M16

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Semestre 2023-2

Laboratorio de Microcomputadoras

Práctica 1:

Introducción a la programación del microcontrolador PIC16F877; “Direccionamiento Directo”

Profesor:

M.I. Ruben Anaya García

Alumnos:	Número de cuenta:
Barreiro Valdez Alejandro	317520888
Zepeda Baeza Jessica	317520747

Grupo Laboratorio: 4

Grupo Teoría: 1

Fecha de realización: 28 de febrero de 2023

Fecha de entrega: 14 de marzo de 2023

Desarrollo

Como introducción a la programación del microcontrolador PIC16F877, se realizaron 5 ejercicios utilizando direccionamiento directo. Para ello, se revisaron características del microcontrolador como: el tamaño de la memoria FLASH y SRAM; el tamaño de sus buses de instrucciones, datos y direcciones; los registros disponibles para el programador y su ubicación en memoria; la distribución de la memoria y el conjunto de instrucciones para la familia PIC. Los ejercicios realizados fueron:

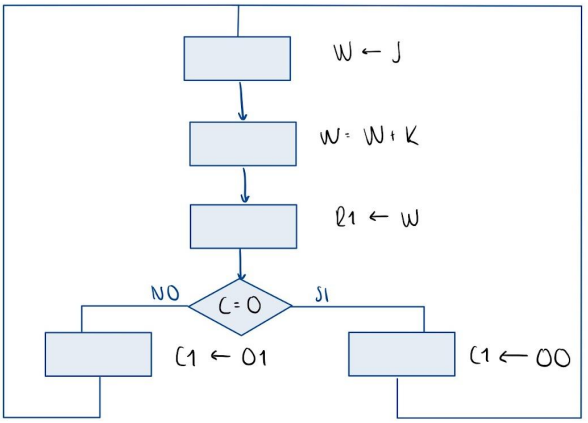
1. Suma de un número en una localidad con 5 y almacenamiento del resultado en otra localidad (solución en el manual).
2. Suma de dos números en distintas localidades y almacenamiento en otra localidad específica (solución en el manual).
3. Suma de dos números en distintas localidades y almacenamiento del resultado y carry en otras dos localidades específicas.
4. Secuencia de potencias de 2 en hexadecimal almacenadas en una misma dirección de memoria.
5. Secuencia de 0 a 20 decimal almacenadas en una misma dirección de memoria.

Todos los programas se realizaron utilizando el IDE MPLAB.

Algoritmos

Ejercicio 3

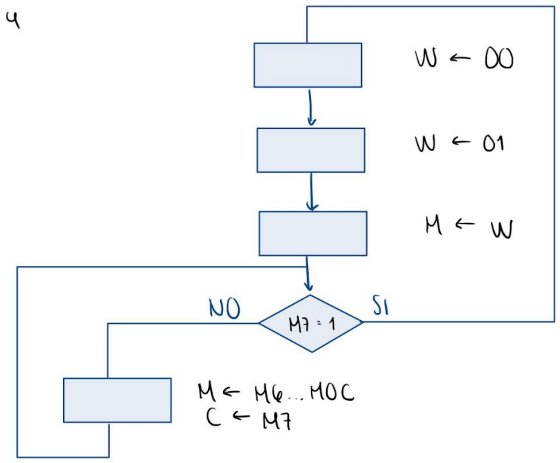
Para el Ejercicio 3 se realizó la siguiente carta ASM:



Donde se guarda el primer número en el registro W, luego se suma con el segundo número y se almacena en W. El resultado se pasa a la localidad de R1. Después se verifica si la operación activó la bandera de Carry. Si no la activó (C=0) el valor en la localidad C1 se vuelve 00 y si sí la activó (C=1) C1 toma el valor de 01.

