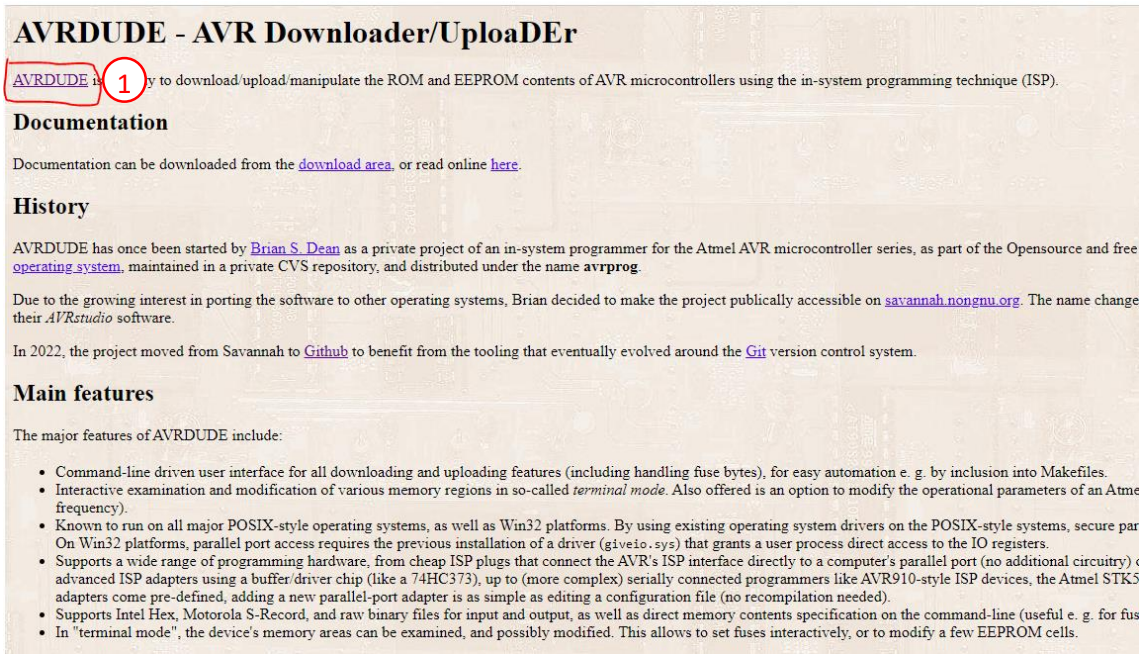
Contenido

Programming the Arduino UNO using ATMEL Studio

Downloading the software programmer

1. By programming Arduino Uno board in Atmel Studio an external tool called AVRdude.exe must be added to the computer. AVRdude.exe can be downloaded at https://www.nongnu.org/avrdude/.



AVRDUDE - AVR Downloader/UploaDEr

AVRDUDE is (1) to download/upload/manipulate the ROM and EEPROM contents of AVR microcontrollers using the in-system programming technique (ISP).

Documentation

Documentation can be downloaded from the download area, or read online here.

History

AVRDUDE has once been started by Brian S. Dean as a private project of an in-system programmer for the Atmel AVR microcontroller series, as part of the Opensource and free operating system, maintained in a private CVS repository, and distributed under the name **avrprog**.

