Segunda Práctica

Diseño de Sistemas Basados en Microprocesador

David Morales Sáez
Alejandro Sánchez Medina

Índice

Objetivos

Nuestro programa va a hacer lo siguiente. En primer mostrará en la pantalla un pequeño mensaje con un menú de opciones.

1 texto A izqda. 2 texto A derecha
3 texto B izqda 4 texto B derecha

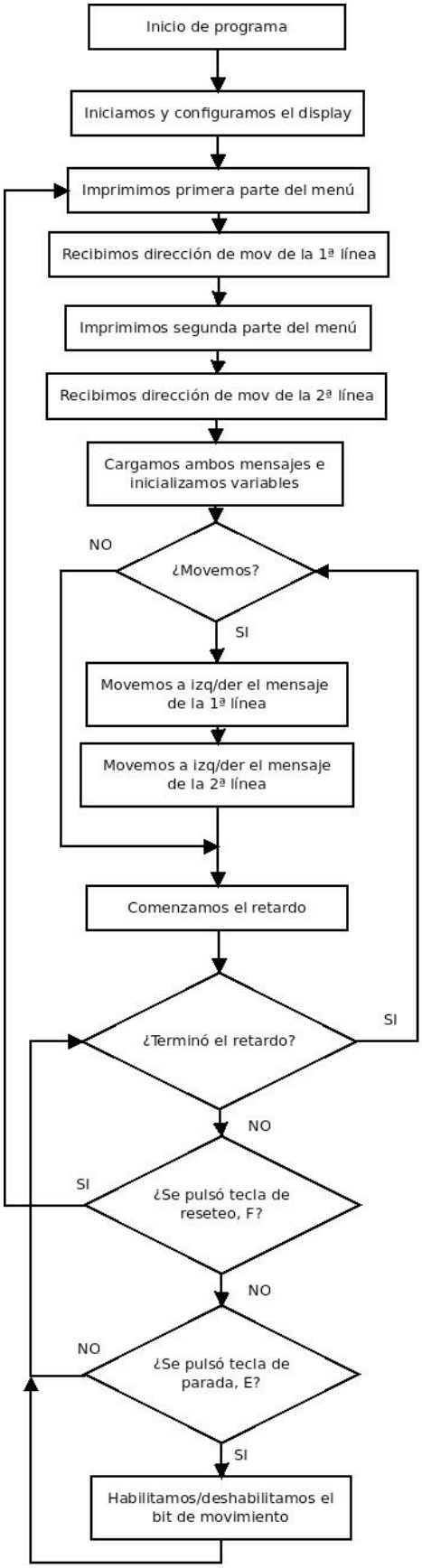
Lo que se está pidiendo es que se muestre el texto A en la línea superior. Y el B en la linea inferior. Izquierda o derecha se refiere a que el texto se irá desplazándose carácter a carácter hacia la izquierda o hacia la derecha hasta desaparecer y volver a aparecer por el otro lado en un movimiento continuo. El usuario podrá entrar por teclado los siguientes valores: 10,20,03,04,13,14,23,24. Una vez que se introduzca la selección los textos aparecerán en pantalla y comenzarán a moverse. Durante el funcionamiento el usuario podrá pulsar la F en el teclado para detener el movimiento de los textos. Una nueva pulsación de la F hará que los textos continúen su movimiento. Por último, la pulsación de la letra E hará que se reinicie todo el proceso y vuelva a mostrarse la pantalla del menu.

Texto A: comenzará en la dirección 0 de la EPROM. El primer byte dice cuantos caracteres contiene el texto. A partir del segundo byte está la ristra de caracteres.

Texto B: comenzará en la dirección 32d de la EPROM. El mismo formato.

