Capítulo 1

Práctica: Proteus, MPLab y Hitec PICC

Aquel que hace una pregunta puede ser un tonto por cinco minutos,
pero aquel que nunca hace una pregunta permanece tonto por siempre
∼ Tom J. Connelly ∼

1.1. Objetivo

[1] Se propone la realización de un sencillo programa en ensamblador y/o lenguaje C para el microcontrolador PIC16F88 que es el objetivo de estudio de este curso. Se busca aprender a manejar las herramientas software y de desarrollo Proteus ISIS, MPLAB y Hitec PICC, así como poner en práctica el ciclo de diseño completo, que va desde la especificación del problema, su diseño, simulación, hasta la puesta en marcha y depuración del sistema.

1.2. Software

MPLAB

MPLAB (actualmente la versión 8.20) es la herramienta que proporciona de manera gratuita el fabricante de los microcontroladores que estudiamos y se puede descargar de su página web [2].

Esta herramienta permite ensamblar, compilar, depurar, depurar en circuito y grabar nuestros programas en el microcontrolador. Permite el uso de compiladores de otros fabricantes e incluso permite una integración con el *software* que vamos a utilizar: Proteus, Hitec PICC.

Proteus ISIS

Proteus es un programa de diseño (actualmente la versión 7.5) asistido por ordenador que permite dibujar el esquema hardware de nuestros diseños y simularlos (tipo SPICE). La ventaja es que permite una simulación híbrida digital/analógica que permite simular también algunos microcontroladores (los más frecuentemente utilizados) y lo que es más importante depurar el funcionamiento de nuestras aplicaciones.

En esta práctica pretendemos hacer uso de esta herramienta que se puede adquirir a través de la página web del *Labcenter Microelectronics*[3]. La versión profesional para principiantes permite simular el microcontrolador 16F877 además del 16F84A y del 18F452 por unas 150 libras

[1] Versión de 2 de marzo de 2009
[2] `http://www.microchip.com`
[3] `http://www.labcenter.co.uk`

esterlinas (20 % de descuento si el uso es educacional). Permite simular prácticamente todos los microcontroladores de estas familias de Microchip por un precio algo más elevado.

Existe una versión *shareware*[4] (versión 6.9 shareware por 30 libras) que permite simular solamente el microcontrolador PIC16F84A.

Hitec PICC

Si deseamos escribir nuestros programas utilizando un lenguaje de alto nivel como puede ser C podemos acudir a los múltiples fabricantes de *software*. Se recomienda un compilador de C de la casa HI-TECH[5]).

La página WEB del HITECH pone a nuestra disposición una versión de evaluación (con prestaciones y duración temporal limitadas)[6]. También disponemos de una versión *freeware*[7].

[4] `http://www.proteuslite.com`

[5] `http://www.htsoft.com/products/picccompiler.php`

[6] `http://www.htsoft.com/downloads/demos.php`

[7] `http://www.htsoft.com/microchip/products/compilers/piccpro-modes.php` con algunas funcionalidades menos (código generado menos eficientemente, etc.)

1.3. Especificaciones de la práctica

El primer programa propuesto consiste en introducir un valor en binario por el PORTA del microcontrolador (5 bits) y sacar por el PORTB (8 bits) el valor leído incrementado en dos unidades. Se aconseja conectar los pines de entrada a los conmutadores del entrenador y las patillas de salida a los LED o al display de 7 segmentos del entrenador. Vamos a emplear el microcontrolador PIC16F88.

Para poder trabajar con él deberemos saber que:

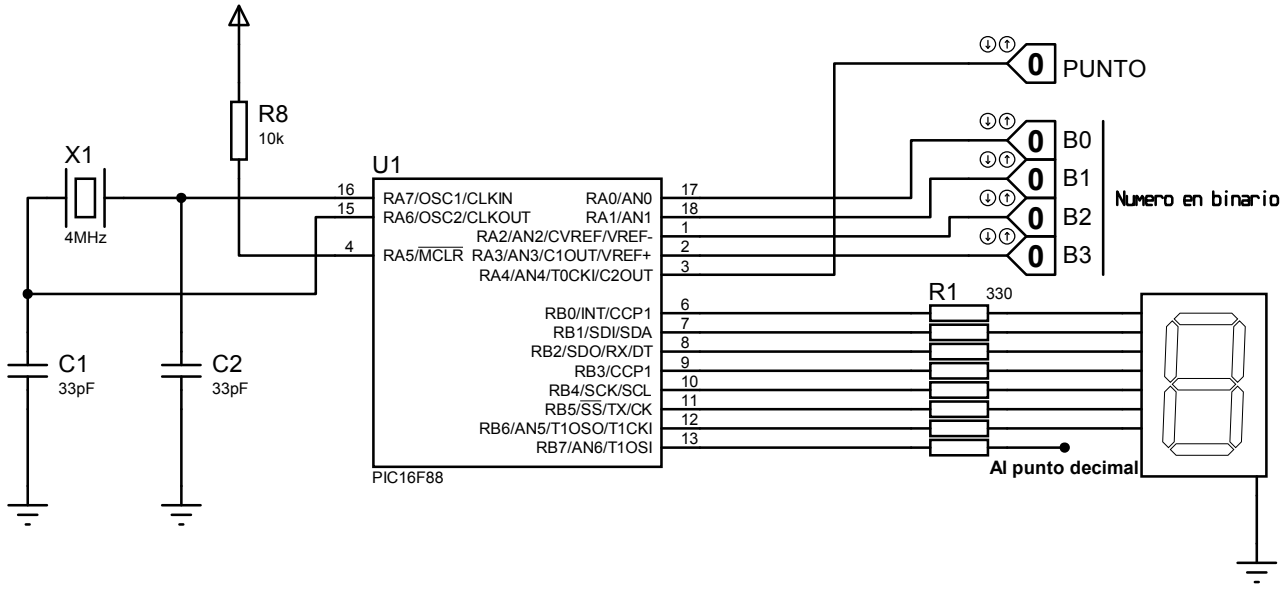
- El primer registro de propósito general utilizable es el : 0x20 (32d) y no el 0x0C (12d) como en el PIC16F84A
- El PORTA al arrancar se configura como de entradas analógicas. Para su uso digital hay que definirlo como sigue:

```
1               bsf STATUS,RP0  ; Banco 1 de registros
2               bcf STATUS,RP1  ; En el 16F88 hay que cambiar tambien RP1
3               clrf ANSEL      ; ANSEL = 0, todo patillas digitales ...
4                               ; Modifica TRISA para el sentido
5               bcf STATUS,RP0  ; Banco 0 de registros
```

En esta ocasión proporcionamos el listado en lenguaje ensamblador:

```
1           LIST p=16F88
2           INCLUDE "P16F88.INC"
3           RADIX DEC
4           ERRORLEVEL -302
5
6           ORG 0
7
8           bsf     STATUS,RP0      ; Selecciona Banco 1
9           bcf     STATUS,RP1
10          movlw   11111111b       ; W = 0FFh (Todo entradas)
11          movwf   TRISA           ; Configuro PORTA
12          movlw   00000000b       ; W = 00h  (Todo salidas)
13          movwf   TRISB           ; Configuro PORTB
14          clrf    ANSEL           ; ANSEL=0, PORTA digital
15          bcf     STATUS,RP0      ; Selecciono Banco 0
16  Bucle   movfw   PORTA           ; Leo W = PORTA
17          andlw   00011111b       ; Me quedo con los 5 bits del PORTA
18          addlw   2               ; W = W + 2
19          movwf   PORTB           ; PORTB = W
20          goto    Bucle           ; Repito indefinidamente
21          END
```

El esquema hardware será:



Para sacar nota

Prueba a cambiar el valor de salida en función de la entrada sacando el complemento del valor de entrada. ¿Sabrías codificar el valor mediante una operación O exclusiva?

