Getting started with the Arduino Due

To connect the Arduino Due to your computer, you'll need a Micro-B USB cable. The USB cable will provide power and allow you to program the board.

Attach the USB micro side of the USB cable to the Due's *Programming* port (this is the port closer to the DC power connector). To upload a sketch, choose **Arduino Due (Programming port)** from the **Tools > Board** menu in the Arduino IDE, and select the correct serial port from the **Tools > Serial Port** menu.

The Due has a dedicated forum for discussing the board.

On this page... (hide)

Differences from ATMEGA based boards

In general, you program and use the Due as you would other Arduino boards. There are, however, a few important differences and functional extensions.

The Due has the same footprint as the Mega 2560.

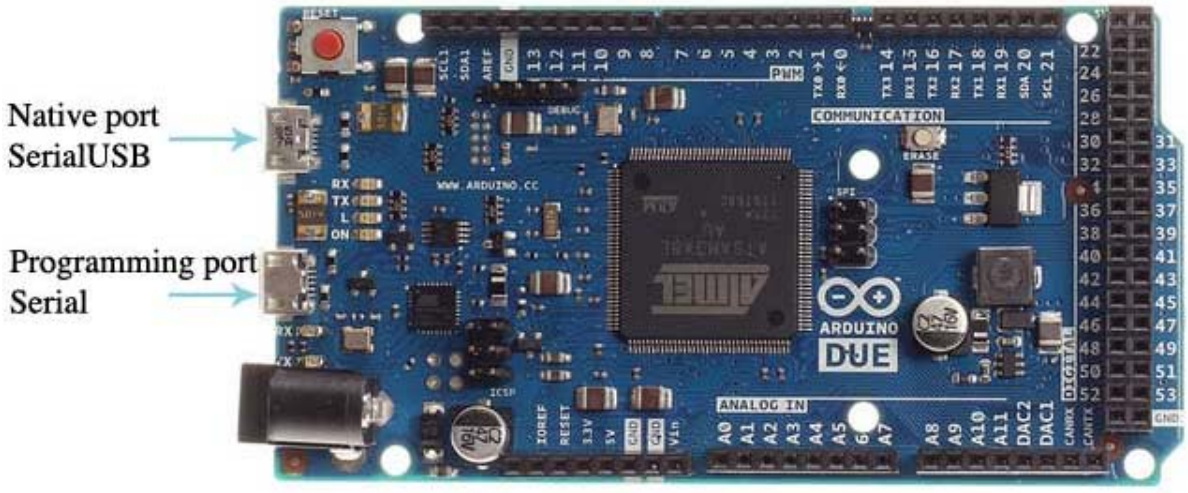
Voltage

The microcontroller mounted on the Arduino Due runs at 3.3V, this means that you can power your sensors and drive your actuartors only with 3.3V. **Connecting higher voltages, like the 5V commonly used with the other Arduino boards will damage the Due.**

The board can take power from the USB connectors or the DC plug. If using the DC connector, supply a voltage between 7V and 12V.

The Arduino Due has an efficient switching voltage regulator, compliant with the USB host specification. If the *Native* USB port is used as host by attaching a USB device to the mirco-A usb connector, the board will provide the power to the device. When the board is used as a usb host, external power from the DC connector is required.

Serial ports on the Due



The Arduino Due has two USB ports available. The *Native* USB port (which supports CDC serial communication using the *SerialUSB* object) is connected directly to the SAM3X MCU. The other USB port is the *Programming* port. It is connected to an ATMEL 16U2 which acts as a USB-to-Serial converter. This *Programming* port is the default for uploading sketches and communicating with the Arduino.

The USB-to-serial converter of the *Programming* port is connected to the first UART of the SAM3X. It's possible to communicate over this port using the "Serial" object in the Arduino programming language.

The USB connector of the *Native* port is directly connected to the USB host pins of the SAM3X. Using the *Native* port enables you to use the Due as a client USB peripheral (acting as a mouse or a keyboard connected to the computer) or as a USB host device so that devices can be connected to the Due (like a mouse, keyboard, or an Android phone). This port can also be used as a virtual serial port using the "SerialUSB" object in the Arduino programming language.

