Ejemplo de utilización del programador universal EETOOLS TOPMAX con el microcontrolador Microchip PIC 16F690 y el compilador CCS PCWH 3.249

Versión 1.0

Autor: Marcos Morales Pallarés

Escola Universitària d'Enginyeria Tècnica Industrial de Barcelona
Laboratorio de Proyectos. Unidad de Electrónica
Universitat Politècnica de Catalunya

Barcelona, noviembre del 2006

Índice

1. Objetivo de este documento 2

2. Aplicación de ejemplo a montar y programar 3

3. Procedimiento 4

3.1. Montaje del circuito en protoboard 5

3.2. Instalación de CCS PCWH 3.249 6

3.3. Instalación del TOPMAX 8

3.4. Creación del programa a ejecutar en el PIC16F690..... 13

3.5. Programación del microcontrolador 28

3.6. Comprobación del correcto funcionamiento 35

1. Objetivo de este documento

El programador universal TOPMAX de la firma EETOOLS está disponible en el laboratorio de PFCs para aquellos/as proyectistas que lo necesiten. El gran número de circuitos integrados que permite programar lo convierte en una herramienta muy versátil.

En muchas ocasiones los estudiantes necesitan volcar programas en microcontroladores, grabar datos en EEPROMs, programar PALs o GALs. Estos dispositivos pueden ser programados con el TOPMAX. Sólo precisamos seleccionar el integrado a emplear, generar el archivo en formato hexadecimal que queramos volcar y programar el integrado.

En el caso de no disponer de los drivers necesarios para programar algún chip, siempre podremos consultar la web del fabricante (http://www.eetools.com/) y comprobar la existencia de actualizaciones y soporte para nuevos dispositivos.

Para que sirva de pauta y a modo de ejemplo, proporcionamos al lector este manual. En él se recogen los pasos necesarios para llevar a cabo el montaje más básico que existe para todo microcontrolador: el control de encendido de un LED.

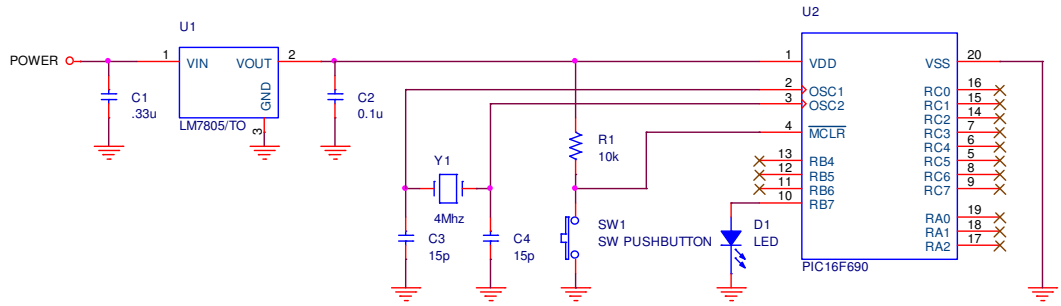
Para ello se ha escogido el PIC16F690, de la casa Microchip. La familia 16FXXX es reconocida por su versatilidad, facilidad de uso y robustez. Nos hemos decantado por el compilador CCS PCWH, pues se trata de una de las herramientas más empleadas al programar código en lenguaje C para los PIC, además de poseer gran cantidad de librerías y ejemplos.

El documento está estructurado a modo de 'guía de pasos' a realizar para llevar a cabo este ejemplo en concreto con éxito. Dada la naturaleza de este documento, no se ha creído conveniente entrar en detalles de funcionamiento.

Tomando este manual como punto de partida, no debería resultar difícil para el lector realizar procesos similares para otros microcontroladores. Obviamente, si lo que el lector precisa es volcar datos a otros circuitos integrados como por ejemplo EPROMs o GALs, deberá emplear otras herramientas software, aunque podrá guiarse de la sección dedicada exclusivamente al programador TOPMAX y el volcado de datos, teniendo en cuenta que algunos de los parámetros aquí expuestos variarán.

2. Aplicación de ejemplo a montar y programar

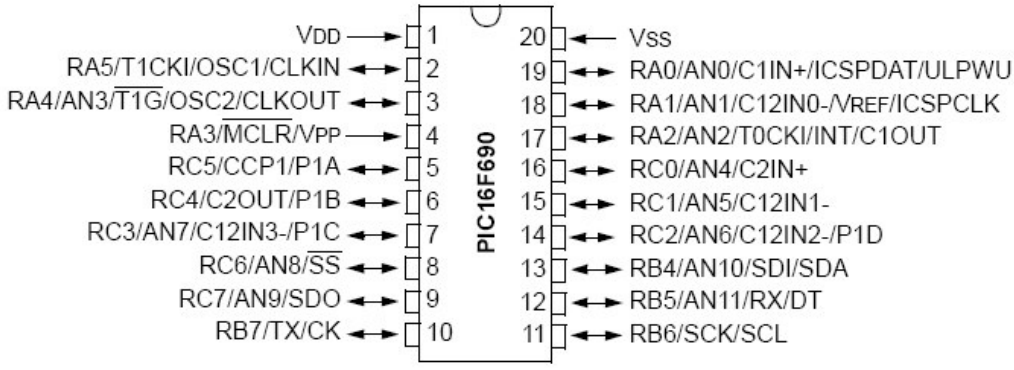
La siguiente figura muestra el circuito que implementaremos en nuestra protoboard.



Aspectos a observar:

- Alimentación regulada por un 7805 a 5 V. Los condensadores de desacoplo de 0.33 uF y 0.1 uF resultan imprescindibles.
- Empleo de un cristal de cuarzo de 4 MHz. En general se trata de la velocidad mínima empleada en todo montaje con microcontrolador.
- Botón de reset.
- LED conectado al pin RB7. Obsérvese la conexión directa al pin del micro, sin emplear ninguna resistencia limitadora de corriente. En general los PIC proporcionan la corriente necesaria para alimentar un diodo LED de forma directa. Esto nos sirve para nuestro ejemplo, aunque en la práctica se recomienda el uso de dichas resistencias.
- No hemos incluído un condensador de desacoplo para el micro, aunque para aplicaciones más complejas puede llegar a ser imprescindible.

A continuación se muestra la distribución de pines proporcionada en el datasheet del fabricante. Consúltese en caso de tener alguna duda.



3. Procedimiento

Los subapartados que se exponen a continuación deben seguirse de forma lineal.

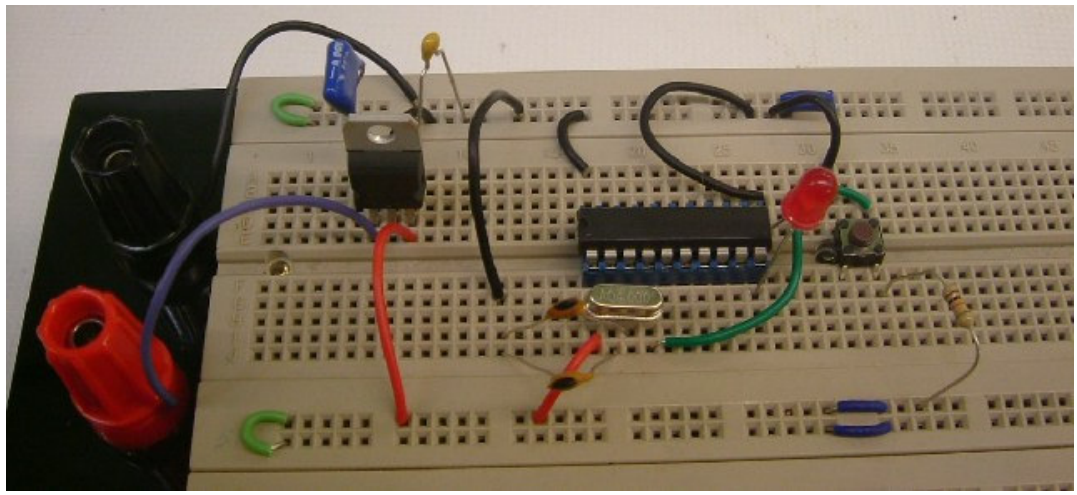
No hace falta decir que, en el caso de tener ya instalado el compilador CCS PCWH o el software del programador universal TOPMAX podremos prescindir de los apartados 3.2 y 3.3, respectivamente.

De todos modos, si la versión del driver que estamos utilizando no soporta el dispositivo que queremos programar, será necesario consultar en la web del fabricante si existe la actualización correspondiente. En el momento de elaborar este manual, ha sido necesario realizar dicha actualización. Puede tomarse el punto 3.3 como ejemplo para realizar esta tarea.

Dentro de cada apartado observaremos que los pasos se encuentran numerados. En cada uno de ellos se incluye una breve explicación de lo que se debe realizar o, en su defecto, una imagen explicativa por sí sola. Los apartados se encuentran separados por barras negras horizontales.

3.1. Montaje del circuito en protoboard

1. El aspecto del circuito una vez montado es el siguiente:

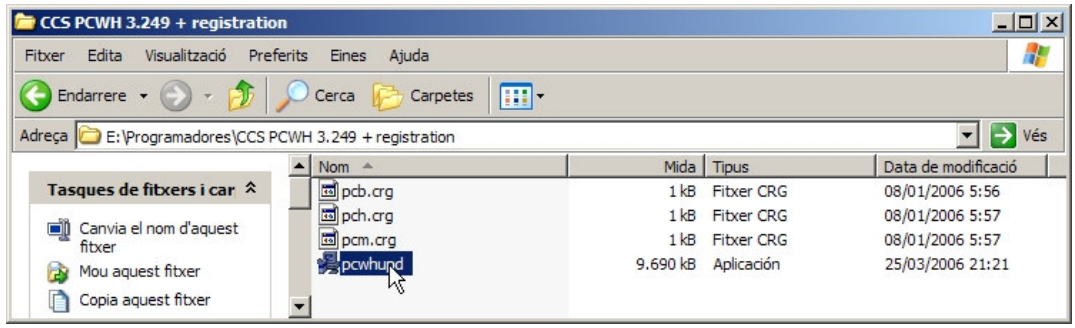


Obsérvese el empleo de un zócalo de 20 pines para nuestro micro. Resulta muy recomendable su uso puesto que al programar varias veces el micro y comprobar su funcionamiento estaremos insertándolo y extrayéndolo de la protoboard continuamente, con lo que podríamos dañar físicamente alguna de sus patitas.

Nuestro objetivo será volcar y ejecutar un programa que encienda y apague el LED de forma intermitente e indefinida. El porqué de realizar un montaje y un programa tan simple es porque nos permite comprobar de un modo sencillo que el proceso de compilación y volcado se ha realizado de forma correcta.

3.2. Instalación de CCS PCWH 3.249

1. Ejecutar 'pcwhund.exe':

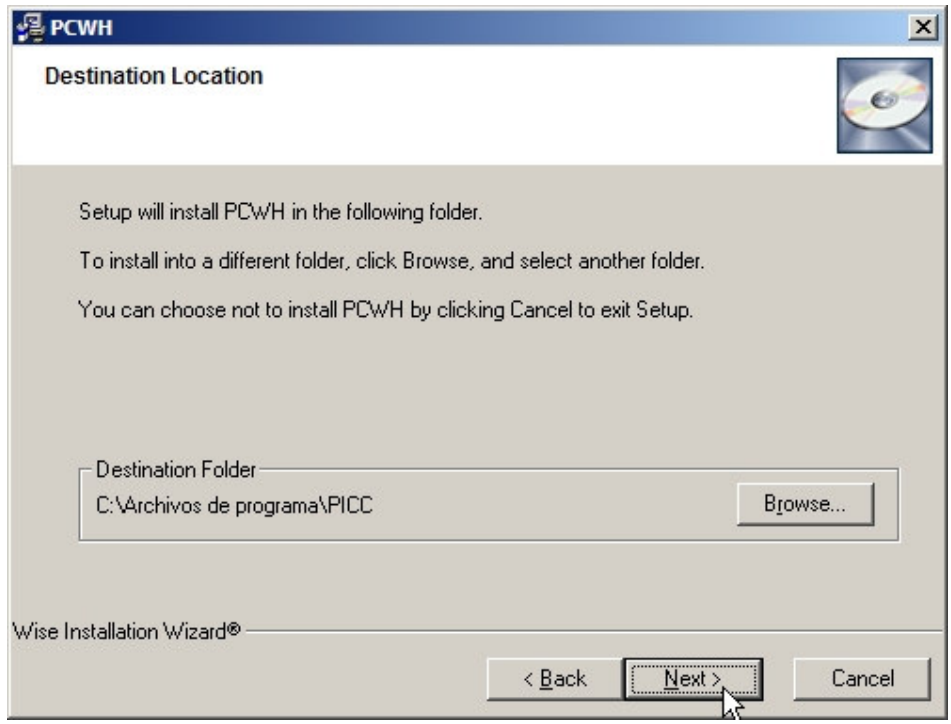


---

2. Clic en 'Next' > Clic en 'Next' > Clic en 'Next'

---

3. Seleccionar la carpeta destino para instalar el programa:

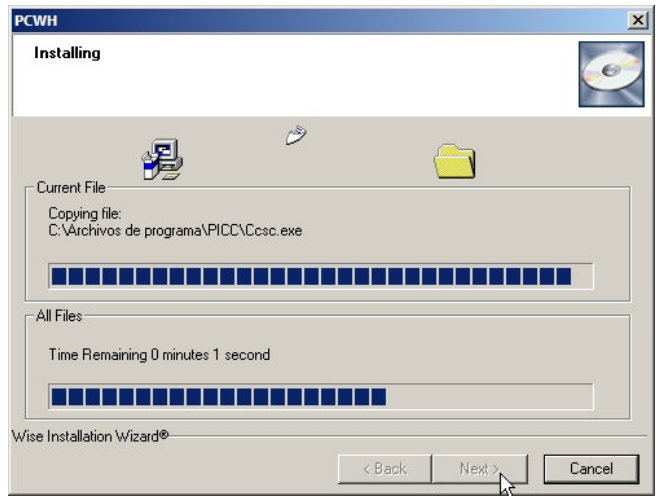


---

4. Clic en 'Next'

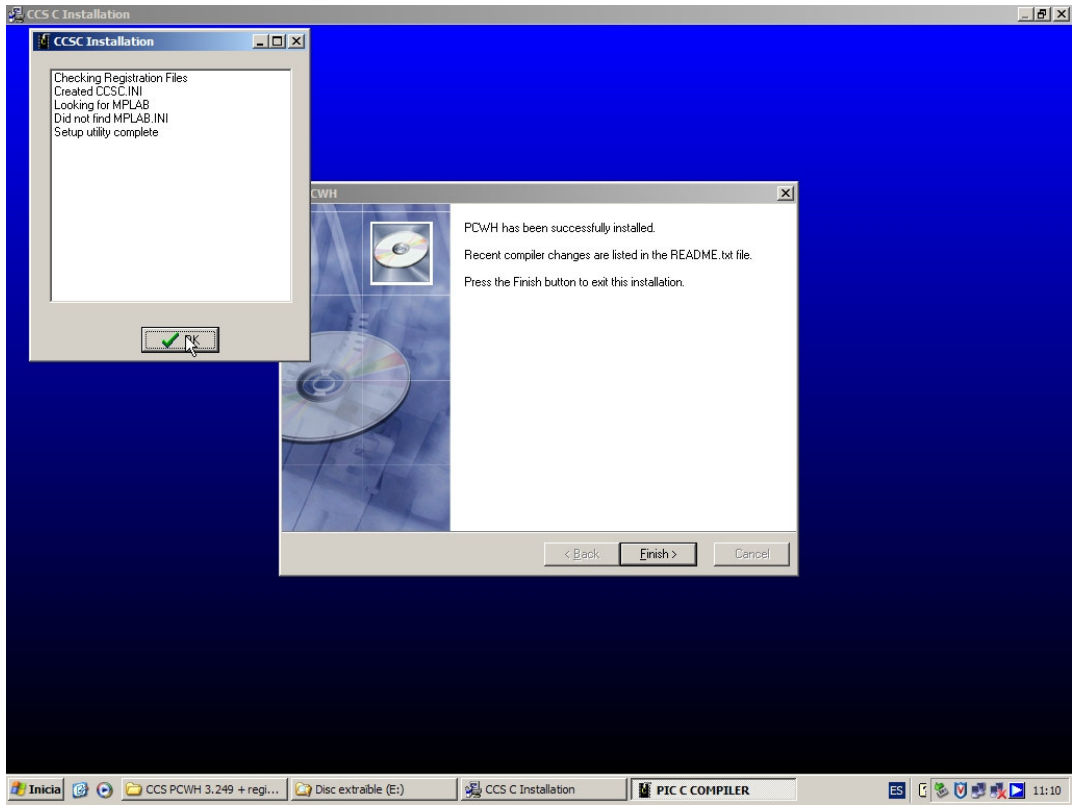
---

5. Esperar a que acabe la copia de archivos:



---

6. Si todo ha ido bien, aparecerán los siguientes cuadros informativos:



Con lo que finalizamos con la instalación de CCS PCWH 3.249.

