Lab Sheet 1

An Introduction to MPLAB

1. The MPLAB Introduction:

Microchip technologies provide the MPLAB X IDE, which is used to develop applications for Microchip microcontrollers. The process of writing an application is knows as a *development cycle*, as all steps from design to implementation are most of the time taken in an iterative process. This *development cycle* is illustrated in Fig. 1 below.

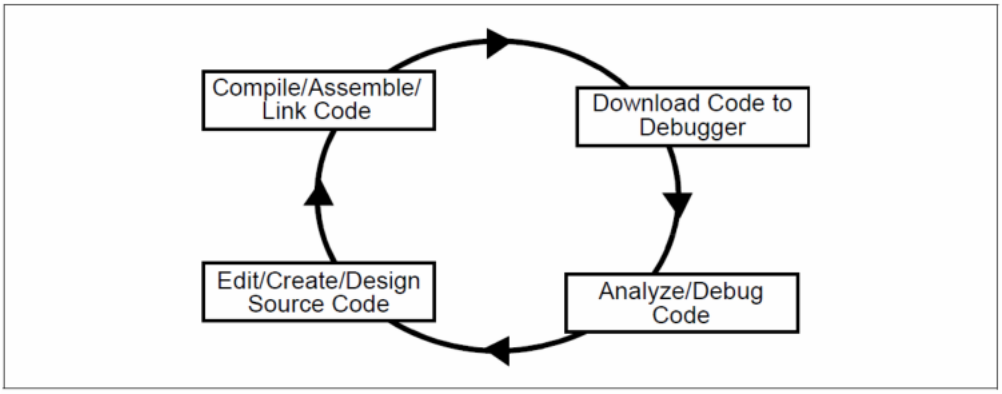

Fig. 1 The development cycle (MPLAB manual).

2. The MPLAB Components:

All the functions of the development cycle are integrated into MPLAB IDE as shown in Fig. 2, which provides a graphical use interface and contains many tools:

1. An editor to write the code,
2. A project manager to organize the files and settings,
3. A compiler or assembler to convert source code into machine code,
4. A debugger allowing breakpoints, single stepping, and watch windows.
5. Some sort of hardware that connects to a target microcontroller (or software that simulates the operation of a microcontroller, an *execution engine*).

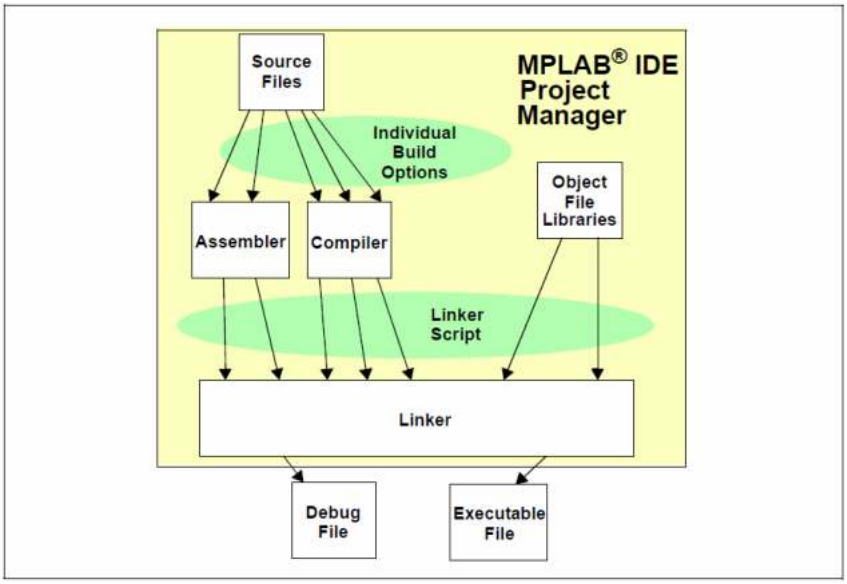

Fig. 2 MPLAB IDE project manager (MPLAB manual).

