EMBEDDED SYSTEMS LAB PROJECT

V SEMESTER, IT B1 BATCH

DEPARTMENT OF I&CT, MIT, MANIPAL

TOPIC: AUTHENTICATOR AND PASSWORD MANAGER SYSTEM

TEAM 4 MEMBERS:

Sr.no	Name	Registration no.	Roll no.	Branch & class
1	Shashvat Tiwari	210911196	36	IT-B1
2	Yash Verdhan Samania	210911066	16	IT-B1
3	Zaid Khan	210911068	17	IT-B1

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Institution of Manipal University

TABLE OF CONTENTS

PROBLEM STATEMENT

Write an embedded C program to create a PASSWORD-AUTHENTICATOR manager using LPC1768 microcontroller and displaying the correct password of a user on LCD. If the user is right the password will then be authenticated and user will get to enter .

ABSTRACT

This project proposes the development of an embedded system leveraging the Philips LPC1768 microcontroller kit, FRC cables, and jumper cables to create a Tiled Password and Authenticator Manager. The primary objective is to design a secure and efficient system for managing passwords and authentication through innovative tiled interface mechanisms.

The LPC1768 microcontroller will serve as the core processing unit, handling the logic and interface between the user and the authentication modules. FRC cables and jumper cables will be utilized for interconnecting components, enabling seamless communication and power supply across the system.

Aims:

1. **Tiled Interface Development:** Implement a tiled interface that allows users to securely input and manage passwords. The tiled mechanism aims to provide an intuitive and visually appealing interaction method, enhancing user experience while ensuring robust security measures.
2. **Advanced Security Implementation:** Employ advanced encryption algorithms and security protocols to safeguard stored passwords and user data. Explore multiple authentication methods, potentially integrating biometric or two-factor authentication, to enhance overall system security.
3. **Versatile Application:** Develop a flexible system suitable for various applications, ranging from personal security to industrial access control systems. The system's adaptability and modularity will be emphasized, enabling integration into diverse environments while maintaining efficient resource utilization.

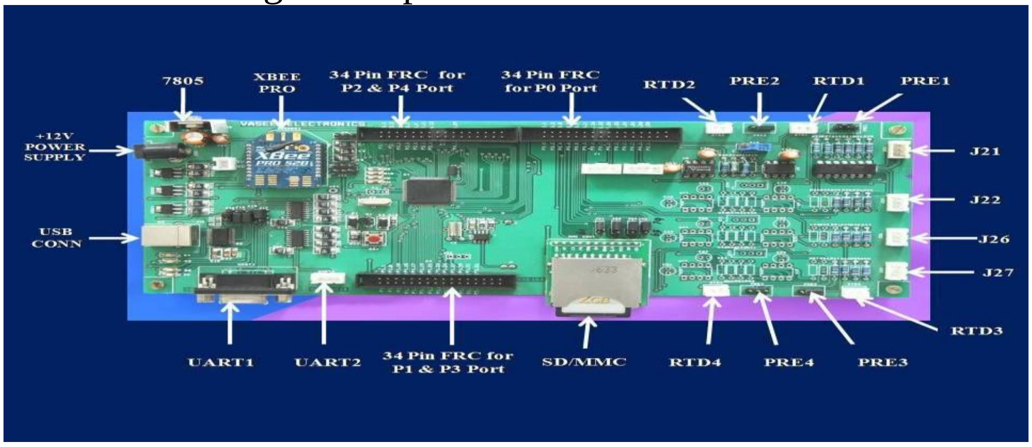
The project aims to showcase the LPC1768 microcontroller's capabilities in conjunction with FRC and jumper cables, creating a secure, adaptable, and user-centric Password and Authenticator Manager system. The innovative tiled interface design, coupled with robust security measures, aims to redefine password management and authentication in embedded systems.

HARDWARE USED

The following components have been used

1. **LPC 1768 microcontroller kit**: It is a powerful 32-bit ARM Cortex-M3 processor running up to 100 MHz with 512 KB flash and 32 KB RAM, which makes it far more capable than popular 8-bit prototyping alternatives.

2. **Power supply:-** The LPC1768 microcontroller operates with a recommended power supply voltage ranging from 2.4V to 3.6V. This versatile microcontroller can be powered through various sources, including USB, external supply voltage, or battery power. It features an integrated voltage regulator that allows for flexible power input, enhancing its adaptability to different applications. The microcontroller's low-power modes and efficient power management capabilities ensure optimized energy consumption, making it suitable for battery-operated devices while maintaining robust performance.



3. **FRC Cables**: It is a powerful 32-bit ARM Cortex-M3 processor running upto 100 MHz with 512 KB flash and 32 KB RAM, which makes it far more capable than popular 8-bit prototyping alternatives.

