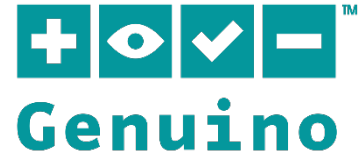


Microcontrollers - how to start?

Connect the Genuino Uno to the computer

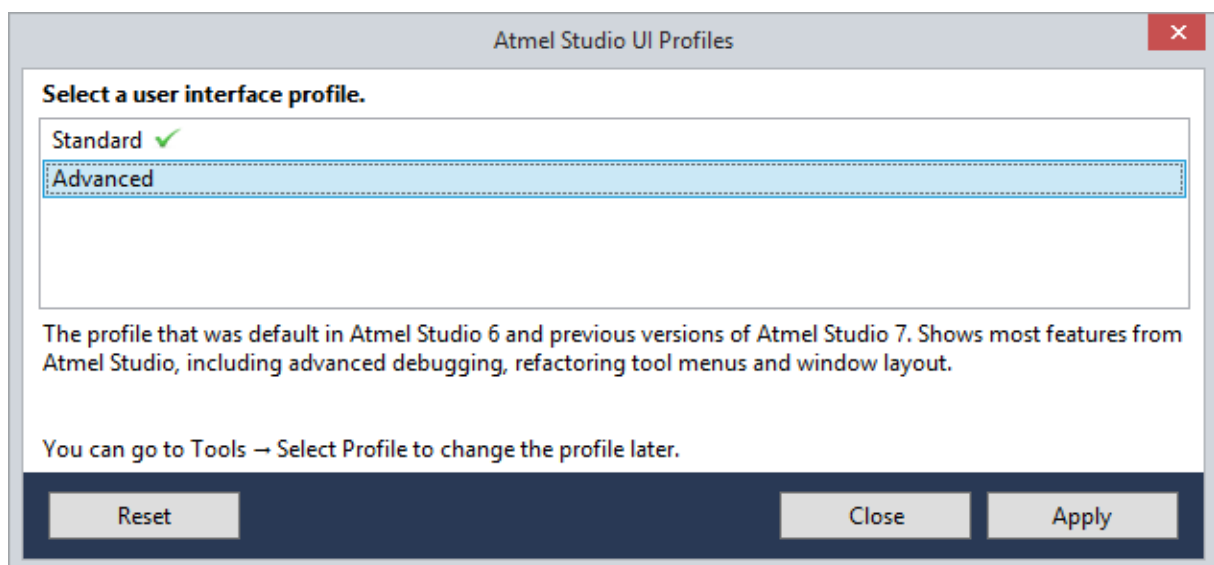
1. Insert the power cable into the Genuino Uno
2. Insert the USB cable (type-B plug) into the Genuino Uno
3. Insert the USB cable (type-A plug) into the computer
4. Wait about 15 seconds as such the Windows Management Instrumentation has the time to update.



