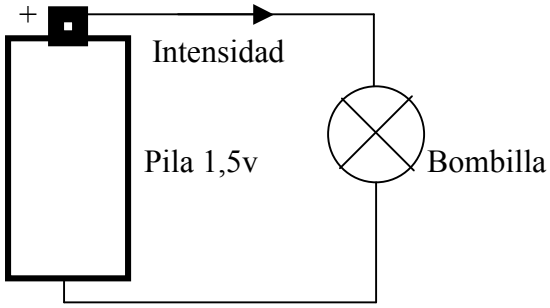
CONCEPTOS BÁSICOS

Si en una frase tuviera que definir que es un microcontrolador, creo que lo más acertado sería definirlo como un controlador de entrada/ salida con capacidad de decisión.

Para poder explicar el funcionamiento de un microcontrolador, primero vamos a intentar comprender el concepto de tensión o diferencia de potencial. El ejemplo más simple de generador de tensión es una pila, esta pose dos polos + y – y como característica principal, esta la diferencia de potencial que puede generar, normalmente los valores estándar suelen ser 1,5v, 4,5v, 9v y 12v. Si ponemos una bombilla entre sus bornes esta se encenderá por que la diferencia de potencial hará que circule una corriente, si desconectamos uno de los bornes, la bombilla se apagara, porque al no haber diferencia de potencial no circulará corriente.



Una fuente de alimentación es básicamente un circuito que transforma los 230v AC de la tensión de red a la tensión continua que nosotros queramos; como si trabajáramos con una pila (por ejemplo 5v).

Los microcontroladores van encapsulados en pastillas (llamadas chips) con un montón de patillas (llamadas pins). Cada pin tiene su función y básicamente las podemos catalogar en tres tipos:

1. **De alimentación:** La función de estas patillas es la de suministra al integrado el voltaje necesario para poder funcionar (generalmente +5v). Hay dos patillas para esta función, un pin para + (llamado generalmente Vdd ó Vcc) y un pin para – (llamado generalmente GND ó Vss). Es decir si tenemos una pila de +5v y conectamos el + a Vss y el – a GND el chip se activará.
2. **Especiales:** Son patillas necesarias para el funcionamiento interno del procesador, por ejemplo, las patillas OSC1 y OSC2 en las que se coloca el Xtal del oscilador o la patilla MCLR que sirve para inicializar el integrado.
3. **De entrada/ salida:** Suelen ser la mayoría y su función es la comunicar el microcontrolador con el exterior. Para realizar esto, el microcontrolador es capaz de poner un determinado pin a 1 haciendo que por esa patilla aparezcan +5v de tensión o a 0 haciendo que la tensión sea 0 (observa que esto es lo mismo que decir que cuando ponemos un 1, hacemos que el integrado conecte una pila entre la salida y masa y cuando ponemos un 0 la desconecte) con esta tensión, podemos activar, por ejemplo, un led, un relé, etc..., como además la

velocidad a la que podemos conectar y desconectar esta señal es muy alta (para el 16F84 con xtal de 4MHz la podemos activar y desactivar un millón de veces por segundo), podemos generar trenes de datos binarios con los que podremos controlar una pantalla, un puerto RS232, etc... Del mismo modo el integrado puede detectar el nivel de tensión que hay en un pin; si conectamos una pila de 5v entre el pin y GND nos dirá que en ese pin hay un 1, si la desconectamos nos dirá que hay 0.

Como ves casi todas las patillas del microcontrolador están destinas a enviar y recibir información del exterior y para poder procesar esa información y decidir que es lo que tiene que hacer en cada momento, dispone de un pequeño ordenador que posee una memoria de tipo ROM (EEPROM, OTP FLASH) en la que se almacenan las instrucciones del programa, una pequeña memoria RAM en la que se almacenan los datos y variables que necesita el programa y una memoria EEPROM, en la que se pueden almacenar ciertas variable o parámetros que no nos interesa que se pierdan si el sistema se apaga.

Ahora pasemos a ver todo esto en el PIC16F84.

