Microcontroladores

Set de Instrucciones

Contenido

- Conocer el ISA del microcontrolador PIC18F4550.
- Conocer el funcionamiento del registro trabajo (W).
- Aprender los tipos de instrucciones básicas en ASM.

1.- Introducción

Los PIC18F4550 construyen el conjunto estándar de 75 instrucciones PIC18, así como un conjunto extendido de ocho nuevas instrucciones para la optimización del código que es recursiva o que utiliza una pila de software.

- **SET DE INSTRUCCIONES ESTÁNDAR:**

La mayoría de las instrucciones son un solo programa con memoria de una palabra (16 bits), pero hay cuatro instrucciones que requieren dos posiciones de memoria de programa.

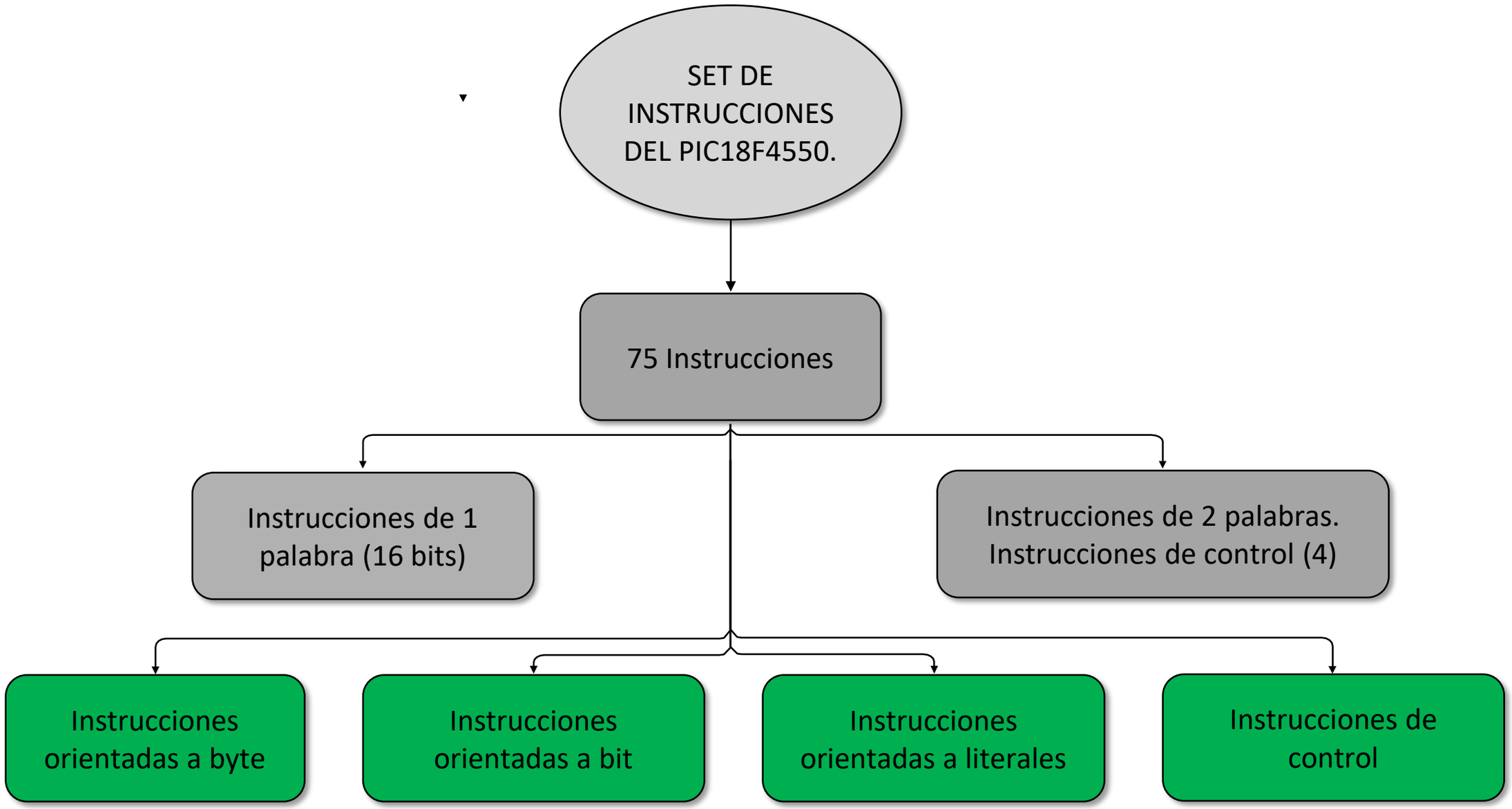
Cada instrucción de una sola palabra es una palabra de 16 bits dividida dentro de un código de operación (OPCODE), que especifica el tipo de instrucción y uno o más operandos, que además especifican la operación de la instrucción.

