UNIDAD IV

PROGRAMACION

Programación del microcontrolador

Para grabar la memoria de programa el microcontrolador debe llevarse a un modo especial de trabajo suministrando 13.5V a la línea MCLR y el voltaje de la línea Vdd debe estabilizarse ente 4.5V y 5.5V. La memoria de programa se puede grabar de manera serial por medio de las líneas data y clock las cuales deben separarse previamente de las líneas del dispositivo para evitar errores durante la programación.

IV.1. Modos de direccionamiento

Las localidades de la memoria RAM se pueden accesar directamente o indirectamente.

Direccionamiento directo

El modo de direccionamiento directo se realiza por medio de direcciones de nueve bits. La dirección del operando se obtiene uniendo los siete bits de una dirección directa indicada en el código de la instrucción, con los bits RP0 y RP1 del registro STATUS, tal como se muestra en la siguiente figura. Cualquier acceso a los registros F (file registers) es un ejemplo de direccionamiento directo. Un file register es cualquier registro SFR o GPR.

```
bsf     STATUS,RP0    ;Selecciona el banco 1.
movlw   0xFF          ;Carga en W un 0xFF.
movwf   TRISA         ;La dirección del registro TRISA
                      ; es tomada del código de la
                      ;instrucción movwf.
```

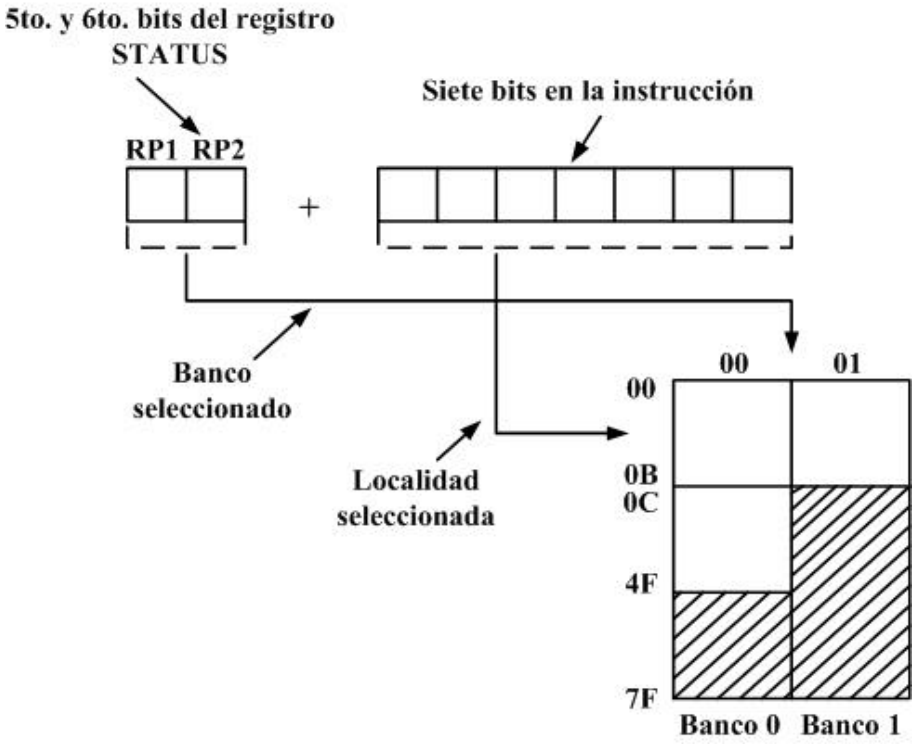


Figura IV.1. Direccionamiento directo

Direccionamiento indirecto

El direccionamiento indirecto, a diferencia del directo, no toma una dirección del código de una instrucción, sino que la construye con la ayuda del bit IRP del registro STATUS y del registro FSR. La localidad direccionada es accesada vía el registro INDF el cual contiene, en efecto, el dato de la dirección indicada por FSR. En otras palabras, cualquier instrucción que use al registro INDF como su registro en realidad estará accesando el dato de la localidad indicada por un registro FSR. Por ejemplo, si un registro de propósito general (GPR) en la dirección 0Fh contiene un valor de 20, al escribir un valor de 0Fh en el registro FSR y leer el registro INDF se obtendrá el valor de 20, lo cual indica que se estará leyendo el GPR sin accesarlo directamente (pero si vía FSR e INDF). Pareciera que este tipo de direccionamiento no tiene ventajas sobre el direccionamiento directo pero ciertas aplicaciones se resuelven más fácil y eficientemente usando direccionamiento indirecto. El registro INDF contiene el dato leído o escrito y el registro FSR la dirección de la localidad de memoria a leer o escribir.

