



**UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA
DEPARTAMENTO DE INGENIERÍA EN
COMPUTACIÓN**

M.I. Rubén Anaya García

LABORATORIO DE MICROCOMPUTADORAS Gpo 7

Práctica #2

Jose Francisco Ugalde Vivo
Hector Montoya Perez
Lara Sala Kevin Arturo

Gpo teoría 5
Gpo teoría 5
Gpo teoría 5

Práctica 2.

Programación en ensamblador direccionamiento indirecto.

Objetivo.

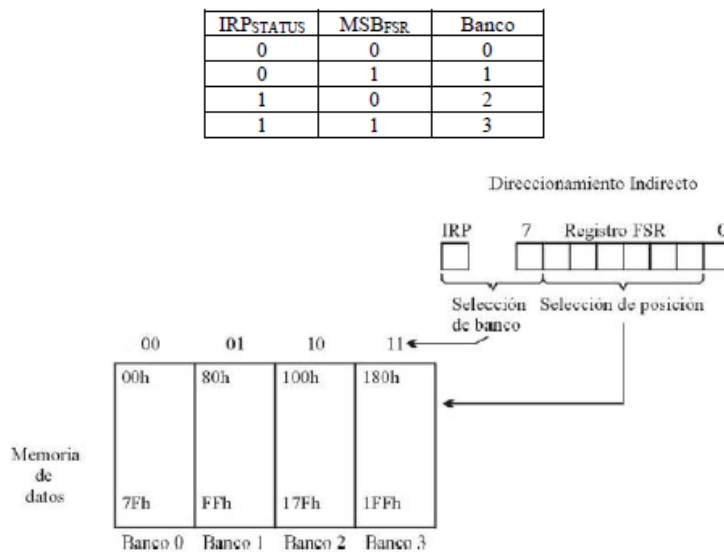
Analizar la programación en lenguaje ensamblador. Realizar algoritmos en lenguaje ensamblador empleando direccionamiento indirecto.

Introducción.

Como fue mencionado en la práctica anterior, este procesador dispone de dos modos de direccionamiento; esta práctica se centrará en el empleo del modo de direccionamiento indirecto.

Direccionamiento indirecto.

El banco de memoria RAM es seleccionado por la codificación de los bits de los registros de STATUS (IRP) y FSR. La dirección dentro del banco será especificada por los bits restantes del registro FSR.



Para poder acceder a la dirección especificada por FSR, deberá ser indicando como parámetro de la instrucción, al registro INDF. Se muestra el siguiente ejemplo.

```
MOVLW 0x20      ; Carga un 0x20 al registro W
MOVWF FSR       ; Mueve el contenido de W al registro FSR (FSR=0x20)
MOVLW 5         ; Carga el valor 5 al registro W
MOVWF INDF      ; Mueve el contenido de W a la dirección apuntada por
                ; el registro FSR; ahora la dirección de memoria 0x20
                ; tendrá como contenido el valor 5.

INCF FSR        ; incrementa al registro FSR (FSR=0x21)
MOVWF INDF      ; en este caso el valor del registro W será almacenado en la localidad
0x21
```

Desarrollo.