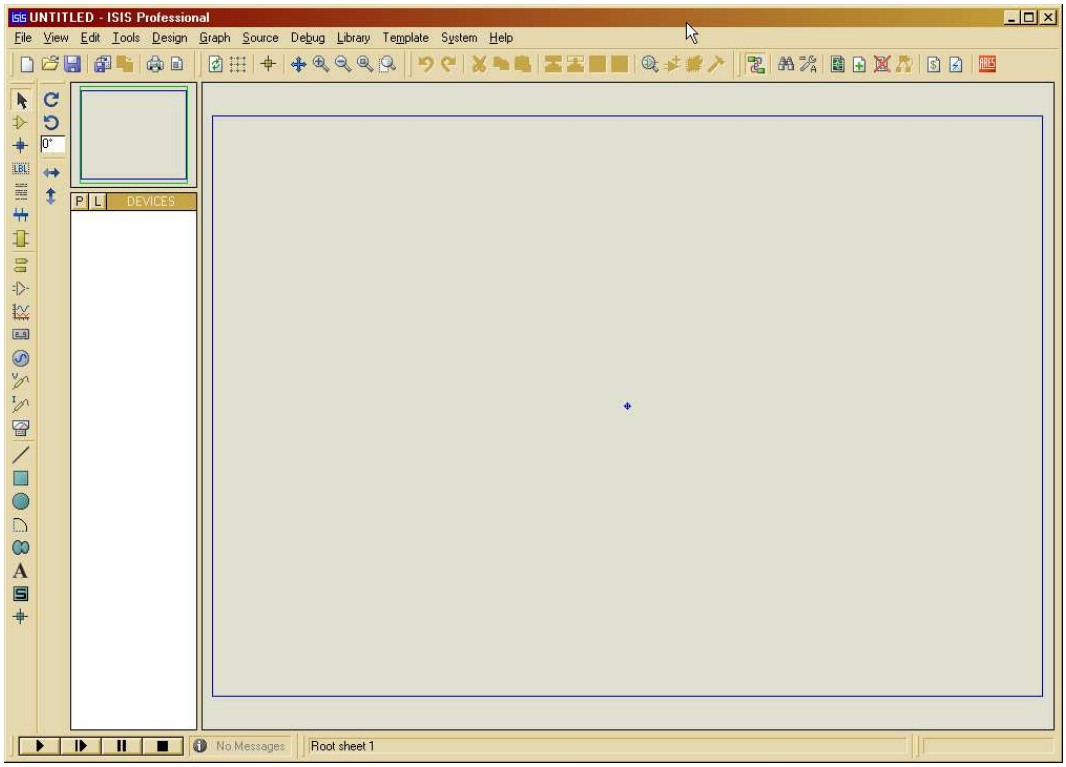
El juego de instrucciones en ensamblador del microcontrolador es el siguiente:

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected
BYTE-ORIENTED FILE REGISTER OPERATIONS					
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z
CLRF	f	Clear f	1	00 0001 lfff ffff	Z
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff	
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff	
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z
MOVWF	f	Move W to f	1	00 0000 lfff ffff	
NOP	-	No Operation	1	00 0000 0xx0 0000	
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff	
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z
BIT-ORIENTED FILE REGISTER OPERATIONS					
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff	
BSF	f, b	Bit Set f	1	01 01bb bfff ffff	
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff	
LITERAL AND CONTROL OPERATIONS					
ADDLW	k	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z
ANDLW	k	AND literal with W	1	11 1001 kkkk kkkk	Z
CALL	k	Call subroutine	2	10 0kkk kkkk kkkk	
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$
GOTO	k	Go to address	2	10 1kkk kkkk kkkk	
IORLW	k	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z
MOVLW	k	Move literal to W	1	11 00xx kkkk kkkk	
RETFIE	-	Return from interrupt	2	00 0000 0000 1001	
RETLW	k	Return with literal in W	2	11 01xx kkkk kkkk	
RETURN	-	Return from Subroutine	2	00 0000 0000 1000	
SLEEP	-	Go into standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$
SUBLW	k	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z
XORLW	k	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z

1.4. Primeros pasos con Proteus

Para comenzar el tutorial lo primero es arrancar el programa que tiene una pantalla inicial, después del logotipo, como la que aparece a continuación. Hay que indicar que el sistema Proteus consta de dos módulos o programas diferenciados:

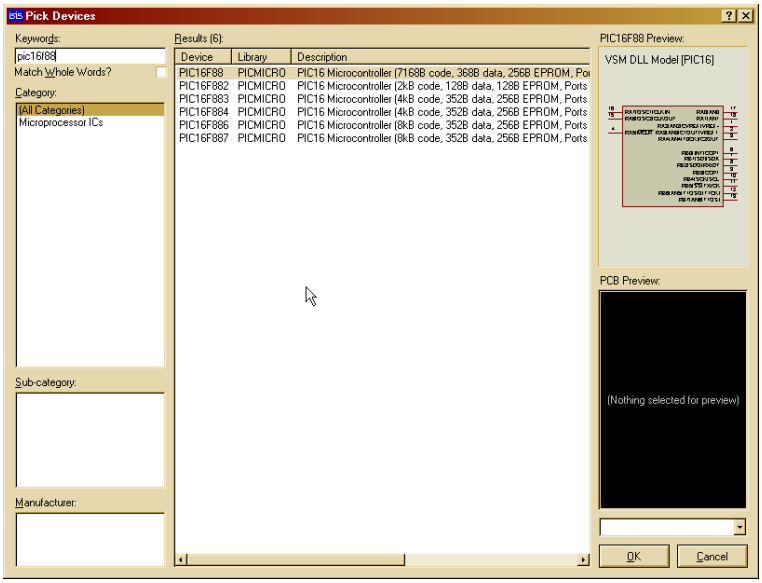
- ISIS que es el módulo que permite hacer la captura esquemática y las simulaciones -y que es objeto de este tutorial- y
- ARES que es el módulo dedicado al diseño de placas de circuito impreso (PCBs).



Inicialmente debemos crear un proyecto vacío (File⇒New) o abrir uno ya existente (File⇒Open). Se distinguen varias partes en la ventana de la aplicación y que son:

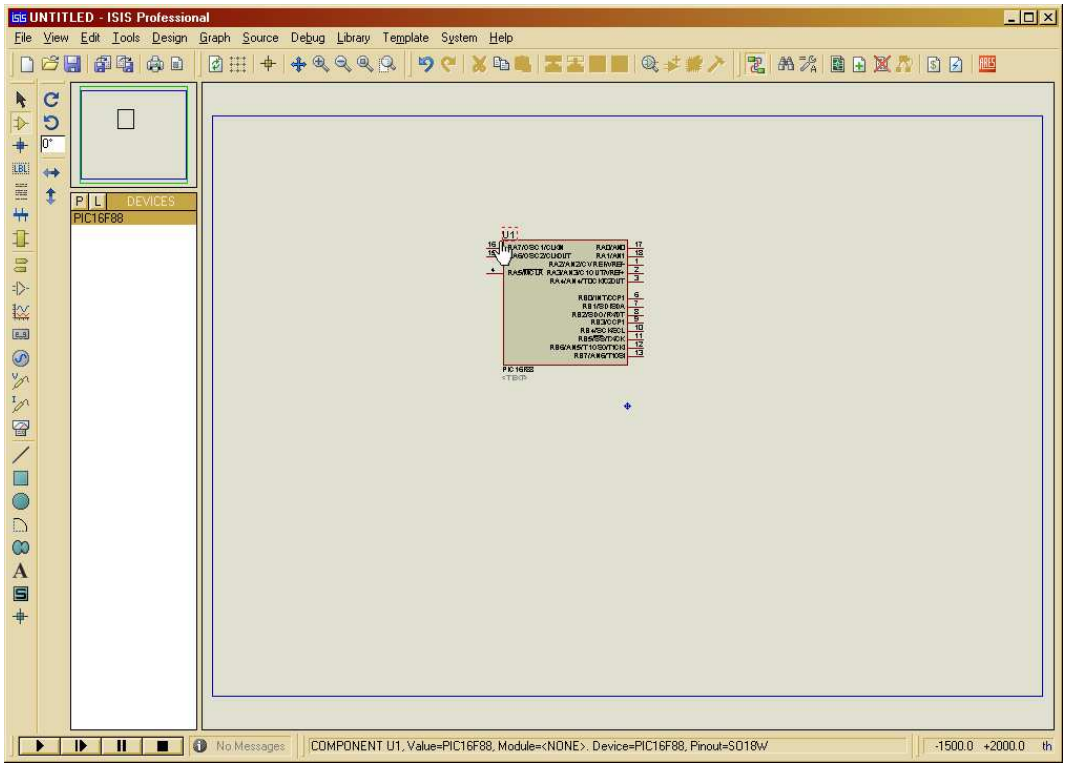
- la barra de herramientas en la parte superior, debajo de los menús,
- una barra de estado que en realidad nos permitirá modificar geométricamente los componentes, y que además permitirá mediante cuatro sencillos botones arrancar la simulación, pararla, ejecutar un paso, etc.
- Además tenemos otra barra de herramientas en formato vertical que va acompañada de una lista de dispositivos.
- Por último tenemos la hoja donde vamos a colocar los distintos componentes.

En un primer diseño vamos a situar el montaje básico de un microcontrolador: el PIC16F88. Para ello es necesario seleccionarlo. Con el atajo de teclado P lograremos situar cualquier componente como se ve en la siguiente figura:

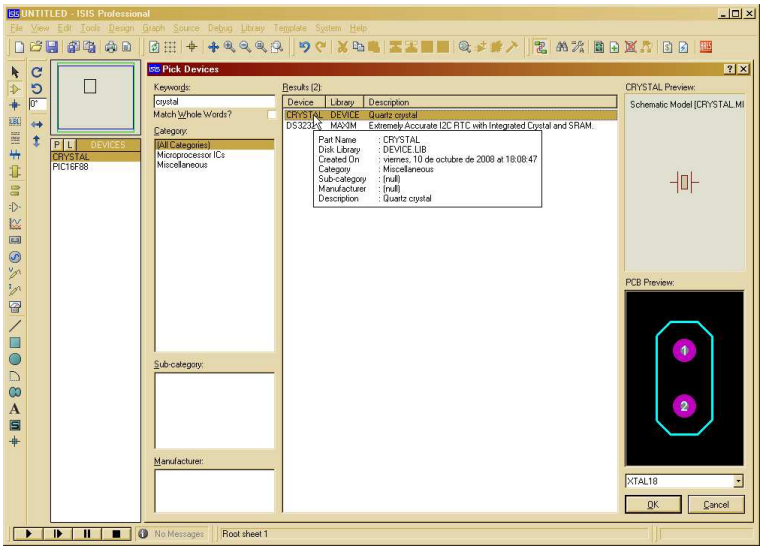


Simplemente debemos teclear el nombre o parte del nombre del dispositivo buscado y nos aparecerá una lista de posibles candidatos y con la selección de uno de ellos el esquema gráfico que lo define.