Ejercicio 4

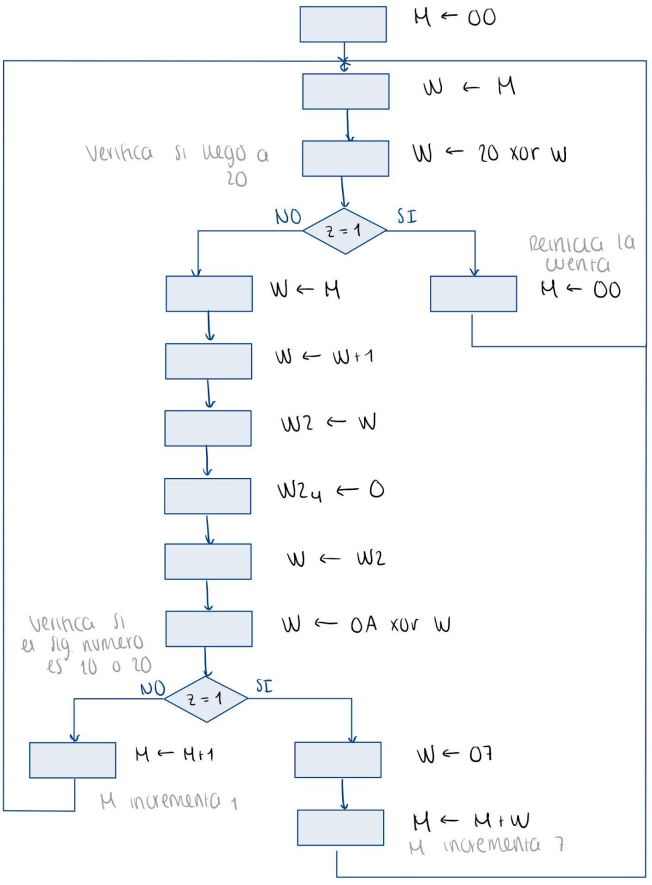
Para el Ejercicio 4 se realizó la siguiente carta ASM:



Se le asigna un valor de 01 a M utilizando el registro W. Se verifica si el bit más significativo de M es 1 y en caso de que no lo sea, se hace una rotación a la izquierda apoyándose del valor del Carry para el bit menos significativo. Se vuelve a verificar el bit más significativo hasta que sí sea 1 y en ese caso se reinicia la secuencia y el programa.

Ejercicio 5

Para el ejercicio 5 se realizó la siguiente carta ASM:



Para este ejercicio se realizaron dos XOR para ubicar cuando la secuencia (el valor de M) llega a números clave: 20 y 0A ya que en caso de ser iguales, la bandera Z se activa.
En un principio se hace un XOR con 20 ya que si se llega a ese número se debe reiniciar M con 00. Pero en caso de que no sean iguales, debe incrementar la secuencia.
Antes de hacer el incremento se verifica si el siguiente número es 0A o 1A ya que este valor se debe representar en decimal como 10 y 20. Por lo tanto se utiliza un registro auxiliar W2 para hacer que tanto 0A como 1A sean 0A y poder hacer el XOR con 0A en W. Si el XOR activa la bandera Z, se debe sumar 7 a M para obtener el 10 o 20 en hexadecimal. Si la bandera no se activa tan solo se le suma 1 a M. Después de hacer los incrementos se regresa al inicio del programa.

Programas comentados

Ejercicio 1

```
PROCESSOR p16f877
INCLUDE <p16f877.inc>

K equ H'26'     ;localidad sumando
L equ H'27'     ;localidad resultado

        ORG 0
        GOTO INICIO     ;salto a inicio de programa

        ORG 5
INICIO: MOVLW H'05'     ;se pone 05 en W
        ADDWF K,0       ;suma de K con W
        MOVWF L         ;se mueve el resultado a L
        GOTO INICIO     ;se regresa al inicio
        END
```

Ejercicio 2

```
PROCESSOR p16f877
INCLUDE <p16f877.inc>

K equ H'26'     ;localidad de sumando
L equ H'27'     ;localidad de sumando
M equ H'28'     ;localidad de resultado

        ORG 0
        GOTO INICIO     ;salto a inicio de programa

        ORG 5
INICIO: MOVF K,W        ;se mueve K a W
        ADDWF L,0       ;se suma K con L en W
        MOVWF M         ;se almacena el resultado en M
        GOTO INICIO     ;se regresa al  inicio
        END
```

Ejercicio 3

```
PROCESSOR p16f877
INCLUDE <p16f877.inc>

J equ H'26'          ;localidad del primer numero
K equ H'27'          ;localidad del segundo numero
C1 equ H'28'         ;localidad del valor del carry
R1 equ H'29'         ;localidad del resultado de la suma

        ORG 0
        GOTO INICIO

        ORG 5
INICIO: MOVF J,W         ;inicio de programa: se mueve num1 a W
        ADDWF K,0        ;se suma W y K y se almacena en W
        MOVWF R1         ;se mueve W a R1
        BTFSC 03H,0      ;se hace un salto si la bandera de carry es 0
        BSF C1,0         ;si no hay salto, carry es 1 y C1 se hace 1
        BCF C1,0         ;si hay salto, carry es 0 y C1 se hace 0
        GOTO INICIO      ;se hace un salto a la etiqueta inicio
        END
```

