Microcontroladores

Laboratorio Dirigido N° 1

Actividades:

- Aprender a realizar diagrama de flujo para código ensamblador, mediante ejemplos demostrativos.
- Conocer los registros necesarios para la configuración del microcontrolador.
- Conocer las instrucciones básicas en ensamblador.
- Ejercicios propuestos para el alumno.

Ejemplo 1: Enunciado

Este ejemplo consiste en la siguiente especificación:

Dado dos números X e Y, calcular:

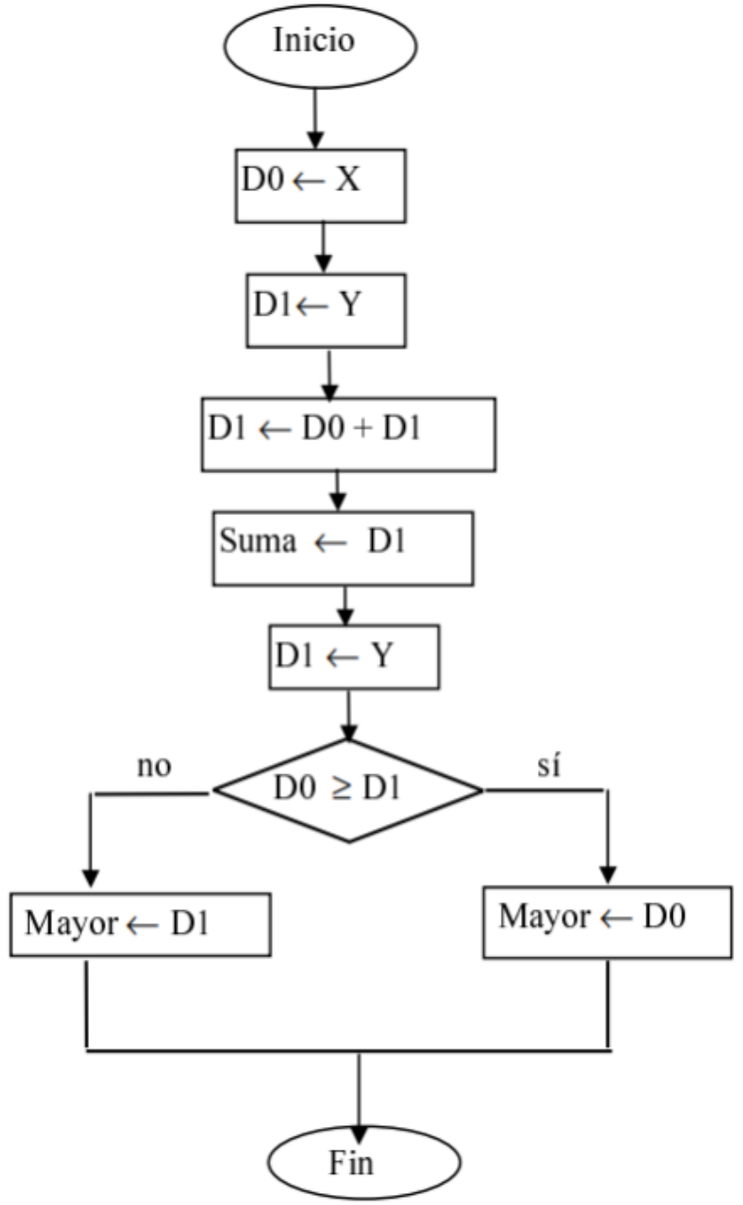
- La suma de ambos números.
- El mayor de ambos números.

