# UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

## FACULTAD DE INGENIERÍA

## LABORATORIO DE MICROCOMPUTADORAS

## GRUPO 04


**ROJAS MÉNDEZ GABRIEL**

**314141712**

**CARREÓN GUZMÁN MARIANA IVETTE**

**312103914**


# PRÁCTICA 8: PROGRAMACIÓN EN C

# PUERTOS PARALELOS E/S, PUERTO SERIE.

**SEMESTRE 2022-1**

**25/11/2021**

**Objetivos:**

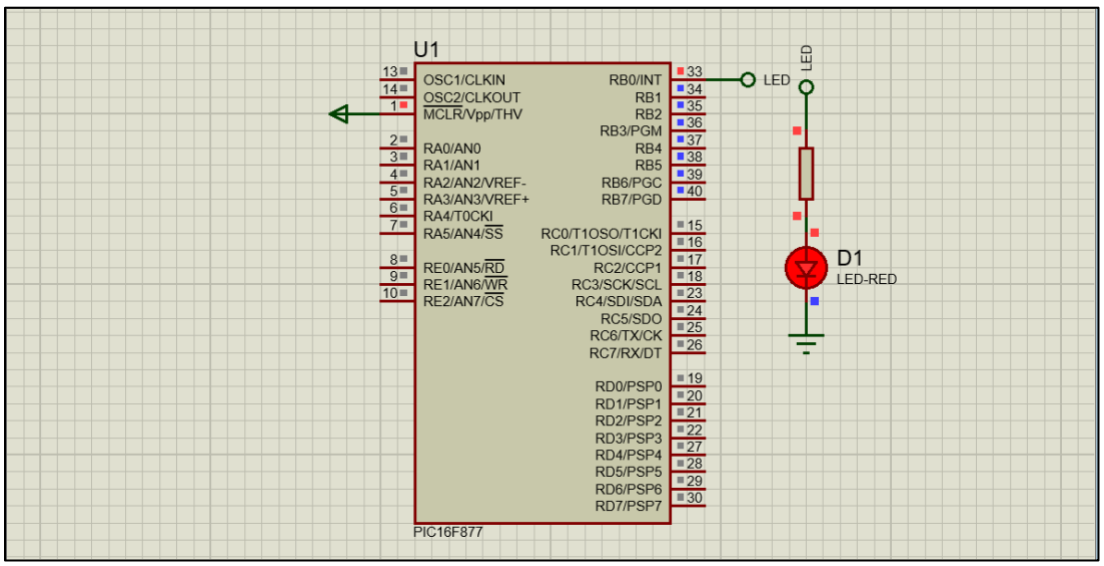
Realización de programas a través de programación en C y empleo del puerto serie para visualización y control.

**Desarrollo:**

1.- Escribir, comentar, compilar, el siguiente programa usando el ambiente del PIC C Compiler y comprobar el funcionamiento.

Ejercicio 1.c*

```
#include <16f877.h>     //Incluye la librería del microControlador
#fuses HS,NOPROTECT,
#use delay(clock=20000000)//Frec. de Oscilación 20Mhz
#org 0x1F00, 0x1FFF void loader16F877(void) {}

//Metodo que enciende y apaga  el bit 0 del puerto B
void main()
{
   while(1) //Ciclo while infinito
   {
      output_b(0x01);  //Enciende el bit 0 del puerto B
      delay_ms(1000);  //Delay de 1s
      output_b(0x00);  //Apaga bits del puerto B
      delay_ms(1000);  //Delay de 1s
   }
}
```

