EEE 174 / CpE 185 Final Project Report

Electronic Safe

Dynamic Duo

Andrew Stites

Vi Than

Team members and responsibilities

Vi Than:

- Product Design
- Hardware Configuration
- Data Logging

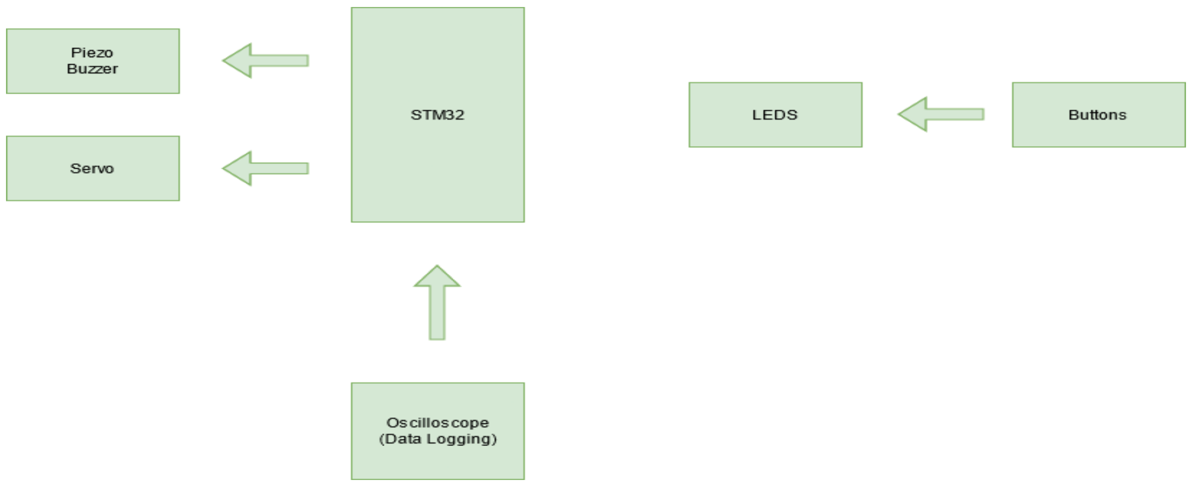
Andrew Stites:

- C Language Implementation
- Hardware Configuration
- Data Logging

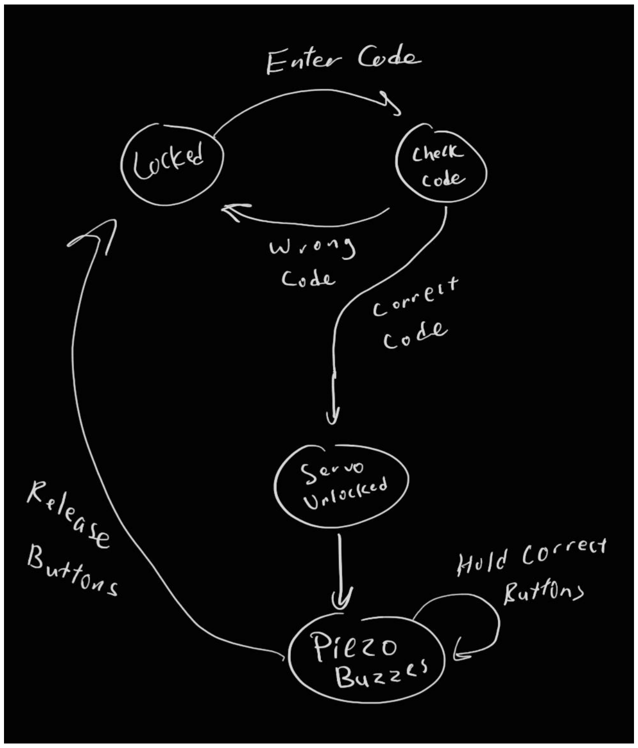
Project Description

The functionality of our project consists of mimicking a simple safe with electronic automation. We utilized the STM32L432KC in conjunction with LEDs, a servo, and a piezo buzzer. We added buttons as input for the LEDs to light up when each respective button is pressed. In addition, the buttons, when all are pressed, activate the "code" for the STM32 to send a PWM signal to the servo to "unlock" the safe door. When the signal is sent to the servo, the piezo buzzer emits a sound alerting to the unlocking process. We implemented the AD2 oscilloscope to collect some data regarding the unlocking of the servo.