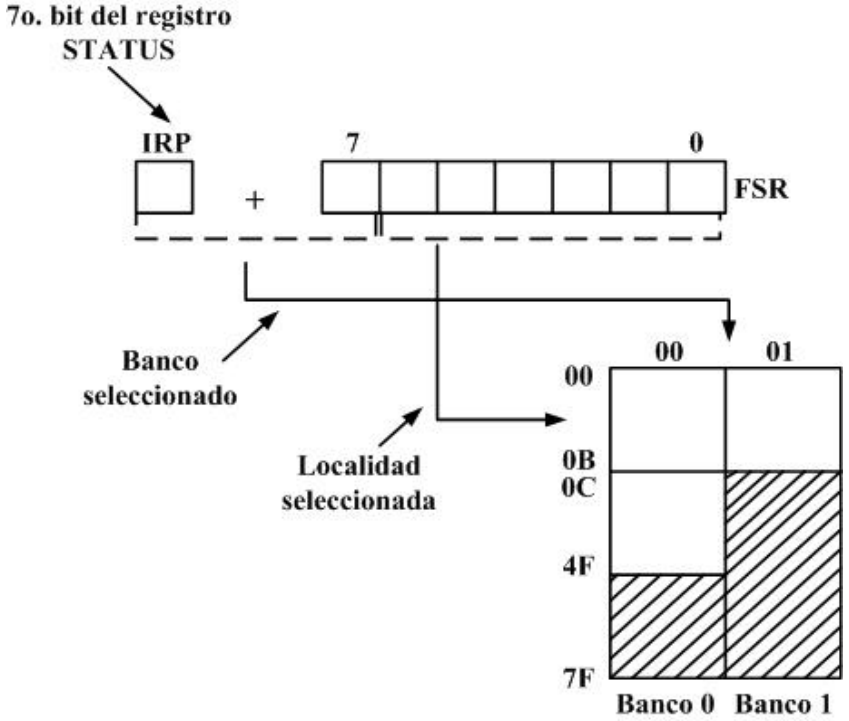


Figura IV.2. Direccionamiento indirecto

Un ejemplo de direccionamiento indirecto es el siguiente, donde se envía un grupo de datos por comunicación serie trabajando con buffer e índices, borrando una parte de la memoria RAM (16 localidades):

```
        movlw   0x0C    ;Establece dirección de inicio.
        movwf   FSR     ;FSR apunta a la dirección 0x0C.
LOOP    clrf    INDF    ;INDF=0.
        incf FSR        :dirección = dirección inicial + 1.
        btfss  FSR,4    ;Están limpias todas las locs.?
        goto LOOP       ;No, regresa al ciclo.
        ....            ;Si, continua con el programa.
```

Una lectura del registro INDF cuando el contenido del registro FSR es cero regresa el valor de cero y una lectura del registro INDF cuando FSR es cero trae como resultado una operación NOP (no operation).

IV.2. Estructura general de un programa

Físicamente, un programa representa un archivo que se encuentra en el disco de la computadora (o en la memoria, si es leído de un microcontrolador) y se escribe de acuerdo a las reglas del ensamblador o algún otro lenguaje que consiste de signos alfabéticos y palabras. Al escribir el programa se deben seguir esas reglas para que un programa intérprete convierta cada instrucción como una serie de ceros y unos que tenga un significado para la lógica interna del microcontrolador.

La conversión se encuentra en un archivo ejecutable y en un archivo con la extensión .HEX, donde .HEX significa hexadecimal, el cual posteriormente se graba en el microcontrolador para su ejecución.

El programa fuente, en lenguaje ensamblador, se hace en un editor de texto y contiene los siguientes elementos básicos:

- Etiquetas.
- Instrucciones.
- Operandos.
- Directivas.
- Comentarios.

Etiquetas

Una etiqueta es una designación textual (generalmente una palabra fácil de leer) para una línea del programa, o sección de un programa a donde puede saltar el microcontrolador o bien el inicio de un conjunto de líneas de un programa. Una etiqueta inicia con una letra del alfabeto o con un caracter underline "_". La longitud de la etiqueta normalmente es de 32 caracteres máximo e inicia en la primera columna.

