C Programming in Atmel Studio 7

Step by Step Tutorial

Sepehr Naimi

BIHE University

12/12/2017

Contents

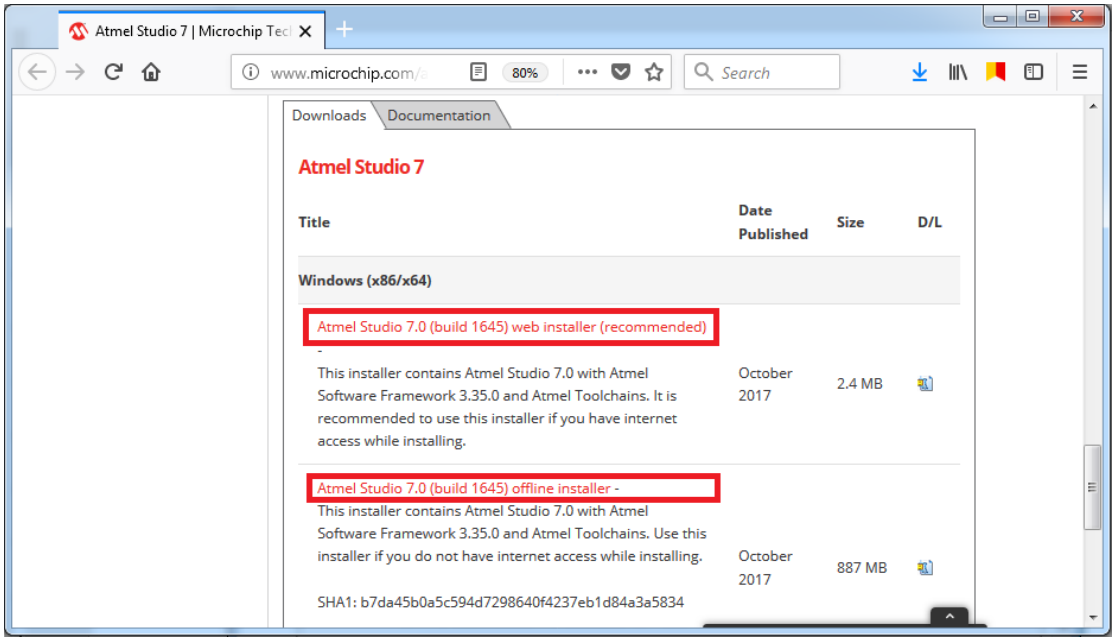
Introduction

This tutorial will teach you how to write, compile, and trace a simple program in Atmel Studio 7.

Downloading and Installing Atmel Studio

Download the newest version of Atmel Studio from the Microchip website:

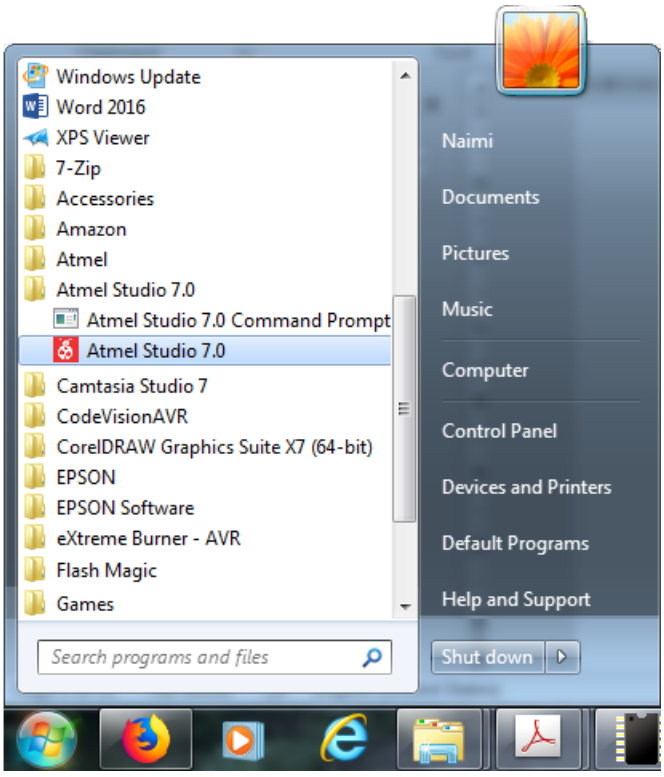
http://www.microchip.com/avr-support/atmel-studio-7



Run the downloaded program to install the Atmel Studio IDE.

