PASSWORD-PROTECTED DOOR ALARM SYSTEM

WHAT IS THE FUNCTION OF THIS SYSTEM?

The system is intended to prevent thieves from entering your house. After installing the system on entrance gate or entrance door of your house, anyone who intends to enter the house should enter the pre-defined password before opening the door. The passwords entered and the system directives can be followed in the small lcd monitor. If the door is opened without entering the correct password, the system will activate the alarm. This alarm will prevent unauthorized people from entering your house.

HOW CAN YOU USE THIS SYSTEM?

Basic procedure for using the installed system is:

1) Enter your initial password. (If you use for the first time)

2) Reenter your initial password for verification.

3) System will control whether the two passwords match or not. If they do not match, you will be redirected to step 1.

4) If two passwords match, this password is set as your valid password.

Now, system is ready to be used.

5) If you want to open the door, you need to enter your valid password.

6) If the password you provided is wrong, you will be redirected to step 5.

7) You may enter a wrong password at most 3 times. If you enter wrong password 4 times, this means you are not authorized. Alarm is activated. ☹

8) If the password you provided is correct, the system allows you to open the door.

9) You are asked whether you want to change your password or not.

10) If no, procedure is over. You need to do nothing other than entering your house safely. ☺

11) If yes, you are asked to enter your old password.

12) If the old password is entered wrong, you are redirected to step 9.

13) If old password is entered correctly, you will enter your new password.

14) Reenter your new password.

15) If the two passwords do not match, you are redirected to step 13.

16) If the two passwords match, new password is set as your valid password.
You are directed to step 5. ☺

Notes:

1) Maximum 12 digit password is allowed. If you want to exceed this limit, the first 12 digit will be accepted as the password. Also, a message will warn you about this.
2) If you want to enter a password with a digit number less than 12, you should press "#" in the keypad in order to terminate your password entrance.
3) When entering password, if you pressed a wrong number by mistake, you can cancel your password entrance by pressing "*" in the keypad.

HOW CAN YOU USE THE PROGRAM IN THIS DOCUMENT?

First, you need to have a C compiler. In this project, CCS C compiler is used. You can download the demo version from "www.*ccsinfo*.com". Save the following program in C source file format. Secondly, you should obtain the hardware components and construct the required circuit. Also, circuit diagram is available in *"www.eeprojects.co.cc"*. Then, you should load the compiled program (the file with '*.hex'* extension) to the circuit you have constructed. Finally, you can begin using the features of "*password protected door alarm system*".

