UNIVERSIDAD

NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Laboratorio de microcomputadoras

PRACTICA #1

“INTRODUCCION GENERAL A UN MICROCONTROLADOR PIC16F877”

EQUIPO:

- BUSTOS RAMÍREZ LUIS ENRIQUE
- EGUIARTE MORETT LUIS ANDRÉS

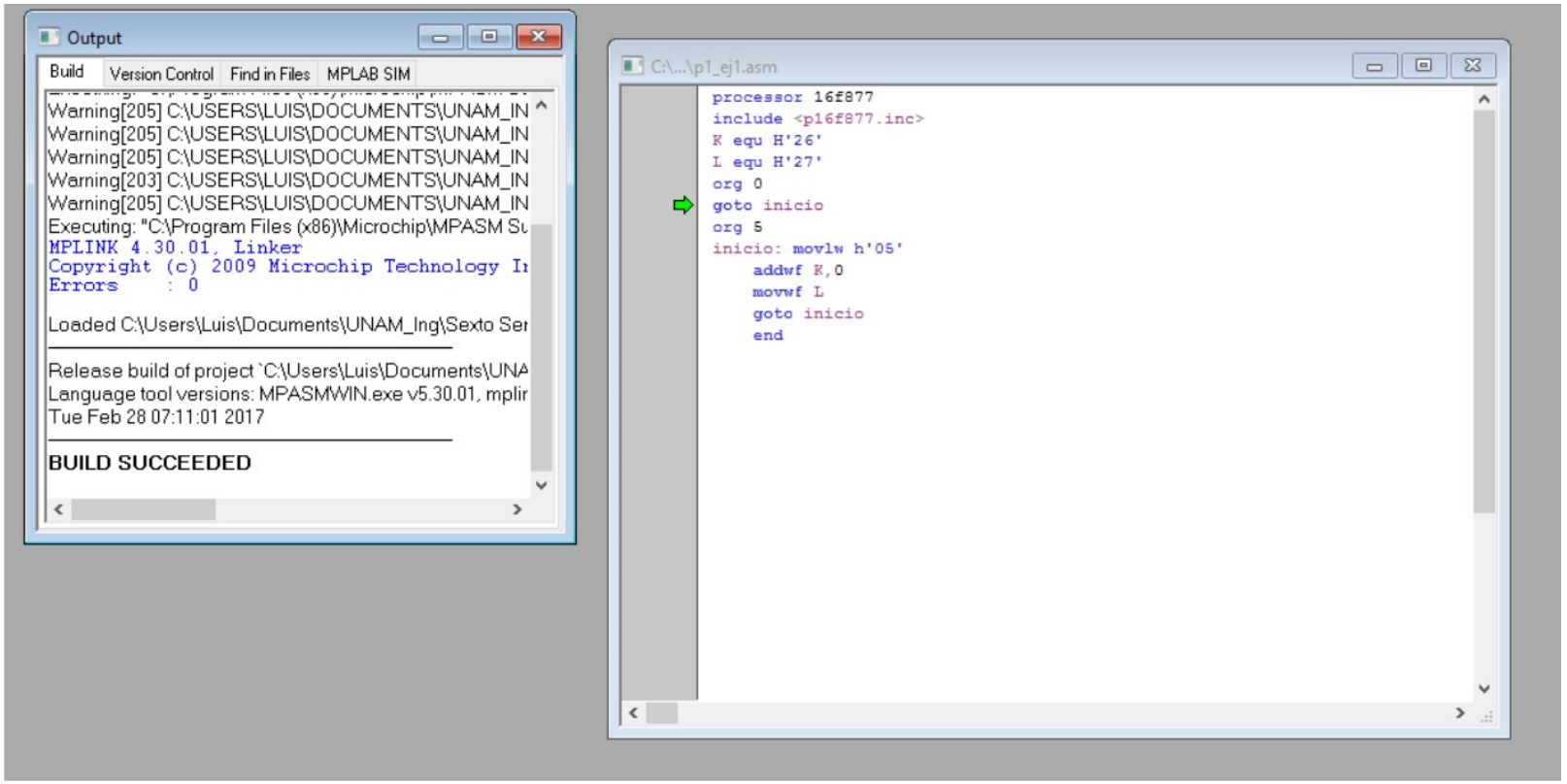
SEMESTRE: 2017-2

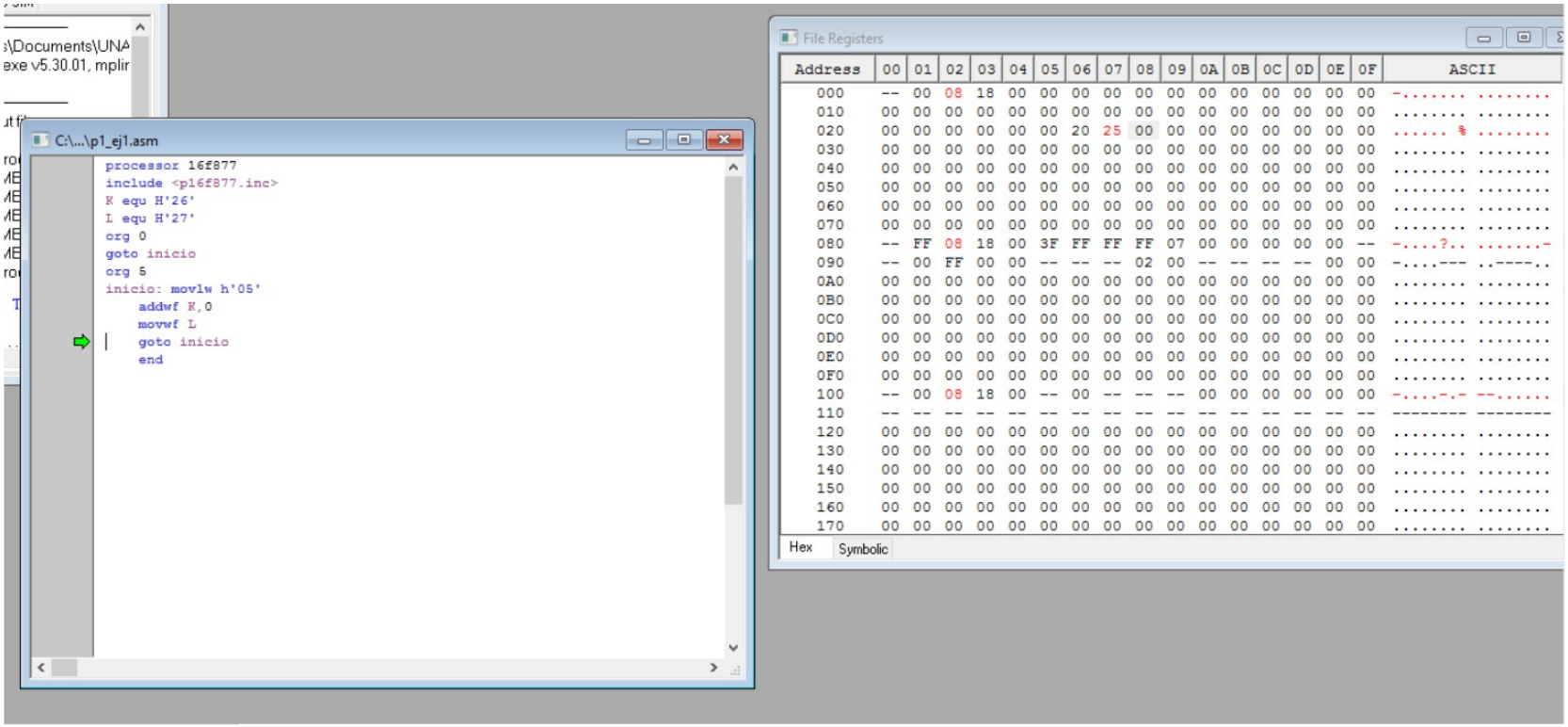
Desarrollo.

Para cada uno de los siguientes ejercicios, realizar los programas solicitados y simular el funcionamiento de ellos.

