Installing programming tools on Windows machines
2/27/24
ECE-304 Spring 2024

Here are the instructions for installing VSCode and the compiler & tool chain for AVR MCU's onto your Windows machine.

VS Code text editor. We will be using *Visual Studio Code (aka VS Code)* as a text editor to write source code. 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.

WinAVR toolchain. This is the *avr-gcc* compiler and assorted tools for Windows machines. This GNU[1] toolchain freely available from the sourceforge web site.

PART 1. Install VS Code.

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.

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.

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

[1] GNU is a recursive acronym for "GNU's Not Unix!" and refers to the GNU Operating System supported by the Free Software Foundation. (A gnu -- lowercase letters -- is a wildebeest, which is a type of antelope.) GNU is pronounced as one syllable with a hard g, like "grew" but "gnew". See https://www.gnu.org/gnu/gnu.html if interested.

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

PART 2. Install the WinAVR compiler from SourceForge

Goto https://sourceforge.net/projects/winavr/files/
and download the file WinAVR-20100110-install.exe
Open the install file and the toolchain will be installed. Accept all defaults.

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
```

Also 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
```

Type

```
>which avrdude.exe
```

and you'll 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).

3. There is no step 3 for Windows!

4. There is no step 4 for Windows!

5. There is no step 5 for Windows!

6. Build your first program.

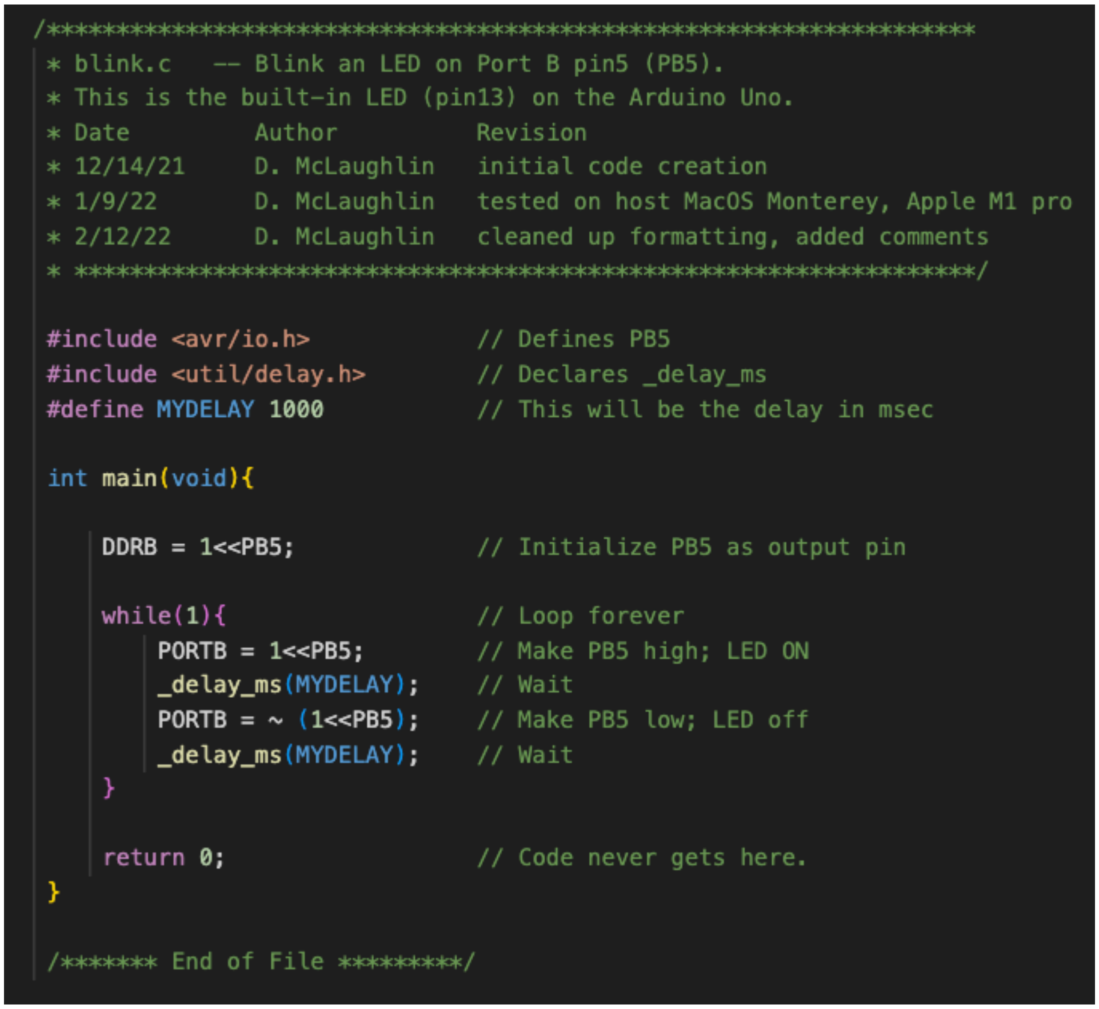
If everything so far has been successful, you are now able to write a source code file in VS Code then compile it from the command line and download the machine code to the flash ROM of the ATmega328P on the Arduino Uno development board. Create the classic "blink" project to test everything.

6.1 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.

6.2 Next, type in the source code shown. You do not need to type in the comments.



```c
/*******************************************************************
 * blink.c   -- Blink an LED on Port B pin5 (PB5).
 * This is the built-in LED (pin13) on the Arduino Uno.
 * Date         Author          Revision
 * 12/14/21     D. McLaughlin   initial code creation
 * 1/9/22       D. McLaughlin   tested on host MacOS Monterey, Apple M1 pro
 * 2/12/22      D. McLaughlin   cleaned up formatting, added comments
 * *****************************************************************/

 #include <avr/io.h>            // Defines PB5
 #include <util/delay.h>        // Declares _delay_ms
 #define MYDELAY 1000           // This will be the delay in msec

 int main(void){

     DDRB = 1<<PB5;             // Initialize PB5 as output pin

     while(1){                  // Loop forever
         PORTB = 1<<PB5;        // Make PB5 high; LED ON
         _delay_ms(MYDELAY);    // Wait
         PORTB = ~ (1<<PB5);    // Make PB5 low; LED off
         _delay_ms(MYDELAY);    // Wait
     }

     return 0;                  // Code never gets here.
 }

 /******* End of File *********/
```

6.3 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. Then type:

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

Next, 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).

Now 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.

6.4 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 cursar in the terminal prompt line and use the up and down arrows to scross through previous commands.

6.5 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 the Week 2 tab of our course Moodle site and move that file to 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.

6.6 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.

6.7 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.

6.8 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