ASM Programming in mplab X



Contents

[Create your Project and Source files 2](#_Toc33368122)

[Writing the First Assembly Project 4](#_Toc33368123)

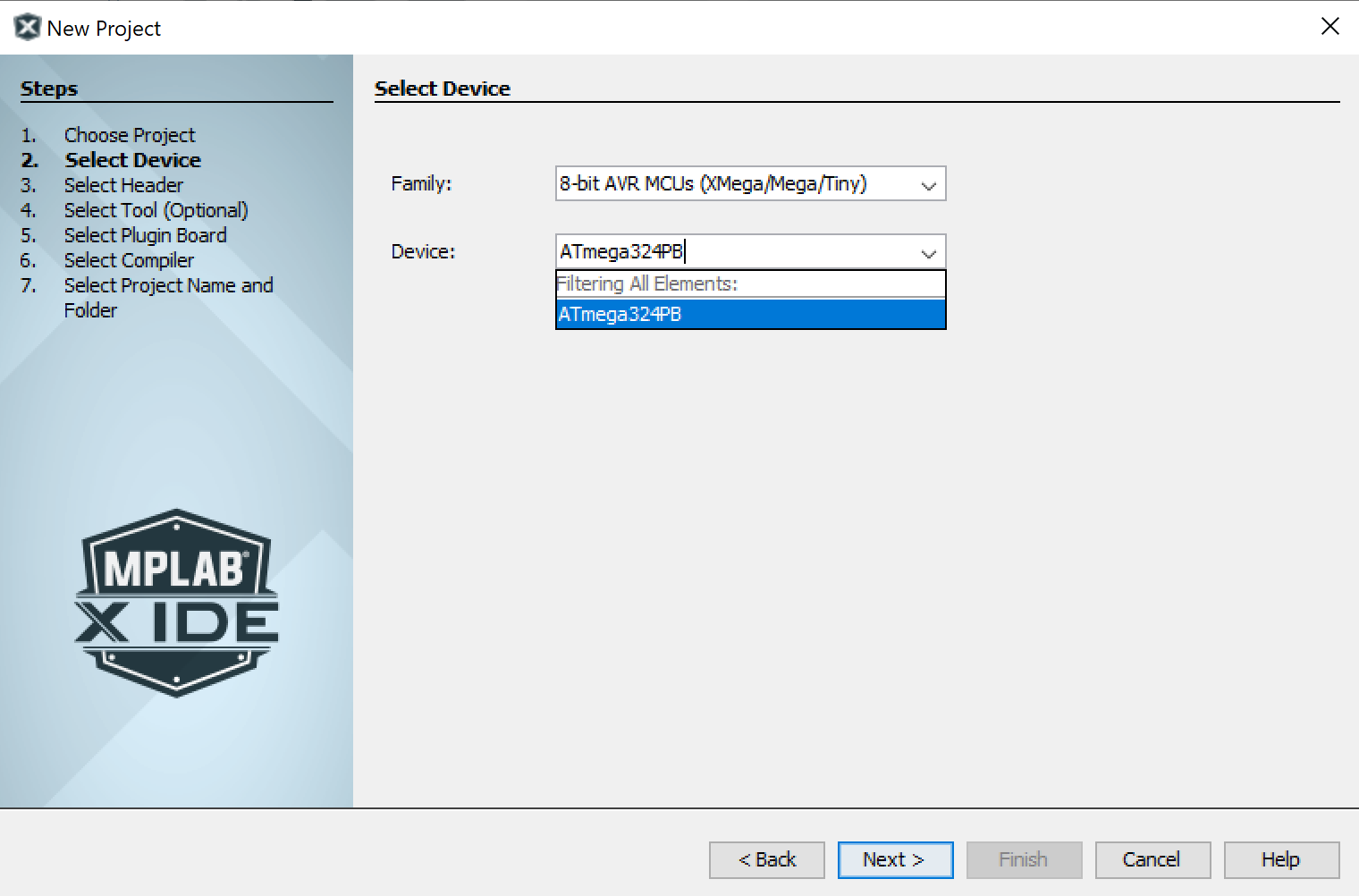
[Building your Project 4](#_Toc33368124)

[Using the Debugger 5](#_Toc33368125)