4. **Jumper Cables:** These cables are used make sophisticated connections between the kit and the sensor. Four female to female jumper cables are usedin the setup.

5. **LED**

Software Requirements:

1. **Keil microvision4 simulator**
2. **Flash Magic**

STEPS PERFORMED/Methodology

Steps Performed:

1. **Requirement Analysis:**
   - Identify and define the functional requirements of the Tiled Password and Authenticator Manager system, considering user needs, security standards, and system constraints.
2. **Component Selection:**
   - Choose the LPC1768 microcontroller kit due to its processing power, flexibility, and integrated peripherals suitable for the project's requirements.
   - Select appropriate FRC cables and jumper cables for reliable interconnections among components.
3. **Tiled Interface Design:**
   - Develop the graphical layout and functionality of the tiled interface using suitable programming languages and tools compatible with the LPC1768 platform.
   - Implement user-friendly features for password input, management, and authentication within the tiled interface.
4. **Security Implementation:**
   - Research and integrate advanced encryption algorithms and security protocols into the system for securing stored passwords and user data.
   - Explore and potentially integrate various authentication methods like biometrics or two-factor authentication to enhance system security.
5. **Hardware Integration:**
   - Connect and integrate the LPC1768 microcontroller with other necessary components using FRC cables and jumper cables, ensuring proper communication and power supply.
6. **Testing and Debugging:**
   - Conduct rigorous testing of the system's functionality, including the tiled interface, authentication processes, and overall security measures.
   - Debug and troubleshoot any identified issues to ensure smooth operation and reliability.

Methodologies Proposed:

1. **Agile Development:**
   - Adopt an iterative approach, allowing for continuous improvements and adaptations based on user feedback and evolving security standards.

2. **Modular Design:**
   - Implement a modular system architecture, enabling scalability and ease of maintenance by separating components and functionalities.

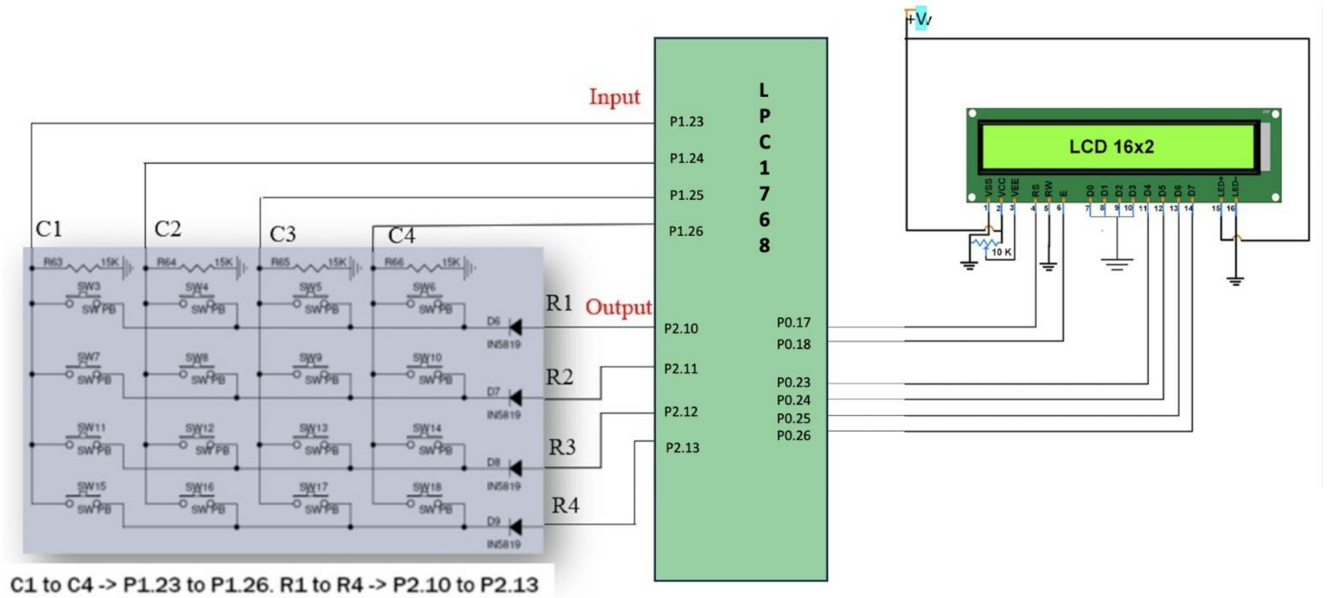
3. **Security-First Approach:**
   - Prioritize security throughout the development process, emphasizing encryption and secure authentication methods to protect user data effectively.