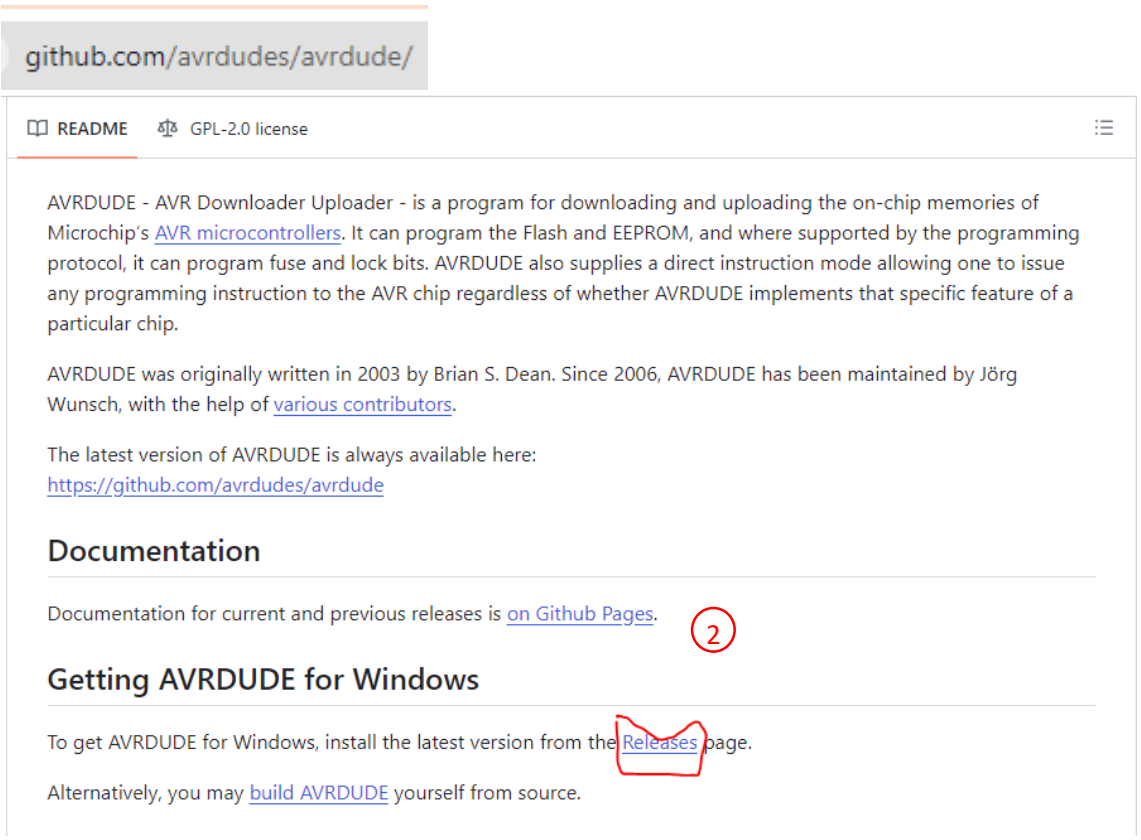
Due to the growing interest in porting the software to other operating systems, Brian decided to make the project publically accessible on savannah.nongnu.org. The name change their *AVRstudio* software.

In 2022, the project moved from Savannah to Github to benefit from the tooling that eventually evolved around the Git version control system.

Main features

The major features of AVRDUDE include:

- Command-line driven user interface for all downloading and uploading features (including handling fuse bytes), for easy automation e. g. by inclusion into Makefiles.
- Interactive examination and modification of various memory regions in so-called *terminal mode*. Also offered is an option to modify the operational parameters of an Atme frequency).
- Known to run on all major POSIX-style operating systems, as well as Win32 platforms. By using existing operating system drivers on the POSIX-style systems, secure par On Win32 platforms, parallel port access requires the previous installation of a driver (`giveio.sys`) that grants a user process direct access to the IO registers.
- Supports a wide range of programming hardware, from cheap ISP plugs that connect the AVR's ISP interface directly to a computer's parallel port (no additional circuitry) advanced ISP adapters using a buffer/driver chip (like a 74HC373), up to (more complex) serially connected programmers like AVR910-style ISP devices, the Atmel STK5 adapters come pre-defined, adding a new parallel-port adapter is as simple as editing a configuration file (no recompilation needed).
- Supports Intel Hex, Motorola S-Record, and raw binary files for input and output, as well as direct memory contents specification on the command-line (useful e. g. for fus
- In "terminal mode", the device's memory areas can be examined, and possibly modified. This allows to set fuses interactively, or to modify a few EEPROM cells.



github.com/avrdudes/avrdude/

README GPL-2.0 license

AVRDUDE - AVR Downloader Uploader - is a program for downloading and uploading the on-chip memories of Microchip's AVR microcontrollers. It can program the Flash and EEPROM, and where supported by the programming protocol, it can program fuse and lock bits. AVRDUDE also supplies a direct instruction mode allowing one to issue any programming instruction to the AVR chip regardless of whether AVRDUDE implements that specific feature of a particular chip.

