Department of I & CT

MIT, Manipal

Secure Access: Password-Based Authentication System with Intrusion Lockout

ICT 3143 – Embedded Systems Lab Mini Project

IVth Sem B.Tech (CCE)

230953010 – Disha Gomes
230953012 – P Durai Naveen
230953050 – Mugdha Chatterjee

ABSTRACT

This project implements a secure password-based authentication system using the LPC1768 ARM Cortex-M3 microcontroller. Users input a 4-digit password via a matrix keyboard, with feedback provided through an LCD display. On successful entry, access is granted and displayed as “Door Open”. If the entered password is incorrect, the system counts failed attempts and shows a numeric countdown on a 7-segment display.

After three failed attempts, the system enters a lockout state for 60 seconds, displaying “Device Locked” and ignoring further input. This prevents brute-force access and demonstrates effective embedded security.

Key features include matrix keyboard interfacing, LCD and 7-segment display outputs, and power-efficient lockout logic. Designed for access control in labs and secure zones, this project illustrates embedded principles such as timer interrupts, GPIO handling, and user interface design using peripherals.

Contents

List of Figures:

1. Introduction

The demand for localized, efficient access control has grown with the increasing need for digital security. Embedded systems play a vital role by offering low-cost, responsive, and customizable solutions. This project presents a keyboard-based access control system, replacing physical keys or cards with password input through a matrix keypad.

The system uses the LPC1768 microcontroller, which interfaces with a matrix keyboard, 16x2 LCD, and a 7-segment display. When a correct password is entered, the system permits access. If incorrect passwords are entered three times, the device locks for 60 seconds, preventing further inputs temporarily.

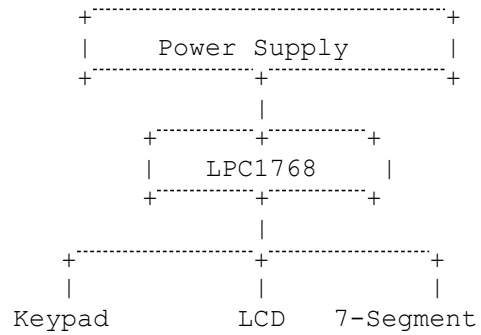
The design focuses on practical applications of keypad interfacing, timers, interrupts, and embedded display control, making it highly suitable for restricted-access environments like labs and secure rooms.

2. Methodology

a. Components Required

- LPC1768 Microcontroller Trainer Kit – Core processor for control and logic.
- 4x4 Matrix Keyboard – For password entry by the user.
- 16x2 LCD Display – Shows messages and password prompts.
- 7-Segment Display – Displays countdown during incorrect attempts.
- Reset Button – To manually reset the system if needed.
- Jumper Wires and Cables – For all component connections.
- Power Supply / USB Cable – Provides power and programming interface.