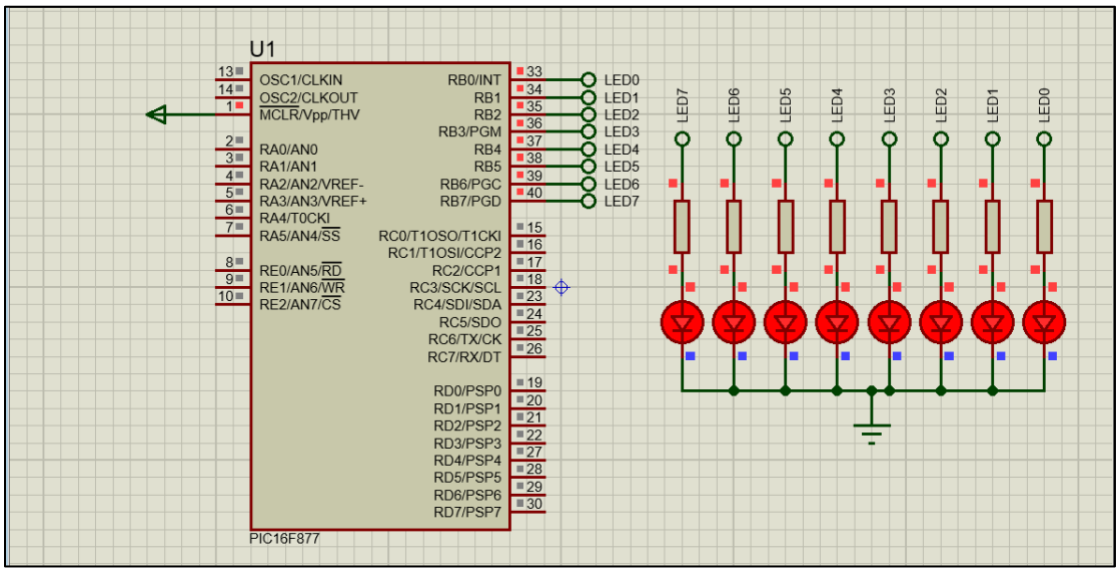



Como se observa, solo se transcribió el programa y se probó en Proteus.

2.- Modificar el programa para que active y desactive todos los bits del puerto B.

Ejercicio 1.c | Ejercicio 2.c

```
#include <16f877.h>        //Incluye la librería del microControlador
#fuses HS,NOPROTECT,
#use delay(clock=20000000)
#org 0x1F00, 0x1FFF void loader16F877(void) {}

//Metodo que enciende y apaga LEDS del puerto B
void main()
{
   while(1)                //Ciclo  while infinito
   {
      output_b(0xFF);      //Pone en alto todos los bits del puerto B
      delay_ms(500);       //Retardo de 1s
      output_b(0x00);      //Pone en bajo todos los bits del puerto B
      delay_ms(500);       //Retrado de 1s
   }
}
```

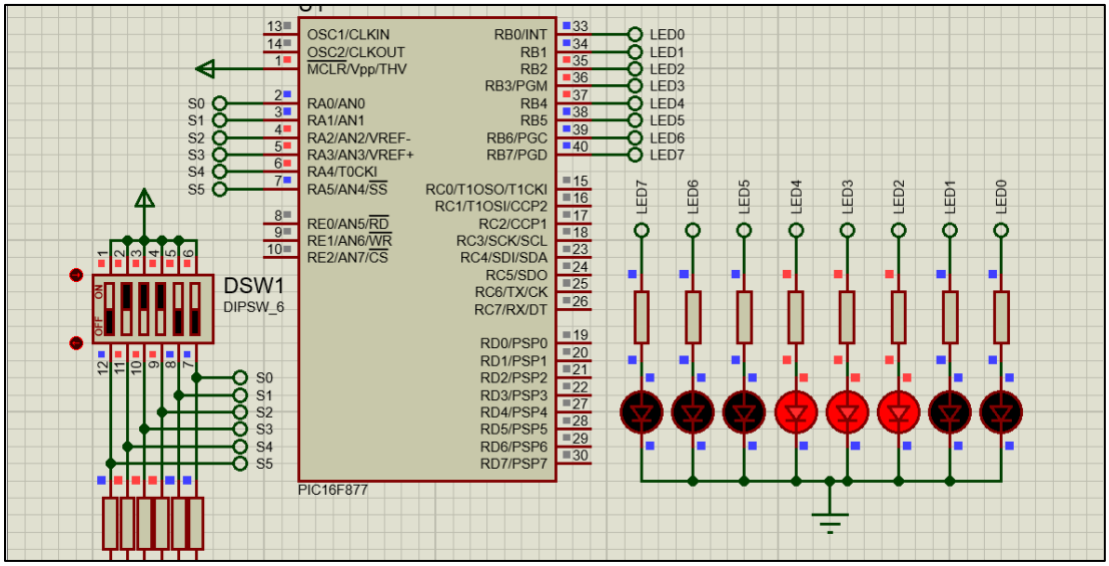



Para este segundo ejercicio, solo fue cuestión de modificar el valor en el método input_b() para que así se pudiera emplear todo el puerto B.

