

ECE 375 Windows Guide

Dongjun Lee

August 2022

1 Programs

Assuming you are running a Windows OS, you will need two basic programs in order to assemble and program your ATmega32U4 chip on the ECE375 TekBots board.






1. avrdude: [AVRDUDE](#) is the standard program for flashing programs onto AVR chips. This will be used to program your ATmega32U4 chip with your compiled code. To get the latest version of AVRDUDE for Windows, go to [the releases folder in github](#): <https://github.com/mariusgreuel/avrdude/releases>



Download the zip file **for your architecture** (e.g., x86, x64, or ARM).

AVRDUDE v7.0-windows Latest

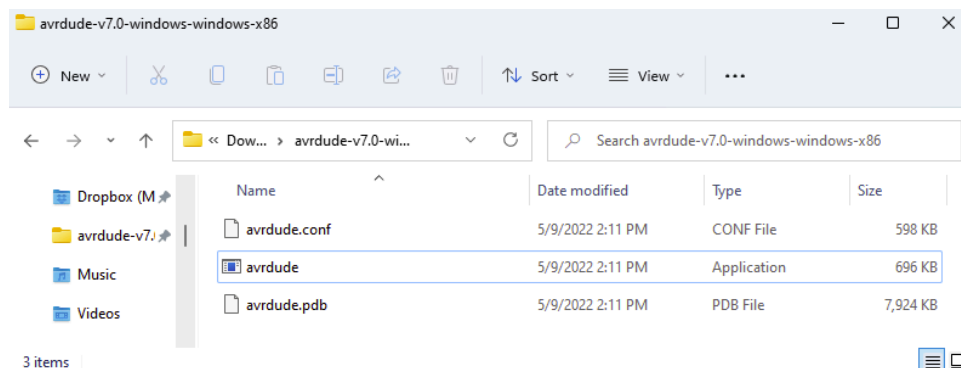
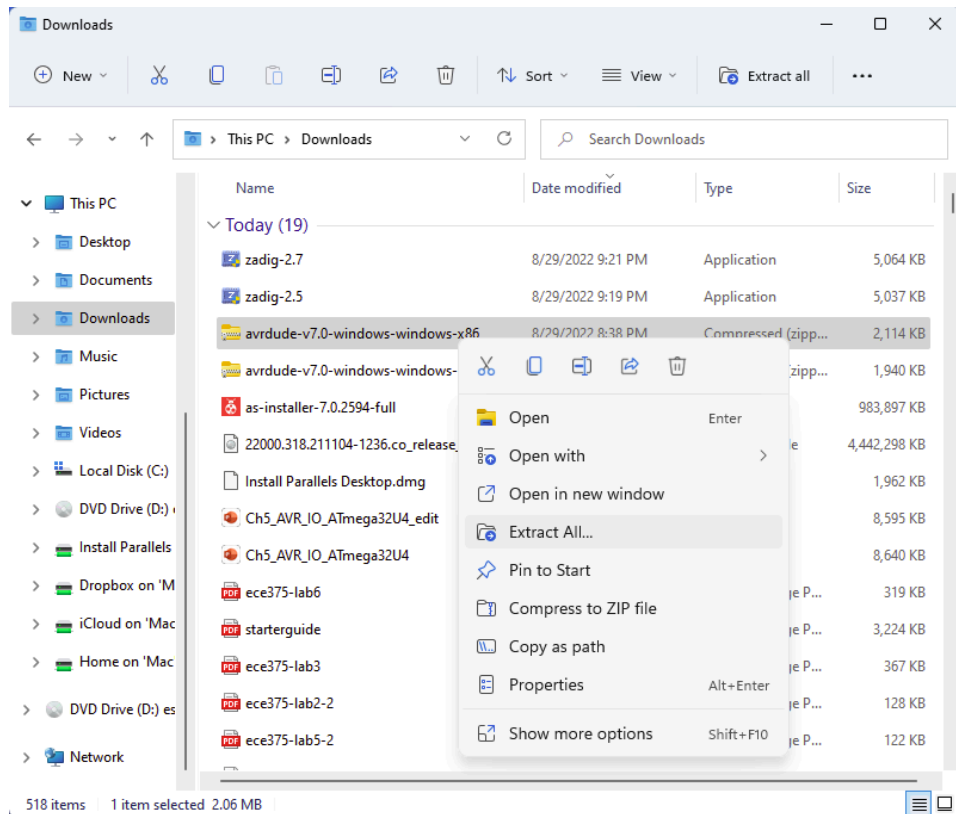
See [Release Notes](#) for changes

▼ Assets 5

 avrdude-v7.0-windows-windows-arm64.zip	1.89 MB	May 09, 2022
 avrdude-v7.0-windows-windows-x64.zip	2.05 MB	May 09, 2022
 avrdude-v7.0-windows-windows-x86.zip	2.06 MB	May 09, 2022
 Source code (zip)		May 09, 2022
 Source code (tar.gz)		May 09, 2022