Automatic (Software) Reset

The SAM3X microcontroller differs from AVR microcontrollers because the flash memory needs to be erased before being re-programmed. A manual procedure would involve holding the erase button for a second, pressing the upload button in the IDE, then the reset button.

Because a manual erase-flash procedure is repetitive, this is managed automatically by both USB ports, in two different ways:

Native port

Opening and closing the "Native' port at the baud rate of 1200bps triggers a “soft erase” procedure: the flash memory is erased and the board is restarted with the bootloader. If, for some reason, the MCU were to crash during this process, it is likely that the soft erase procedure wouldn't work as it's done in software by the MCU itself.

Opening and closing the *Native* port at a baudrate other than 1200bps will not reset the SAM3X. To use the serial monitor, and see what your sketch does from the beginning, you'll need to add few lines of code inside the setup(). This will ensure the SAM3X will wait for the SerialUSB port to open before executing the sketch:

```
while (!Serial) ;
```
[Get Code]

Pressing the Reset button on the Due causes the SAM3X reset as well as the USB communication. This interruption means that if the serial monitor is open, it's necessary to close and reopen it to restart the communication.

Programming port

The *Programming* port uses a USB-to-serial chip connected to the first UART of the MCU (RX0 and TX0). The USB-to-serial chip has two pins connected to the Reset and Erase pins of the SAM3X. When you open this serial port, the USB-to-Serial activates the Erase and Reset sequence before it begins communicating with the UART of the SAM3X. This procedure is much more reliable and should work even if the main MCU has crashed.

To communicate serially with the *Programming* port, use the "Serial" object in the IDE. All existing sketches that use serial communication based on the Uno board should work similarly. The *Programming* port behaves like the Uno's serial port in that the USB-to-Serial chip resets the board each time you open the serial monitor (or any other serial communication).

Pressing the Reset button while communicating over the *Programming* port doesn't close a USB connection with the computer because only the SAM3X is reset.

USB Host

The Due has the ability to act as a USB host for peripherals connected to the SerialUSB port. For additional information and examples, see the USB host reference page.

When using the Due as a host, it will be providing power to the attached device. It is strongly recommended to use the DC power connector when acting as a host.

ADC and PWM resolutions

The Due has the ability to change its default analog read and write resolutions (10-bits and 8-bits, respectively). It can support up to 12-bit ADC and PWM resolutions. See the analog write resolution and analog read resolution pages for information.

Expanded SPI functionality

The Due has expanded functionality on its SPI bus, useful for communicating with multiple devices that speak at different speeds. See the Due extended SPI library usage page for more details.