3.3. Instalación del TOPMAX

1. El pack del programador universal TOPMAX incluye los siguientes elementos:
    - Módulo programador.
    - CD con el software de utilización (versión de fábrica).
    - Cable para el puerto paralelo.
    - Cable de alimentación.

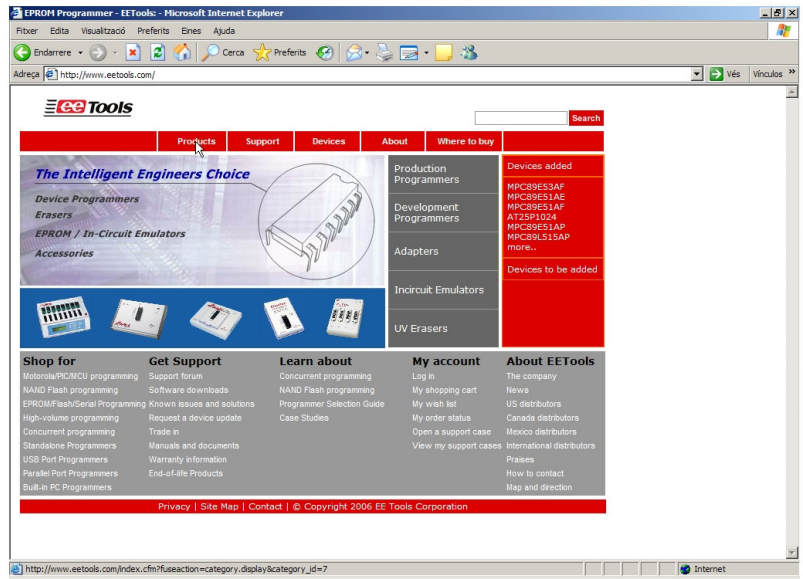
En caso de faltar algún elemento será preciso avisar a uno de los responsables del laboratorio.

---

2. Desde nuestro navegador, introducimos la dirección de la web del fabricante:
http://www.eetools.com/

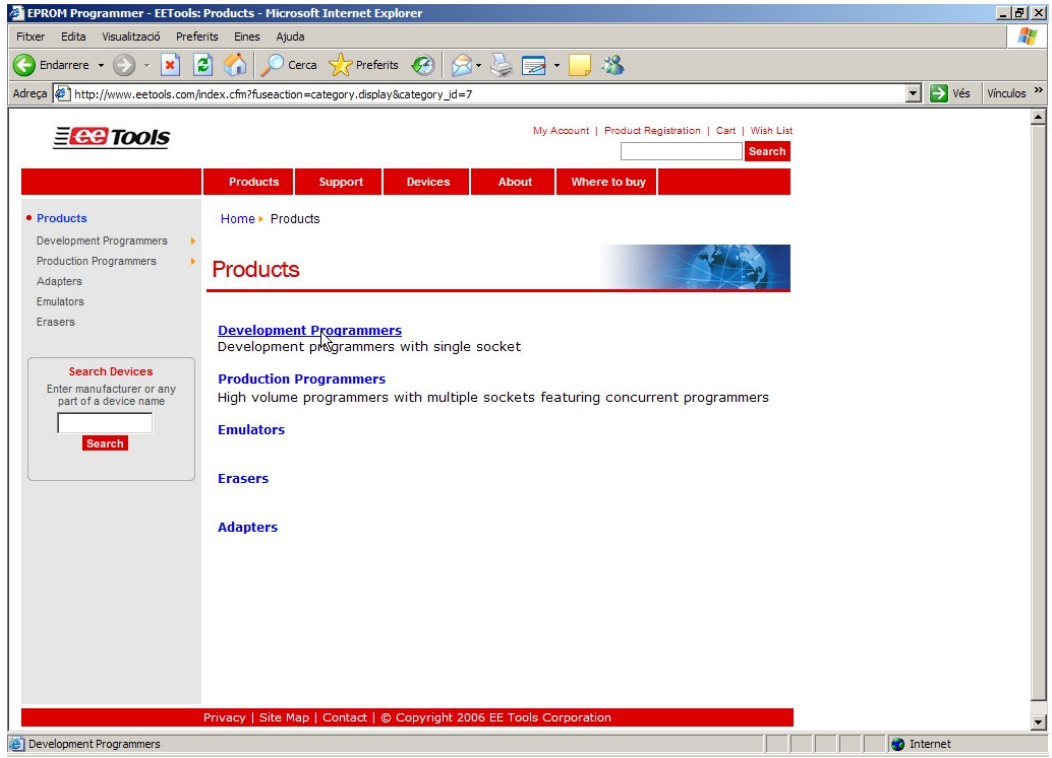
---

3. Click en 'Products'.

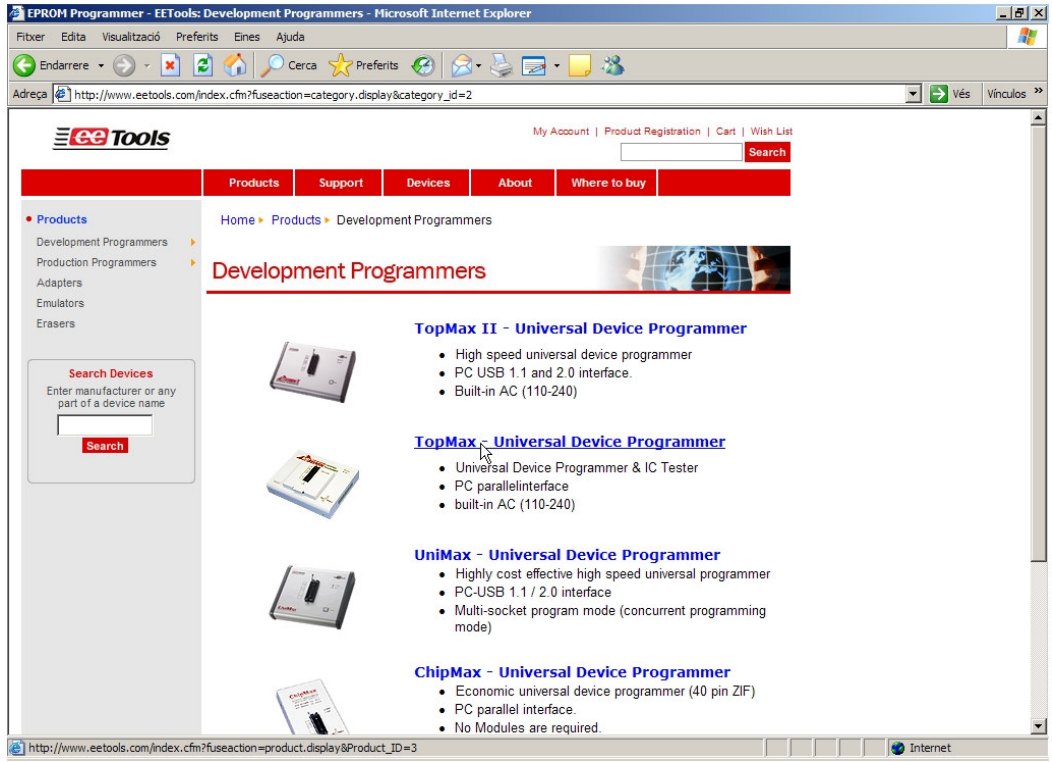


---

4. Click en 'Development Programmers'.

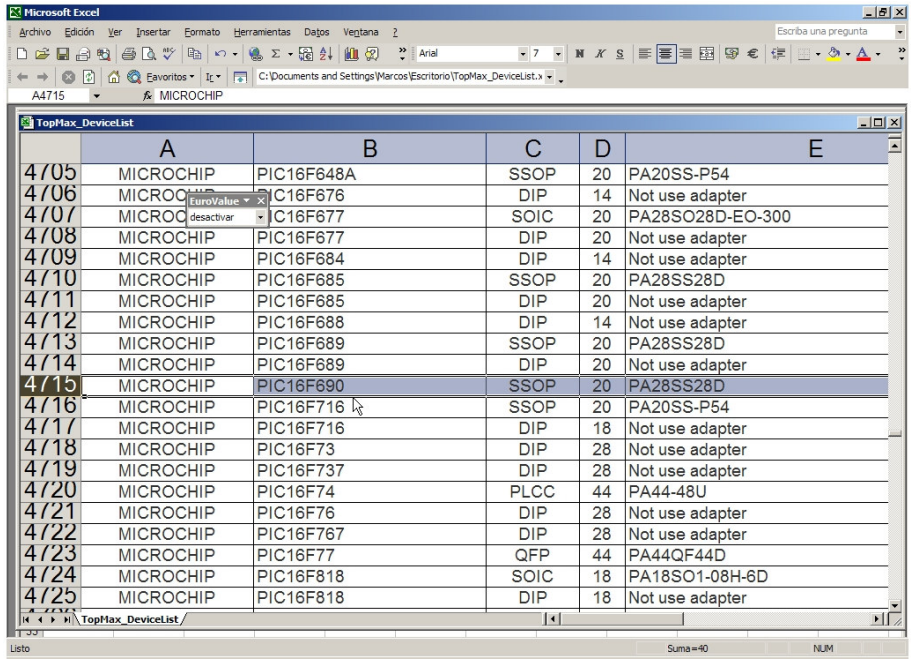


5. Click en 'TopMax – Universal Device Programmer'.



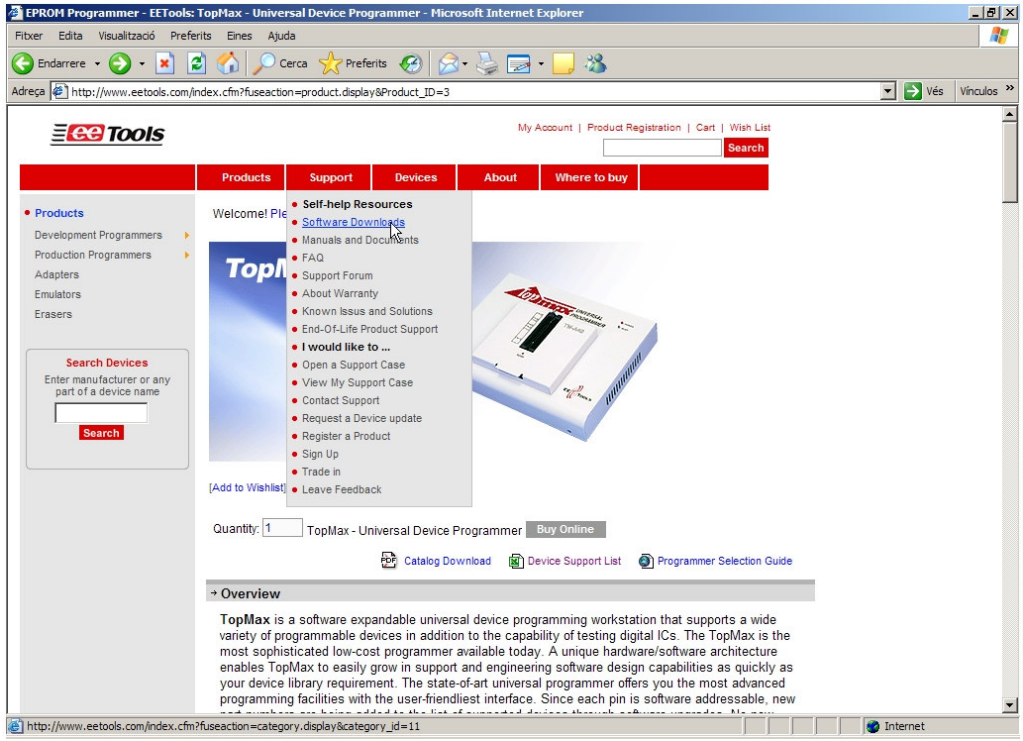
6. Vamos a consultar la lista de dispositivos soportados hasta la fecha. Para ello nos bajaremos el fichero en excel del link 'Device Support List'.

7. Al navegar por el archivo, comprobamos que, efectivamente, nuestro 16F690 se encuentra como dispositivo programable por el TOPMAX.

