VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

B. Tech – Electronics & Communication Engineering

BECE403E – EMBEDDED SYSTEMS DESIGN

LAB RECORD

(lab slot L27+L28)

Submitted By

22BEC1205 – Jayakrishnan Menon

Submitted To

Dr. S.Muthulakshmi

DATE: 13/03/2025

Slot: L27+L28

Date: 13/03/2025

LAB – 09: Working with Keypad

AIM: To understand the interfacing of a Keypad with Nucleo64-STM32L152RE Board for applications involving the measure of distances and proximity. We will also perform the following tasks:

Lab Task-1: Write a mbed C++ program to accepts input from a 4x3 keypad and displays it on the serial monitor and LCD. Assume LCD operates in 4-bit with EN and RS active state. Design and verify this logic on Nucleo 152RE board using online Keil Studio platform.

Lab Task-2: Write a mbed C++ program to design a password based door locking system in which the system accept 4-digit password (last 4-digit of your reg. no) via keypad. Use the LCD display to message "Enter Password" on the first line and display * symbol on the second line of the LCD to represent every digit of the password entered. Check whether entered password matches with actual password, if matched activate the servo to open the door else also display message "Correct Password" on LCD first line and "Door opening" message on the LCD second line. If password not matched, activate the buzzer also display message "Incorrect Password" on LCD first line and "Door can't open" message on the LCD second line. Assume LCD operates in 4-bit with EN and RS active state. Design and verify this logic on Nucleo 152RE board using online Keil Studio platform.

Lab Task-3: Write a mbed C++program to design home security system that consists of three main modules (1) Intruder detection (2) Activate fence light during night time (2) password based door lock system. (Challenging Task)

SOFTWARE REQUIRED: ARM Keil Studio (Mbed Online Compiler), Tera Term

HARDWARE REQUIRED: Micro USB cable, NUCLEO64-STM32L152 Board, Potentiometer, Jumper Wires (M-F and M-M), Breadboard, LCD, Servo Motor and an Ultrasonic Sensor (HCSR04)

PROCEDURE:

1. Go to ARM Keil Studio (https://studio.keil.arm.com) and log in
2. Select File → New → Mbed Project
3. Click the Example project drop-down list and select "mbed2-example-blinky"
4. In Project name field, provide the name of the new project and click Add project
5. Double click on the "main.cpp" file from the newly created project folder
6. Modify the code in the editor window as per the logic of your application
7. Check for any errors in the program under the "Problems" tab of the panels window
8. If no errors, connect the Nucleo Board to the computer using Micro USB Cable
9. Click Play icon (Run project) to upload and start the code execution on the board.

PROGRAMS:

Lab Task 1: Write a mbed C++ program to accepts input from a 4x3 keypad and displays it on the serial monitor and LCD. Assume LCD operates in 4-bit with EN and RS active state. Design and verify this logic on Nucleo 152RE board using online Keil Studio platform.

Code:

```
#include "mbed.h"
#include "keypad.h"
#include "TextLCD.h"

Serial pc(USBTX, USBRX);`
TextLCD lcd(PC_0, PC_1, PB_0, PA_4, PA_1, PA_0);
Keypad keypad(PA_10, PB_3, PB_5, PB_4, PB_10, PA_8, PA_9, PC_7);

char key;
int main() {
    keypad.enablePullUp();
    lcd.printf("Pressed Key: ");
    pc.printf("Pressed Key: \n\r");
    while (1) {
        while (keypad.getKey() == '\0');
        key = keypad.getKey();
        lcd.locate(12,0);
        lcd.printf("%c", key);
        pc.printf("%c\r\n", key);
        wait(1);
    }
}
```

Output:

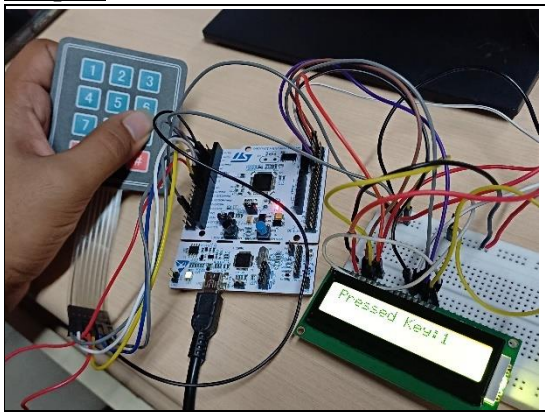
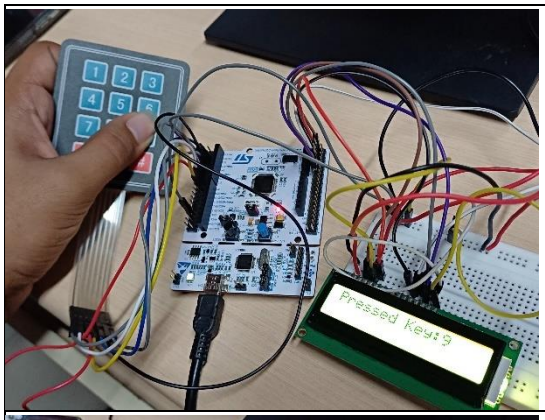
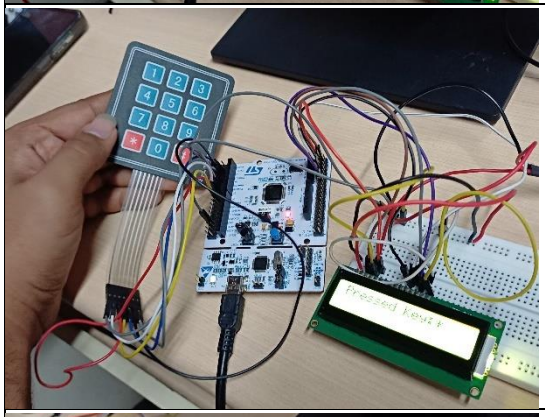
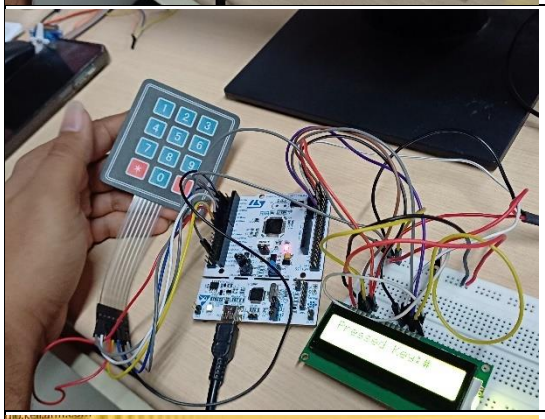
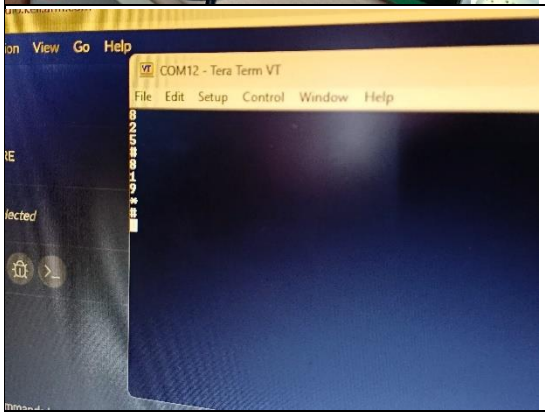


Fig. 1.1: When key "1" is pressed on the keypad, it appears on the LCD.

	**Fig. 1.2:** When key “9” is pressed on the keypad, it appears on the LCD.
	**Fig. 1.3:** When key “*” is pressed on the keypad, it appears on the LCD.
	**Fig. 1.4:** When key “#” is pressed on the keypad, it appears on the LCD.
	**Fig. 1.5:** Whenever a key is pressed, the corresponding character also appears on the Serial Terminal. Here “1”, “9”, “*” and “#” appears one after the other due to them being pressed in that order

