Ejemplo de utilización del programador MICROCHIP PICSTART PLUS con el microcontrolador Microchip PIC 16F690, el entorno de desarrollo MPLAB IDE 7.50 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, diciembre del 2006

Índice

1. Objetivo de este documento

El programador PICSTART PLUS de la casa MICROCHIP está disponible en el laboratorio de PFCs para aquellos/as proyectistas que lo necesiten. Se trata de uno de los primeros programadores que Microchip proporcionó para programar su amplia familia de microcontroladores de 8 bits, comúnmente conocidos como los "PIC".

En el momento de editar este manual el PICSTART PLUS sigue comercializándose y recibiendo soporte por parte de MICROCHIP, con actualizaciones constantes del firmware (programa interno del módulo programador que permite reconocer los PICs).

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.

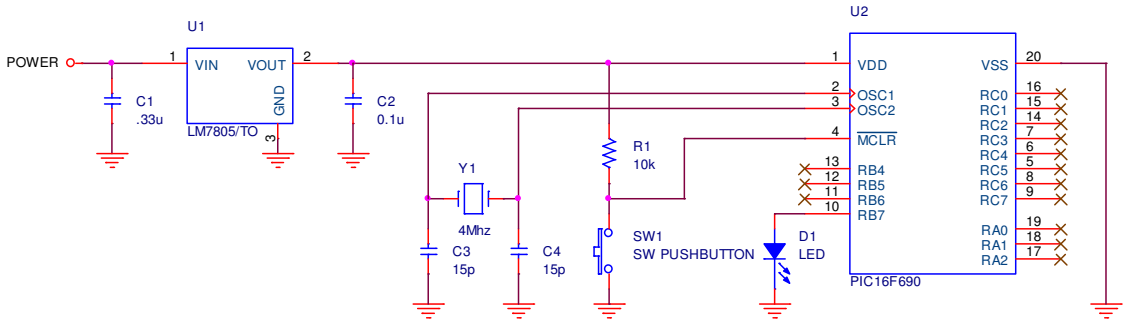
No obstante, es necesario emplear el entorno de desarrollo propio de Microchip, el MPLAB IDE, para realizar el volcado del programa al micro, pues es gratuito y nos permite trabajar con el PICSTART PLUS, así como realizar actualizaciones del firmware del mismo.

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.

Finalmente comentar que el laboratorio de proyectos dispone de un adaptador USB 1.1-RS232 (puerto serie), muy útil para aquellos estudiantes que quieran emplear sus propios portátiles como estaciones de programación y volcado, y no dispongan de puertos de comunicaciones. En este manual se explica su instalación, configuración y uso.

2. Aplicación a montar

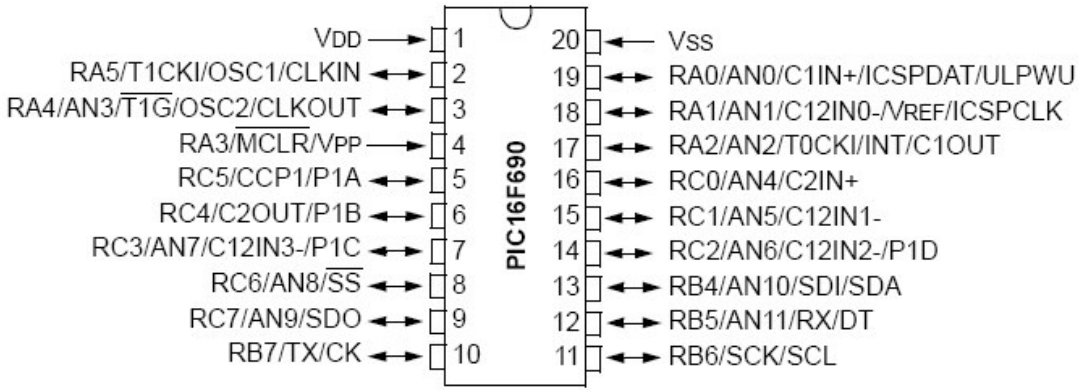
La siguiente figura muestra el circuito que implementaremos en nuestra protoboard.



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 resulta imprescindible 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

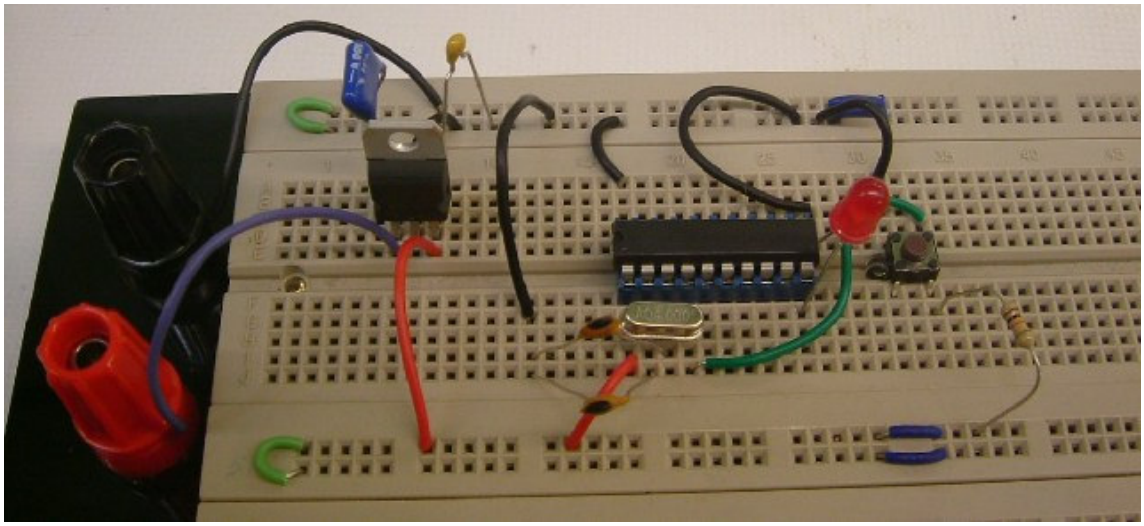
Los subapartados que se exponen a continuación deben seguirse de forma lineal.

No hace falta decir que, en el caso de tener ya instalado el compilador CCS PCWH, el MPLAB o el plugin del CCS PCWH para el MPLAB, podremos prescindir de los apartados 3.2, 3.5 y 3.6, respectivamente. Si no necesitamos emplear el adaptador USB-RS232 puede omitirse el apartado 3.4.

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:

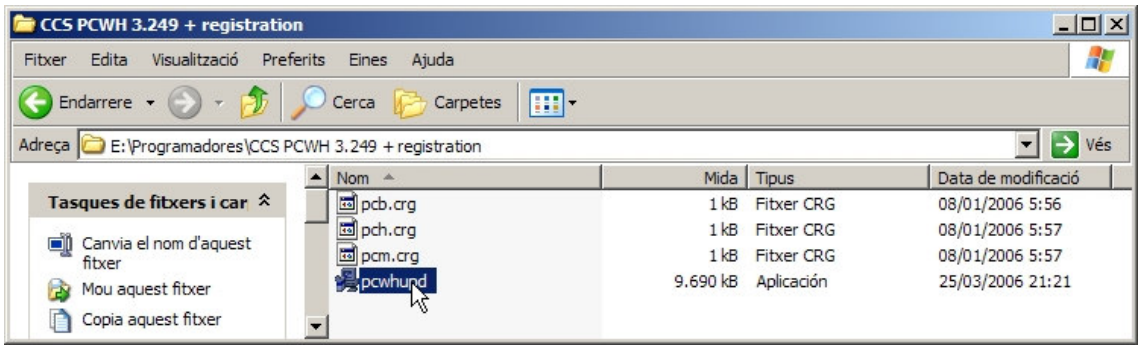


Obsérvese el empleo de un zócalo de 20 pines para nuestro micro. Resulta muy recomendable su uso puesto que al programar varias veces el micro y comprobar su funcionamiento estaremos insertándolo y extrayéndolo de la protoboard continuamente, con lo que podríamos dañar físicamente alguna de sus patitas.

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':

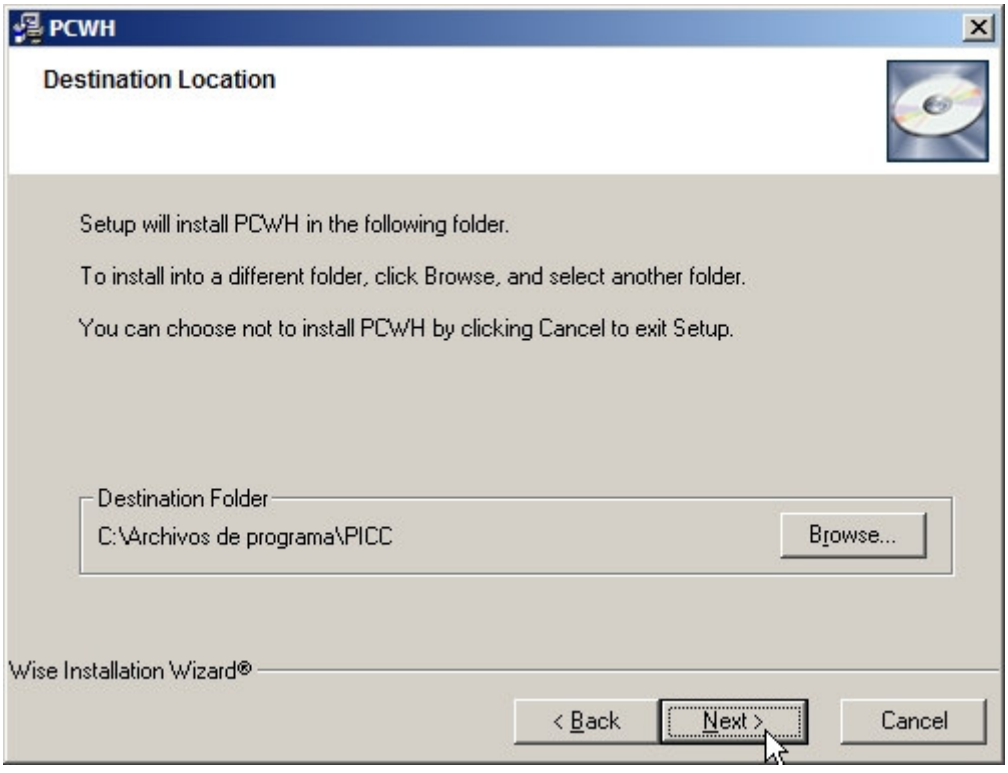


---

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

---

3. Seleccionar la carpeta destino para instalar el programa:

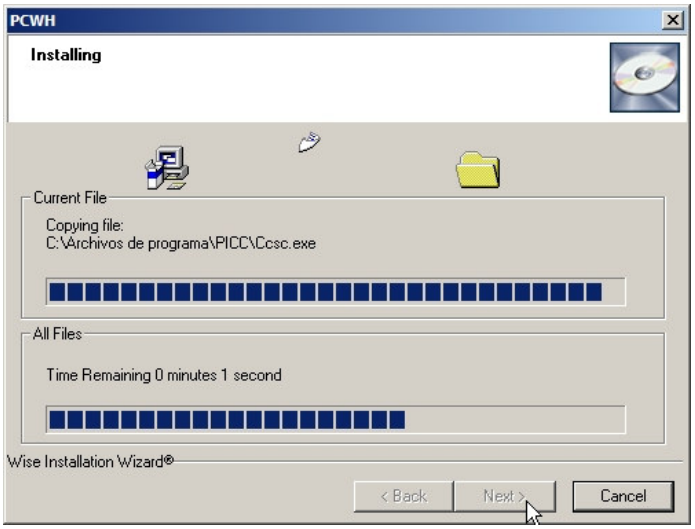


---

4. Clic en 'Next'

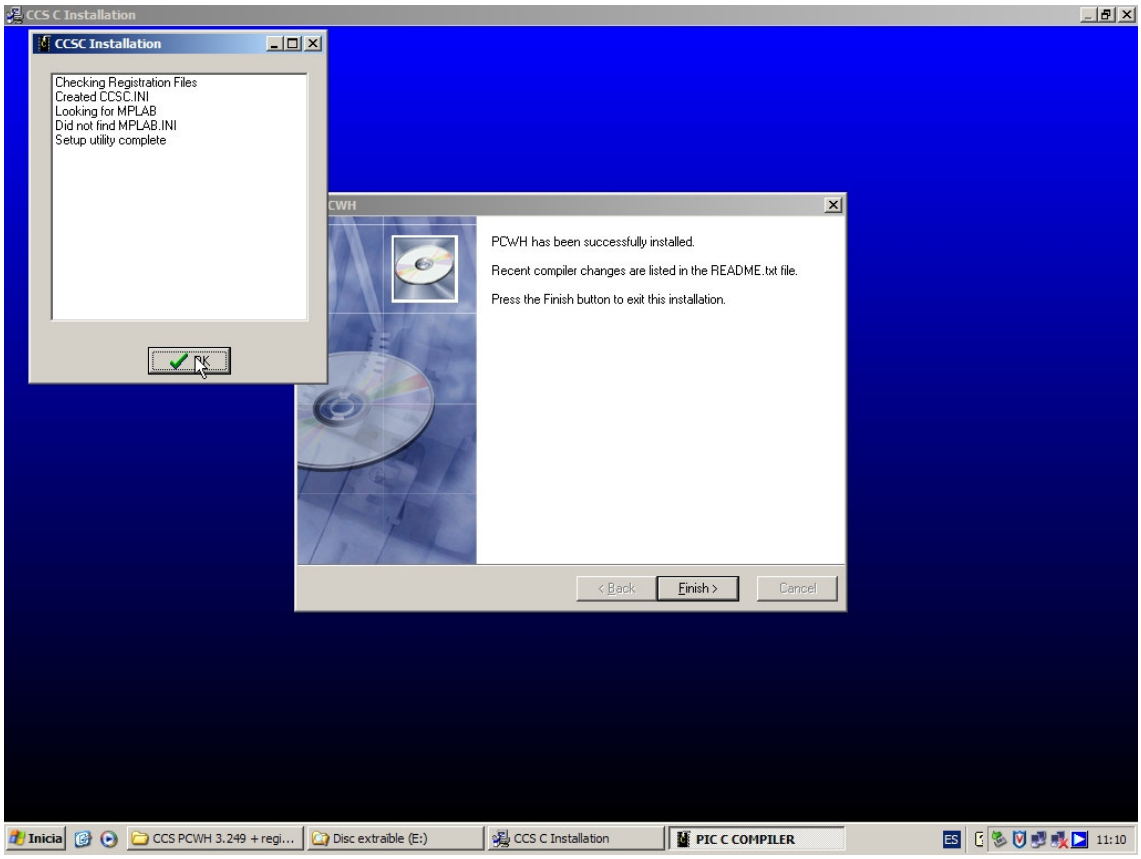
---

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. Creación del programa en C, a ejecutar en el PIC16F690

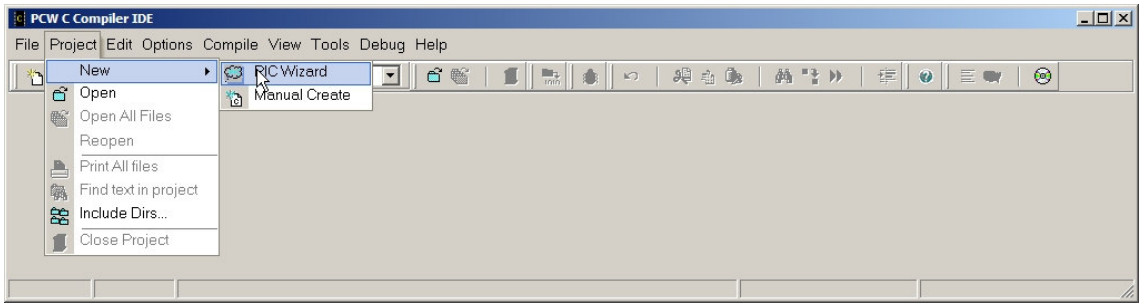
1. Iniciamos el CCS PCWH PIC C Compiler:

Inicio > Programas > PIC-C > PIC C Compiler

---

2. Ejecutamos el asistente para la creación de proyectos:

Project > New > PIC Wizard

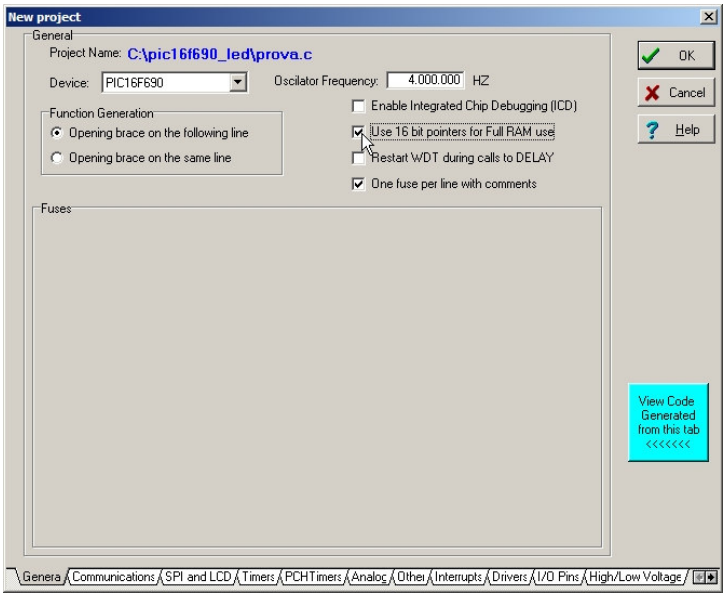


---

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

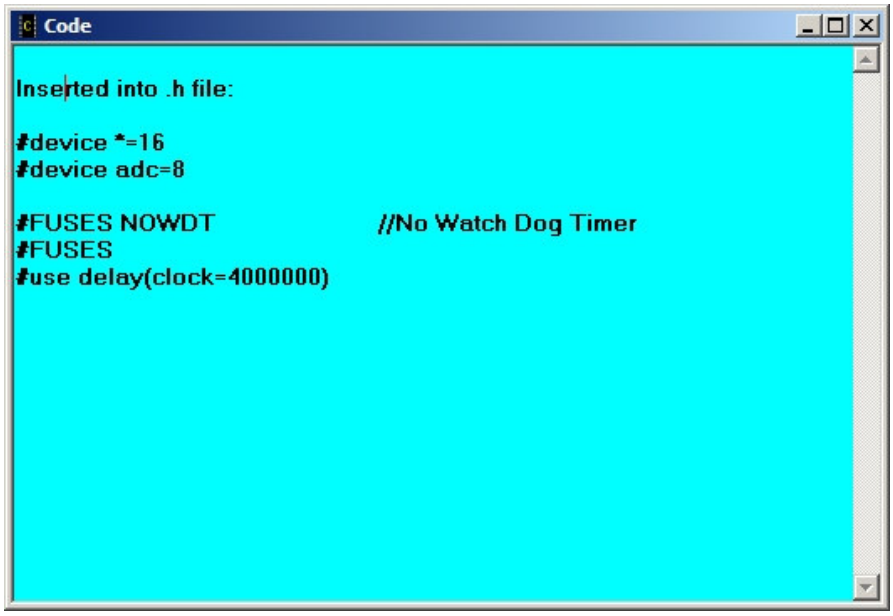
---

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.



---

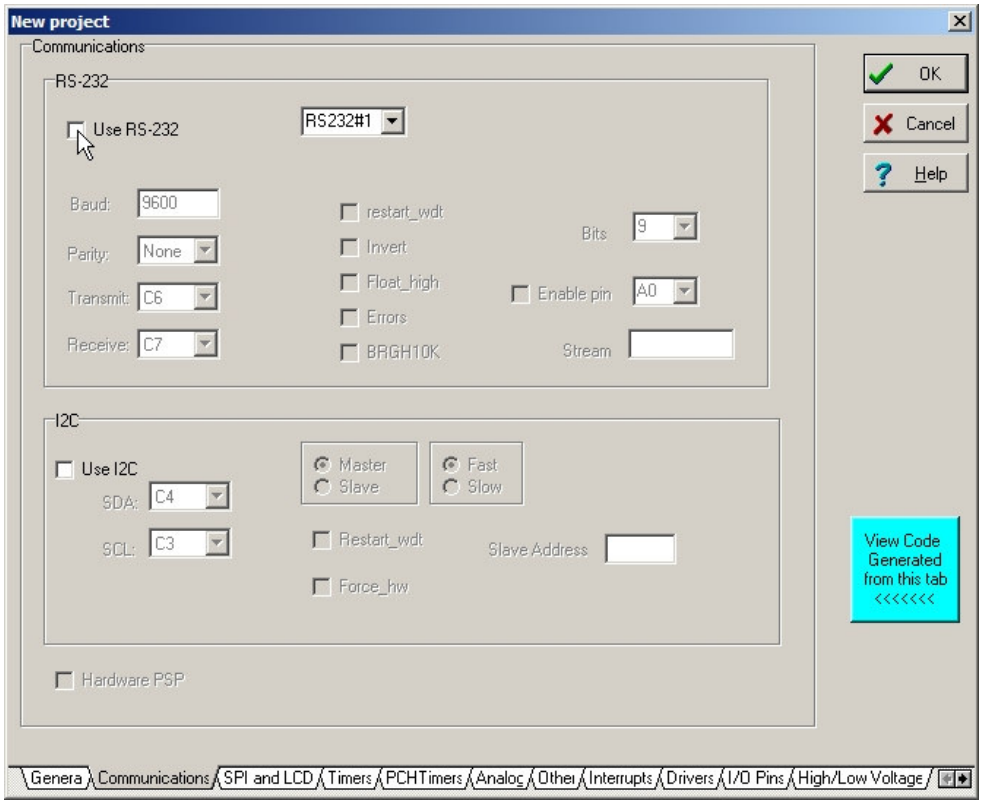
5. Si clicamos en 'View Code Generated from this tab' observaremos la ventana inferior.



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.

