MPLAB X IDE TUTORIAL

The aim of this tutorial is to show how to use MPLAB X IDE.

Introduction to MPLAB X IDE

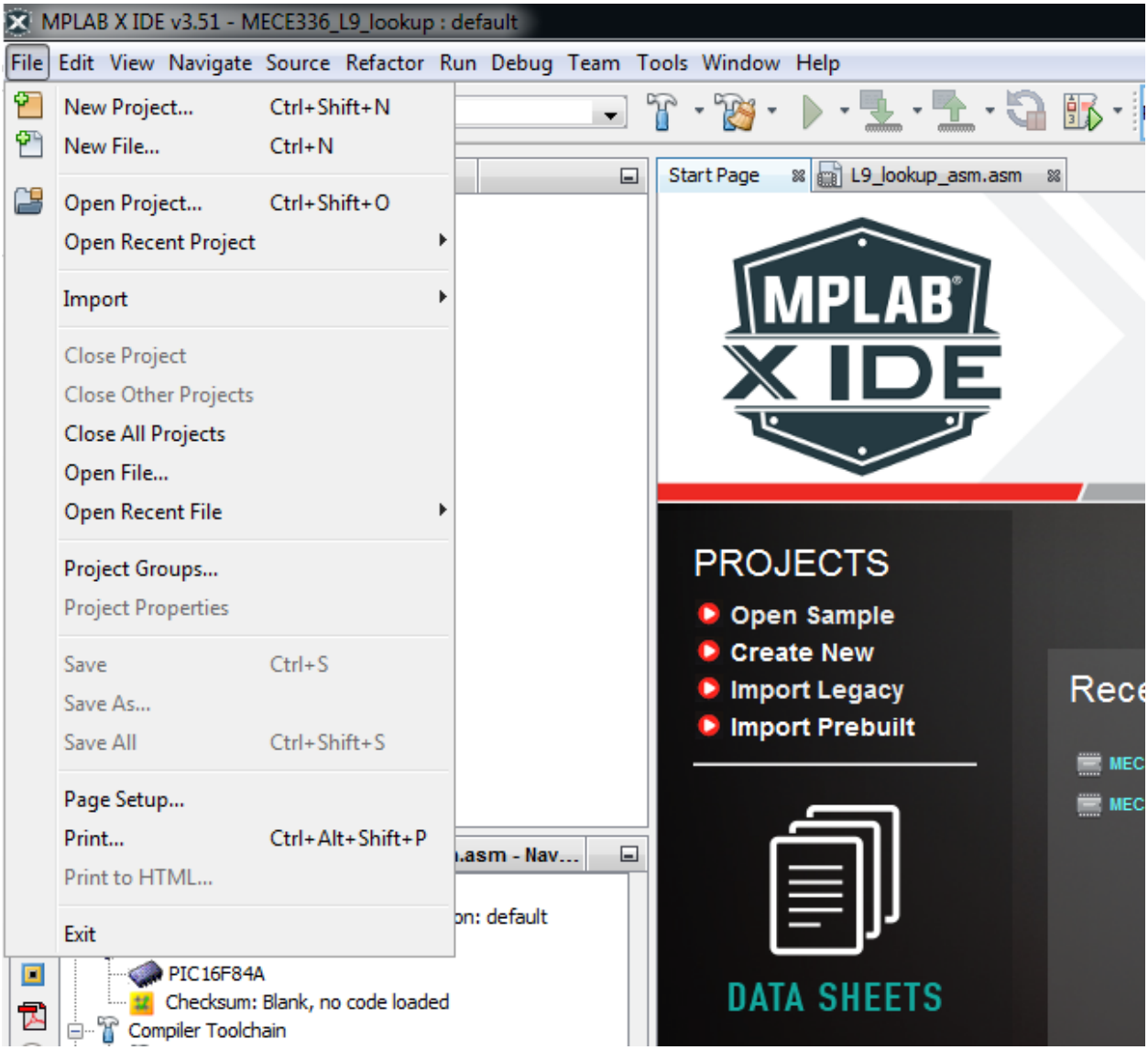
MPLAB X IDE has three main components that work together to generate machine code to be loaded on a PIC microcontroller:

- **Project Manager**. The preferred way of developing programs in MPLAB is by creating a project. An MPLAB project groups all the files together that relate to the project and ensures that they interact with each other in an appropriate way and are updated as needed.
- **Text Editor**. This allows entry of the source code. It behaves to some extent like a simple text editor such as Notepad, but it can recognise the main elements of the programming language that is being used. Thus in Assembler it codes instructions in one colour, labels in another and comments in a third. In this way the programmer can immediately see if there is a misconception in his placing or use of text within the Assembler line.
- **Assembler and Linker**. The Assembler converts the Assembly language to machine language. In advanced projects, the code may be created from a number of different files. The role of the Linker is to put these together, give each its correct location in memory, and ensure that branches and calls from one file to the other are correctly established.

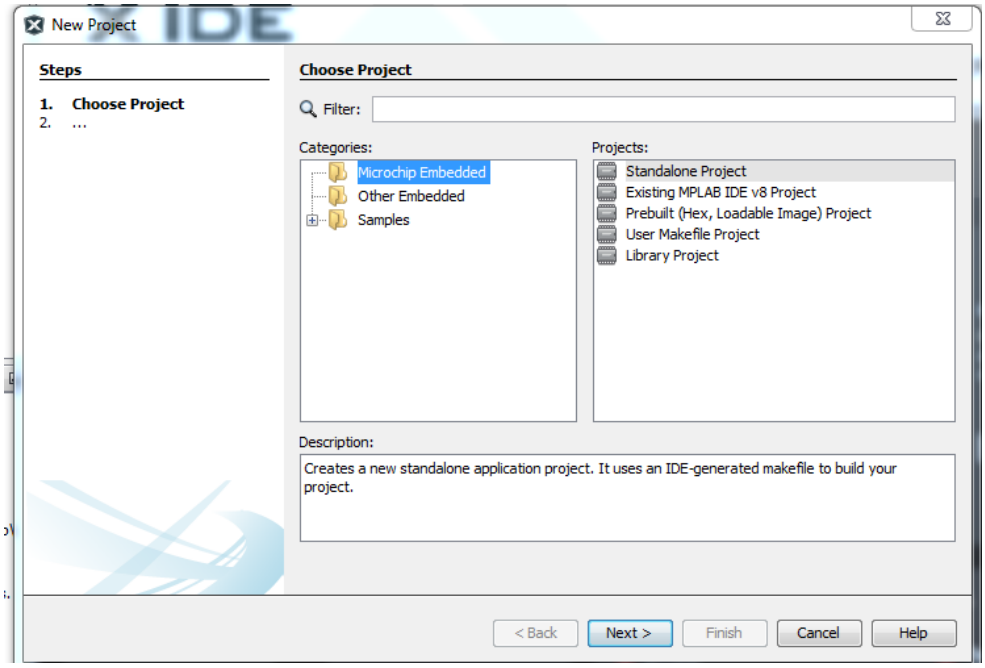
Creating a Project in MPLAB X IDE

1) Open MPLAP IDE. Icon of the editor can be seen in **Figure-1**.