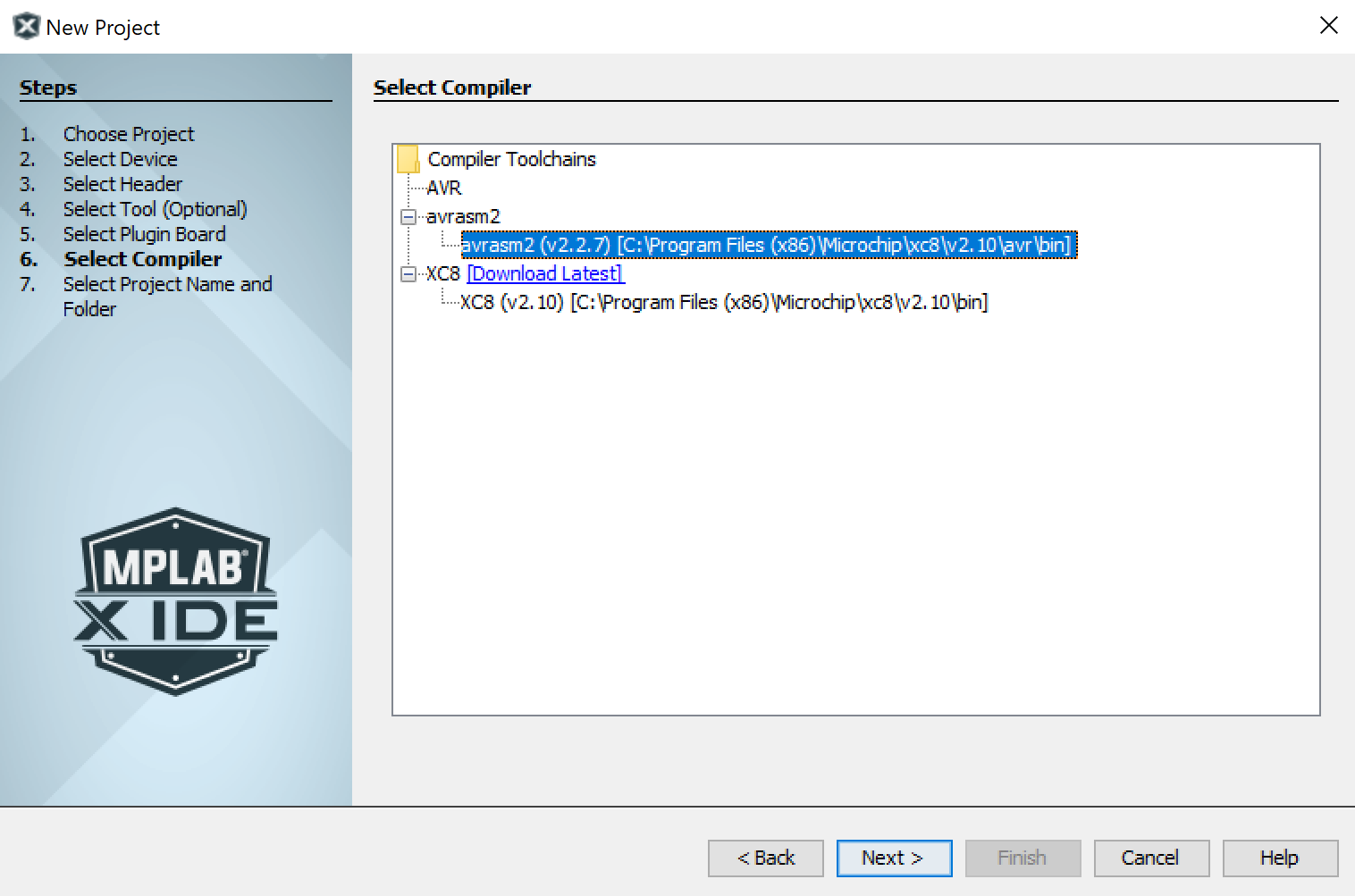
[Using Breakpoints 8](#_Toc33368126)