Ejemplo 1: Solución

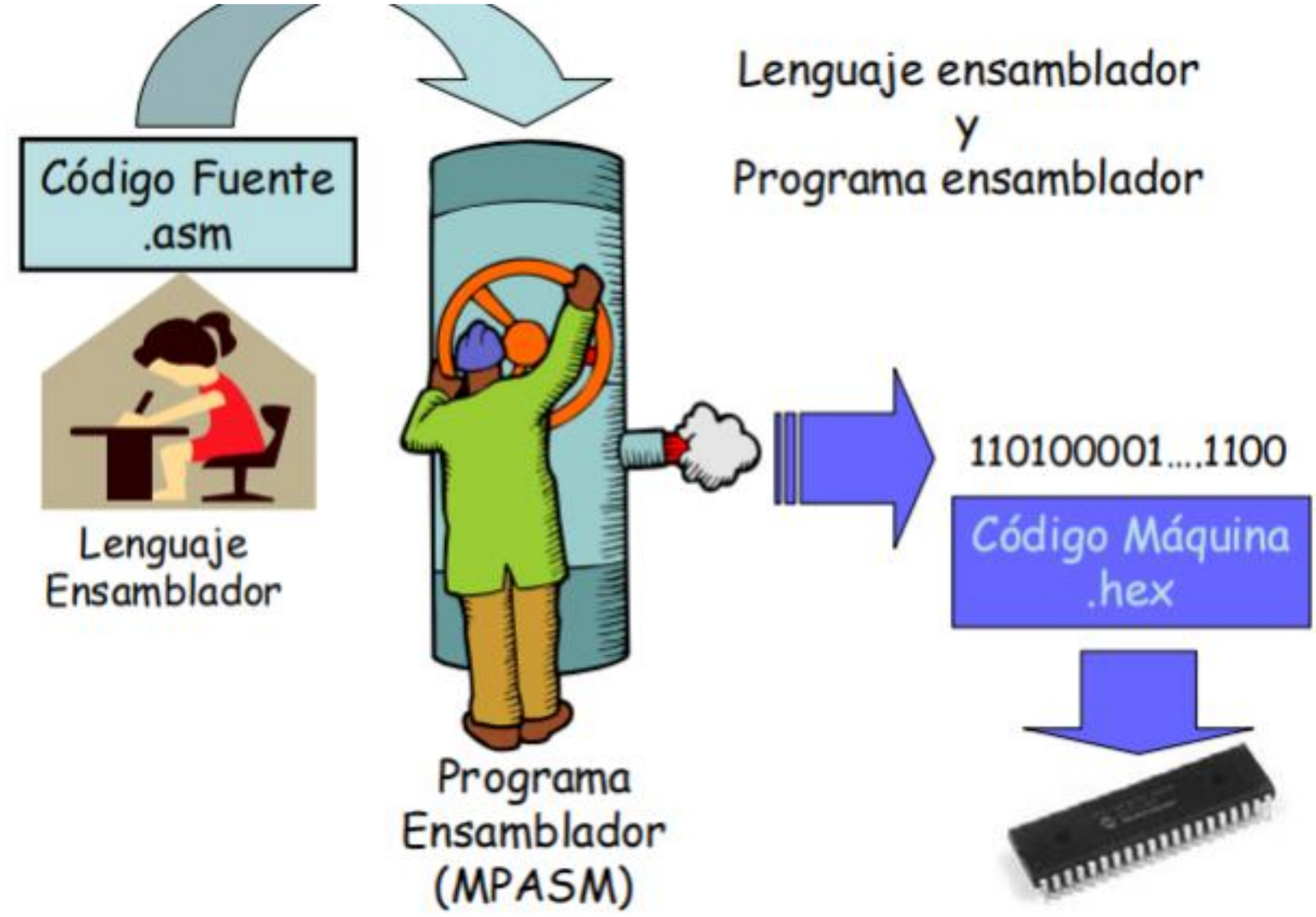
$$Suma = X + Y$$

$$Mayor = \begin{cases} X, \text{ si } X \geq Y \\ Y, \text{ si } X < Y \end{cases}$$

Un programa en ensamblador no se compone únicamente de instrucciones que expresan de forma abstracta el algoritmo que implementa, sino que, al contrario de lo que ocurre en alto nivel, el programador necesita sopesar las distintas opciones que la arquitectura final ofrece: dónde y cómo almacenar variables, cómo manejar datos



1) Programa Ensamblador(MPASM)



2) El código fuente:

- Está compuesto por una sucesión de líneas de texto.

- Cada línea puede estructurarse en hasta cuatro campos o columnas separados por uno o más espacios o tabulaciones entre sí.