PROGRAM TO BE LOADED TO YOUR CIRCUIT

```
#include<16f877A.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=4000000)
#opt 9
#define use_portb_lcd TRUE
#include<lcd.c>

void input_ver();      void scanning();          void password_gir();
void ilk_sifre_gir();  void yeni_sifre_gir();    void iceri_gir();
void sifreyi_yaz();    void alarm();             void kiyas_ilk();
void kiyas_yeni();     void password_degis();    void prm();
void tmp();            void temp2perm();         void password_sifirlama();
void temp_sifirlama(); void perm_sifirlama();    void lcd_input_ver();     void sifre_vazgec();
int i,j,k,p1,p2,cikis,y,d,length,length_temp,length_perm,change,yeni,vazgec,sayac,itut,ktut;
int password[12], temp[12], perm[12];
float x,v,volt;
float read_voltage(BYTE ch);
void init_hardware();

main(void)
{
  lcd_init();       delay_ms(1);              init_hardware();
  setup_ccp1(CCP_PWM);
  delay_ms(1000);
  ilk_sifre_gir();

  while(TRUE)
{
  iceri_gir();
  printf(lcd_putc,"\f");            lcd_gotoxy(1,1);          printf(lcd_putc,"enter house: 1");
  printf(lcd_putc,"\n");            lcd_gotoxy(1,2);          printf(lcd_putc,"change password: 2");
  lcd_input_ver();
  if(change==1)
      {   printf(lcd_putc,"\f");    lcd_gotoxy(1,1);
          printf(lcd_putc,"Welcome! You can\nenter.");    delay_ms(2000); }
  else  { password_degis(); }
}
  return 0;
}

void init_hardware()
{
  setup_adc(ADC_CLOCK_DIV_8);
  setup_adc_ports(ALL_ANALOG);
}
```

```
float read_voltage(BYTE ch)
{
  set_adc_channel(ch);
  delay_us(100);
  v = read_adc();
  return(v);
}

void scanning()
{
  for(i=0;i<4;i++)
  {  x=read_voltage(i);
    if(x>245) {  sayac++;   break; }}
  if(i==4)         { i=3;   }
  if((i<3)&&(sayac<2))  {j=j+1; }      else {j=j; }    k=k+1;
}

void input_ver()
{
  password_sifirlama();
  vazgec=0;
  itut=10;
  sayac=0;
  ktut=0;
  output_low(PIN_d2);    output_low(PIN_d3);    output_low(PIN_c4);    output_low(PIN_c5);
  while(TRUE)
{
  cikis=0;   k=0;
  output_high(PIN_d2);    delay_ms(1);   scanning();   password_gir();    delay_ms(1);
  output_low(PIN_d2);     delay_ms(1);
  if(cikis==1 || j==12 || vazgec==1) { break;}
  output_high(PIN_d3);    delay_ms(1);   scanning();   password_gir();    delay_ms(1);
  output_low(PIN_d3);     delay_ms(1);
  if(cikis==1 || j==12 || vazgec==1) { break;}
  output_high(PIN_c4);    delay_ms(1);   scanning();    password_gir();   delay_ms(1);
  output_low(PIN_c4);     delay_ms(1);
  if(cikis==1 || j==12 || vazgec==1) { break;}
  output_high(PIN_c5);    delay_ms(1);   scanning();    password_gir();   delay_ms(1);
  output_low(PIN_c5);     delay_ms(1);
  if(cikis==1 || j==12 || vazgec==1) { break;}
}}

void lcd_input_ver()
{
  j=0;
  change=0;
```

```
  while(TRUE)
{
  output_high(PIN_d3);
  for(i=0;i<4;i++)
        {       volt=read_voltage(i);           if(volt>245) { break; }    }
  if(i==0) { change=1; break;}
  if(i==2) { change=2; break;}
}
  output_low(PIN_d3);    delay_ms(1800);
}

void password_gir()
{
  p1=k*k+i;   p2=k*k+i-2;
  if((k==ktut) && (x<245))  { sayac=0;}
  if(i<3)
{
  if(k==1 || k==2)
        {   password[j-1]=p1;   ktut=k;  itut=i;  if(sayac<2) { printf(lcd_putc,"*");  }      }
  if(k==3)
        {   password[j-1]=p2;   ktut=k;  itut=i;   if(sayac<2) { printf(lcd_putc,"*");  }     }
  if(k==4 && i==0)
        {   delay_ms(250);        password_sifirlama();        vazgec=1;     }
  if(k==4 && i==1)
        { password[j-1]=0;    ktut=k;  itut=i;   if(sayac<2) { printf(lcd_putc,"*”);  }       }
  if(k==4 && i==2)    {   cikis=1;    }
}}

void sifreyi_yaz()
{
    if(j==12)
        {       delay_ms(400);   length=j; printf(lcd_putc,"\f");      lcd_gotoxy(1,1);
                printf(lcd_putc,"max. length\nreached");            delay_ms(1500);       }
    if(j<12)   {   length=j-1;  }
}

void ilk_sifre_gir()
{
   while(TRUE)
{
   j=0;      d=0;
   while(TRUE)
{
   printf(lcd_putc,"\f");          lcd_gotoxy(1,1);          printf(lcd_putc,"Enter initial pass:");
   printf(lcd_putc,"\nPW=");      input_ver();
   if(vazgec==1)          { sifre_vazgec();}
   else { break; }
}
```

```
   sifreyi_yaz(); prm();  password_sifirlama();    delay_ms(500);