# Create your Project and Source files

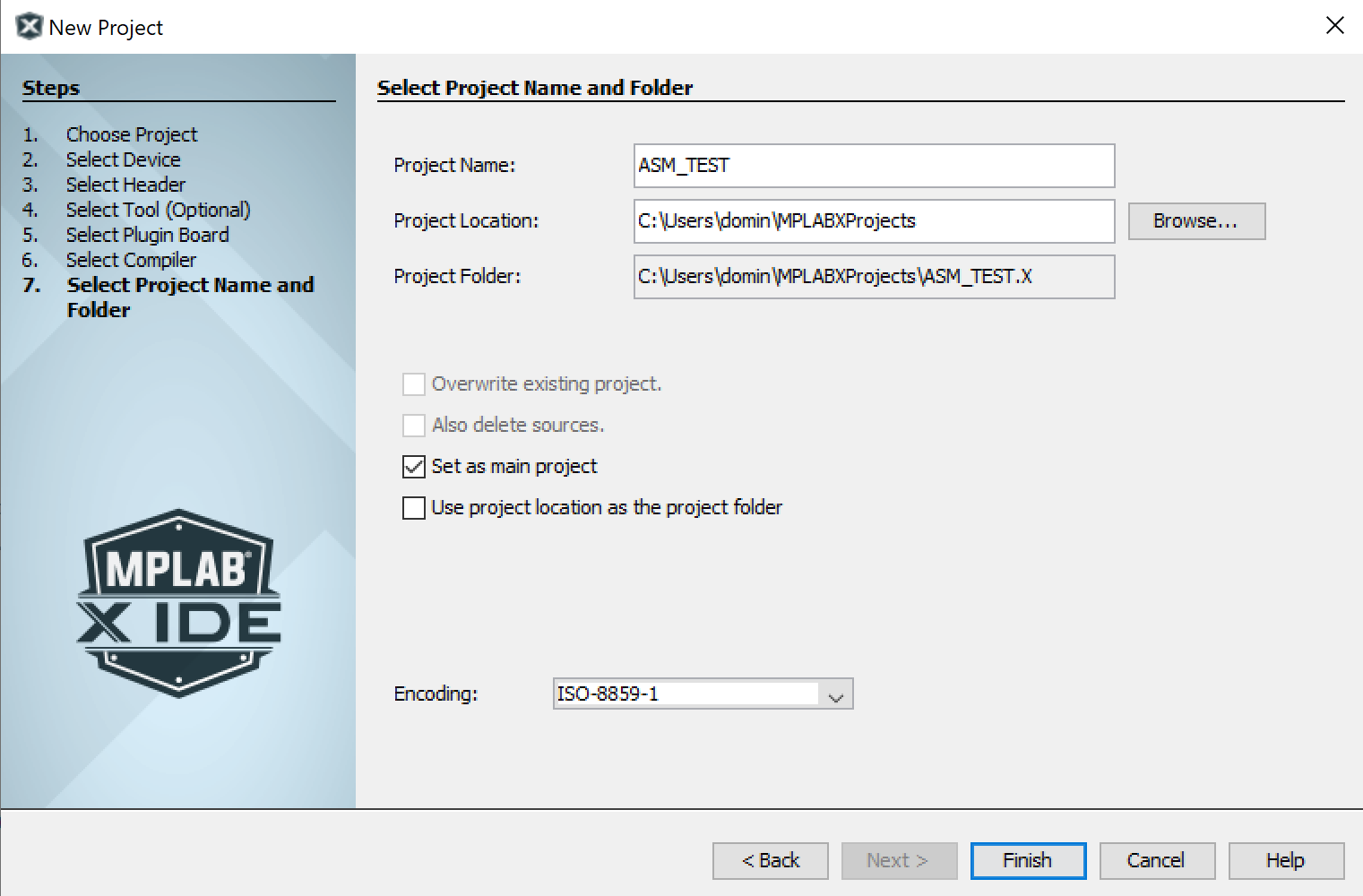
* 1. Create a new project by clicking the  icon
  2. Set the Family as 8-bit AVR MCUs and Device as ATmega324PB



