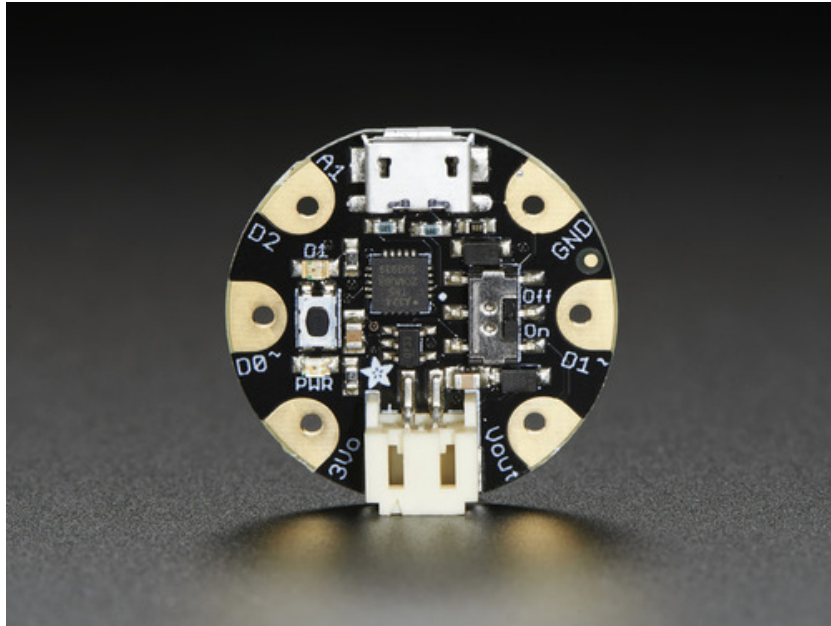




Introducing Gemma

Created by lady ada



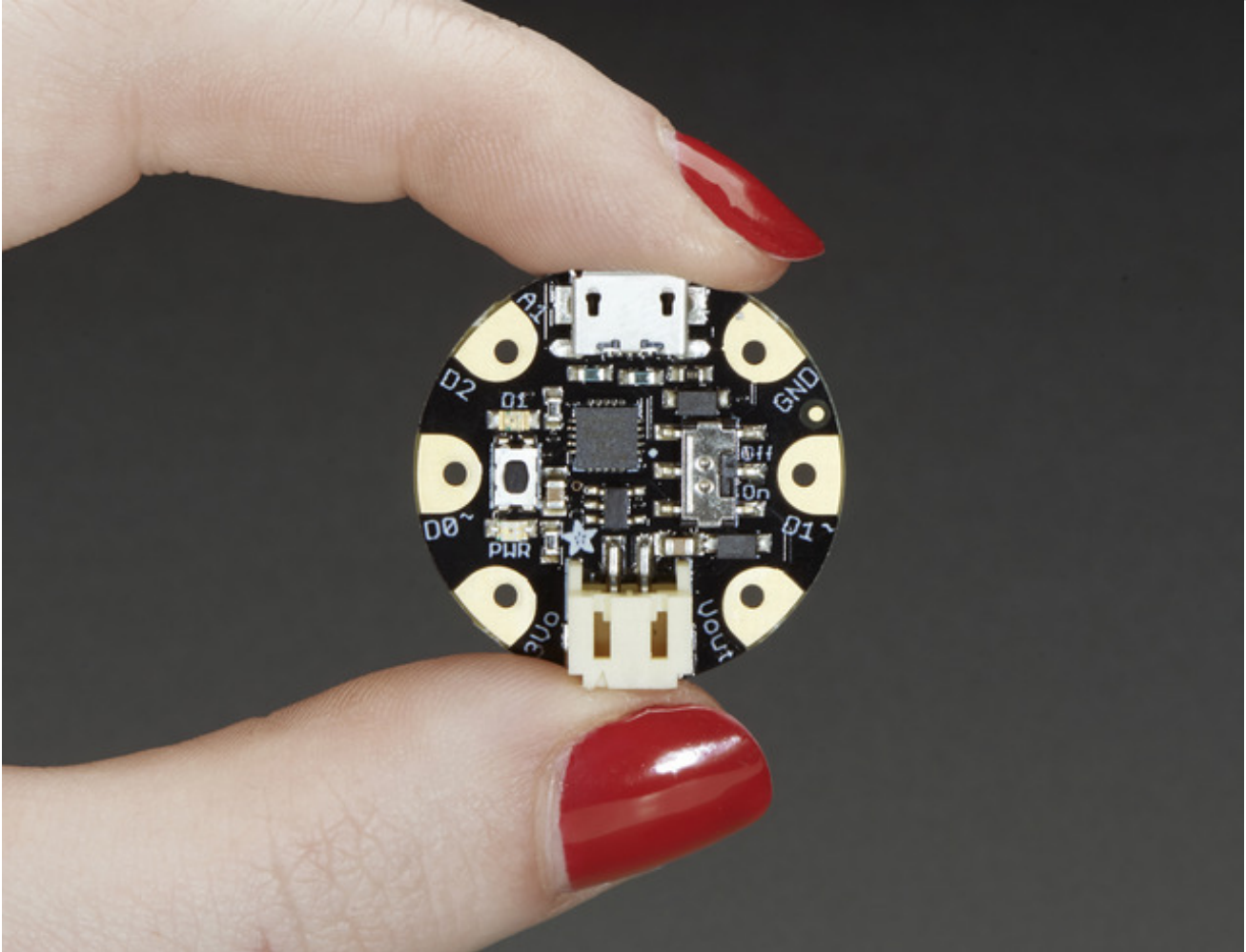
Last updated on 2015-08-29 02:52:34 PM EDT

Guide Contents

Guide Contents	2
Introduction	4
Guided Tour	7
Pinouts	10
JST Battery Input	11
Power Pads	11
Input/Output Pads	12
Secret Reset Pad	12
Windows Driver Installation	13
Adafruit Trinket/Gemma Driver Installation	13
Arduino Gemma Driver Installation	14
Flora Driver Installation	14
Windows 8 & 10	14
Windows 7	15
Locating the Driver File (Win 7 & XP)	17
Windows 10 Driver Installation	20
About the Bootloader	29
Special Notes on using Gemma with Linux	30
How to start the bootloader	30
Setting up with Arduino IDE	31
Arduino IDE Setup	31
Blink!	31
Adafruit Gemma (Black Gemma)	32
Arduino Gemma (Teal Gemma)	33
Get Into Bootloader Mode	34
Something Went Wrong!	35
If you get the error message avrdude: Error: Could not find USBtiny device (0x1781/0xc9f)	35
If you get a lot of red text, errors and also a warning about Verification Failed	36
On Linux if you get the error message "usbtiny_receive: error sending control message: Protocol error (expected 4, got -71)"	37
Programming with Arduino IDE	38

pinMode() & digitalWrite() & digitalRead()	38
analogRead()	39
analogWrite()	40
More...	40
Downloads	42
Datasheets	42
Source code	42
Schematics	42
FAQ	44

Introduction



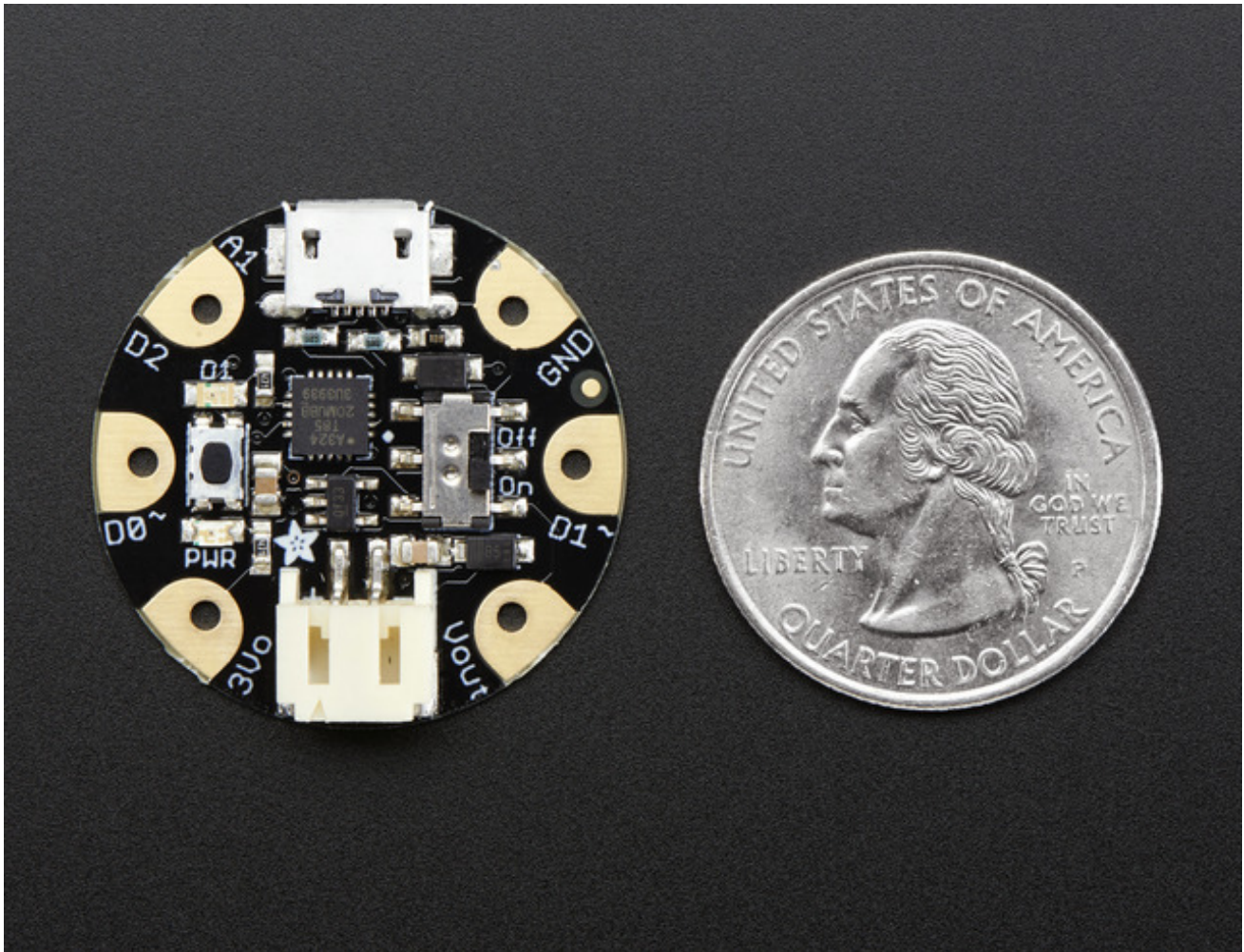
Love Flora but want a bite-sized version? Look no further, Gemma is a tiny wearable platform board with a lot of might in a 1" diameter package. Powered by a Attiny85 and programmable with an Arduino IDE over USB, you'll be able to realize any wearables project!

We wanted to design a microcontroller board that was small enough to fit into any project, and low cost enough to use without hesitation. Perfect for when you don't want to give up your Flora and you aren't willing to take apart the project you worked so hard to design. It's our lowest-cost sewable controller!

The Attiny85 is a fun processor because despite being so small, it has 8K of flash, and 5 I/O pins, including analog inputs and PWM 'analog' outputs. We designed a USB bootloader so you can plug it into any computer and reprogram it over a USB port just like an Arduino (it uses 2 of the 5 I/O pins, leaving you with 3). In fact we even made some simple modifications to the Arduino IDE so that it works like a mini-Flora. Perfect for small & simple projects the Gemma will be your go-to wearable electronics platform.

Even though you can program Gemma using the Arduino IDE, it's not a fully 100% Arduino-compatible. There are some things you trade off for such a small and low cost microcontroller!