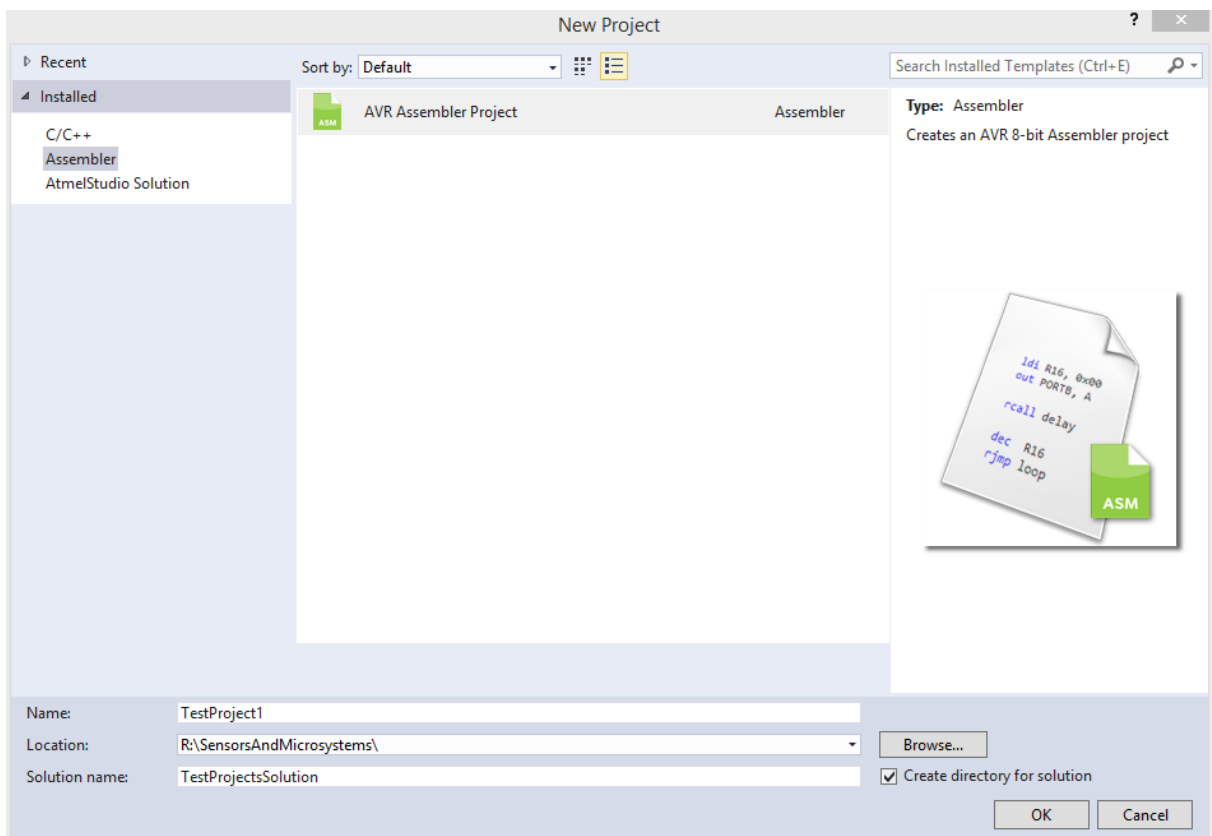
A Windows 7, 8, 8.1 or 10 operating system is required!

Programming: Atmel Studio 7

1. Start Atmel Studio 7.0
2. At first launch, atmel studio will ask you to use the standard or advanced interface, choose Advanced, click apply



3. Start a new project (File >> New >> Project)
4. Choose Assembler >> AVR Assembler Project
5. Choose a name for a new project (example: TestProject1)
6. Choose a location to save the project (tip: make a new folder on your personal U: drive) (example: SensorsAndMicrosystems)
7. Choose a solution name. A solution is a combination of several projects. (example: TestProjectsSolution). If you want to create a directory for the solution, check the square. If you created already a directory, uncheck the square.
8. Press OK
9. In Device Selection, choose ATmega328P (this is the microcontroller on the Genuino Uno)
10. Press OK >> Your solution and project are created



11. You can start coding now

Assembly coding: Atmel Studio 7.0

1. All information about assembly coding in Atmel Studio can be found on <http://www.atmel.com/webdoc/avrassembler/index.html>
2. Assembler directives that you need to use:
 - a. `.INCLUDE "m328pdef.inc"` : this directive includes the definition file of the microcontroller we will program. It contains the addresses of all System Function Registers (SFRs) and other definitions.
 - b. `.ORG` : Compiler command that tells the compiler to go to a specific address. The subsequent instruction will be written at this address. Keep in mind that the Atmega328P always boots at address 0x0000
 - c. `.DEF` : Compiler command to set a symbolic name as an alias for a register
 - d. `.EQU` : Compiler command to set a symbolic name as an alias for an expression
 - e. `RJMP` : Jump to a specific address or to the label which corresponds to an address
3. Some remarks:
 - a. Assembly language is case insensitive
 - b. A comment starts with a semicolon ("`;`"), or looks like "`/* some comment */`"
 - c. The code in the program memory of the Atmega328P is non-volatile; the data memory is volatile, except the EEPROM.

