

# **Ejemplo de utilización del programador universal ELNEC PIKPROG con el microcontrolador Microchip PIC 16F57 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

<b>1. Objetivo de este documento .....</b>	<b>2</b>
<b>2. Aplicación de ejemplo a montar y programar .....</b>	<b>3</b>
<b>3. Procedimiento .....</b>	<b>4</b>
<b><i>3.1. Montaje del circuito en protoboard .....</i></b>	<b>5</b>
<b><i>3.2. Instalación de CCS PCWH 3.249 .....</i></b>	<b>6</b>
<b><i>3.3. Instalación del PIKPROG .....</i></b>	<b>8</b>
<b><i>3.4. Creación del programa a ejecutar en el PIC16F57 .....</i></b>	<b>14</b>
<b><i>3.5. Programación del microcontrolador .....</i></b>	<b>24</b>
<b><i>3.6. Comprobación del correcto funcionamiento .....</i></b>	<b>31</b>

## **1. Objetivo de este documento**

El programador PIKPROG de la firma ELNEC 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.

Para realizar el volcado sólo precisamos seleccionar el integrado a emplear, generar el archivo en formato hexadecimal que queramos volcar y programar el chip.

En el momento de elaborar este guión el programador PIKPROG ha dejado de fabricarse por parte de ELNEC. Esto quiere decir que, probablemente, ya no se van a realizar más actualizaciones de software y drivers para nuevos dispositivos que salgan al mercado. En la página web del fabricante (<http://www.elnec.com/>) se proporciona la lista de chips soportados.

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 PIC16F57, 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.

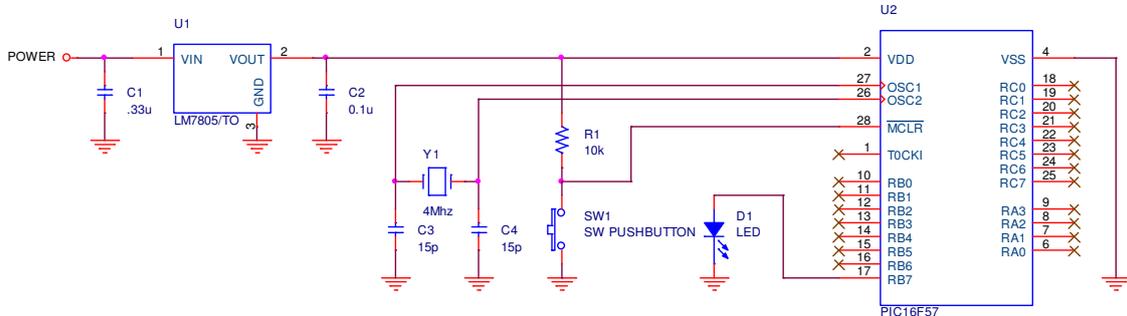
El 16F57 es un micro de prestaciones más reducidas que el 16F690 (este último utilizado como ejemplo en otros manuales de programadores del laboratorio). Se ha elegido el primero puesto que el segundo no está soportado por el programador, tal y como se verá más adelante.

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.

## 2. Aplicación a montar

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

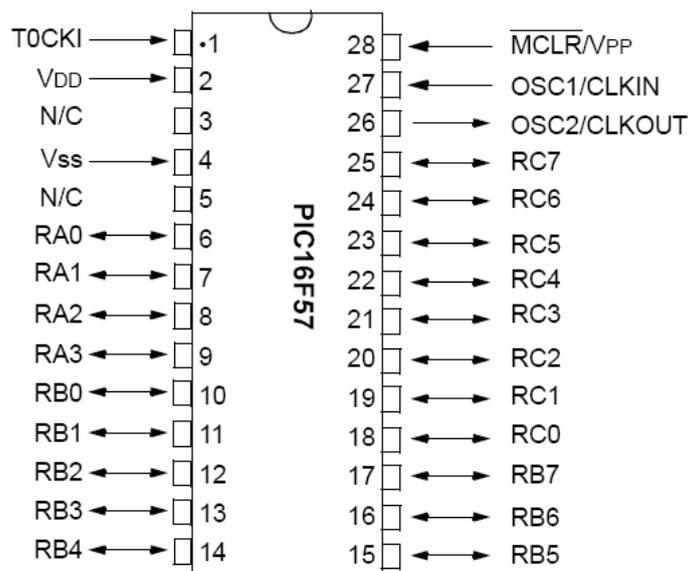


Aspectos a observar:

- 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 incluido 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.

### PDIP, SOIC



### **3. Procedimiento**

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

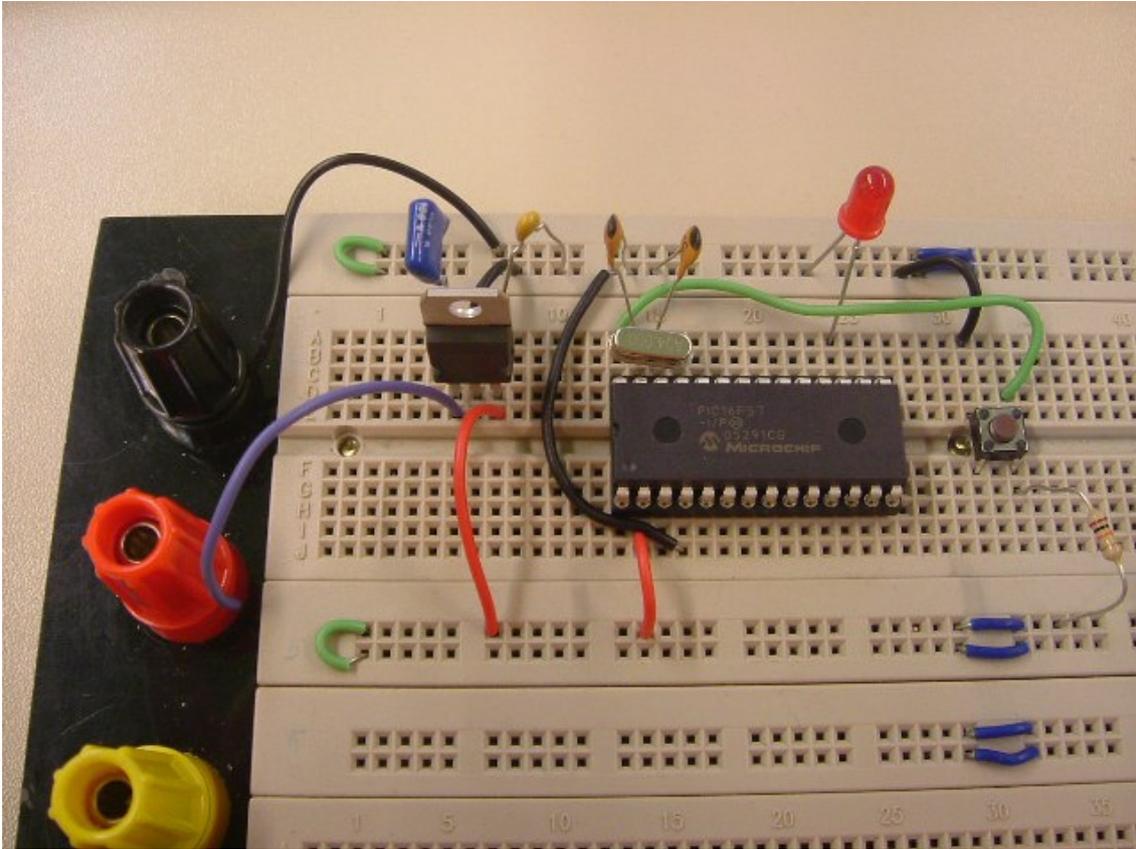
No hace falta decir que, en el caso de tener ya instalado el compilador CCS PCWH o el software del programador universal PIKPROG podremos prescindir de los apartados 3.2 y 3.3, respectivamente.

En el momento de elaborar este manual, ha sido necesario descargar los últimos drivers disponibles. Puede tomarse el punto 3.3 como ejemplo para realizar esta tarea en caso de necesidad, si bien en el pack se incluye un CD con el soft necesario.

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:

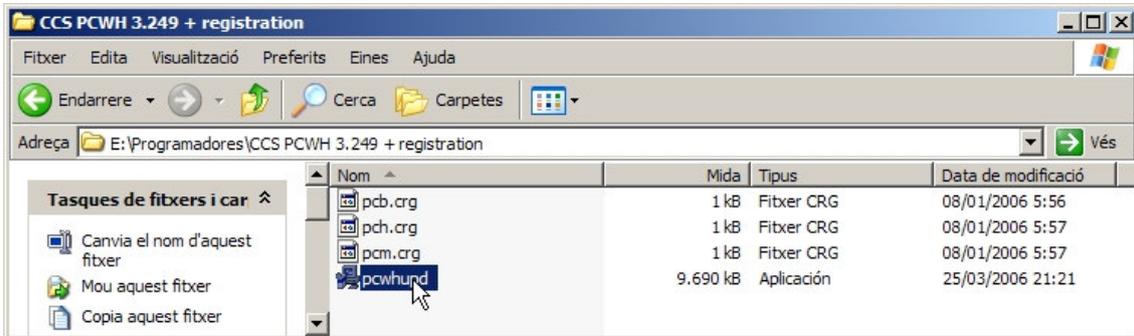


Obsérvese el empleo de un zócalo de 28 pines tipo DIL 600 para nuestro micro. Resulta muy recomendable su uso puesto que al programar varias veces el micro y comprobar su funcionamiento estaremos insertándolo y extrayéndolo de la protoboard continuamente, con lo que podríamos dañar físicamente alguna 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.

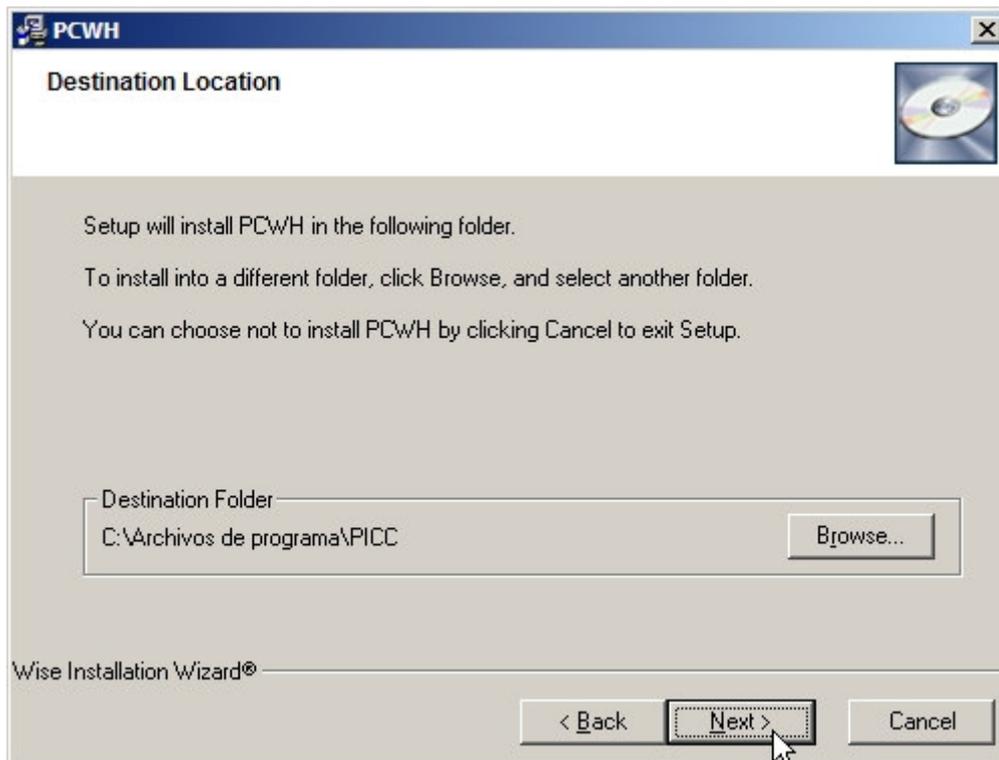
### 3.2. Instalación de CCS PCWH 3.249

1. Ejecutar 'pcwhund.exe':



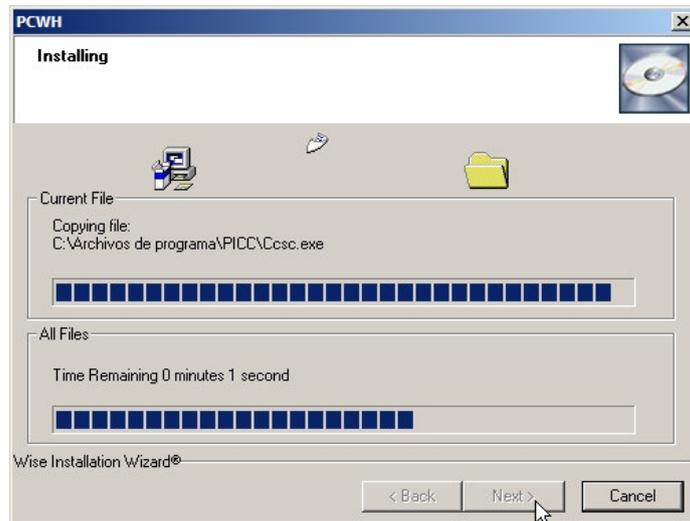
2. Clic en 'Next' > Clic en 'Next' > Clic en 'Next'

