UNNOBA

MANUAL DE PROGRAMACIÓN Y SIMULACIÓN: MPLAB - PROTEUS

Guía de estudio

Cátedra Arquitectura;Lucas Benjamin Cicerchia;Pablo Addante

2014

www.unnoba.edu.ar

MPLAB X

MPLAB X IDE es un entorno de desarrollo integrado o interfaz de desarrollo (IDE) que permite crear aplicaciones para microcontroladores de la empresa Microchip. Brinda la posibilidad de desarrollar código para diferentes microcontroladores embebidos y se integra con distintos programadores siendo así capaz de grabar directamente sobre el dispositivo.

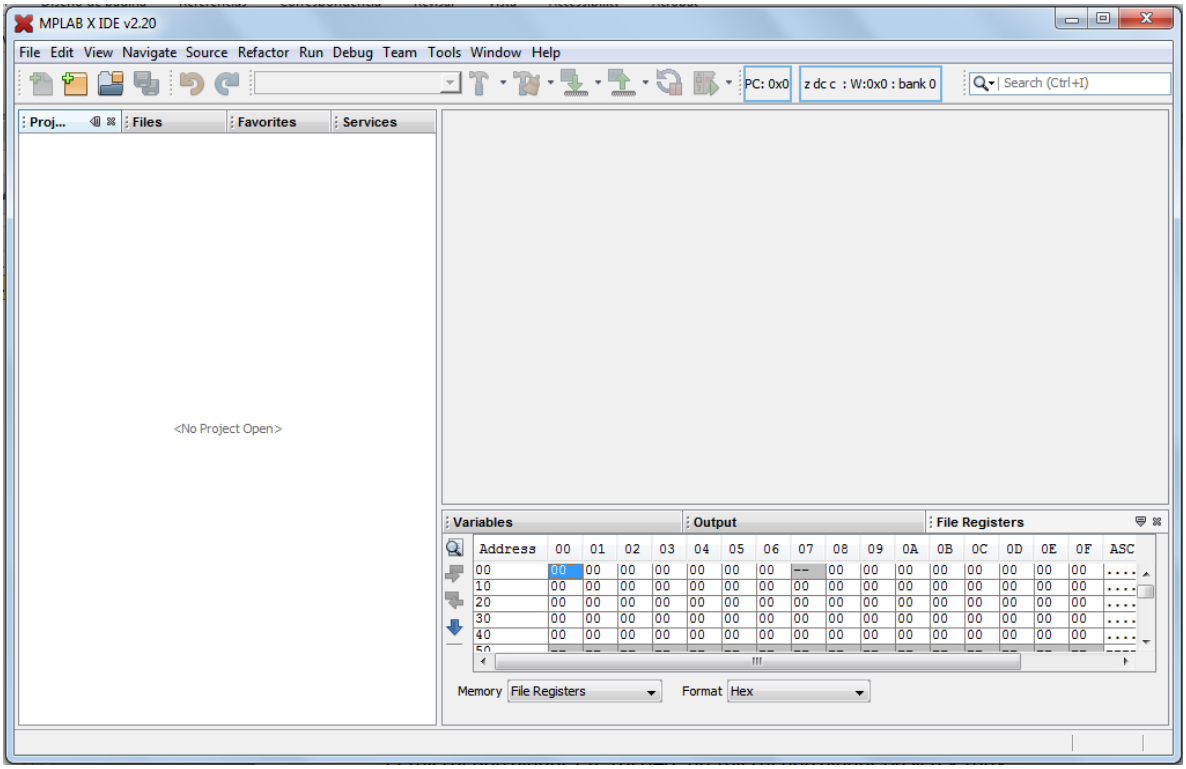

Ilustración 1 - Ventana Principal MPLAB X IDE

DESARROLLO DE UNA APLICACIÓN

Se mostrará cómo hacer para desarrollar una aplicación en MPLAB X para el microcontrolador PIC16F84A, un microcontrolador básico y muy utilizado.

CREACIÓN DE UN PROYECTO

El primer paso es la creación de un proyecto. Para ello, en la ventana principal podemos usar el ícono que se indica con un recuadro rojo en la imagen a continuación, ir a *File* → *New Project* o presionar CTRL+SHIFT+N.

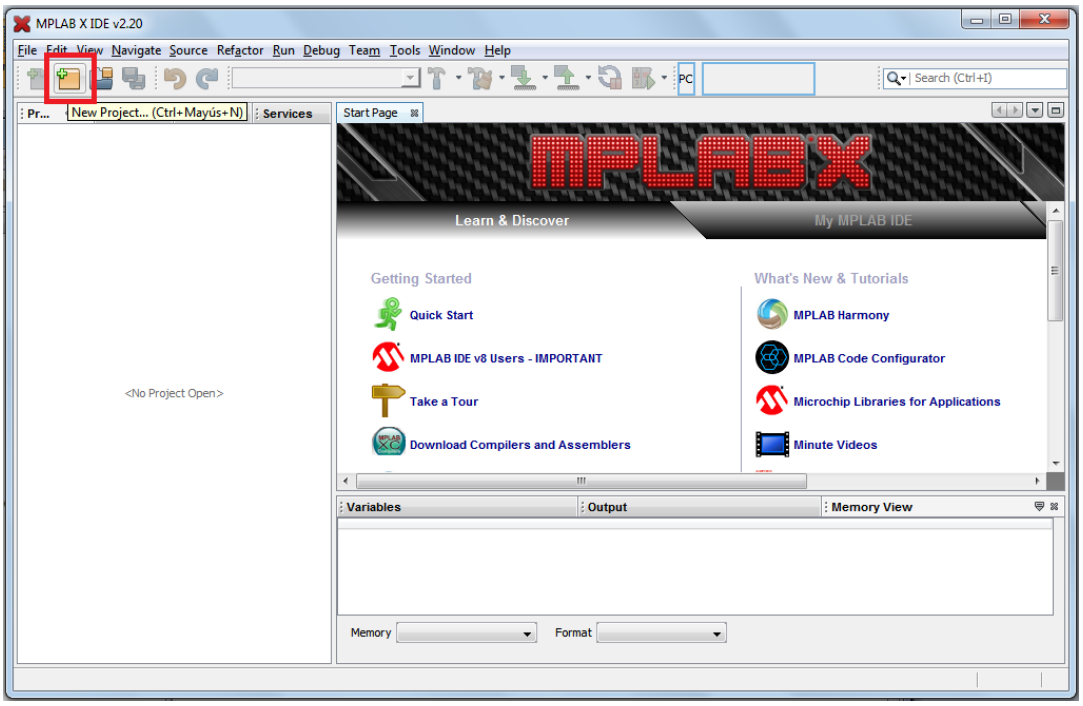

Ilustración 2 - Creación de un nuevo proyecto

Aparecerá una ventana para seleccionar el tipo de proyecto. En nuestro caso, debemos dejar seleccionado el que viene por defecto (*Microchip Embedded -> Standalone Project*) y presionar *Next*.