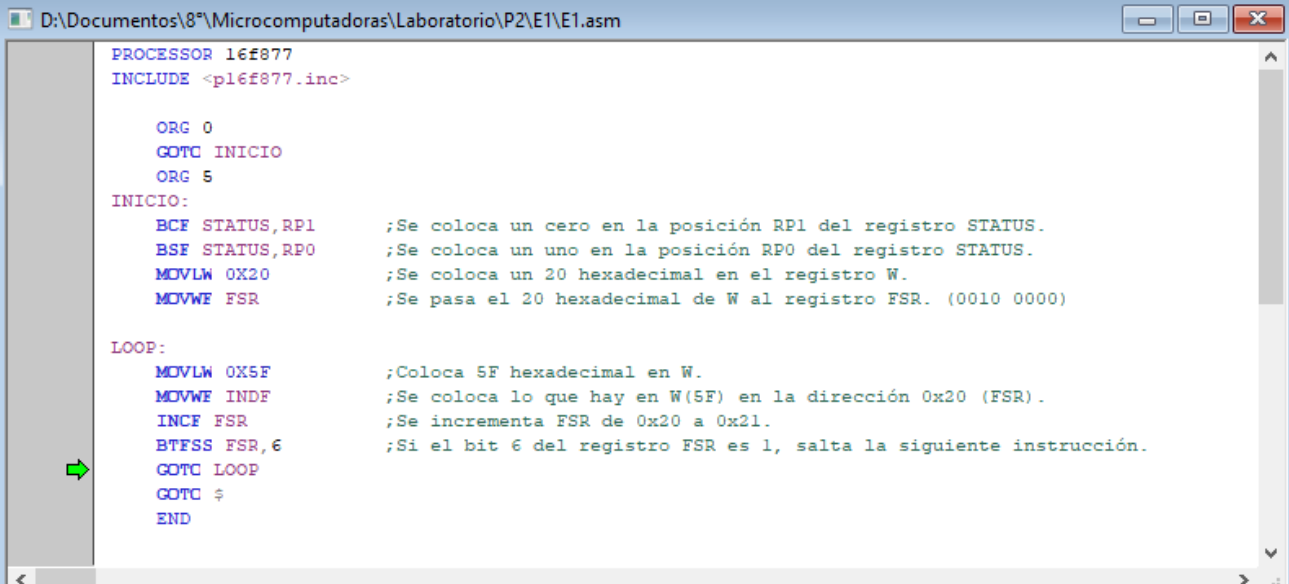
1. Escribir, comentar y ejecutar la simulación del siguiente programa:

```
PROCESSOR 16f877
INCLUDE <p16f877.inc>
    ORG 0
    GOTO INICIO
    ORG 5
INICIO:
    BCF STATUS,RP1
    BSF STATUS,RP0
    MOVLW 0X20
    MOVWF FSR
LOOP:
    MOVLW 0X5F
    MOVWF INDF
    INCF FSR
    BTFSS FSR,6
    GOTO LOOP
    GOTO $
    END
```

Describir el funcionamiento.

Solución.

Código comentado:



```
D:\Documentos\8*\Microcomputadoras\Laboratorio\P2\E1\E1.asm

PROCESSOR 16f877
INCLUDE <p16f877.inc>

    ORG 0
    GOTO INICIO
    ORG 5
INICIO:
    BCF STATUS,RP1      ;Se coloca un cero en la posición RP1 del registro STATUS.
    BSF STATUS,RP0      ;Se coloca un uno en la posición RP0 del registro STATUS.
    MOVLW 0X20           ;Se coloca un 20 hexadecimal en el registro W.
    MOVWF FSR           ;Se pasa el 20 hexadecimal de W al registro FSR. (0010 0000)

LOOP:
    MOVLW 0X5F           ;Coloca 5F hexadecimal en W.
    MOVWF INDF           ;Se coloca lo que hay en W(5F) en la dirección 0x20 (FSR).
    INCF FSR             ;Se incrementa FSR de 0x20 a 0x21.
    BTFSS FSR,6          ;Si el bit 6 del registro FSR es 1, salta la siguiente instrucción.
    GOTO LOOP
    GOTO $
    END
```

Funcionamiento.

Lo que hace el código, es colocar el valor 5F hexadecimal en los registros de la memoria a partir de la dirección 0x20, haciendo uso del direccionamiento indirecto. El ciclo se repite 32 veces, ya que se verifica el bit 6 del registro ($2^6 = 64$; $2^5 = 32$; $64 - 32 = 32$.)

Simulación:

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
000	--	00	0E	38	40	00	00	00	00	00	00	00	00	00	00	00	-.8@...
010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
020	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F
030	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F	5F
040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
080	--	FF	0E	38	40	3F	FF	FF	FF	07	00	00	00	00	00	--	-.8@?..
090	--	00	FF	00	00	--	--	--	02	00	--	--	--	--	00	00
0A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
100	--	00	0E	38	40	--	--	--	--	--	00	00	00	00	00	00	-.8@...

- Elaborar un programa que encuentre el número menor de un conjunto de datos ubicados entre las localidades 0x20 a 0x3F. Mostrar el valor en la dirección 40H.