Instrucciones

Las instrucciones ya están definidas por el microcontrolador específico a usar, de manera tal que solo resta seguir su uso en el lenguaje ensamblador. La forma de escribir las instrucciones se le llama sintaxis de la instrucción. En el ejemplo siguiente se puede reconocer un error de escritura donde las instrucciones **movlp** y **gotto NO** existen para el microcontrolador PIC16F84.

Instrucciones escritas correctamente

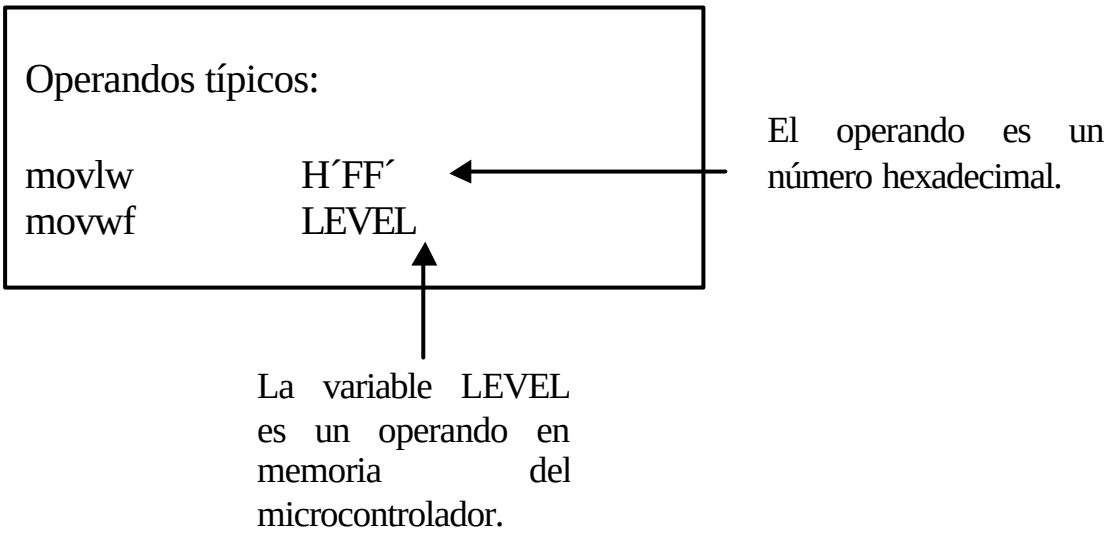
```
movlw   H´01FF´
goto    Start
```

Instrucciones escritas incorrectamente

```
movlp   H´01FF´
gotto   Start
```

Operandos

Los operandos son elementos de las instrucciones que necesitan éstas para poderse ejecutar. Usualmente los operandos son registros, variables o constantes.



Comentarios

Un comentario es una serie de palabras que escribe el programador para hacer más claro y legible el programa. Los comentarios comúnmente se colocan después de una instrucción e inician con un punto y coma “;”.

Directivas

Una directiva es similar a una instrucción, pero a diferencia de una instrucción, la directiva es independiente del modelo del microcontrolador y representa una característica del lenguaje ensamblador mismo. Las directivas se usan para dar un significado poderoso a variables o registros. Por ejemplo, el nombre **NIVEL** se puede usar para designar una variable en la localidad de memoria RAM 0Dh. De esta forma es más fácil para el programador entender o recordar que la localidad de memoria 0Dh contiene información acerca del **NIVEL**. Las directivas que a continuación se toman como ejemplo pertenecen al ensamblador MPASM de Microchip.