3.- Escribir, comentar, compilar, el siguiente programa usando el ambiente del PIC C Compiler y comprobar el funcionamiento.

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c

```
1  #include <16f877.h>            //Incluye la librería del microControlador
2  #fuses HS,NOPROTECT,
3  #use delay(clock=20000000)
4  #org 0x1F00, 0x1FFF void loader16F877(void) {}
5
6  int var1;                      //Define una variable tipo entero
7                                 //Metodo que lee el contenido del puerto A y mues
8                                 //el resultado en el puerto B
9  void main()
10 {
11    while(1)                    // Ciclo while infinito
12    {
13       var1=input_a();          // Guarda en var1 el contenido del puerto A
14       output_b(var1);          // Manda el contenido de var1 al puerto B
15    }
16 }
```

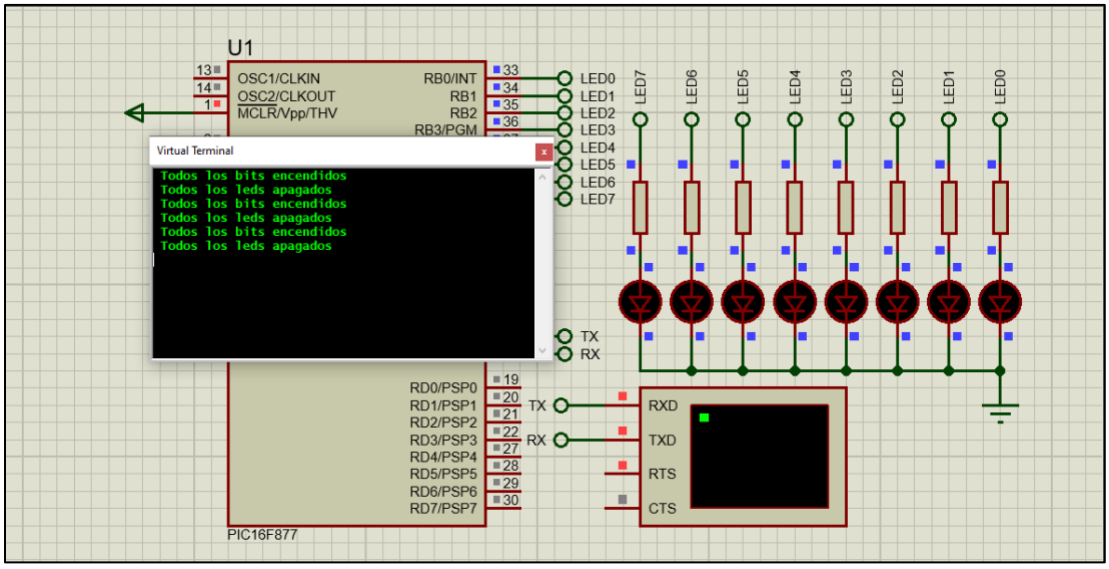



De igual manera para desarrollar este ejercicio solo se transcribió el ejemplo proporcionado en la práctica y se conoció que hay método específicos para cada puerto que permite capturar o transmitir datos, en este caso la función input_a() permitió capturar los datos que estaban ingresando por el puerto A.

4.- Escribir, comentar, compilar, el siguiente programa usando el ambiente del PIC C Compiler y comprobar el funcionamiento.

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c

```c
#include <16f877.h>        //Incluye la librería del microControlador
#fuses HS,NOPROTECT,
#use delay(clock=20000000)
//Configura y habilita el puerto de comunicaión SERIAL
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void) {}

// Metodo que enciende y apaga los bits del puerto B
// ademas de mostrar en la hyperterminal el mensaje
// de encendido y apagado.

void main()
{
    while(1)  //Ciclo while infinito
    {
        output_b(0xff); //Enciende los bits del puerto B
        // Muestra mensaje en la Hyperterminal
        printf(" Todos los bits encendidos \n\r");
        delay_ms(1000);
        output_b(0x00);  //Apaga todos los bits del puerto B
        // Muestra mensaje en la hyperterminal
        printf(" Todos los leds apagados \n\r");
        delay_ms(1000);
     }
}
```

