PROGRAMACIÓN DEL MICROCONTROLADOR PIC 16C64 COMO CONTROLADOR MULTIEJE PARA MOTORES PASO

. Gustavo Caamaño*
; Coordinador de Tecnología en Electrónica

En este artículo se describe la programación de un PIC 16C64 para que funcione como un generador universal de secuencia para cuatro motores paso a paso, con selectores de medio paso o paso completo, tipo de secuencia, sentido de giro y entradas independientes de reloj para cada motor. Se describe el algoritmo para la discriminación de los flancos de subida al trabajar con la característica de interrupción por cambio de la parte alta del puerto B. La programación es también un ejemplo de utilización de todos los puertos del microcontrolador.

Introducción

Los motores paso a paso son ampliamente utilizados en aplicaciones electrónicas, con la particularidad de que la señal de avance debe ser entregada por un circuito secuenciador.

Existen varios fabricantes de circuitos integrados que ofrecen secuenciadores para motores paso a paso de diversos tipos[1].

En este artículo mostraremos cómo se pueden programar cuatro secuenciadores universales en un solo circuito integrado, el cual no es mas que un microcontrolador: el PIC 16C64 de Microchip (ver Figura 1).

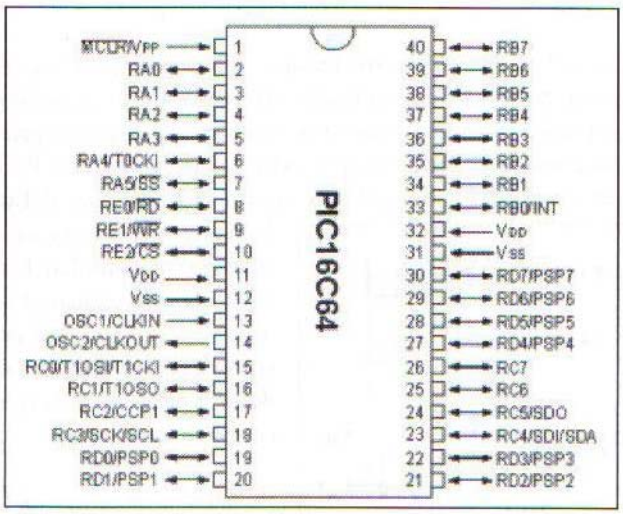


Figura 1 Distribución de pines del PIC16C64

* Ingeniero Electrónico, Coordinador de Tecnología en Electrónica. Especialización en Bioingeniería de la Universidad Distrital. Actualmente se encuentra desarrollando el Doctorado en telecomunicaciones con la Universidad Central de las Villas Santa Clara de Cuba.

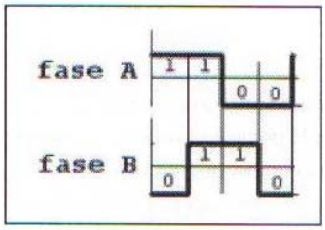


Figura 2 Señal bifásica (90[4]) necesaria para excitar un motor paso a paso

Básicamente, los motores paso a paso son motores bifásicos con separación eléctrica entre polos de 90° (ver Figura *2*)*. Por ser bifásicos necesitan un par de bobinas de excitación (el campo es producido, generalmente, por imán permanente en el rotor).

Dependiendo de la conexión que se haga a las bobinas de excitación, estos motores se clasifican en *monopolares* y *bipolares* (ver Figura 3.), en donde la forma general es el motor bipolar llamado así porque es necesario emplear una fuente de poder dual de corriente directa[2] para invertir el flujo del campo magnético en el núcleo de las bobinas. Sin embargo, la presentación más usual es el motor monopolar, ya que con él se logra utilizar una fuente de polaridad única y un driver más sencillo a expensas del tamaño del motor pues en realidad lo que se hace es duplicar las bobinas. ,

[1] Un ejemplo es la pastilla LS297 fabricada por la SGS.
[2] También es usual utilizar un puente de transistores para evitar el empleo de una fuente dual.

En la Figura 4 se describe la secuencia de manejo de un motor bipolar y en la 5 la de uno monopolar[3].

La secuencia

Es fácil ver que las polaridades necesarias para accionar un motor paso a paso tienen una traducción al lenguaje de unos y ceros utilizado en la electrónica digital como se muestra en la Figura 6.