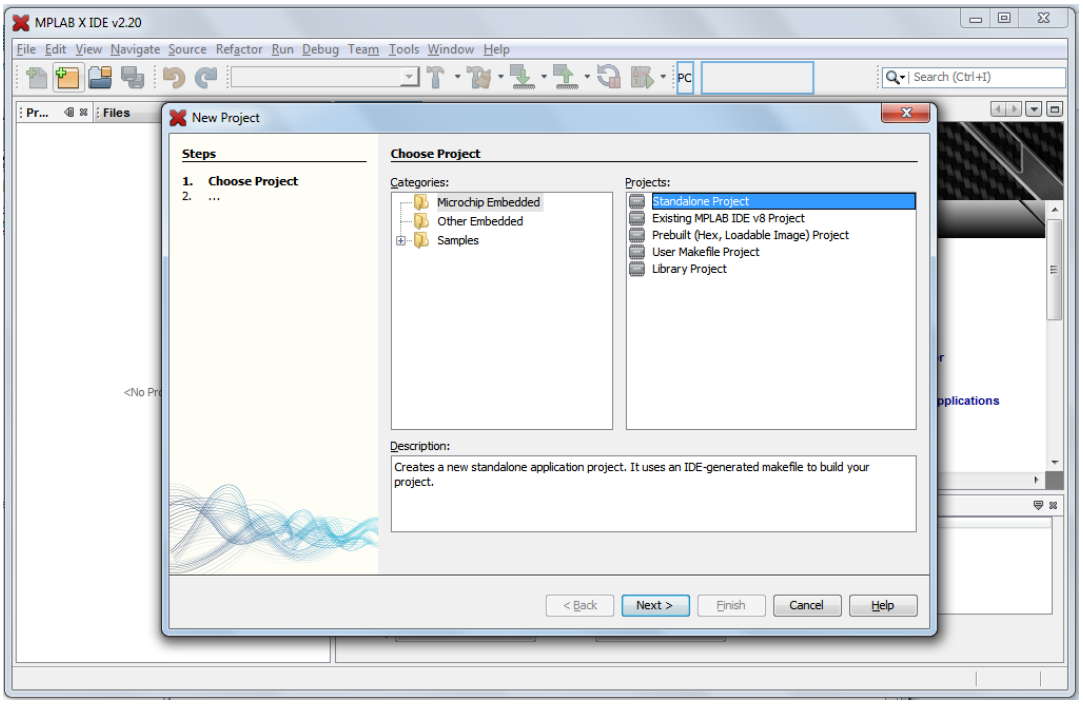

Ilustración 3 - Selección del Tipo de Proyecto

Elegimos luego el dispositivo, en nuestro caso *PIC16F84A*, y presionamos *Next*.

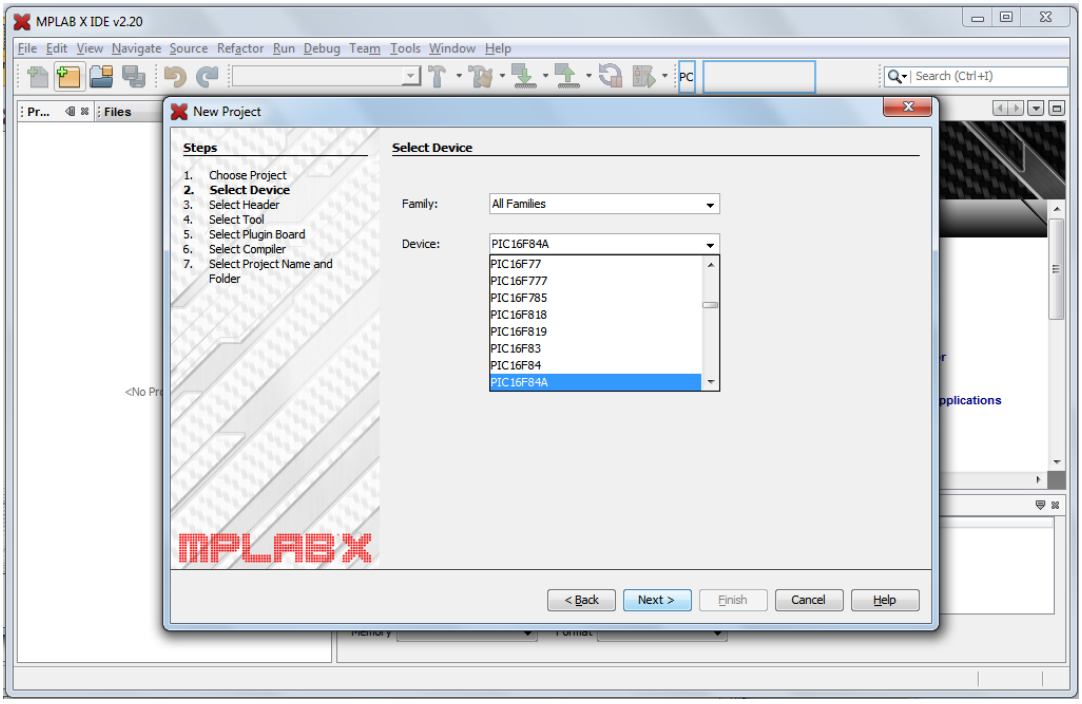

Ilustración 4 - Selección del Microcontrolador PIC

A continuación debemos elegir el programador a utilizar. El programador sirve para transferir nuestros programas al PIC. Si bien no lo utilizaremos, debemos selecciona una opción para continuar. Elegimos *Simulator* y presionamos *Next*.

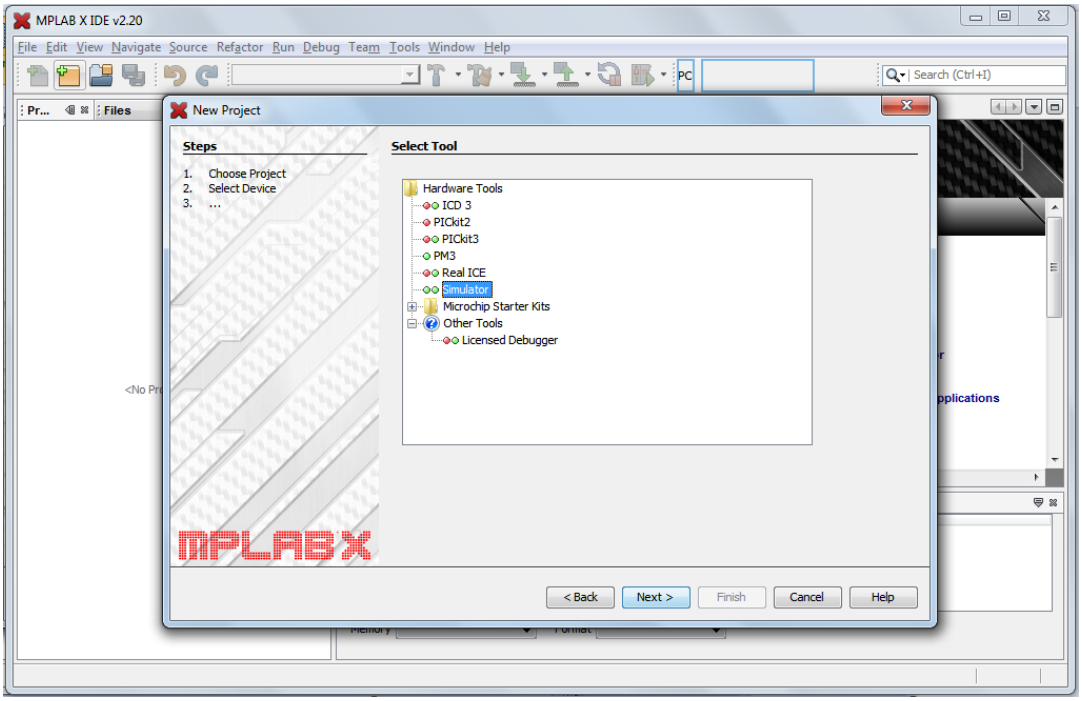

Ilustración 5 - Selección del Programador de PIC

Luego seleccionaremos el compilador, que dependerá del lenguaje que usemos. Elegimos *MPASM*, para programar en Assembler, y presionamos *Next*.