  - Campo de etiquetas. Expresiones alfanuméricas escogidas por el usuario para identificar una determinada línea. Todas las etiquetas tienen asignado el valor de la posición de memoria en la que se encuentra el código al que acompañan.
  - Campo de código. Corresponde al nemónico de una instrucción, de una directiva o de una llamada a macro.
  - Campo de operandos y datos. Contiene los operandos que precisa el nemónico utilizado. Según el código, puede haber dos, uno o ningún operando.
  - Campo de comentarios. Dentro de una línea, todo lo que se encuentre a continuación de un punto y coma (;) será ignorado por el programa ensamblador y considerado como comentario.

3) Registros Importantes:

TRISX: REGISTRO DE CONFIGURACIÓN DEL PUERTO X

TRISx7	TRISx6	TRISx5	TRISx4	TRISx3	TRISx2	TRISx1	TRISx0
Bit7	Bit6	Bit5	Bit4	Bit	Bit2	Bit1	Bit0

LATX: REGISTRO DE DATOS DEL PUERTO X

LATx7	LATx6	LATx5	LATx4	LATx3	LATx2	LATx1	LATx0
Bit7	Bit6	Bit5	Bit4	Bit	Bit2	Bit1	Bit0

PORTX: REGISTRO DE ACCESO A LECTURA DEL PUERTO X

PORTx7	PORTx6	PORTx5	PORTx4	PORTx3	PORTx2	PORTx1	PORTx0
Bit7	Bit6	Bit5	Bit4	Bit	Bit2	Bit1	Bit0

INTCON2: REGISTRO DE CONTROL DE INTERRUPCIONES 2

RBPU	INTEDG0	INTEDG1	INTEDG2	-	TMR0IP	-	RBIP
Bit7	Bit6	Bit5	Bit4	Bit	Bit2	Bit1	Bit0

Bit7 **RBPU:** Bit que permite las **resistencias PULL-UP** para el PUERTO B.
1= **Desactiva** las Resistencias **PULL-UP**.
0= **Activa** las resistencias **PULL-UP**.

A continuación se explicará cómo configurar, escribir o leer un Pin de cada **PUERTO**;

- **TRISX:** Escribiendo en cada **bit** ("1" lógico = **Entrada** y "0" lógico = **Salida**).
- **LATX:** Escribiendo en cada **bit** la salida tendrá un **nivel** de Voltaje ("1" lógico = 5v | "0" lógico = 0v).
- **PORTX:** Este registro es de **solo Lectura** (5v ="1" lógico | 0v ="0" lógico).

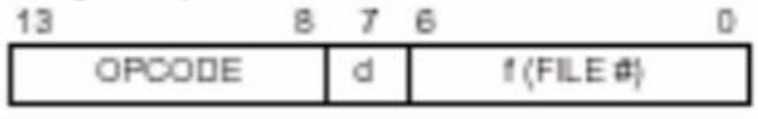
4) Campo de operandos y datos:

TIPO	SINTAXIS		
Decimal	D'<valor>'	d'<valor>'	.<valor>
Hexadecimal	H'<valor>'	h'<valor>'	0x<valor>
	<valor>H	<valor>h	
Octal	O'<valor>'	o'<valor>'	
Binario	B'<valor>'	b'<valor>'	
ASCII	A'<carácter>'	a'<carácter>'	'<carácter>'
Cadena	"<cadena>"		

5) Instrucciones según su formato:

1.- Orientadas al byte

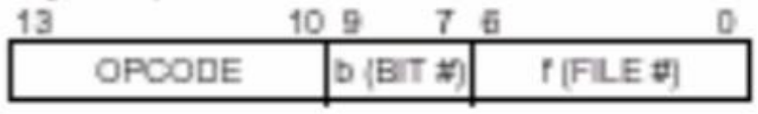
Byte-oriented file register operations



d = 0 for destination W
d = 1 for destination f
f = 7-bit file register address

2.- Orientadas al bit

Bit-oriented file register operations

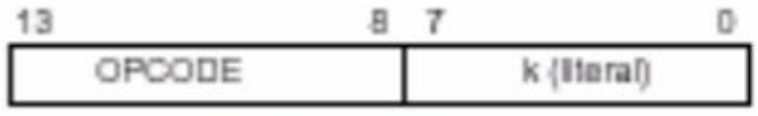


b = 3-bit bit address
f = 7-bit file register address

3.- Literales y de control

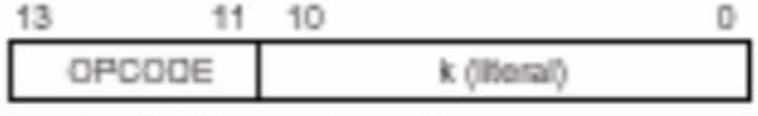
Literal and control operations

General



k = 8-bit literal (immediate) value

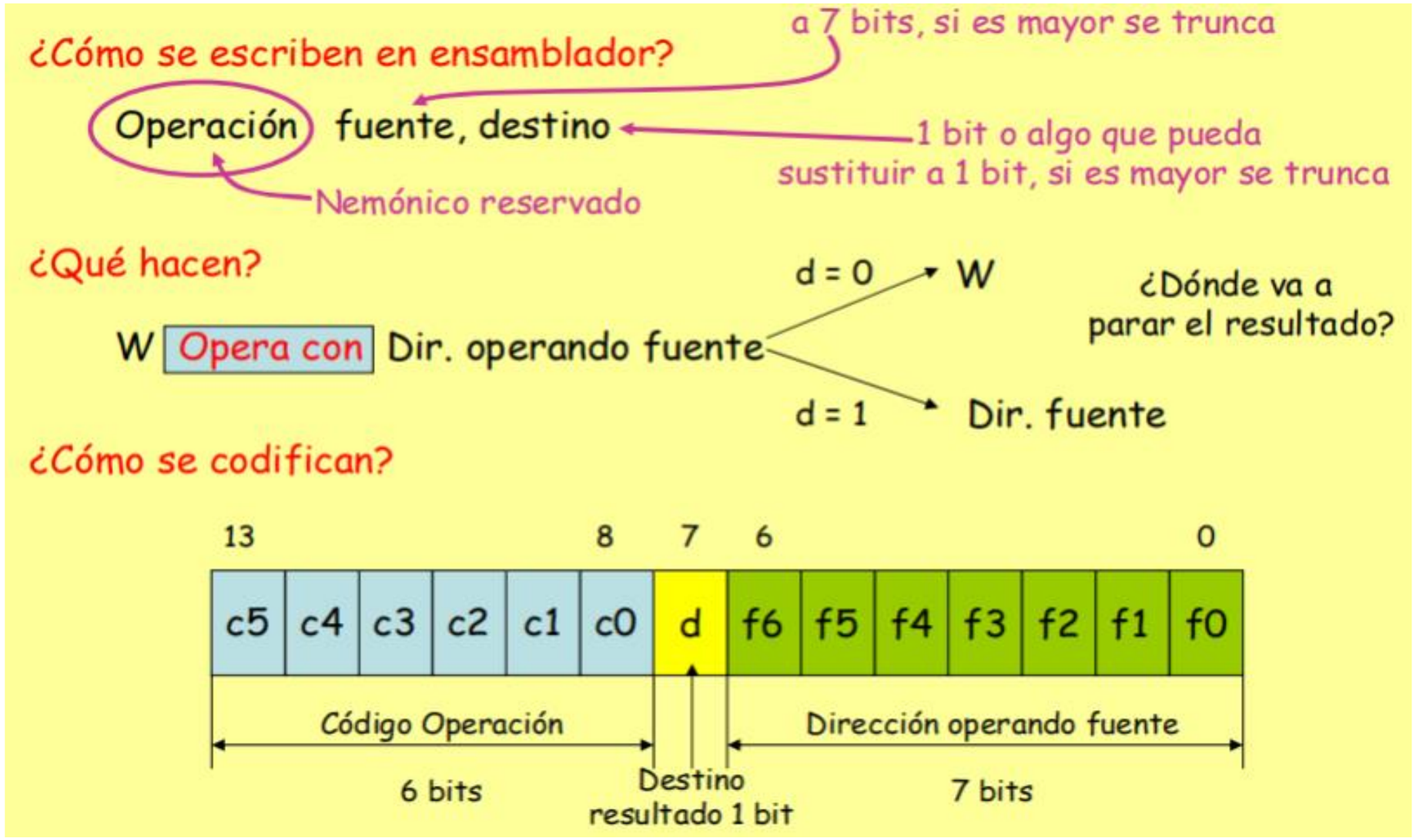
CALL and GOTO instructions only



k = 11-bit literal (immediate) value

5) Instrucciones según su formato:

5.1) Instrucciones orientadas al byte:



5) Instrucciones según su formato:

5.2) Instrucciones orientadas al bit:

¿Cómo se escriben en ensamblador?

