

PROGRAMACIÓN DE PICs EN LENGUAJE C

Oscar Luis Vele G

oscar_vele@yahoo.es

Cuenca – Ecuador

1. Introducción.

Una de las ramas de la electrónica que ha tomado un mayor auge y desarrollo en la actualidad es la del estudio de los Microcontroladores, los mismos que presentan grandes ventajas al momento de desarrollar sistemas embebidos, sobre todo en lo que se refiere al precio, tamaño, software de desarrollo, etc. En el presente tutorial nos dedicaremos al estudio de la programación en lenguaje C de Microcontroladores Microchip (PICs) mediante el software de desarrollo gratuito de Hi-Tech. Las preguntas podrían ser: porqué microcontroladores Microchip? Y porqué lenguaje C? La respuesta a la primera pregunta está en función de la variedad de estos dispositivos que están disponibles en nuestro medio y con respecto a la segunda pregunta, existen algunas razones de peso que se destacan a continuación:

- Creación de programas utilizando un lenguaje de alto nivel, de aquí, que la programación estructurada facilita el entendimiento y depuración, disminuyendo el tiempo de diseño.
- Manejo de librerías especializadas para el tratamiento de cadenas de caracteres, matemáticas, etc.
- Creación de código reutilizable y portable.

Cuántas veces en el desarrollo de nuestros proyectos nos hemos “roto la cabeza” tratando de realizar una simple división o intentando trabajar con números en formato de punto flotante; mediante este compilador las cosas son mucho más fáciles, sin embargo, muchas personas prefieren programar en lenguaje ensamblador y en justificación a eso se debe decir que muy difícilmente un compilador supera la eficiencia de un buen programador en lenguaje ensamblador. He aquí la disyuntiva, y la elección depende del tipo de programa o proyecto a realizar. Un compilador estaría bien para fines educativos o proyectos en los que una hipotética falla del programa (causada por el compilador) no represente un mayor riesgo, de aquí, que si hablamos de compiladores, existen de todos los tipos y con diferentes grados de confiabilidad, de donde se establece su precio, desde unos cientos a algunos miles de dólares. El compilador PICC de Hi-Tech es bastante bueno, a juzgar por los resultados y prestaciones del mismo.

En el presente documento se trabajará con microcontroladores de la familia media de Microchip y se asumirá que el lector ha trabajado antes con PICs¹ en MPLAB.

2. Desarrollo de un programa para PICs en lenguaje C.

En general, una aplicación en lenguaje C pasa por seis etapas antes de ejecutarse, y cuando se trabaja con microcontroladores, no es la excepción; estas etapas son: edición, preprocesado, compilación, enlace, carga y ejecución (Figura 1).

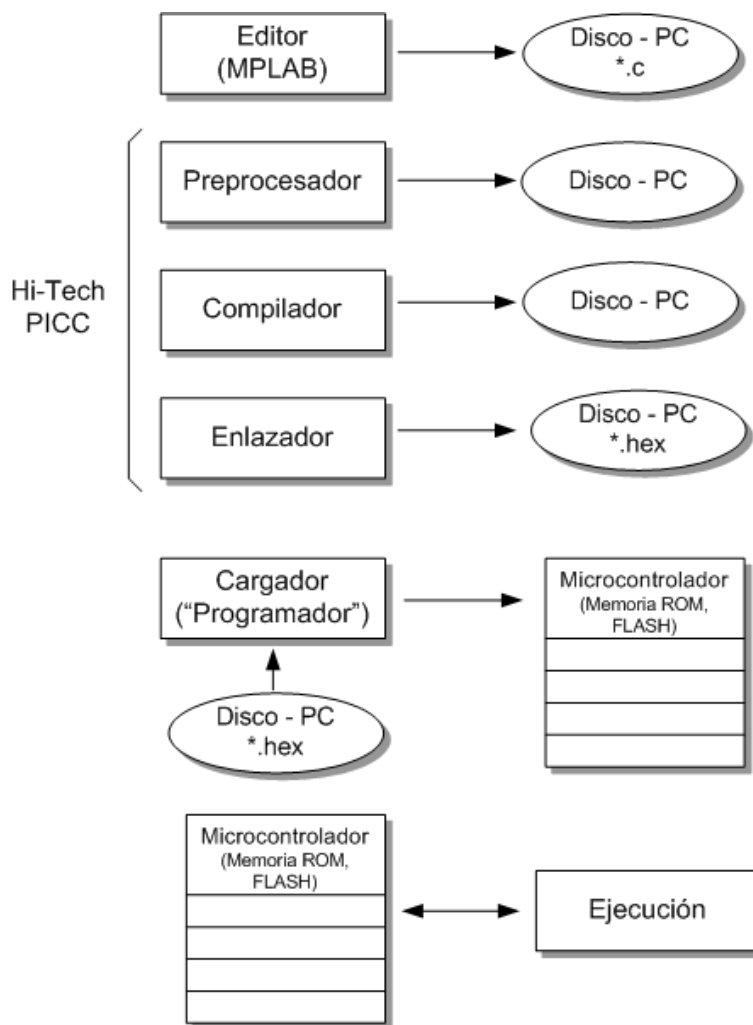


Figura 1. Desarrollo de un programa para PICs en lenguaje C.