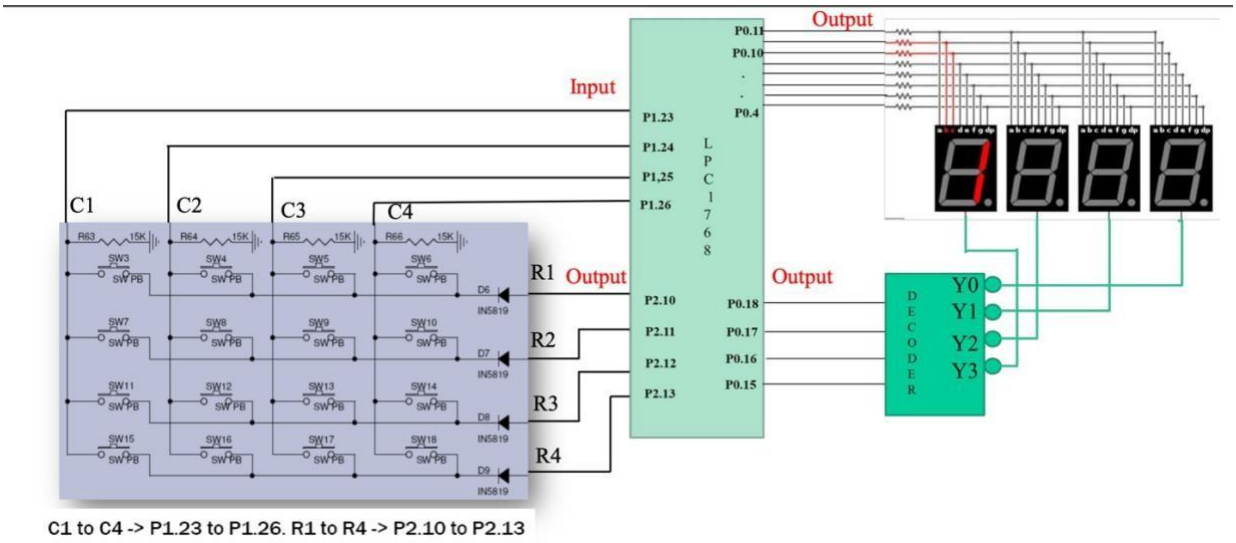
b. Block Diagram



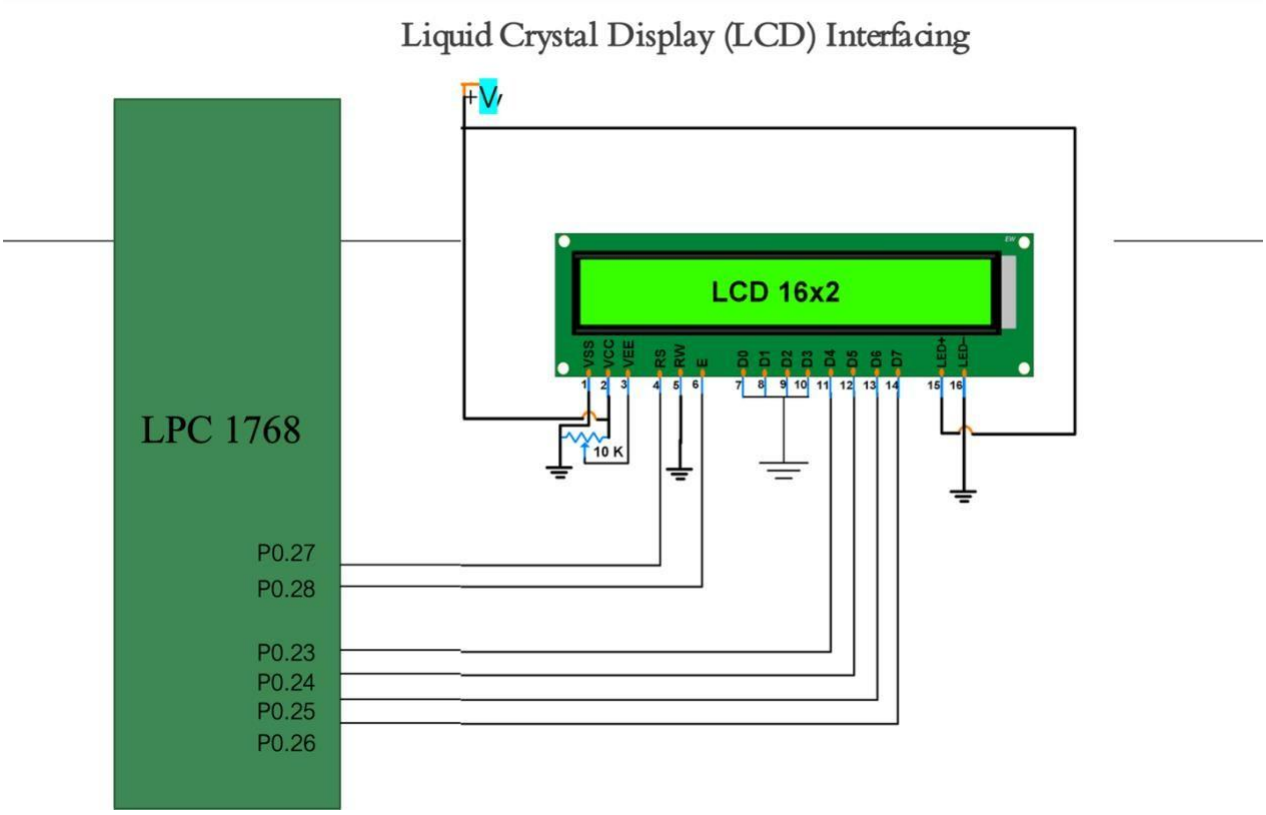
c. Connection Description

- Matrix Keyboard: Columns connected to pins P1.23 to P1.26; Rows to P2.10 to P2.13. Allows scanning to detect key presses.
- LCD Display: Connected in 4-bit mode to P0.23–P0.26 (data), RS to P0.27, EN to P0.28. • 7-Segment Display: Connected to GPIO pins for numeric display of remaining attempts (3→0).
- Reset Button: Connected to GPIO pin for user to reset the system if needed.
- Power: Supplied via USB or external 5V source.

