Aplicaciones de Microcontroladores

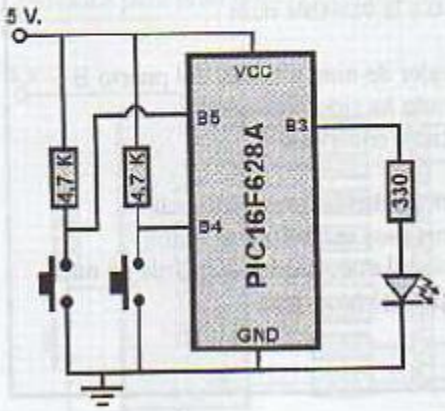
Semana 5

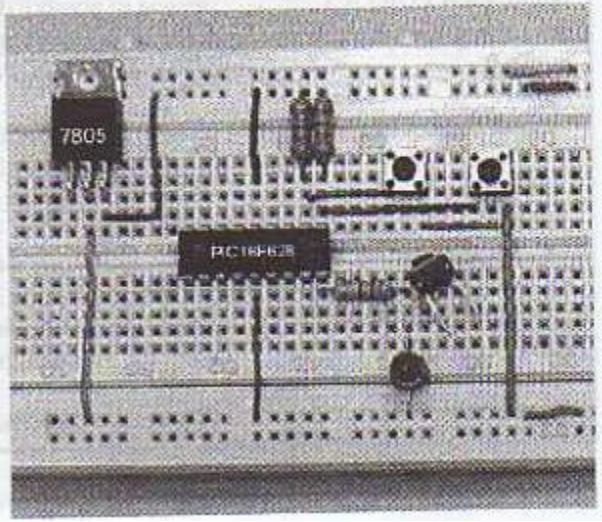
Haga un proyecto en el que al presionar un botón este encienda un led intermitente de 8 repeticiones de 250 mls. Luego el led permanece apagado y el programa vuelve a sensar el pulsador.

Con un pulsador haga que 8 leds conectados en el puerto B, se enciendan de derecha a izquierda uno a la vez, empezando de B0 a B7, al final este último permanece encendido, con otro pulsador haga que los leds se desplacen uno a uno hacia la derecha, es decir desde B7 que fue el último y que está actualmente encendido se desplace hasta B0, las pausas son de 300 mls.

MATERIALES.

-1 LED de 5mm.
-1 resistencias de 330Ω a ½ vatio, naranja-naranja-café
-2 resistencia de 4,7 KΩ a ½ vatio, amarillo-violeta-rojo
-2 pulsadores para protoboard normalmente abierto como los de la figura 5.3.3.2






```
Pbaja VAR portb.5      ;el portb.5 se llamará pbaja
Psube VAR portb.4      ;el portb.4 se llamará psube
led VAR portb.3        ;el portb.3 se llamará led
xy VAR byte            ;crea la variable xy con tamaño de 255
veces VAR byte         ;crea la variable veces con tamaño de 255
veces = 100            ;carga con 100 a la variable veces

inicio:
   HIGH led            ;encender el led
   GOSUB timer         ;ir y retornar de timer
   LOW led             ;apaga el led
   GOSUB timer         ;ir y retornar de timer
GOTO inicio

timer:
   IF psube = 0 THEN GOSUB restar     ;pregunta si presionó psube
   IF pbaja = 0 THEN GOSUB sumar      ;pregunta si presionó pbaja
FOR xy = 1 TO veces                   ;repite desde 1 hasta el valor que contenga veces
   pause 5                            ;retardo de 5 mls
   NEXT                               ;siguiente repetición
RETURN                                ;retornar al que le envió

sumar:
IF veces>150 THEN RETURN              ;retorna si veces excede de 150
   veces=veces+5                      ;suma 5 a la variable veces
   RETURN                             ;retorna hacia el que le envió

restar:
   IF veces<10 THEN RETURN            ;retorna si veces es menor que 10
   veces =veces-5                     ; resta 5 a la variable veces
   RETURN                             ;retorna hacia el que le envió


END                                   ; fin de la programación
```

Figura 5.3.3.3. led variable.pbp *Programa para el led intermitente de velocidad variable.*

5.4.1. MANEJO DE UN DISPLAY DE 7 SEGMENTOS CON EL CI. 7447.

Los displays son muy utilizados para visualizar datos. Para esta práctica se utilizá como periférico de salida un display tipo ánodo común, para lo cual se facilita el diagrama en la figura 5.4.1.2. El proyecto consiste en hacer un contador decimal (0,..,9),con intervalos de 0,5 segundos.

