BRAC UNIVERSITY

Inspiring Excellence

Department of Electrical and Electronic Engineering

Project Report Submission

Semester: Fall 2021
Course Code: EEE-366
Course Title: **MICROPROCESSOR AND INTERFACING LABORATORY**

Project Title: PASSWORD PROTECTED DOOR LOCK

Group No: 4

Group members:

SL.	***Name***	***ID***
1.	*Lazib Sharar Shaiok*	*19121141*
2.	*Rajia Jannat Liya*	*18121038*
3.	*Mohammad Tasnimul Hasan*	*19121063*

Date of Submission: 31st December,2021

Abstract

In this project a different iteration of the popular password protected lock was designed and implemented using Proteus and CodeVision AVR.The key criteria were that the door should be locked if the passwords don't match and unlock if they do.Furthermore, the user should be able to set and reset the password. Both these key criteria were fulfilled and further features were added to the system to make it more robust and dynamic.

I. Introduction

In the 21st century, petty crime is rising in all parts of the world and security is a big concern for all socioeconomic classes. Thus in this project, we aim to develop a password-protected door lock that is resistant to brute force lock-picking and technical hacking.

II. Scopes & objective

In this project assigned in Microprocessor and Interfacing Laboratory (EEE-366) we are to design a password-protected door lock for a door with an anti-theft protection system. The key requirements of this project are the following:

- The door will only open if the password matches. If it does not then the door shall remain locked.
- Users can both set and reset the password of the door lock i.e that the password is NOT hard-coded into the program.

More features were added to ensure total protection of users and dynamicity which will be described later in this report.

Due to the ongoing pandemic, the project was implemented in software only. However, hardware implementation of this project could be very easily done.

III. Apparatus & software

Apparatus:

- AT Mega 32
- SPST Push Button
- Resistor
- DC Operated Buzzer
- Keypad-Phone
- 16 × 2 Alphanumeric LCD
- Servo Motor
- Power Supply
- Oscilloscope (To monitor the PWM wave)

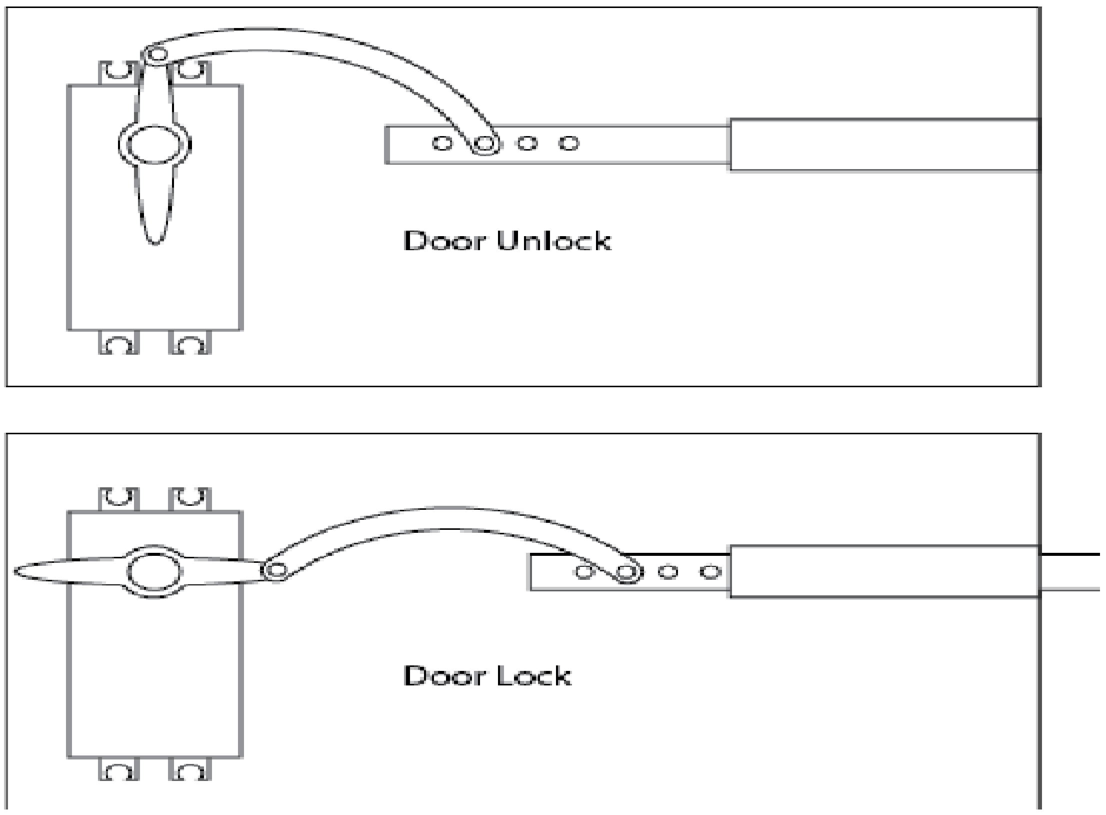
Software:

- CodeVision AVR
- Proteus 8 Professional

IV. Project specification

The aim of this project is to provide security to the user. Hence a lot of features were added as such.

- The door will only open if the password matches. If it does not then the door shall remain locked. The door-lock mechanism was achieved by means of a servo-motor's rotor being connected with an actual lock. If the servo motor's rotor moves from -90 to 90 degrees (clockwise) then the movement of the rotor will cause the door to unlock. If the servo motor's rotor moves from 90 to -90 degrees (anti-clockwise) then the movement of the rotor will cause the door to lock.



A lock using a servo motor to lock and unlock.

- Another main aim of this project is to make the password setting dynamic. That is that the user should be able to set and reset the password. This feature was added by using two keypads instead of one and some nifty coding. The idea is that on both sides of the door there will be a keypad. There are two faces in each door, one is inward-facing (inside the home/vault) and another is outward-facing(outside the home/vault).

  Now when the user first SETS up the lock he will use the keypad-A which is on the inward face of the door (inside the home/vault) to do so. Then from then on whenever he will try to get into the house/vault he will have to type the password in keypad-B which is on the outward face of the door (outside the home/vault) to do so. If he decides to RESET the password again, he will have to at first type the old password using keypad-B then when the door unlocks he will press a switch also on the inward face of the door and then type the new password using keypad-A.

  In short keypad-A and a switch are used to SET and RESET the password. They are on the inward face of the door and keypad-B is on the outward face of the door.

Keypads like this will be placed on both sides of the door.

- If an incorrect password is typed three times straight a buzzer will sound.
- The LCD display will display “ * “ whenever a password is being typed. To prevent anyone else from viewing the password.
- The LCD display guides the user in setting, resetting, and entering the password.
- # character of both the keypads can be used to lock the door again after unlocking it.
- A 4-digit password is taken only to make it practical. But theoretically, a 16 digit password could be used but this is impractical.

- Numerical passwords only. (Other characters could also be used if a different keypad were to be used.)
- It is worth noting that the lock is only as secure from the brute force as the torque the servo motor can exert. If servo motors like SG-90, the ones we used in the lab are used, then an adult should be able to rotate the rotor and unlock the doors small servos like those only exert 2.5kg/cm torque. However, larger servos like SPT Servo SPT5435LV-180 35kg/cm torque.