d. Pin Diagram

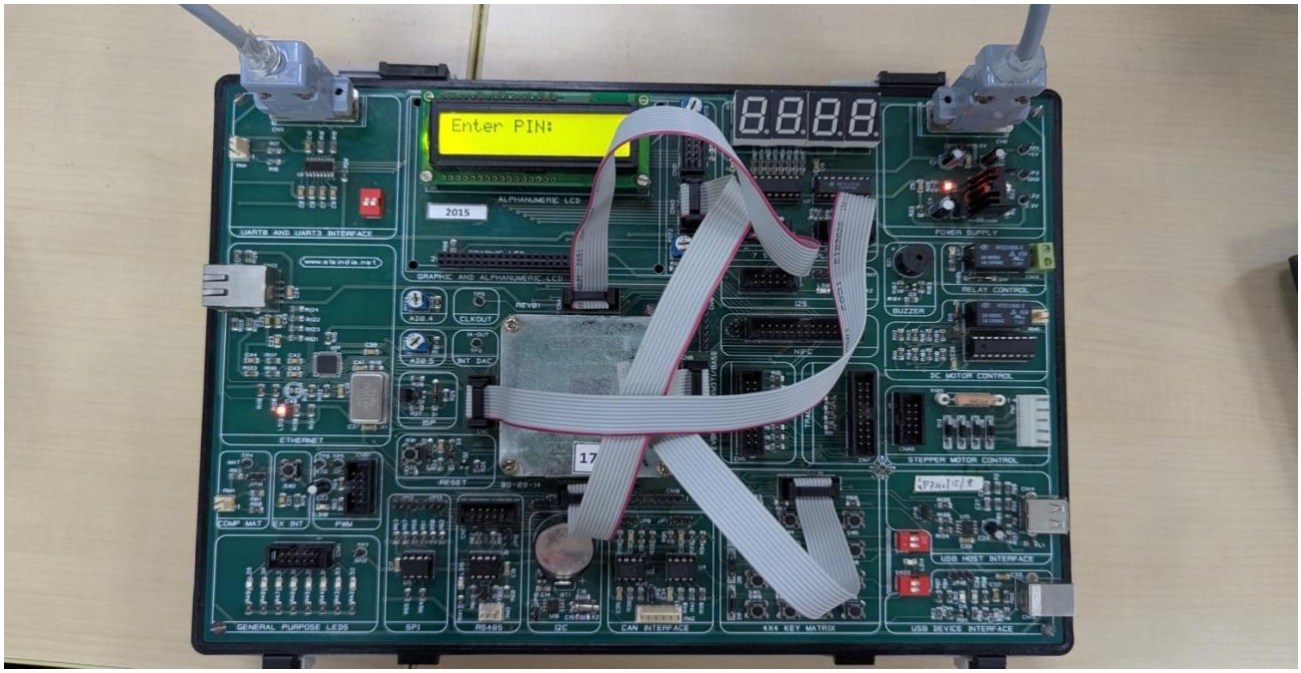


Keyboard Connection & Seven Segment Interfacing