Matthew Horotn

EE380 Projects Lab

Project 1 Report

**Problem Description**

Create an electronic lock with a 6-digit passcode that can open a solenoid and provide visual confirmation of whether the code was entered correctly or incorrectly.

**Design**

Using the number pad and servo provided, and the STM32L432 Nucleo board implement the lock.

Power

I wanted all power to be able to be sourced from the Nucleo board. This was not feasible as the motor’s CCA’s are way higher than what a Nucleo device can provide. So an external MOSFET and power supply is required to drive the motor. Other than that all power was able to be provided by the onboard voltage regulator. The MOSFET should be a PMOS device driven into saturation when driven low by the Nucleo, probably in open drain configuration and with a pull up resistor to 5v. It’s Source-Drain current rating must be greater than the current draw of the servo. The stock room ran out of motors so I was unable to implement this portion of the project.

Code

The code consists of a basic state machine implemented in C, compiled with the ARM 6 MDK compiler for uVision Keil.

Global variables

array used to map row column input to characters

success condition

current index in pin number

integer array with pin number in it

Main

Initialization function calls

Register initialization

While loop containing calls to numberpad scanning function and pin number check function

Conditional for completion of pin number successfully and function for LED success on

Return error main exits

Functions

Numpad get key pressed

Process key entry

Please see Attached Code in appendix A

**Theoretical Results**

Two electrical systems needed calculations. The pull up resistors on the numberpad columns and the servo system.

The servo system is pending design until we have enough servos for me to use one. As described earlier I imagine a PMOS with a pull up resistor on its gate being controlled by an open drain GPIO pin should be enough. There’s no way the nucleo can source enough amps to overcome the initial amperage draw of the servo. An external power supply should be used with additional care and protection to isolate the grounds of the control system and the servo driver circuit.

For the keypad pull up resistor calculations I just used the reference calculations from the book design. (Yifeng, p365, Embedded Systems with Cortex M)

**Simulation Results**

My mapping of characters from the numberpad scan was not as clean as I’d hoped. The numberpad scan function follows the traditional design. All columns are pulled high by external pull up resistors and all columns are open drain. When a button is pressed its row goes high and column goes low. Then the columns and rows are scanned to pinpoint the row/column combination or button pressed.

I implemented a filter to prevent the idle character from constantly filling my pin number comparison buffer. As long as the next character entered is different from the last character entered AND its not the idle character, push the new character into the queue.

My filter needed more conditions to make sure that only one button entry was admitted into it per press. The debugger (simulation ?) highlighted this problem.

**Experimental Design**

See the circuit schematic in Appendix B and code in Appendix A

**Measurement Results**

See the circuit schematic in Appendix B and code in Appendix A

**Conclusion**

This was a fun project, I should’ve allocated more than 2 days to it though because I was unable to meet the deadline for physical testing. In total I’d say 2 hours were spent designing and implementing the physical system and 12 hours the software.