El programa es muy similar al del 5.3.2. contador binario, con la diferencia que sólo se necesita los 4 bits más bajos (B.0, B.1, B.2, y B.3), el decodificador binario a 7 segmentos (7447), es el encargado de transformar el número binario que ingresa a número decimal.

MATERIALES.
-1 DISPLAY ánodo común preferible como el de la figura 5.4.1.2, ideal para protoboards.
-7 resistencias de 330Ω a ½ vatio, naranja-naranja-café
-1 CI. 7447 decodificador BCD

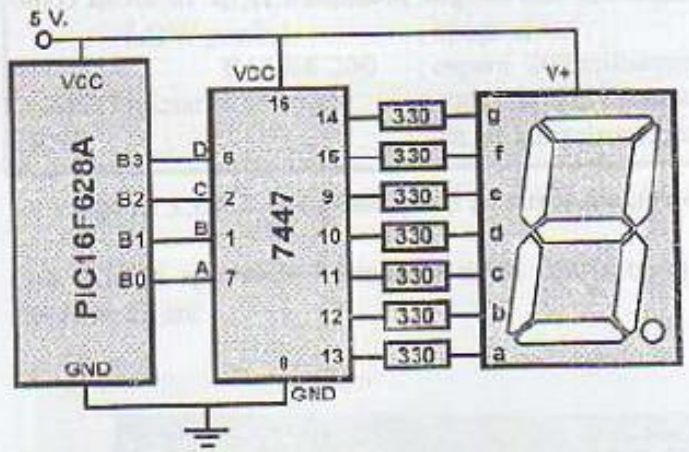


Figura 5.4.1.1. Diagrama de conexión de un display ánodo común con el BCD 7447

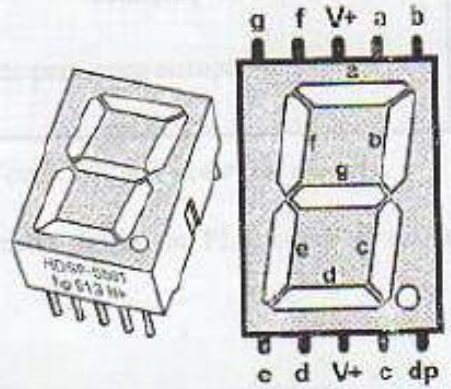


Figura 5.4.1.2. Esquema de pines de un display a. c. HDSP-5501

```
trisb=%11110000          ;hace salidas sólo los bits más bajos de Puerto B
numero VAR BYTE          ;crea la variable número con valor 255

encerar:
  numero = 0             ;carga con cero a la variable número
display:
  portb=numero           ;sacar por el puerto b el contenido de número
  PAUSE 500              ;esperar 0,5 segundos
  IF numero=9 THEN encerar ;si número es =9 encerar número =0
  numero=numero + 1      ;sumar 1 a la variable número
 GOTO display            ;ir a display
END
```

```
trisb=%11110000          ;hace salidas sólo los bits más bajos de Puerto B
numero VAR BYTE          ;crea la variable número con valor 255
bot VAR portb.4          ;nombre para el puerto B.4
encerar:
  numero = 0             ;carga con cero a la variable número
display:
  portb=numero           ;sacar por el puerto b el contenido de número
  IF bot=0 THEN aumentar ; si el botón es pulsado ir a aumentar
GOTO display             ;ir a display

aumentar:
  IF bot=0 THEN aumentar      ; si el botón sigue pulsado encerrar en aumentar
  PAUSE 200                   ;esperar 0,2 segundos
  IF numero=9 THEN encerar    ;si número es =9 encerar número =0
  numero=numero + 1           ;sumar 1 a la variable número
 GOTO display                 ;ir a display
END
```

Figura 5.4.2.1. display7seg boton.pbp *Programa para el display de 7 segmentos con pulsador.*

5.4.3. MANEJO DE UN DISPLAY DE 7 SEGMENTOS SIN EL CI. 7447.

Como se dijo en un comienzo al PIC se le puede programar para reemplazar a casi cualquier circuito integrado, en esta ocasión haremos que el propio PIC sea como un CI. 7447, para esto debemos saber que para sacar el número 3 por ejemplo, debemos calcular el número decimal que hace que se enciendan los segmentos correctos del display, esto se hace de la siguiente forma.

