Laboratorio de Microcomputadoras
Práctica No. 1
Introducción General Microcontrolador PIC16F877

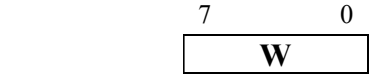
Objetivo. Familiarizar al alumno en el conocimiento del ensamblador, el simulador, el conjunto de instrucciones de un microcontrolador y ejecutar programas en tiempo de simulación.

Introducción

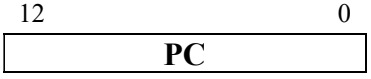
Algunas de las características más importantes del microcontrolador son:

- 8K de memoria FLASH
- 368 bytes de memoria RAM
- 255 bytes de memoria EEPROM
- 35 instrucciones
- 5 puertos paralelos (A, B, C, D, E)
- Convertidor Analógico Digital
- Comunicación Serie Asíncrona
- Comunicación Serie Síncrona (SPI, I2C)
- Tres módulos temporizadores
- Dos módulos CCP que pueden operar como Comparación, Captura o PWM
- 14 fuentes de interrupción

Los registros disponibles para el programador son:

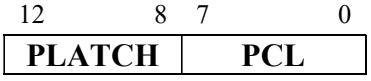


Registro de trabajo W



Registro Contador de Programa

El registro PC se forma con dos registros de ocho bits:



Registro de banderas

El registro STATUS está compuesto por las siguientes banderas:

7	6	5	4	3	2	1	0
IRP	RP1	RP0	T0	PD	Z	DC	C

IRP — Selecciona el banco de memoria RAM en direccionamiento indirecto en conjunto con la bandera más significativa del registro FSR.

RP1, RPO — Seleccionan el banco de memoria RAM en direccionamiento directo.

TO,PD — Banderas de interrupciones.

Z — Indica el estado del resultado de la última operación, si Z=1 el resultado fue cero y si Z=0 el resultado previo fue diferente de cero.

DC — Indica si existió un medio acarreo.

C — Indica si fue generado in acarreo como resultado de la última operación; si C=1 existió un acarreo y si C=0 no existió acarreo.

Memoria de datos

Consta de un total de 368 localidades de memoria a los cuales se les denomina registros; existen registros de propósito específico, los cuales tendrán la función de configurar o indicar el estado de algún periférico o recurso del microcontrolador; los registros PC y STATUS pertenecen a este grupo.

También existen los registros de propósito general, los cuales serán útiles en la ejecución de los programas para almacenar datos y resultados de operaciones.

La memoria es dividida en cuatro bloques (denominados bancos) de 128 localidades de memoria (registros). Para acceder a un registro deberá ubicarse en el banco de memoria en el cuál se encuentra, se configuran las banderas RP0 y RP1 en el registro STATUS (direccionamiento directo).

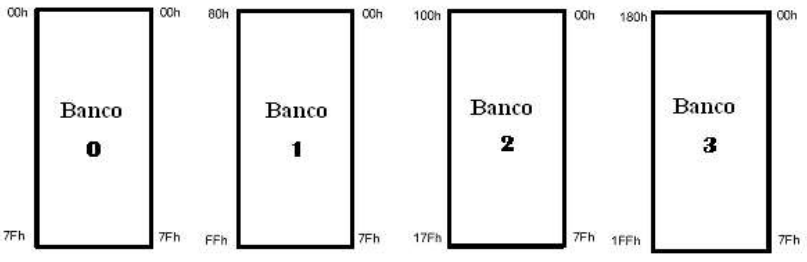


Figura 1.1 Bancos de memoria de datos

El mapa de memoria y distribución de registros se muestra en la figura 1.2.