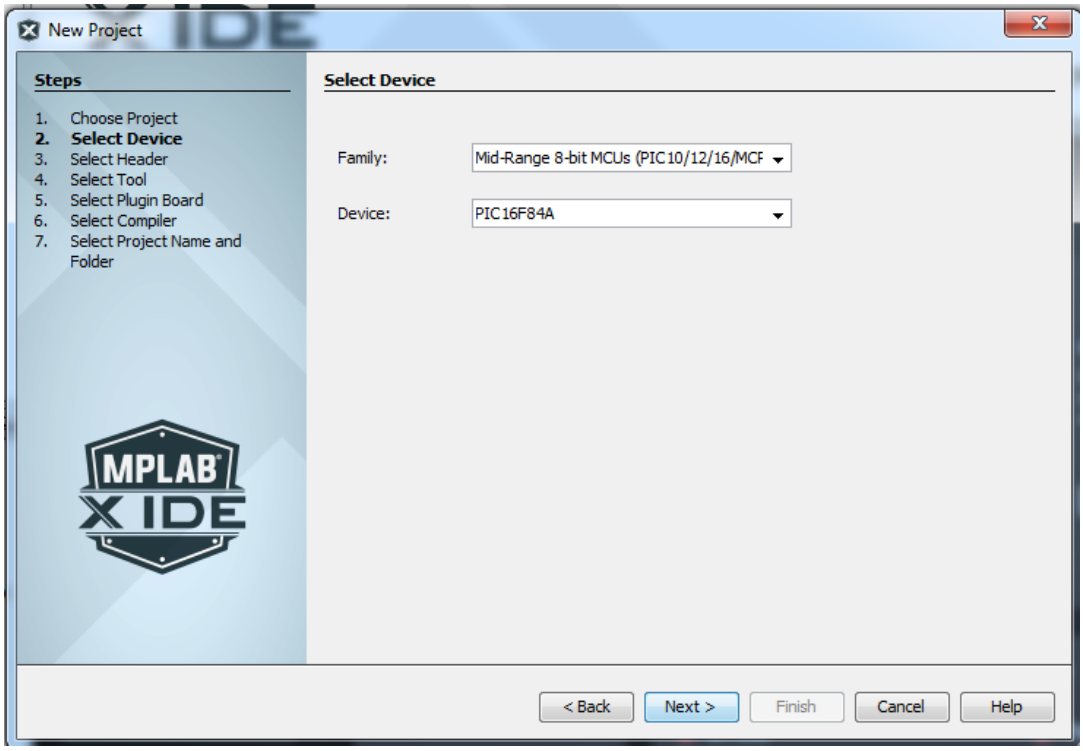
2) From the **"Project"** choose **"New Project"**.



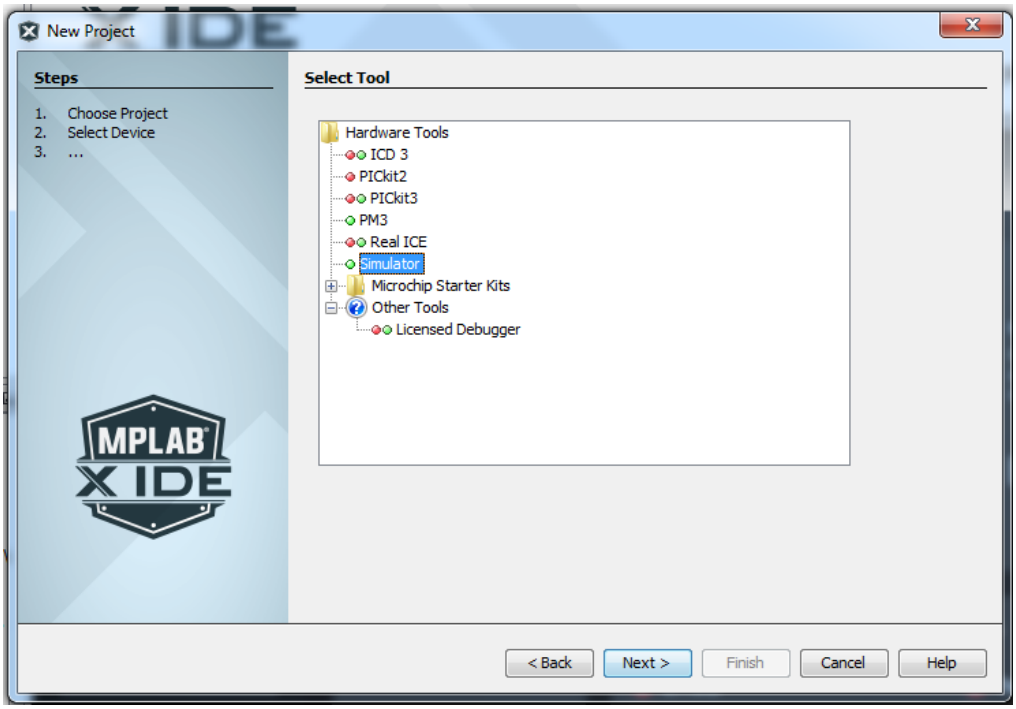
3) Select **Microchip Embedded** (Categories) and **Standalone Project** (Projects). Then click **Next**.



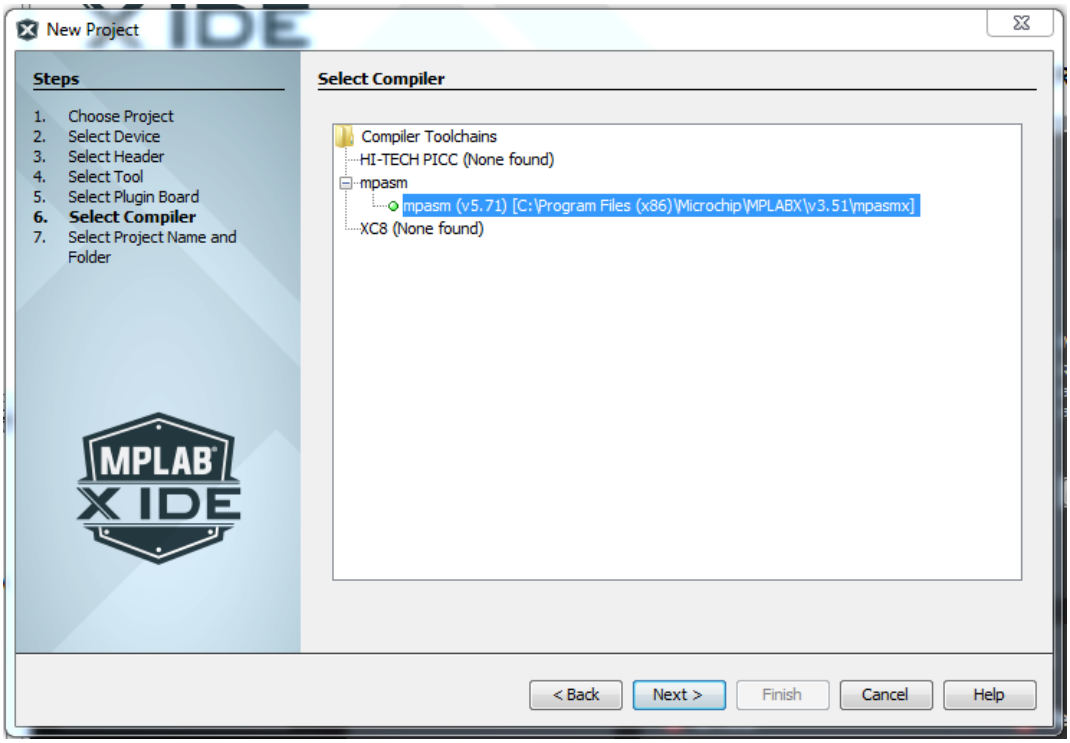
4) Select the device. In our case choose the family Mid-Range 8-bit MCUs and the Device **PIC16F84A.** Click "**Next**".