3. Seleccionar la carpeta destino para instalar el programa:

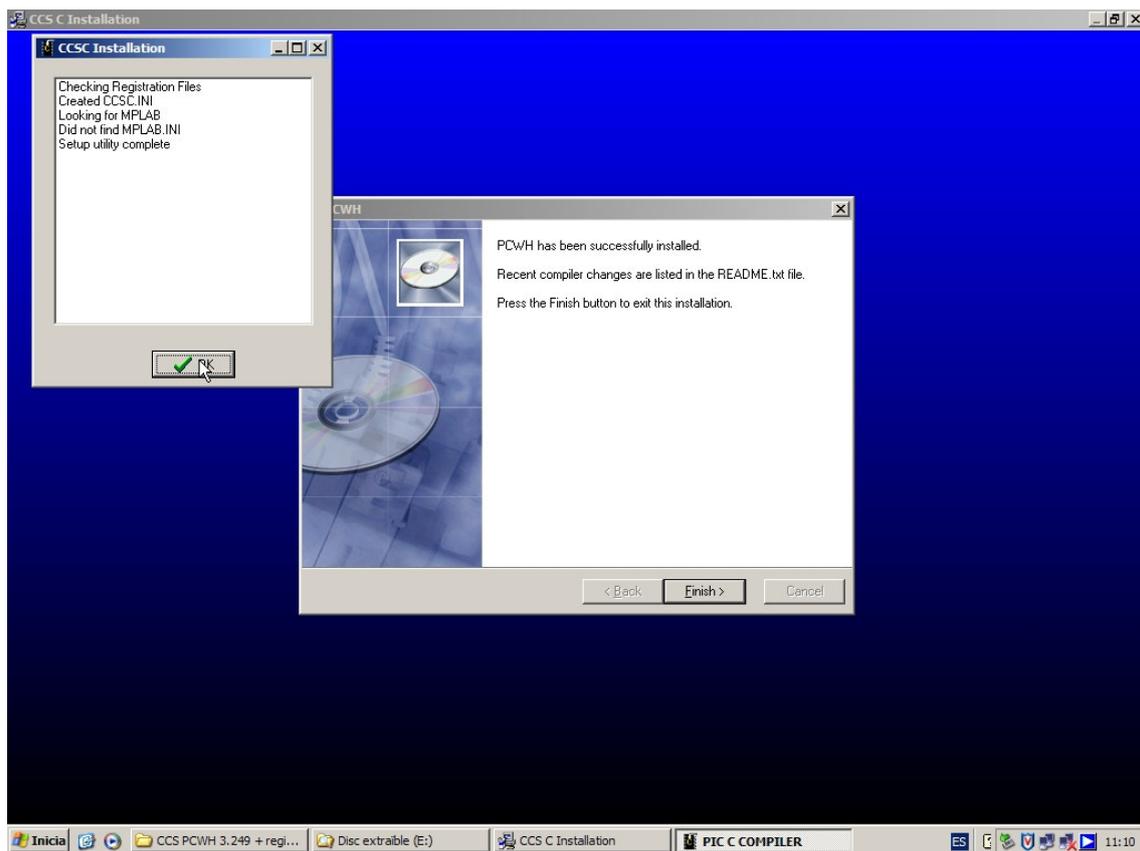


4. Clic en 'Next'

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 PIKPROG**

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



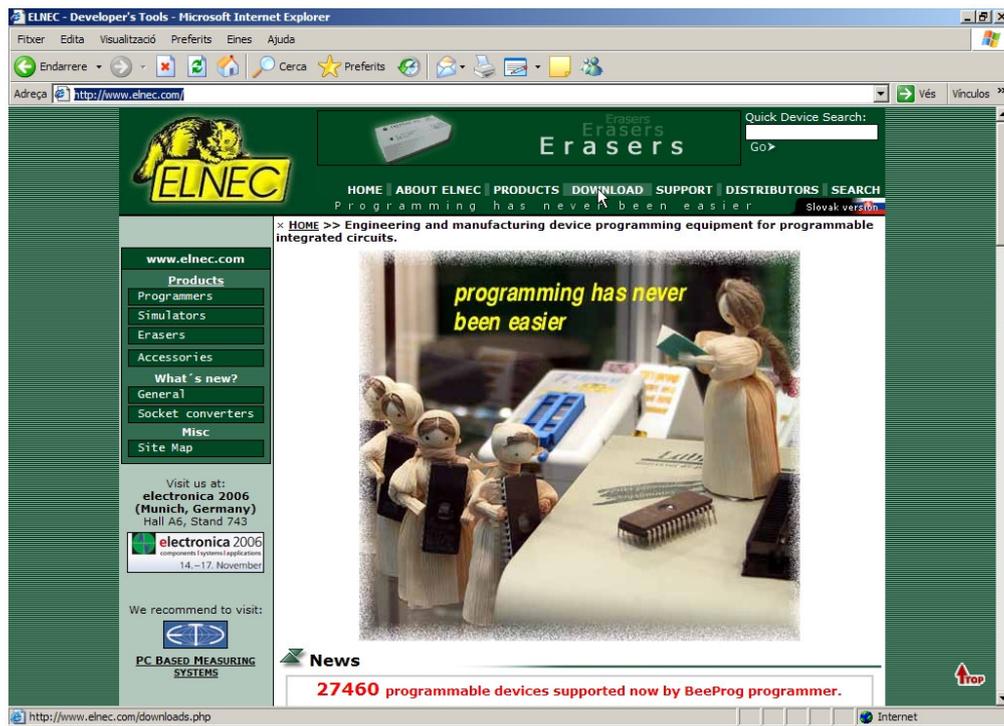
En caso de faltar algún elemento será preciso avisar a uno de los responsables del laboratorio.

- 
2. Desde nuestro navegador, introducimos la dirección de la web del fabricante:

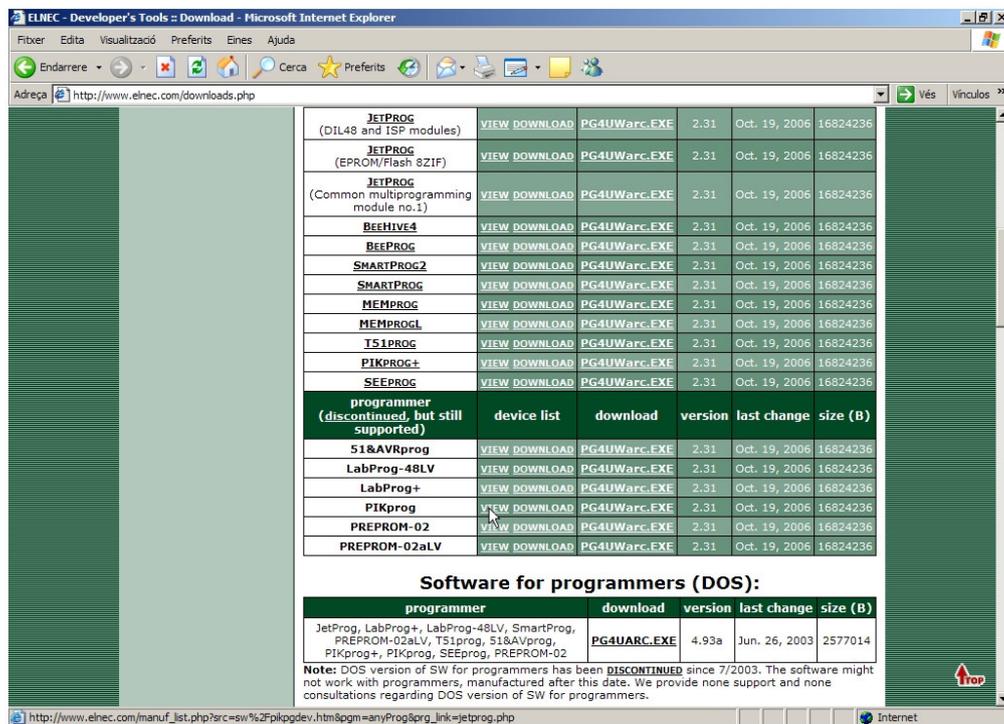
<http://www.galep.com/>

---

3. Click en 'DOWNLOAD'.



4. Nos deslizamos hacia la parte inferior de la página empleando la barra de desplazamiento vertical. En la pantalla siguiente,

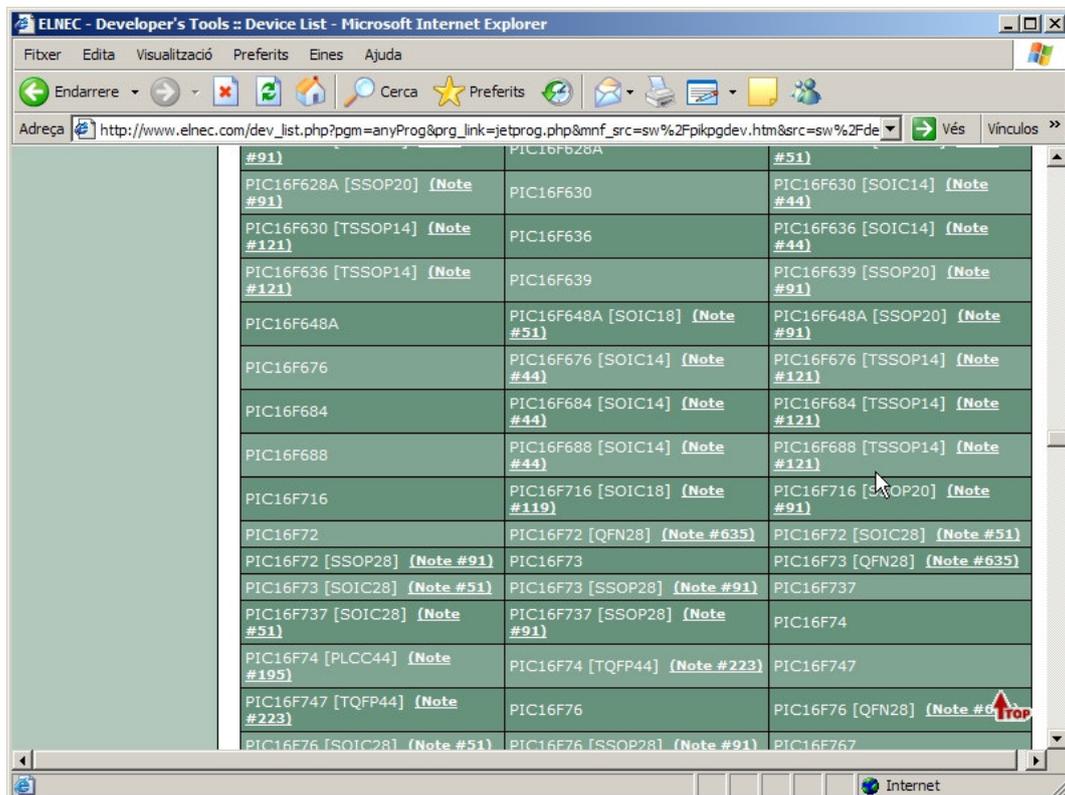


observamos como el programador PIKprog se encuentra en la sección 'discontinued, but still supported'. Clicamos en VIEW (columna 'device list').