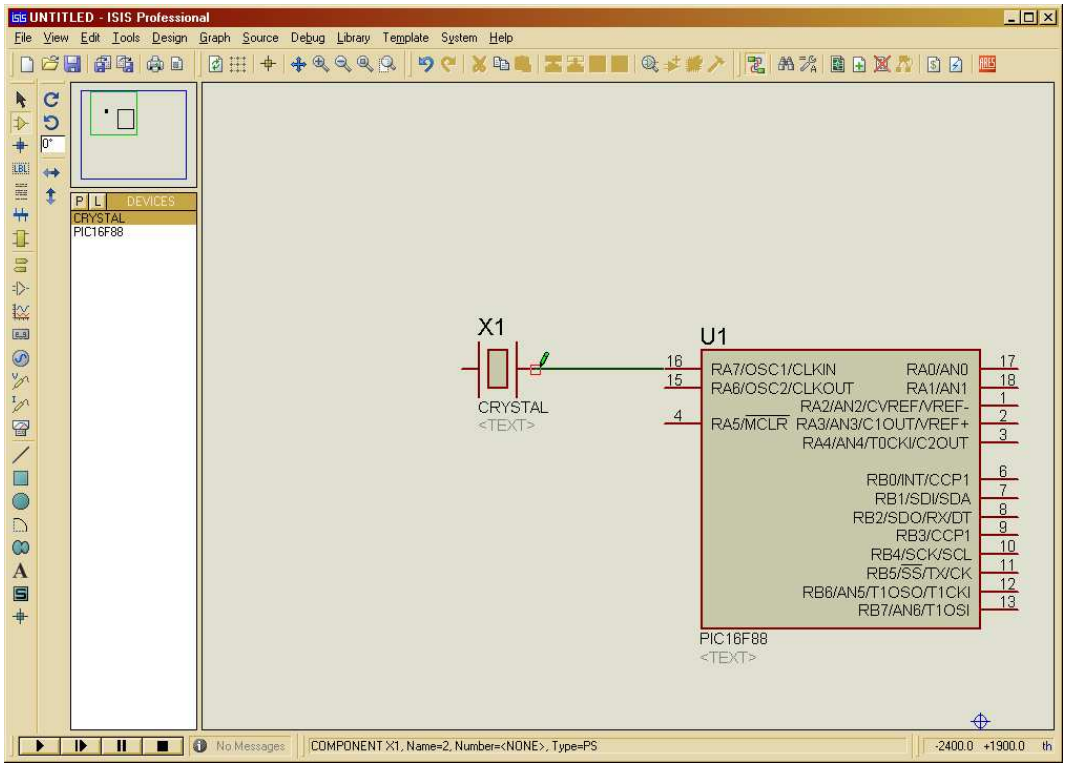
Una vez aceptado el componente solo tendremos que seleccionar un lugar con el ratón y pulsar el botón izquierdo para situarlo en la hoja de diseño:



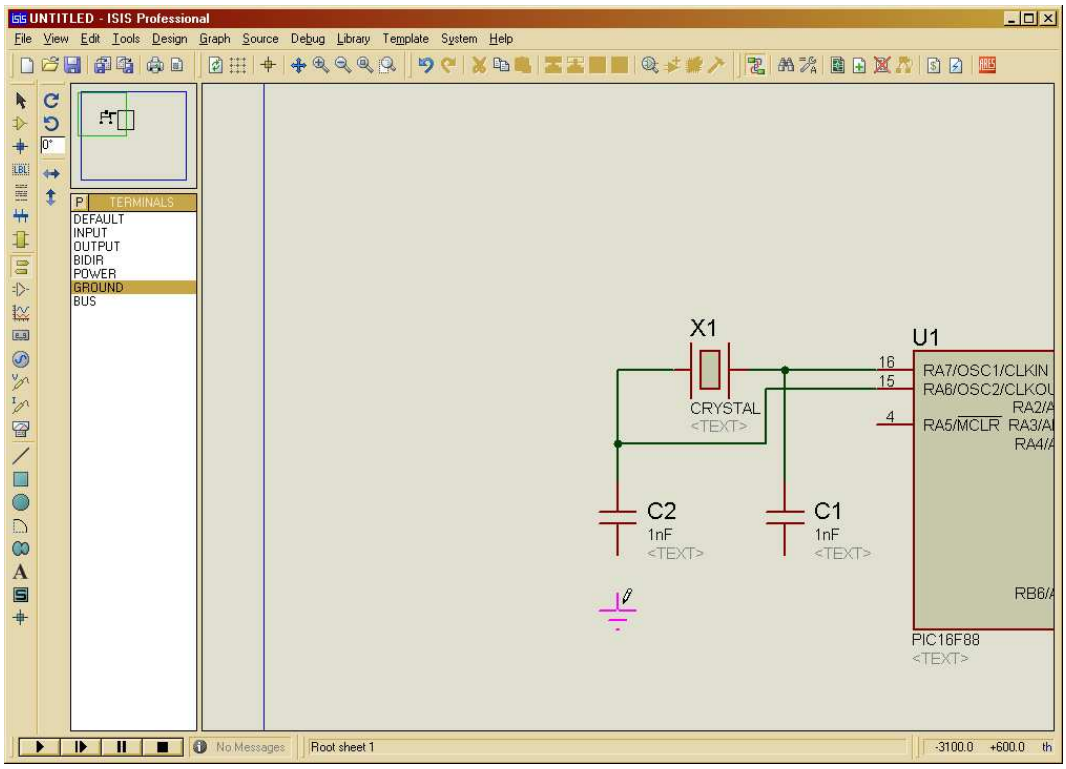
Haremos lo mismo con otros componentes necesarios como el cristal de cuarzo (CRYSTAL).



Para trazar un cable entre dos elementos simplemente aproximaremos el cursor hasta la patilla correspondiente y pulsaremos el botón izquierdo del ratón para trazar automáticamente el cable. Si queremos que el cable recorra una figura determinada, simplemente pulsaremos a lo largo del camino a recorrer y terminaremos con el segundo elemento.

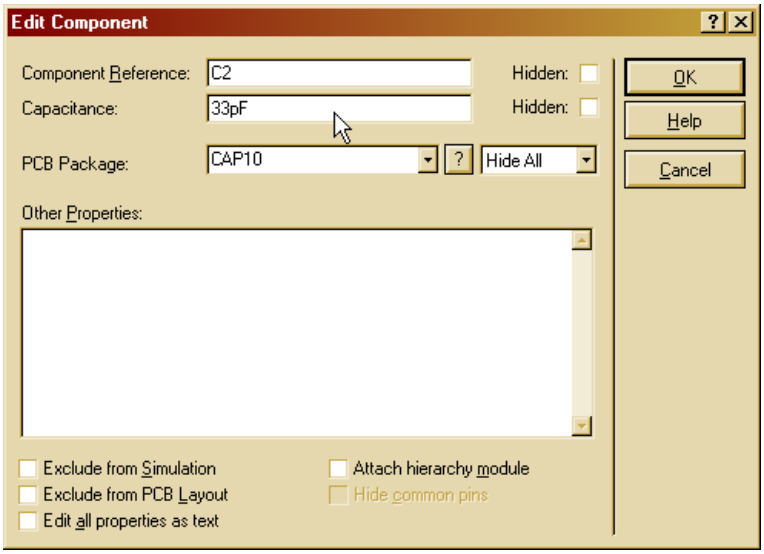


Después de haber colocado y conectado otros elementos como los condensadores (CAP), necesitaremos poner algunas tierras (GROUND) y alimentaciones (POWER). Estas se encuentran pulsando en la barra de herramientas vertical el icono relacionado con los terminales.