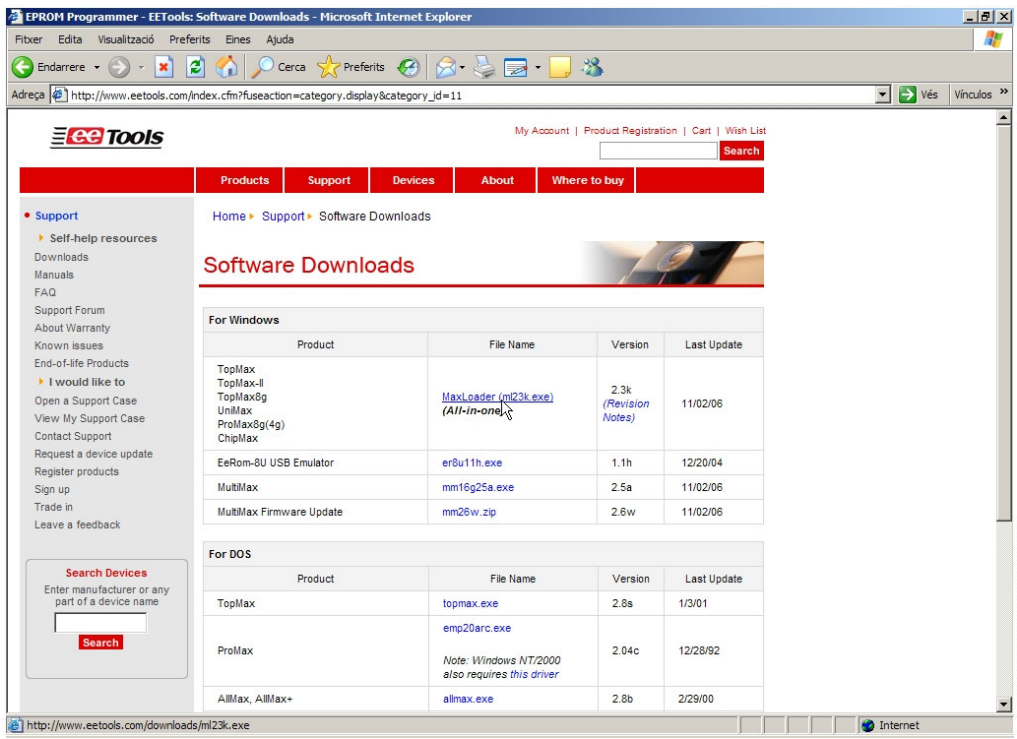


	A	B	C	D	E
4705	MICROCHIP	PIC16F648A	SSOP	20	PA20SS-P54
4706	MICROCHIP	PIC16F676	DIP	14	Not use adapter
4707	MICROCHIP	PIC16F677	SOIC	20	PA28SO28D-EO-300
4708	MICROCHIP	PIC16F677	DIP	20	Not use adapter
4709	MICROCHIP	PIC16F684	DIP	14	Not use adapter
4710	MICROCHIP	PIC16F685	SSOP	20	PA28SS28D
4711	MICROCHIP	PIC16F685	DIP	20	Not use adapter
4712	MICROCHIP	PIC16F688	DIP	14	Not use adapter
4713	MICROCHIP	PIC16F689	SSOP	20	PA28SS28D
4714	MICROCHIP	PIC16F689	DIP	20	Not use adapter
4715	MICROCHIP	PIC16F690	SSOP	20	PA28SS28D
4716	MICROCHIP	PIC16F716	SSOP	20	PA20SS-P54
4717	MICROCHIP	PIC16F716	DIP	18	Not use adapter
4718	MICROCHIP	PIC16F73	DIP	28	Not use adapter
4719	MICROCHIP	PIC16F737	DIP	28	Not use adapter
4720	MICROCHIP	PIC16F74	PLCC	44	PA44-48U
4721	MICROCHIP	PIC16F76	DIP	28	Not use adapter
4722	MICROCHIP	PIC16F767	DIP	28	Not use adapter
4723	MICROCHIP	PIC16F77	QFP	44	PA44QF44D
4724	MICROCHIP	PIC16F818	SOIC	18	PA18SO1-08H-6D
4725	MICROCHIP	PIC16F818	DIP	18	Not use adapter

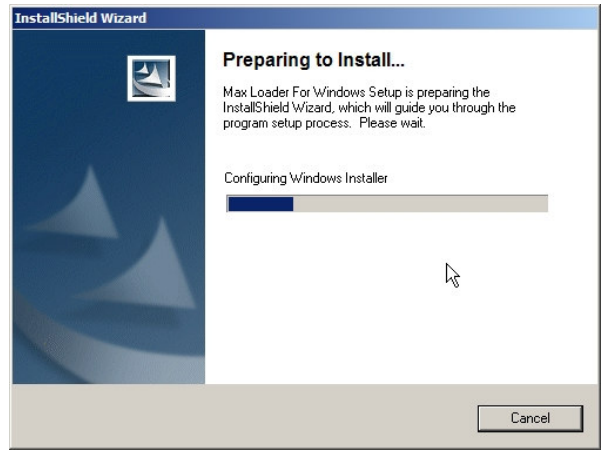
8. Vamos a descargar la última versión de los drivers:
Click en 'Support' > Click en 'Software Downloads'



9. El archivo a descargar es ml23k.exe (link MaxLoader, apartado para Windows).



10. Ejecutamos el archivo .exe descargado.



---

11. Click en 'Next'

---

12. Seleccionar la carpeta de destino.

---

13. Esperamos a que acabe la transferencia de ficheros.

---

14. Click en 'Finish'.

3.4. Creación del programa a ejecutar en el PIC16F690

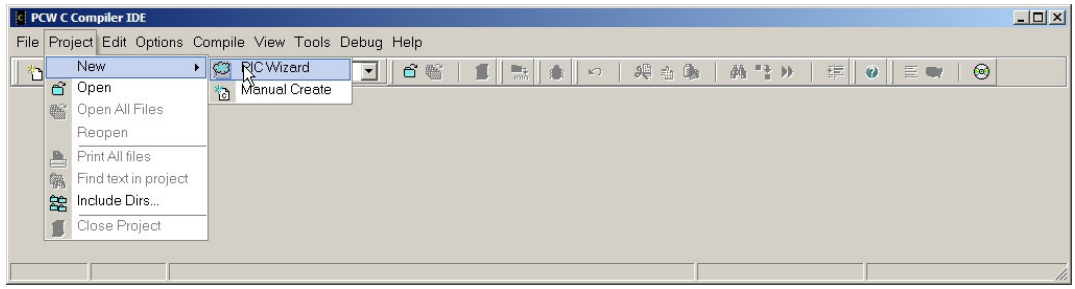
1. Iniciamos el CCS PCW PIC C Compiler:

Inicio > Programas > PIC-C > PIC C Compiler

---

2. Ejecutamos el asistente para la creación de proyectos:

Project > New > PIC Wizard

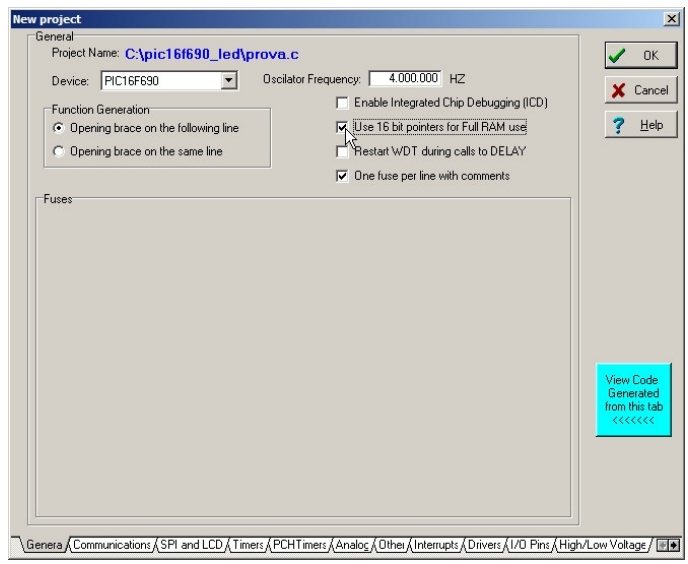


---

3. Se nos preguntará por el nombre y la ubicación de nuestro proyecto. Es recomendable dedicar un directorio exclusivamente al proyecto, pues el número de archivos generados puede ser considerable.

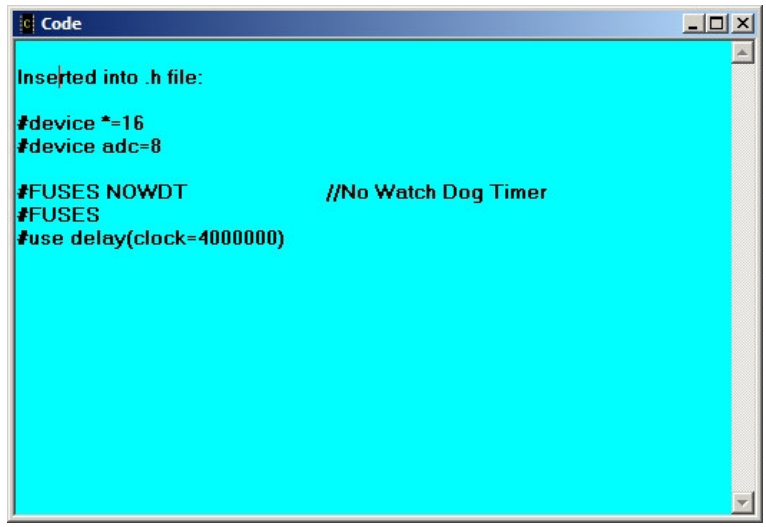
---

4. Aparecerá el asistente abierto por la pestaña 'General':
En 'Device' seleccionamos nuestro modelo de microcontrolador (PIC16F690).
Introducimos la frecuencia de trabajo de nuestro cristal oscilador (4000000).
Seleccionamos el uso de punteros de 16 bits para poder utilizar toda la RAM del micro.



---

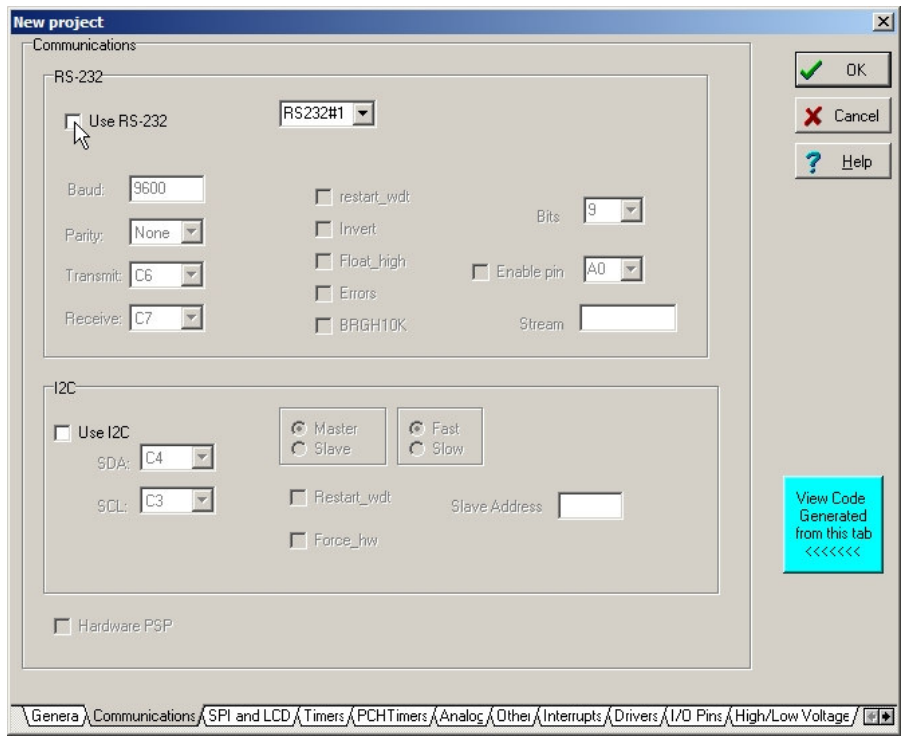
5. Si clicamos en 'View Code Generated from this tab' observaremos la ventana inferior.



En ella podemos observar el código C que el asistente generará (y dónde lo insertará) teniendo en cuenta las opciones seleccionadas de la pestaña del asistente en la que nos encontramos.

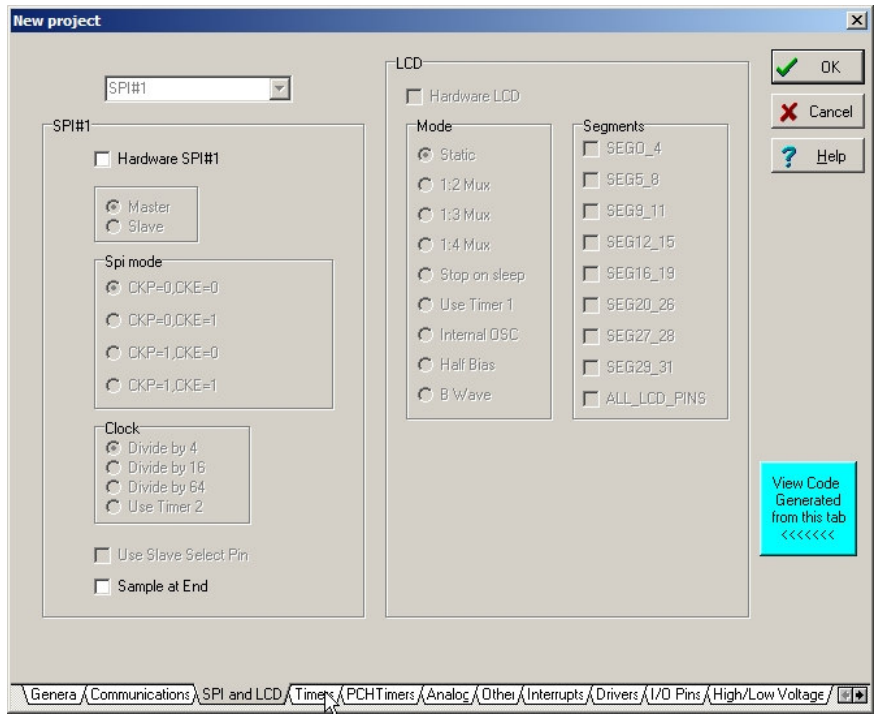
---

6. Pestaña 'Communications':

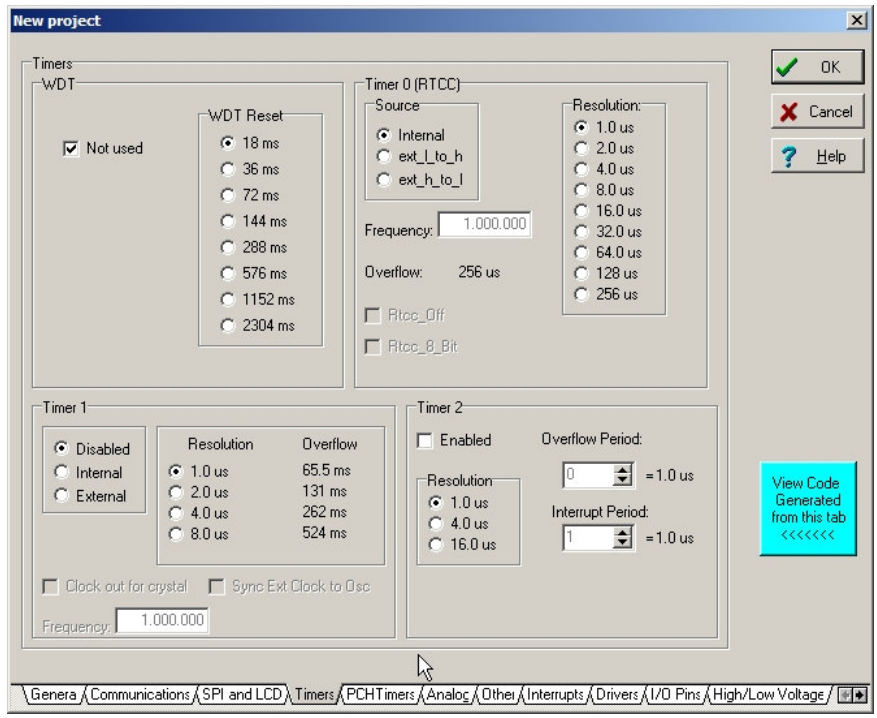


---

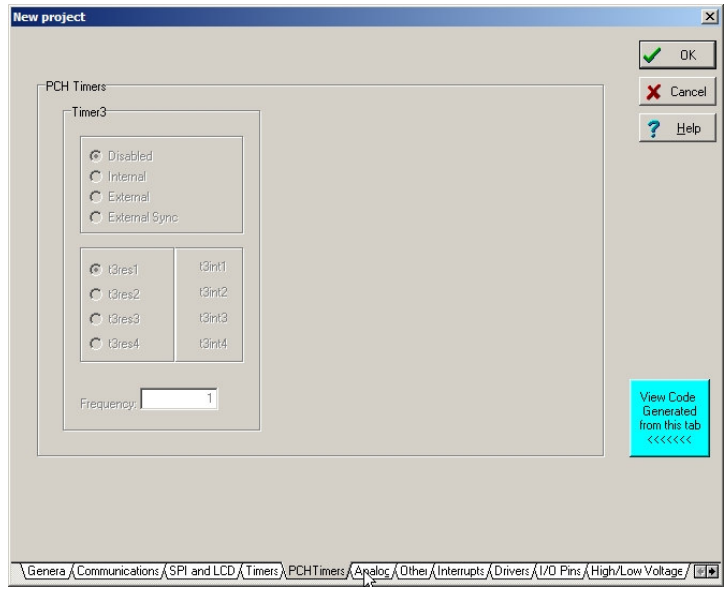
7. Pestaña 'SPI and LCD':