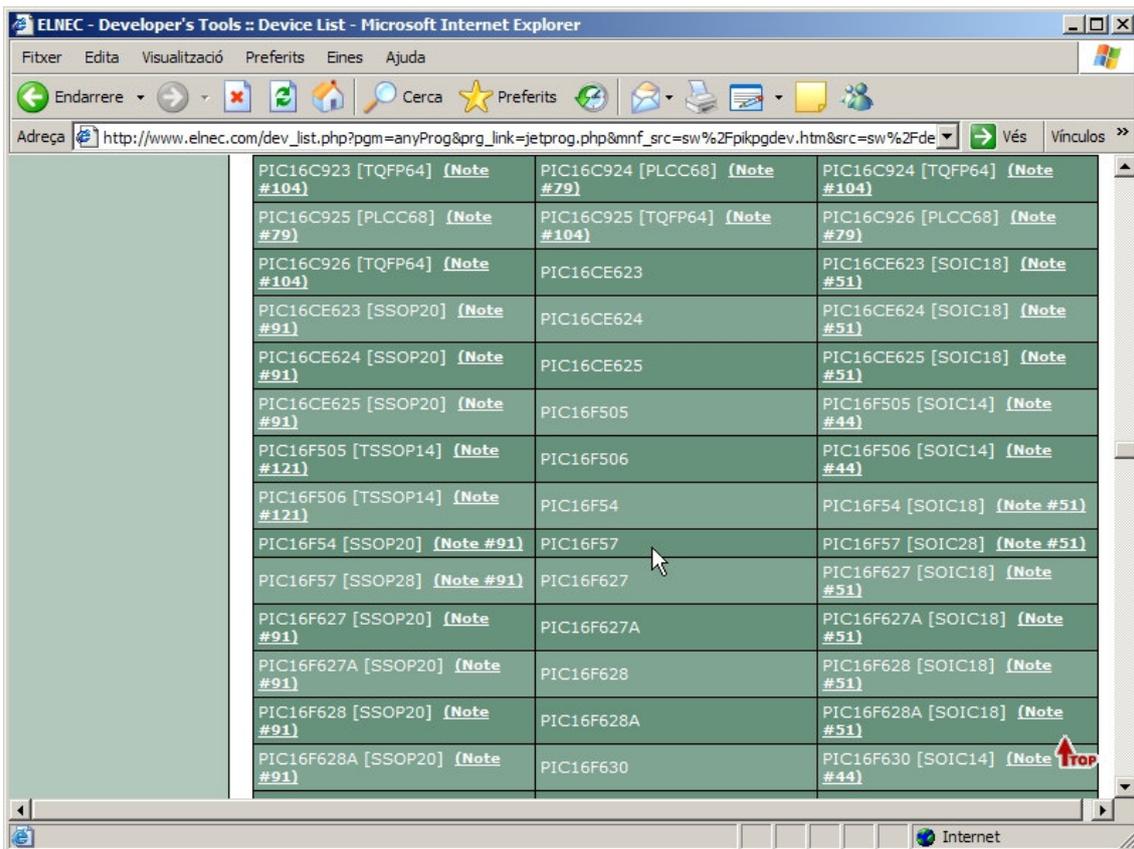
5. Click en 'Microchip'.



6. Si buscamos el PIC16F690 (el que inicialmente teníamos pensado emplear) observaremos que no se encuentra en la lista.



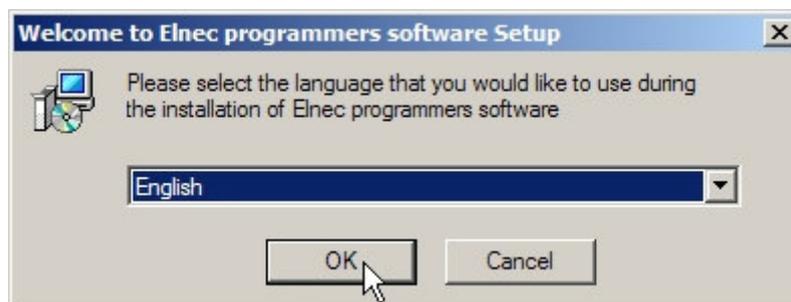
7. Vamos a buscar ahora el 16F57:



Como se puede ver, sí que está en la lista.

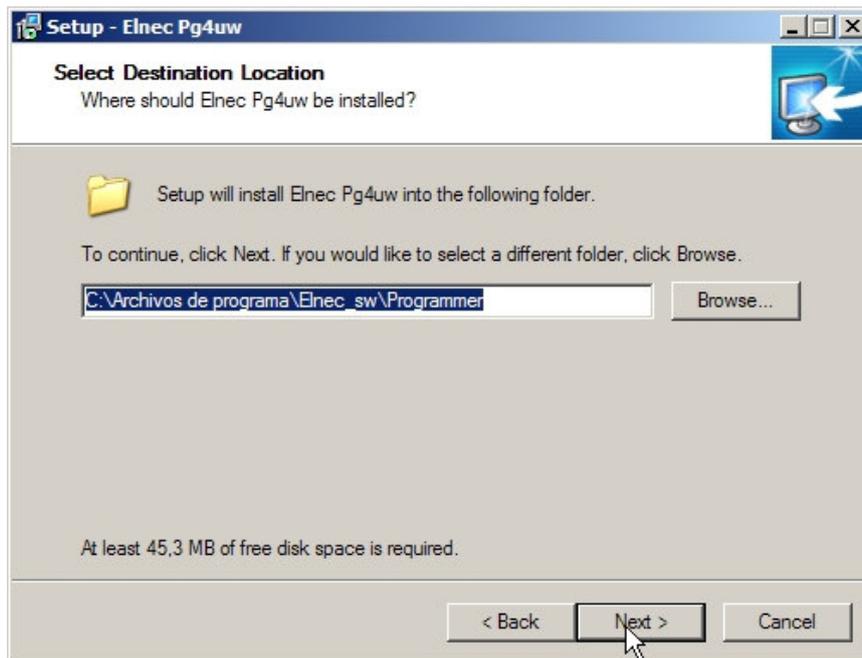
8. Volvemos a la pantalla anterior y hacemos click en el link 'PG4UWarc.EXE' (fila PIKPROG), con lo que iniciaremos la descarga del fichero de instalación.

9. Al ejecutar el fichero descargado, lo primero que se nos pregunta es el idioma de la interface.



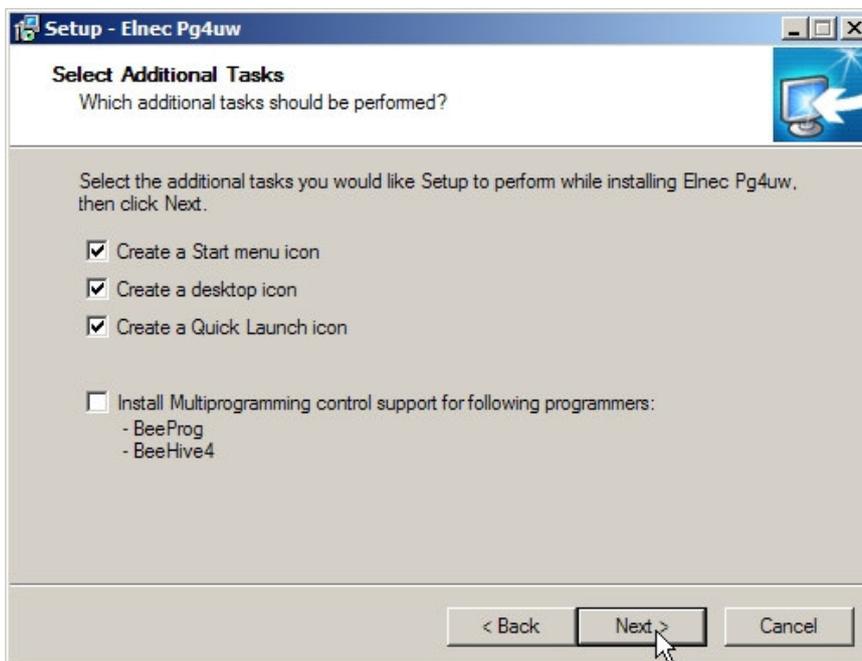
10. Click en 'Next'.

11. Seleccionamos la ruta de destino para la aplicación a instalar.



12. Introducimos el nombre de la carpeta del menú Inicio. (Click en 'Next').

13. Click en 'Next'.

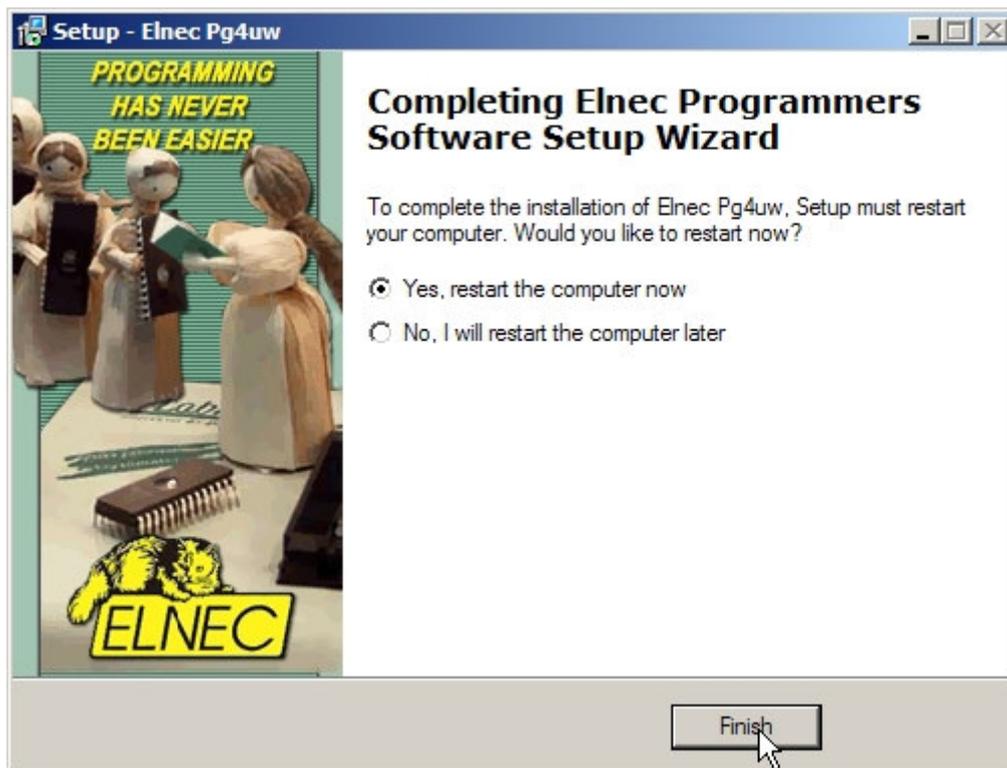


14. Click en 'Install'. Esperamos a que finalice la copia de archivos.

15. Si aparece el siguiente cuadro de diálogo, pulsamos 'Continuar igualmente'.



16. Al finalizar la instalación, el ordenador reiniciará.



### 3.4. Creación del programa a ejecutar en el PIC16F57.

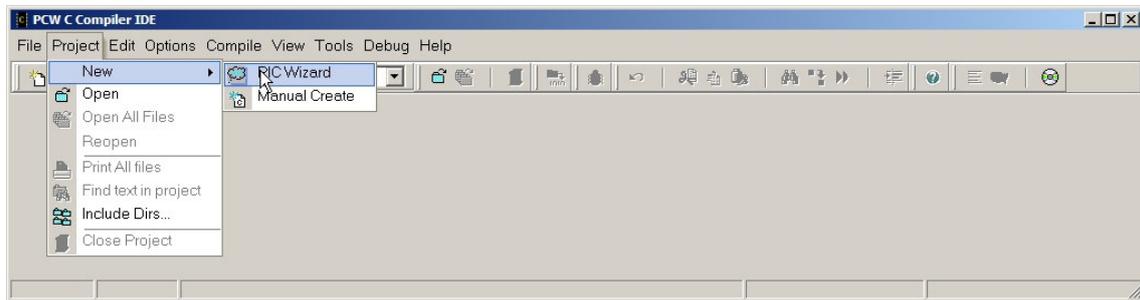
1. Iniciamos el CCS PCW PIC C Compiler:

Inicio > Programas > PIC-C > PIC C Compiler



2. Ejecutamos el asistente para la creación de proyectos:

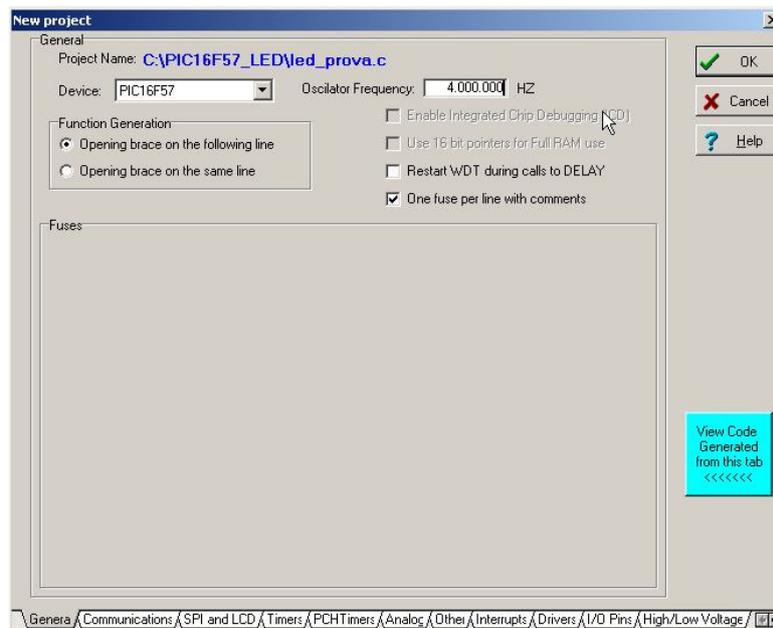
Project > New > PIC Wizard



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 (PIC16F57).  
Introducimos la frecuencia de trabajo de nuestro cristal oscilador (4000000).