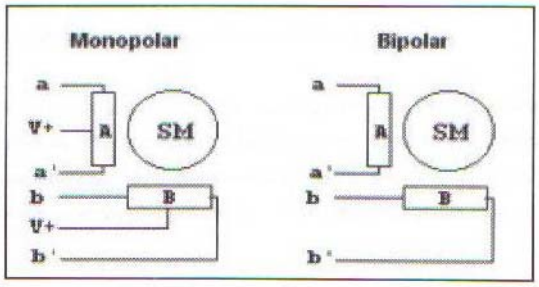


Figura 3 Esquema de un motor monopolar y uno bipolar.

La secuencia de señales binarias entregadas por el circuito generador será traducida por el driver de potencia adecuado para producir las correspondientes corrientes en las bobinas del motor con el fin de hacerlo rotar en un sentido determinado; si se invierte la secuencia, se invierte el sentido de giro del motor.

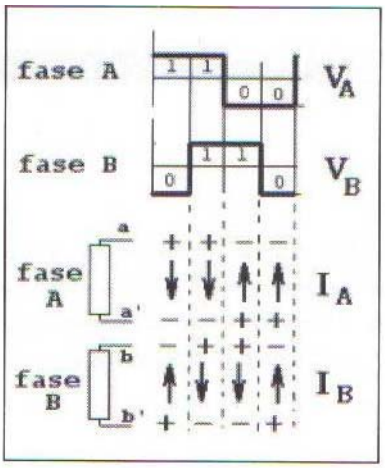
Si se alarga la secuencia intercalando un estado de apagado cada vez que se va a invertir la polaridad sobre una bobina, se obtendrá como consecuencia lo que se llama el modo de medio paso. Este consiste en hacer que el rotor se estacione a mitad de camino entre los polos magnéticos. La secuencia de medio paso también se muestra en la Figura 6.

Para el motor monopolar existe, además, otra secuencia (ver figura 7) llamada wave, muy utilizada por ser más fácil de implementar que las anteriores con circuitos discretos[4], y consiste en alimentar una sola bobina a la vez, consecutivamente. También se emplea la secuencia wave de dos bobinas en la cual se alimentan dos bobinas a la vez. Al igual que en las secuencias clásicas, de la secuencia wave se puede derivar una de medio paso[5] .

[3] En algunos textos se refieren al motor monopolar como motor unipolar de cuatro fases.
[4] Ver http://www.lonestar.texas.net/~diana/stepper.gif
[5] La secuencia wave de medio paso no se implemento en este programa

El programa

Nos proponemos diseñar un programa que permita manejar cuatro motores paso a paso con la adecuada cantidad de controles que rijan el funcionamiento de cada uno de ellos por separado. Estos controles se muestran



corrientes en las bobinas de un motor paso a paso bipolar.

controlador ya que los tres restantes tienen idéntico comportamiento. Nótese que para lograr el avance de un motor es necesario que exista un frente de subida en la entrada de reloj (xCK). Además, se desea que el funcionamiento de los secuenciadores sea lo más independiente q

Asignación de pines

En la Figura 8. Se muestra la asignación de pines, con la cual se empieza a trabajar el diseño del programa. Se ha tenido especial cuidado en reservar la mitad alta del puerto B para las entradas de reloj y así aprovechar la característica del PIC de interrupción por cambio en la parte alta del puerto B. En la Figura 8. se pueden observar con facilidad los pines que son entradas y los que son salidas. Así es como deben configurarse desde el principio de la ejecución del programa.

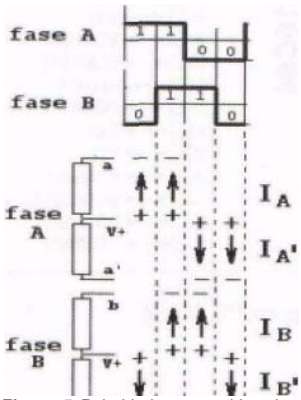
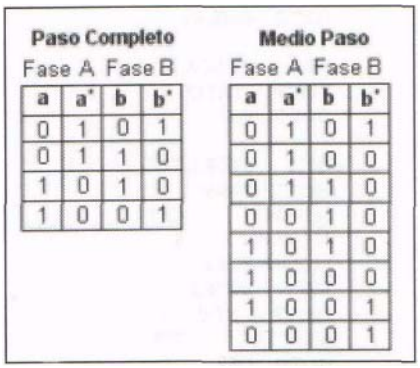


Figura 5 Polaridades y sentidos de las corrientes en las bobinas de un motor monopolar paso *a* paso.

