

# Las Instrucciones

A continuación vamos a presentar el conjunto de instrucciones básico de los Microcontroladores Picmicro.

En general las instrucciones de los microcontroladores pueden clasificarse como:

## CISC:

*Complex Instruction Set Computer*

Juego de Instrucciones Complejo, más de 80 instrucciones

## RISC

*Reduced Instruction Set Computer*

Juego de Instrucciones Reducido, unas 35 instrucciones. Los microcontroladores PICmicro son de este tipo.

## SISC

*Specific Instruction Set Computer*

Juego de Instrucciones Específico.

Para una mejor presentación las instrucciones se pueden clasificar según la función que desempeñan en un programa, por ejemplo para:

- Mover
- Cambiar el contenido de los registros
- Controlar el flujo del programa
- Controlar el microcontrolador
- Realizar operaciones lógicas
- Realizar operaciones aritméticas.

Las instrucciones realizan operaciones y movimientos entre el "acumulador" o "registro de trabajo" y los registros de la memoria RAM del microcontrolador.

El acumulador está representado con la letra W (Work)

## Instrucciones para mover

### **MOVLW    k**

Carga un número en el acumulador W.

El número que se va a cargar en el acumulador está representado por **k**, este número puede escribirse en decimal, hexadecimal o binario

Ejemplo:

```
MOVLW    d'255'           ; decimal
MOVLW    0xFF             ; hexadecimal
MOVLW    b'11111111'     ; binario
```

### **MOVWF f**

Mueve una copia del acumulador W al registro **f**.

Ejemplo:

```
portb    equ    0x6
Contador  equ    10

MOVLW    0x25           ; carga el acumulador con 0x25
MOVWF    portb          ; el registro portb contiene 0x25
MOVWF    Contador       ; el registro Contador contiene 0x25
```

### **MOVF f,d**

Mueve una copia del registro **f** al destino **d**.

El destino puede ser:

Si **d=0** el destino es el acumulador.

Mueve una copia del registro **f** al Acumulador **W**

Si **d=1** el destino es el registro **f**. En este caso el formato es un poco confuso. Mueve del registro **f** al registro **f**. No pasa nada. El dato se queda en el mismo lugar. No hay porque preocuparse el formato de la instrucción es así. Como veremos más adelante, en otras instrucciones, el formato de destino **d=1** es mas claro. Se puede decir que en general con esta instrucción **d=1** no se utiliza. Siga adelante, entenderá mejor en un momento.

Ejemplo:

```
W        equ    0
f        equ    1
Contador  equ    10

MOVF     Contador,W     ; mueve una copia del contenido del
                       ; Contador al acumulador
```

## **Instrucciones para Cambiar el contenido de los registros**

### **CLRF f**

El contenido del registro **f** se pone en ceros: 0x00

portb equ 0x6

Ejemplo:

CLRF portb ; el registro portb se pone en 0x00

### **CLRWF**

El contenido del acumulador se pone en ceros 0x00

### **COMF f,d**

Complementa el contenido del registro **f**

Los 1 unos los pone en 0, los 0 los pone en 1.

El resultado de esta operación lo coloca en el destino **d**.

Si **d=0** el resultado lo coloca en el acumulador

Si **d=1** el resultado se coloca en el mismo registro **f**

Ejemplo:

W equ 0  
f equ 1  
Contador equ 10

MOVLW b'00001111'  
MOVWF Contador ; Contador contiene b'00001111'  
COMF Contador,W ; el acumulador contiene b'11110000'

MOVLW b'00001111'  
MOVWF Contador ; Contador contiene b'00001111'  
COMF Contador,f ; Contador contiene b'11110000'

### **DECF f,d**

Decrementa el registro **f**

El resultado de esta operación lo coloca en el destino **d**.

Si **d=0** el resultado lo coloca en el acumulador

Si **d=1** el resultado se coloca en el mismo registro **f**

Si el contenido del registro se encuentra en 0xFF y se ejecuta un decremento el resultado es 0x00 .

Ejemplo:

```
W      equ  0
f      equ  1
Contador equ 10
```

```
MOVLW  d'10'
MOVWF  Contador      ; Contador contiene d'10'
DECF   Contador,W    ; El acumulador contiene d'9'
```

```
MOVLW  d'10'
MOVWF  Contador      ; Contador contiene d'10'
DECF   Contador,f    ; Contador contiene d'9'
```

**INCF**      **f,d**

Incrementa el registro **f**

El resultado de esta operación lo coloca en el destino **d**.

