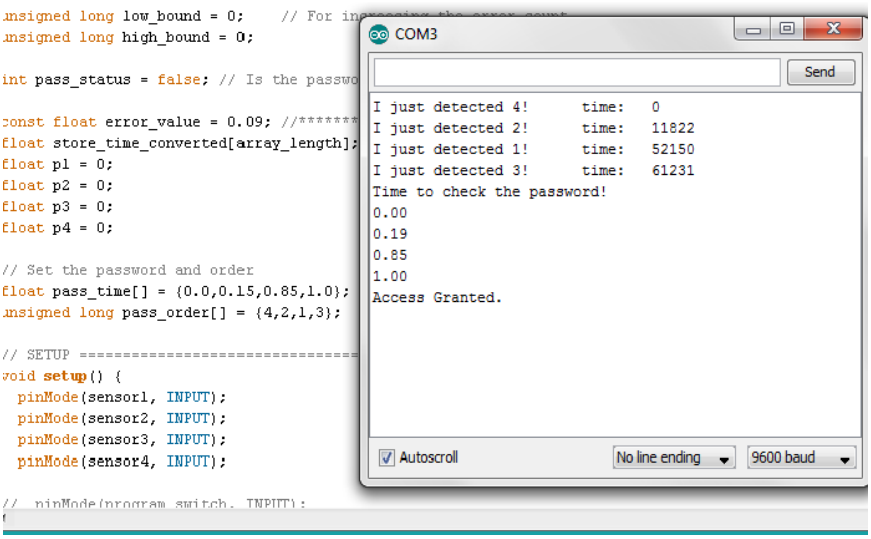

Figure 6: Access granted. Pattern and rhythm match. The code has a password "key" that is a ratio between knocks and the location pattern. If the input knocks match the key within the bounds of the error value, the green LED will flash.

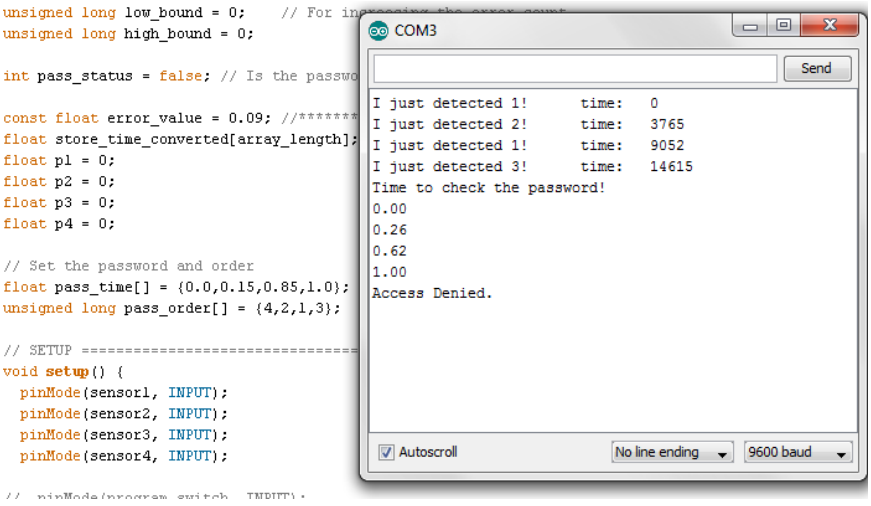

Figure 7: Access denied. Pattern and rhythm do not match.

5 Conclusions

Through a series of vibrational sensors, comparisons, and programmed analysis, a device that could detect and compute a knock's position and pattern was constructed. Through this project, we

have understood more about how comparators operate (although we did not use one directly), the capabilities of the Arduino, and signal analysis. In this lab, we all gained a greater understanding of how piezo sensors work, how to develop a PCB layout, how to use an Arduino to collect data, and we all improved on our debugging skills.

5.1 Technical Difficulties

One of the main issues that we kept contiuously running into was the speed of sound. Becuase it travels relatively fast, we were always doubting whether the arduino would have a fast enough clock cycle to be able to sense the delay between the sensors. We also questioned whether the Arduinpo would have enough memory to store the knocks with their relative times. This problem was overcome when we discovered "unsigned long" which allowed us to set our variables to only store positive integers. this allowed to to greatly increase the number storage on the Arduino.w

Another issue we ran into was the mechanical design of the board. The first version was clamped to the table. This prevented the board from vibrating and the sensors could only pick up very strong knocks. However, a simple redesign using thinner wood and legs improved the sensors' sensitivity as well as accuracy.

5.2 Lessons Learned

Throughout this project, one theme came up constantly, which was the integration of different parts to create one coherent deliverable. We had the board and programming component developing in parallel. A problem arose in which the programming worked with a "dummy" setup of buttons instead of knocks, and the board seemed to work with the oscilloscope. However, we spent a long time trying to integrate these two components together. This is interesting because throughout this project, we had issues with using two, 2-channel oscilloscopes and having them timed correctly as well as retrieving data from a 4-channel oscilloscope. This project showed us how important being able to integrate working parts to create one whole working system.

5.3 Future Work

Future work includes attaching our device to a door, creating code to be able to program in knocks, and replacing the Green LED with a motor that controls the doors lock.

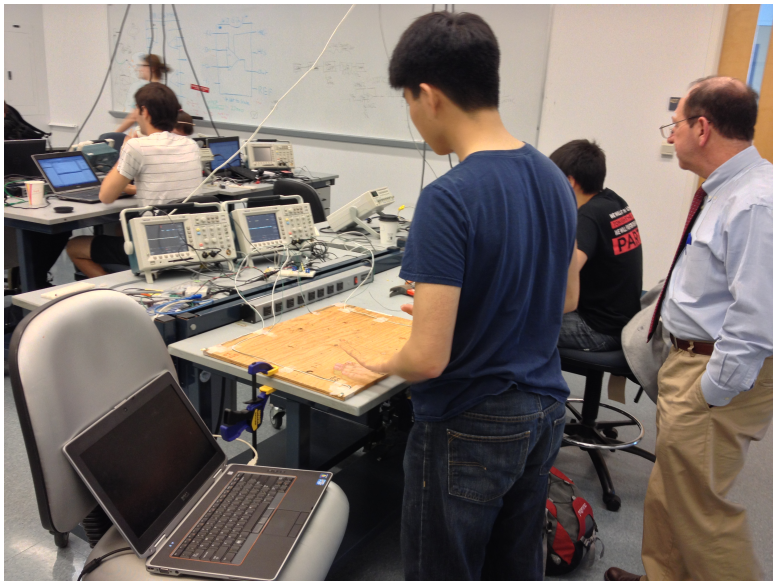

Figure 8: Testing configuration 1. The door was clamped directly to the table which caused sensor issues. We are also using two, 2-channel oscilloscopes.

Figure 9: Testing configuration 2.0. The door was raised via four legs to improve sensor sensitivity. The breadboard was replaced with a PCB and we implemented Arduino calculation.