Universidad Nacional
Autónoma de México

Facultad de ingeniería

Microcomputadoras
Grupo:1

Proyecto 2
LCD 16x2 en PIC16F877

Bárcenas Martínez Edgar Daniel
416075773

27/03/2019

Introduccion:

La pantalla de cristal líquido o LCD (Liquid Crystal Display) es un dispositivo microControlado de visualización gráfico para la presentación de caracteres, símbolos o incluso dibujos (en algunos modelos), es este caso dispone de 2 filas y de 16 caracteres cada una y cada carácter dispone de una matriz de 5x7 puntos (pixels), aunque los hay de otro número de filas y caracteres. Este dispositivo está gobernado internamente por un microcontrolador y regula todos los parámetros de presentación, este modelo es el más comúnmente usado y esta información se basará en el manejo de este u otro LCD compatible.

El LCD tiene un aspecto físico como el mostrado en la figura 1. Está constituido por un circuito impreso en el que están integrados los controladores del display y los pines para la conexión del display. Sobre el circuito impreso se encuentra el LCD en sí, rodeado por una estructura metálica que lo protege. En total se pueden visualizar 2 líneas de 16 caracteres cada una, es decir, 2x16=32 caracteres, como se muestra en la figura 2. A pesar de que el display sólo puede visualizar 16 caracteres por línea, puede almacenar en total 40 por línea. Es el usuario el que especifica qué 16 caracteres son los que se van a visualizar.

Figura 1: Aspecto físico del LCD

Figura 2: Capacidad de visualización de caracteres del display

LA MEMORIA DEL LCD El LCD

Dispone de dos tipos de memorias independientes: la DD RAM y la CG RAM.

DD RAM (Display Data Ram)

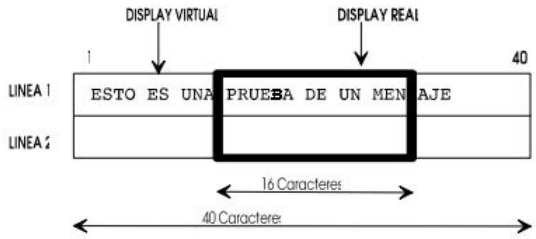


Figura 4: Display virtual y display real

En esta memoria se almacenan los caracteres que están siendo visualizados o que se encuentran en posiciones no visibles. El display almacena en esta memoria dos líneas de 40 caracteres pero sólo se visualizan 2 líneas de 16 caracteres. Por ello la DD RAM tiene un tamaño de 2x40=80 bytes. Debido a esta peculiar disposición de la DD RAM se puede pensar en el display como un display virtual constituido por dos líneas de 40 caracteres cada una (Fig. 4). La posición situada más a la izquierda de cada línea es la posición 1 y la situada más a la derecha es la posición 40. Para localizar los elementos dentro del display virtual se va a utilizar un par de coordenadas (x,y) donde x representa la posición horizontal (comprendida entre 1-40) e y representa la línea (1-2). El display real es una ventana en la que se visualizan dos líneas de 16 caracteres. Es lo que el usuario está viendo.

LA CG RAM (Character Generator RAM)

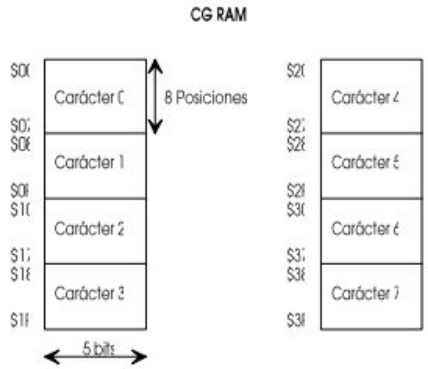


Figura 7: Mapa de memoria de la CG RAM

La CG RAM es la memoria que contiene los caracteres definibles por el usuario. Está formada por 64 posiciones, con direcciones $00-$3F. Cada posición es de 5 bits. La memoria está dividida en 8 bloques, correspondiendo cada bloque a un carácter definible por el usuario. Por ello el usuario puede definir como máximo 8 caracteres, cuyos códigos van del 0 al 7. En la figura 7 se ha representado la CG RAM. Todas las direcciones están en hexadecimal. Como se representó en la figura 3, cada carácter está constituido por una matriz de 5 columnas x 8 filas. Para definir un carácter y asignarle por ejemplo el código 0 habrá que almacenar en las posiciones $00-$07 los valores binarios de las 8 filas del carácter del usuario. Un bit con valor 1 representa un punto encendido. Un bit con valor 0 es un punto apagado.