AVRDUDE was originally written in 2003 by Brian S. Dean. Since 2006, AVRDUDE has been maintained by Jörg Wunsch, with the help of various contributors.

The latest version of AVRDUDE is always available here:
https://github.com/avrdudes/avrdude

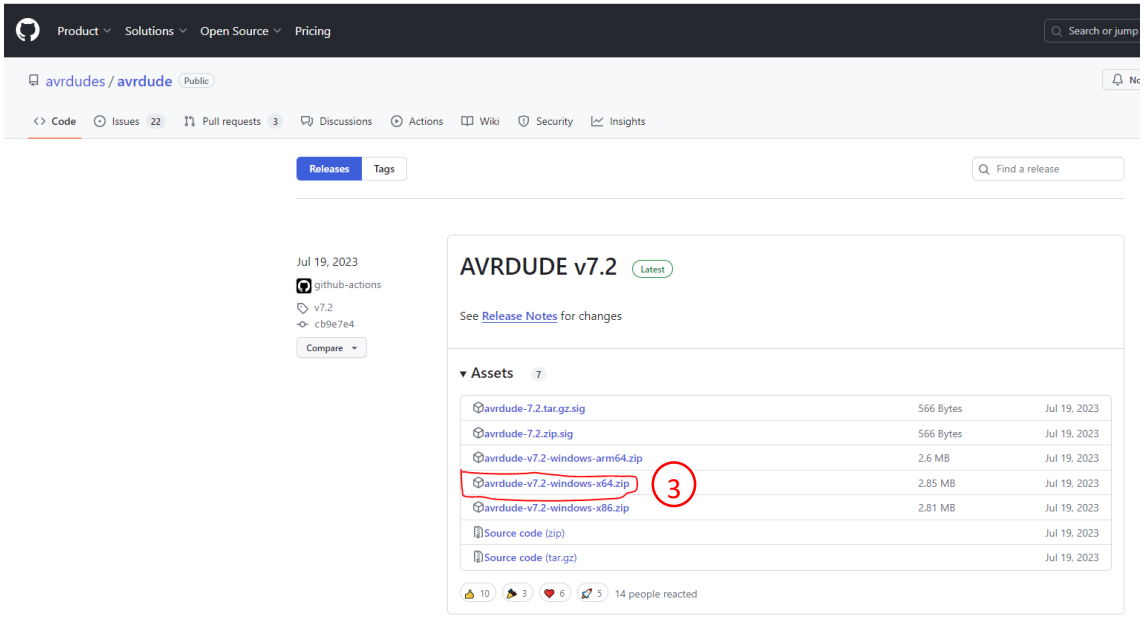
Documentation

Documentation for current and previous releases is on Github Pages. (2)

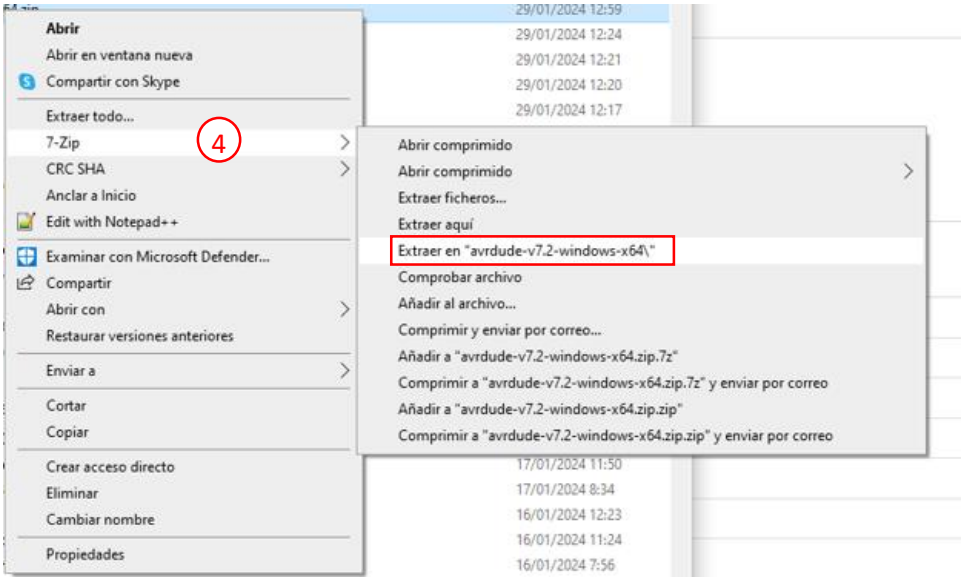
Getting AVRDUDE for Windows

To get AVRDUDE for Windows, install the latest version from the Releases page.

