MICROCONTROLLERS PROJECT

“Smart Door Locking System Using ESP32 Microcontroller”

\

BY:

GUNTURU JEEVACHETAN – BT22EEE055

VENKANNA BHUKYA – BT22EEE062

NUTULA KRANTHI REDDY - BT22EEE037

BALAGAM SWAROOP NAIDU - BT22EEE049

AIM: Smart Door Locking System using ESP32 Microcontroller

OBJECTIVE:

- To show the Secure Door Access Using Keypad Input, which allows users to enter a predefined passcode on the 4x4 keypad to unlock the door, enhancing security compared to traditional keys.
- Making Customised Access Control, as the user can change the password instead of breaking the whole system.
- To use ESP32 and upload the IDE code to predefine the password, also change whenever needed.

INTRODUCTION:

Smart Door Lock is a password-based electronic code lock that we made using an ESP32 Microcontroller. It allows users to unlock doors using a digital code instead of a traditional key. This system typically includes a keypad for password entry and an electronic lock mechanism controlled by a microcontroller, like the **ESP32**.

Why use a smart door locking system with a password?

- **Keyless Entry:**
  No physical keys are needed, reducing the risk of losing keys or unauthorised duplication.
- **Customizable Access:**
  You can easily change passwords for security or give temporary access to guests, cleaners, etc.
- **Improved Security:**
  Can be combined with other features like auto-lock, alarm on failed attempts.
- **Remote Control:**
  When integrated with IoT systems, doors can be locked/unlocked remotely.

Why build it using ESP32?

We can build the digital door using many ways, ex, 8051 microcontroller, Arduino, ESP32, and other types of microcontrollers. But we tried to build it with ESP32 because it is a powerful microcontroller with built-in Wi-Fi and Bluetooth, making it perfect for smart IoT applications. But we haven't performed using Bluetooth and Wi-Fi.

Advantages of using ESP32:

- **Low Power Consumption:**

  Great for battery-operated smart locks.

- **Affordable and Readily Available:**

  ESP32 is cost-effective for personal and commercial use.

- **GPIO Pins:**

  Supports sensors (e.g., motion, temperature) and output devices (e.g., buzzers, servos for locks).

- **Secure Communication:**

  Supports SSL/TLS for secure data transmission (e.g., when sending logs or receiving commands).