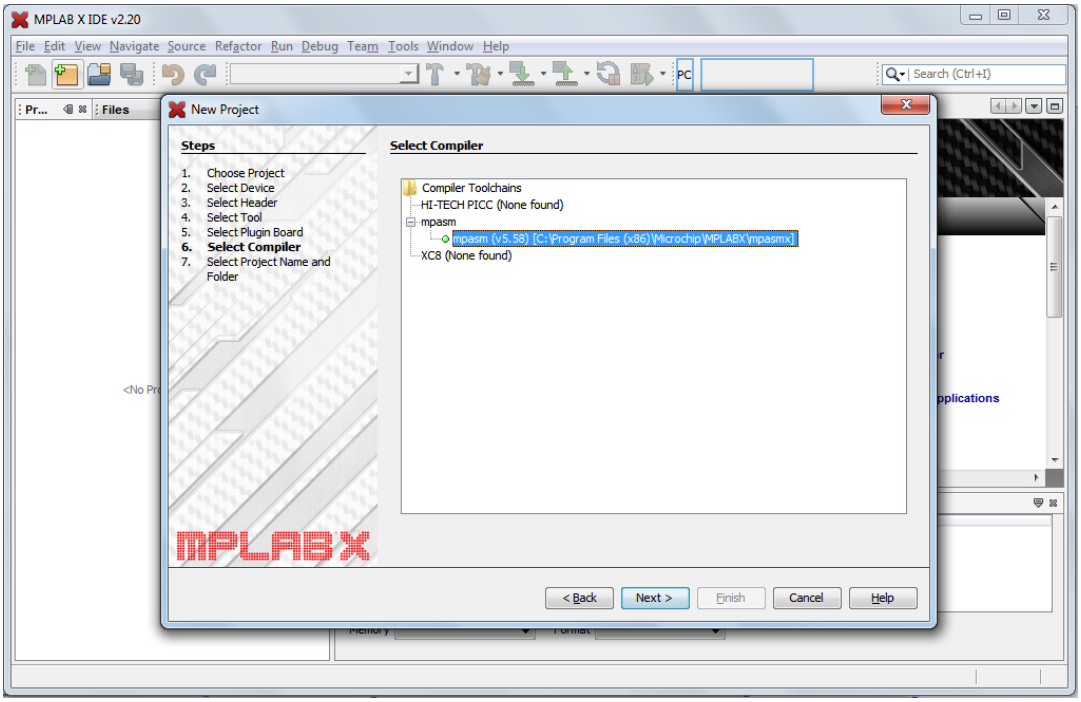

Ilustración 6 - Selección del Compilador

Por último, le asignamos un nombre al proyecto en el campo *Proyect Name* y luego presionamos *Finish*.

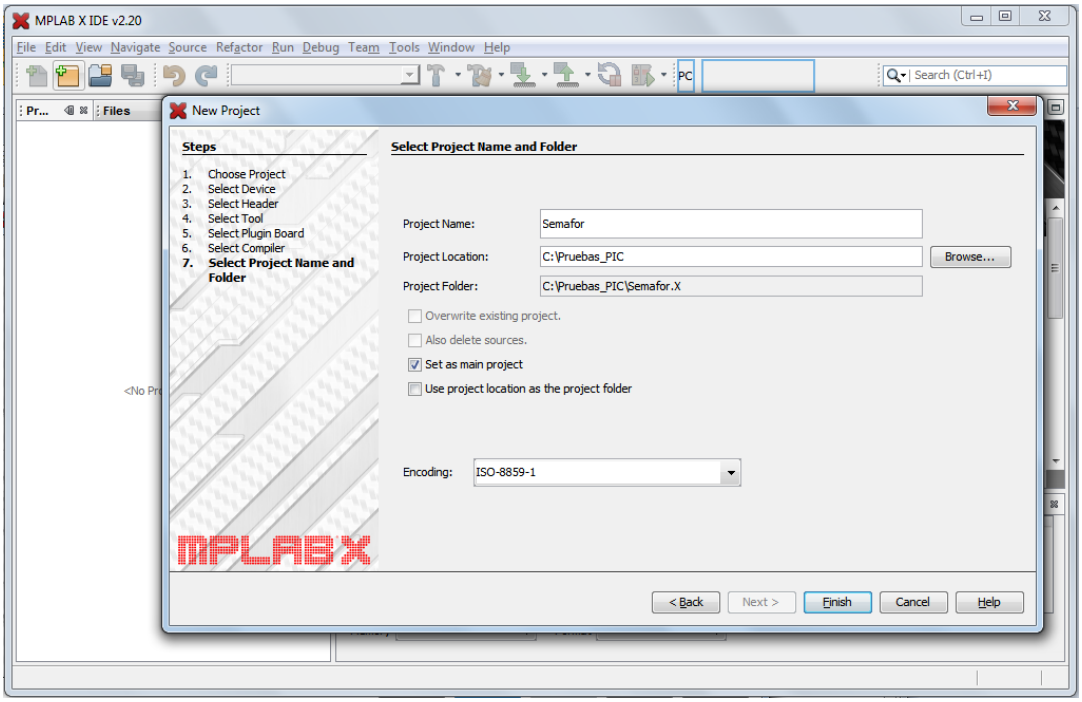

Ilustración 7 - Selección del Nombre del proyecto

Allí aparecerá el proyecto abierto.

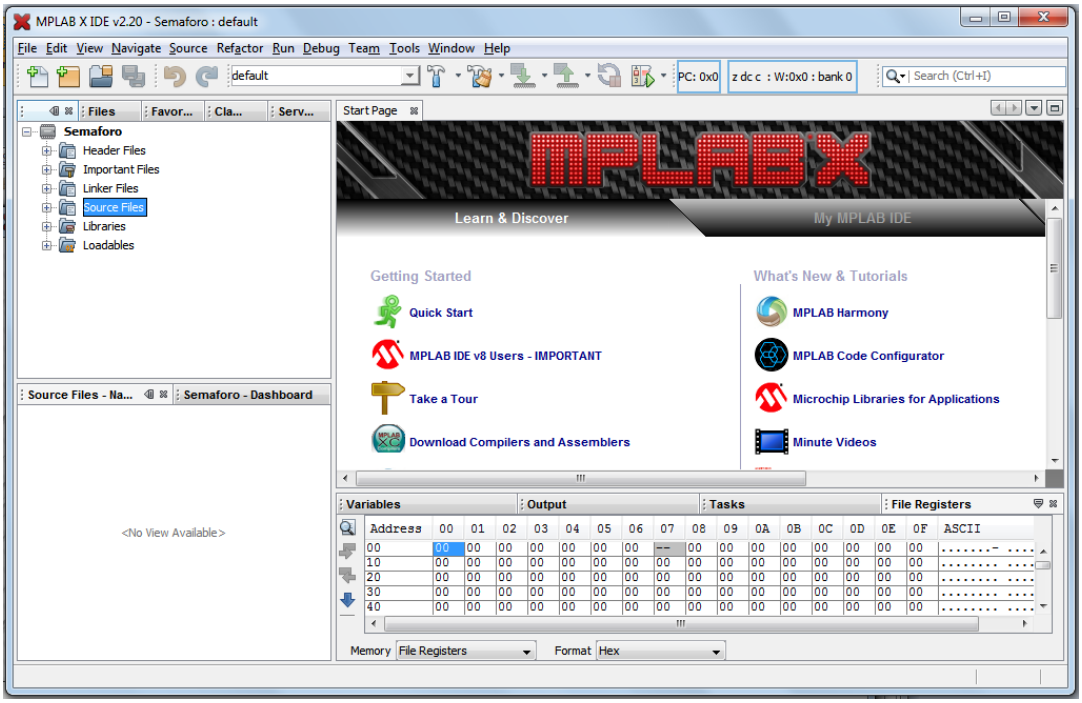

Ilustración 8 - Proyecto creado y abierto

A este punto, solo resta escribir el programa. Para ello podemos presionar el ícono marcado con un recuadro rojo en la imagen a continuación, elegir *File* → *New File* en el menú, o presionar CTRL+N.