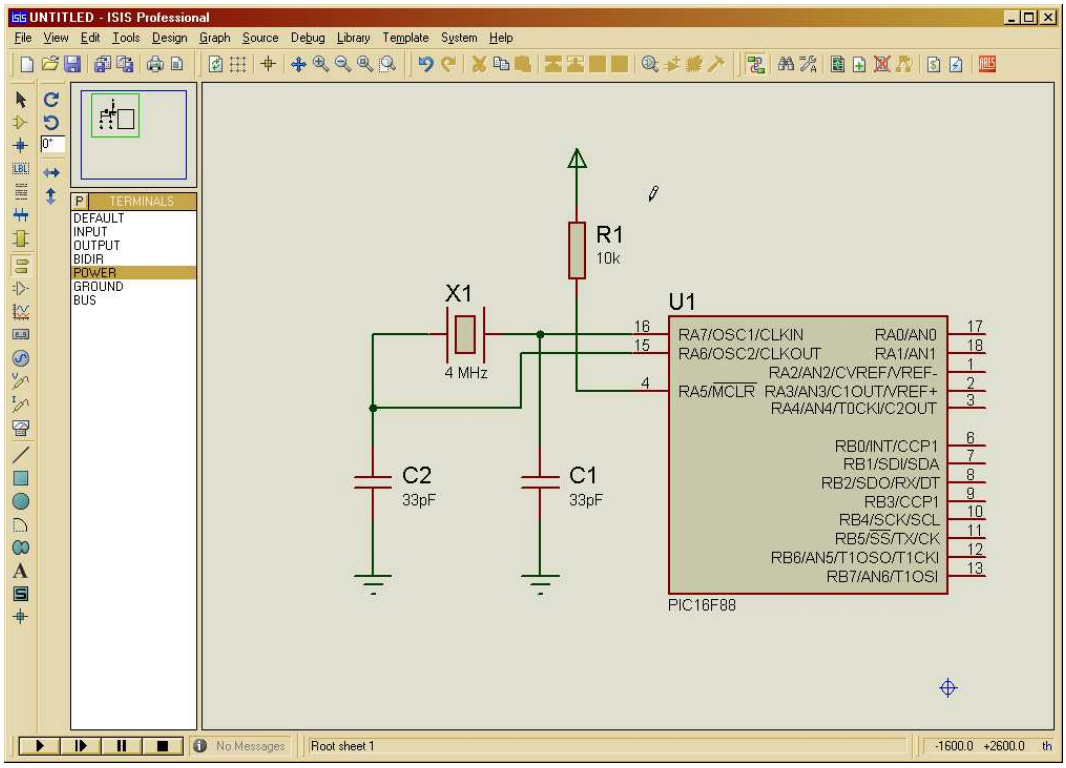


La conexión de estos elementos se hace de la misma manera. Si necesitamos ver ampliada la hoja de diseño se puede hacer a través del menú, de la barra de herramientas horizontal o utilizando los atajos de teclado (teclas F5, F6, F7, F8).

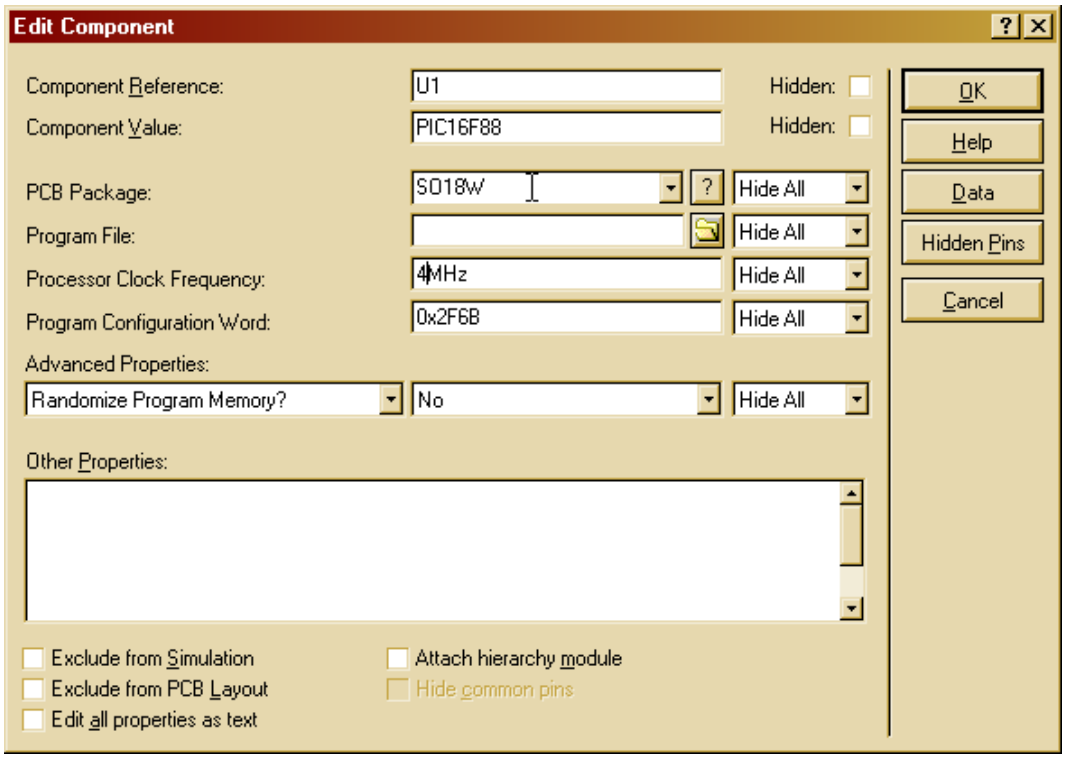
Una vez colocados los componentes querremos modificar su valor. Para ello seleccionamos el componente concreto pulsando sobre él con el botón derecho del ratón (o haciendo doble click en el valor del componente), y a continuación con el botón izquierdo. Cuidado por que si pulsamos dos veces con el botón derecho eliminaremos el componente. El atajo de teclado U nos ayudará a recuperar (deshacer la última operación) el componente borrado por error.



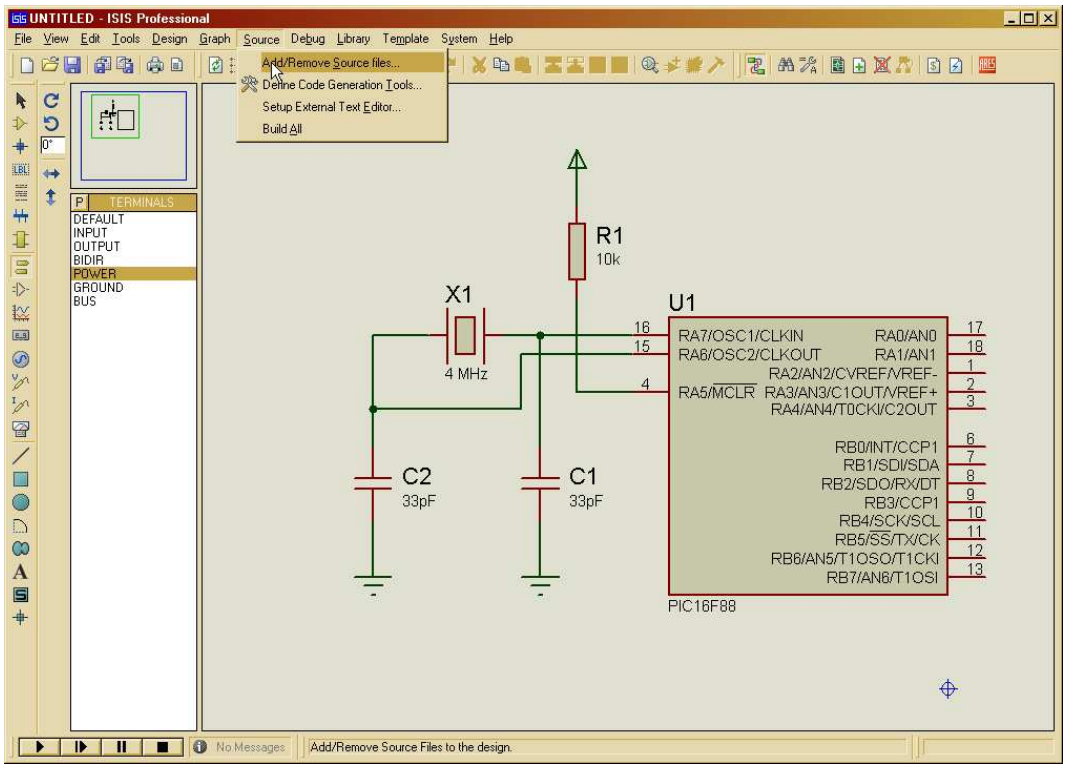
Después de haber modificado los valores como los que aparecen en la figura (resistencias: RES), deberemos indicarle al microcontrolador con qué frecuencia va a ser simulado.