V. Non-technical constraints

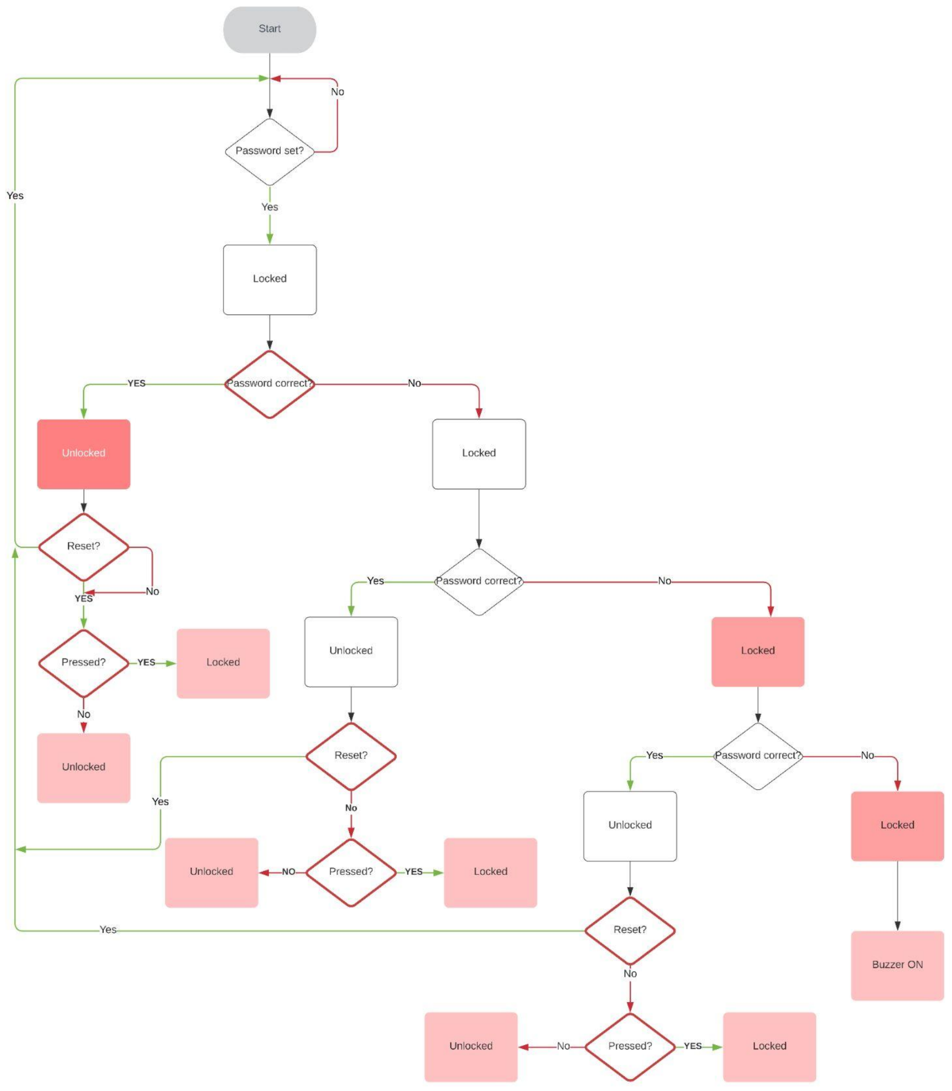
Security is a major issue in the 21st century as break-ins are getting common day by day.No matter the socio-economic background everyone wants to feel a sense of security. However, the prices of security systems in the market prevent lower-income families from buying them. The system designed in this project is quite cost-effective and can be accessed by even lower-income families.

Moreover, the set password is being saved on the microcontroller, which will be present on location, instead of a distant server thus data-leaks can't be possible with this system.

VI. Methodology

Flowchart:

This section will explain the working of the physical system by means of an algorithm.



Design Process:

This section will explain how the entire design process was done. For further elaboration, one can view the source code found at the end of this report.

Challenge #1 (To store the data being pressed in the keypads):

In our system, there are two keypads. Keypad-A is used to set and reset the password and Keypad-B is used to take in the attempted password.

Using the user-defined **keypadscan() function** we took in the input to the Keypad-B and stored the attempted password in an unsigned char array called password[]. This array was 4 chars long as our password will also be 4 chars long. The row scanning technique was utilized to see which key was being pressed. Now for example, if the key "1" was being pressed then **password[i]=1** was written to ensure that the char "1" was stored within our array password[] (i is just an integer used to keep track of how many times the key was pressed/ how many chars of the password array got filled) . Using keypadscan() the password being typed was also displayed on the LCD as "*". The code lcd_gotoxy(i, 0); lcd_putsf("*"); was utilized.

keypadscan2() was the corresponding user-dfeined function to Keypad-A and also utilized the same logic.

So in short **keypadscan()** and **keypadscan2()** utilized the row scanning technique to not only keep track of which keys were being pressed but also fill two unsigned char arrays and display "*" symbols on the LCD.

Challenge #2 (To match the attempted password with the set password. If they match take appropriate action and if they don't match take appropriate action):

For this, a user-defined function **check()** was written.In this function using an if condition each element of the array password (where we stored the attempted password) and the array password_enter (where we stored the set password) was checked. If they all matched then a message "password correct. lock opened" was displayed on the LCD using **password_correct()** function. Also, the servo was rotated from -90 to 90 using the **rotate_minus_to_ninety()** function.So the lock opens.

If the array elements did not match then the message "password incorrect" was displayed on the lcd using **password_incorrect()** function. Also, the servo was rotated from 90 to -90 using the **rotate_ninety_to_minus()** function.So the lock remains locked.

Challenge #3 (Lock and Unlock):

Whenever the **rotate_minus_to_ninety()** function was called the servo was rotated from -90 to 90 degrees(clockwise).Thus the lock opens.

Whenever the **rotate_ninety_to_minus()** function was called the servo was rotated from 90 to -90 degrees(anti-clockwise).Thus the lock closes.

To control the servo motor a PWM was generated using the PD5/OC1A pin. The 5% duty cycle wave was corresponding to the -90 degree position and the 10% duty cycle wave was corresponding to the +90 degree position. Total Pulse-Width=20ms and min on time=1ms and max on time=2ms.

According to the above specifications TCCR1, ICCR1, and OCR1A register values were set.

Challenge #4 (If the user gets the password wrong letting him try multiple times):

At first what was happening was that if the user typed in the attempted password as “1 2 3 4” then in the array password[4] was being stored permanently. So new entry was being taken if the user got the password wrong the first time.

However by setting **i=0** whenever the **check()** function was called regardless of whether the user got the password right or wrong this problem was solved. As the next time **keypad_scan()** was called the new entries just overwrote the old entries.

Challenge #5 (Forcing the user to SET the password the first time and allowing for the RESET capability):

By placing the **keypadscan2()** function and a few LCD calls outside the while loop of the main function. We basically ensured that the user SETs the password first.

Now for the reset feature, one must understand keypad-A and the switch of INT0 are placed on the inward face of the door/ inside the vault. So intruders don’t have access to keypad-A or switch of INT0.

One can only get access to those items if the door is already unlocked, which is when the user is setting it up for the first time and whenever he opens the door.

To reset the user must press the switch then type in the new password for keypad-A.

So whenever the switch is pressed ISR of INT0 is called. The value of k is set to 0. Allowing us to rewrite new chars over the old chars in the array where the previously set password was stored. **keypadscan2()** function was called again. Thus readying our microcontroller to take in new entries from keypad-A.

#asm(“sei”) was written in each function. This was done to ensure that password reset has the most priority. This is necessary as during the time of the simulation we don’t know where our active code is .