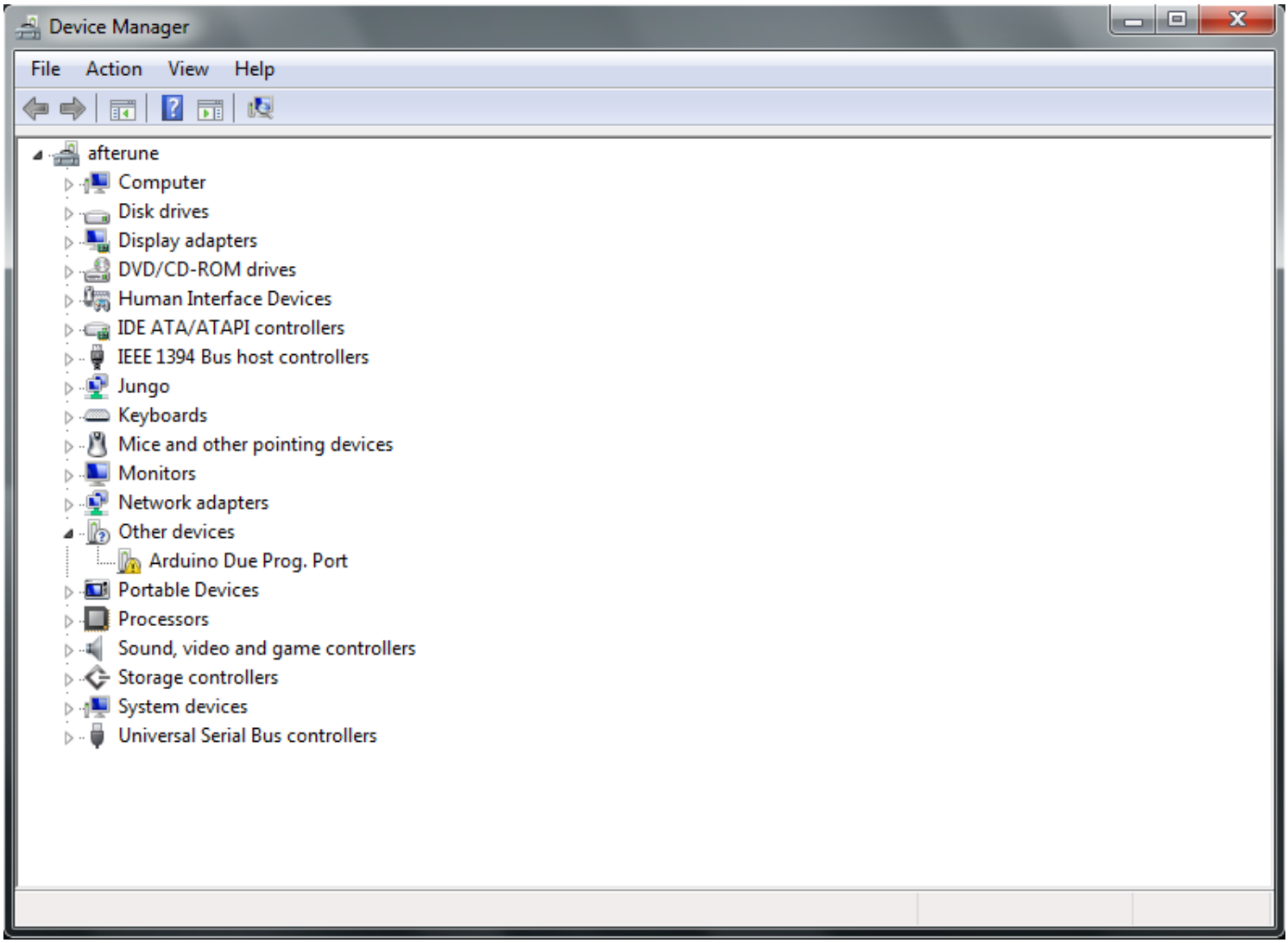
Installing Drivers for the Due

OSX

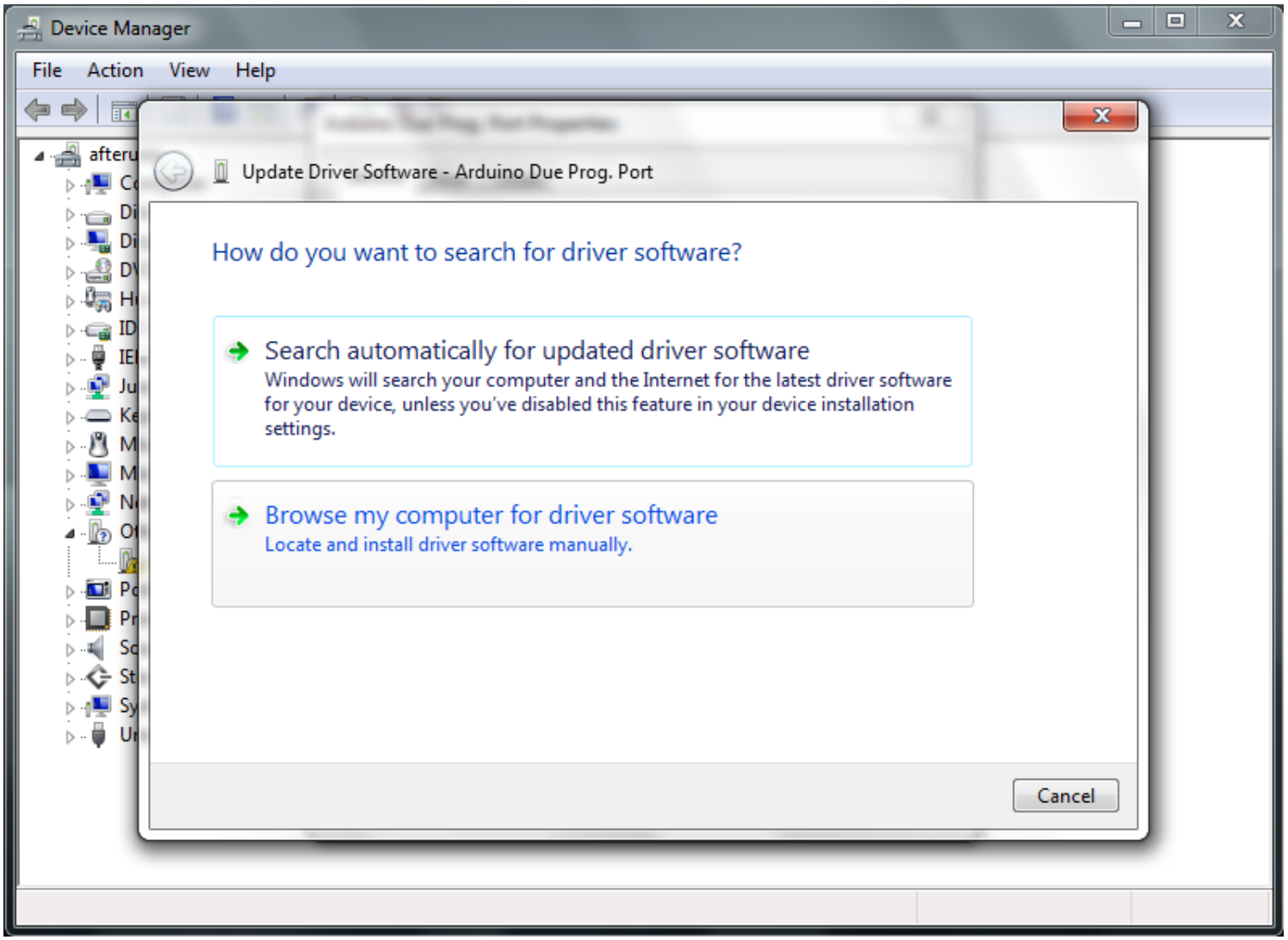
- No driver installation is necessary on OSX. Depending on the version of the OS you're running, you may get a dialog box asking you if you wish to open the “Network Preferences”. Click the "Network Preferences..." button, then click "Apply". The Uno will show up as “Not Configured”, but it is still working. You can quit the System Preferences.

Windows (tested on XP and 7)