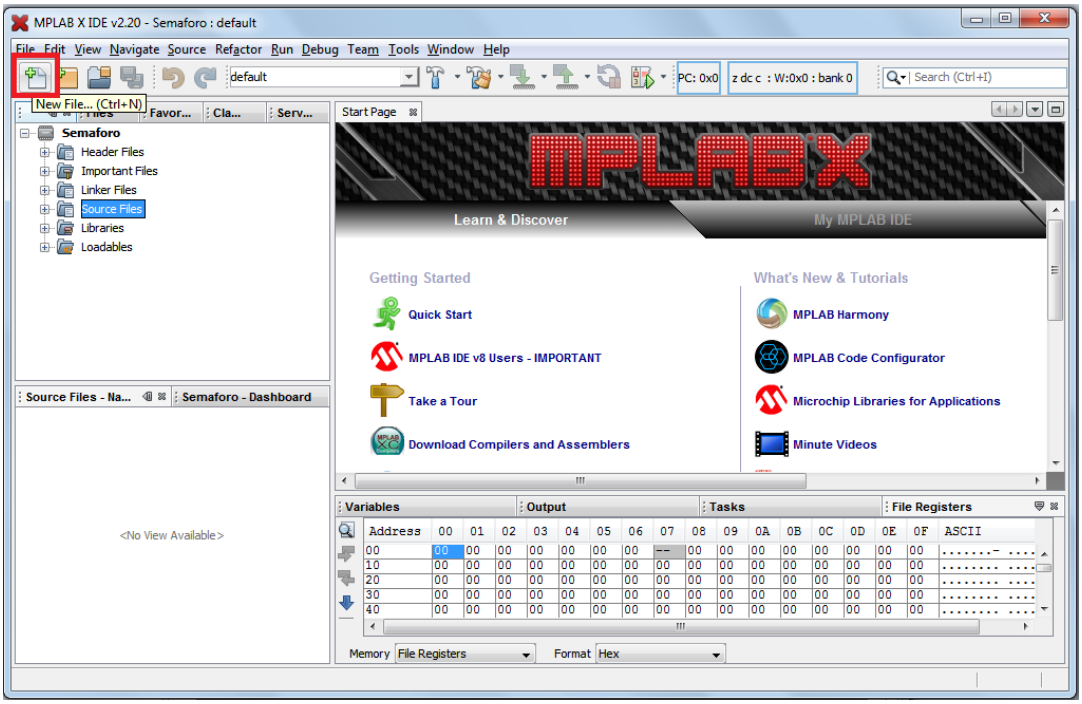

Ilustración 9 - Creación de un nuevo archivo Assembler

Allí se debe seleccionar qué lenguaje de programación se va a utilizar y el tipo de archivo. En nuestro caso, elegimos el lenguaje *Assembler*, seleccionamos el tipo de archivo *.asm* y entonces presionamos *Next*.

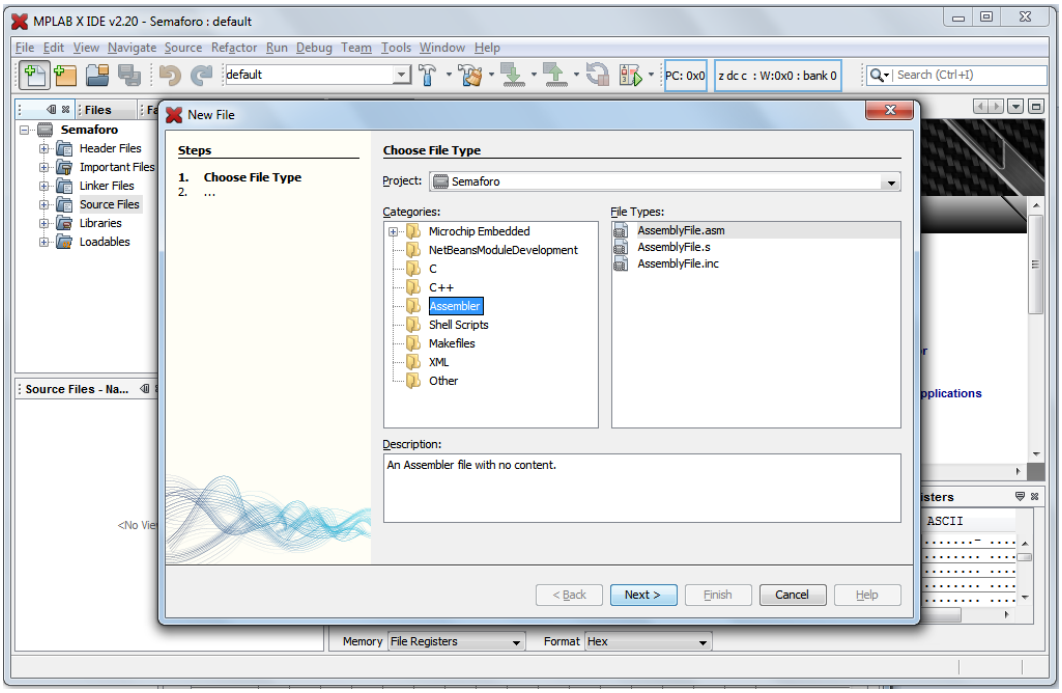

Ilustración 10 - Selección del tipo de archivo de programa (Assembler)

Por último, le ponemos un nombre al archivo en el campo de texto *File Name* y luego presionamos *Finish*. Allí solo resta comenzar a escribir nuestro programa.

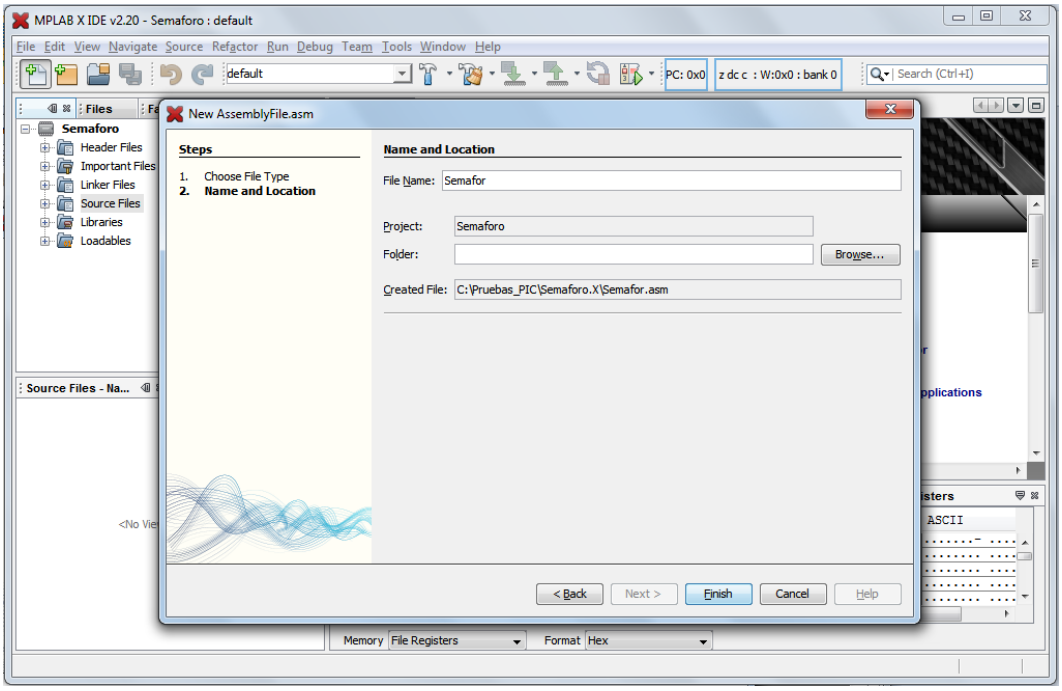

Ilustración 11 - Selección del nombre del archivo

COMPROBACIÓN Y PRUEBA DEL PROGRAMA

Una vez que el programa ya está escrito, necesitamos probar si funciona como corresponde o si contiene algún error. Para ello, debemos ensamblarlo. Después del ensamblado se nos informará si el proceso fue exitoso o si falló porque nuestro código contiene errores. En caso de que haya errores, el reporte nos indicará en qué línea se encuentran y cuál es el motivo.