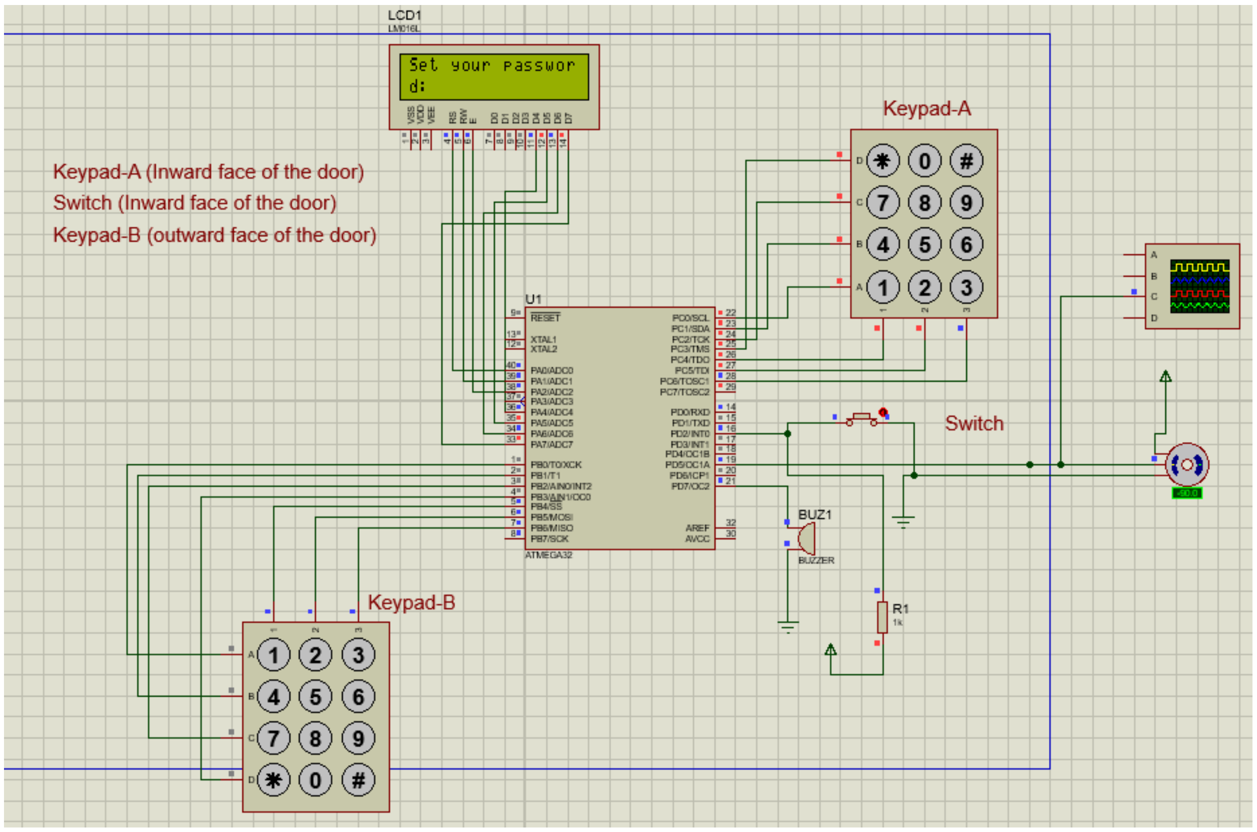
Challenge #6 (Buzzing the buzzer if three subsequent incorrect passwords are entered):

Whenever the **password_incorrect()** function gets called an integer is incremented and the **buzzer()** function is called. In the **buzzer()** function there is an if condition which states that the if conditions will run when the integer reaches the value three or else it will not. If the **password_correct()** is called **incorrect=0 i**s set to ensure that three subsequent incorrect entires ring the buzzer not three incorrect entries since the time of the simulation.

VII. Design choices & alternatives

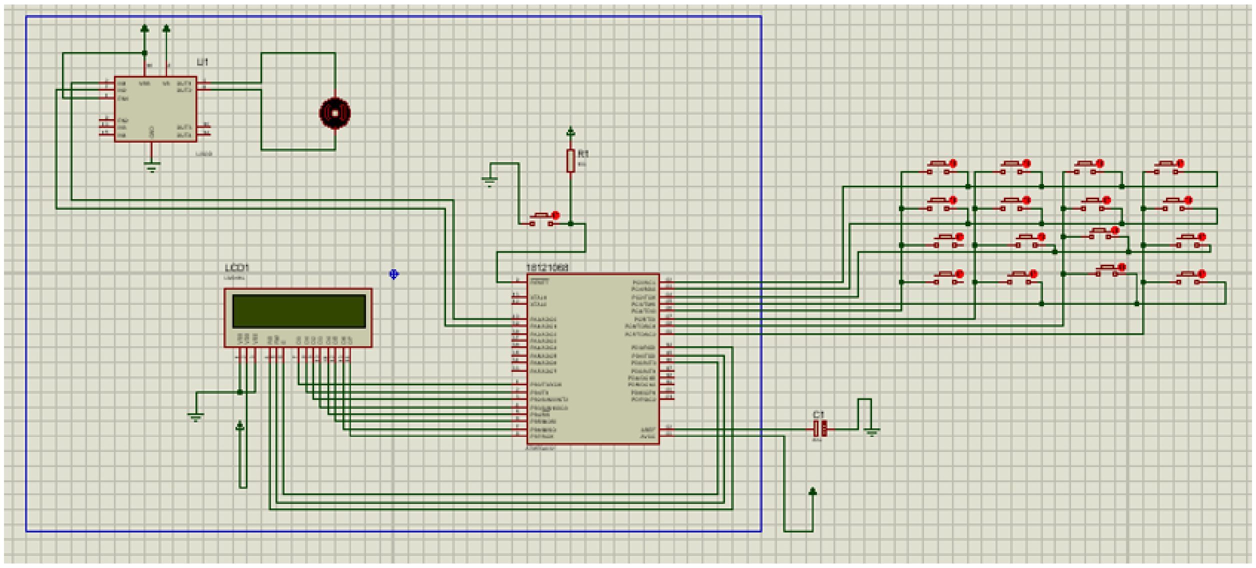
The must-have criteria of this project are that the door should unlock only when the attempted password matches the set password and remain locked if it does not. Also, the user should be able to set and reset the password dynamically,i.e that the set password should not be programmed into the code.
The following three designs all fulfill these minimum criteria.

A. Design approach 1:


LCD1
LM016L
Set your passwor
d:
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

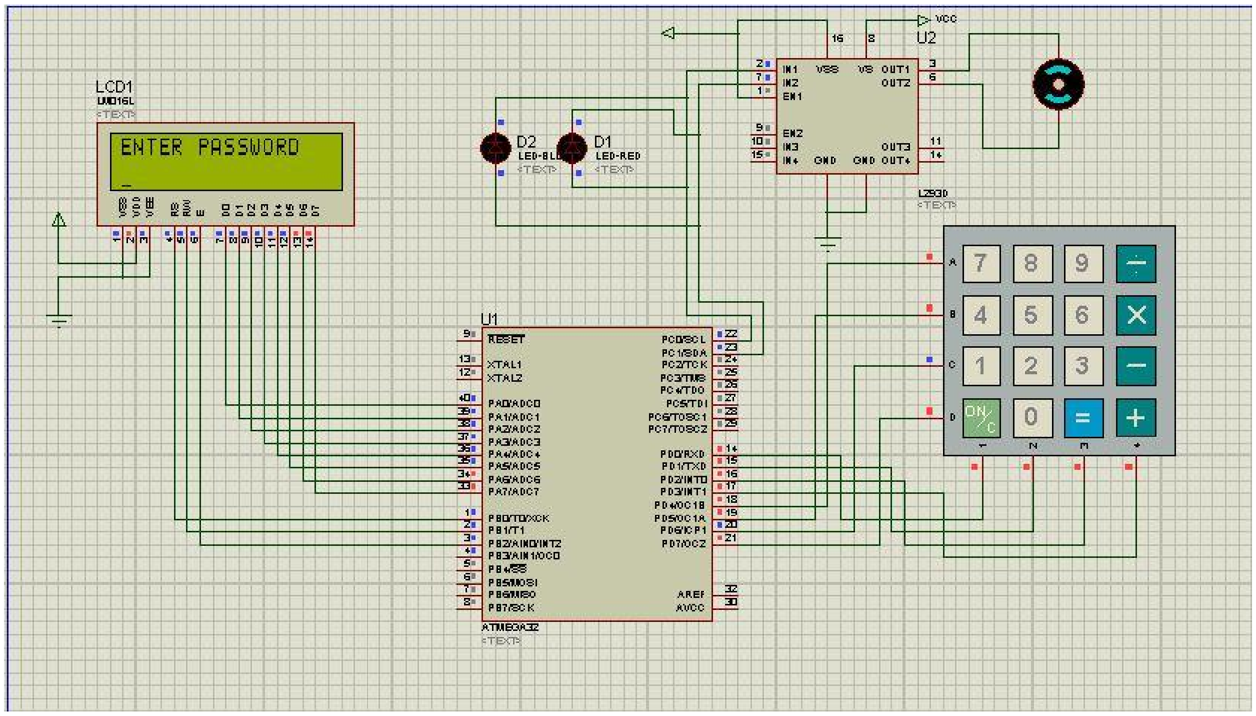
This system was designed and implemented in this project.
Keypad-A and Switch are on the inward face of the door and used for SET and RESET purposes.
Since they are on the inward face of the door, intruders don't have access to these components.
Keypad-B is on the outward face of the door and is used to take entry for the attempted password.
LCD display, Mobile-Keypad, Buzzer, and Servo Motor are also interfaced.

