PROJECT REPORT

EMBEDDED SYSTEMS PROJECT (EE3401)

Password based Door lock system using 8051 Microcontroller

Project Title: PASSWORD BASED DOOR LOCK SYSTEM USING 8051 MICROCONTROLLER

Group members:

T. Srimannarayana Rao(121EE1110)

M. Gunavarshin(121EE1046)

P. Nitish Chandra(121EE0835)

Introduction:

Automated systems have less manual operations, so that the flexibility, reliabilities are high and accurate. Nowadays most of the systems are automated in order to face new challenges and present day requirements to achieve good results.

Objective:

Our objective is to utilize different electronic parts available in the market and build an integrated home security system based on ICs, Microcontroller and LCD screen and another main objective of designing this password based door lock system is to provide many modern security features than mechanical lock.

Principle:

The password which is entered through the 4X3 Keypad is compared with the predefined password. If the entered password is correct, then the system opens the door by rotating door motor and displays the status of door on LCD. If the password is wrong, then door is remain closed and displays “Password is wrong” on LCD.

Hardware Required:

1. 8051 Microcontroller developer board

2. Connecting wires

3. 4 * 4 Matrix Keypad

4. 16 *2 LCD display

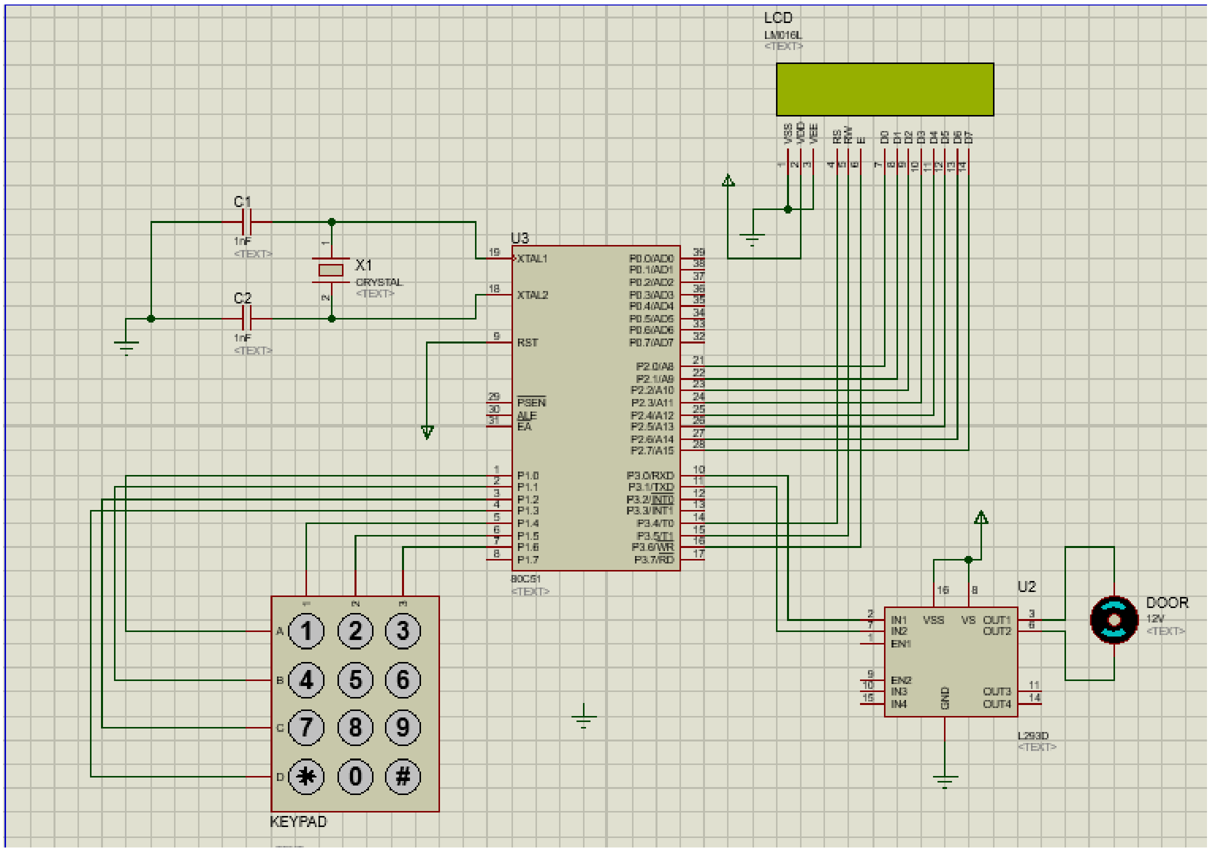
5. DC Motor

6. L293D Motor Driver

7. Power

8. Ground

Proteus Design:



Description of Components:

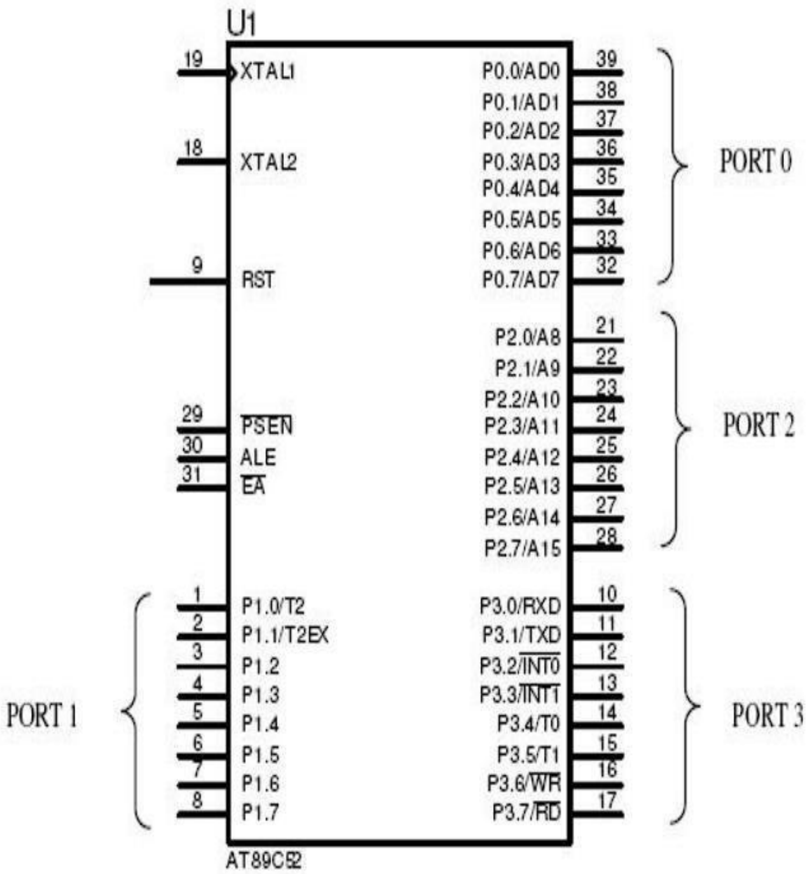
L.C.D: Here we are using a 16x2 L.C.D which is a Liquid Crystal Display. Its function is to display the alphanumeric symbols to indicate the status message of the circuit. This L.C.D can display the two lines and each line contains 16 characters.