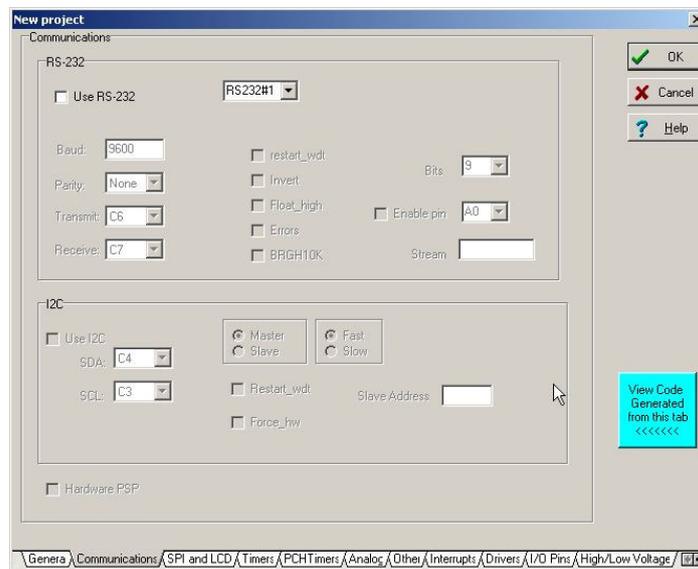
5. Si clicamos en 'View Code Generated from this tab' observaremos la ventana inferior.



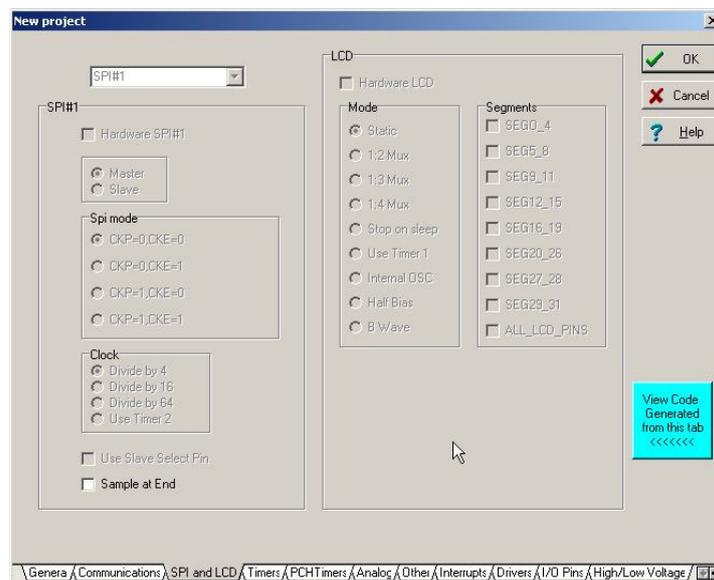
```
Code
Inserted into .h file:
#FUSES NOWDT           //No Watch Dog Timer
#FUSES
#fuse delay(clock=4000000)
```

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.

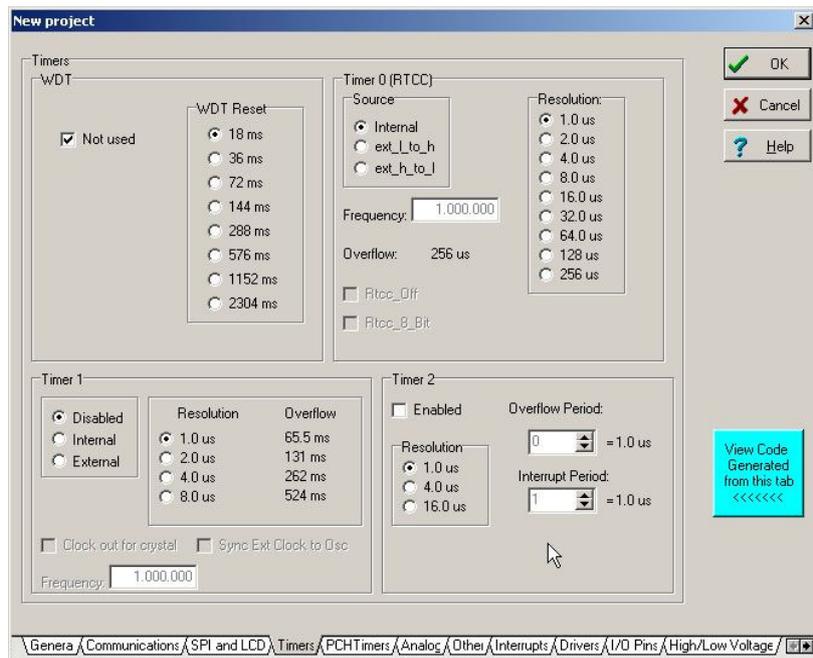
### 6. Pestaña 'Communications':



### 7. Pestaña 'SPI and LCD':



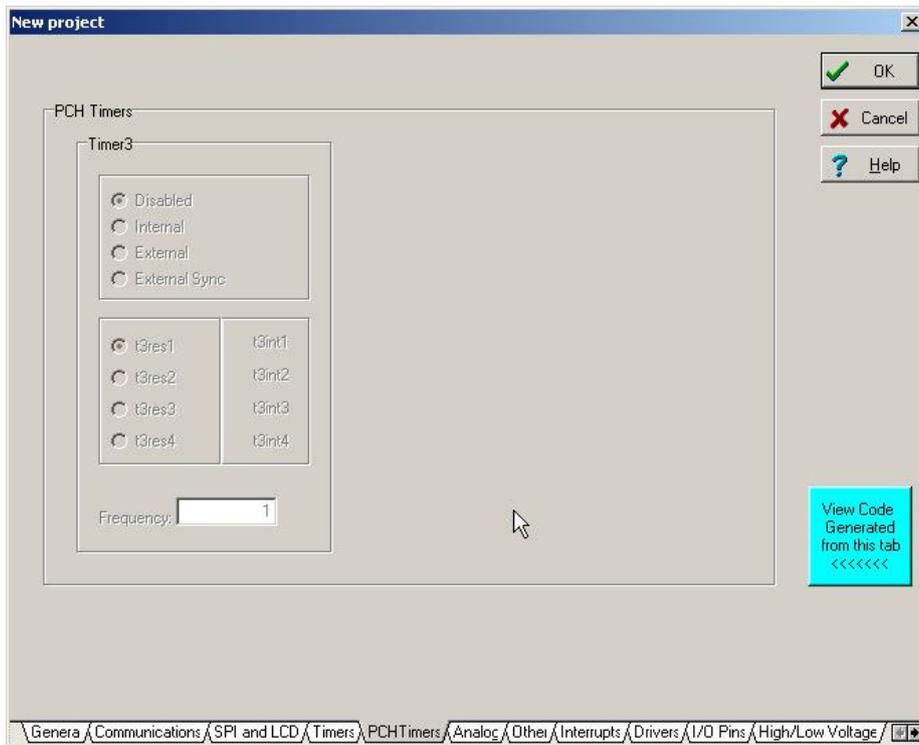
## 8. Pestaña 'Timers':



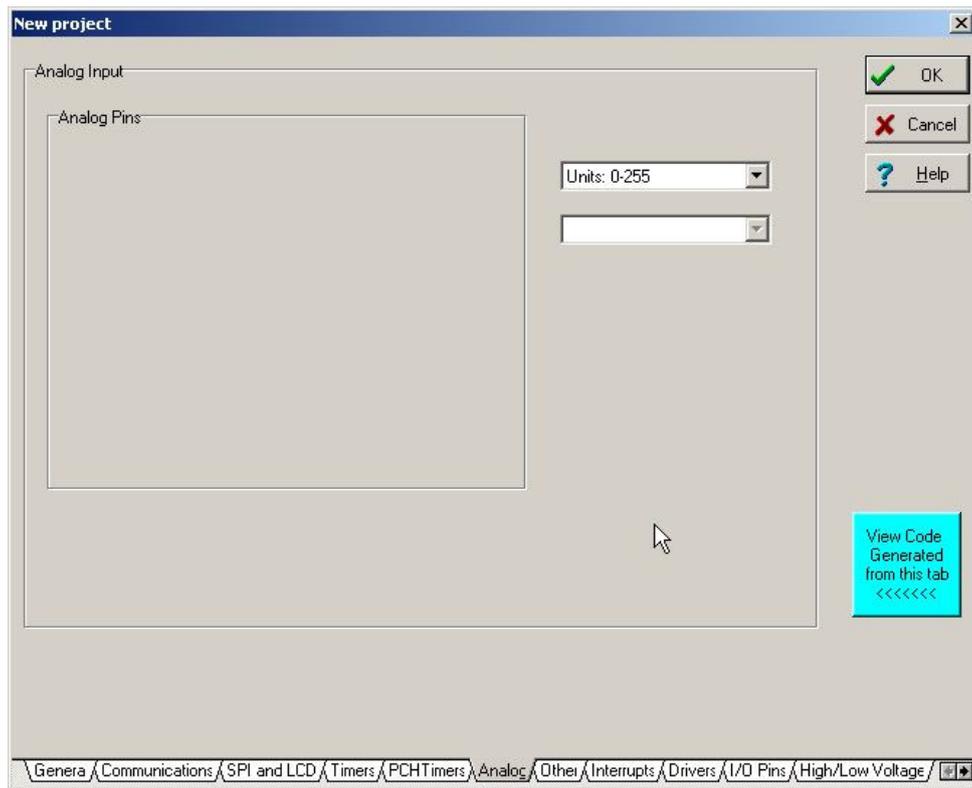
```

Code
Inserted into .c file in main():
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
    
```

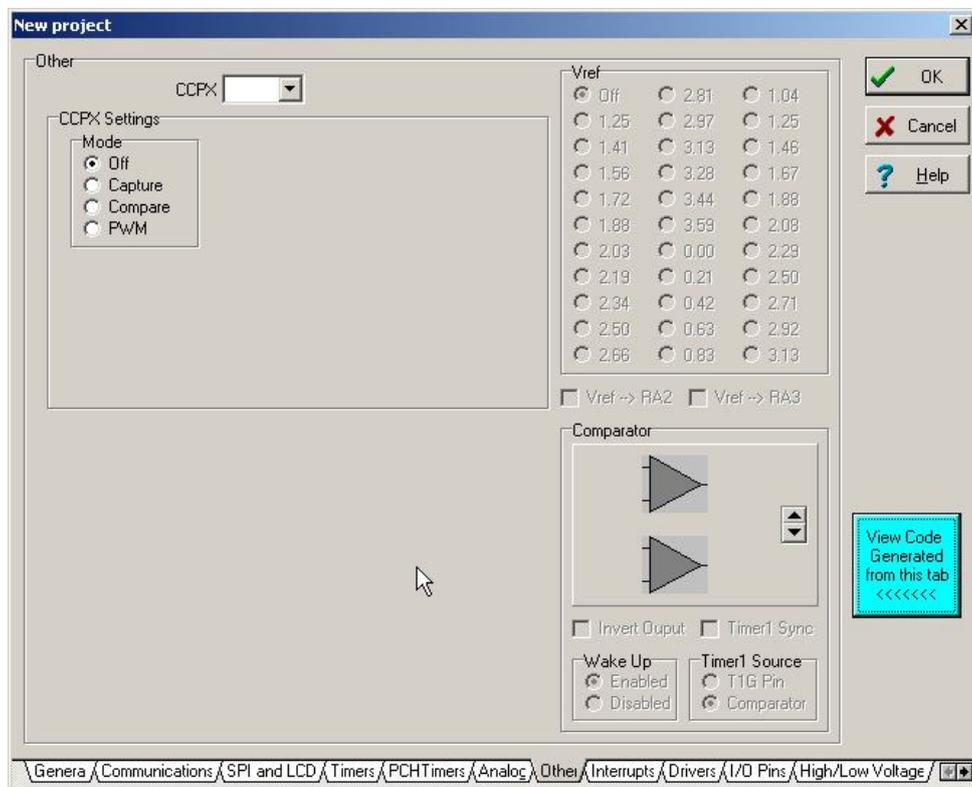
## 9. Pestaña 'PCH Timers':