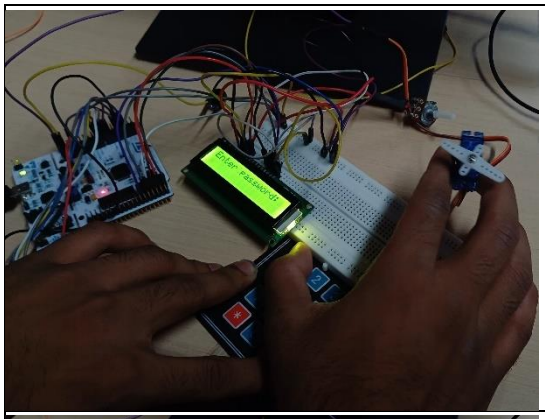
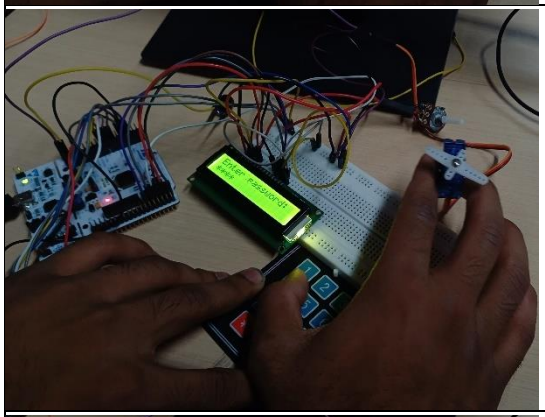
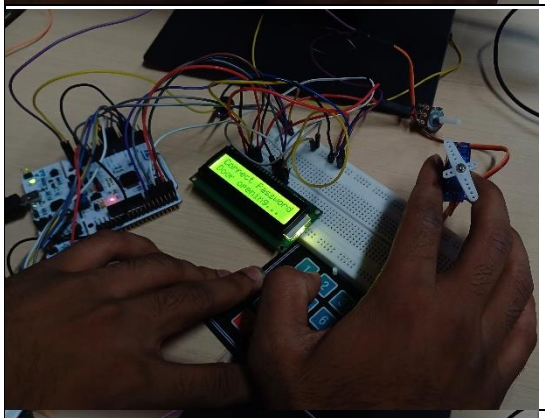
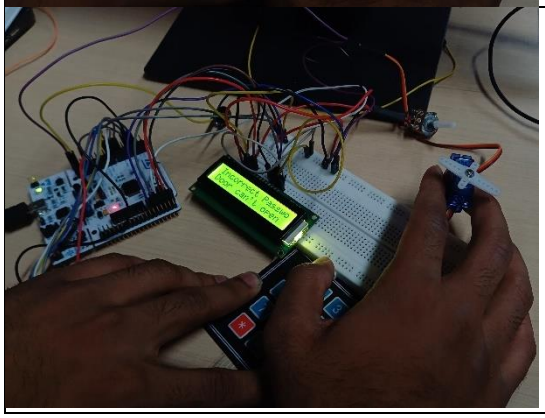
Lab Task 2: Write a mbed C++ program to design a password based door locking system in which the system accept 4-digit password (last 4-digit of your reg. no) via keypad. Use the LCD display to message "Enter Password" on the first line and display * symbol on the second line of the LCD to represent every digit of the password entered. Check whether entered password matches with actual password, if matched activate the servo to open the door else also display message "Correct Password" on LCD first line and "Door opening" message on the LCD second line. If password not matched, activate the buzzer also display message "Incorrect Password" on LCD first line and "Door can't open" message on the LCD second line. Assume LCD operates in 4-bit with EN and RS active state. Design and verify this logic on Nucleo 152RE board using online Keil Studio platform.