La *edición* consiste en la escritura del programa en un editor (en nuestro caso será el programa MPLAB de Microchip), el mismo que además permitirá realizar las correcciones necesarias. El archivo fuente es guardado en el disco de la computadora con la extensión *.c*. El *preprocesador* se encarga de realizar ciertas manipulaciones en el programa antes de la

¹ Si no tiene conocimientos de PICs, Ud. podría revisar el siguiente tutorial: "Curso de Microcontroladores Microchip" en <http://loslocoselectro.blogspot.com/>

compilación, obedeciendo a comandos denominados *directivas del preprocesador*¹. El *compilador* realiza la traducción del programa en C a código en lenguaje de máquina “entendible” por el microcontrolador. El *enlazador* (linker) realiza el enlace del programa fuente con las referencias a funciones y datos definidos en otra parte, tales como bibliotecas estándar o privadas que constituyen una determinada aplicación, y así crea una imagen ejecutable de nuestro programa (archivo *.hex*). La siguiente fase realiza la *carga* de la imagen ejecutable en la memoria del Microcontrolador (ROM o FLASH), para esto se dispone de “*programadores*” que realizan este trabajo; se tiene a PICSTARPLUS de Microchip y algunos gratuitos como IC-PROG, PONY-PROG, etc., que conjuntamente con el hardware adecuado (JDM, Propic, etc., cuyos circuitos están disponibles en la red) son una alternativa interesante cuando no se tiene mucho dinero. Por último, la *ejecución* del programa se realiza en el microcontrolador (después de una conexión y alimentación adecuada) una instrucción a la vez.

3. Hi-Tech PICC.

Hi-Tech PICC es un compilador de alto rendimiento para las familias 10/12/14/16/17 de Microchip, el mismo que implementa en forma completa ISO/ANSI C (excepto recursión de funciones); además, maneja varios tipos de variables, incluso las de punto flotante de 24 bits e IEEE de 32 bits. Este compilador genera un código altamente optimizado, brindando al programador capacidades solamente limitadas por las características del microcontrolador. Entre las características más importantes se pueden destacar:

- Confiabilidad probada.
- Manejo automático de páginas y bancos de memoria.
- Múltiples niveles de optimización² en código C.
- Optimización en ensamblador.
- Librerías estándar de ANSI C.
- Inclusión de lenguaje ensamblador entre código C.
- Ilimitado número de archivos fuente.
- Compatible con MPLAB IDE, MPLAB ICD y otras herramientas de desarrollo.
- Compatible con varias plataformas: Windows, Linux, Unix, Mac OS X, S Solaris.

Dado que Hi-Tech PICC es un compilador bastante confiable, su precio es relativamente alto, por esta razón, a través de este tutorial se trabajará con la versión gratuita Hi-Tech PICC-Lite, que es igual al compilador comercial, sin embargo, presenta algunas restricciones con

¹ Son comandos del lenguaje que se encargan de realizar algunas tareas (definición de constantes simbólicas y macros, inclusión de archivos, etc.) antes de la compilación y estas directivas siempre comienzan con #.

² Grado de eficiencia en el tamaño del código generado y en la velocidad de ejecución del mismo.

respecto a los microcontroladores que se pueden programar y la cantidad de memoria que puede ser usada. En la tabla 1 se muestra la lista de microcontroladores así como sus limitaciones.

Microcontrolador	Limitaciones
12F629	Sin limitaciones
12F675	Sin limitaciones
16C84	Sin limitaciones
16F627	2 bancos de memoria RAM
16F627A	2 bancos de memoria RAM
16F684	1 banco de RAM, 1K de memoria de programa
16F690	2 bancos de RAM, 2K de memoria de programa
16F84A	Sin limitaciones
16F877	2 bancos de RAM, 2K de memoria de programa
16F877A	2 bancos de RAM, 2K de memoria de programa

Tabla 1. Microcontroladores soportados y limitaciones en memoria.

4. Instalación y configuración.

Las siguientes herramientas son necesarias para la realización de un proyecto (Se muestran las direcciones Web para descargar gratuitamente estos programas):

- MPLAB IDE v. 7.xx <http://www.microchip.com/>
- HI-TECH PICC-Lite ¹ <http://www.htsoft.com/downloads/demos.php>

Es recomendable instalar primero MPLAB IDE antes de instalar el compilador PICC. En el presente tutorial se trabajará con la versión 7.31 de MPLAB IDE y con la versión 9.50PL2 de PICC-Lite.

