Project Report

By Tariq Abdullah Alshaya

Table of Contents

Table of Figures

INTRODUCTION

This paper shall discuss and summarize the outcomes that I have learned in embedded systems where I have designed a project using *LPC1768* board. In this project I have designed a car security system that allows the user to access this system without a key. I have added certain features that modern cars do not provide that shall protect the engine's health.

ENGINEERING PROBLEM IDENTIFICATION AND FORMULATION

The user will be able to access the car using a password that will grant the user to access system features that includes:

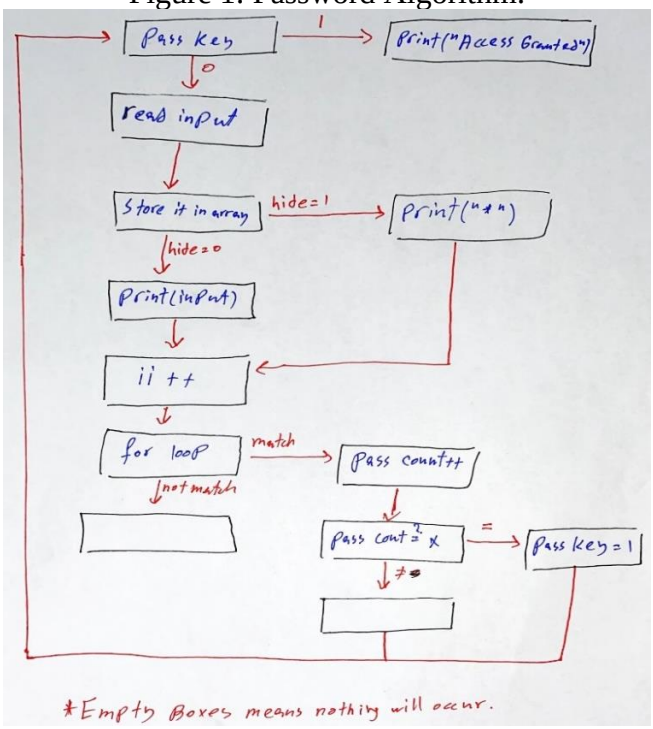
- **Keypad**: User input for password.
- **Potentiometer**: To vary the weathers' temperature where specific scenarios will occur for each weather:
    - If the weather is hot (Temperature $> 30\ °C$): *LED3* will be on to warn the user that the temperature outside is hot, and a message will be printed when the user turn ON the system for the first time to warn the user that the *RPM* of the engine should be low.
    - If the weather is cold (Temperature $\leq 15\ °C$): A message will be printed when the user turn ON the system for the first time to remind the user to warm the engine before driving.

- **Light Detector Sensor (LDR)**: When its dark outside *LED2* will be *ON*, as an automatic way to turn *ON* the car's light when it gets dark.
- **Seven Segment Display**: A counter that will provide the user a real-timer counter, that counts from 0 to 9 in seconds.
- **Ultrasonic Sensor**: Sensor that will measure the distance of the object that is in front of the sensor, if this object is close to the sensor *LED1* will be blinking to warn the user of a close object, and the user can issue a command to know how far is that object.
- **Liquid Crystal Display (LCD):** An *LCD* screen that will provide the user with information regarding temperature, distance, and the password.

DESCRIPTION OF ANY USED ALGORITHM

Password algorithm: This algorithm not only checks if the entered password is correct it also provides the user the functionality whether to hide or show the entered password that is printed on the *LCD*.

Figure 1: Password Algorithm.



PROBLEMS FACED AND HOW THEY WERE SOLVED

The *LCD* have 16 pins and due its size it will take space in the *Breadboard* which will not allow me to install the *Seven Segment Display* and the *LDR*. I used *Female-to-Male* wires to connect the *LCD* without installing it in the *Breadboard* to have enough space for the other components.

When printing the statement "*Powering ON the system*" I printed after it three dots that should print a dot after 0.7 seconds three times to simulate real system. I used this command "*printf("*")*" which printed the asterisks all together at once, after researching and experimenting I found out that I did not prefix this command to *Serial*, and the following command solved this issue "*pc.printf("*")*" where *pc* is *Serial* object.