Limitaremos el tamaño de cada texto a 15 caracteres. Si comprobamos que alguna de los textos tiene un tamaño mayor daremos un error: “Formato del texto no válido”

Algoritmo


Inicio de programa
Iniciamos y configuramos el display
Imprimimos primera parte del menú
Recibimos dirección de mov de la 1ª línea
Imprimimos segunda parte del menú
Recibimos dirección de mov de la 2ª línea
Cargamos ambos mensajes e inicializamos variables
NO
¿Movemos?
SI
Movemos a izq/der el mensaje de la 1ª línea
Movemos a izq/der el mensaje de la 2ª línea
Comenzamos el retardo
¿Terminó el retardo?
SI
NO
SI
¿Se pulsó tecla de reseteo, F?
NO
NO
¿Se pulsó tecla de parada, E?
SI
Habilitamos/deshabilitamos el bit de movimiento

Como aclaración, nuestro algoritmo mueve el cursor, en lugar de la pantalla, ya que consideramos que es más rápido, fluido y más sencillo de implementar en comparación con la modificación de la ristra o de la pantalla.

Código fuente

EEPROM.asm

```
                #include "p16f84a.inc"

GLOBAL SAVE_E, LOAD_E, E_ADRESS

lcd_data        udata_ovr
Lcd_Temp_1      res     1       ;Para la espera del LCD.
Lcd_Temp_2      res     1       ;Para la espera del LCD.
E_ADRESS        res 1

EEPROM          code
; Guardar en la posicion E_ADRESS y lo que este en W
SAVE_E
        MOVWF       EEDATA
        MOVFW       E_ADRESS
        MOVWF       EEADR
        BSF         STATUS, RP0
        BSF         EECON1,WREN
        BCF         STATUS, RP0
        BCF         INTCON,GIE
        BSF         STATUS, RP0
        MOVLW       0x55
        MOVWF       EECON2
        MOVLW       0xAA
        MOVWF       EECON2
        BSF         EECON1, WR
        BCF         EECON1, WREN
        BCF         STATUS, RP0
        BSF         STATUS, RP0
espera1
        BTFSC       EECON1, WR
        GOTO        espera1
        BCF         EECON1, EEIF    ; Apagar el flag de interrupciones
        BCF         STATUS, RP0
        RETURN
; Leera la posicion E_ADRESS y guardara lo leido en W
LOAD_E
        MOVFW       E_ADRESS
        MOVWF       EEADR
        BSF         STATUS, RP0
        BSF         EECON1,RD
        BCF         STATUS, RP0
        MOVFW       EEDATA
        RETURN
        end
```