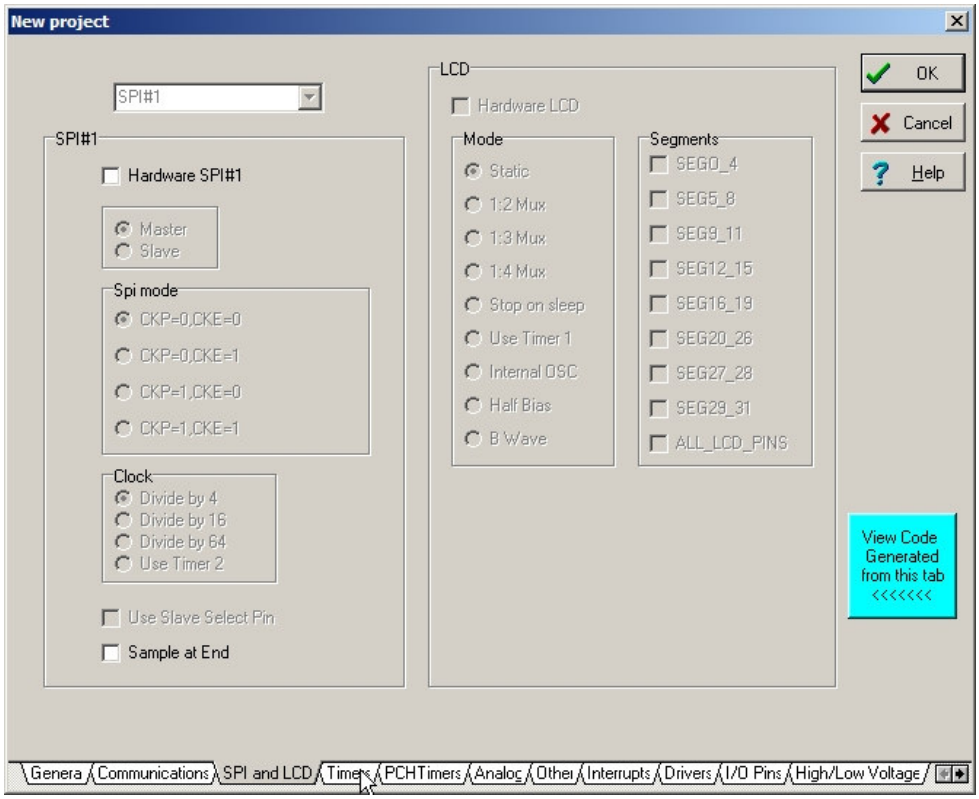
---

6. Pestaña 'Communications':



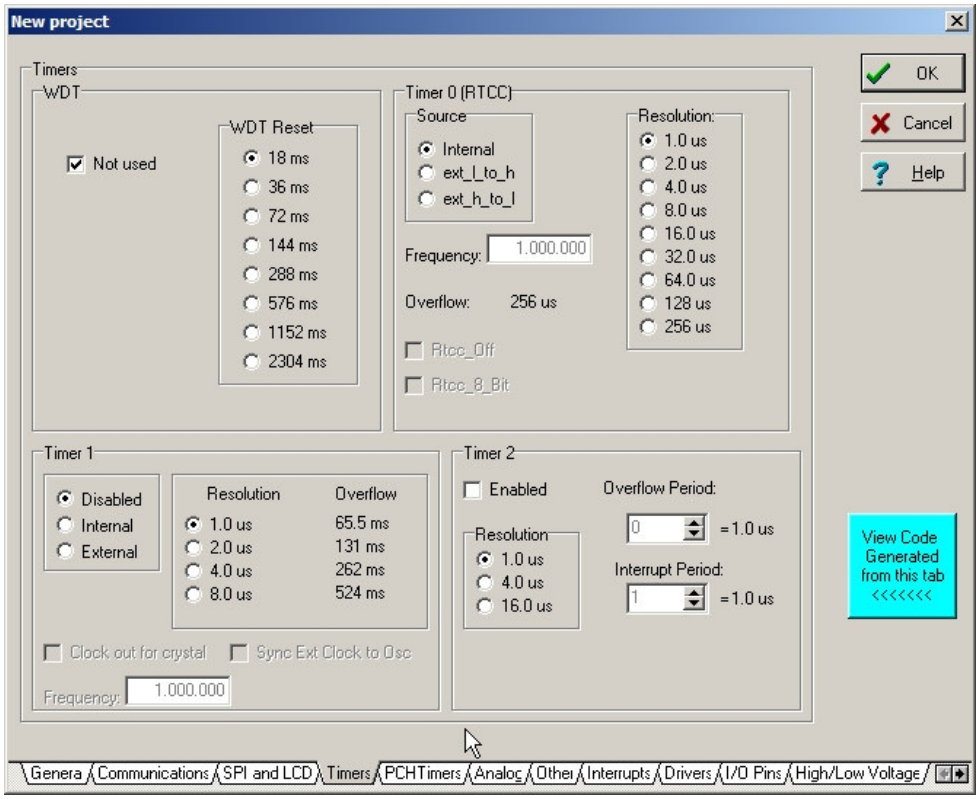
---

7. Pestaña ‘SPI and LCD’:

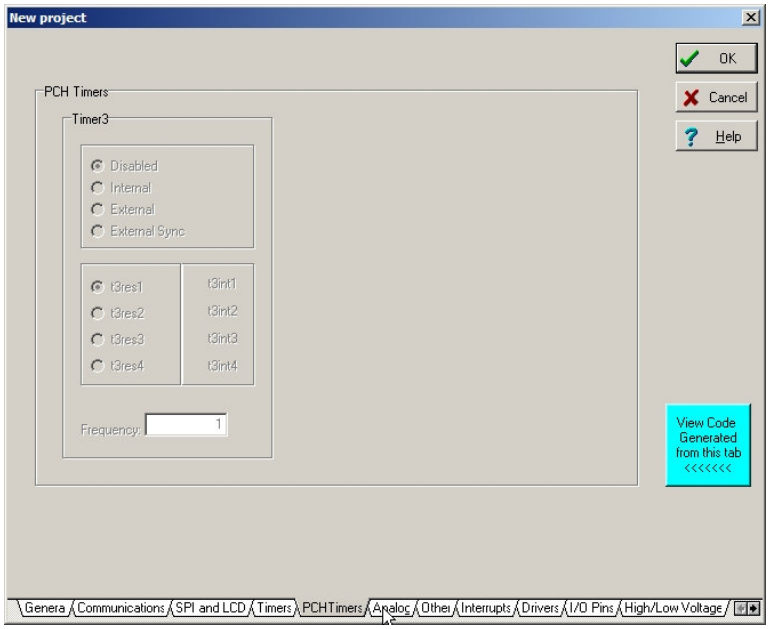


---

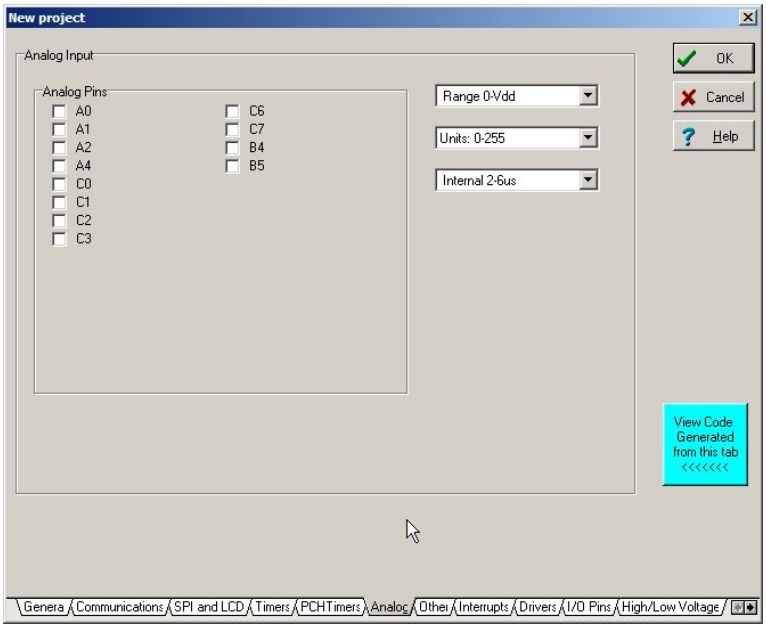
8. Pestaña ‘Timers’:

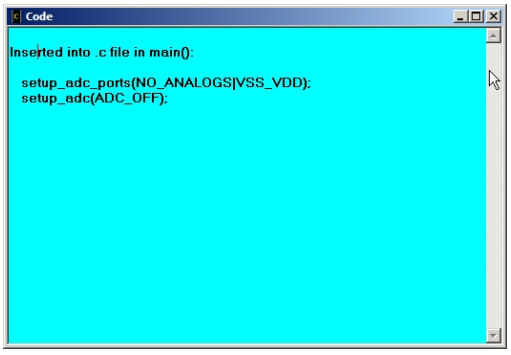


9. Pestaña ‘PCH Timers’:

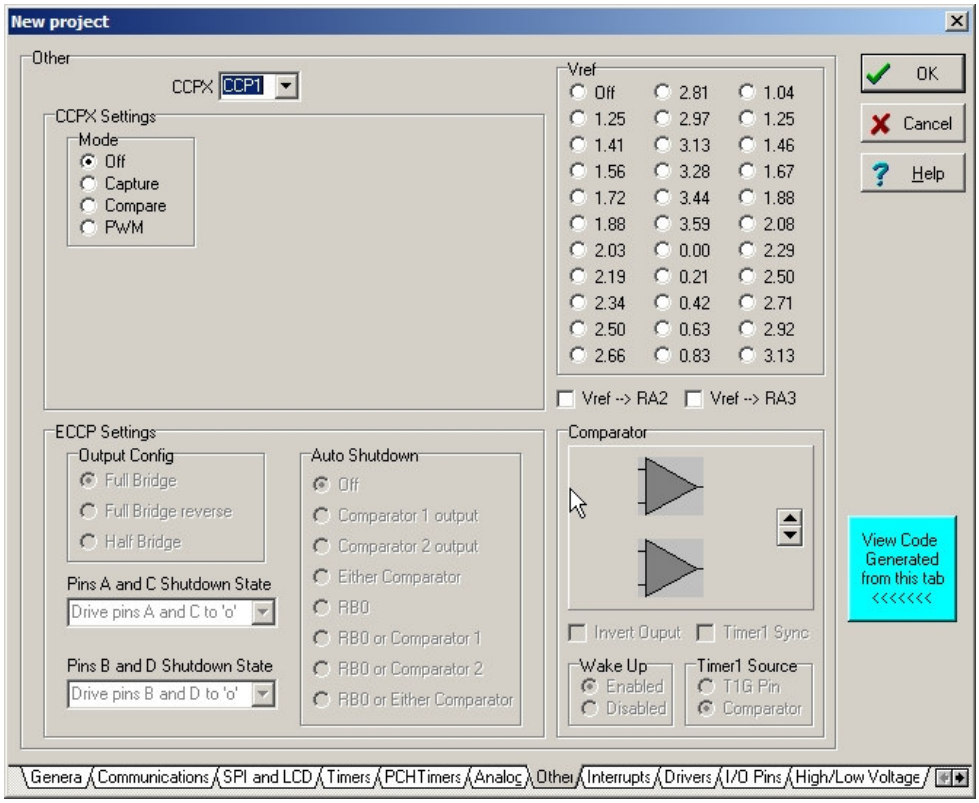


10. Pestaña ‘Analog’:



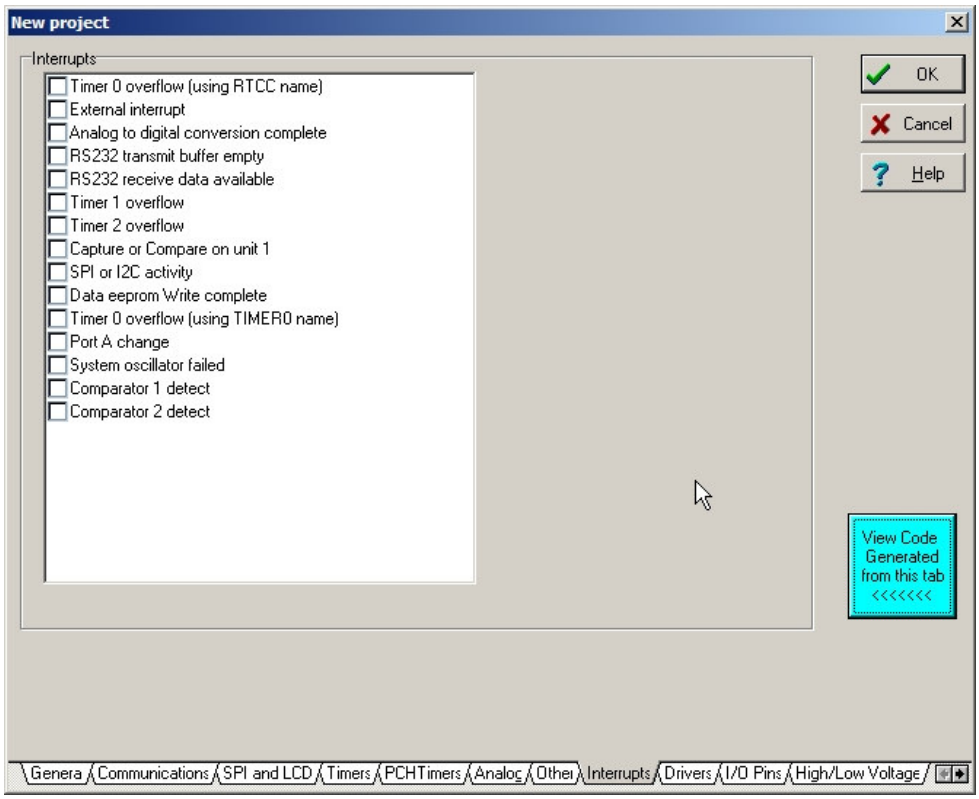


11. Pestaña ‘Other’:

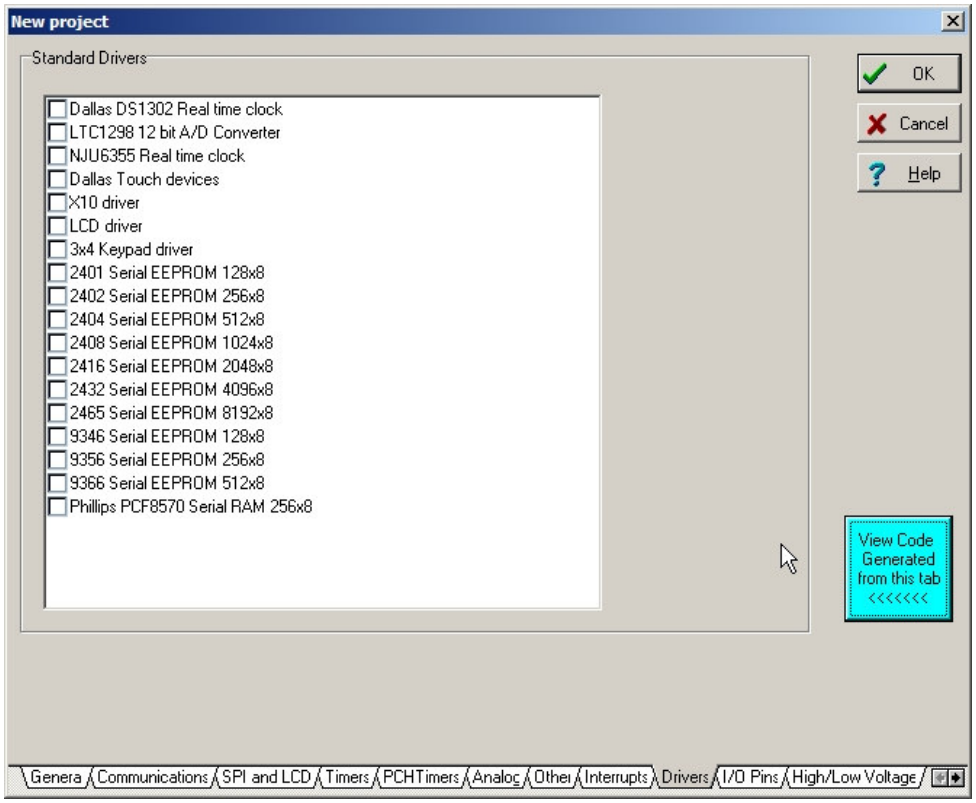


---

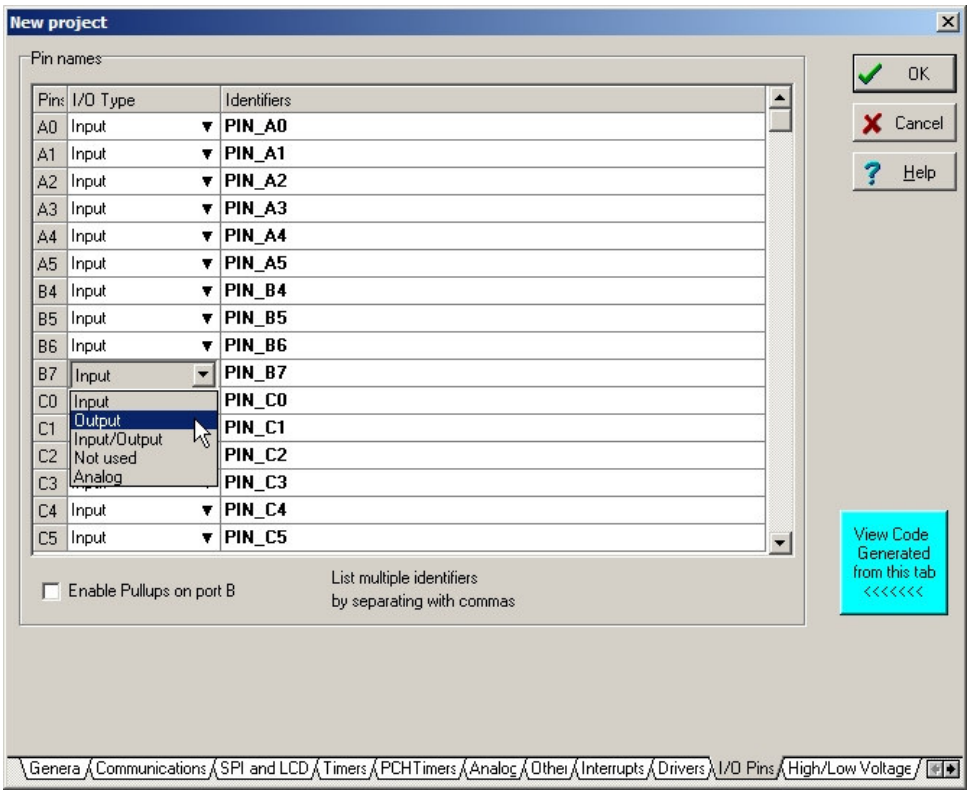
12. Pestaña ‘Interrupts’:



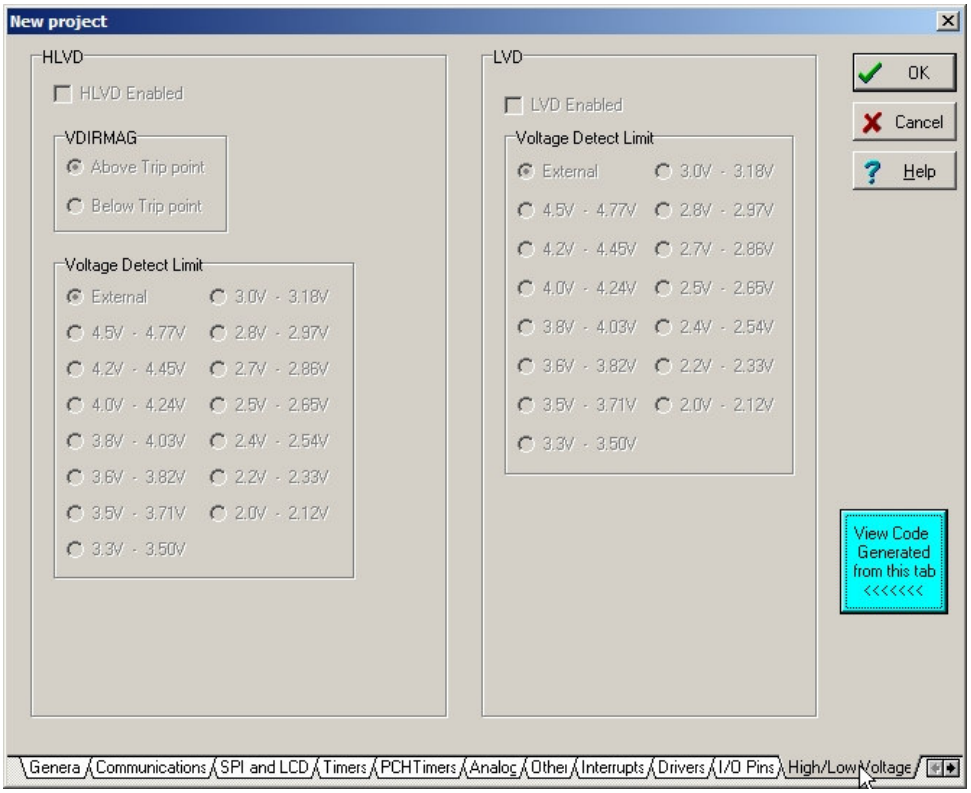
13. Pestaña 'Drivers':



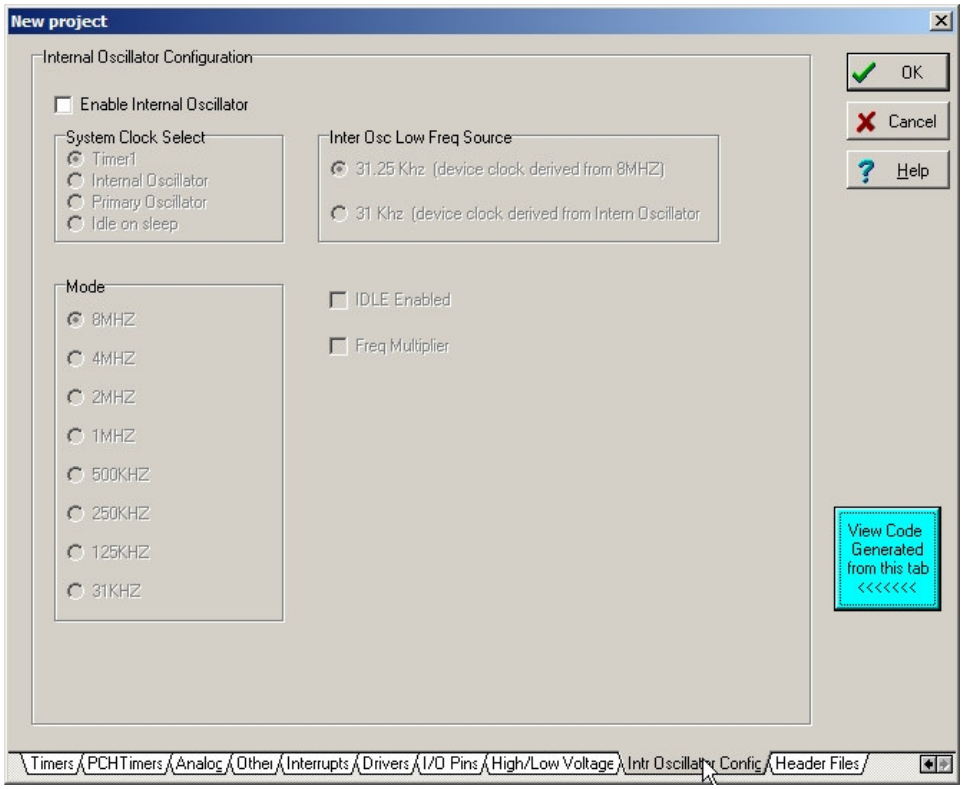
14. Pestaña 'I/O Pins':