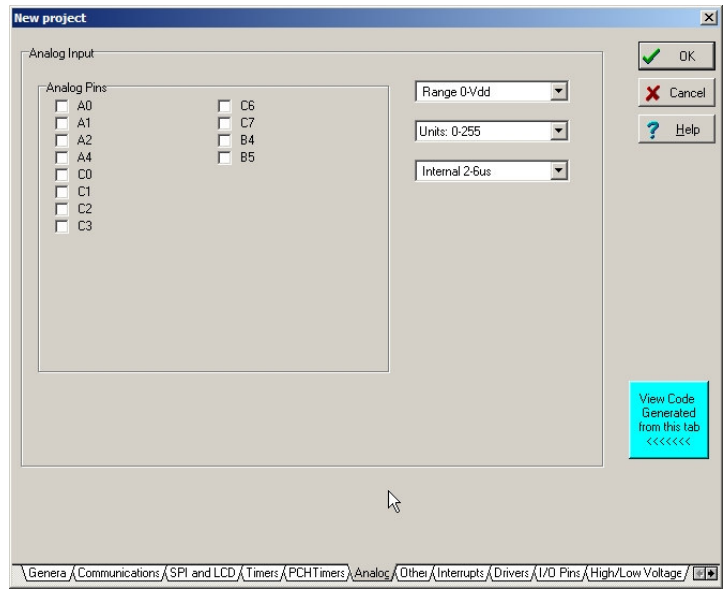
8. Pestaña 'Timers':

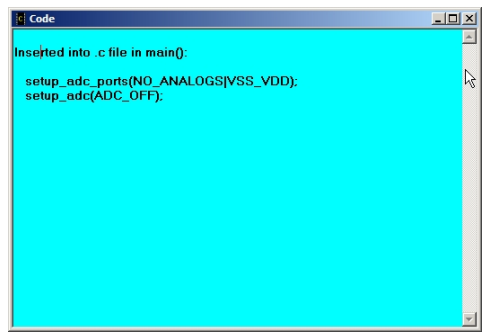


9. Pestaña ‘PCH Timers’:

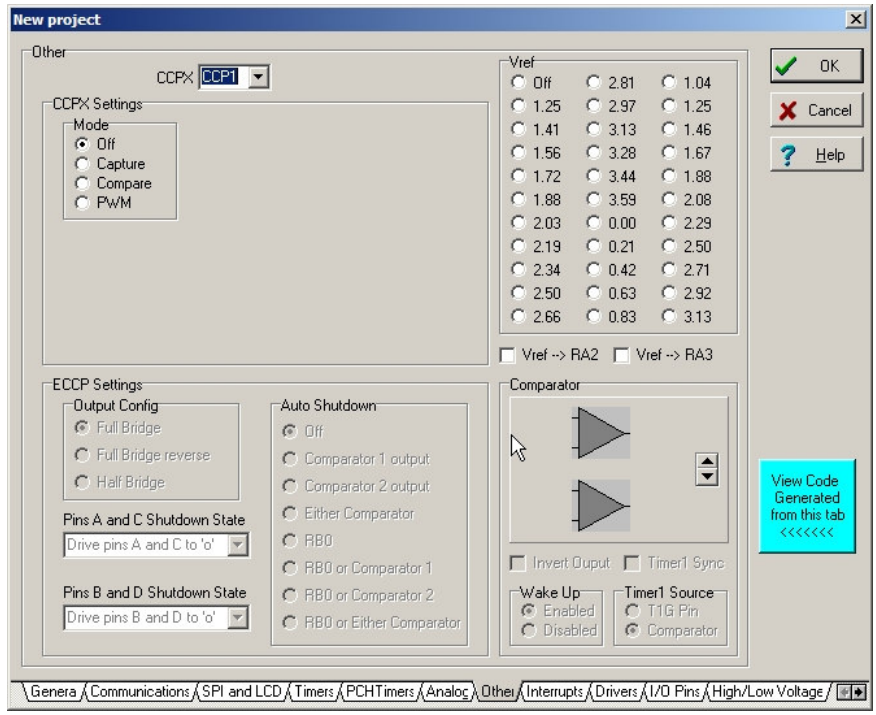


10. Pestaña ‘Analog’:

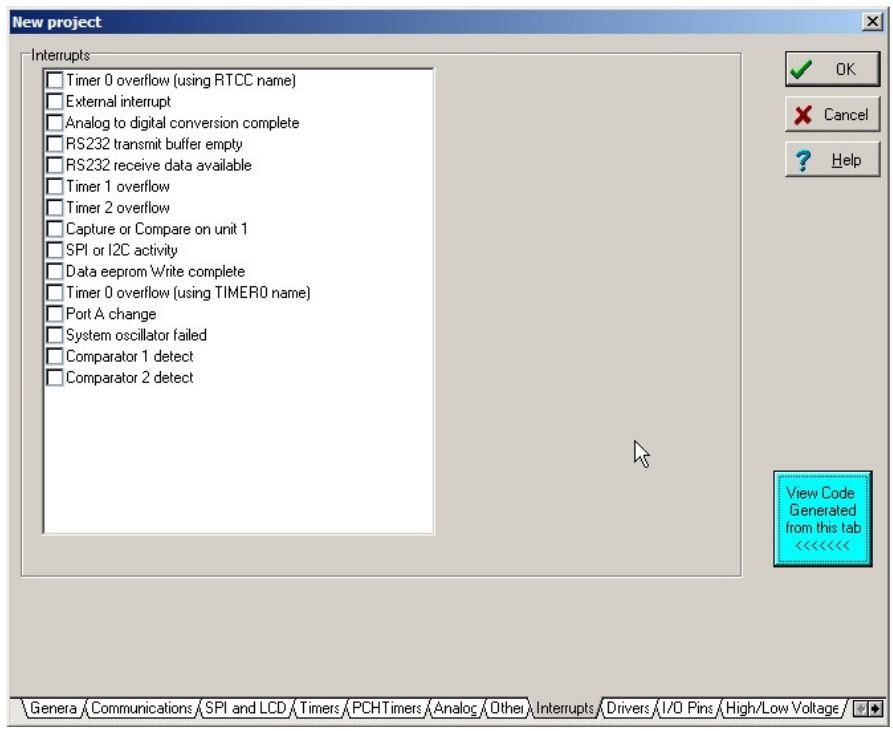




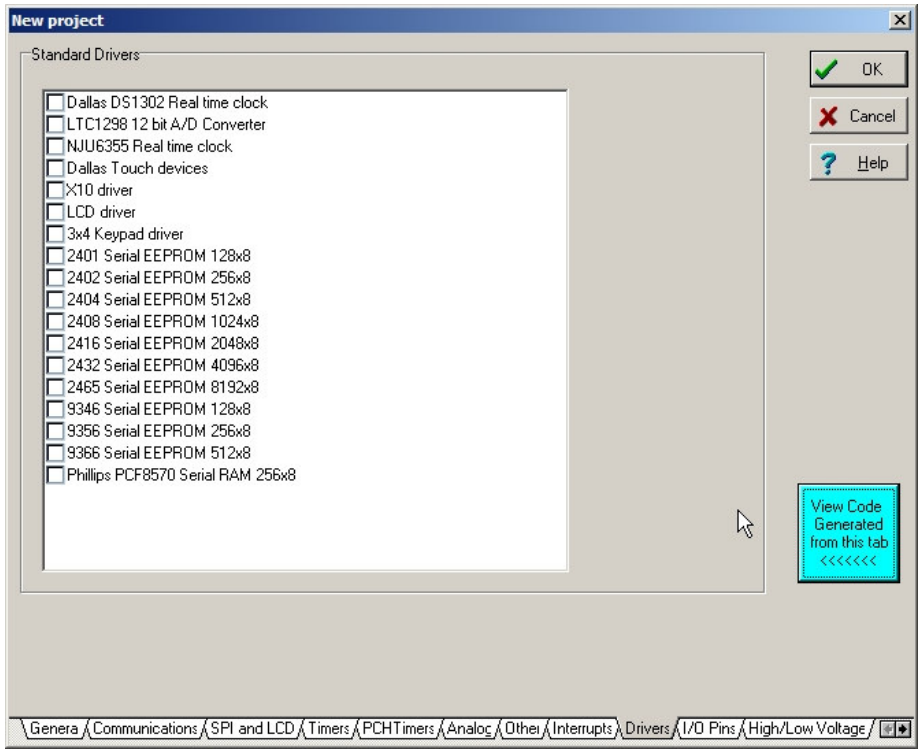
11. Pestaña 'Other':



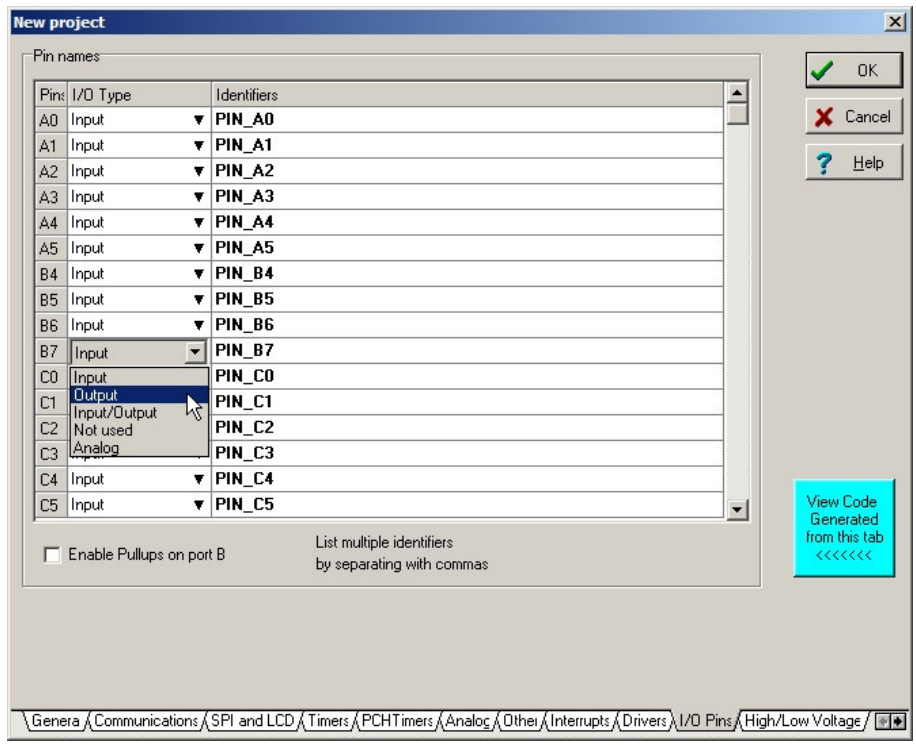
12. Pestaña 'Interrupts':



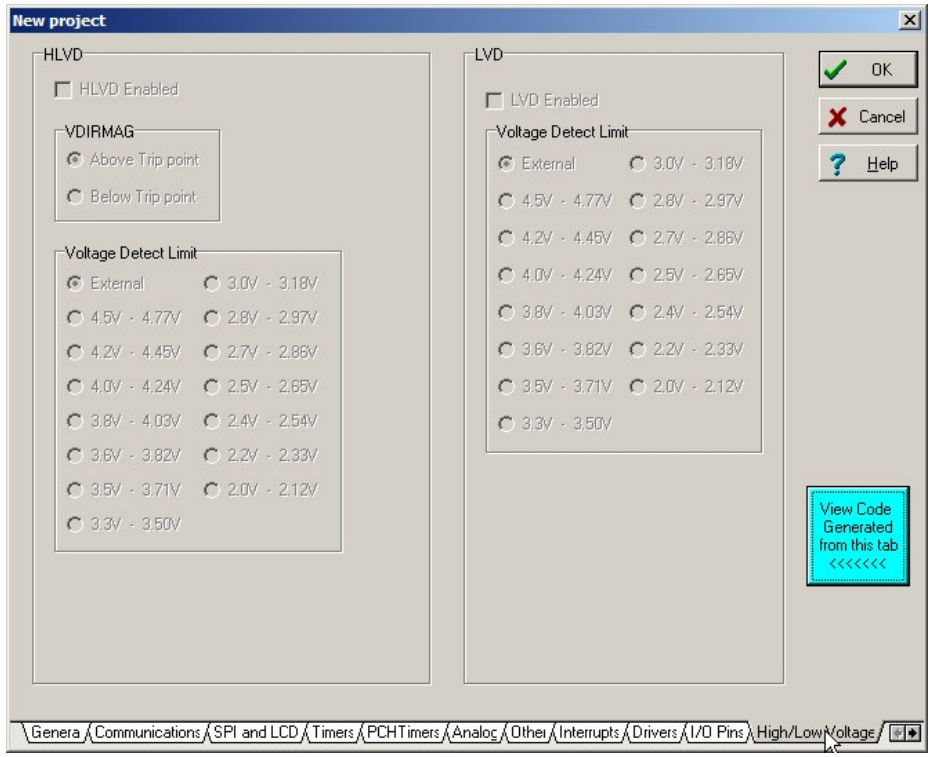
13. Pestaña 'Drivers':