3. Creating a Project in MPLAB:

To create a project in MPLAB, we will use the project wizard provided by the MPLAB IDE. Please follow the shown steps (Open MPLAB through Start Menu → All Programs → Microchip → MPLAB X IDE v5.05):

Step 1: Initiate a project. Select "File → New Project" or click on "Create New "

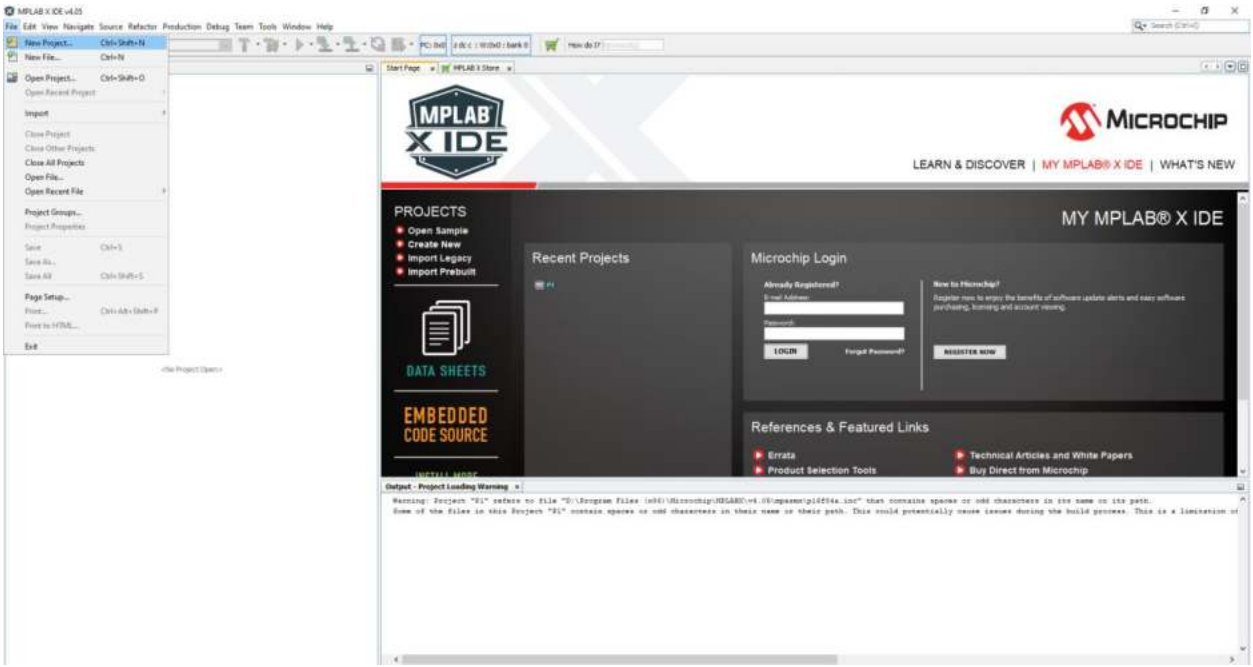

Fig. 3 Selecting Project → Project Wizard

This will display a project wizard window. Select Microchip Embedded under Categories and Standalone Project under Projects.