Si **d=0** el resultado lo coloca en el acumulador

Si **d=1** el resultado se coloca en el mismo registro **f**

Si el contenido del registro se encuentra en 0xFF y se ejecuta un Incremento el resultado es 0x00 .

Ejemplo:

```
W      equ  0
f      equ  1
Contador equ 10
```

```
MOVLW  d'10'
MOVWF  Contador      ; Contador contiene d'10'
INCF   Contador,W    ; El acumulador contiene d'11'
```

```
MOVLW  d'10'
MOVWF  Contador      ; Contador contiene d'10'
INCF   Contador,f    ; Contador contiene d'11'
```

**BCF**      **f,b**

Pone en cero el bit **b** del file **f** .

Los bits del registro **f** se numeran de 0 a 7.

Ejemplo:

```
Contador    equ    10

MOVLW      b'11111111'
MOVWF      Contador    ; Contador b'11111111'
BCF        Contador,0    ; Contador b'11111110'
BCF        Contador,7    ; Contador b'01111110'

BSF        f,b
```

Pone en uno el bit **b** del file **f** .

Ejemplo:

```
Contador    equ    10

MOVLW      b'00000000'
MOVWF      Contador    ; Contador b'00000000'
BSF        Contador,0    ; Contador b'00000001'
BSF        Contador,7    ; Contador b'10000001'

RLF        f,d
```

Rota el contenido del registro **f** una posición a la izquierda.

El bit se rota a través de la bandera “carry” . La bandera carry esta en el bit 0 del registro STATUS, que se localiza en la posición 0x03 de la memoria del microcontrolador. Cada vez que se ejecuta la instrucción los bits se rotan una posición a la izquierda, el bit menos significativo es ocupado por el contenido de la bandera carry y el bit más significativo pasa a ocupar el lugar de la bandera carry.

El resultado de la operación se coloca en el destino **d**

Ejemplo:

```
W          equ    0
f          equ    1

STATUS     equ    0x03
C          equ    0

Contador   equ    10

BCF        STATUS,0    ; carry a 0
```

```

MOVLW    b'11111111'
MOVWF    Contador          ; Contador b'11111111'

RLF      Contador,f        ; Contador b'11111110'

RRF      f,d

```

Rota el contenido del registro **f** una posición a la derecha

El bit se rota a través de la bandera “carry”. Cada vez que se ejecuta la instrucción los bits se rotan una posición a la derecha, el bit más significativo es ocupado por el contenido de la bandera carry y el bit menos significativo pasa a ocupar el lugar de la bandera carry.

El resultado de la operación se coloca en el destino **d**

Ejemplo:

```

W        equ 0
f        equ 1

STATUS   equ 0x03
C        equ 0

Contador equ 10

BCF      STATUS,0        ; carry a 0

MOVLW    b'11111111'
MOVWF    Contador        ; Contador b'11111111'

RRF      Contador,f      ; Contador b'01111111'

```

**SWAPF f,d**

Intercambia el nibble más significativo y el nibble menos significativo.

El resultado de la operación se coloca en el destino **d**.

Ejemplo:

```

W        equ 0
f        equ 1

Contador equ 10

```

```

MOVLW    b'11110000'
MOVWF    Contador          ; Contador b'11110000'
SWAPF    Contador          ; Contador b'00001111'

```

### Instrucciones para controlar el flujo del programa.

**GOTO k**  
Salta a la etiqueta **k**

Ejemplo:

```
Contador equ 10
```

```

MOVLW    b'00000000'
MOVWF    Contador          ; Contador b'00000000'
GOTO     Programa1        ; El programa continua en Programa1
-----
-----
-----

```

Programa1

```

BSF      Contador,0        ; Contador b'00000001'
BSF      Contador,7        ; Contador b'10000001'

```

**CALL k**  
Salta a una subrutina en la etiqueta **k**

Ejemplo:

```
Contador equ 10
Contador1 equ 11
```

```

BSF      Contador,0        ; Contador b'00000001'
BSF      Contador,7        ; Contador b'10000001'

```

```

CALL     CargaContador    ; ejecuta la subrutina CargaContador
----- ; aquí sigue después de subrutina
-----
-----
-----

```

CargaContador

```

MOVLW    b'00000000'
MOVWF    Contador          ; Contador b'00000000'

```

Return

**RETURN**

Regresa de una subrutina

**RETLW k**

Regresa de una subrutina. Con el numero **k** en el acumulador.