En la Ficha de programa 1 se encuentran estas asignaciones. Primero se clarean todos los registros de los puertos a modo de inicialización y después se accede al banco 1, con el fin de alterar los registros de dirección de entrada — salida, de los cuales sólo tocamos los del puerto C

y del D pues, por defecto, en el arranque todos los puertos se configuran automáticamente como entradas. El siguiente paso es leer qué tipo de secuencia se desea para cada motor, inspeccionando los pines xWAVE y xFULL (ver Tabla 1). Así, se determinan y anotan, por única vez cuales son las tablas con las que se quiere trabajar (ver Ficha de programa 3).



Paso Completo

Fase A Fase B

a	a'	b	b'
0	1	0	1
0	1	1	0
1	0	1	0
1	0	0	1

Medio Paso

Fase A Fase B

a	a'	b	b'
0	1	0	1
0	1	0	0
0	1	1	0
0	0	1	0
1	0	1	0
1	0	0	0
1	0	0	1
0	0	0	1

Figura 6 Secuencia binaria para los modos de paso completo *y* medio paso.

Las posibles secuencias se tienen en unas tablas o bloques de memoria (ver Ficha de programa 2), a saber: paso completo (FULL), medio paso (HALF), wave sencilla (WAVE1) y wave de dos bobinas al tiempo (WAVE2); cuyos valores concuerdan con los que se muestran en las Figuras 6 y 7.

El propósito es que la tabla informe acerca del siguiente valor de la secuencia para el avance del motor, ya sea en sentido horario o antihorario. Se utiliza aquí la convención de que si el sentido deseado para el eje del motor es horario se leerá la tabla en forma ascendente; en el caso contrario se lee la tabla de manera descendente.

↑CK	↑WAVE	↑FULL	↑CW	OPERACION
↑	0	0	0	Medio paso. Sentido antihorario
↑	0	0	1	Medio paso. Sentido horario.
↑	0	1	0	Paso completo. Sentido antihorario.
↑	0	1	1	Paso completo. Sentido horario.
↑	1	0	0	Wave sencillo. Sentido antihorario.
↑	1	0	1	Wave sencillo. Sentido horario.
↑	1	1	0	Wave dos bobinas. Sentido antihorario
↑	1	1	1	Wave dos bobinas. Sentido antihorario

Tabla 1 Controles que determinan el funcionamiento del secuenciador del motor

Para anotar el bloque de memoria que se usará para cada motor, se lee el bit xWAVE; si es cero, se trata de la primera o segunda tabla (FULL o HALF) y por el contrario, si es uno, se refiere entonces a la tercera o cuarta (WAVE1 O WAVE2). La información se completa leyendo el BIT xFULL: si es cero se requiere la primera o tercera tabla, pero si es uno podría ser la segunda o la cuarta.

Paso Completo

Fase A Fase B

a	a'	b	b'
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Figura 7 Secuencia WAVE sencilla

En k rutina QUER se determinan y anotan, por única vez, cuales son las tablas con las que se quiere trabajar inspeccionando los pines xWAVE y xFULL. Hay cuatro registros en los que se va a guardar ésta información: TAB_M4, TAB_M3, TAB_M2, TAB_M1, en los cuales se guardará el modo de funcionamiento de cada motor así: O si es paso completo, 1 si es medio paso, 2 si es wave sencillo y 3 si es wave de dos bobinas (Ver Ficha de programa 3).

```
CONFIG:
        CLRF   PORTA
        CLRF   PORTB
        CLRF   PORTC
        CLRF   PORTD
        CLRF   PORTE

        BSF    STATUS,RP0   ;Banco 1

        CLRF   PORTC; PUERTO C ES SALIDA
                    ; MOTOR 1 Y MOTOR 2
        CLRF   PORTD; PUERTO D ES SALIDA
                    ; MOTOR 3 Y MOTOR 4

    ;LOS DEMAS PUERTOS SON ENTRADAS

        BCF    STATUS,RP0   ;REGRESA AL
                            ;BANCO 0
```

Ficha de Programa 1 Configuración de los puertos.