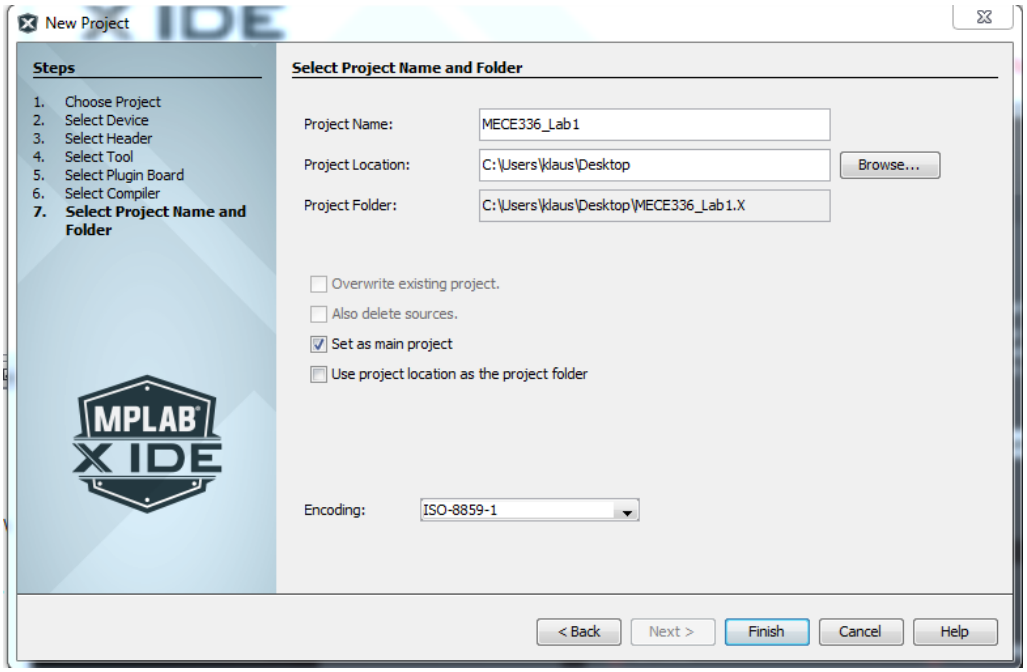
5) Choose the Hardware Tool. If we want to use the MPLAB Simulator, select **Simulator**.



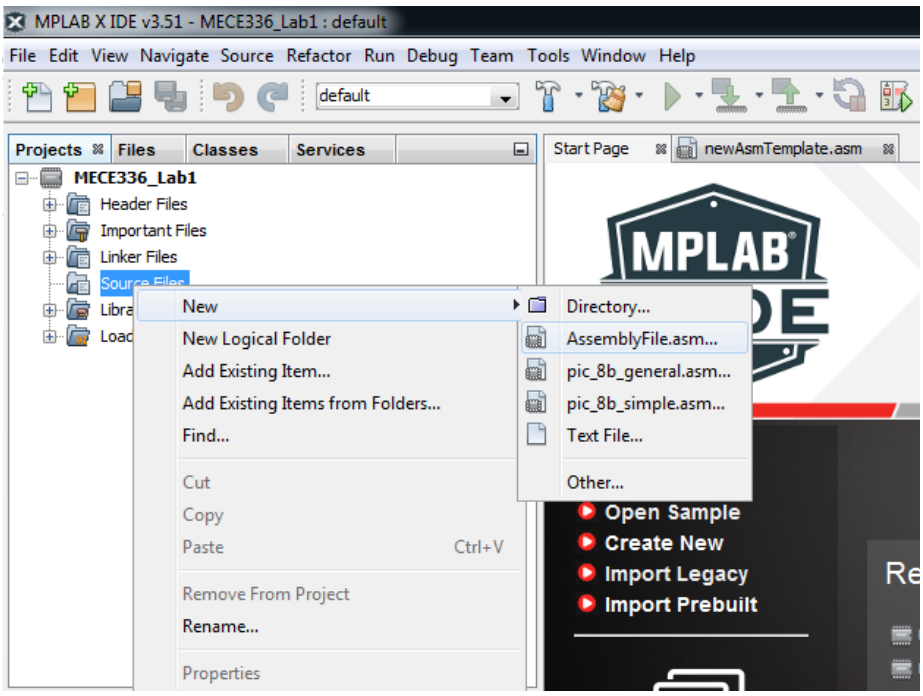
6) Choose the programming language. If you want to use Assembly choose **"mpasm",** then click **"Next"**.



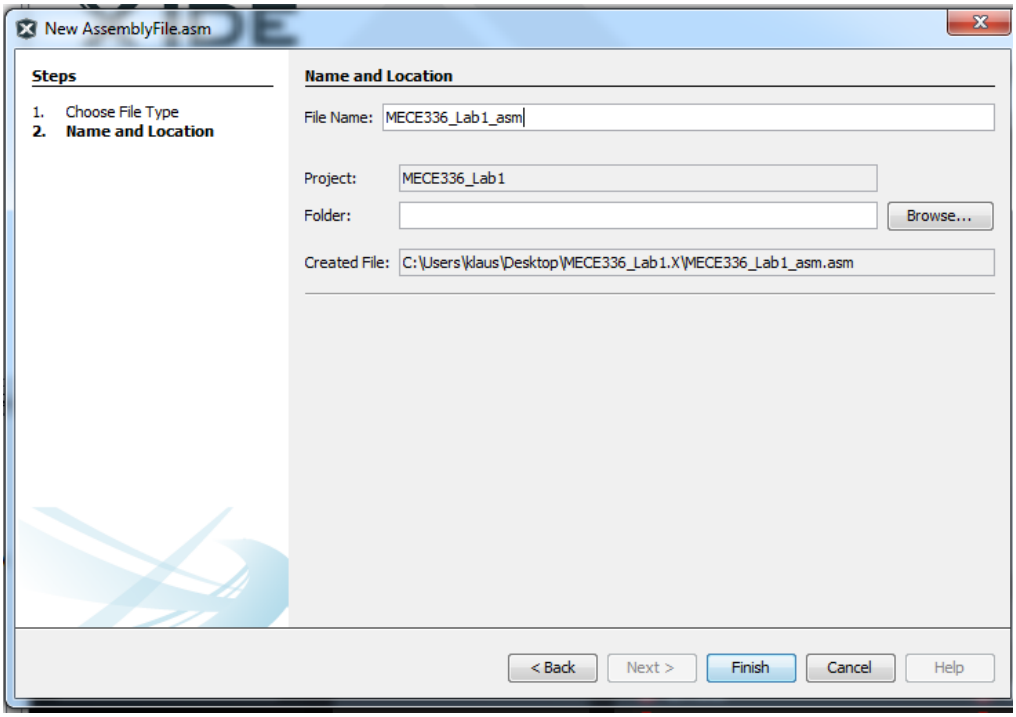
7) Choose a directory and a name for the project. Click **"Finish"**.