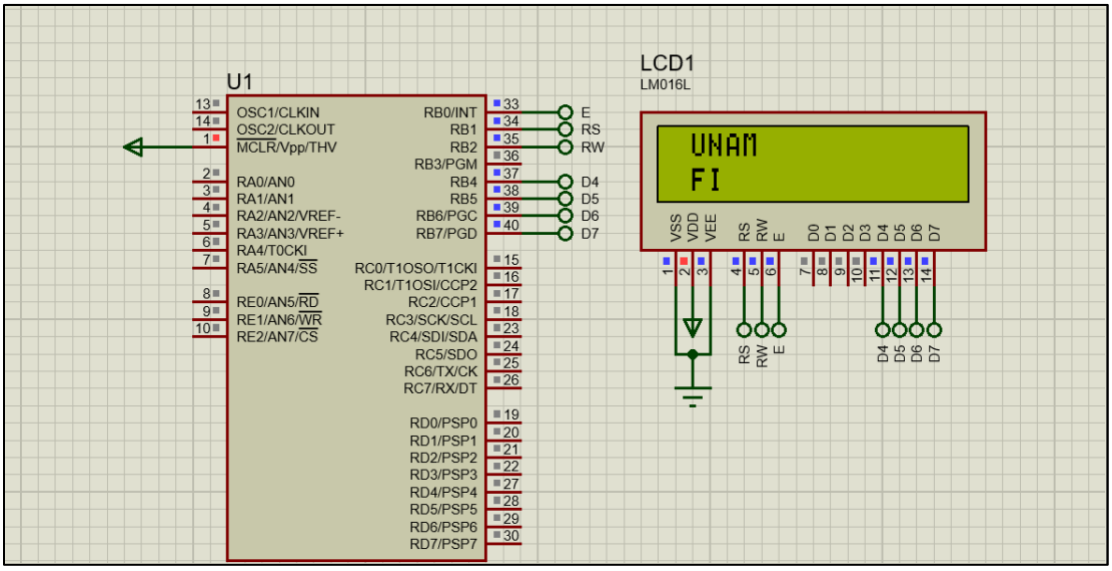



Para la realización de este ejercicio, se empleó la configuración de la comunicación serial del puerto serie, de esta manera se puedo trabajar con una terminal virtual.

5.- Escribir, comentar, compilar, el siguiente programa usando el ambiente del PIC C Compiler y comprobar el funcionamiento.

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c | Ejercicio 5.c

```
#include <16f877.h>              //Incluye la librería del microControlador
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)
#define use_portb_lcd true       //Declara el puerto B como comunicación con
#include <lcd.c>                 //el LCD. Librería que contiene el LCD

void main()
{
    lcd_init();                  //Inicialización del LCD
    while(TRUE)                  //Ciclo while infinito
    {
        lcd_gotoxy(1,1);         //Función para determinar la posición del LCD
        printf(lcd_putc," UNAM \n");
        lcd_gotoxy(1,2);
        printf(lcd_putc," FI \n");
        delay_ms(300);
     }
}
```




Para el desarrollo de este ejercicio se tuvo que emplear la librería del LCD, la declaración del puerto B como puerto paralelo para la comunicación con este elemento.

6.- Realizar un programa empleando el compilador de C, para ejecutar las acciones mostradas en la siguiente tabla, estas son controladas a través del puerto serie; usar retardos de $\frac{1}{2}$ segundo.

| DATO | ACCION<br>Puerto B | *Ejecución* |
|---|---|---|
| 0 | Todos los bits apagados | *00000000* |
| 1 | Todos los bits encendidos | *11111111* |
| 2 | Corrimiento del bit más significativo hacia la derecha | *10000000*<br>*……..*<br>*00000001* |
| 3 | Corrimiento del bit menos significativo hacia la izquierda | *00000001*<br>*……..*<br>*10000000* |
| 4 | Corrimiento del bit más significativo hacia la derecha y a la izquierda | *10000000*<br>*……..*<br>*00000001*<br>*……..*<br>*10000000* |
| 5 | Apagar y encender todos los bits. | *00000000*<br>*11111111* |

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c | Ejercicio 5.c | Ejercicio 6.c