1.- Siguiendo las indicaciones previas, escribir el siguiente programa, ensamblar y simular el funcionamiento de este.

```
processor 16f877

include <p16f877.inc>


K equ H '26'

L equ H '27'


org 0H


goto inicio

        org 05H


inicio: movlw h '05'

        addwf K, 0 ;sumar K con cero y guardar en W

        movwf L; copiar W a L

        goto inicio

        end
```


Output
Build
Version Control
Find in Files
MPLAB SIM
Warning[205] C:\USERS\LUIS\DOCUMENTS\UNAM_IN
Warning[205] C:\USERS\LUIS\DOCUMENTS\UNAM_IN
Warning[205] C:\USERS\LUIS\DOCUMENTS\UNAM_IN
Warning[203] C:\USERS\LUIS\DOCUMENTS\UNAM_IN
Warning[205] C:\USERS\LUIS\DOCUMENTS\UNAM_IN
Executing: "C:\Program Files (x86)\Microchip\MPASM Su
MPLINK 4.30.01, Linker
Copyright (c) 2009 Microchip Technology In
Errors : 0
Loaded C:\Users\Luis\Documents\UNAM_Ing\Sexto Ser
Release build of project `C:\Users\Luis\Documents\UNA
Language tool versions: MPASMWIN.exe v5.30.01, mplir
Tue Feb 28 07:11:01 2017
BUILD SUCCEEDED
C:\...\p1_ej1.asm
processor 16f877
include <p16f877.inc>
K equ H'26'
L equ H'27'
org 0
goto inicio
org 5
inicio: movlw h'05'
addwf K,0
movwf L
goto inicio
end


C:\...\p1_ej1.asm
processor 16f877
include <p16f877.inc>
K equ H'26'
L equ H'27'
org 0
goto inicio
org 5
inicio: movlw h'05'
addwf K,0
movwf L
goto inicio
end
File Registers
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F ASCII
000 -- 00 08 18 00 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 00 00 00 00 00 00 20 25 00 00 00 00 00 00 00 00 ..... %
030 00 00 00 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
080 -- FF 08 18 00 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 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 08 18 00 -- 00 -- -- -- 00 00 00 00 00 00 -....-.- --......
110 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -------- --------
120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Hex
Symbolic

2.- Modificar el programa anterior, para que ahora los datos que operará se encuentren en las localidades reservadas para **J** y **K** respectivamente y el resultado almacenarlo en otras direcciones, reservadas para **C1** y **R1** donde C1 representará el valor de la bandera de acarreo y R1 el resultado.

Algoritmo:

1. Inicio
2. Suma J y K
3. Guardar resultado en R1
4. Hubo medio acarreo?
5. Si no regresa al inicio vuelve a sumar
6. Si poner un uno en C1
7. Fin

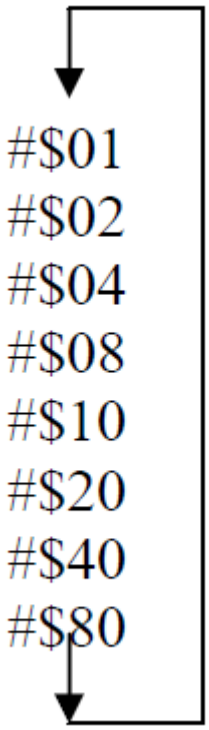
Código:

```
processor 16f877     ;Procesador a utilizar
  include<p16f877.inc> ;Incluir libreria


J  equ h'20'         ;Reservación de localidades en memoria
K  equ h'21'
R1 equ h'22'
C1 equ h'23'


    org 0           ;Vector de RESET
    goto INICIO
    org 5


INICIO:
    movf K,w        ;Carga el contenido de K en W
    addwf J,w       ;Suma W+J y guarda en W
    movwf R1        ;Guarda el contenido de W en R1
    btfss STATUS,DC ;Verifica si hubo medio acarreo
    goto INICIO     ;No, ve a Inicio
    movlw 0x01      ;Si,Carga 0x1 en W
    movwf C1        ;Pon el contenido de W en C1
```

goto INICIO ;Vuelve a iniciar el ciclo

end ;Fin de programa

3. Realice un programa que ejecute la siguiente secuencia, misma que deberá ver en la dirección de memoria de su elección.