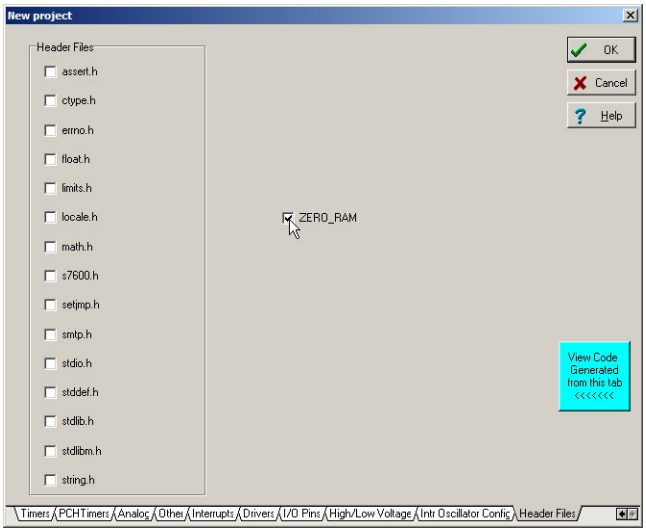
15. Pestaña ‘High/Low Voltage’:

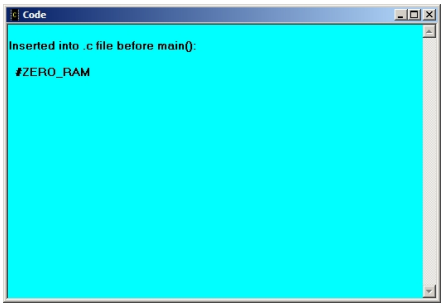


16. Pestaña ‘Intr Oscillator Config’:



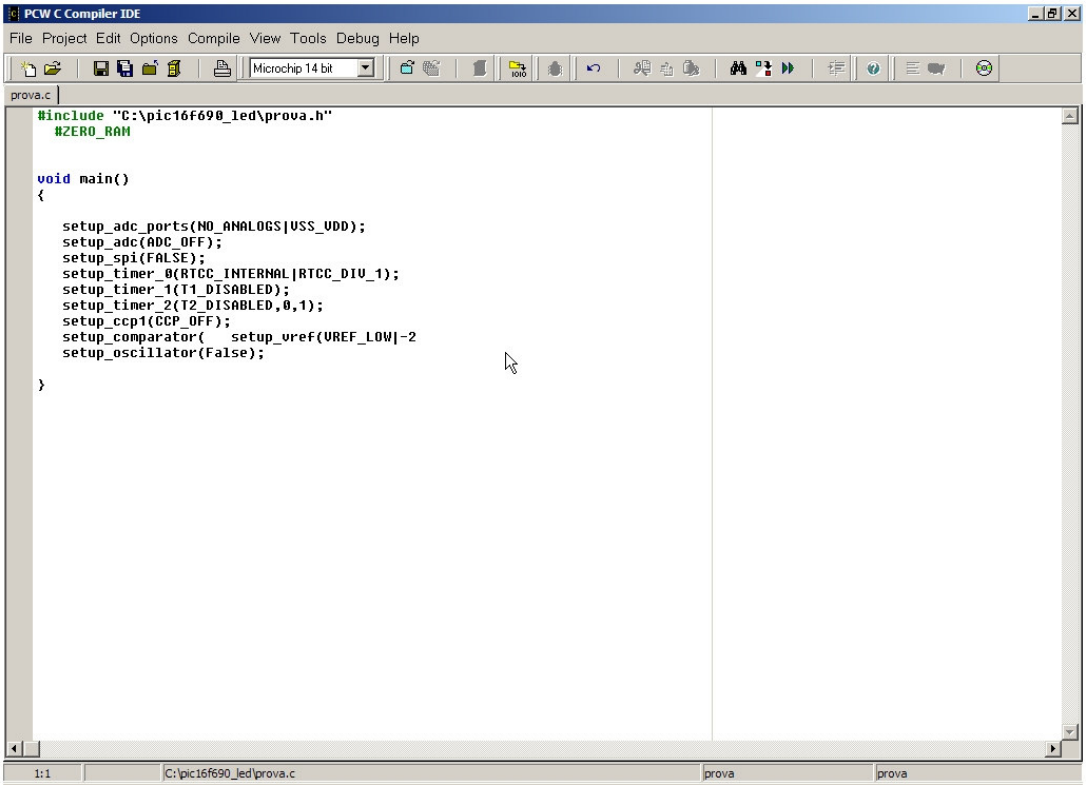
17. Pestaña ‘Header Files’:





---

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



Podemos realizar varias observaciones:

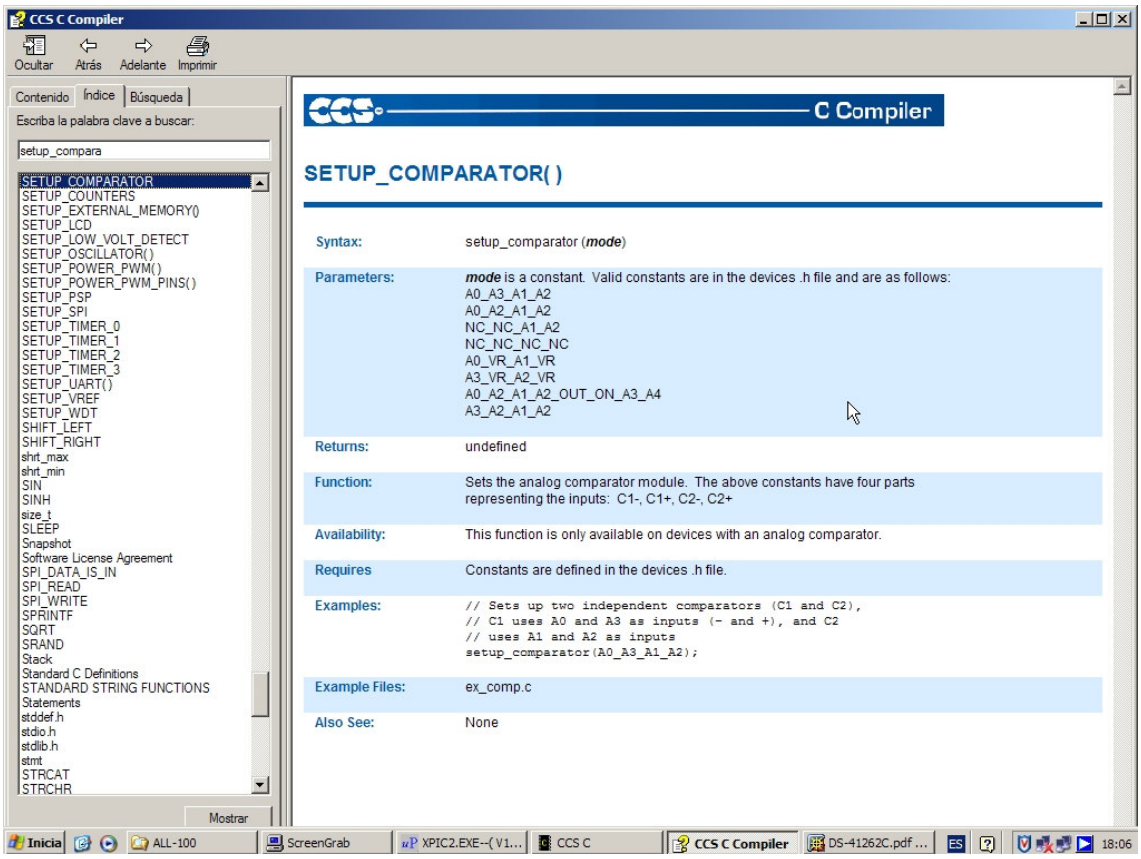
- Nuestro proyecto se ha creado con un archivo de código principal 'prova.c', y un archivo cabecera 'prova.h'.
- En la rutina principal (main) podemos observar una serie de llamadas a funciones de configuración que proporciona el propio compilador: setup_xxx.
- El asistente ha redactado un par de instrucciones 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.



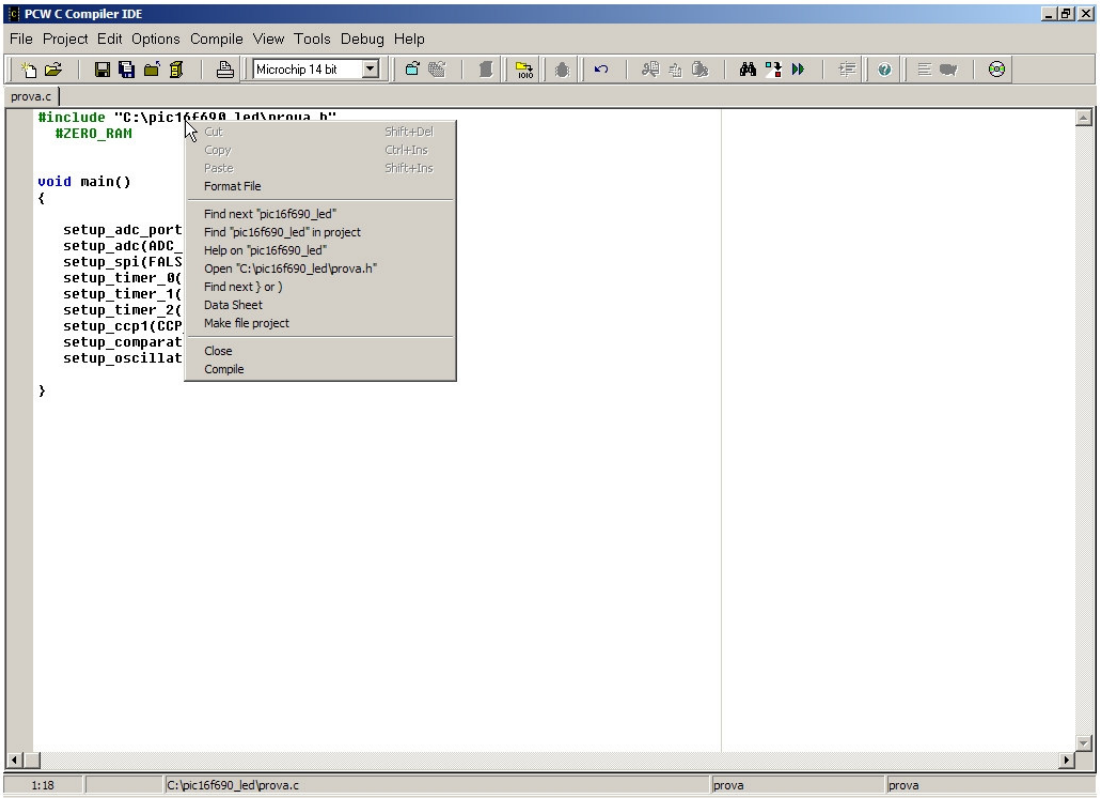
Como se puede observar, la función setup_comparator consta de un único parámetro 'mode', que especifica la configuración de los comparadores del micro. Como no vamos a emplear ninguno, usaremos la constante NC_NC_NC_NC.

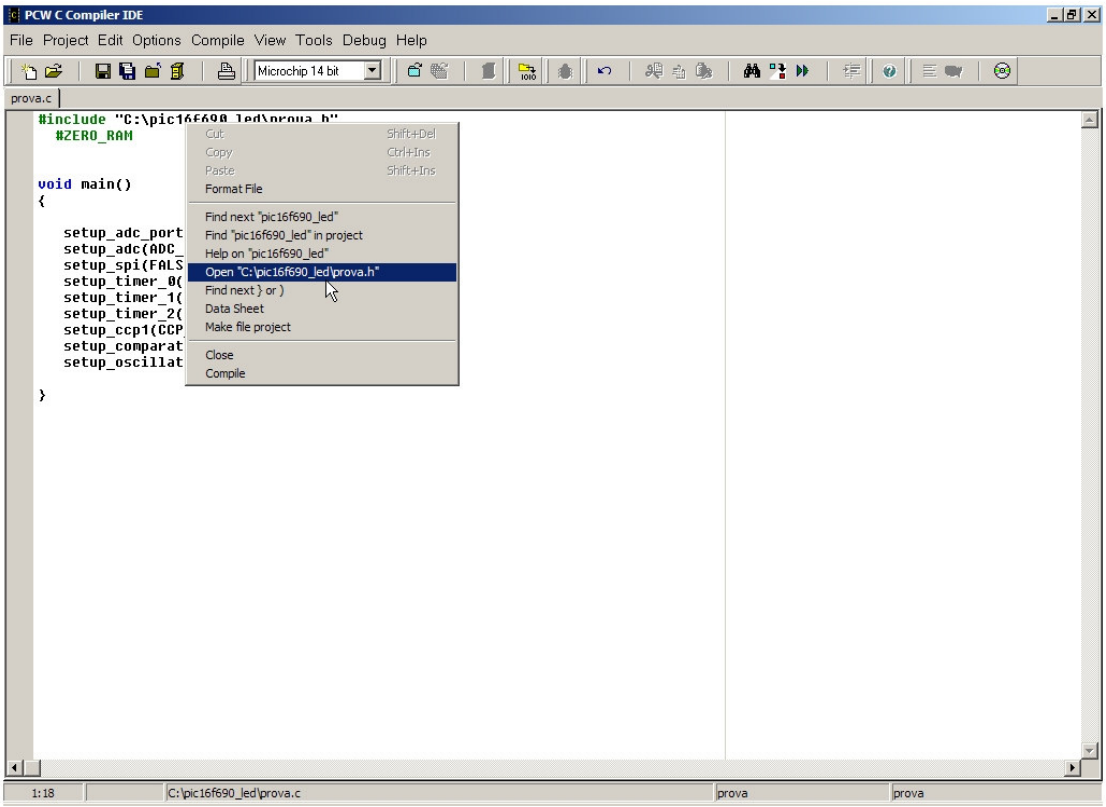
---

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.

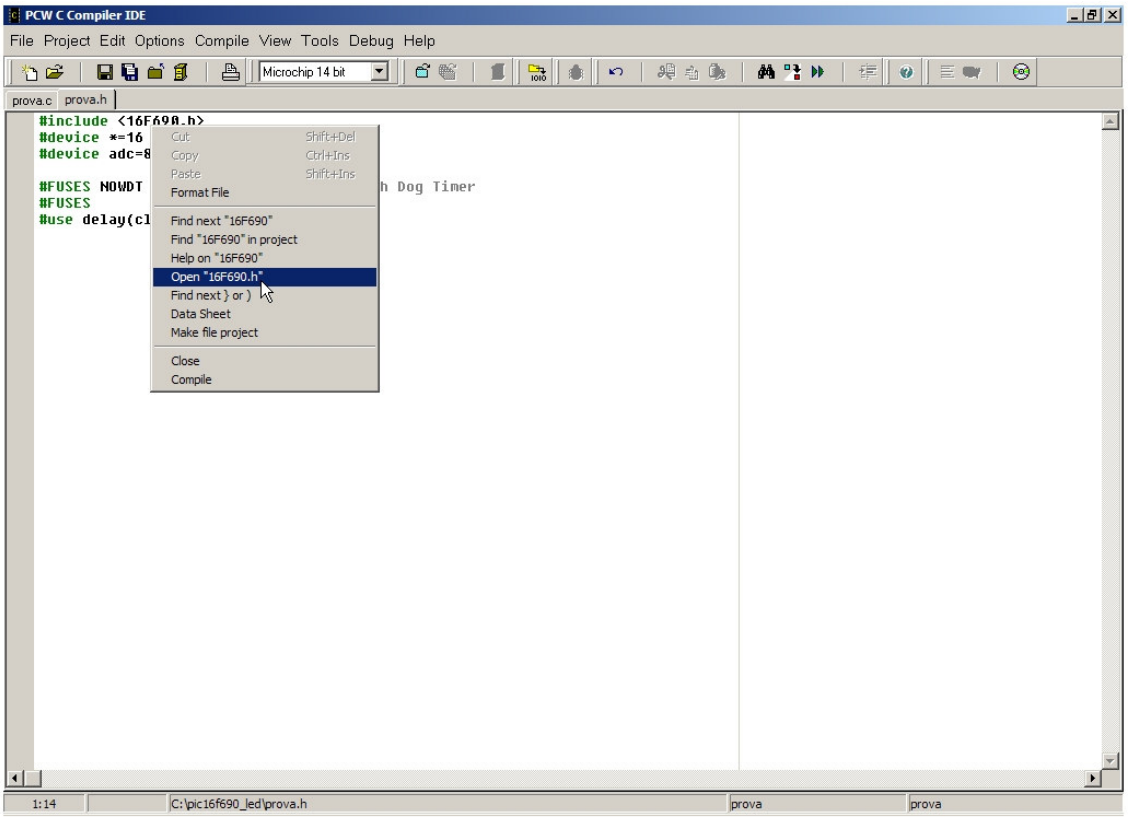
Para acceder al fichero 16F690.h podemos hacer:

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

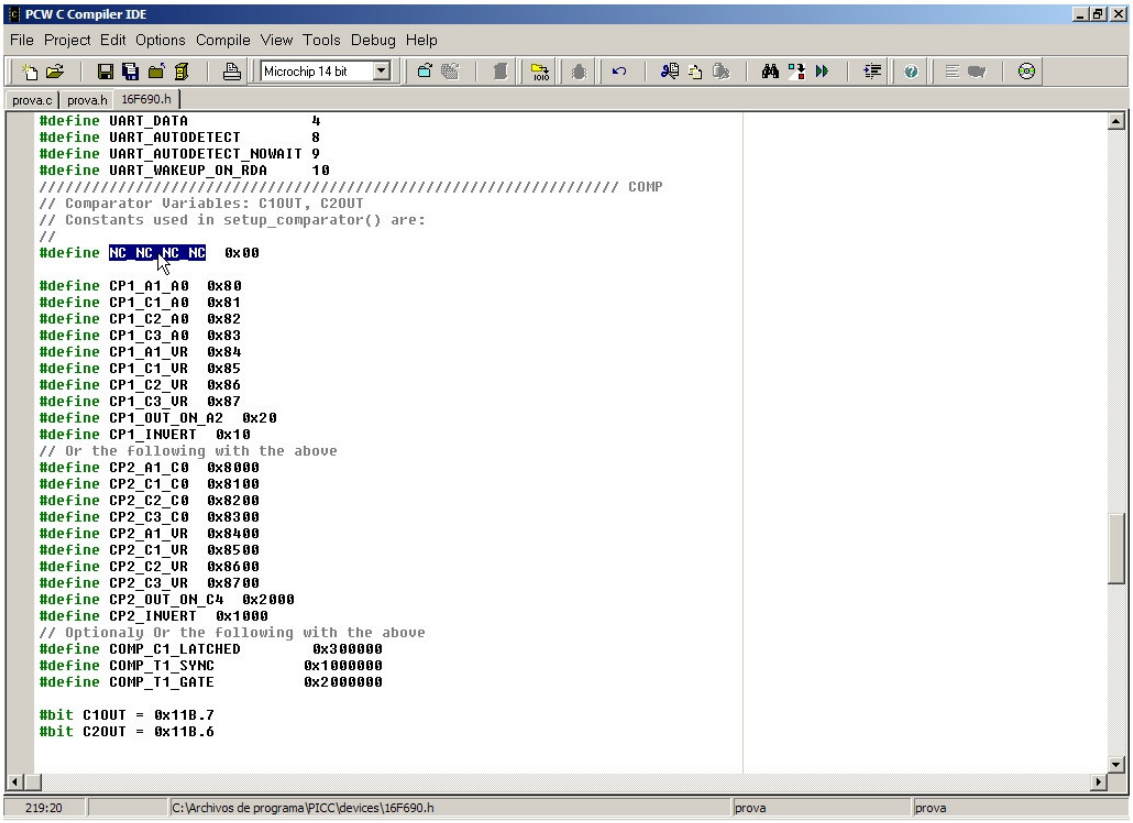




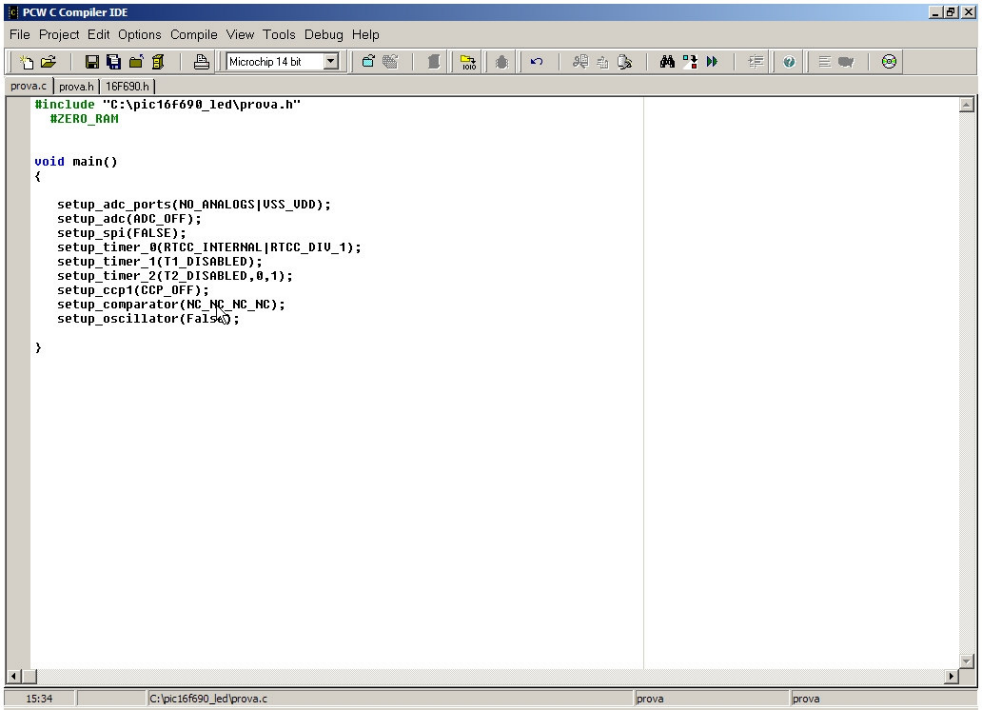
Una vez abierto 'prova.h', repetimos el proceso con '16f690.h', cuya inclusión se realiza desde prova.h.



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.