Al realizar la instalación del compilador PICC-Lite es aconsejable seguir la configuración por defecto del instalador, sobre todo cuando no se tiene mucha experiencia. Después de haber instalado el compilador, lo primero que se debe comprobar es el correcto enlace entre los dos programas², tal como se muestra en la figura 2.

¹ La descarga debe ser realizada en la sección de software gratuito (Free software), no en la sección de demos, ya que éstos expiran dentro de un determinado tiempo. Para poder descargar los programas, el usuario debe estar registrado.

² Normalmente, esto ha sido realizado por el instalador (PICC-Lite).

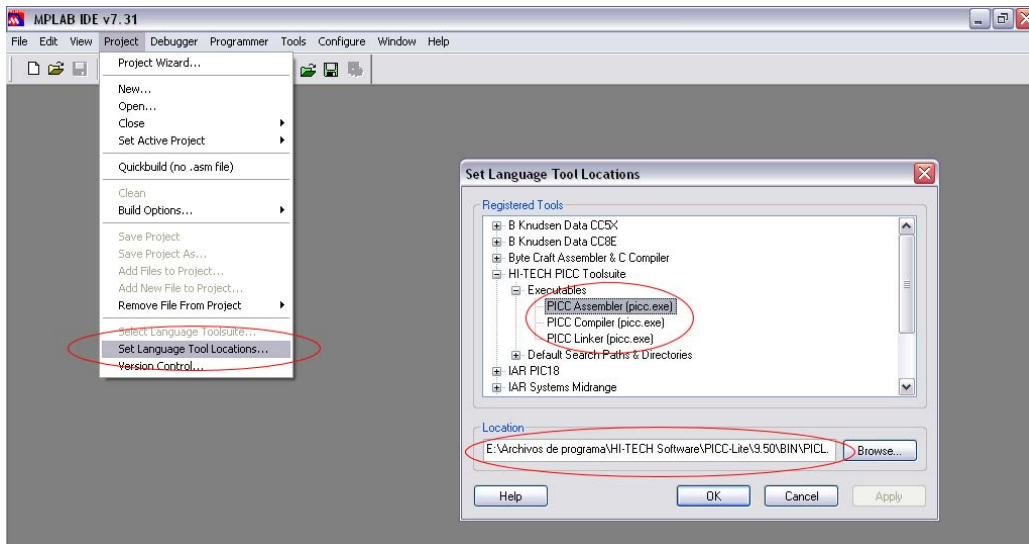


Figura 2. Comprobación de la configuración de MPLAB.

Dentro de la ventana “Set Lenguaje Tool Locations” asegúrese de que para “PICC Asembler”, “PICC Compiler” y “PICC Linker” el archivo **PICL.EXE** esté correctamente direccionado (Por lo general dentro de **Raíz:\Archivos de programa\HI-TECH Software\PICC-Lite\9.50\BIN**), en caso contrario, utilizar la herramienta de configuración “Configure MPLAB” en el menú del programas o ejecutar el archivo **MPLABConfig.EXE** que se encuentra dentro de la carpeta **BIN** en el directorio de instalación del compilador.

5. Realización de un nuevo proyecto en MPLAB utilizando el compilador PICC-Lite.

Quizás la forma más eficiente de comenzar un nuevo proyecto es mediante el asistente de proyectos “**Project Wizard...**”, tal como se ve en la figura 3.

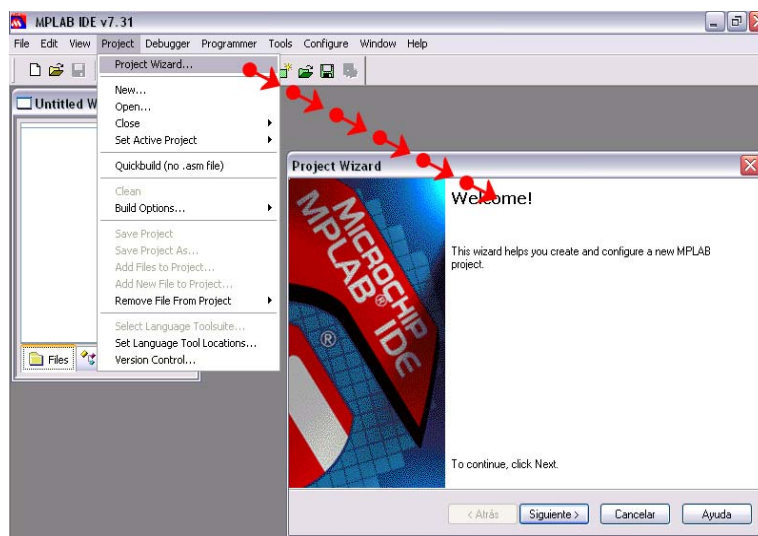


Figura 3. Realización de un nuevo proyecto.