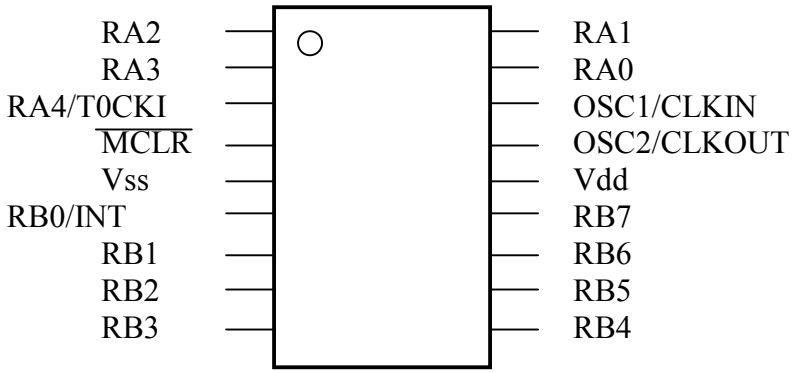
INTRODUCCIÓN AL PIC16F84.

En este apartado vamos a hacer una introducción al chip, mirando los puntos básicos necesarios para poder trabajar con él. Todo lo que aparece a continuación es aplicable a cualquier PIC de gama media (arquitectura de 14 bits).

El microcontrolador PIC16F84 posee las siguientes características:

- Memoria FLASH de 1k x 14 bits, en la que almacenaremos el programa.
- Memoria RAM de 68 x 8 bits, en la que almacenaremos los datos del programa.
- Memoria EEPROM de 64 x 8 bits.
- 1 Timer TMR0.
- 4 fuentes de interrupción.
- 13 líneas de entrada/ salida divididas en dos puertos RA y RB.
- Una ULA (unidad aritmético lógica) de 8 bits con banderines de Z (cero) C (acarreo) DC (semi-acarreo).
- 1 Watchdog timer. Este dispositivo sirve para vigilar (Watchdog significa perro guardián) que el microcontrolador funcione correctamente. Es básicamente un contador que cuando llega al final de la cuenta resetea el microcontrolador, para evitar que esto pase hay que introducirle, cada cierto tiempo, una palabra clave en un registro, si el microcontrolador esta colgado esto no se hará y el WDT reseteará el integrado.
- Power up reset. Este dispositivo permite al circuito inicializarse cuando la alimentación pasa desde 0 voltios a +5v, pero no funciona ante un Brown-out, es decir, la alimentación cae por debajo del nivel mínimo necesario para que el integrado funcione pero luego vuelve a subir sin llegar a 0.

El encapsulado de 18 pines, es el siguiente:



En la siguiente tabla aparece la descripción de cada pin:

Pin	Tipo	Descripción
OSC1/CLKIN	E	Xtal del oscilador o entrada de reloj externa.
OSC2/CLKOUT	S	Xtal del oscilador o salida de reloj externa.
-MCLR	E	Entrada de RESET. Si ponemos un 0 el chip se inicializa.
RA0..RA4	E/S	Puerto A de entrada/ salida. RA4 esta conectada al TMR0
RB0..RB7	E/S	Puerto B de entrada/ salida. En el RB0 es la entrada de interrupción externa y RB4..RB7 se pueden configurar como entradas de interrupción por cambio.
Vss		GND
Vdd		+5v

Para poder gestionar los recursos del microcontrolador (puertos, timers, ALU, etc...), el microcontrolador posee una serie de posiciones de memoria en RAM, llamadas registros de control, cada una de las cuales se encarga de controlar una determinada parte del chip. Estos registros son para el 16F8X los siguientes:

Dir.	Registros Pagina 0	Registros Pagina 1	Dir
00h	INDF	INDF	80h
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h	----	----	87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2	89h
0Ah	PCLATCH	PCLATCH	8Ah
0Bh	INTCON	INTCON	8Bh

Desde la posición 0Ch hasta la 4Fh tenemos posiciones de memoria libres que podemos utilizar como variables (a estas mismas posiciones podemos acceder desde las direcciones 8Ch a CFh).