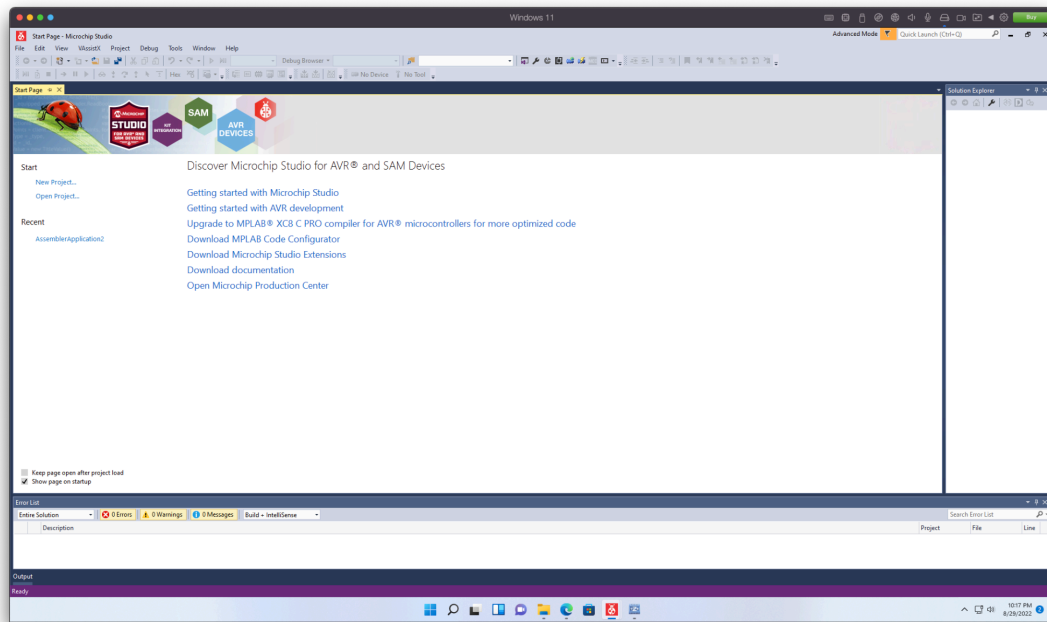
 2  4 5 people reacted

Extract the file in your preferred directory:

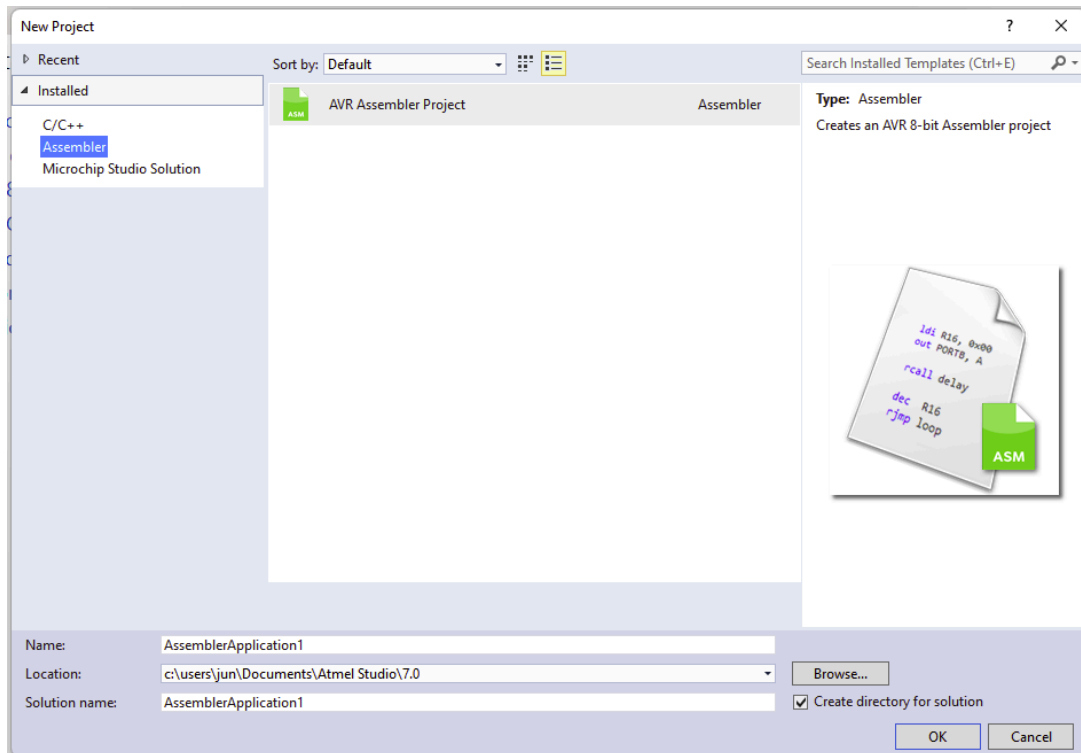


2. **Microchip Studio:** Microchip Studio is an IDE for developing and debugging AVR microcontroller applications. You can use Microchip Studio with the debuggers, programmers and development kits. Download **Microchip Studio Installer** here;
<https://www.microchip.com/en-us/tools-resources/archives/avr-sam-mcus>

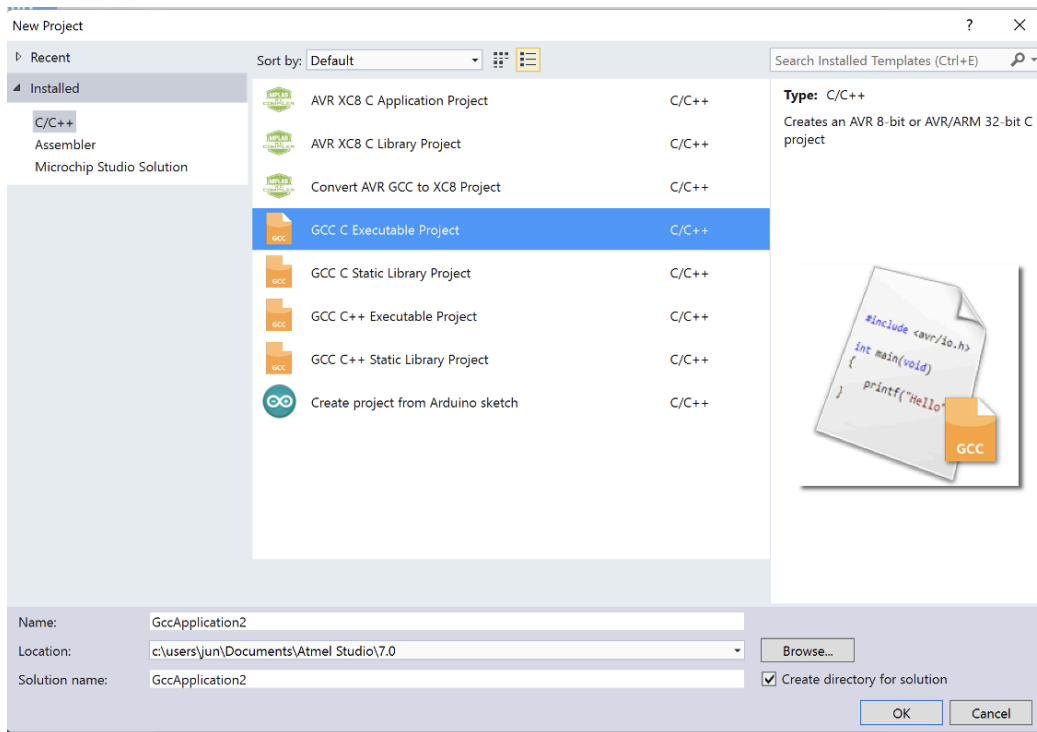
Install and **launch** the Microchip Studio. Microchip Studio should display a Start Page. To create a new AVR project, click on the **New Project...**



Select **AVR Assembler Project** as the project type.