Como debemos encender los segmentos a, b, c, d, y g., revisamos los pines del PIC que les corresponde y estos son: B0, B1, B2, B3, Y B6, respectivamente, estos debemos ponerlos en estado cero lógico para que los segmentos se enciendan (recuerde que el display es ánodo común), y los demás 1 lógico para que permanezcan apagados:

B6	B5	B4	B3	B2	B1	B0
0	1	1	0	0	0	0

equivale al decimal 48, el display saca el número 3

LA DECLARACIÓN LOOKUP. Sirve para obtener un valor constante de una tabla, esto lo hace según el número de veces que repita el **FOR NEXT**, por ejemplo: la 1ra vez toma el dato que se encuentra en el lugar 0, es decir el Nro. 64, la segunda el dato del lugar 1el Nro.121, así sucesivamente, y lo va guardando en la variable dat.

MATERIALES.
-1 DISPLAY ánodo común. preferible como el de la figura 5.4.1.2
-7 resistencias de 330Ω a ½ vatio, naranja-naranja-café

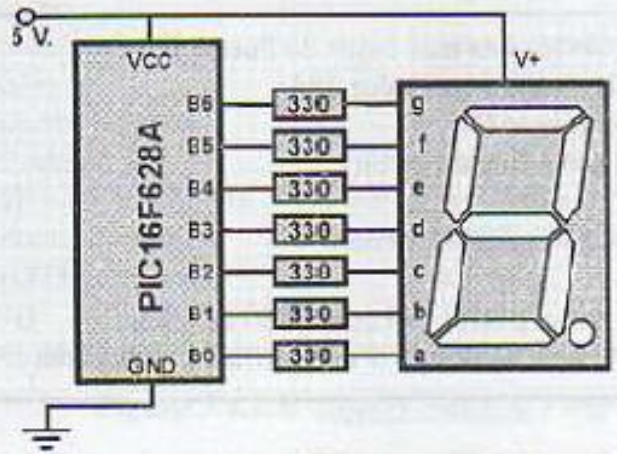


Figura 5.4.3.1. Diagrama de conexión de un display de 7 segmentos directamente al PIC.

```
di  VAR BYTE                          ;crea variable di
dat VAR BYTE                          ;crea variable dat
TRISB = 0                             ;todo el puerto B como salida
prog:
  FOR di=0 TO 15                      ;para repeticiones de 0 a 15
  LOOKUP di,[64,121,36,48,25,18,2,120,0,16,8,3,70,33,6,14],dat    ;toma uno por uno cada
                                      ; valor de la tabla constante y lo guarda en la variable dat
  portb=dat                           ;sacar el contenido de dat por el puerto B
  PAUSE 500                           ;espera de 0,5 seg.
  NEXT di                             ;siguiente repetición
GOTO prog
END
```

Figura 5.4.3.2. display7seg lookup.pbp *Programa para manejar el display directamente.*

5.4.4. MANEJO DE 4 DISPLAYS DE 7 SEGMENTOS CON EL CI. 7447.

El siguiente proyecto debe encender 4 displays para poder mostrar cualquier número desde el 0 hasta el 9999, esto lo conseguimos gracias al transistor tipo PNP, que nos ayudará a multiplexar cada uno de los displays, el funcionamiento es bastante sencillo, debemos conectar los 4 bits más altos a cada transistor y los cuatro bits más bajos al CI. 7447, si por ejemplo queremos sacar el número 6874, primero habilitamos el 4to transistor, el de la derecha y enviamos el número 4, el CI. 7447 se encarga de formar el 4 en el display, luego pasamos a cero lógico el 2do transistor, y los demás lo mantenemos en nivel alto, al mismo tiempo sacamos el número 7 por los bits menos significativos del puerto B, y así consecutivamente, el tiempo que debemos mantener activado cada transistor no puede ser mayor que 5 milisegundos, es decir que los cambios son tan rápidos que el ojo humano ve todos los displays encendidos al mismo tiempo, cuando en realidad sólo se enciende uno a la vez.

Ejemplo: para sacar el Nro 8 en las centenas debemos sacar (176+8), es decir el número 184 porque si analizamos en código binario, tenemos que los bits más bajos entran al CI. 7447, y los bits más altos, son los encargados de encender el display que le corresponde a las centenas.