B. Design approach 2:



In this design approach push buttons are utilized instead of the built-in keyboards available in the Proteus. Moreover, instead of a servo motor, a DC motor interfaced using an L293D motor-driver is used. The reset button and reference voltage pins are also enabled.

C. Design approach 3:



In this design approach calculator keypad is used. Moreover, instead of a servo motor, a DC motor interfaced using an L293D motor-driver is used. Two LEDs are connected to the motor-driver to indicate clock-wise and anti-clockwise rotation of the DC Motor.

D. Justification of the best approach:

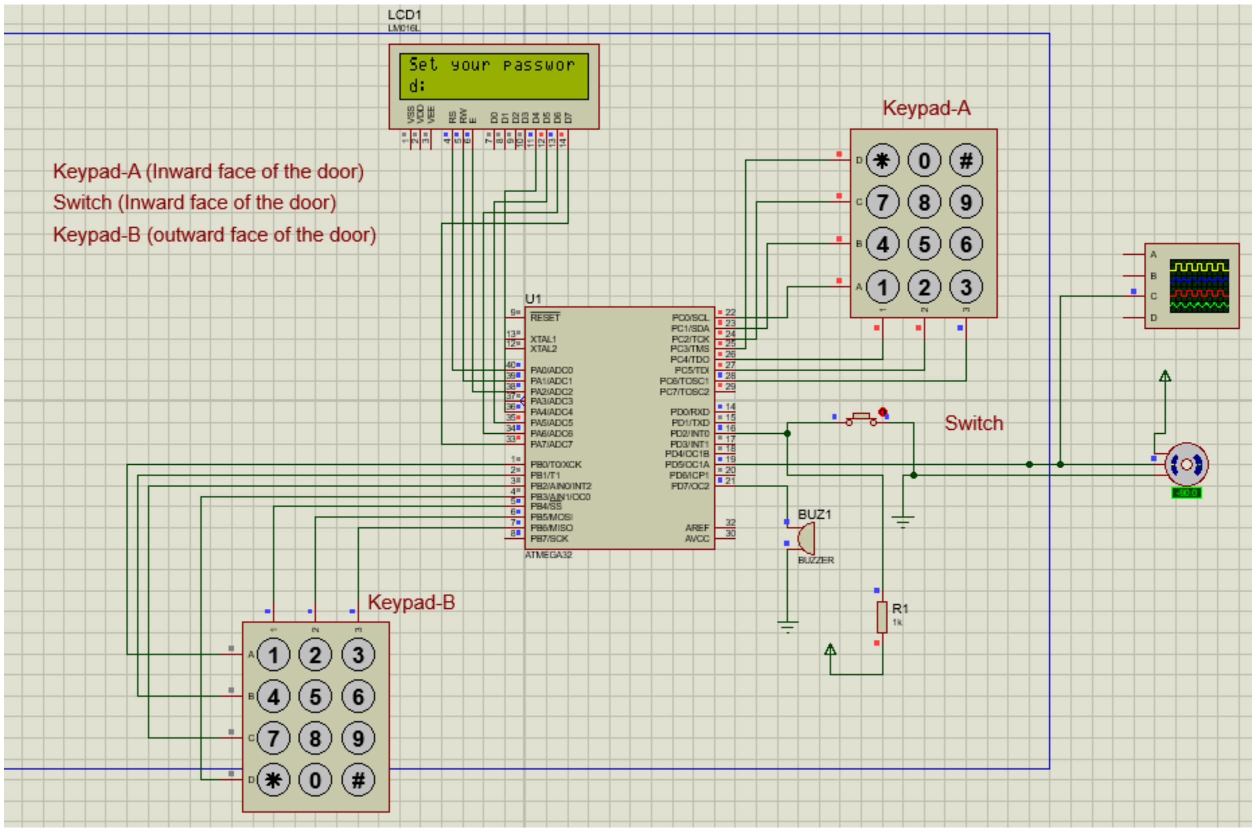
Design approach-1 is the best approach.

Design-1 made use of a servo motor whilst the other two designs made use of DC motors.
The servo motor is far superior to the DC motor for this particular application for two reasons.
The first reason is the DC Motor requires a motor driver to operate if interfaced with an ATMega32.(Because ATMega32 can't keep up with power requirements of DC Motor) This motor driver increases the cost of design-2 and 3 compared to design-1. Servo Motor requires no such motor drivers.
The second reason is that the servo motor provides much more precise rotation when compared to DC Motor. This is important because when using a lock with a motor we want -180 degrees to be exactly -180 degrees to ensure that the lock is completely shut.

Design-1 made use of two keypads and a switch. Keypad-A and the switch are used for SET and RESET purposes and are on the inward face of the door whilst Keypad-B is on the outward face of the door and is used for taking in the entry of attempted passwords.
This ensures that intruders outside can never have access to SET and RESET features as they don't have access to Keypad-A and the switch which are on the inside.
On the other hand, the other two designs use just one keypad for both SET and RESET purposes and for taking in entries for the attempted password. There is not even a switch to provide control on such features. This has the potential to be a security flaw as the intruder can use the lone keypad to enable SET and RESET features.

Furthermore, the use of two separate keypads for two separate functions allows for less wear and tear.

VIII. Circuit diagram


LCD1
LM016L
Set your passwor
d:
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

In this section the connections will be explained.

Keypad-A:
PC.4-6 is initialized as o/p pins as they are connected to the columns of the keypad.
PC.0-3 is initialized as i/p pins as they are connected to the rows of the keypad.

Keypad-B:
PB.4-6 is initialized as o/p pins as they are connected to the columns of the keypad.
P.C 0-3 is initialized as i/p pins as they are connected to the rows of the keypad.

LCD:
All the PORT-A pins are o/p pins. PA.0-2 to is connected to RS,RW and E (control of LCD) whilst PA.3-7 is connected to D4-D7(writing data to the LCD)

Switch:
PD.2 is initialized as i/p and is used as an interrupt which is INT0. PD3(INT1) could also have been used.

Buzzer:
PD.7 is initialized as an o/p and is used to drive the buzzer.