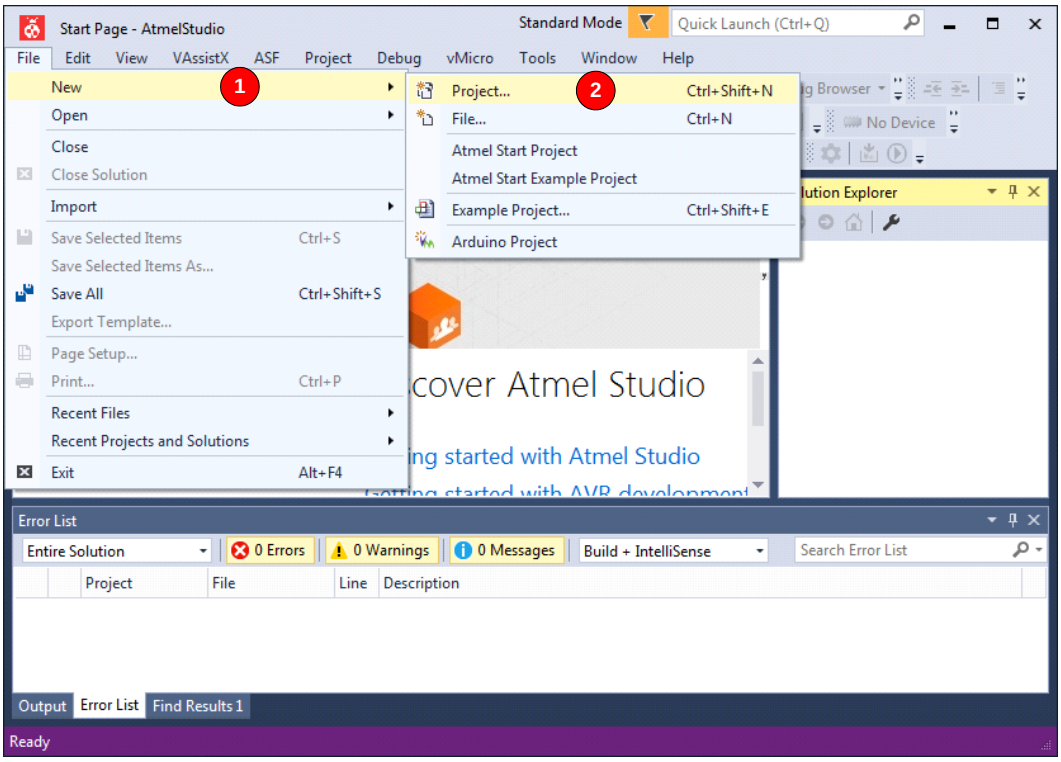
Opening Atmel Studio

Go to the ***Start*** menu and open Atmel Studio.