Banco 0		Banco 1		Banco 2		Banco 3	
00H	**INDF**	80H	**INDF**	100H	**INDF**	180H	**INDF**
01H	**TMR0**	81H	**OPTION_REG**	101H	**TMR0**	181H	**OPTION_REG**
02H	**PCL**	82H	**PCL**	102H	**PCL**	182H	**PCL**
03H	**STATUS**	83H	**STATUS**	103H	**STATUS**	183H	**STATUS**
04H	**FSR**	84H	**FSR**	104H	**FSR**	184H	**FSR**
05H	**PORTA**	85H	**TRISA**	105H		185H	
06H	**PORTB**	86H	**TRISB**	106H	**PORTB**	186H	**TRISB**
07H	**PORTC**	87H	**TRISC**	107H		187H	
09H	**PORTD**	89H	**TRISD**	109H		189H	
09H	**PORTE**	89H	**TRISE**	109H		189H	
0AH	**PCLATCH**	8AH	**PCLATCH**	10AH	**PCLATCH**	18AH	**PCLATCH**
0BH	**INTCON**	8BH	**INTCON**	10BH	**INTCON**	18BH	**INTCON**
0CH	**PIR1**	8CH	**PIE1**	10CH	**EEDATA**	18CH	**EECON1**
0DH	**PIR2**	8DH	**PIE2**	10DH	**EEADR**	18DH	**EECON2**
0EH	**TMR1L**	8EH	**PCON**	10EH	**EEDATH**	18EH	
0FH	**TMR1H**	8FH		10FH	**EEADRH**	18FH	
10H	**T1CON**	90H		110H		1A0H	
11H	**TMR2**	91H	**SSPCON2**	111H		1A1H	
12H	**T2CON**	92H	**PR2**	112H		1A2H	
13H	**SSPBUF**	93H	**SSPADD**	113H		1A3H	
14H	**SSPCON**	94H	**SSPSTAT**	114H		1A4H	
15H	**CCPR1L**	95H		115H		1A5H	
16H	**CCPR1H**	96H		116H		1A6H	
17H	**CCPCON1**	97H		117H		1A7H	
18H	**RCSTA**	98H	**TXSTA**	118H		1A8H	
19H	**TXREG**	99H	**SPBRG**	119H		1A9H	
1AH	**RCREG**	9AH		11AH		1AAH	
1BH	**CCPR2L**	9BH		11BH		1ABH	
1CH	**CCPR2H**	9CH		11CH		1ACH	
1DH	**CCP2CON**	9DH		11DH		1ADH	
1EH	**ADRESH**	9EH	**ADRESL**	11EH		1AEH	
1FH	**ADCON0**	9FH	**ADCON1**	11FH		1AFH	
20H	**96 REGISTROS DE PROPÓSITO GENERAL**	A0H	**80 REGISTROS DE PROPÓSITO GENERAL**	120H	**80 REGISTROS DE PROPÓSITO GENERAL**	1A0H	**80 REGISTROS DE PROPÓSITO GENERAL**
		EFH		16FH		1EF	
		F0H		170H		1F0	
7FH		FFH		17FH		1FFH	

No disponible

Figura 1.2 Mapa de memoria de datos

Memoria de Programa

Memoria Flash de 8K x 14 bits, cuya función será almacenar los programas que ejecutará el procesador; el mapa de memoria es el siguiente:

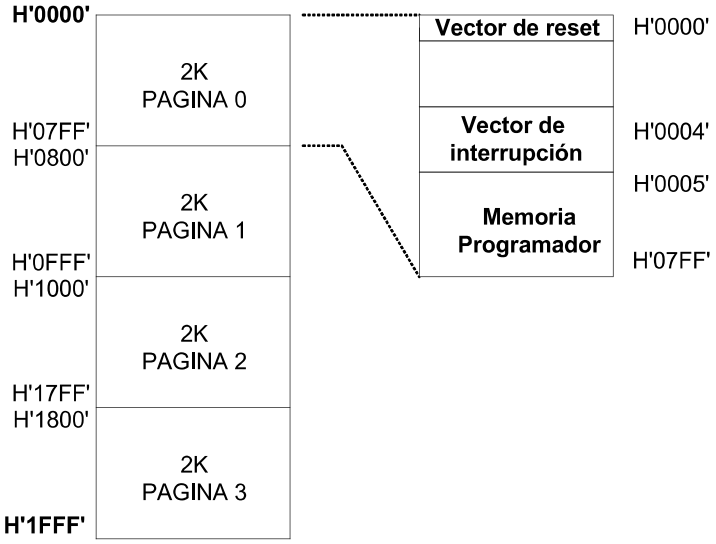


Figura 1.3 Mapa de memoria de programa

De acuerdo a lo anterior; el programa en ensamblador deberá cumplir el siguiente formato.

```
Etiquetas   Directivas y/o Instrucciones   Comentarios

            PROCESSOR 16f877               ; Indica la versión de procesador
            INCLUDE <p16f877.inc>          ; Incluye la librería de la versión del procesador


            ORG 0                          ; Especifica un origen (vector de reset)
            GOTO INICIO                    ; Código del programa

            ORG 5                          ; Indica origen para inicio del programa
INICIO:     INSTRUCCIÓN 1                  ; Instrucciones que se ejecutan una sola vez
                  :

LOOP:       INSTRUCCIÓN N                  ; Instrucciones que se repiten


            INSTRUCCCION M
            GOTO LOOP                      ; repite el ciclo (salta a la etiqueta LOOP)

            END                            ; Directiva de fin de programa
```

La gama media de la familia de los PIC's tiene el siguiente conjunto de instrucciones.

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb			LSb	Status Affected
BYTE-ORIENTED FILE REGISTER OPERATIONS								
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff	
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff	
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z
MOVWF	f	Move W to f	1	00	0000	lfff	ffff	
NOP	-	No Operation	1	00	0000	0xx0	0000	
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff	
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z
BIT-ORIENTED FILE REGISTER OPERATIONS								
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff	
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff	
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff	
LITERAL AND CONTROL OPERATIONS								
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk	
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk	
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk	
RETFIE	-	Return from interrupt	2	00	0000	0000	1001	
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk	
RETURN	-	Return from Subroutine	2	00	0000	0000	1000	
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z

Figura 1.4 Conjunto de instrucciones del PIC 16F877

Descripción de las instrucciones

a. Instrucciones orientadas a registros

Formato: **INSTRUCCIÓN F,D**

INSTRUCCIÓN Corresponde al mnemónico de la instrucción a realizar.

F Es el registro a operar en la instrucción.

D Es el destino de la instrucción.

Donde, si:

D = 1, F, f ,DEF_REG ó nada ; el resultado se almacena en el registro.

Ejemplo:
ADDWF **OX20**
ADDWF OX30,**1**
ADDWF 0X40,**F**
ADDWF REG,**REG**
ADDWF **REG**

D = W, w ó 0 ; el resultado se almacena en el registro W.

Ejemplo:
ADDWF 0X20,**0**
ADDWF 0X30,**W**
ADDWF REG,**W**
ADDWF REG,**w**

b. Instrucciones orientadas a manejo de bits

Formato: **INSTRUCCIÓN F,B**

INSTRUCCIÓN Corresponde al mnemónico de la instrucción a ejecutar.

F Es el registro a utilizar en la instrucción.

B Es la posición del bit en el registro (0,1,2,3,4,5,6,7).

MSB							***LSB***
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Ejemplo:

```
BCF 0X20,0          ; Coloca en 0 al bit 0 del registro 0x20.

BSF REG,7           ; Coloca en 1 al bit 7 del registro definido como  REG.

BTFSC REG,2         ;Prueba el bit 2 del registro REG y salta una instrucción si este
INST_FALSA          bit es cero.
INST_VERDADERA
```

c. Instrucciones orientadas a manejo de constantes e instrucciones de control

Formato: **INSTRUCCIÓN K**

INSTRUCCION Corresponde al mnemónico de la instrucción a realizar.