with LPC17C8

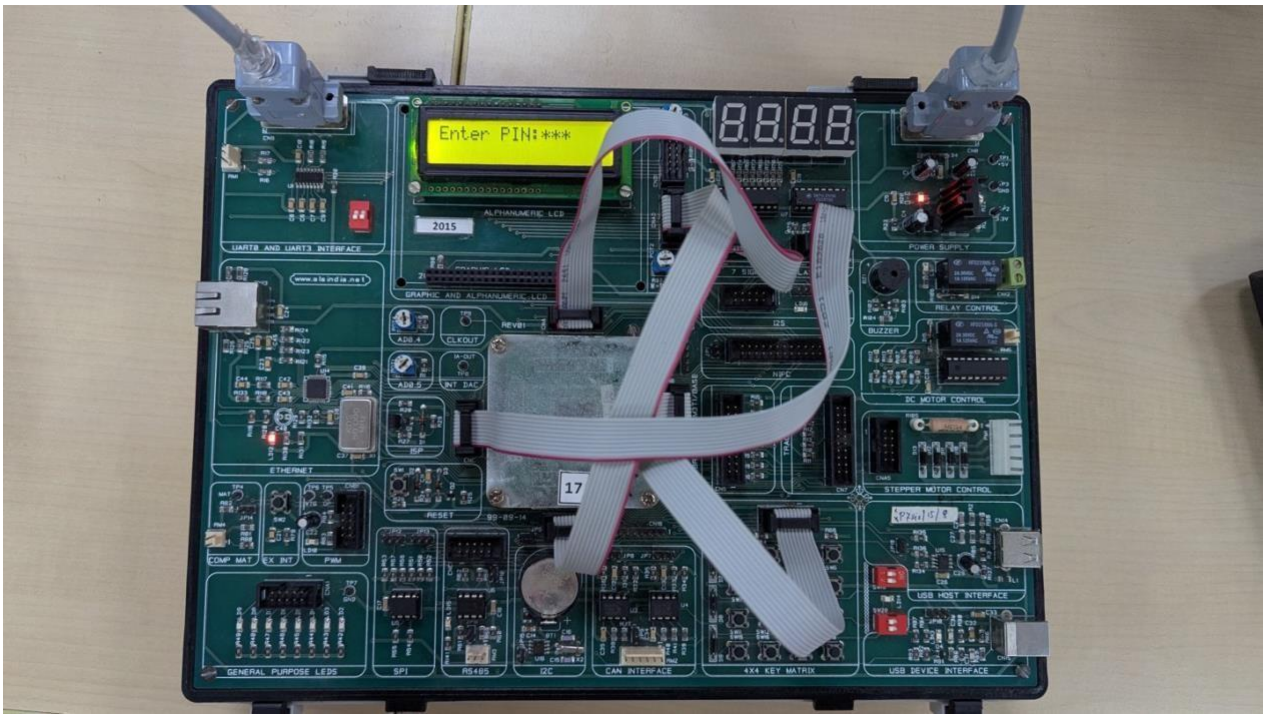


LCD Interfacing with LPC17C8