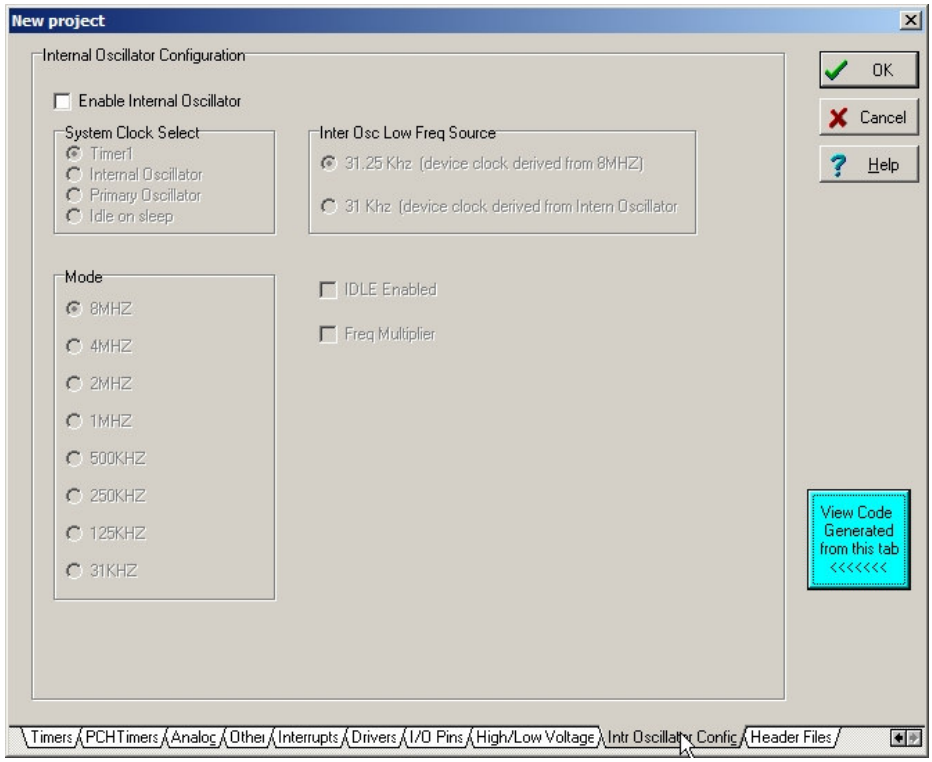
14. Pestaña 'I/O Pins':



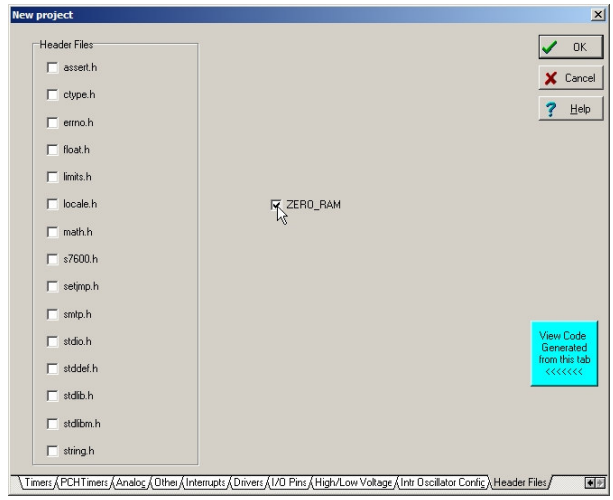
15. Pestaña 'High/Low Voltage':

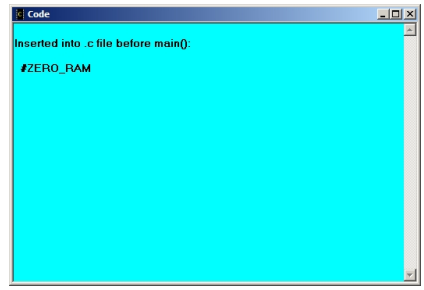


16. Pestaña 'Intr Oscillator Config':



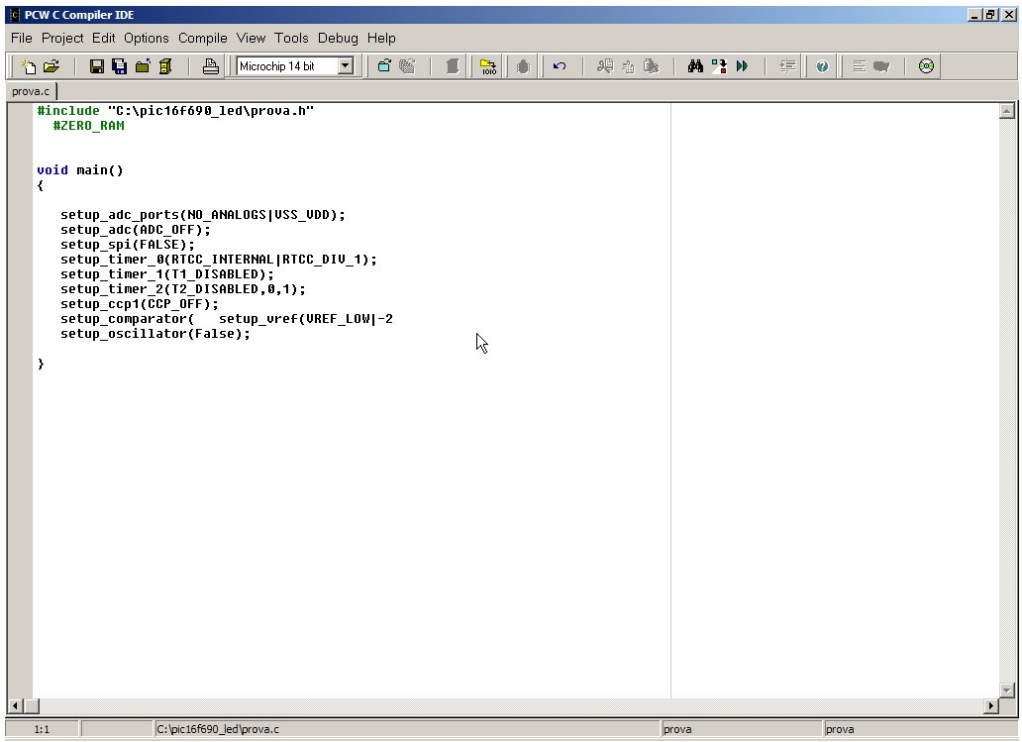
17. Pestaña 'Header Files':





---

18. Finalmente clicamos en 'OK' y obtendremos el siguiente código:



Podemos realizar varias observaciones:

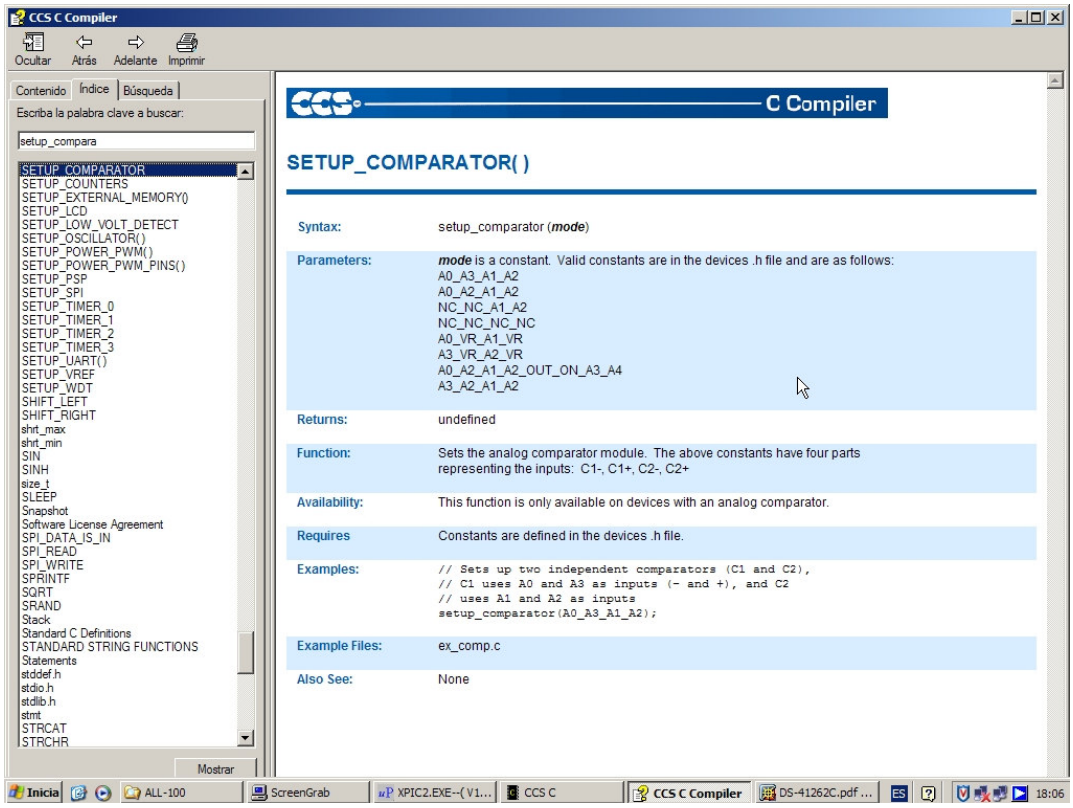
- Nuestro proyecto se ha creado con un archivo de código principal 'prova.c', y un archivo cabecera 'prova.h'.
- En la rutina principal (main) podemos observar una serie de llamadas a funciones de configuración que proporciona el propio compilador: setup_xxx.
- El asistente ha redactado un par de instrucciones con errores de sintaxis: setup_comparator(x) y setup_vref(x). Se trata de un bug de CCS PCWH que deberemos solucionar.

---

19. Consultamos la ayuda para conocer la sintaxis de setup_comparator(x):

Help > Contents

Clicamos en la pestaña 'Índice' y accedemos a la entrada SETUP_COMPARATOR de la lista.



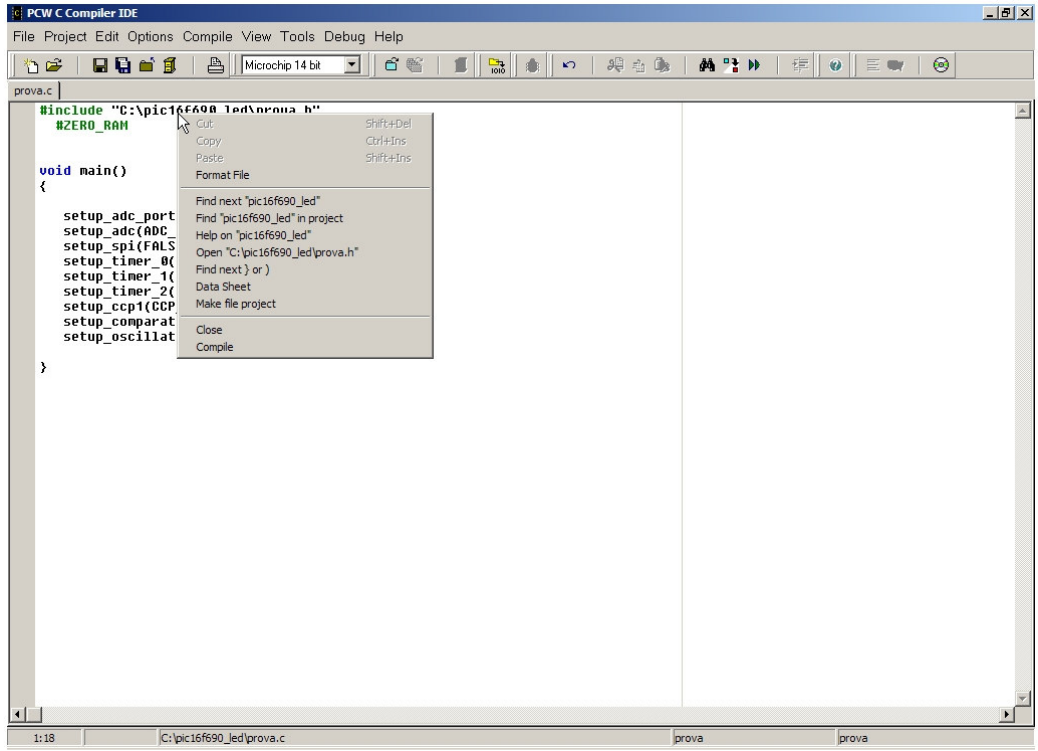
Como se puede observar, la función setup_comparator consta de un único parámetro 'mode', que especifica la configuración de los comparadores del micro. Como no vamos a emplear ninguno, usaremos la constante NC_NC_NC_NC.

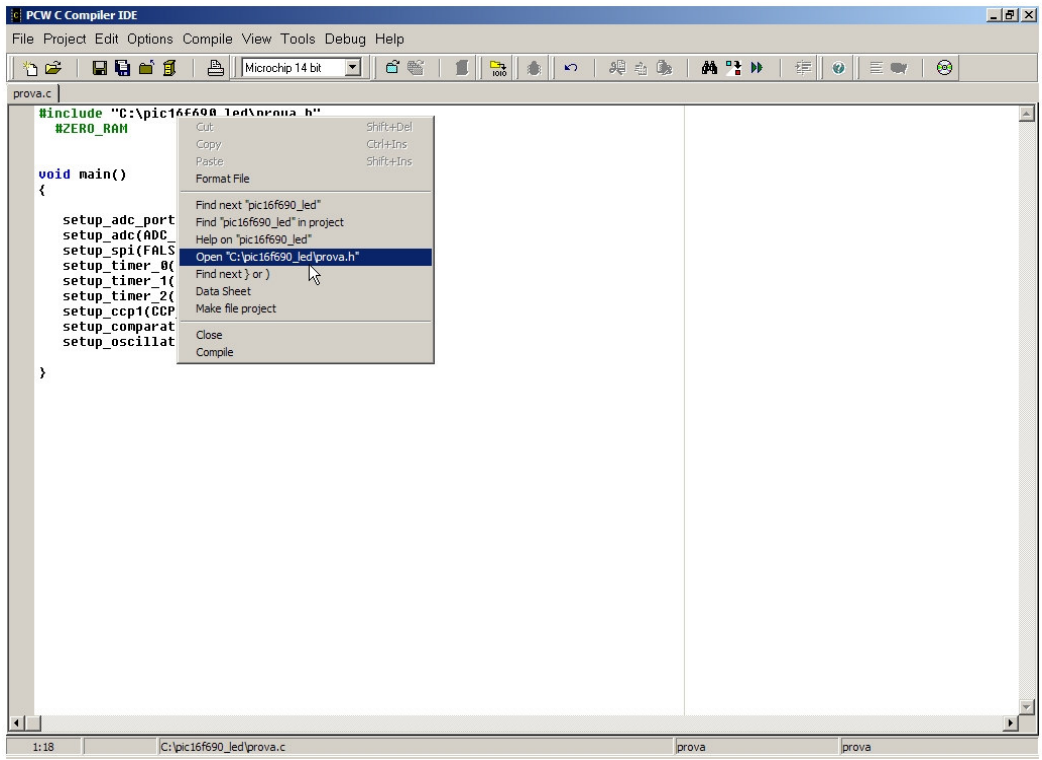
---

20. El paso anterior se puede realizar también accediendo al fichero de cabecera específico del 16F690, y consultando la sección que contiene las definiciones correspondientes a los comparadores, y consultando a su vez en el datasheet el valor de los valores numéricos que las deficiones nos indican.