184 = % 10111000 — 1011 1000 — Este número entra al 7447 el cual saca el 8

Este habilita el 3er transistor, el de las centenas

```
trisb=0              ;convierte en salida todo el puerto B
display:
  portb=224+8        ;%11100000,activa el transistor de las unidades y presenta el 8
 PAUSE 5
  portb=208+7        ;%11010000,activa el transistor de las decenas y presenta el 7
 PAUSE 5
  portb=176+6        ;%10110000,activa el transistor de las centenas y presenta el 6
 PAUSE 5
  portb=112+5        ;%01110000,activa el transistor de los millares y presenta el 5
 PAUSE 5
 GOTO display        ; encierra en este lazo
END
```

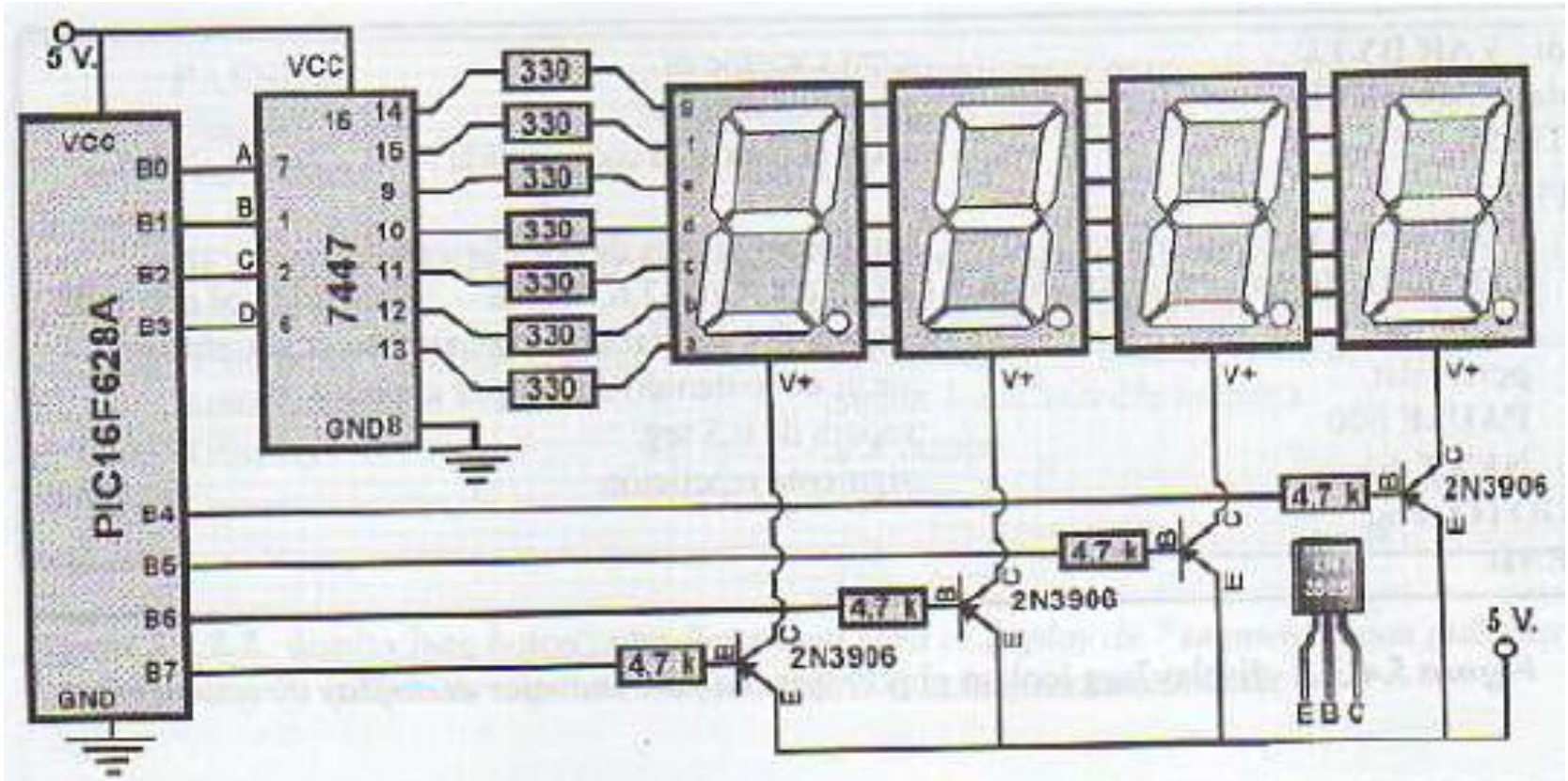


Figura 5.4.4.1. Configuración para manejar 4 displays de 7 segmentos con el CI. 7447.