Para ensamblar el proyecto podemos usar el ícono con un martillo, marcado con rojo en la imagen a continuación, o el menú *Run* → *Build Main Project*.

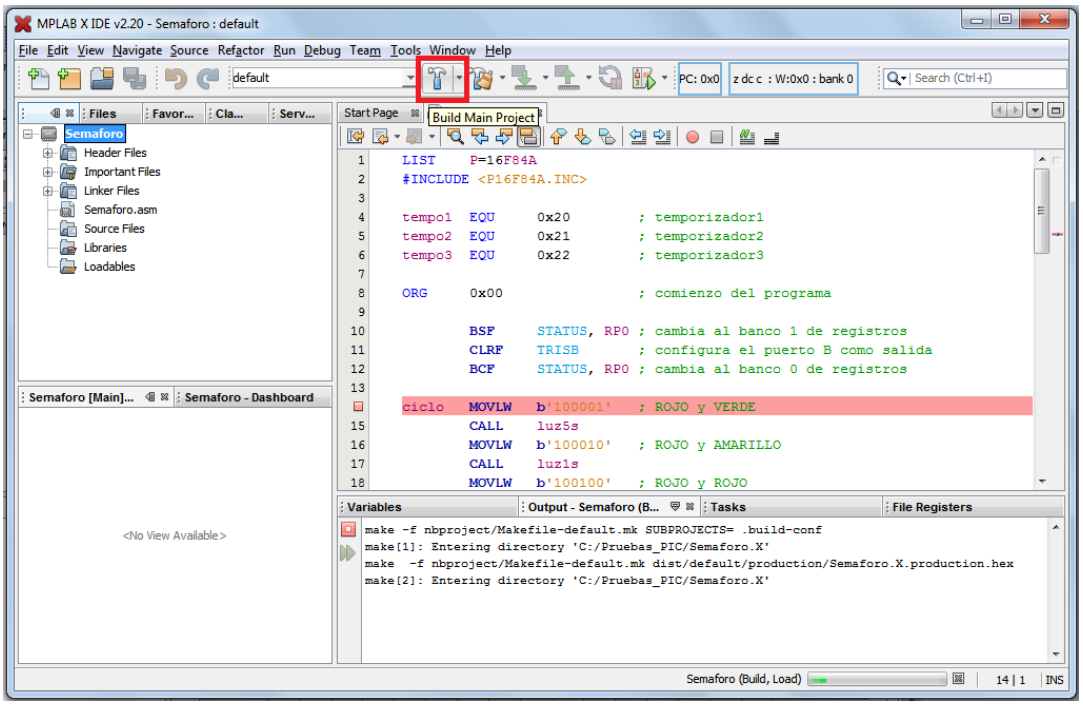

Ilustración 12 - Compilación del proyecto

Si el ensamblado fue exitoso, es decir, si nuestro programa no contiene errores, podemos pasar a probarlo usando el *debugger*. Primero tendremos que configurar ciertas vistas para nos muestren la información relevante.

Para mostrar una pestaña con los registros, iremos a *Window* → *Debugging* → *Variables*

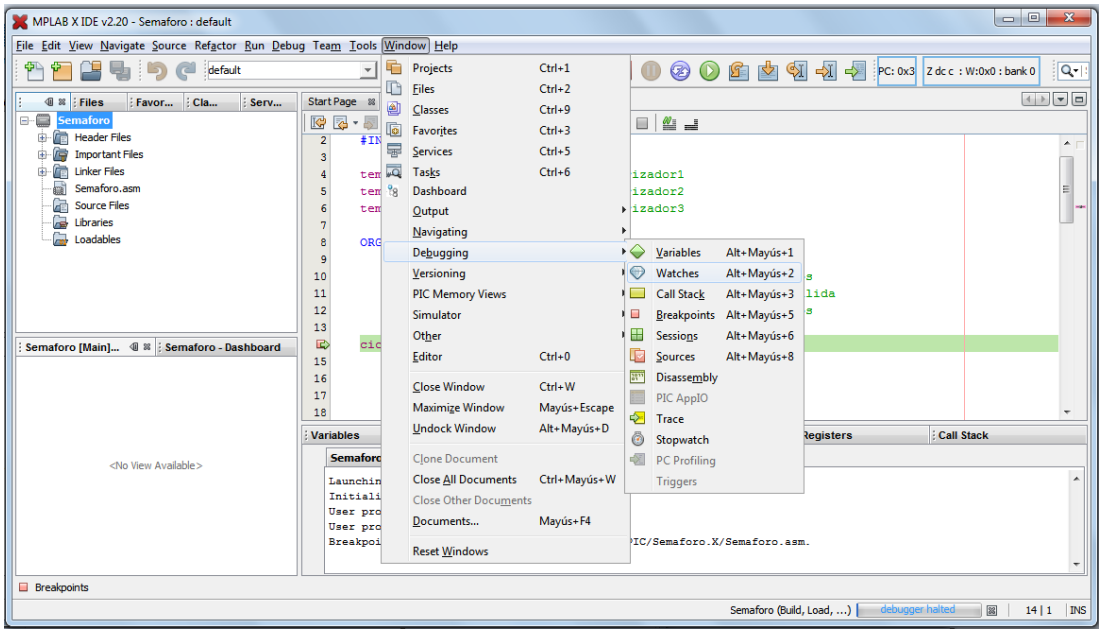

Ilustración 13 - Pestaña de registros

Esto nos permitirá ver los registros del PIC en la parte inferior. Si alguno no aparece, se puede añadir con el último de los íconos de la izquierda. Cada vez que en el programa se modifique uno de ellos, aparecerá remarcado en rojo.

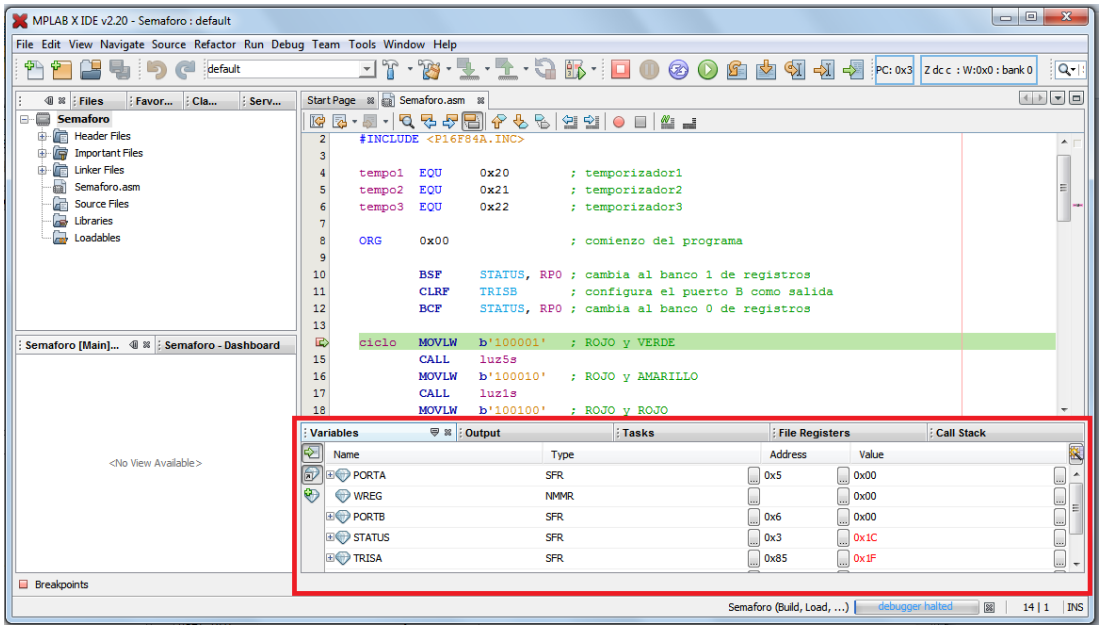

Ilustración 14 - Cuadro con la pestaña de registros

Luego para poder ver la memoria del PIC y cómo modifican su valor las "variables", se debe ir a *Window* → *PIC Memory Views* → *File Registers*