REGISTROS:

- **STATUS (03h ,83h):** Este es el registro de estado, en el se encuentra información básica sobre el integrado, como es el estado de los banderines de Z, C y DC y la página de RAM que estamos utilizando. La descripción de sus bits es la siguiente:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRP	RP1	RP0	-TO	-PD	Z	DC	C

  - **IRP:** No se usa en el PIC16F84 y debe ser siempre 0.
  - **IRP1:IRP0:** Si te fijas en la dirección de los registros hay dos paginas , estos bits se utilizan para indicar en que pagina estamos (00= Página 0, 01= Página 1, 10= Página 2, 11= Página 3). Como el PIC16F84 solo tienen dos paginas RP1 valdrá siempre 0 y el valor de RP0 indicará la página en la que estamos.
  - **–TO:** Indica si el Watch-dog se activó.
  - **–PD:** Si vale 0 esta en modo standbye.
  - **Z:** Si este banderín vale 1, el resultado de la ultima operación aritmética es 0.
  - **DC:** Si el valor de este banderín es 1, existe un acarreo del cuarto bit en la ultima operación.
  - **C:** Si el valor de este banderín es 1, exite un acarreo del bit más significativo en la ultima operación.
- **OPTION (81h):** Este registro posee la siguiente información:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

  - **–RBPU:** Si lo ponemos a 0, hacemos que el puerto B tenga resistencias de pull-up activadas cuando estas están configuradas como entradas.
  - **INTEDG:** Selecciona el flanco por el que se activa la interrupción del pin RB0/INT. Si INTEDG=1 se hará por flanco de subida, si es 0 se hará por flanco de bajada.
  - **T0CS:** Indica cual es el origen de los pulsos de clk para el timer; si es 1 el origen es la señal introducida por el pin RA4/T0CKI, si es 0 el origen será el ciclo de instrucción (CLKOUT = FrecuenciaXtal/4).
  - **T0SE:** Indica el flanco por el que se incrementará la cuenta del contador TMR0. Si vale 0 se hará por el flanco de bajada, si es 1 se hará por el flanco de subida.
  - **PSA:** El chip tienen internamente un divisor, este se puede asignar mediante este bit al WDT con un 1 o al TMR0 con un 0.
  - **PS2:PS0:** Este es el valor de la preescala, según la siguiente tabla de valores.

Valor PS2:PS0	TMR0	WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

- **INTCONT (0Bh, 8Bh):** Este registro posee la siguiente información:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF

  - **GIE:** Si lo ponemos a 1, activa el permiso para que puedan haber interrupciones. Si este bit esta a 0, aunque habilites otras interrupciones estas no serán permitidas.
  - **EEIE:** Si lo ponemos a 1, permite que se genere una interrupción cuando haya termina de efectuar una escritura en la EEPROM.
  - **T0IE:** Si lo ponemos a 1, permite una interrupción cada vez que el TMR0 pase de 255 a 0.
  - **INTE:** Si lo ponemos a 1, permite la interrupción en el pin RB0/INT.
  - **RBIE:** Si lo ponemos a 1, permite la interrupción por cambio en el estado de los pines RB7:RB4.
  - **T0IF:** Si esta a 1, indica que se ha producido una interrupción en el TMR0.
  - **INTF:** Si esta a 1, indica que se ha producido una interrupción en la patilla RB0/INT.
  - **RBIF:** Si esta a 1, indica que se ha producido una interrupción porque al menos uno de los bits RB7:RB4 ha cambiado de estado.
- **PCL (02h 82h):** Este registro contiene el valor de los 8 bits menos significativos del contador de programa.
- **PCLATCH (0Ah 8Ah):** Este es un latch en el que almacenamos la parte alta del contador de programa.
- **INDF y FSR:** INDF se utiliza para hacer direccionamiento indirecto, de tal forma que el valor que leamos o escribamos en él, será el valor escrito o leído en el registro, cuya dirección este en FSR. Es decir, si FSR = 0Ch y leemos INDF, lo que obtendremos será el valor almacenado en 0Ch, del mismo modo, si escribimos un 5 en INDF escribiremos un 5 en el registro 0Ch.
- **TMR0:** Contiene el contador de 8 bits del timer 0.
- **TRISA, TRISB:** Sirve para indicar cuales pines son de salida, poniendo un 0 en el correspondiente bit, y cuales son de entrada puniendo un 1 en el correspondiente bit.
- **PORTA, PORTB:** Sirve para indicar si un determinado pin del puerto, que esta configurado como salida, esta a 1 o a 0.
- **EECON1, EECON2, EEDATA y EEADR:** Sirven para gestionar la transferencia de datos con la EEPROM y los estudiaremos más adelante con un ejemplo.