Operación fuente, bit

a 7 bits, si es mayor se trunca

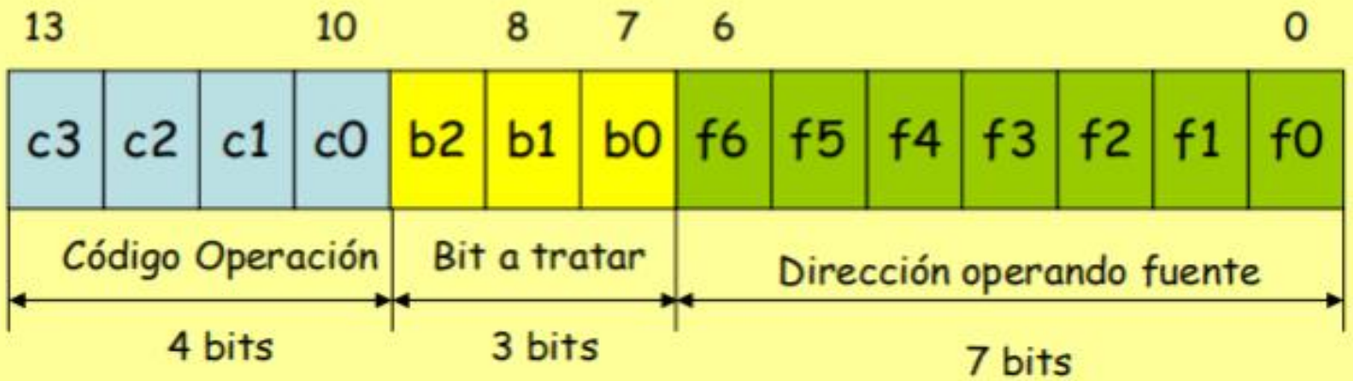
3 bits o algo que pueda sustituir a 3 bits, si es mayor se trunca

Nemónico reservado

¿Qué hacen?

Opera o explora el bit de la posición #bit del operando fuente

¿Cómo se codifican?



5) Instrucciones según su formato:

5.3) Instrucciones literales y de control:

¿Cómo se escriben en ensamblador?

Operación valor literal

Nemónico reservado

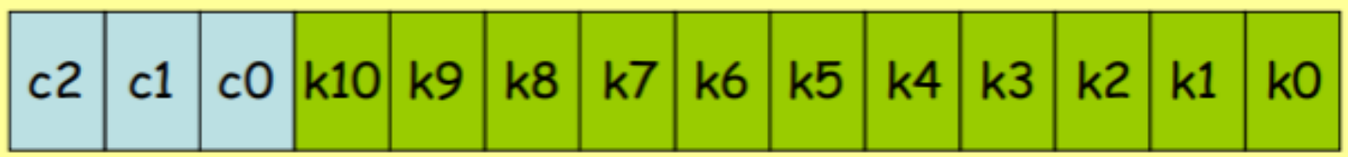
8 bits o algo que pueda sustituir a 8 bits, si es mayor se trunca en GOTO y CALL el valor es de 11 bits

¿Qué hacen?

Operan con el valor literal directamente

¿Cómo se codifican?

En el caso GOTO y CALL



6) Juego de Instrucciones:

En la hoja de datos del PIC18F4550, encontrar:

- Instrucciones orientadas a byte.
- Instrucciones orientadas a bit.
- Instrucciones literales y de control.