P2.asm

```
                List            p=16f84
                include "P16F84.INC"
                include "lcd.inc"
                include "EEPROM.inc"
                include "tecla.inc"

__CONFIG _WDT_OFF & _XT_OSC

Iter            equ     0x0d
Iter2           equ     0x0e

        udata
Aux             res     1
Aux2            res     1

Band            res     1       ; Band0 aka B[0]
                                ; B[0] = 1 : Se ha pulsado una tecla y se esta esperando a 80h
                                ; B[1] = 1 : Tecla leida
                                ; B[2] = 1 : Nothing
                                ; B[3] = 0 : El mensaje 1 ira hacia la derecha
                                ; B[4] = 0 : El mensaje 2 ira hacia la derecha
                                ; B[5] = 1 : Se para el movimiento de los textos
                                ; B[7] = 1 : El display est· parado
TPulsada        res     1

PosIni          res     1
PosIni2 res     1
Pos             res     1

Den             res     1
DenMen1         res     1
DenMen2         res     1
TamMen1         res     1
TamMen2         res     1

TamDisp         res     1
TamMen          res     1
FinMen res      1

Linea           res     1

Rts             res     1
Tmp             res     1
Prm             res     1

                org     0x00
                goto    Inicio

                org     0x05

;Retardo        movlw           05h
                movwf           Rts

;Retard movlw           0ffh
                movwf           Tmp

;Sigu2          decfsz          Prm, 1
                goto            CompTecla
```

```
                goto            final

;continua       decfsz          Tmp, 1
                goto            continua
                goto            Sigu2

;final          return


CompTecla       CALL            LeeTeclado
                BTFSS Band, 1                   ; Comprobamos tecla v·lida
                goto            continua
                MOVLW           h'F'            ; Si pulsa F, reseteamos
                SUBWF           TPulsada, W
                BTFSC STATUS, Z
                goto            Inicio
                MOVLW           h'E'
                                                ; Si pulsa E, "activamos/desactivamos" display
                SUBWF           TPulsada, W
                BTFSS STATUS, Z
                goto            continua
                                                ; No hemos recibido ni E ni F
                BTFSS Band, 7
                                                ; Si el display est· "desactivado"
                goto            ActivaB7

                BCF             Band, 7
                                                ; "Desactivamos" display
                goto            continua

ActivaB7        BSF             Band, 7
                                                ; "Activamos" display
                goto            continua

Retardomovlw            05h
                movwf           Rts

Retard2         decfsz          Rts, 1
                goto            Retard
                goto            final

Retard  movlw           0ffh
                movwf           Tm

Sigu2           decfsz          Prm, 1
                goto            CompTecla
                goto            Retard2

continua        decfsz          Tmp, 1
                goto            continua
                goto            Retard

final           return

Modulo          SUBWF           Aux, F
                BTFSC STATUS, C
                goto            Modulo
```

```
                ADDWF       Aux, W
                Return

MsgIni1                                 ; Primer mensaje inicial
                MOVLW       'M'
                CALL        LCD_DATO
                MOVLW       'S'
                CALL        LCD_DATO
                MOVLW       'G'
                CALL        LCD_DATO
                MOVLW       '1'
                CALL        LCD_DATO
                MOVLW       ':'
                CALL        LCD_DATO
                MOVLW       d'64'
                                        ; Desplazar el cursor a la segunda linea
                MOVWF       Aux
                MOVLW       b'10000000'
                                        ; Mover cursor
                ADDWF       Aux,W
                CALL        LCD_REG
                MOVLW       '1'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       'I'
                CALL        LCD_DATO
                MOVLW       'Z'
                CALL        LCD_DATO
                MOVLW       'Q'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       '2'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       'D'
                CALL        LCD_DATO
                MOVLW       'E'
                CALL        LCD_DATO
                MOVLW       'R'
                CALL        LCD_DATO
                RETURN

MsgIni2                                 ; Segundo mensaje inicial
                MOVLW       'M'
                CALL        LCD_DATO
                MOVLW       'S'
                CALL        LCD_DATO
                MOVLW       'G'
                CALL        LCD_DATO
                MOVLW       '2'
                CALL        LCD_DATO
                MOVLW       ':'
                CALL        LCD_DATO
                MOVLW       d'64'
