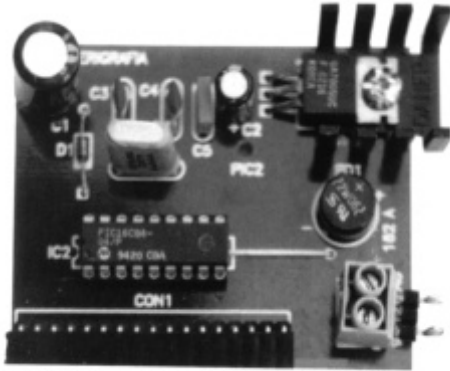


Programación y diseño de dispositivos mediante microcontroladores PIC



-Foto 1.-Imagen del kit del Módulo-01 montando un PIC16C84. Se puede apreciar el conector CN1.

Hemos elegido el microcontrolador PIC16F84 por las características que se explicarán más adelante. Nuestro microcontrolador PIC16F84 (ver Figura 1) que es un chip de 18 pines grabable, por lo que necesitamos un dispositivo que permita grabar el programa elaborado dentro del PIC 16F84, es decir, un grabador o programador. Además, es necesario un

compilador que produzca el código ejecutable de dicho programa.

DESCRIPCIÓN PINES PIC16F84 16C84				
Nombre	Nº	Tip. I/O/P	Tip. Buffer	Descripción
OSC1 CLKIN	16	I	ST / CMOS	Entrada oscilador a cristal / fuente externa de reloj
OSC1 CLKOUT	15	O	--	Salida oscilador a cristal. Conectar al cristal de cuarzo
MCLR	4	I / P	ST	Reset a nivel 0. Entrada estroboscópica de programación
RA0	17	I / O	TTL	El Puerto A es bidireccional
RA1	18	I / O	TTL	El pin RA4 del Puerto A, si se programa como salida es de colector abierto. Como entrada puede seleccionarse en funcionamiento normal o como entrada del contador / temporizador TMR0 en modo TMR0
RA2	1	I / O	TTL	
RA3	2	I / O	TTL	
RA4 TMR0	3	I / O	ST	
RB0 INT	6	I / O	TTL / ST	El pin 6 del Puerto B, RB0 puede programarse como entrada de interrupciones externas INT
RB1	7	I / O	TTL	El Puerto B es bidireccional. Mediante software y cuando se programa como entrada pueden activarse resistencias de potencia interna
RB2	8	I / O	TTL	Los pines RB4 a RB7 pueden programarse para que respondan a interrupción por cambio de estado
RB3	9	I / O	TTL	
RB4	10	I / O	TTL	
RB5	11	I / O	TTL	
RB6	12	I / O	TTL / ST	Modo programación entrada RELOJ
RB7	13	I / O	TTL / ST	Modo programación DATOS
Vss	5	P	--	Pin de masa GND
Vdd	14	P	--	Pin de alimentación positiva

-Figura 1.- Breve descripción y disposición de patillas del PIC 16F84 y 16C84.

Una vez introducido el programa en la memoria del microcontrolador, éste funcionará mediante un conjunto mínimo de componentes externos (ver Figura 2).

registros. El consumo típico es de 2 mA a 4 MHz y unos 40 μ A funcionando en modo Sleep.

