UNIVERSIDAD SIMÓN BOLÍVAR

DEPARTAMENTO DE ELECTRÓNICA Y CIRCUITOS

LABORATORIO DE MICROPROCESADORES EC-3074

INFORME - PRÁCTICA #4

Profesor

Mauricio Pérez

Estudiante

Giancarlo Torlone 20-10626

ÍNDICE

INTRODUCCIÓN

En la siguiente práctica de laboratorio se diseña una cerradura electrónica similar a la implementada en la práctica anterior (P3), pero esta vez utilizando un display LCD en lugar de un display de 7 segmentos para mostrar el código ingresado así como también algunos mensajes adicionales como el nombre ficticio de una empresa, “Ingresar Pin”, “Pin Correcto” y “Pin Incorrecto”.

Esta práctica fue realizada en C con el compilador XC8 de MPLAB.

ANÁLISIS DE RESULTADOS

Oscilador

El oscilador del microcontrolador se configuró para que sea HS (High Speed) y para que trabaje a una frecuencia de 20 MHz.

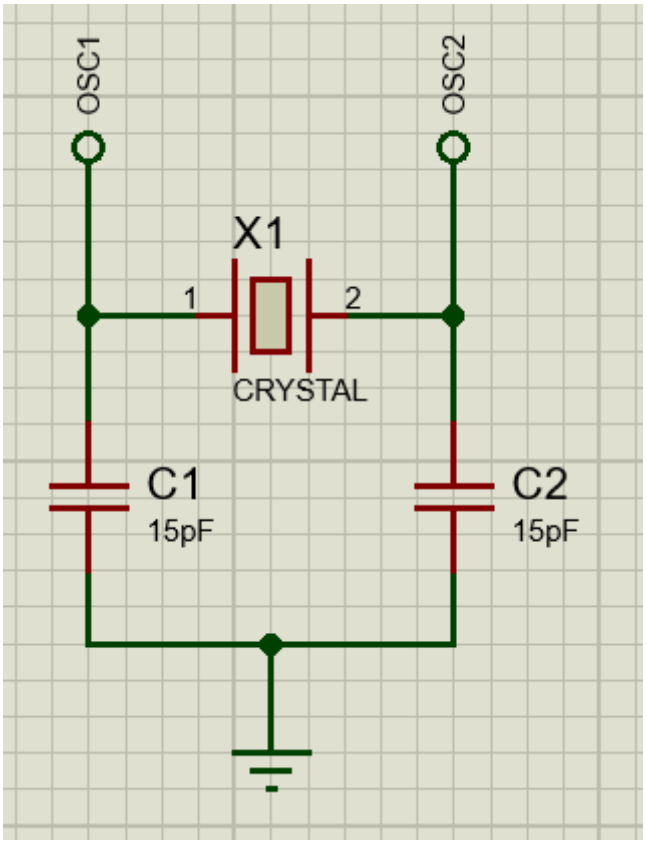

Figura 1. Conexiones para el oscilador

MCLR

Para el MCLR o reset se realizó la siguiente conexión en PULL UP.