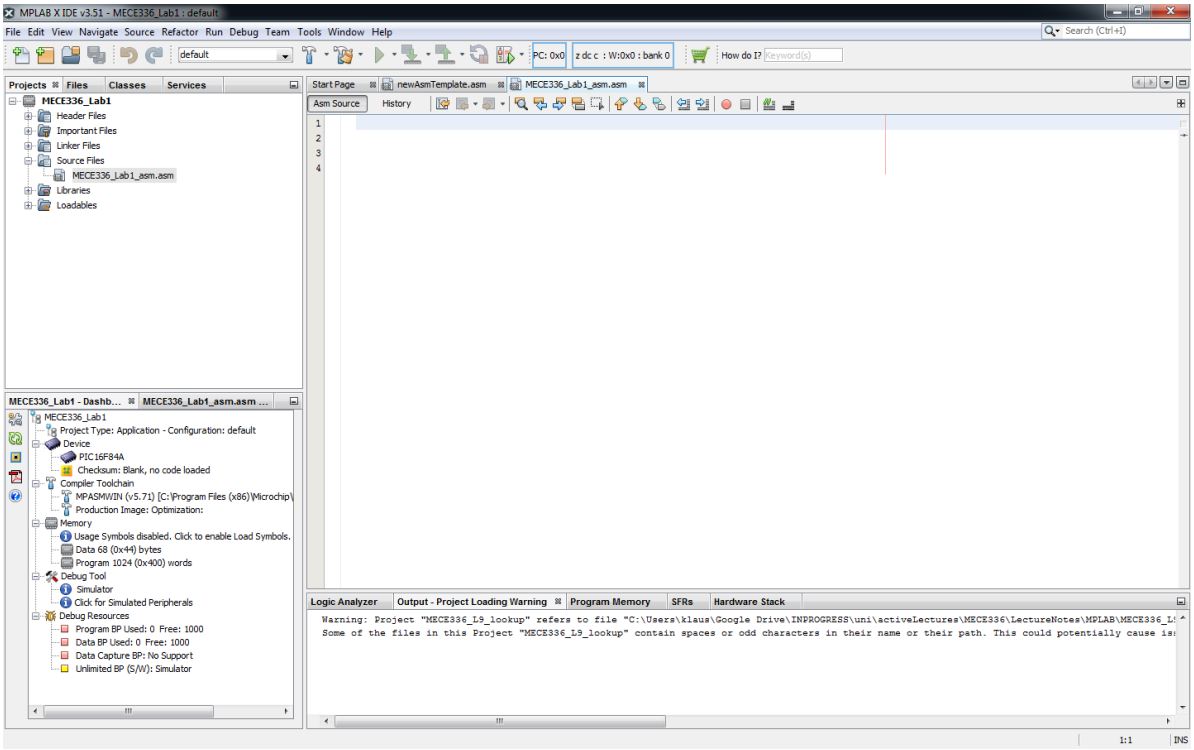
8) A new project is created. Your window should look like in the next figure. The next step is to write our assembly code. Expand the tree under Projects and select **Source Files -> New -> AssemblyFile.asm.**



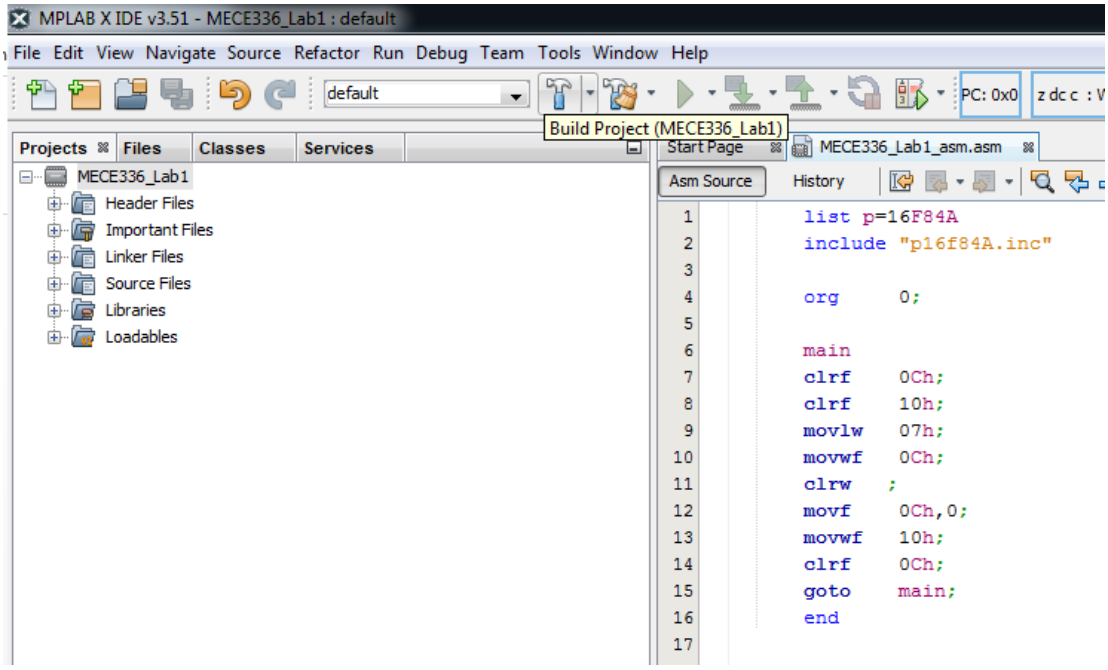
9) Write a file name and click **Finish**.



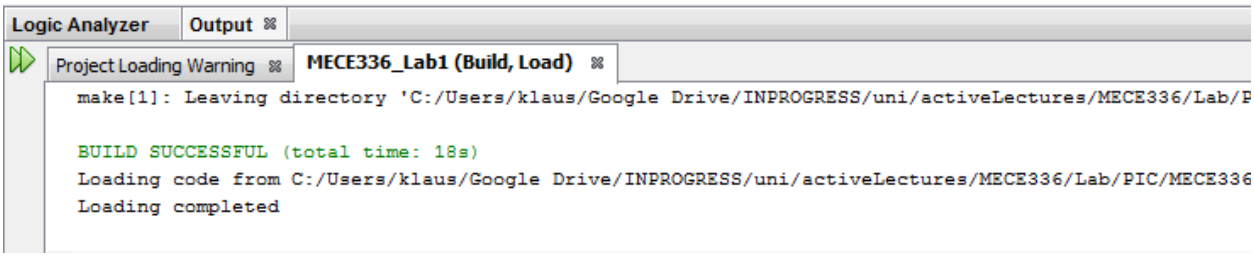
10) Your project window should now look as in the following figure. You can start writing assembly code.