4. **User-Centric Design:**
   - Focus on creating an intuitive and user-friendly tiled interface, considering user behaviors and preferences to enhance usability.

5. **Documentation and Knowledge Sharing:**
   - Maintain comprehensive documentation of the system design, implementation details, and testing procedures to facilitate knowledge transfer and future enhancements.

By following these steps and methodologies, the development of the Tiled Password and Authenticator Manager using LPC1768 aims to create a robust, secure, and user-centric system for managing passwords and authentication.

CIRCUIT DIAGRAM



C1 to C4 -> P1.23 to P1.26. R1 to R4 -> P2.10 to P2.13

CODE

```
#include <stdio.h>
#include <LPC17xx.h>
#include <math.h>
#include <string.h>

#include <lpc17xx.h>
#define RS 27 // P0.27
#define EN 28 // P0.28
#define DT 23 // P0.23 to P0.26 data lines

unsigned long int temp1 = 0, temp2 = 0, i, j, x, count = 0, fail = 0, supcount = 0;
unsigned char flag1 = 0, flag2 = 0;
unsigned char ans;
unsigned char ams[10];
unsigned char op1, op2, oper, ans;
unsigned char msg[] = {"Enter the Password="};
unsigned char msg2[] = {"Checking the password "};
unsigned char msg3[] = {"succesfull authentication"};
unsigned char msg4[] = {"failed authentication retry "};
unsigned char msg5[] = {"all tries are over check again "};
unsigned char msg6[] = {"Enter the Password THAT SHOULD BE OF 4 CHARACTER"};
unsigned char msg7[] = {"IT SHOULD CONTAIN 2 NUMBER 2 ALPHABETS"};

unsigned char message[4][4] = {{'1', '2', '3', '4'}, {'D', 'E', 'F', 'G'}, {'9', 'A', 'B', 'C'}, {'5', '6', '7', '8'}};

unsigned char entered_pwd[4];
unsigned char correct_pwd[4];

int help = 0;
unsigned char col, row, flag, key;
unsigned long var;
unsigned int r;
void scan();

void lcd_write(void);
void port_write(void);
void delay_lcd(unsigned int);
unsigned long int init_command[] = {0x30, 0x30, 0x30, 0x20, 0x28, 0x0c, 0x06, 0x01, 0x80};

void TIM1delay()
{
   LPC_TIM0->TCR = 0X02;
   LPC_TIM0->MCR = 0X02;
   LPC_TIM0->CTCR = 0X0;
   LPC_TIM0->EMR = 0X20;

   LPC_TIM0->PR = 2;
```

```
    LPC_TIM0->MR0 = 1999996;
    LPC_TIM0->TCR = 0X01;
    while (LPC_TIM0->EMR && 0X01 == 0)
        ;
}

void port_write1()
{
    // LPC_GPIO0->FIOPIN = flag1 << 27;
    LPC_GPIO0->FIOPIN = temp2 << 23;
    if (flag1 == 0)
        LPC_GPIO0->FIOCLR = 1 << 27; // RS command
    else
        LPC_GPIO0->FIOSET = 1 << 27; // RS data
    // sending command/data

    LPC_GPIO0->FIOSET = 1 << 28;
    for (i = 0; i < 50; i++)
        ;
    LPC_GPIO0->FIOCLR = 1 << 28;
    for (i = 0; i < 50000; i++)
        ;
}
void delay_lcd(unsigned int r1)
{

    for (r = 0; r < r1; r++)
        ;

    return;
}
void lcd_write1()
{
    temp2 = temp1 & 0xF0;
    temp2 = temp2 >> 4;
    port_write1();

    temp2 = (temp1 & 0x0F);
    port_write1();
}

int main(void)
{
    SystemInit();
    SystemCoreClockUpdate();
    LPC_GPIO0->FIODIR = 1 << RS | 1 << EN | 0XF << DT;
    LPC_PINCON->PINSEL3 &= 0xFFC03FFF; // P1.23 to P1.26 MADE
                                       // GPIO
    LPC_PINCON->PINSEL4 &= 0xF00FFFFF; // P2.10 t P2.13 made
                                       // GPIO
    LPC_GPIO2->FIODIR |= 0x00003C00;   // made output P2.10 to
```

```
                               // P2.13 (rows)
LPC_GPIO1->FIODIR &= 0xF87FFFFF;   // made input P1.23 to
                               // P1.26 (cols)
flag1 = 0;                         // Write the commands in command register
for (i = 0; i < 9; i++)
{
   temp1 = init_command[i];
   lcd_write();
}
//_________________________________________________________________________________
______________________________________________________________

flag1 = 1; // Write the data in data register
i = 0;

while (msg6[i] != '\0')
{
   temp1 = msg6[i];
   lcd_write();
   i += 1;
   if (i == 16)
   {
      temp1 = 0xc0; // Enable  the second line and to write commands in command register
      flag1 = 0;
      lcd_write();
      flag1 = 1;
   }
}
delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();

flag1 = 1; // Write the data in data register
i = 0;

while (msg7[i] != '\0')
{
   temp1 = msg7[i];
   lcd_write();
   i += 1;
   if (i == 16)
   {
      temp1 = 0xc0; // Enable  the second line and to write commands in command register
      flag1 = 0;
      lcd_write();
      flag1 = 1;
   }
}
delay_lcd(1500000);
flag1 = 0;
```

```
    temp1 = 0x01;
    lcd_write();
    //_______________________________________________________________________