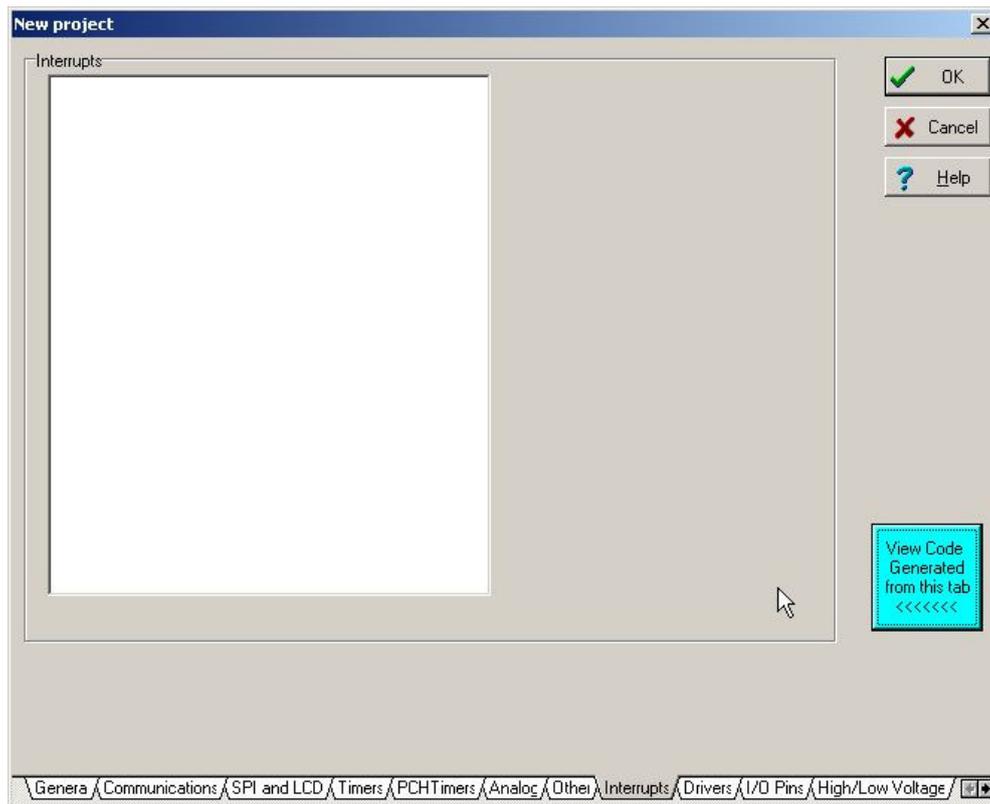
## 10. Pestaña 'Analog':



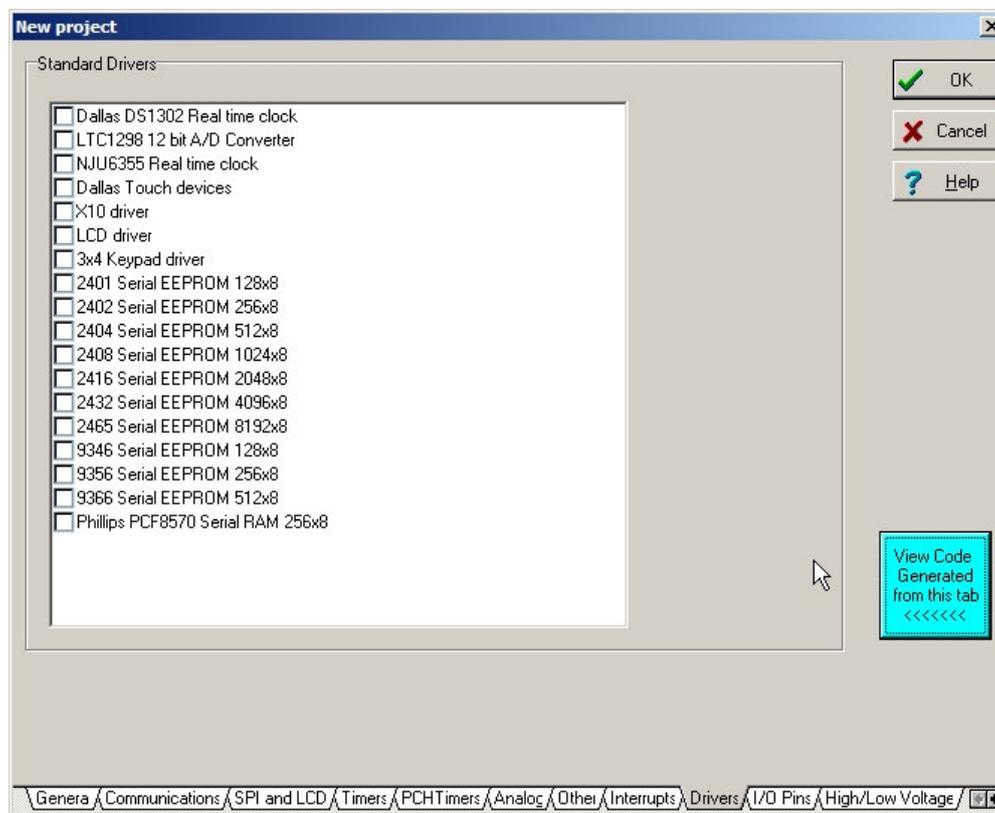
## 11. Pestaña 'Other':



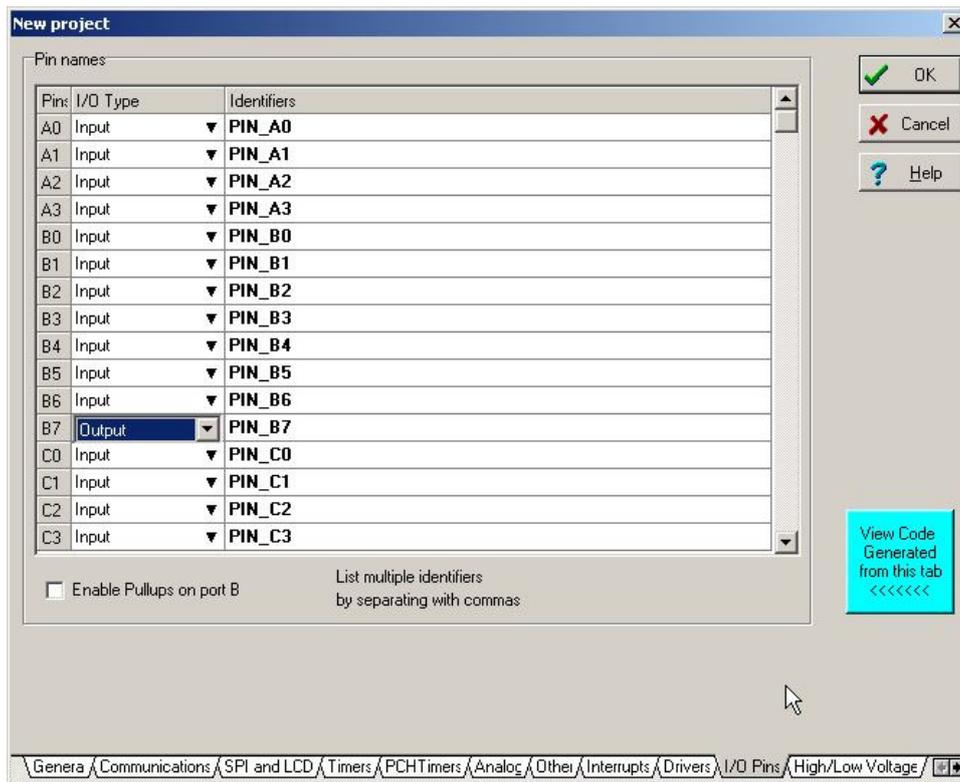
## 12. Pestaña 'Interrupts':



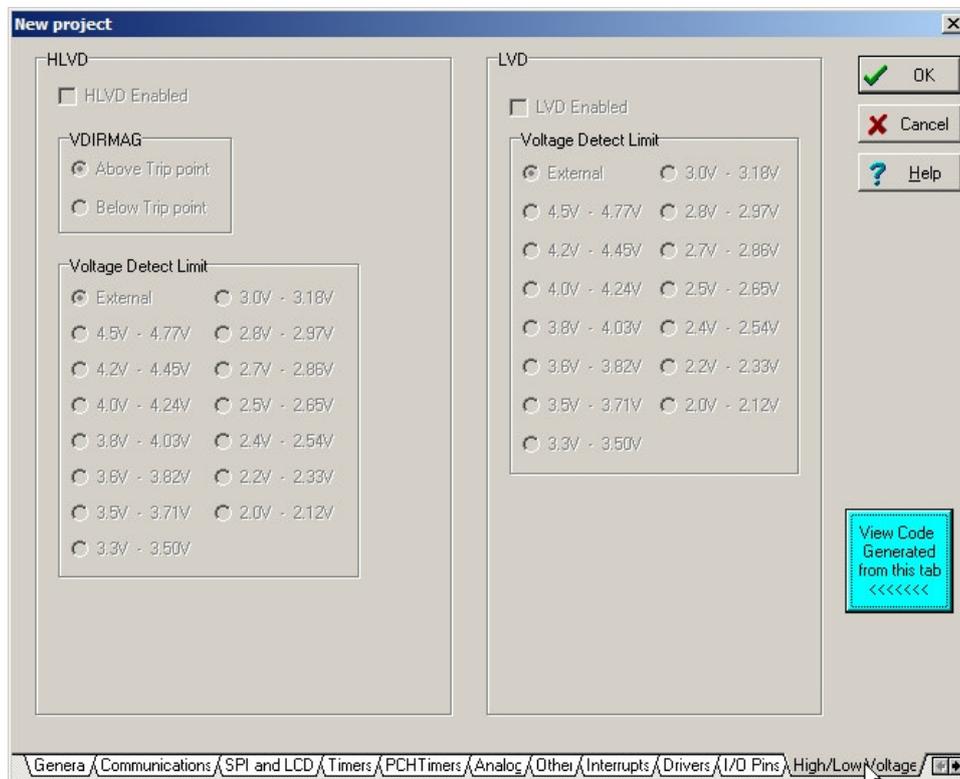
## 13. Pestaña 'Drivers':



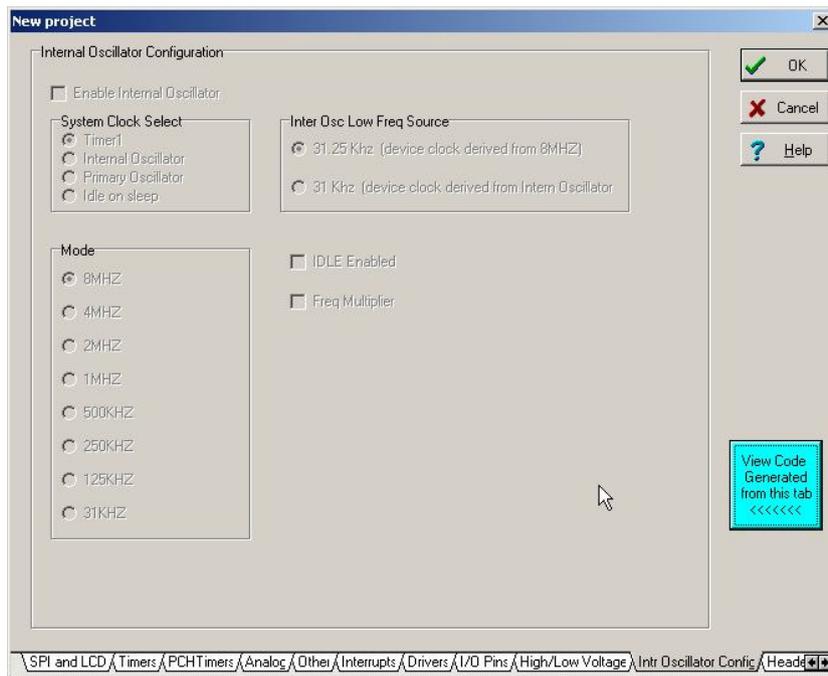
#### 14. Pestaña 'I/O Pins':