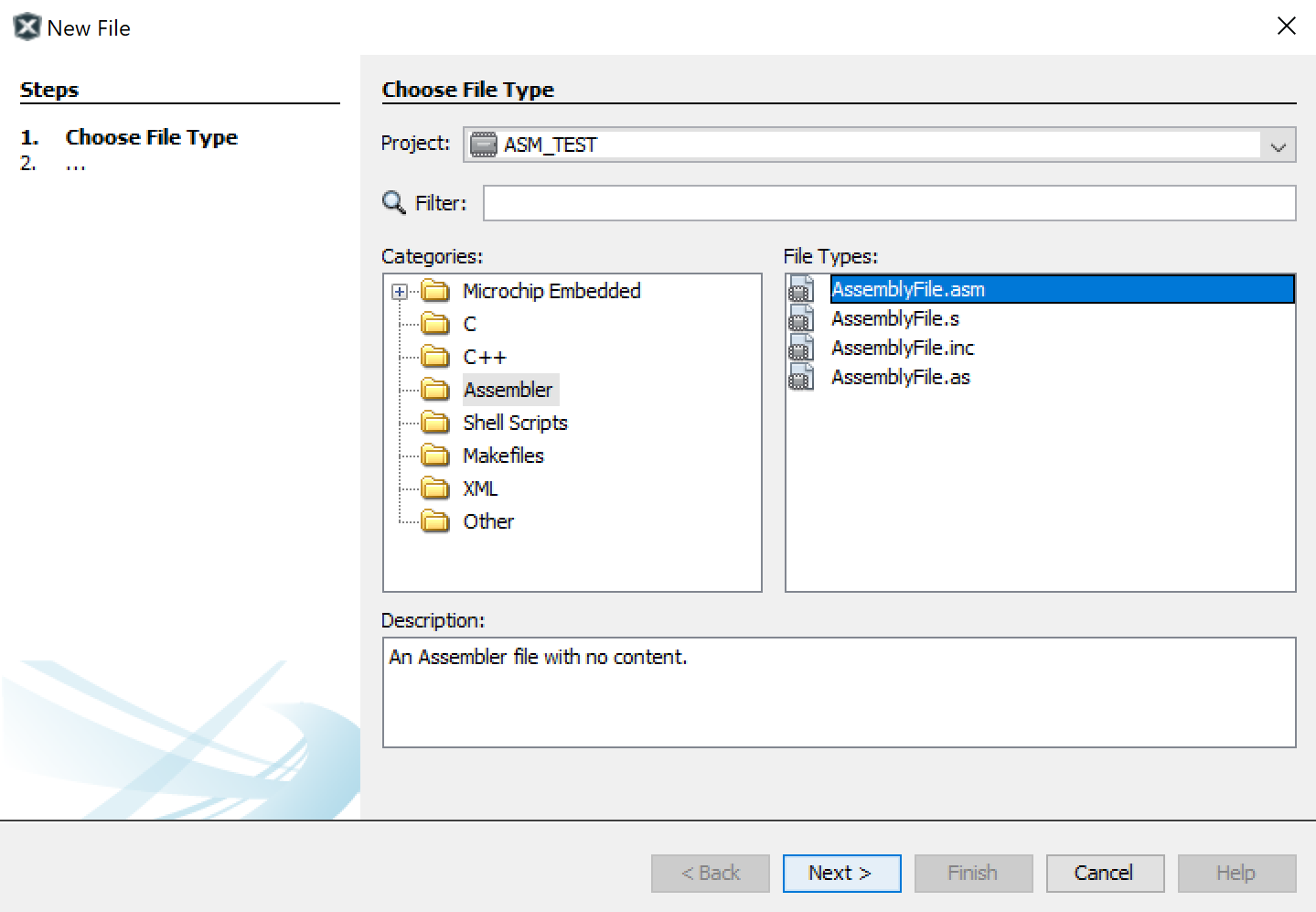
* 1. Set the toolchain as avrasm2



* + 1. If you do not have this toolchain available, follow the Getting Started with MPLAB Tutorial.
  1. Set the project directory to a reasonable location in your file system.