MATERIALES.

-4 DISPLAYS ánodo común
-4 transistores 2N3906
-7 resistencias de 330Ω a ½ vatio, naranja-naranja-café
-4 resistencias de 4,7 KΩ a ½ vatio, amarillo-violeta-rojo

```
cmcon=7                 ;convierte todo el puerto A en digital
trisb=0                 ;convierte todos los pines del puerto B en salidas
trisa=0                 ;convierte todos los pines del puerto A en salidas
texto:
   porta=14             ;%1110 activa el display de la derecha
   portb=8              ;%0001000 forma la letra A
   PAUSE 5
   porta=13             ;%1101 activa el siguiente display
   portb=71             ;%1000111 forma la letra L
   PAUSE 5
   porta=11             ;%1011 activa el siguiente display
   portb=64             ;%1000000 forma la letra O
   PAUSE 5
   porta=7              ;%0111 activa el último display
   portb=9              ;%0001001 forma la letra H
   PAUSE 5
GOTO texto
END
```

5.5.1. MANEJO DE UN MÓDULO LCD.

Los módulos LCD (Display de Cristal Líquido), son utilizados para mostrar mensajes que indican al operario el estado de la maquina, o para dar instrucciones de manejo, mostrar valores, etc. El LCD permite la comunicación entre las máquinas y los humanos, este puede mostrar cualquier caracter ASCII, y consumen mucho menos que los displays de 7 segmentos, existen de varias presentaciones por ejemplo de 2 líneas por 8 caracteres, 2x16, 2x20, 4x20, 4x40, etc. sin backlight (14 pines) o con backlight (16 pines, iluminado de pantalla), el LCD más popular es el 2x16, 2 líneas de 16 caracteres cada una.

1 16

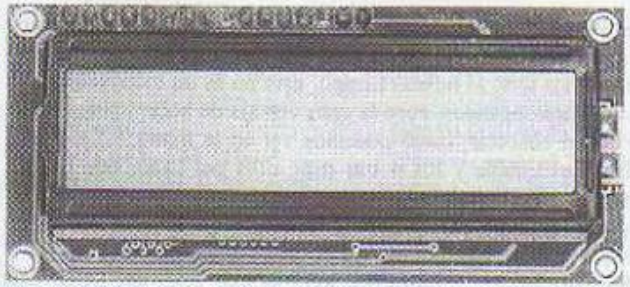


Figura 5.5.1.1. Fotografía de un LCD 2x16 con controlador Hitachi 44780 y Backlight en color amarillo

Pin	Simb.	Descripción
1	Vss	Tierra de alimentación GND
2	Vdd	Alimentación de +5V CC
3	Vo	Ajuste del contraste del cristal líquido (0 a +5V)
4	RS	Selección del registro control/datos RS=0 reg. control RS=1 reg. datos
5	R/W	Lectura/escritura en LCD R/W=0 escritura (Write) R/W=1 lectura (Read)
6	E	Habilitación E=0 módulo desconectado E=1 módulo conectado
7	D0	Bit menos significativo (bus de datos bidireccional)
8	D1	
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7	Bit más significativo (bus de datos bidireccional)
15	A	Alimentación del backlight +3,5 V o +5V CC (según especificación técnica)
16	K	Tierra GND del backlight

***Figura 5.5.1.2.** Función de cada pin del LCD.*

LA DECLARACIÓN LCDOUT. Sirve para mostrar items en una pantalla de cristal líquido, se utiliza escribiendo: **LCDOUT**, luego escribiendo $FE, y seguido por el comando a utilizar, el siguiente cuadro muestra los comandos más utilizados:

Comando	Operación
$FE, 1	Limpia el visor del LCD
$FE, 2	Vuelve al inicio (comienzo de la primera línea)
$FE, $0C	Apagar el cursor
$FE, $0E	Subrayado del cursor activo (—)
$FE, $0F	Parpadeo del cursor activo ( )
$FE, $10	Mover el cursor una posición a la izquierda
$FE, $14	Mover el cursor una posición a la derecha
$FE, $80	Mueve el cursor al comienzo de la primera línea
$FE, $C0	Mueve el cursor al comienzo de la segunda línea
$FE, $94	Mueve el cursor al comienzo de la tercera línea
$FE, $D4	Mueve el cursor al comienzo de la cuarta línea

Los LCD se puede conectar con el PIC con un bus de 4 u 8 bits, la diferencia está en el tiempo que se demora, pues la comunicación a 4 bits, primero envía los 4 bits más altos y luego los 4 bits más bajos, mientras que la de 8 bits envía todo al mismo tiempo, esto no es un inconveniente si consideramos que el LCD trabaja en microsegundos. Pero la gran ventaja de hacer conexión a 4 bits, son los pocos cables que se deben conectar, como podemos ver en la figura 5.5.1.4. sólo debemos conectar el bit de Registro, el Enable y los 4 bits más altos del LCD, con esto es suficiente para enviar los mensajes.