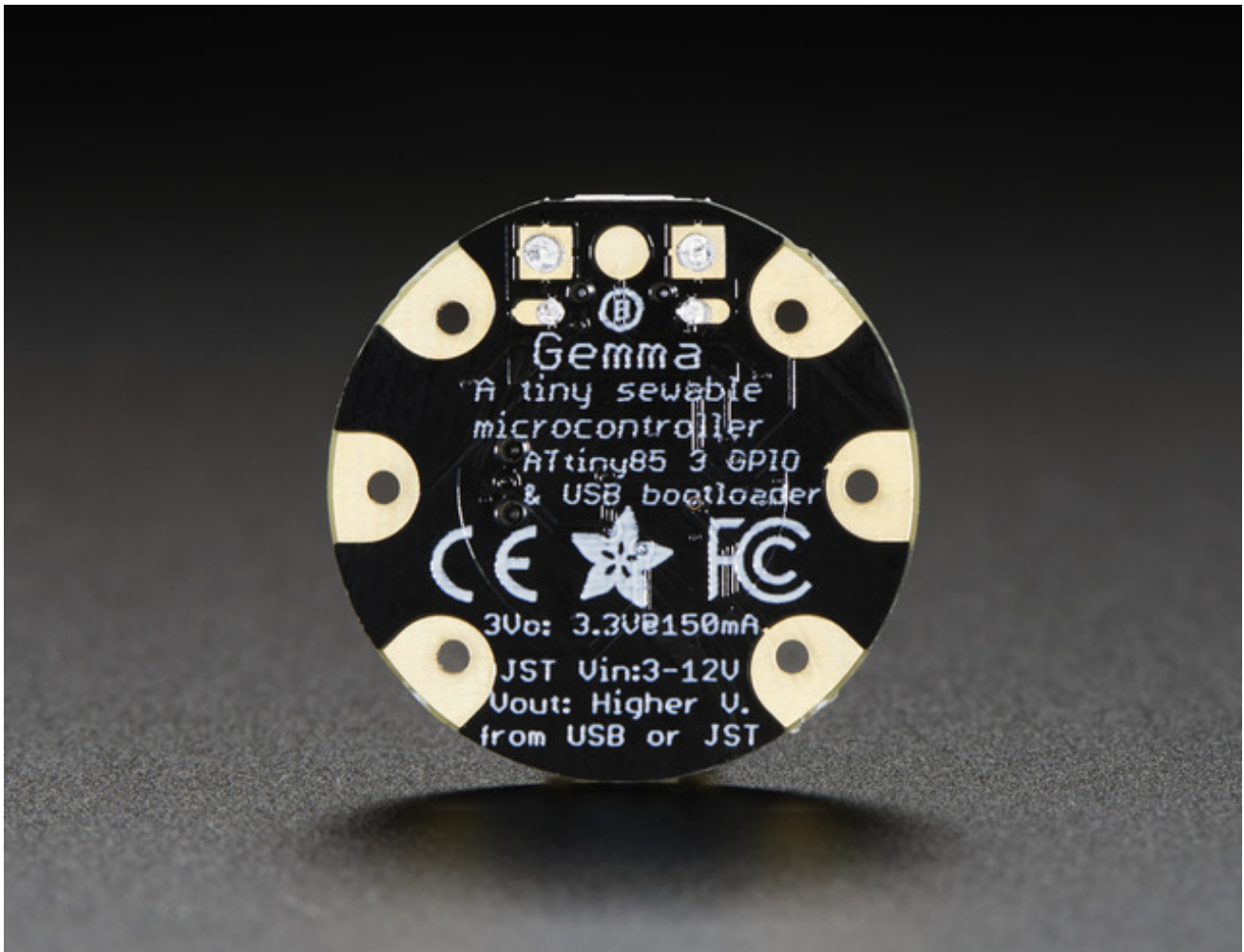
- Gemma does not have a Serial port connection for debugging so the serial port monitor will not be able to send/receive data
- Some computers' USB v3 ports don't recognize the Gemma's bootloader. Simply use a USB v2 port or a USB hub in between
- Gemma is not supported on Linux operating system at this time - try Mac OS or Windows!



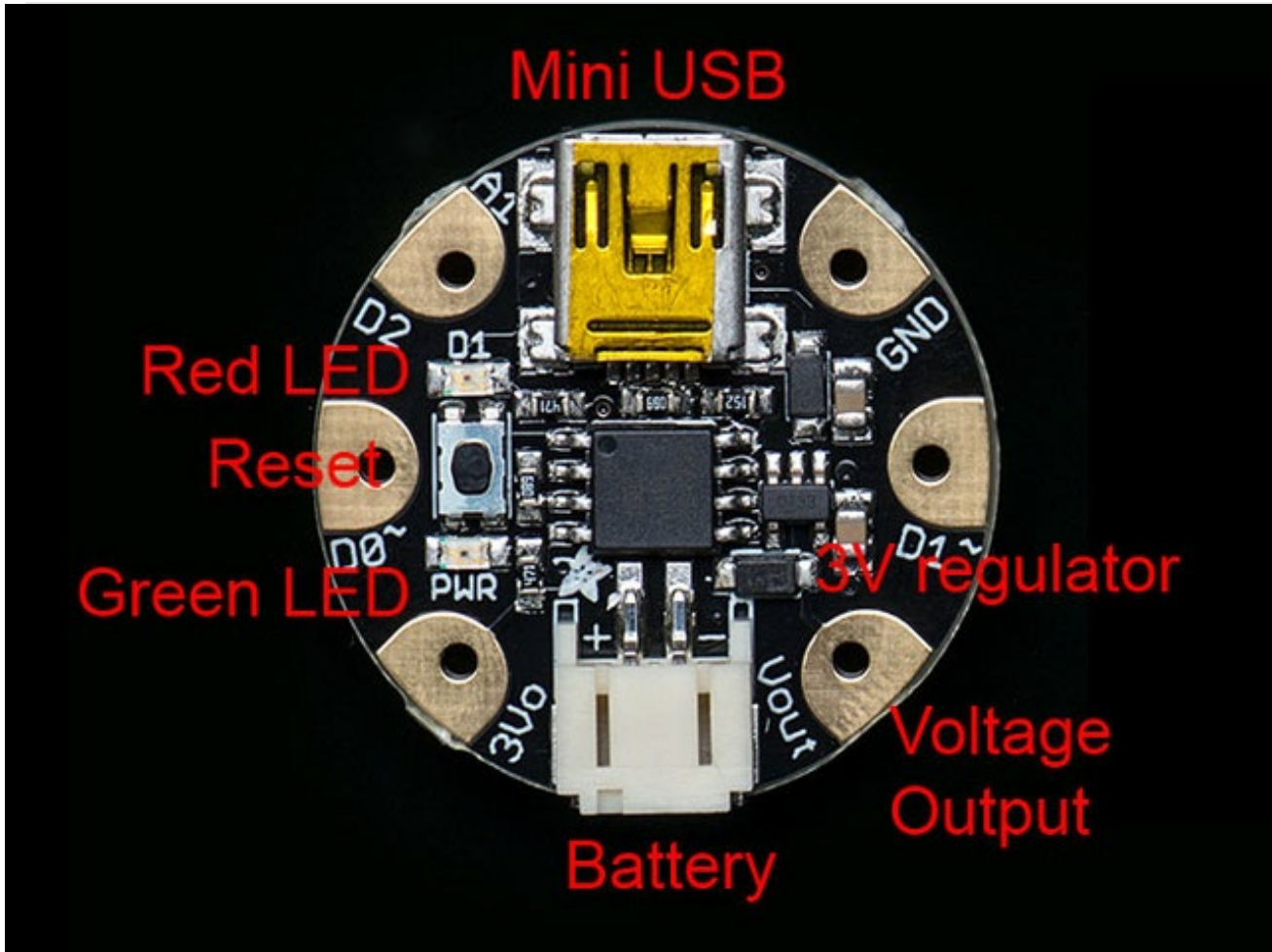
Here are some useful specifications!

- Super small, only 1.1" / 28mm diameter and 0.28" / 7mm thick.
- Easy-to-sew or solder pads for embedding in your wearable project
- Low cost enough, you can use one for every weekend project
- ATtiny85 on-board, 8K of flash, 512 byte of SRAM, 512 bytes of EEPROM
- Internal oscillator runs at 8MHz
- Ultra low power, draws only 9 mA while running
- USB bootloader with a nice LED indicator looks just like a USBtinyISP so you can program it

- with the Arduino IDE (with a few simple config modifications)
- Mini-USB jack for power and/or USB uploading, you can put it in a box or tape it up and use any USB cable for when you want to reprogram.
 - We really worked hard on the bootloader process to make it rugged and foolproof
 - ~5.25K bytes available for use (2.75K taken for the bootloader)
 - On-board 3.3V power regulator with 150mA output capability and ultra-low dropout. Up to 16V input, reverse-polarity protection, thermal and current-limit protection.
 - Power with either USB or external output (such as a battery) - it'll automatically switch over
 - On-board green power LED and red pin #1 LED
 - Reset button for entering the bootloader or restarting the program.
 - 3 GPIO - The 3 independent IO pins have 1 analog input and 2 PWM output as well.
 - Hardware I2C capability for breakout & sensor interfacing.



Guided Tour

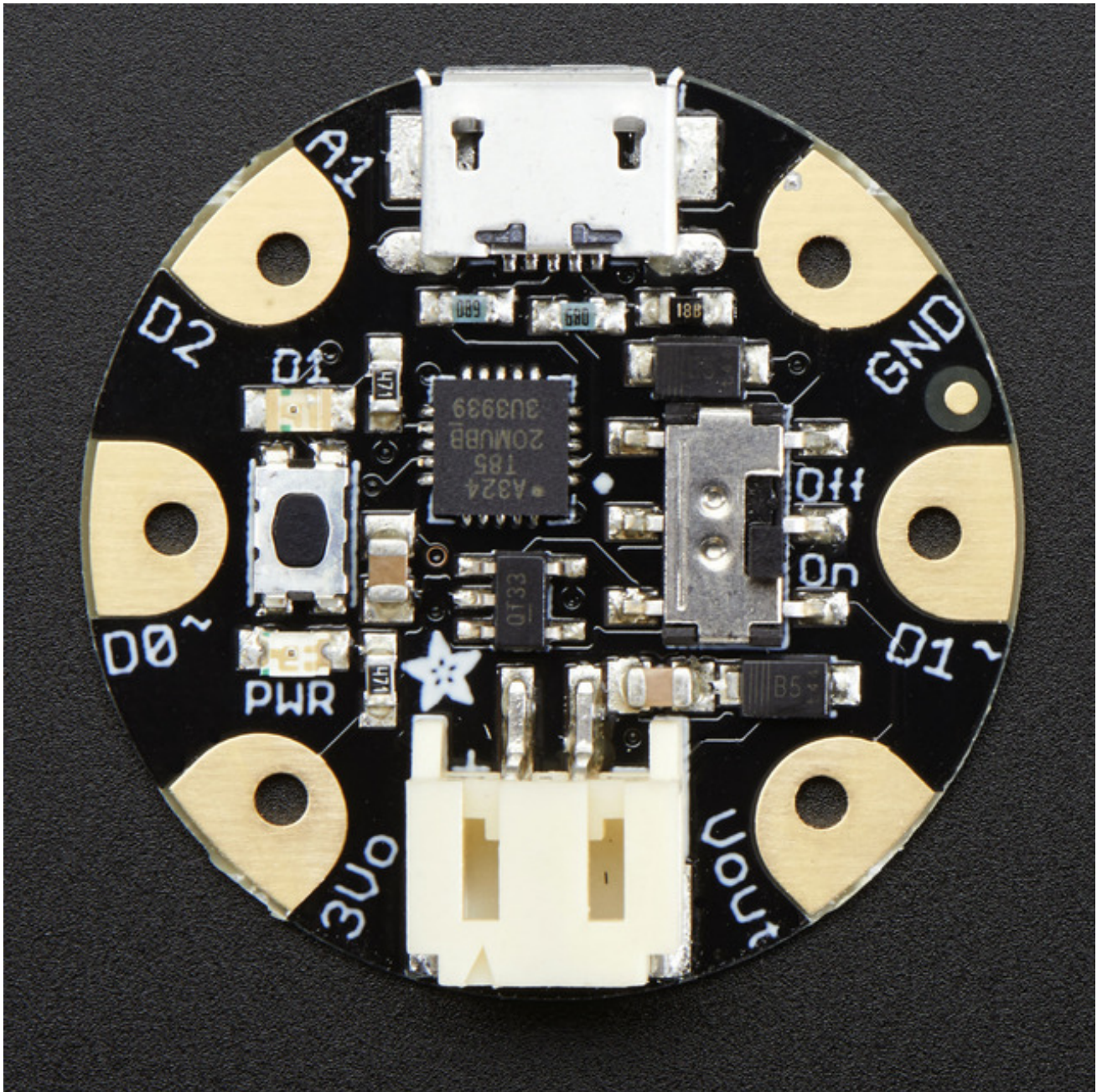


Let me take you on a tour of your Gemma! Each Gemma is assembled here at Adafruit and comes chock-full of good design to make it a joy to use.