4. A good template to start from:

```
main.asm  [X]
;
; Template.asm
;
; Created: 9/02/2017 14:25:53
; Author : RobinDeleener
;
;
; Definition file of the ATmega328P
#include "m328pdef.inc"

; Your own register definitions
.def    JOYSTICK_POISITION = R2          ;give a meaningful label to R2

; Your own constants
.equ    NUMBER_OF_ROWS = 7              ;Define a constant value that can be used in the code
.equ    SCREEN_ARRAY_ADDRESS = 0x01000  ;Define the address of the first byte of the screen array

; Boot code (microcontroller starts @ adress 0x0000)
.org 0x0000
    rjmp init

; Interrupt address vectors
.org 0x0002
    rjmp ISR1



init: /*
      Put some initialisation code here
      */
    rcall MyFunction1
    rjmp main

main: /*
      Put your main program hereh
      */
    rjmp main    ;jump back to main to create an endless while loop

/* Interrupt handlers */
ISR1: /*
      Put your Interrupt Service Routine here
      */
    RETI          ;return from an interrupt

/* Own Functions */
MyFunction1:
    /*
      Put your function here
      */
    RET            ;return from a function

/* Code memory data */
```

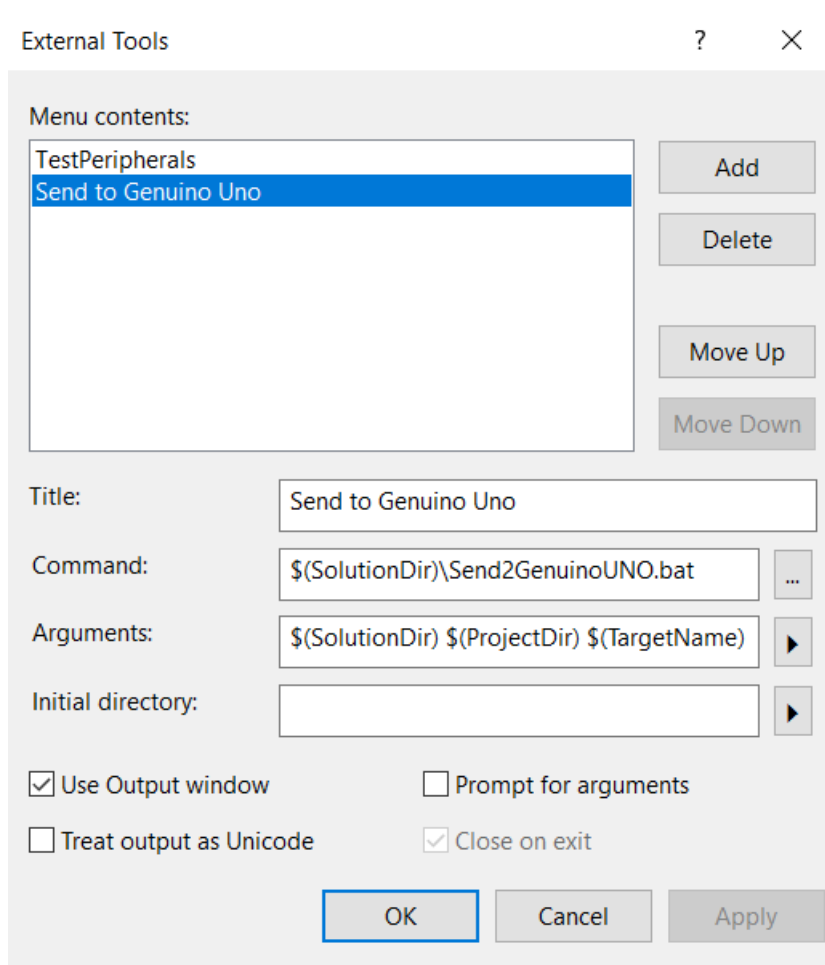
5. After writing you code, you should press “Build ***” ( or Build >> Build ***) in order to build the program and generate the binary code for the Atmega328P. You can also press “Build solution” ( or Build >> Build Solution or F7) which will build all projects in the current solution.
6. You are now ready to upload the instructions to the Genuino Uno.