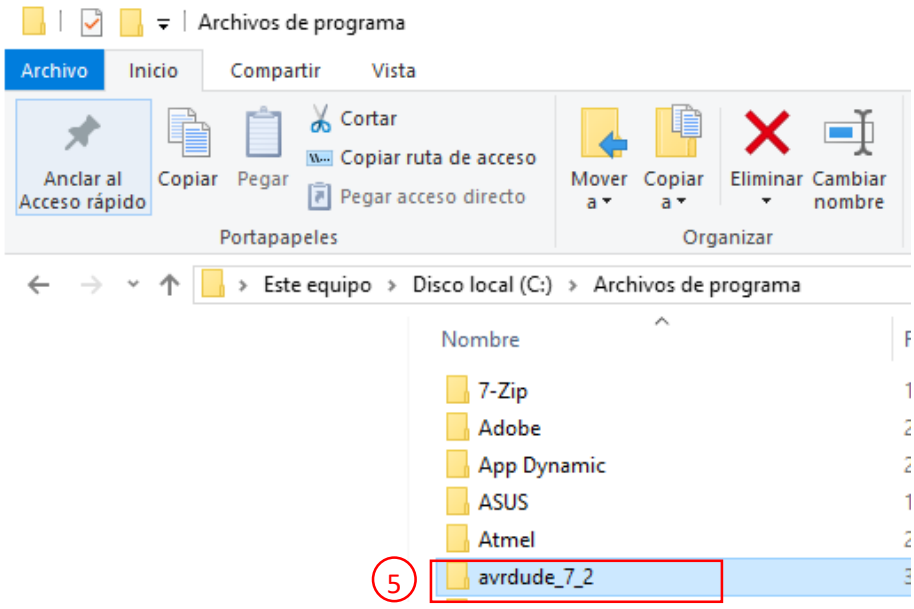
Alternatively, you may build AVRDUDE yourself from source.



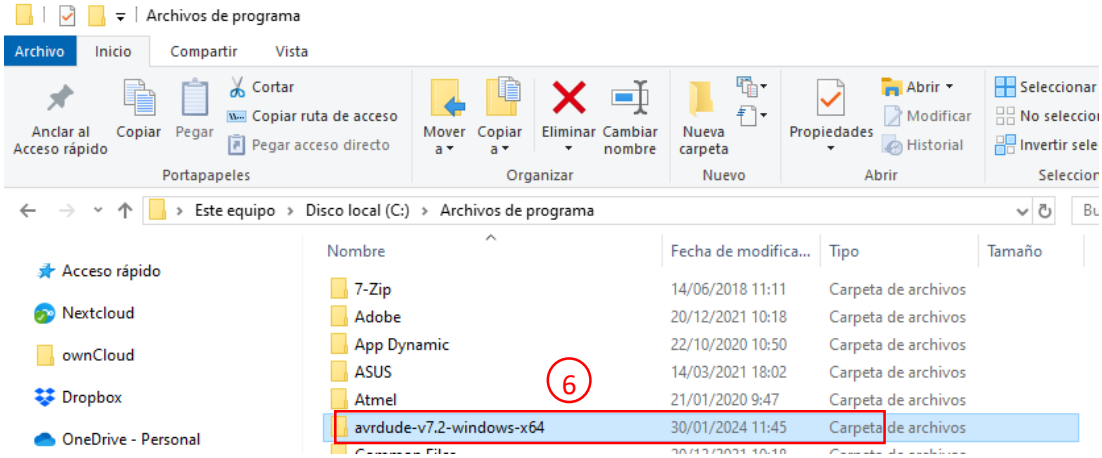
2. Now, unzip the file in the same folder. Be aware that your selection is → **Extraer en "avrdude-v7.2-windows-x64\"**