**RETFIE**

Regresa de una interrupción.

**BTFSC f,b**

Prueba el bit **b** del registro **f** .

Salta la siguiente instrucción si bit b es 0.

Ejemplo:

Entrada equ 11

Switch equ 0

ChecaEISwitch

BTFSC Entrada, Switch ; Checa que Switch este en 0

GOTO ChecaEISwitch ; El Switch esta en 1, regresa

----- ; El Switch ya esta en 0

-----

-----

**BTFSS f,b**

Prueba el bit **b** del registro **f** .

Salta la siguiente instrucción si bit b es 1.

Ejemplo:

Entrada equ 11

Switch equ 0

ChecaEISwitch

BTFSS Entrada, Switch ; Checa que Switch este en 1

GOTO ChecaEISwitch ; El Switch esta en 0, regresa

----- ; El Switch ya esta en 1

-----

-----

**DECFSZ f,d**

Decrementa el registro **f**

Salta la siguiente instrucción si el resultado es 0

El resultado de la operación se coloca en el destino **d**.



Ejemplo:

```
W      equ  0
f      equ  1

Contador equ  10

MOVLW  d'10'
MOVWF  Contador ; Contador en d'10'
```

Contando

```
-----
-----
DECFSZ Contador,f ; decrementa Contador
GOTO    Contando  ; Contador mayor que 0 decrementa de nuevo
-----
; Contador en 0
-----
```

### **INCFSZ f,d**

Incrementa el registro **f**

Salta la siguiente instrucción si el resultado es 0

El resultado de la operación se coloca en el destino **d**

### **NOP**

Esta instrucción no hace nada durante un ciclo.

Se puede ocupar para realizar retardos.

### **Instrucciones para controlar el microcontrolador.**

#### **CLRWDT**

Pone a 0 el temporizador Watchdog.

#### **OPTION**

El contenido del acumulador se envía al registro OPTION

#### **SLEEP**

Pone el microcontrolador en SLEEP (dormir) para reducir el consumo

#### **TRIS f**

El contenido del acumulador determina las Entradas Salidas el Puerto **f**.

Ejemplo:

Portb equ 0x06

MOVLW b'00000000'

TRIS Portb ; Todo el Puerto B como salidas

### Instrucciones para realizar operaciones lógicas

#### **ANDLW k**

AND el acumulador y el numero k.

Resultado en el acumulador.

Ejemplo

MOVLW b'00001111' ; acumulador b'00001111'

ANDLW b'00000001' ; acumulador b'00000001'

#### **ANDWF f,d**

AND el contenido del acumulador con el registro f.

El resultado de la operación se coloca en el destino **d**.

#### **IORWF k**

OR el acumulador y el numero k.

Resultado en el acumulador.

#### **IORWF f,d**

OR el contenido del acumulador con el registro f.

El resultado de la operación se coloca en el destino **d**.

#### **XORLW k**

XOR el acumulador y el numero k.

Resultado en el acumulador.

#### **XORWF f,d**

XOR el contenido del acumulador con el registro f.

El resultado de la operación se coloca en el destino **d**.

### Instrucciones para realizar operaciones aritméticas

#### **ADDWF f,d**

Suma el contenido de **W** con el contenido del registro **f**

#### **ADDLW k**

Suma el contenido del acumulador **W** con el numero **k**

**SUBLW     k**  
Realiza la resta **k – W**

**SUBWF     f,d**  
Ejecuta la resta **f – W**

Como comentario final a esta breve presentación de las instrucciones básicas de los microcontroladores PICmicro cabe hacer notar que no basta con conocerlas, leerlas o memorizarlas, hay que poner manos a la obra y ejercitarse en su uso. La mejor manera de aprender a usarlas es practicando muchas veces, cometer errores y aprender de ellos. Así que a practicar, abra MPLAB y diviértase un buen rato. Para aprender a programar es necesario conocer las instrucciones pero conocer las instrucciones no es saber programar, pero vamaos por buen camino.