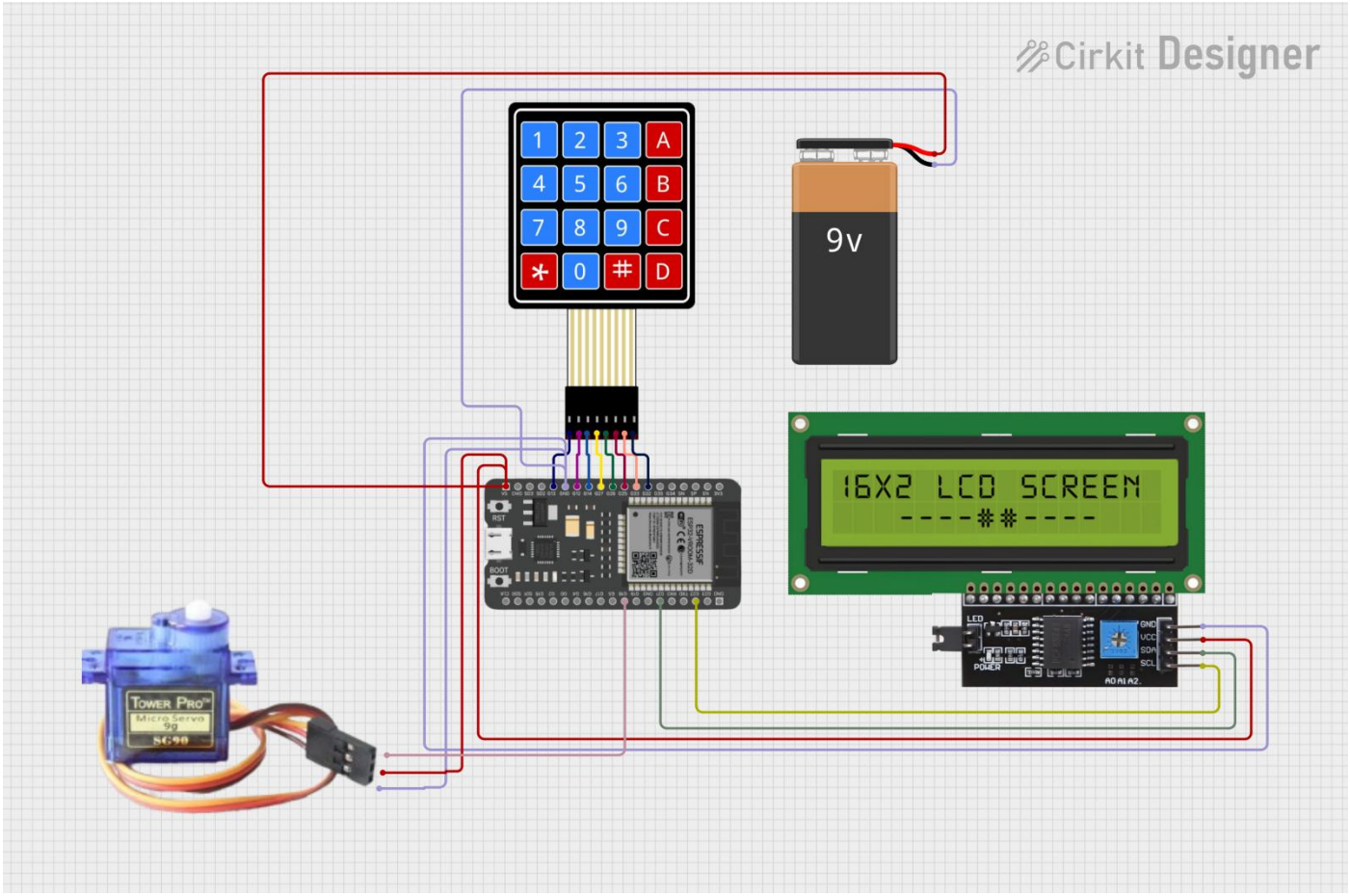
- **Built-in Wi-Fi & Bluetooth:**

  Easily connect the lock system to a smartphone app or a web interface for remote control

About Our Project:

Our Digital Code Lock project is a simple electronic number lock system using EPS32, which has a preset ***4-digit password*** stored inside the program. The system collects a 4-digit user input and compares the user input with the preset password inside the program. If the user input and stored password match, access will be granted (by opening the door, i.e., rotating the servomotor 90 degrees for a few seconds (5 sec) and closing it automatically after the stipulated time). If there is a mismatch between user input and stored password, access will be denied (by not opening the closed door).

Circuit Diagram:



Components used:

- **ESP32 module:**

  A powerful microcontroller with built-in Wi-Fi and Bluetooth, used to control this locking system, process user password through keypad input, drives the servo, and display corresponding messages on the LCD.

- **4x4 keypad:**

  A matrix keypad allows users to enter password; it sends the pressed key signals to the ESP32 for verification by comparing it with predetermined password and unlocking.

- **Servomotor SG90:**

  A small, precise actuator used to rotate and lock/unlock the door mechanism based on correct input given by user received to the microcontroller.

- **Power supply (9v battery and power bank):**

  Provides necessary electrical power to run the ESP32, servo, and LCD, ensuring stable operation without interruption.

- **USB to B-type cable**

  Used to upload IDE code to the ESP32 and optionally deliver additional power from a power bank or USB source when battery alone is insufficient.

- **16x2 LCD Display with I2C module:**

  A screen that displays system status, prompts, and messages, while the I2C module simplifies wiring using only two data lines.

- **LED lights** (Red and Green)
- **Jumper wires**

Hardware Setup

Wiring

- **Keypad:**
  - 4 rows → GPIO 13, 12, 14, 27
  - 4 cols → GPIO 26, 25, 33, 32
- **Servo:**
  - Signal → GPIO 18
  - Power → 9V
  - GND → GND (shared with ESP32)

- **LCD Display:**
  - GND → GND
  - Vcc→ Vin
  - SDA → GPIO 21
  - SCL → GPIO 22

Note: If your servo draws a lot of current, power it with an external 9v source and connect the grounds.