Code:

```
#include "mbed.h"
#include "keypad.h"
#include "TextLCD.h"

TextLCD lcd(PC_0, PC_1, PB_0, PA_4, PA_1, PA_0);
Keypad keypad(PA_10, PB_3, PB_5, PB_4, PB_10, PA_8, PA_9, PC_7);
DigitalOut buzz(PC_6);
PwmOut servo(PC_8);

const char ACTUAL_PW[4] = {'1', '2', '3', '4'};
char PW_ENTERED[4];
int count;

int main() {
    keypad.enablePullUp();
    while(1) {
        lcd.cls();
        lcd.locate(0, 0);
        lcd.printf("Enter password:");
        servo.period_ms(20);
        servo.pulsewidth_us(500);
        buzz = 0;

        count = 0;
        while(count < 4) {
            char key = '\0';
            while (key == '\0') {
                key = keypad.getKey();
            }
            PW_ENTERED[count] = key;
            lcd.locate(count, 1);
            lcd.printf("*");
            count++;
            wait(0.5);
        }

        bool correct = true;
        for (int i = 0; i < 4; i++) {
            if (PW_ENTERED[i] != ACTUAL_PW[i]) {
```

```
                correct = false;
                break;
            }
        }

        lcd.cls();
        if (correct) {
            lcd.locate(0, 0);
            lcd.printf("Correct Password");
            lcd.locate(0, 1);
            lcd.printf("Door opening...");
            servo.pulsewidth_us(2000);
            wait(3);
            servo.pulsewidth_us(500);
        } else {
            lcd.locate(0, 0);
            lcd.printf("Incorrect Password");
            lcd.locate(0, 1);
            lcd.printf("Door can't open");
            buzz = 1;
            wait(2);
            buzz = 0;
        }
    }
}
```

Output:

	**Fig. 2.1:** Starting Screen with a prompt to enter the password.
	**Fig. 2.2:** Entered password appears as ****.
	**Fig. 2.3:** Entering the correct password will display a prompt indicating that the password was correct. The Servo motor also actuates an angular displacement as an indication of the door opening.
	**Fig. 2.4:** Entering the wrong password will display a prompt indicating that the password was wrong. The Servo motor stays at its position.

Lab Task 3: Write a mbed C++program to design home security system that consists of three main modules (1) Intruder detection (2) Activate fence light during night time (3) password based door lock system.

- The intruder detection system consists of PIR sensor interfaced with processing unit to detect and alert under human presence condition.
- Automatically activate the fence lights during night time using LDR.
- Password based door lock system uses numeric keypad to accept the password (last 4-digit of your reg. no) from user and LCD to display the message whether permission is granted or not. Upon receiving correct password signal, enable motor to open the door.

In case of password mismatch or intruder detection condition activate the buzzer. Assume LCD operates in 4-bit with EN and RS active state. Design and verify this logic on Nucleo 152RE board using online Keil Studio platform.

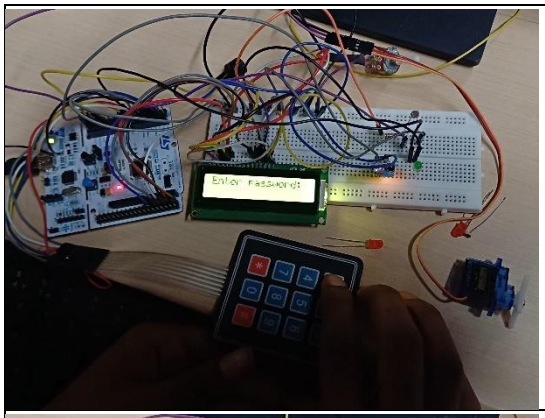
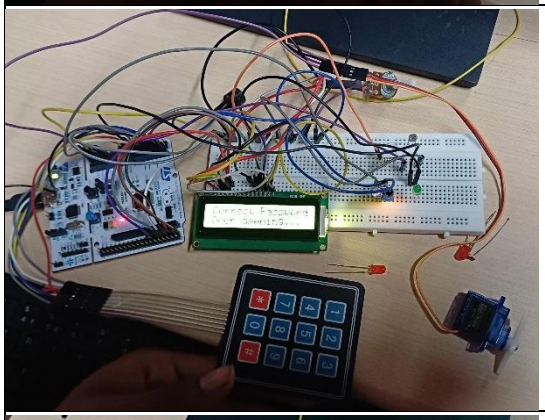
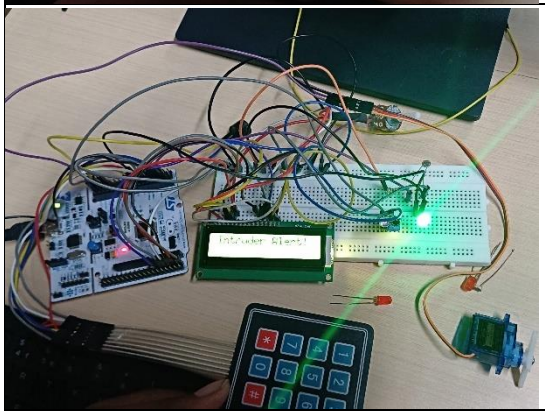
Code:

```
#include "mbed.h"
#include "keypad.h"
#include "TextLCD.h"

TextLCD lcd(PC_0, PC_1, PB_0, PA_4, PA_1, PA_0);
Keypad keypad(PA_10, PB_3, PB_5, PB_4, PB_10, PA_8, PA_9, PC_7);

DigitalOut buzz(PC_6);
PwmOut servo(PC_8);
DigitalIn pirSensor(PC_9);  // PIR motion sensor
AnalogIn ldr(PC_4);         // LDR for light detection
DigitalOut fenceLight(PB_2);

const char ACTUAL_PW[4] = {'1', '2', '4', '5'};
char PW_ENTERED[4];
int count;

int main() {
    keypad.enablePullUp();
    servo.period_ms(20);
    servo.pulsewidth_us(500);  // Lock door position
    buzz = 0;

    while (1) {
        // *Intruder Detection (PIR Sensor)*
        if (pirSensor.read() == 1) {
            lcd.cls();
            lcd.locate(0, 0);
            lcd.printf("Intruder Alert!");
            buzz = 1;
            wait(2);
            buzz = 0;
        }

        // *Fence Light Activation (LDR)*
        if (ldr.read() < 0.3) {  // If it's dark
```