El compilador PBP soporta módulos LCD'S con controlador Hitachi 44780 o equivalentes y por defecto, asume que se conectó en el pin A4 el bit de Registro, en el pin B3 el bit Enable y en el puerto A empezando desde A0 hasta A3, los bits más altos del LCD. Esta configuración predefinida, se lo puede cambiar de acuerdo a la necesidad, como lo veremos más adelante.

MATERIALES.
-1 DISPLAY LCD 2x16 (preguntar si es compatible con los PIC'S)
-1 resistencia de 10Ω a ½ vatio, café-negro-negro
-1 potenciómetro de 10 KΩ.

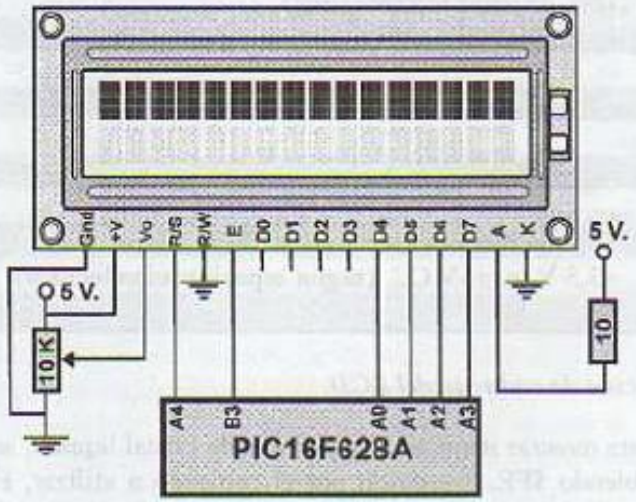


Figura 5.5.1.4. *Conexión de un LCD. a 4 bits predefinido por el compilador PBP, la resistencia de 10 Ω conectado a la alimentación del backlight, sirve para evitar altas temperaturas, noten además que el bit R/W se encuentra conectado a tierra, esto es porque la declaración* ***LCDOUT****, es de escritura únicamente.*

```
PAUSE 200                ;retardo para esperar que funcione el LCD
LCDOUT $FE, 1,"Hola"     ;limpiar pantalla y sacar el texto Hola
LCDOUT $FE, $C0,"microPIC"  ;pasar al comienzo de la segunda linea
                         ;y escribir microPIC
END                      ;fin de instrucciones
```

Figura 5.5.1.5. LCD Hola.pbp *Programa para presentar la palabra Hola microPIC.*

NOTA: algunos LCD'S no requieren de ningún PAUSE al inicio, pero existen otros modelos que necesitan unos pocos milisegundos para estar listos, por eso colocamos un **PAUSE** 200 al comienzo del programa.

Bien una vez visto el texto notaremos que las dos palabras están al lado izquierdo, si queremos que salgan centradas en nuestro LCD, tenemos 2 maneras de hacerlo, la primera es dando espacios antes de cada palabra ejemplo:

LCDOUT $FE, 1, " Hola" y **LCDOUT** $FE, $C0, " microPIC"

Lo cual es sencillo pero no es muy recomendable porque ocupa más espacio en el PIC, la segunda manera es asignando el lugar donde se quiere que aparezca cada palabra ejemplo:

```
LCDOUT $FE, 1                ;limpia la pantalla y coloca el cursor al comienzo
LCDOUT $FE, $86, "Hola"      ;pasa el cursor al 7ma casilla de la 1era Linea y escribe
LCDOUT $FE, $C4, "microPIC"  ;pasa a la casilla 5 de la 2da línea y escribe microPIC
```

Se debe entender que existe un cursor que aunque no lo vemos, pues este es el que indica donde aparecerá la siguiente letra, para poder entender haremos un ejercicio completo, así podrán aprender más del LCD y las funciones de cada uno de los comandos. Primero que nada haremos visible el cursor y luego pondremos PAUSES para poder ver el funcionamiento.