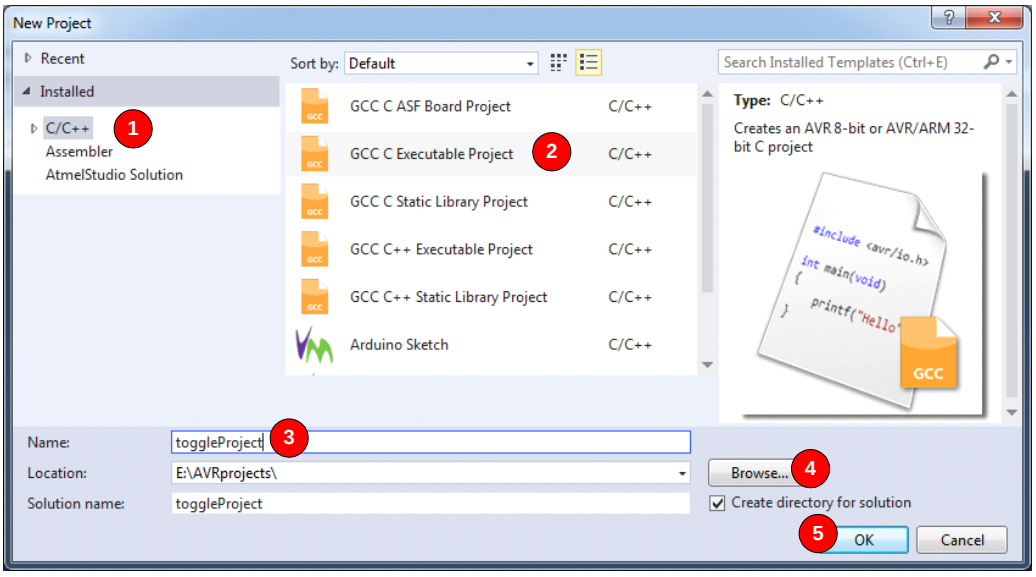


Creating the first project

1. Go to the ***File*** menu. Choose ***New*** and then ***Project***.

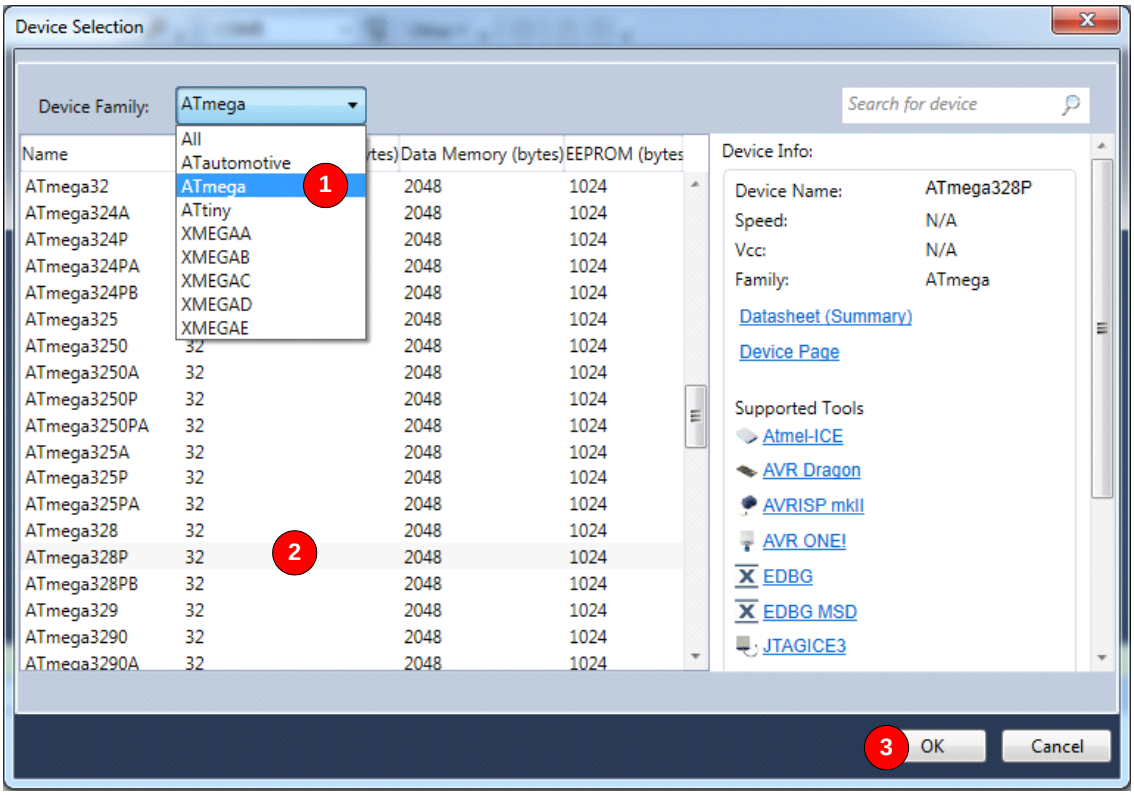


2. In the opened dialog,
   a. Choose ***C/C++***.
   b. Select ***GCC C Executable Project***.
   c. Name the project as ***toggleProject***.
   d. Choose the path where you like to save the project by clicking on the ***Browse*** button.
   e. Press ***OK***.

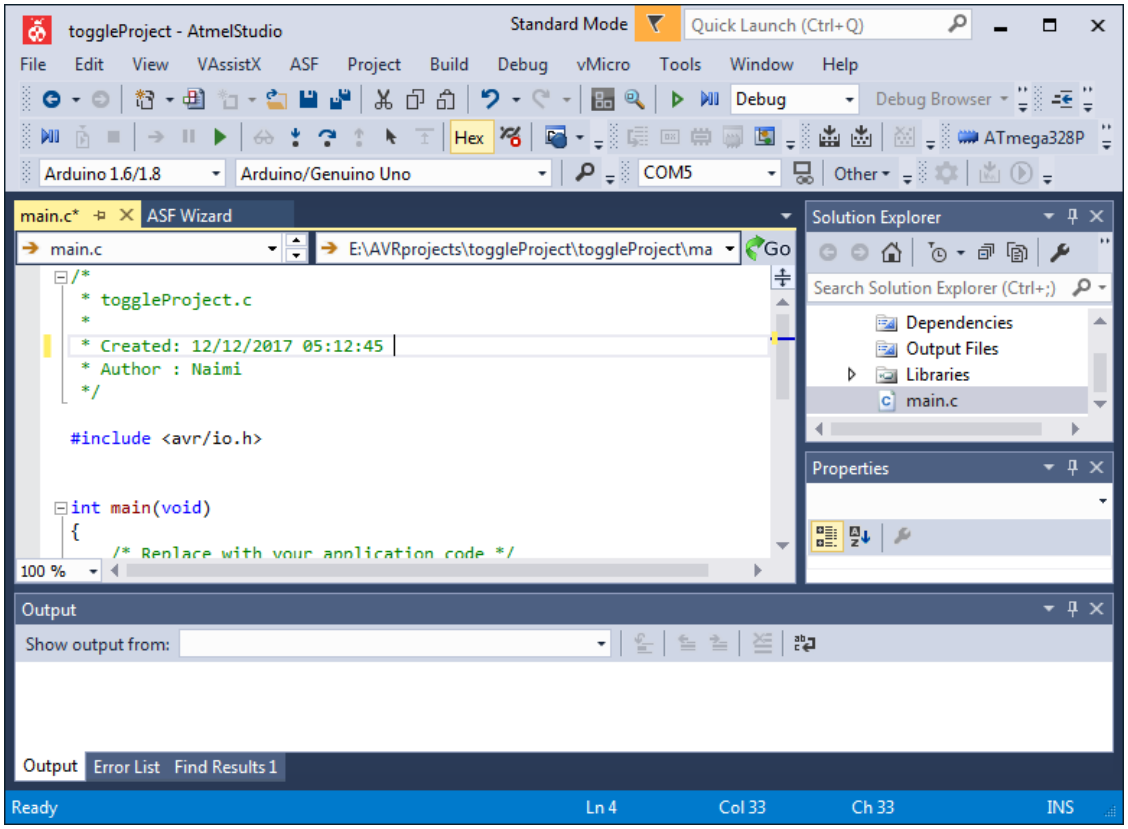


3. In the ***Device Selection*** dialog

a. Select ***ATmega*** as the ***Device family***.
b. Choose ***ATmega328*** (or any other Chips you want to use)
c. Select ***OK***.



The compiler automatically makes the ***toggleProject*** and adds a C file to it.



Writing the first C program

Type the following program.