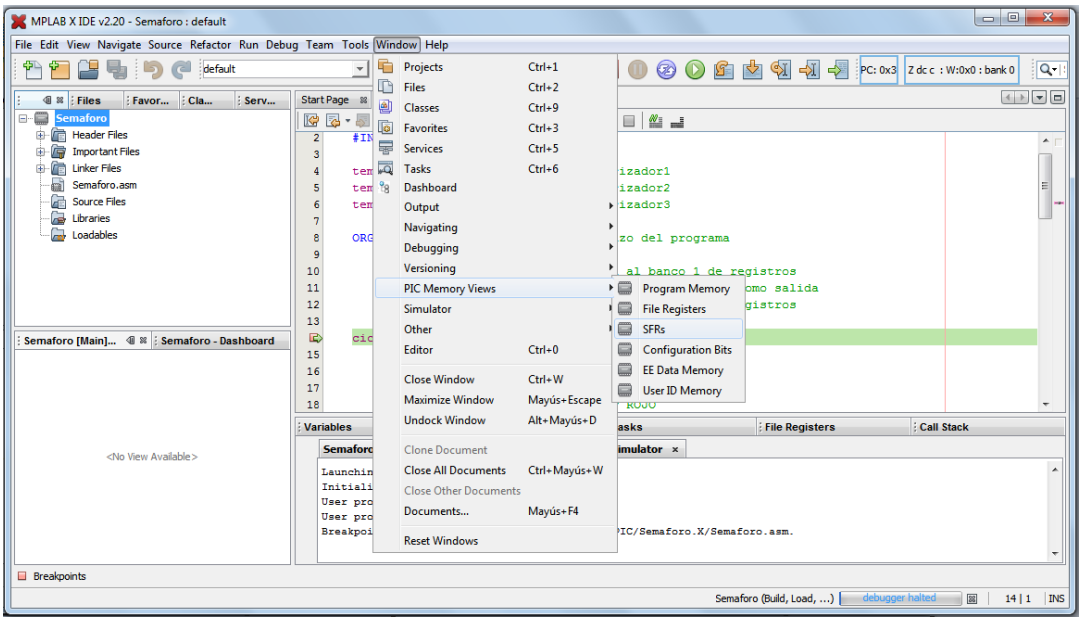

Ilustración 15 - Ventana de memoria del PIC

Allí aparecerá la memoria del programa y podrá verse como se modifican los valores de la misma, indicando de color rojo cuando alguna celda es modificada.

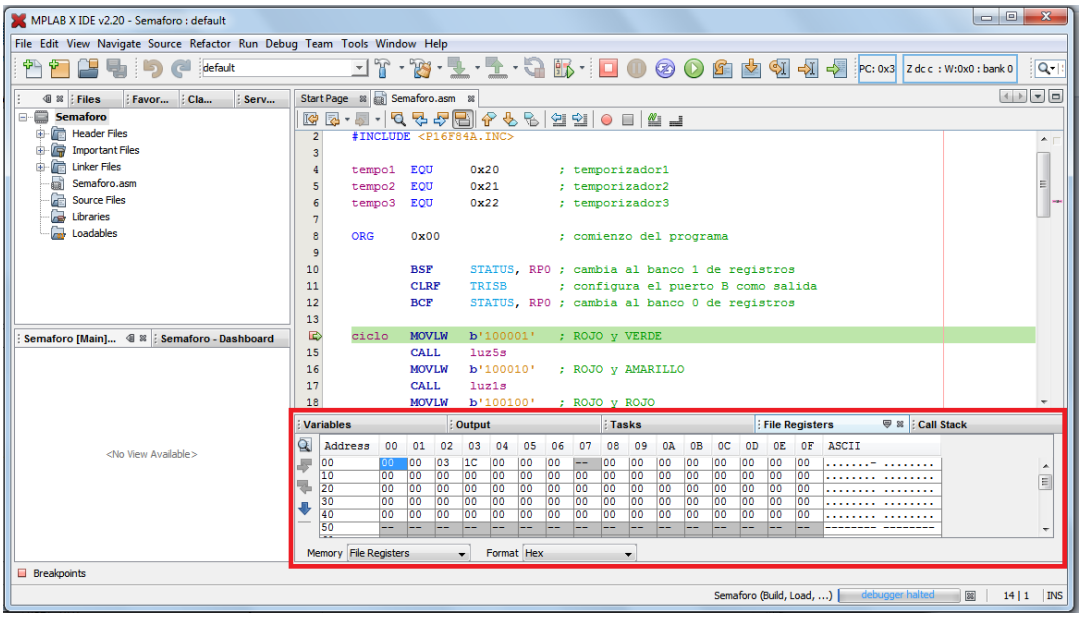

Ilustración 16 - Cuadro con la pestaña de memoria del PIC

Una vez que se tienen estas pestañas, podemos empezar a depurar (*debuggear*) nuestro programa, teniendo en cuenta algunos aspectos.

Por lo general, los programas no tienen una condición de finalización, con lo que se ejecutan todo el tiempo. Si queremos controlar su funcionamiento, tendremos que colocar algún *breakpoint* para que al llegar a ese punto se frene

y poder ejecutarlo paso a paso. Para ello, podemos hacer doble clic sobre el número de la línea del programa en donde queremos frenar, o posicionar el cursor en la línea y presionar CTRL+F8, o ir a *Debug → Toggle Line Breakpoint.*

También debemos considerar que si vamos a utilizar algún puerto de entrada, no podremos controlar qué valor tiene el mismo, con lo cual deberemos simular los diferentes valores a mano, haciendo una asignación directamente al registro W y trabajar con ese valor ingresado.

Una vez realizadas estas salvedades y colocado el o los *breakpoints* para depurar el programa, se debe presionar el ícono marcado en rojo en la imagen a continuación, o a través del menú *Debug → Debug Main Project*.