Algoritmo:

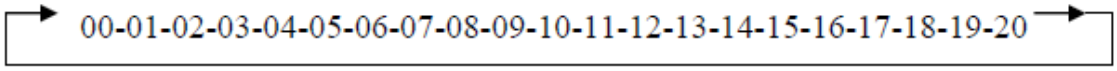
1. Inicio
2. Inicializa un valor de 1 y lo carga en 20H
3. Inicio de ciclo:
4. Hacer un rotamiento a la izquierda de un bit en la dirección de memoria 20H
5. Valor en 20H es 80H?
6. Si no regresa al inicio del ciclo
7. Si vuelve a inicio, inicializa la secuencia de nuevo
8. Fin

Código:

```
processor 16f877     ;Procesador a utilizar

  include<p16f877.inc> ;Incluir libreria
```

```
    org 0           ;Vector de RESET
    goto INICIO
    org 5


INICIO:
    movlw h'1'    ;Carga en W en valor inicial de 1
    movwf h'20'   ;Mueve el valor de W a la Localidad 20h
CICLO:
    rlf h'20',1   ;Hace un rotamiento a la Izquierda y guarda en 20h
    btfss h'20',7 ;Comprueba si el valor que hay en 20h es 80h
    goto CICLO    ;NO, vuelve a hacer corrimiento
    goto INICIO   ;SI, vuelve a iniciar secuencia
FIN
    end
```

4.-Desarrolla un programa que presente la cuenta en numeración decimal en la localidad de memoria de su elección, como se indica a continuación.



Algoritmo:

1. Inicio
2. Se incrementa en uno el número
3. Se verifica el valor 09 en binario mediante el bit 3 y 0
4. Si es 9 en la parte baja se le suman 7
5. Se repite el proceso desde el punto 2
6. Fin

Código:

```
processor 16f877
include <p16f877.inc>
        I equ H'20'
                org 0
                goto inicio
        inicio:
                org 5
```

```
                incf I;  I++
                btfss I,3; omite si esta en 1 el bit
                goto inicio
                btfsc I,0; omite si esta en 1 el bit
                goto sig
                goto inicio
        sig:
                movlw 0x07; w = 7
                addwf I; w= w+I
        inc:
                incf I; I = I + 1
                btfss I,5; omite si esta en 1 el bit
                goto inicio
                movlw 0x00; w = 0
                movwf I; I = w
                goto inicio
end
```

5.-Elabora un programa que encuentre el número menor, de un conjunto de datos ubicados entre las localidades de memoria 20h a 40h: mostrar el valor en la dirección 41h.

Algoritmo:

1. Inicio
2. Contenido de localidad 20h es igual al menor
3. Se guarda el valor en W
4. Se mueve el apuntador a la localidad siguiente
5. Se resta W y Xh donde Xh es el valor de la localidad siguiente
6. Si el resultado es negativo, el contenido de Xh se vuelve el menor
7. Si no se mantiene el número menor
8. W se iguala al número menor
9. Se repite el ciclo desde el punto 4
10. FIn

Código:

```
processor 16f877
include <p16f877.inc>
        K equ H'41'; numero menor
                org 0
                goto inicio
inicio:
                org 5
                movlw 0x20; W = 20
                movwf FSR; FSR = W =20
```

```
                movf INDF, W; W = [FSR] = INDF
                movwf K; K=W
evalua:
                btfsc FSR,6 ; FSR[6] = 0?
                goto inicio
                movf K,W ; W = K
                subwf INDF,Q ; Q=INDF-W en complemento a 2
                btfsc STATUS, C; C=0? es negativo?
                goto conserva ;resultado positivo de la operación subwf
                goto cambia ;resultado negativo de la operación subwf
cambia:
                movf INDF,W; W = INDF
                movwf K; K= número menor = W
                incf FSR ; FSR + 1
                goto evalua
conserva:
                incf FSR ; FSR +1
                goto evalua
end
```

Conclusiones.

Bustos Ramírez Luis Enrique: Esta práctica me ayudo a familiarizarme con el set de instrucciones del PIC16F877, ya que te pide implementar direccionamiento indirecto, barrido de bits y condicionales para poder resolver los problemas propuestos. En general fue una práctica exitosa pues se pudieron resolver todos los problemas.

Eguiarte Morett Luis Andres: Con esta práctica me fue posible tener un nivel de aprendizaje mucho mayor del funcionamiento del microcontrolador PIC16F877 así como reconocer el funcionamiento del set de instrucciones y utilizar este conjunto de instrucciones para lograr efectuar los algoritmos necesarios para lograr resolver los ejercicios propuestos en el ensamblador correspondiente a este microcomputador.