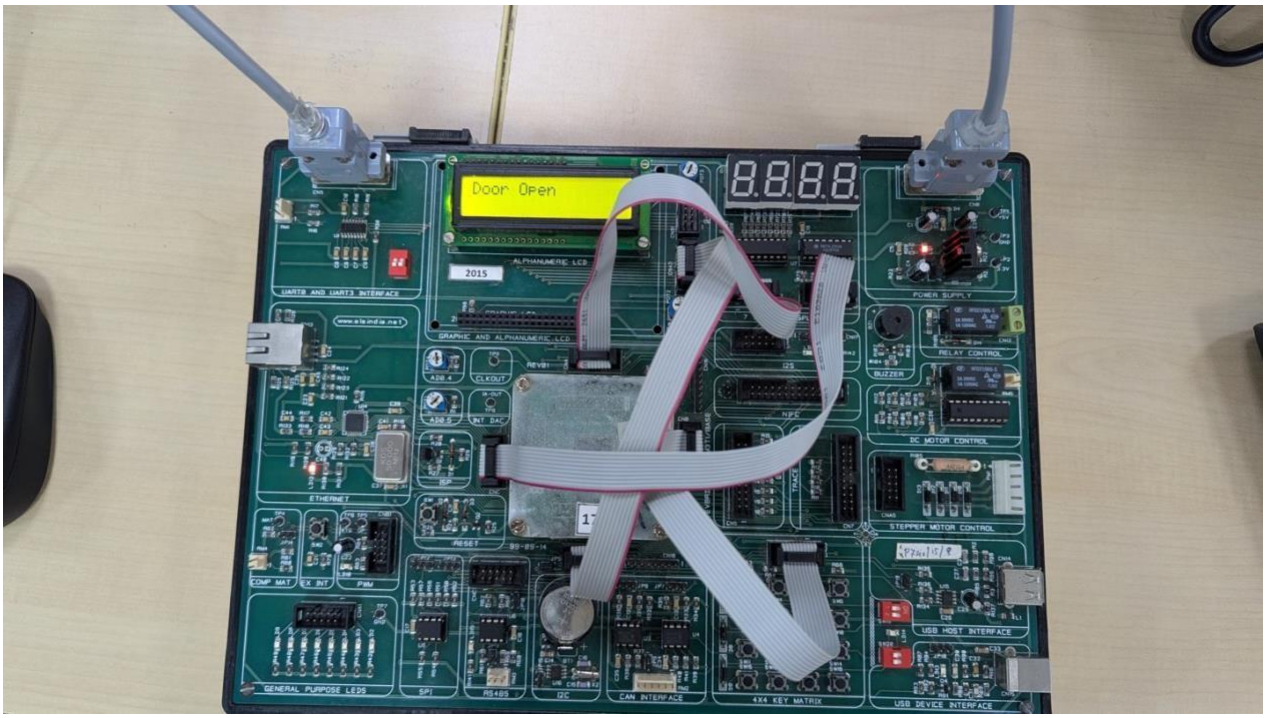
e. Results



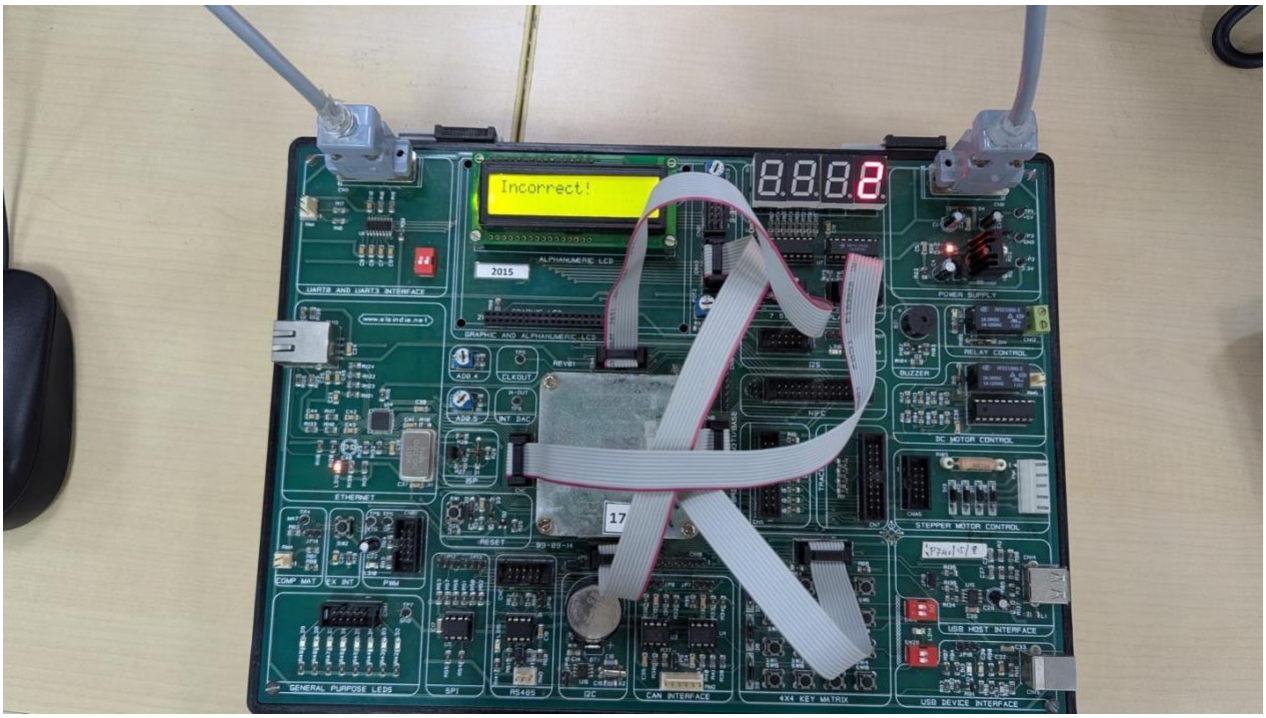
Enter Pin