- **Mini-B USB connector (on Version 1)** - We went with the tried and true mini-B USB connector for power and/or USB bootloading on our first version of the Gemma
- **Green Power LED** - you'll know that the board is powered up when this bright LED is lit
- **Red #1 LED** - this LED does double duty. Its connected with a series resistor to the digital #1 GPIO pin. It pulses nicely when the Gemma is in bootloader mode, and its also handy for when you want an indicator LED.
- **JST Battery Input** - take your Gemma anywhere and power it from an external battery. This pin can take up 16V DC input, and has reverse-polarity, over-current and thermal protections. The circuitry inside will use either the battery or USB power, safely switching from one to the other. If both are connected, it will use whichever has the higher voltage. Works great with a Lithium Polymer battery or our 3xAAA battery packs with a JST connector on the end
- **Voltage Output** - This pin will give you either the battery power or USB power, whichever has a higher voltage. Its great when you want to power something like NeoPixels, that might use

more than the 150mA available from the onboard regulator

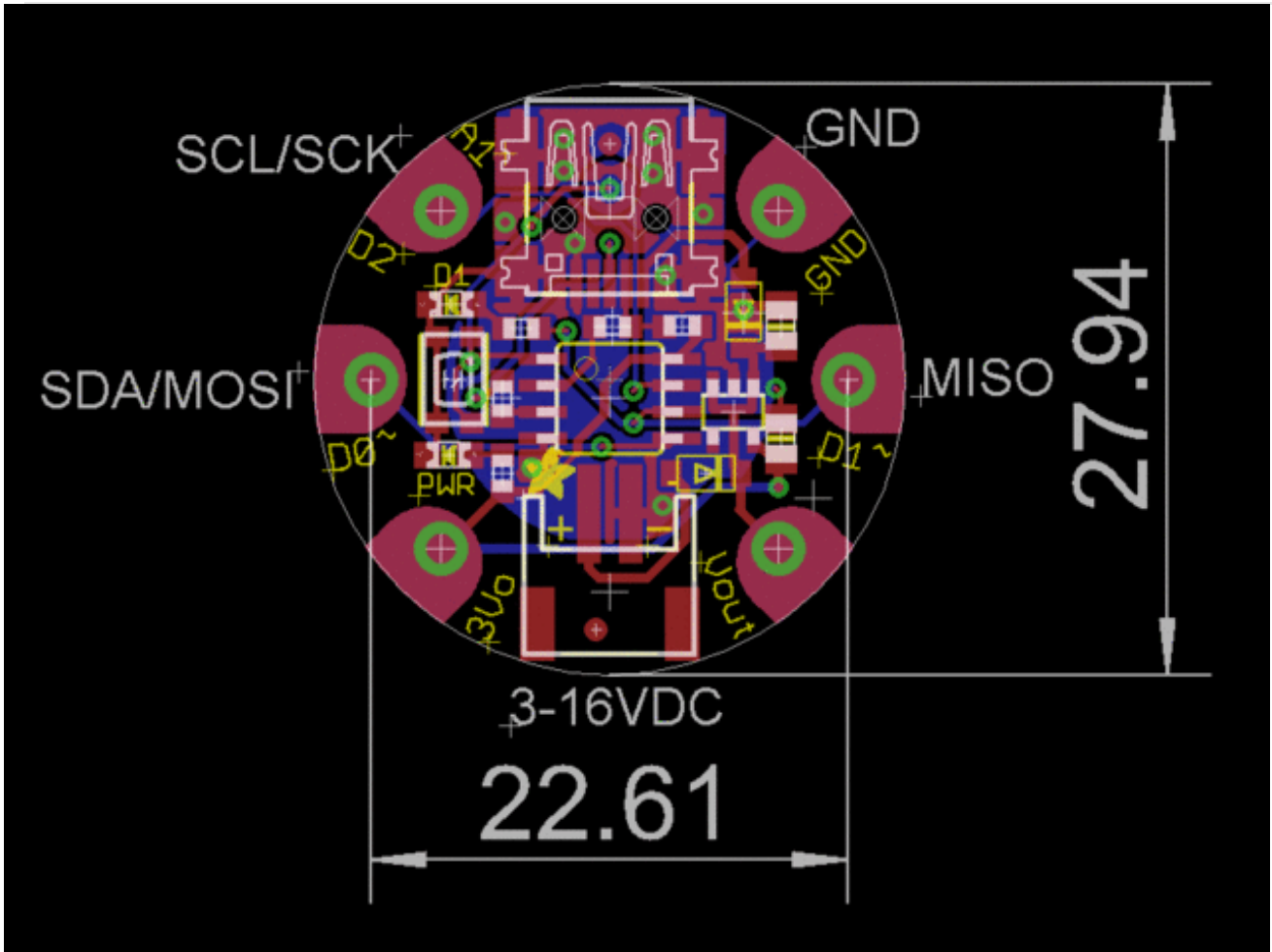
- **3V Regulator Out** - The on-board voltage regulator can supply up to 150mA at a steady 3.3V from up to 16VDC
- **Sewing friendly pads** - You can easily sew to these pads, and they're gold plated so they won't corrode (oxidize). You can also use alligator clips or solder directly to them.
- **GPIO!** - 3 GPIO pins, at 3V logic, check the next section for a detailed pinout guide
- **Reset Button** - an onboard reset button will launch the bootloader when pressed and the Gemma is plugged into a computer. If it is not connected to a computer, it's smart enough to go straight to the program.



New in Version 2

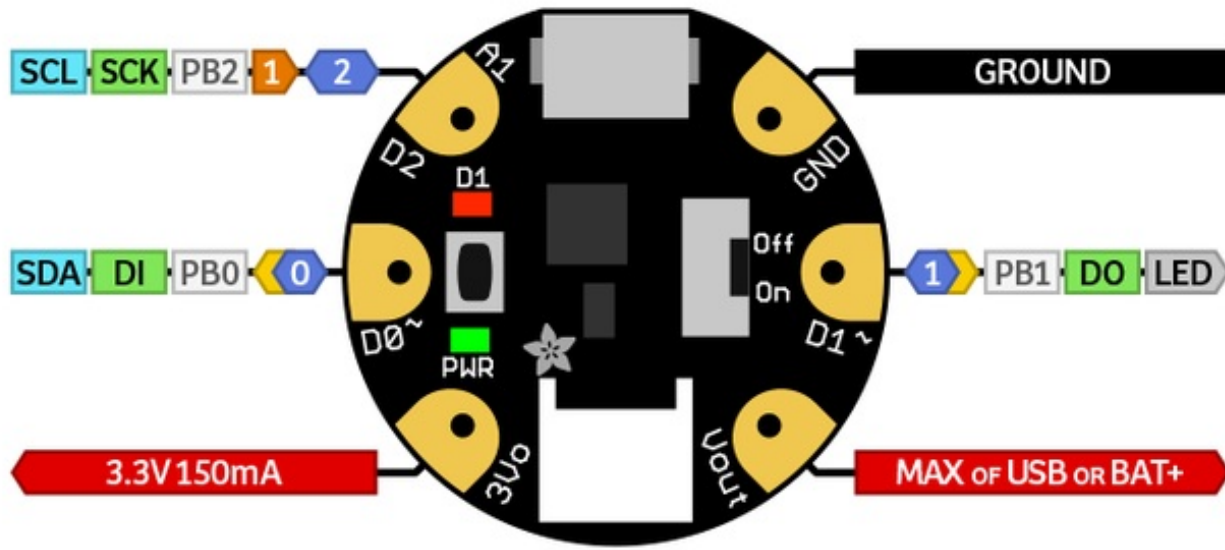
- **Micro-B USB connector (on Version 2)** - In version 2 we upgraded to a micro USB connector, which is now the standard for all cell phones and other devices. It is used for power and/or USB bootloading on our first version of the Gemma
- **On/Off switch (in version 2 only)** - With the space we saved by moving to a smaller chip and micro USB connector we had enough space to add an on/off slide switch!

Pinouts



This diagram shows the physical size of the Gemma (diameter in mm) and the distance from pad to pad. The pads are at exact 90 & 45 degree angles except for 0 and 180 which is where the USB/JST connectors go

GEMMA



POWER	
GROUND	
DIGITAL READ/WRITE	
ANALOG:	READ WRITE
SERIAL	I2C SPI
PORT PIN	
MISC	

JST Battery Input

There is no battery INPUT pin on the Gemma. You can connect a battery via the JST jack. We have found that [Lipoly batteries](http://adafru.it/cFB) (<http://adafru.it/cFB>), [coin-cells](http://adafru.it/783) (<http://adafru.it/783>), and [AAA's](http://adafru.it/727) (<http://adafru.it/727>) work great. You can also make your own battery input pack using a plain JST cable (<http://adafru.it/261>). And use a JST extension cable if necessary (<http://adafru.it/1131>).

You can plug anything from around 4VDC up to 16VDC, but we suggest 4-6V since higher voltages just get wasted as heat. This input is polarity protected. If the green PWR LED lights up, you're good to go. There is no off switch on the Gemma, so unplug or switch off the battery pack when done.

Power Pads

Half of the pads on the Gemma are related to power in and out: **3Vo**, **Vout** and **GND**

- **Vout** - This is a voltage **OUTPUT** pin, it will be connected to *either* the USB power or the

battery input, whichever has the higher voltage. This output does not connect to the regulator so you can draw as much current as your USB port / Battery can provide (in general, that's about 500mA)

- **3Vo** - This is the **3.3V OUTPUT** pad from the voltage regulator. It can provide up to 150mA at a steady 3.3V. Good for sensors or small LEDs or other 3V devices.
- **GND** is the common ground pin, used for logic and power. It is connected to the USB ground and the power regulator, etc. This is the pin you'll want to use for any and all ground connections

Input/Output Pads

Next we will cover the 3 GPIO (General Purpose Input Output) pins! For reference you may want to also check out the datasheet-reference above for the core ATtiny85 pin

All the GPIO pins can be used as digital inputs, digital outputs, for LEDs, buttons and switches etc. They can provide up to 20mA of current. Don't connect a motor or other high-power component directly to the pins! [Instead, use a transistor to power the DC motor on/off \(http://adafru.it/aUD\)](http://adafru.it/aUD)

On a Gemma, the GPIO are 3.3V output level, and should not be used with 5V inputs. In general, most 5V devices are OK with 3.3V output though.

The 3 GPIO pins are completely 'free' pins, they are not used by the USB connection so you never have to worry about the USB interface interfering with them when programming

- **Pad #0** - this is connected to **PB0** on the ATtiny85. This pin can be used as a PWM output, and is also used for I2C data, and SPI data input.
- **Pad #1** - this is connected to **PB1** on the ATtiny85. This pin can be used as a PWM output, and is also used for SPI data output. This pin is also connected to the onboard LED (like pin 13 on a regular Arduino).
- **Pad #2** - this is connected to **PB2** on the ATtiny85. This pin can be used as an analog input (known as **Analog A1**), and is also used for I2C clock and SPI clock.

Secret Reset Pad

On the off chance you want to reprogram your Gemma with an AVR burner, the bottom of the board has a large pad that is connected to the Reset pin. We use it for testing and you will likely never need it but it is there if you do.

Windows Driver Installation

To use a Trinket, Gemma, or Flora on Windows you will need to install a driver for your board. Follow the steps below depending on the board you're using:

- [Adafruit Trinket/Gemma \(black board\) Driver Installation \(http://adafru.it/eTU\)](http://adafru.it/eTU)
- [Arduino Gemma \(teal board\) Driver Installation \(http://adafru.it/eTU\)](http://adafru.it/eTU)
- [Flora Driver Installation \(http://adafru.it/eTU\)](http://adafru.it/eTU)

Note on Mac OSX or Linux you don't need to install any driver to use Adafruit's boards. Only follow these steps if you're using Windows.

Adafruit Trinket/Gemma Driver Installation

If you're using an **Adafruit Trinket** or **Adafruit Gemma** (the black circuit board Gemma, *not* the teal or blue/green Arduino Gemma--for Arduino Gemma [skip below to the next section \(http://adafru.it/eTU\)](http://adafru.it/eTU)) follow the steps below to install a custom driver for the board.