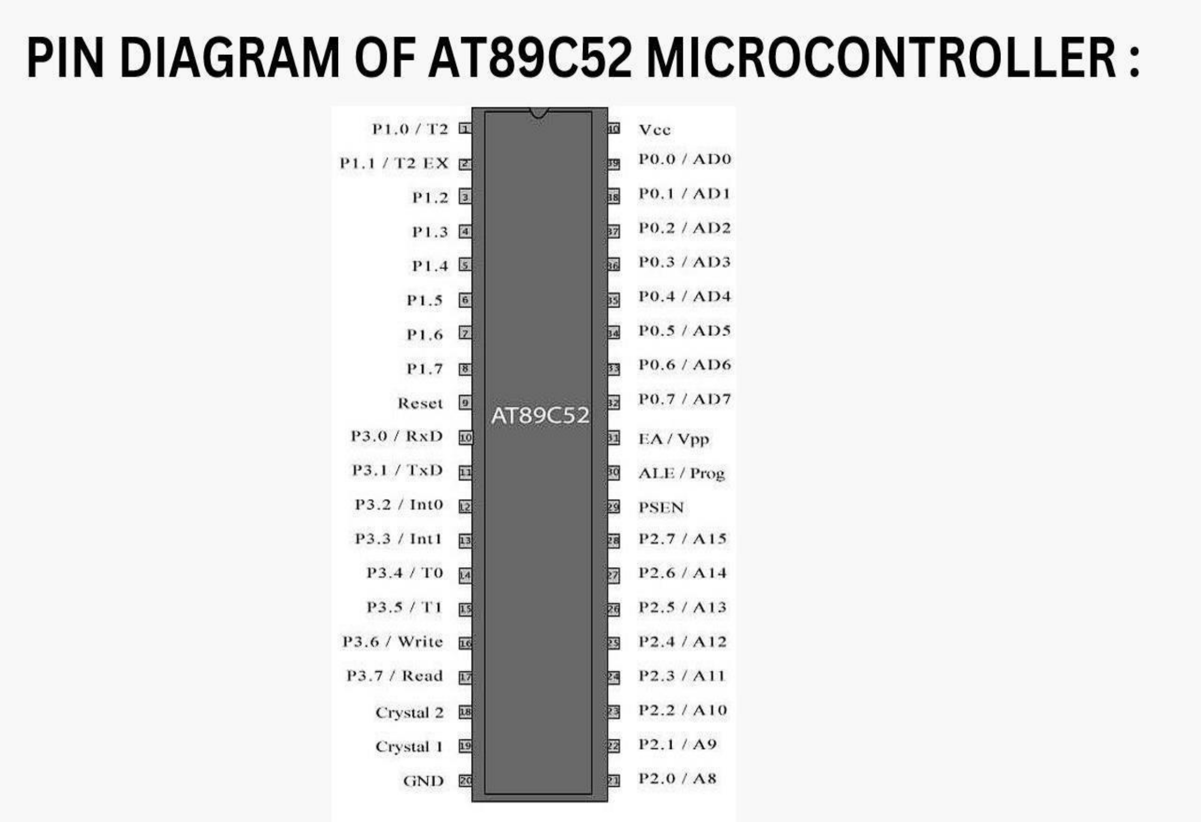
Data pins of LCD is connected to port P2 from P2.0-P2.7 describing the current status of system and three control pins RS, RW and E are connected to port P3 from P3.4- P3.6 respectively.

D.C Motor Drive: It is a machine used to convert the electrical energy into mechanical energy. This D.C motor uses a 5V battery as an input and rotates to open the door or to lock the door. To interface this D.C motor drive with the Micro Controller we use an L293D I.C. Dc motor is connected to port P3 from P3.0-P3.1 .

4x4 Matrix Keyboard: This keyboard contains a number of switches arranged in a matrix format. Each rows and each column are connected to the pins of micro controller. This keyboard contains numbers from 0 to 9, a ‘*’ button and a ‘#’ button. These switches are generally a numbers of push buttons. With the help of this Keyboard an individual can enter the password to unlock the door. 4x4 keypad is connected to port P1 from P1.0-P1.7 allowing user to enter user id.

I/O Port Pin and microcontroller connection diagram:





Application:

User can unlock the door using pre-defined passcode. When a user try to unlock the door using wrong passcode a message will be triggered. The door will automatically open when only passcode is matched. The main component in the circuit is 8051 micro controller which is basically used to send and accept text message from user. 4*4 Keypad is used to enter the password. The entered password is compared with the pre-defined password. If it is correct password, the system opens the door by rotating servo motor and display the status of the door on LCD. If the password is wrong then door remain closed and display message to the user.