2.- Tipos de Instrucciones

El conjunto de instrucciones se agrupa en cuatro categorías básicas:

- Operaciones orientadas a byte.
- operaciones orientadas a bit.
- Operaciones literales.
- Las operaciones de control.

2.- Tipos de Instrucciones



2.- Tipos de Instrucciones

Instrucciones orientadas a byte.

3 OPERANDOS:

- Registro de archivo (f)
- Destino de resultado (d)
- Memoria Accesada (a)

El designador del registro de archivos ‘f’ especifica que archivo de registro es utilizado por la instrucción.

El designador de destino “d” especifica donde se va a colocar el resultado de la operación a realizar.

D = 0
El resultado se almacena en WREG.

D = 1
El resultado se coloca en el archivo que aparece en la instrucción

2.- Tipos de Instrucciones

2.1.- Instrucciones Orientadas a Bit:

Todas las instrucciones orientadas a bits tienen tres operandos:

- El registro de archivo (especificado por 'f')
- Los bits en el registro de archivos (especificados por "b")
- La memoria a la cual se tiene acceso (especificado por "a")

El designador de campo de bits 'b' selecciona el número del bit afectados por la operación, mientras que el registro de archivo designador 'f' Representa el número del archivo en donde el bit se encuentra.

2.- Tipos de Instrucciones

2.2.- Instrucciones Orientadas a Literales:

- Un valor literal que se carga en un fichero de registro (especificado por 'k’).
- El registro FSR deseado para cargar el valor literal en (especificado por 'f’).
- No se requiere operando (especificado por '-')

2.- Tipos de Instrucciones

2.3.- Instrucciones Orientadas a Control:

- Una dirección de memoria de programa (especificado por ‘n’).
- El modo de las instrucciones CALL o Retorno (especificado por 's’)
- El modo de la mesa de lectura y escritura de mesa instrucciones (especificados por 'm’)
- No se requiere operando (especificado por '-')