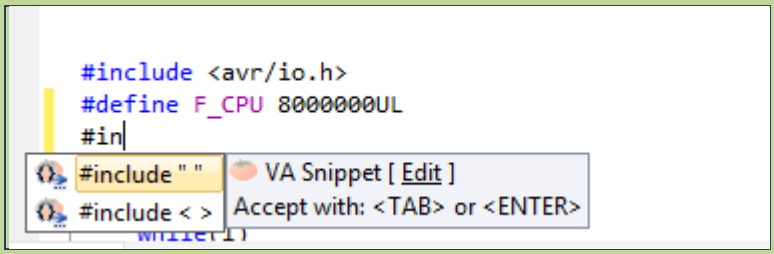
```
#include <avr/io.h>
#define F_CPU 16000000UL
#include "util/delay.h"

int main(void)
{
	DDRB = 0xFF; //make port B as output port

	while(1)
	{
		PORTB = 0xFF; //make all pins HIGH
		_delay_ms(1000); //wait 1 sec
		PORTB = 0x00; //make all pins LOW
		_delay_ms(1000); //wait 1 sec
	}
}
```

Note: Auto Complete

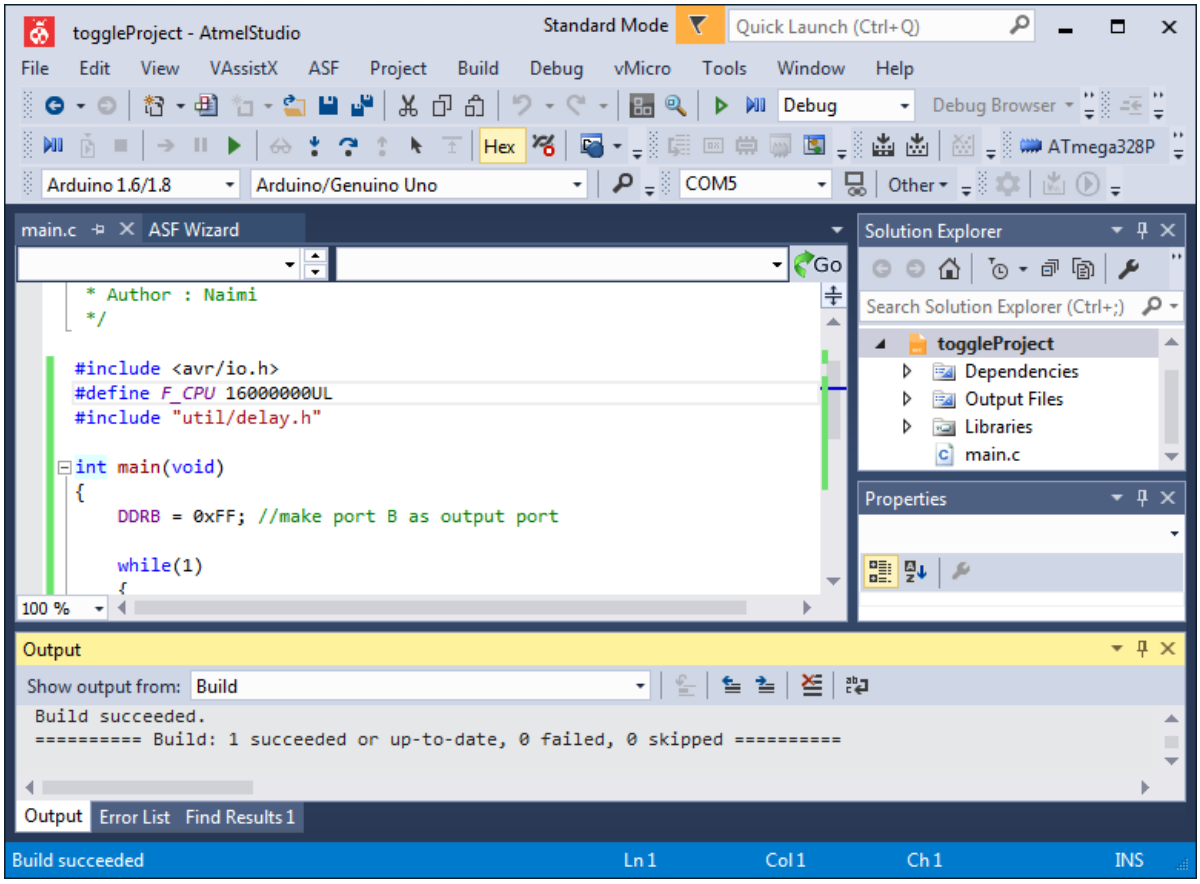
The editor knows the available variables and functions. For example, if you type the first few letters of #include, e.g. #in, the following list appears and you can choose ***#include*** from the list, by pressing the ***Enter*** button instead of typing ***#include***.



While editing the program, if you press ***Ctrl+Space***, the auto complete list appears, and you can choose between the available choices.