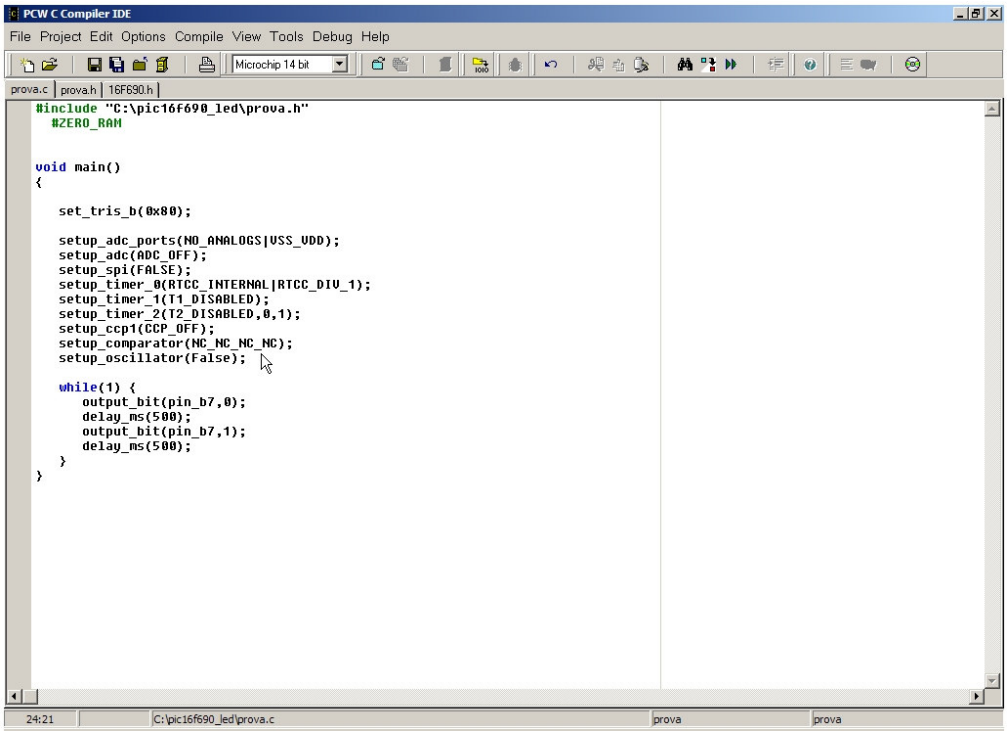


A destacar:

- 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:

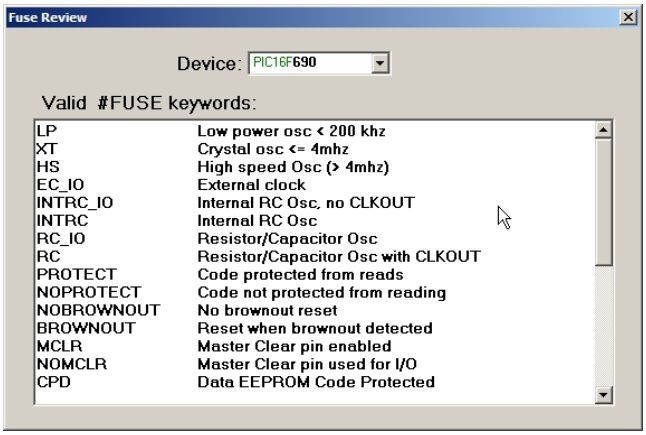


Como se puede ver, se trata de un bucle infinito (while(1)), en cuyo interior se activa y desactiva el bit 7 del puerto b (RB7), con retardo de 0,5 segundos. Así pues, si todo ha ido bien, veremos parpadear al led conectado a dicho pin.

---

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

View > Valid Fuses.



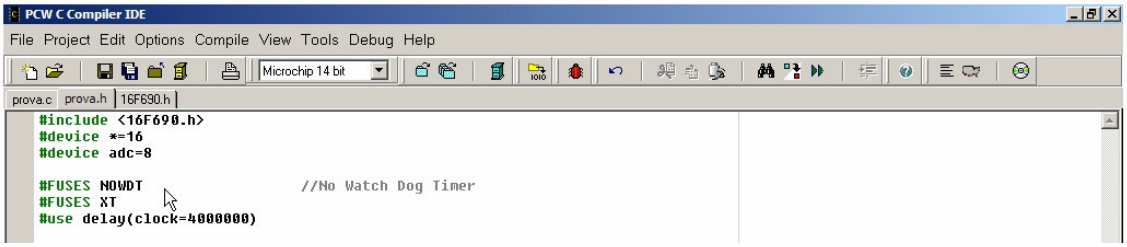
Añadiremos la directiva que indica que nuestro cristal funciona a una frecuencia inferior o igual a 4 MHz.

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 con los parámetros aquí escogidos.

---

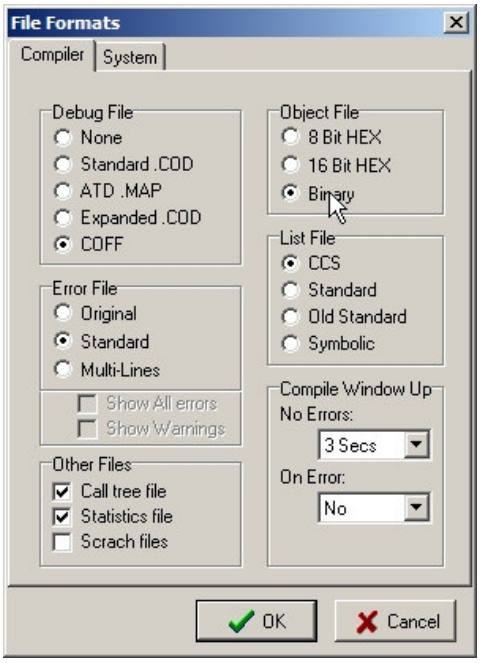
24. Modificamos prova.h añadiendo el parámetro XT a la directiva #FUSES:



---

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'.

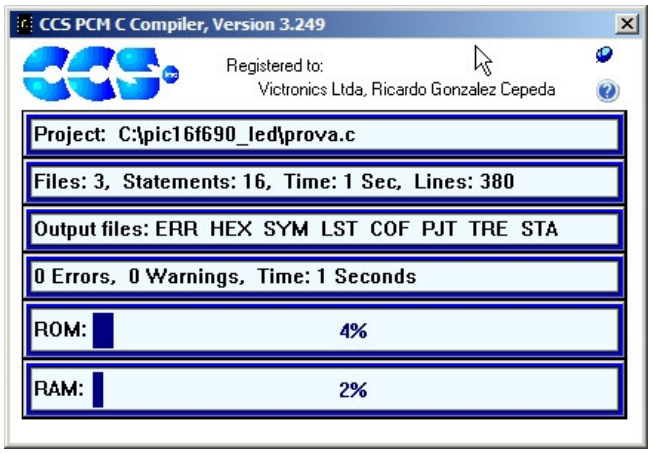


---

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:

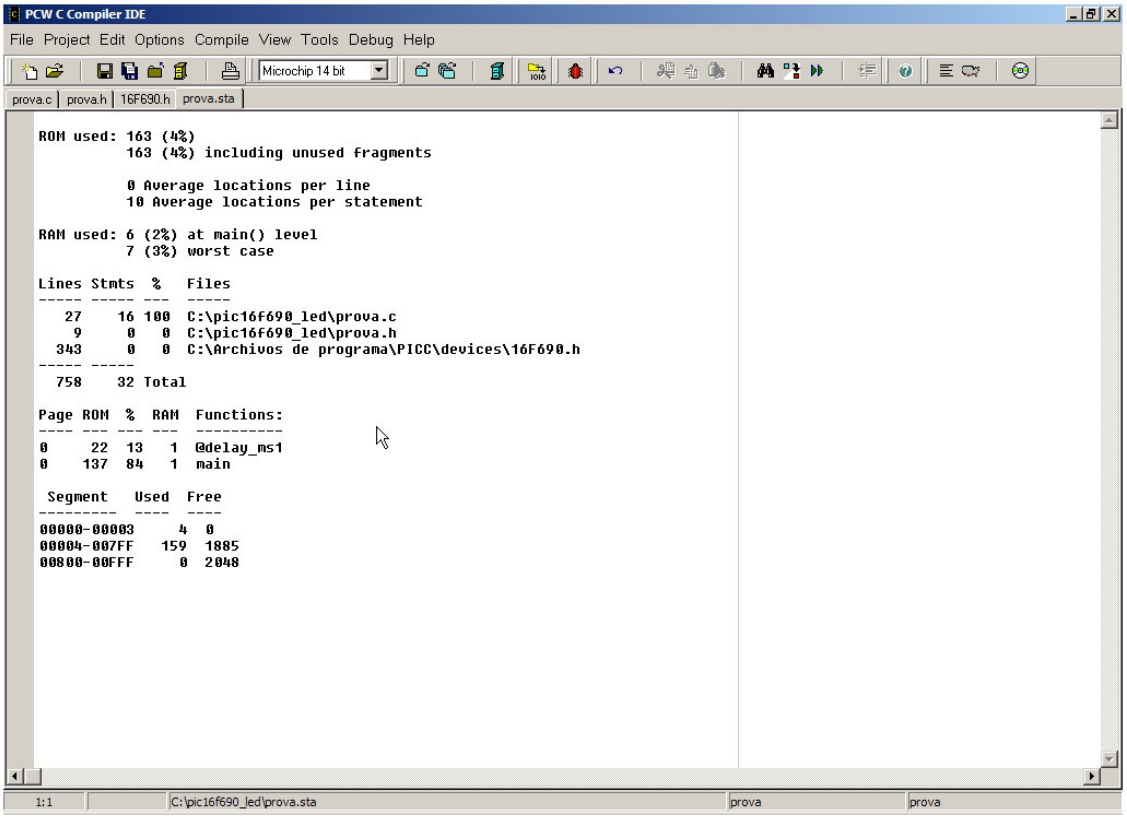


Obsérvese como una aplicación tan simple apenas consume memoria ROM (programa) y RAM (variables y registros de tiempo de ejecución).

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

---

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.



3.4. Instalación del adaptador USB-RS232

1. Antes de empezar, comentar que el adaptador USB-RS232 funciona a la perfección en equipos que no tengan ningún puerto serie.

Sin embargo, tras realizar algunas pruebas, se ha constatado que pueden surgir problemas en algunos equipos con puerto(s) serie incorporados. Aunque solamente tiene sentido emplear el adaptador en PCs con puerto serie incorporado si estos ya estan ocupados.

---

2. Al introducir el CD autoarrancable con los drivers se nos pedirá el idioma en el que queremos visualizar los textos del menú de instalación.

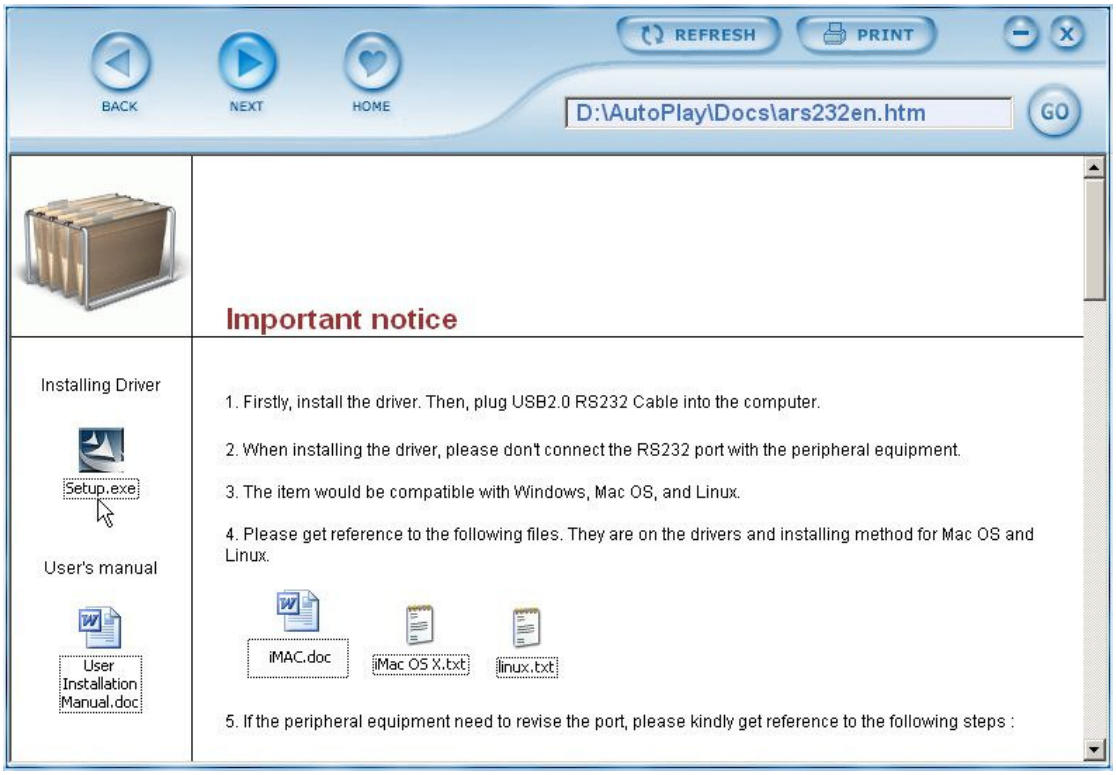
En el caso de no ejecutarse la aplicación autoarrancable, la podremos iniciar haciendo doble clic sobre el archivo 'Autorun.exe' de la raíz del CD.

---

3. Seleccionamos 'USB 1.1 TO RS232 Converter'.

---

4. A continuación se mostrará un fichero .html con la información necesaria para instalar y configurar los drivers del adaptador.

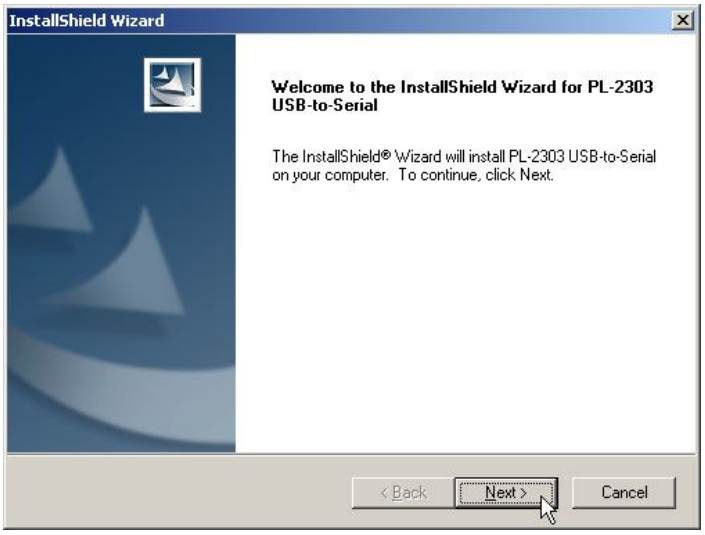


Es importante no conectar el adaptador al módulo programador mientras se instala el driver.

Ejecutamos el 'setup.exe'.

---

5. Aparecerá el asistente de instalación. Click en 'siguiente'.



---

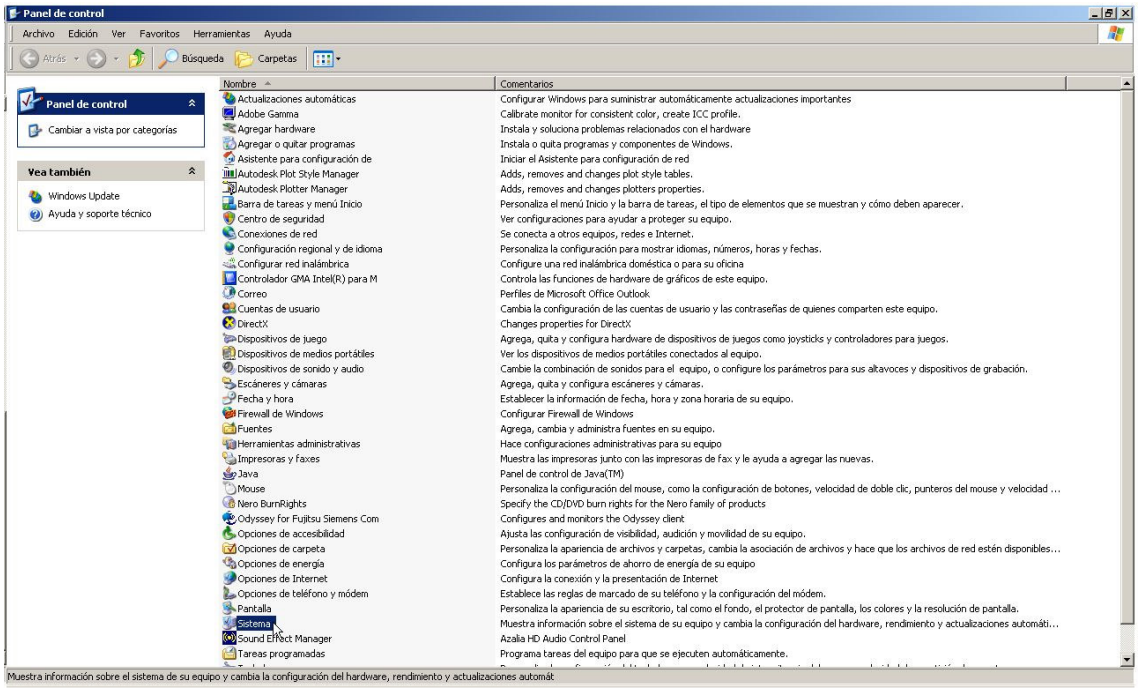
6. Esperamos a que finalice el proceso. Click en 'Finish'.

---

7. Reiniciamos el equipo.

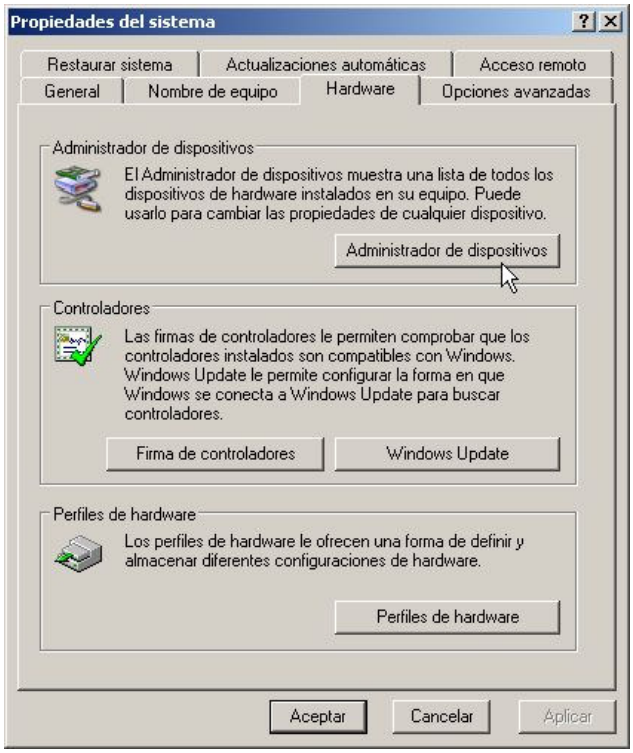
---

8. Accedemos al panel de control y entramos en el apartado ‘Sistema’.



9. Seleccionamos la pestaña ‘Hardware’.

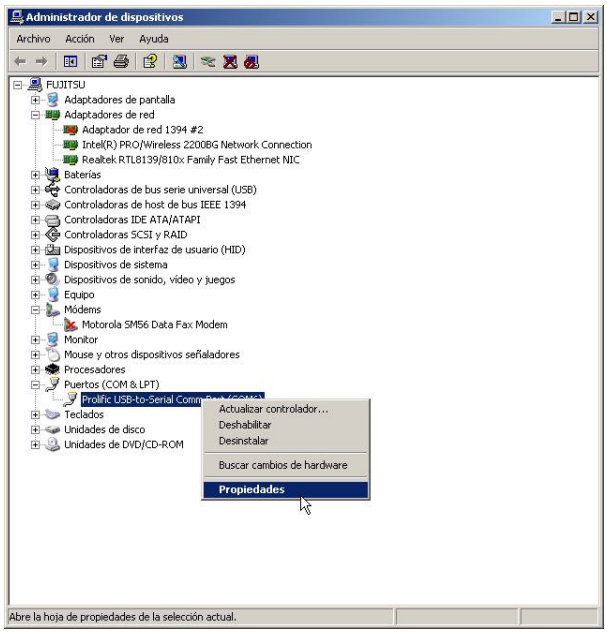
10. Click en ‘Administrador de dispositivos’.



---

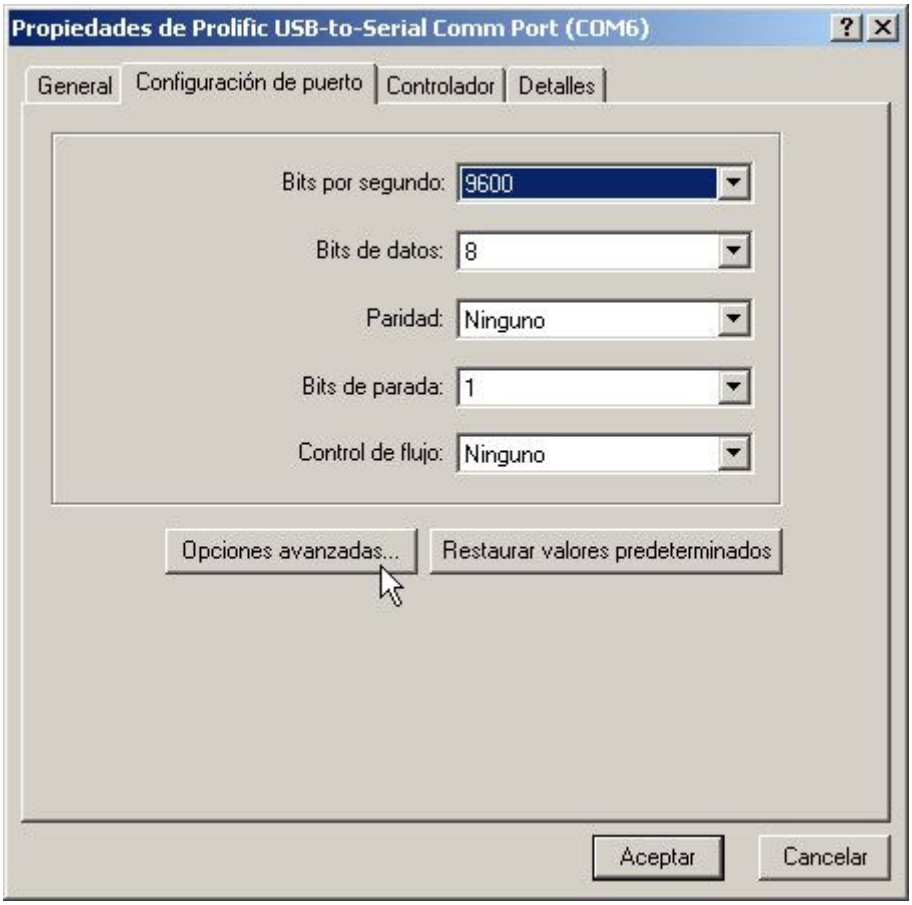
11. Dentro del apartado ‘Puertos (COM & LPT)’ podremos observar el nuevo puerto que tenemos instalado (deberá estar conectado).