Block Diagram

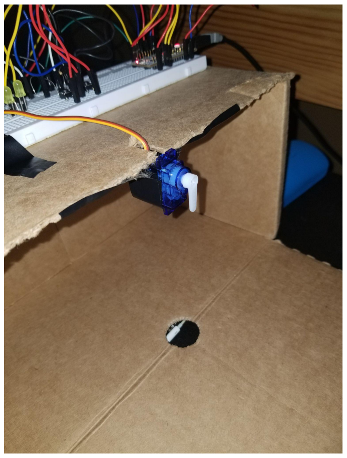


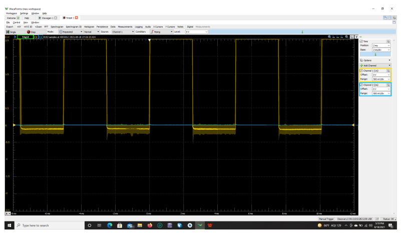
FSM Diagram



Background

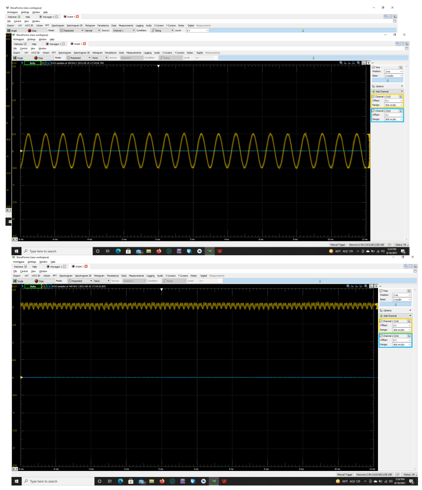
Servo





A servo is a DC motor that is coupled with a positional sensor that places the arm in at the correct degree due to the Pulse Width Modulated signal given. The photos above captured with the AD2 oscilloscope show square waves with edge detection. When the correct "code" is pressed and the servo is rotated from 90 degrees to 0 degrees, the duty cycle increases and the edge of the wave goes from 1.3ms to 2.3ms.

Piezo Buzzer



A piezo buzzer produces an alarm noise due to the fluctuation of the ceramic/metal disk when a current is pushed through it. The frequency of the current is what determines the sound output. This frequency is referred to as the resonance frequency. The pictures above show the voltage spike when the buttons are pressed for the correct "code" with the AD2 oscilloscope. The wave of the first picture atop to the right shows a peak amplitude of 0.5V and a frequency of 1kHz due to the wave period of 1ms. The picture below to the right shows a peak amplitude of 0.1V, but with a frequency of 10kHz due to the wave period of .1ms.

Results

Test Cases

Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
Check servo with correct codes	1. Press all three buttons 2. See all three LEDs light 3. Piezo Buzzer emits sound	Servo should move to the correct angle of 0 degrees from 90 degrees	As expected	Pass

Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
Check servo with incorrect codes	1. Press less than three buttons 2. See less than three LEDs light 3. Piezo Buzzer does not emit sound	Servo does not move	As expected	Fail

Conclusion

This project was originally going to be completely different. The end result was supposed to be a very simplified synthezier utilizing the PIC24F Microchip, potentiometers, buttons, and a speaker, but we could not find enough external resources to facilitate that endeavor. Ultimately, we discarded the PIC24F Microchip and replaced it with the superior STM32L432KC. Considering our timeframe and novice knowledge on microprocessors, we decided to simplify our project and implement older ideas from previous lab teachings. Before finding an external resource to help with the servo, we used the Raspberry Pi as the servo controller and were successful with the servo functioning, but faltered on the connection from the STM32L432KC to the Raspberry Pi for proper signaling. As a last ditch effort, we were going to implement a DC motor in lieu of the servo to showcase proper I/O processing. Unfortunately, we could not get the DC motor to effectively turn on and off when a button was pressed. This was later concluded to be the result of bad wire splicing from the DC motor to jumper cables. Thankfully, the servo, due to serendipity, was implemented and the Electronic Safe lived up to its name properly. Before the piezo buzzer, we tried using a speaker with another GPIO pin sending a PWM signal to generate a sound, but failed in the execution. We briefly looked into the STM32L432KC’s DAC GPIO pin, but decided to scrape the idea altogether and focus on the simpler piezo buzzer. Overall, the final product is everything we planned for and are very ecstatic that all the components worked properly.

References

https://www.engineersgarage.com/interfacing-servo-motor-with-stm32/

-This link aided in the implementation of the servo.

https://www.americanpiezo.com/standard-products/buzzers.html

-This link aided with knowledge regarding the Piezo Buzzer

https://www.electrical4u.net/electrical/working-of-servomotor/

-This link aided with knowledge regarding a servo motor’s functionality

https://learn.sparkfun.com/tutorials/pulse-width-modulation/all

-This link aided with knowledge regarding the PWM