Abdullah Aljandali

EE 380 - Keypad Doorlock

2/28/2019

Problem Description:

This product uses a keypad, LEDs and a Solenoid to represent a door lock. The user inputs a password using a keypad. If the password is correct, the solenoid is released for 10 seconds and a green LED blinks 5 times. Otherwise, if the password is wrong, a red LED blinks 5 times and the user is allowed to enter another code. The system is controlled using the STM32L432KC microcontroller.

Project Design:

A. Hardware Details

Components used:

- STM32L432KC Microcontroller - x1
- Solenoid -x1
- Keypad -x1
- 12v source -x1
- 2200 ohms resistor -x4
- LED -x4

Hardware Design:

Only port A was used in this project, and all the references I will make about port pins is referring to port A pins. The project will require a 3x4 keypad, which has 3 pins for the columns and 4 pins for the rows. I will connect the columns pins as inputs to port pins 4,5 and 6 because these are only-input pins, leaving space to use the others for the output. The input pins will have an external pull up to 3.3v using 2.2kΩ resistors. Since I will use an external pull up, I will leave the input pins to default no pullup no pulldown. This summarises all the inputs use for the project.

Secondly, we will use 9 pins for the output. The rows pins will be set as the output pins, and will then be connected to the port pins 0,1,2,3 as open drains. The project will also use 4 LEDS: blue for button click, red for wrong code, green for correct code, and yellow for wait.

These will be connected to port pins 7,8,9,10 respectively. Pin 11 will also be used for the solenoid as discussed next.

This project requires a solenoid to turn on for 10 seconds when the correct code is entered. Though, the solenoid requires 12v and 0.5A which cannot be supplied from the microcontroller board. Therefore, we need to use an alternative method. We will use a 12v voltage source as our source for the solenoid. Though, since we need it to be controlled by the microcontroller, we need to use some sort of switch controlled by the microcontroller for the 12v source. Since MOSFETs are current driven, and we have BS170 MOSFETs readily available, I will use a BS170 in this case. Also, BS170 can handle 0.5A of current, which will be enough for the solenoid. We will connect port pin 11 to the gate which will supply 3.3v when high. There will also be a pulldown 2.2kΩ resistor to allow the right amount of current. The gate will be connected to one side of the solenoid, and the other side of the solenoid will be connected to to power. The source will be connected to ground. The ground of the power source has to be connected to the ground of the microcontroller.

B. Software Design:

The basic idea is for the software to wait for button clicks and store those keys in an array. Once there has been 6 clicks, the array will be compared to the passcode. If they match, the password is correct and the green LED lights up and the solenoid turn on. If the password is wrong, the red LED lights up.

The way the key clicks are scanned is by first checking what column is pressed (which pin is = 0). If there is no column pressed, an arbitrary value is returned indicating no button click. If one column is = 0, the function will then check which row it is. To do that, we loop over the output pins and give 0 to one of them and 1s to the rest. We then check which of these rows keep the column we found before as a 0. Once we have found the col and the row, we can find which key has been pressed.

As for the way we will calculate the wait time for the LEDs and such. We can use for loops and find out how many loops it takes for a millisecond. Once we find that, we can multiply it and use it for our delays. A better method to do this would be by using timers.

Theoretical Results:

Based on the design I made, I expect the project to behave like what the problem statement states. In other words, I expect it to blink the green LED and turn on the solenoid when the correct password is inputted and to blink the yellow LED for 10 seconds. Otherwise, I expect it to blink the red LED 5 times. I also expect it to be sensitive with minimal errors because of the "unpress" algorithm discussed in the "Design" section.

Experimental Design:

To test the keypad, I first hard coded the password to be: {1,2,3,4,5,6}, and performed this test:

1. Connect the Microcontroller to the computer through USB
2. Enter in the password {1,2,3,4,5,6}
3. For every click, the blue LED should light up. If it doesn't then there is an issue.
4. Once you enter in the code, check to make sure the green LED is blinking and the solenoid turned on
5. Use a stopwatch to make sure there solenoid has been open for 10 seconds
6. Now try entering any wrong password
7. Check to make sure that the red LED blinks for 5 times
8. Now repeat the process multiple times and count how many times it succeeds

Measurement Results:

The product overall behaved very closely as expected. There were a few unexpected errors, which I fixed by making some changes. The first issue I encountered was that the keypad sensitivity wasn't perfect. Sometimes, even though the correct password is inputted, the red LED blinks. To fix this issue I made some changes to the code. Now, the program only stores characters in the array if the button has been "unpressed". For example, if I press '1', '1' is stored in the input array and a flag is set to true. The program won't store any more characters until the flag is set back to false. For the flag to be set back to false, the keypad-scan functions has to have return the arbitrary character that refers to "no key pressed" 100 times.

Another issue was with the MOSFET heating up, and I realized that it is because BS170's only handle 0.5A of current and the Solenoid was at least taking that much current. Therefore, I put another BS170 in parallel to the other one, which resulted fixed the issue.