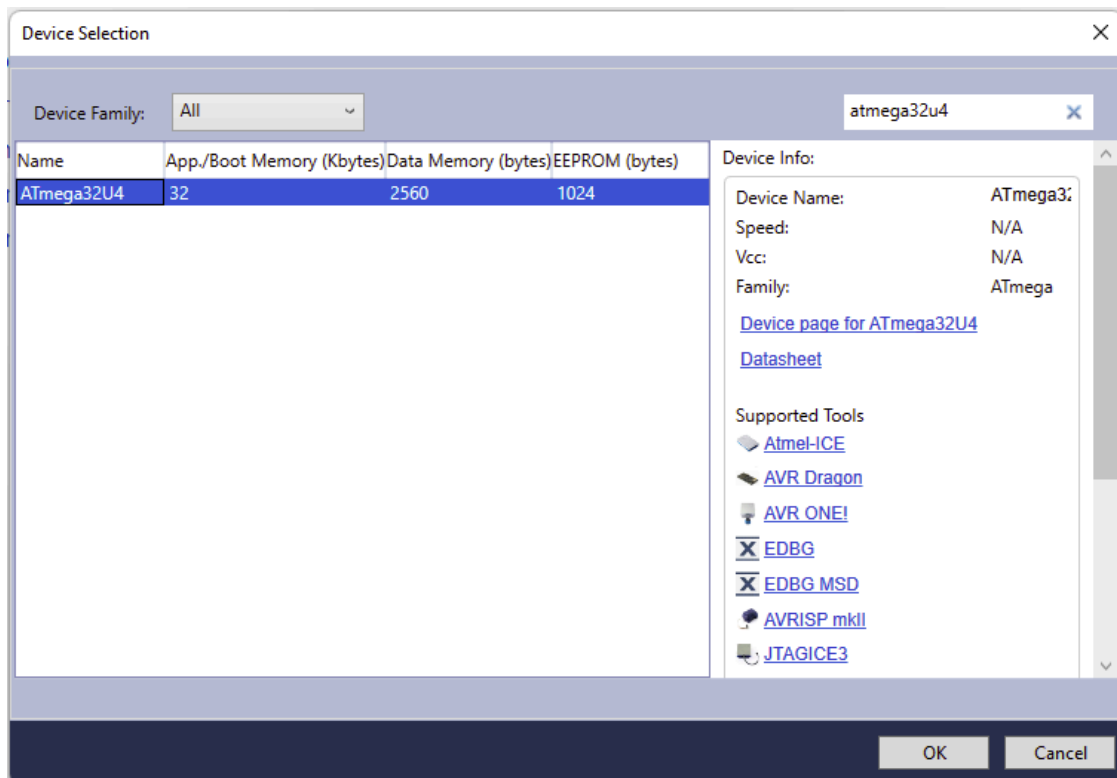


(For the C project in Lab1 Part 2, select **GCC C Executable Project** as shown below

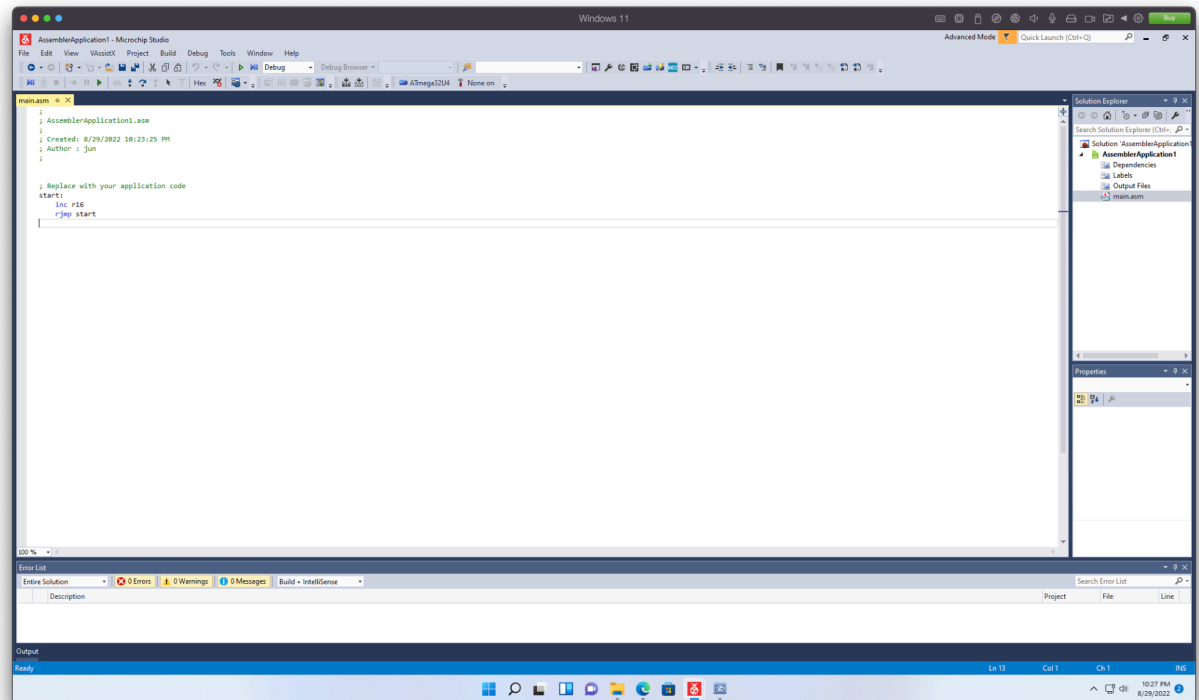


)

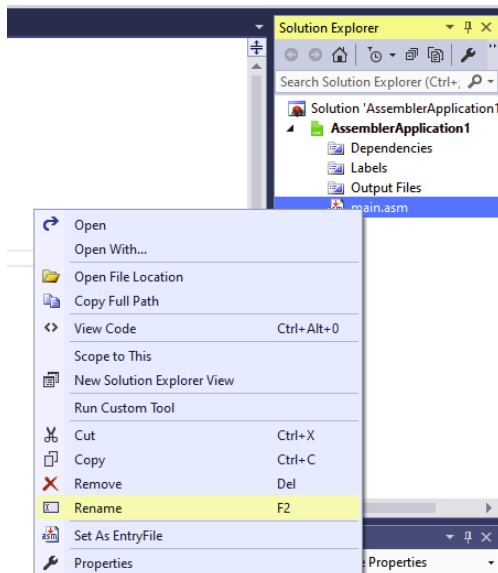
Select **ATmega32U4** as the device type.