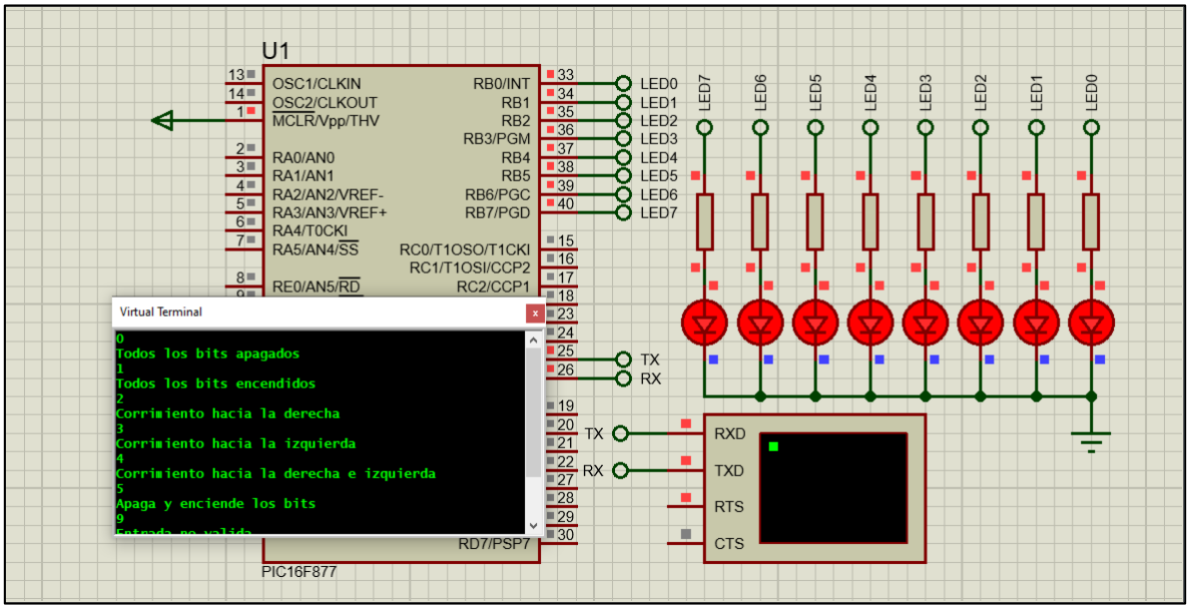
```
#include <16f877.h>   //Incluye la librería del microprocesador
#fuses HS,NOPROTECT,
#use delay(clock=20000000) //Frec. de oscilación 20Mhz

//Configura y activa el puerto SERIAL
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void) {}

char var1;   //Define una variable tipo CHAR
int var2;    //Define una variable tipo entero

void main()
{
    while(1) //Ciclo While infinito
    {
        //Lee la variable que es escrita en la
        //Hyperterminal y la guarda en var1
        var1=getchar();

        if (var1=='0'){
           printf("0\n\r");
           printf("Todos los bits apagados\n\r");
           output_b(0x00);   //Pone en bajo los bit's del puerto B
        }
        if (var1=='1'){
           printf("%c\n\r", var1);
           printf("Todos los bits encendidos\n\r");
           output_b(0xff);   //Pone en alto los bit del puerto B
        }
        if (var1=='2'){
           printf("%c\n\r", var1);
           printf("Corrimiento hacia la derecha\n\r");
           var2=0x80;       //Inicializa con el valor de "10000000"
```

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c | Ejercicio 5.c | Ejercicio 6.c

```
35              delay_ms(500);    //Retardo 1/2s
36              //Se define un ciclo que realizar un corrimiento
37              //a la derecha
38              do{
39                   var2=var2/2; //Realiza una divisón entre 2 lo que
40                   //permite realizar un corrimiento a la derecha
41                   output_b(var2);//Muestra var2 en el puerto B
42                   delay_ms(500);
43              }while (var2!=1); //Repite el ciclo mientas var2 sea
44                   //diferente de 0x01="00000001"
45          }
46          if (var1=='3'){
47             printf("%c\n\r", var1);
48             printf("Corrimiento hacia la izquierda\n\r");
49             var2=0x01;       //Inicializa con el valor de "00000001"
50             output_b(var2);   //Manda var2 al puerto B
51             delay_ms(500);    //Retardo 1/2s
52             do{
53                  var2+=var2;    //Realiza corrimiento a la izquierda
54                  output_b(var2);//Manda var2 al puerto B
55                  delay_ms(500);
56             }while (var2!=0x80);//Repite el ciclo mientras var2 sea
57                                 //diferente de 0x80=b"10000000"
58          }
59          if (var1=='4'){
60              printf("%c\n\r", var1);
61              printf("Corrimiento hacia la derecha e izquierda\n\r");
62              //Seinicializa var2=b"10000000"
63              var2=0x80;
64              output_b(var2);
65              delay_ms(500);
66
67              //Ciclo que realiza corrimiento a la derecha
68              do{
```