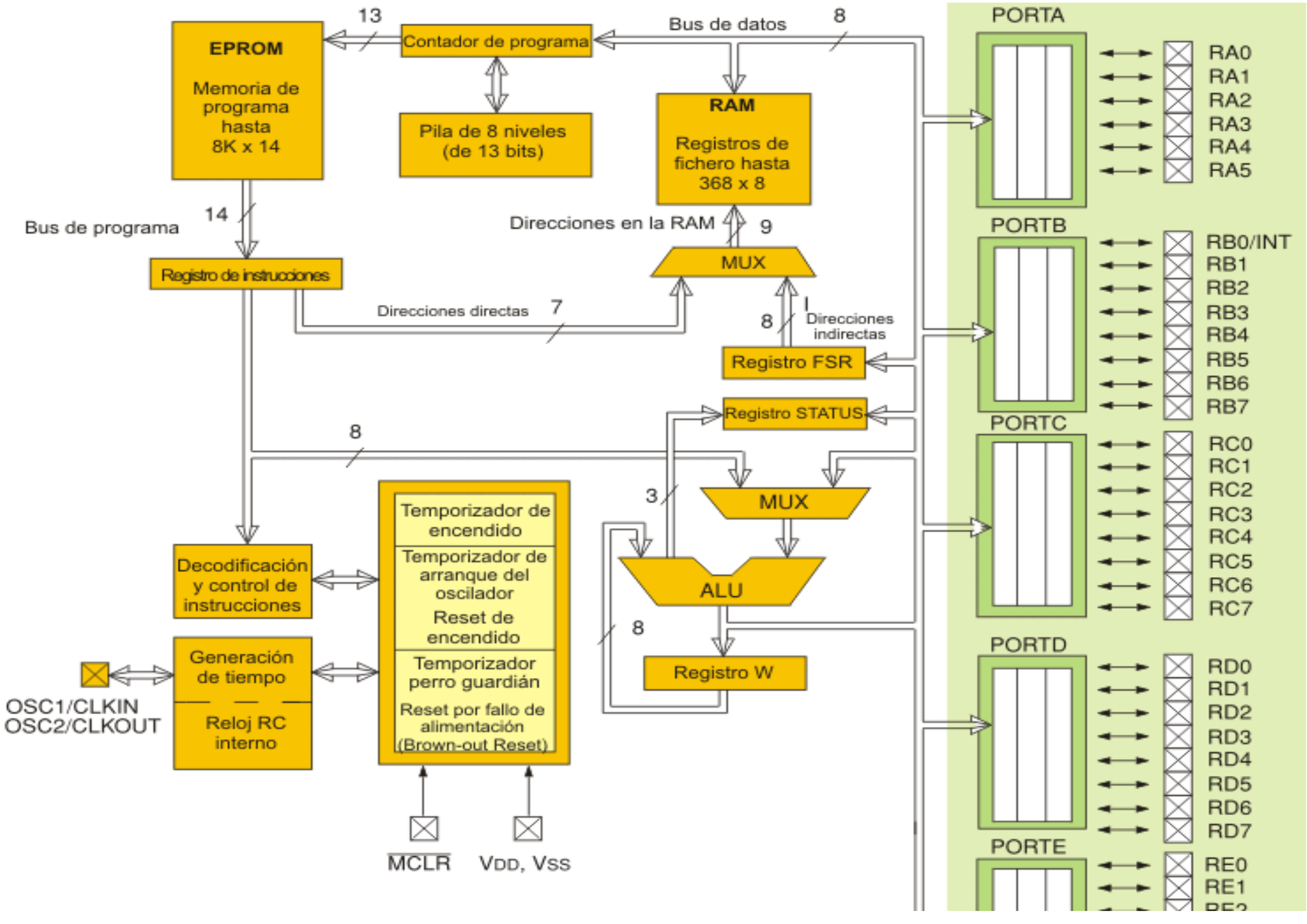
3.- Características de las Instrucciones

- Todas las instrucciones son de palabra única, excepto en cuatro instrucciones es de palabra doble.
- Estas instrucciones fueron hechas a doble palabra para contener la información requerida en 32 bits.
- Todas las instrucciones de palabra única se ejecutan en un solo ciclo de instrucción, menos una prueba condicional que es verdadera o cuando el contador de programa cambia como resultado de la instrucción.