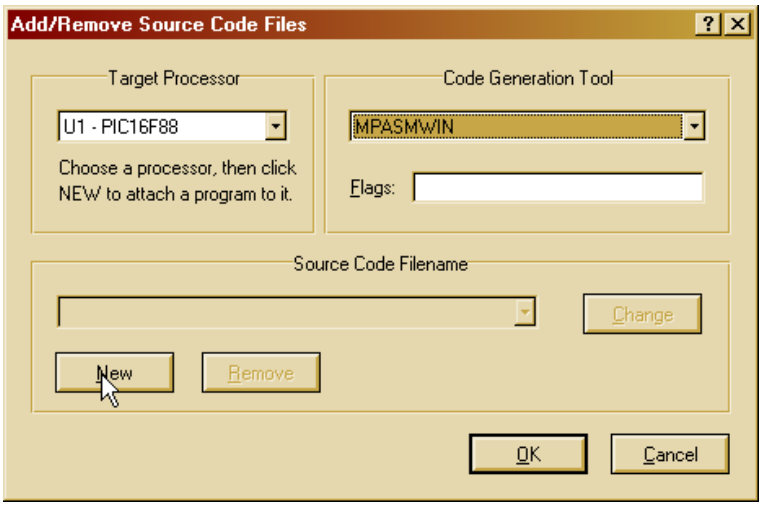
Seleccionamos el micro y editamos sus propiedades



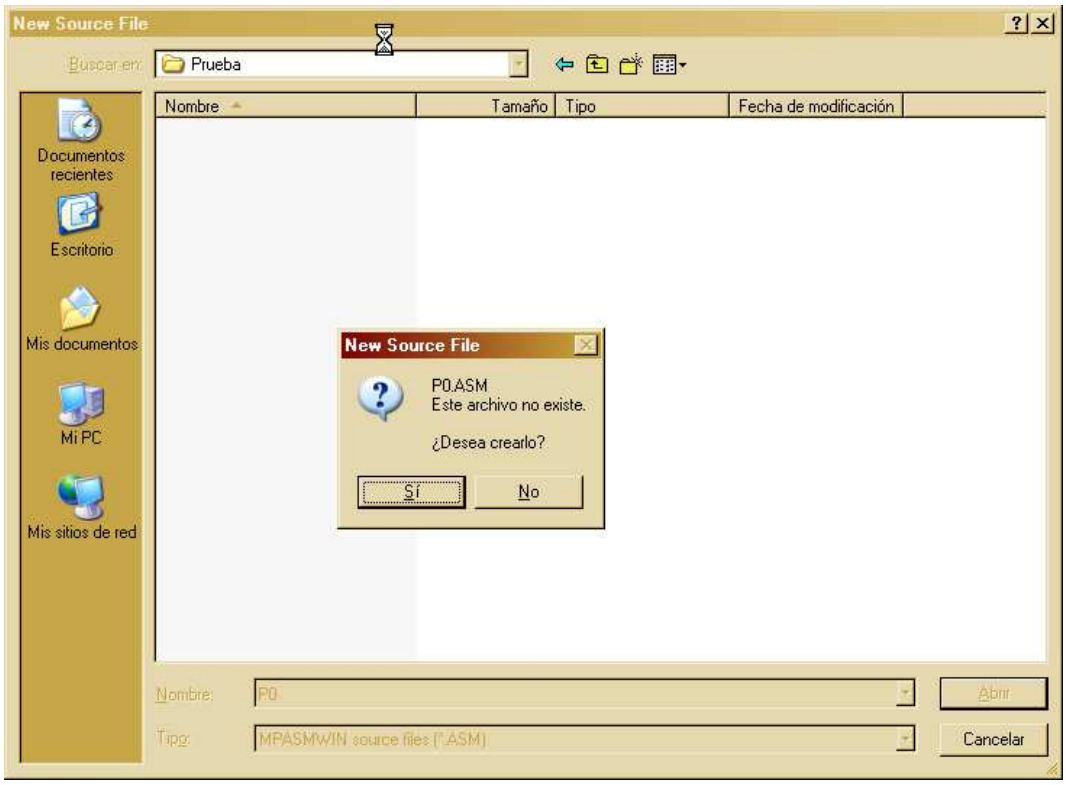
Colocaremos la frecuencia de reloj coincidente con la del cristal de cuarzo del esquema. Ahora ha llegado el momento de añadir código al microcontrolador.



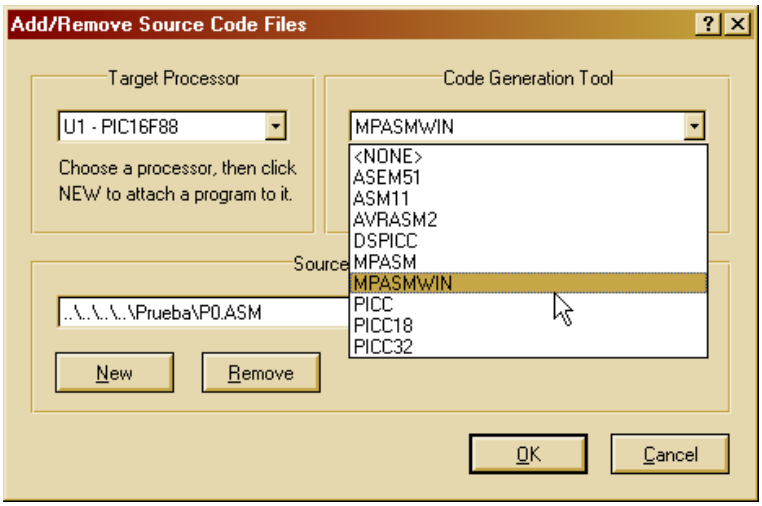
Por lo tanto, en el menú Source⇒Add/Remove Source Files .. seleccionaremos el listado en ensamblador o C (si se dispone del compilador correspondiente) que queremos colocar en el micro (el programa principal; los demás ficheros estarán incluidos de alguna manera en el fichero principal).



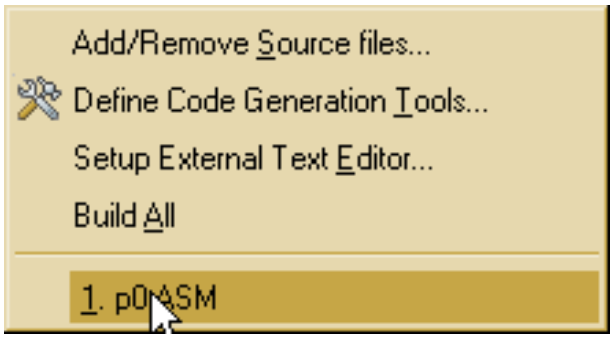
Si el fichero no existe no importará ya que será creado al editarlo



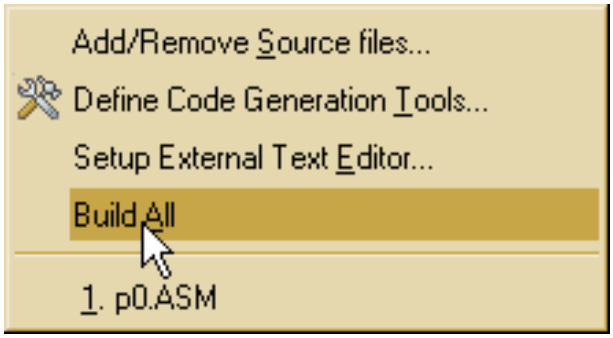
Al mismo tiempo seleccionaremos la herramienta de compilación, en este caso seleccionaremos MPASMWIN.



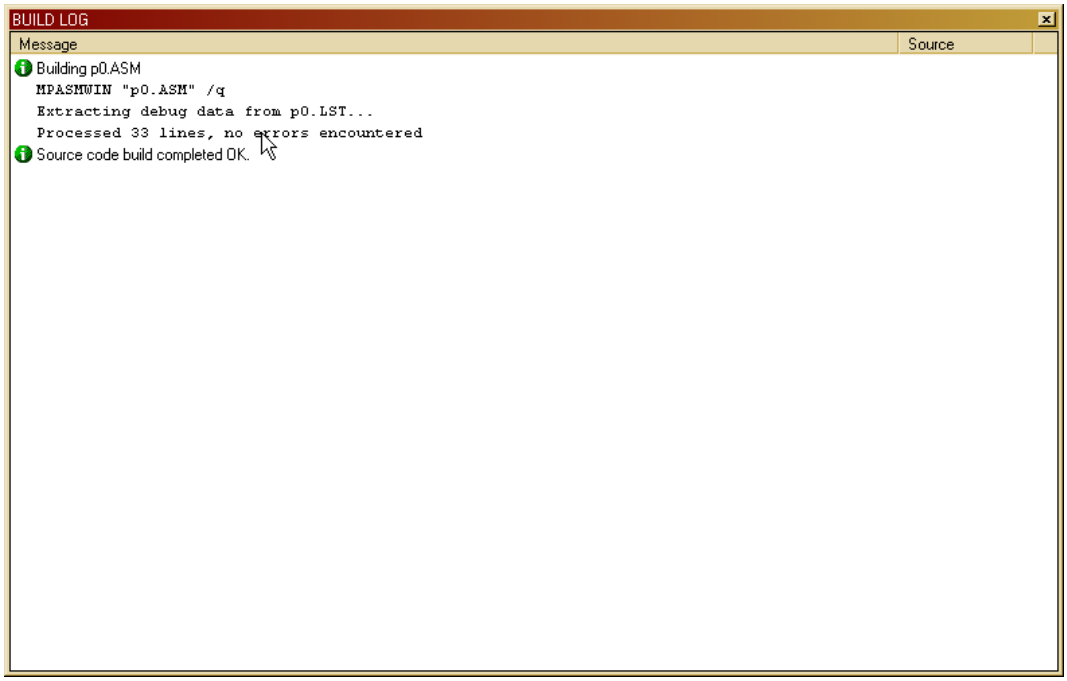
Dentro del menú Source aparecerá el fichero relacionado. Pulsando esa opción se arrancará un editor para modificar/crear el programa.