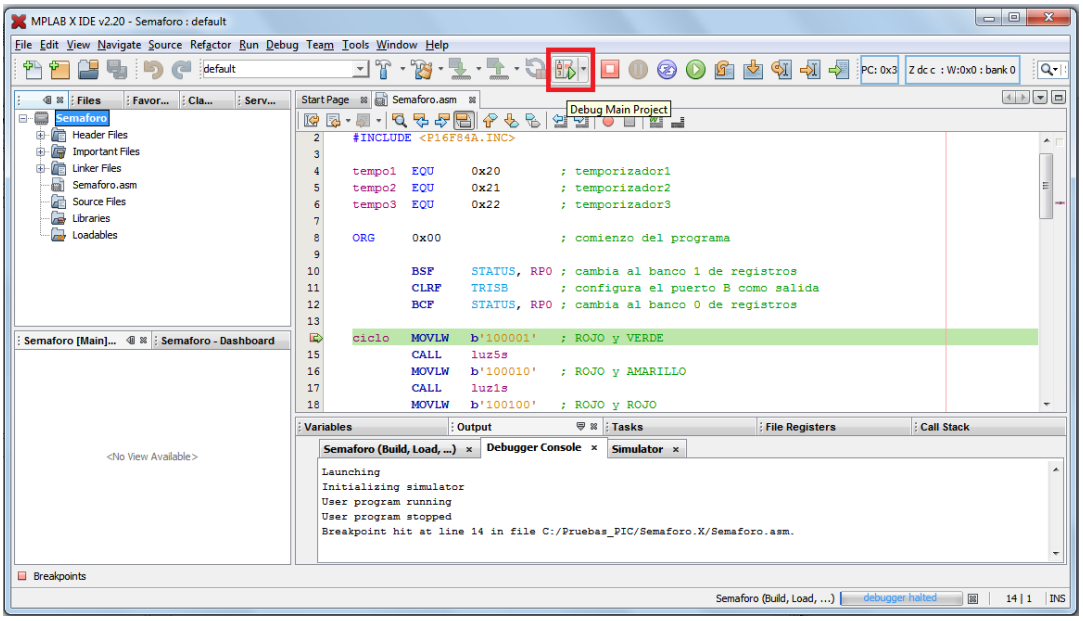

Ilustración 17 - Debug del programa

De este modo, comenzará a ejecutarse el programa y se frenará en la línea en la que fue colocado el *breakpoint*. Allí se podrá comenzar a ejecutar el código paso a paso con la tecla F7 o a través del menú de íconos que aparece arriba del código del programa, como se puede ver en la imagen a continuación.

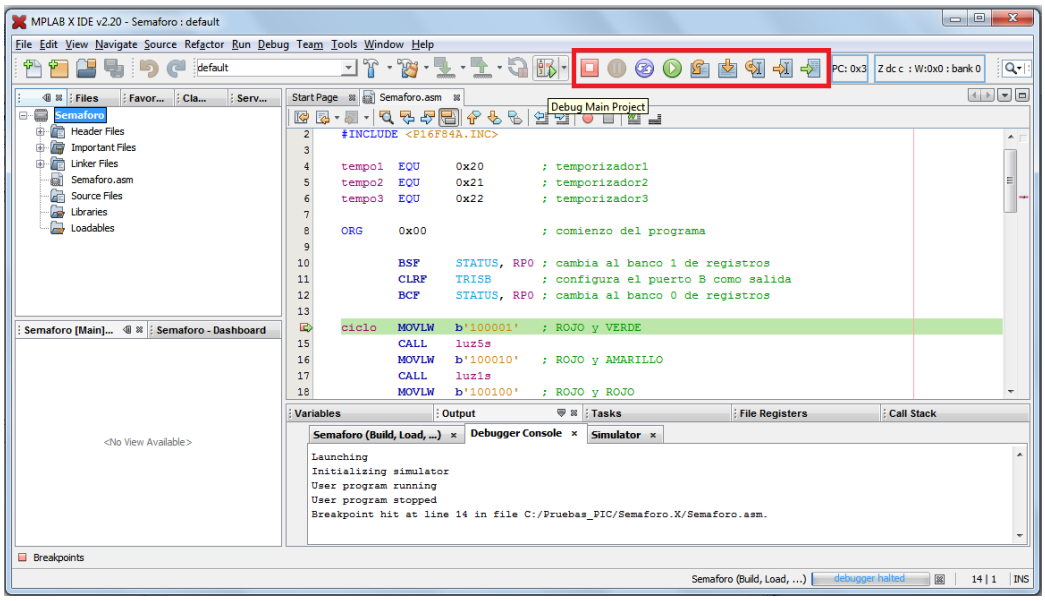

Ilustración 18 - Botones de ejecución de instrucciones para *debugging*

SIMULACIÓN EN PROTEUS

Una vez controlado, el programa se puede pasar, mediante alguno de los programadores, directamente al PIC para el que fue programado.

Otro modo de probar el programa realizado, que es el que vamos a utilizar en este curso, es a través de una simulación en la aplicación Proteus. Esta aplicación permite armar circuitos electrónicos, incluyendo microcontroladores y simular su funcionamiento del mismo. Veremos cómo hacerlo.

A continuación se puede ver la ventana de Proteus con un circuito que tiene un microcontrolador PIC16F84A con 6 LEDs conectados al Puerto B.

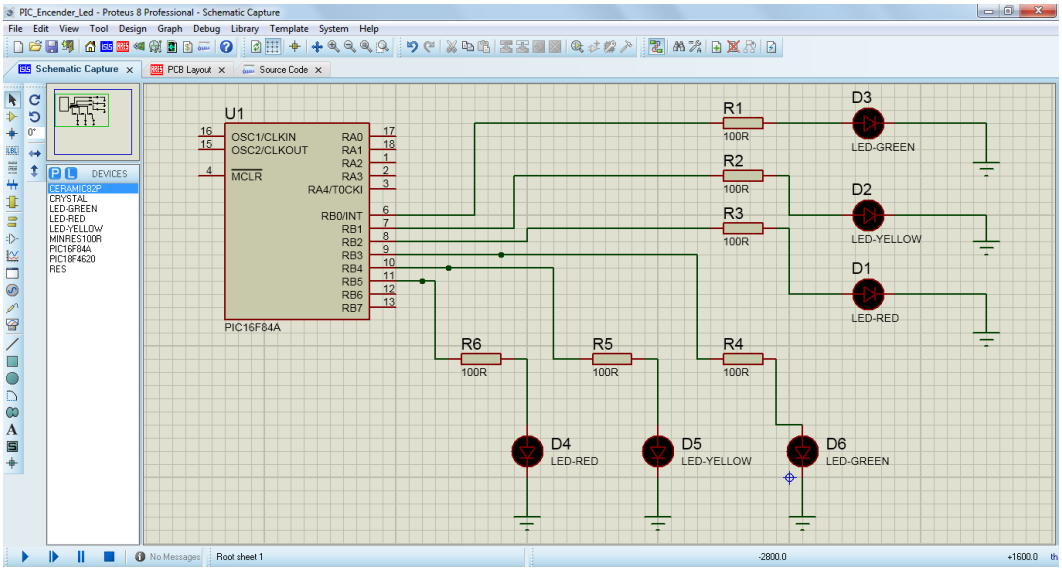

Ilustración 19 - Aplicación Proteus con un circuito electrónico cargado

No es el objetivo de este manual explicar cómo realizar los circuitos en Proteus sino, utilizando circuitos prearmados, ver cómo cargar programas en el microcontrolador para simular su funcionamiento de manera más real. Para ello, al hacer doble clic sobre el microcontrolador aparecerá un cuadro con las características y configuración del dispositivo, como se ve a continuación.