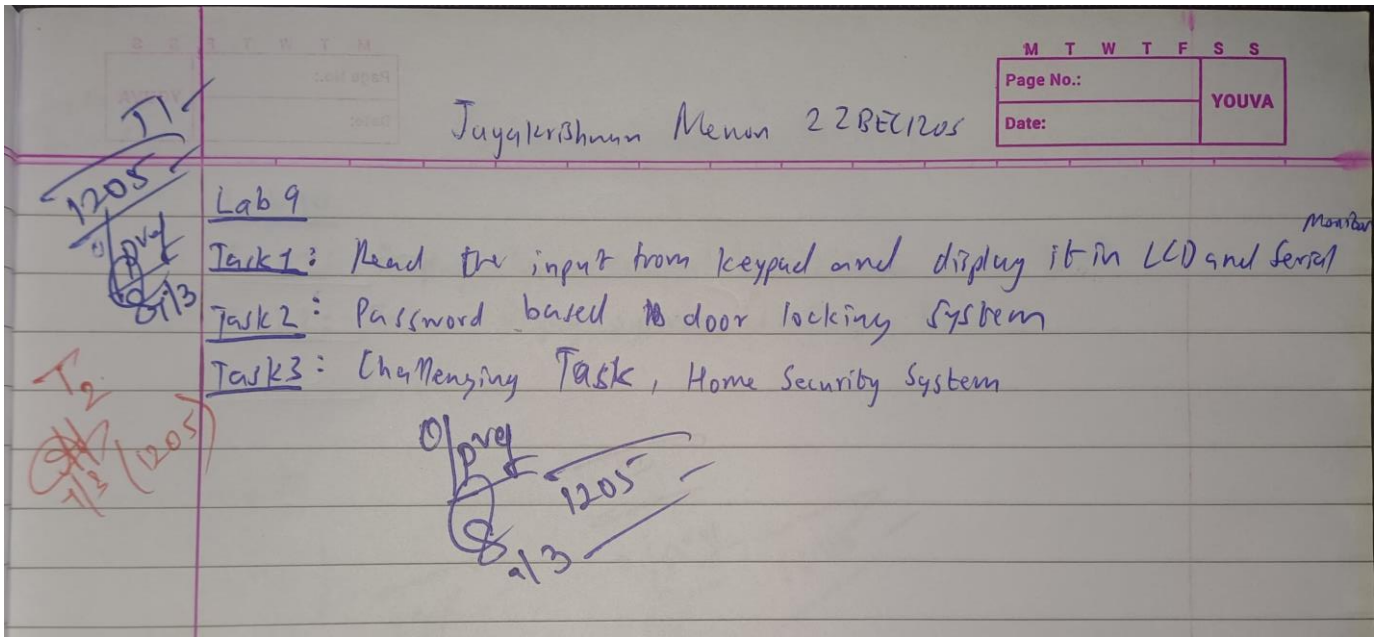
```
            fenceLight = 1;      // Turn on lights
        } else {
            fenceLight = 0;      // Turn off lights
        }

        // *Password-Based Door Lock*
        lcd.cls();
        lcd.locate(0, 0);
        lcd.printf("Enter password:");
        count = 0;

        while (count < 4) {
            char key = '\0';
            while (key == '\0') {
                key = keypad.getKey(); // Wait for key press
            }
            PW_ENTERED[count] = key;
            lcd.locate(count, 1);
            lcd.printf(""); // Display '' for entered digits
            count++;
            wait(0.5); // Debounce delay
        }

        bool correct = true;
        for (int i = 0; i < 4; i++) {
            if (PW_ENTERED[i] != ACTUAL_PW[i]) {
                correct = false;
                break;
            }
        }

        lcd.cls();
        if (correct) {
            lcd.locate(0, 0);
            lcd.printf("Correct Password");
            lcd.locate(0, 1);
            lcd.printf("Door opening...");
            buzz = 0;
            servo.pulsewidth_us(2000); // Unlock door
            wait(3);
            servo.pulsewidth_us(500);  // Lock door again

        } else {
            lcd.locate(0, 0);
            lcd.printf("Incorrect Password");
            lcd.locate(0, 1);
            lcd.printf("Door can't open");
            buzz = 1;
            wait(2);
            buzz = 0;
```

```
        }
    }
}
```