Editaremos el código deseado Y lo compilaremos mediante la opción Source⇒Build All.

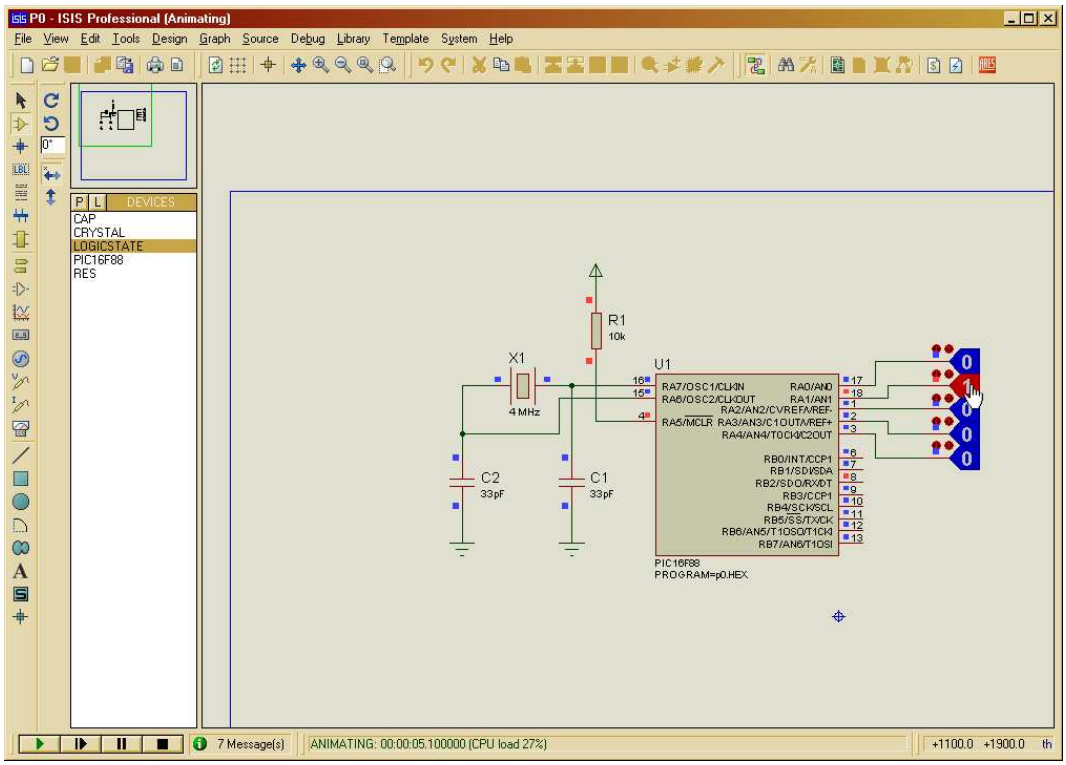


Si todo fue bien y no cometimos ningún error de sintaxis aparecerá la siguiente ventana indicándonos que todo fue bien. El ensamblador/compilador utilizado es externo al entorno, luego se pueden añadir herramientas de este tipo para programar el microcontrolador en el lenguaje que deseemos.



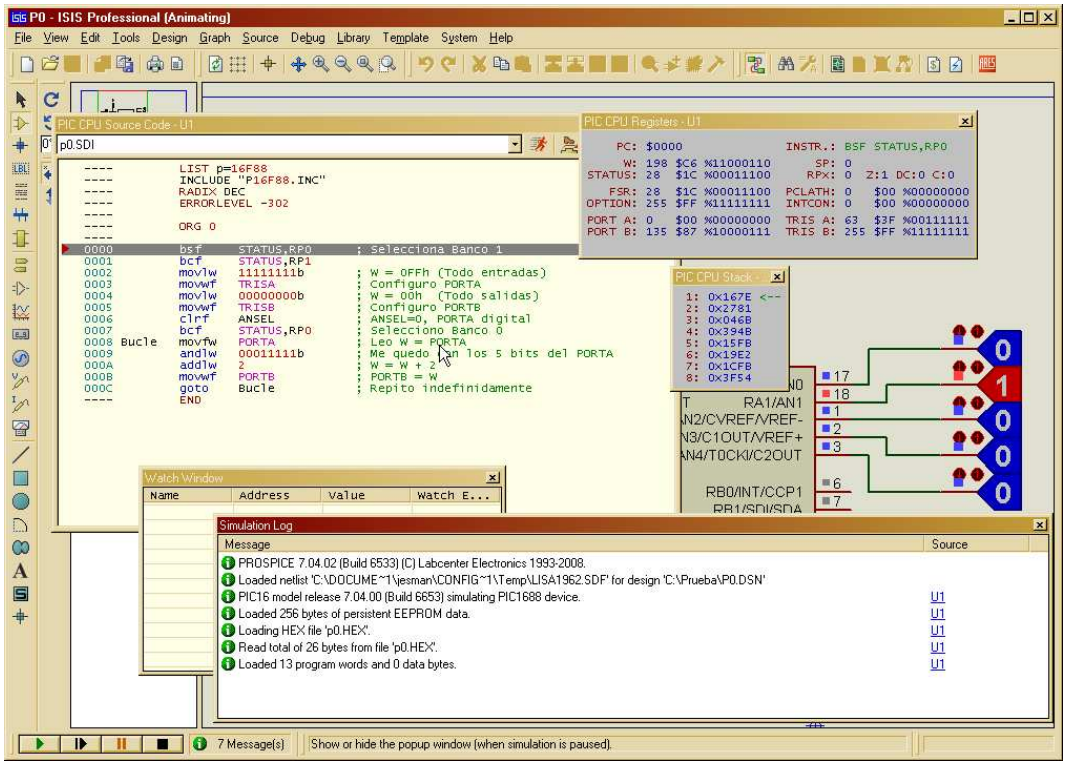
Finalmente deberemos incluir el código compilado en el microcontrolador. Seleccionamos el micro y editamos sus propiedades. En este caso rellenaremos el campo Program File con el fichero generado que tendrá extensión .HEX

Ahora podemos proceder a simular el circuito dando a la tecla PLAY de la barra de estado. Se generará una lista de nodos, se compilará todo lo necesario, y se comenzará la simulación



Nos faltará añadir algunos elementos más para ver el funcionamiento de forma más correcta (LOGICSTATE).

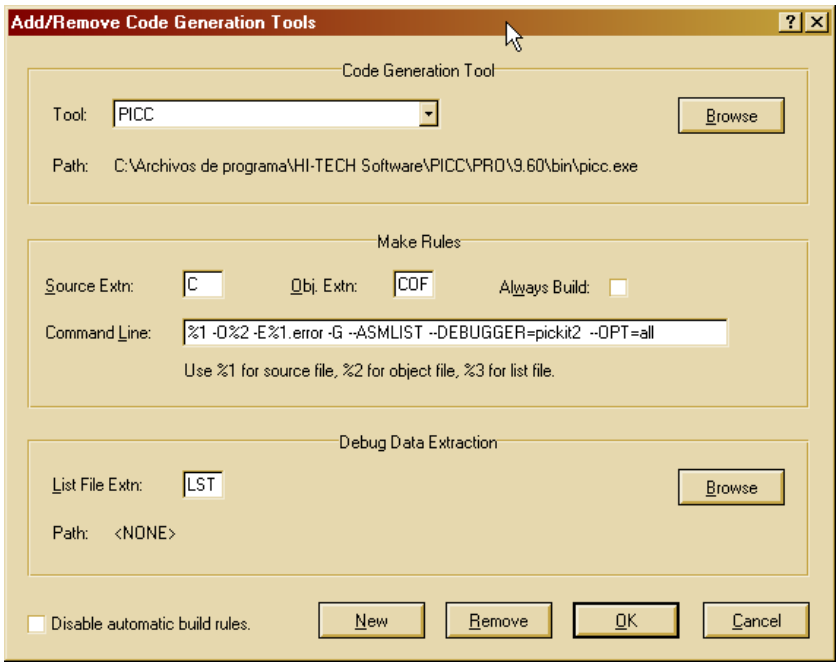
Para depurar el programa no tenemos más que pulsar la tecla PAUSE de la barra de estado y podremos ejecutar paso a paso las instrucciones, examinar la memoria, los registros, nuestras variables, etc. Todo esto se selecciona en el menú Debug.



Al tiempo que se ejecutan paso a paso las instrucciones, el esquema se actualizará encendiendo y apagando los LEDs conectados, funcionando los instrumentos virtuales colocados, los displays, etc.