_________________________________________________________________________
______
    j = 0;
    flag1 = 1;

    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                correct_pwd[0] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    //_______________________________________________________________________
____________________________________________________
    j = 0;
    flag1 = 1;
```

```
   help = 0;

   while (1)
   {

      for (row = 0; row < 4; row++)
      {
         LPC_GPIO2->FIOPIN = 1 << (10 + row);
         x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
         for (i = 0; i < 500; i++)
            ;
         if (x)
         {
            if (x == 1)
               col = 0;
            else if (x == 2)
               col = 1;
            else if (x == 4)
               col = 2;
            else if (x == 8)
               col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
               ;
            help = 1;
            correct_pwd[1] = message[row][col];
         }
      }
      if (help == 1)
      {
         break;
      }
   }

   delay_lcd(1500000);
   flag1 = 0;
   temp1 = 0x01;
   lcd_write();
   j = 0;
   flag1 = 1;

  help = 0;
   while (1)
   {

      for (row = 0; row < 4; row++)
      {
         LPC_GPIO2->FIOPIN = 1 << (10 + row);
         x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
         for (i = 0; i < 500; i++)
```

```
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            correct_pwd[2] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
j = 0;
flag1 = 1;
help = 0;
while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            correct_pwd[3] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
delay_lcd(1500000);
flag1 = 1; // Write the data in data register
i = 0;
while (msg[i] != '\0')
{
    temp1 = msg[i];
    lcd_write();
    i += 1;
    if (i == 16)
    {
        temp1 = 0xc0; // Enable  the second line and to write commands in command register
        flag1 = 0;
        lcd_write();
        flag1 = 1;
    }
}
delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
flag1 = 1;
j = 0;
flag1 = 1;

while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
```

```
  {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[0] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write;
    j = 0;
    flag1 = 1;

    help = 0;

    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            entered_pwd[1] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
j = 0;
flag1 = 1;

//________________________________________________________________________________
________________________________________________________

help = 0;
while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            entered_pwd[2] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
j = 0;
flag1 = 1;

help = 0;
while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            entered_pwd[3] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
```

```
temp1 = 0x01;
lcd_write();


help = 0;

flag1 = 1; // Write the data in data register
i = 0;

while (msg2[i] != '\0')
{
   temp1 = msg2[i];
   lcd_write();
   i += 1;
   if (i == 16)
   {
      temp1 = 0xc0; // Enable  the second line and to write commands in command register
      flag1 = 0;
      lcd_write();
      flag1 = 1;
   }
}
delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();

//______________________________________________________________________________________________
_________________________________________________________________________

// checking the password

for (i = 0; i < 4; i++)
{
   if (entered_pwd[i] != correct_pwd[i])
   {
      fail = 1;
      break;
   }
}

if (fail == 0)
{
   flag1 = 1; // Write the data in data register
   i = 0;
   while (msg3[i] != '\0')
   {
      temp1 = msg3[i];
      lcd_write();
      i += 1;
      if (i == 16)
```

```
        {
            temp1 = 0xc0; // Enable  the second line and to write commands in command register
            flag1 = 0;
            lcd_write();
            flag1 = 1;
        }
    }
    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();
    supcount = 3;
}
else
{
    flag1 = 1; // Write the data in data register
    i = 0;

    while (msg4[i] != '\0')
    {
        temp1 = msg4[i];
        lcd_write();
        i += 1;
        if (i == 16)
        {
            temp1 = 0xc0; // Enable  the second line and to write commands in command register
            flag1 = 0;
            lcd_write();
            flag1 = 1;