Código:

```

PROCESSOR 16f877
INCLUDE <pl6f877.inc>

ORG 0
GOTO INICIO
ORG 5

INICIO:
    BCF STATUS,RP1
    BCF STATUS,RP0

    MOVLW 0x20
    MOVWF FSR           ;Nos colocamos en la primera dirección.
    MOVWF INDF           ;Pasamos el valor al registro W.
    MOVWF H'40'         ;Copiamos el valor que hay en W a la dirección 40 (donde irá el resultado).

LOOP:
    INCF FSR
    SUBWF INDF, W        ;Se hace la resta del apuntador INDF menos el valor en W.
    BTFSS STATUS, 0      ;Si la bandera de Carry es 1, se salta la siguiente línea.
    MOVWF INDF           ;Se pasa el valor del apuntador a W.
    BTFSS STATUS, 0
    MOVWF H'40'         ;Se manda el valor actualizado a la dirección 0x40.

    BTFSS FSR, 6         ;Se repite hasta que se acaben los datos.
    GOTO LOOP

    GOTO $
END

```

Funcionamiento:

Se tienen valores aleatorios en los registros 0x20 hasta 0x3F. Se almacena el valor que hay en 0x20 en el registro W. Con el apuntador FSR se va a ir iterando desde 0x21 hasta 0x3F, realizando una resta de lo que haya en memoria menos el valor que hay en W. Si la resta produce un carry en el registro STATUS, significa que W es menor, por lo que no se hace nada, si la resta no produce carry, significa que W es mayor, por lo que se actualiza y se manda a la dirección 0x40 como se solicitó.

Registros y simulación:

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
000	--	00	13	08	40	00	00	00	00	00	00	00	00	00	00	00	-...@... ..
010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
020	FF	FE	FD	FC	FB	FA	89	88	87	86	85	84	83	82	81	80
030	6B	69	68	67	40	41	42	43	44	45	46	47	03	49	90	91	king@ABC DEFG.I..
040	03	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
080	--	FF	13	08	40	3F	FF	FF	FF	07	00	00	00	00	00	--	-...@?... ..
090	--	00	FF	00	00	--	--	--	02	00	--	--	--	--	00	00	-...-... ..
0A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0C0	FE	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
100	--	00	13	08	40	--	00	--	--	--	00	00	00	00	00	00	-...@... ..

Como se puede ver, en la dirección 0x40 se encuentra el valor 03, que es el menor en todo el arreglo de datos.

3. Desarrollar el algoritmo y el programa que ordene de manera ascendente un conjunto de datos ubicados en el banco 0 del registro 0x20 al 0x2F.

Código

```

C:\...\Ejercicio_3.asm
PROCESSOR 16f077
INCLUDE <pl6f077.inc>

ORG 0
GOTO INICIO
ORG 5

INICIO:
    MOVLW H'16'    ;W = 16 - Numero de pasadas
    MOVWF H'30'    ;30 = W

PASADA:
    DECF H'30'     ;Decrementamos a 30
    BTFSK STATUS, Z ;Si no es cero salta
    GOTO $         ;FIN
    MOVLW H'20'    ;W = 20
    MOVWF FSR      ;FSR = W

SIGUENTE:
    MOVLW H'2F'    ;W = 2F
    XORWF FSR, 0   ;W = W XOR FSR
    BTFSK STATUS, Z ;Si no es cero salta
    GOTO PASADA    ;Nueva pasada
    MOVF INDF, 0   ;W = INDF
    MOVWF H'31'    ;30 = W
    INCF FSR, 1    ;FSR++
    SUBWF INDF, 0   ;W = INDF - W
    BTFSK STATUS, C ;If (Positivo) salta
    GOTO SIGUENTE
    GOTO CAMBIO

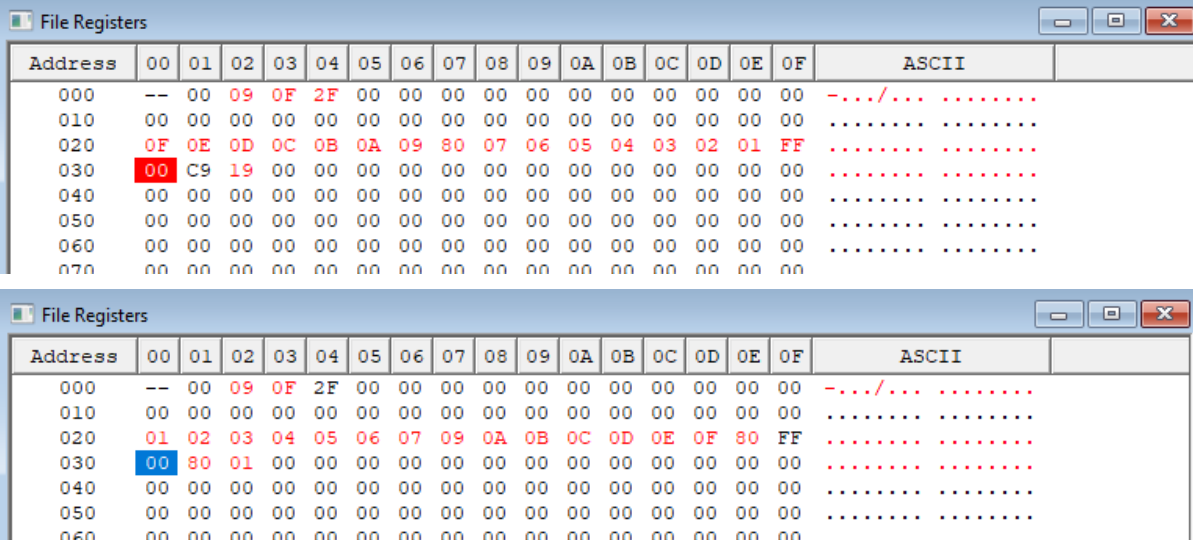
CAMBIO:
    MOVF INDF, 0
    MOVWF H'32'
    MOVF H'31', W
    MOVWF INDF
    DECF FSR, 1
    MOVF H'32', W
    MOVWF INDF
    GOTO SIGUENTE
END