Características principales:
-Pantalla de caracteres ASCII, además de los caracteres Kanji y Griegos.
-Desplazamiento de los caracteres hacia la izquierda o la derecha.
-Proporciona la dirección de la posición absoluta o relativa del caracter.
-Memoria de 40 caracteres por línea de pantalla.
-Movimiento del cursor y cambio de su aspecto.
-Permite que el usuario pueda programar 8 caracteres.
-Conexión a un procesador usando un interfaz de 4 u 8 bits

Funcionamiento:
Para comunicarse con la pantalla LCD podemos hacerlo por medio de sus patitas de entrada de dos maneras posibles, con bus de 4 bits o con bus de 8 bits, este último es el que explicare y la rutina también será para este. En la siguiente figura vemos las dos maneras posibles de conexionar el LCD con un pic16F84.

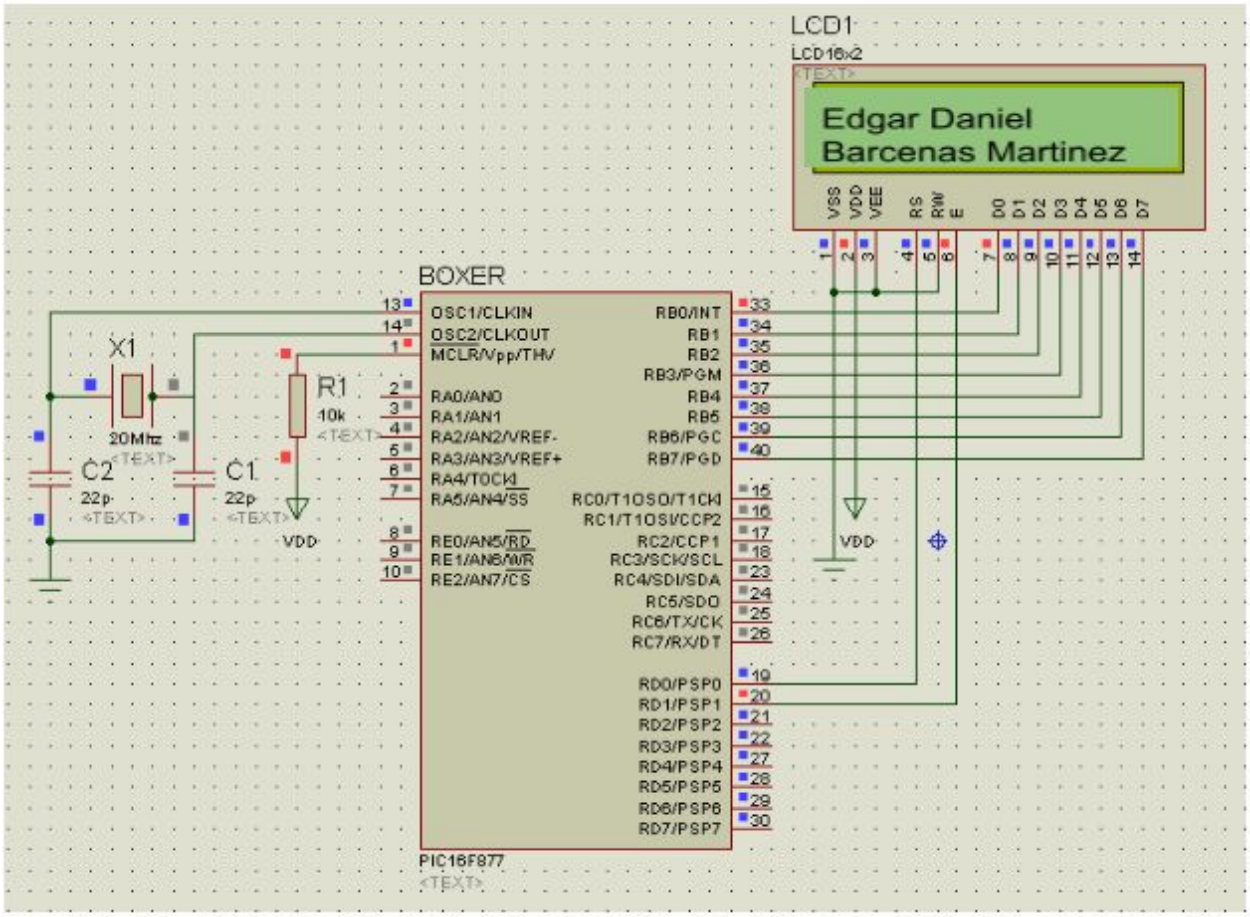
Descripción de Pines:

PIN Nº	SIMBOLO	DESCRIPCION
1	Vss	Tierra de alimentación GND
2	Vdd	Alimentación de +5V CC
3	Vo	Contraste del cristal liquido. (0 a +5V)
4	RS	Selección del registro de control/registro de datos: RS=0 Selección registro de control RS=1 Selección registro de datos
5	R/W	Señal de lectura/escritura: R/W=0 Escritura (Write) R/W=1 Lectura (Read)
6	E	Habilitación del modulo: E=0 Módulo desconectado E=1 Módulo conectado
7-14	D0-D7	Bus de datos bidireccional.

Objetivo:

Realizar un programa en lenguaje ensamblador que muestra un mensaje en un LCD 16x2 cuando reciba como entrada un 0,recorrer hacia la derecha mi nombre cuando reciba un 1, recorrer hacia la izquierda cuando reciba un 2 y mostrar un carácter definido por el usuario cuando reciba un 3.

Diagrama:



Desarrollo:

Inicialización,Comando y Envío de Datos.

```
INICIA_LCD:
    bcf PORTD,0       ; RS=0 MODO INSTRUCCION
    movlw 0x01        ; El comando 0x01 limpia la pantalla en el LCD
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    movlw 0x0C        ; Selecciona la primera línea
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    movlw 0x3C        ; Se configura el cursor
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    bsf PORTD, 0      ; Rs=1 MODO DATO
    return

    ;Subrutina para enviar comandos
COMANDO:
    bsf PORTD,1       ; Pone ENABLE en 1
    call DELAY        ; Tiempo de espera
    call DELAY
    bcf PORTD, 1      ; ENABLE=0
    call DELAY
    return

    ;Subrutina para enviar un dato
ENVIA:
    bsf PORTD,0       ; RS=1 MODO DATO
    call COMANDO      ; Se da de alta el comando
    return
    ;Configuración Línea1 2 LCD
```

1.Cargar los Caracteres que muestran mi nombre.

```
M3:
     ;Edgar
     movlw 'E'          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
      movlw 'D'          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
     movlw 'G'          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
      movlw 'A'          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
     movlw 'R'          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
      movlw ' '          ;Mueve 'H' a W
     movwf PORTB        ;Mueve lo que hay en W a PORTB
     call ENVIA
```

2.Desplazar caracteres a la derecha.

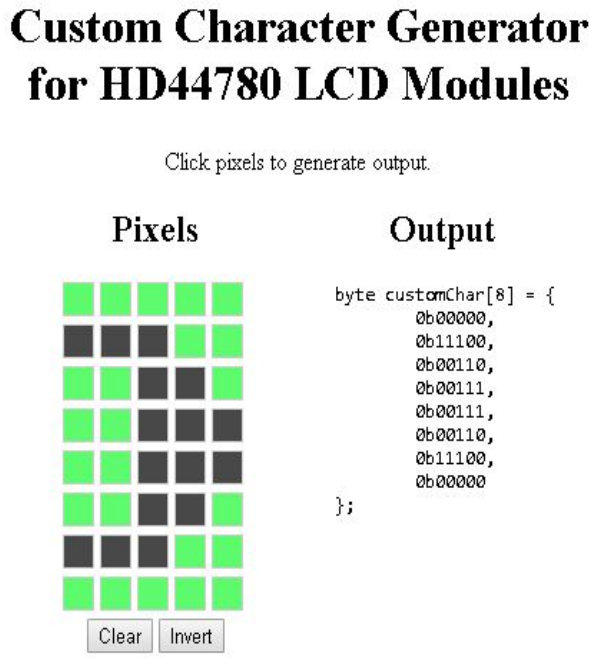
```
bcf PORTD,0
movlw 0x1C
movwf PORTB
call COMANDO
```