Algorithm:

```
ORG 0H

; Port initialization

MOV P0, #0FFH ; Set P0 as output for the control lines of the LCD display
```

```
MOV P1, #0FFH ; Set P1 as input for the 4x4 keypad
MOV P2, #0FFH ; Set P2 as output for the 14-pin LCD data lines
MOV P3, #0FFH ; Set P3 as output for motor control lines

; Define constants
DELAY_COUNT EQU 0C80H ; Hexadecimal value for the delay count (5 seconds)
CORRECT_PASSWORD_HIGH EQU 016H ; High byte of the correct password (56)
CORRECT_PASSWORD_LOW EQU 078H ; Low byte of the correct password (78)

MAIN:
   CALL INIT_PORTS ; Initialize ports
   CALL INIT_LCD ; Initialize the LCD
   CALL GET_NEXT_DIGIT ; Get the entered password from the user
   CALL CHECK_PASSWORD ; Check if the entered password is correct

   JZ WRONG_PASSWORD ; If the password is wrong, display "Wrong Password"
   CALL CLEAR_LCD ; Clear the LCD display

   CALL DISPLAY_DOOR_OPENING ; Display "Door is opening" message
   ; Rotate the motor...

   MOV P3, #01H ; Set P3.0 for the motor
   CALL DELAY_MOTOR ; Delay for 5 seconds

   MOV P3, #02H ; Set P3.1 for the motor
   CALL DELAY_MOTOR ; Delay for 5 seconds

   MOV P3, #00H ; Clear P3 to stop the motor
   CALL DELAY_LCD ; Delay before clearing the LCD
   CALL CLEAR_LCD ; Clear the LCD display
   JMP MAIN ; Restart the process

WRONG_PASSWORD:
```

```
    CALL CLEAR_LCD ; Clear the LCD display
    CALL DISPLAY_WRONG_PASSWORD ; Display "Wrong Password" message

    CALL CLEAR_LCD ; Clear the LCD display
    JMP MAIN ; Restart the process

GET_NEXT_DIGIT:
    ; Initialize variables
    MOV R2, #00H ; Initialize R2 to 0 (higher nibble)
    MOV R3, #00H ; Initialize R3 to 0 (lower nibble)

    ; Scan rows of the keypad
    MOV P1, #0FEH ; Enable row 0
    MOV A, P1 ; Read columns
    JNB P1.0, ROW0_CHECK ; Check if column 0 is low
    JNB P1.1, ROW1_CHECK ; Check if column 1 is low
    JNB P1.2, ROW2_CHECK ; Check if column 2 is low
    JNB P1.3, ROW3_CHECK ; Check if column 3 is low

ROW0_CHECK:
    MOV R2, #01H ; Set R2 to 1 (digit 1, higher nibble)
    JMP KEY_READ_DONE ; Exit the loop

ROW1_CHECK:
    MOV R2, #02H ; Set R2 to 2 (digit 2, higher nibble)
    JMP KEY_READ_DONE ; Exit the loop

ROW2_CHECK:
    MOV R2, #03H ; Set R2 to 3 (digit 3, higher nibble)
    JMP KEY_READ_DONE ; Exit the loop

ROW3_CHECK:
    MOV R2, #0AH ; Set R2 to 10 (digit 0, higher nibble)
```

```
    JMP KEY_READ_DONE ; Exit the loop

KEY_READ_DONE:
    ; Now, R2 contains the higher nibble of the digit read from the keypad
    ; Process the digit and update R0 accordingly
    ADD A, R2 ; Add the higher nibble to the accumulated password
    MOV R0, A ; Update R0 with the accumulated password

    ; Display the digit on the LCD
    CALL DISPLAY_ON_LCD ; Call subroutine to display on LCD

    ; Shift R0 to make room for the lower nibble
    MOV A, R0 ; Copy the accumulated password
    SWAP A ; Swap the nibbles
    MOV R0, A ; Update R0 with the shifted value

    ; Scan rows of the keypad again for the lower nibble
    MOV P1, #0FEH ; Enable row 0
    MOV A, P1 ; Read columns
    JNB P1.0, ROW0_CHECK_LOWER ; Check if column 0 is low
    JNB P1.1, ROW1_CHECK_LOWER ; Check if column 1 is low
    JNB P1.2, ROW2_CHECK_LOWER ; Check if column 2 is low
    JNB P1.3, ROW3_CHECK_LOWER ; Check if column 3 is low

ROW0_CHECK_LOWER:
    MOV R3, #01H ; Set R3 to 1 (digit 1, lower nibble)
    JMP KEY_READ_DONE_LOWER ; Exit the loop

ROW1_CHECK_LOWER:
    MOV R3, #02H ; Set R3 to 2 (digit 2, lower nibble)
    JMP KEY_READ_DONE_LOWER ; Exit the loop

ROW2_CHECK_LOWER:
```

```
    MOV R3, #03H ; Set R3 to 3 (digit 3, lower nibble)
    JMP KEY_READ_DONE_LOWER ; Exit the loop

ROW3_CHECK_LOWER:
    MOV R3, #0AH ; Set R3 to 10 (digit 0, lower nibble)
    JMP KEY_READ_DONE_LOWER ; Exit the loop

KEY_READ_DONE_LOWER:
    ; Now, R3 contains the lower nibble of the digit read from the keypad
    ; Process the digit and update R0 accordingly
    ADD A, R3 ; Add the lower nibble to the accumulated password
    MOV R0, A ; Update R0 with the accumulated password

    ; Display the digit on the LCD
    CALL DISPLAY_ON_LCD ; Call subroutine to display on LCD

    DEC R1 ; Decrement the loop counter
    JZ PASSWORD_ENTERED ; If all digits are entered, jump to PASSWORD_ENTERED

    JMP GET_NEXT_DIGIT ; Repeat the loop to get the next digit

PASSWORD_ENTERED:
    ; Add code here to handle when the password is entered successfully
    ; You can display a message or perform additional actions
    JMP MAIN ; Restart the process

CHECK_PASSWORD:
    ; Compare the entered password with the correct password
    MOV A, R0 ; Load the entered password
    CJNE A, #CORRECT_PASSWORD_LOW, PASSWORD_WRONG ; If not equal, jump to PASSWORD_WRONG
    MOV A, R1 ; Load the high byte of the entered password
    CJNE A, #CORRECT_PASSWORD_HIGH, PASSWORD_WRONG ; If not equal, jump to PASSWORD_WRONG
    RET ; Return if the password is correct
```