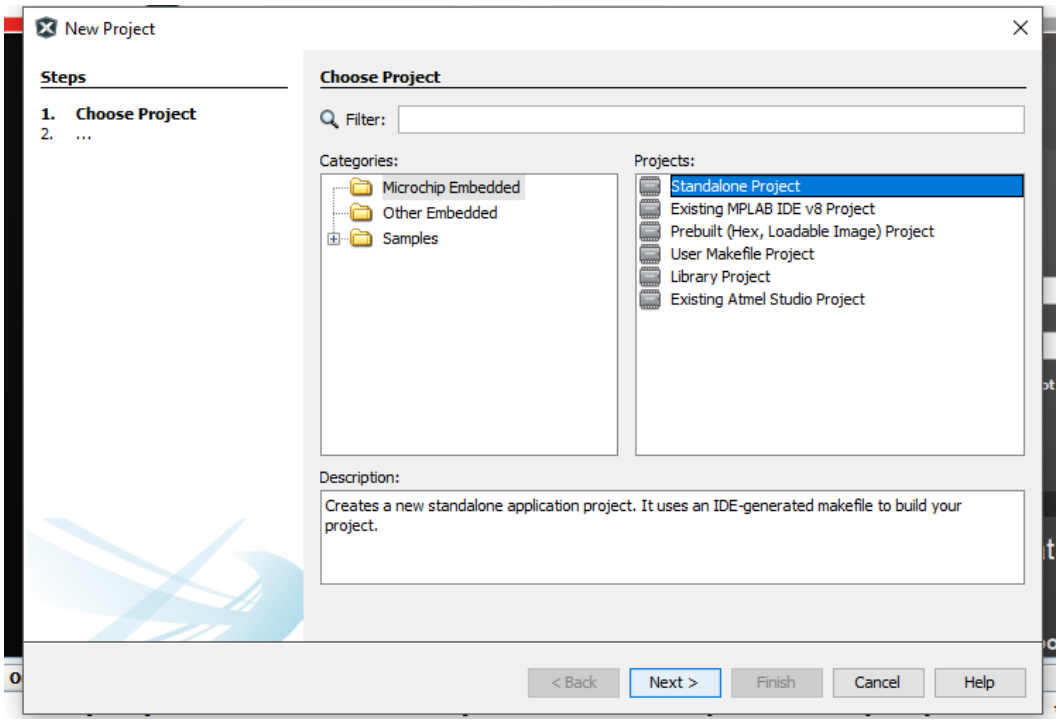

Fig. 4 Choosing Project Type.

Step 2: Select a device. The capabilities of MPLAB IDE vary according to which device is selected. Device to be selected is "**PIC16F84A**".

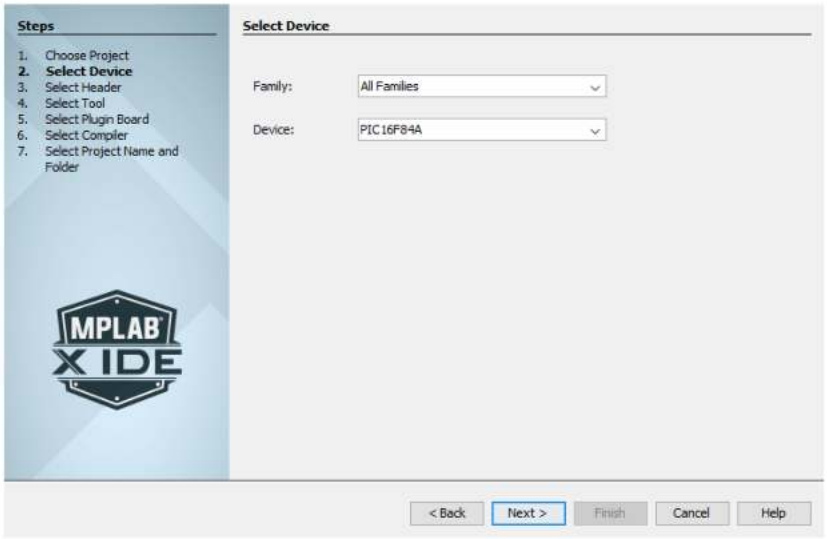

Fig. 5 Selecting the device.

Step 3: Select a language tool suite. If you wish to upload your code to hardware select PICkit3. However, since today's lab is just software simulations, we will select the **Simulator** tool.