3.Desplazar caracteres a la izquierda.

```
bcf PORTD,0
movlw 0x18
movwf PORTB
call COMANDO
```

4.Crear Caracter.

```
ESCRIBIR:
    bcf PORTD,0
    movlw 0x40        ; CGRAM
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    bsf PORTD,0
    movlw b'00000100'
    movwf PORTB
    call COMANDO
    movlw b'00000100'
    movwf PORTB
    call COMANDO
    movlw b'00001110'
    movwf PORTB
    call COMANDO
    movlw b'00001110'
    movwf PORTB
    call COMANDO
    movlw b'00011111'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    return
```



Codigo:

```
    processor 16F877a
    include <P16F877a.INC>

;Variables para DELAY
val1 equ 0x30
val2 equ 0x31
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
cte1 equ 100h
cte2 equ 500h
cte3 equ 600h


;Definicion de variables a utilizar para
;comparar las entradas a traves del puerto A
v0 equ h'24'
v1 equ h'25'
v2 equ h'26'
v3 equ h'27'
c0 equ 0h
c1 equ 1h
c2 equ 2h
c3 equ 3h

    org 0               ;Vector de RESET
    goto INICIO
    org 5               ;Inicio del Programa

    ;Configuración de puertos
INICIO:
    clrf PORTB          ;Limpia PORTB
    clrf PORTD          ;Limpia PORTD
    clrf PORTA
    bsf STATUS, RP0
    bcf STATUS, RP1     ;Selecciona el banco 1
    clrf TRISB          ;Configura PORTB como salida
    clrf TRISD          ;Configura PORTD como salida
    movlw 06h       ;Configura puertos A y E como digitales
    movwf ADCON1
    movlw 3fh       ;Configura el Puerto A como entrada
    movwf TRISA
    bcf STATUS,RP0      ;Regresa al banco 0
    call ESCRIBIR

CICLO:
       movlw c0
       movwf v0
       movfw PORTA      ;Mueve lo que hay en PORTA a W
       xorwf v0,w       ;Verifica si la entrada es $00
       btfsc STATUS,Z   ;z=0?
       goto START_LCD         ;NO, entonces v0=W
                        ;SI, entonves v0!=W
       movlw c1
       movwf v1
       movfw PORTA
       xorwf v1,w       ;Verifica si la entrada es $01
       btfsc STATUS,Z
       goto START_LCD1

       movlw c2
       movwf v2
       movfw PORTA
       xorwf v2,w       ;Verifica si la entrada es $02
       btfsc STATUS,Z
       goto START_LCD2

       movlw c3
       movwf v3
       movfw PORTA
       xorwf v3,w       ;Verifica si la entrada es $03
       btfsc STATUS,Z
       goto  START_LCD3
;;;;;;;;;;
START_LCD:
    call INICIA_LCD     ;Configura el LCD
    call M1             ;Muestra Mensaje
    call LINEA2         ;Configura linea 2
    call M2             ;Muestra Mensaje
    call RETARDO
    goto CICLO
START_LCD1:
    call INICIA_LCD     ;Configura el LCD
    call M3             ;Muestra Mensaje
    goto CICLO
START_LCD2:
    call INICIA_LCD     ;Configura el LCD
    call LINEA2         ;Configura linea 2
    call M4             ;Muestra Mensaje
    goto CICLO
START_LCD3:
    call INICIA_LCD     ;Configura el LCD
    call M5            ;Muestra Mensaje
    goto CICLO

M5:
    call CENTRO
    movlw   0x00
    movwf   PORTB
    call    ENVIA
    call    RETARDO
    return
```

```
ESCRIBIR:
    bcf PORTD,0
    movlw 0x40        ; CGRAM
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    bsf PORTD,0
    movlw b'00000100'
    movwf PORTB
    call COMANDO
    movlw b'00000100'
    movwf PORTB
    call COMANDO
    movlw b'00001110'
    movwf PORTB
    call COMANDO
    movlw b'00001110'
    movwf PORTB
    call COMANDO
    movlw b'00011111'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    movlw b'00010001'
    movwf PORTB
    call COMANDO
    return

;Mensaje a enviar
M3:
    ;Edgar
    movlw 'E'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'D'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'G'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'A'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'R'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw ' '           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'B'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'A'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'R'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'C'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'E'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'N'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'A'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'S'           ;Mueve 'H' a W
    movwf PORTB         ;Mueve lo que hay en W a PORTB
    call ENVIA

    bcf PORTD,0
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
```