Hacemos click con el botón derecho y vamos a ‘Propiedades’.



---

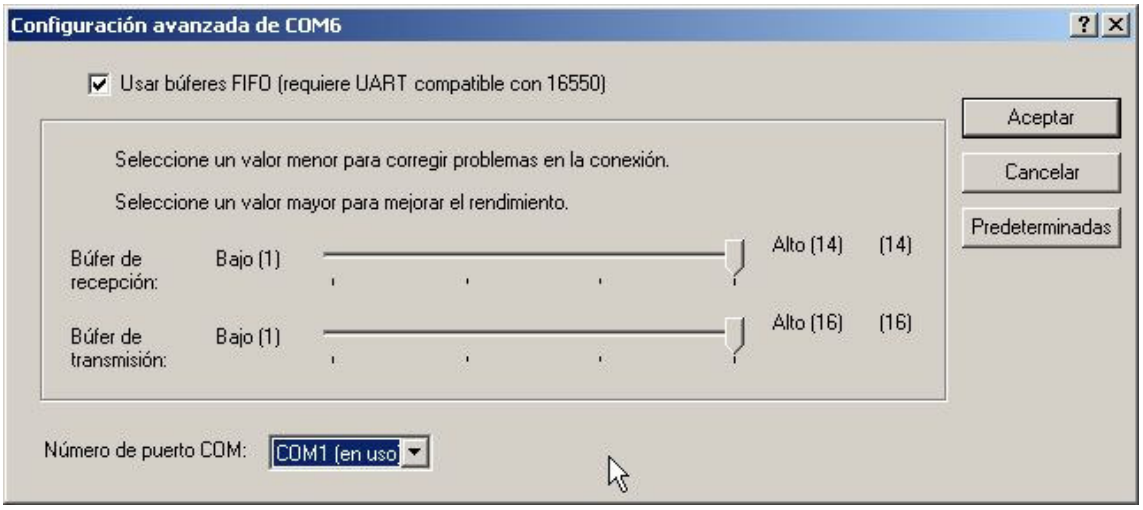
12. Click en la pestaña 'Configuración de puerto'.



Y accedemos a 'Opciones avanzadas'.

---

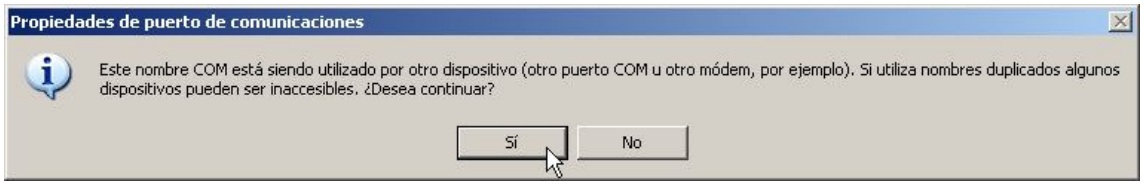
13. Seleccionamos el número de puerto COM que vayamos a utilizar. En nuestro caso: COM1.



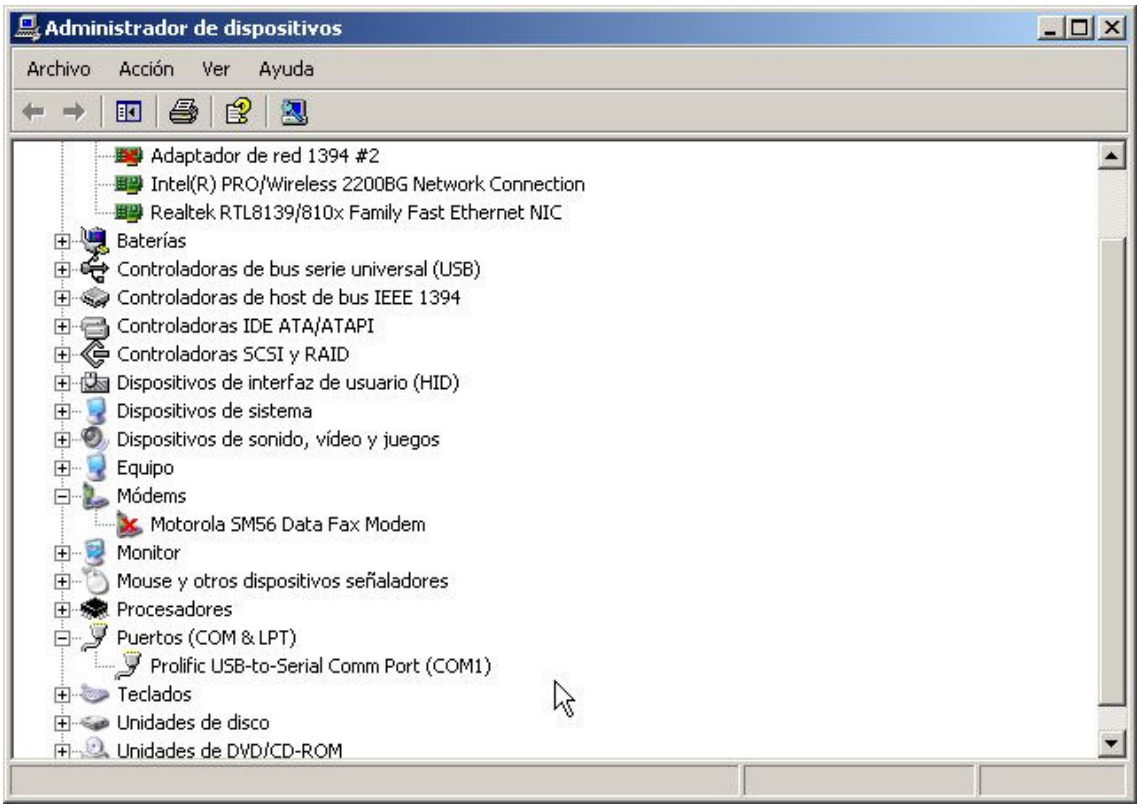
Click en 'Aceptar'.

---

14. Si aparece el siguiente mensaje, nos cercioraremos de no tener ningún otro dispositivo empleando el mismo recurso COM.



15. Finalmente comprobamos en el 'Administrador de dispositivos' que el puerto USB-COM se ha actualizado al número seleccionado.

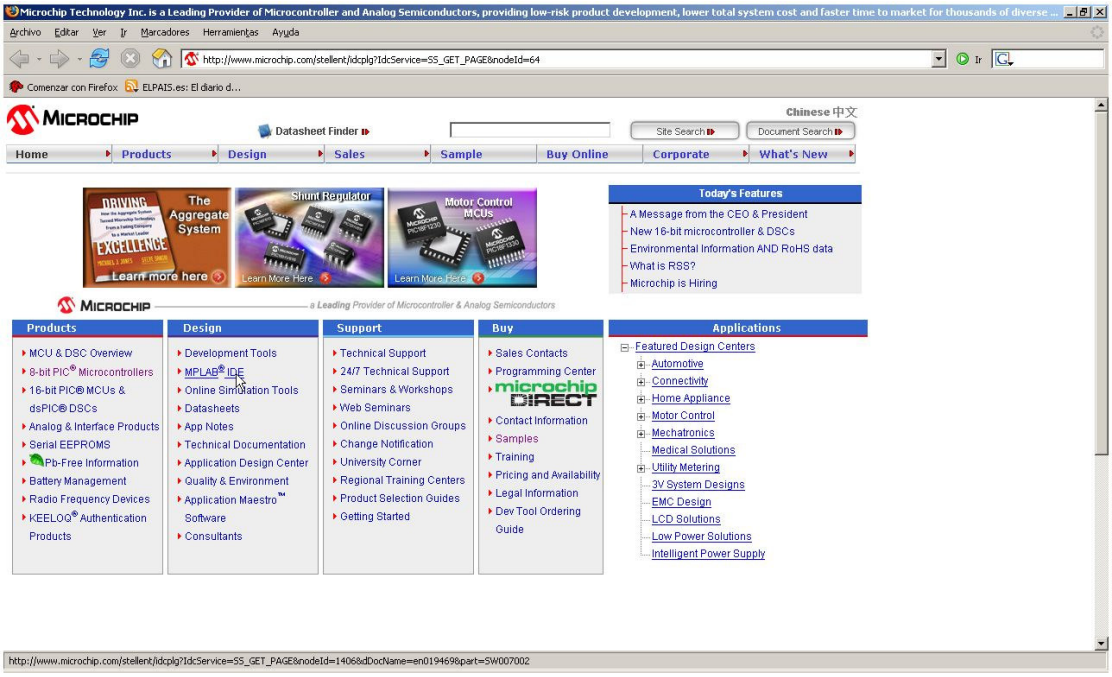


3.5. Instalación de MPLAB IDE 7.50

1. Vamos a descargar el entorno de desarrollo gratuito de MICROCHIP, el MPLAB IDE 7.50, de su página web, cuya dirección es: www.microchip.com

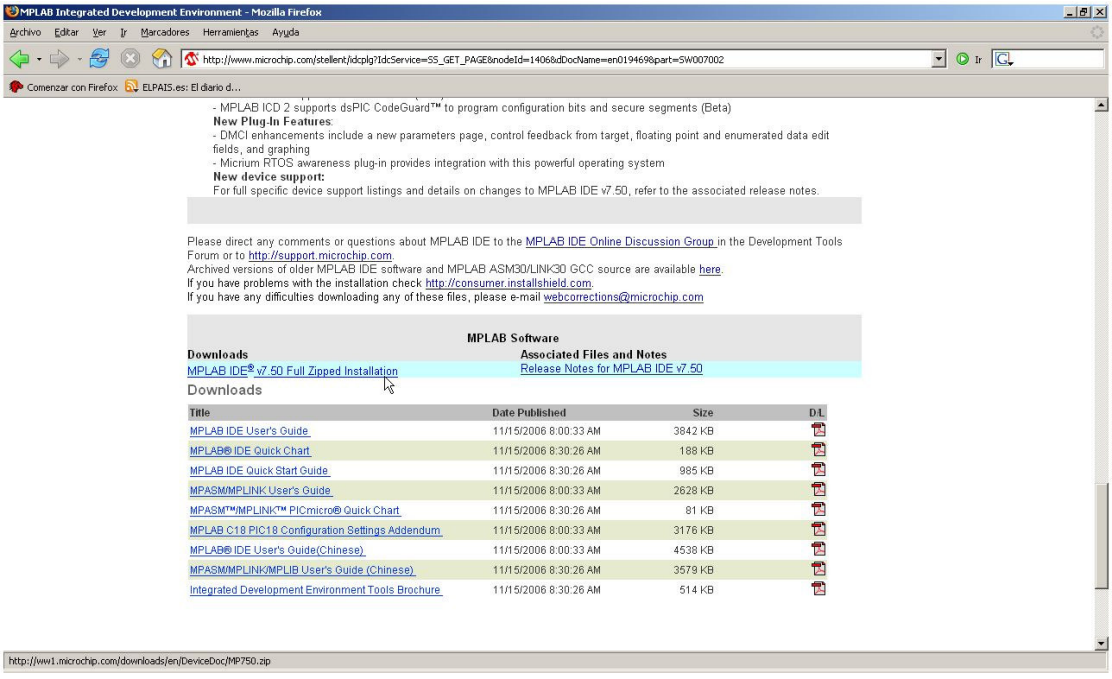
---

2. Clicamos en MPLAB IDE.



---

3. En la ventana informativa sobre este producto, nos desplazaremos hasta encontrar el link que permite descargar el instalable.

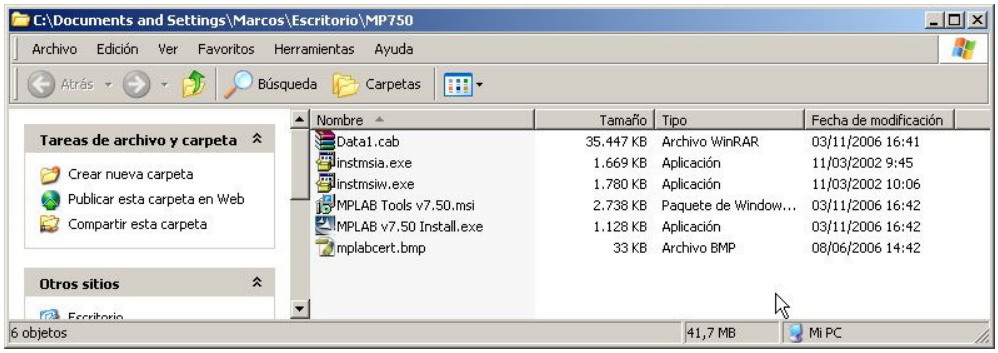


---

4. Esperamos a que finalice la descarga.

---

5. Descomprimimos el archivo y ejecutamos el archivo 'MPLAB v7.50 Install.exe'.



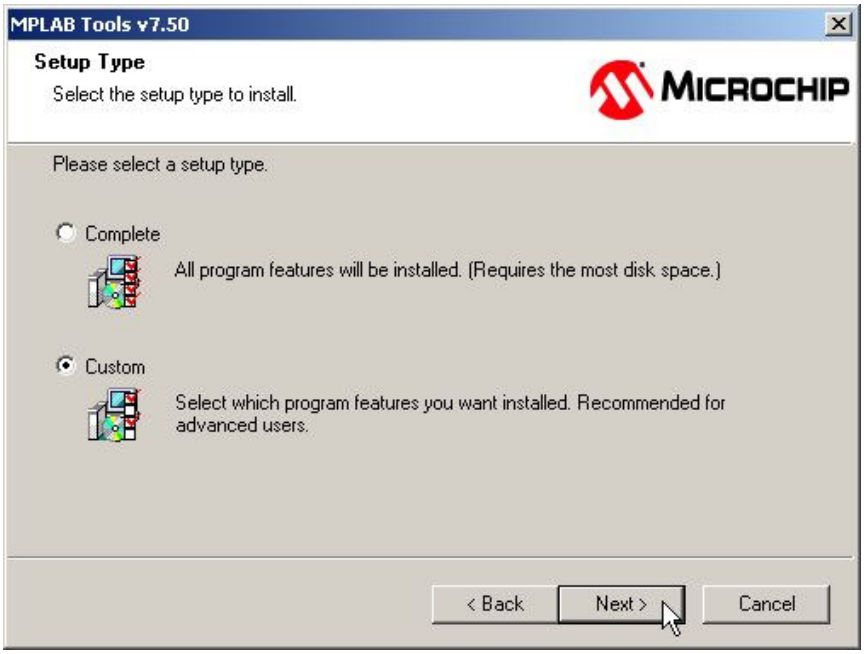
---

6. Es importante recordar que no debe haber ninguna aplicación ejecutándose mientras instalamos el software.

Click en 'Next' > Click en 'Next'

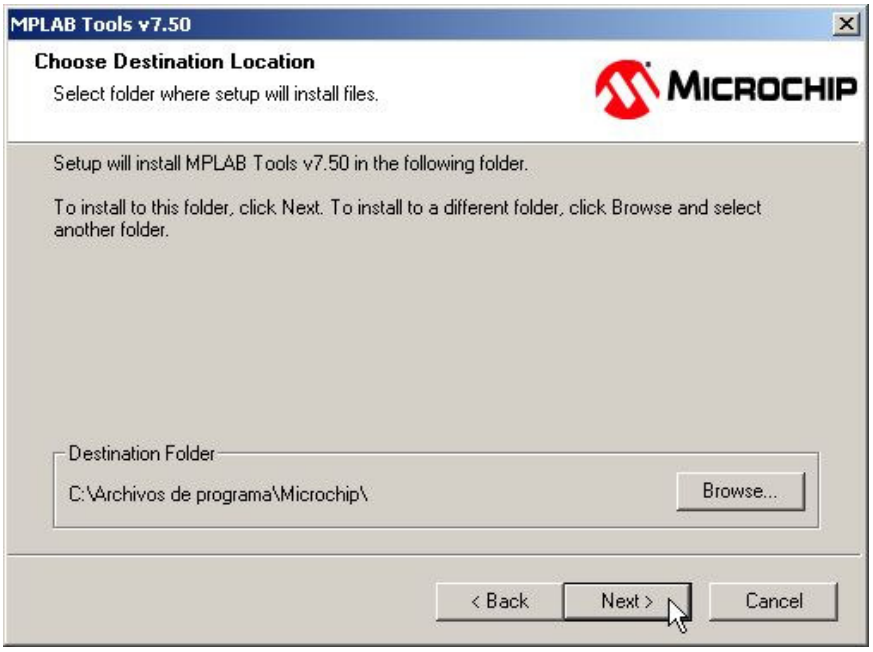
---

7. Seleccionamos la 'Custom' como tipo de instalación y continuamos.



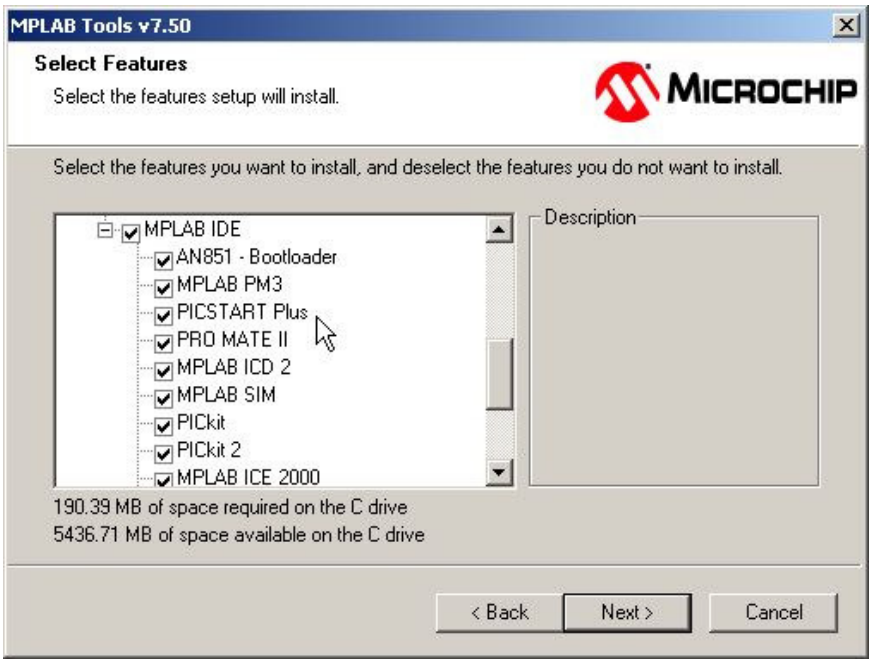
---

8. Introducimos la ruta de destino. Click en 'Next'.



---

9. A continuación se nos pedirá qué programadores queremos que MPLAB tenga configurados. Nos aseguraremos de que PICSTART Plus esté seleccionado.



Click en 'Next'.

---

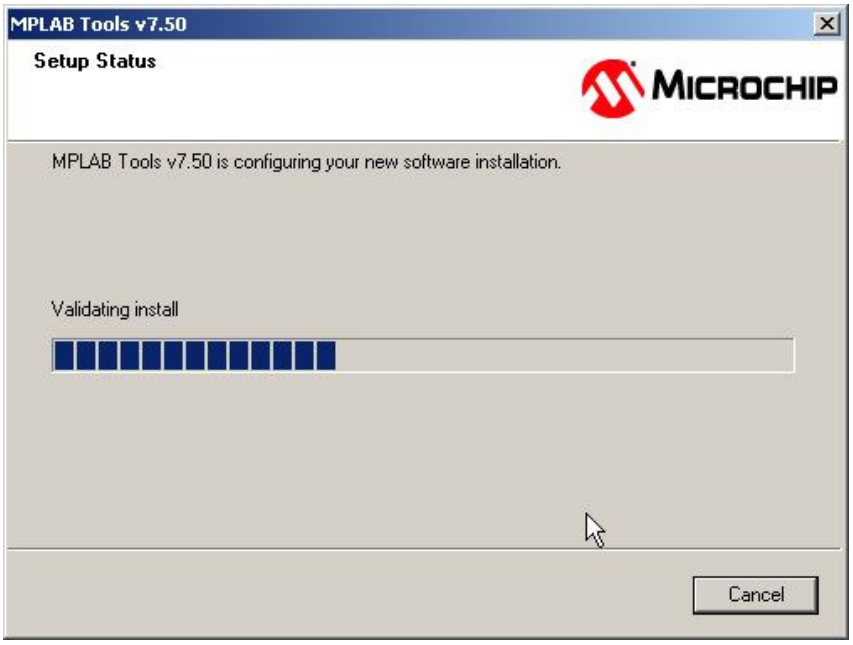
10. Acetamos el acuerdo de licencia y continuamos.