First download the signed Windows 10, 8, 7, Vista, XP driver below and unzip its contents to a folder on your computer:

Download Signed Adafruit
Trinket/Gemma Driver

<http://adafru.it/djr>

Remember this driver is for the Adafruit Gemma board and not the Arduino Gemma board. If your Gemma has a black circuit board use this driver, but if it has a teal or green/blue circuit board skip down to the next section.

If for some reason you need it there is also an [older unsigned driver for Windows 7, Vista, XP \(http://adafru.it/eFa\)](http://adafru.it/eFa). Stick with using the signed driver above unless you know you need the unsigned driver.

Then plug in the Trinket or Gemma to a known-good USB cable to start the process. You should see the green power LED lit and the red bootloading LED pulse indicating that the Trinket is ready to start bootloading. If you've programmed the Trinket since getting it, you can always get it back to the bootloader state by pressing the small onboard reset button.

Finally, follow the steps on this page to install the custom Trinket/Gemma driver. (<http://adafru.it/cDY>)

If you're using Windows 10 you might need to [follow the steps at the bottom of the page to manually install the driver \(http://adafru.it/eTU\)](#).

Arduino Gemma Driver Installation

If you're using the **Arduino Gemma** (teal or blue/green circuit board, not the black circuit board Adafruit Gemma--if you have a Adafruit Gemma [skip back up to the previous section \(http://adafru.it/eTU\)](#)) follow the steps below to install a custom driver for the board.

First download the driver below and unzip it to a folder on your computer:

Download Arduino Gemma Driver

<http://adafru.it/fSO>

Remember this driver is for the *Arduino Gemma* board and not the *Adafruit Gemma* board. If your Gemma has a teal or blue/green circuit board use this driver, but if it has a black circuit board skip back up to the previous section.

Then plug in the Gemma to a known-good USB cable to start the process. You should see the green power LED lit and the red bootloading LED pulse indicating that the Gemma is ready to start bootloading. If you've programmed the Gemma since getting it, you can always get it back to the bootloader state by pressing the small onboard reset button.

Finally, [follow the steps on this page to install the custom Gemma driver. \(http://adafru.it/cDY\)](#)

If you're using Windows 10 you might need to [follow the steps at the bottom of the page to manually install the driver \(http://adafru.it/eTU\)](#).

Flora Driver Installation

If you're using the Flora board follow the steps below to install a custom driver for the board. For Trinket or Gemma [skip back up to the section at the top \(http://adafru.it/eTU\)](#).

Begin by plugging in the Flora board. Look for a green power LED (left of the mini USB jack) and a pulsing red LED (right of the mini USB jack).

WinXP users, skip down past these few photos, you should get a popup asking you for a driver location, just browse to it as shown.

Windows 8 & 10

We now have signed Windows 8/10 drivers for flora. Download the driver by clicking here. You should be able to right-click the **inf** file to install it directly, or point Windows to this uncompressed folder when prompted!

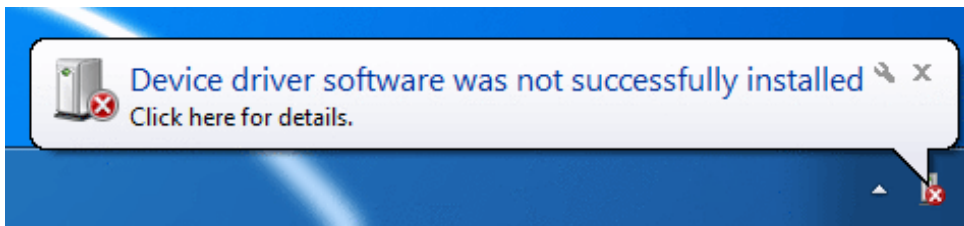
If you're using Windows 10 you might need to [follow the steps at the bottom of the page to manually install the driver \(http://adafru.it/eTU\)](http://adafru.it/eTU).

Downloads the Windows XP/7/8
signed Flora Driver

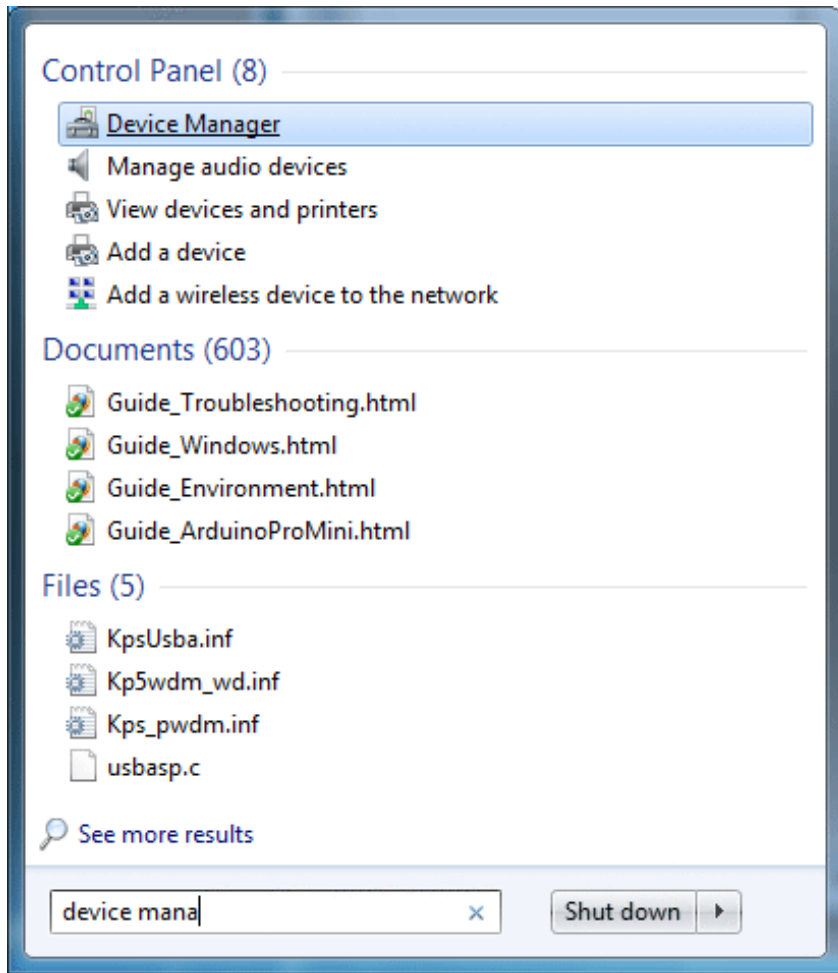
<http://adafru.it/dpT>

Windows 7

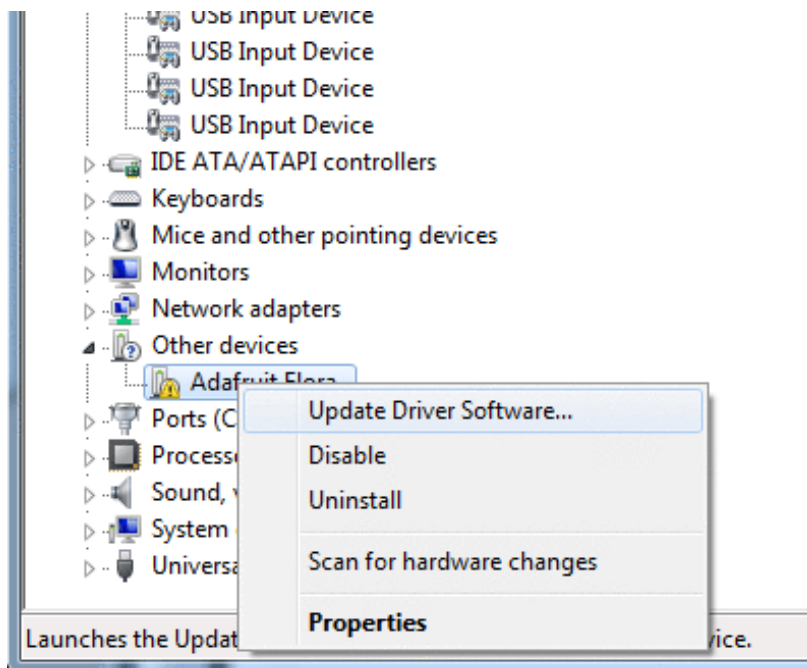
You'll see this pop up in the bottom right corner:



Now go to the Start menu (bottom left corner, the round windows logo) and click on it and start typing in **Device Manager**

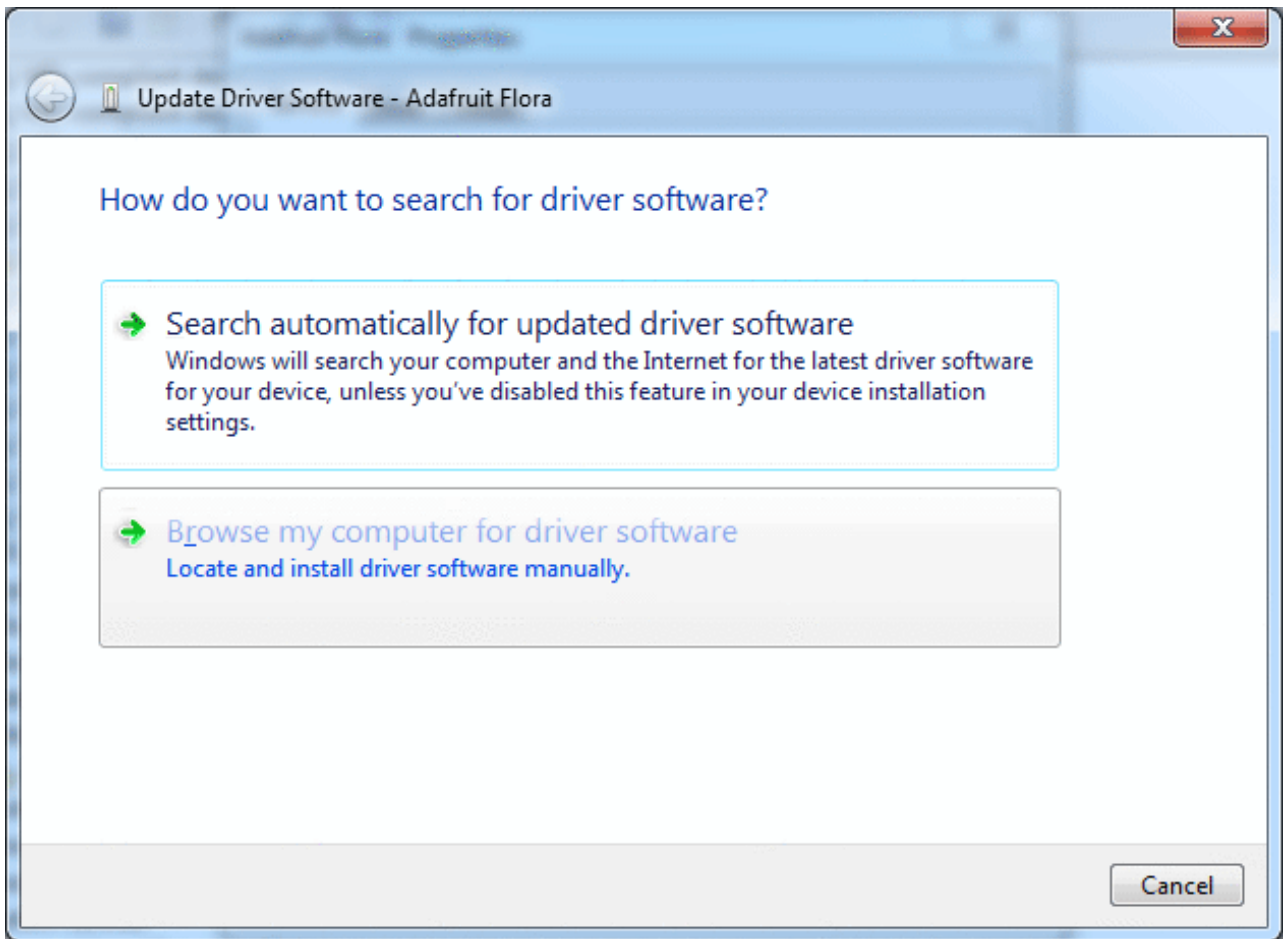


Click on the **Device Manager** (top icon in the menu) to launch it, scroll down until you see Adafruit Flora with a ! mark. Right click and select **Update Driver Software**