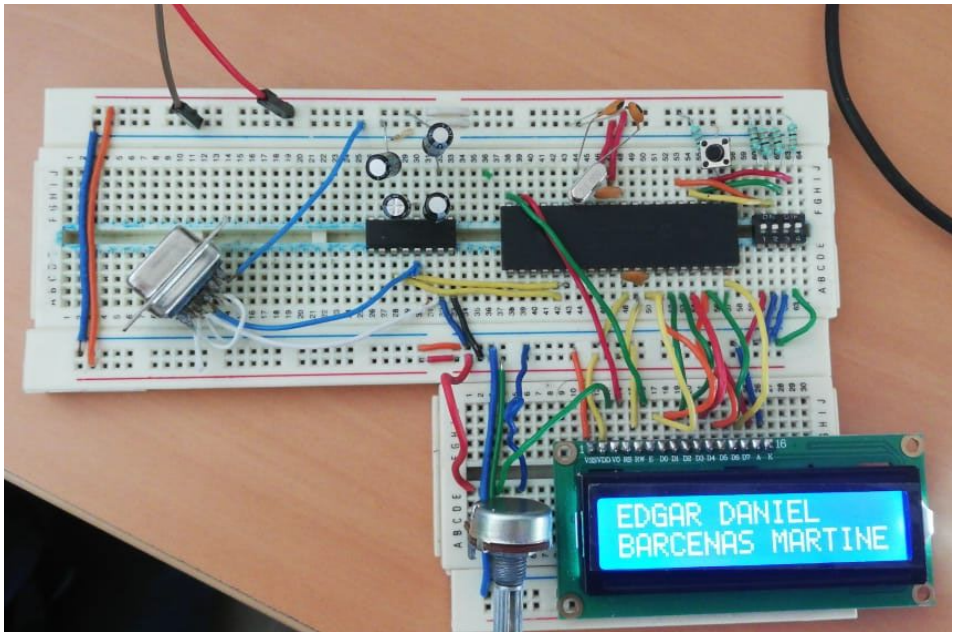
```
M4:
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
    movlw 0x1C
    movwf PORTB
    call COMANDO
```

```
    movlw 'E'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'D'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'G'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'A'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'R'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw ' '        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA

    movlw 'B'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'A'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'R'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'C'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'E'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'N'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
    movlw 'A'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA
     movlw 'S'        ;Mueve 'H' a W
    movwf PORTB      ;Mueve lo que hay en W a PORTB
    call ENVIA

    bcf PORTD,0
    movlw 0x18
    movwf PORTB
    call COMANDO
    movlw 0x18
    movwf PORTB
    call COMANDO
    movlw 0x18
    movwf PORTB
```

