🔐CircuitGuard: An Embedded PIN-Based Security System using PIC16F877A✨

C code for MPLAB IDE

```
#include <htc.h>
#include <string.h>
#define _XTAL_FREQ 20000000

// LCD pins
#define RS RB0
#define EN RB1
#define D4 RB4
#define D5 RB5
#define D6 RB6
#define D7 RB7

// Keypad
#define ROW1 RD0
#define ROW2 RD1
#define ROW3 RD2
#define ROW4 RD3
#define COL1 RD4
#define COL2 RD5
#define COL3 RD6

// LEDs and Buzzer
#define GREEN_LED RC0
#define RED_LED RC1
#define BUZZER RC2

const char password[5] = "0625";
char entered[5];
char key;
unsigned int attempt = 0;

// LCD Functions
void lcd_cmd(char cmd);
void lcd_data(char data);
void lcd_init();
void lcd_clear();
void lcd_print(const char *str);
void lcd_set_cursor(char row, char col);

// Keypad
char get_key();
```

```
char scan_keypad();

void main() {
    TRISB = 0x00;
    TRISC = 0x00;
    TRISD = 0b11110000; // RD0–RD3 rows as output, RD4–RD6 columns as input

    PORTB = PORTC = 0x00;
    PORTD = 0xF0; // Set all rows high to avoid floating

    lcd_init();
    lcd_set_cursor(1, 1);
    lcd_print("Protected By"); // line 1
    lcd_set_cursor(2, 1);
    lcd_print("CircuitGuard");       // line 2
    __delay_ms(10);

    lcd_clear();
    lcd_set_cursor(1, 1);
    lcd_print("Hi! Enter PIN");

    while (1) {
        int i = 0;
        memset(entered, 0, sizeof(entered));
        lcd_set_cursor(2, 1);

        while (i < 4) {
            key = get_key();
            if (key != 'N') {
                entered[i++] = key;
                lcd_data(key); // Show actual digit
            }
        }

        __delay_ms(20);

        if (strcmp(entered, password) == 0) {
            GREEN_LED = 1;
            RED_LED = BUZZER = 0;

            lcd_clear();
            lcd_set_cursor(1, 1);
            lcd_print("Welcome");
            lcd_set_cursor(2, 1);
            lcd_print("Access Granted");
            __delay_ms(50);
```

```
        lcd_clear();
        lcd_set_cursor(1, 1);
        lcd_print("Protected By"); // line 1
        lcd_set_cursor(2, 1);
        lcd_print("CircuitGuard");       // line 2
        __delay_ms(10);
        lcd_clear();
        lcd_set_cursor(1, 1);
        lcd_print("Hi! Enter PIN");
        GREEN_LED = 0;
        attempt = 0;
    } else {
        RED_LED = 1;
        GREEN_LED = BUZZER = 0;

        lcd_clear();
        lcd_set_cursor(1, 1);
        lcd_print("Access Denied");
        lcd_set_cursor(2, 1);
        lcd_print("Try Again");
        __delay_ms(50);
        RED_LED = 0;
        attempt++;

        if (attempt >= 3) {
            lcd_clear();
            lcd_set_cursor(1, 1);
            lcd_print("System Locked");
            lcd_set_cursor(2, 1);
            lcd_print("Try after 10s");

            __delay_ms(600); // Remaining time
            attempt = 0;

            lcd_clear();
            lcd_set_cursor(1, 1);
            lcd_print("Protected By"); // line 1
            lcd_set_cursor(2, 1);
            lcd_print("CircuitGuard");       // line 2
            __delay_ms(10);
            lcd_clear();
            lcd_set_cursor(1, 1);
            lcd_print("Hi! Enter PIN");
        } else {
            lcd_clear();
            lcd_set_cursor(1, 1);
            lcd_print("Protected By"); // line 1
            lcd_set_cursor(2, 1);
```

```
                lcd_print("CircuitGuard");       // line 2
                __delay_ms(10);
                lcd_clear();
                lcd_set_cursor(1, 1);
                lcd_print("Hi! Enter PIN");
            }
        }
    }
}

char get_key() {
    char k = 'N';
    while (k == 'N') {
        k = scan_keypad();
    }

    while (scan_keypad() != 'N'); // Wait until key released
    __delay_ms(20); // Debounce
    return k;
}

char scan_keypad() {
    char keypad[4][3] = {
        {'1', '2', '3'},
        {'4', '5', '6'},
        {'7', '8', '9'},
        {'*', '0', '#'}
    };

    PORTD = 0xF0; // All rows high by default

    for (int row = 0; row < 4; row++) {
        PORTD = ~(1 << row); // Drive one row low, others high
        __delay_ms(1);

        if (!COL1) return keypad[row][0];
        if (!COL2) return keypad[row][1];
        if (!COL3) return keypad[row][2];
    }

    return 'N'; // No key pressed
}

// LCD
void lcd_cmd(char cmd) {
    RS = 0;
    PORTB = (PORTB & 0x0F) | (cmd & 0xF0);
    EN = 1; __delay_ms(1); EN = 0;
```

```
    __delay_ms(1);
    PORTB = (PORTB & 0x0F) | (cmd << 4);
    EN = 1; __delay_ms(1); EN = 0;
    __delay_ms(2);
}

void lcd_data(char data) {
    RS = 1;
    PORTB = (PORTB & 0x0F) | (data & 0xF0);
    EN = 1; __delay_ms(1); EN = 0;
    __delay_ms(1);
    PORTB = (PORTB & 0x0F) | (data << 4);
    EN = 1; __delay_ms(1); EN = 0;
    __delay_ms(2);
}

void lcd_init() {
    __delay_ms(20);
    lcd_cmd(0x02); // Initialize 4-bit mode
    lcd_cmd(0x28); // 2-line display, 5x8 font
    lcd_cmd(0x0C); // Display ON, Cursor OFF
    lcd_cmd(0x06); // Entry mode
    lcd_cmd(0x01); // Clear display
    __delay_ms(2);
}

void lcd_clear() {
    lcd_cmd(0x01);
    __delay_ms(2);
}

void lcd_print(const char *str) {
    while (*str) lcd_data(*str++);
}

void lcd_set_cursor(char row, char col) {
    char pos[] = {0x80, 0xC0};
    lcd_cmd(pos[row - 1] + col - 1);
}
```