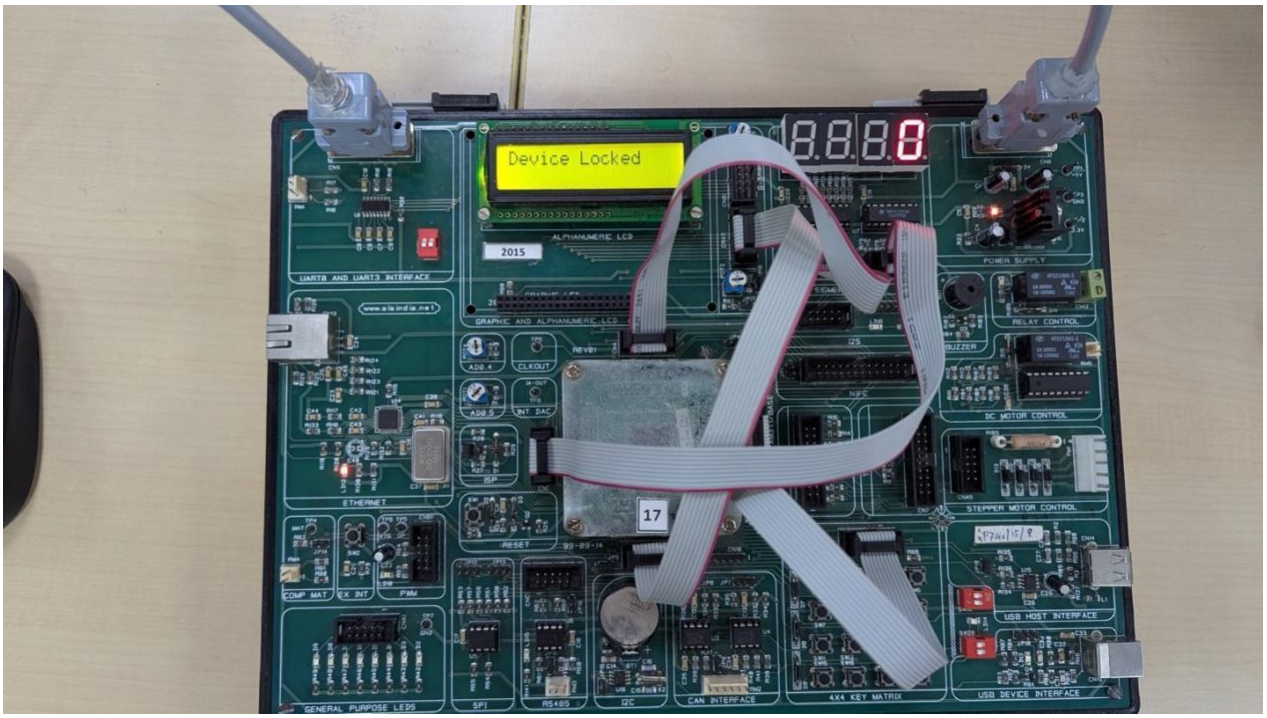
After Entering Pin (For Password Protection the pin have been hashed)