Programming the Genuino Uno

1. Go to Canvas and download Software >> Portables.zip
2. Place the content of this archive in the solution folder

	avrdude	9/02/2017 14:36	File folder	
	Task1	9/02/2017 14:37	File folder	
	Send2GenuinoUNO	7/02/2017 20:07	Windows Batch File	2 KB
	StudentTasks_Arduino	6/10/2016 11:44	ATMEL Studio 7.0 ...	9 KB
	TestPeripherals	7/02/2017 20:56	Windows Batch File	2 KB
	TestPeripherals.hex	17/11/2016 13:35	HEX File	6 KB

3. Go to Tools >> External Tools
4. Copy the Title, Command and Arguments from the picture below (ignore the TestPeripherals)



5. Click Apply >> OK
(You only have to do steps 1-5 once)

6. Click on Tools >> Send to Genuino Uno
7. The instructions are uploaded to the Genuino Uno
8. If everything goes well, the output window shows:

```
Output
Show output from: Send to arduino UNO
avrdude.exe: verifying flash memory against R:\SensorsAndMicrosystems\TestProjectsSolution\TestProject1\Debug\TestProject1.hex:
avrdude.exe: load data flash data from input file R:\SensorsAndMicrosystems\TestProjectsSolution\TestProject1\Debug\TestProject1.hex:
avrdude.exe: input file R:\SensorsAndMicrosystems\TestProjectsSolution\TestProject1\Debug\TestProject1.hex contains 14 bytes
avrdude.exe: reading on-chip flash data:

Reading | ##### | 100% 0.02s

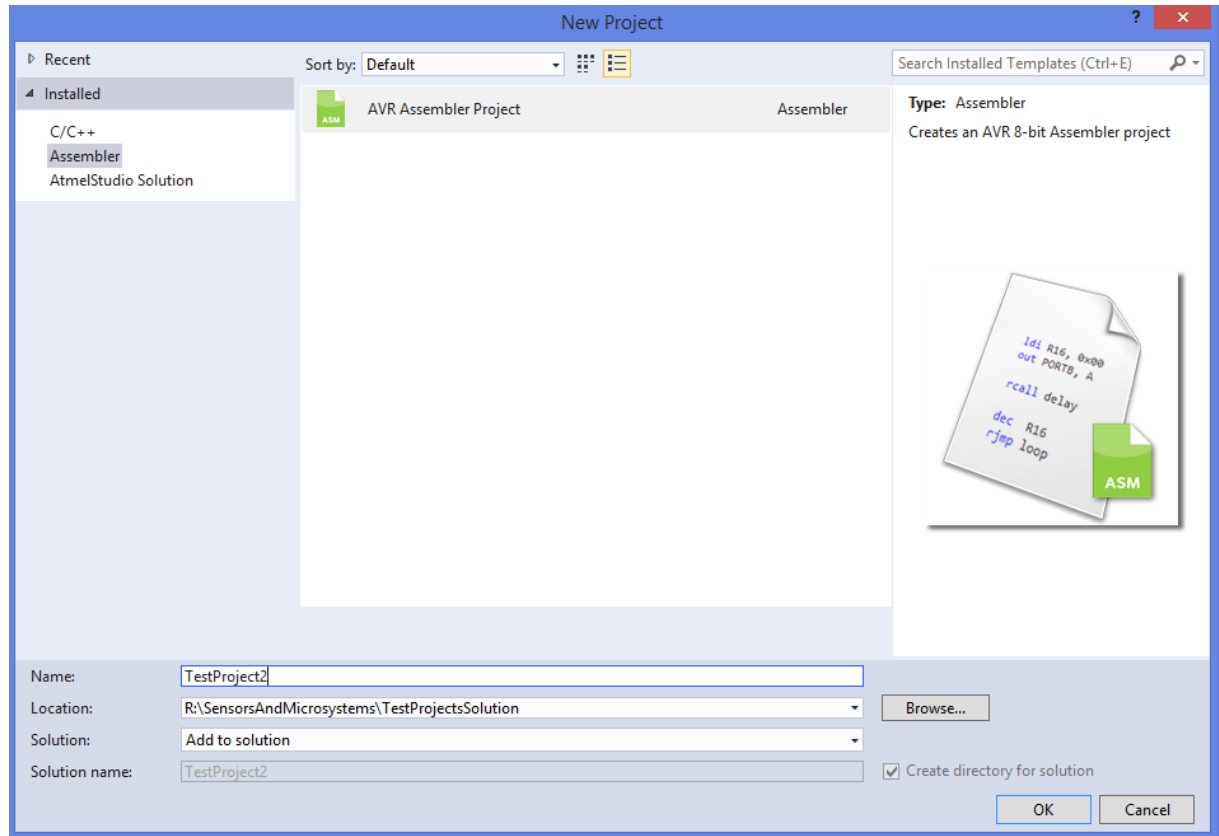
avrdude.exe: verifying ...
avrdude.exe: 14 bytes of flash verified



avrdude.exe: safemode: Fuses OK (H:00, E:00, L:00)

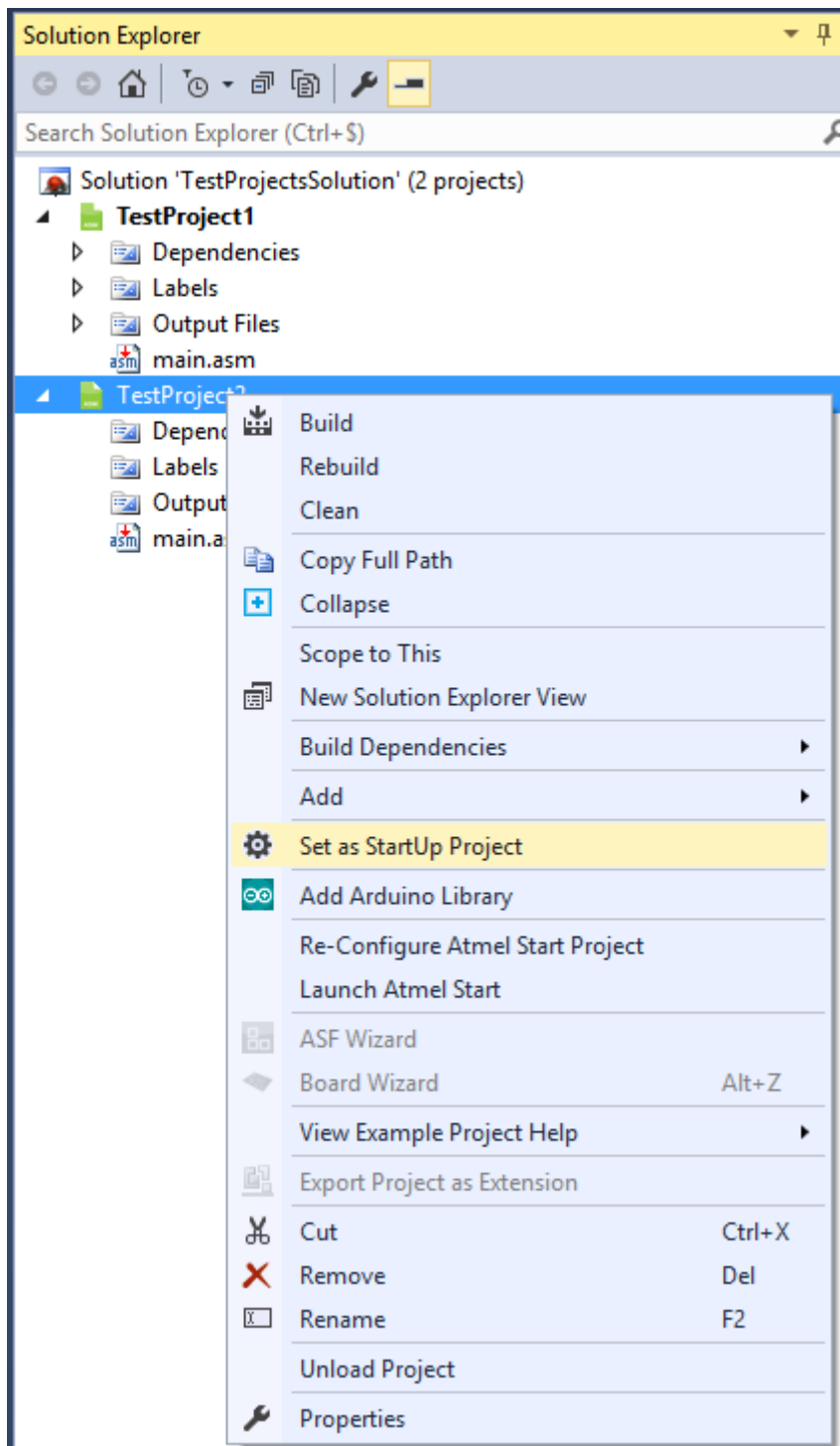
avrdude.exe done. Thank you.
```