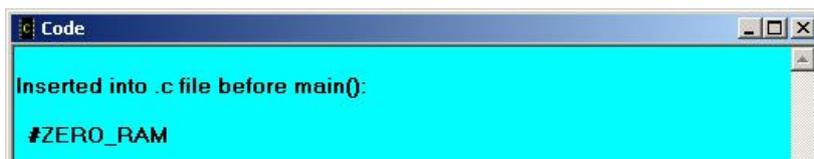
#### 15. Pestaña 'High/Low Voltage':



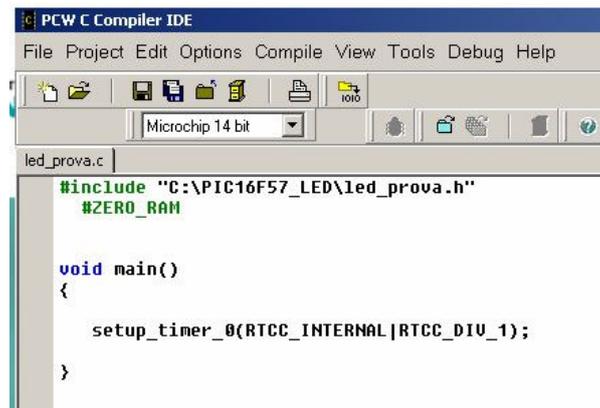
## 16. Pestaña 'Intr Oscillator Config':



## 17. Pestaña 'Header Files':



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



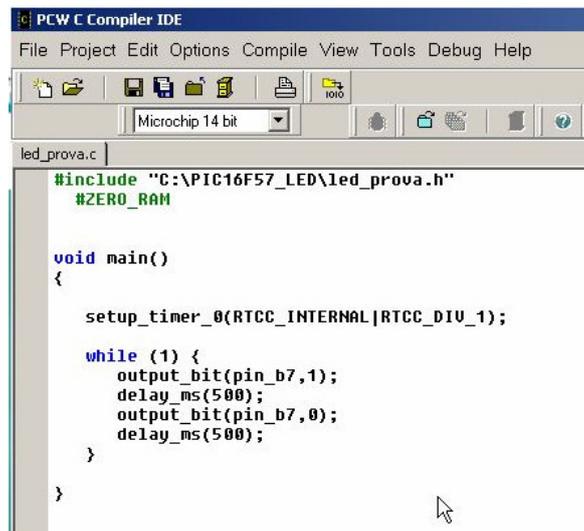
```
PCW C Compiler IDE
File Project Edit Options Compile View Tools Debug Help
Microchip 14 bit
led_prova.c
#include "C:\PIC16F57_LED\led_prova.h"
#ZERO_RAM

void main()
{
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
}
```

Podemos realizar varias observaciones:

- Nuestro proyecto se ha creado con un archivo de código principal 'led\_prova.c', y un archivo cabecera 'led\_prova.h'.
- En la rutina principal (main) podemos observar una llamada a una función de configuración del timer 0 que proporciona el propio compilador: setup\_timer\_0.

19. Escribimos el código necesario para controlar el encendido/apagado del LED cada medio segundo.



```
PCW C Compiler IDE
File Project Edit Options Compile View Tools Debug Help
Microchip 14 bit
led_prova.c
#include "C:\PIC16F57_LED\led_prova.h"
#ZERO_RAM

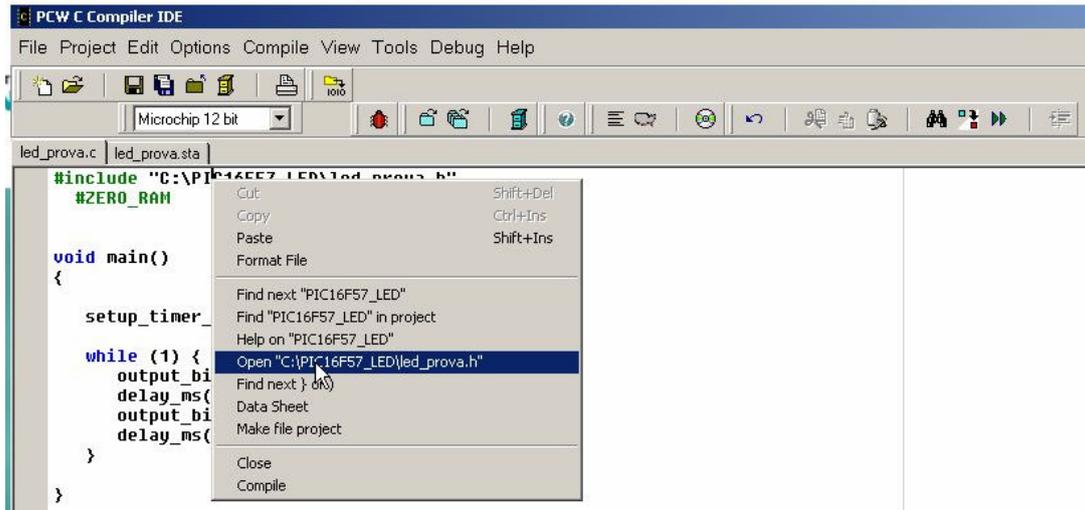
void main()
{
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);

    while (1) {
        output_bit(pin_b7,1);
        delay_ms(500);
        output_bit(pin_b7,0);
        delay_ms(500);
    }
}
```

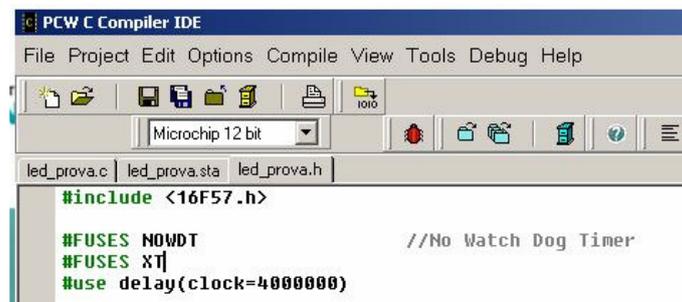
Como se puede ver, se trata de un bucle infinito (while(1)), en cuyo interior se activa y desactiva el bit 7 del puerto b (RB7), con retardo de 0,5 segundos.

20. Vamos a añadir el parámetro XT a la directiva #FUSES en el archivo de cabecera.

Podemos abrirlo clicando con el botón derecho en el fragmento de código que incluye dicho fichero y haciendo click sobre 'Open "C:\PIC16F57\_LED\led\_prova.h"'.  
Añadiendo dicho parámetro establecemos que nuestro cristal funciona a una frecuencia inferior o igual a 4 MHz.

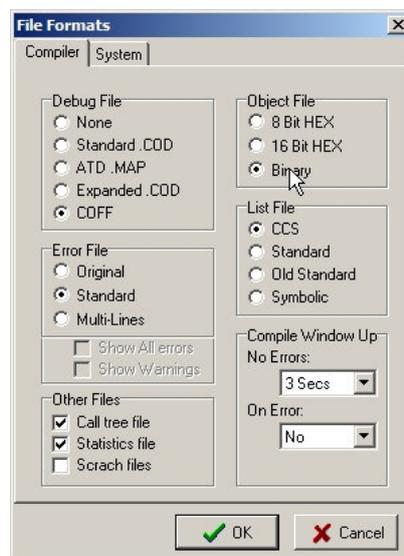


21. Modificamos prova.h añadiendo el parámetro XT a la directiva #FUSES:



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

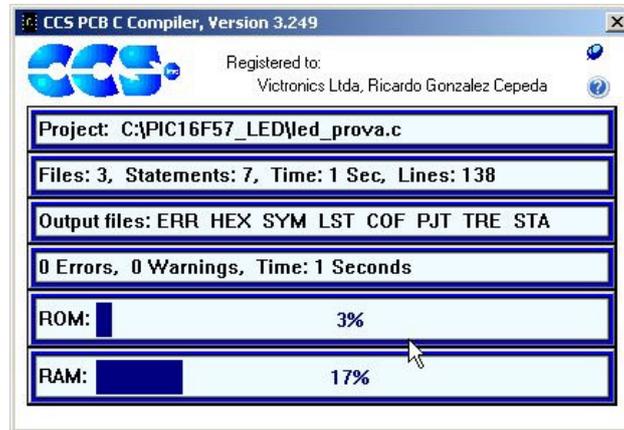
Accedemos al menú Options > File Formats y seleccionamos la opción 'Binary' del marco 'Object File'.



23. 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.

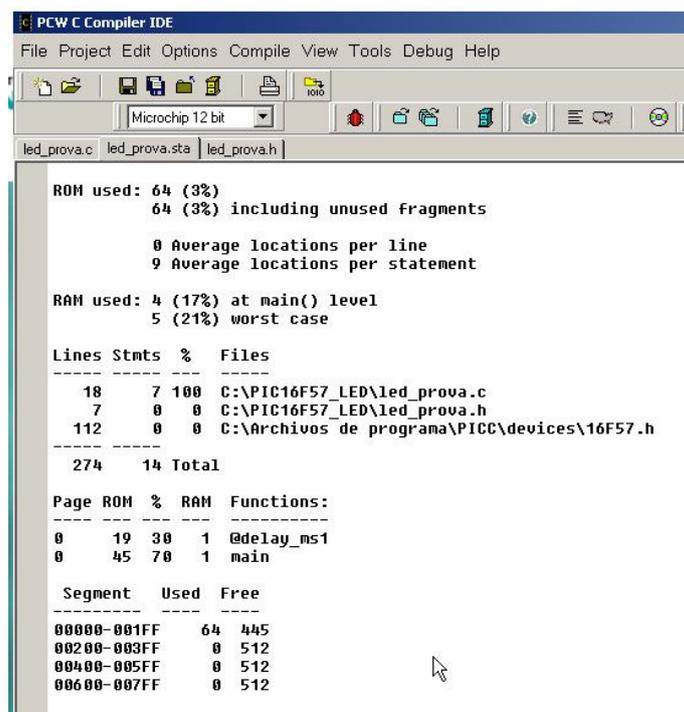
Si todo ha ido bien veremos la siguiente figura:



Obsérvese como una aplicación tan simple apenas consume memoria ROM (programa).

Ya tenemos en el directorio de nuestro proyecto el fichero 'led\_prova.hex' con el código máquina de nuestro micro, cuyo contenido volcaremos con el PIKPROG.

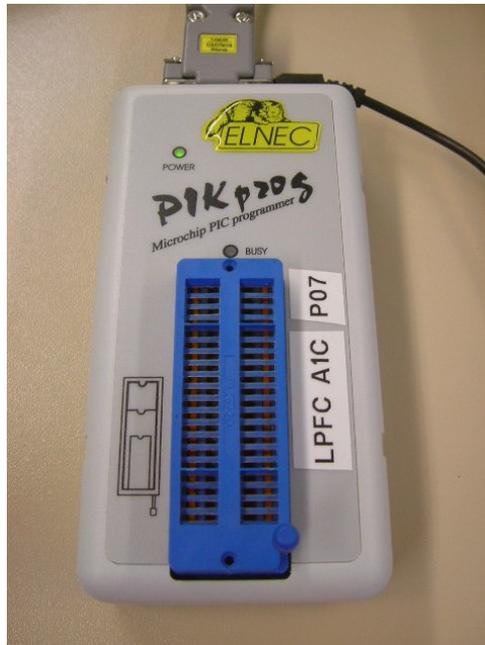
24. Antes de pasar al programador, puede ser útil saber que después de compilar podemos acceder al menú View > Statistics. Se nos mostrará un archivo recopilatorio de datos estadísticos útiles sobre el rendimiento de nuestro programa.



### 3.5. Programación del microcontrolador

1.