3. Modify the original name, “avrdude-v7.2-windows-x64”, and write the new name as **“avrdude_7_2”.**



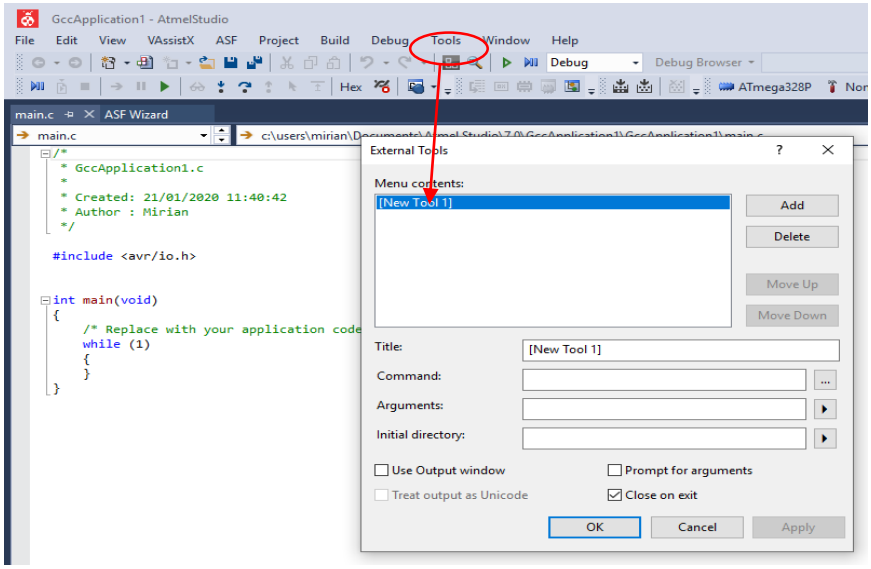
4. Once the “AVRdude” folder has been renamed, move it to the “**c:\Archivos de programa**” folder.



Creating a new Tool in Atmel Studio

In Atmel Studio create a new external tool as follows:

1. Select “Tools→External Tools…”



2. Type a name for the new tool in the field “Title”, “ArduinoUNO”.

3. In the “Command” field type the path for “AVRdude.exe”.

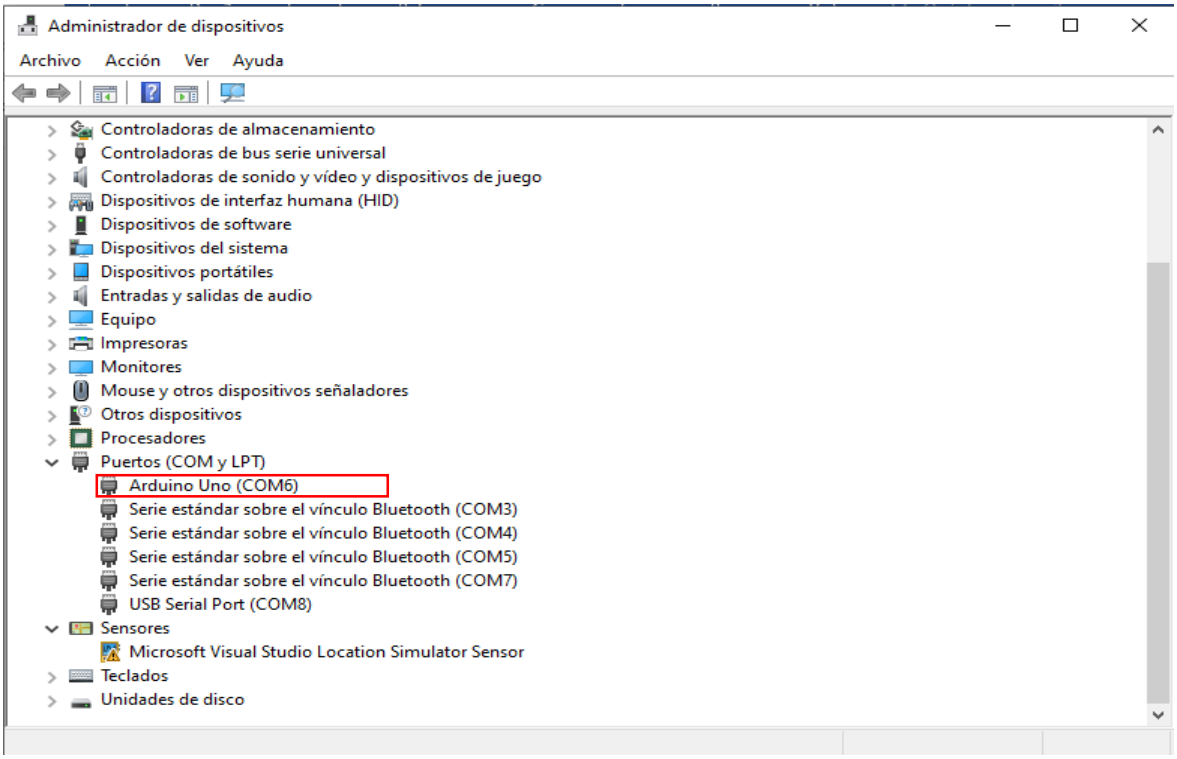
```
C:\Program Files\avrdude_7_2\avrdude.exe
```