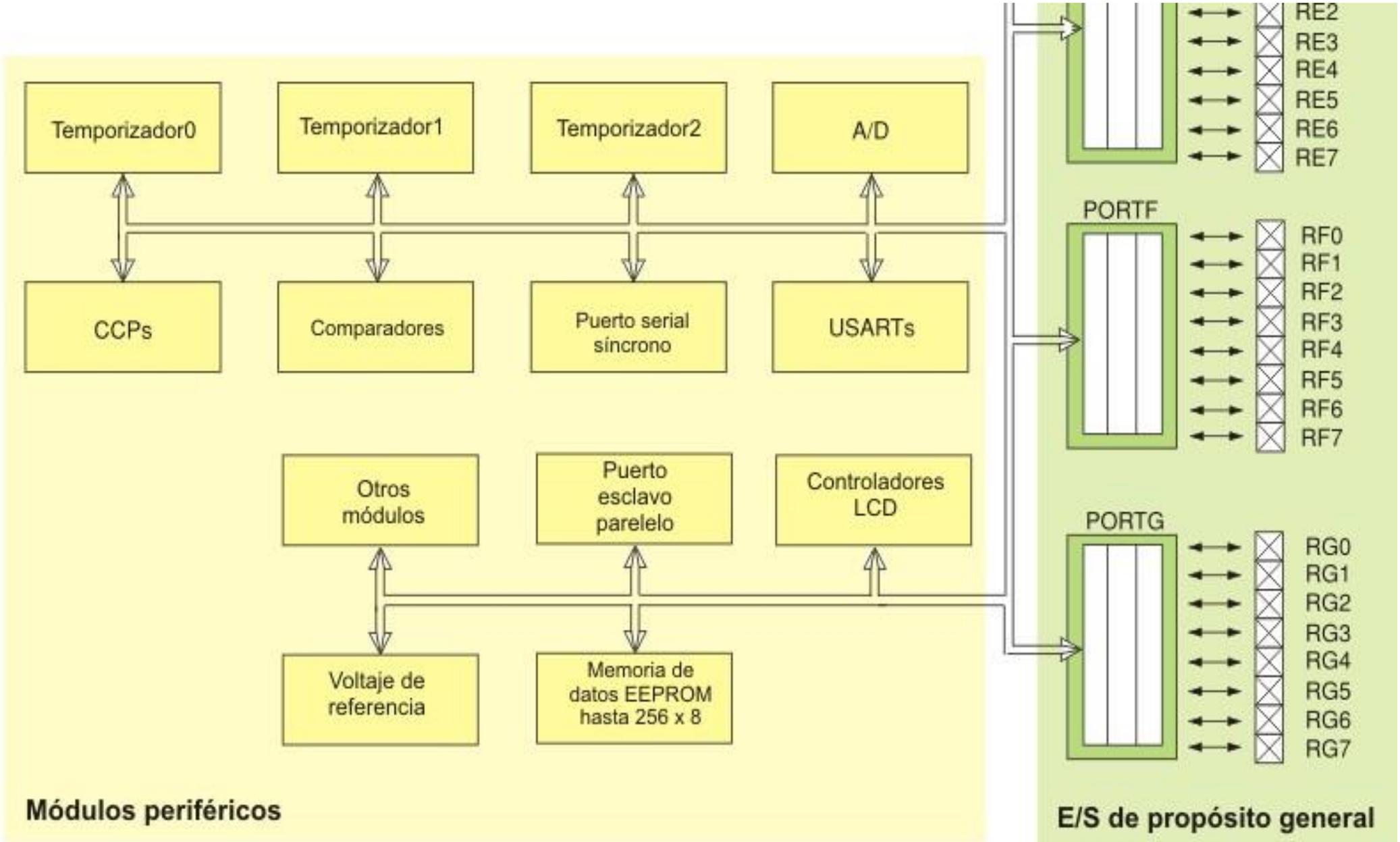
3.- Características de las Instrucciones

- Un ciclo de instrucción consta de cuatro períodos de oscilador.
- Por lo tanto, para un oscilador de frecuencia de 4 MHz, el tiempo normal de ejecución de la instrucción es de 1us.
- Si una prueba condicional resulta verdadero , o el contador de programa cambia como resultado de la ejecución de una instrucción, el tiempo de ejecución de la instrucción es de 2 us.
- Las instrucciones de salto (branch) de doble palabra (si es verdadera) tomarían 3 uS

4.- Diagrama modular del PIC



4.- Diagrama modular del PIC



5.- Descripción del FILE OPCODE

Field	Description
a	RAM access bit a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
C, DC, Z, OV, N	ALU Status bits: Carry, Digit Carry, Zero, Overflow, Negative.
d	Destination select bit d = 0: store result in WREG d = 1: store result in file register f
dest	Destination: either the WREG register or the specified register file location.
f	8-bit register file address (00h to FFh) or 2-bit FSR designator (0h to 3h).
f_s	12-bit register file address (000h to FFFh). This is the source address.
f_d	12-bit register file address (000h to FFFh). This is the destination address.
GIE	Global Interrupt Enable bit.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value).
label	Label name.
mm * *+ *- +*	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: No change to register (such as TBLPTR with table reads and writes) Post-Increment register (such as TBLPTR with table reads and writes) Post-Decrement register (such as TBLPTR with table reads and writes) Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions or the direct address for Call/Branch and Return instructions.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.