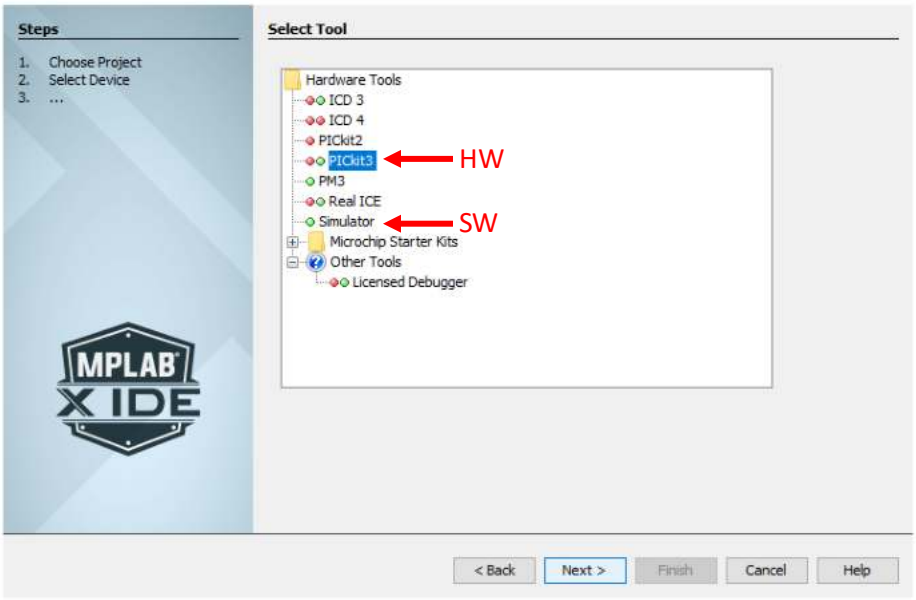

Fig. 6 Selecting a language tool suite.

Step 4: Selecting the compiler. Throughout the lab sessions you will be required to program in assembly or C. "mpasm" is used for assembly and XC8 is used for C. For now select **mpasm**.

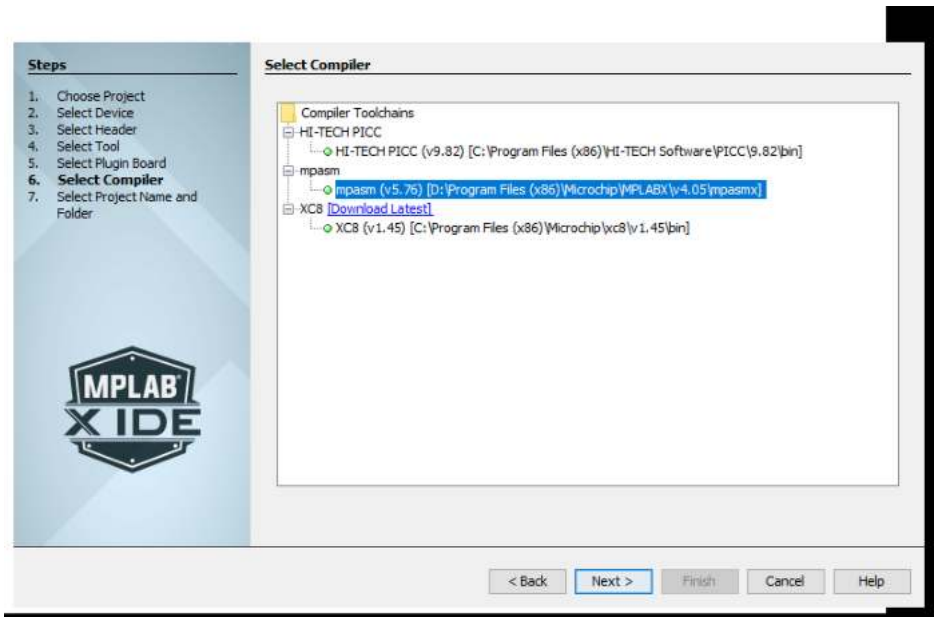

Fig 7. Selecting a compiler.

Step 5: Create a new project. **Always prepare a different folder for every new project (for every lab session)**. Write the Project Name and select the appropriate folder then click Finish.