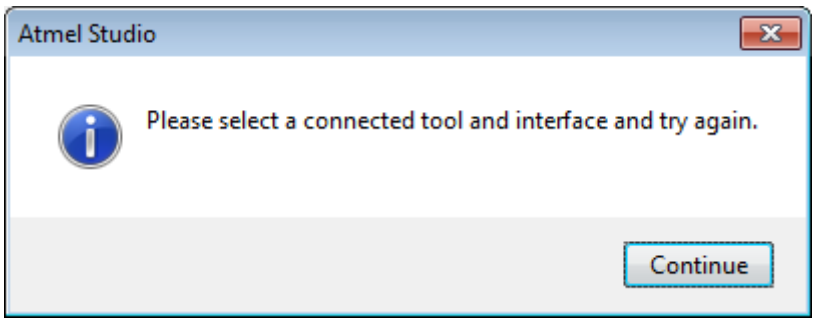
Compiling

Press ***F7*** to compile, or choose ***Build Solution*** from the ***Build*** menu. The results of compile are shown in the ***Output*** window.

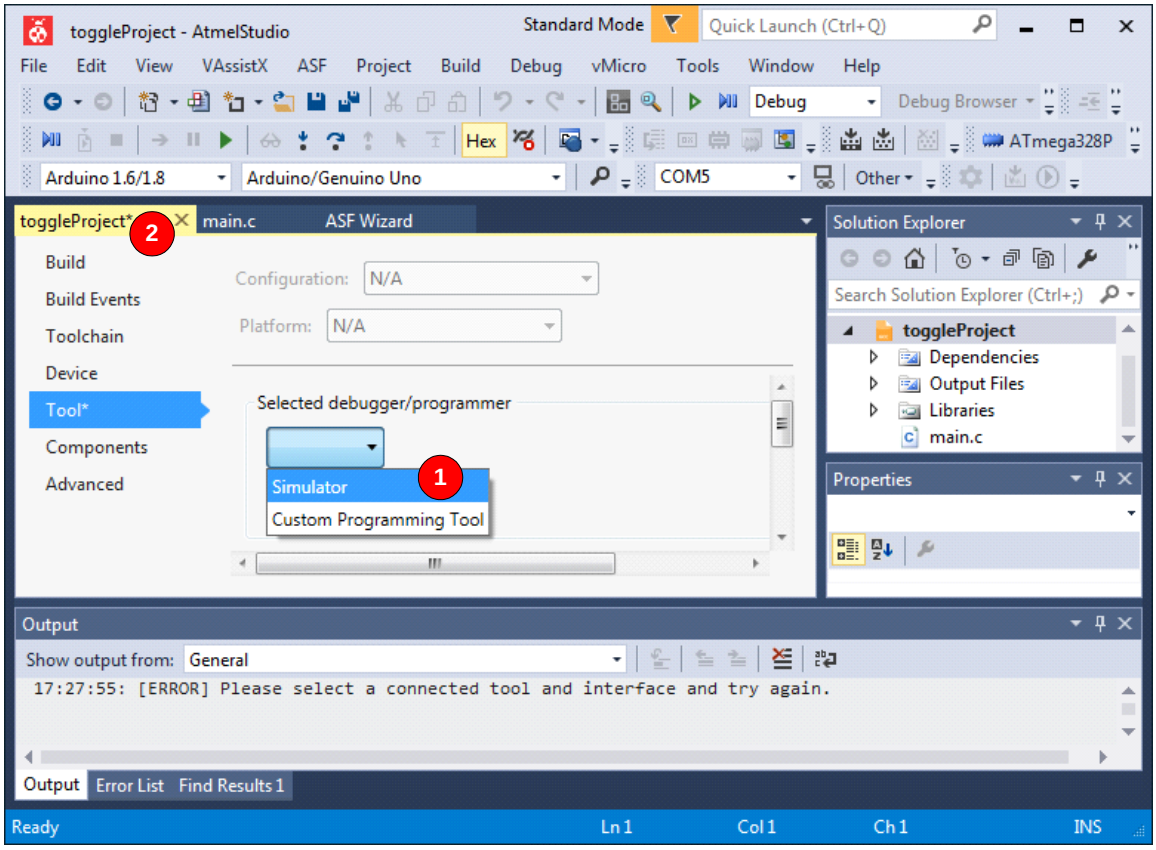


Debugging

1. To start debugging, press ***Alt+F5*** or choose ***Start Debugging*** from the ***Debug*** menu.
2. The following Dialog appears and asks you to select the debugging tool. Press ***Continue***.