Interrupciones

Otro de los procesos preparatorios a realizar es la selección de las fuentes de interrupciones[6]. En nuestro caso sólo se utilizará una fuente de interrupción: la interrupción por cambio de la mitad alta del puerto B.

```
FULL: MOVF FLAGA,0
      ADDWF     PCL,1
      RETLW     5       ;     0101
      RETLW     6       ;     0110
      RETLW     0A      ;     1010
      RETLW     9       ;     1001

HALF: MOVF FLAGA,0
      ADDWF     PCL,1
      RETLW     5       ;     0101
      RETLW     4       ;     0100
      RETLW     6       ;     0110
      RETLW     2       ;     0010
      RETLW     0A      ;     1010
      RETLW     8       ;     1000
      RETLW     9       ;     1001
      RETLW     1       ;     0001

WAVE1:      MOVF FLAGA,0
      ADDWF     PCL,1
      RETLW     8       ;     1000
      RETLW     4       ;     0100
      RETLW     2       ;     0010
      RETLW     1       ;     0001

WAVE2:      MOVF FLAGA,0
      ADDWF     PCL,1
      RETLW     0C      ;     1100
      RETLW     6       ;     0110
      RETLW     3       ;     0011
      RETLW     9       ;     1001
```

Ficha de Programa 2 Tablas descriptoras de secuencias

Para disponer esta fuente se requiere únicamente activar el bit de habilitación RBIE, aunque por precaución se recomienda clarear previamente la bandera testigo de interrupción RBIF. Además es necesario preparar los habilitadores globales de interrupción (ver Ficha de Programa 4).

Servicio a las interrupciones

Otro elemento de la programación es la subrutina de servicio a las interrupciones, pues es la encargada de discriminar cuál de las entradas de reloj, xCK, produjo un frente de subida y hacer que el motor correspondiente se mueva en el sentido indicado por xCW (recordar la

```
QUER: MOVLW     4
      MOVWF     CONTEO

QE1:  CLRF   INDF ;TAB_Mx = 0
      RRF    FLAGB,1
      BTFSC  STATUS,C
      GOTO   ONDAS

      RRF    FLAGA,1
      BTFSS  STATUS,C
      GOTO   QE2

      INCF   INDF,1;TAB_MX = 1
      MOVF   FSR,0
      MOVWF       CKT
      MOVLW       8
      SUBWF FSR,1
      BSF    INDF,2
      MOVF   CKT,0
      MOVWF       FSR
      GOTO   QE2

ONDAS:      BSF    INDF,1;TAB_Mx = 2

      RRF    FLAGA,1
      BTFSC  STATUS,C
      INCF   INDF,1;TAB_MX = 3

QE2:  INCF   FSR,1
      DECFSZ      CONTEO,1
      GOTO   QE1
```

Ficha de Programa 3 La rutina QUER define el modo de funcionamiento de cada motor.

Hay un inconveniente inicial a salvar: la interrupción por cambio, como su nombre lo indica, se produce cuando en la entrada xCK ocurre un flanco de subida o un flanco de bajada de la señal de reloj, y no solamente en los flancos de subida como se dictaminó en el planteamiento del diseño. Entonces se requiere un algoritmo que discrimine los tipos de flancos e ignore los de bajada (ver Ficha de Programa 5), lo cual se puede lograr si se conoce el valor anterior del puerto, así:

$$FS = (xCK_T \oplus xCK_{T-1}) \bullet xCK_T$$

Donde FS es uno si el valor anterior del puerto, x_p es cero y el valor actual, xCKp es uno. De lo contrario FS es cero. A continuación puede preguntarse para cada

[6] De hecho el PIC16C64 tiene 9 fuentes de interrupción. 21.

secuenciador si se requiere desplazar el apuntador de tabla respectivo dependiendo del resultado particular de FS[7].

Nótese que el valor del puerto siempre es actualizado sin importar si cambia o no. Ya que quien realmente refleja el movimiento del motor es el índice o apuntador de tabla (IND_Mx), el cual se decrementa si se requiere sentido de giro antihorario, se incrementa si es horario, o simplemente no varía si no se desea que el motor ejecute alguna acción.

El resultado final de la programación se muestra en la Tabla 2.1listado general del Programa.

Conclusiones y recomendaciones

El interés principal del presente experimento es ilustrar la utilidad que tiene un microcontrolador cuando se requiere implementar algunos tipos de circuitos secuenciales. El programa podría ser más simple o el microcontrolador más pequeño pues no siempre se requiere comandar cuatro motores paso a paso en una misma aplicación. Una recomendación que se deja al lector es el estudio de las características mecánicas del motor, de acuerdo con el tipo de secuencia que se use.

