```
// PARCIAL 3 - DISEÑO CON uP y uC. 2024-1.
// NOMBRE: Diego Andrés García Díaz.
// CÓDIGO: 2195533.
// --------------------------------------------------------------------------
//    RA2 Pin 5 === SDA; RA1 pin 06 === SCL. PIC12F1822
// ***
#include "stdint.h"
#define i2c_soft
#ifndef  SOFT_I2C
#define LCD_I2C_Start  I2C1_Start
#define LCD_I2C_Write  I2C1_Wr
#define LCD_I2C_Stop   I2C1_Stop
#else
#define LCD_I2C_Start  Soft_I2C_Start
#define LCD_I2C_Write  Soft_I2C_Write
#define LCD_I2C_Stop   Soft_I2C_Stop
#endif

#define LCD_BACKLIGHT         0x08
#define LCD_NOBACKLIGHT       0x00
#define LCD_FIRST_ROW         0x80
#define LCD_SECOND_ROW        0xC0
#define LCD_THIRD_ROW         0x94
#define LCD_FOURTH_ROW        0xD4
#define LCD_CLEAR             0x01
#define LCD_RETURN_HOME       0x02
#define LCD_ENTRY_MODE_SET    0x04
#define LCD_CURSOR_OFF        0x0C
#define LCD_UNDERLINE_ON      0x0E
#define LCD_BLINK_CURSOR_ON   0x0F
#define LCD_MOVE_CURSOR_LEFT  0x10
#define LCD_MOVE_CURSOR_RIGHT  0x14
#define LCD_TURN_ON           0x0C
#define LCD_TURN_OFF          0x08
#define LCD_SHIFT_LEFT        0x18
#define LCD_SHIFT_RIGHT       0x1E

#ifndef LCD_TYPE
#define LCD_TYPE 4          // 0=5x7, 1=5x10, 2=2 lines
#endif

unsigned int m; //----------------------------------------------------------------------
void Expander_Write(uint8_t value);
void LCD_Write_Nibble(uint8_t n);
void LCD_Cmd_(uint8_t Command);
void LCD_Goto(uint8_t col, uint8_t row);
void LCD_PutC(char LCD_Char);
void LCD_Print(char* LCD_Str);
void LCD_Begin(uint8_t _i2c_addr);
void Backlight();
void noBacklight();

bit RS;
unsigned short i2c_addr, backlight_val = LCD_BACKLIGHT;

void Expander_Write(uint8_t value)
                 {
                  LCD_I2C_Start();
                  LCD_I2C_Write(i2c_addr);
                  LCD_I2C_Write(value | backlight_val);
                  LCD_I2C_Stop();
                 }

void LCD_Write_Nibble(uint8_t n)   //ok
                {
                 n |= RS;
                 Expander_Write(n & 0xFB);
                 //delay_us(1);
                 asm nop;  //0,6 uS
                 Expander_Write(n | 0x04);
                 //delay_us(1);
                 asm nop;  //0,6 uS
                 Expander_Write(n & 0xFB);
                 //delay_us(100);
                 for (m = 0; m < 20; m = m +1){}//asm: nop   1 = 5 us
                 }

void LCD_Cmd_(uint8_t Command)   //ok
               {
                RS = 0;
                LCD_Write_Nibble(Command & 0xF0);
                LCD_Write_Nibble((Command << 4) & 0xF0);
                if((Command == LCD_CLEAR) || (Command == LCD_RETURN_HOME))
                 //delay_ms(2);
                 for (m = 0; m < 400; m = m +1){}//asm: nop   1 = 5 us
               }

void LCD_Goto(uint8_t col, uint8_t row)
                    {
                     switch(row)
                     {
                      case 2:
                       LCD_Cmd_(LCD_SECOND_ROW + col - 1);
                       break;
                      case 3:
                       LCD_Cmd_(LCD_THIRD_ROW  + col - 1);
                       break;
                      case 4:
                       LCD_Cmd_(LCD_FOURTH_ROW + col - 1);
                      break;
                      default:     // case 1:
                       LCD_Cmd_(LCD_FIRST_ROW  + col - 1);
                     }

                    }

void LCD_PutC(char LCD_Char)
              {
               RS = 1;
               LCD_Write_Nibble(LCD_Char & 0xF0);
               LCD_Write_Nibble((LCD_Char << 4) & 0xF0);
              }

void LCD_Print(char* LCD_Str)
               {
                uint8_t i = 0;
                RS = 1;
                while(LCD_Str[i] != '\0')
                {