                                        ; Desplazar cursor a la segunda line
```

```
                MOVWF       Aux
                MOVLW       b'10000000'     ; Mover cursor
                ADDWF       Aux,W
                CALL        LCD_REG
                MOVLW       '4'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       'I'
                CALL        LCD_DATO
                MOVLW       'Z'
                CALL        LCD_DATO
                 MOVLW      'Q'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       '5'
                CALL        LCD_DATO
                MOVLW       ' '
                CALL        LCD_DATO
                MOVLW       'D'
                CALL        LCD_DATO
                MOVLW       'E'
                CALL        LCD_DATO
                MOVLW       'R'
                CALL        LCD_DATO
                RETURN

LeeTeclado
                BCF         Band, 1         ; Aún no se ha leÌdo tecla v·lida
                CALL        Key_Scan        ; Escanear      teclado
                BTFSC Band,0                ; Si no Band[0] = 0 se sigue normal
                GOTO  Leer1                 ; Si Band[0] = 1 hay que esperar hasta que se
suelte la tecla del todo para evitar rebotes
                MOVLW       h'80'           ; Si no se toco tecla
                SUBWF       Tecla,W
                BTFSC STATUS, Z
                RETURN                      ; salimos
                CALL        Cods_Tecla      ; Se ha pulsado una tecla
                MOVFW       POS
                MOVWF       TPulsada        ; Se guarda la tecla pulsada
                BSF         Band,0          ; Y se esperara a que se suelte la tecla
                RETURN

Leer1
                MOVLW       h'80'           ; Si aun no se ha soltado la tecla
                SUBWF       Tecla,W
                BTFSS STATUS, Z
                RETURN                      ; salimos
                BCF         Band,0          ; Se estabilizo la seÒal por lo que se apaga el
Band[0]
                BSF         Band, 1         ; Indicamos que en TPulsada hay una tecla
v·lida
                RETURN

Inicio
                call        LCD_INI
                call        DISPLAY_CONF
```

```
                MOVLW       d'16'           ;Inicializamos el tamaÒo del display
                MOVWF       TamDisp

                MOVLW       30
                MOVWF       Aux

                movlw       0
                movf        Iter,1
                call        MsgIni1

                                            ; Imprimimos el primer mensaje y esperamos la
                                        respuesta

                CLRF        Band

LCom1                                       ; Lectura del primer comando
                CALL        LeeTeclado
                BTFSS Band, 1
                goto        LCom1
                MOVLW       d'1'            ; Si pulsa 1, movemos a la iz
                SUBWF       TPulsada, W
                BTFSC STATUS, Z
                goto        Com1Iz
                MOVLW       d'2'            ; Si pulsa 2, movemos a la der
                SUBWF       TPulsada, W
                BTFSS STATUS, Z
                goto        LCom1           ; No hemos recibido ni 1 ni 2
                goto        aLCom2

Com1Iz
                BSF         Band, 3

aLCom2
                MOVLW       b'00000001'     ; Clear display
                CALL        LCD_REG
                call        MsgIni2

                                            ; Imprimimos el segundo mensaje y esperamos
                                        la respuesta

LCom2                                       ; Lectura del segundo comando
                CALL        LeeTeclado

                BTFSS Band, 1
                goto        LCom2
                MOVLW       d'4'            ; Si pulsa 4, movemos a la iz
                SUBWF        TPulsada, W
                BTFSC STATUS, Z
                goto        Com2Iz
                MOVLW       d'5'            ; Si pulsa 5, movemos a la der
                SUBWF       TPulsada, W
                BTFSS STATUS, Z
                goto        LCom2           ; No hemos recibido ni

                goto        FinLeer

Com2Iz
```

```
                BSF             Band, 4

FinLeer                                         ; Segun los comandos pulsados se iniciara una
                                                ;rutina u otra para cada linea
                MOVLW           b'00000001'     ; Clear display
                CALL            LCD_REG

                ;   Bucle principal
                CALL            IniMen1
                CALL            IniMen2

                CLRF            PosIni          ; La posiciÛn inicial en ambas lineas
                CLRF    PosIni2
Yeas
                                                ; B[7] = 1 : El display est· parado
                BTFSC Band, 7
                goto            Ret

                                                ; B[3] = 0 : El mensaje 1 ira hacia la derecha
                                                ; B[4] = 0 : El mensaje 2 ira hacia la derecha
                BTFSS Band, 3
                goto            B1
                CALL            MoverIzq1
                goto            B2
B1              CALL            MoverDer1