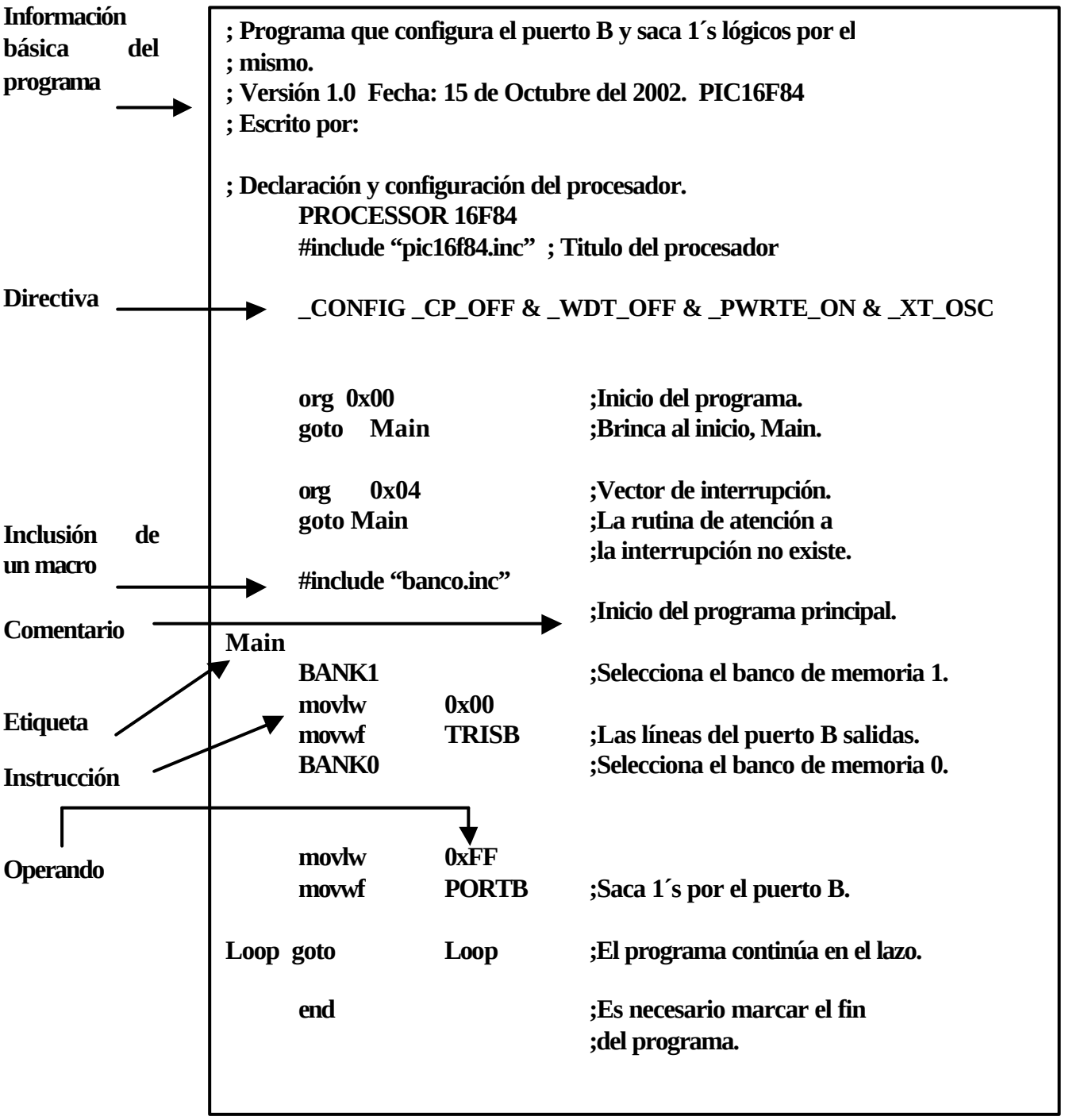
Algunas directivas más frecuentemente usadas:

```
PROCESSOR 16F84
#include    “p16f84.inc”

__CONFIG  _CP_OFF &  _WDT_OFF  &  _PWRTE_ON  &  _XT_OSC
```

Al escribir un programa, existen reglas obligatorias o estrictas y reglas que no son especificadas pero que es recomendable seguirlas. Algunas de estas reglas son las siguientes: escribir al inicio del mismo el nombre del programa, lo que hace el programa, la versión, la fecha cuando fue escrito, el tipo de microcontrolador a usar y el nombre del programador.

El ejemplo siguiente muestra un programa escrito en lenguaje ensamblador respetando las reglas básicas anteriores.