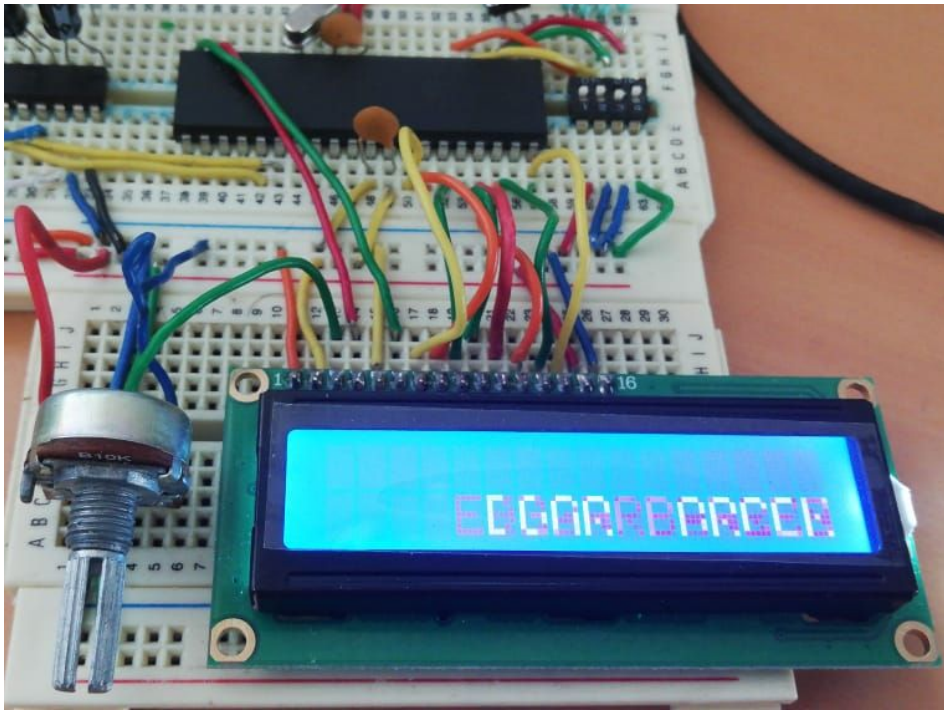
```
INICIA_LCD:
    bcf PORTD,0       ; RS=0 MODO INSTRUCCION
    movlw 0x01        ; El comando 0x01 limpia la pantall
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    movlw 0x0C        ; Selecciona la primera línea
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    movlw 0x3C        ; Se configura el cursor
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    bsf PORTD, 0      ; Rs=1 MODO DATO
    return

    ;Subrutina para enviar comandos
COMANDO:
    bsf PORTD,1       ; Pone ENABLE en 1
    call DELAY        ; Tiempo de espera
    call DELAY
    bcf PORTD, 1      ; ENABLE=0
    call DELAY
    return

    ;Subrutina para enviar un dato
ENVIA:
    bsf PORTD,0       ; RS=1 MODO DATO
    call COMANDO      ; Se da de alta el comando
    return
    ;Configuración Líneal 2 LCD
LINEA2:
    bcf PORTD, 0      ; RS=0 MODO INSTRUCCION
    movlw 0xc0        ; Selecciona línea 2 en el LCD
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    return

CENTRO:
    bcf PORTD, 0      ; RS=0 MODO INSTRUCCION
    movlw 0xc7        ; Selecciona línea 2 en el LCD
    movwf PORTB
    call COMANDO      ; Se da de alta el comando
    return

RETARDO:
     movlw cte1
     movwf valor1
tres movwf cte2
     movwf valor2
dos  movlw cte3
     movwf valor3
uno  decfsz valor3
     goto uno
     decfsz valor2
     goto dos
     decfsz valor1
     goto tres
     return
```

Resultado:

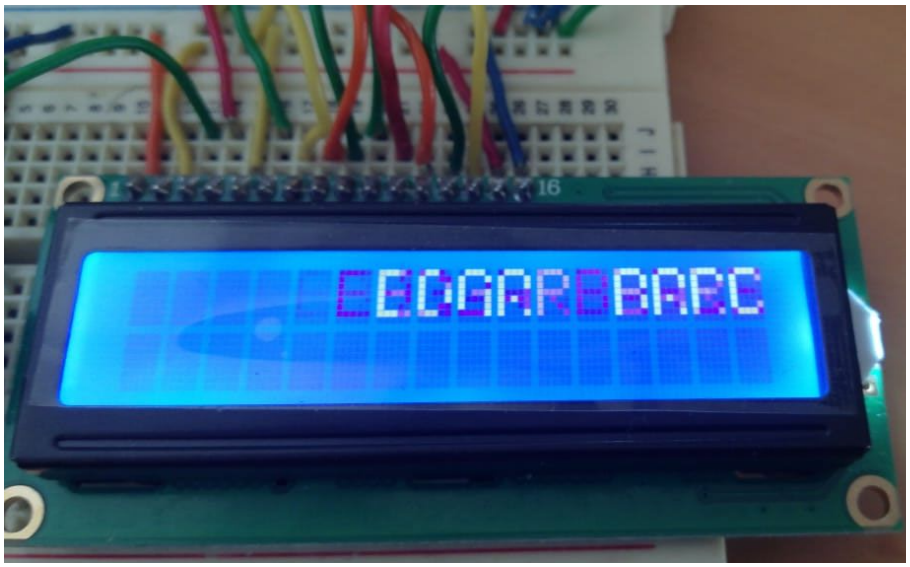
1.Cargar los Caracteres que muestran mi nombre.