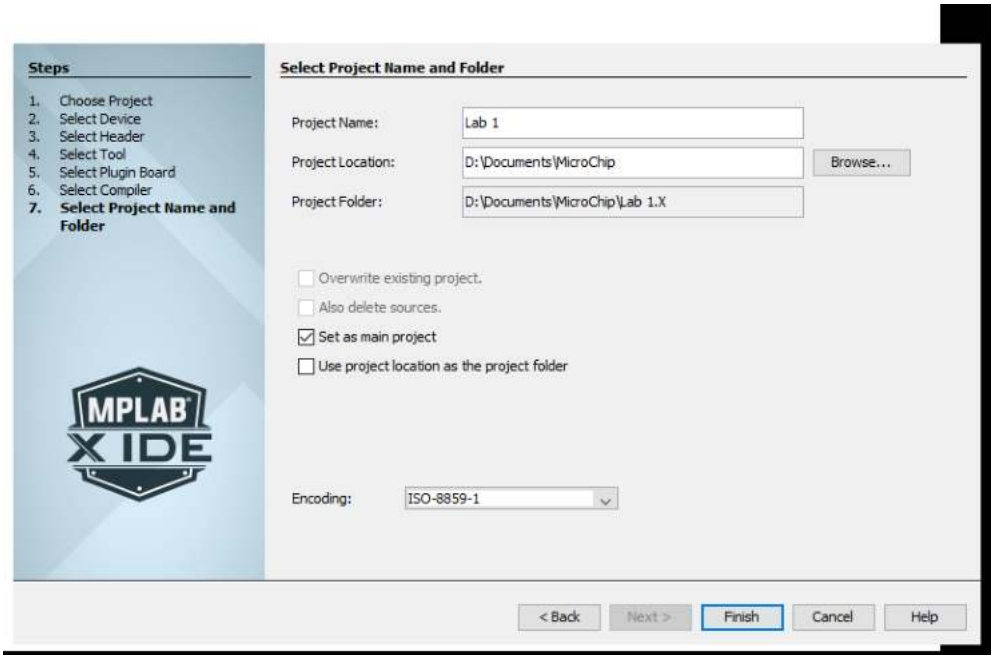

Fig. 8 Creating a new project.

Step 6: Add existing files to your project. Add the **p16f84a.inc** file available in "C:\Program Files (x86)\Microchip\MPLABX\ v5.05\mpasmx\p16f84a.inc".

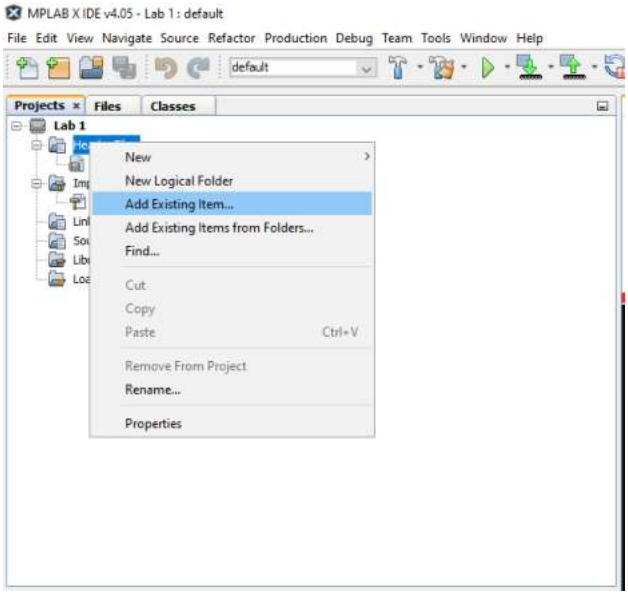

Fig. 9 Adding files to the created project.

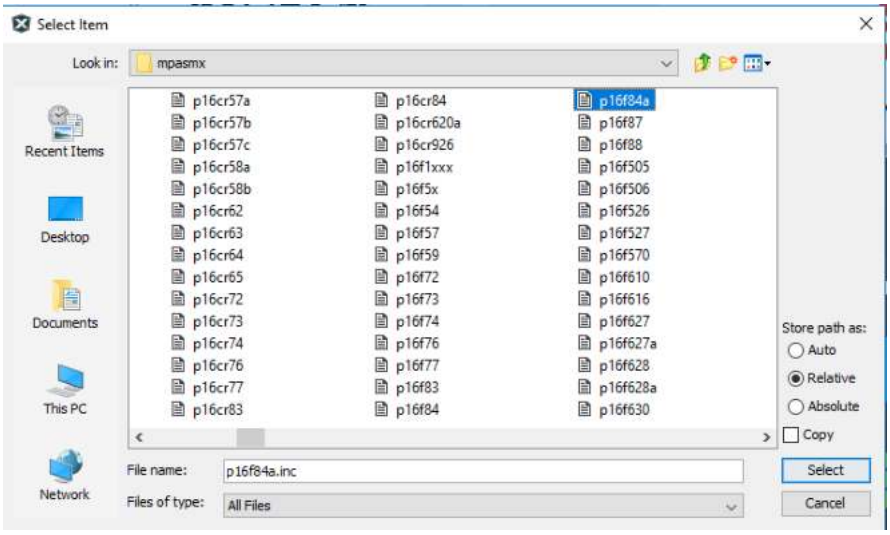

Fig. 10 Adding the appropriate header file

Step 7: To add source files, download the **Assembly Files** available on **Moodle** to your project folder. Make sure to extract the zip folder in the project directory. Then, right click on "Source File" and choose "Add Existing Item" to select the files you want.