   while(TRUE)
{
   printf(lcd_putc,"\f");              lcd_gotoxy(1,1);            printf(lcd_putc,"Reenter:\nPW=");
   j=0;               input_ver();
   if(vazgec==1) { sifre_vazgec();}  else { break; }
}
   sifreyi_yaz();  d=0;   kiyas_ilk();     delay_ms(100);
   if(d==j && length==length_perm)
      {   printf(lcd_putc,"\f");       lcd_gotoxy(1,1);
          printf(lcd_putc,"pw verified");     delay_ms(2000); break;  }
   else
      {   printf(lcd_putc,"\f");       lcd_gotoxy(1,1);
           printf(lcd_putc,"passwords do not\nmatch!");      delay_ms(2000);    perm_sifirlama();  }
}
   printf(lcd_putc,"\f");              lcd_gotoxy(1,1);
   printf(lcd_putc,"your password\nsaved.");      delay_ms(2000);
}

void yeni_sifre_gir()
{
   while(TRUE)
{
   delay_ms(500);   j=0;   d=0;
   printf(lcd_putc,"\f");              lcd_gotoxy(1,1);             printf(lcd_putc,"Enter your new\npassword:");
   input_ver();               sifreyi_yaz();                tmp();
   password_sifirlama();             delay_ms(1000);
   printf(lcd_putc,"\f");              lcd_gotoxy(1,1);            printf(lcd_putc,"Reenter:\nPW=");
   j=0;    input_ver();   sifreyi_yaz();  d=0;
   kiyas_yeni();     delay_ms(100);
   if(d==j && length==length_temp)
      {   printf(lcd_putc,"\f");      lcd_gotoxy(1,1);      printf(lcd_putc,"password saved");
          delay_ms(2000);      temp2perm();       temp_sifirlama();  break;  }
   else
      {   printf(lcd_putc,"\f");  lcd_gotoxy(1,1); printf(lcd_putc,"passwords do not\nmatch!");
          delay_ms(2000);   temp_sifirlama();  }
}
  printf(lcd_putc,"\f");      lcd_gotoxy(1,1);        printf(lcd_putc,"you can use use\nyour password.");
  delay_ms(2000);
}

void iceri_gir()
{
    y=0;      printf(lcd_putc,"\f");
    lcd_gotoxy(1,1); printf(lcd_putc,"Enter password: ");  delay_ms(100);
    while(TRUE)
{
    j=0;   password_sifirlama();
```

```
   lcd_gotoxy(1,2);        printf(lcd_putc,"PW= ");            input_ver();
   sifreyi_yaz();    d=0;       kiyas_ilk();
   if(d==j && length==length_perm)
        { printf(lcd_putc,"\f");       lcd_gotoxy(1,1);
          printf(lcd_putc,"password correct");       break;   delay_ms(2000);   }
   else
        { printf(lcd_putc,"\f");  lcd_gotoxy(1,1); printf(lcd_putc,"password wrong");  delay_ms(2000);
          y=y+1;  }

   printf(lcd_putc,"\f"); lcd_gotoxy(1,1); printf(lcd_putc,"%d trial remained",4-y);
   if(y==4)
        {    printf(lcd_putc,"\f"); lcd_gotoxy(1,1); printf(lcd_putc,"You are not\nauthorized");
             delay_ms(2000);
             printf(lcd_putc,"\f"); lcd_gotoxy(1,1); printf(lcd_putc,"Alarm will be\nactivated.");
             delay_ms(3000);
             y=0;              alarm();                    }
}
   printf(lcd_putc,"\f");                    lcd_gotoxy(1,1);
   printf(lcd_putc,"password correct");  delay_ms(2000);
   printf(lcd_putc,"\f");                    lcd_gotoxy(1,1);
   printf(lcd_putc,"You can\nenter");    delay_ms(2000);
}

void password_degis()
{
   yeni=0;
   while(TRUE)
{
   printf(lcd_putc,"\f");                    lcd_gotoxy(1,1);
   printf(lcd_putc,"Enter old pass:");     printf(lcd_putc,"\nPW=");
   j=0;             password_sifirlama();
   input_ver();   sifreyi_yaz();             d=0;                kiyas_ilk();
   if(d==j && length==length_perm)
        { yeni_sifre_gir();  yeni=1;   break;   }
   else
        {  change=0; printf(lcd_putc,"\f");           lcd_gotoxy(1,1);
           printf(lcd_putc,"Wrong!! Are you\nsure to change?");    delay_ms(1500);
           printf(lcd_putc,"\f");    lcd_gotoxy(1,1);          printf(lcd_putc,"change pw: 1");
           printf(lcd_putc,"\n");    lcd_gotoxy(1,2);          printf(lcd_putc,"disregard: 2");
          lcd_input_ver();           if(change!=1)             { break; }
          }
}
   if(yeni!=1)  { printf(lcd_putc,"\f");       lcd_gotoxy(1,1);
                  printf(lcd_putc,"valid pw is the\nold one.");  }
}

void sifre_vazgec()
{
```

```
printf(lcd_putc,"\f"); lcd_gotoxy(1,1);
printf(lcd_putc,"Password entrance\naborted.");          delay_ms(2000);          j=0;                  }

