FACULTAD DE INGENIERÍA, UNAM

LABORATORIO DE MICROCOMPUTADORASS

SEMESTRE 2023-2

GRUPO 11

PREVIO PRÁCTICA 2

PROGRAMACIÓN EN ENSAMBLADOR DIRECCIONAMIENTO INDIRECTO.

NOMBRE DEL ALUMNO:

ARRIAGA MEJÍA JOSÉ CARLOS

PROFESOR

ING. ROMAN V. OSORIO COMPARAN

FECHA DE ENTREGA: **03 DE MARZO DE 2023** **CALIFICACION**

Objetivo

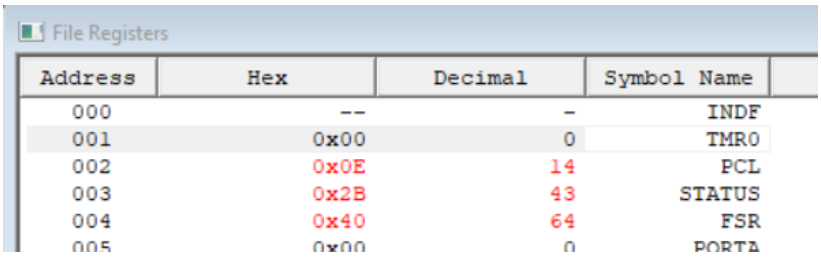
Analizar la programación en lenguaje ensamblador. Realizar algoritmos en lenguaje ensamblador empleando direccionamiento indirecto.

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

a. Describir el funcionamiento

```
PROCESSOR 16f877            ;Indica la versión del procesador
INCLUDE <p16f877.inc>       ;Incluye la biblioteca de la versión del procesador
ORG 0                       ;Especifica el origen
GOTO INICIO                 ;Manda al inicio del programa
ORG 5                       ;Especifica el origen del programa
INICIO:
    BCF STATUS, RP1         ;Coloca en 0 al bit RP1 del registro STATUS
    BSF STATUS, RP0         ;Coloca en 1 al bit RP0 del registro STATUS
                            ;Selección del banco 01
    MOVLW 0x20              ;Carga 0x20 al registro W
    MOVWF FSR               ;Mueve el contenido de W al registro FSR (FSR=0x20)
LOOP:
    MOVLW 0x5F              ;Inicia el loop y carga a W 0x5F
    MOVWF INDF              ;Retorna el valor que apunta FSR
    INCF FSR                ;Incrementa al registro FSR (FSR=0x21)
    BTFSS FSR,6             ;Verifica si el bit 6 de FRS sea 1 (FRS=0x40)
    GOTO LOOP               ;Repite el loop
    GOTO $                  ;Se queda ejecutando esta operación
    END                     ;Fin del programa
```



2.- Elaborar un programa que encuentre el número menor, de un conjunto de datos ubicados entre las localidades de memoria 0x20 a 0X3F; mostrar el valor en la dirección 40H.

Ejemplo:

Dirección	Dato
20	FF
21	FE
22	FD
23	FC
24	FB
25	FA
26	89
27	88
28	87
29	86
2A	85
2B	84
2C	83
2D	82
2E	81
2F	80

Dirección	Dato
30	6B
31	69
32	68
33	67
34	40
35	41
36	42
37	43
38	44
39	45
3A	46
3B	47
3C	48
3D	49
3E	90
3F	**01**

Dirección	Dato
40	01

D:\Escuela\Semestre 23-2\LabMicros\Previos\P2\EJ2.asm

```
PROCESSOR 16f877
#INCLUDE <p16f877.inc>
resultado equ H'40'
ORG 0
GOTO INICIO
ORG 5
INICIO:
    MOVLW 0x20
    MOVWF FSR ;Ya tenemos el FSR preparado en 0x20
    MOVF INDF,0 ;El valor de la localidad FSR se va a W
    MOVWF resultado ;W se vuelve el menor, yéndose a 0x40
CHECK:
    INCF FSR ;El apuntador va a la siguiente localidad
    BTFSC FSR,6 ;Estamos en 0x40?
    GOTO TERMINAR ;SI, ya terminamos de ver todo. Termina el programa
    MOVF INDF,0 ;NO, Lo que hay en la localidad actual se mueve a W
    SUBWF resultado,0 ; RES >= W? (resultado-W)
    BTFSS STATUS,C ;Lo que hay en RES es mayor a W?
    GOTO RES_MENOR ;NO, se va a MENOR
    GOTO RES_MAYOR ;SI, se continúa el proceso
    RES_MENOR: ;Si RES es MENOR no se hace nada, ya es el menor hasta ese punto
    GOTO CHECK
RES_MAYOR: ;Si RES es MAYOR, existe un valor menor hasta ese punto, se actualiza
    MOVF INDF,0
    MOVWF resultado
    GOTO CHECK
TERMINAR:
    SLEEP
    END
```

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.

a. Comprobar el funcionamiento de su programa con distintos conjuntos de datos.