                 LCD_Write_Nibble(LCD_Str[i] & 0xF0);
                 LCD_Write_Nibble( (LCD_Str[i++] << 4) & 0xF0 );
                }
               }

void LCD_Begin(uint8_t _i2c_addr)            // ok
                {
                 i2c_addr = _i2c_addr;
                 Expander_Write(0);

                 //delay_ms(40);
                 for (m = 0; m < 8000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(3);
                 //delay_ms(5);
                 for (m = 0; m < 1000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(3);
                 //delay_ms(5);
                 for (m = 0; m < 1000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(3);
                 //delay_ms(5);
                 for (m = 0; m < 1000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(LCD_RETURN_HOME);
                 //delay_ms(5);
                 for (m = 0; m < 1000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(0x20 | (LCD_TYPE << 2));
                 //delay_ms(50);
                 for (m = 0; m < 10000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(LCD_TURN_ON);
                 //delay_ms(50);
                 for (m = 0; m < 10000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(LCD_CLEAR);
                 //delay_ms(50);
                 for (m = 0; m < 10000; m = m +1){}//asm: nop   1 = 5 us
                 LCD_Cmd_(LCD_ENTRY_MODE_SET | LCD_RETURN_HOME);
                 //delay_ms(50);
                 for (m = 0; m < 10000; m = m +1){}//asm: nop   1 = 5 us
                }

void Backlight() {                  //ok
          backlight_val = LCD_BACKLIGHT;
          Expander_Write(0);
        }

void noBacklight() {                //ok
           backlight_val = LCD_NOBACKLIGHT;
           Expander_Write(0);
          }
// ***
// ---------------------------------------------------------------------------------

char keypadPort at PORTA; // Puerto del micro para conectar el teclado.
#define LED_RED     PORTA.F4    // Se define el pin RA4 del micro para controlar el LED.
#define LED_GREEN     PORTA.F5  // Se define el pin RA5 del micro para controlar el LED.
char tecla;
char texto[10];
char retardo_cad[10];
unsigned int retardo=100;
//maquina de estado para capturar varias teclas
enum teclado_estado {n1,n2,n3,n4,en};
char estado_teclado=n1;
char tec=0;
char teclado_deco(){
  char tec;
  tec = Keypad_Key_Press();     //mejor
   if (tec!=0){
    switch (tec){
     case 1:tec='7';break;
     case 2:tec='8';break;
     case 3:tec='9';break;
     case 4:tec='/';break;
     case 5:tec='4';break;
     case 6:tec='5';break;
     case 7:tec='6';break;
     case 8:tec='*';break;
     case 9:tec='1';break;
     case 10:tec='2';break;
     case 11:tec='3';break;
     case 12:tec='-';break;
     case 13:tec='E';break;
     case 14:tec='0';break;
     case 15:tec='=';break;
     case 16:tec='A';break;
     }//Fin del switch
    return (tec);
    }//si hay una tecla valida
}

void teclado_estado(){
    tecla=teclado_deco();
    switch (estado_teclado){
      case n1:
              if ( (tecla!=0)  && (isdigit(tecla))){
              LED_GREEN=1;
              retardo_cad[0]=tecla;
              estado_teclado=n2;
              //Lcd_Chr(2,8,tecla);
              LCD_Goto(2,8);
              LCD_PutC(tecla);
              Delay_ms(400); // retardo antirrebote
              }
              break;
      case n2:
              if ((tecla!=0)  && (isdigit(tecla))){
              retardo_cad[1]=tecla;
              estado_teclado=n3;
              //Lcd_Chr(2,9,tecla);
              LCD_Goto(2,9);
              LCD_PutC(tecla);
              Delay_ms(400); // retardo antirrebote
              }
              break;
    case n3:
        if ((tecla!=0)  && (isdigit(tecla))){
        retardo_cad[2]=tecla;