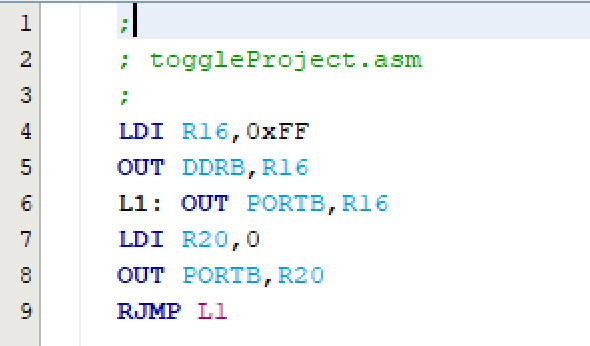


* + 1. Be sure the path name contains no spaces!
  1. Create a new source file by clicking the  icon
  2. Chose Assembler > Assembly File



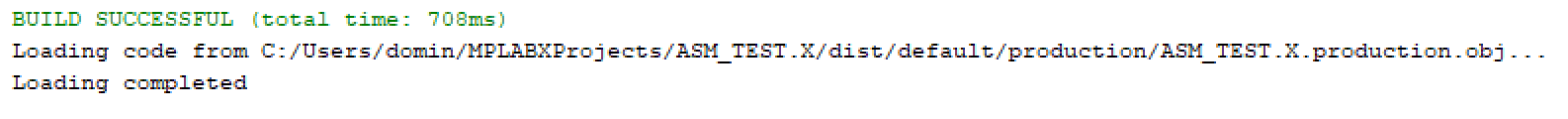
# Writing the First Assembly Project

* 1. Type the following code. This program sets the **Data Direction Register** for **PORTB (DDRB)** by writing 0xFF, then it loops and toggles the outputs at **PORTB**.



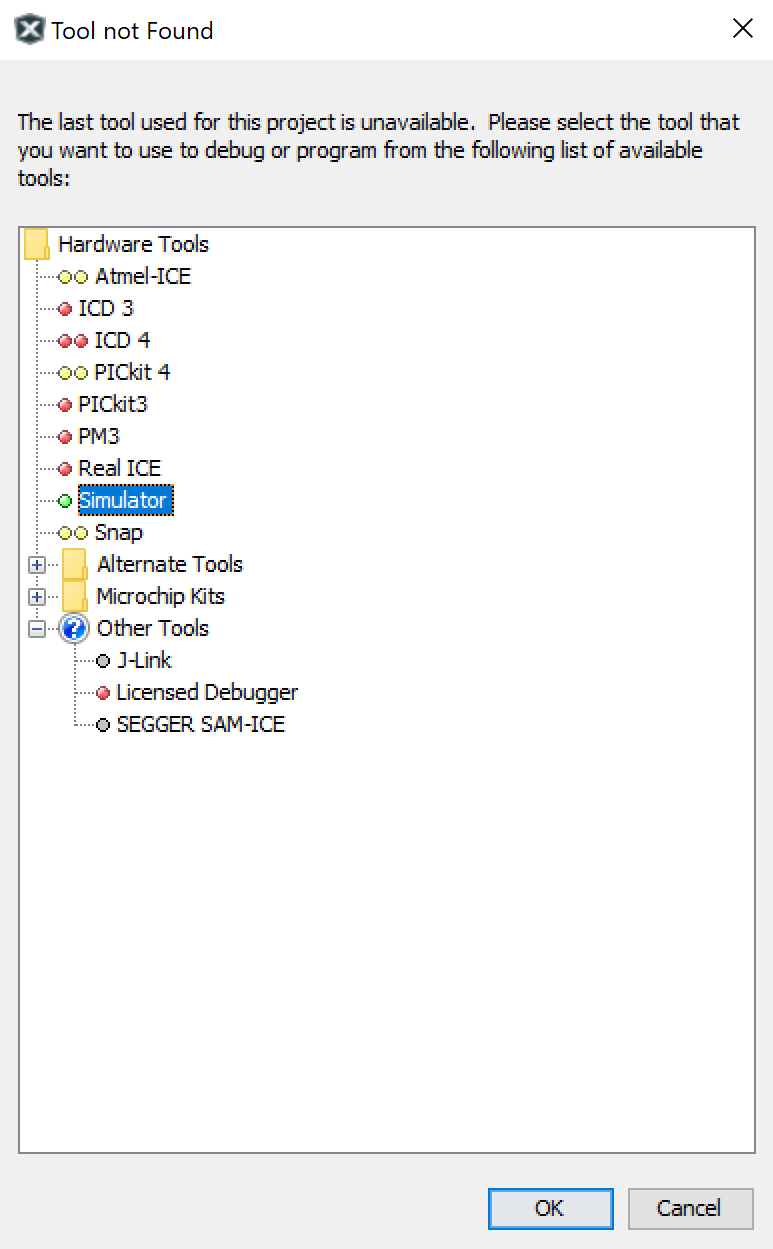
# Building your Project

* 1. Building is like compiling in on other platforms.
  2. Press the  icon to Build or the  icon to clean and build.
  3. Like compiling, Building will tell you if your program has any errors.
  4. Building is complete when you see this message



# Using the Debugger

* 1. Click the  icon to start the debugger
  2. The following dialog box will appear. Chose the Simulator.