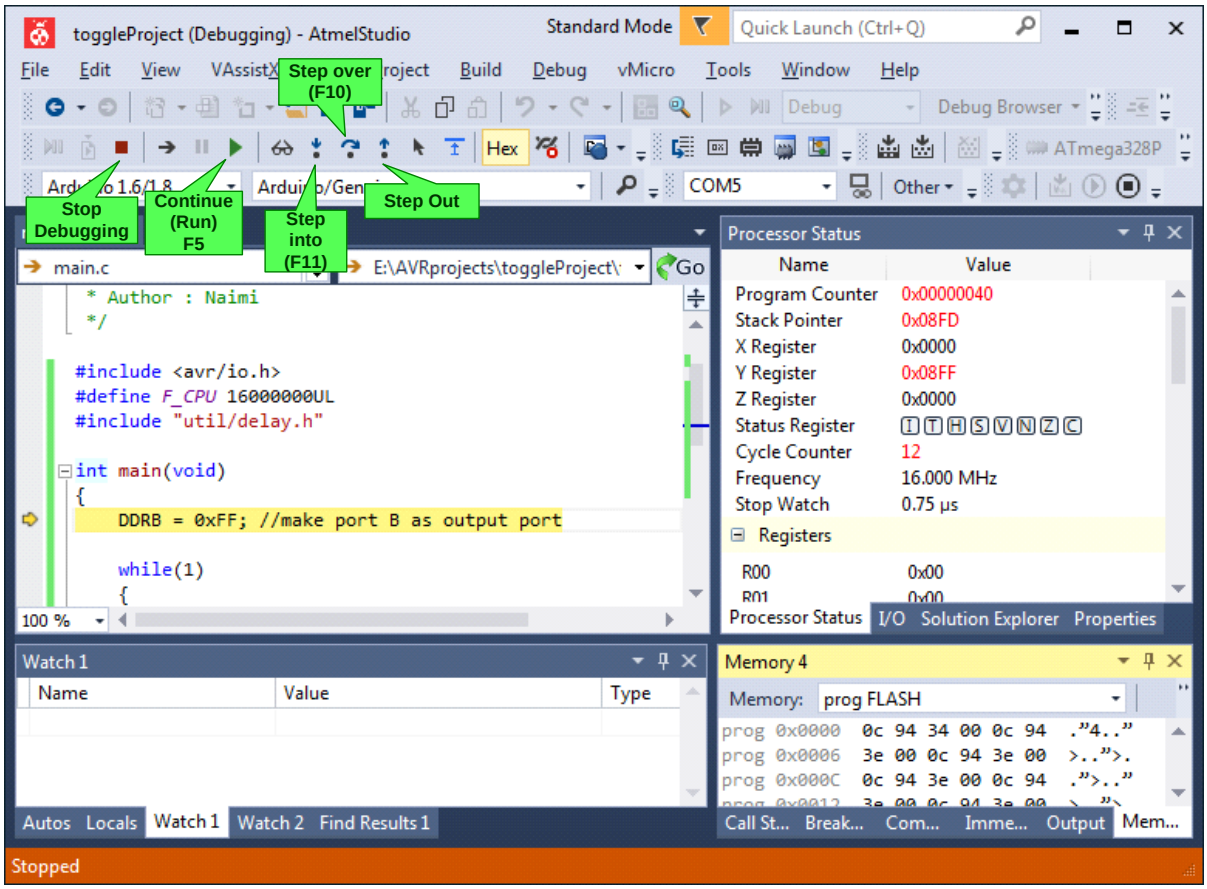
3. In the following window, choose ***Simulator*** as the debugger and then close it by pressing the x next to the ***toggleProject***.



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.

4. Press ***Alt+F5*** again. Now a yellow cursor is on the first line of the main program and the IDE is ready to debug the program.



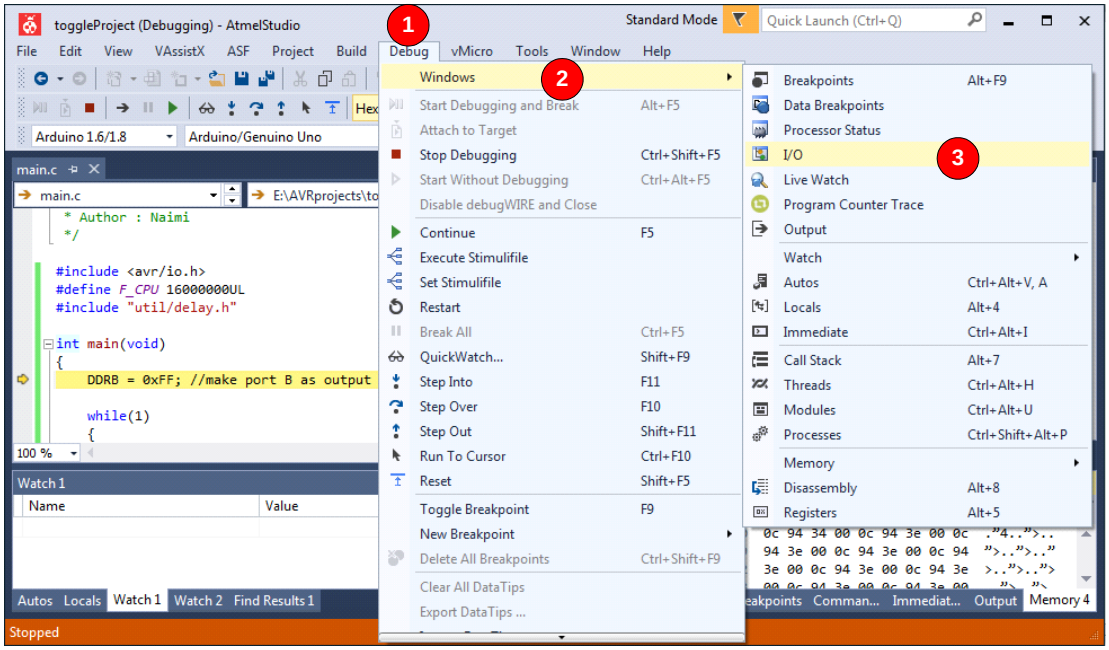
5. To execute the instructions line by line press ***F10*** or click on the ***Step over*** icon.

Step Into vs. Step Over
Both ***F10 (Step over)*** and ***F11 (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.