Información básica del programa
Directiva
Inclusión de un macro
Comentario
Etiqueta
Instrucción
Operando
; Programa que configura el puerto B y saca 1´s lógicos por el
; mismo.
; Versión 1.0 Fecha: 15 de Octubre del 2002. PIC16F84
; Escrito por:
; Declaración y configuración del procesador.
PROCESSOR 16F84
#include "pic16f84.inc" ; Titulo del procesador
_CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC
org 0x00 ;Inicio del programa.
goto Main ;Brinca al inicio, Main.
org 0x04 ;Vector de interrupción.
goto Main ;La rutina de atención a
;la interrupción no existe.
#include "banco.inc"
;Inicio del programa principal.
Main
BANK1 ;Selecciona el banco de memoria 1.
movlw 0x00
movwf TRISB ;Las líneas del puerto B salidas.
BANK0 ;Selecciona el banco de memoria 0.
movlw 0xFF
movwf PORTB ;Saca 1´s por el puerto B.
Loop goto Loop ;El programa continúa en el lazo.
end ;Es necesario marcar el fin
;del programa.

En el programa anterior se puede observar que después de los comentarios iniciales se definen varios parámetros importantes del microcontrolador como por ejemplo, el tipo del oscilador, el encendido/apagado del watchdog timer y la habilitación/deshabilitación del circuito interno de reloj, con la directiva siguiente:

__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

Cuando todos los elementos necesarios han sido definidos, se puede empezar a escribir un programa. Primero, es necesario determinar una dirección desde la cual inicia el microcontrolador al encender la fuente de alimentación (org 0x00). A continuación definir la dirección de inicio de la rutina de atención a la interrupción (org 0x04) y posteriormente iniciar el programa principal.

IV.3. Conjunto de instrucciones

El conjunto de instrucciones del PIC16F84 incluye 35 instrucciones ya que se trata de un microcontrolador RISC cuyas instrucciones han sido optimizadas considerando la velocidad de trabajo, arquitectura simple y código compacto.

Las instrucciones del PIC16F84 están clasificadas de la siguiente manera:

- Instrucciones de transferencia de datos.
- Instrucciones aritméticas y lógicas.
- Instrucciones de manejo de bits.
- Instrucciones de transferencia de control.
- Instrucciones especiales.

Instrucciones de transferencia de datos

La transferencia de datos en el PIC16F84 se hace usando el registro de trabajo (W) y un registro f (file register) el cual representa cualquier localidad de la RAM interna (independientemente de que se trate de un SFR o un GPR).

f cualquier localidad de memoria del microcontrolador.
W el registro de trabajo W.
b posición de un bit en el registro "f".
d bit destino.

Mnemónico	Descripción	Operación	Banderas afectadas	Ciclos	Observaciones
MOVLW k	Mueve la constante k a W.	k ✍ W		1	
MOVWF f	Mueve W a f.	W ✍ f		1	
MOVF f,d	Mueve f.	f ✍ d	Z	1	1,2
CLRW	Limpia W.	0 ✍ W	Z	1	
CLRF f	Limpia f.	0 ✍ f	Z	1	2
SWAPF f,d	Intercambia los nibbles de f.	f(7:4),(3:0) ✍ f(3:0),(7:4)		1	1,2

Notas:

1-Si el operando fuente es un puerto, se lee el estado de las líneas del microcontrolador.
2-Si se ejecuta esta instrucción sobre el registro TMR0 y d=1, el prescalar asignado al timer automáticamente se limpia.

Las tres primeras instrucciones de la tabla anterior realizan las siguientes acciones: escribir una constante en el registro W (MOVLW significa MOVe Literal to W), copia un dato del registro W en la RAM y copia un dato de la RAM al registro W (o en la misma localidad de la RAM, en cuyo caso solo cambia el estado de la bandera Z). La instrucción CLRF escribe la constante 00h en el registro f, mientras que la instrucción CLRW escribe la constante 00h en el registro W. La instrucción SWAPF intercambia de lugar los dos nibbles de 4 bits de un registro.

Instrucciones aritméticas y lógicas

De todas la operaciones matemáticas, el PIC16F84, como muchos microcontroladores, únicamente soporta la sustracción y la adición. Se afectan las banderas C, DC y Z de acuerdo a la operación realizada, con una sola excepción: ya que la sustracción se realiza como una adición de un valor negativo, la bandera C toma un valor inverso después de una sustracción. En otras palabras, esta bandera se activa si la operación es posible y se limpia cuando es sustraído un número grande de uno más pequeño.