A continuación se debe elegir el microprocesador con el cual se va a trabajar (Se debe tener en cuenta las limitaciones que presenta esta versión gratuita del compilador, es decir, se deben elegir únicamente los microcontroladores soportados – Tabla 1).

El siguiente paso es elegir el lenguaje de programación, en nuestro caso, “HI-TECH PICC Toolsuite”.

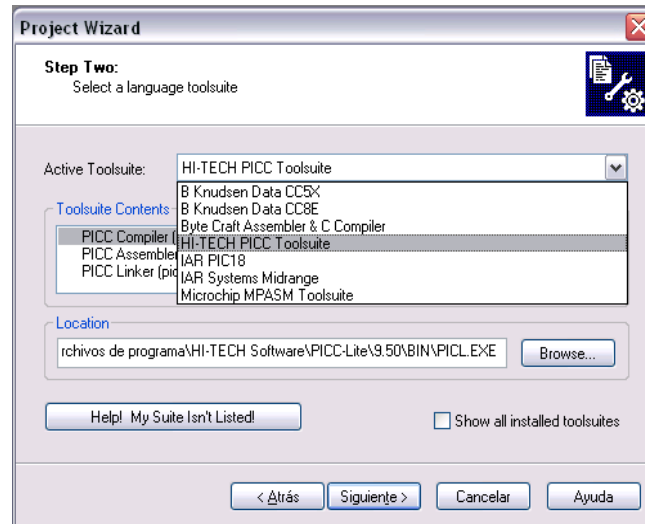


Figura 4. Elección del lenguaje de programación.

Una vez realizado esto, ingresar el nombre del proyecto y elegir la carpeta en la cual se grabarán los archivos.

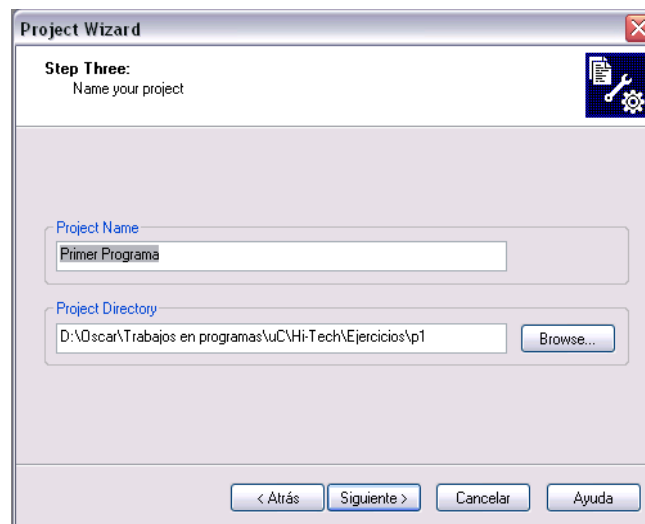


Figura 5. Nombre y directorio del proyecto.

Opcionalmente, se pueden incluir archivos en nuestro proyecto, tales como: librerías personalizadas, archivos de cabecera, etc. En nuestro primer ejemplo no incluimos ningún archivo.

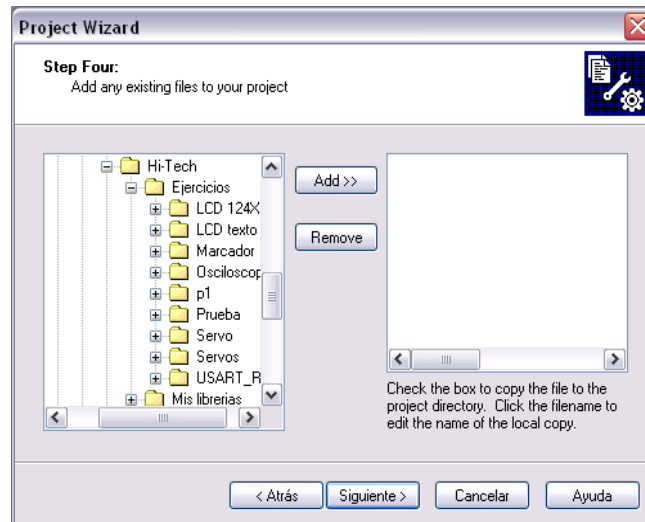


Figura 6. Inclusión de archivos al proyecto.

Finalmente, se presenta un resumen de la configuración del proyecto.

