```
;************************************************
;Programa que muestra los números del 9 al 0
;de manera descendente en una matriz de 5x7
;leds mediante multiplexación. Los ánodos (filas)
;están conectados al puerto B (RB0/RB6) mientras
;que los cátodos (columnas) al puerto A (RA0/RA4).
;La multiplexación se realiza recorriendo un cero
;por el puerto A y mostrando por el puerto B el
;valor de la columna deseada.
;
;Fecha: 05.01.08    Autor: Jorge A. Bojórquez
;http://micropic.wordpress.com
;************************************************

    list        p=16f628a
    include     p16f628a.inc
    __config    0x3f18

    cblock  0x20
    w_temp
    s_temp
    p_temp
    num1
    num2
    num3
    num4
    num5
    NUMERO
    d1
    d2
    d3
    endc

    org     0x00
    goto    INICIO

    org     0x04
ISR
    movwf   w_temp      ; Guarda W
    swapf   STATUS,W    ; Swap STATUS en W
    movwf   s_temp      ; Guarda STATUS
    movfw   PCLATH      ; Mueve PCLATH a W
    movwf   p_temp      ; Guarda PCLATH

    btfss   PIR1,TMR2IF ; Checa si TMR2 interrumpio
    goto    SAL_ISR     ; No, sale de la ISR
    bcf     PIR1,TMR2IF ; Si, borra bandera de interrupcion

CHECKCOL
```

```
    btfss   PORTA,4
    goto    UNO1
    btfss   PORTA,0
    goto    UNO2
    btfss   PORTA,1
    goto    UNO3
    btfss   PORTA,2
    goto    UNO4
    btfss   PORTA,3
    goto    UNO5

UNO1
    movlw   0xFF        ;
    movwf   PORTA       ; Apaga matriz
    movf    num1,w
    movwf   PORTB       ;load columns
    movlw   b'11111110'
    movwf   PORTA       ; Enciende columna 1
    goto    SAL_ISR

UNO2
    movlw   0xFF        ;
    movwf   PORTA       ; Apaga matriz
    movf    num2,w
    movwf   PORTB       ;load columns
    movlw   b'11111101'
    movwf   PORTA       ; Enciende columna 2
    goto    SAL_ISR

UNO3
    movlw   0xFF        ;
    movwf   PORTA       ; Apaga matriz
    movf    num3,w
    movwf   PORTB       ;load columns
    movlw   b'11111011'
    movwf   PORTA       ; Enciende columna 3
    goto    SAL_ISR

UNO4
    movlw   0xFF        ;
    movwf   PORTA       ; Apaga matriz
    movf    num4,w
    movwf   PORTB       ;load columns
    movlw   b'11110111'
    movwf   PORTA       ; Enciende columna 4
    goto    SAL_ISR

UNO5
    movlw   0xFF        ;
```

```
    movwf   PORTA       ; Apaga matriz
    movf    num5,w
    movwf   PORTB       ;load columns
    movlw   b'11101111'
    movwf   PORTA       ; Enciende columna 5
    goto    SAL_ISR

SAL_ISR
    movfw   p_temp
    movwf   PCLATH      ; Recupera PCLATH
    swapf   s_temp,W
    movwf   STATUS      ; Recupera STATUS
    swapf   w_temp,F
    swapf   w_temp,W    ; Recupera W
    retfie

INICIO
    movlw   0x07        ;
    movwf   CMCON       ; Deshabilita comparadores analogos
    bsf     STATUS,RP0  ; Cambio al banco 1
    clrf    TRISA       ; Puerto A como salida
    clrf    TRISB       ; Puerto B como salida
    bcf     STATUS,RP0  ; Cambio al banco 0
    clrf    PORTA       ; Apaga todos los LEDs
    clrf    PORTB       ; Limpia columna

    clrf    NUMERO

;*************************
;Configuración del Timer 2
;*************************
    movlw   b'00010110' ; Postscaler /4, prescaler /16, TMR2 ON
    movwf   T2CON
    bsf     STATUS,RP0  ; Cambio al banco 1
    movlw   .10         ; Carga valor del comparador
    movwf   PR2
;*************************
;Configuración de interrupciones
;*************************
    bsf     PIE1,TMR2IE ; Habilita interrupcion TMR2
    bcf     STATUS,RP0  ; Cambio al banco 0
    bsf INTCON,PEIE     ; Habilita interrupciones de perifericos
    bsf INTCON,GIE      ; Habilita interrupciones globales

    movlw   b'11101111' ;
    movwf   PORTA       ; Precarga fila

;*******************
;Programa principal
```

```
;*******************
CARGA_INICIAL
    movlw   .9
    movwf   NUMERO
CICLO
    call    DECOD_TIEMPO
    call    Delay1Sec
    decfsz  NUMERO,f
    goto    CICLO
    call    DECOD_TIEMPO
    call    Delay1Sec
    goto    CARGA_INICIAL


;*******************
;Subrutinas
;*******************
DECOD_TIEMPO
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .9          ; Checa si NUMERO es 9
    btfsc   STATUS,Z    ; ¿NUMERO = 9?