Ejercicio 4

```
PROCESSOR P16f877
INCLUDE <p16f877.inc>

M equ H'20'          ;localidad donde se vera el resultado

        ORG 0
        GOTO INICIO

        ORG 5
INICIO: CLRW             ;inicio del programa: se limpia el valor de W
        MOVLW 01         ;se coloca 01 en W
        MOVWF M          ;se mueve el valor de W a M
ROTA:   BTFSS M,7        ;se revisa el valor del bit 7 de M, si es 1 hay un salto
        GOTO SIGUE       ;si el bit 7 de M es 0, salta a la etiqueta sigue
        GOTO INICIO      ;si el bit 7 de M es 1, salta al inicio del programa
SIGUE:  RLF M            ;rotan los bits en M hacia la izquierda usando el carry
        GOTO ROTA        ;salta a la etiqueta rota
        END
```

Ejercicio 5

```
PROCESSOR p16f877
INCLUDE <p16f877.inc>

M equ H'20'          ;localidad del resultado
DIEZ equ H'22'       ;localidad que guarda el valor 0A
V equ H'23'          ;localidad que guarda el valor 20
W2 equ H'24'         ;localidad auxiliar

        ORG 0
        CLRF M               ;se limpia el valor en M
        GOTO INICIO          ;salta al inicio de programa

        ORG 5
INICIO:                      ;bloque que verifica si se llegÃ³ a 20 para volver al 0
        MOVF M,W             ;inicio de programa: se mueve M a W
        XORWF V,W            ;W xor V y se almacena el resultado en W
        BTFSS 03H,2          ;salta si la bandera z es 1 (como resultado del xor)
        GOTO INCREMENTO      ;si z es 0, salta a incremento
        CLRF M               ;si z es 1, se limpia M
        GOTO INICIO          ;salta al inicio del programa

INCREMENTO:                  ;bloque que verifica si el siguiente numero 10 o 20
        MOVF M,W             ;mueve M a W
        ADDLW 01             ;incrementa en 1 a W
        MOVWF W2             ;mueve W a W2
        BCF W2,4             ;covierte en 0 al bit 4 de W2
        MOVF W2,W            ;mueve W2 a W
        XORWF DIEZ,W         ;W xor 0A y se almacena el resultado en W
        BTFSS 03H,2          ;salta si la bandera z es 1 (como resultado del xor)
        GOTO NODIEZ          ;si z es 0, salta a sidiez

SIDIEZ:                      ;bloque que aumenta 7 a 09 y 19 para obtener 10 y 20
        MOVLW 07             ;mueve 07 a W
        ADDWF M              ;suma W y M y almacena el resultado en M
        GOTO INICIO          ;salta al inicio del programa

NODIEZ:                      ;bloque que incrementa el numero actual
        INCF M               ;se incrementa M
        GOTO INICIO          ;salta al inicio del programa
        END
```

Conclusiones y/o comentarios

Zepeda Baeza Jessica:

Con esta primera práctica fue posible retomar conocimientos sobre ensamblador aprendidos en materias anteriores y aplicarlos utilizando el conjunto de instrucciones del procesador PIC. Con los programas realizados me pude familiarizar un poco más con los nombres de las instrucciones básicas como cargar un dato, mover datos entre registros, sumar y restar registros, hacer saltos, hacer operaciones lógicas, etc. También aprendí sobre los componentes de este procesador como memorias, buses, registros y banderas y sus tamaños. En todos los ejercicios se hizo uso del direccionamiento directo haciendo referencia a la localidad exacta a la cual se quería acceder. Por último, aunque fueron ejercicios sencillos de secuencias y sumas, al hacerlos aprendimos herramientas como que

los corrimientos a la izquierda multiplican un número por 2 y que se puede verificar la igualdad entre dos números usando la bandera de Z después de un XOR.

Barreiro Valdez Alejandro:

En esta práctica se pudo aplicar en una serie de programas el conocimiento que se obtuvo sobre instrucciones en el lenguaje ensamblador para el procesador PIC. Se realizaron operaciones básicas como mover datos entre registros, sumar dos registros, restar dos registros y controlar el flujo de un programa mediante etiquetas y saltos. Además, se utilizaron otras operaciones como el corrimiento de bits en un registro y operaciones lógicas como XOR para verificar si dos números son iguales. También, se obtuvo conocimiento de todas las banderas involucradas en el desarrollo de estos programas. Se realizaron todos estos ejercicios en un programa llamado MPLab del cual también se obtuvo conocimiento sobre sus herramientas y forma de uso. Por último, se aprendió la estructura del procesador incluyendo sus buses, la organización de sus registros y la forma de direccionamiento. Con todas estas actividades se logró entender un uso básico del procesador PIC para la resolución de ejercicios de programación básica.