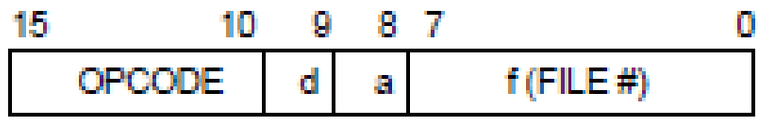
5.- Descripción del FILE OPCODE

Todas las instrucciones llevan alguno, dependiendo de la función que desempeñen:

- Instrucciones orientadas a byte.
- Instrucciones orientadas a bit.
- Instrucciones de manejo de constantes (en la literatura de Microchip, las constantes se denominan “literals”) y, finalmente instrucciones de salto CALL y GOTO.
- El trabajo de decodificación de cada formato es realizado por el programa ensamblador, razón por la cual el programador no requiere de decodificar manualmente cada instrucción.

6.- Formato General-Instrucciones

Byte-oriented file register operations

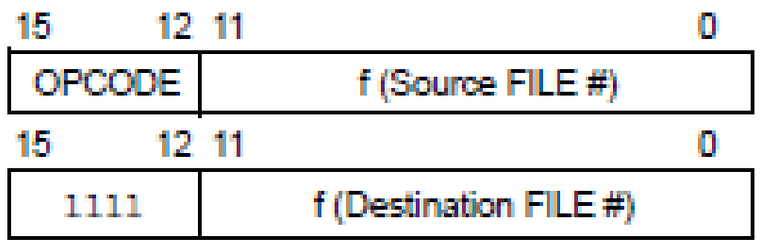


d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Example Instruction

```
ADDWF MYREG, W, B
```

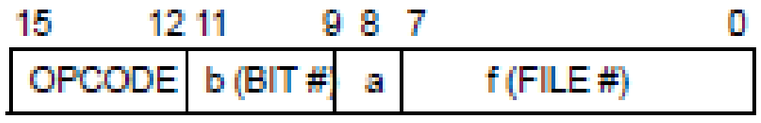
Byte to Byte move operations (2-word)



f = 12-bit file register address

```
MOVFF MYREG1, MYREG2
```

Bit-oriented file register operations



b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

```
BSF MYREG, bit, B
```

6.- Formato General-Instrucciones

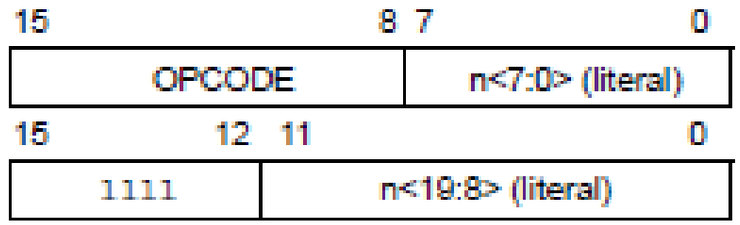
Literal operations

k = 8-bit immediate value

```
MOVLW 7Fh
```

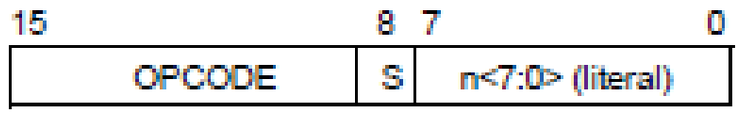
Control operations

CALL, GOTO and Branch operations



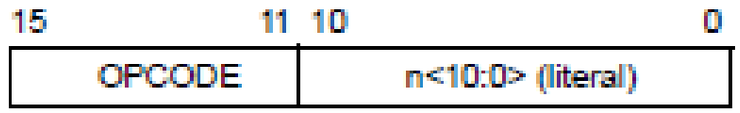
n = 20-bit immediate value

```
GOTO Label
```