    call    NUEVE
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .8          ; Checa si NUMERO es 8
    btfsc   STATUS,Z    ; ¿NUMERO = 8?
    call    OCHO
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .7          ; Checa si NUMERO es 7
    btfsc   STATUS,Z    ; ¿NUMERO = 7?
    call    SIETE
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .6          ; Checa si NUMERO es 6
    btfsc   STATUS,Z    ; ¿NUMERO = 6?
    call    SEIS
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .5          ; Checa si NUMERO es 5
    btfsc   STATUS,Z    ; ¿NUMERO = 5?
    call    CINCO
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .4          ; Checa si NUMERO es 4
    btfsc   STATUS,Z    ; ¿NUMERO = 4?
    call    CUATRO
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .3          ; Checa si NUMERO es 3
    btfsc   STATUS,Z    ; ¿NUMERO = 3?
    call    TRES
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .2          ; Checa si NUMERO es 2
    btfsc   STATUS,Z    ; ¿NUMERO = 2?
```

```
    call    DOS
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .1          ; Checa si NUMERO es 1
    btfsc   STATUS,Z    ; ¿NUMERO = 1?
    call    UNO
    movf    NUMERO,w    ; Mueve NUMERO a W
    sublw   .0          ; Checa si NUMERO es 0
    btfsc   STATUS,Z    ; ¿NUMERO = 0?
    call    CERO
    return

NUEVE
    movlw   b'00000110'
    movwf   num1
    movlw   b'01001001'
    movwf   num2
    movlw   b'01001001'
    movwf   num3
    movlw   b'00101001'
    movwf   num4
    movlw   b'00011110'
    movwf   num5
    return

OCHO
    movlw   b'00110110'
    movwf   num1
    movlw   b'01001001'
    movwf   num2
    movlw   b'01001001'
    movwf   num3
    movlw   b'01001001'
    movwf   num4
    movlw   b'00110110'
    movwf   num5
    return

SIETE
    movlw   b'00000011'
    movwf   num1
    movlw   b'00000001'
    movwf   num2
    movlw   b'01110001'
    movwf   num3
    movlw   b'00001001'
    movwf   num4
    movlw   b'00000111'
    movwf   num5
    return
```

```
SEIS
    movlw   b'00111100'
    movwf   num1
    movlw   b'01001010'
    movwf   num2
    movlw   b'01001001'
    movwf   num3
    movlw   b'01001001'
    movwf   num4
    movlw   b'00110000'
    movwf   num5
    return

CINCO
    movlw   b'00100111'
    movwf   num1
    movlw   b'01000101'
    movwf   num2
    movlw   b'01000101'
    movwf   num3
    movlw   b'01000101'
    movwf   num4
    movlw   b'00111001'
    movwf   num5
    return

CUATRO
    movlw   b'00011000'
    movwf   num1
    movlw   b'00010100'
    movwf   num2
    movlw   b'00010010'
    movwf   num3
    movlw   b'01111111'
    movwf   num4
    movlw   b'00010000'
    movwf   num5
    return

TRES
    movlw   b'00100001'
    movwf   num1
    movlw   b'01000001'
    movwf   num2
    movlw   b'01000101'
    movwf   num3
    movlw   b'01001011'
    movwf   num4
```

```
    movlw   b'00110001'
    movwf   num5
    return

DOS
    movlw   b'01000010'
    movwf   num1
    movlw   b'01100001'
    movwf   num2
    movlw   b'01010001'
    movwf   num3
    movlw   b'01001001'
    movwf   num4
    movlw   b'01000110'
    movwf   num5
    return

UNO
    movlw   b'00000000'
    movwf   num1
    movlw   b'01000010'
    movwf   num2
    movlw   b'01111111'
    movwf   num3
    movlw   b'01000000'
    movwf   num4
    movlw   b'00000000'
    movwf   num5
    return

CERO
    movlw   b'00111110'
    movwf   num1
    movlw   b'01010001'
    movwf   num2
    movlw   b'01001001'
    movwf   num3
    movlw   b'01000101'
    movwf   num4
    movlw   b'00111110'
    movwf   num5
    return


Delay1Sec
            ;1999996 cycles
    movlw   0x11
    movwf   d1
    movlw   0x5D
```

```
    movwf   d2
    movlw   0x05
    movwf   d3
Delay1Sec_0
    decfsz  d1, f
    goto    $+2
    decfsz  d2, f
    goto    $+2
    decfsz  d3, f
    goto    Delay1Sec_0

            ;4 cycles (including call)
    retlw   0x00

    END
```