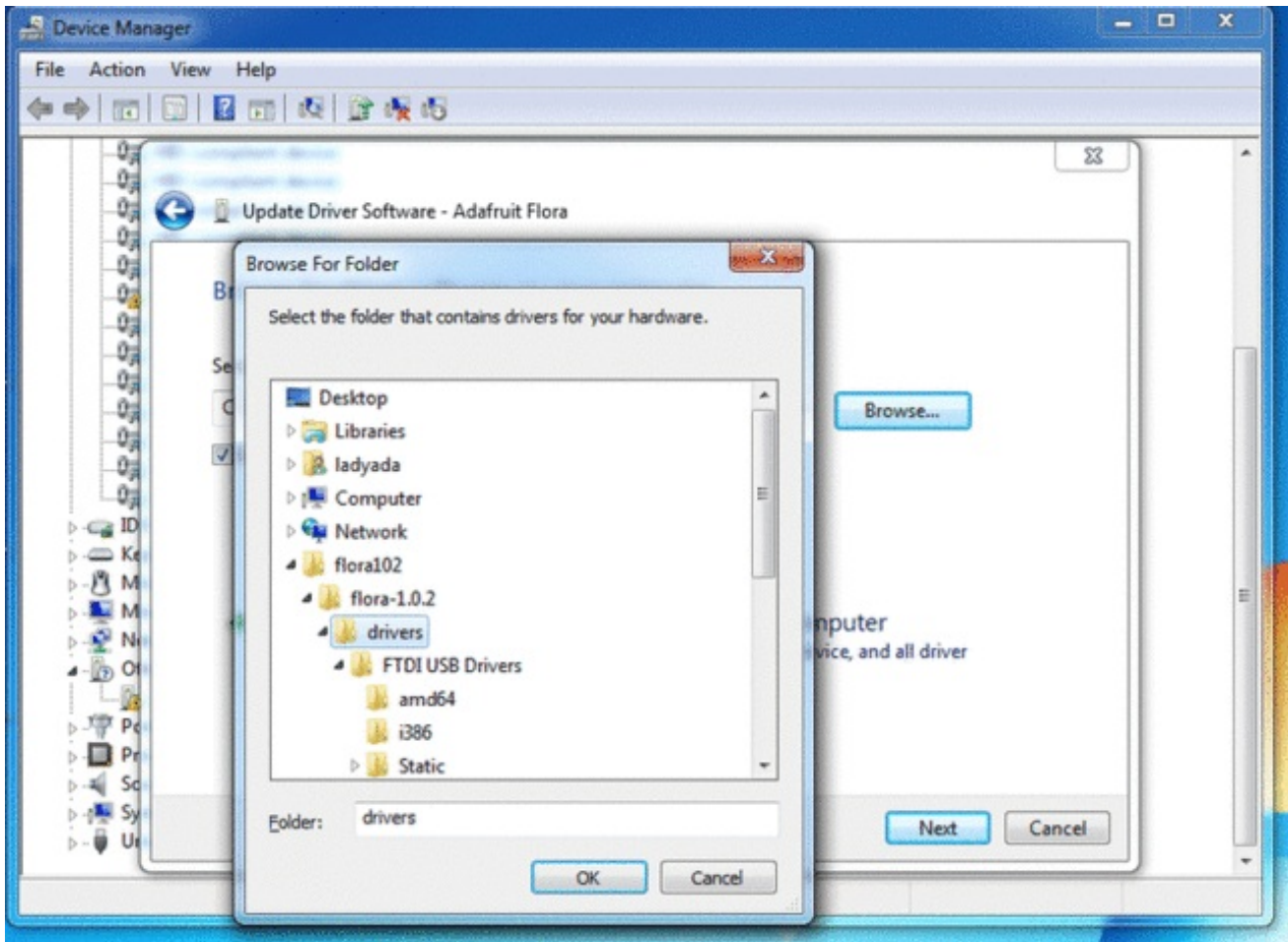


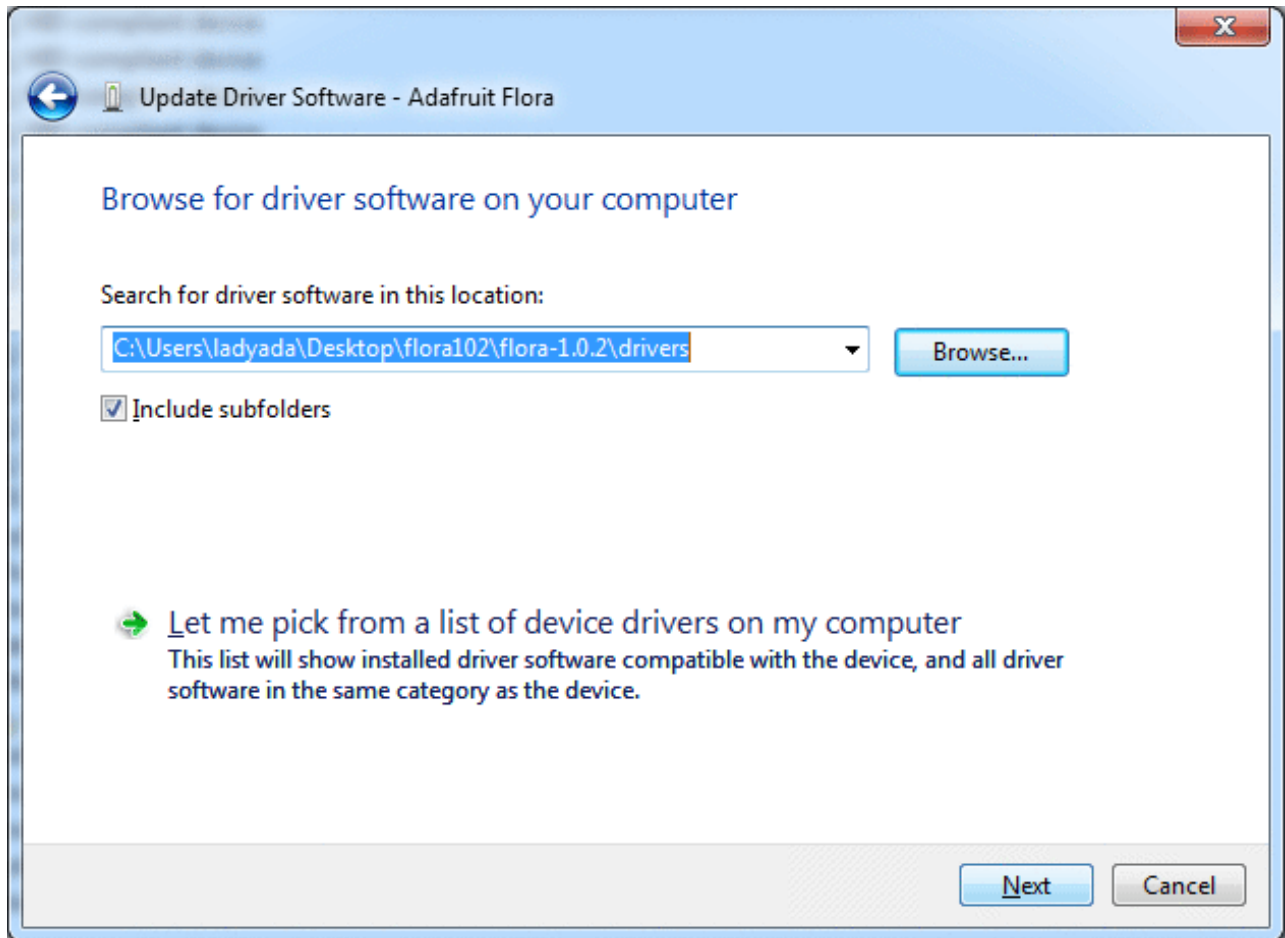
Locating the Driver File (Win 7 & XP)

OK if you have XP you'll basically start from here, click **Browse my computer for driver software**



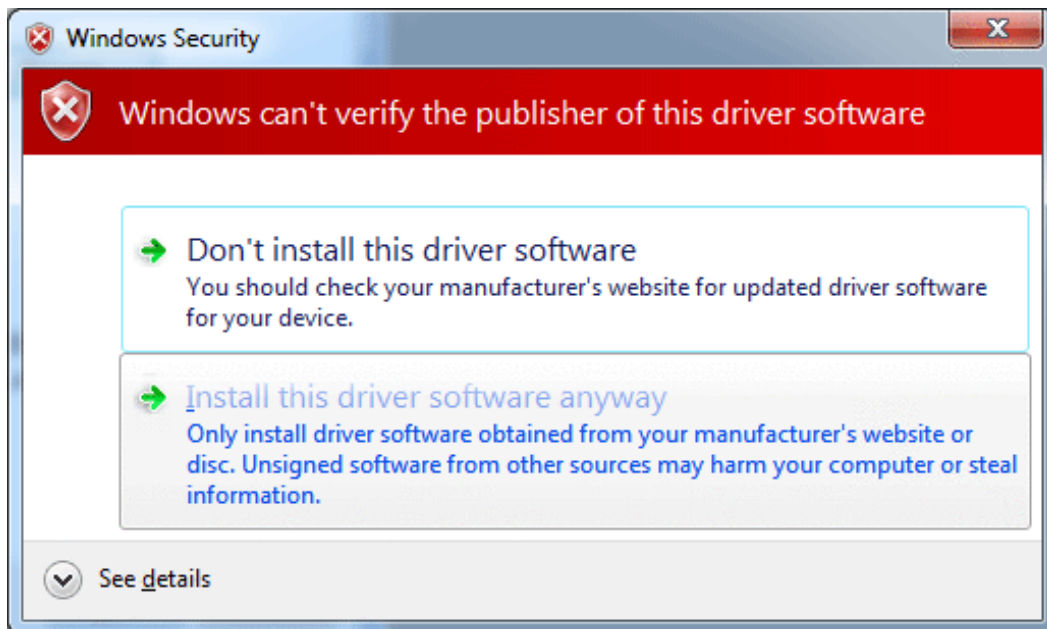
Then browse to the uncompressed Flora IDE folder, and **select the drivers folder** (not any other folder!) and click **OK**.





Make sure the folder next to the **Browse...** button is correct, then click **Next**.

When you get this scary window next, click **Install this driver software anyway**.



That's it! Thankfully you only have to do this once.

