

Installing programming tools on Windows machines

3/31/24

ECE-231 Part 2 - 8 Bit Systems, Spring 2024

Version 2.0

This note provides instructions for downloading the VSCode editor and the WinAVR toolchain (compiler and other tools) onto a computer running Windows. You should be using an up-to-date version of Windows such as Win10 or Win11.

VS Code text editor

Visual Studio Code (aka VS Code) is a no-cost text editor owned by Microsoft. This is a popular editor among programmers because it has a number of features (eg, color coding, indentation, syntax, error checking, code folding, file management, etc..) that help to develop source code. Note that VS Code is not the same as another similarly-named Microsoft product, Visual Studio. Studio is a full blown Integrated Development Environment, while VS Code is simply a text editor that happens to have many features helpful for creating source code.

Installing VS Code.

1. Goto <https://code.visualstudio.com/docs/setup/windows> and follow the Installation instructions there to download and install VS Code to your Applications folder. The installer will add VS Code to your environmental PATH variable by default.

2. To test your installation, quit VS Code. Then open the Command Prompt and type

```
> code .
```

(Note: In this document, I have highlighted in yellow whenever you have something to type on the command line. The >, or a similar symbol, is the prompt you will see on your Command Prompt terminal. Don't type it. Also: don't forget the space followed by the period after code. When you hit return, VS Code should open.

3. Under the VS Code file menu, be sure that autosave is checked. If not, check it.

4. Pin the Command Promot to your dock, since we'll be using it frequently. Quit terminal and quit VS Code.

WinAVR toolchain

Use the WinAVR compiler and a set of associated tools (collectively called a tool-chain) on for Windows machines. This GNU toolchain is freely available from the Sourceforge web site. Goto <https://sourceforge.net/projects/winavr/files/> and download the file WinAVR-20200220-install.exe. Open the install file and the toolchain will be installed. Accept all defaults. The avr-gcc compiler, make, avrdude, and other files will be installed. Below, we will check that key files have been installed correctly:

avr-gcc compiler

Open your Command Prompt and check the installation by typing:

```
> which avr-gcc.exe
```

(NOTE: Commands you need to type at the Command Prompt are highlighted in yellow. The > at the beginning represents the Command Prompt. Don't type that.)

and you should see something like

```
C:\WinAVR-20100110\bin\avr-gcc.exe
```

Now type

```
> avr-gcc --version
```

and you should see something like

```
avr-gcc (WinAVR 20100110) 4.3.3
```

make

Make is a program that automates the multi-step process of compiling, linking, copying and flashing a program to a microcontroller. A version of make will have been installed as part of WinAVR. To test you make program, type

```
>which make.exe
```

and you should see

```
C:\WinAVR-20100110\utils\bin\make.exe
```

Type

```
>make --version
```

And you'll see

GNU Make 3.81

avrdude

avrdude (avr downloader/uploader) is the program used to flash compiled code to an AVR microcontroller. This will have been installed with WinAvr. Check the avrdude installation by typing

```
>which avrdude.exe
```

and you should see

```
C:\WinAVR-20100110\bin\avrdude.exe
```

finally, type

```
>avrdude
```

And you'll see a screen full of options.

All this means you have successfully installed the avr toolchain. (Be glad you're not using a Mac, since this process is substantially more complicated on macOS).

Build & Flash your first program

If everything so far has been successful, you are now able to write a source code file in VS Code then compile it from the command line and download the machine code to the flash ROM of your AVR MCU. Create the classic “blink” project to test everything. The following instructions assume you are coding for an ATmega328p MCU on an Arduino Uno development (dev) board.

1. First, make a directory called “codingprojects” in your home directory. This is where you will be storing all your projects during the semester. Next, move into this new directory. Next, create a new directory called “blink” for the project that you will now build. Move into that

directory. Then open VS Code from that directory. You do all this with the following Command Prompt terminal commands:

```
> mkdir codingprojects  
> cd codingprojects  
> mkdir blink  
> cd blink  
> code .
```

This will open up VS Code in the blink project folder. Create a source code file called "blink.c" by first hovering, then clicking on the + file box toward the top of the window.

2. Next, type in the source code for a blinking LED. You can copy it from here:

<https://github.com/ProfMcL/ECE231/blob/main/blink/blink.c>

3. Now, compile the code by typing:

```
> avr-gcc -Wall -Os -DF_CPU=16000000 -mmcu=atmega328p -o main.elf blink.c
```

If there are any errors, fix them and re-type the line above.

4. Next, type:

```
> avr-objcopy -j .text -j .data -O ihex main.elf main.hex
```

This will create a text file containing a hexadecimal (base 16) version of the compiled source code that is understandable by the MCU. In the next step, you will upload this hex file to the flash ROM of the MCU. This is called "flashing" the MCU.

5. Connect your Arduino Uno to your computer using the USB cable. Determine which COM port the Uno is connected to by opening Device Manager from the Start menu, then clicking on Ports (COM & LPT) you should see something like USB Serial Device (COM4). Flash the code to the ATmega328P mcu by typing

```
> avrdude -c Arduino -b 115200 -P COMX -p atmega328p -U flash:w:main.hex:i
```

In place of X type the number of your COM port identified in the previous step. You should now see the internal LED on the Arduino board blinking on and off at 1 second each.

6. Now modify the code by changing MYDELAY to 100.

Redo the commands `avr-gcc`, `avr-objcopy`, and `avrdude` and the LED should be blinking much faster. Note: you do not need to type in these commands. Position your cursor in the terminal prompt line and use the up and down arrows to scross through previous commands.

7. You have undoubtedly noted that the three lines we're using to compile and flash are tedious to type. A makefile is a text file that contains a set of instructions for building code without needing to type in all the command line steps. Download the makefile from here: <https://github.com/ProfMcL/ECE231/blob/main/blink/makefile> and move th makefile into your blink project folder. You will notice that this file is now listed along with `blink.c`, `blink.hex`, and other files in the explorer panel of VS Code. Your laptop may have appended `.txt` or another extension to makefile during the download process. If so, right-click on `makefile.txt` and change its name to `makefile` without any extension.

8. Read through the makefile using VSCode and update the `PORT` variable to the correct COM port for your Arduino Uno. You can find this port in your Device Manager.

9. Go back to `blink.c` and change `MYDELAY` to some other value, such as 3000, corresponding to 3 seconds.

From the terminal command line, type

```
> make
```

This will compile your code. Then type

```
> make flash
```

This will flash your code to your device.

10. Unplug your USB cable from your laptop and plug it into a wall outlet via a USB adapter if you have one.

Congratulations on your first embedded coding project!

Document History

Revised on	Version	Author	Description
2/13/23	1.0	D. McLaughlin	Initial document creation
2/17/23	1.1	D. McLaughlin	added instruction 6.6 related to PORT in makefile
3/31/24	2.0	D. McLaughlin	updates for Spring 2024