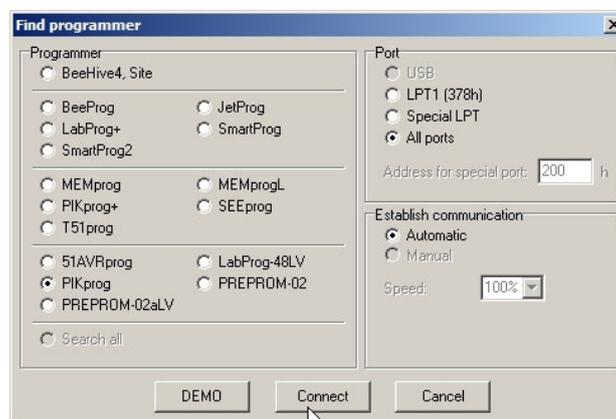
- Conectamos el cable de puerto paralelo al módulo programador y al PC.
- Alimentamos el PIKPROG con el transformador. El LED de power (verde) se encenderá (es importante no tener ningún chip en el zócalo).



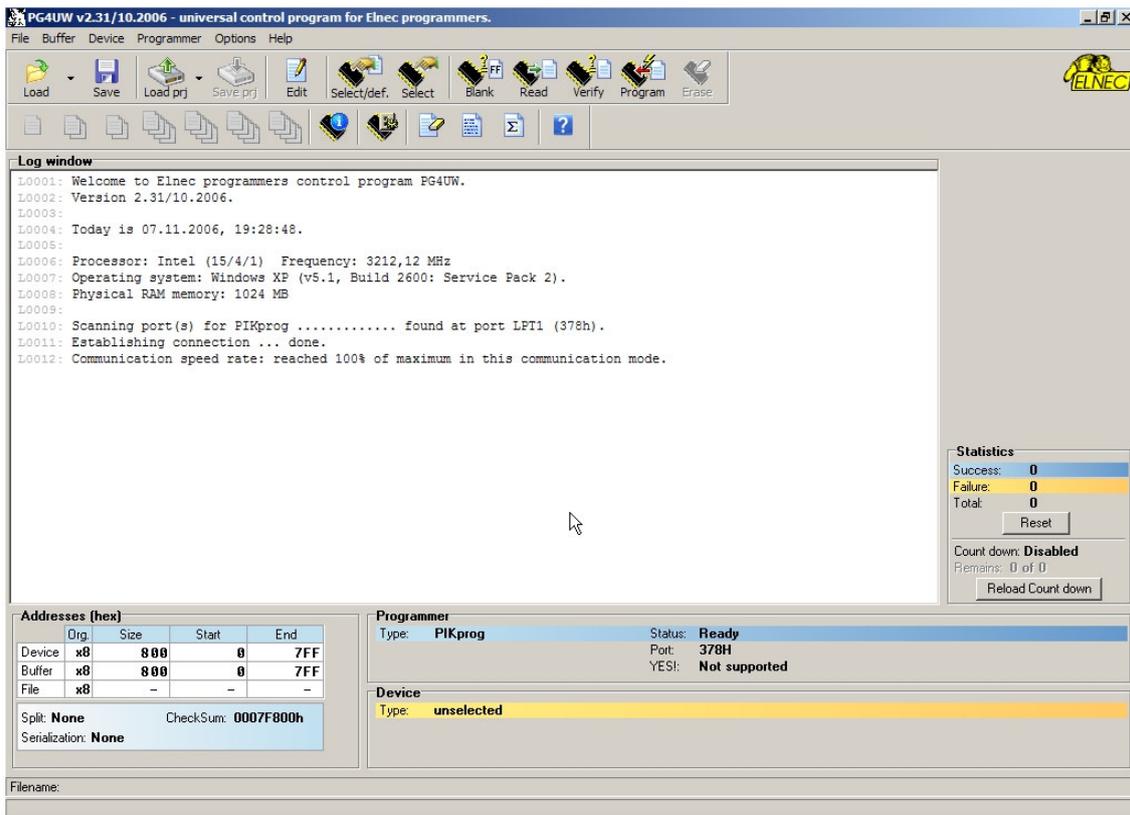
2. Iniciamos el Pg4uw, la aplicación para programar los integrados.



3. Se nos preguntará qué programador queremos utilizar. Seleccionamos el PIKprog. Click en 'Connect'.



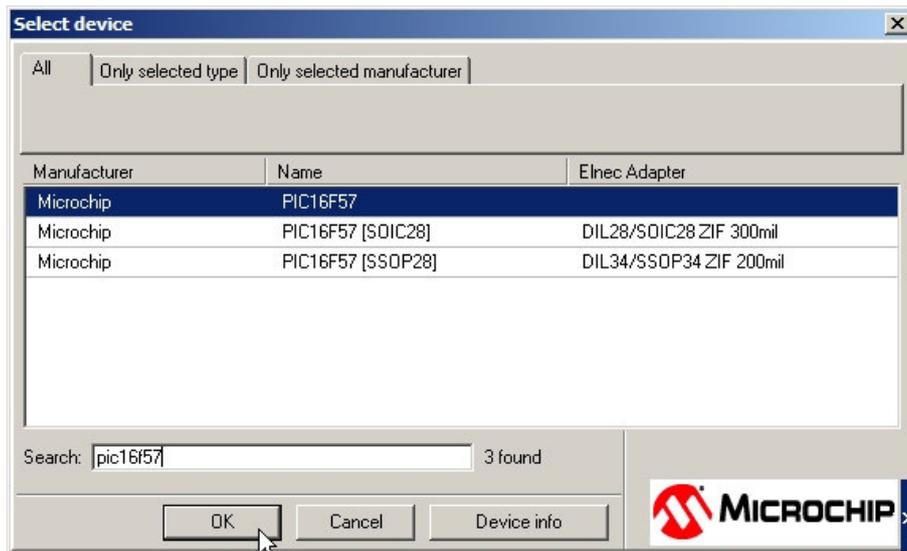
4. Si se establece comunicación con el módulo programador de forma satisfactoria observaremos la siguiente pantalla:



5. Seleccionaremos el micro a programar. Para ello:

Device > Select device...

6. Buscamos el PIC16F57. Hay que tener en cuenta que de los tres que se nos ofrecen nos quedaremos con aquél que no necesita adaptador de encapsulado.



7. Nos aparecerá el siguiente cuadro de diálogo informándonos que la palabra de configuración está establecida a los valores por defecto. Más adelante la modificaremos.

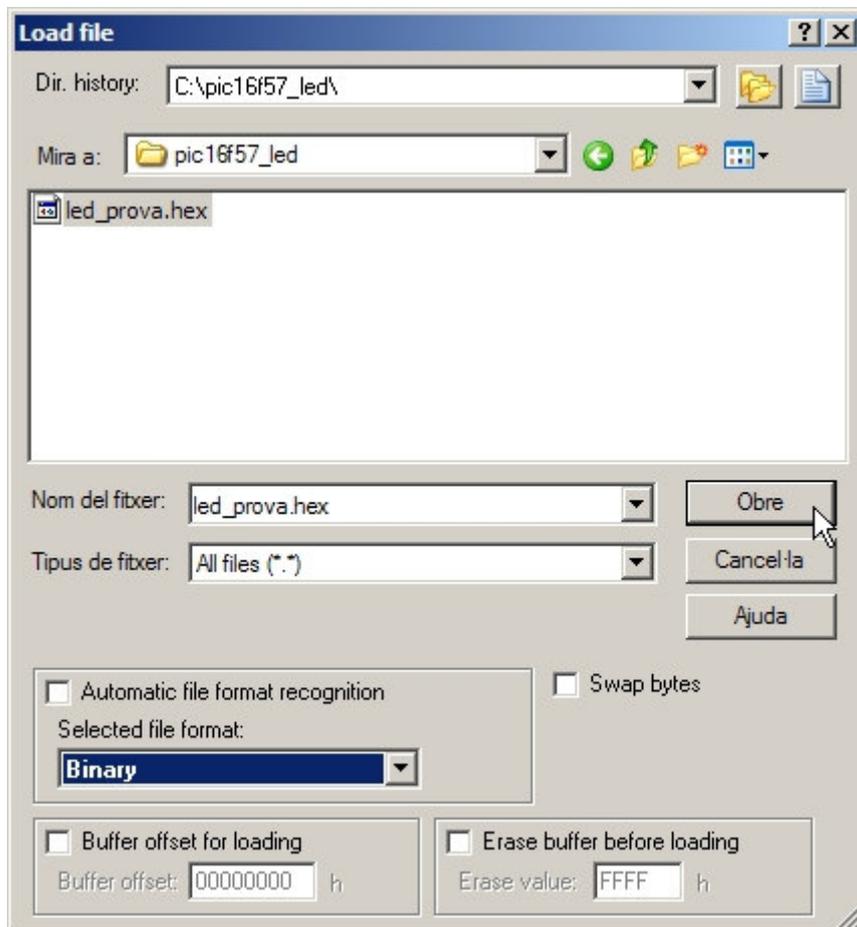


---

8. Click sobre el botón LOAD.

---

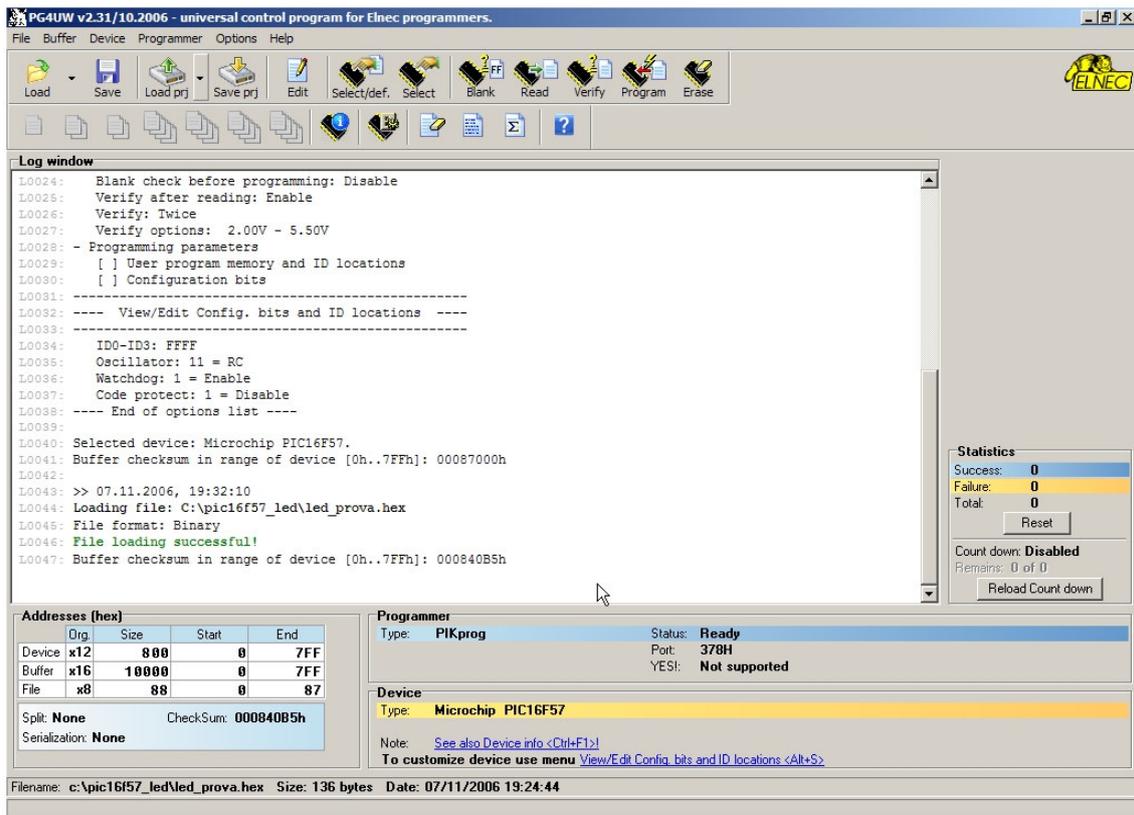
9. Seleccionamos el archivo .hex creado con el compilador.



Obsérvese como el formato que se interpretará es 'Binary'.

---

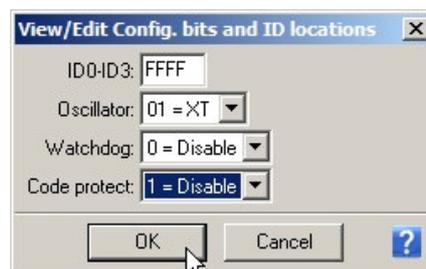
10. Si la lectura ha sido correcta observaremos la imagen que se muestra en la a continuación.



11. Lo siguiente a realizar será editar los bits de la 'Config. Word'.

Device > Device options > View/Edit Config. Bits and ID locations...

12. Desactivamos el Watchdog, la protección del código de programa contra lecturas y seleccionamos XT como oscilador del montaje.