Windows 10 Driver Installation

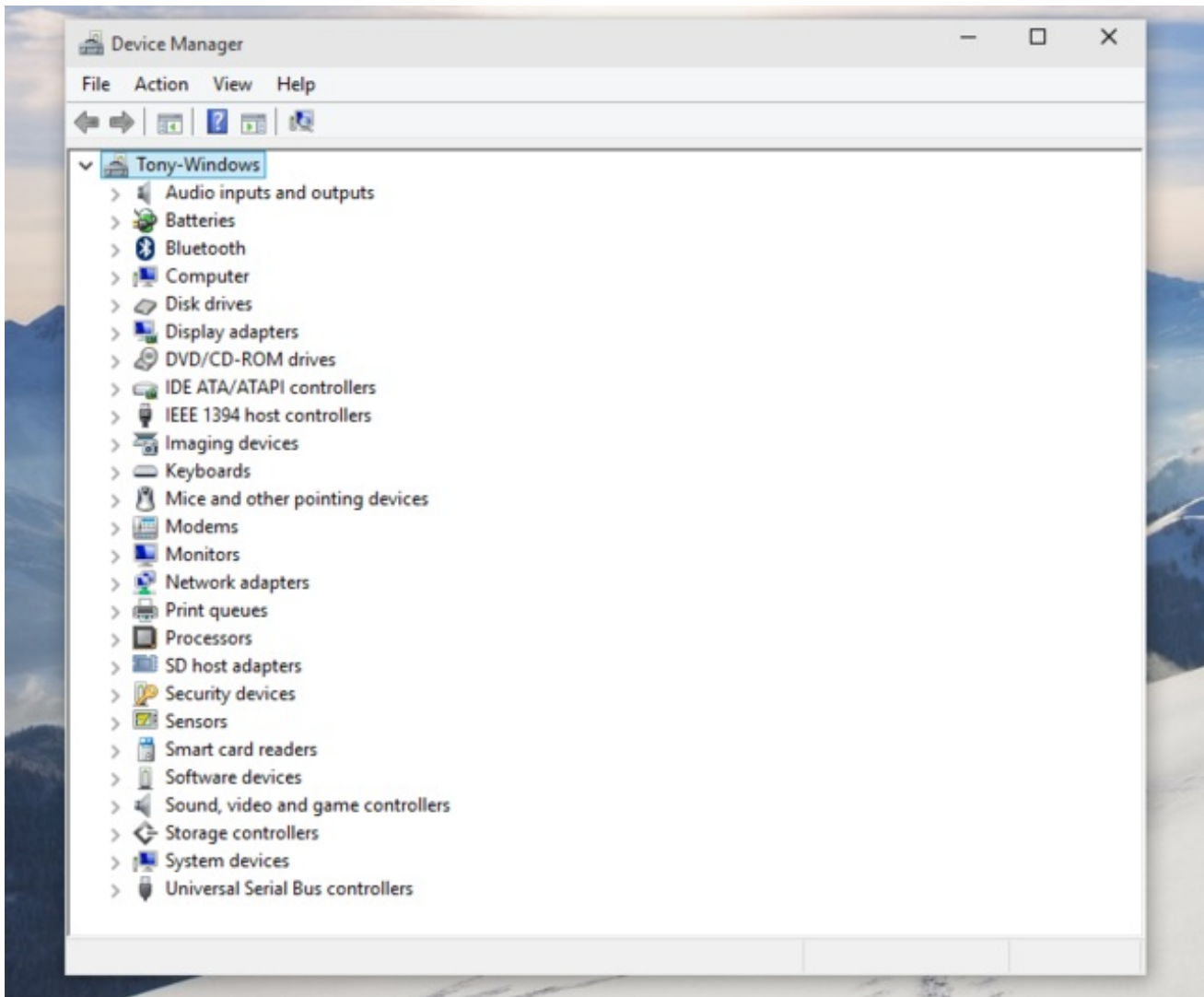
If you're using Windows 10 the process to install a custom driver is a little different than previous Windows versions. When you plug in your device, like a Trinket/Gemma/Flora, you might not get a pop up telling you to install a driver for the device. However you can follow the steps below to manually install the driver.

These instructions show how to install the driver for a Trinket, but the steps should also apply to Gemma and Flora (just expect to see a different name for the board).

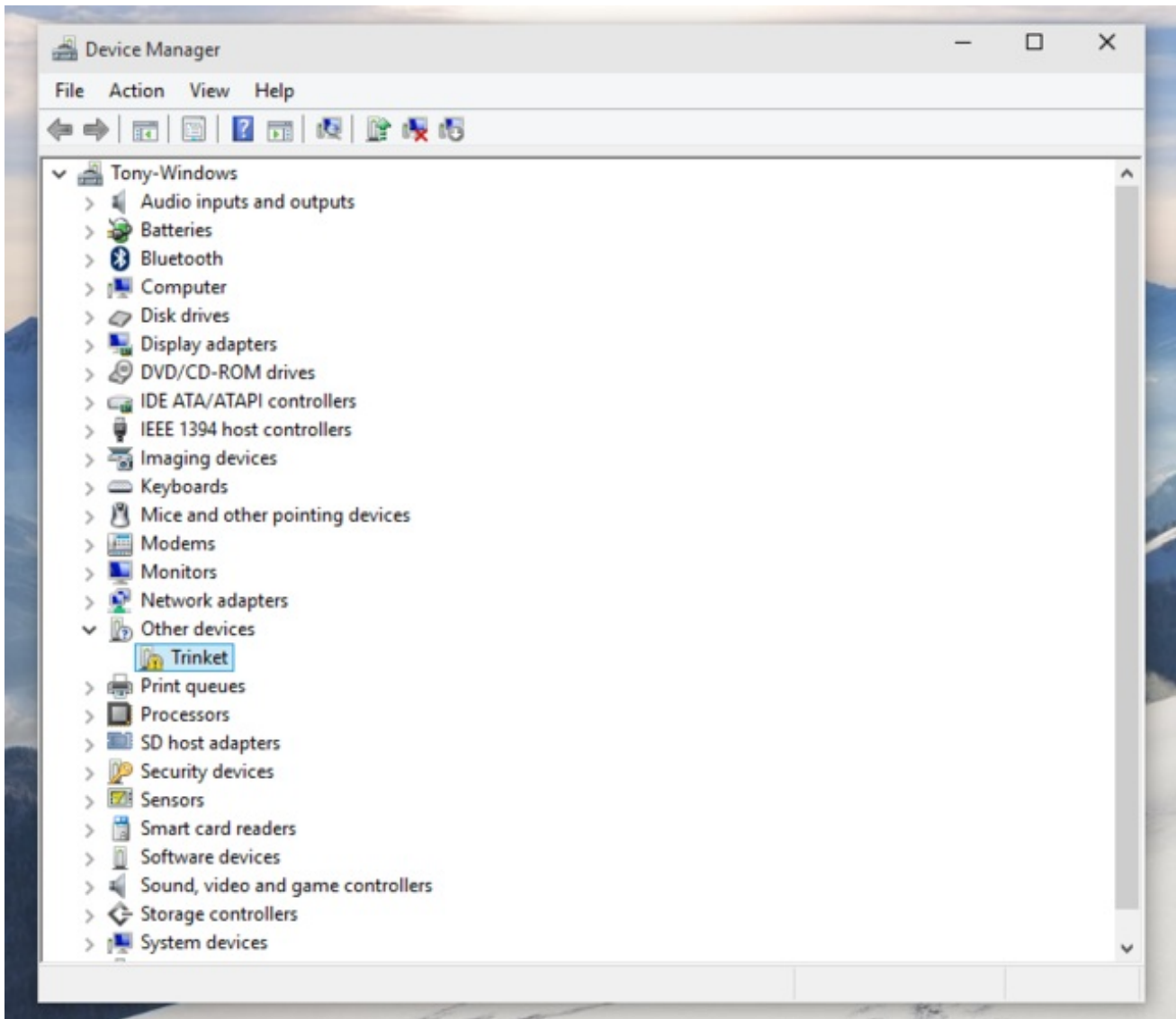
First make sure you've downloaded the signed driver for the board from the sections above. Extract the contents of the driver to a folder and remember where you put it.

Then plug in the Trinket/Gemma/Flora. You might hear a 'ding' from Windows that a new device appeared but no dialog to install a driver. Don't worry keep following along to get the driver installed.

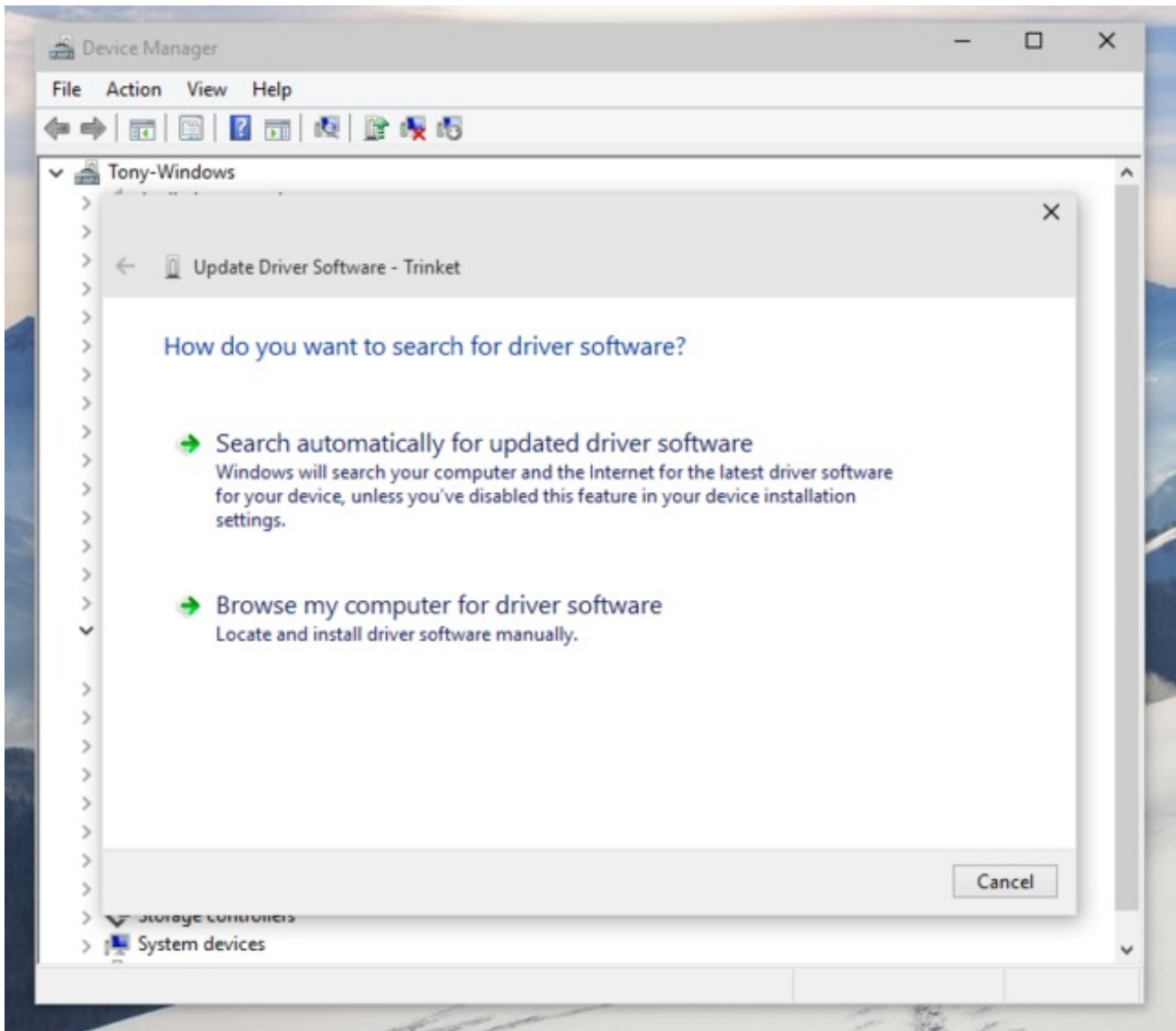
Now open **Device Manager** by clicking the start button and typing **Device Manager**, then clicking the device manager result to run it. You should see something like the following:



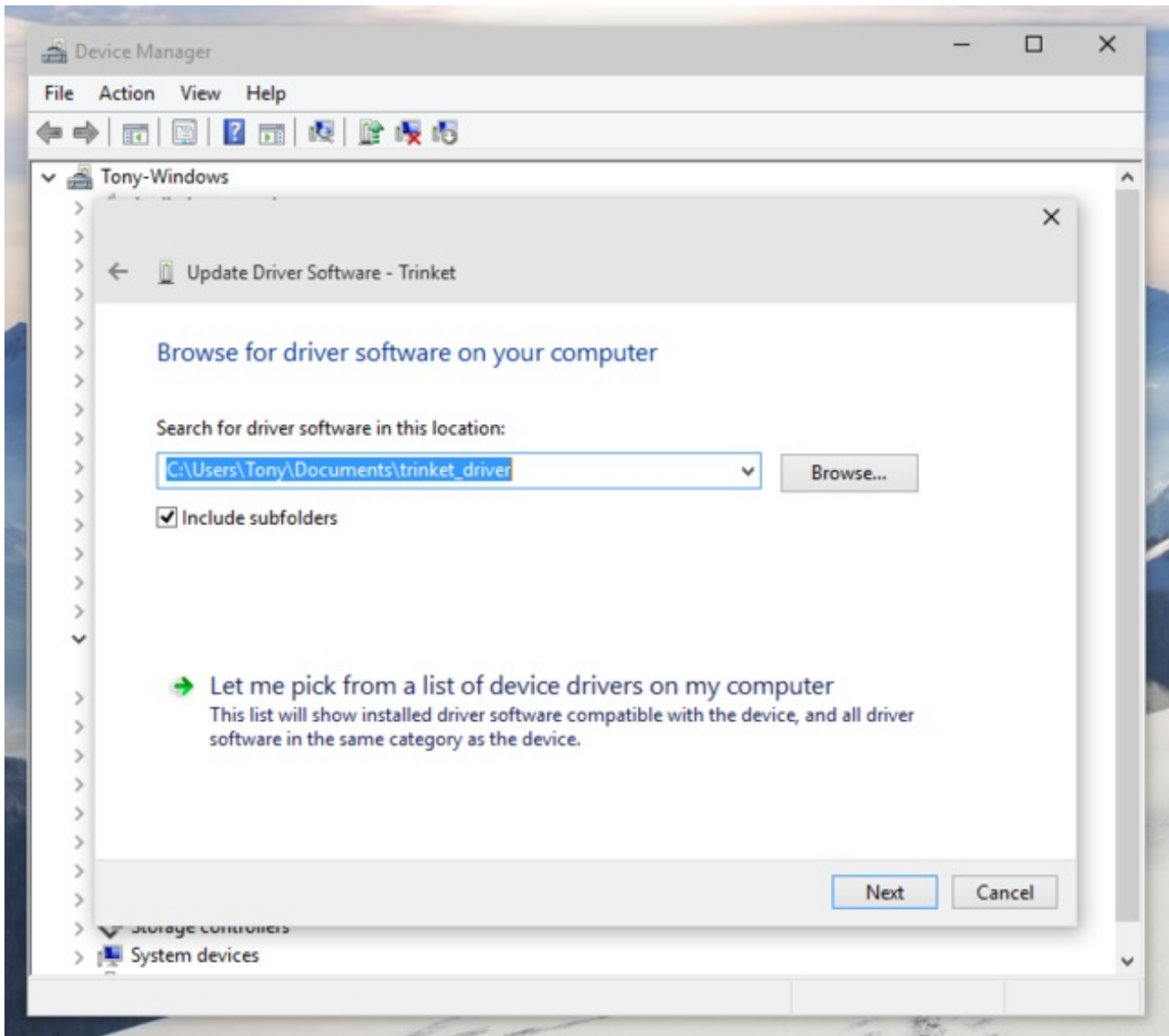
Press the bootloader button on the Trinket/Gemma and after a few moments you should see device manager refresh and display the board under an **Other devices** node:



Right-click the board (like Trinket in this example) and click the **Update Driver Software...** option at the top. This should open a new dialog that looks like the following:

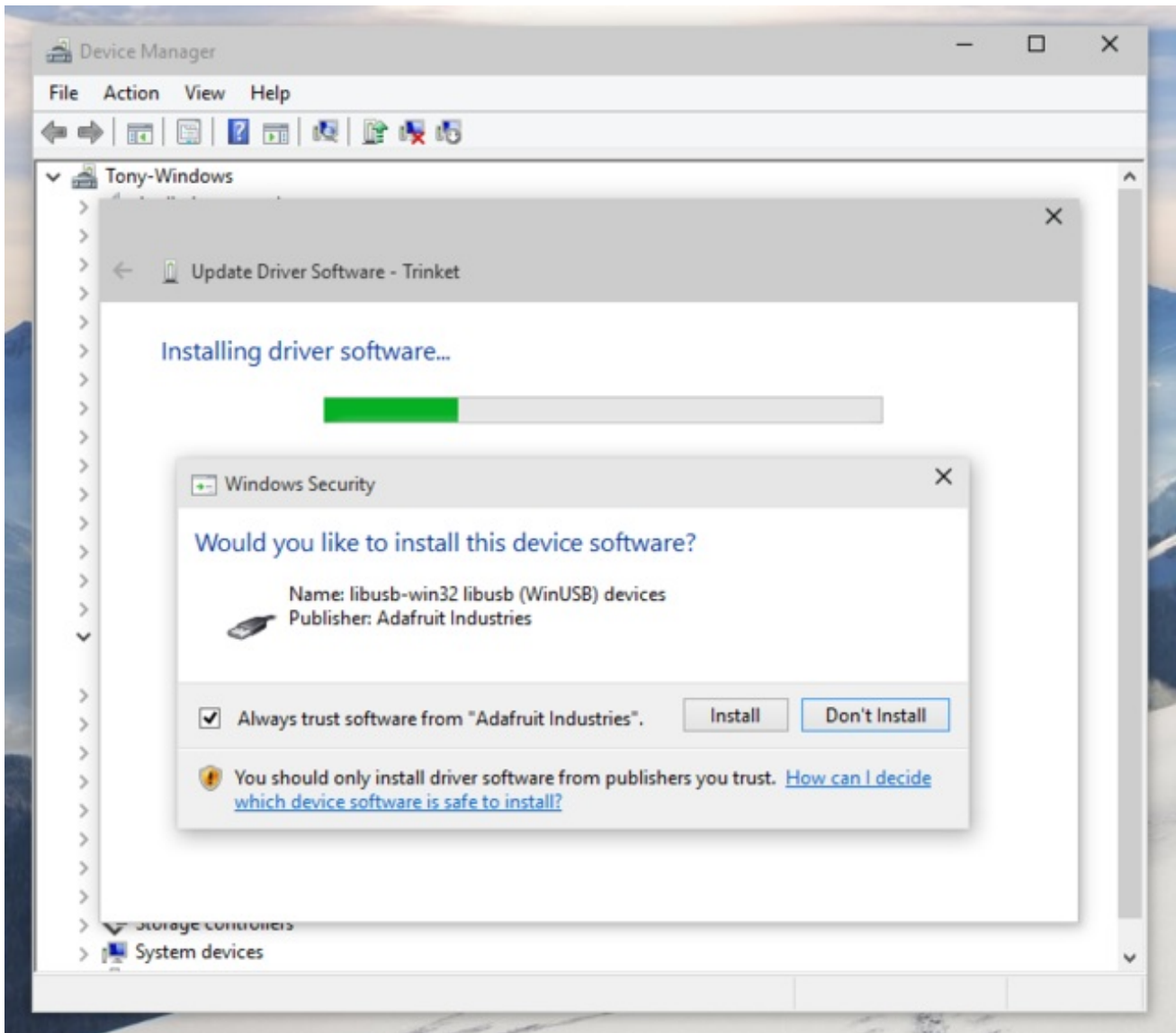


Select the **Browse my computer for driver software** option at the bottom. Now you should see a dialog to find the driver like below:



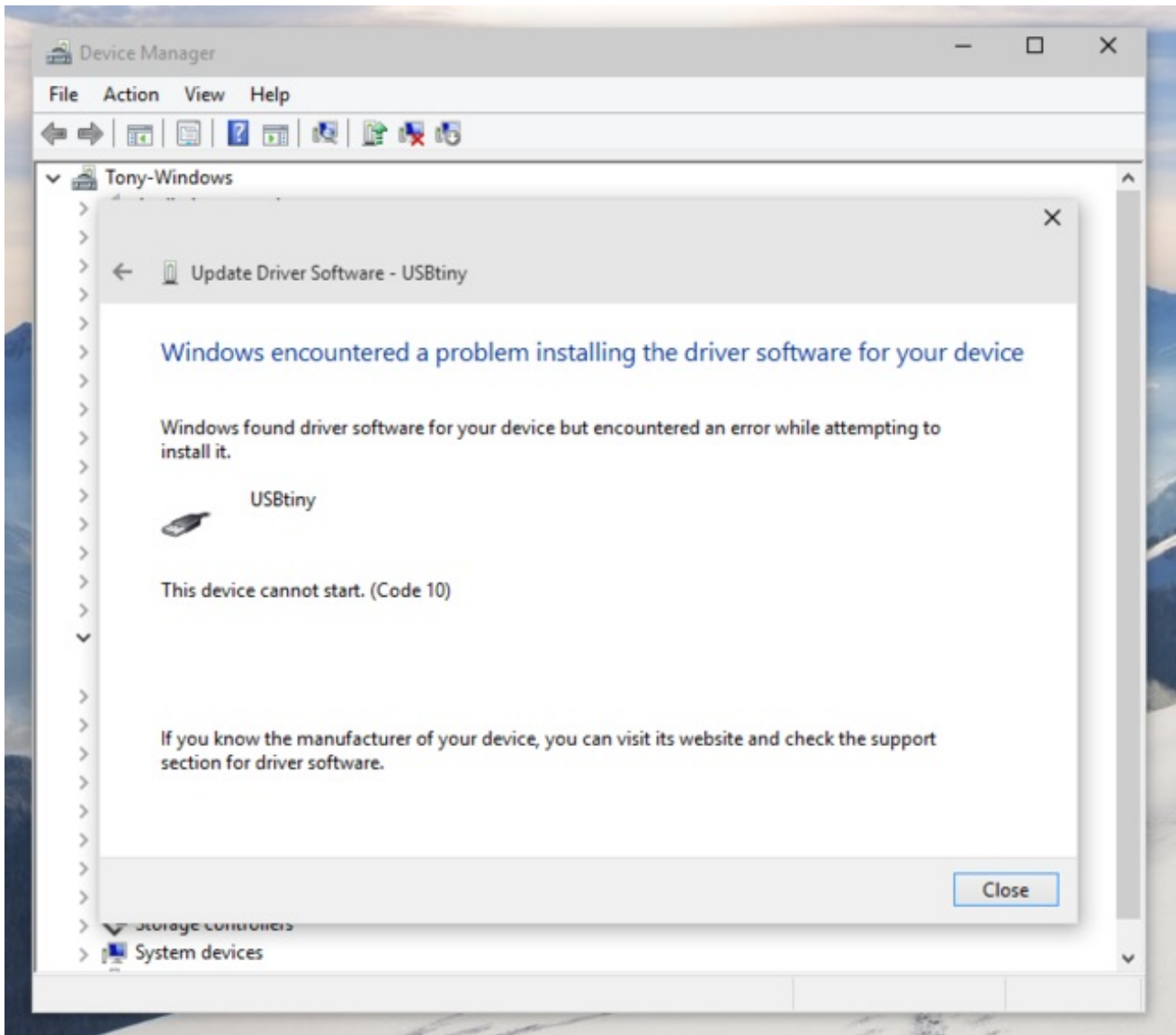
Click **Browse...** and navigate to the folder where you unzipped the driver that was downloaded earlier. Make sure the 'Include subfolders' option is checked so Windows is sure to find the driver. Then click the **Next** button at the bottom.

You should see the driver installation start. During the installation you will probably see a pop-up like below that asks if you want to allow the Adafruit driver to be installed:



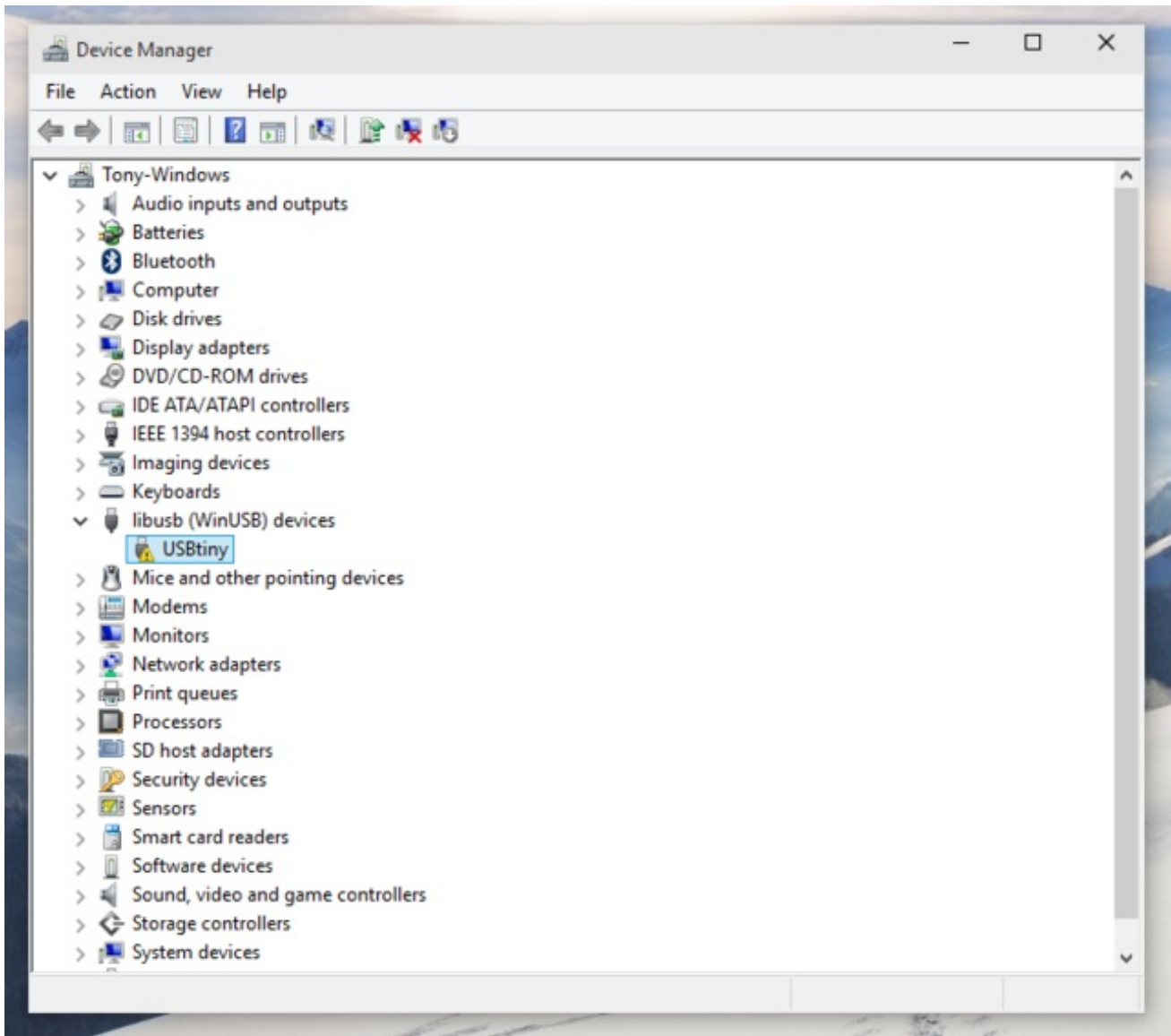
Make sure **Always trust software from "Adafuit Industries"** is checked, then click **Install**.