[7] Como se puede apreciar en listado general del programa a partir d

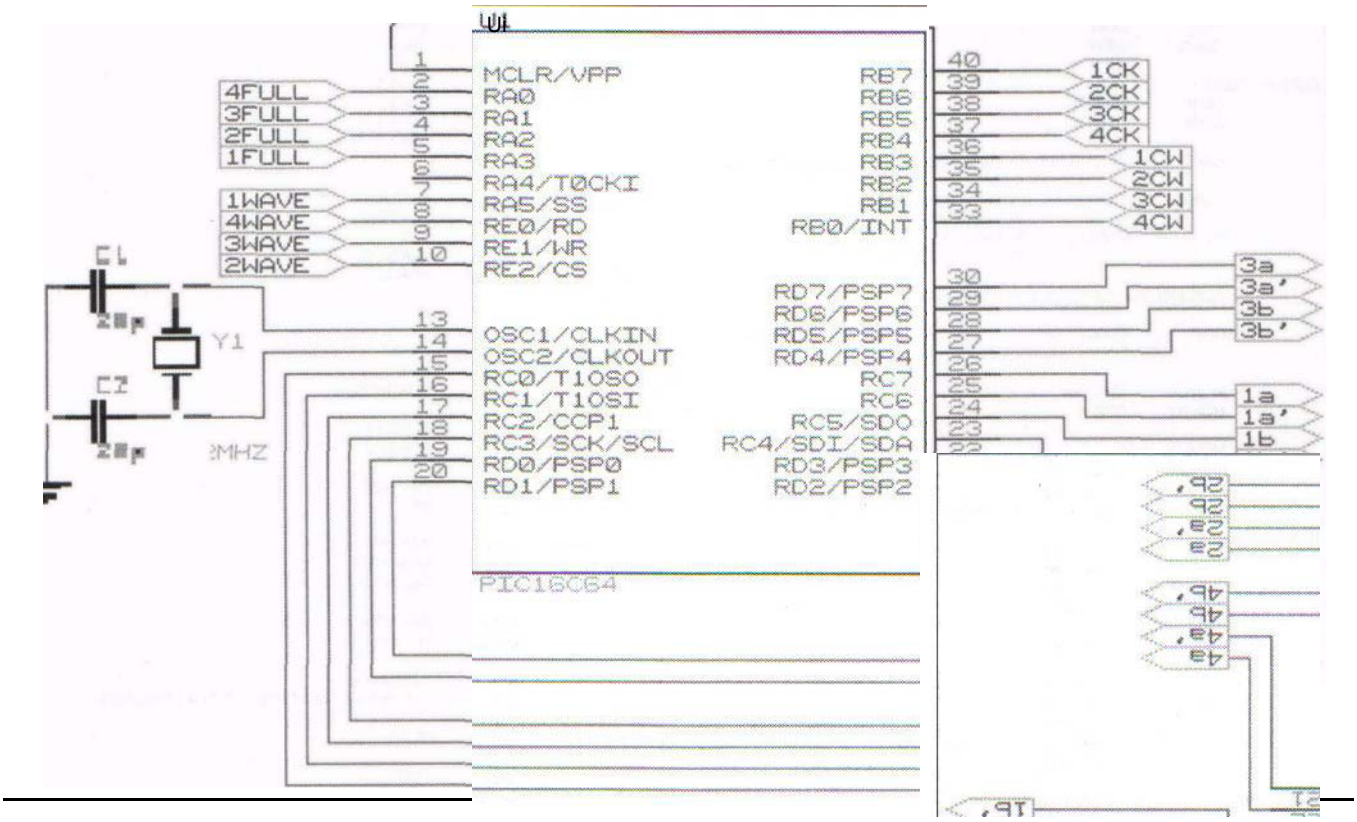


Figura No. 8 Asignación de Pines

LISTADO GENERAL DEL PROGRAMA

```
        LIST    P=16C64


        INCLUDE "p16c64.¡nc"

CONTEO  EQU     20      ;REGISTRO AUXILIAR

PLAGA   EQU     21      ;REGISTRO AUXILIAR
FLAGB   EQU     22      ;REGISTRO AUXILIAR

MAS_M4  EQU     23      ¡MASCARA MOTOR 4
MAS_M3  EQU     24      ;MASCARA MOTOR 3
MAS M2  EQU     25      ;MASCARA MOTOR 2
MAS_M1  EQU     26      ;MASCARA MOTOR 1

IND_M4  EQU     27      ;INDICE SECUENCIA MOTOR 4
IND M3  EQU     28      ;INDICE SECUENCIA MOTOR 3
IND M2  EQU     29      ;INDICE SECUENCIA MOTOR 2
IND_M1  EQU     2A      ;INDICE SECUENCIA MOTOR 1

TAB M4  EQU     28      ;MODO  OPERACIÓN MOTOR 4
TAB M3  EQU     2C      ;MODO  OPERACIÓN MOTOR 3
TAB_M2  EQU     20      ;MODO OPERACIÓN MOTOR 2
TAB_M1  EQU     2E      ;MODO OPERACIÓN MOTOR 1

CKT     EQU     2F      ;VALOR ACTUAL MITAD ALTA DEL PTO. B
CKT_1   EQU     30      ;VALOR ANTERIOR MITAD ALTA PTO. B
FS      EQU     31      ¡FLANCOS DE SUBIDA MITAD ALTA PTO. B


        ORG     0X00
CONFIG:
        CLRF    PORTA
        CLRF    PORTE
        CLRF    PORTC
        GOTO    CONFIG_CONT

        ORG     0X04
        GOTO    ISER

CONFIG.. CONT:
        CLRF    PORTO
        CLRF    PORTE

        BSF     STATUS, RPO     ;Sube al banco 1 para configurar puertos.
        CLRF    PORTC   ; PUERTO C ES SALIDA
                ; MOTOR 1 Y MOTOR 2
        CLRF    PORTO   ; PUERTO D ES SALIDA
                ; MOTOR 3 Y MOTOR 4

        ;LOS DEMÁS PUERTOS SON ENTRADAS
        BCF     STATUS.RPO      ;REGRESA AL BANCO 0


        MOVLW   MAS_M4
        MOVWF   FSR

        MOVLW   4
        MOVWF CONTEO    ;Son cuatro motores

        MOVLW   3
P1:     MOVWF   INDF
        INCF    FSR.1
        DECFSZ  CONTE0.1
        GOTO    P1
        MOVLW   4

        MOVWF   CONTEO
P2:     CLRF    INDF
        INCF    FSR.1
        DECFSZ  CONTEO.1
        GOTO    P2

MOVF    PORTE, 0
        ANDLW   07
        MOVWF   FLAGB
        RLF     FLAGB.1
        RLF     FLAGB.1
        MOVF    PORTA.O
        MOVWF   FLAGA
        ANDLW   20
        IORWF   FLAGB.1
        RRF     FLAGB.1
        RRF     FLAGB.1

ESK2:   MOVWF   FLAGA

SWAPF           FLAGA.1
MOVF            PORTC.O
ANDLW           OE
IORWF   FLAGA,0