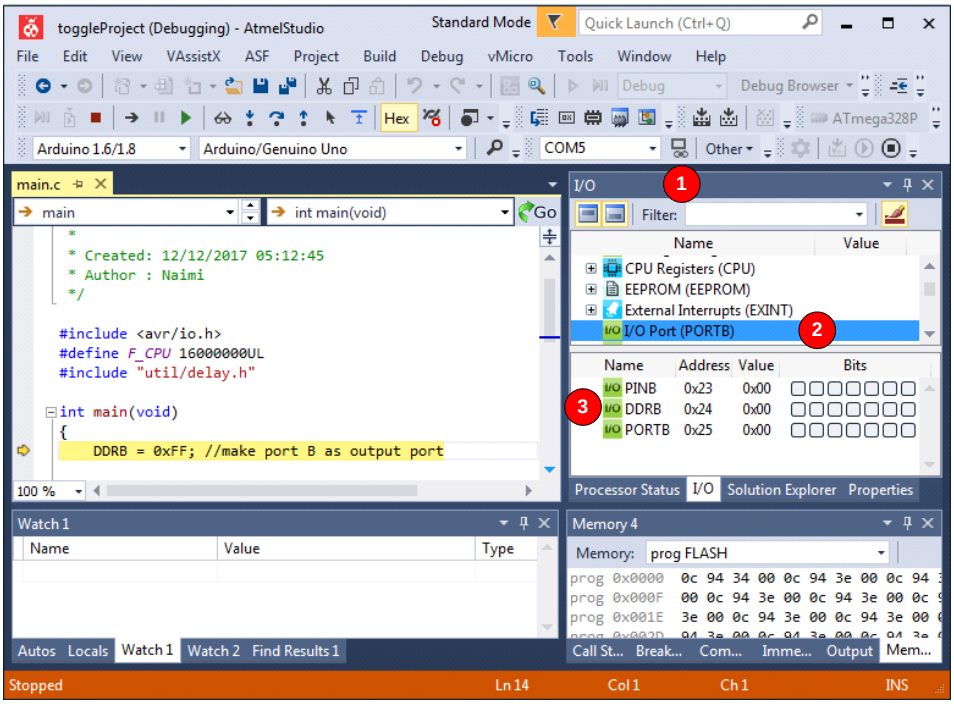
Step Out
If the execution is in a function, you can execute the function to the end by pressing the ***Step Out***.

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.

6. To monitor the peripherals, including the I/O ports, click on the ***Debug*** menu, choose ***Windows*** and then ***I/O view***.



7. The ***I/O*** tab appears on the right hand side 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.



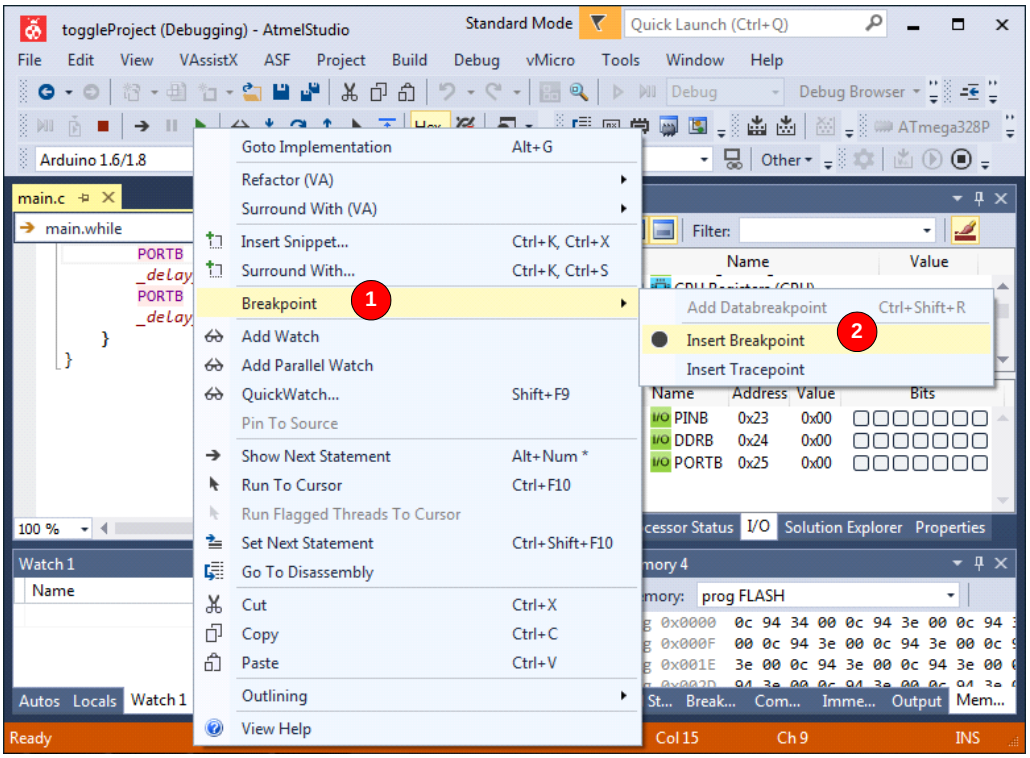
8. Press ***F10 (Step Over)*** a few times and see the PORTB register changes in the ***I/O***.