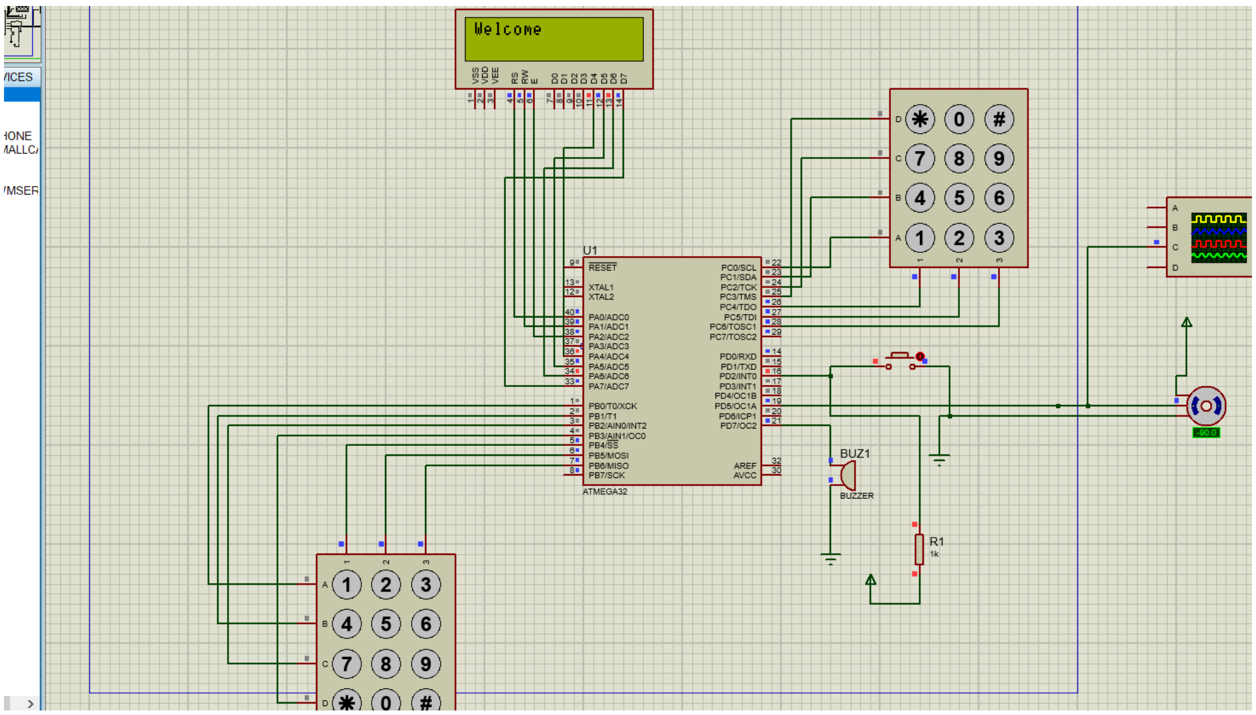
Servo Motor:

Since we are using a PWM to control the servo motor thus we have to use either PD5(OC1A) or PD4(OC1B). We went with PD5 and initialized it as o/p.

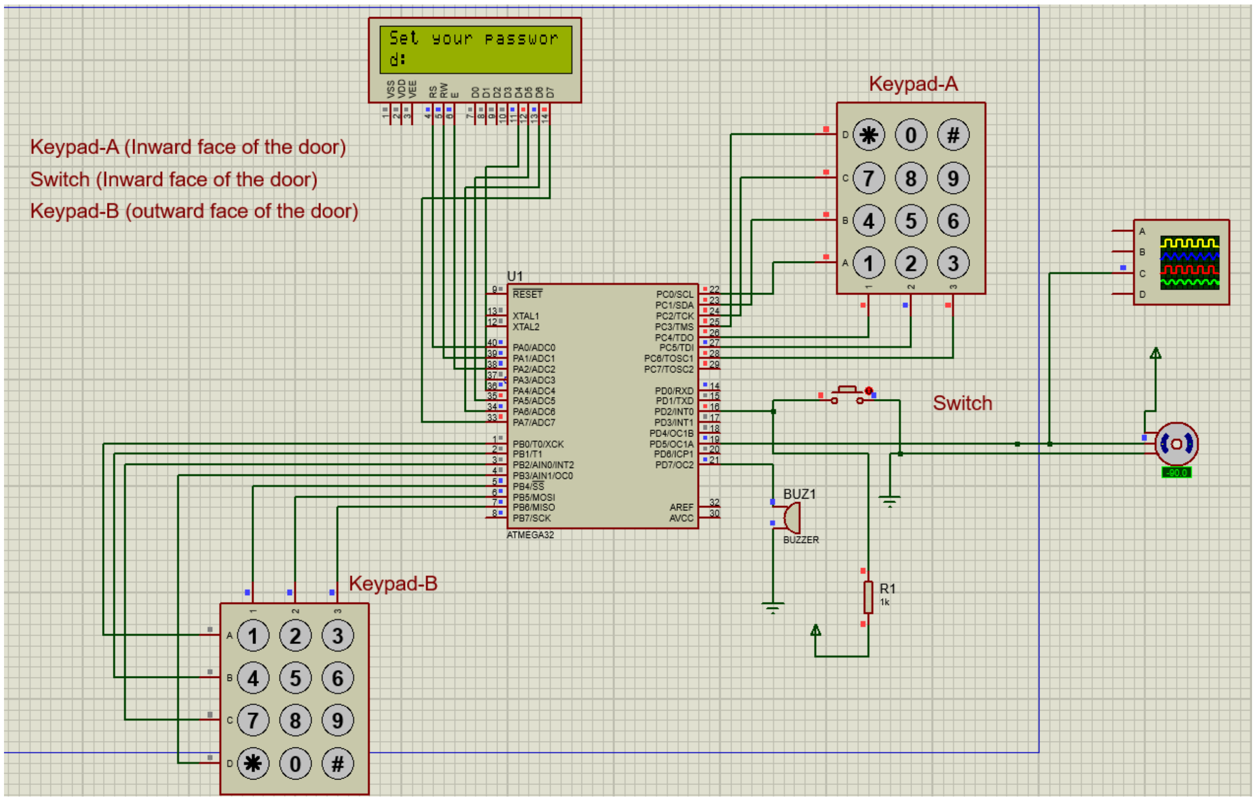
IX. *Results & discussion*

Results:

[In this section, we will try to explain the functionality of the entire system through images and text. However, things like the motor rotation, buzzer sound, and update of LCD display are very hard to visualize through text and image only for which please refer to the simulation video attached as a drive-link at the end of the report.]

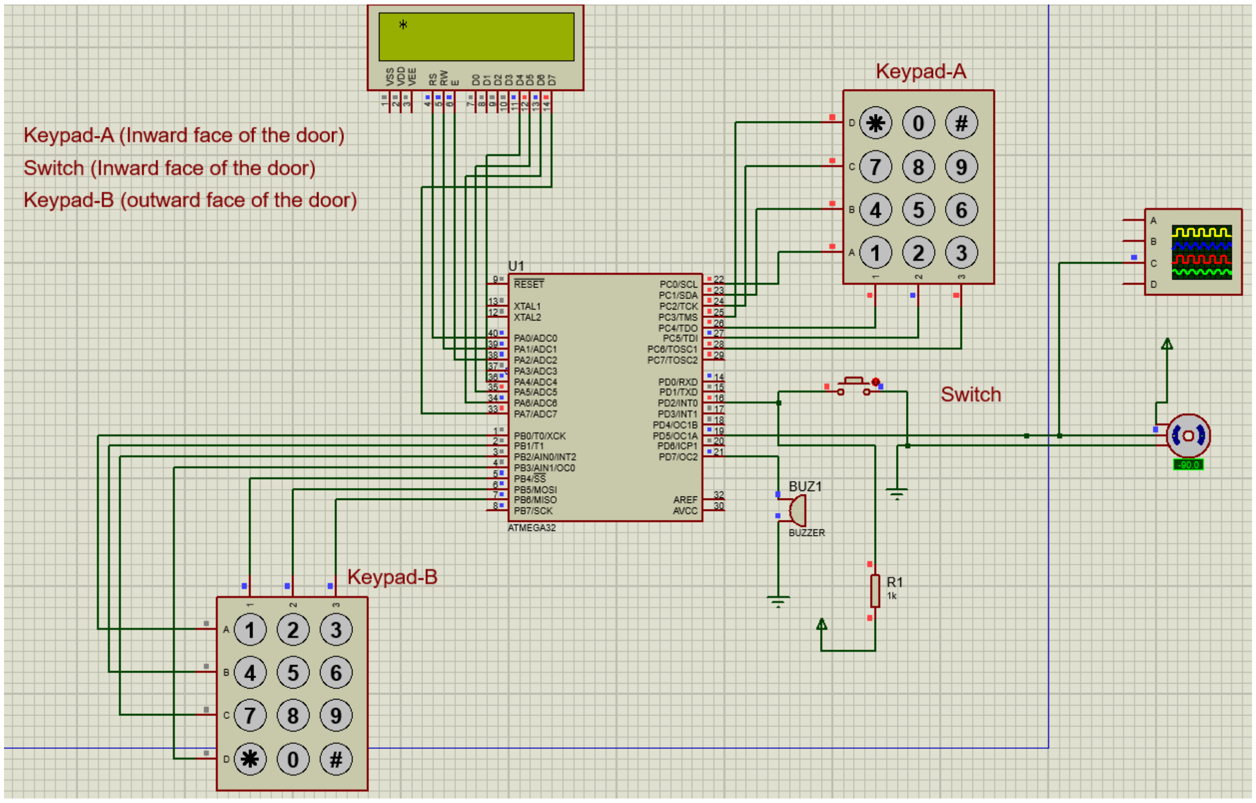


The user will be greeted with a welcome message when the simulation is run for the first time.