MOVWF           PORTC

SEGUMOT:
BTFSS           FS,6
GOTO            TERMOT

BTFSC           PORTB.2
GOTO            HORAR2

ANTIH2: DECF    IND M2,1
GOTO            HH2

HORAR2: INCF    IND M2,1
HH2:    MOVF    MAS_M2,0
ANDWF           IND_M2,1

MOVF            IND_M2,0
MOVWF           FLAGA

BCF             STATUS, C
RLF             TAB_M2,0    ;MULTIPLICA X 2 LA DIRECCIÓN

ADDWF           PCL.1

CALL            FULL
GOTO            ESK3
CALL            HALF
GOTO            ESK3
CALL            WAVE1
GOTO            ESK3
CALL            WAVE2
GOTO            ESK3

ESK3:   MOVWF   FLAGA
MOVF            PORTC.O
ANDLW           OFO
IORWF           FLAGA.O
MOVWF           PORTC


TERMOT:
BTFSS           FS,5
GOTO            CUARMOT

BTFSC           PORTB.1
GOTO            HORAR3

ANTIH3: DECF    IND M3,1
GOTO            HH3

HORAR3: INCF    IND_M3,1
HH3:    MOVF    MAS_M3,0
ANDWF           IND_M3,1
MOVF            IND M3,0
MOVWF           FLAGA

BCF             STATUS.C
RLF             TAB_M3,0    ;MULTIPLICA X 2 LA DIRECCIÓN

ADDWF           PCL.1

CALL            FULL
GOTO            ESK4
CALL            HALF
GOTO            ESK4
CALL            WAVE1
GOTO            ESK4
CALL            WAVE2
GOTO            ESK4
```

```
QUER:   MOVLW   4
        MOVWF   CONTEO

QE1:    CLRF    INDF    ;TAB_Mx = 0
        RRF     FLAGB,1
        BTFSC   STATUS,C
        GOTO    ONDAS

        RRF     FLAGA,1
        BTFSS   STATUS,C
        GOTO    QE2

        INCF    INDF,1  ;TAB_MX = 1
        MOVF    FSR,0
        MOVWF   CKT     ;Salva temporalmente el FSR.
        MOVLW   8
        SUBWF   FSR,1
        BSF     INDF,2  ;Enmascara hasta el tercer bit.
                        ;indicando tabla de ocho valores.
        MOVF    CKT,0
        MOVWF   FSR
        GOTO    QE2

ONDAS:  BSF     INDF,1  ;TAB_Mx = 2

        RRF     FLAGA,1
        BTFSC   STATUS,C
        INCF    INDF,1  ;TAB_MX = 3

QE2:    INCF    FSR,1
        DECFSZ  CONTEO,1
        GOTO    QE1

PREPAINT:
        BCF     INTCON,RBIF
        BSF     INTCON,RBIE

        BSF     INTCON,GIE

        GOTO    $


ISER:   BCF     INTCON,RBIF
        MOVF    PORTB,0
        ANDLW   0F0
        MOVWF   CKT
        XORWF   CKT_1,0
        ANDWF   CKT,0
        MOVWF   FS
        MOVF    CKT,0
        MOVWF   CKT_1

PRIMOT:
        BTFSS   FS,7
        GOTO    SEGUMOT