K Constante a operar con la instrucción

```
MOVLW H´10´   ; Mueve el número 10 en hexadecimal al registro W.

GOTO LOOP     ; Cambia el control del programa a la ubicación de la etiqueta LOOP.

CALL RET      ; Ejecuta la subrutina RET (el procesador almacena en la pila la
              dirección de la siguiente instrucción).

RETURN        ; Regresa de la subrutina (recupera de la pila la dirección de la
              instrucción pendiente).
```

Modos de direccionamiento

Esta versión de microcontrolador dispone de dos modos de direccionamiento:

a. Directo
b. Indirecto

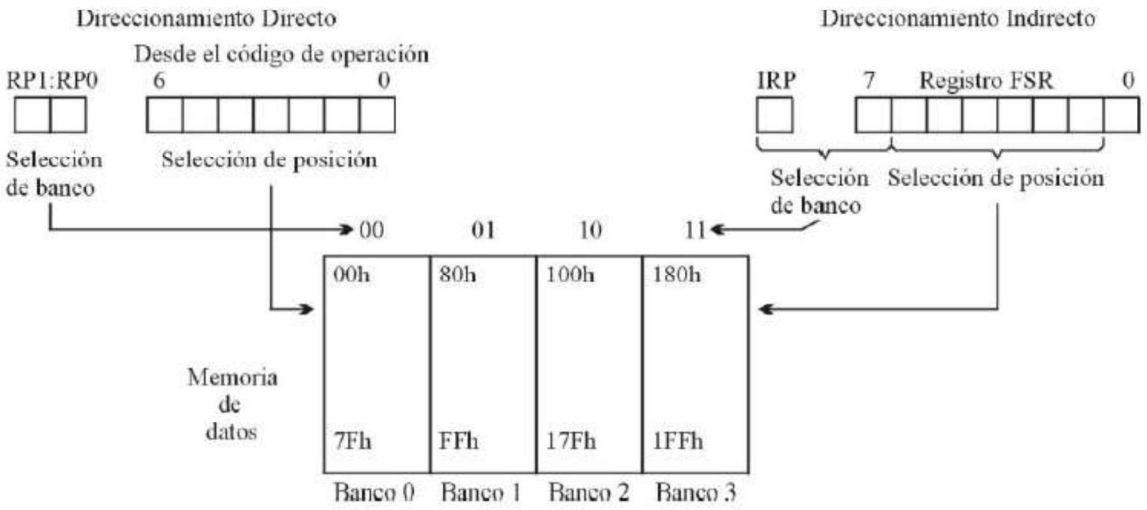
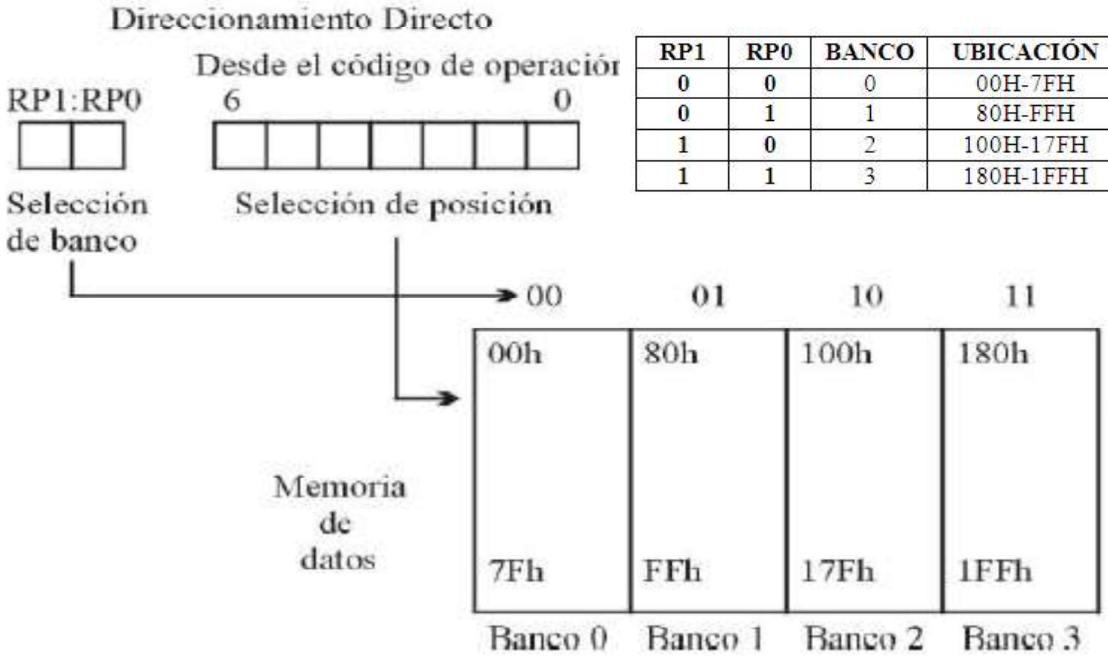