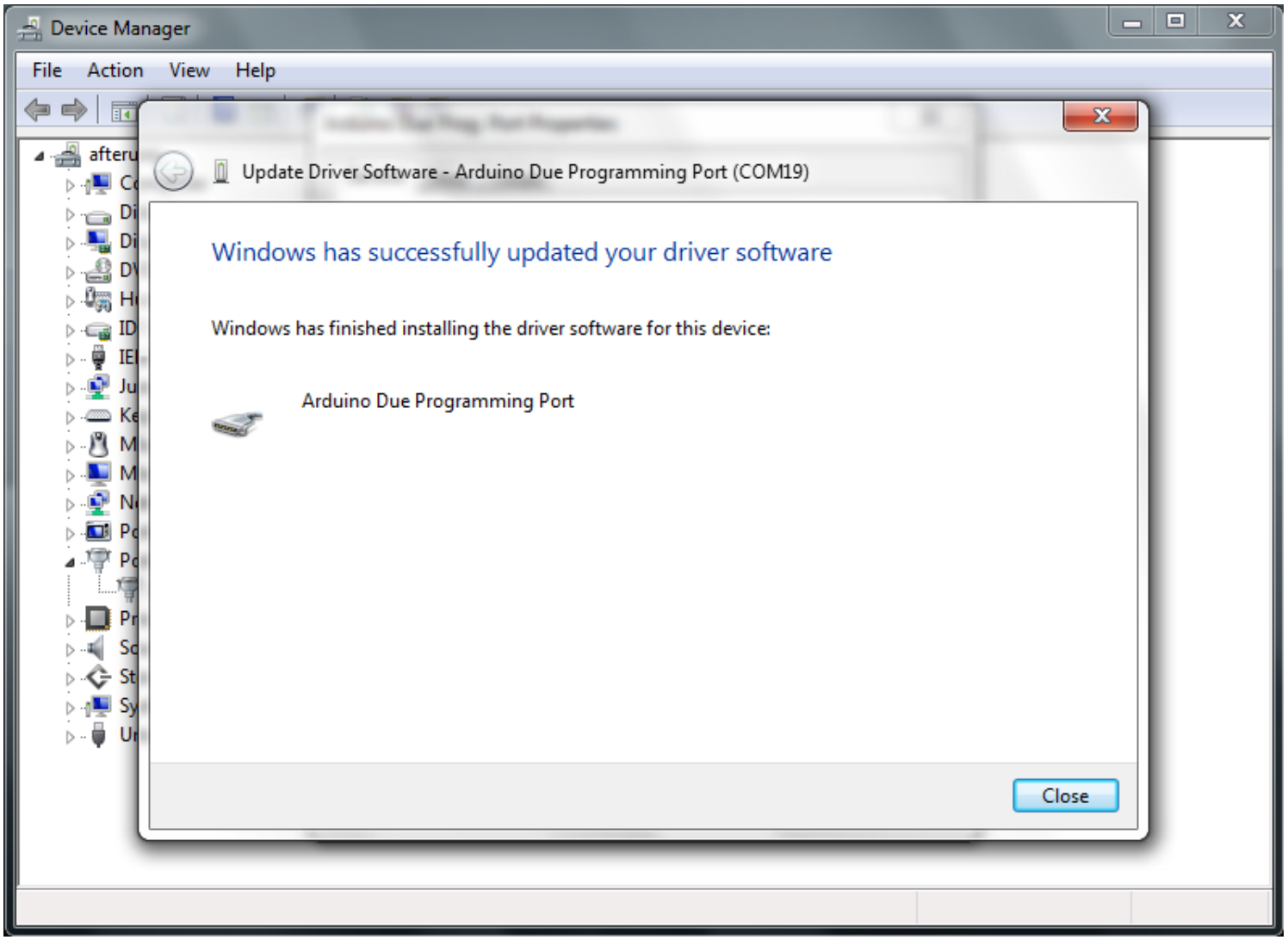
- Download the Windows version of the Arduino software. When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure.
- Connect the Due to your computer with a USB cable via the *Programming* port.
- Windows should initiate its driver installation process once the board is plugged in, but it won't be able to find the driver on its own. You'll have to tell it where the driver is.
- Click on the Start Menu and open the Control Panel
- Navigate to “System and Security”. Click on System, and open the Device Manager.
- Look for the listing named “Ports (COM & LPT)”. You should see an open port named “Arduino Due Prog. Port”.
- Select the “Browse my computer for Driver software” option.