Software Setup:

- After making above hardware setup. Connect ESP32 Module to Computer using a USB to B type cable.
- Open Arduino IDE and Install Libraries (Keypad, ESP32Servo, Liquid Crystal_I2C).
- Connect to Board **ESP32 Dev Module** and corresponding Port **COM9**(USB).
- Write the below C++ code in IDE.
- Verify and upload the code until the it gets compiled and get uploaded status.

Code:

```
#include <Keypad.h>
#include <ESP32Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// LCD setup (I2C address 0x27 is common)
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Servo & LED Pins
const int servoPin = 18;

// Password
String input = "";
String correctCode = "1234";

// Servo
Servo lockServo;

// Keypad setup
const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {13, 12, 14, 27};
byte colPins[COLS] = {26, 25, 33, 32};
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
  Serial.begin(115200);
```

```
  lockServo.attach(servoPin);
  lockServo.write(0); // Locked

  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);

  // LCD setup
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Enter Passcode");
}

void loop() {
  char key = keypad.getKey();

  if (key) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Key: ");
    lcd.print(key);

    if (key == '#') {
      if (input == correctCode) {
        unlockDoor();
      } else {
        failure();
      }
      input = "";  // reset
    } else if (key == '*') {
      input = "";
      lcd.clear();
      lcd.print("Cleared");
    } else {
      input += key;
    }
  }
}

void unlockDoor() {
  Serial.println("Access Granted");
  lcd.clear();
  lcd.print("Access Granted");
  digitalWrite(greenLedPin, HIGH);
  digitalWrite(redLedPin, LOW);
  lockServo.write(90);  // Unlock
  delay(5000);
  lockServo.write(0);   // Lock
  digitalWrite(greenLedPin, LOW);
  lcd.clear();
  lcd.print("Enter Passcode");
}

void failure() {
  Serial.println("Access Denied");
  lcd.clear();
  lcd.print("Access Denied");
  digitalWrite(redLedPin, HIGH);
  delay(2000);
  digitalWrite(redLedPin, LOW);
  lcd.clear();
  lcd.print("Enter Passcode");
}
```

How It Works:

1. “ENTER PASSWORD”: The user enters a password on the keypad.
2. Press # to confirm.
3. The ESP32 checks if the entered code matches the saved password.
4. If correct:
    - Display shows “Access Granted”
    - It sends a signal to the servo motor to unlock the door (rotate to unlock position).
    - After a 5 sec delay, the servo returns to the lock position.
5. Press * to clear the input.
6. If incorrect:
    - It displays “Access Denied”
    - It doesn’t send a signal to the servo motor and will not rotate.

Control via Web Page (ESP32 as Web Server)

We can also setup Control though mobile as ESP32 can host a simple web page that you access on your phone over Wi-Fi.

6. ESP32 connects to your Wi-Fi.
7. Hosts a webpage with a lock/unlock button.

1. You open the ESP32's IP address on your phone browser.
2. Press button → ESP32 moves the servo.

How It Works:

- ESP32 connects to your Wi-Fi network.
- It hosts a simple webpage on a local IP (e.g., 192.168.1.100).
- When you visit the IP on your phone, you’ll see a button (e.g., "Unlock Door").
- Clicking the button sends a command to move the servo.

But we haven't made this advancement in our project to make it simpler.

Conclusion:

This smart door locking system is a compact and efficient solution for modern home security. It uses an ESP32 microcontroller to process input from a 4x4 keypad, allowing access only through a correct password. A servo motor operates the locking mechanism based on the verified input. The 16x2 LCD display with I2C provides clear, real-time feedback to users, such as prompts and access status. Powered by a 9V battery and/or a power bank, the system is portable and reliable. The USB to B-type cable is used for programming and additional power. Overall, it's a cost-effective, user-friendly smart security system.