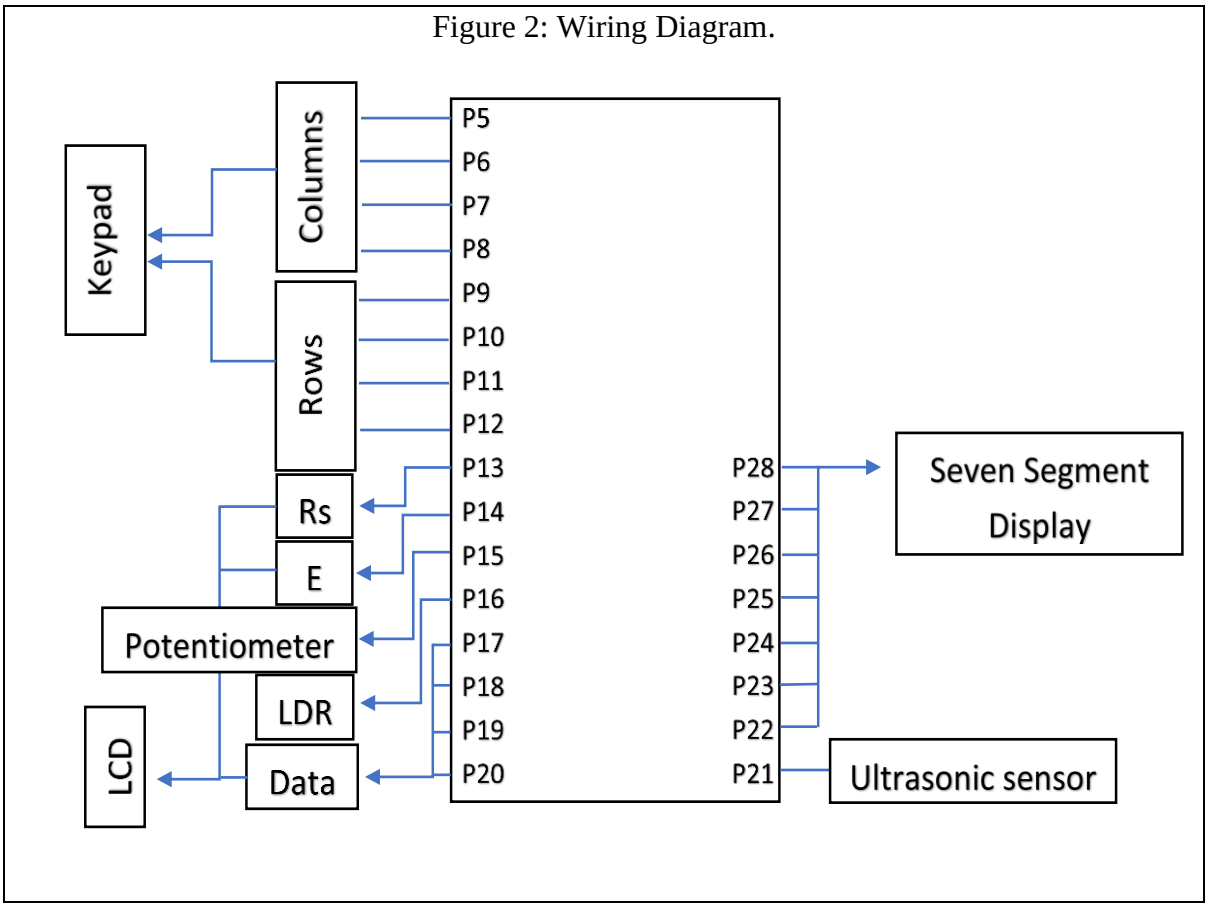
When implementing the "*Hide_Show()*" function, if the user entered a password "*741A*" then the user issues the command to hide the password the output will be "****" if the user entered more digits the output will be "****8520" here the issue is that the written part was only hidden, and further inputs was not. Here the '*hide*' variable solves this issue, so that if the user wants to hide the password '*hide = 1'* which will keep printing asterisks instead of the input itself, but if the user wants to show the input '*hide = 0'* the input of the user will be printed.

When writing the "*Read_Input()*" function I was including the "*Hide_Show()*" function inside it, which caused a security breach where the user can access certain features without entering the password. This breach allows the user to access the temperature, distance, and the counter features. Where I solved this breach by separating the "*Hide_Show()*" in a different function to provide more secure system.

DESCRIBE THE FUNCTIONALITY

LPC1768 board: In *Figure 2*, is the summarized version of the wiring diagram that is used in this project.

Figure 2: Wiring Diagram.



Keypad: The *Keypad* consist of four rows and four columns. Choosing between rows and columns, so that one of them can be set to an input while the other is output. Fixing the columns as an output and set them to *Logic 0*, since in *LPC1768* input's state is by default *HIGH*. As can be seen in *Figure 4*, if a button was pressed the row will short circuit with its corresponding column resulting in *Logic 0* in that row. For example, if the user issues the most top left button, *R1* will short circuit with *C1* resulting in *Logic 0* in *R1*.

Figure 3: 4x4 Keypad pins.