        }
    }
    delay_lcd(1500000);
    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    flag1 = 1;
    i = 0;
    while (msg[i] != '\0')
    {
        temp1 = msg[i];
        lcd_write();
        i += 1;
        if (i == 16)
        {
            temp1 = 0xc0; // Enable  the second line and to write commands in command register
            flag1 = 0;
            lcd_write();
            flag1 = 1;
        }
```

```
}
delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();

j = 0;
flag1 = 1;

while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            entered_pwd[0] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();

//______________________________________________________________________
_____________________________________________________________
j = 0;
flag1 = 1;
```

```
help = 0;

while (1)
{

    for (row = 0; row < 4; row++)
    {
        LPC_GPIO2->FIOPIN = 1 << (10 + row);
        x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
        for (i = 0; i < 500; i++)
            ;
        if (x)
        {
            if (x == 1)
                col = 0;
            else if (x == 2)
                col = 1;
            else if (x == 4)
                col = 2;
            else if (x == 8)
                col = 3;
            temp1 = message[row][col];
            lcd_write1();
            for (i = 0; i < 1500000; i++)
                ;
            help = 1;
            entered_pwd[1] = message[row][col];
        }
    }
    if (help == 1)
    {
        break;
    }
}

delay_lcd(1500000);
flag1 = 0;
temp1 = 0x01;
lcd_write();
j = 0;
flag1 = 1;

//________________________________________________________________________________
________________________________________________________________________

help = 0;
while (1)
{

    for (row = 0; row < 4; row++)
```

```
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[2] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();
    j = 0;
    flag1 = 1;

    //____________________________________________________________________
____________________________________________________________
    help = 0;
    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[3] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    //____________________________________________________________________________
____________________________________________________________________________
    help = 0;

    flag1 = 1; // Write the data in data register
    i = 0;

    while (msg2[i] != '\0')
    {
        temp1 = msg2[i];
        lcd_write();
        i += 1;
        if (i == 16)
        {
            temp1 = 0xc0; // Enable  the second line and to write commands in command register
            flag1 = 0;
            lcd_write();
            flag1 = 1;
        }
    }
    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();
    fail = 0;
```

```
        for (i = 0; i < 4; i++)
        {
            if (entered_pwd[i] != correct_pwd[i])
            {
                fail = 1;
                break;
            }
        }

        if (fail == 0)
        {
            flag1 = 1; // Write the data in data register
            i = 0;
            while (msg3[i] != '\0')
            {
                temp1 = msg3[i];
                lcd_write();
                i += 1;
                if (i == 16)
                {
                    temp1 = 0xc0; // Enable  the second line and to write commands in command
register
                    flag1 = 0;
                    lcd_write();
                    flag1 = 1;
                }
            }
            delay_lcd(1500000);
            flag1 = 0;
            temp1 = 0x01;
            lcd_write();
            supcount = 3;
        }
        else
        {
            flag1 = 1; // Write the data in data register
            i = 0;

            while (msg4[i] != '\0')
            {
                temp1 = msg4[i];
                lcd_write();
                i += 1;
                if (i == 16)
                {
                    temp1 = 0xc0; // Enable  the second line and to write commands in command
register
                    flag1 = 0;
                    lcd_write();
                    flag1 = 1;
                }
```

```
    }
    delay_lcd(1500000);
    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    flag1 = 1;
    i = 0;
    while (msg[i] != '\0')
    {
        temp1 = msg[i];
        lcd_write();
        i += 1;
        if (i == 16)
        {