Output:

	**Fig. 3.1:** Initially, a prompt to enter the password is displayed.
	**Fig. 3.2:** If the password is correct the servo motor actuates an angular displacement.
	**Fig. 3.3:** If the password is wrong or the infrared sensor detects an object, the buzzer beeps and the LCD indicated an intruder alert. If the environment is dark, the LED night light turns ON.

OUTPUT VERIFICATION:



INFERENCE:

1. The Keypad.h library is designed to work with various keypad configurations, including 4x4, 3x4, and other matrix layouts, making it versatile for different applications.
2. The getKey() method is the core function for reading keypad input. It scans the keypad and returns the character corresponding to the pressed key.
3. void enablePullUp() Enables internal PullUp resistors on the columns pins.
4. Using the "DigitalIn" API, we can read the value of a digital pin. And by using the "DigitalOut" API, we can write logical high or low values to a digital pin
5. The PwmOut API in mbed allows you to generate Pulse Width Modulated (PWM) signals on pins that support PWM functionality. This is particularly useful for controlling devices such as LEDs, Buzzers, and servos. Its Syntax is: "PwmOut Identifier(PinName)".
6. The PwmOut API provides two methods, period and period_ms, to configure the period of the Pulse Width Modulation (PWM) signal. These methods are critical for setting the frequency of the PWM signal.
7. The write method in the PwmOut API sets the duty cycle of the PWM signal. The duty cycle determines the proportion of the PWM period for which the signal is HIGH (ON) versus LOW (OFF). This is essential for controlling the intensity, speed, or position of devices like LEDs, motors, and servos.
8. The TextLCD library allows a Nucleo board to control Liquid Crystal Displays (LCDs) based on the Hitachi HD44780 chipset in 4-bit mode. It can be used by including the header file,

"TextLCD.h". The mbed TextLCD library performs the laborious LCD setup routines and also tells the LCD object which pins are used for which functions.

9. The Syntax for using this API is: “TextLCD lcd(rs, enable, d4, d5, d6, d7)”.
10. It also provides 4 main useful functions. The cls() function clears the screen and locates the cursor to 0,0. The locate() function can be used to move the cursor to any of the 16x2 positions available on the LCD. The putc() function writes a character to the LCD. The printf() function writes a formatted string to the LCD.
11. Programmable Delays can be implemented through the use of the “wait()” function.
12. A program can be set to run indefinitely using the “while(1)” loop.

RESULT:

Thus, the interfacing of an Keypad with Nucleo64-STM32L152RE Board for applications involving the measure of distances and proximity was understood and the tasks were also performed successfully.