Se debe entender que existe un cursor que aunque no lo vemos, pues este es el que indica donde aparecerá la siguiente letra, para poder entender haremos un ejercicio completo, así podrán aprender más del LCD y las funciones de cada uno de los comandos. Primero que nada haremos visible el cursor y luego pondremos PAUSES para poder ver el funcionamiento.

```
x VAR BYTE                ;crea la variable x de un tamaño de 255
pepe CON $FE              ;asigna el nombre de pepe a la constante $FE

  PAUSE 2000
LCDOUT pepe, 1            ;limpia el visor del LCD
  PAUSE 2000
LCDOUT pepe,$0F           ;muestra el cursor en casilla negra
  PAUSE 2000
LCDOUT pepe,$0E           ;subraya el cursor
  PAUSE 2000
LCDOUT pepe,$14           ;desplaza el cursor una casilla a la derecha
  PAUSE 2000
LCDOUT, "MIKRO"           ;escribe mikro, desde donde se encuentre el cursor
  PAUSE 2000
 FOR x = 1 TO 3           ; repite 3 veces las siguientes instrucciones
      LCDOUT pepe,$10     ; desplaza el cursor una casilla a la izquierda
      PAUSE 1000
  NEXT
LCDOUT, 67                ; envia el caracter ASCII "C" para corregir MICRO
  PAUSE 2000
                                                          continua ....
LCDOUT pepe,$C0+12,"PIC"          ;escribe en la segunda linea casillero 13
                                  ;esto equivale a $CC o 204
PAUSE 2000
LCDOUT pepe,2,"1"                 ;vuelve al inicio de la 1era fila y escribe 1
END
```

5.5.2. PRESENTACIÓN DE CARACTER POR CARACTER EN LCD.

En las prácticas anteriores se presentaron mensajes completos en un instante, en esta nueva práctica incluimos la declaración **LOOKUP**, que nos servirá para enviar caracter por caracter con un intervalo de 400mls, dando como resultado un efecto especial en la visualización. Como conexión para esta práctica utilizaremos la misma de la figura 5.5.1.4.

```
PAUSE 200                       ;retardo para esperar que funcione el LCD
x VAR BYTE                      ;crear la variable x de 255
abc VAR BYTE                    ;crear la variable abc de 255
ini:
        LCDOUT $FE,1            ;limpiar pantalla
        FOR x = 0 TO 15         ;repetir 16 veces
LOOKUP x,["Microcontrolador"],abc       ;tomar caracter por caracter y guardar en abc
        LCDOUT, abc             ;sacar en LCD el contenido de abc
        PAUSE 400               ;esperar 400 mls
        NEXT                    ;siguiente repetición
        PAUSE 2000
GOTO ini
END
```

Figura 5.5.2.1. LCD especial.pbp *Programa para mostrar uno por uno cada caracter.*

5.5.4. CONTADOR DE PULSOS CON LCD.

Este proyecto, consiste en contar el número de pulsos que ingresan por un pin en un determinado período, este a su vez se visualiza en un LCD, si la cantidad de este supera a los 120 pulsos por segundo es decir 120 HZ, se encenderá una alarma que en este caso será un led rojo, y si la cantidad de pulsos baja a menos de 100 HZ, este encenderá un led verde, si la frecuencia se mantiene entre estos 2 rangos, no se encenderá ningún led.

Este proyecto tiene muchas aplicaciones como por ejemplo para un regulador de voltaje en el que a más de indicarnos el voltaje de salida podría además indicarnos la frecuencia.

Para esta práctica utilizaremos un CI. 555 que nos ayuda a generar un tren de pulsos variable, el cual lo conectamos al PIC para su posterior conteo.

LA DECLARACIÓN COUNT. Sirve para contar el número de pulsos que ingresan por un pin en un determinado tiempo, este a su vez lo guarda en una variable para su posterior procesamiento, la manera de utilizarlo es la siguiente:

```
COUNT portb.0, 1000, abc
```

El cual se interpreta así: cuenta pulsos a través del puerto B0 en un período de 1000 milisegundos y lo guarda en la variable previamente creada llamada abc, el período podemos variarlo de 1 a 65535.