1.5. Configuración de *Proteus*^{tm} para usar el compilador PICC

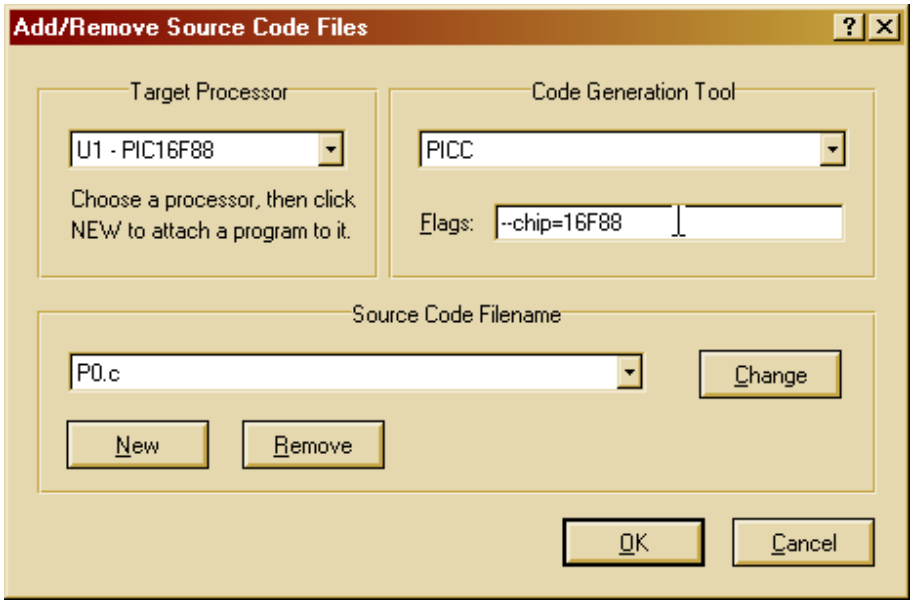
Para poder utilizarlo con ***Proteus***^{tm} se debe hacer lo siguiente en el Menú: `Source` ⇒ `Define Code Generation Tools`. Se crea una nueva entrada (PICC) que se rellena como sigue:



Entre las opciones marcadas para el compilador (Versión 9.60) están

- -ASMLIST. Genera un fichero de listado en ensamblador (.LST).
- –OPT=all. Compilación optimizada y optimización global.
- -O %2. Especificamos el fichero de salida.
- -E %1.error. Especificamos el fichero (`%1.error`) donde se escribirán los mensajes de error. Este fichero lo deberemos visualizar para comprobar los errores.
- -G. Le indicamos que queremos información para la depuración.

Al crear o añadir el fichero fuente en C, hay que indicar en el campo `Flags` con el valor `--chip=16F88` que dependerá del microcontrolador para el que se compile el código.



Crearemos un nuevo fichero fuente según el listado:

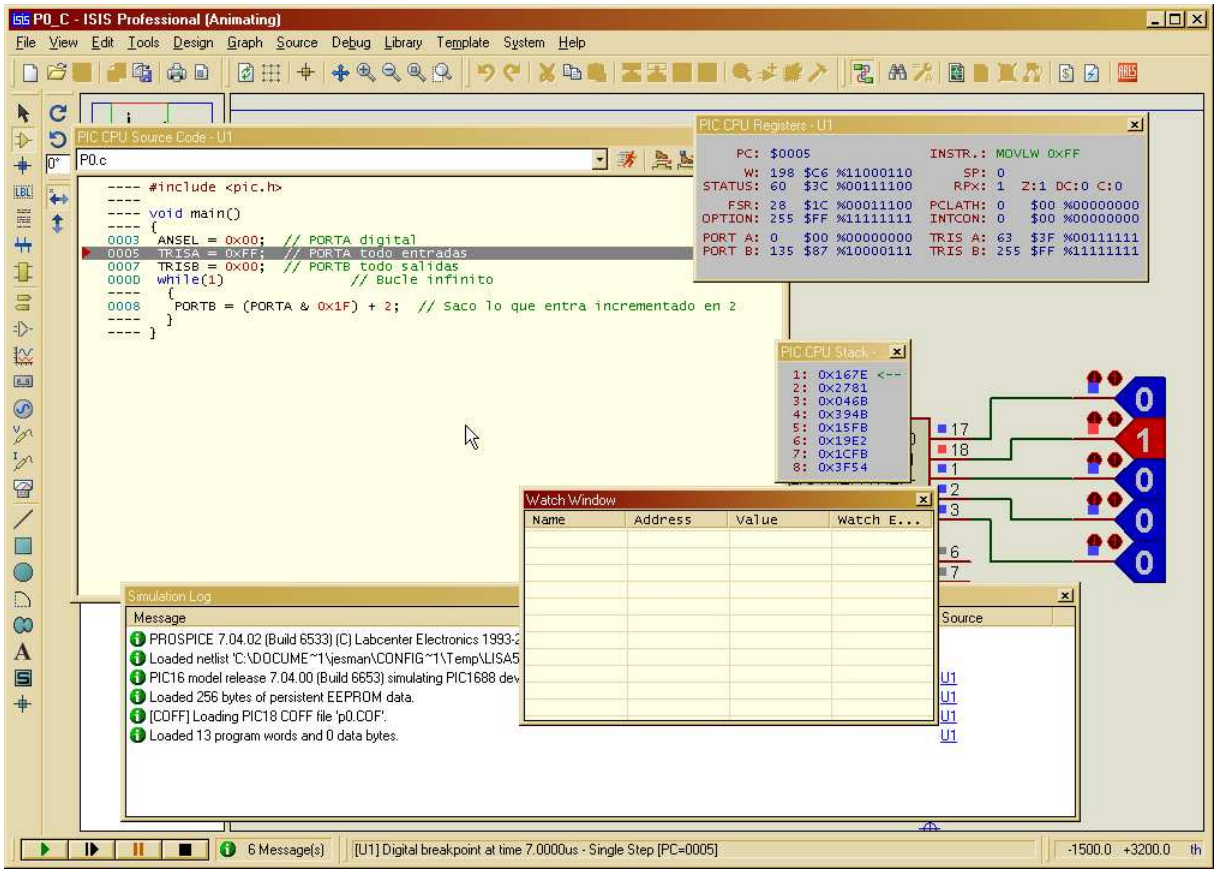
```
1  #include <pic.h>
2
3  void main()
4  {
5    TRISA=0xFF;   // PORTA todo entradas
6    TRISB=0x00;   // PORTB todo salidas
7    ANSEL=0X00;   // PORTA digital
8    while(1)      // Bucle infinito
9    {
10    PORTB = (PORTA&0x1F) + 2; // Saco lo que entra + 2
11   }
12 }
```

y le asignaremos como compilador el que acabamos de definir. Para compilar lo haremos de la forma habitual. Si da un error genérico y no aparece en el directorio de trabajo el fichero `p0.c.error` el problema es que ha habido un error al ejecutar el comando y por tanto habrá que revisar la linea introducida en el campo `Command Line`:.

```
%1 -O%2 -E%1.error -G --ASMLIST --DEBUGGER=pickit2 --OPT=all
```

El fichero a incluir en el microcontrolador será, si se desea, el `*.hex` que será el empleado desde el programa grabador (ICProg), pero si se requiere la depuración en lenguaje C se deberá emplear el fichero `*.cof`. Con esto estaremos preparados para depurar nuestras aplicaciones escritas en C con ***Proteus***^{tm} .

Si se incluye en el microcontrolador el fichero `*.cod` se verá a la vez el código C y el ensamblador generado aunque la depuración se hará a nivel ensamblador.



Capítulo 2

Práctica: Programador para los Microcontroladores PIC

Si supiese que es lo que estoy haciendo,
no lo llamaría investigación, ¿verdad?
∼ Albert Einstein ∼

2.1. Objetivo

[1] Para poder trabajar con el microcontrolador PIC16F84A/PIC16F88 y con los procesadores de señal digitales dsPIC (segunda parte de la asignatura) necesitamos construirnos un programador. El mecanismo de programación se realiza en formato serie a través de cinco líneas: VCC, /MCLR(VPP), tierra, la señal de datos RB7 y la señal de reloj RB6. El micro permite la programación en circuito (*ICSP: In Circuit Serial Programming*[2]).