When we Enter Correct Pin



When we enter Incorrect Pin



If Incorrect Pin Entered 3 times the Device gets Locked

F. Code

```
#include <lpc17xx.h>
#include <stdio.h>
#include <string.h>

#define RS_CTRL 0x08000000 // P0.27
#define EN_CTRL 0x10000000 // P0.28
#define DT_CTRL 0x07800000 // P0.23 to P0.26
#define PIN_LENGTH 4

unsigned long temp=0, temp11=0, temp12=0, i;
unsigned char flag=0, flag1=0, flag2=0; unsigned
char col, row, j=3, attempts_left=3; unsigned char
key[4][4] = {{'0','1','2','3'},
                        {'4','5','6','7'},
                        {'8','9','A','B'},
                        {'C','D','E','F'}};
const char correct_pin[PIN_LENGTH] = {'1','2','3','4'}; // Change your PIN here
unsigned char input_pin[PIN_LENGTH]; unsigned char pin_index = 0;

unsigned char seven_seg[4]={0x3F,0x06,0x5B,0x4F}; // 3,2,1,0 unsigned long
init_command[] = {0x33,0x32,0x28,0x0C,0x06,0x01,0x80};

void    lcd_write(void);    void
port_write(void);           void
scan(void);                 void
delay_lcd(unsigned  int);   void
counter(void);              void
delay_ms(unsigned  int);    void
lcd_clear(void);            void
lcd_print(char     *str);   void
system_lockout(void);

int main() {
    SystemInit();
    SystemCoreClockUpdate();
    // GPIO Configuration
    LPC_GPIO0->FIODIR |= 0x00078FF0;
    LPC_GPIO0->FIODIR |= DT_CTRL | RS_CTRL | EN_CTRL;
```

```
    LPC_GPIO2->FIODIR |= 0x00003C00; // Rows P2.10-P2.13
    LPC_GPIO1->FIODIR &= ~(0x07800000); // Columns P1.23-P1.26

    // LCD Initialization for(i=0;
    i<7; i++) {
        temp11 = init_command[i]; lcd_write();
    }
lcd_clear();
    lcd_print("Enter PIN:");


    while(1) {
        for(row=0; row<4; row++) {
            LPC_GPIO2->FIOPIN = (1 << (row + 10)); scan();

            if(flag && pin_index < PIN_LENGTH) {
                input_pin[pin_index++] = key[row][col];
                temp11 = '*'; flag1 = 1; lcd_write();
                delay_ms(200);
            }

            if(pin_index == PIN_LENGTH) {
                if(memcmp(input_pin, correct_pin, PIN_LENGTH) == 0) {
                    lcd_clear(); lcd_print("Door Open");
                    LPC_GPIO0->FIOSET = (1<<19); // Buzzer (if connected)
                    delay_ms(2000);
                    LPC_GPIO0->FIOCLR = (1<<19);
                    pin_index = 0; attempts_left = 3;
                    lcd_clear();
                    lcd_print("Enter PIN:");
                }
                else {
                    attempts_left--; lcd_clear();
                    lcd_print("Incorrect!");
                    counter();

                    if(attempts_left == 0) {
                        system_lockout();
                        attempts_left = 3;
```

```
                }

                pin_index       =       0;
                delay_ms(1000);
                lcd_clear();
                lcd_print("Enter PIN:");
            }
        }
    }
}

void scan() { unsigned long temp3 = LPC_GPIO1->FIOPIN &
  0x07800000; flag = 0;

  if(temp3) {
    delay_ms(20); // Debounce temp3 = LPC_GPIO1-
    >FIOPIN & 0x07800000; if(temp3) {
      flag = 1; if(temp3 & (1<<23)) col =
      0; else if(temp3 & (1<<24)) col =
      1; else if(temp3 & (1<<25)) col =
      2; else if(temp3 & (1<<26)) col =
      3;
    }
  }
}

void counter() {
    LPC_GPIO0->FIOPIN = seven_seg[attempts_left] << 4; delay_ms(1000);

}

void system_lockout() {
  lcd_clear();
  lcd_print("Device   Locked");
  for(i=60; i>0; i--) {
    LPC_GPIO0->FIOPIN = seven_seg[0] << 4; // Display 0 delay_ms(1000);
  }
```

```
    lcd_clear();    lcd_print("Enter
    PIN:");
}

void lcd_print(char *str) {
    flag1 = 1; while(*str)
    {  temp11  =  *str++;
    lcd_write();
    }
}

void   lcd_clear()   {
    flag1      =      0;
    temp11  =  0x01;
    lcd_write();
    delay_ms(100);
}

void lcd_write() {
    temp12 = temp11 & 0xF0; temp12 =
    temp12 << 19; port_write(); temp12
    =   (temp11   &   0x0F)   <<   23;
    port_write();
}

void port_write() {
    LPC_GPIO0->FIOPIN = temp12;
    LPC_GPIO0->FIOCLR  =  RS_CTRL  |  EN_CTRL;
    if(flag1)   LPC_GPIO0->FIOSET   =   RS_CTRL;
    LPC_GPIO0->FIOSET = EN_CTRL;
    delay_lcd(500);
    LPC_GPIO0->FIOCLR = EN_CTRL;
    delay_lcd(500);
}

void  delay_ms(unsigned  int  ms)  {
unsigned  int  o,j;  for(o=0;  o<ms;
o++){ for(j=0; j<20000; j++);}
}
```

```
void delay_lcd(unsigned int r1) { while(r1--)
   __NOP();
}
```

4. References

- **LPC1768 User Manual – NXP Semiconductors**

Official documentation for the ARM Cortex-M3 LPC1768 microcontroller.

📎 https://www.nxp.com/docs/en/user-guide/UM10360.pdf

- **Matrix Keypad Interfacing with LPC1768 – Electronics-Lab**

Tutorial on connecting and scanning a 4x4 matrix keypad with the LPC1768 microcontroller.