        estado_teclado=n4;
        //Lcd_Chr(2,10,tecla);
        LCD_Goto(2,10);
        LCD_PutC(tecla);
        Delay_ms(400); // retardo antirrebote
        }
        break;
    case n4:
        if ((tecla!=0)  && (isdigit(tecla))){
        retardo_cad[3]=tecla;
        estado_teclado=en;
        //Lcd_Chr(2,11,tecla);
        LCD_Goto(2,11);
        LCD_PutC(tecla);
        Delay_ms(400); // retardo antirrebote
        }
        break;
     case en:
         if ((tecla!=0)  && (tecla!='=')){  // no se presiono el igual
         LED_GREEN=0;
         LED_RED=1;
         estado_teclado=n1; // vuelve al estado de  captura del primer numero
         //Lcd_Out(2,8,"      ");// borra la pantalla
         LCD_Goto(2,8);
         LCD_Print("      ");
         Delay_ms(600); // retardo antirrebote
         }
         // si si se presiono el enter
         else if ((tecla!=0)  && (tecla=='=')){  //si se presiono el igual
         LED_GREEN=0;
         retardo_cad[4]=0;  //final de cadena
         retardo=atoi(retardo_cad); // lo pasa a numero y calcula el retardo deseado
         estado_teclado=n1; // vuelve al estado de caprura del primer numero.
         //Lcd_Out(2,8,"      ");// borra la pantalla
         LCD_Goto(2,8);
         LCD_Print("      ");
         Delay_ms(600); // retardo antirrebote
         }
         break;
    }
}

void miretardo(signed int valoor){
   while (valoor>0){
       Delay_us(800);
       teclado_estado();
       valoor--;
       }
}

char usuario[5]=" ";
char fija[5]="791A";
int i=0, error=0;
void main(){
    //TRISA=0;
    ANSELA=0;
    OSCCON = 0b11110011;
    TRISA4_bit=0; //salida led rojo
    LED_RED=1;
    TRISA5_bit=0; //salida led verde
    LED_GREEN=0;
    //LATA.F1=0;
    //LCD_INIT();                  // Initialize LCD
    LCD_Cmd_(LCD_CLEAR);           // Clear display
    LCD_Cmd_(LCD_CURSOR_OFF);              // Cursor off
    Keypad_Init();
    //Lcd_Out(1,6,"PASSWORD:");
    LCD_Goto(1,6);
    LCD_Print("PASSWORD: ");
    // Lcd_out(2,1,"CONTRA: ");
    LCD_Goto(2,1);
    LCD_Print("PASS: ");

    while (1){
        LED_RED=~LED_RED;//invierte el led
        miretardo(retardo);

                            usuario[i]= teclado();
        //Lcd_chr_cp('*');
                            LCD_PutC('*');
        i++;
        if(i==4){
            LCD_Cmd_(1);
            //Lcd_out(1,1,"VALIDANDO");
                                        LCD_Goto(1,1);
                                        LCD_Print("VALIDANDO");
            for(i=0; i<=3; i++){
                if(fija[i] != usuario[i]){
                  error++;
                }
                //Lcd_chr(2,i+1,'*');
                                  LCD_Goto(2,i+1);
                                  LCD_PutC('*');
                delay_ms(200);
            }
            if(error==0){
                verde=1;
                rojo = 0;
                LCD_Cmd_(1);
                //Lcd_out(1,1,"CONTRASENA VALIDA");
                                  LCD_Goto(1,1);
                                  LCD_Print("CONTRASENA VALIDA");
            }
            else{
                verde=0;
                rojo = 1;
                LCD_Cmd_(1);
                //Lcd_out(1,1,"ERROR CONTRASENA");
                                  LCD_Goto(1,1);
                                  LCD_Print("ERRPR CONTRASENA");
            }
            delay_ms(2000);
             LCD_Cmd_(1);
             //Lcd_out(1,1,"DIGITE CONTRASENA");
                           LCD_Goto(1,1);
                           LCD_Print("DIGITE CONTRASENA";
             //Lcd_out(2,1,"CONTRA: ");
                           LCD_Goto(2,1);
                           LCD_Print("PASS:");
             i=0;
             error=0;
        }
       }
}
```