4. In the “Arguments” field type the arguments you can see below.

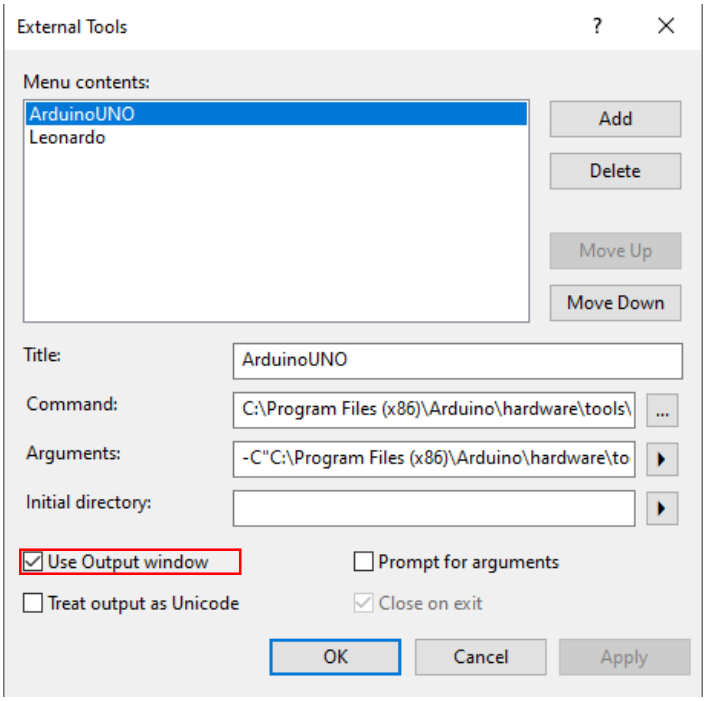
```
-c "C:\Program Files\avrdude_7_2\avrdude.conf" -v -patmega328p
-carduino -PCOM3 -b115200 -D -U
flash:w:"$(ProjectDir)Debug\$(TargetName).hex":i
```

Be aware of copy-paste the “Arguments” from this document to the Atmel Studio as some hidden characters will be included in the command line. As a result, the External Tool created is not going to work.