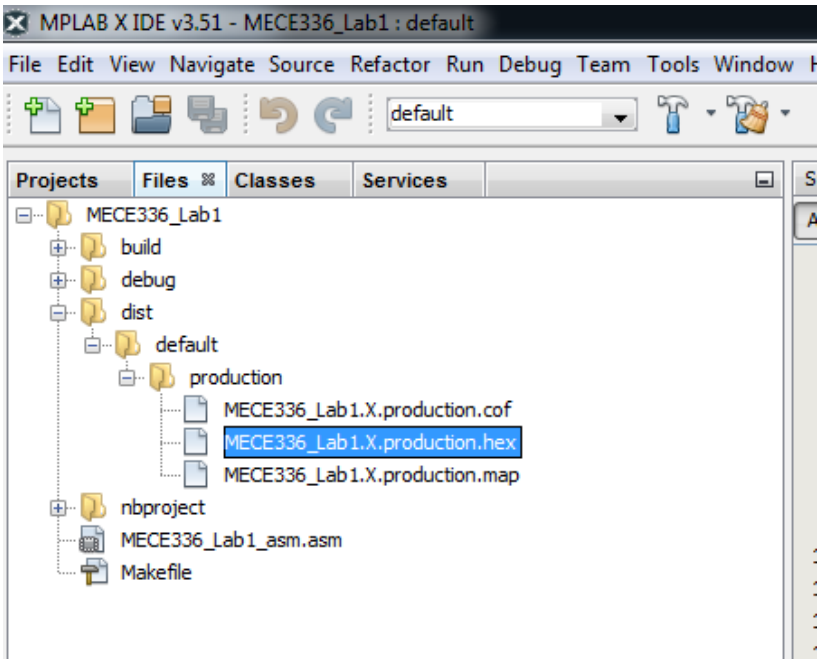
11) Special commands in the written code are colored as blue and keywords are colored as magenta. You can now compile your code by clicking **Build Project**.



12) If your code is correct in terms of assembly syntax rules then the output window shows "BUILD SUCCESSFUL". The "**.hex**" file is created and we are ready for the simulation.



13) If you want to look at the .hex file, you can find it under **Files->ProjectName->dist->default->production**.



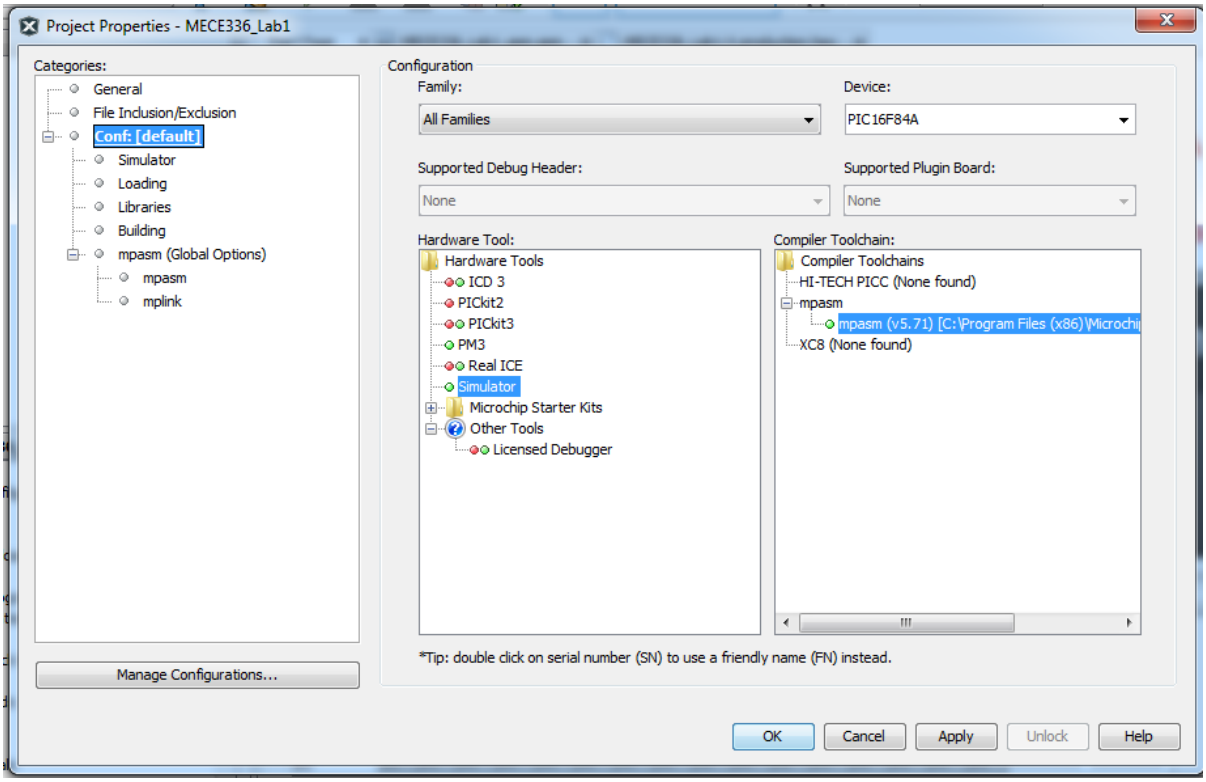
B Simulate your assembly code in MPLAB X IDE

MPLAB X IDE contains a **Software Simulator and Debugger**. A software simulator allows a program to be tested by running it on a simulated CPU in the host computer. Inputs can also be simulated and outputs and memory values can be observed. The debugger contains the tools which allow program execution to be fully examined, for example by single stepping through the program, running at slow speed, or halting at a particular location.