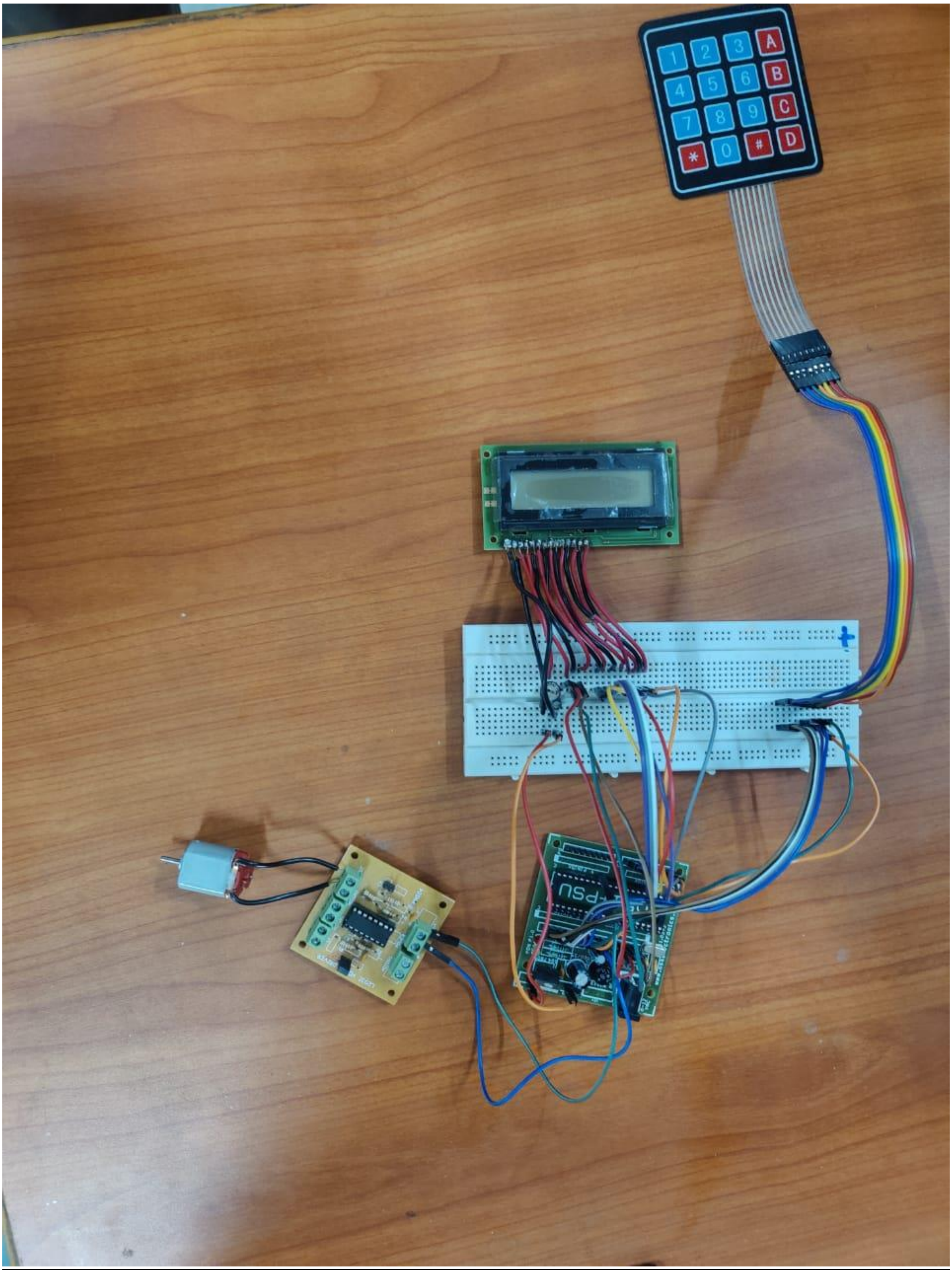
```
PASSWORD_WRONG:
  MOV R0, #00H ; Reset the accumulated password
  RET ; Return

DELAY_MOTOR:
  MOV R2, #0C8H ; Load delay count for the motor delay (5 seconds)
  MOV R3, #00H ; Clear R3
DELAY_MOTOR_LOOP:
  CALL DELAY ; Call the delay subroutine
  DJNZ R2, DELAY_MOTOR_LOOP ; Decrement R2 and jump if not zero
  RET

DELAY_LCD:
  MOV R2, #08H ; Load delay count for the LCD delay
  MOV R3, #00H ; Clear R3
DELAY_LCD_LOOP:
  CALL DELAY ; Call the delay subroutine
  DJNZ R2, DELAY_LCD_LOOP ; Decrement R2 and jump if not zero
  RET

CLEAR_LCD:
  CLR P0.0 ; RS = 0 (Command mode)
  CLR P0.1 ; RW = 0 (Write mode)
  SETB P0.2 ; E = 1 (Enable)
  MOV P2, #01H ; Send clear display command
  CALL DELAY_LCD ; Delay to ensure the command is processed
  CLR P0.2 ; E = 0 (Disable)
  RET

DISPLAY_ON_LCD:
  SETB P0.0 ; RS = 1 (Data mode)
  CLR P2.0 ; RW = 0 (Write mode)
```

```
    SETB P2.1 ; E = 1 (Enable)
    MOV P2, A ; Send the data (digit) to be displayed
    CALL DELAY_LCD ; Delay to ensure the data is processed
    CLR P2.1 ; E = 0 (Disable)
    RET

DISPLAY_DOOR_OPENING:
    CLR P0.0 ; RS = 0 (Command mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #80H ; Set DDRAM address to the beginning of the second line
    CALL DELAY_LCD ; Delay to ensure the command is processed
    CLR P0.2 ; E = 0 (Disable)

    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'D' ; Send 'D'
    CALL DELAY_LCD ; Delay to ensure the data is processed
    RET
DISPLAY_WRONG_PASSWORD:
    CLR P0.0 ; RS = 0 (Command mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #80H ; Set DDRAM address to the beginning of the second line
    CALL DELAY_LCD ; Delay to ensure the command is processed
    CLR P0.2 ; E = 0 (Disable)

    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'W' ; Send 'W'
    CALL DELAY_LCD ; Delay to ensure the data is processed
```

```
    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'r' ; Send 'r'
    CALL DELAY_LCD ; Delay to ensure the data is processed

    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'o' ; Send 'o'
    CALL DELAY_LCD ; Delay to ensure the data is processed

    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'n' ; Send 'n'
    CALL DELAY_LCD ; Delay to ensure the data is processed