Figura 1.5 Modos de direccionamiento

Direccionamiento directo

La dirección de memoria estará indicada por el valor de las banderas RP1 y RP0 accediendo al banco deseado; así como la dirección de memoria (registro) indicado en la instrucción.



RP1	RP0	BANCO	UBICACIÓN
0	0	0	00H-7FH
0	1	1	80H-FFH
1	0	2	100H-17FH
1	1	3	180H-1FFH

Figura 1.6 Direccionamiento directo

Ejemplo:

```
MOVWF 0X20      ; Mueve el contenido del registro W a la dirección del registro 0x20
INCF 0X20       : Incrementa el contenido del registro 0X20
DECF 0X20       ;Decrementa el contenido del registro 0X20
```

Nota: *En caso que RP0=0 y RP1=0 el contenido de W será enviado a la dirección 0x20 del banco 0; alguna otra codificación posicionará el dato en el banco seleccionado.*

Herramienta de desarrollo MPLAB

El MPLAB es un Ambiente de Desarrollo Integrado ***IDE***, que permite escribir, ensamblar y simular un programa, e incluso usando determinado hardware, se puede simular en circuito y programar al microcontrolador. Este ***IDE*** lo puedes bajar de manera gratuita de la dirección electrónica de Microchip (www.microchip.com).

Al ejecutar MPLAB, presenta una pantalla como la siguiente:

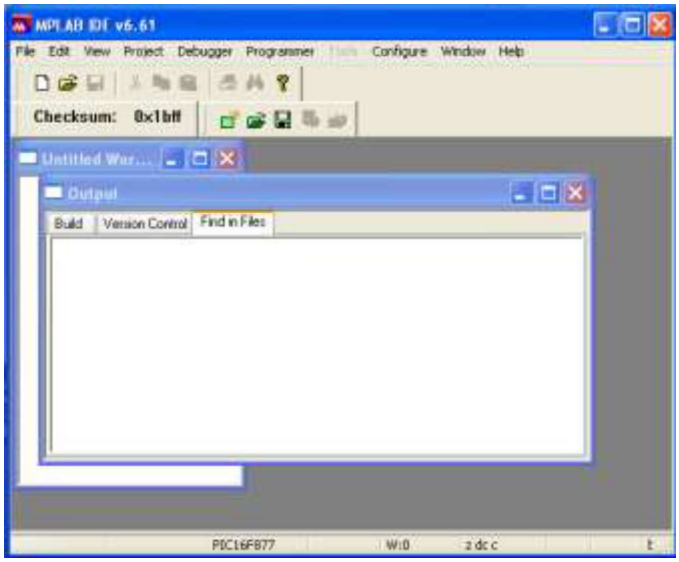


Figura 1.7 Entorno de MPLAB

En el menú **File** seleccionar **New**, entonces aparece la ventana de trabajo con el encabezado **Untitled**, escribir el programa en esta área, una vez terminado, salvarlo usando nuevamente el menú File y el submenú **Save as** del tipo ASM.

Para ensamblar el programa usar el comando **Project**, buscar el submenú **Quickbuild**, donde aparecerá incluido el nombre del programa a ensamblar que es el que está activo en el área de captura.