B2              BTFSS Band, 4
                goto            B3
                CALL            MoverIzq2
                goto            Ret
B3              CALL            MoverDer2

Ret             CALL            Retardo
                goto            Yeas

IniMen1                                         ; Rutina para que el mensaje de la primera linea
                                                ; vaya hacia la derecha
                MOVLW           d'0'            ; Se carga de la EEPROM el tamaÒo de la
ristra
                MOVWF           E_ADRESS
                MOVLW           d'18'
                MOVWF           TamMen1         ; Guardamos el tamaÒo del mensaje1

                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'M'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'E'
                CALL SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'N'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'S'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'A'
                CALL            SAVE_E
```

```
                INCF            E_ADRESS, F
                MOVLW           'J'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'E'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           '_'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'M'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'a'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           's'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           '_'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'L'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'a'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'r'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'g'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           'o'
                CALL            SAVE_E
                INCF            E_ADRESS, F
                MOVLW           '_'             ;18 car
                CALL SAVE_E

                                                ; Cargamos las variables  necesarias para el
mensaje1
                MOVF  TamMen1, W
                MOVWF           DenMen1
                INCF            DenMen1, F

                SUBWF           TamDisp, W
                BTFSS STATUS, C
                                                ; Si TamDisp < TamMen1 dejamos como
                                                denominador TamMen1 +1
                goto Sal1
                MOVF            TamDisp, W
                                                ; Si no el denominador es TamDisp
                MOVWF           DenMen1
                INCF            DenMen1, F

Sal1            RETURN
```

```
IniMen2                                 ; Rutina para que el mensaje de la primera linea
                                        ; vaya hacia la derecha
            MOVLW       d'32'           ; Se carga de la EEPROM el tamaÒo de la
ristra
            MOVWF       E_ADRESS
            MOVLW       d'13'
            MOVWF       TamMen2
                                        ; Guardamos el tamaÒo del mensaje1
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'M'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'E'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'N'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'S'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'A'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'J'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'E'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       ' '
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'C'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'o'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'r'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       't'
            CALL        SAVE_E
            INCF        E_ADRESS, F
            MOVLW       'o'
            CALL        SAVE_E      ;13 car

                                        ; Cargamos las variables  necesarias para la
mover el mensaje2
            MOVF  TamMen2, W
            MOVWF       DenMen2
            INCF        DenMen2, F
            SUBWF       TamDisp, W
            BTFSS STATUS, C
                                        ; Si TamDisp < TamMen1 dejamos como
                                        denominador TamMen1 +1
```

```
                goto            Sal2
                MOVF            TamDisp, W
                                        ; Si no el denominador es TamDisp
                MOVWF           DenMen2
                INCF            DenMen2, F

Sal2            RETURN

; Variables empleadas:
; TamDisp       --> Tamano del display
; TamMen        --> Tamano del mensaje+1 (+1 para introducir un espacio "flotante")
; FinMen        --> Posicion final del mensaje
; E_ADRESS --> PosiciÛn a leer en la EEPROM
; Den            --> Denominador para el modulo, ser· el mayor valor entre TamDisp y TamMen
; Pos           --> Posicion en la que comenzamos a imprimir
; Linea  --> Linea en la que se imprimir· la frase

çImpMsg                                 ; Rutina que calcula los caracteres visibles del
mensaje
                                        ; y los imprime.
        INCF            E_ADRESS, F
        INCF            FinMen, W
                                        ; Sumamos 1 porque tiene que llegar a FinMem
        SUBWF           E_ADRESS, W