Mnemónico	Descripción	Operación	Banderas afectadas	Ciclos	Observaciones
ADDLW k	Adiciona una constante y W.	W + k ✍ W	C,DC,Z	1	
ADDWF f,d	Adiciona W y f.	W + f ✍ d	C,DC,Z	1	1,2
SUBLW k	Sustrae W de una constante.	k - W ✍ W	C,DC,Z	1	
SUBWF f,d	Sustrae W de f.	f - W ✍ W	C,DC,Z	1	1,2
ANDLW k	AND de constante con W.	W AND k ✍ W	Z	1	
ANDWF f,d	AND de W con f.	W AND f ✍ d	Z	1	1,2
IORLW k	OR de constante con W.	W OR k ✍ W	Z	1	
IORWF f,d	OR de W con f.	W OR f ✍ d	Z	1	1,2
XORLW k	OR Exclusiva de constante con W.	W XOR k ✍ W	Z	1	1,2
XORWF f,d	OR Exclusiva de W con f.	W XOR f ✍ d	Z	1	
INCF f,d	Incrementa f.	f + 1 ✍ f	Z	1	1,2
DECF f,d	Decrementa f.	f - 1 ✍ f	Z	1	1,2
RLF f,d	Rota a la izquierda f a través del carry.		C	1	1,2

RRF f,d	Rota a la derecha f a través del carry.		C	1	1,2
COMF f,d	Complementa f.	f ✍ d	Z	1	1,2

Notas:

1-Si el operando fuente es un puerto, se lee el estado de las líneas del microcontrolador.

2-Si se ejecuta esta instrucción sobre el registro TMR0 y d=1, el prescalar asignado al timer automáticamente se limpia.

La unidad lógica del PIC tiene la capacidad de realizar la operaciones de AND, OR, EXOR, complemento (COMF) y rotación (RLF y RRF).

Las instrucciones de rotación mueven los bits del registro por medio de la bandera de carry (C) un espacio a la izquierda o a la derecha. El bit que sale del registro se escribe en la bandera C y el valor de la bandera C se escribe en el bit d el lado opuesto del registro.

Instrucciones de manejo de bits

Las instrucciones BCF y BSF establecen a 1 lógico o a 0 lógico, respectivamente, un bit. Aunque esta es una operación muy simple, cuando la CPU la ejecuta primero lee el byte de la localidad de memoria, cambia el bit y después escribe el byte en el mismo lugar.

Mnemónico	Descripción	Operación	Banderas afectadas	Ciclos	Observaciones
BCF f,b	Limpia el bit b de f.	0 ✍ f(b)		1	1.2
BSF f,b	Establece a 1 lógico el bit b de f.	1 ✍ f(b)		1	1.2

Notas:

1-Si el operando fuente es un puerto, se lee el estado de las líneas del microcontrolador.

2-Si se ejecuta esta instrucción sobre el registro TMR0 y d=1, el prescalar asignado al timer automáticamente se limpia.

Instrucciones de transferencia de control

Las instrucciones GOTO, CALL y RETURN se ejecutan de manera similar que en otros microcontroladores, únicamente que el stack es independiente de la memoria RAM, fuera del alcance del programador y limitado a ocho niveles. La instrucción RETLW k es idéntica que la instrucción RETURN, excepto que antes de regresar de un subprograma se escribe en el registro W una constante definida por el operando de la instrucción. Esta instrucción permite diseñar fácilmente tablas de búsqueda (listas). Se usa más comúnmente para determinar la posición en una tabla adicionándole a la dirección de inicio de la tabla la constante definida por la instrucción y leer el dato de esa localidad (la cual se encuentra usualmente en memoria de programa).

La tabla se puede formar como un subprograma que consiste de una serie de instrucciones RETLW k, donde las constantes k son miembros de la tabla.

```
Main    movlw   2
        call    Lookup

Lookup  addwf   PCL,f
        retlw   k
        retlw   k1
        retlw   k2
        :
        :
        retlw   kn
```

En el segmento anterior de un programa se debe escribir la posición de un miembro de la tabla en el registro W, y usando la instrucción CALL se llama a un subprograma que crea la tabla. La primera línea del subprograma, ADDWF PCL,f, suma la posición de un miembro, almacenada en el registro W, a la dirección de inicio de la tabla

encontrada en el registro PCL para encontrar la dirección real del dato localizado en memoria de programa. Al regresar del subprograma se tendrá en el registro W el contenido de un miembro de la tabla direccionada. En el ejemplo anterior, después de ejecutar la instrucción retlw se encontrará en el registro W la constante k2.