At this point, an editor window appears within Microchip Studio and you are able to begin composing your assembly program.

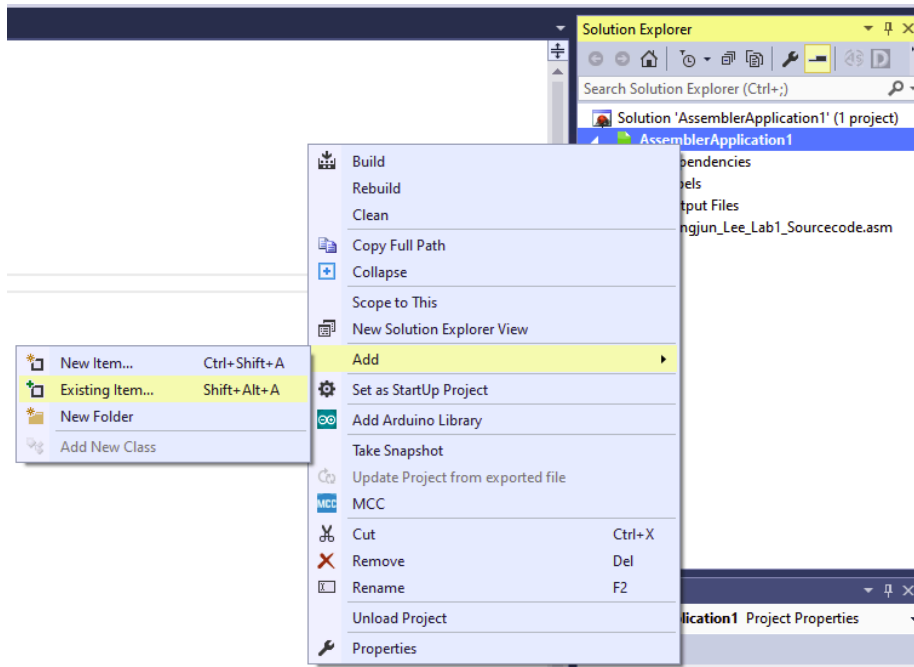


In order to run Lab 1 of ECE 375 as an example, one must obtain the [BasicBumpBot.asm](#) file from the [lab website](#). You can incorporate this code into this new project in one of two ways. First, you can simply open BasicBumpBot.asm file with a text editor and copy-paste of its contents directly into the open editor window within Microchip Studio - this copies your code into the file created for you, e.g., main.asm. If you want to change file name, right-click on the .asm file in the **Solution Explorer** on the right hand side of the Microchip Studio window. Then, select **Rename** and type **YourName_Lab#_Sourcecode**.

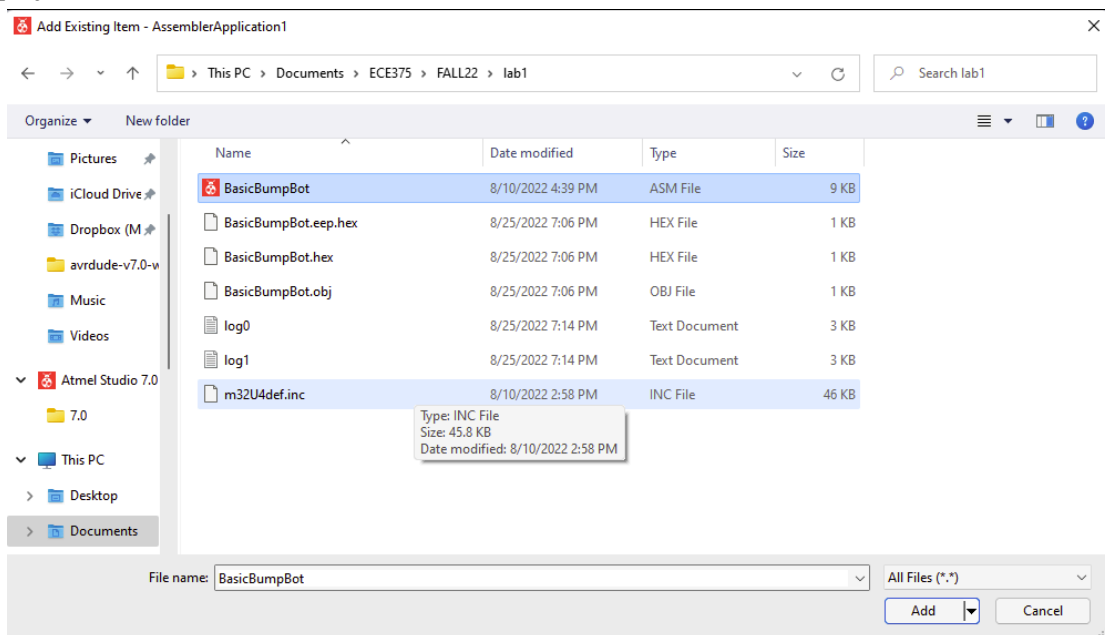


 Dongjun_Lee_Lab1_Sourcecode.asm

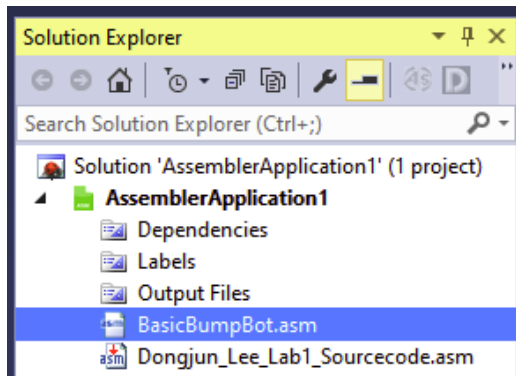
If you want to include an entire existing file into your newly-created project, right-click on the name of your project (e.g., AssemblerApplication1) and select Add => Existing Item...