---

11. Click en 'Next'.

---

12. Esperamos a que finalice la instalación.



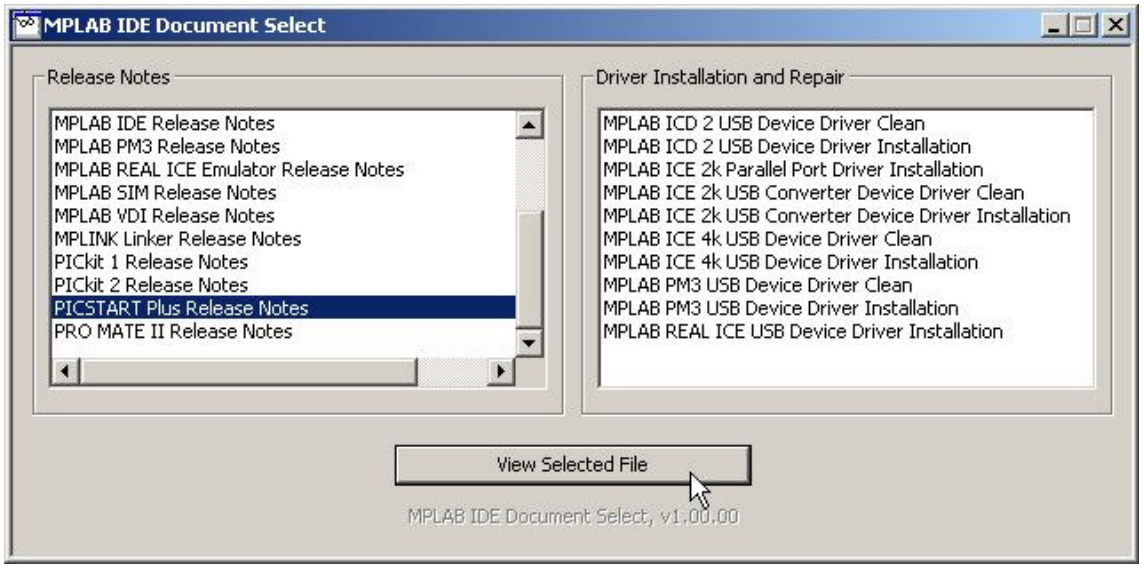
---

13. Click en 'Finish'.

---

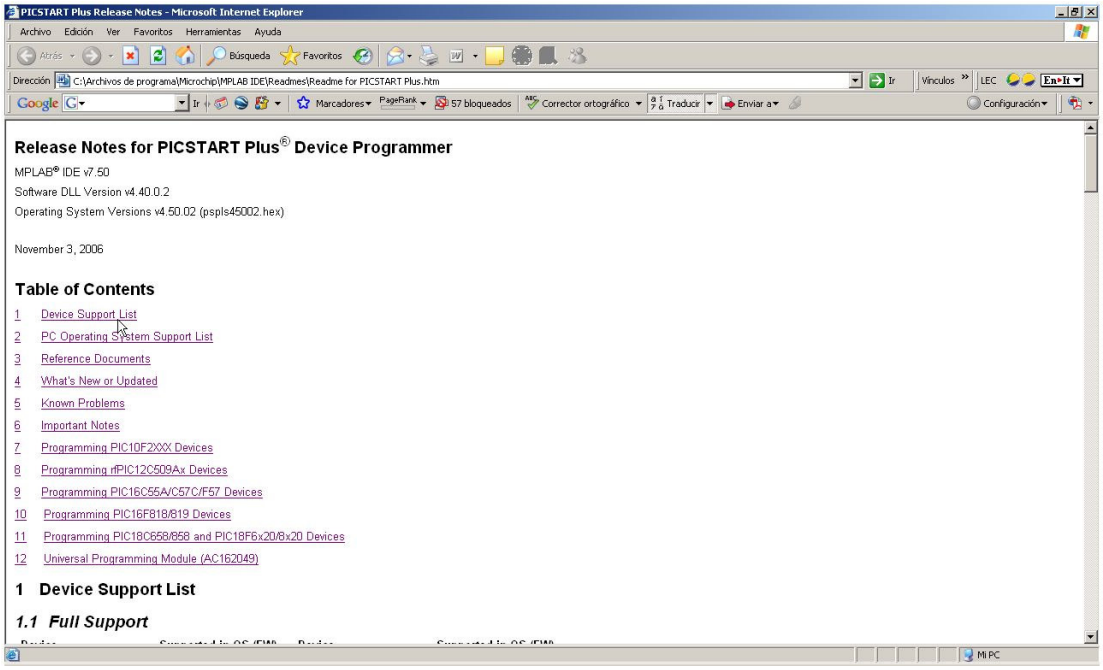
14. Al finalizar se nos ofrece visualizar de entre un conjunto de documentos relativos a aspectos concretos o programadores soportados aquellos que seleccionemos.

Seleccionamos el archivo 'PICSTART Plus Release Notes' y clicamos en 'View Selected File'.



---

15. El archivo que se nos mostrará es un documento en html como el siguiente:

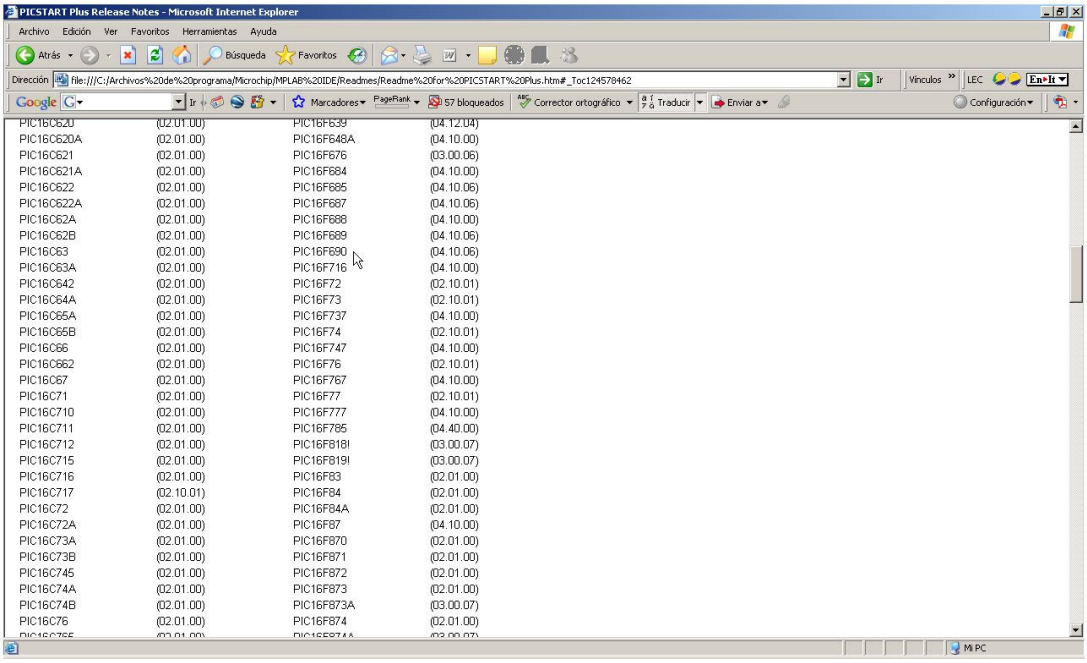


En él se recogen diversos aspectos de interés como el listado de microcontroladores soportados para cada versión (actualización) de firmware, notas sobre el uso de PICSTART PLUS con ciertas familias que puedan tener consideraciones especiales, etc.

Así pues, clicamos en 'Device Support List'.

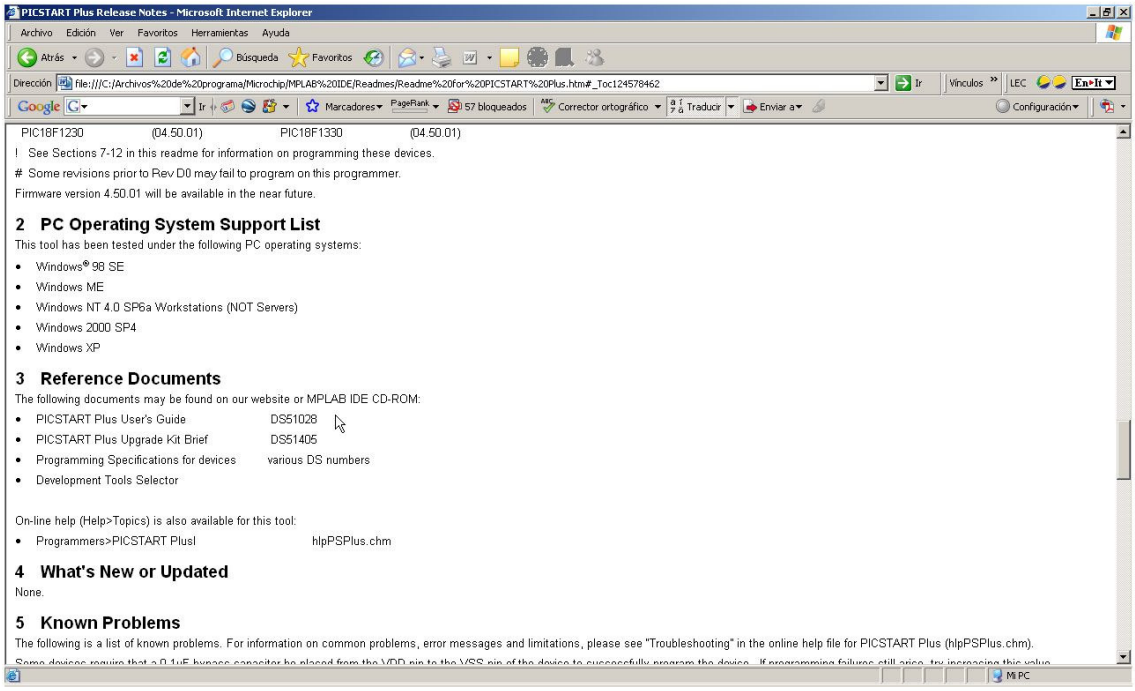
---

16. Como se puede ver, nuestro dispositivo (PIC16F690) está en la lista, y la versión requerida del firmware es la 4.10.06.

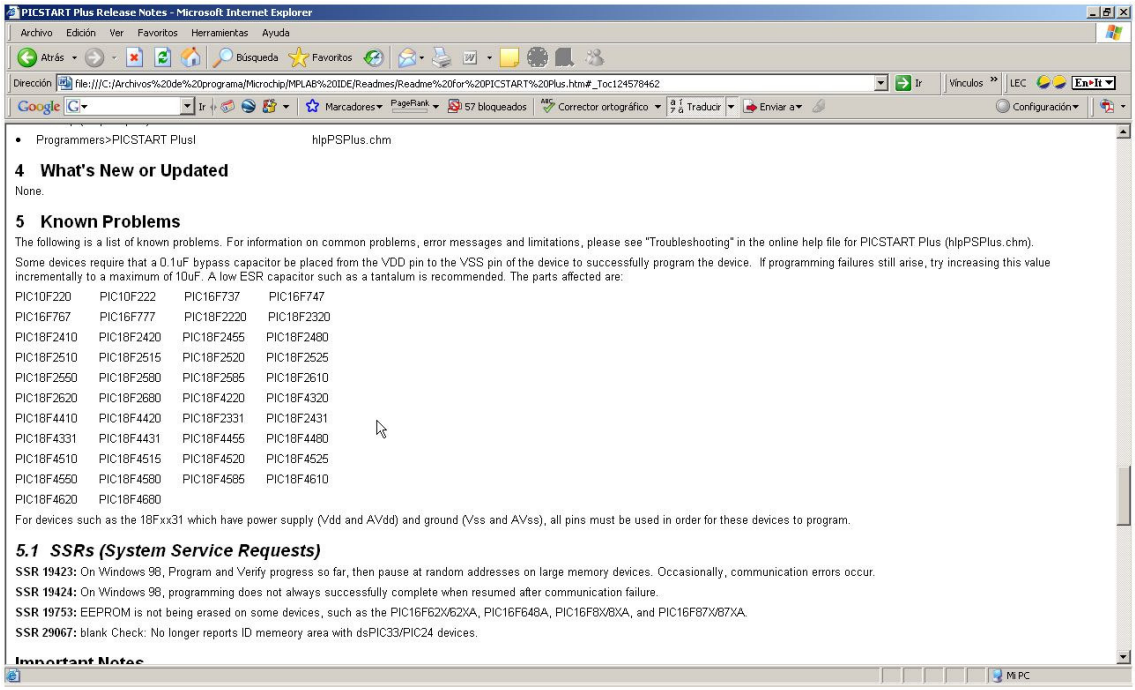


---

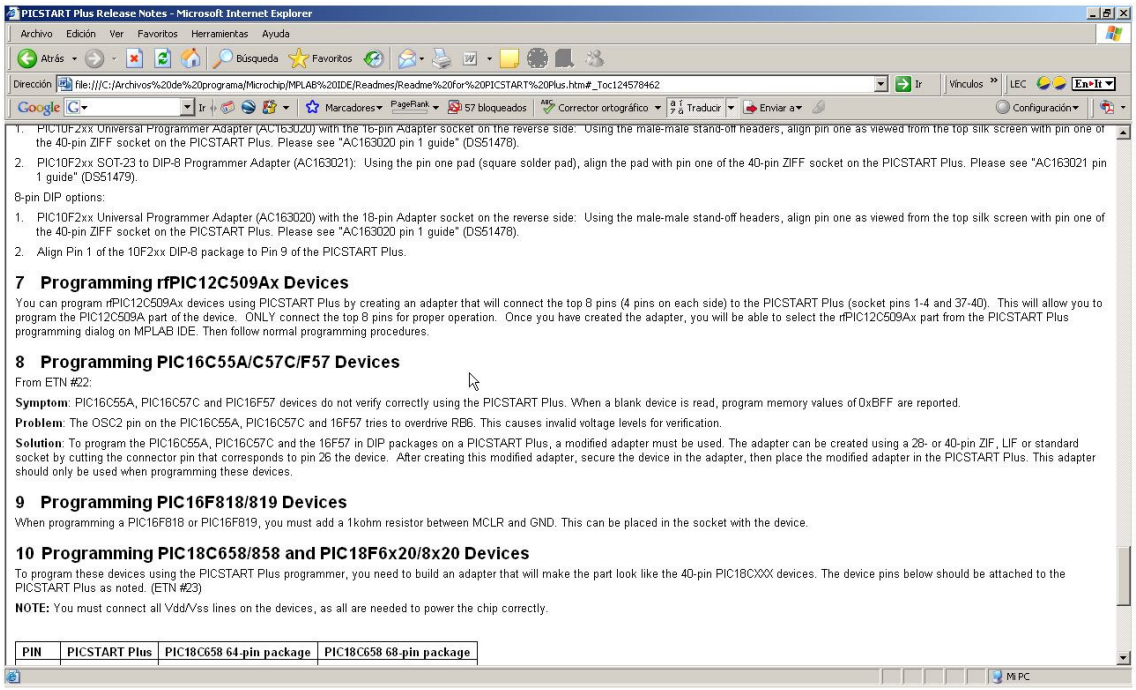
17. Como ya se ha comentado anteriormente, el documento recoge otros aspectos relevantes. El apartado 3 nos indica que la guía del usuario del PICSTART PLUS y las notas para actualizar el kit (Upgrade Kit Brief) se encuentran disponibles en la web de MICROCHIP.



18. El apartado 5 muestra un listado con dispositivos que tienen problemas de programación.



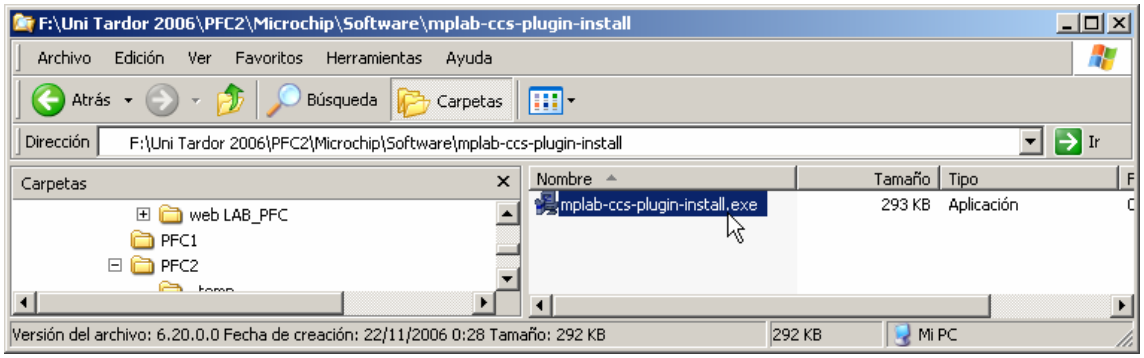
19. Otros apartados como los mostrados en la figura inferior dan a conocer aspectos (en general problemas y sus soluciones) para algunos micros o familias de micros, en relación al PICSTART PLUS.



3.6. Instalación del plugin del CCS PCWH para MPLAB

1. La instalación del plugin resulta imprescindible si queremos emplear CCS desde MPLAB, por ejemplo para compilar directamente desde el IDE.

Ejecutamos el archivo 'mplab-ccs-plugin-install.exe'.

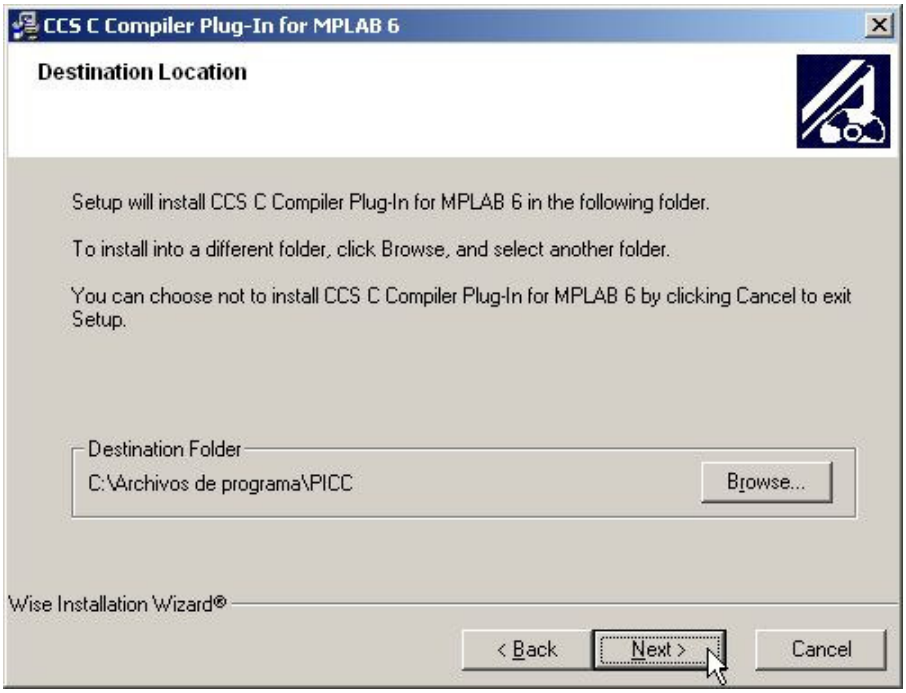


---

2. Al iniciarse el asistente clicaremos en 'Next'.

---

3. Seleccionamos la ruta de destino (que por defecto será la misma que la del CCS PCWH).



---

4. Click en 'Next' > Click en 'Next'.

---

5. Esperamos a que finalice la instalación. Click en 'Finish'.

3.7. Programación del microcontrolador con MPLAB IDE 7.50

1. Iniciamos el entorno MPLAB. Para ello:

Inicio > Programas > Microchip > MPLAB IDE v7.50 > MPLAB IDE

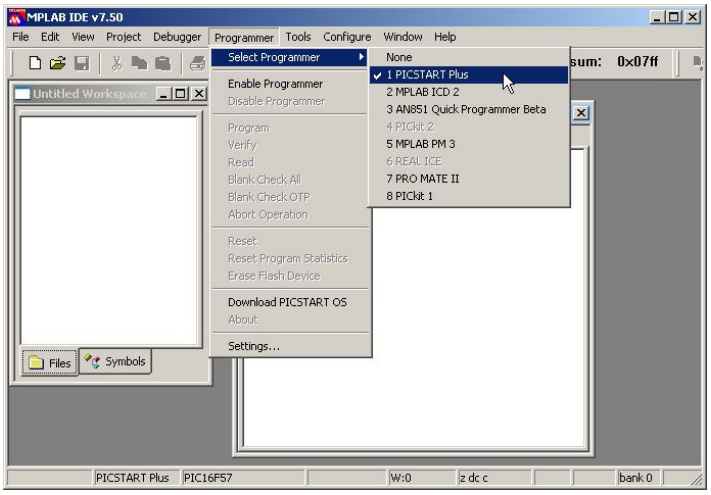
---

2. Conectamos el programador a la alimentación (habiéndonos asegurado de que no hay ningún micro insertado en el zócalo) y al puerto serie o al USB, bien empleando el cable RS232 habitual, bien empleando el conversor USB-RS232. El LED verde (Power) deberá encenderse.

Si faltase algún componente será preciso avisar a alguno de los responsables de laboratorio.

---

3. Una vez dentro del entorno, seleccionamos el PICSTART Plus de entre todos los posibles.



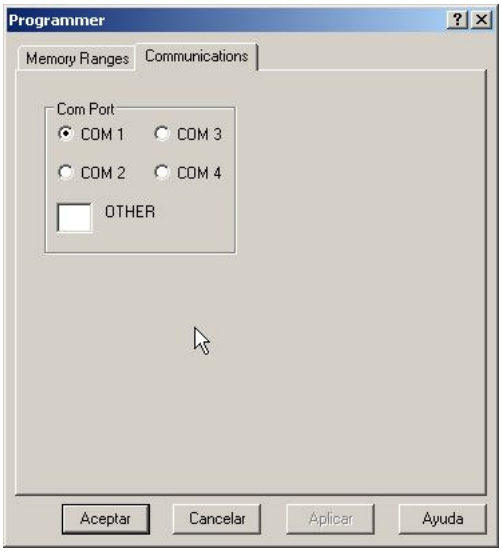
---

4. Ahora pasamos a configurar el dispositivo.

Programmer > Settings…

---

5. En la pestaña 'Communications' seleccionaremos el número de puerto serie en el que tenemos conectado el PICSTART PLUS. En nuestro caso tenemos el adaptador USB-RS232 configurado en el COM1.

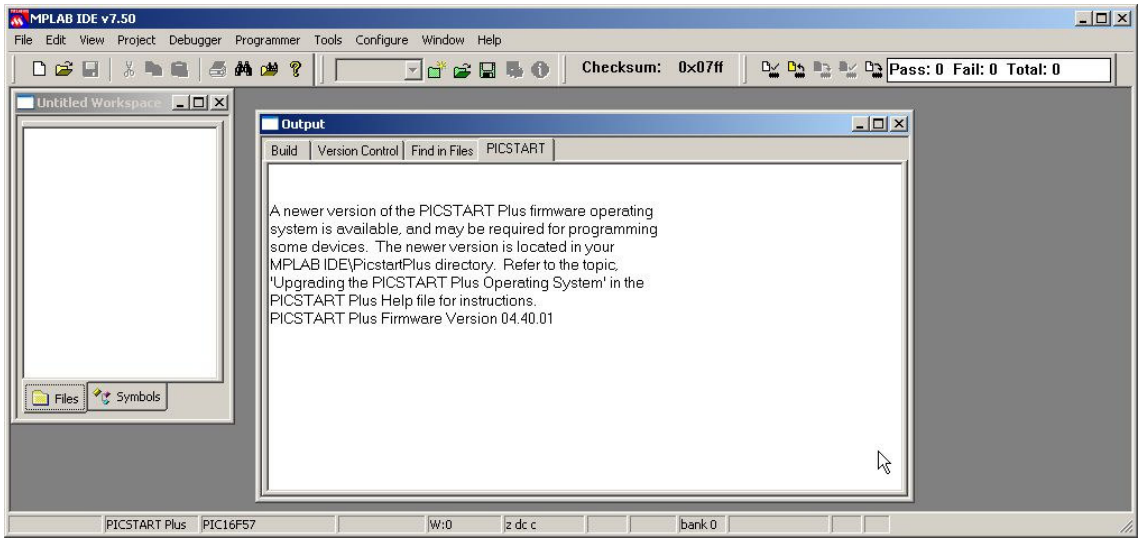


Aceptamos y volvemos al entorno.