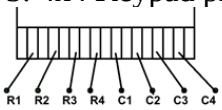
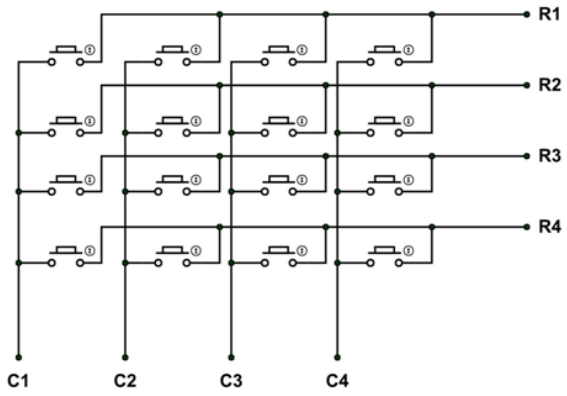
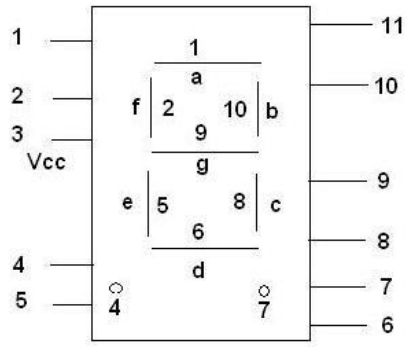


Figure 4: 4x4 Keypad diagram.



Seven Segment Display: In *Figure 5*, is the *Seven Segment Display* diagram where it is shown what pins can turn *ON* or *OFF* different bars. *Logic 0* in pin number 10 will turn *ON* bar '*b*', whereas *Logic 1* will turn it *OFF*. For example, to print number Zero in this display, all bars must be *ON*, *Logic 0*, except these bars '*g, 4, 7*' must be *OFF* using *Logic 1*.

Figure 5: Seven Segment Display diagram.



Ultrasonic Sensor: This sensor consists of two parts; one is *Transmitter*, and the other is *Receiver*. The *Transmitter* will send sound waves that will be reflected by an object that is in front of it, and this reflected wave will be received by the *Receiver* which will calculate the time delay in order to calculate the distance. This sensor has three pins, one for ground, one for voltage, and the other is a *Half-duplex Signal* pin that reads and sends data from and to the sensor.

CONCLUSION

In this project I have implemented different concepts that I have learned in this course, I have used *Interrupts* and *Timers* in the *Ultrasonic* sensor, *Analog-to-Digital* in both of *Potentiometer* and *Light Detector Sensor*, *UART* communication to read and output data from and to *CoolTerm* software. I have learned how to use *Mbed* Libraries to maintain a clean organized code and avoid reinventing the wheel.

CODE APPENDIX

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

Serial pc(USBTX, USBRX);//UART
DigitalOut Pins[] = { (LED1), (LED2), (LED3) };// LED1 for distance, LED2 for LDR, LED3
for Temperature.
TextLCD lcd(p13, p14, p17, p18, p19, p20); // rs, e, d[4-7].
BusOut Display(p22, p23, p24, p25, p26, p27, p28);
Keypad keypad(p8, p7, p6, p5, p12, p11, p10, p9);//Columns, Rows.
AnalogIn LDR(A1);// A1 = A0.1 p16.
AnalogIn Potentiometer(A0);// A0 = A0.0 p15.
Ping p(p21); //Ping object.

char input, key;
double Temperature, ADC_Result;
int range, ii = 0;
int PassKey = 0;
int hide = 0;
int PassCount = 0;
char Password[8];
char TruePassword[9] = "741A8520";//Here you can change the password. And remember to
change line #187 "PassCount"
```

```
/*_____________________________________________________________________*/

void Temperature_Calc() {
    Temperature = Potentiometer.read() * 50;//To limit the temperature [0-50]C.
}

void Print_Number(int number) { // 0 is ON, 1 is OFF.
    if (number == 0)
        Display = 0b0100000;
    if (number == 1)
        Display = 0b0101111;
    if (number == 2)
        Display = 0b0010010;
    if (number == 3)
        Display = 0b0000110;
    if (number == 4)
        Display = 0b0001101;
    if (number == 5)
        Display = 0b1000100;
    if (number == 6)
        Display = 0b1000000;
    if (number == 7)
        Display = 0b0101110;
    if (number == 8)
        Display = 0b0000000;
    if (number == 9)
        Display = 0b0000100;
}

void Hide_Show() {//Hide_Show was separated from Read_Input for security purposes so
that the user won't be able to access temperature till he types the password.
    if (pc.readable()) {// if the user pressed a key.
        input = pc.getc();
        if (input == 'h') {
            lcd.locate(0, 1);
            hide = 1;
            for (int j = 0; j < ii; j++)
                lcd.printf("*");
        }
        if (input == 's') {
            hide = 0;
            lcd.locate(0, 1);
            for (int j = 0; j < ii; j++)
                lcd.printf("%c", Password[j]);
        }
    }
}

void Read_Input() {
    if (pc.readable()) {// if the user pressed a key.
        input = pc.getc();
        if (input == 'd') {
            lcd.cls();//Clear LCD in go to (0,0) location on the LCD.
            lcd.printf("Distance:");
            lcd.locate(0, 1);
            lcd.printf("%d CM", range);
            pc.printf("The distance = %d cm\n\r", range);
```