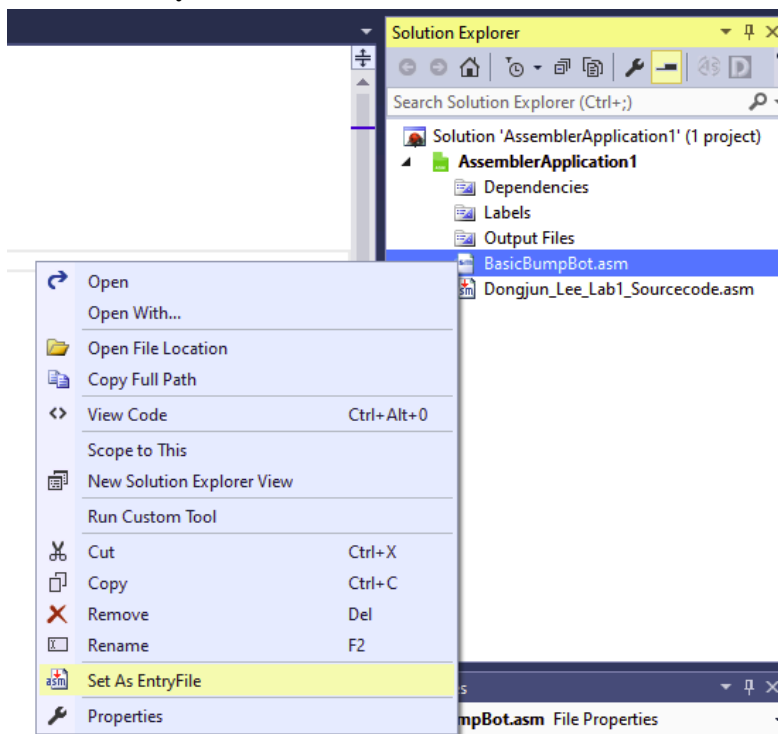
Navigate to the existing assembly code file (e.g., BasicBumpBot.asm) that you would like to use for this project, select it, and click **Add**.



Your existing code file will now appear in the **Solution Explorer** under the heading of your project. Double-click on the file name and it will open in a new editor tab.



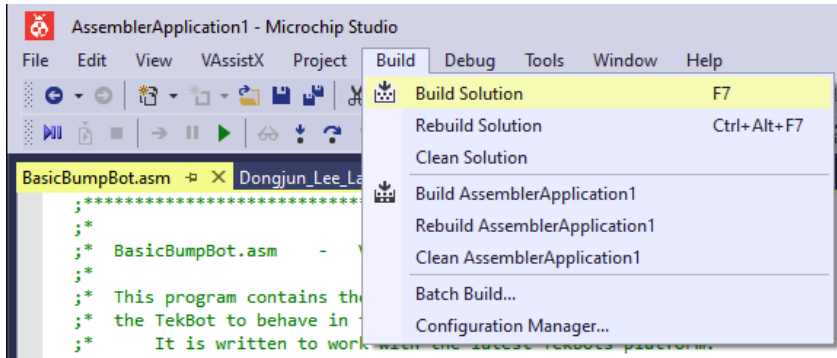
If this existing file is to be the “main” assembly file of your project, right-click on the file name and select **Set As Entry File**.



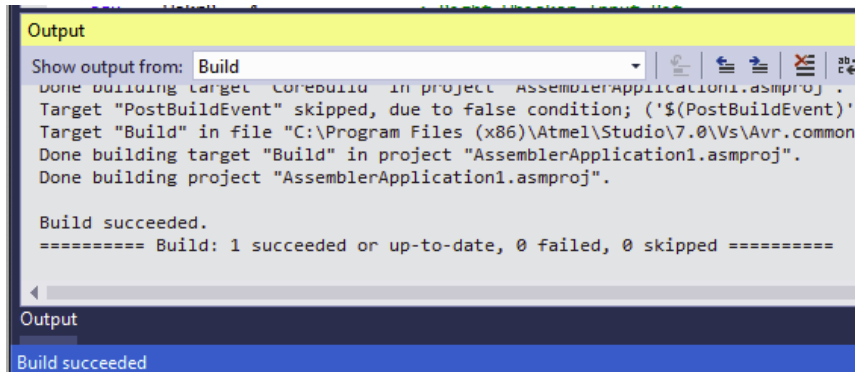
For more information, refer to the 2.1 Startup Tutorial, 2.2 Simulation Tips, 2.3 Debugging Strategies in **AVR starterguide.pdf**