    SETB P0.0 ; RS = 1 (Data mode)
    CLR P0.1 ; RW = 0 (Write mode)
    SETB P0.2 ; E = 1 (Enable)
    MOV P2, #'g' ; Send 'g'
    CALL DELAY_LCD ; Delay to ensure the data is processed

    RET

INIT_PORTS:
    MOV P0, #0FFH ; Set P0 as output for the control lines of the LCD display
    MOV P1, #0FFH ; Set P1 as input for the 4x4 keypad
    MOV P2, #0FFH ; Set P2 as output for the 14-pin LCD data lines
    MOV P3, #0FFH ; Set P3 as output for motor control lines
    RET
```

```
INIT_LCD:
  MOV P2, #038H ; Function Set: 2 Lines, 8-bit, 5x8 dots
  CALL COMMAND_DELAY ; Delay for command execution

  MOV P2, #00CH ; Display On/Off Control: Display ON, Cursor OFF, Blinking OFF
  CALL COMMAND_DELAY ; Delay for command execution

  MOV P2, #006H ; Entry Mode Set: Increment cursor, no display shift
  CALL COMMAND_DELAY ; Delay for command execution

  MOV P2, #001H ; Clear Display
  CALL COMMAND_DELAY ; Delay for command execution
 RET

COMMAND_DELAY:
  MOV R2, #0FFH ; Load delay count for command execution
COMMAND_DELAY_LOOP:
  NOP ; No operation
  NOP ; No operation
  NOP ; No operation
  NOP ; No operation
  DJNZ R2, COMMAND_DELAY_LOOP ; Decrement R2 and jump if not zero
  RET
END
```

Hardware interfacing:



Advantages:

- You won’t have to carry around a large set of keys and they will be less likely to be lost or stolen
- In a company building, you can control and restrict who goes into what part of the building and it reduces the risk of anything getting stolen.
- the prevent break-ins because burglars are unable to pick or ‘bump’ the lock. Criminal’s methods of breaking and entering are improving and the majority of criminals can pick an ordinary key lock.

Conclusion:

In conclusion, the development of an embedded system for door access control through soft keys addresses the common issue of forgetting physical keys. This innovative solution not only enhances convenience but also improves security by allowing access control through password authentication. The integration of soft keys provides a flexible and user-friendly approach, offering both local and remote accessibility for door management. This project aligns with the contemporary need for smart and efficient solutions in access control systems, contributing to a more seamless and secure user experience.

Thank You