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c | Ejercicio 5.c | Ejercicio 6.c

```
67              //Ciclo que realiza corrimiento a la derecha
68              do{
69                  var2=var2/2;
70                  output_b(var2);
71                  delay_ms(500);
72              }while (var2!=1);
73              //Seinicializa var2=b"00000001"
74              var2=0x01;
75              output_b(var2);
76              delay_ms(500);
77
78              //Ciclo que realiza corrimiento a la izquierda
79              do{
80                  var2+=var2;
81                  output_b(var2);
82                  delay_ms(500);
83              }while (var2!=0x80);
84          }
85          if (var1=='5'){
86             printf("%c\n\r", var1);
87             printf("Apaga y enciende los bits\n\r");
88             output_b(0xff);     //Enciende los bits del puerto B
89             delay_ms(500);
90             output_b(0x00);     //Apaga los bits del puerto B
91             delay_ms(500);
92          }
93          if (var1 > '5'){
94             printf("%c \n\r", var1);
95             printf("Entrada no valida\n\r");
96          }
97      }
98  }
```




Para el desarrollo de este ejercicio solo se tuvo que emplear la función getchar() para capturar el dato ingresado en la terminal virtual y mediante estructuras condicionales determinar verificar el valor del dato y determinar el comportamiento de esta.

**Algoritmo:**

a) Configurar la comunicación serie.
b) Capturar el dato recibido por la comunicación serie.
c) Verificar el valor del ato y determinar:
    a. Si dato = 0, apagar todos los bits.
    b. Si dato = 1, encender todos los bits.
    c. Si dato = 2, corrimiento hacia la derecha.
    d. Si dato = 3, corrimiento hacia la izquierda.
    e. Si dato = 4, corrimiento a la derecha e izquierda.
    f. Si dato = 5, encender y apagar todos los bits.
    g. Si dato > 5 entrada no válida.

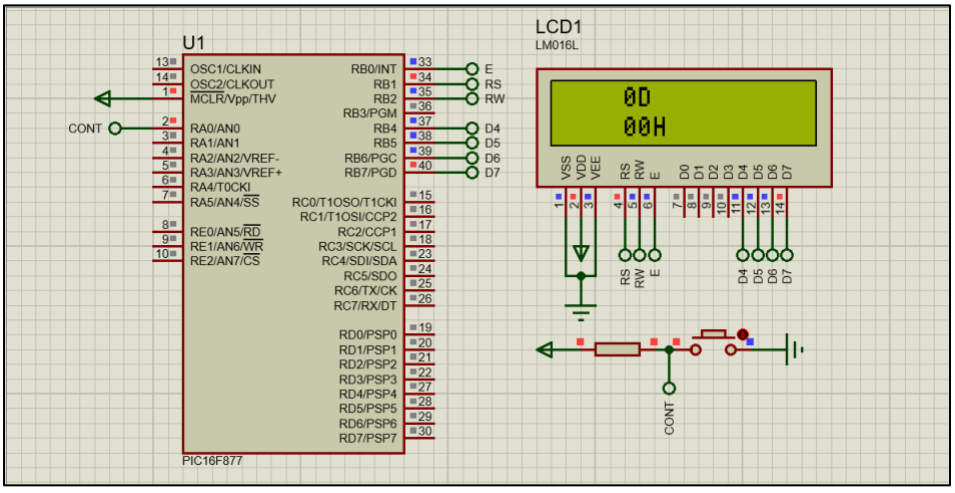
7.- Realizar un programa que muestre en un Display de Cristal Líquido, la cantidad de veces que se ha presionado un interruptor, el cual está conectado a la terminal A0.