void kiyas_ilk()
{   for(i=0;i<j;i++)    { if(password[i]==perm[i]) {d=d+1;  } }
}

void kiyas_yeni()
{   for(i=0;i<j;i++)    { if(password[i]==temp[i]) {d=d+1;  } }
}

void prm()
{   for(i=0;i<12;i++)    { perm[i]=password[i];   }     length_perm=length;     }

void tmp()
{   for(i=0;i<12;i++)    { temp[i]=password[i]; }       length_temp=length;    }

void temp2perm()
{   for(i=0;i<12;i++)    { perm[i]=temp[i];    }       length_perm=length_temp; }

void password_sifirlama()
{   for(i=0;i<12;i++)  { password[i]=0; }    }

void perm_sifirlama()
{   for(i=0;i<12;i++)  { perm[i]=0; }       }

void temp_sifirlama()
{   for(i=0;i<12;i++)  { temp[i]=0; }       }

void alarm()
{   printf(lcd_putc,"\f");            lcd_gotoxy(1,1);
    printf(lcd_putc,"Alarm is active\nin 2 minutes.");                  delay_ms(2000);
}



// NOTE THAT THE CODE BELOW THIS LINE SHOULD BE PUT IN A SEPARATE SOURCE FILE BUT IN THE
//SAME FOLDER WITH THE CODE ABOVE.  THE TITLE OF THE SOURCE FILE SHOULD BE “lcd.c”


struct lcd_pin_map {              // This structure is overlayed
        BOOLEAN enable;           // on to an I/O port to gain
        BOOLEAN rs;              // access to the LCD pins.
        BOOLEAN rw;               // The bits are allocated from
        BOOLEAN unused;           // low order up.  ENABLE will
        int     data : 4;        // be pin B0.
      } lcd;
```

```
#if defined use_portb_lcd
  #locate lcd = getenv("sfr:PORTB")    // This puts the entire structure over the port
  #ifdef __pch__
   #locate lcd = 0xf81
  #else
   #locate lcd = 6
  #endif
  #define set_tris_lcd(x) set_tris_b(x)
#else
  #locate lcd = getenv("sfr:PORTD")    // This puts the entire structure over the port
  #ifdef __pch__
   #locate lcd = 0xf83
  #else
   #locate lcd = 8
  #endif
  #define set_tris_lcd(x) set_tris_d(x)
#endif

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

#define lcd_line_two 0x40    // LCD RAM address for the second line


BYTE const LCD_INIT_STRING[4] = {0x20 | (lcd_type << 2), 0xc, 1, 6};
                    // These bytes need to be sent to the LCD
                    // to start it up.


                    // The following are used for setting
                    // the I/O port direction register.

struct lcd_pin_map const LCD_WRITE = {0,0,0,0,0}; // For write mode all pins are out
struct lcd_pin_map const LCD_READ = {0,0,0,0,15}; // For read mode data pins are in


BYTE lcd_read_byte() {
    BYTE low,high;
    set_tris_lcd(LCD_READ);
    lcd.rw = 1;
    delay_cycles(1);
    lcd.enable = 1;
    delay_cycles(1);
    high = lcd.data;
    lcd.enable = 0;
    delay_cycles(1);
    lcd.enable = 1;
```

```
    delay_us(1);
    low = lcd.data;
    lcd.enable = 0;
    set_tris_lcd(LCD_WRITE);
    return( (high<<4) | low);
}

void lcd_send_nibble( BYTE n ) {
    lcd.data = n;
    delay_cycles(1);
    lcd.enable = 1;
    delay_us(2);
    lcd.enable = 0;
}

void lcd_send_byte( BYTE address, BYTE n ) {

    lcd.rs = 0;
    while ( bit_test(lcd_read_byte(),7) ) ;
    lcd.rs = address;
    delay_cycles(1);
    lcd.rw = 0;
    delay_cycles(1);
    lcd.enable = 0;
    lcd_send_nibble(n >> 4);
    lcd_send_nibble(n & 0xf);
}

void lcd_init() {
   BYTE i;
   set_tris_lcd(LCD_WRITE);
   lcd.rs = 0;
   lcd.rw = 0;
   lcd.enable = 0;
   delay_ms(15);
   for(i=1;i<=3;++i) {
     lcd_send_nibble(3);
     delay_ms(5);
   }
   lcd_send_nibble(2);
   for(i=0;i<=3;++i)
     lcd_send_byte(0,LCD_INIT_STRING[i]);
}

void lcd_gotoxy( BYTE x, BYTE y) {
  BYTE address;

  if(y!=1)
```

```
    address=lcd_line_two;
  else
    address=0;
  address+=x-1;
  lcd_send_byte(0,0x80|address);
}

void lcd_putc( char c) {
  switch (c) {
    case '\f'   : lcd_send_byte(0,1);
                  delay_ms(2);
                                  break;
    case '\n'   : lcd_gotoxy(1,2);       break;
    case '\b'   : lcd_send_byte(0,0x10);  break;
    default     : lcd_send_byte(1,c);     break;
  }
}

char lcd_getc( BYTE x, BYTE y) {
  char value;

   lcd_gotoxy(x,y);
   while ( bit_test(lcd_read_byte(),7) ); // wait until busy flag is low
   lcd.rs=1;
   value = lcd_read_byte();
   lcd.rs=0;
   return(value);
}
```