2 Assembling and Running Programs

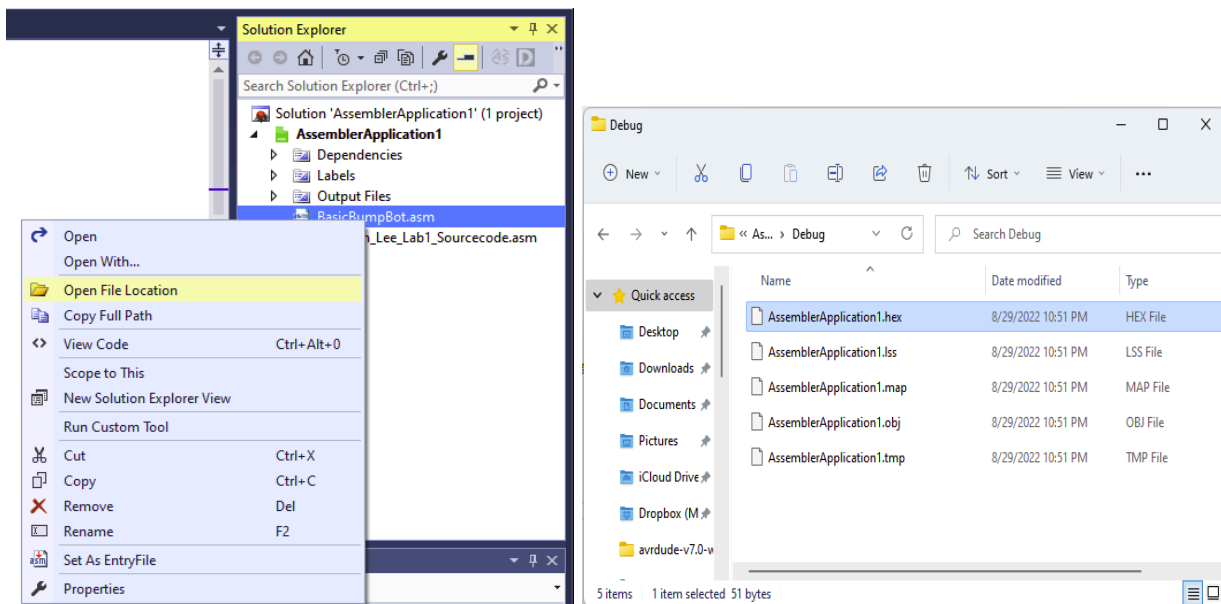
In order to translate the assembly source into machine code, navigate to **Build => Build Solution** in the main Microchip Studio menu.



If the code was successfully compiled, a message in the **Output** window at the bottom should read “Build Succeeded”.

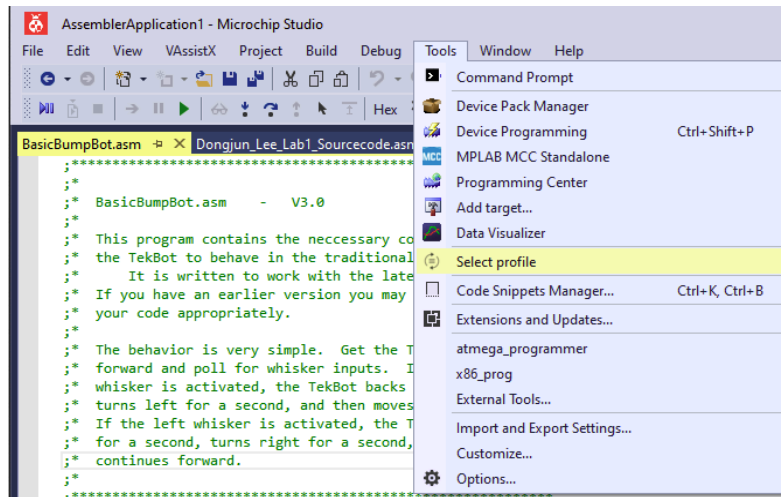


A **hex file** should be created in the **Debug folder** in the **current project directory** that can be moved into the ATmega32U4’s flash program memory. You can check its directory by right-clicking on the file name and select **Open File Location**.

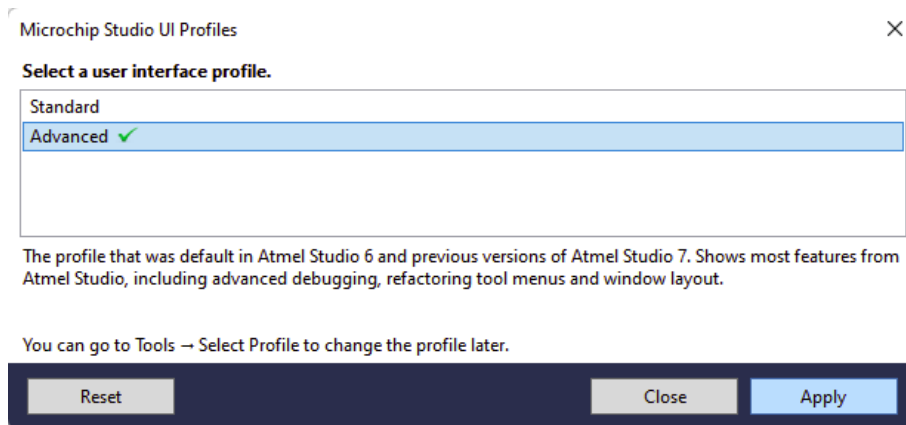


To program, you need to figure out a port connection first. Zadig helps you to identify VID and PID of your USB device.

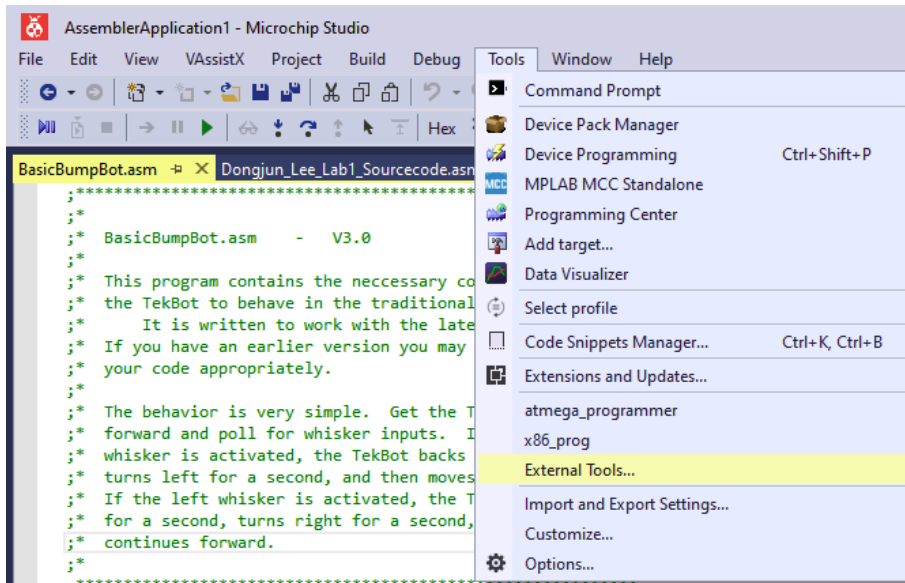
Now we will set up the **external programmer** in the Microchip Studio. Navigate to **Tools => Select profile** in the main Microchip Studio menu.