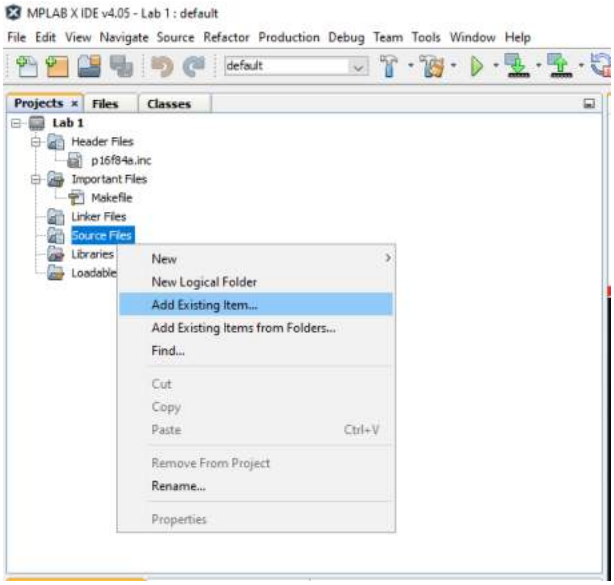

Fig. 11 Adding a template source file to the project - 1.

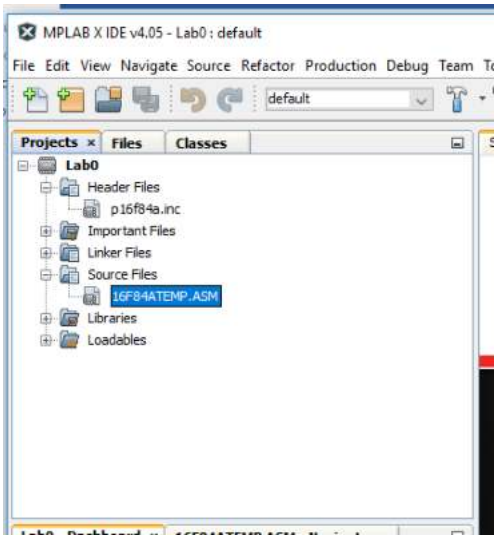

Fig. 12 Adding a template source file to the project - 2.

Finally, note that the new file appeared under the "Source Files" of the project.

4. Your First Program:

For your first assembly program, we will be testing some simple literal and byte-oriented instructions. To write your assembly program, you just need to insert your code between the "**main**" label and the "**END**" directive in the source file. For this example, the following code has already been added for you:

```
CLRW
MOVLW       0xD2
ADDLW       D'10'
ADDLW       B'00110011'
NOP
```

Code 1 – Program 1 Code

The next step is to build your program. The normal way is to select "Production → Build Main Project" or press "F11". This is for **uploading the code to the microcontroller**.

For Debugging however, you will need to build the project from "Debug → Discrete Debugger Operation → Build for Debugging Main Project." If the build is successful, the message "BUILD SUCCESSFUL" should be printed out in the output window as shown below