Además de estos registros el PIC contiene un registro interno especial, llamado W, que es el registro de trabajo de la ALU (unidad aritmético lógica) y que se utiliza para hacer cálculos y transferencias de datos entre registros.

INSTRUCCIONES:

El conjunto de instrucciones de todos los PIC de gama media, esta formado por 32 nemónicos, en los que nos podemos encontrar 5 tipos de etiquetas:

- **f:** Este es un registro de memoria y su valor puede estar entre 00h y 7Fh. (cuando accedemos a la dirección, por ejemplo, 8Ah, realmente accedemos a 0Ah pero de la pagina 1 - bit RP0=1 del registro STATUS-).
- **W:** registro interno de trabajo del PIC.
- **b:** Este se utiliza para indicar, en las instrucciones de manipulación de bits, a que bit nos referimos.
- **k:** Esta es una etiqueta de constante.
- **d:** Cuando d vale 0 el resultado de la instrucción se almacena en W, cuando es 1 el resultado se almacena en f.

La lista completa de las instrucciones del PIC es la siguiente:

Nemónico	Status	Descripción
ADDWF f,d	C,DC,Z	Suma al registro f y W y el resultado lo almacena según d.
ANDWF f,d	Z	Hace un AND entre f y W y el resultado lo almacena según d
CLRF f	Z	Pone a 0 todos los bits del registro f.
CLRW	Z	Pone a 0 W.
COMF f,d	Z	Hace un NOT a f y el resultado lo almacena según d.
DECF f,d	Z	Decrementa f y el resultado lo almacena según d.
DECFSZ f,d		Decrementa f y se salta la siguiente instrucción si f=0. El resultado lo almacena según d.
INCF f,d	Z	Incrementa f y el resultado lo almacena según d.
INCFSZ f,d		Incrementa f y se salta la siguiente instrucción si f=0. El resultado lo almacena según d.
IORWF f,d	Z	Hace un OR entre f y W y el resultado lo almacena según d.
MOVF f,d	Z	Mueve el contenido de f al destino indicado por d.
MOVWF f		Hace f = W.
NOP		No operación.
RLF f,d	C	Rotación a la izquierda a través del acarreo y el resultado lo almacena según d.
RRF f,d	C	Rotación a la derecha a través del acarreo y el resultado lo almacena según d.
SUBWF f,d	C,DC,Z	Resta a f W y el resultado lo almacena según d.
SWAPF f,d		Intercambia los nibbles del registro f y el resultado lo almacena según d.
XORWF f,d	Z	Hace una XOR entre f y W y el resultado lo almacena según d.
BCF f,b		Pone a cero el bit b del registro f.
BSF f,b		Pone a uno el bit b del registro f.

BTFSC f,b		Si el bit b del registro f es 0 se salta la siguiente instrucción.
BTFSS f,b		Si el bit b del registro f es 1 se salta la siguiente instrucción.
ADDLW k	C,DC,Z	Suma k a W
ANDLW k	Z	Hace un AND entre W y k y el resultado se almacena en W.
CALL k		Llama a la subrutina de dirección k.
CLRWDT	TO,PD	Inicia el contador del Watchdog.
GOTO k		Salta a la dirección k.
IORLW k	Z	Hace un OR entre W y k y el resultado se almacena en W.
MOVLW k		hace W = k.
RETFIE		Vuelve de interrupción.
RETLW k		Vuelve de subrutina con W = k.
RETURN		Vuelve de subrutina.
SLEEP	TO,PD	Pone el integrado en modo de bajo consumo.
SUBLW k	C,DC,Z	Resta a k W y lo almacena en W.
XORLW k	Z	Haz un XOR entre k y W y almacénalo en W.

En los siguientes ejercicios iremos profundizando en las diferentes partes del integrado y en el uso de estas instrucciones, de todas formas puedes conseguir el datasheet del integrado (en ingles) en www.microchip.com buscando 16F8X.