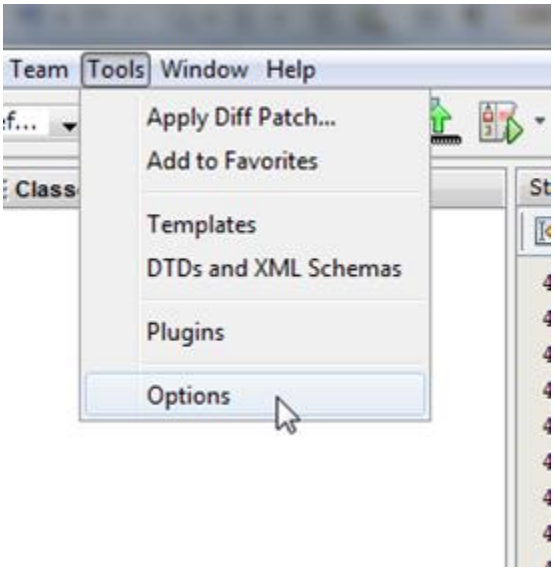
1. Open a Project as described above

2. Select Simulator as the Hardware Tool

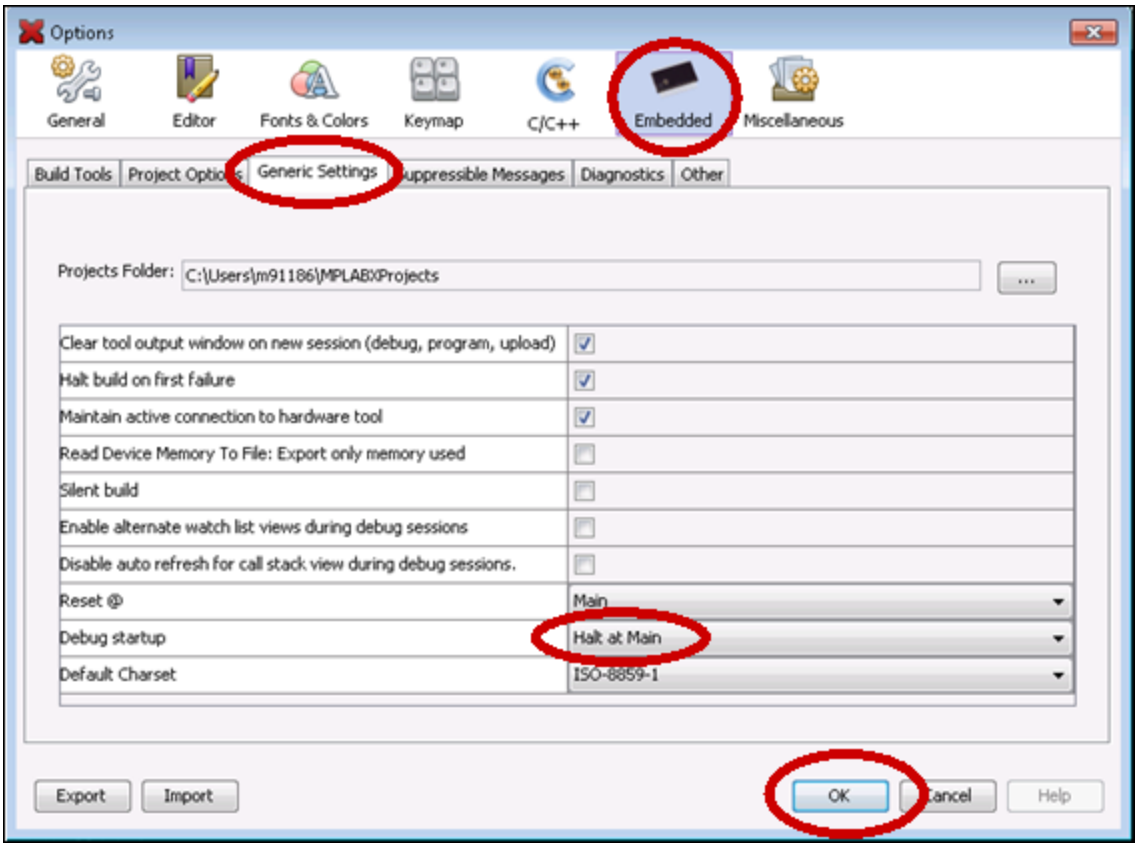
- Select MECE336_Lab1 from the project list
- Right click
- Select “Properties”
- Click Conf:[default]
- Under “Hardware Tools”, verify that Simulator is selected



3. Setting the Simulator to start at the beginning of main() function

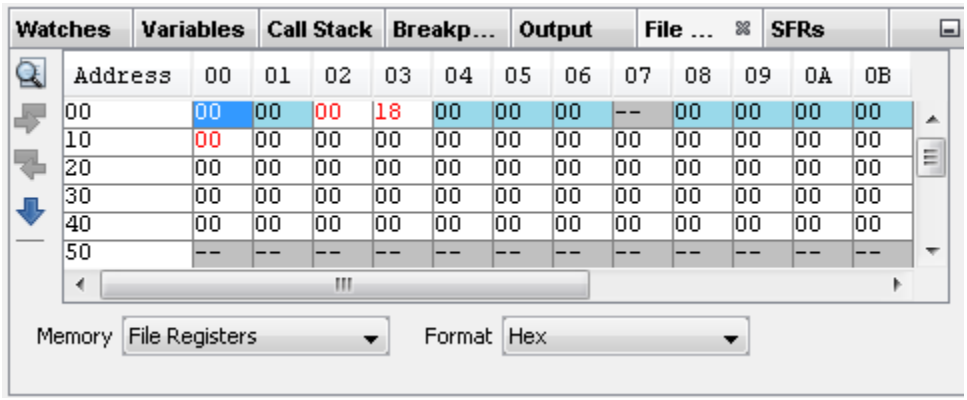


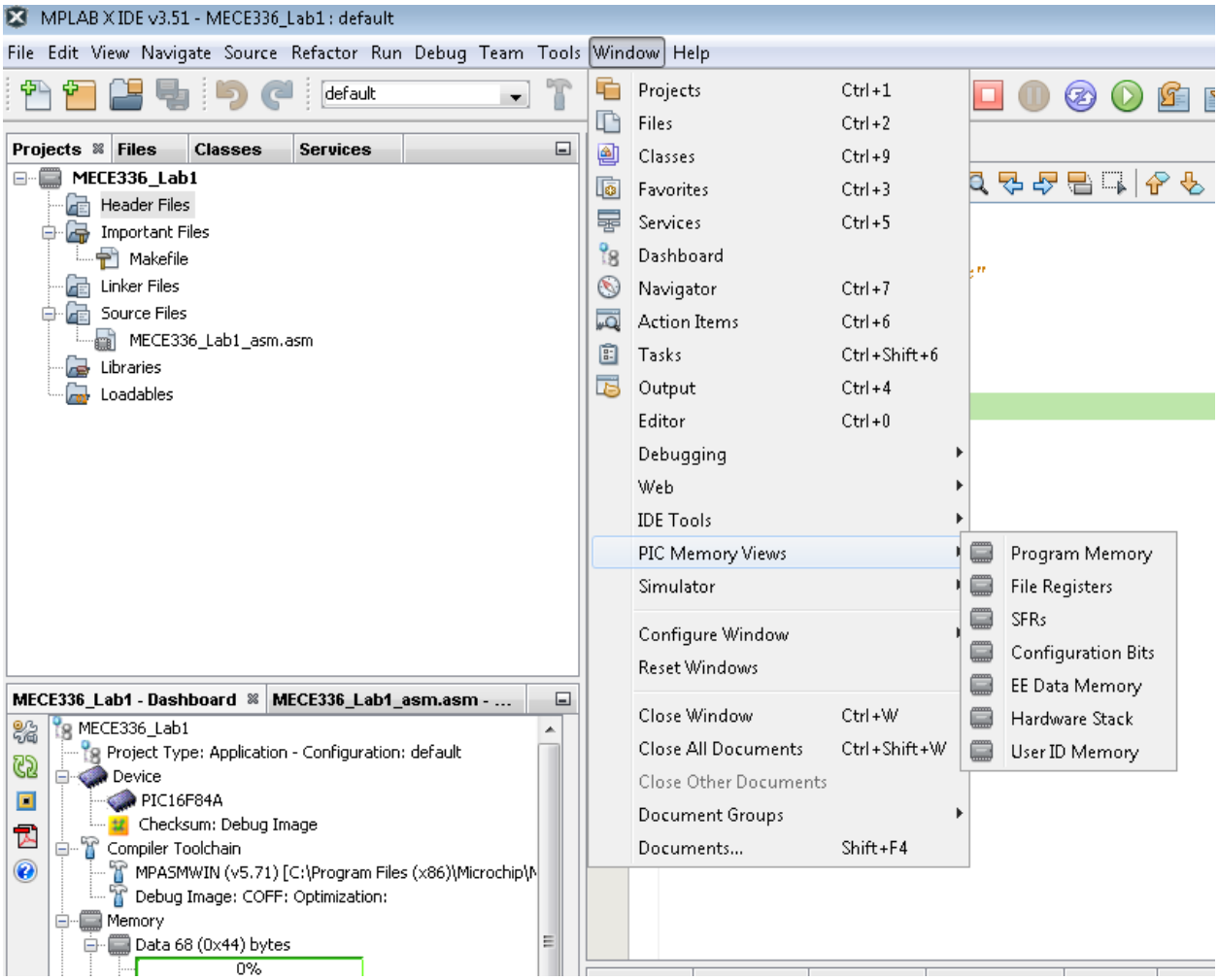
- From the “Tools” pull down menu select “Options”
- Select the Embedded icon
- Select the “Generic Settings” tab
- Ensure the “Debug startup” is set to ‘Halt at Main’