Después de terminar de configurar el asistente de proyectos, creamos un nuevo archivo y para poder compilar el proyecto, se requiere mínimo el lazo principal (main) del lenguaje ANSI C¹.

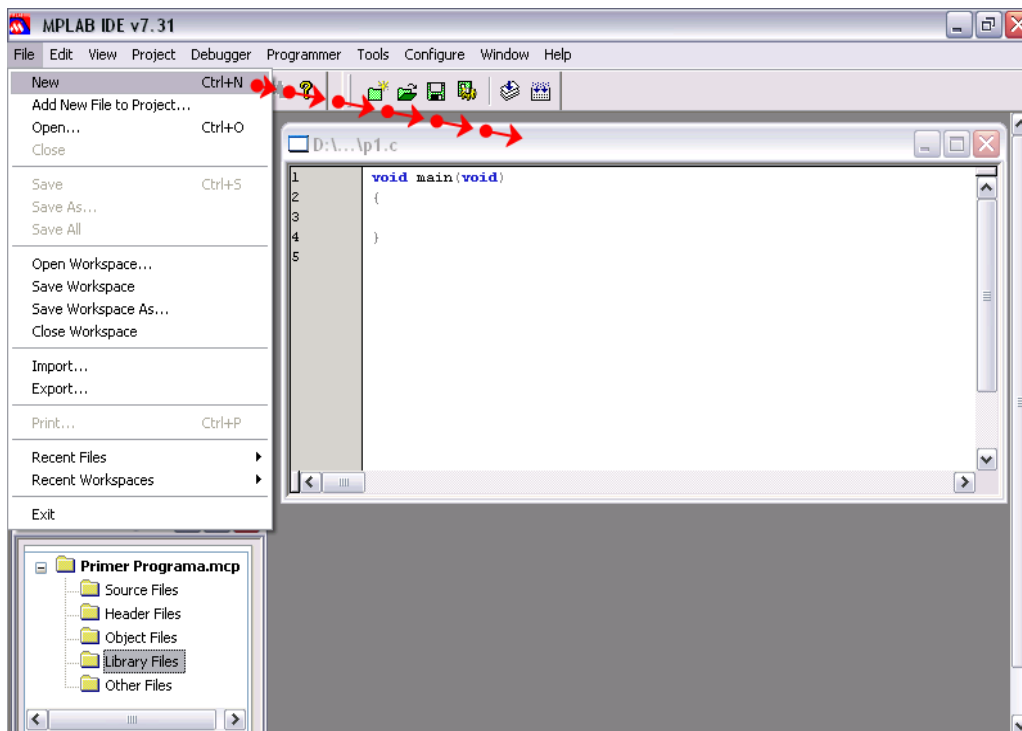


Figura 7. Creación del archivo fuente.

A continuación se debe guardar el archivo con la extensión “.c”, dentro de la carpeta en la cual se inició el proyecto (en nuestro caso, **p1**), tal como muestra la figura 8.

¹ Se asumirá que el lector tiene conocimientos básicos del lenguaje de programación ANSI C.