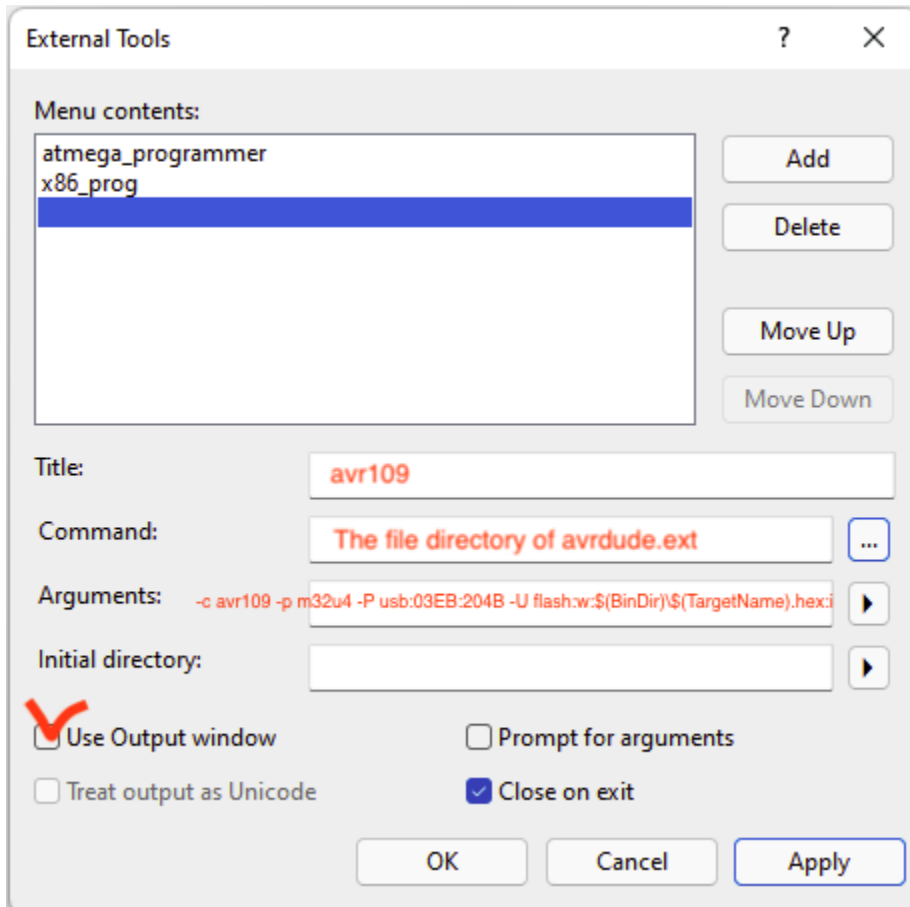
Select **Advanced** and click **Apply**.



Navigate to the Tools => External Tools...



Add a new tool by clicking **Add** and fill in the blank as below:



Title:

avr109 (or the name of your choice)

Commands:

the path of avrdude.exe that you downloaded from the section 1.1.

Arguments:

```
-c avr109 -p m32u4 -P usb:03EB:204B -U flash:w:$(BinDir)\$(TargetName).hex:i
```

Here, the flags are as follows:

- c specifies the type of hardware programmer to use (avr109 is what comes in the lab kit.)

- p specifies the type of chip to be programmed (ATmega32U4 is the AVR MCU on the board.)

- P specifies the connection port.

- U specifies a type of command for avrdude to execute; here, it is writing (w) the flash memory of the ATmega32U4 with your program (the .hex file).

If you want to know other flags, type `avrdude --help` in the command line.

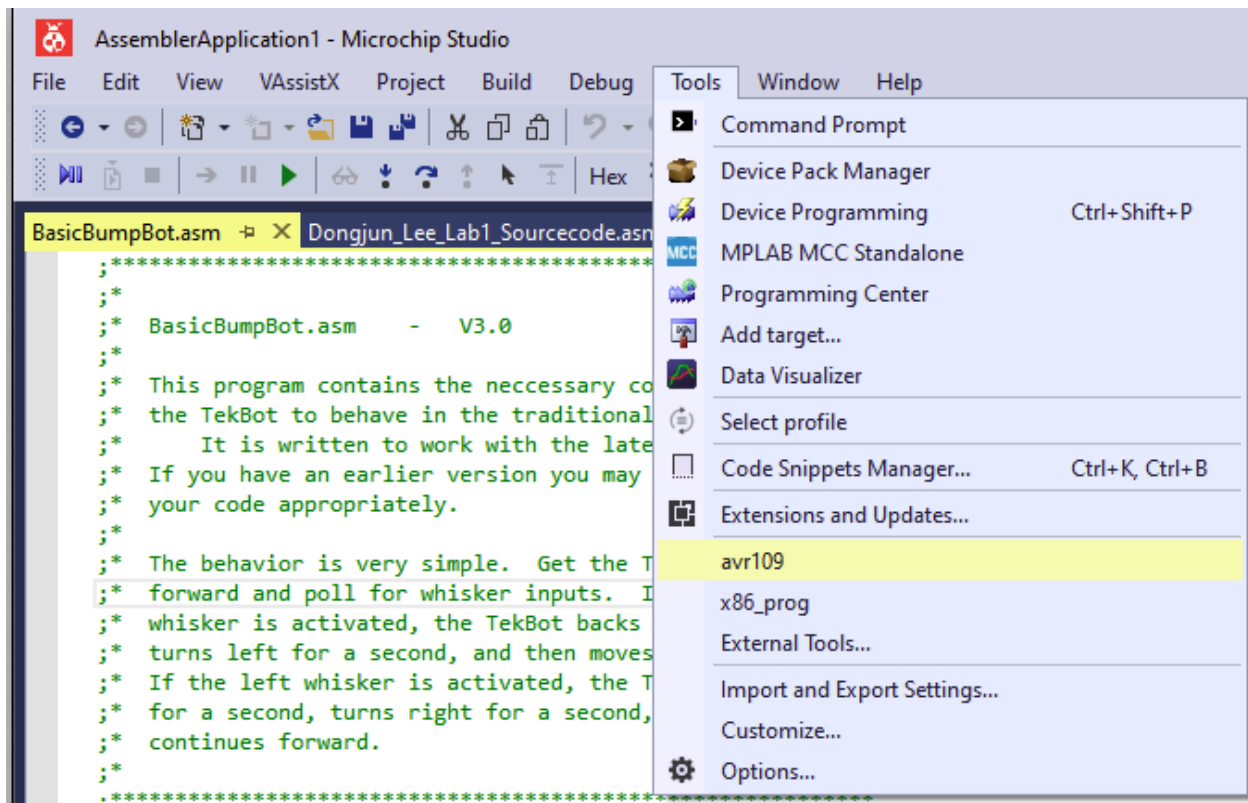
You don't need to replace `$(BinDir)\$(TargetName)` with yours.