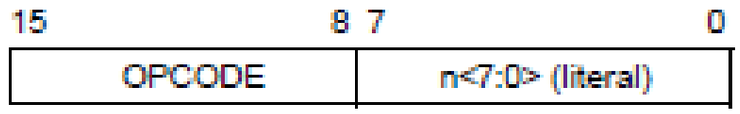


S = Fast bit

```
CALL MYFUNC
```




```
BRA MYFUNC
```




```
BC MYFUNC
```

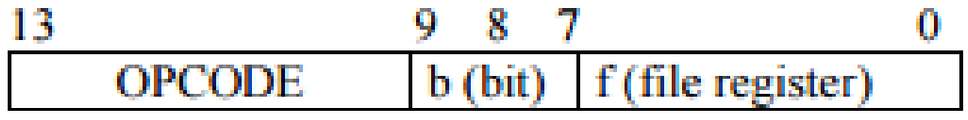
6.- Formato General- Instrucciones

INSTRUCCIONES ORIENTADAS A BYTES:

Si d=0, el destino es w, si d=1, el destino es f

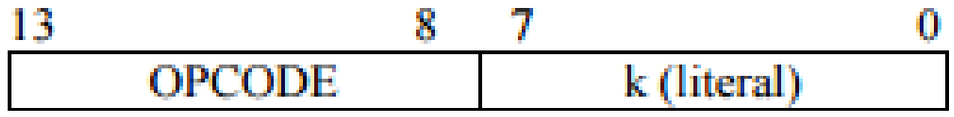
INSTRUCCIONES ORIENTADAS A BITS:

b selecciona el bit del registro f, (valor de 0 a 7)



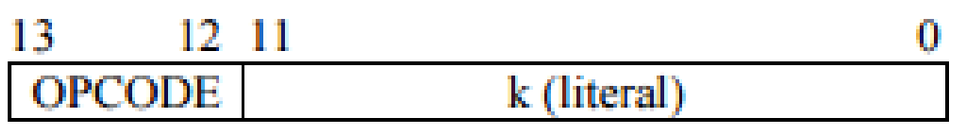
INSTRUCCIONES MANEJO DE CONSTANTES:

k es la constante en la instrucción.



INSTRUCCIONES CALL Y GOTO:

k es la dirección inmediata en 12 bits.



7.- Operaciones orientadas a BYTES

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
BYTE-ORIENTED OPERATIONS									
ADDWF	f, d, a	Add WREG and f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	4
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	4
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	None	
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	1, 2
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	1, 2
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	1, 2
SUBFWB	f, d, a	Subtract f from WREG with borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	Subtract WREG from f with borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	

8.- Operaciones orientadas a BIT

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
BIT-ORIENTED OPERATIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, d, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2
CONTROL OPERATIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	1 (2)	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	2	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call subroutine 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C	
GOTO	n	Go to address 1st word	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx	None	4
POP	—	Pop top of return stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push top of return stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software device Reset	1	0000	0000	1111	1111	All	
RETFIE	s	Return from interrupt enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

9.- Operaciones LITERAL.