Adding a new project to the solution

1. Start a new project (File >> New >> Project)
2. Choose Assembler >> AVR Assembler Project
3. Choose a name for this project (example: TestProject2)
4. Choose a location to save the project, this can be the same location as before because Atmel will create a new folder with the new project name
5. Solution >> Add to solution



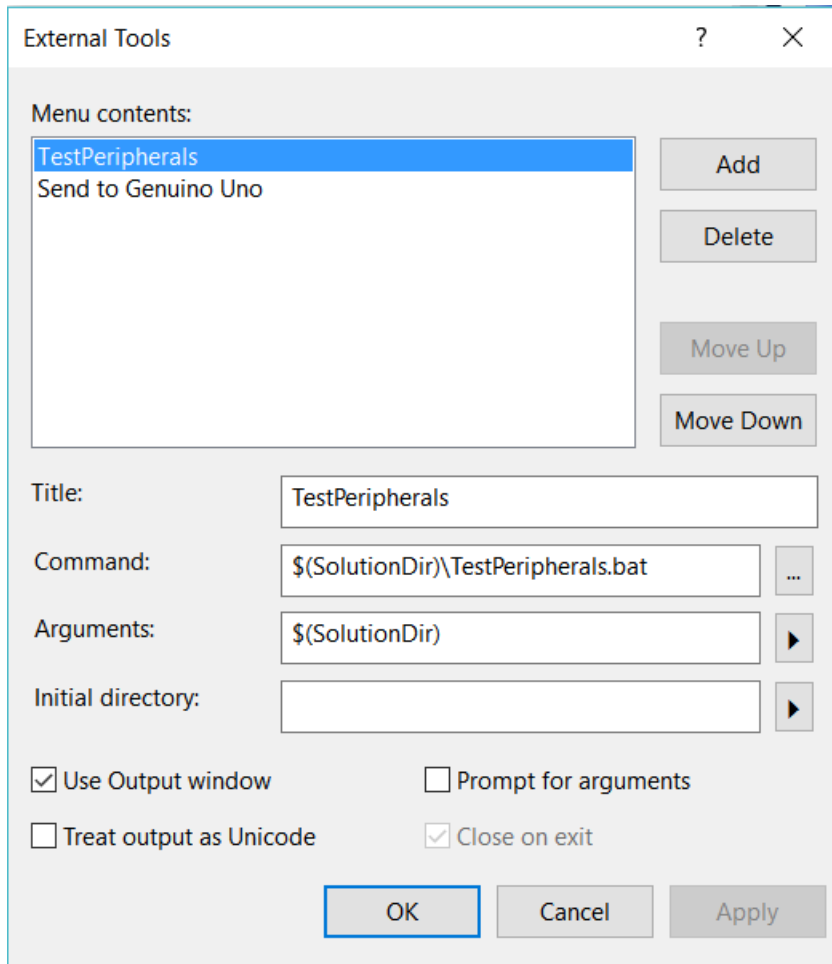
6. Press OK
7. In Device Selection, choose ATmega328P (this is the microcontroller on the Genuino Uno)
8. Press OK. The project is created and added to the solution
9. You can start coding now.
10. After writing you code, you should press "Build ***" ( or Build > Build ***) in order to build the code and generate the binary code for the Atmega328P. You can also press "Build solution" ( or Build > Build Solution or F7) which will build all projects in the current solution.
11. You are now ready to upload the instructions to the Genuino.
12. To choose which project in the solution you want to send to the Genuino Uno, you right-click on the project that you want to send and choose "Set as StartUp project". The project name becomes bold after that.



9. Click on Tools >> Send to Genuino Uno
10. The instructions are now uploaded to the Genuino Uno

Testing the board in case of doubt

1. We have created a program to test all the functionalities of the board. In case you have troubles and want to check that the board is working correctly, you can upload this program to the Genuino Uno.
2. Go to Tools >> External Tools
3. Click Add and copy the Title, Command and Arguments from the picture below:



4. Click Apply >> OK
(You only have to do steps 1-4 once)
5. Click on Tools >> TestPeripherals
6. The program is uploaded to the Genuino Uno and you can test all functionalities of the board

!!! GOOD LUCK !!!