            temp1 = 0xc0; // Enable  the second line and to write commands in command
register
            flag1 = 0;
            lcd_write();
            flag1 = 1;
        }
    }
    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    j = 0;
    flag1 = 1;

    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[0] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();

    //____________________________________________________________________
____________________________________________________________________
    j = 0;
    flag1 = 1;

    help = 0;

    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[1] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }

    delay_lcd(1500000);
    flag1 = 0;
    temp1 = 0x01;
    lcd_write();
    j = 0;
    flag1 = 1;

    //____________________________________________________________________________
____________________________________________________________

    help = 0;
    while (1)
    {

        for (row = 0; row < 4; row++)
        {
            LPC_GPIO2->FIOPIN = 1 << (10 + row);
            x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
            for (i = 0; i < 500; i++)
                ;
            if (x)
            {
                if (x == 1)
                    col = 0;
                else if (x == 2)
                    col = 1;
                else if (x == 4)
                    col = 2;
                else if (x == 8)
                    col = 3;
                temp1 = message[row][col];
                lcd_write1();
                for (i = 0; i < 1500000; i++)
                    ;
                help = 1;
                entered_pwd[2] = message[row][col];
            }
        }
        if (help == 1)
        {
            break;
        }
    }
```

```
        delay_lcd(1500000);
        flag1 = 0;
        temp1 = 0x01;
        lcd_write();
        j = 0;
        flag1 = 1;

        //______________________________________________________________________________
______________________________________________________________________
        help = 0;
        while (1)
        {

            for (row = 0; row < 4; row++)
            {
                LPC_GPIO2->FIOPIN = 1 << (10 + row);
                x = (LPC_GPIO1->FIOPIN >> 23) & 0xF;
                for (i = 0; i < 500; i++)
                    ;
                if (x)
                {
                    if (x == 1)
                        col = 0;
                    else if (x == 2)
                        col = 1;
                    else if (x == 4)
                        col = 2;
                    else if (x == 8)
                        col = 3;
                    temp1 = message[row][col];
                    lcd_write1();
                    for (i = 0; i < 1500000; i++)
                        ;
                    help = 1;
                    entered_pwd[3] = message[row][col];
                }
            }
            if (help == 1)
            {
                break;
            }
        }

        delay_lcd(1500000);
        flag1 = 0;
        temp1 = 0x01;
        lcd_write();

        //______________________________________________________________________________
______________________________________________________________________________
        help = 0;
        flag1 = 1; // Write the data in data register
        i = 0;

        while (msg2[i] != '\0')
        {
           temp1 = msg2[i];
           lcd_write();
           i += 1;
           if (i == 16)
           {
              temp1 = 0xc0; // Enable  the second line and to write commands in command
register
              flag1 = 0;
              lcd_write();
              flag1 = 1;
           }
        }
        delay_lcd(1500000);
        flag1 = 0;
        temp1 = 0x01;
        lcd_write();
        fail = 0;
        for (i = 0; i < 4; i++)
        {
           if (entered_pwd[i] != correct_pwd[i])
           {
              fail = 1;
              break;
           }
        }

        if (fail == 0)
        {
           flag1 = 1; // Write the data in data register
           i = 0;
           while (msg3[i] != '\0')
           {
              temp1 = msg3[i];
              lcd_write();
              i += 1;
              if (i == 16)
              {
                 temp1 = 0xc0; // Enable  the second line and to write commands in command
register
                 flag1 = 0;
                 lcd_write();
                 flag1 = 1;
              }
           }
           delay_lcd(1500000);
```