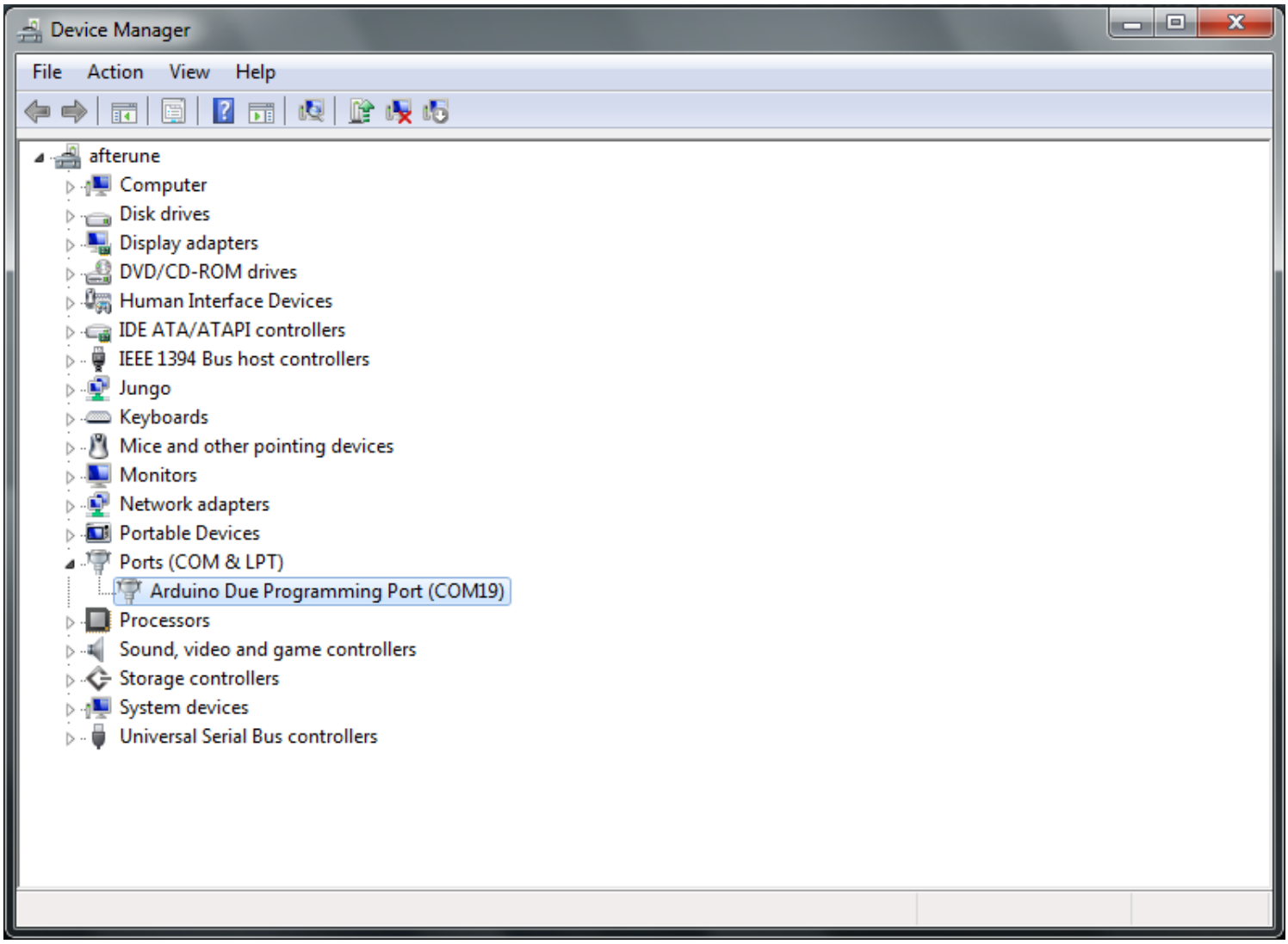
- Right click on the “Arduino Due Prog. Port” and choose “Update Driver Software”. Descriptive text



- Navigate to the folder with the Arduino IDE you downloaded and unzipped earlier. Locate and select the “Drivers” folder in the main Arduino folder (not the “FTDI USB Drivers” sub-directory). Press “OK” and “Next” to proceed.
- If you are prompted with a warning dialog about not passing Windows Logo testing, click “Continue Anyway”.
- Windows now will take over the driver installation.



- You have installed the driver on your computer. In the Device Manager, you should now see a port listing similar to “Arduino Due Programming Port (COM4)”.



Linux

- No driver installation is necessary for Linux.

Uploading Code to the Due

The uploading process on the Arduino Due works the same as other boards from a user's standpoint. It is recommended to use the *Programming* port for uploading sketches, though you can upload sketches on either of the USB ports.

For uploading with the *Programming* port follow this steps:

- Connect your board to the computer by attaching the USB cable to the Due's *Programming* port (this is the port closer to the DC power connector).
- Open the Arduino IDE.
- In the "Tools" menu choose "Serial Port" and select the serial port of the Due
- Under the "Tools > Boards" menu select "Arduino Due (Programming port)"

You are now ready to upload sketches to your Arduino Due.

For more details on the Arduino Due, see the hardware page.