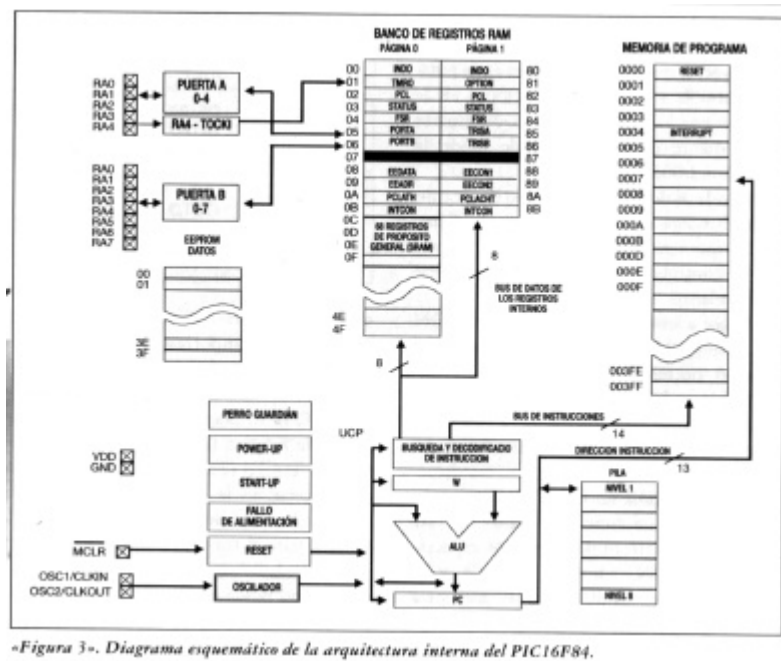
El encapsulado de 18 pines así como la descripción del patillaje puede verse en la «Figura 1 », mientras en la «Figura 2» se presenta el esquema eléctrico del mismo. Contiene sólo y exclusivamente el microcontrolador PIC16F84 así como la circuitería necesaria para su alimentación partiendo de una entrada en alterna de 9-12 V A.C. o desde una pila de 9V. Incluye además un conector de 17 pines (CN1) donde se encuentran todas las líneas útiles del microcontrolador así como las señales de masa, + 5V y el voltaje rectificado y filtrado del transformador PWR. Con la sola sustitución del cristal de cuarzo y dos condensadores puede hacerse funcionar el microcontrolador a 4 o 10 MHz.

Arquitectura interna

Las altas prestaciones de los microcontroladores PIC derivan de las características de su arquitectura (ver «Figura 3»). Están basados en una arquitectura tipo Harvard, es decir, disponen de dos memorias independientes: una que contiene sólo instrucciones y otra sólo con datos. Ambas poseen sus **respectivos sistemas de buses** y es posible realizar operaciones de **acceso (lectura o escritura) simultáneamente** en ellas. Esto los hace más rápidos que los microcontroladores basados en la arquitectura tradicional de Von Neumann.

Otra característica es su juego de instrucciones reducido (35 instrucciones) RISC, que en su mayoría se ejecutan en un solo ciclo de reloj excepto las instrucciones de salto que necesitan dos ciclos.

Los microcontroladores PIC 16F84 poseen dos bloques de memoria separados, la memoria de programa y los bancos de registros. (ver Figura 3).



«Figura 3». Diagrama esquemático de la arquitectura interna del PIC16F84.

La memoria de programa está organizada con palabras de 14 bits con un total de 1K. Es del tipo EEPROM y en funcionamiento es de sólo lectura. Los bancos de registros RAM son de 8 bits (byte) a excepción del contador del programa que es de 13 bits. Como puede verse en la Figura 3 las direcciones bajas contienen los registros especiales y los registros de recursos, encontrándose a continuación los registros de propósito general. El PIC 16F84 posee 68 registros que van de la dirección \$00 a la \$4F, aunque no todas estas direcciones se usan.

El vector de *reset* se encuentra en la posición 0x0000h y el de interrupciones en la 0004h, por lo que la memoria de usuario propiamente dicha se extiende desde la dirección 0005h a la 03ffh.

La pila es de ocho niveles. **No hay ningún flag** que marque si **está llena**, por lo que será el programador el responsable de controlar que no se produzca su desbordamiento.

Posee además una memoria EEPROM de datos de 8 bits. Ésta no forma parte del espacio normal direccionable y sólo es accesible en lectura y escritura a través de dos registros: para los datos el EEDATA que se encuentra en la posición 0008h del banco de registros RAM y para las direcciones el EEADR en la 0009h.

Para definir el modo de funcionamiento de esta memoria se utilizan dos registros especiales, el EECON1 dirección 0088h y el EECON2 en 0089h. Esta memoria EEPROM no emplea ningún recurso externo de alimentación. Su programación dura unos 10 ms y se controla mediante un temporizador interno.