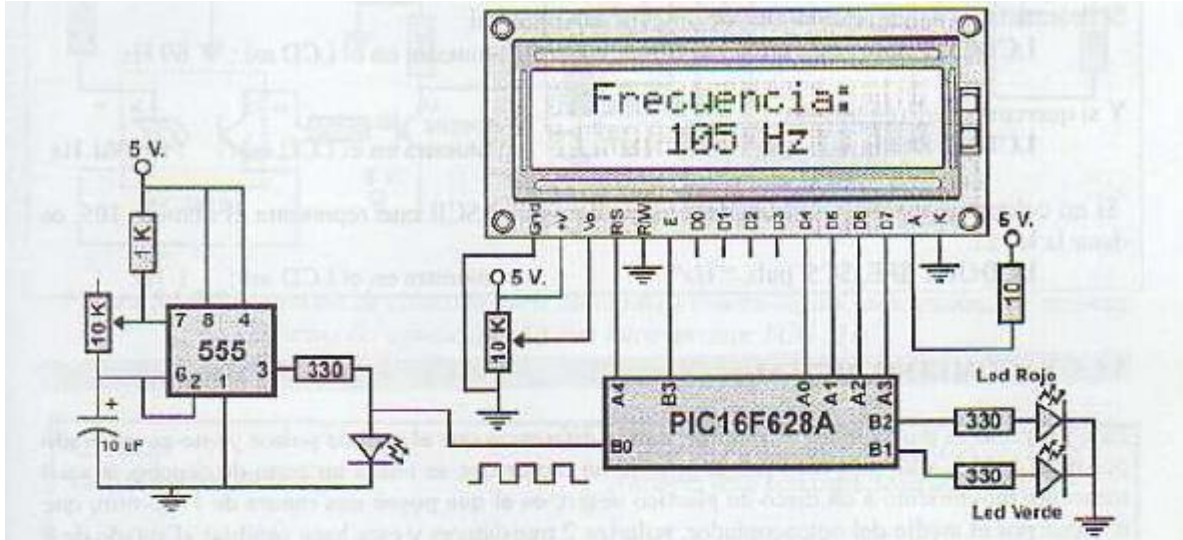


Figura 5.5.4.1. Esquema de conexión para el frecuencímetro con LCD.

```
CMCON = 7                          ;convierte en digitales el puerto A
LEDRED VAR PORTB.2                 ;el puerto B.2 se llamará LEDRED
LEDVER VAR PORTB.1                 ;el puerto B.1 se llamará LEDVER
puls VAR WORD                      ;variable puls con capacidad de 0 a 65535
prog:
  COUNT portb.0,1000,puls          ;contar pulsos en el puerto B.0
  LCDOUT $FE, 1                    ;limpiar pantalla LCD
  LCDOUT $FE,$82,"Frecuencia:"     ;cursor en el casillero 3 de la 1era linea
  LCDOUT $FE,$C5, DEC puls," Hz"   ;en el comienzo 2da linea casilla 6 sacar
                                   ;el valor decimal de la variable puls
  IF puls>120 THEN alarmah         ;si la variable excede de 120
  IF puls<100 THEN alarmal         ;si la variable baja de 100
  LOW LEDRED : LOW LEDVER          ;apagar los 2 leds
GOTO prog
alarmah: HIGH LEDRED               ;encender led rojo
         LOW LEDVER                ;apagar led verde
  GOTO prog
alarmal: HIGH LEDVER               ;encender led verde
         LOW LEDRED                ;apagar led rojo
  GOTO prog
```

Figura 5.5.4.2. frecuenciaLCD.pbp *Programa para un contador de pulsos.*

LA PALABRA DEC. Sirve para mostrar el número de la variable en decimal, también se lo puede representar por el signo (#), además existe las palabras **BIN** y **HEX**, el siguiente es un ejemplo de cómo mostraría el LCD si puls = 105:

```
LCDOUT $FE, $C5, DEC puls, " Hz"        ; muestra en el LCD así :   105 Hz
```

También se lo puede utilizar el signo #, que equivale a **DEC**.

```
LCDOUT $FE, $C5, # puls, " Hz"          ; muestra en el LCD así :   105 Hz
```

Si deseamos ver la variable en hexadecimal pondríamos así:

```
LCDOUT $FE, $C5, HEX puls, " Hz"        ; muestra en el LCD así :   69 Hz
```

Y si queremos verlo en binario:

```
LCDOUT SFE, $C5,  BIN puls, " Hz"       ; muestra en el LCD así :   1101001 Hz
```

Si no colocamos ninguna instrucción nos mostraría el ASCII que representa el número 105, es decir la letra i.

```
LCDOUT $FE, $C5, puls, " Hz"            ; muestra en el LCD así :   i Hz
```