        BTFSC STATUS, Z
                                        ; mientras (E_ADRESS < FinMen+1) entonces
        return

        MOVF  E_ADRESS, W
                                        ; Calculamos la posicion relativa del
                                    E_ADRESS
        MOVWF           Aux             ; en la frase (E_ADRESS mod 32).
        MOVLW           d'32'           ; 32 es el ˙ltimo byte relativo dedicado para
        CALL            Modulo          ; la informaciÛn de cada mensaje
        MOVWF           Aux

        DECF   Aux, W
                                        ;No hay que decrementar;
                                    Pos+Pos_Rel_E_ADRESS-1
        ADDWF           Pos, W

        MOVWF           Aux
        MOVF  Den, W
        CALL            Modulo          ; W = (PosIni+E_ADRESS-1) mod Den
        MOVWF           Aux                     ; Guardamos el modulo

        SUBWF           TamDisp, W      ; TamDisp-W
        BTFSC STATUS, Z                 ; No imprimimos si W >= TamDisp
        goto            ImpMsg
        BTFSS STATUS, C
                                        ; En una resta el carry se activa cuando no es
                                    negativo
        goto            ImpMsg

        MOVF            Aux, W          ; Recuperamos el modulo
        ADDWF           Linea, W
                ; Posicionamos el cursor en la posicion que nos da el modulo
        CALL            LCD_REG
```

```
        CALL    LOAD_E          ; Se lee un caracter
        CALL    LCD_DATO        ; Se imprime

        goto    ImpMsg

MoverIzq1
        MOVLW   b'00000001'     ; Clear display
        CALL    LCD_REG

        MOVF  TamMen1, W
        MOVWF   TamMen          ; Guardamos el tamaÒo del mensaje

        MOVWF   FinMen          ; El mensaje 1 acaba en el TamMen+1
        MOVF  DenMen1, W
        MOVWF   Den
                                ; Guardamos el mensaje correspondiente
        MOVF  PosIni, W         ; Pasamos la posicion inicial
        MOVWF   Pos

        MOVLW   b'10000000'
                                ; Indicamos que imprimiremos en la linea 1
        MOVWF   Linea

        CLRF    E_ADRESS        ; Reseteamos E_ADRESS
        CALL    ImpMsg
        DECF   PosIni, F
        MOVLW   h'FF'
        SUBWF   PosIni, W
                                ; En el caso que Pos haya llegado a -1, lo
                                reseteamos
        BTFSS STATUS, Z
        RETURN
        MOVF  DenMen1, W
        MOVWF   PosIni
        DECF   PosIni, F

        RETURN

MoverDer1
        MOVLW    b'00000001'    ; Clear display
        CALL    LCD_REG
        MOVLW   b'10000000'     ; Cambiamos linea
        CALL    LCD_REG

        MOVF  TamMen1, W
        MOVWF   TamMen          ; Guardamos el tamaÒo del mensaje

        MOVWF   FinMen          ; El mensaje 1 acaba en el TamMen+1
        MOVF  DenMen1, W
        MOVWF   Den
                                ; Guardamos el mensaje correspondiente
        MOVF  PosIni, W         ; Pasamos la posicion inicial
        MOVWF   Pos

        CLRF    E_ADRESS        ; Reseteamos E_ADRESS
        MOVLW   b'10000000'
                                ; Indicamos que imprimiremos en la linea 1
        MOVWF   Linea
```

```
        CALL    ImpMsg

        INCF    PosIni, F
        MOVF  DenMen1, W
        SUBWF   PosIni, W
                                ; En el caso que Pos haya llegado a 16, lo
                reseteamos
        BTFSC STATUS, Z
        clrf    PosIni

        RETURN

MoverIzq2
        MOVLW   b'11000000'     ; Cambiamos linea
        CALL    LCD_REG
        MOVF  TamMen2, W
        MOVWF   TamMen          ; Guardamos el tamaÒo del mensaje

        MOVF  DenMen2, W
        MOVWF   Den
                                ; Guardamos el mensaje correspondiente
        MOVF  PosIni2, W        ; Pasamos la posicion inicial
        MOVWF   Pos
        MOVLW   b'11000000'
                                ; Indicamos que imprimiremos en la linea 1
        MOVWF   Linea
        MOVLW   d'32'
        MOVWF   E_ADRESS

        ADDWF   TamMen, W       ; El mensaje 1 acaba en el 32+TamMen
        MOVWF   FinMen

        CALL    ImpMsg

        DECF   PosIni2, F
        MOVLW   h'FF'
        SUBWF   PosIni2, W
                                ; En el caso que Pos haya llegado a -1, lo
                        reseteamos
        BTFSS STATUS, Z
        RETURN
        MOVF  DenMen2, W
        MOVWF   PosIni2
        DECF   PosIni2, F
        RETURN

MoverDer2
        MOVF  TamMen2, W
        MOVWF   TamMen          ; Guardamos el tamaÒo del mensaje

        MOVF  DenMen2, W
        MOVWF   Den             ; Guardamos el mensaje correspondiente
        MOVF  PosIni2, W        ; Pasamos la posicion inicial
        MOVWF   Pos

        MOVLW   b'11000000'
                                ; Indicamos que imprimiremos en la linea 1
```

```
        MOVWF   Linea

        MOVLW   d'32'
        MOVWF   E_ADRESS

        ADDWF   TamMen, W       ; El mensaje 1 acaba en el 32+TamMen
        MOVWF   FinMen

        CALL    ImpMsg

        INCF    PosIni2, F
        MOVF  DenMen2, W
        SUBWF   PosIni2, W
                                ; En el caso que Pos haya llegado a 16, lo reseteamos
        BTFSC STATUS, Z
        clrf    PosIni2
        RETURN

Aqui
        goto    Aqui

        end
```