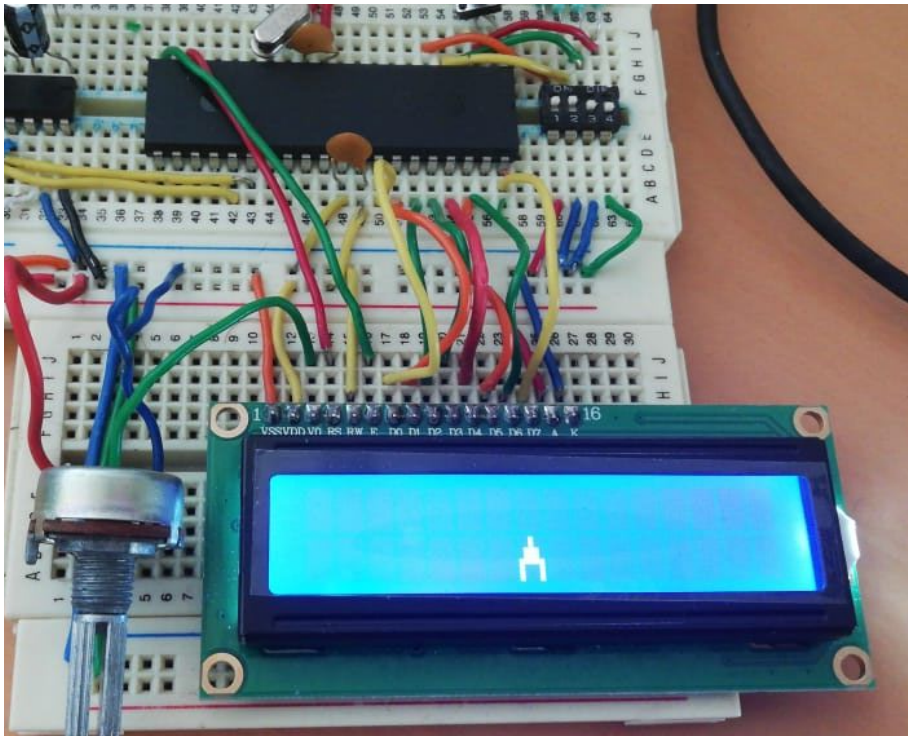
2.Desplazar caracteres a la izquierda



3.Desplazar caracteres a la derecha



4.Crear Caracter.



Conclusiones:

En este segundo proyecto aprendí a manejar el LCD, aprendi a generar instrucciones para el LCD con el uso de ensamblador en el PIC16f877A.

Logre entender más acerca de la configuración de los puertos de entrada y salida del PIC, logre que en el LCD se mostrará un mensaje, también aprendí a desplazar los caracteres hacia la derecha e izquierda y también a generar un nuevo carácter con la CGRAM, entre las dificultades que encontré durante el desarrollo de este proyecto fue entender los comandos que se envían al LCD para que realice las diferentes configuraciones, otra de mis dificultades fue la realización del nuevo carácter porque no sabía cómo guardar el caracter y luego volverlo a llamar.

.

Referencias:

http://www.x-robotics.com/rutinas.htm#LCD

https://omerk.github.io/lcdchargen/

http://www.openboxer.260mb.com/asignaturas/microcomp/LCD16x2.html

http://www.electronica60norte.com/mwfls/pdf/displayLCD.pdf