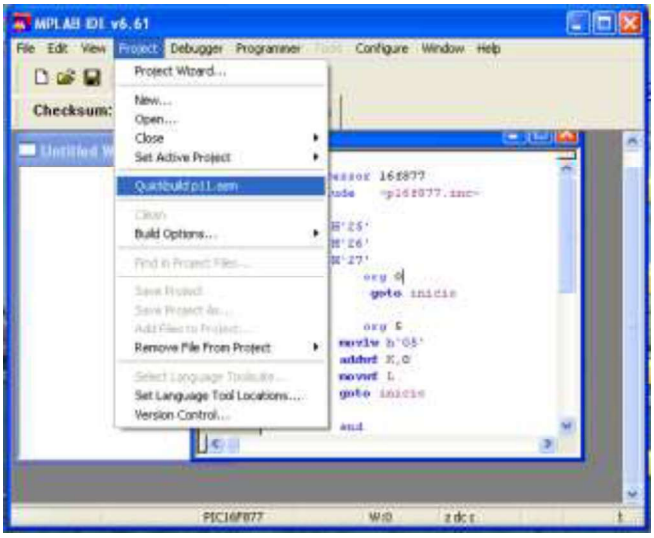


Figura 1.8 Ensamblar un programa

Al no existir errores de sintaxis, se genera el mensaje **BUILD SUCCEEDED**, lo cuál índica que el proceso de ensamblado ha concluido satisfactoriamente.

El siguiente proceso será simular el programa, para lo cuál del menú se elige el comando **View** y las opciones requeridas.

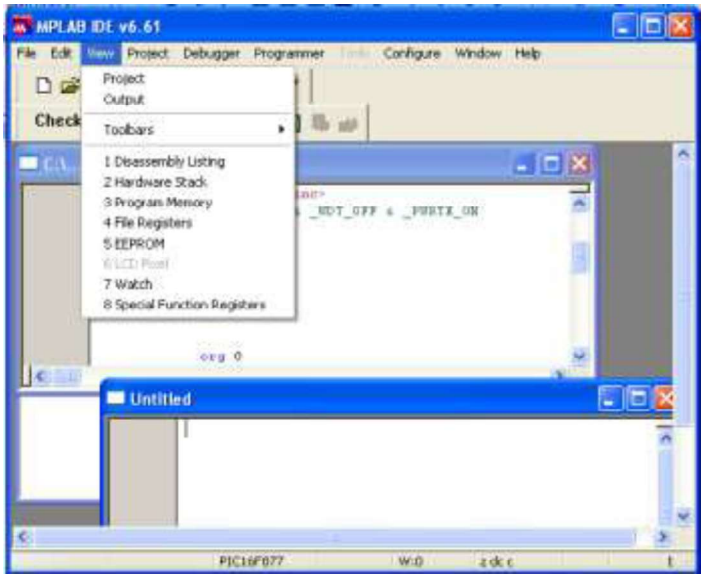


Figura 1.9 Selección de ventanas de visualización para el proceso de simulación

Por lo general solo se selecciona **File Registers**, el cuál muestra los registros y sus valores actuales; para modificar el contenido de alguna localidad, sólo se tiene que escribir el valor deseado y si el programa genera un valor, este será actualizado.