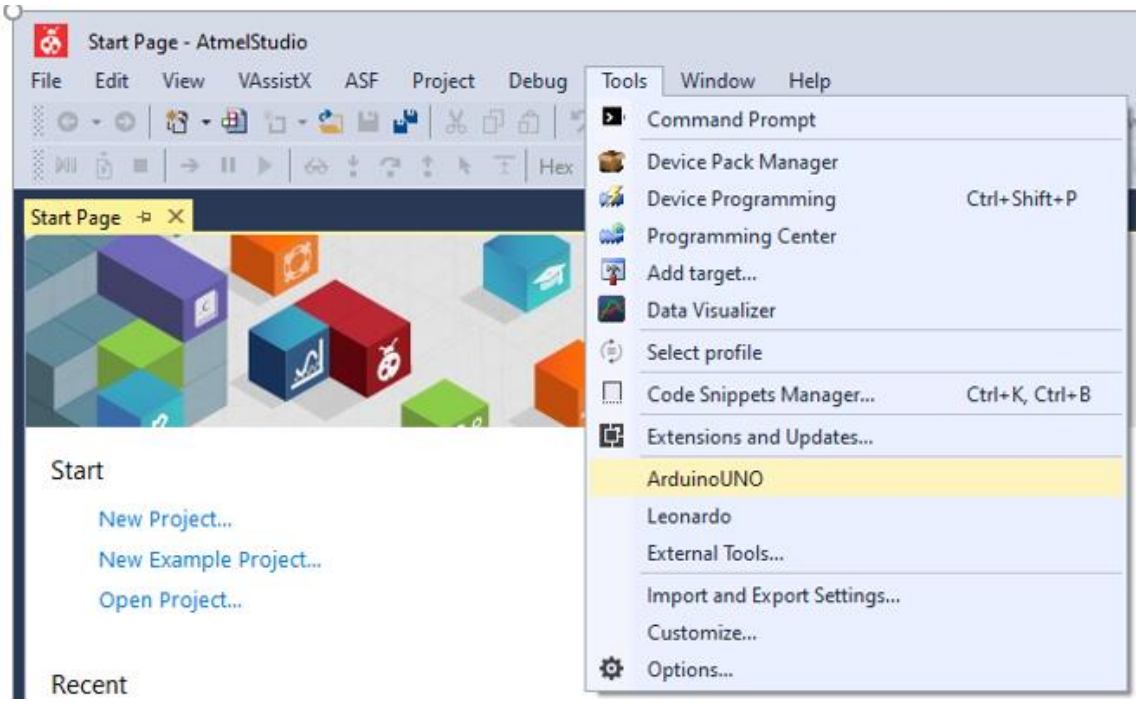
5. To make the parameters compatible with the Atmel Studio environment and our computer, we need to make a small modification. This involves adjusting the serial number port for the COM, which depends on the port number assigned to the Arduino board by Windows. To do this, connect the Arduino board to your computer and check “Administrador de dispositivos” to see which number has been assigned to it. This will ensure that the correct port is being used.



6. Make sure the “Use output windows” is selected.



7. Now, the new tool “**ArduinoUNO**” is available to be used in the Tools menu.



AVRdude setup

"AVRdude" has many switches to modify its setup, the following are explained those have been used in this tutorial.

- **-p <partno>:** This is just to tell it what microcontroller it's programming. For example, if you are programming an ATtiny2313, use attiny2313 as the part number
- **-C <config-file>:** The configuration file tells "AVRrdude" about all the different ways it can talk to the programmer. There is a default configuration file. The path of the configuration file must be typed.
- **-c <programmer>:** Here is where we specify the programmer type, if you're using an STK500 use **stk500**, if you're using a DT006 programmer use **dt006**, etc.
- **-D:** This disables erasing the chip before programming. We don't want that so don't use this command switch.
- **-P <port>:** This is the communication port to use to talk to the programmer. It might be COM1 for serial or LPT1 for parallel or USB for, well, USB.
- **-v:** This gives you 'verbose' output...in case you want to debug something. If you want you can use it, but in general we won't.
- **-U <memtype>:r|w|v:<filename>[:format]:** It is the parameter that actually does the programming. The **<memtype>** is either **flash** or **eeprom** (or **hfuse**, **lfuse** or **efuse** for the chip configuration fuses. The **r|w|v** means you can use **r** (read) **w** (write) or **v** (verify) as the command. The **<filename>** is, well, the file that you want to write to or read from. and **[:format]** means there is an optional format flag. We will always be using "Intel Hex" format, so use **i** So, for example, if you wanted to write the file **test.hex** to the flash memory, you would use **-U flash:w:test.hex:i**. If you wanted to read the EEPROM memory into the file "eedump.hex" you would use **-U eeprom:r:eedump.hex:i**