📎 https://www.electronics-lab.com/project/interfacing-4x4-matrix-keypad-lpc1768-microcontroller

- **Keil uVision IDE – Arm Development Tools**

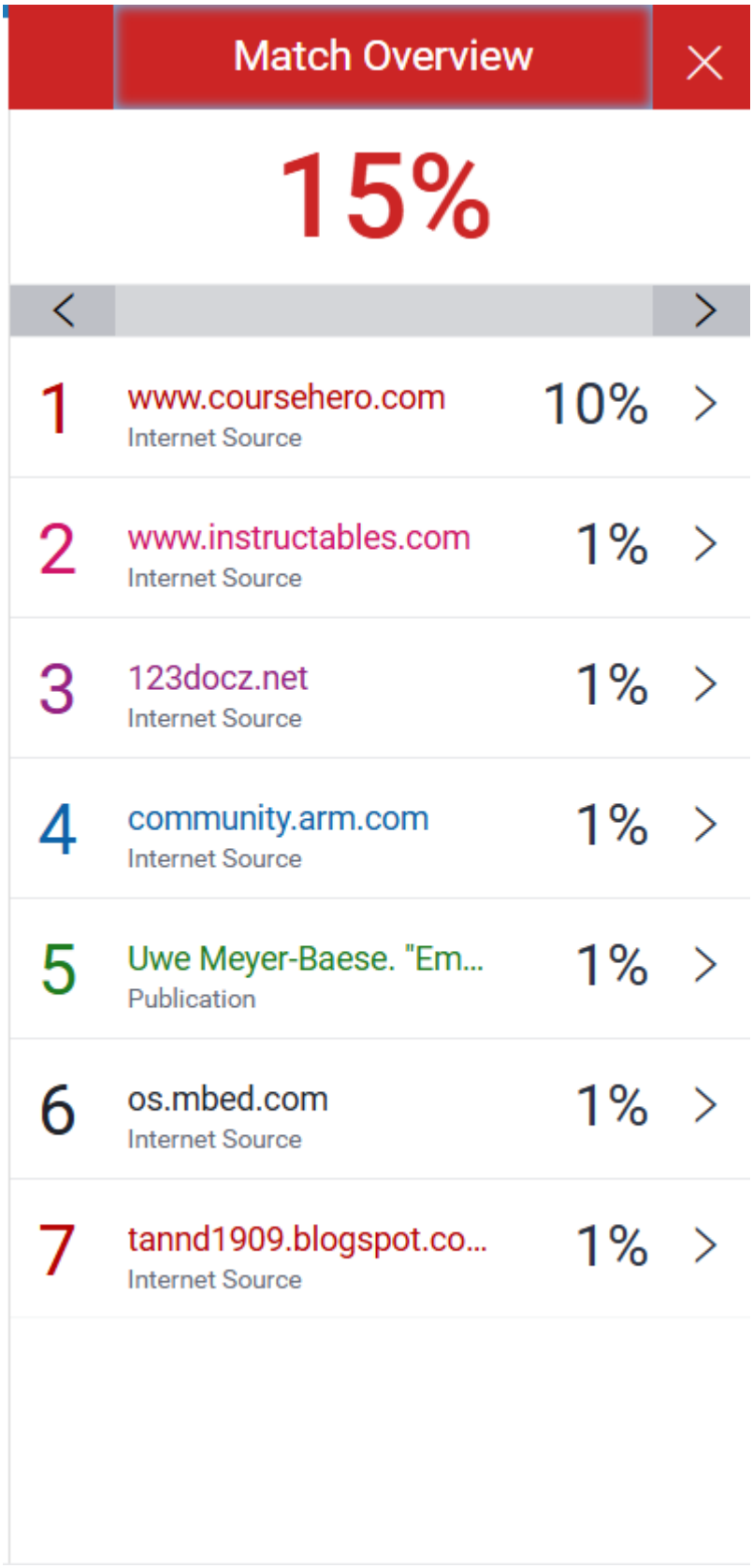
Official IDE for developing and debugging embedded C applications on ARM microcontrollers.

📎 https://www.keil.com/mdk5/

- **LCD Interfacing with LPC1768 – BASCOM Tutorials**

Guide for connecting and programming a 16x2 LCD in 4-bit mode with LPC1768.

📎 https://www.bascom-tutorials.com/lcd-interfacing-with-lpc1768


Match Overview
15%
1
www.coursehero.com
Internet Source
10%
2
www.instructables.com
Internet Source
1%
3
123docz.net
Internet Source
1%
4
community.arm.com
Internet Source
1%
5
Uwe Meyer-Baese. "Em...
Publication
1%
6
os.mbed.com
Internet Source
1%
7
tannd1909.blogspot.co...
Internet Source
1%