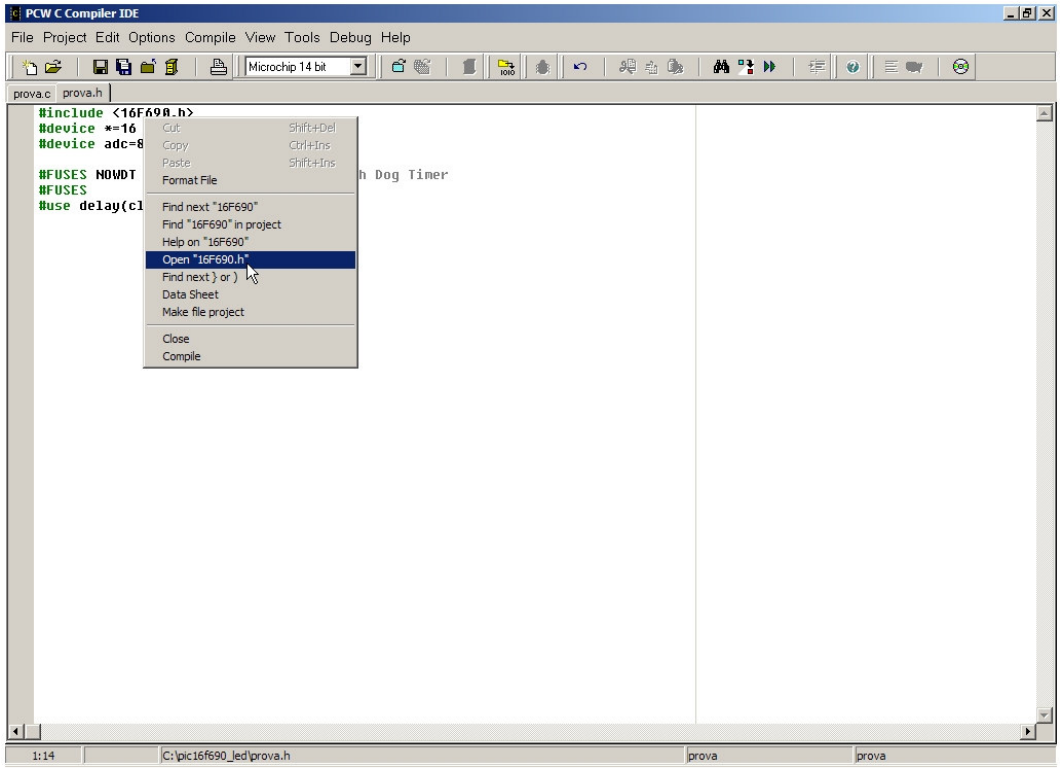
Para acceder al fichero 16F690.h podemos hacer:

- Clic con el botón derecho en el fragmento de código que incluye prova.h.
- Clic en Open '…\prova.h'

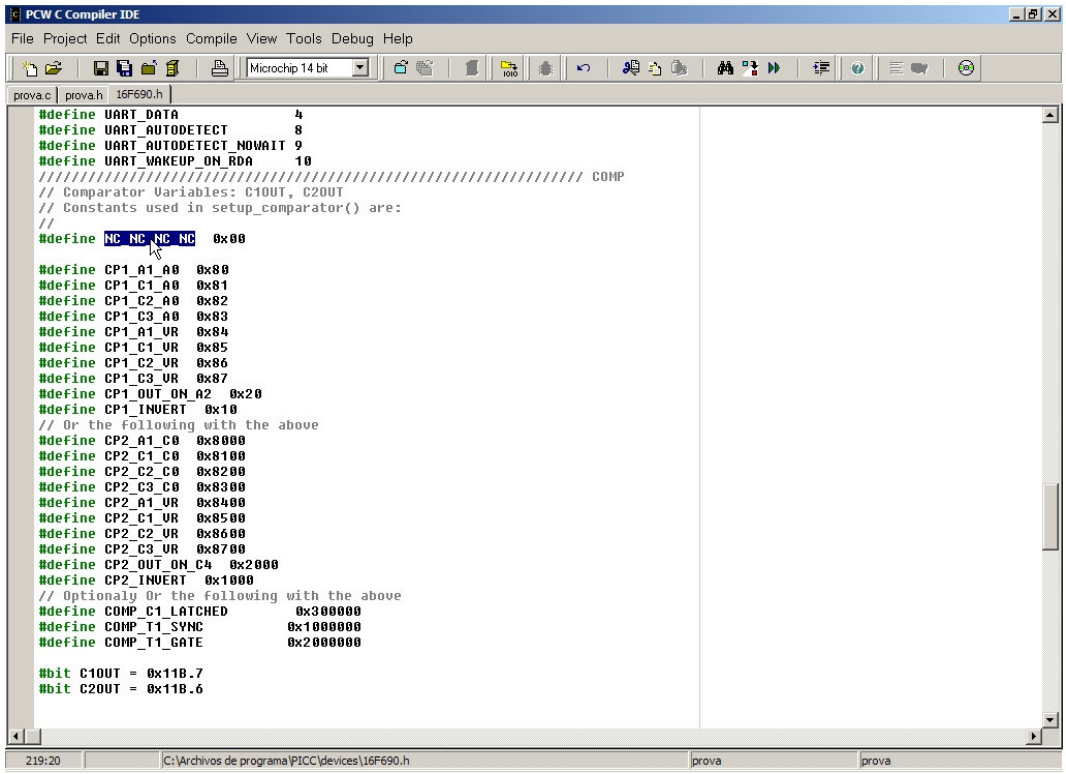




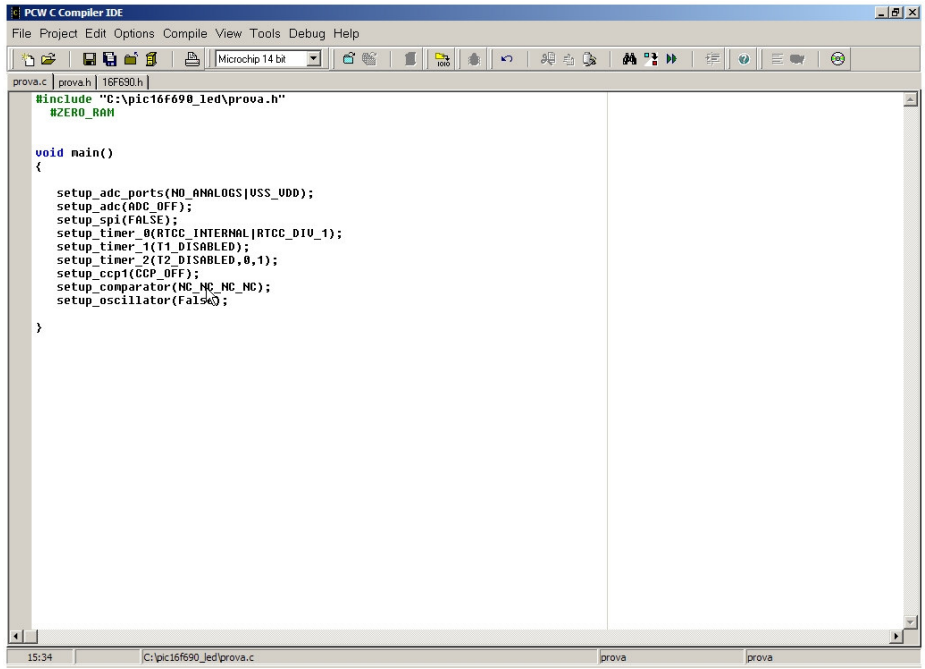
Una vez abierto 'prova.h', repetimos el proceso con '16f690.h', cuya inclusión se realiza desde prova.h.



En la figura inferior se puede observar la sección de comparadores del fichero '16f690.h':



21. En la figura inferior podemos observar el código final después de los arreglos.

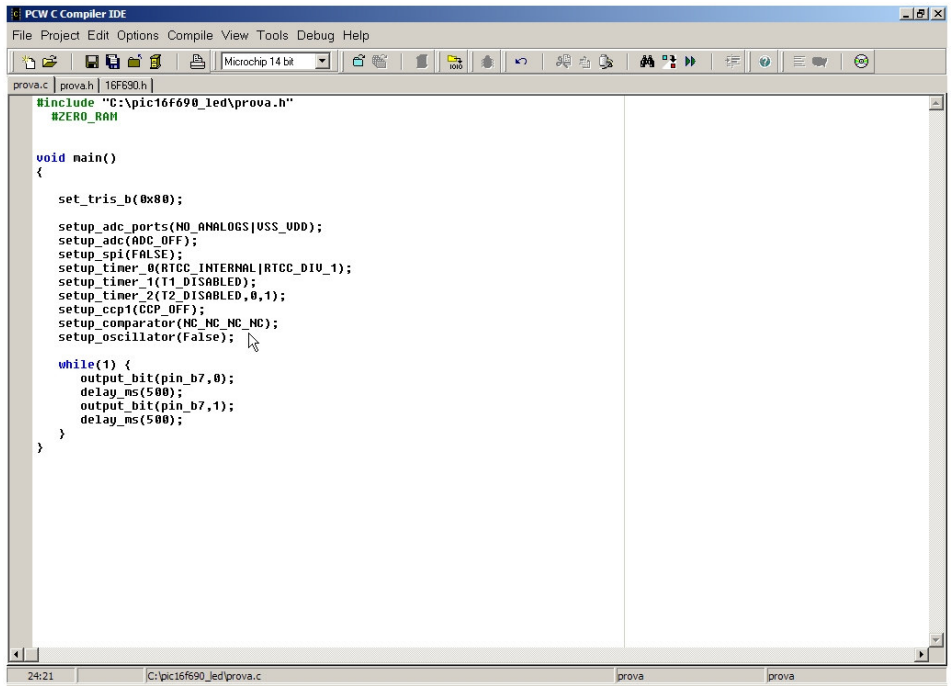


A destacar:

- La llamada a la función setup_comparator, con la constante que deshabilita los comparadores.
- La supresión de la llamada a la función que configura la tensión de referencia del A/D (setup_vref). Al no emplearla no será necesario configurar nada.

---

22. Escribimos el resto del código:

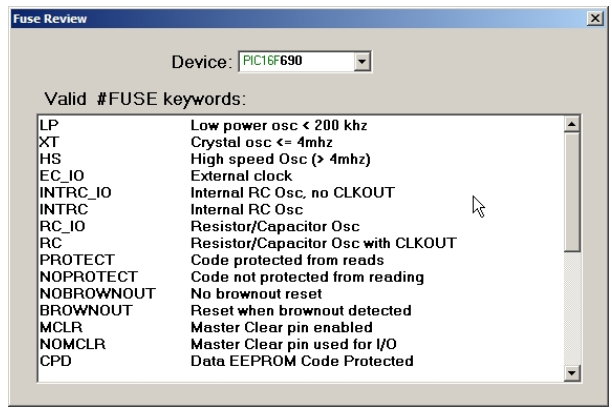


Como se puede ver, se trata de un bucle infinito (while(1)), en cuyo interior se activa y desactiva el bit 7 del puerto b (RB7), con retardo de 0,5 segundos. Así pues, si todo ha ido bien, veremos parpadear al led conectado a dicho pin.

---

23. A continuación vamos a ver qué constantes podemos asignar a la directiva #FUSES.

View > Valid Fuses.



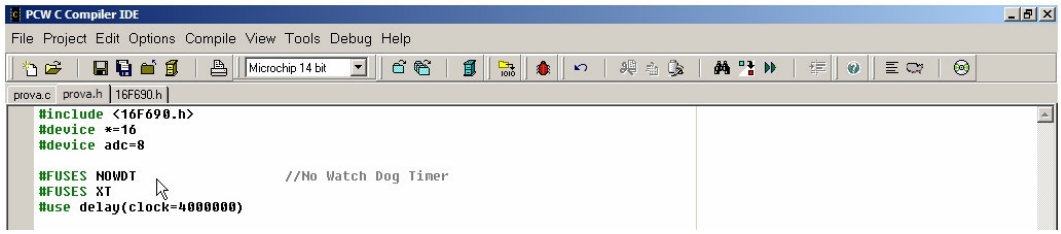
Añadiremos la directiva que indica que nuestro cristal funciona a una frecuencia inferior o igual a 4 MHz.

Como se puede observar, existen otras constantes interesantes. Por ejemplo, si quisiéramos proteger nuestro código de posibles lecturas, emplearíamos la constante PROTECT.

Deberemos tener en cuenta qué directivas empleamos en nuestro código porque deberemos configurar al programador universal en consonancia a los parámetros aquí escogidos.

---

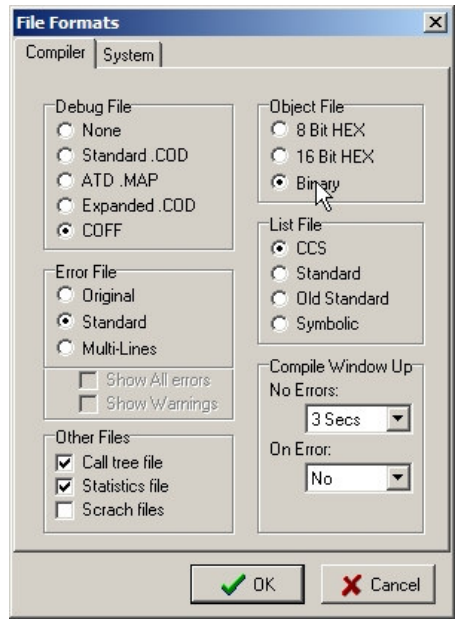
24. Modificamos prova.h añadiendo el parámetro XT a la directiva #FUSES:



---

25. Antes de compilar deberemos especificar el formato del archivo .hex que cargaremos en nuestro micro.

Accedemos al menú Options > File Formats y seleccionamos la opción 'Binary' del marco 'Object File'.

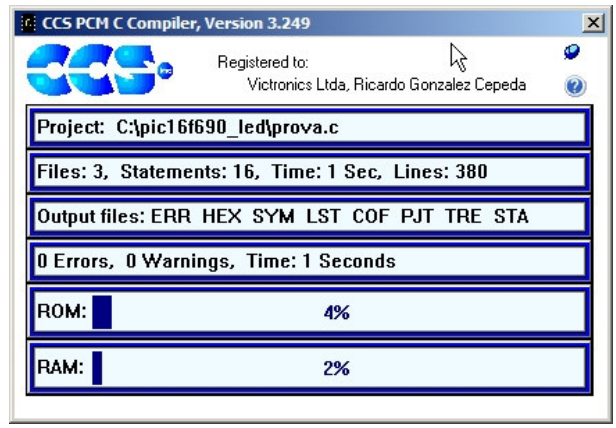


---

26. Recordemos salvar nuestro proyecto de vez en cuando. Una buena costumbre es hacerlo siempre antes de compilar cualquiera que sea el entorno de programación en el que trabajemos.