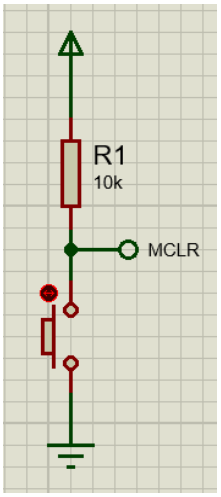

Figura 2. Conexión para el MCLR

Montaje Final

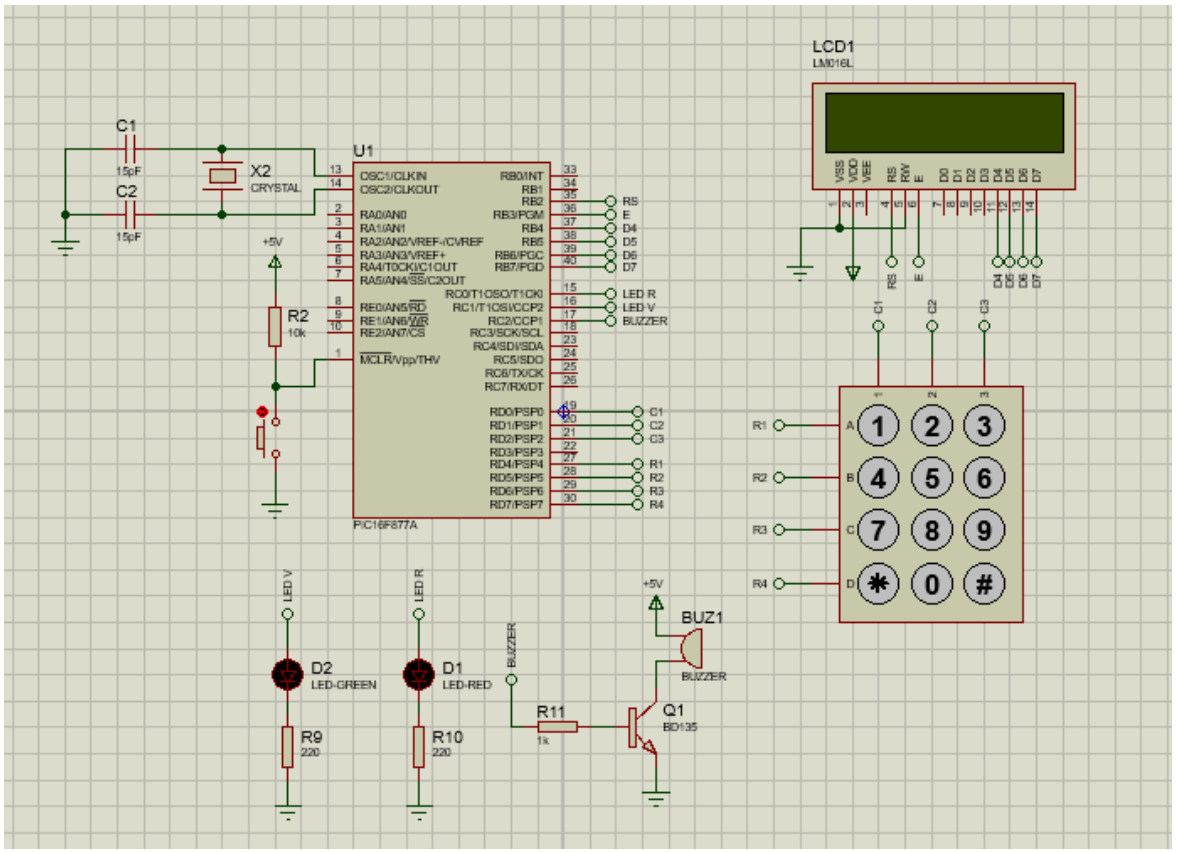

Figura 3. Montaje final del circuito

Código

Para facilitar el código, se hizo uso de librerías, las cuales permiten utilizar el teclado y manejar la pantalla LCD de manera fácil y rápida. Dichas librerías pueden ser encontradas por internet en diversos proyectos o foros. Además de las librerías mencionadas, se utilizaron dos librerías propias del lenguaje C para poder realizar ciertas funciones.

Las librerías fueron incluidas al inicio del código.

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

#include "lcd.h" // libreria para la pantalla LCD
#include "kbd4x3.h" // libreria para teclado matricial 4x4
```

Se comienza definiendo las configuraciones del microcontrolador

```
// PIC16F877A Configuration Bit Settings

// CONFIG
#pragma config FOSC = HS
#pragma config WDTE = OFF
#pragma config PWRTE = ON
#pragma config BOREN = OFF
#pragma config LVP = OFF
#pragma config CPD = OFF
#pragma config WRT = OFF
#pragma config CP = OFF

#define _XTAL_FREQ 20000000
```

Luego se definen los puertos para el Led Verde, el Led Rojo y el Buzzer o Zumbador.
También se definen las variables para almacenar la tecla presionada, la clave ingresada y la clave para acceder, la cual será **0626.**
Cabe destacar que dicha clave puede ser cambiada por otra reemplazándola en el código.

```
#define LED_ROJO PORTCbits.RC0 // led de acceso incorrecto
#define LED_VERDE PORTCbits.RC1 // led de acceso correcto
#define BUZZER PORTCbits.RC2 // buzzer

char tecla;                  // Almacena el valor de la tecla presionada
char clave[5];               // Almacena la clave ingresada por el usuario
char clave_enter[5] = "0626"; // Clave para acceder
```

A continuación se encuentra el programa principal, el cual se encarga de comparar la clave ingresada por el usuario con la correcta y en función de si es correcta o incorrecta, ejecutar una serie de instrucciones para cada caso.

```
while (1) {
    int i = 0;   // Contador para las veces que se pulsa alguna tecla
    Lcd_Set_Cursor(1,2);
    Lcd_Write_String("INGRESAR PIN"); // Mensaje de solicitud para ingresar clave

    while(i < 4)
    {
        tecla = Keypad_Get_Char(); // Lee el dato de la tecla presionada
        if(tecla != 0)  // Verifica si se ha presionado alguna tecla
        {
            clave[i] = tecla;   // Almacena cada tecla presionada en el arreglo
            Lcd_Set_Cursor(2,2+i);
            Lcd_Write_Char(tecla); // Muestra la tecla presionada. Puede ser reemplazado por * (para ocultarla)
            i++;  // Incrementa el contador
        }
    }
    __delay_ms(200);
    Lcd_Clear();   // Limpia la pantalla lcd