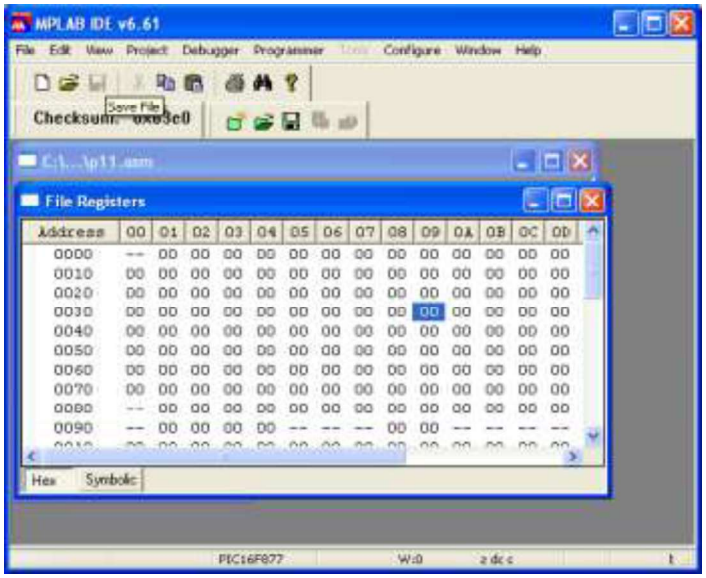


Figura 1.10 Mapa de memoria RAM

Para iniciar el proceso de simulación se debe seleccionar el simulador **MPLAB SIM**, accediendo al menú principal, dar click en **Debugger**, luego seleccionar **Select Tool** y entonces **Mplab Sim**; se habilitarán los iconos de simulación.

Figura 1.11 Iconos de simulación

Permitirá iniciar el proceso de simulación por instrucción o en forma continua, también es posible simular usando teclas de función, acceder al comando Debbuger del menú principal.

Nota: Es importante mencionar que las instrucciones y directivas pueden ser escritas en mayúsculas y minúsculas de manera indistintas, no así las etiquetas y variables que sean declaradas por el alumnos; estas deben ser usadas tal como hayan sido definidas. En el caso de usar las definidas dentro del archivo PIC16F877.INC, deben ser referidas en mayúsculas.

Desarrollo de la práctica 1:

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   0
                GOTO  INICIO

                ORG 5
INICIO:         MOVLW H'05'
                ADDWF  K,0
                MOVWF  L
                GOTO  INICIO
                END
```

Recomendaciones:

a. Las etiquetas y la definición de variables en la primer columna.
b. Las instrucciones deben iniciar a partir de la segunda columna.
c. Las instrucciones y directivas pueden ser en mayúsculas o minúsculas de manera indistintas, no así las variables.

Ingresar un dato de 8 bits al la dirección reservada a la variable **K** y ejecutar la simulación del programa utilizando diferentes valores.

K	L
66	**6B**

2.- Escribir, ensamblar y ejecutar el siguiente programa:

```
        PROCESSOR 16f877
        INCLUDE <p16f877.inc>

K       equ     H'26'
L       equ     H'27'
M       equ     H'28'

                ORG   0
                GOTO  INICIO

                ORG 5
INICIO:         MOVF K,W
                ADDWF  L,0
                MOVWF  M
                GOTO  INICIO
                END
```

a. Comentar e indicar que hace el programa
b. Realizar la ejecución con diferentes valores en K y L
c. Revisar el valor que se genera en la bandera C

K	L	M
50	55	**A5**

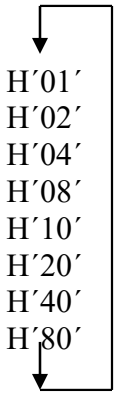
3.- Modificar el programa anterior, para que ahora los datos que operará se encuentren en las localidades **J** y **K** respectivamente, el resultado almacenarlo en otras direcciones, reservadas para **C1** y **R1;** C1 representa el valor de la bandera de acarreo y R1 el resultado.

Un ejemplo de datos y del resultado de la suma es:

J	K	C1	R1
FF	FF	01	FE

4.- Realice un programa que ejecute la siguiente secuencia, misma que deberá ver en la dirección de memoria (registro) de su elección.

Secuencia:



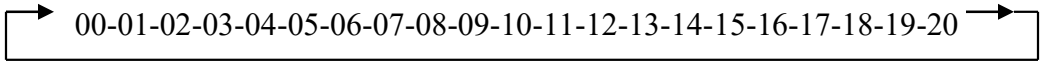
Donde H´01´ indica que el dato esta dado en hexadecimal.

En caso e seleccionar el registro cuya dirección es 0X20

DIR	20	20	20	20	20	20	20	20
DATO	**01**	**02**	**04**	**08**	**10**	**20**	**40**	**80**

Nota: La secuencia indicada, deberá mostrarse en una misma dirección de memoria.

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



Nota: La secuencia indicada, deberá mostrarse en una misma dirección de memoria, tal como fue realizado en el ejercicio anterior.