Ya podemos compilar nuestro proyecto mediante el menú Compile > Compile.

Si todo ha ido bien veremos la siguiente figura:

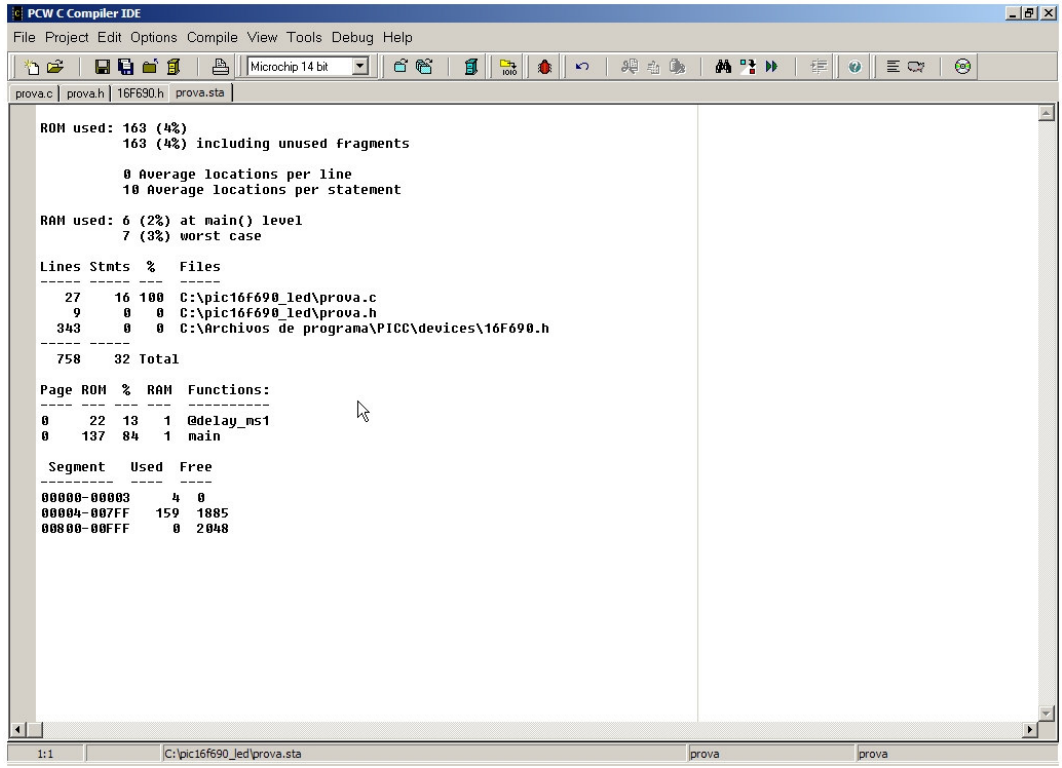


Obsérvese como una aplicación tan simple apenas consume memoria ROM (programa) y RAM (variables y registros de tiempo de ejecución).

Ya tenemos en el directorio de nuestro proyecto el fichero 'prova.hex' con el código máquina de nuestro micro, cuyo contenido volcaremos con TOPMAX.

---

27. Antes de pasar al programador, puede ser útil saber que después de compilar podemos acceder al menú View > Statistics. Se nos mostrará un archivo recopilatorio de datos estadísticos útiles sobre el rendimiento de nuestro programa.



3.5. Programación del microcontrolador

1. Nos aseguramos de que el interruptor del módulo programador está en posición OFF.

---

2. Conectamos el cable de impresora al módulo programador y al PC.

---

3. Conectamos el cable de alimentación.

---

4. Sin colocar ningún integrado en el zócalo, encendemos el módulo. El LED de POWER (rojo, parte superior derecha) se encenderá.

---

5. Iniciamos el soft de programación, el MaxLoader.

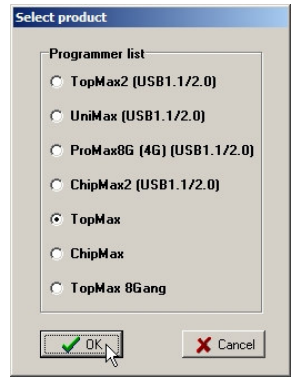
---

6. Seleccionamos el módulo a emplear:

Config > Select product

---

7. Seleccionamos la opción 'TopMax'



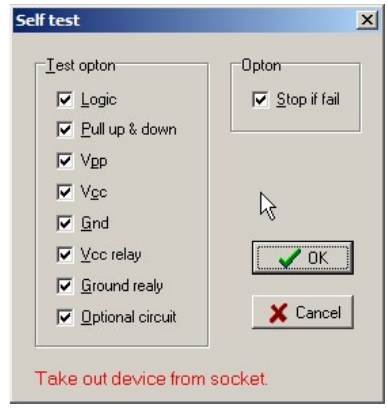
---

8. Vamos a realizar un test del módulo programador:

Config > Hardware Test

---

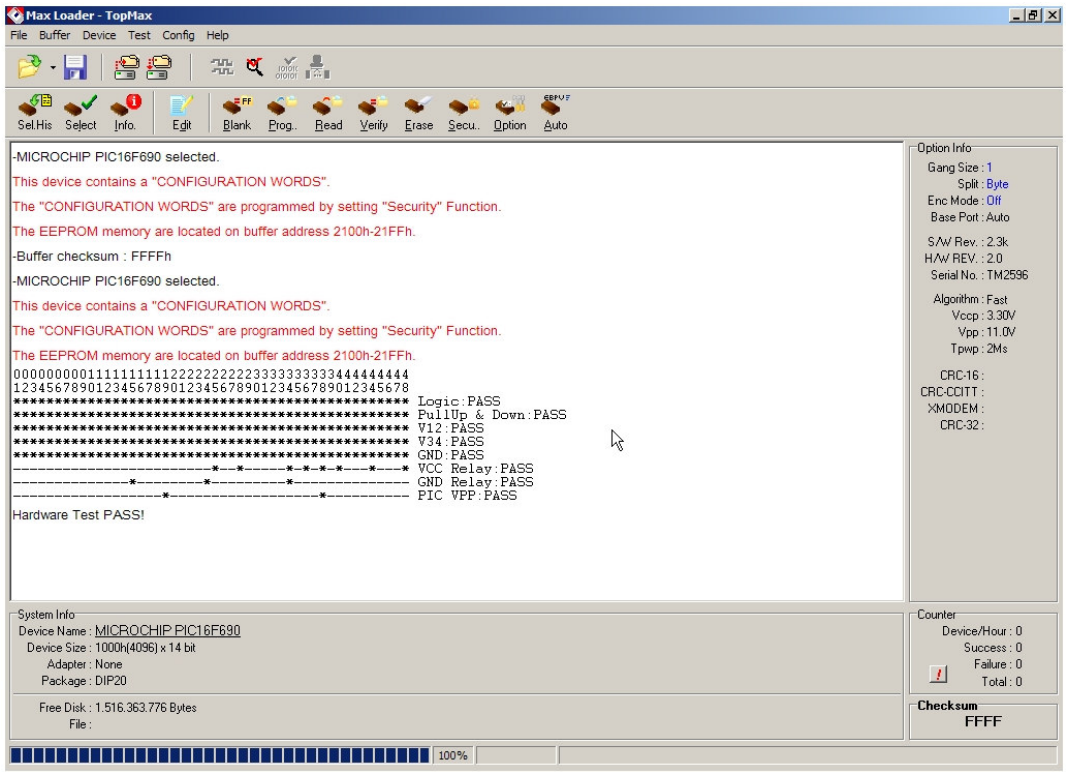
9. Seleccionamos todas las opciones y pulsamos en 'OK'.



Tal y como se indica en la figura superior, NO DEBE HABER NINGÚN CIRCUITO INSERTADO en el módulo programador.

---

10. Si todo va bien, se mostrará un mensaje como el siguiente:



---

11. Cerramos el MaxLoader.

---

12. Apagamos el módulo.

---

13. SIN ENCENDER EL PROGRAMADOR, insertamos el chip en el zócalo DIL (el pin 1 quedará en la parte superior izquierda). Para ello, tiramos de la palanca, colocamos el integrado y bajamos el bracito.

---

14. Encendemos el módulo programador.

Es importante saber que tanto al insertar como al extraer un chip mantener la unidad APAGADA (obsérvese la etiqueta amarilla indicadora, justa a la derecha del zócalo).

---

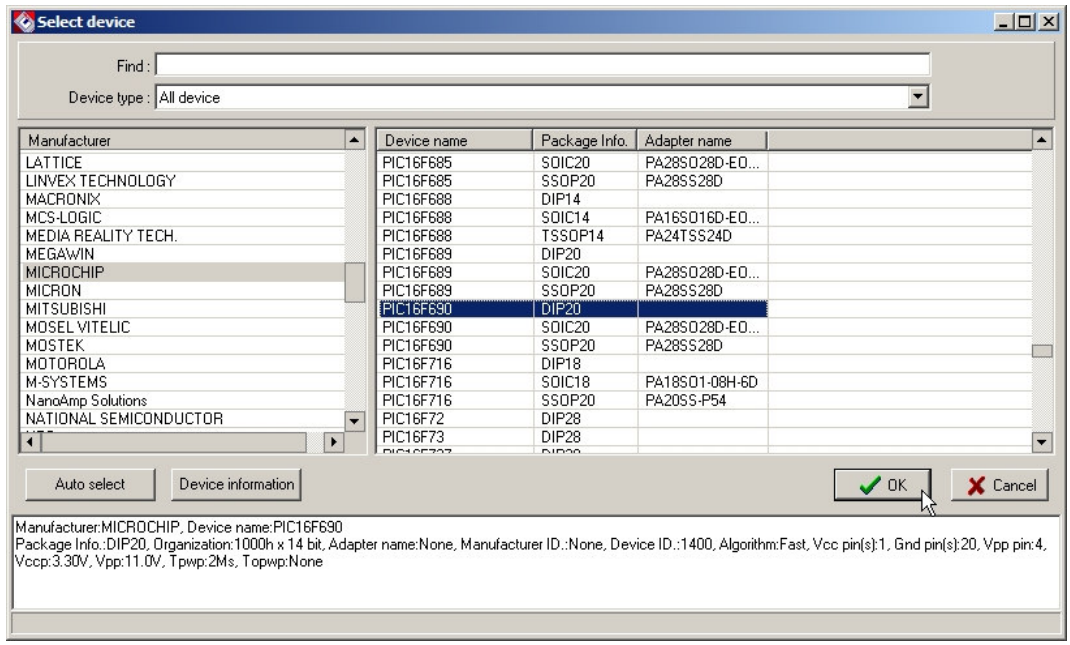
15. Ejecutamos el MaxLoader.

---

16. Click en Device > Select (para seleccionar el micro).

---

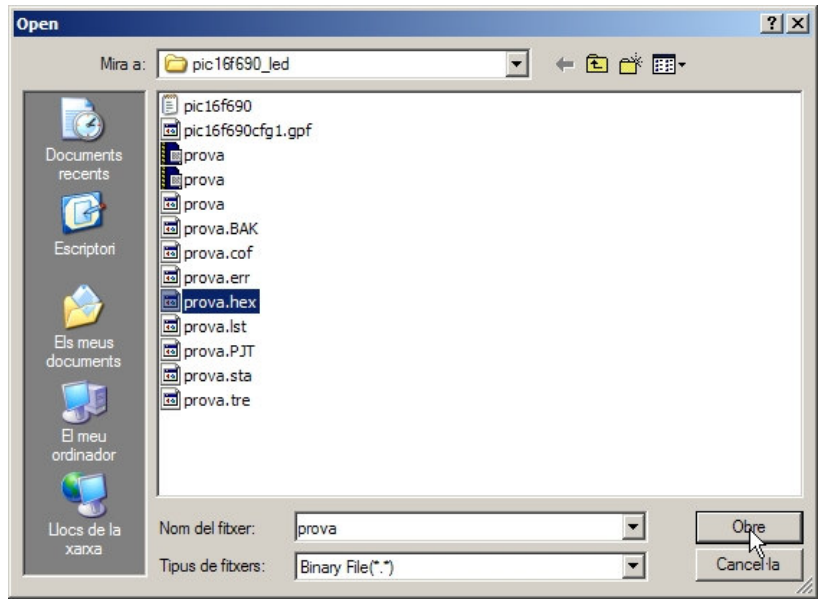
17. Buscamos la entrada PIC16F690 (DIP20, de MICROCHIP) y pulsamos en OK.



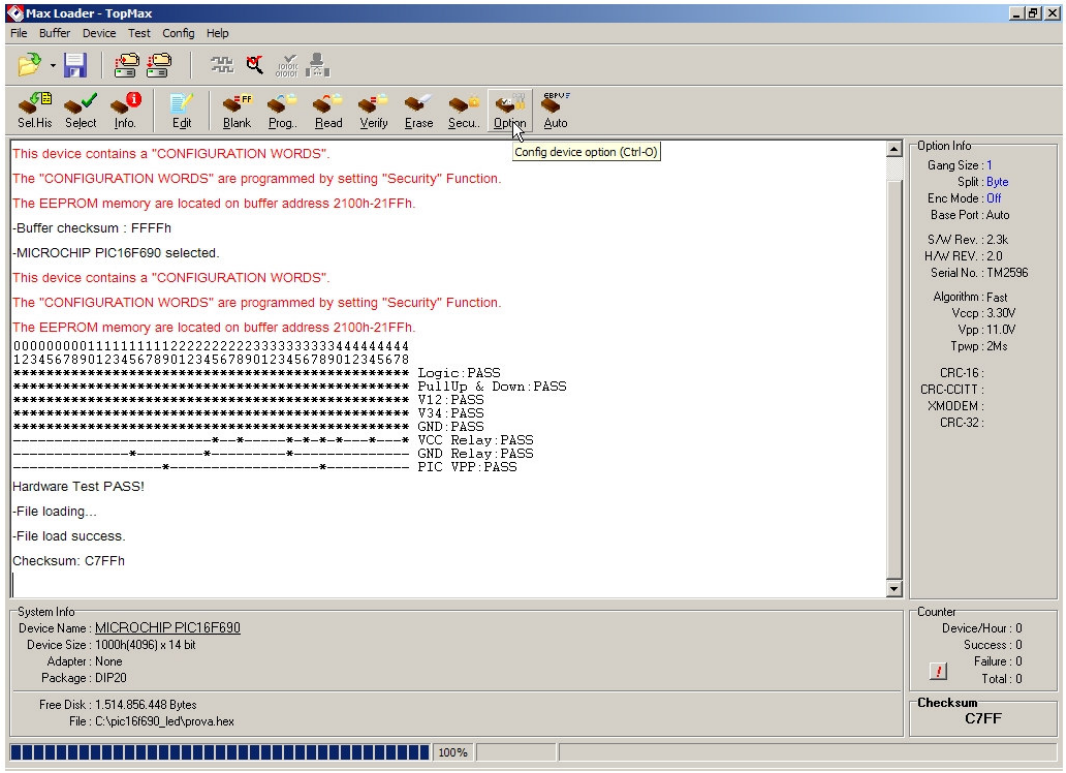
18. Cargamos el archivo .hex. Para ello:

File > Load

19. En el cuadro de diálogo open seleccionamos el archivo 'prova.hex', asegurándonos que el tipo de fichero está seleccionado como 'Binary File' (archivo en formato binario).