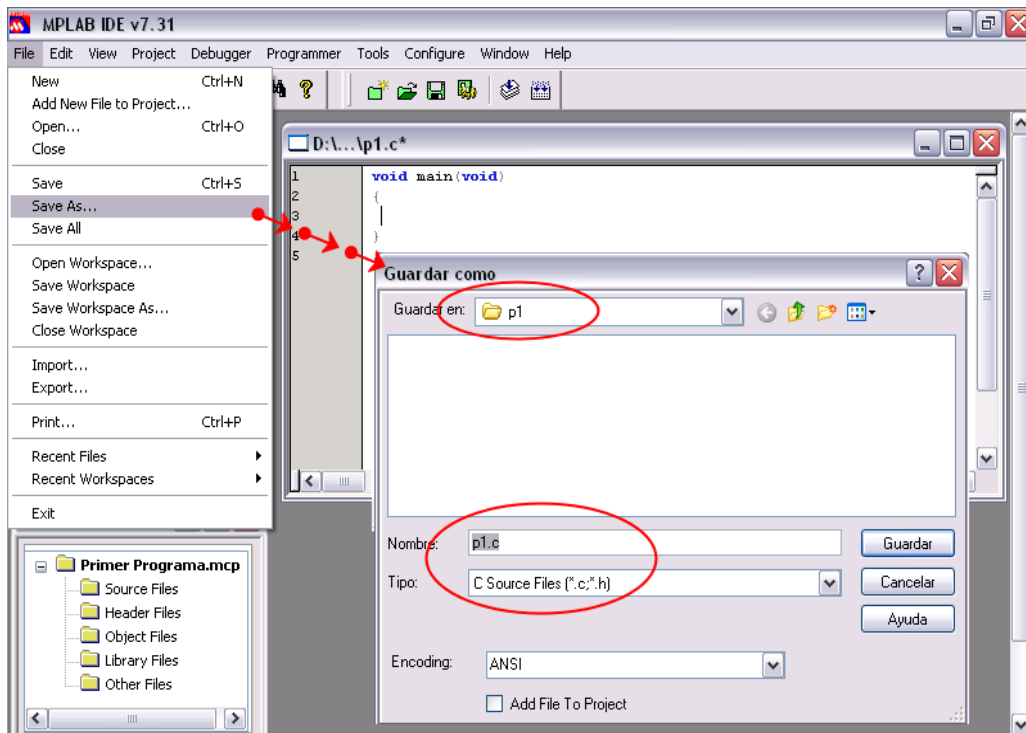


Figura 8. Creación del archivo fuente.

Finalmente, se debe incluir el archivo recién guardado dentro de los archivos fuente del proyecto (click derecho en “Source Files”).

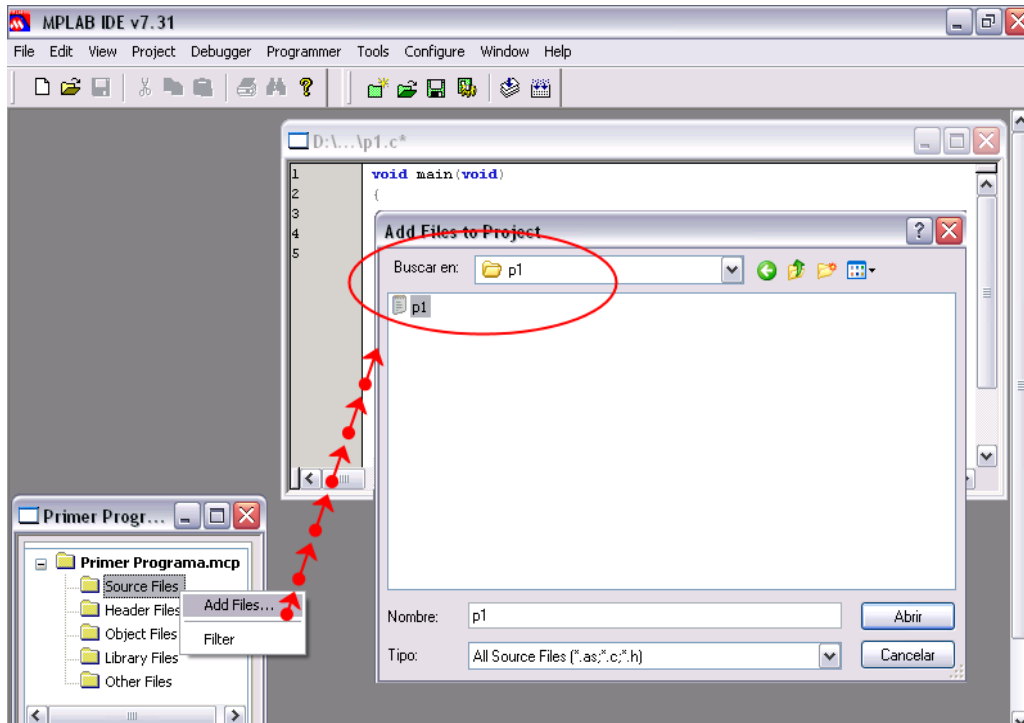


Figura 9. Inclusión del archivo fuente.

A partir de ahora, se puede compilar, simular, etc., el proyecto. Si la compilación ha sido exitosa, se presentará un resumen de consumo de memoria en la ventana de salida, tal como se muestra en la figura 10,