2.2. Esquema básico

Listado de componentes

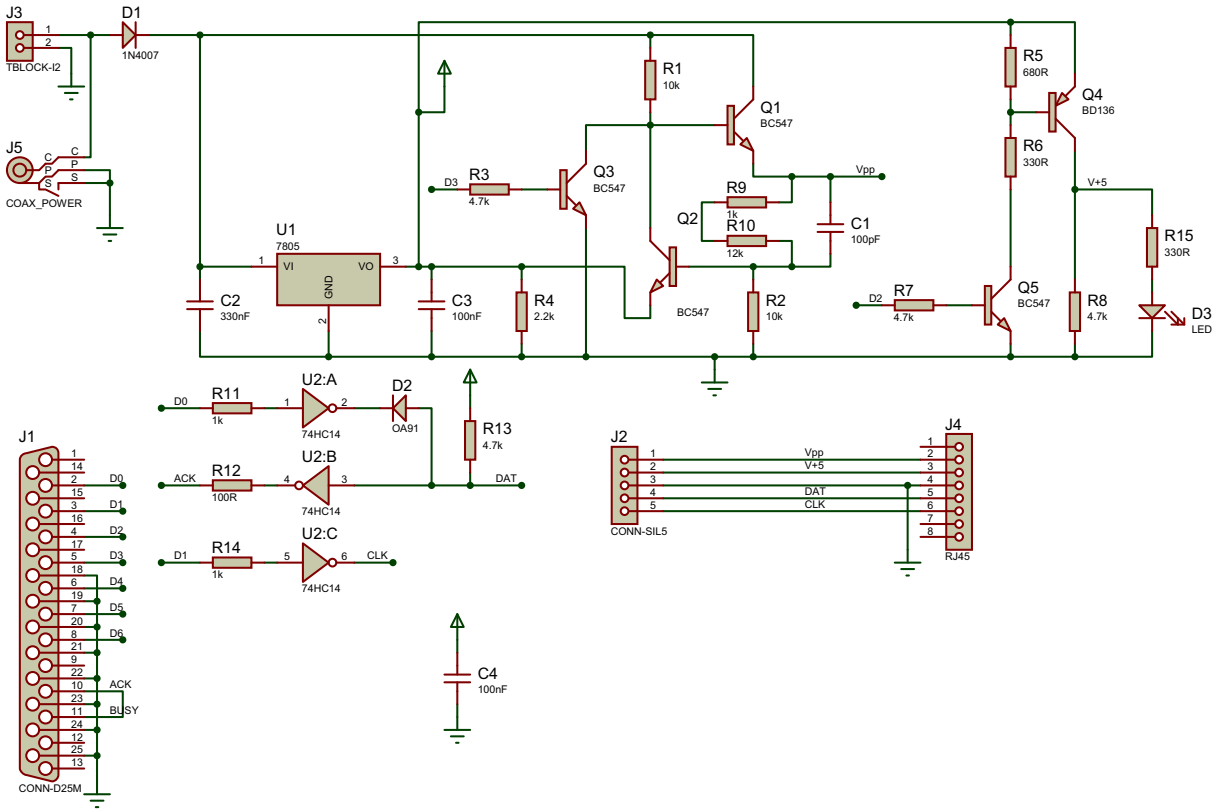
```
Bill Of Materials
=================

QTY  PART-REFS                    VALUE                              PACKAGE
---  ---------                    -----                              -------
Resistors (1/4 W)
---------
2    R1,R2                        10k                                RES40
4    R3,R7,R8,R13                 4.7k                               RES40
1    R4                           2.2k                               RES40
1    R5                           680R                               RES40
2    R6,R15                       330R                               RES40
3    R9,R11,R14                   1k                                 RES40
1    R10                          12k                                RES40
1    R12                          100R                               RES40

Capacitors
----------
1    C1                           100pF                              CAP20
1    C2                           330nF /25V                         CAP20
2    C3,C4                        100nF                              CAP20

Integrated Circuits
-------------------
```

[1] Versión de 2 de marzo de 2009
[2] Ver documento: *In-Circuit Serial Programming for PIC16F8X FLASH MCUs*

```
1     U1                          7805                                TO-220
1     U2                          74HC14                              DIL14

Transistors
-----------
4     Q1-Q3,Q5                    BC547                               TO92-100
1     Q4                          BD136                               TO126

Diodes
------
1     D1                          1N4007                              DO41
1     D2                          OA91                                DO7
1     D3                          LED Rojo                            LED 5mm

Connectors
----------
1     J1                          Conector DB25 Macho para PCB        D-25-M-R
1     J3                          Borna 2 vias atornillable para PCB 5 mm pìtch
2     J4,J5                       RJ45 hembra para PCB acodado        RJ45-90

Miscelaneous
------------
1                                 Latiguillo RJ45-RJ45 1m largo (cable de red)
1                                 Zócalo 14 pines                     DIP14
1                                 Zócalo 18 pines                     DIP18
1                                 Tira de pines macho (100th pitch)
```

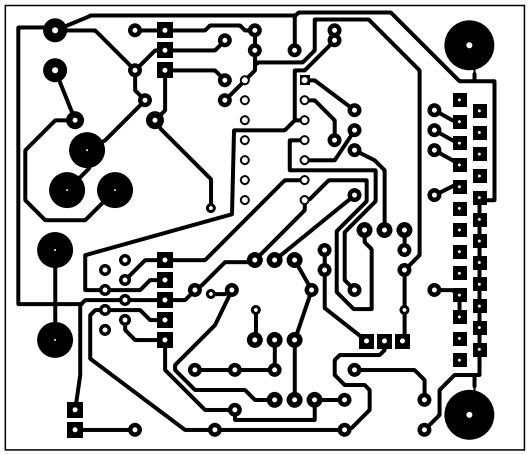
Esquema eléctrico

El esquema del programador a realizar emplea tan solo unos pocos componentes. Este programador es conocido como SETI-Prog y su esquema de conexionado se muestra en la figura:

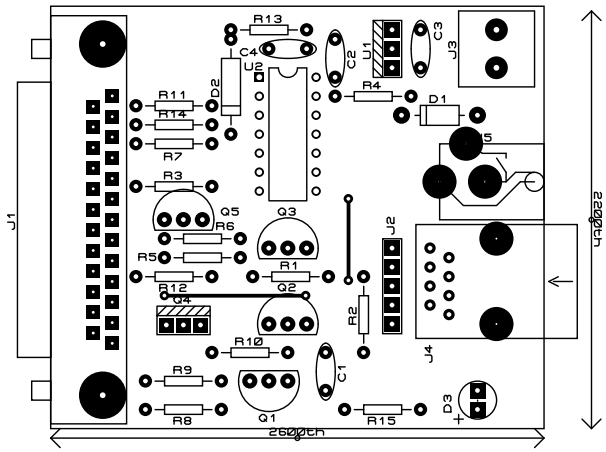


Montaje

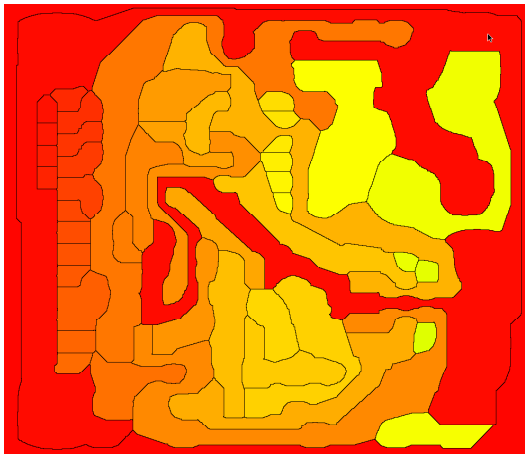
La placa de circuito impreso queda como sigue:
Cara de soldadura



Cara de componentes



Después de un proceso de engrosado de pistas obtenemos la siguiente cara de soldadura:
Cara de soldadura engrosada para su uso con la máquina fresadora CNC



2.3. Software

Para utilizar y verificar el programador necesitaremos el programa WinPIC800 (v3.61)[3] ó el programa WinPIC[4]

En WinPIC800 salvaremos la configuración del ProPIC2 como SETI-Prog y se modificará como sigue:

Con el software WinPIC la configuración será la siguiente: `DataIn=!bsy VppOnOff=!D3 VddOnOff=D2 ClockOut=!D1 DataOut=!D0`.

2.3.1. Nota

Al comienzo del programa ensamblador será necesario añadir la definición de la palabra de configuración del microcontrolador para evitarnos tener que definirlo con el software que vamos a emplear:

```
1    ; Para el PIC16F84A
2    __CONFIG _CP_OFF & _WDT_OFF & _XT_OSC & _PWRTE_ON
```

```
1    ;Para el PIC16F88
2    ;Program Configuration Register 1 (Ojo: todo escrito en la misma linea!)
3    __CONFIG    _CONFIG1, _CP_OFF & _CCP1_RB0 & _DEBUG_OFF & _WRT_PROTECT_OFF
4                & _CPD_OFF & _LVP_OFF & _BODEN_OFF & _MCLR_ON & _PWRTE_OFF
5                & _WDT_OFF & _XT_OSC
6
7    ;Program Configuration Register 2
8    __CONFIG    _CONFIG2, _IESO_OFF & _FCMEN_OFF
```

Que nos indicara que no está protegido, que el perro guardián está desactivado, que seleccionamos el modo XT para el oscilador, y que el temporizador de arranque está desactivado (activo en baja)[5].

Si el programa está escrito en lenguaje C, entonces:

```
1    __CONFIG(WDTDIS & XT & UNPROTECT & PWRTDIS); // PIC16F84A
```

```
1    __CONFIG(WDTDIS & XT & UNPROTECT & PWRTDIS & CCPRB0
2            & DEBUGDIS & LVPDIS & BORDIS & MCLREN
3            & FCMDIS & IESODIS); // PIC16F88
```

En algunos casos Proteus no hará caso de esto y será necesario comprobar mediante el `Simulation log` que la palabra de configuración es la correcta.

[3] http://www.winpic800.com
[4] http://freenet-homepage.de/dl4yhf/winpicpr.html
[5] Para el PIC16F88 indica algunas cosas más.