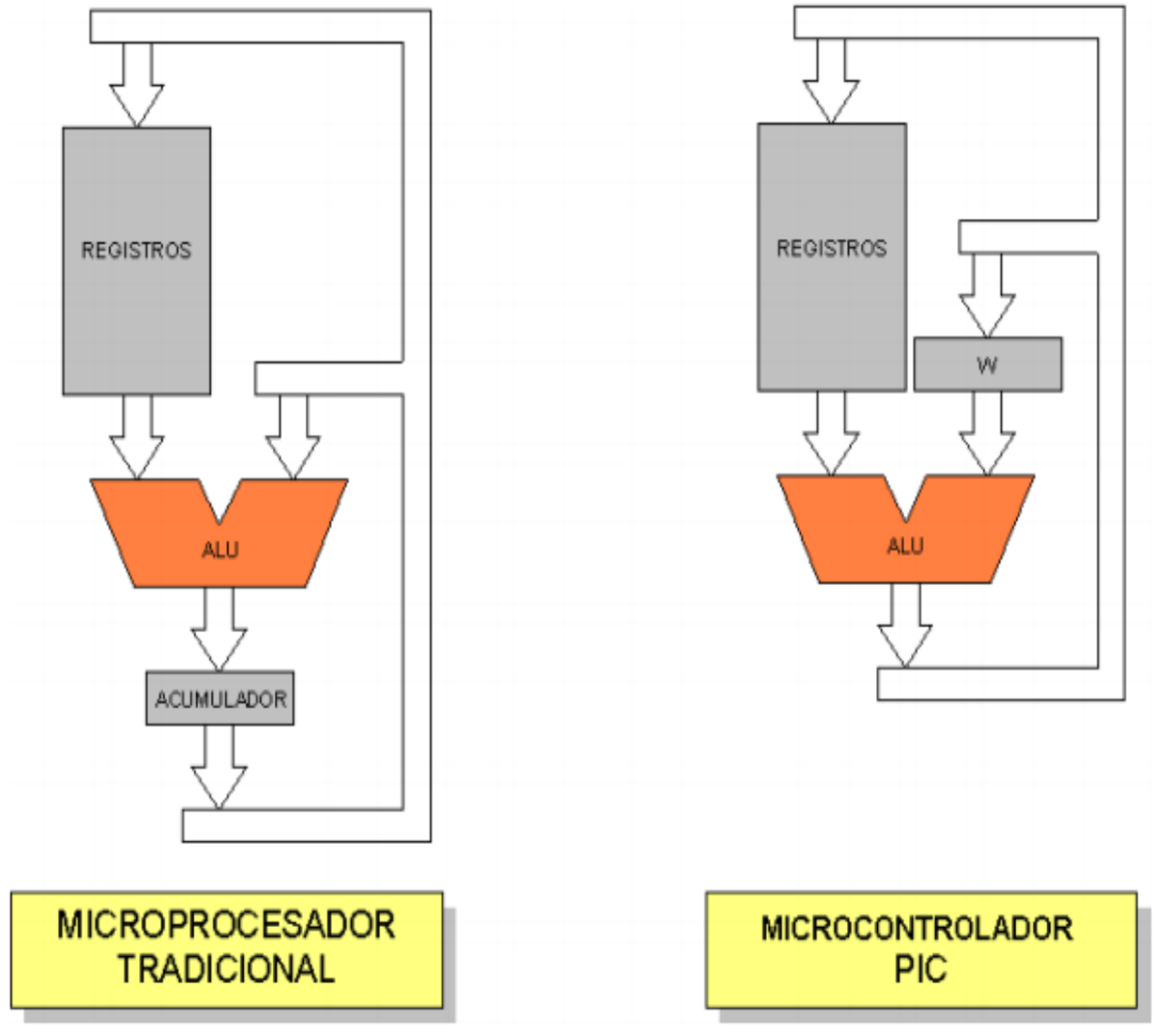
Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS						
ADDLW	k	Add literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW	k	AND literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW	k	Inclusive OR literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR	f, k	Move literal (12-bit) 2nd word to FSR(f) 1st word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB	k	Move literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW	k	Move literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW	k	Multiply literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW	k	Return with literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW	k	Subtract WREG from literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR literal with WREG	1	0000 1010 kkkk kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS						
TBLRD*		Table Read	2	0000 0000 0000 1000	None	
TBLRD*+		Table Read with post-increment		0000 0000 0000 1001	None	
TBLRD*-		Table Read with post-decrement		0000 0000 0000 1010	None	
TBLRD+*		Table Read with pre-increment		0000 0000 0000 1011	None	
TBLWT*		Table Write	2	0000 0000 0000 1100	None	
TBLWT*+		Table Write with post-increment		0000 0000 0000 1101	None	
TBLWT*-		Table Write with post-decrement		0000 0000 0000 1110	None	
TBLWT+*		Table Write with pre-increment		0000 0000 0000 1111	None	

10.- Registro de Trabajo (W)

- Es importante recordar que el PIC18F4550 opera con una arquitectura HARVARD.
- Distinto manejo de tablas que en una VON NEUMANN, en donde la memoria de código y datos es compartida, y entonces, a través de direccionamiento indirecto es posible recuperar los datos de la tabla.
- En el caso de la arquitectura Harvard, la tabla está en la memoria de código y debe entonces de manejarse como parte del programa ejecutable.