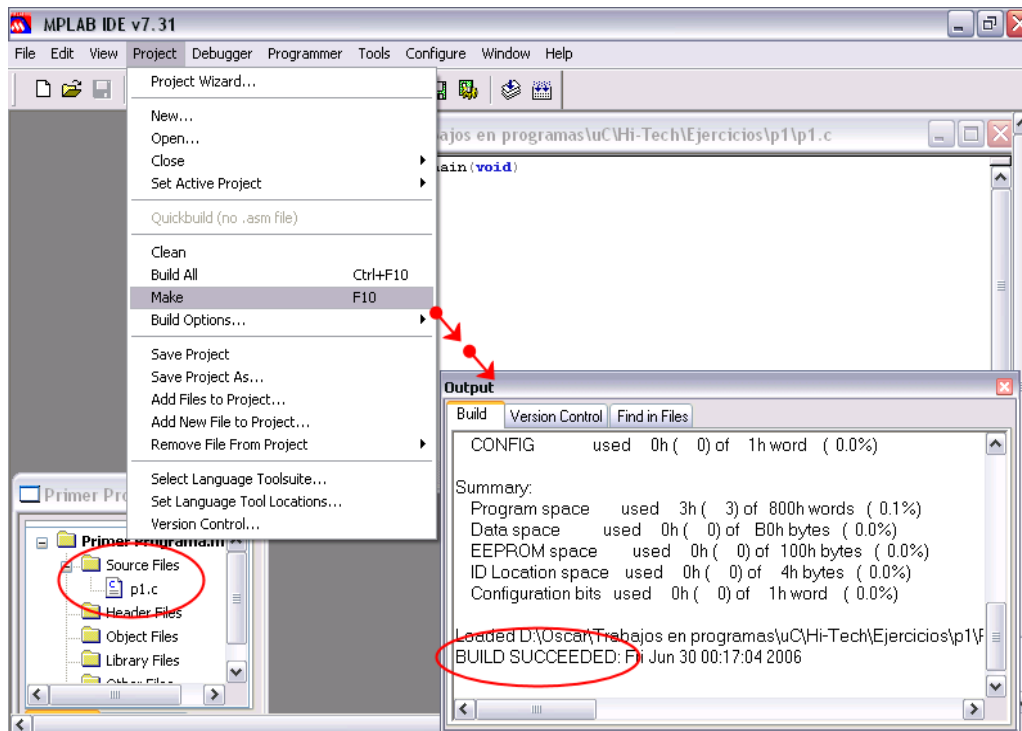


Figura 10. Compilación del proyecto.

6. Simulación.

Una de las características más importantes al utilizar el compilador PICC dentro del entorno de MPLAB, es el empleo del simulador con todas sus capacidades. En la figura 11 se muestra el entorno de simulación de un programa demostrativo (no tiene aplicación) sencillo que realiza un conteo cada 500ms, así como una salida que oscila con el mismo periodo. A continuación se puede observar el código de este programa.

```

//EJERCICIO 1
//
//                                     Oscar Luis Vele
//                                     18/09/2006
//*****
#define FC 16                          //Frecuencia del oscilador
//                                     //(para la secuencia de demoras)

#include <htc.h>                        //Librería general (siempre va)
#include "demoras.h"                   //Librería personalizada de retardos

__CONFIG(0x3D32); //Palabra de configuracion del uC
//-----
void main(void)
{
    long cont=0;                       //Variable entera de 16 bits
    TRISA=0;                           //Salidas
    TRISE=255;                         //Entradas
    ADCON1=7;                          //Líneas digitales
    while(1)                            //Lazo infinito
    {
        if (RBO==1)
            cont++;                     //Incremento del contador
        else
            cont--;                     //Decremento del contador
        RA0=1;                          //Salida en alto
        dem_ms(250);                    //Demora de n milisegundos
        RA0=0;                          //Salida en bajo
        dem_ms(250);                    //Demora de n milisegundos
    }
}