```
            if (range < 30)
                pc.printf("Be careful! Too close.\n\r");
        }
        if (input == 't') {
            lcd.cls();//Clear LCD in go to (0,0) location on the LCD.
            lcd.printf("Temp:");
            lcd.locate(0, 1);
            lcd.printf("%.2f C", Temperature);
            pc.printf("The temperature = %.2f C\n\r", Temperature);
            if (Temperature > 35)
                pc.printf("Be careful! The temperature is high.\n\r");
        }
        if (input == 'c') {
            pc.printf("Timer started, you will not have access to the system until the
timer finishes.\n\r");
            for (int i = 0; i < 10; i++) {
                Print_Number(i);
                wait(1);
            }
            Display = 0b1111111;    //Turn OFF 7segments LEDs.
        }
    }
}

void Tempreture_Distance_Check() {
    Temperature_Calc();
    if (Temperature > 35) {//Hot
        Pins[2] = 1;
        wait(0.3);
    }
    if (Temperature <= 35) {//Normal.
        Pins[2] = 0;
    }
    p.Send();
    wait_ms(30);
    range = p.Read_cm();
    if (range < 30) {
        Pins[0] = 1;
        wait(0.3);
    }
    if (range <= 30) {
        Pins[0] = 0;
    }
}

void LDR_Check() {
    if (LDR.read() < 0.48)
        Pins[1] = 1;
    else
        Pins[1] = 0;
}

void print_Output() {
    pc.printf("|_____________________________________________________________\n\r");
    pc.printf("|    Powering ON the system");
    for (int i = 0; i < 4; i++) {
        pc.printf(".");
        wait(0.7);
```

```
    }
    pc.printf("\n\r");
    wait(0.1);
    Temperature_Calc();
    pc.printf("|    The temperature is : %.2f \n\r", Temperature);
    if (Temperature > 35) //Hot
        pc.printf("|    It is hot outside, try to keep the RPM below 4. \n\r");
    if (Temperature <= 15) //Cold.
        pc.printf("|    It is cold outside remember to warm the engine. \n\r");

    pc.printf("|    The time is : ");
    LDR_Check();
    if (Pins[1] == 0)
        pc.printf("Day\n\r");
    else
        pc.printf("Night\n\r");
    pc.printf("|    Press 'd' to print the distance\n\r");
    pc.printf("|    Press 't' to print the temperature\n\r");
    pc.printf("|    Press 'c' to start the timer\n\r");
    pc.printf("|_____________________________________________________________\n\r");
}

void User_Login() {
    pc.baud(9600); // Baud rate.
    pc.format(8, SerialBase::None, 1); // 8-bits, no parity, 1 stop bit.
    pc.printf("Enter the password using the Keypad\n\r");
    lcd.printf("Password");
    lcd.locate(0, 1);
    pc.printf("Enter 'h' to hide the password\n\r");
    pc.printf("Enter 's' to show the password\n\r");
}

void Keypad_Init() {
    while (PassKey == 0) {
        Hide_Show();
        key = keypad.getKey();
        if (key != KEY_RELEASED) {
            Password[ii] = key;
            if (hide == 0)
                lcd.printf("%c", Password[ii]);
            if (hide == 1)
                lcd.printf("*");
            ii = ii + 1;
            wait(0.6);

            for (int i = 0; i < ii; i++) {
                if (TruePassword[i] == Password[i]) {
                    PassCount = PassCount + 1;
                }
            }
            if (PassCount == 36)
                PassKey = 1;
        }
    }
    pc.printf("Access Granted!\n\r");
    print_Output(); //Print it if the password is True.
}
/*_____________________________________________________________*/
```

```
int main() {
    User_Login();
    Display = 0b1111111;    //Turn OFF 7segments LEDs.
    while (1) {
        if (PassKey == 0)
            Keypad_Init();
        Temperature_Calc();
        Read_Input();
        Tempreture_Distance_Check();
        LDR_Check();
    }
}
```

REFERENCES

Project Implementation: https://youtu.be/NEGA-UIUzho

Figure 3: https://components101.com/misc/4x4-keypad-module-pinout-configuration-features-datasheet

Figure 4: https://components101.com/misc/4x4-keypad-module-pinout-configuration-features-datasheet

Figure 5: http://www.cs.binghamton.edu/~steflik/cs423/7segment/sevenseg.html