---

6. Habilitamos el programador:

Programmer > Enable Programmer

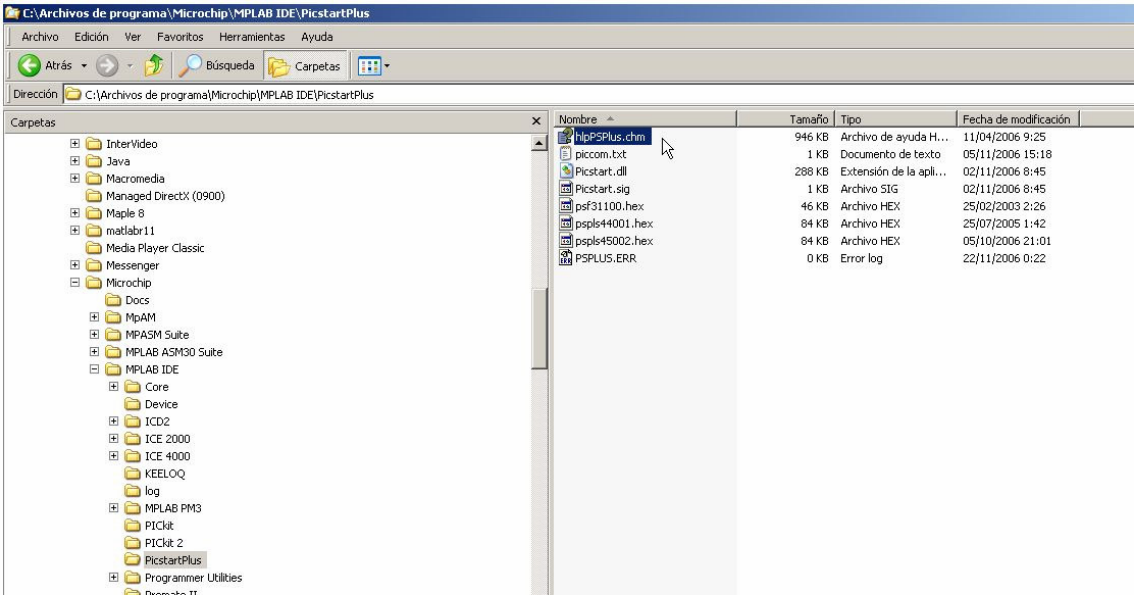


En el momento de editar este manual la versión del firmware residente en el módulo programador es anterior a la soportada por la versión 7.50 de MPLAB.

Procederemos a actualizar el firmware.

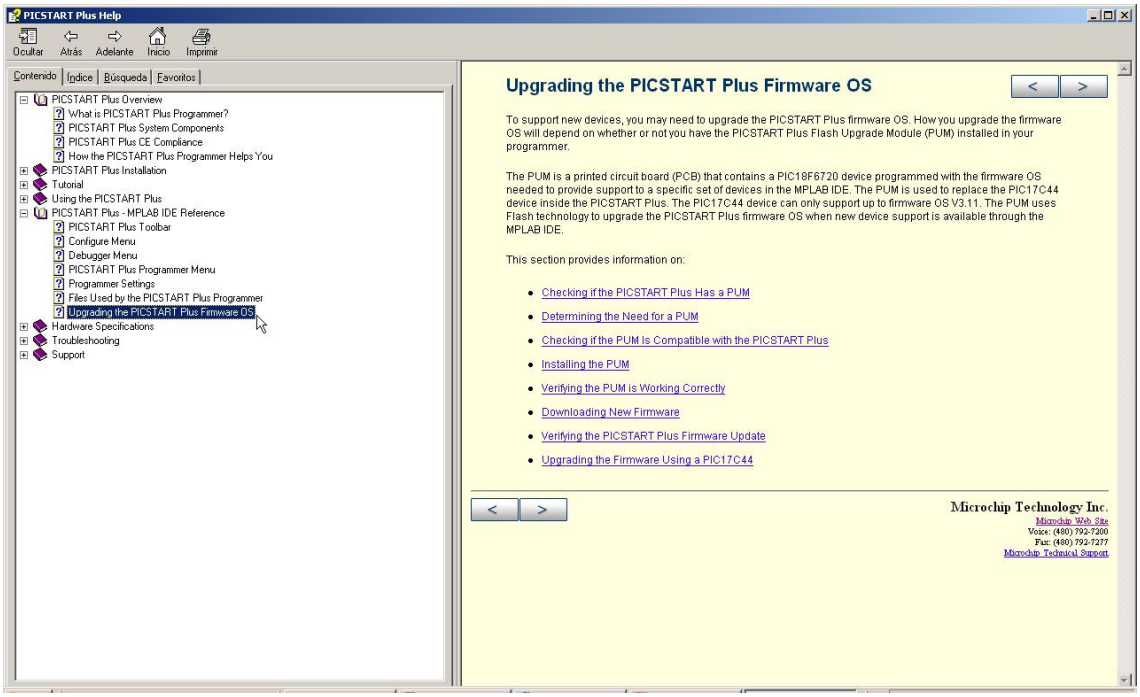
---

7. Tal y como se nos explica en la ventana 'Output', la actualización se encuentra en el subdirectorio 'PicstartPlus' del directorio 'MPLAB IDE'. Se trata de un archivo hexadecimal que será volcado sobre el micro que incorpora el programador (en realidad es un PIC). No obstante, antes de proceder consultaremos el archivo de ayuda que se encuentra en el mismo directorio.



---

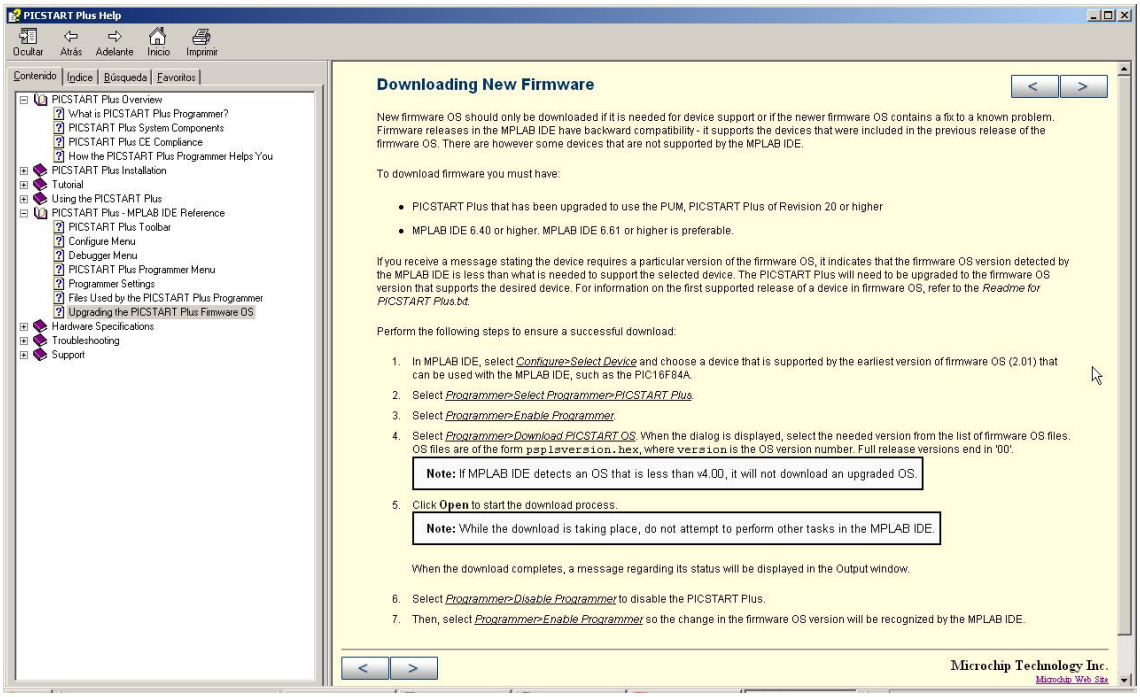
8. Dentro del archivo de ayuda accedemos al apartado 'Upgrading the PICSTART Plus Firmware OS'.



Aquí obtendremos toda la información necesaria acerca de si es necesario o no actualizar el firmware, como volcarlo, etc.

---

9. Clicamos en 'Downloading the firmware'. Aquí se nos muestran los pasos necesarios para realizar el volcado.



Consúltese en caso de necesidad.

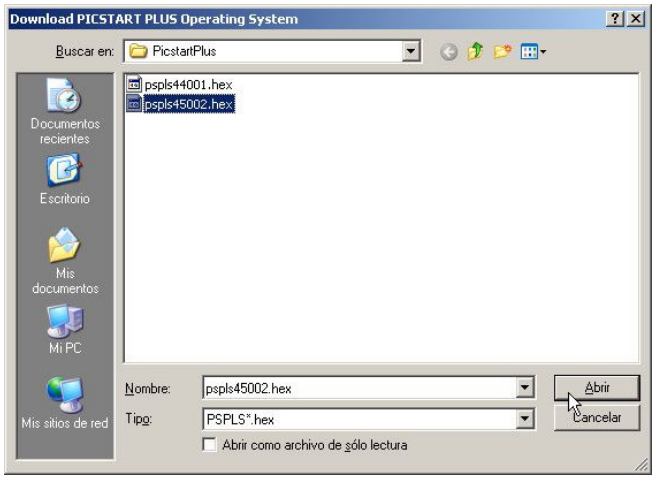
---

10. Tal y como se nos explica en el archivo de ayuda, procedemos a descargar el firmware. Para ello hacemos, en el MPLAB IDE:

Programmer > Download PICSTART OS

---

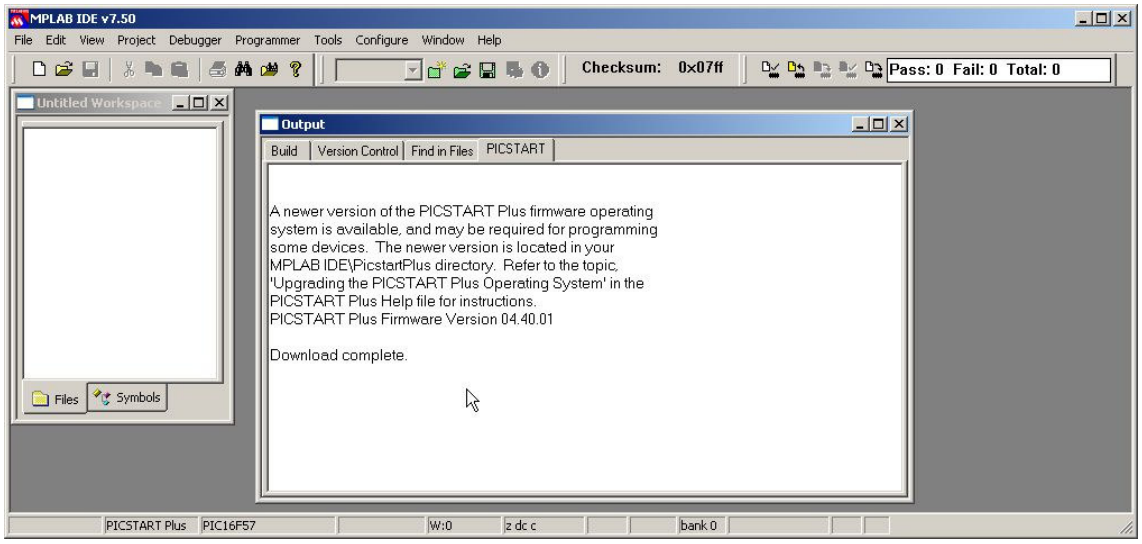
11. Se nos preguntará por el archivo .hex a descargar. Deberemos seleccionar aquél de mayor versión ubicado en la carpeta 'PicstartPlus'.



Como se puede observar, en nuestro caso es el archivo 'pspls45002.hex' (correspondiente a la versión 4.50.02).

---

12. Si todo ha ido bien se nos mostrará el siguiente mensaje:



---

13. Una vez preparado el entorno, ya podemos crear el proyecto que nos servirá de base para acceder al proyecto que hicimos en CCS, pero desde MPLAB.

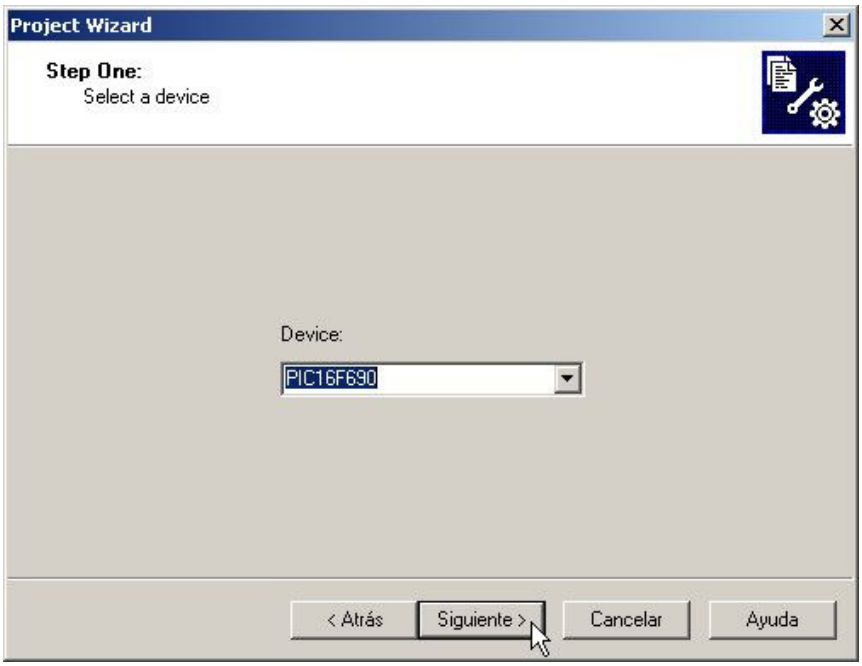
Accedemos a Project > Project Wizard…

---

14. Click en 'Next'.

---

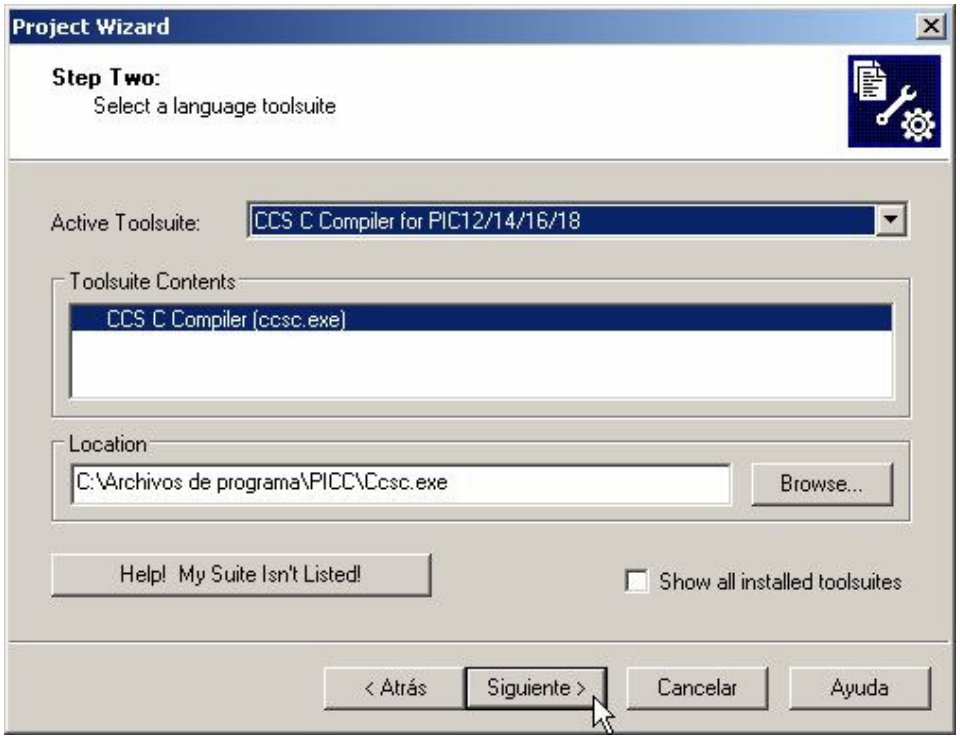
15. Seleccionamos el PIC a utilizar. En nuestro caso el PIC16F690.



Click en 'Siguiente'.

---

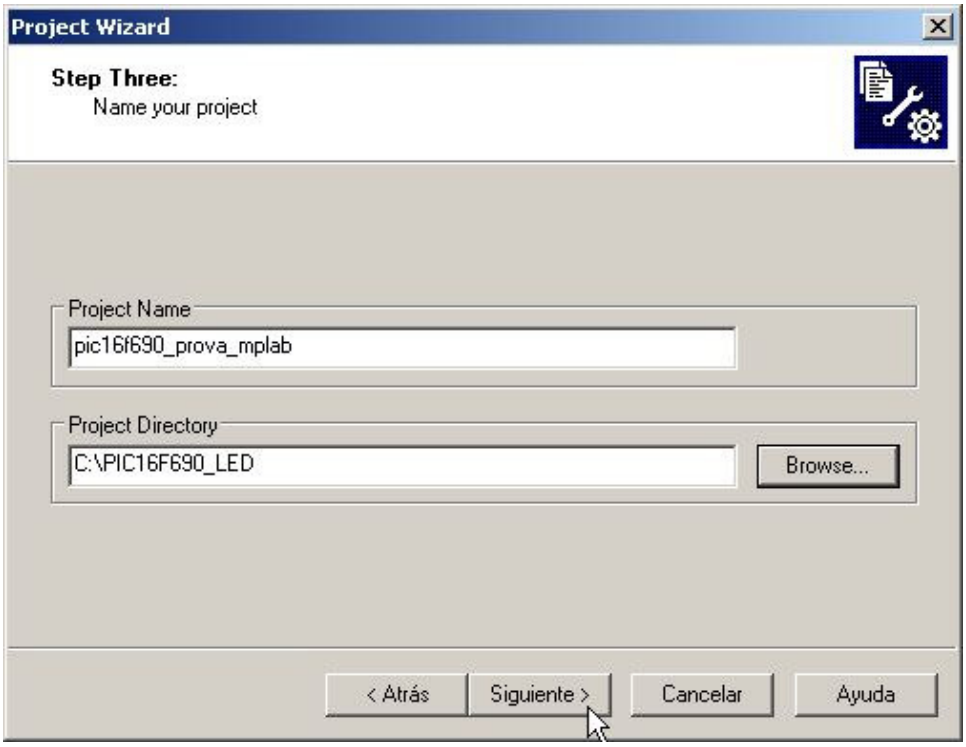
16. Ahora se nos pedirá que seleccionemos el compilador a emplear. Si hemos instalado correctamente el plugin, el CCS PCWH aparecerá por defecto.



Click en 'Siguiente'.

---

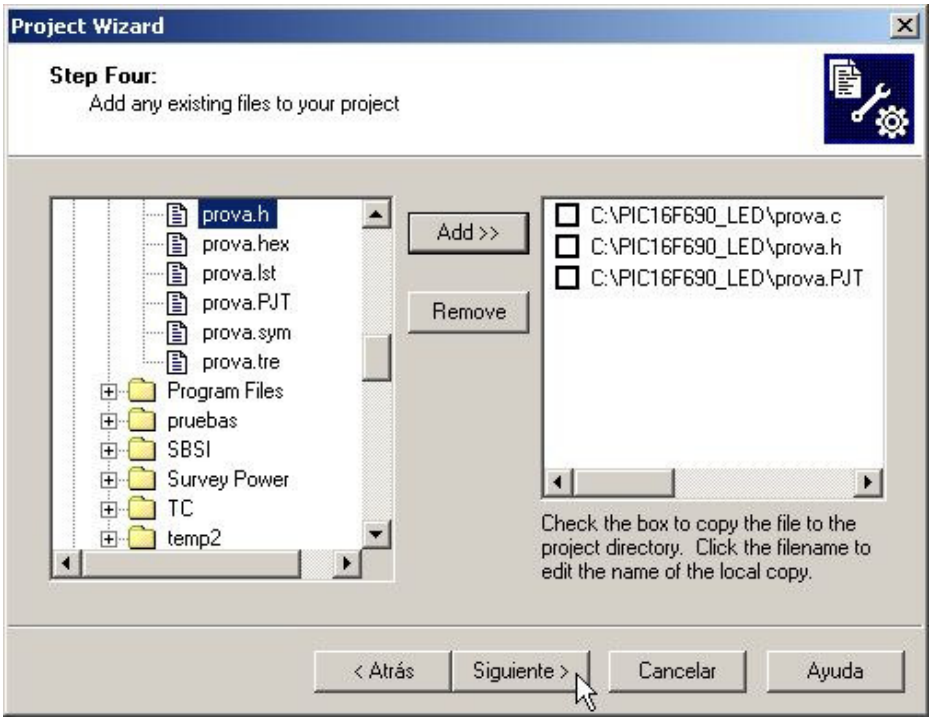
17. Asignamos un nombre a nuestro proyecto y su ubicación.



Click en 'Siguiente'.

---

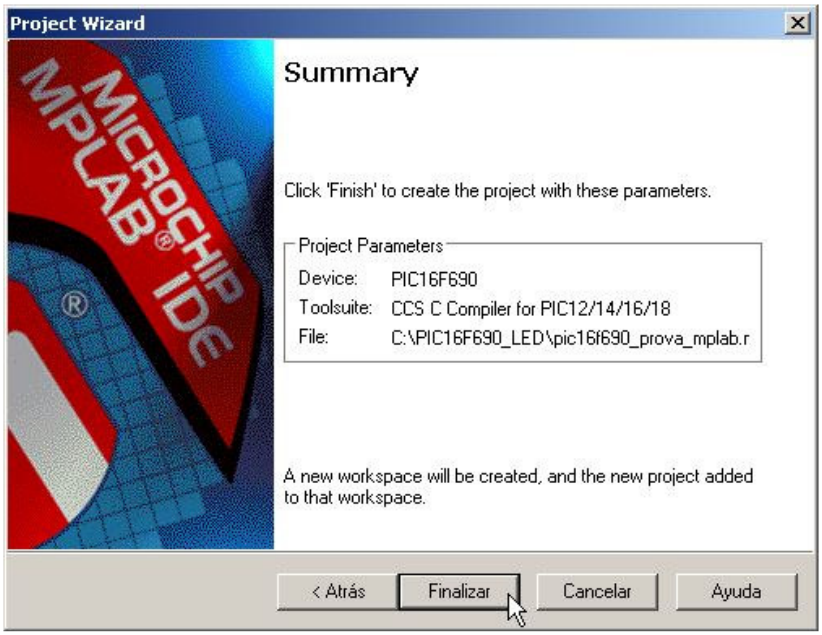
18. Ahora deberemos seleccionar los archivo que componen el proyecto que ya hemos hecho en CCS PCWH. Estos son 'prova.c' (archivo C principal), su cabecera 'prova.h' y el archivo de proyecto CCS 'prova.pjt'.