Mnemónico	Descripción	Operación	Banderas afectadas	Ciclos	Observaciones
BTFSC f,b	Prueba el bit b de f y salta si es 0 lógico.	Salta si f(b)=0		1(2)	3
BTFSS f,b	Prueba el bit b de f y salta si es 1 lógico.	Salta si f(b)=1		1(2)	3
DECFSZ f,d	Decrementa f y salta si es 0 lógico.	f - 1 ✍ d, salta si Z=1		1(2)	1,2,3
INCFSZ f,d	Incrementa f y salta si es 0 lógico.	f + 1 ✍ d, salta si Z=1		1(2)	1,2,3
GOTO k	Salta a la dirección o etiqueta k.			2	
CALL k	Llama a una subrutina k.			2	
RETURN	Regresa de una subrutina.			2	
RETLW k	Regresa con una constante en W.			2	
RETFIE	Regresa de una interrupción.			2	

Notas:

1-Si el operando fuente es un puerto, se lee el estado de las líneas del microcontrolador.

2-Si se ejecuta esta instrucción sobre el registro TMR0 y d=1, el prescalar asignado al timer automáticamente se limpia.

3-Si se modifica el PC o el resultado de la prueba es 1 lógico, la instrucción se ejecuta en dos ciclos.

La instrucción RETFIE (RETurn From Interrupt and Interrupt Enable) sirve para regresar de una rutina de una interrupción y difiere de la

instrucción RETURN solo en que automáticamente establece a 1 lógico el bit GIE (Global Interrupt Enable). Cuando sucede una interrupción se limpia este bit y solo el valor del program counter se coloca en el tope del stack.

Los saltos condicionales se resumen en dos instrucciones: BTFSC y BTFSS. Dependiendo del bit de que se esté probando del registro f, se ejecutan o no las instrucciones siguientes al BTFSC o BTFSC.

Instrucciones especiales

Mnemónico	Descripción	Operación	Banderas afectadas	Ciclos	Observaciones
NOP	No operation.			1	
CLRWDT	Limpia el Watchdog Timer.	0 ✍ WDT, 1 ✍ TO, 1 ✍ PD	TO, PD	1	
SLEEP	Pasa al modo standby.	0 ✍ WDT, 1 ✍ TO, 0 ✍ PD	TO, PD	1	

IV.4. Periodo de ejecución de las instrucciones

Todas las instrucciones se ejecutan en un ciclo de máquina, excepto las instrucciones de salto condicional que se ejecutan en dos ciclos de máquina si la condición se cumple, o si el contenido del program counter es cambiado por alguna instrucción. En ese caso, la ejecución requiere dos ciclos y durante el segundo ciclo de instrucción se ejecuta una instrucción NOP (No Operation). Un ciclo de instrucción está compuesto de cuatro pulsos de la señal de reloj, por lo que si la frecuencia del oscilador para dicha señal es de 4Mhz, el tiempo para ejecutar una instrucción es de 1 µs, y en caso de saltos condicionales, el periodo de ejecución es de 2 microsegundos.

IV.5. Archivos creados al ensamblar un programa

Como resultado del proceso de ensamblado de un programa se obtienen los siguientes archivos:

- Archivo ejecutable en formato Intel (Nombre_del_programa.HEX).
- Archivo de errores del programa (Nombre_del_programa.ERR).
- Archivo de listado del programa (Nombre_del_programa.LST).

El primer archivo contiene el programa ensamblado y que será grabado en el microcontrolador. El segundo archivo contiene los posibles errores al escribir el programa y que fueron detectados por el ensamblador. Los errores se pueden ver también en el archivo de listado del programa, lo cual es muy útil en programas grandes.

El tercer archivo es el más útil para el programador, ya que contiene mucha información acerca de la ubicación de instrucciones y variables en memoria o señalización de errores, en este archivo se muestra, normalmente, en la parte superior de cada página el nombre del archivo, la fecha cuando fue ensamblado y el número de página. Este archivo se divide en varias columnas, de las cuales la primera indica la dirección en memoria de programa donde se almacena la instrucción de la línea correspondiente del programa. La segunda columna contiene el valor de las variables definidas por las directivas: SET, EQU, VARIABLE, CONSTANT o CBLOCK. La tercera columna está reservada para el ensamblador y la cuarta columna contiene las instrucciones y comentarios del programa. Los errores posibles aparecerán después de la línea donde ocurrió el error.