4. Select Memory Locations to Observe

- Select Window->PIC Memory Views from the Menu bar
- You can for example observe the file registers, special function registers (SFRs) or the program memory
- The figure below shows the file register starting from address 0x00.

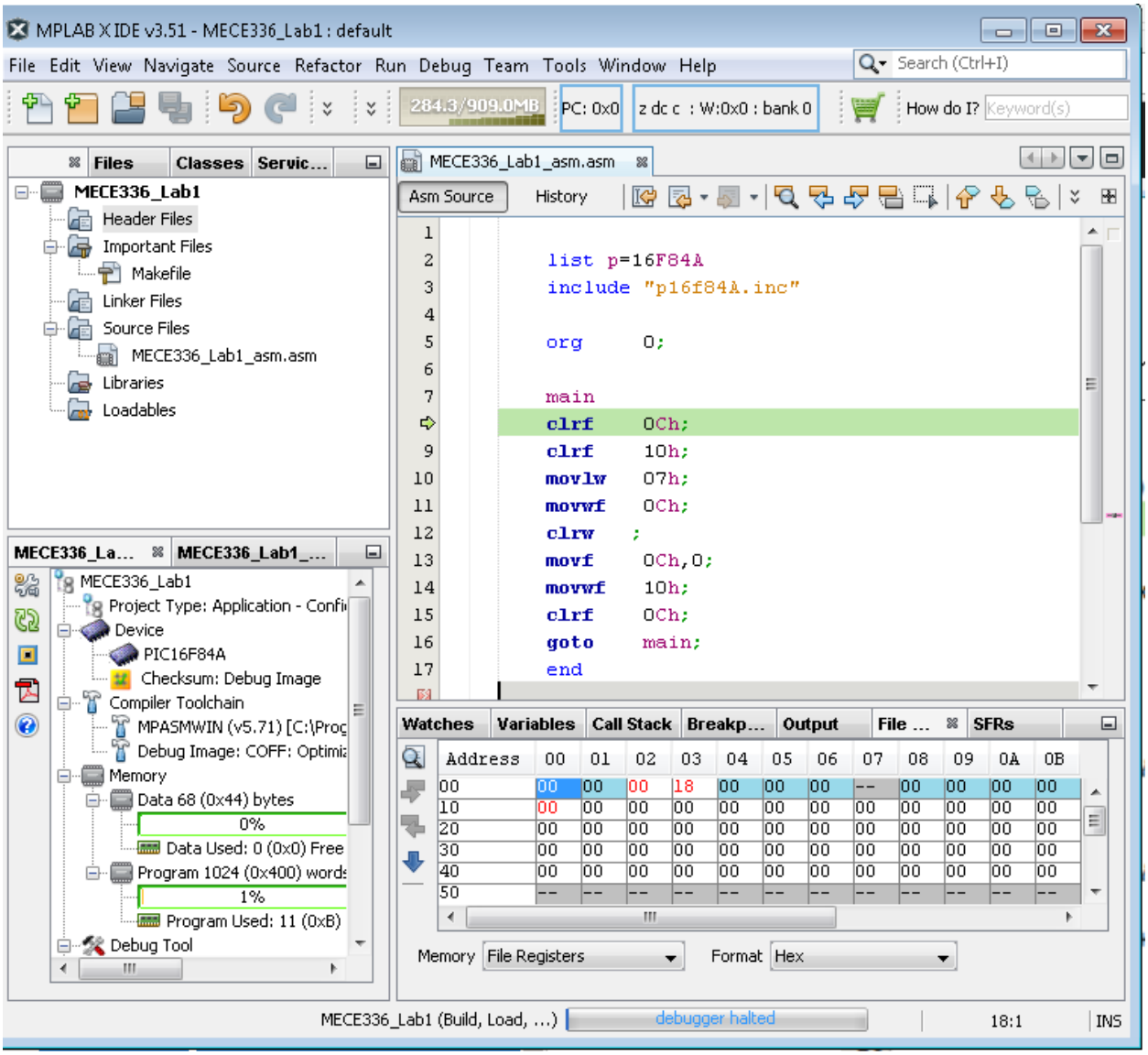




5. Starting the Debug Session

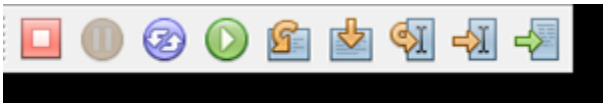
Build a debug version of the open project by clicking on the Debug Project icon 

You will notice that clicking the “Debug Project” icon will build the project, download it to the simulator, start the simulator session, and run until the program reaches “main”.