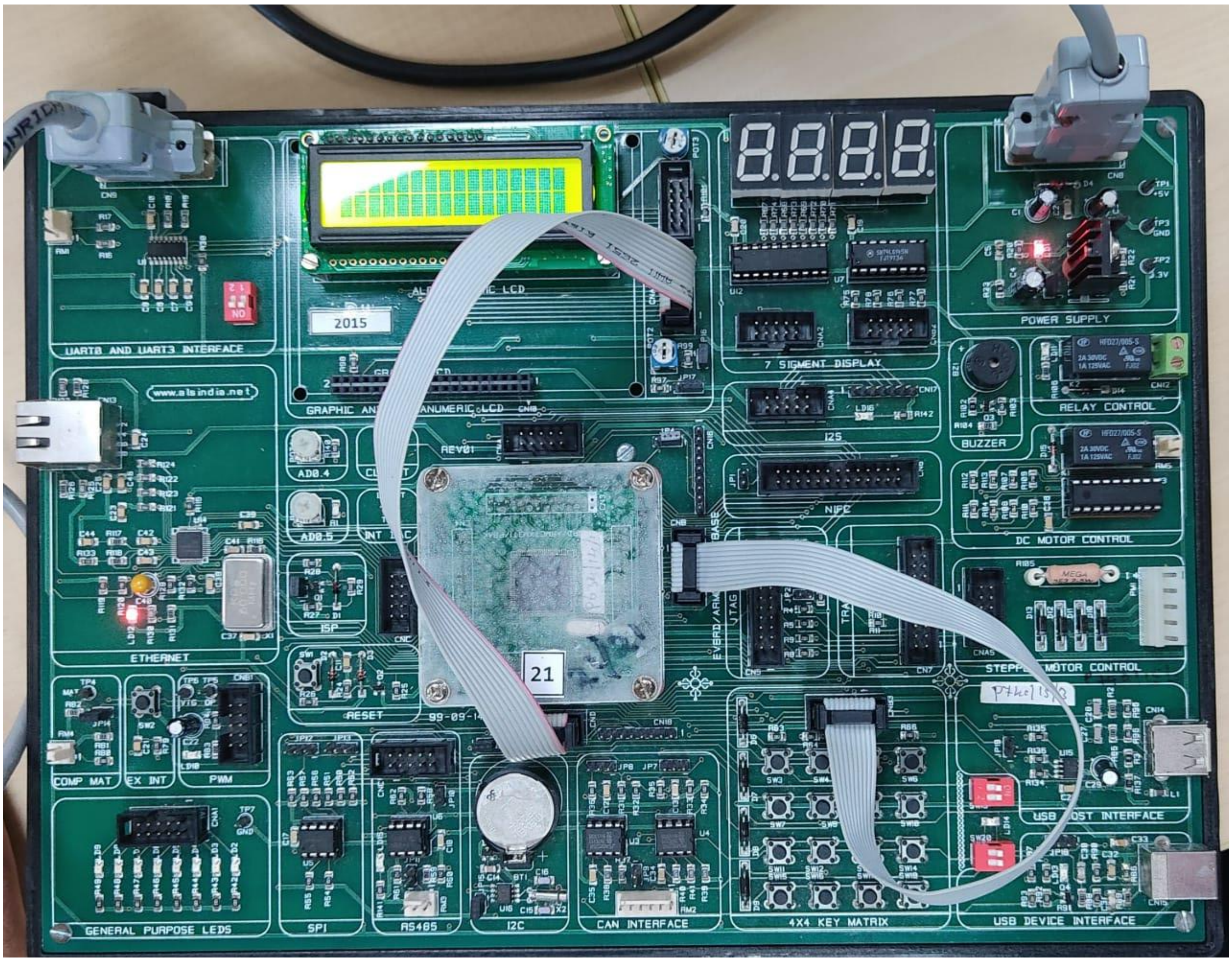
```
            flag1 = 0;
            temp1 = 0x01;
            lcd_write();
            supcount = 3;
         }
         else
         {
            flag1 = 1; // Write the data in data register
            i = 0;
            while (msg5[i] != '\0')
            {
               temp1 = msg5[i];
               lcd_write();
               i += 1;
               if (i == 16)
               {
                  temp1 = 0xc0; // Enable  the second line and to write commands in command
register
                  flag1 = 0;
                  lcd_write();
                  flag1 = 1;
               }
            }
            delay_lcd(1500000);
            flag1 = 0;
            temp1 = 0x01;
            lcd_write();
            supcount = 3;
         }
      }
   }
}

void lcd_write(void)
{
   flag2 = (flag1 == 1) ? 0 : ((temp1 == 0x30) || (temp1 == 0x20)) ? 1
                                                          : 0;
   temp2 = temp1 & 0xf0; // move data (26-8+1) times : 26 - HN place, 4 - Bits
   temp2 = temp2 >> 4;
   temp2 = temp2 << DT; // data lines from 23 to 26
   port_write();
   if (!flag2)
   {
      temp2 = temp1 & 0x0f; // 26-4+1
      temp2 = temp2 << DT;
      port_write();
   }
}

void port_write(void)
{
   LPC_GPIO0->FIOPIN = 0;
   LPC_GPIO0->FIOPIN = temp2;
   if (flag1 == 0)
      LPC_GPIO0->FIOCLR = 1 << RS;
   else
      LPC_GPIO0->FIOSET = 1 << RS;

   LPC_GPIO0->FIOSET = 1 << EN;
   delay_lcd(25);
   LPC_GPIO0->FIOCLR = 1 << EN;
   delay_lcd(30000);
}
```