Using Breakpoints

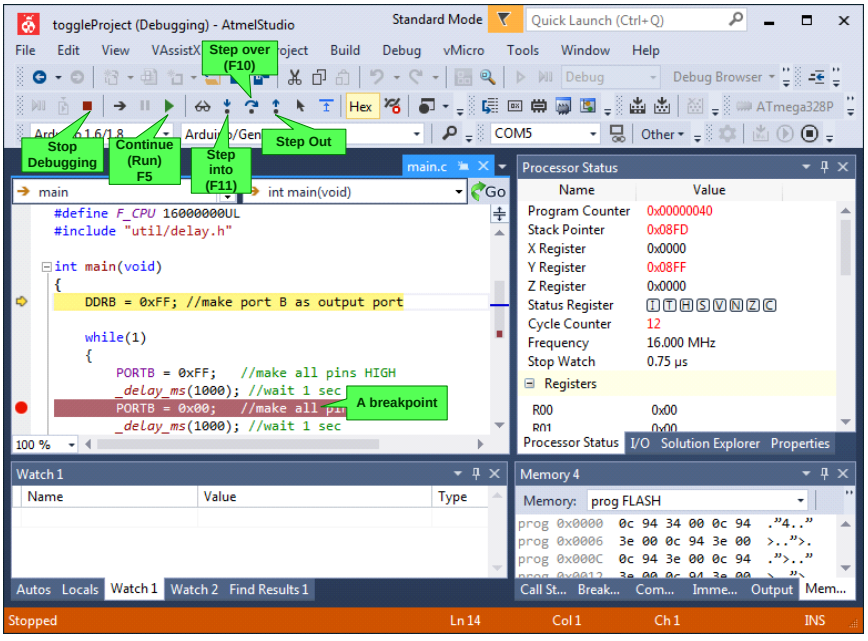
If you want to debug a portion of a program, add a breakpoint to the beginning of this part of the code and press the run button. The IDE runs the program and when it reaches the breakpoint, it

stops running and the yellow cursor is shown on the breakpoint line. Below, you see the steps in detail.

1. Right click on the "`PORTB = 0x00;`" instruction. A pop-up menu appears. Choose ***Breakpoint*** and then Insert Breakpoint. A red bullet appears on the left hand side of the "`PORTB = 0x00;`" instruction. To add a Breakpoint, you can also click on the left hand side of instructions where the red bullet appears.



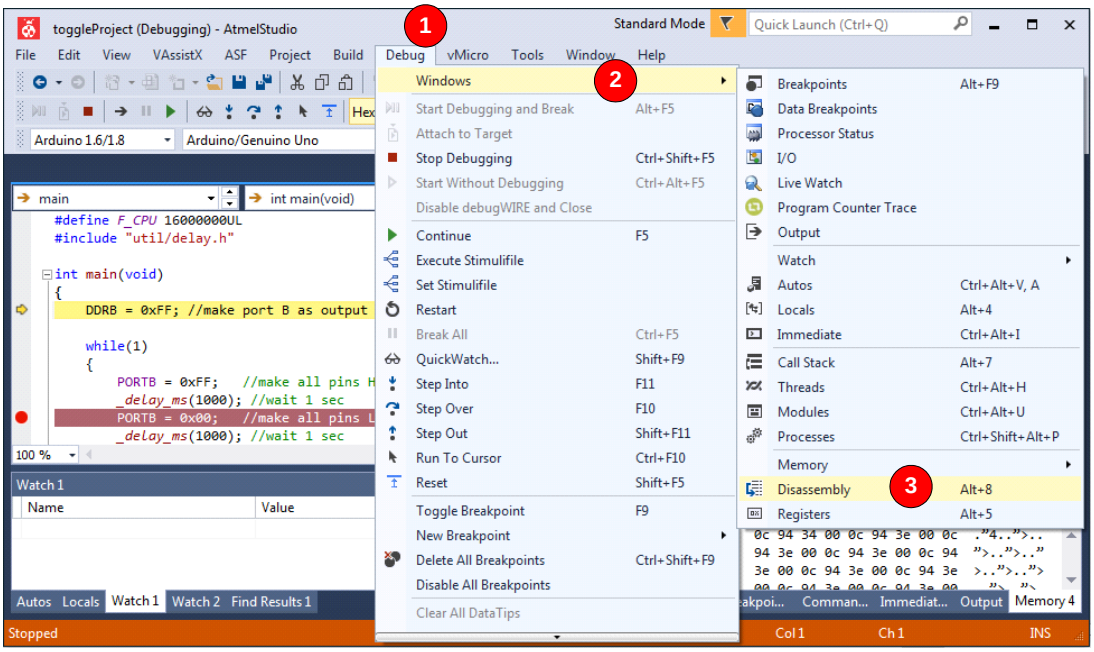
2. Press ***F5*** or the ***Continue*** button. The IDE runs program until it reaches the Breakpoint. Now, you can continue debugging from the breakpoint using the ***Step into*** and ***Step over*** buttons.



Disassembly

The Compiler converts C programs to machine instructions. To see the assembly equivalent of your program:

1. Click on the ***Debug*** menu, choose ***Windows*** and then ***Disassembly***.



2. The following window appears.
    a. The black texts show the C program instructions. The assembly equivalent of the C instruction is shown below, in gray.
    b. The first column shows the memory address where the assembly instruction is located.
    c. The second column shows the assembly instruction.