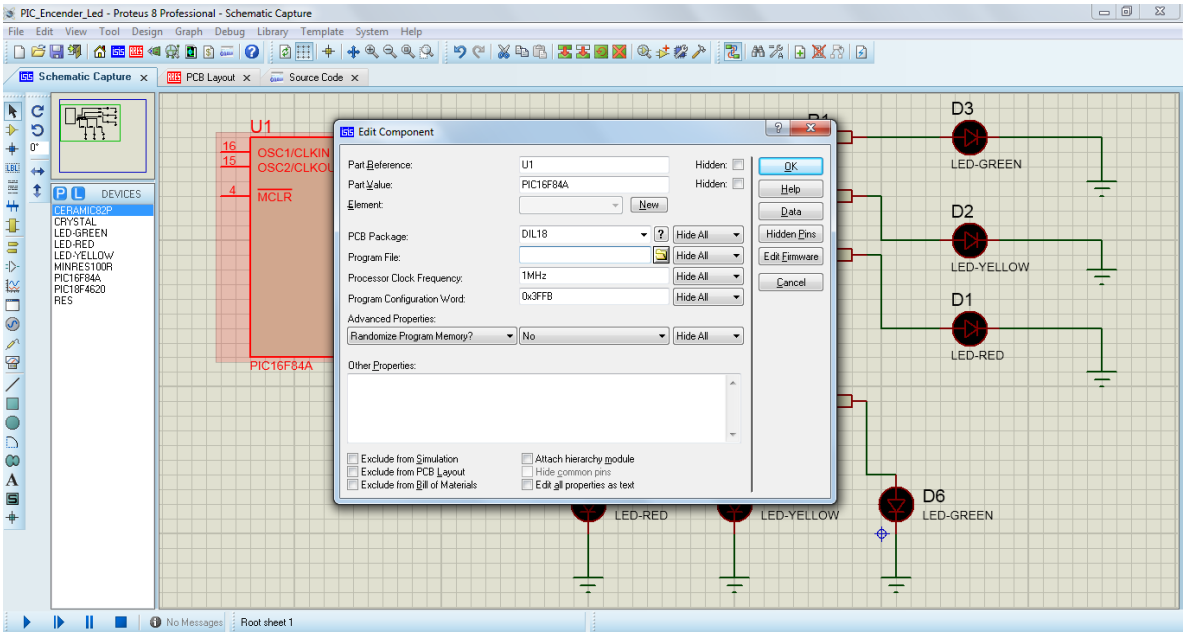

Ilustración 20 - Cuadro de configuración de PIC

Allí se puede cargar en el PIC el programa realizado en MPLAB X IDE. Para ello debemos tener el binario, es decir, el archivo con extensión .HEX, que es nuestro programa realizado en el IDE pero ensamblado para ser ejecutado por el microcontrolador. Este archivo se genera cuando se ensambla el programa en MPLAB y queda guardado en la carpeta del proyecto en las siguientes subcarpetas:

"DirectorioProyecto\NombreProyecto.X\dist\default\production\Programa.HEX"

Dicho programa debe ser cargado en el campo *Program File* que se puede ver en la imagen anterior y luego presionar *Ok*.

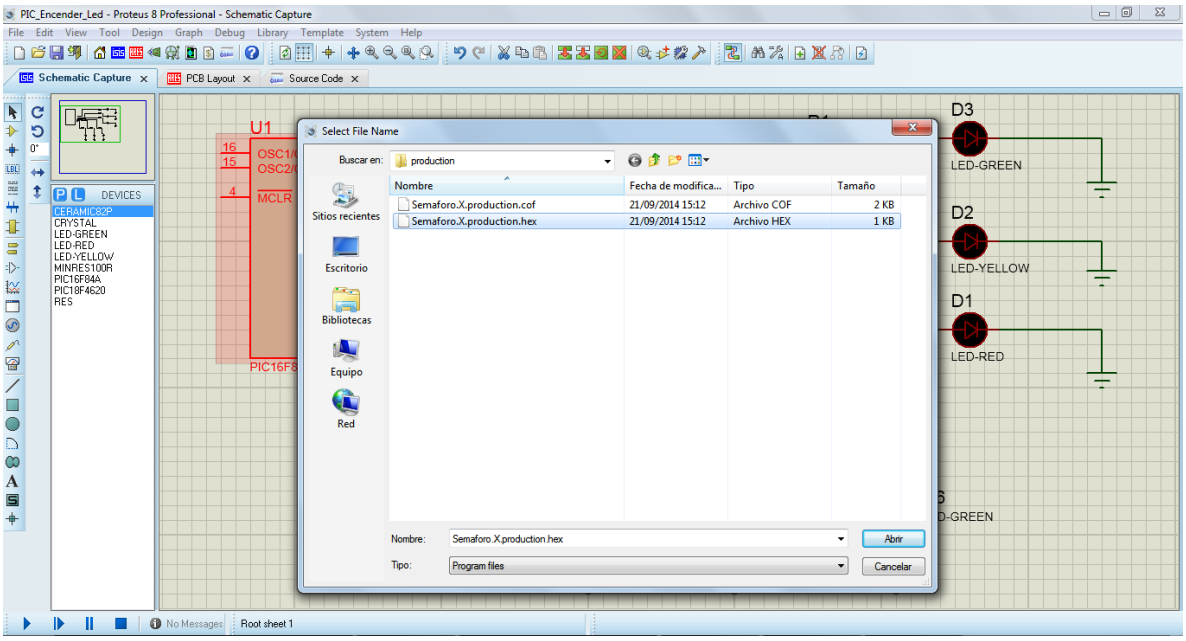

Ilustración 21 - Selección de programa para PIC

Una vez seleccionado allí solo queda ejecutarlo para ver si funciona como se espera. Para ello en la parte inferior izquierda del programa se encuentran una serie de botones de control, con un botón de play, uno de pausa y uno de stop el cual permitirá ejecutar el programa, pausarlo o pararlo.

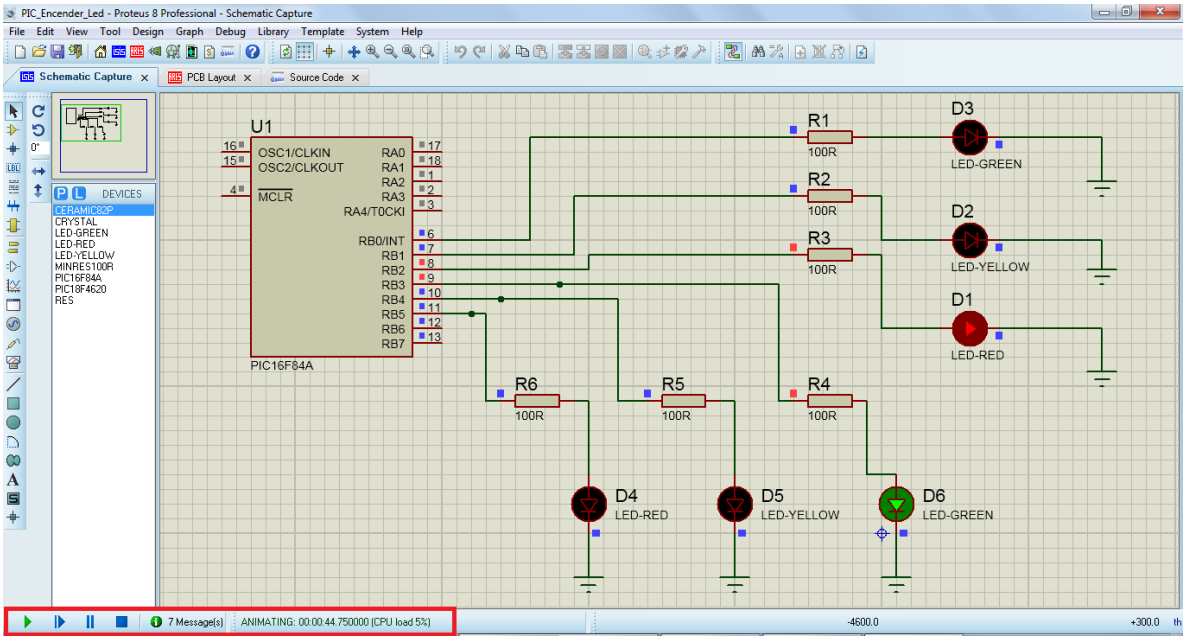

Ilustración 22 - Ejecución del programa dentro del circuito