10.- Registro de Trabajo (W)

- En los microcontroladores tradicionales todas las operaciones se realizan sobre el acumulador.
- La salida del acumulador esta conectada a una de las entradas de la Unidad Aritmética y Lógica (ALU), y por lo tanto éste es siempre uno de los dos operandos de cualquier instrucción.



10.- Registro de Trabajo (W)

- Por convención, las instrucciones de simple operando (borrado, incremento, decremento, complemento), actúan sobre el acumulador.
- La salida de la ALU va solamente a la entrada del acumulador, por lo tanto el resultado de cualquier operación siempre quedará en este registro.
- Para operar sobre un dato de memoria, luego realizar la operación siempre hay que mover el acumulador a la memoria con una instrucción adicional.

10.- Registro de Trabajo (W)

- En los microcontroladores PIC, la salida de la ALU va al registro W y también a la memoria de datos, por lo tanto el resultado puede guardarse en cualquiera de los dos destinos.
- En las instrucciones de doble operando, uno de los dos datos siempre debe estar en el registro W, como ocurría en el modelo tradicional con el acumulador.
- En las instrucciones de simple operando el dato en este caso se toma de la memoria (también por convención).

10.- Registro de Trabajo (W)

La gran ventaja de la nueva arquitectura del acumulador es que permite un gran ahorro de instrucciones ya que el resultado de cualquier instrucción que opere con la memoria, ya sea de simple o doble operando, puede dejarse en la misma posición de memoria o en el registro W, según se seleccione con un bit de la misma instrucción.

Las operaciones con constantes provenientes de la memoria de programa (literales) se realizan solo sobre el registro W.

11.- Ejemplo de Instrucciones

ADDLW Suma un literal

Sintaxis: [label] ADDLW k
Operandos: $0 \le k \le 255$
Operación: : (W) + (k)⇒ (W)
Flags afectados: C, DC, Z
Código OP: 11 111x kkkk kkkk

Descripción: Suma el contenido del registro W y k, guardando el resultado en W.

Ejemplo: ADDLW 0xC2

Antes: W = 0x17
Después: W = 0xD9

ANDWF W AND F

Sintaxis: [label] ANDWF f,d
Operandos: $d \in [0,1]$, $0 \le f \le 127$
Operación: (W) AND (f) ⇒ (dest)
Flags afectados: Z
Código OP: 00 0101 dfff ffff

Descripción: Realiza la operación lógica AND entre los registros W y f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : ANDWF REG,0

Antes: W = 0x17., REG = 0xC2
Después: W = 0x17, REG = 0x02

CALL Salto a subrutina

Sintaxis: [label] CALL k
Operandos: $0 \le k \le 2047$
Operación: PC ⇒ Pila; k ⇒ PC
Flags afectados: Ninguno
Código OP: 10 0kkk kkkk kkkk

Descripción: Salto a una subrutina. La parte baja de k se carga en PCL, y la alta en PCLATCH. Ocupa 2 ciclos de reloj.

Ejemplo:ORIGEN CALL DESTINO

Antes: PC = ORIGEN
Después: PC = DESTINO

COMF Complemento de f

Sintaxis: [label] COMF f,d
Operandos: $d \in [0,1]$, $0 \le f \le 127$
Operación: : (/ f), 1 ⇒ (dest)
Flags afectados: Z
Código OP: 00 1001 dfff ffff

Descripción: El registro f es complementado. El flag Z se activa si el resultado es 0. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f..

Ejemplo: : COMF REG,0

Antes: REG = 0x13
Después: REG = 0x13, W = 0XEC

DECF Decremento de f

Sintaxis: [label] DECF f,d
Operandos: $d \in [0,1]$, $0 \le f \le 127$
Operación: : (f) – 1 ⇒ (dest)
Flags afectados: Z
Código OP: 00 0011 dfff ffff

Descripción: Decrementa en 1 el contenido de f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : DECF CONT,1

Antes: CONT = 0x01, Z = 0
Después: CONT = 0x00, Z = 1

GRACIAS