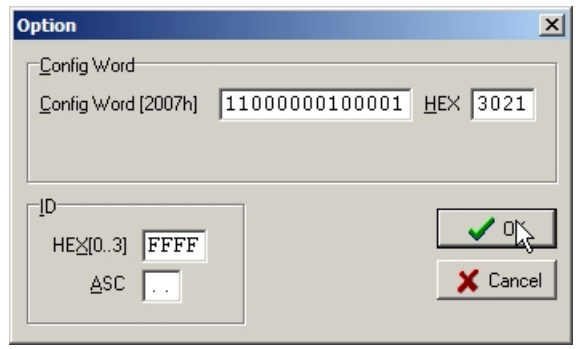


20. Pulsamos sobre el botón 'Option'.



21. Sobre el cuadro abierto editaremos la palabra de configuración (Config Word).

Cada bit corresponde a un parámetro que determina el comportamiento del micro (que también habremos tenido en cuenta al compilar el código en C). En otras aplicaciones de programación de micros se permite editar la palabra seleccionando las opciones mediante casillas de verificación (checkboxes). En nuestro caso deberemos editarla manualmente como sigue:



El significado de cada bit debe consultarse en el datasheet (página siguiente).

REGISTER 14-1: CONFIG: CONFIGURATION WORD REGISTER

Reserved	Reserved	FCMEN	IESO	BOREN1(1)	BOREN0(1)	$\overline{CPD}$(2
bit 13						bit 7

$\overline{CP}$(3)	MCLRE(4)	$\overline{PWRTE}$	WDTE	FOSC2	FOSC1	FOSC0
bit 6						bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable'	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 13-12 **Reserved:** Reserved bits. Do Not Use.

bit 11 **FCMEN:** Fail-Safe Clock Monitor Enabled bit
1 = Fail-Safe Clock Monitor is enabled
0 = Fail-Safe Clock Monitor is disabled

bit 10 **IESO:** Internal External Switchover bit
1 = Internal External Switchover mode is enabled
0 = Internal External Switchover mode is disabled

bit 9-8 **BOREN<1:0>:** Brown-out Reset Selection bits(1)
11 = BOR enabled
10 = BOR enabled during operation and disabled in Sleep
01 = BOR controlled by SBOREN bit of the PCON register
00 = BOR disabled

bit 7 **$\overline{CPD}$:** Data Code Protection bit(2)
1 = Data memory code protection is disabled
0 = Data memory code protection is enabled

bit 6 **$\overline{CP}$:** Code Protection bit(2)
1 = Program memory code protection is disabled
0 = Program memory code protection is enabled

bit 5 **MCLRE:** $\overline{MCLR}$ Pin Function Select bit(3)
1 = $\overline{MCLR}$ pin function is $\overline{MCLR}$
0 = $\overline{MCLR}$ pin function is digital input, $\overline{MCLR}$ internally tied to VDD

bit 4 **$\overline{PWRTE}$:** Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 3 **WDTE:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits
111 = RC oscillator: CLKOUT function on RA4/OSC2/CLKOUT pin, RC on RA5/OSC1/CLKIN
110 = RCIO oscillator: I/O function on RA4/OSC2/CLKOUT pin, RC on RA5/OSC1/CLKIN
101 = INTOSC oscillator: CLKOUT function on RA4/OSC2/CLKOUT pin, I/O function on RA5/OSC1/CLKIN
100 = INTOSCIO oscillator: I/O function on RA4/OSC2/CLKOUT pin, I/O function on RA5/OSC1/CLKIN
011 = EC: I/O function on RA4/OSC2/CLKOUT pin, CLKIN on RA5/OSC1/CLKIN
010 = HS oscillator: High-speed crystal/resonator on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN
001 = XT oscillator: Crystal/resonator on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN
000 = LP oscillator: Low-power crystal on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN

Note 1: Enabling Brown-out Reset does not automatically enable Power-up Timer.
2: The entire data EEPROM will be erased when the code protection is turned off.
3: The entire program memory will be erased when the code protection is turned off.
4: When $\overline{MCLR}$ is asserted in INTOSC or RC mode, the internal clock oscillator is disabled.

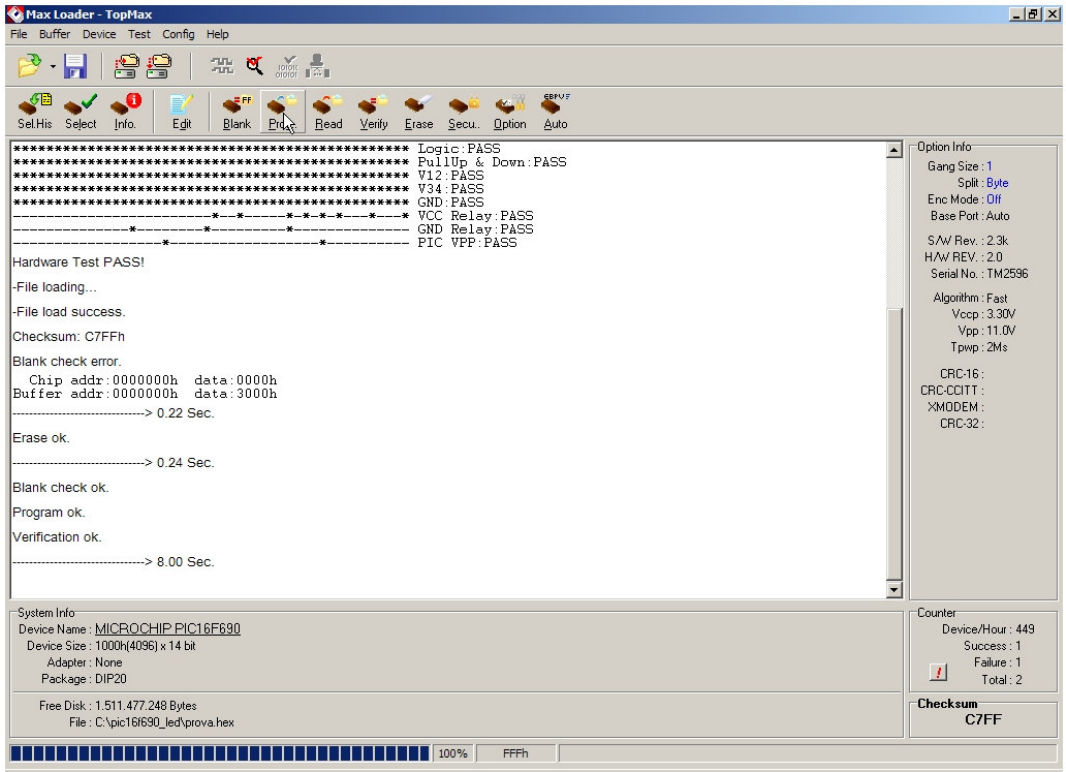
---

22. Click en el botón 'Erase'. Acabamos de borrar el contenido previo del micro.

---

23. Click en el botón 'Program'.

Empieza el proceso de volcado. Si todo ha ido bien, se mostrará una pantalla como la de la siguiente página.



---

24. Ahora debemos fijar la palabra de configuración en el micro. Para ello, hacemos click en el botón 'Secure'.

---

25. Se nos pedirá confirmación. Aceptamos.

---

26. Esperamos a que finalice el proceso de fijado de la 'Config Word'.

---

27. Para extraer el chip, apagamos el módulo programador y luego sacamos el integrado.

---

28. Todas las opciones de configuración seleccionadas (incluído el archivos .hex) las podremos salvar mediante:

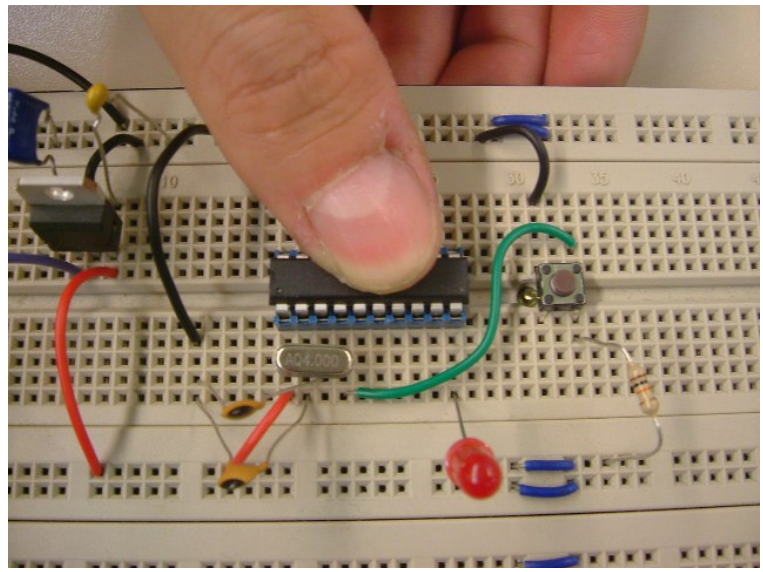
File > Save Project

---

29. Indicaremos el nombre del autor y una breve descripción.

3.6. Comprobación del correcto funcionamiento

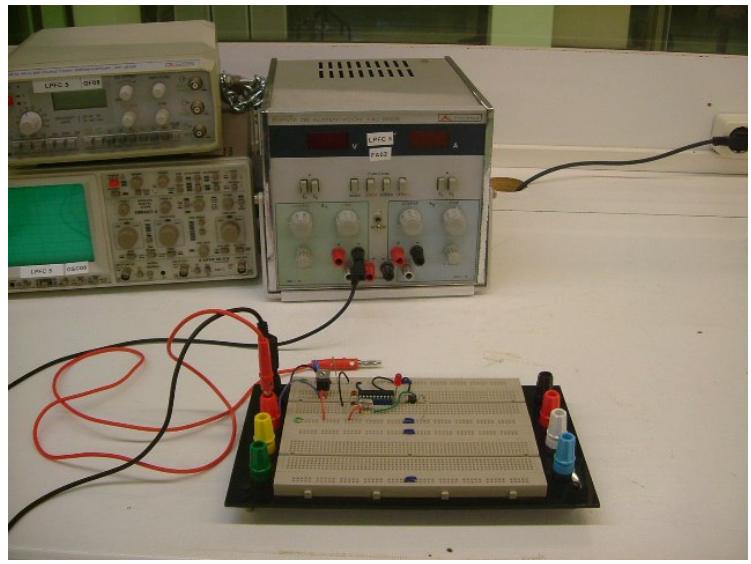
1. Insertamos el micro en la posición que le corresponde de la protoboard, asegurándonos de que queda bien fijado y las patitas hacen contacto con el sustrato metálico de la placa de pruebas.



---

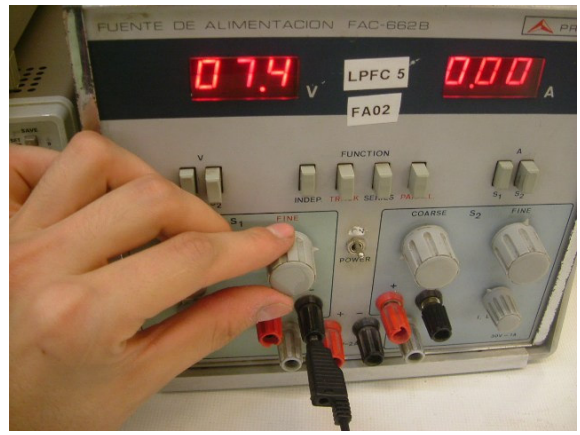
2. Conectamos los cables banana-banana a la protoboard y a la fuente de alimentación, a excepción del borne positivo que corresponde a la fuente. El porqué de proceder así radica en el pico de tensión y corriente que proporciona la fuente al encenderla. Podríamos dañar nuestro micro.

Más conveniente es conectar el positivo de la placa a la fuente una vez esté encendida.



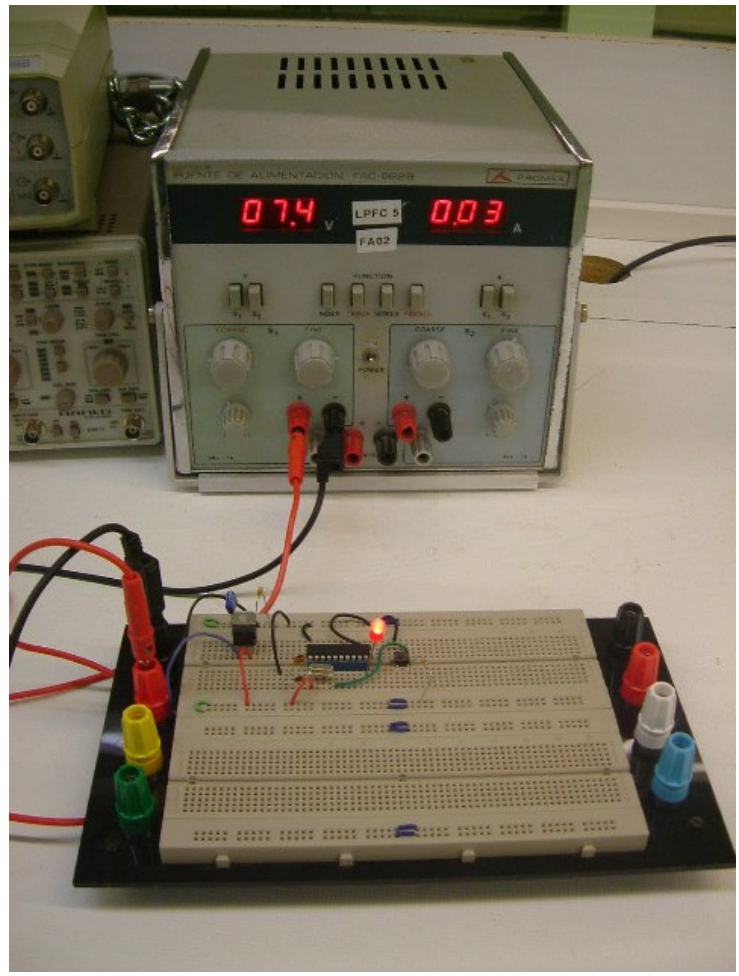
---

3. Encendemos la fuente de alimentación y seleccionamos una tensión superior a los 7 V (requisito del 7805) e inferior a 8 V.



---

4. ¡Llegó el momento de la verdad! Conectamos el la placa a la fuente. Si todo ha ido bien, nuestro LED se encenderá y apagará alternativamente cada medio segundo.



---

5. Finalmente comprobamos que el botón de reset realiza su función correctamente: al mantenerlo pulsado el circuito debería permanecer inactivo (el LED deja de encenderse). Al soltarlo vuelve a encenderse de forma intermitente.