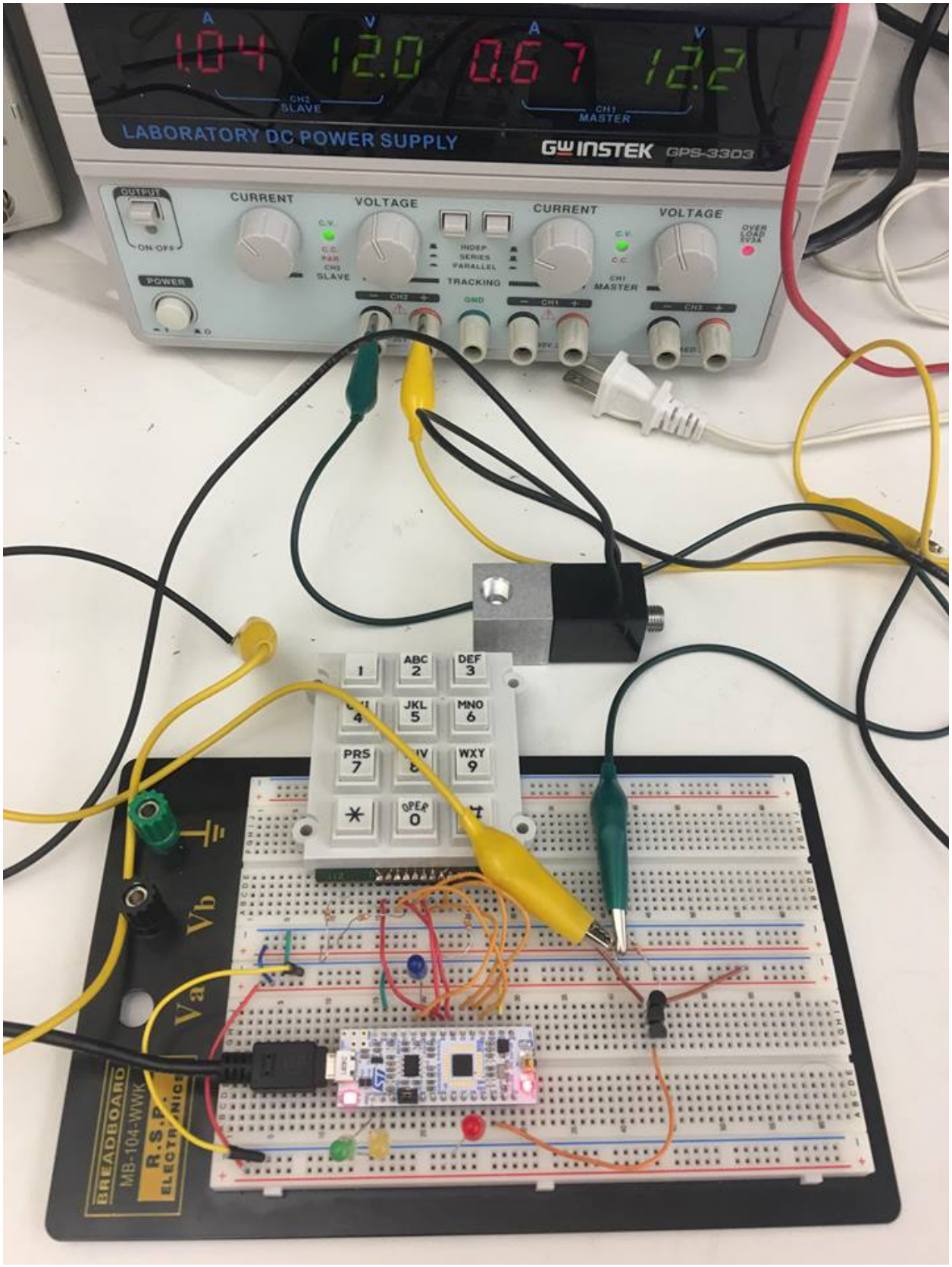
Conclusion:

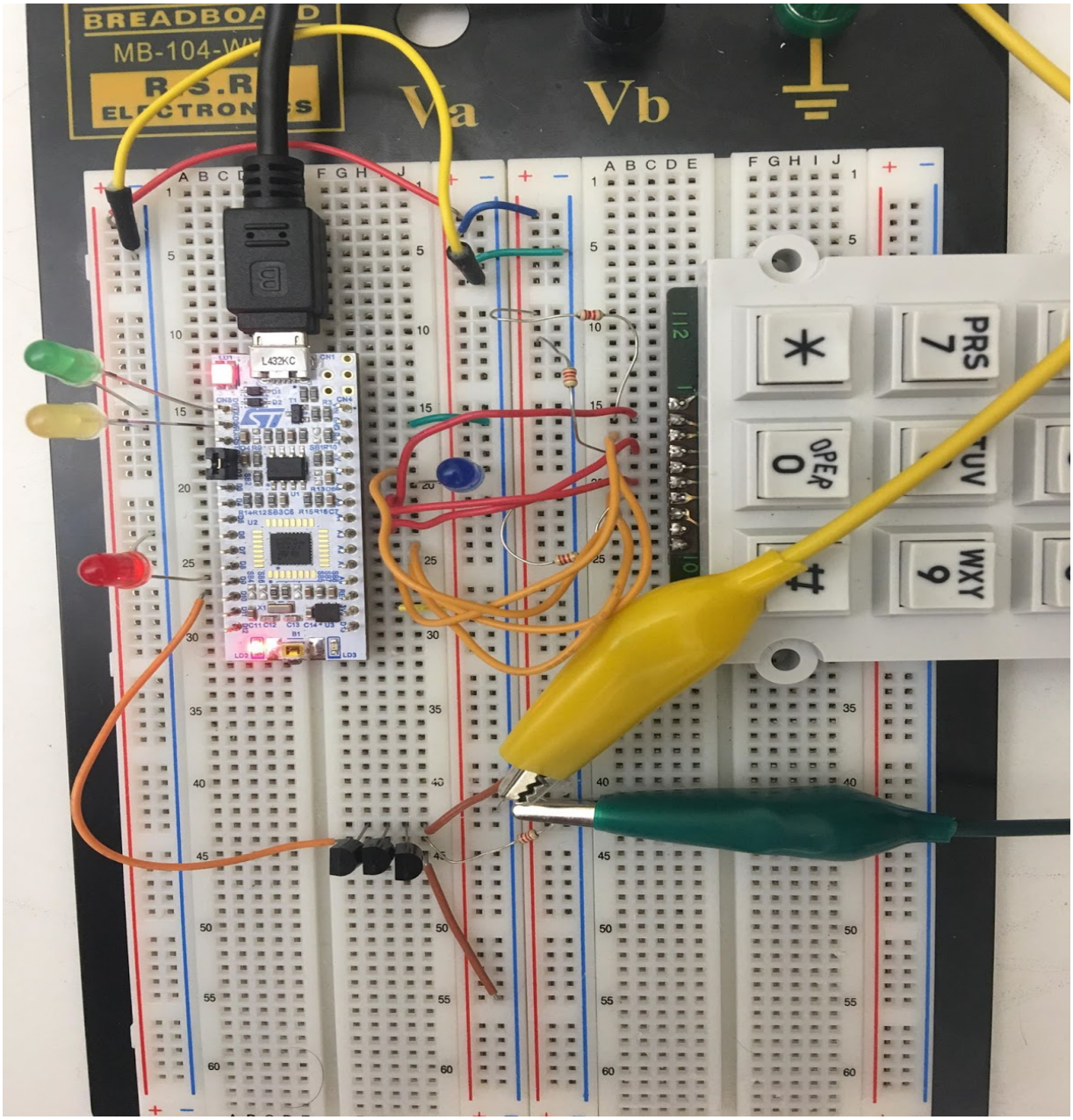
The project satisfies the minimal requirements with minimal error. There are also novel features that have been added (as discussed later). The project as whole has been a good learning experience as it makes sure the student understand multiple areas well. First of all, having to use a MOSFET to turn on the solenoid needs understanding of knowledge. Also, a big part of the project was coding, which required good coding and problem solving skills. Not to mention, we also needed to figure out how to use new devices we have never seen (keypad, solenoid), which shows that we will do a good job in the real world.

Novel Features

The project included three additional features that are not captured in the problem statement:

1) Password Change: It is possible for the user to change the password in this product. They can do that by clicking the * button six times. The yellow LED blinks 3 times to indicate that they can enter their password. Once they have entered in their password, the yellow LED blinks again indicating that they should confirm their password. If the two entries match, the green LED blinks and the password is changed. Otherwise, if the two entries don't match, the red LED blinks and the password remains the same. If they attempt to use '*' as part of their password, the red LED blinks indicating an invalid password, that's because the key '*' is used for changing the password.
2) Button Click Confirmation: Every time a key is clicked, a blue LED blinks indicating that the click has been captured. This helps the user know if their click was captured or not, which helps avoiding issues that lead to them having to re-enter the password multiple times.
3) Wait indicator: The project also uses a yellow LED that acts as an indication for the user to wait. It is used in multiple occasions. For example, if they enter in a correct password, after the door unlocks, the yellow LED blinks indicating how much time they have left before the door locks again.


1.04
12.0
0.67
12.2
LABORATORY DC POWER SUPPLY
GW INSTEK GPS-3303
CURRENT
VOLTAGE
CURRENT
VOLTAGE
TRACKING
BREADBOARD
MB-104-WWK


BREADBOARD
MB-104-W
Va
Vb
L432KC
PRS
7
OPER
0
WXY
9