6) Juego de Instrucciones:

Instrucciones de CARGA			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
clrf	f	00 → (f)	Z
clrw		00 → (W)	Z
movf	f,d	(f) → (destino)	Z
movlw	k	k → (W)	Ninguno
movwf	f	(W) → (f)	Ninguno

Instrucciones de BIT			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
bcf	f,b	Pone a 0 el bit 'b' del registro 'f'	Ninguno
bsf	f,b	Pone a 1 el bit 'b' del registro 'f'	Ninguno

Instrucciones ARITMÉTICAS			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
addlw	k	(W) + k → (W)	C - DC - Z
addwf	f,d	(W) + (f) → (destino)	C - DC - Z
decf	f,d	(f) - 1 → (destino)	Z
incf	f,d	(f) + 1 → (destino)	Z
sublw	k	K - (W) → (W)	C - DC - Z
subwf	f,d	(f) - (W) → (destino)	C - DC - Z

6) Juego de Instrucciones:

Instrucciones LÓGICAS			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
andlw	k	(W) AND k → (W)	Z
andwf	f,d	(W) AND (f) → (destino)	Z
comf	f,d	(/f) → (destino)	Z
iorlw	k	(W) OR k → (W)	Z
iorwf	f,d	(W) OR (f) → (destino)	Z
rlf	f,d	Rota (f) a izquierda → (destino)	C
rrf	f,d	Rota (f) a derecha → (destino)	C
swap	f,d	Intercambia nibbles (f) → (destino)	Ninguno
xorlw	k	(W) XOR k → (W)	Z
xorwf	f,d	(W) XOR (f) → (destino)	Z

6) Juego de Instrucciones:

Instrucciones de SALTO			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
btfsc	f,b	Salta si el bit 'b' de 'f' es 0	Ninguno
btfss	f,b	Salta si el bit 'b' de 'f' es 1	Ninguno
decfsz	f,d	(f) - 1 → (destino) y salta si es 0	Ninguno
incfsz	f,d	(f) + 1 → (destino) y salta si es 0	Ninguno
goto	k	Salta a la dirección 'k'	Ninguno

Instrucciones de manejo de SUBRUTINAS			
NEMÓNICO		DESCRIPCIÓN	FLAGS AFECTADOS
call	k	Llamada a subrutina	Ninguno
retfie		Retorno de una interrupción	Ninguno
retlw	k	Retorno con un literal en (W)	Ninguno
return		Retorno de una subrutina	Ninguno

6) Juego de Instrucciones:

Instrucciones ESPECIALES		
NEMÓNICO	DESCRIPCIÓN	FLAGS AFECTADOS
clrwdt	Borra Timer del Watchdog	/TO - /PD
nop	No operación	Ninguno
sleep	Entra en modo de bajo consumo	/TO - /PD

Ejemplo 2: Enunciado

Se requiere implementar una compuerta NOT en lenguaje ensamblador, con el PIC18F4550.

- Realizar un diagrama de flujo.

- Realizar un código en ensamblador, basado en el diagrama de flujos.

Ejemplo 3: Enunciado

Se requiere implementar una compuerta AND en lenguaje ensamblador, con el PIC18F4550.

- Realizar un diagrama de flujo.
- Realizar un código en ensamblador, basado en el diagrama de flujos.

Ejercicios Propuestos

Ejercicio Propuestos:

Se tienen las siguientes funciones lógicas:

1) F = AB + C
2) F = $\bar{A}BC$
3) F = $\overline{ABC}$
4) F = A + $\overline{BC}$
5) F = $\bar{A} + \bar{B}C$

Para cada ejercicio propuesto, se pide realizar lo siguiente:

- Realizar un diagrama de flujo, del comportamiento de la función lógica.
- Realizar un código en ensamblador, basado en el diagrama de flujos.

Indicaciones:

Presentar los siguientes archivos:

- Un reporte con los pasos seguidos y el diagrama de flujo en PDF.
- Presentar el Lab1_Apellido.X del programa solicitado.
- Presentar el archivo de la simulación en PROTEUS. (Lab1_Apellido).

La entrega es personal a la actividad creada en el BlackBoard.

Fecha de entrega: Depende de la sección de laboratorio

- Martes entrega Viernes hasta el mediodía
- Miércoles entrega Sábado hasta el mediodía
- Sábado entrega hasta martes al mediodía