    if(!strcmp(clave, clave_enter)) // Compara si la clave es la correcta
    {
        LED_VERDE = 1; // enciende el led verde
        LED_ROJO = 0; // apaga el led rojo
        BUZZER = 1; //  enciende el buzzer
        Lcd_Set_Cursor(1,2);
        Lcd_Write_String("INSTRUMENT C.A");
        Lcd_Set_Cursor(2,2);
        Lcd_Write_String("PIN CORRECTO");
        __delay_ms(2000);
    }
    else   // Sino es la clave correcta, no permite el acceso
    {
        LED_ROJO = 1;
        LED_VERDE = 0;
        BUZZER = 1;
        Lcd_Set_Cursor(2,2);
        Lcd_Write_String("PIN INCORRECTO");
        __delay_ms(400);
        BUZZER = 0;
        __delay_ms(400);
        BUZZER = 1;
        __delay_ms(400);
        BUZZER = 0;
        __delay_ms(400);
        BUZZER = 1;
        __delay_ms(400);
        BUZZER = 0;
    }
    i = 0;          // Reinicia el contador
    LED_ROJO = 0;  // Apagar led rojo
    LED_VERDE = 0; // apagar led verde
    BUZZER = 0; // apaga buzzer
    Lcd_Clear();  // Limpia la pantalla lcd
}
```

Código completo

```
// PIC16F877A Configuration Bit Settings

// CONFIG
#pragma config FOSC = HS
#pragma config WDTE = OFF
#pragma config PWRTE = ON
#pragma config BOREN = OFF
#pragma config LVP = OFF
#pragma config CPD = OFF
#pragma config WRT = OFF
#pragma config CP = OFF

#define _XTAL_FREQ 20000000
#include <xc.h>
#include <stdio.h>
#include <string.h>

#include "lcd.h" // libreria para la pantalla LCD
#include "kbd4x3.h" // libreria para teclado matricial 4x4

#define LED_ROJO PORTCbits.RC0 // led de acceso incorrecto
#define LED_VERDE PORTCbits.RC1 // led de acceso correcto
#define BUZZER PORTCbits.RC2 // buzzer

char tecla;                    // Almacena el valor de la tecla presionada
char clave[5];                 // Almacena la clave ingresada por el usuario
char clave_enter[5] = "0626"; // Clave para acceder
```

```
void main(void) {
    ADCON1bits.PCFG = 0x0F; // coloca todos los pines como digitales
    TRISC = 0x00; // PUERTO C como salida (leds y buzzer)
    LED_VERDE = 0; // led verde inicialmente apagado
    LED_ROJO = 0; // led rojo inicalmente apagado
    BUZZER = 0; // buzzer incialmente apagado
    Keypad_Init();   // Inicializa el teclado matricial 4x4
    Lcd_Init();      // Inicializa la pantalla lcd

    while (1) {
        int i = 0;   // Contador para las veces que se pulsa alguna tecla
        Lcd_Set_Cursor(1,2);
        Lcd_Write_String("INGRESAR PIN"); // Mensaje de solicitud para ingresar clave

        while(i < 4)
        {
            tecla = Keypad_Get_Char(); // Lee el dato de la tecla presionada
            if(tecla != 0)  // Verifica si se ha presionado alguna tecla
            {
                clave[i] = tecla;   // Almacena cada tecla presionada en el arreglo
                Lcd_Set_Cursor(2,2+i);
                Lcd_Write_Char(tecla); // Muestra la tecla presionada. Puede ser reemplazado por * (para ocultarla)
                i++;  // Incrementa el contador
            }
        }
        __delay_ms(200);
        Lcd_Clear();   // Limpia la pantalla lcd
```

```
    if(!strcmp(clave, clave_enter)) // Compara si la clave es la correcta
    {
        LED_VERDE = 1; // enciende el led verde
        LED_ROJO = 0; // apaga el led rojo
        BUZZER = 1; //  enciende el buzzer
        Lcd_Set_Cursor(1,2);
        Lcd_Write_String("INSTRUMENT C.A");
        Lcd_Set_Cursor(2,2);
        Lcd_Write_String("PIN CORRECTO");
        __delay_ms(2000);
    }
    else   // Sino es la clave correcta, no permite el acceso
    {
        LED_ROJO = 1;
        LED_VERDE = 0;
        BUZZER = 1;
        Lcd_Set_Cursor(2,2);
        Lcd_Write_String("PIN INCORRECTO");
        __delay_ms(400);
        BUZZER = 0;
        __delay_ms(400);
        BUZZER = 1;
        __delay_ms(400);
        BUZZER = 0;
        __delay_ms(400);
        BUZZER = 1;
        __delay_ms(400);
        BUZZER = 0;
    }
```

```
       i = 0;          // Reinicia el contador
       LED_ROJO = 0;  // Apagar led rojo
       LED_VERDE = 0; // apagar led verde
       BUZZER = 0; // apaga buzzer
       Lcd_Clear();  // Limpia la pantalla lcd
   }
}
```