13. Llegó el momento de programar el dispositivo. Insertamos el chip en el zócalo, levantando previamente la palanquita de fijación. Préstese especial atención en la disposición del integrado. Puede tomarse como referencia la ilustración de la página siguiente (el pin 1 se encuentra en la parte superior izquierda).

Una vez insertado, bajamos el bracito del zócalo.



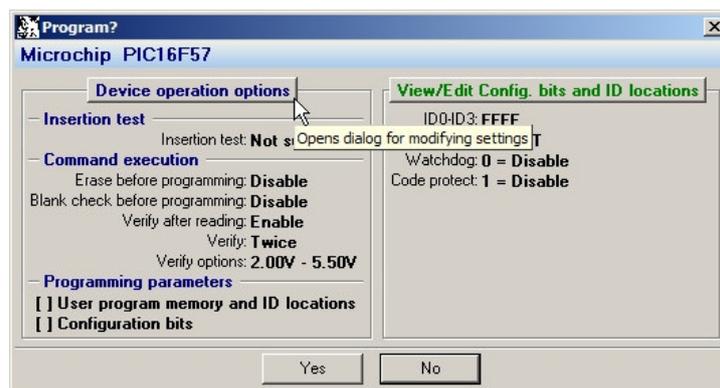
---

14. Hacemos click en el botón 'Program'.

---

15. Aparecerá un cuadro de diálogo indicándonos las operaciones que se van a realizar sobre el dispositivo (mitad izquierda).

En la mitad derecha aparecen las opciones seleccionadas para la palabra de configuración.



---

16. Click en el botón 'Device operation options'.

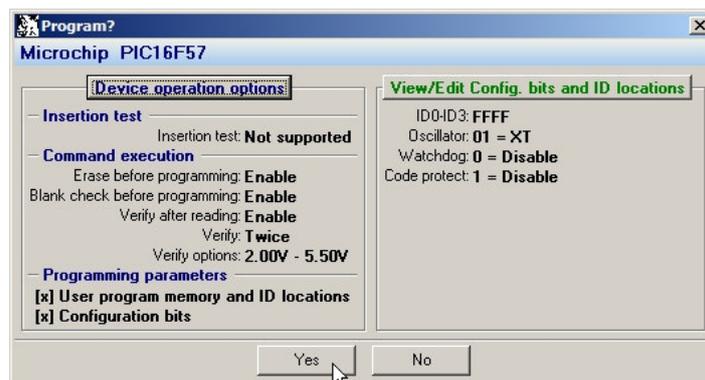
---

17. Activamos la acción 'Erase before programming' y 'Blank check before programming'. De este modo nos aseguramos que el 16F57 no contiene nada antes de programarlo.

También seleccionaremos las dos casillas de verificación de la parte inferior. Con esta acción le indicamos al programa que vamos a escribir lo que hemos cargado en el buffer de la memoria de programa y además que sobrescribiremos la palabra de configuración con las opciones seleccionadas anteriormente.

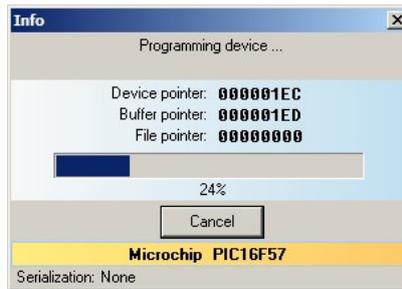


17. La pantalla resumen queda como sigue. Click en 'Yes'.

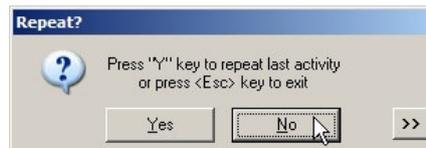


18. El proceso de volcado empieza. El led 'BUSY', de color rojo se encenderá.

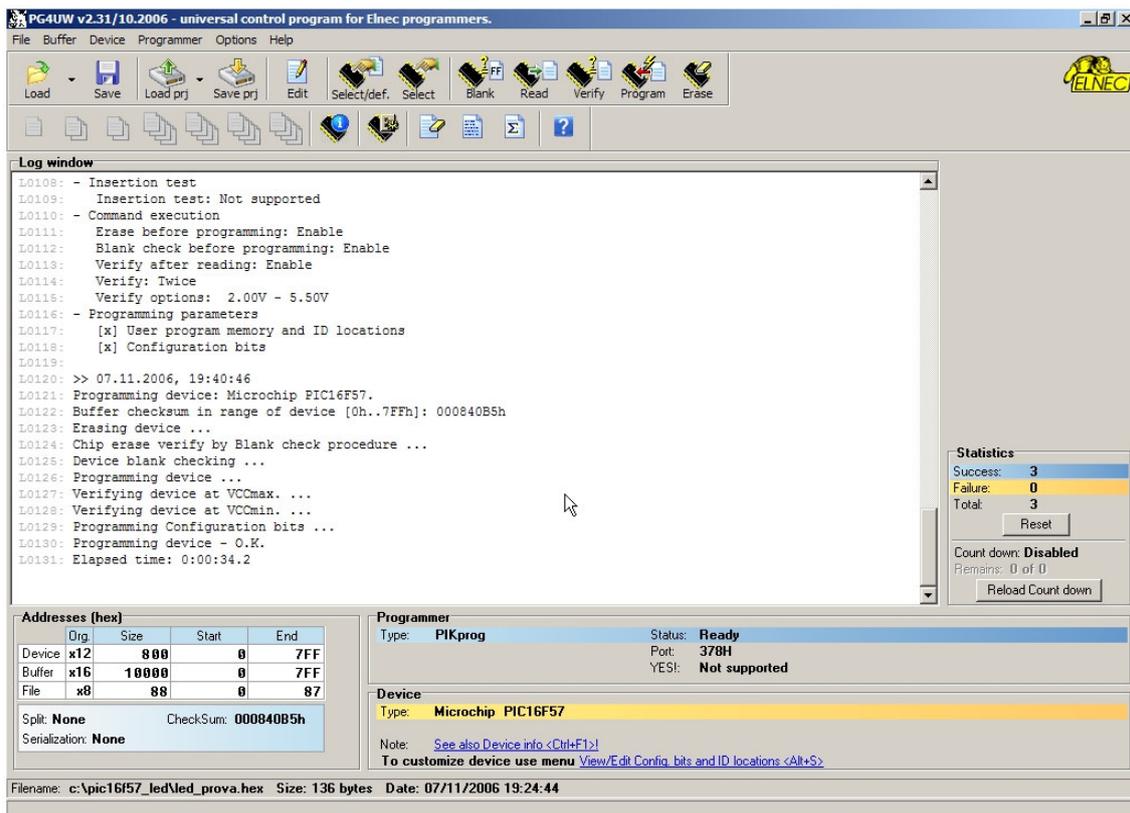




19. Se nos preguntará si queremos repetir el proceso.



20. Si el proceso ha finalizado con éxito, observaremos lo siguiente:



21. Vamos a salvar nuestro proyecto, para emplearlo en futuras ocasiones:

File > Save project

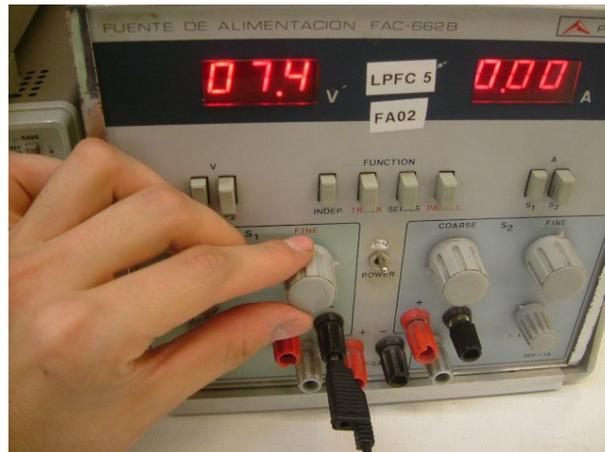
22. Indicamos el nombre y la ruta del archivo .eproj

### 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.

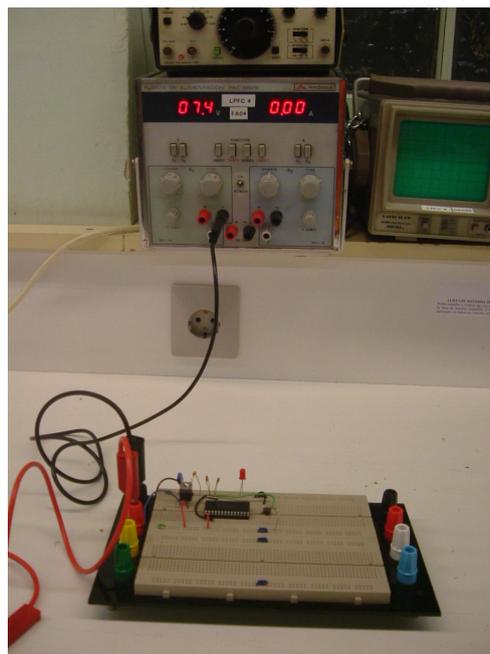
---

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

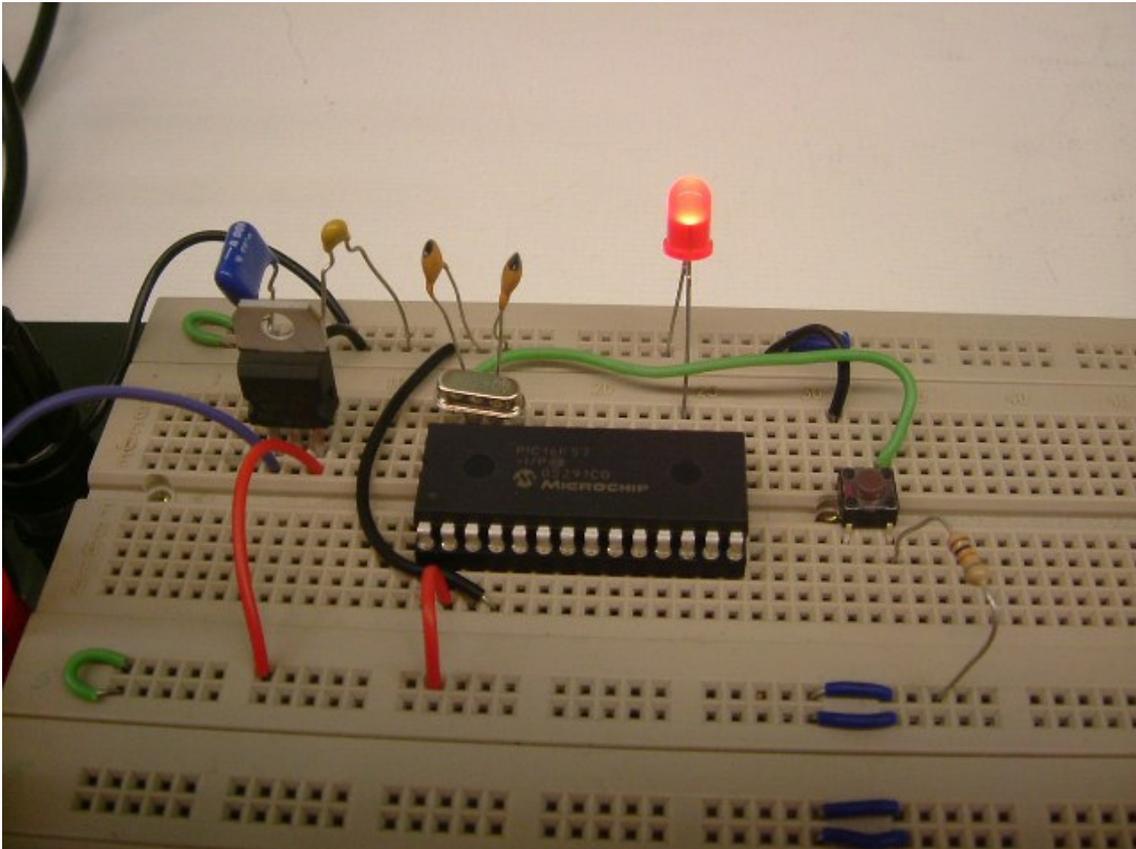


3. Conectamos los cables banana-banana a la protoboard y a la fuente de alimentación, a excepción del borne positivo que corresponde a la fuente. El porqué de proceder así radica en el pico de tensión y corriente que proporciona la fuente al encenderla. Podríamos dañar nuestro micro.

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



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.

---