        BTFSC   PORTB,3
        GOTO    HORAR1

ANTIH1: DECF    IND_M1,1
        GOTO    HH1

HORAR1: INCF    IND_M1,1
HH1:    MOVF    MAS_M1,0
        ANDWF   IND_M1,1

        MOVF    IND_M1,0
        MOVWF   FLAGA

        BCF     STATUS,C
        RLF     TAB_M1,0  ;MULTIPLICA X 2 LA DIRECCION

        ADDWF   PCL,1

        CALL    FULL
        GOTO    ESK2
        CALL    HALF
        GOTO    ESK2
        CALL    WAVE1
        GOTO    ESK2
        CALL    WAVE2

ESK4:   MOVWF   FLAGA
        SWAPF   FLAGA,1
        MOVF    PORTD,0
        ANDLW   0F
        IORWF   FLAGA,0
        MOVWF   PORTD


CUARMOT:
        BTFSS   FS,4
        GOTO    NONA

        BTFSC   PORTB,0
        GOTO    HORAR4

ANTIH4: DECF    IND_M4,1
        GOTO    HH4

HORAR4: INCF    IND_M4,1
HH4:    MOVF    MAS_M4,0
        ANDWF   IND_M4,1

        MOVF    IND_M4,0
        MOVWF   FLAGA

        BCF     STATUS,C
        RLF     TAB_M4,0  ;MULTIPLICA X 2 LA DIRECCION

        ADDWF   PCL,1

        CALL    FULL
        GOTO    ESK5
        CALL    HALF
        GOTO    ESK5
        CALL    WAVE1
        GOTO    ESK5
        CALL    WAVE2


ESK5:   MOVWF   FLAGA
        MOVF    PORTD,0
        ANDLW   0F0
        IORWF   FLAGA,0
        MOVWF   PORTD


NONA:   RETFIE


FULL:   MOVF    FLAGA,0
        ADDWF   PCL,1
        RETLW           5       ;       0101
        RETLW           6       ;       0110
        RETLW           0A      ;       1010
        RETLW           9       ;       1001


HALF:   MOVF    FLAGA,0
        ADDWF   PCL,1
        RETLW           5       ;       0101
        RETLW           4       ;       0100
        RETLW           6       ;       0110
        RETLW           2       ;       0010
        RETLW           0A      ;       1010
        RETLW           8       ;       1000
        RETLW           9       ;       1001
        RETLW           1       ;       0001

WAVE1:  MOVF    FLAGA,0
        ADDWF   PCL,1
        RETLW           8       ;       1000
        RETLW           4       ;       0100
        RETLW           2       ;       0010
        RETLW           1       ;       0001

WAVE2:  MOVF    FLAGA,0
        ADDWF   PCL,1
        RETLW           0C      ;       1100
        RETLW           6       ;       0110
        RETLW           3       ;       0011
        RETLW           9       ;       1001

        END
```