```

Funcionamiento

El funcionamiento está basado en el método de ordenamiento de la burbuja donde validará en cada uno su lado derecho y determinará si estos son mayores o iguales y hará el cambio si es necesario, si no continuara.

Registro y Simulación



Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
000	--	00	09	0F	2F	00	00	00	00	00	00	00	00	00	00	00	-. . . /
010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
020	0F	0E	0D	0C	0B	0A	09	80	07	06	05	04	03	02	01	FF
030	00	C9	19	00	00	00	00	00	00	00	00	00	00	00	00	00
040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
000	--	00	09	0F	2F	00	00	00	00	00	00	00	00	00	00	00	-. . . /
010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
020	01	02	03	04	05	06	07	09	0A	0B	0C	0D	0E	0F	80	FF
030	00	80	01	00	00	00	00	00	00	00	00	00	00	00	00	00
040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Hector Montoya Perez

Dentro de los modos de direccionamiento hay diferentes maneras de especificar especifica la forma de calcular la dirección de memoria efectiva de un operando mediante el uso de la información contenida en registros o constantes, contenida dentro de una instrucción de la máquina o en otra parte.

Su funciones principales son: Dar versatilidad, sirve para manejar estructuras de datos complejas. Y reducir el número de bits del campo de operando. Es tal la importancia de los modos de direccionamiento que la potencia de una máquina se mide tanto por su repertorio de instrucciones como por la variedad de modos de direccionamiento que es capaz de admitir.

Lara Sala Kevin Arturo

El direccionamiento indirecto significa que la dirección de los datos se mantiene en una ubicación intermedia, de modo que la dirección primero se 'busca' y luego se usa para ubicar los datos en sí. Muchos programas utilizan bibliotecas de software que el cargador carga en la memoria en tiempo de ejecución. También se puede preguntar, ¿qué es el modo de direccionamiento indirecto de registros? Registro de modo de direccionamiento indirecto El registro de direccionamiento indirecto significa que la ubicación de un operando se mantiene en un registro. También se denomina direccionamiento indexado o direccionamiento base. El modo de direccionamiento indirecto de registro requiere tres operaciones de lectura para acceder a un operando.

Jose Francisco Ugalde Vivo

El registro FSR sirve como puntero para direccionamiento indirecto además de servir para seleccionar el banco activo. La posición 00 del mapa de RAM es la llamada dirección indirecta.

Si en cualquier instrucción se opera con la dirección 00, en realidad se estará operando con la dirección a donde apunte el contenido del FSR.

Por ejemplo si el FSR contiene el valor 1Ah, una instrucción que opere sobre la dirección 0, en realidad lo hará sobre la dirección 1Ah. Puede decirse que la posición 1Ah de memoria fue direccionada en forma indirecta a través del puntero FSR.

Referencias Bibliográficas

- • Microchip.(2003). PIC16F87XA Data Sheet. Recuperado de <https://ww1.microchip.com/downloads/en/devicedoc/39582b.pdf>
- • Explaining the Instruction Set for the PIC12c508A. Recuperado de http://users.tpg.com.au/users/talking/explaining_instruction_set.html
- <https://findanyanswer.com/what-is-indirect-addressing-mode>