Ejemplo:

20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
0F	**0E**	**0D**	**0C**	**0B**	**FF**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**0A**	**01**

Solución:

20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
01	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0A**	**0B**	**0C**	**0D**	**0E**	**0F**	**FF**

```
PROCESSOR 16f877
#INCLUDE <p16f877.inc>
menor equ H'30' ;Nos ayudara a saber el menor valor al terminar el recorrido
inc_auxiliar equ H'31' ;APUNTADOR Nos ayudara a no perder el indice del incremento cuando recorramos consecutivamente
menor_posicion equ H'32' ;APUNTADOR Nos ayudara a saber la posición del menor valor
valor_aux equ H'33' ;Nos ayudara a guardar el valor en el apuntador de aux, para reemplazarlo en la posicion del menor
ORG 0
GOTC INICIO
ORG 5
INICIO: ;Las preparaciones para la primera iteracion
    MOVLW 0x20 ;Se empieza en la localidad 20
    MOVWF inc_auxiliar ;El auxiliar le pasara su valor a FSR, se controlara FSR, pero sirve para tener un valor fijo que incrementar
SIG_ITERACION:
    MOVF inc_auxiliar,0 ;Se mueve el auxiliar a W
    MOVWF FSR ;Se mueve W a FSR. Prepara FSR apuntando a 0x20, el inicio de los valores
    MOVF INDF,0 ;El valor de la localidad FSR se va a W
    MOVWF menor ;W se vuelve el menor, yendose a 0x30. De aqui partimos
    MOVWF valor_aux ;Este no se modificara, solo se guarda el valor para ordenar posteriormente
CHECK:
    INCF FSR ;El apuntador va a la siguiente localidad
    MOVFW FSR ;Se mueve el valor de FSR a W y se checa la localidad actual
    SUBLW 30h ;Se resta 30 a W y se checara si da 0, para confirmar que ya estamos en 30
    BTFSC STATUS,Z ;El valor de Z=0?S
    GOTC ORDENAR ;NO, Ya llegamos a la localidad 30
    MOVF INDF,0 ;SI, Lo que hay en la localidad actual se mueve a W
    SUBWF menor,0 ;RES >= W? (resultado-W)
    BTFSS STATUS,C ;Lo que hay en RES es mayor a W?
    GOTC RES_MENOR ;NO, se va a MENOR
    GOTC RES_MAYOR ;SI, se continúa el proceso
RES_MENOR: ;resultado menor es el valor más pequeño actualmente
    GOTC CHECK
RES_MAYOR: ;Existe un valor menor al resultado menor actual
    MOVF INDF,0 ;El valor donde se encuentra se manda a W
    MOVWF menor ;Se actualiza el menor valor
    MOVF FSR,0 ;Se mueve la posicion a W
    MOVWF menor_posicion ;Se guarda la posicion del menor
    GOTC CHECK

ORDENAR: ;En este proceso se ordenan los valores
    MOVF inc_auxiliar,0 ;El valor de esta localidad manda a un apuntador al inicio de la lista
    MOVWF FSR ;Actualizamos el FSR, para poder manipular esta localidad
    MOVF menor,0 ;El menor se mueve a W para ordenar
    MOVWF INDF ;Se mueve el menor a FSR o inicio de la iteracion actual
    MOVF menor_posicion,0 ;Se guarda en W el valor anterior del menor lugar
    MOVWF FSR ;La posicion de FSR se actualiza al lugar anterior del valor menor
    MOVF valor_aux,0 ;Se manda el valor auxiliar, anterior primer valor, a W
    MOVWF INDF ;Se actualiza la posicion con el valor de W (El primer valor actual de la lista)
    INCF inc_auxiliar ;Ahora empezaremos desde el siguiente valor
    CLRF menor ;Se limpian los valores de los auxiliares, por si acaso
    CLRF valor_aux
    CLRF menor_posicion
    MOVF inc_auxiliar,0 ;Se mueve el valor del auxiliar a W para revisar posicion de localidad
    SUBLW 30h ;Se resta el valor de la localidad final a W
    BTFSS STATUS,Z ;Skip if Z=1, Se salta si la resta da 0, indicando que ya terminamos
    GOTC SIG_ITERACION ;Se repite mientras no se haya llegado al final
TERMINAR:
    SLEEP
    END
```