Check **Use Output window** then click **Apply** to save the setting.

(If needed, replace `03EB:204B` with the **USB ID** that you jot down from **Zadig.exe** in the **Troubleshooting** section.)

Now, it's ready to program the BasicBumpBot.hex. **Press the reset button on the ATmega32U4 board.**

You have about 9 sec before the bootloader times out and starts any resident user program. **Before it times out**, invoke programming via External Tools that you just created (i.e., avr109). Navigate to the **Tools => avr109**. This should be a one-time set up unless you connect the board with a different USB port.



```
Output
Show output from: avr109

Connecting to programmer: .
Found programmer: Id = "LUFACDC"; type = S
    Software Version = 1.0; No Hardware Version given.
Programmer supports auto addr increment.
Programmer supports buffered memory access with buffersize=128 bytes.

Programmer supports the following devices:
    Device code: 0x44

avrdude.exe: AVR device initialized and ready to accept instructions

Reading | ##### | 100% 0.00s

avrdude.exe: Device signature = 0x1e9587 (probably m32u4)
avrdude.exe: NOTE: "flash" memory has been specified, an erase cycle will be performed
    To disable this feature, specify the -D option.
avrdude.exe: erasing chip
avrdude.exe: reading input file "c:\users\jun\Documents\Atmel Studio\7.0\AssemblerApplication1\AssemblerApplication1\Debug\AssemblerApplication1.hex"
avrdude.exe: writing flash (4 bytes):

Writing | ##### | 100% 0.01s

avrdude.exe: 4 bytes of flash written
avrdude.exe: verifying flash memory against c:\users\jun\Documents\Atmel Studio\7.0\AssemblerApplication1\AssemblerApplication1\Debug\AssemblerApplication1.hex:
Reading | ##### | 100% 0.00s

avrdude.exe: 4 bytes of flash verified

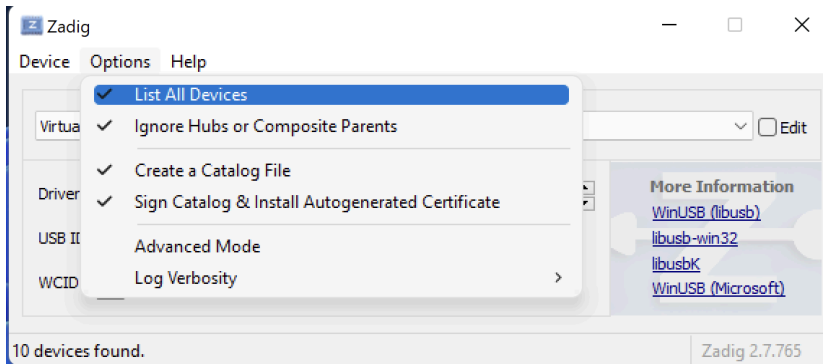
avrdude.exe done. Thank you.
```

If you see none of the prompts in the output window, you may be using avrdude.exe with the wrong architecture version. Make sure you downloaded the right one (in general x64 or x86 not arm64).

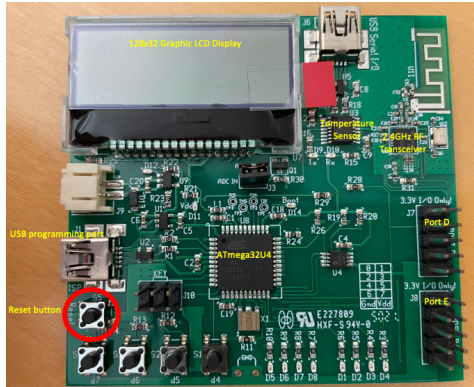
3 Troubleshooting

If the microchip studio cannot detect your device, you want to identify your device ID.

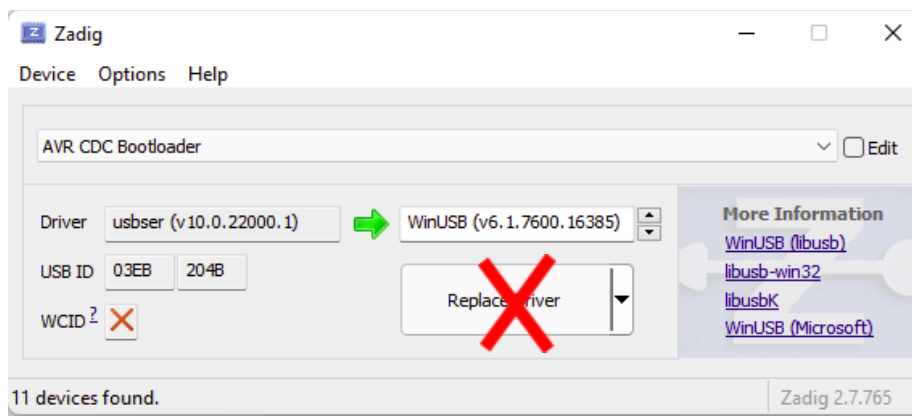
Download Zadig.exe [here: https://zadig.akeo.ie](https://zadig.akeo.ie) and launch the program. Then, enable the List All Devices in the Options.



Connect the ATmega32U4 board to your laptop and **press the reset button on board first**. You have about 9 sec before the bootloader times out and starts any resident user program.



Before it times out, Zadig should detect AVR CDC Bootloader. Jot down the **USB ID** (e.g., 03EB:204B) somewhere.



Do NOT click Replace Driver!!!

4 Tested environment

August 2022: M1 Mac - Windows 11 ARM