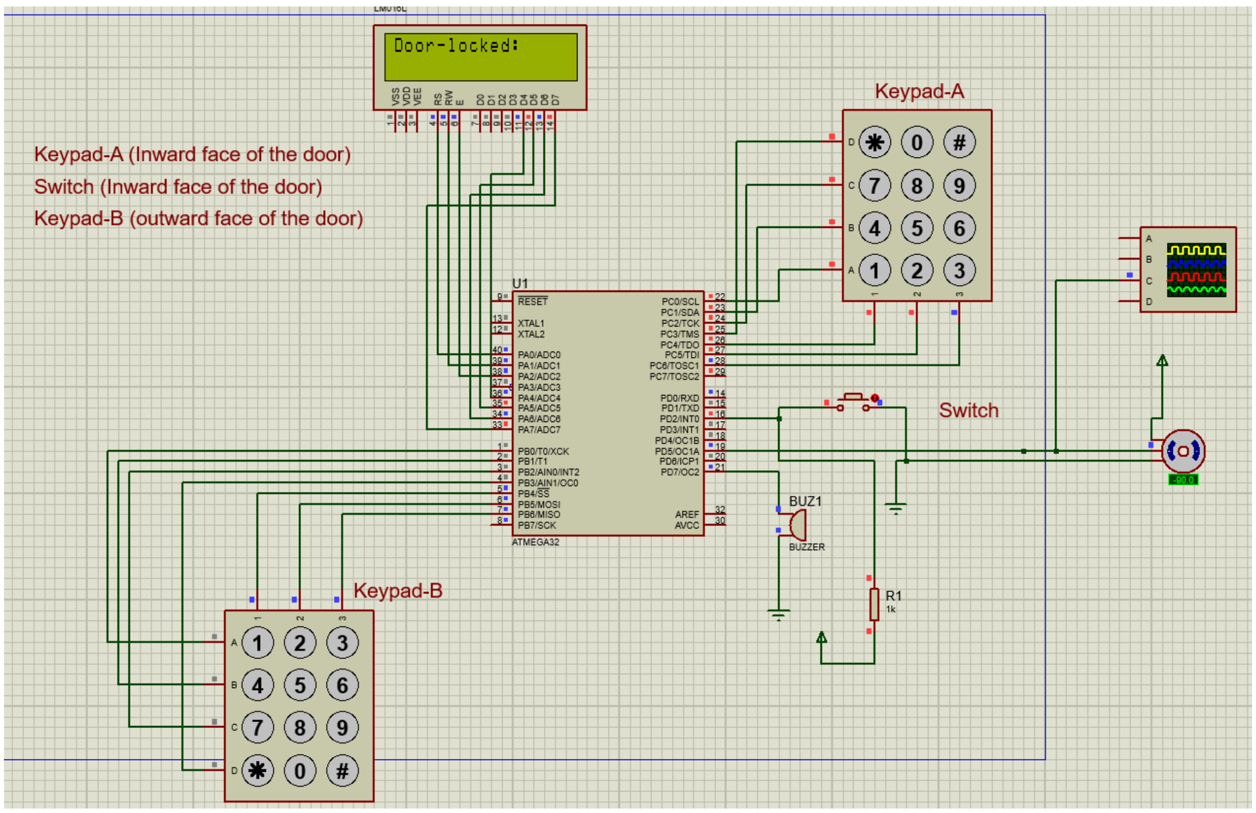


Set your passwor
d:
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

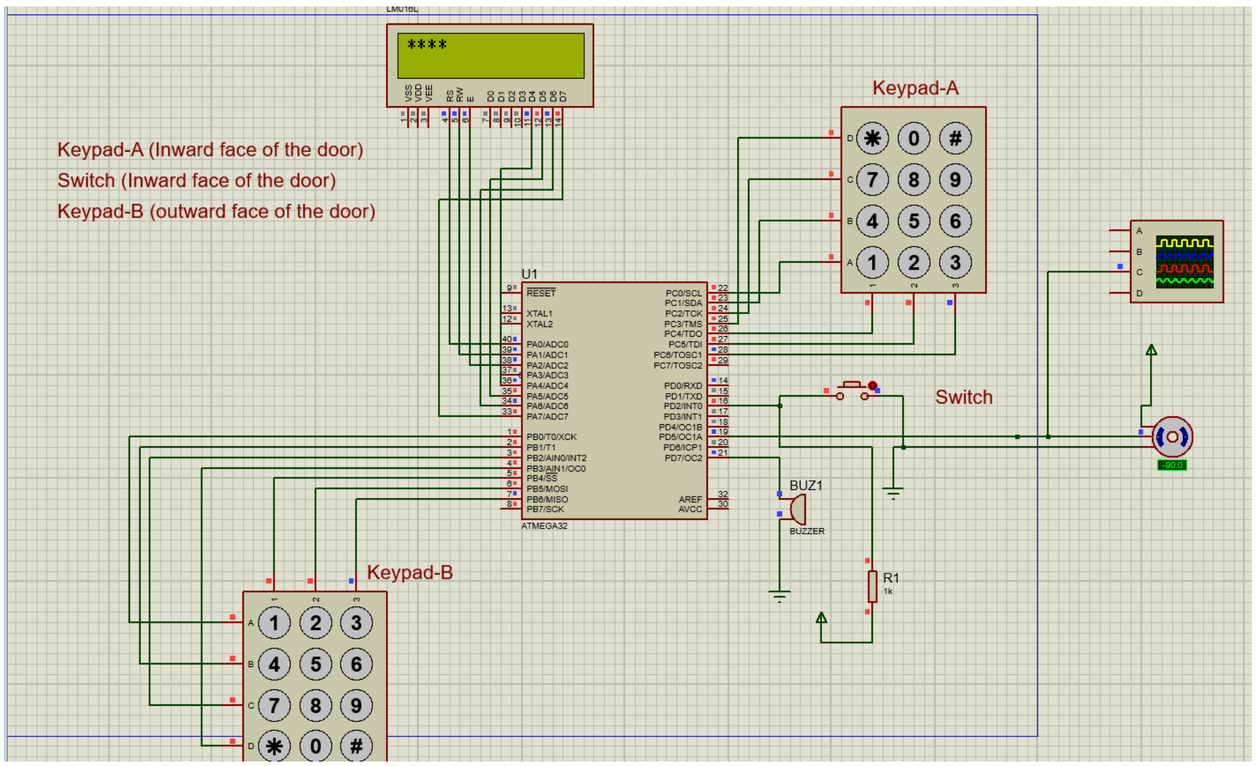
The user will be prompted to SET a password.


Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
Keypad-A
Keypad-B
Switch
U1
ATMEGA32
BUZ1
BUZZER
R1
1k

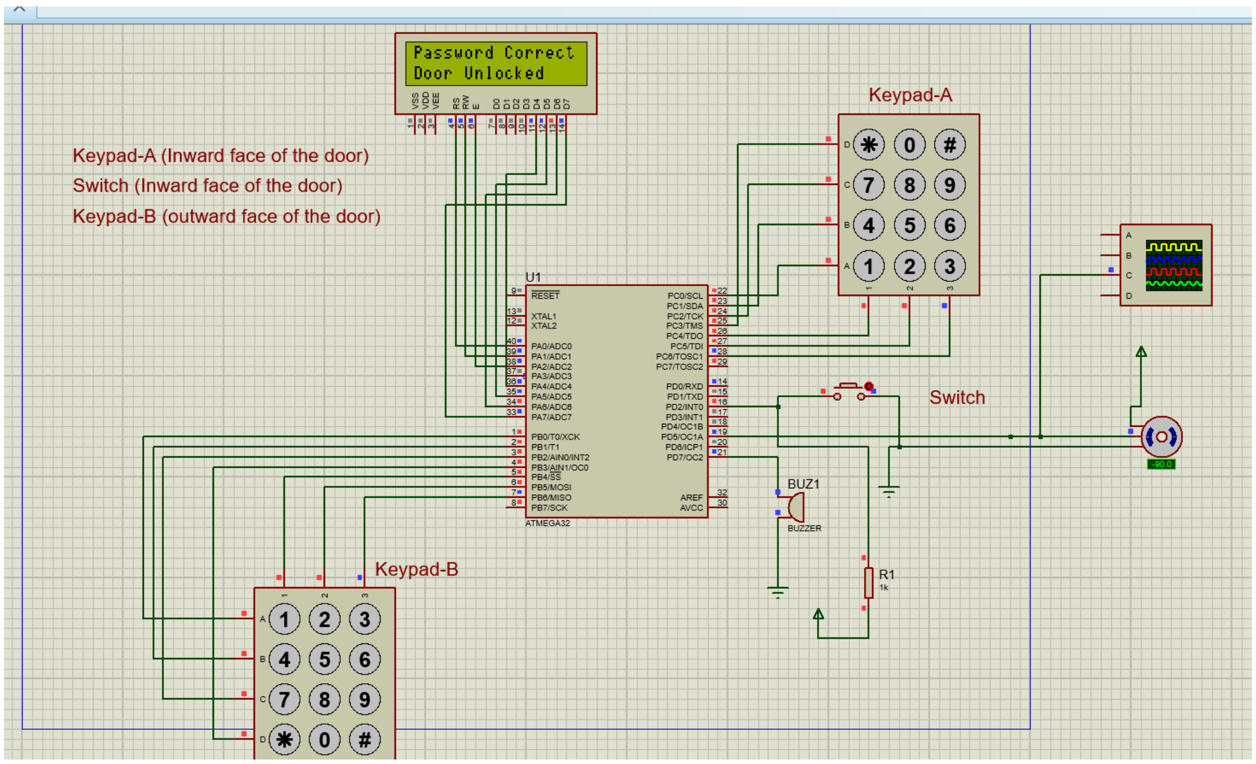
The user is now setting the password for the lock using keypad-A.

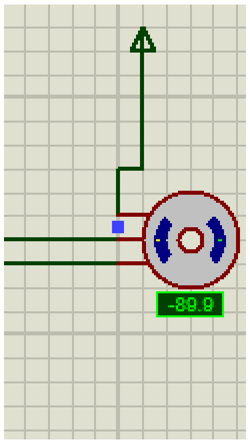

LM016L
Door-locked:
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

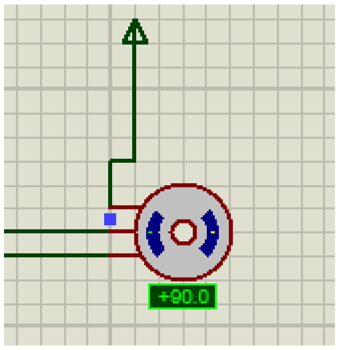
After the password has been set, the door will lock itself.


LM016L
****
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

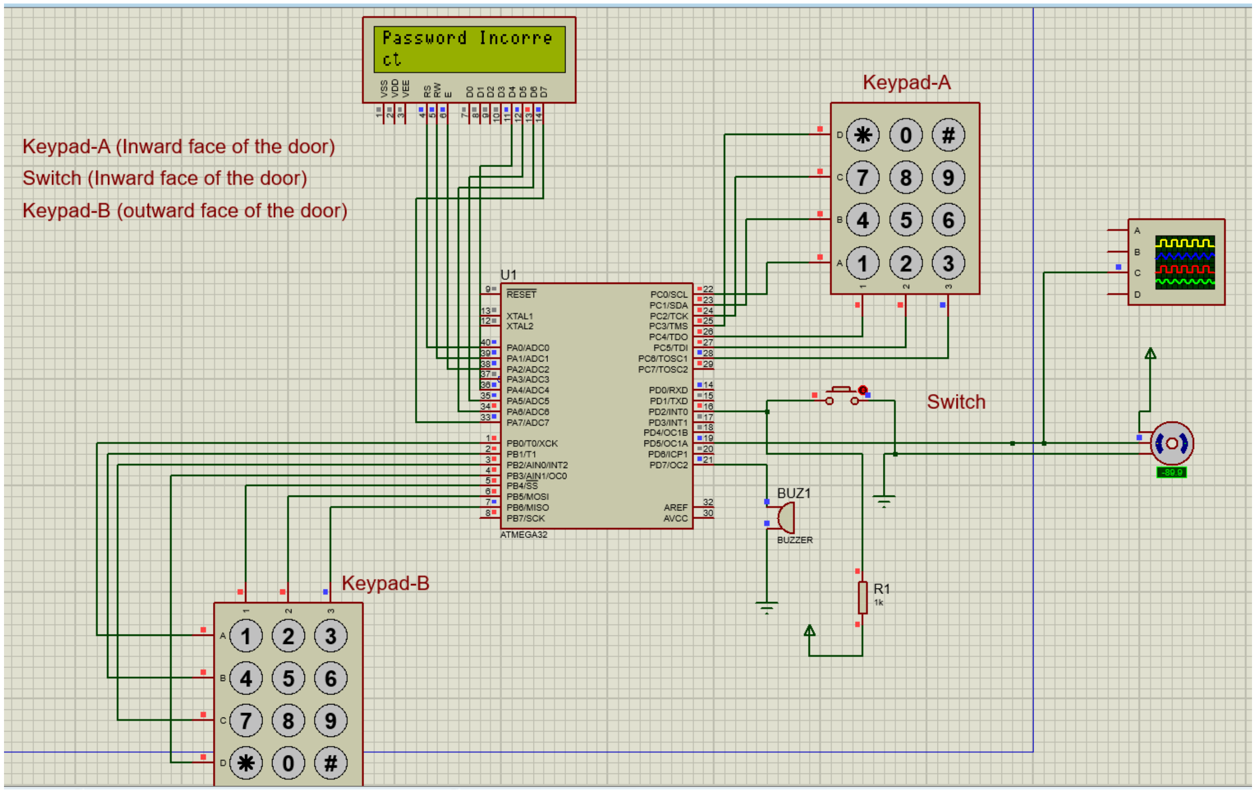
The user is now attempting to open the door by entering a password using keypad-B.