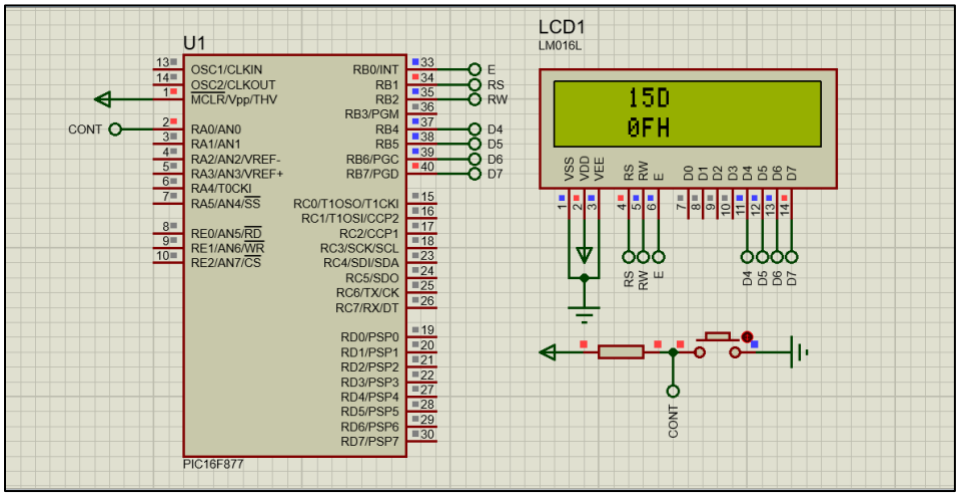
El despliegue para mostrar es:

a) Primera línea, columna 5: La cuenta decimal.
b) Segunda línea, columna 5: la cuenta en hexadecimal.

Ejercicio 1.c | Ejercicio 2.c | Ejercicio 3.c | Ejercicio 4.c | Ejercicio 5.c | Ejercicio 6.c | Ejercicio

```
#INCLUDE <16F877.H>
#FUSES HS,NOWDT,NOPROTECT,NOLVP
#USE DELAY(CLOCK=20000000)
#ORG 0x1F00, 0x1FFF VOID LOADER16F877(VOID) {}
#DEFINE USE_PORTB_LCD TRUE
#INCLUDE <LCD.C>

INT VAR1, VAR2 = 0;

VOID MAIN()
{
    LCD_INIT();
    WHILE(TRUE)  //CICLO WHILE INFINITO
    {
        VAR1=INPUT_A();
        IF(VAR1==0){
          VAR2 = VAR2 + 1;
        }
        LCD_GOTOXY(5,1);
        PRINTF(LCD_PUTC,"%dD", VAR2);
        LCD_GOTOXY(5,2);
        PRINTF(LCD_PUTC,"%XH", VAR2);
        DELAY_MS(250);
     }
}
```






Para la realización de este ejercicio, se combinaron las lecciones de los ejercicios anteriores en donde manejó el LCD y la captura de datos de los puertos paralelos, así que sólo se emplearon las funciones antes vistas.

**Algoritmo:**

a) Determinar variable acumulativa y variable de estado.
b) Si variable de estado es = 0 entonces aumenta en 1 variable acumulativa.
c) Se imprime la variable acumulativa con los formatos solicitados.

**Conclusiones:**

**Carreón Guzmán Mariana**

En esta práctica pudimos comprender el funcionamiento de el PIC, hicimos uso de algoritmos para la creación de las soluciones, esto resultó un poco más sencillo que en las prácticas anteriores ya que en este caso usamos lenguaje C, al tener conocimientos más amplios en dicho lenguaje facilitó la realización de los ejercicios. Por otra parte pude observar como controlar el puerto fuente y hacer uso de él en los ejercicios planteados.

**Rojas Méndez Gabriel:** con el desarrollo de esta práctica me familiarice con el entorno de C y el PIC, ya que de esta manera la mayoría de las cosas solicitadas en la práctica fueron muy sencillas y rápidas, solo con investigar un poco sobre el entorno se podían resolver las problemáticas planteadas, además, cabe mencionar que programar en un lenguaje de alto nivel es mucho más fácil y más práctico que en uno de bajo nivel como lo es ensamblador.