Los archivos seleccionados mediante el botón 'Add >>' los visualizaremos en la lista de la derecha.

---

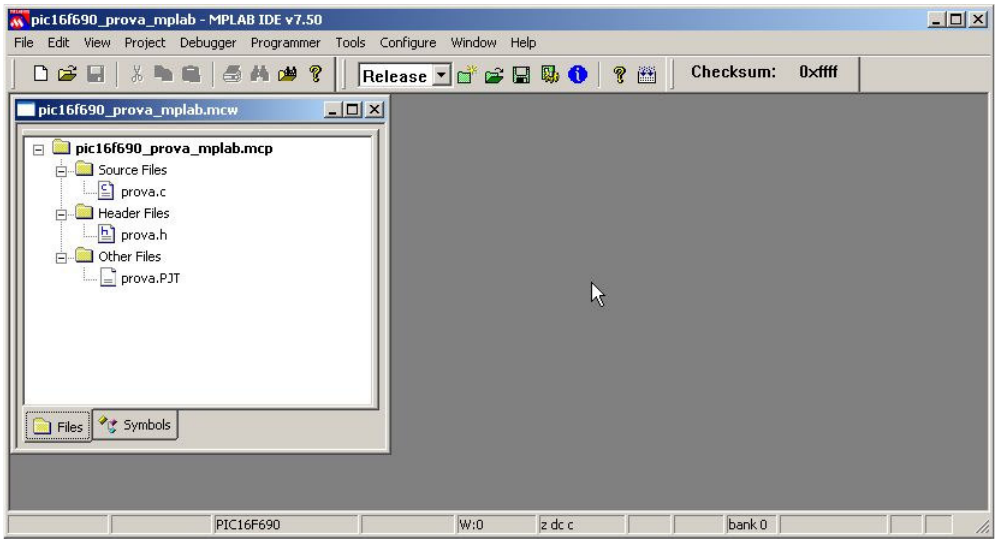
19. Ya hemos terminado con el asistente.



Click en 'Finalizar'.

---

20. Ya tenemos nuestro proyecto listo para compilar y volcar. Como se puede ver en la figura inferior, los archivos que hemos seleccionado han sido añadidos a nuestro proyecto MPLAB.



---

21. Volvemos a abrir la ventana de mensajes 'Output':

View > Output

---

22. De nuevo seleccionamos el PICSTART PLUS.

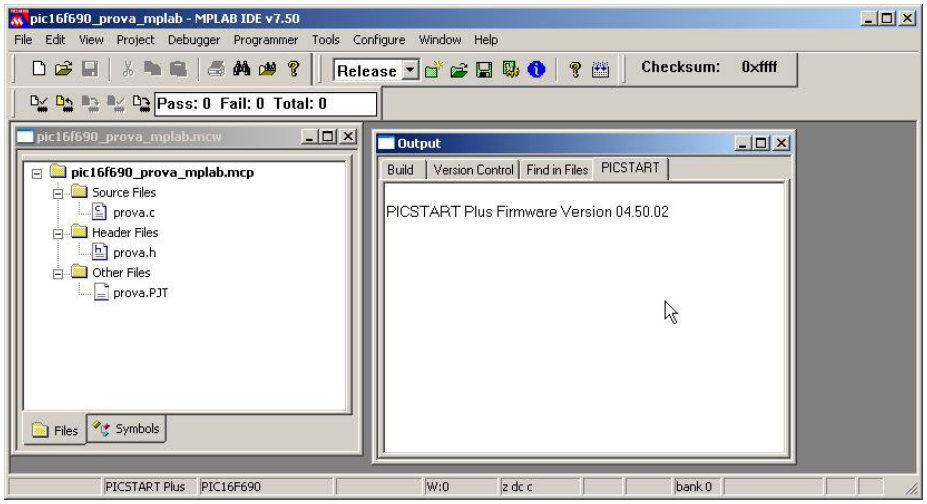
Programmer > Select Programmer > 1 PICSTART Plus

---

23. Lo habilitamos de nuevo.

Programmer > Enable Programmer

Obsérvese como ahora aparece la versión (actualizada) del firmware en la ventana 'Output'.



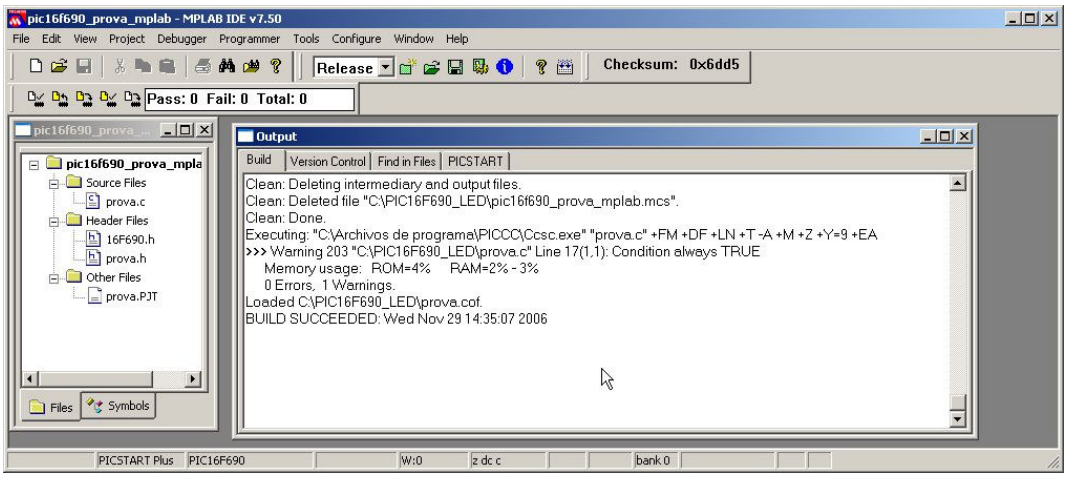
---

24. Vamos a compilarlo de nuevo. Para ello:

Project > Compile

---

25. MPLAB invocará a CCS PCWH y veremos como, rápidamente, compila nuestro proyecto. Si todo ha ido bien, al finalizar el proceso veremos aparecer el siguiente mensaje:

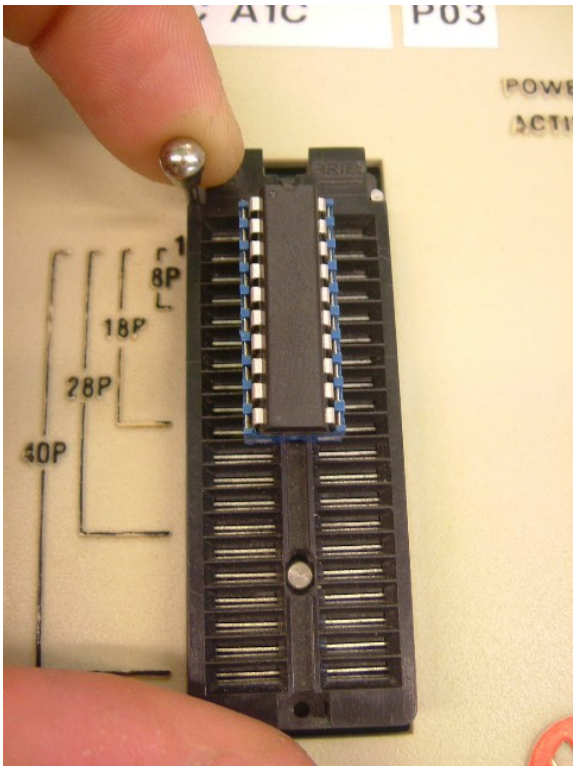


---

26. Llegó el momento de insertar el micro en el zócalo del módulo programador. Lo primero es tirar de la palanquita hacia arriba.

---

27. Insertamos el PIC16F690 teniendo en cuenta que el pin número 1 está situado en la parte superior izquierda.



---

28. Bajamos la palanquita.

---

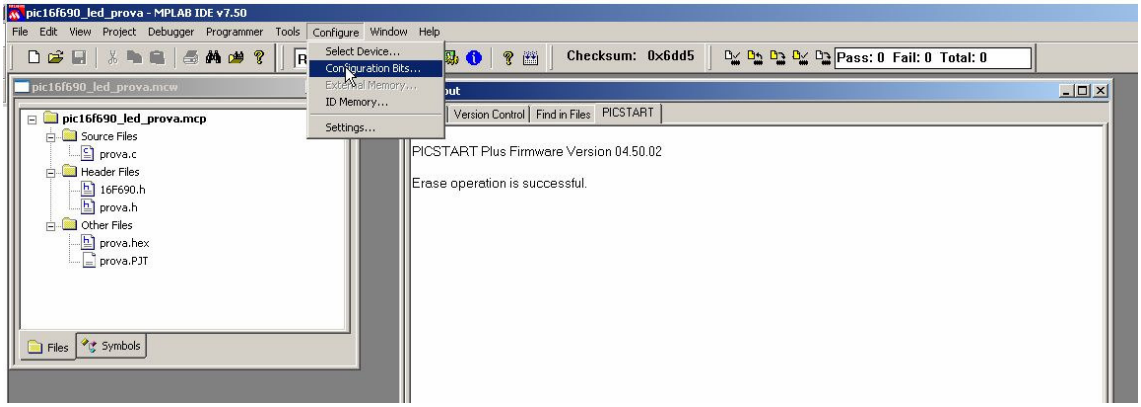
29. Procedemos a borrar el contenido previo.

Programmer > Erase Flash Device

---

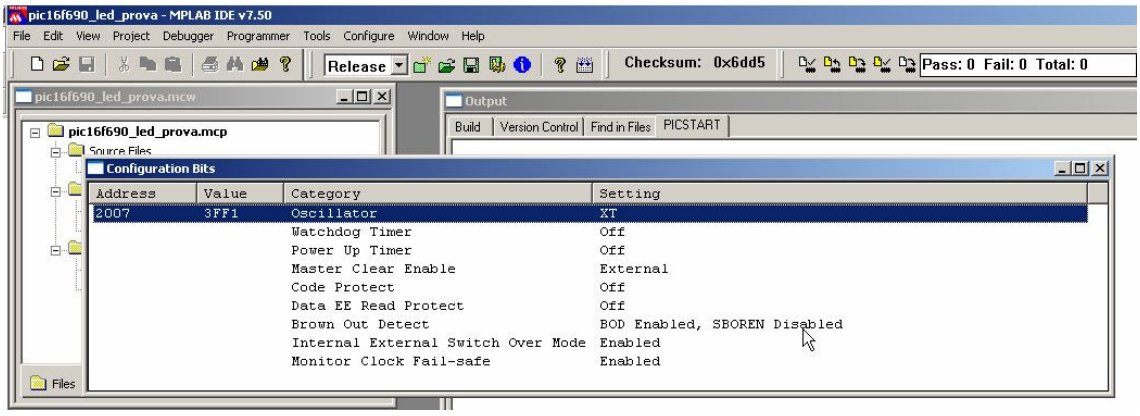
30. Después del mensaje de borrado exitoso, accedemos al menú:

Configure > Configuration Bits.



---

31. Los parámetros deberán coincidir con los aquí mostrados.



En el caso de haber alguna directiva tipo #FUSES en nuestro programa, resulta imprescindible que los parámetros allí establecidos coincidan con los parámetros aquí mostrados.

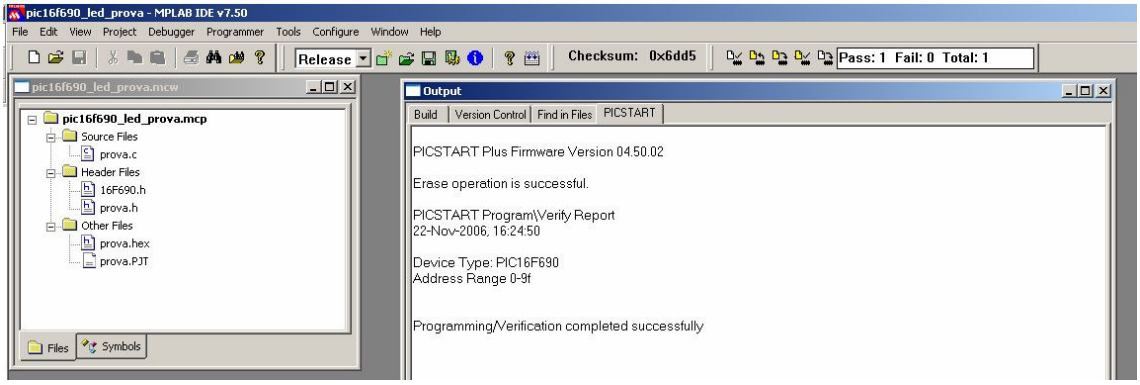
---

32. Ya podemos programar el micro. Para realizar esto:

Programmer > Program

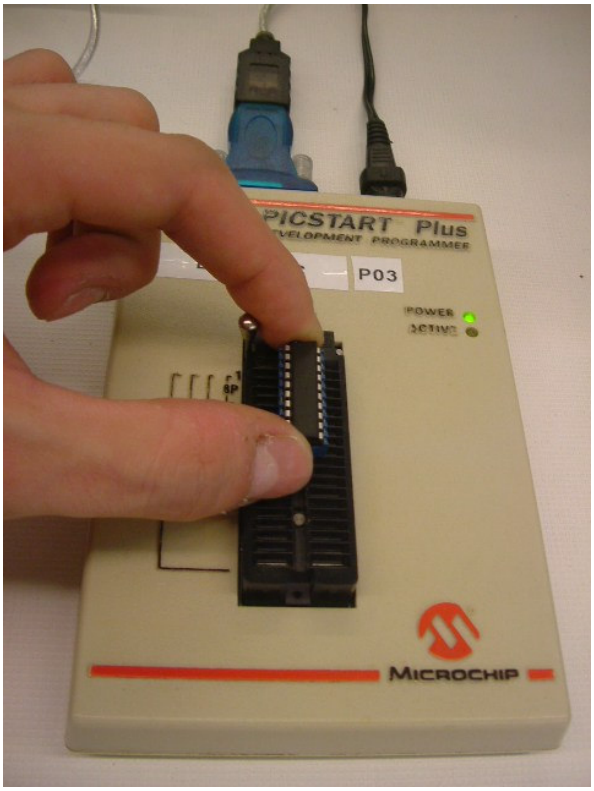
---

33. Si todo ha ido bien podremos ver el siguiente mensaje en la ventana 'Output'.



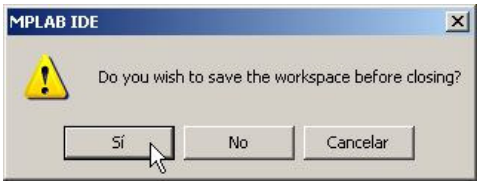
---

34. Ya podemos extraer el microcontrolador del módulo programador, levantando previamente la palanquita.



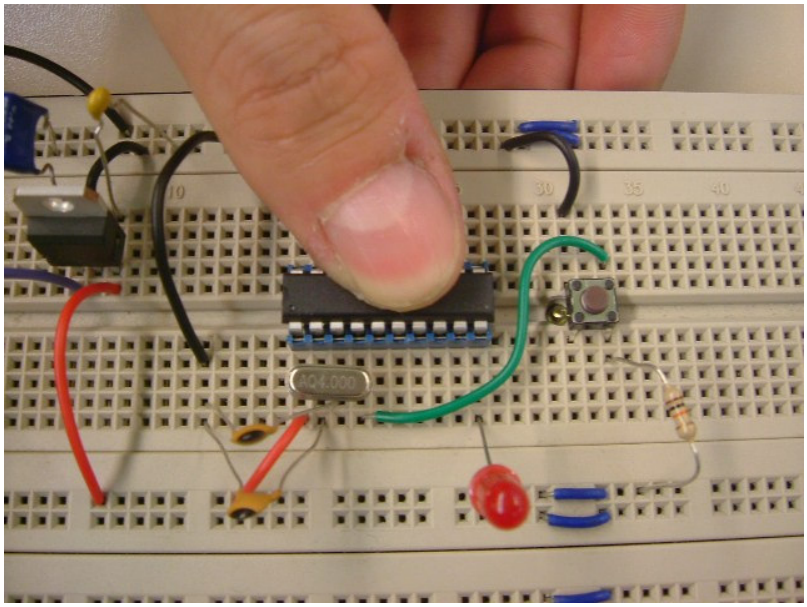
---

35. Al cerrar MPLAB IDE se nos preguntará si queremos guardar la configuración del entorno (Workspace). Es recomendable hacerlo si se va a volver a trabajar con el mismo proyecto de forma frecuente.



3.8. Comprobación del correcto funcionamiento

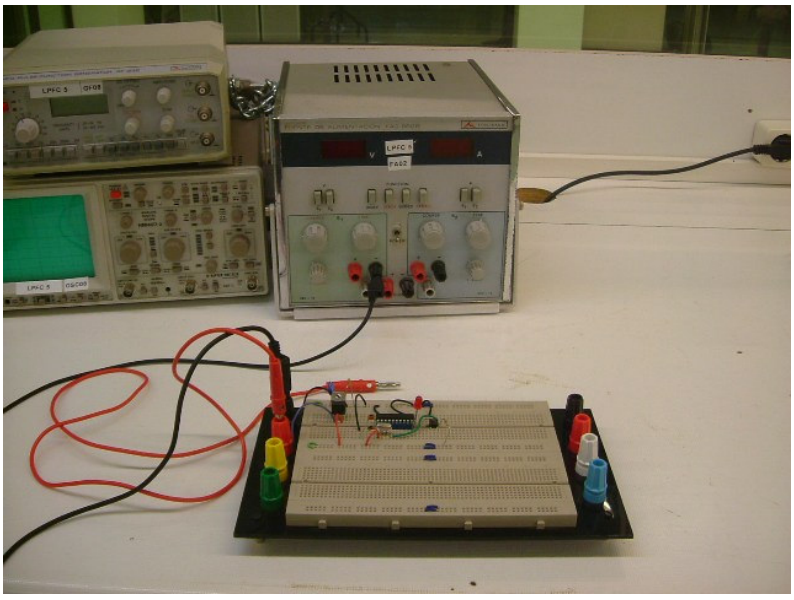
1. Insertamos el micro en la posición que le corresponde de la protoboard, asegurándonos de que queda bien fijado y las patitas hacen contacto con el sustrato metálico de la placa de pruebas.



---

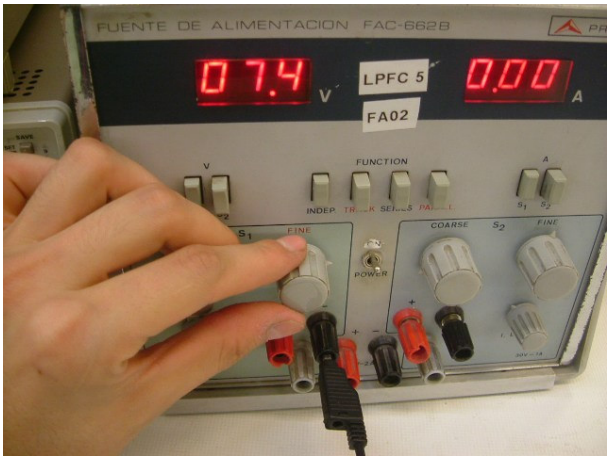
2. Conectamos los cables banana-banana a la protoboard y a la fuente de alimentación, a excepción del borne positivo que corresponde a la fuente. El porqué de proceder así radica en el pico de tensión y corriente que proporciona la fuente al encenderla. Podríamos dañar nuestro micro.

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



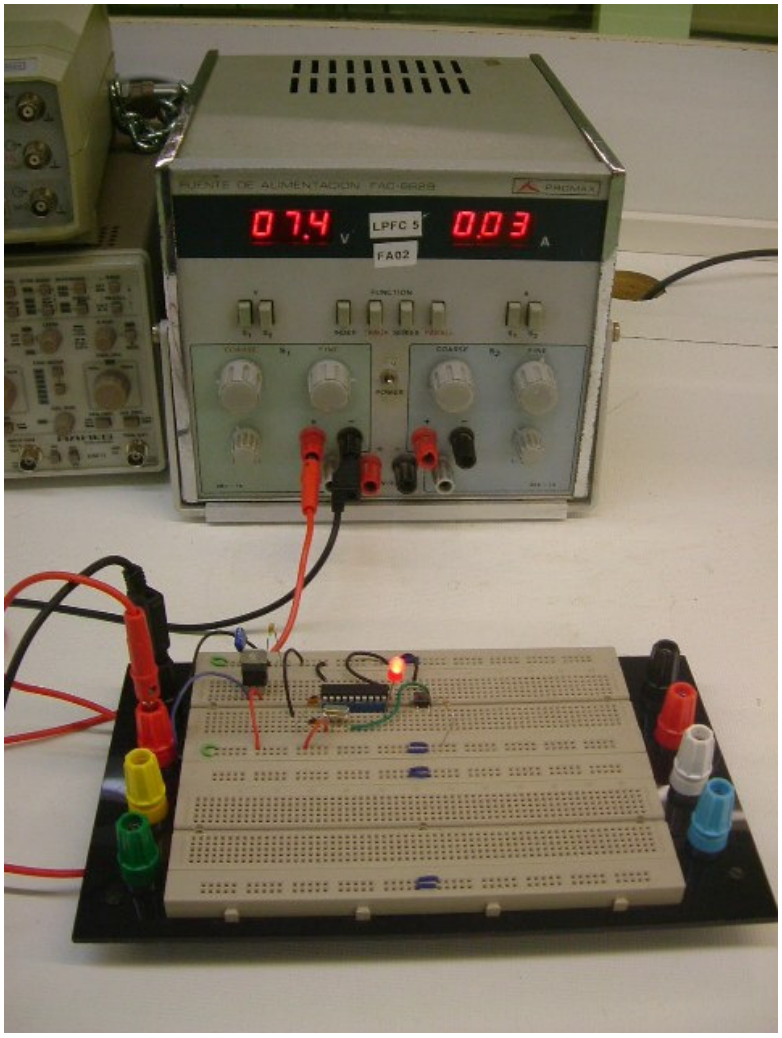
---

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.