```

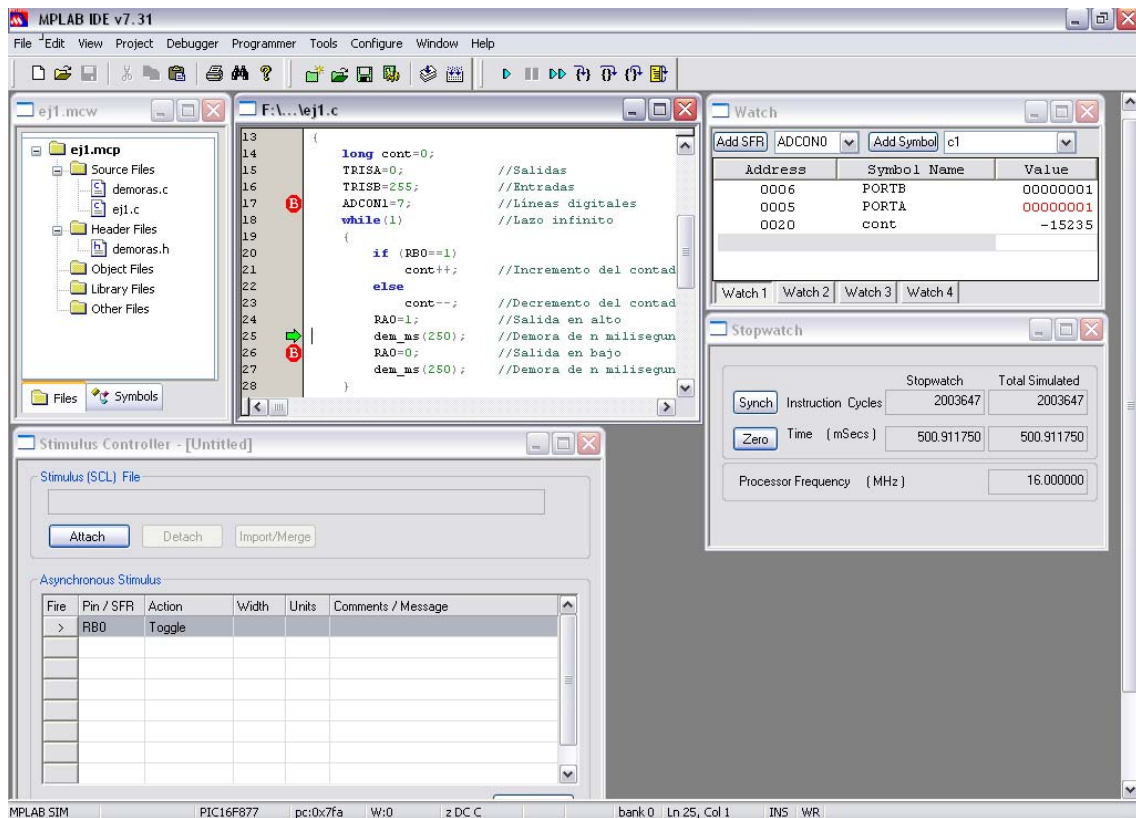


Figura 11. Simulación del proyecto.

Como se observa en el programa, éste incorpora un archivo de retardos “demoras.c”, el mismo que se detalla a continuación¹.

```
//ARCHIVO DE DEMORAS EN ms y s
//
//                                Oscar Luis Vele
//                                14/01/2005
//*****
//Directivas
//Colocar la frecuencia del cristal
#if !defined(FC)
#define FC 16 //Frecuencia del cristal en MHz
#endif
//*****
//DEMORA de 0 a 255 ms
void dem_ms(unsigned char d)
{
    unsigned char c1,c2,c3,c4;
    c4=34;
    for (c3=0;c3<d;c3++)
    {
        for (c1=0;c1<FC;c1++)
        {
            for (c2=0;c2<c4;c2++){ //Demora de lms con 1MHz
            }
        }
    }
}
//*****
//DEMORA de 0 a 255 seg
void dem_seg(unsigned char d)
{
    unsigned char c1;
    for (c1=0;c1<d;c1++)
    {
        dem_ms(250); //Demora de 1 segundo
        dem_ms(250);
        dem_ms(250);
        dem_ms(250);
    }
}
```

7. Utilización de Interrupciones.

El compilador PICC maneja las interrupciones mediante la función *interrupt*, la misma que opera de manera diferente a las demás funciones, ya que respalda registros importantes al ingresar a la interrupción y los restituye al salir de la misma.

Cabe destacar que dentro de la función interrupción solamente se pueden utilizar variables globales.

Para los microcontroladores de la familia media el encabezado de la función interrupción siempre es **void interrupt tc_int(void)**.

A continuación se muestra parte del código de un programa que utiliza interrupciones del Timer0 y Timer2.

¹ Tanto este como otros programas demostrativos se encuentran disponibles en la página WEB: <http://loslocoselectro.blogspot.com/>

```
//DECLARACIONES Y PALABRA DE CONFIGURACIÓN...
unsigned char dc=124; //Variable global
-----
void main(void)
{
//PROGRAMA PRINCIPAL
while(1) //Lazo infinito
{
//SE ELIMINÓ ESTA PARTE DEL PROGRAMA
}
}
-----
//INTERRUPCIÓN
void interrupt tc_int(void)
{
if(TOIF==1) //Atención al TMR0?
{
PORTC=0xFF;
PR2=dc; //Utilización de la variable global "dc"
TMR2=0;
TMR2ON=1;
TOIF=0; //Borrado de bandera
}

if(TMR2IF==1) //Atención al TMR2?
{
PORTC=0;
TMR2ON=0;
TMR2IF=0; //Borrado de bandera
}
}
```

En el sitio WEB se pueden descargar dos ejemplos de programas; el uno que ya se revisó anteriormente y el otro es una aplicación utilizando un LCD de texto. Una gama amplia de ejemplos se instala con el compilador dentro de **Raíz:\Archivos de programa\HI-TECH Software\ PICC-Lite\9.50\samples**. Entre estos ejemplos destacan aplicaciones con ADC, I2C, puerto serial, etc.

Finalmente, es justo mencionar que muchos aspectos de este compilador quedan fuera del alcance de este tutorial, pues la intención no fue realizar un manual detallado, sino un tutorial introductorio. Para información detallada de este compilador el lector puede referirse al manual (“User Manual”) que se instala conjuntamente con el compilador.

Que lo disfrute!!

8. Referencias.

- DEITEL, “Cómo programar en C++”, Cuarta edición.
- “PICC User Manual”. <http://www.htsoft.com/>