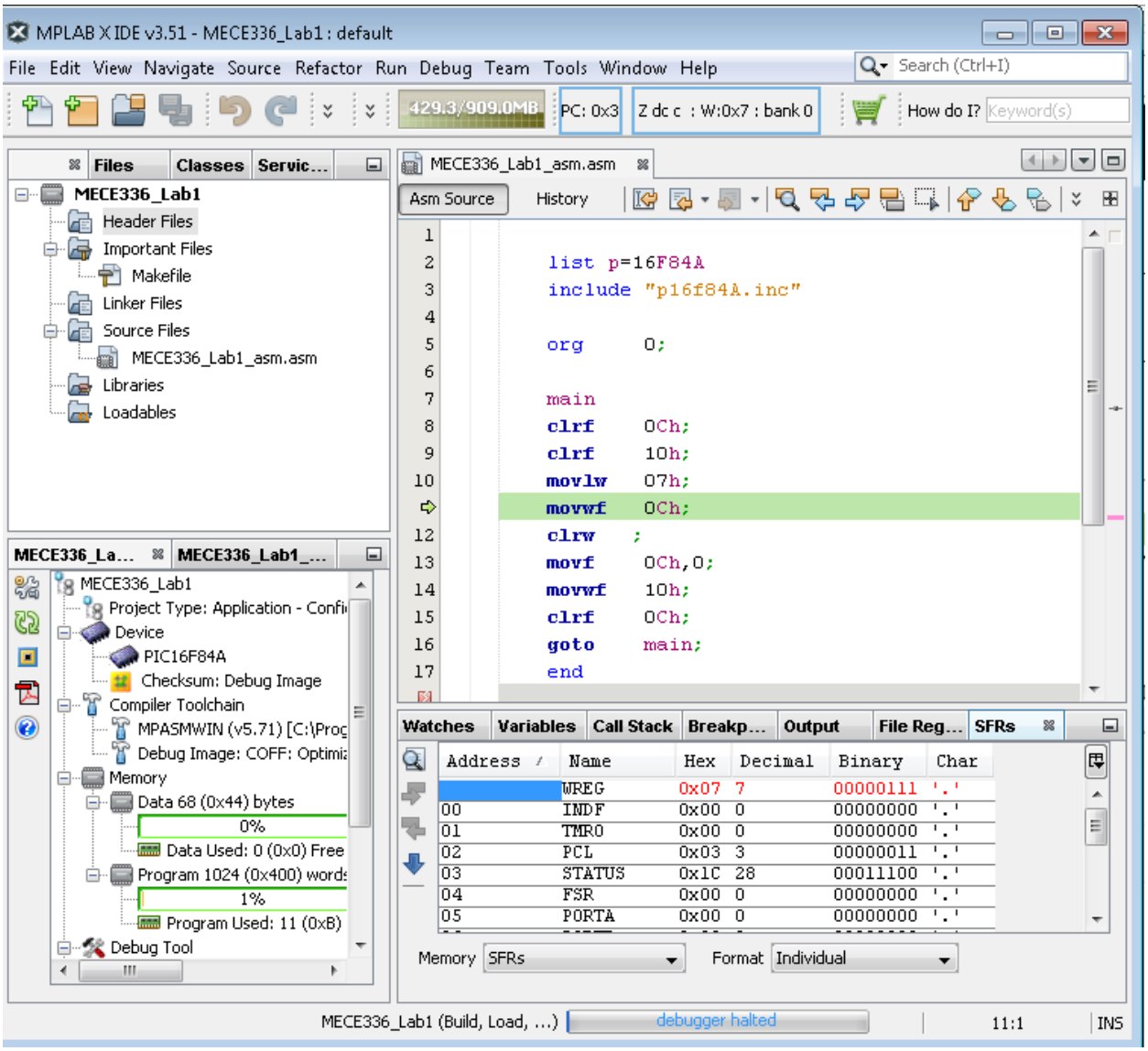
Observe the green code highlight on the first line after main. The simulation has started but stopped at this place in the code. The green line indicates the current position of the PC during a debug session.

Notice the addition of the Debug Toolbar



6. Step Into

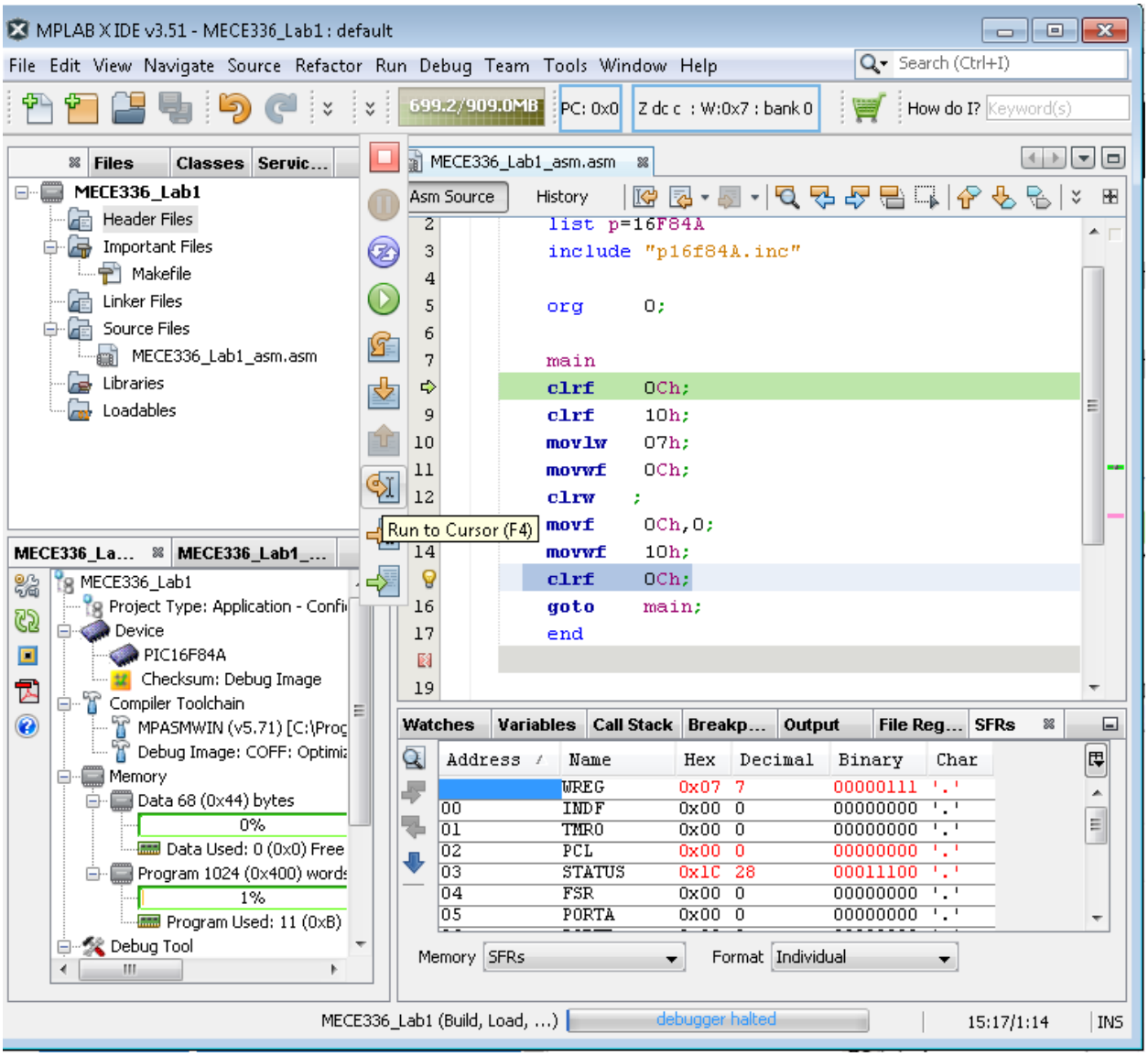
Click “Step Into”  icon to execute one line of source code. Notice the PC increment as the green line goes to the next line in code



7. Run to Cursor

We could continue to step through the program one line at a time to get to a place of interest but this may be time consuming.

Place the cursor on the line of code containing the desired instruction, for example “clrf 0Ch”;. Then, right click and select Run to Cursor



Prepared by Assoc. Prof. Dr. Klaus Werner Schmidt

Department of Mechatronics Engineering

Cankaya University