Password Correct
Door Unlocked
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
Keypad-A
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
Keypad-B

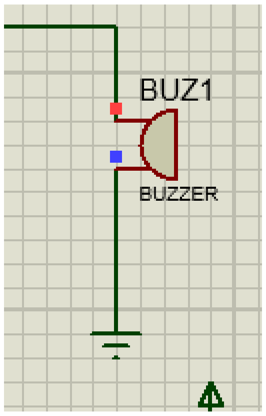




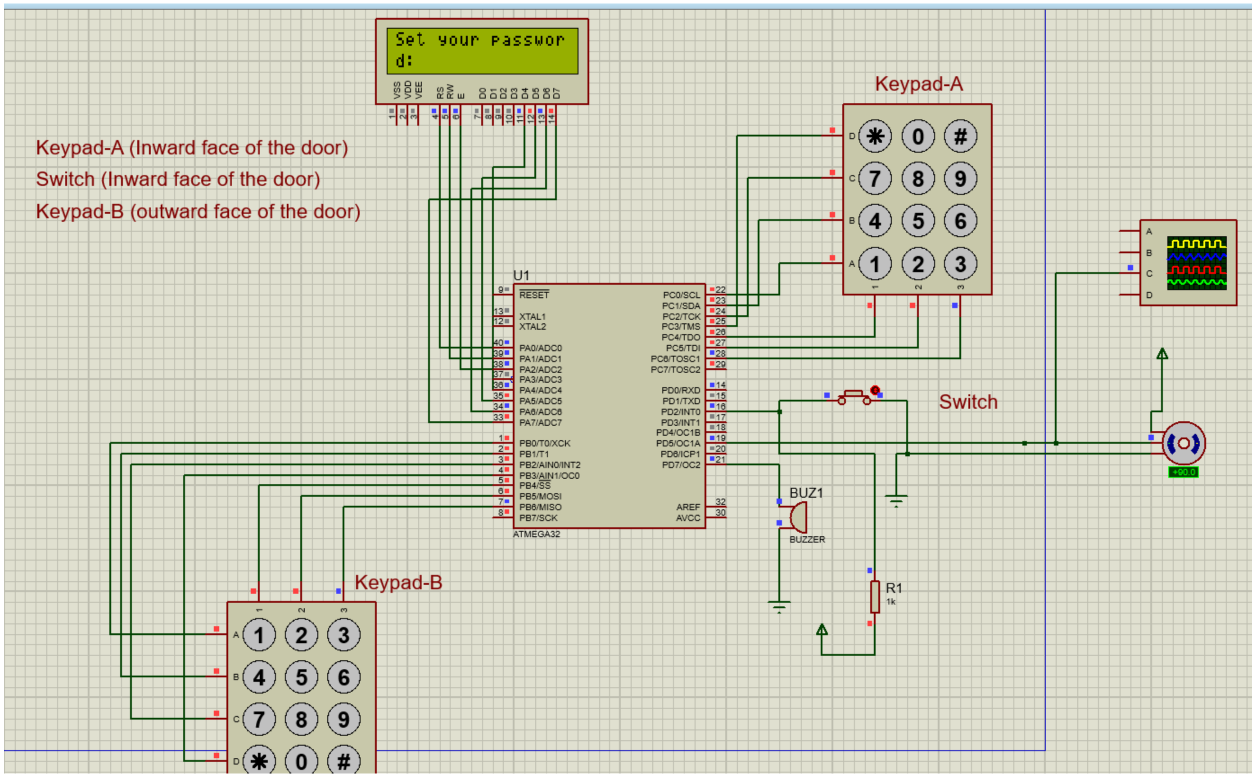
The attempted password matches the previously set password. Thus the door unlocks and the servo motor rotates from -90 to 90 degrees.


Password Incorre
ct
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
Keypad-A
Keypad-B
Switch
U1
ATMEGA32
BUZ1
BUZZER
R1
1k
-89.9

The attempted password does not match the previously set password. Please notice that the servo motor remains at -90 degrees meaning that the door will remain locked.



The attempted password is wrong three times straight so the buzzer sets of.


Set your passwor
d:
Keypad-A
Keypad-A (Inward face of the door)
Switch (Inward face of the door)
Keypad-B (outward face of the door)
U1
ATMEGA32
Switch
BUZ1
BUZZER
R1
1k
Keypad-B

The switch connected to INT-0 will allow for reset capabilities. That is that the user has to press the switch first then reset the password.

Discussion:

This project was implemented entirely on ATMega 32 using Proteus and CodeVision AVR software. Several key concepts taught in the Microprocessor and Interfacing Laboratory (EEE-366) were utilized whilst implementing this project. For example, the row-scanning technique, interrupt, generation of PWM wave to control servo motor and interfacing components like LCD,keypad, buzzer, etc.

In the coding phase of the project, there were two particular problems faced which was particularly difficult.

The first problem faced was that the two global int variables were being used to keep track of the number of times the keys of both the keyboard were being pressed were not being updated, This was important in storing the entry to arrays and for display purposes.

The cause for this was that :

```
if (input_data .0 == 0) {
    lcd_gotoxy(i, 0);
    lcd_putsf("*");
    password[i] = '3';
    i = i + 1;
}
```

The above is the faulty code.This faulty code was repeated for the other keys in the row scanning method of the keypad. What was happening here was that the entire keypad scanning code was being run through very quickly as the ATMega32 operates at 16Hz.So the global int variable which was supposed to keep track of the times the keys were being pressed was continually being updated.

The solution to this problem was to ensure that our code remains within the loop until the key is released. This was done using the following lines of code:

```
if (input_data .0 == 0) {
   lcd_gotoxy(i, 0);
   lcd_putsf("*");
   password[i] = '3';
   i = i + 1;
   while (input_data .0 != 1) {}
```

The second problem faced was that the attempted password and set password were being stored permanently in their respective arrays.

For example, what was happening was that if the user typed in the attempted password as “1 2 3 4” then in the array password[4] was being stored permanently. So new entry was being taken if the user got the password wrong the first time.

However by setting i=0 whenever the check() function was called regardless of whether the user got the password right or wrong this problem was solved. As the next time keypad_scan() was called the new entries just overwrote the old entries.

The reset password issue was also fixed using similar logic.

X. Applications

The dynamicity of the password protected lock designed ensures that it can be used in any application where there is a door or gate of some sort. Examples include but are not limited to:

- Cars
- Houses
- Vaults
- Factories

XI. Future scope

In the future we would like to add a biometric sensor like fingerprint sensor to the system.Furthermore, we would like to interface a GSM module using serial communication concepts.This would allow the system to send an SMS alert if an incorrect password had been entered multiple times.Both this features would add extra-layers of security to the system.

XII. References

- https://drive.google.com/drive/folders/12yAxauC9dEkmh7RR-lL7pXBI4VzUiAgA?usp=sharing
- https://docs.google.com/document/d/1BGTD_f9jJTH_EjX4zVJIKk_7iiaimpYc/edit?usp=sharing&ouid=115615944943590333611&rtpof=true&sd=true
- https://www.researchgate.net/publication/341873045_Smart_Door_Monitoring_and_Locking_System_using_SIM900_GSM_Shield_and_Arduino_UNO/figures
- https://www.hackster.io/athulakumar10/digital-lock-using-atmega32-b3f479#story
- https://www.amazon.in/Servo-Motor-Micro-Helicopter-Airplane/dp/B07GFH1K3N
- https://www.banggood.com/SPT-Servo-SPT5435LV-180-35KG-Large-Torque-Metal-Gear-Digital-Servo-For-RC-Robot-Arm-RC-Car-p-1577509.html?gmcCountry=US&currency=USD&cur_warehouse=CN&createTmp=1&utm_source=googleshopping&utm_medium=cpc_bgs&utm_content=sandra&utm_campaign=sandra-ssc-us-all-0407&ad_id=512762581403&gclid=CjwKCAiAzrWOBhBjEiwAq85QZ-a6XHHczSW75VSbSiIFuhidmoJtcexUB3u0FnVV16h5OKnqpGyzrRoC-4YQAvD_BwE
- https://thepihut.com/blogs/raspberry-pi-roundup/whats-the-difference-between-dc-servo-amp-stepper-motors#:~:text=The%20position%20of%20servo%20motors,power%2C%20ground%20%26%20control)
- https://atmega32-avr.com/4x4-keypad-based-password-atmega32-lcd-display/

XIII. Google drive link of source code,schematic and simulation video

Source-code:
https://drive.google.com/drive/folders/1fEkIyVZBSQjphB0rL75mBxG0dV4z3SFd?usp=sharing
(In the file named keypad.c the source code can be found.)

Schematic:
https://drive.google.com/drive/folders/10jkCog3rZIBBS7ifI8W78tIZEnkuO_bm?usp=sharing
(Please open the file Q4.pdsprj to avail the schematic)

Simulation Video:
https://drive.google.com/file/d/19RDJduOGp5DNJu-UnaG0bI35e89yp5Zx/view?usp=sharing