Mencionar por último que el microcontrolador dispone de dos puertos de entrada y salida (E/S). El puerto A con 5 líneas de la RA0 a la RA4 y el puerto B que dispone de 8 líneas de entrada/salida que van de la RB0 a la RB7.

Programa de ejemplo

El primer programa que haremos estará escrito en ensamblador. La finalidad del mismo consiste en contar en binario a través del puerto B (patitas RBO RB7), iluminando, en su caso los correspondientes diodos LED. El primer paso consiste en la escritura del programa CONTAR haciendo uso de un editor de textos ASCII. El segundo paso es el ensamblado de este fichero (CONTAR.ASM) empleando el compilador-ensamblador MPASM.EXE que distribuye libremente el fabricante. El ensamblador produce varios ficheros, aunque de entre ellos el que necesitamos es el fichero CONTAR.HEX.

PROGRAMA CONTAR.ASM

```

;*****
; Programa CONTAR.ASM                      Fecha :
; *
; Este programa cuenta de 0 a 255 sacando el dato binario a los LED      *
; conectados a la Puerta B introduciendo un retardo durante la cuenta    *
; Revisión : 0.0                      Programa para PIC16C84/16F84      *
; Velocidad del reloj: 4 MHz          Reloj Instrucción: 1 MHz = 1 uS   *
; Perro guardián : Deshabilitado     Tipo de Reloj : XT                *
; Protección del código : OFF
; *
;*****

;***** IGUALDADES *****
; ***** Igualdades que designa los destinos *****
w          EQU          0
f          EQU          1

;***** Igualdades de la UCP y del Mapa de memoria *****
PORTA      EQU          05h    ; Puerta A
PORTB      EQU          06h    ; Puerta B
TRISA      EQU          05h    ; Registro triestado Puerta A. Página 1
TRISB      EQU          06h    ; Registro triestado Puerta B. Página 1
STATUS     EQU          03h    ; Registro STATUS
RP0        EQU          05h    ; Bit 5 registro STATUS
Contador1  EQU          0Ch    ; Registro para primer contador
Contador2  EQU          0Dh    ; Registro para segundo contador

; ***** Sección Código de reset *****
;          ORG          00h    ; Dirección del Vector de reset
;          GOTO         Inicializa ; Comienza el programa detrás
;                               ; del Vector interrupción
;          ORG          05h    ; Una posición a continuación
;                               ; del vector de interrupción

; ***** Sección Inicializa *****
; Inicializa  BSF          STATUS,RP0 ; Selecciona página 1
;             CLRF        TRISB      ; Programa Puerta B todo salida
;             BCF         STATUS,RP0 ; Vuelve a página 0
;             CLRF        PORTB      ; Apaga los LEDs borrando la Puerta B
;             CLRF        Contador1  ; Inicializa (borra) Contador1 (0Ch)
;             CLRF        Contador2  ; y Contador2 (0Dh)

; ***** Sección Principal *****
; Principal  INCF         PORTB,f    ; Incrementa en 1 la Puerta B
; ***** Sección Bucle *****
; Bucle      DECFSZ      Contador1,f ; Decrementa Contador1 y chequea si = 0
;             GOTO       Bucle      ; Si no 0, decrementa Contador1
;             DECFSZ      Contador2,f ; Si es 0, decrementa Contador2
;             GOTO       Bucle      ; Si Contador2 no es 0, decrementa
;                               ; Contador1 otras 256 veces
;             GOTO       Principal  ; Actualiza los LED
;                               ; después de 65536 iteraciones
;             END

```

Este programa utiliza dos registros de 8 bits para realizar un contador anidado, en el que por cada iteración del contador 1 se producen 256 iteraciones del contador 2.

Llegados a este punto podemos comparar el número de líneas de código en ensamblador con su equivalente en C utilizando un compilador para PIC's:

```
main ( )
{   char i, j, k;           //Declaro dos variables auxiliares para contar
    set_tris_b (0);
    for (k=0;k<255;k++)
        {   output_port-b(k);
            for (i=0;i<255;i++)
                for (j=0;j<255;j++)
            }
    }
```