Note: Simulator vs. debugger

Using the simulator, you can execute the instructions, and watch the registers and variables. If you have a debugger, e.g. AVRISP mkII or Atmel-ICE, you can connect a trainer board to your computer. In the case, the microcontroller of the board executes the same instructions, when you trace the program. This facilitates you to check the hardware while monitoring the variables in the IDE.

* 1. This  is the debugger toolbar. While the debugger is running the only options available are to exit or pause the debugger.
  2. With the debugger paused, the other options become available



* 1. The buttons from left to right are Quit, Pause, Reset, Continue, Step Over, Step Into, Step Out, Run to Cursor, Set Program Counter to Cursor, Focus Cursor at Program Counter

Step Into vs. Step Over

Both F8 (Step over) and F7 (Step into) execute one instruction and go to the next instruction. But they work differently when the cursor is on a function call. If the cursor is on the function call, Step into goes into the first instruction of the function, but Step Over executes the whole function and goes to the next instruction.

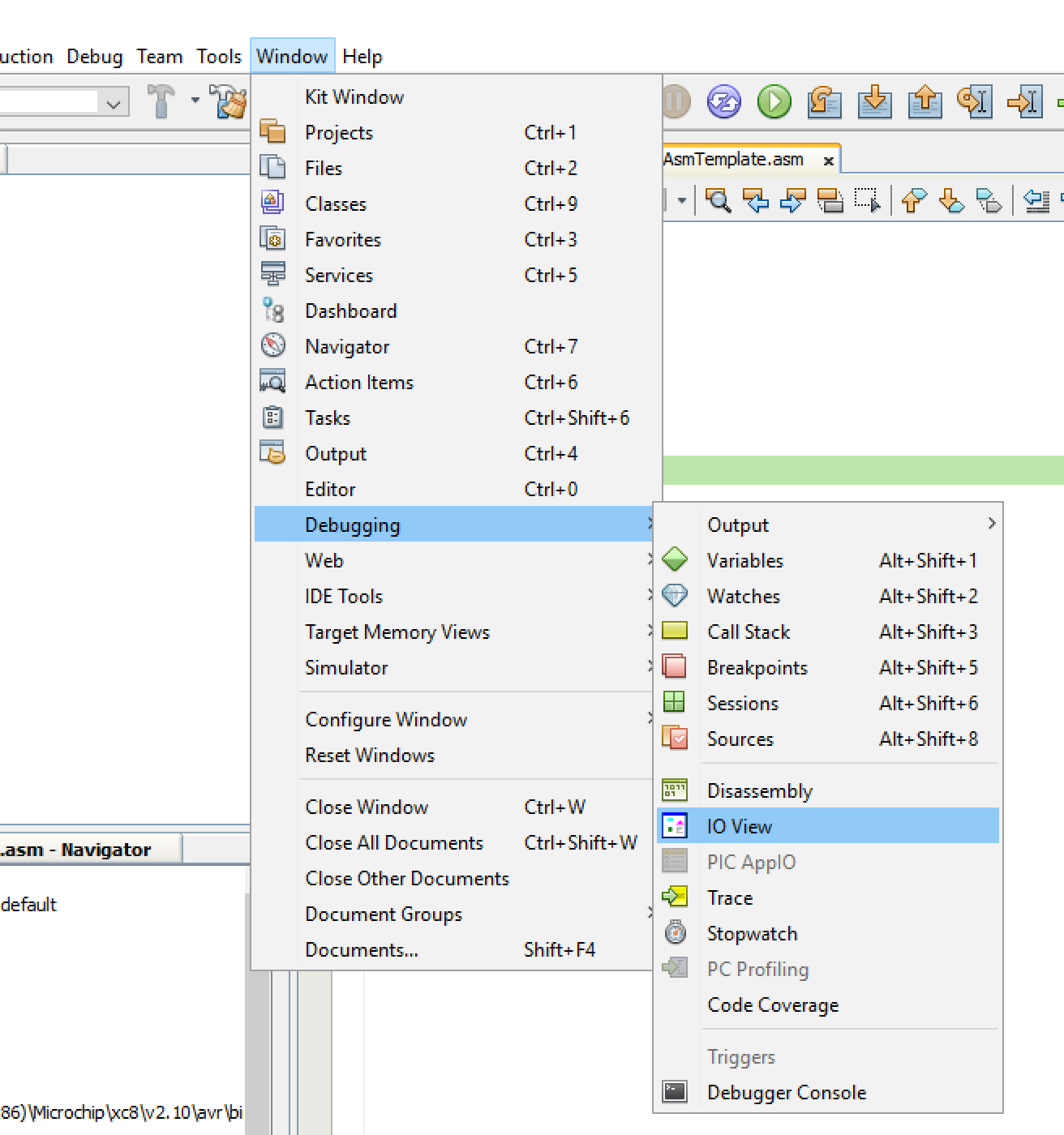
Step Out

If the execution is in a function, you can execute the function to the end by pressing the Step Out.

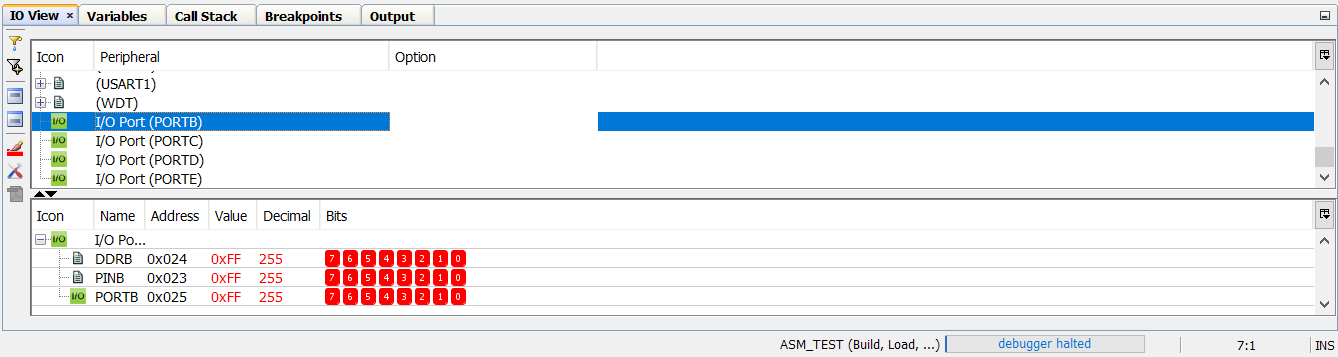