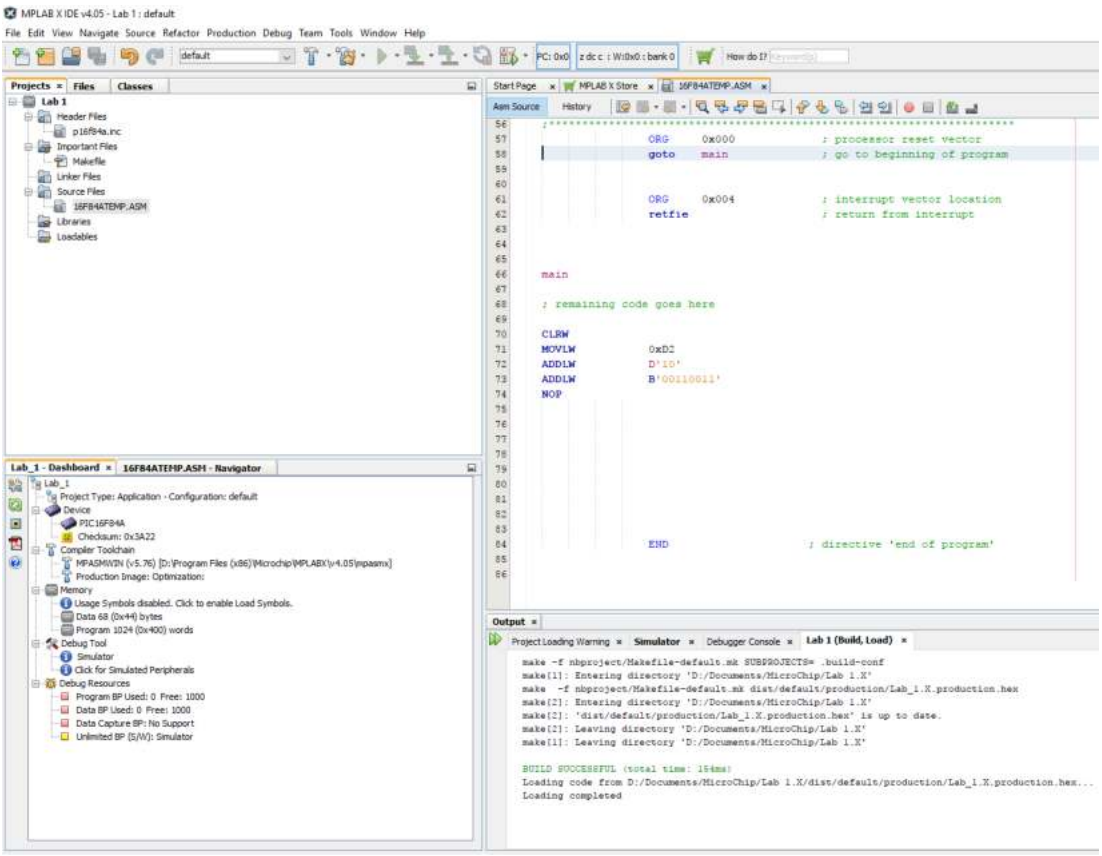

Fig. 14 A successful build operation.

Now we are ready to simulate this program by executing it step-by-step and monitoring how the values of the different registers change during execution. In order to do this, you need to run your program in debugger mode (AFTER YOU BUILD) by going to Debug → Discrete Debugger Option → Launch Debugger Main Project. You should then see the Debugger buttons on the toolbar as shown in Figure 13.

Fig. 13 Debugger buttons.

In order to view the values inside the special function registers, select "Windows → Memory Views → SFRs". The special function registers window will appear as shown in the figure below.