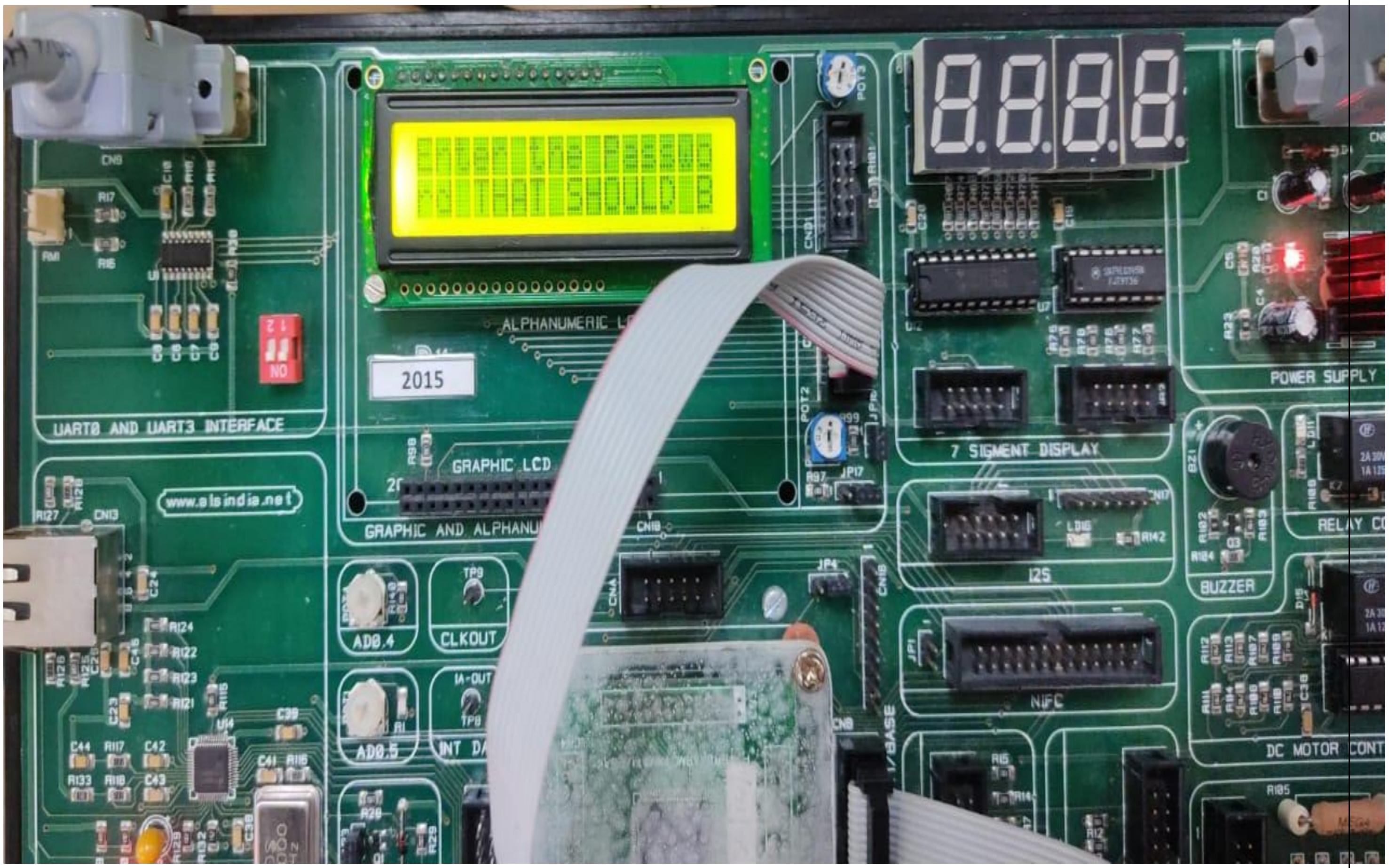
CONNECTION



UART0 AND UART3 INTERFACE
ETHERNET
COMP MAT
EX INT
PWM
GENERAL PURPOSE LEDS
SPI
RS485
I2C
CAN INTERFACE
4X4 KEY MATRIX
USB DEVICE INTERFACE
USB HOST INTERFACE
STEPPER MOTOR CONTROL
DC MOTOR CONTROL
RELAY CONTROL
BUZZER
POWER SUPPLY
7 SEGMENT DISPLAY
I2S
NIFC
GRAPHIC AND ALPHANUMERIC LCD
ISP
RESET
AD0.4
AD0.5
INT ISAC
2015
21
www.alsindia.net

1.Enter Password

2.Password Signup



3.Password Strength Check
Check for alphabets count if
Count==4|count==3
Print strong password

Else if count==2
Print average password
Else
Print weak password
4.Password Hashing
Hashing is also used with a secret key
5.Password Access Control using Timer
Access control btw tries is controlled using timer
6..Password Login
Password authentication in 3 tries

CONCLUSION

Password and Authenticator Manager using the LPC1768 microcontroller has culminated in a robust system that addresses the critical aspects of password management and user authentication. This project leveraged a multifaceted approach, integrating innovative design elements with advanced security measures to create a solution that balances functionality, usability, and security.

The project's key strength lies in the implementation of a tiled interface, offering a visually intuitive and user-friendly platform for managing passwords securely. The use of the LPC1768 microcontroller, in conjunction with FRC and jumper cables, facilitated seamless integration and efficient communication among system components, ensuring a cohesive and reliable system architecture.

Moreover, the emphasis on security was a cornerstone of this project. By incorporating advanced encryption algorithms and considering various authentication methods, such as biometrics or two-factor authentication, the system ensures a robust defense against unauthorized access and data breaches. This security-first approach aligns with contemporary standards, providing users with a trustworthy platform to safeguard their sensitive information.

The project's iterative and modular design approach allows for scalability and adaptability, enabling future enhancements and accommodating evolving user needs and security protocols. It stands as a testament to the importance of a user-centric design philosophy, where the emphasis on usability and practicality enhances the overall user experience.

In conclusion, the Tiled Password and Authenticator Manager project represents a successful convergence of technology and user-centric design principles. It showcases the LPC1768 microcontroller's capabilities while addressing the critical need for secure password management and authentication. This project serves as a foundation for further innovation in embedded systems, emphasizing the significance of user interaction, security, and adaptability in modern technological solutions.