Run to Cursor

You can put the cursor on an instruction and then press the Run to Cursor button. In the case, the program runs until it reaches the instruction which the cursor is on it.

* 1. To monitor peripherals like I/O, click window, debugging, I/O view



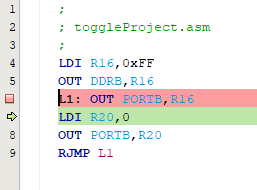
* 1. The **I/O** tab appears on the bottom which shows the peripherals of the microcontroller, including the **I/O** ports. Select **PORTB**. The values of the related registers (**PINB**, **DDRB**, and **PORTB**) will be shown below.



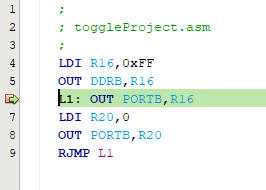
* 1. Press **F8 (Step Over)** a few times and see the **PORTB** register changes in the **I/O** window.

# Using Breakpoints

* 1. If you want to debug a portion of a program, add a breakpoint to the beginning of this part of the code and press the run button. The IDE runs the program and when it reaches the breakpoint, it stops running and the yellow cursor is shown on the breakpoint line. Below, you see the steps in detail.
  2. With the debugger active and halted, click on the line you wish to insert a breakpoint.



* 1. Press **Continue (F5)** to resume the debugger. The debugger will halt when the Program Counter reaches that instruction.



* 1. The bottom left corner of the screen shows how many breakpoints are in use and available for use.