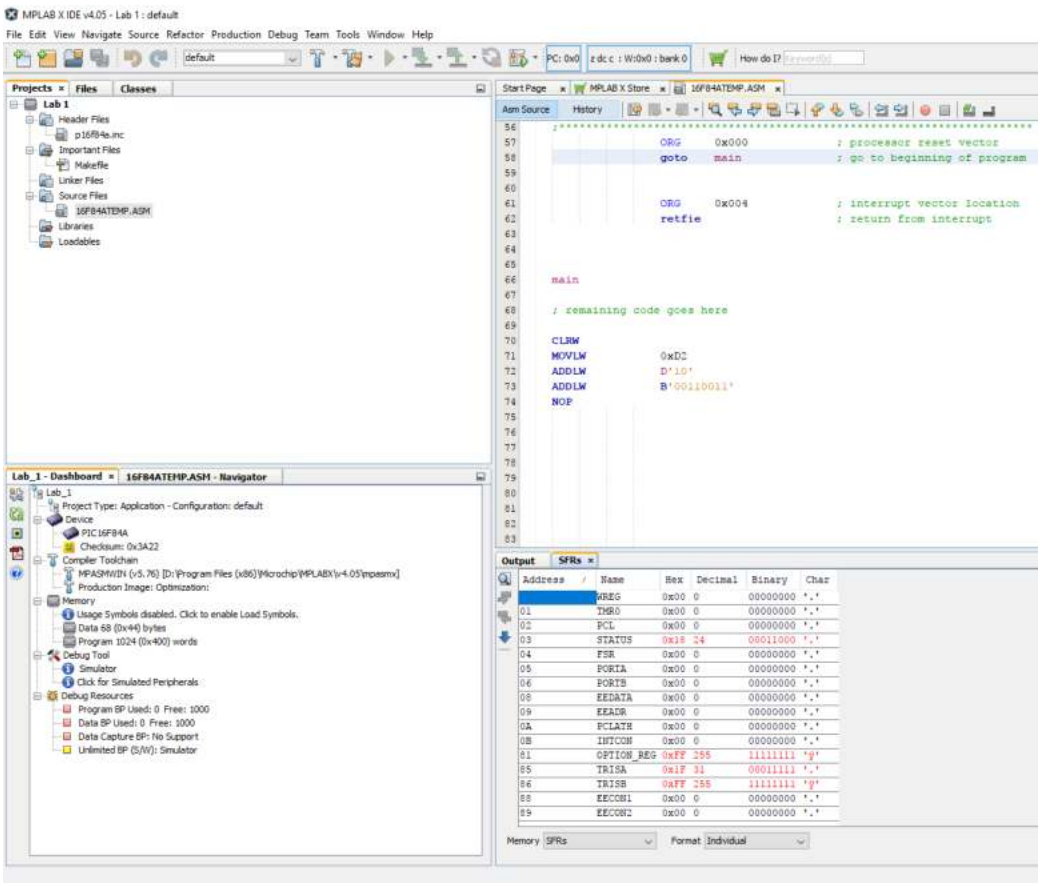

Fig. 15 Special function registers window.

Once you run the debugger notice how a line of code is **highlighted in green**. This indicates which instruction the program is currently at.

In this window you can see the different values for special function registers in the controller. For example, The STATUS register contains some useful "flags". The PC register shows the counter of the current instruction. We'll learn more about these in future labs.

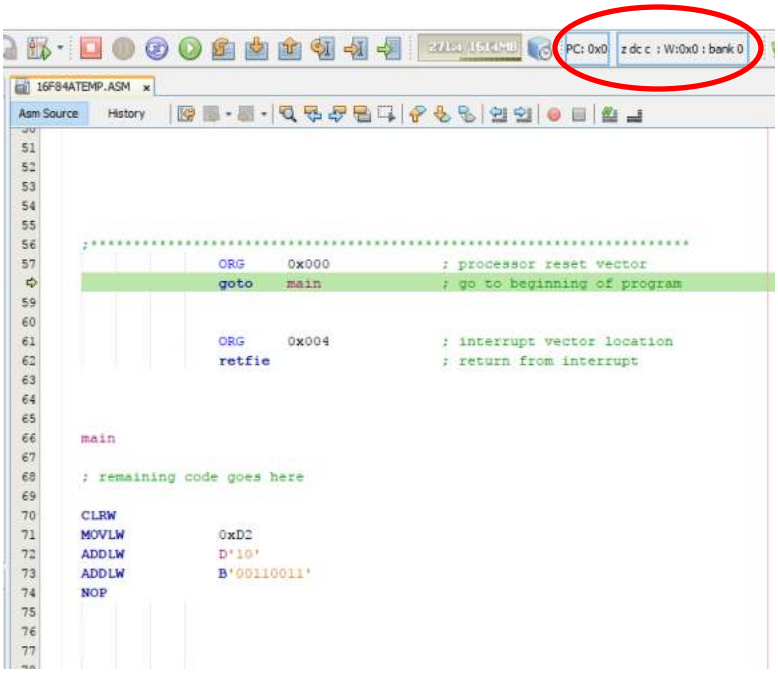

Fig. 16 Debugging the code

For some of the special bits in the STATUS register, we can directly see their values from circles top right display in Figure 16. An uppercase letter(s) indicates a '1' value, and lower case indicates a '0' value for that flag.