The driver should finish installing. Note that you might see a "This device cannot start (Code 10)" failure like the following:

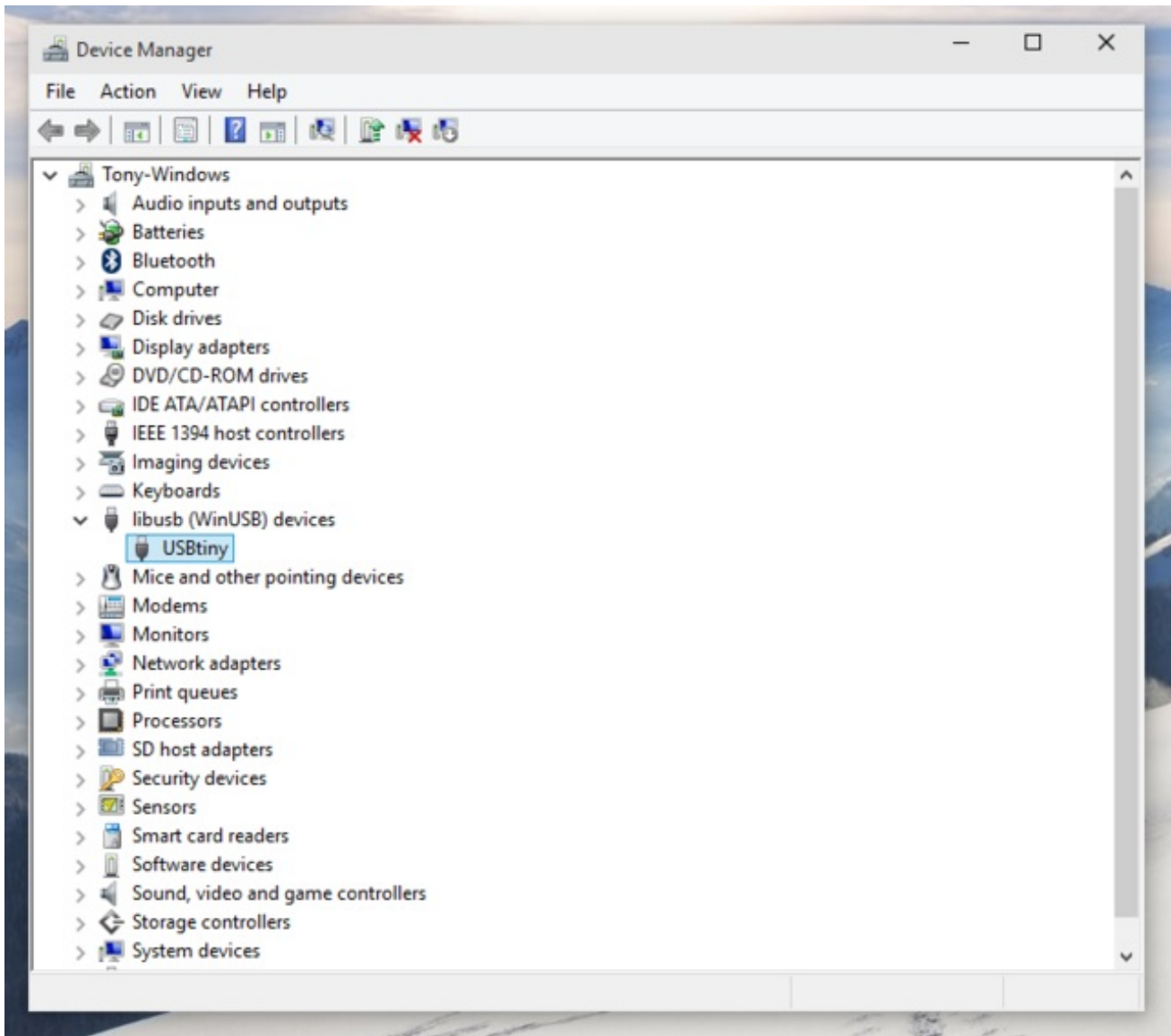


Don't worry! This just means the Trinket/Gemma bootloader timed-out before the driver finished installing. Everything should still be fine and you can continue along with the installation. Click **Close** to finish the install.

Now device manager should show a **libusb (WinUSB) devices** node with the **USBtiny** device below it:



Don't worry if the device has an exclamation point next to it indicating a problem. Press the Trinket/Gemma bootloader button and after a moment device manager should update to show the device is working:



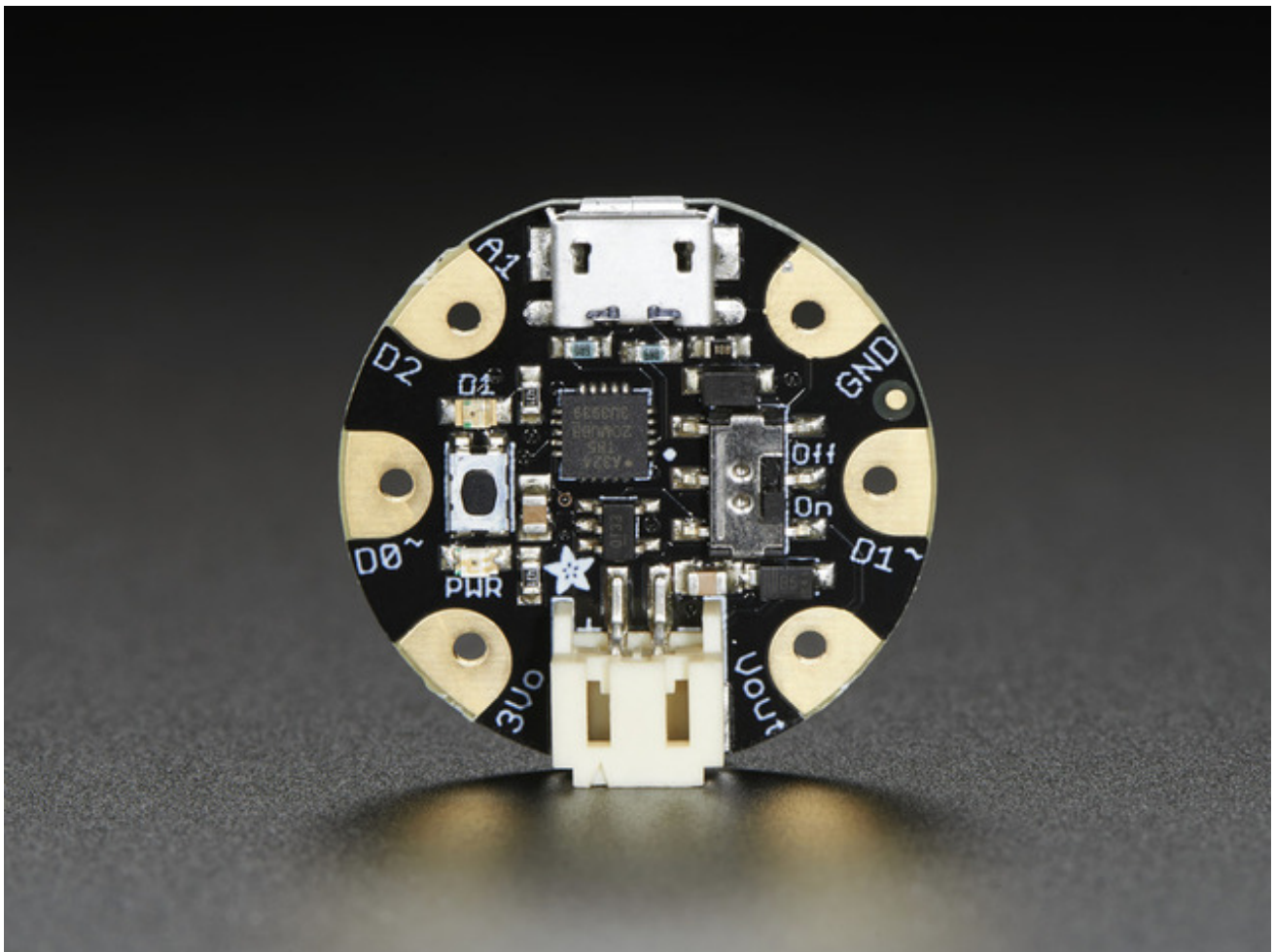
That's it, your board should be ready to go and program with the Arduino IDE!

About the Bootloader

One of the challenges with the Gemma is that we wanted to have a built-in USB bootloader, but the ATtiny85 doesn't have built-in USB hardware! There are existing USB bootloaders that can work on the 't85 but they use other companies' USB VID/PIDs. Since it not permitted by [the USB developer's group](http://adafru.it/cDW) to use others' VID/PIDs we had to adapt one of these existing bootloaders to use our USB ID, but we also wanted to not have to re-compile avrdude or the Arduino IDE since that's such a pain.

So instead, [Frank \(our awesome engineer with mad USB chops\)](http://adafru.it/cDX) created a USB bootloader that combines the elegance of V-USB with the well-supported and tested nature of the USBtinyISP. This bootloader looks just like a USBtinyISP - and since it uses the unique Adafruit VID/PID we own and that we added to avrdude so long ago, it works with only very minimal configuraton tweaks. No need to recompile anything, whew!

Please note: you cannot use the Adafruit USB VID/PID for your own non-Gemma products or projects. Purchase a USB VID for yourself at <http://www.usb.org/developers/vendor/>



Don't forget to plug in the Gemma via a known-good USB cable to start the process. You should see the green power LED lit and the red bootloading LED pulse indicating that the Gemma is ready to start programming. If you've programmed the Gemma since getting it, you can always get it back to the bootloader state by pressing the small onboard reset button.

Special Notes on using Gemma with Linux

Gemma is guaranteed supported on Linux operating system at this time - try Mac OS or Windows! However, you can try the following - it does work for some computers

Linux is fairly picky about who can poke and prod at the USB port. You can always run **avrdude** or **Arduino IDE** as root, which will make sure you have the proper permissions. If you want to be super-cool you can add a *udev* rule which will let any user (who is not root) connect to the USBtiny driver. That way you don't have to be root all the time!

Check <http://learn.adafruit.com/usbtinyisp/avrdude#for-linux> (<http://adafru.it/cf3>) for what to add to your udev file.

How to start the bootloader

Before you try to upload code to the Gemma it must be in the Bootloader Mode. That means its listening for a sketch or program to be sent to it

When the Gemma is in bootloader mode, the red LED will be pulsing. Once the red LED stops pulsing, you must press the reset button to re-enter bootloader mode

The Gemma must be connected to a computer via a USB cable to enter bootloader mode. You can enter the bootloader mode by pressing the little button on the board with your fingernail. The bootloader will 'time out' after 10 seconds, so to re-enter the bootloader mode just re-press the button!

Don't press-and-hold the reset button, be sure to press-and-release!

See the video below for what it looks like to plug it in, have the LED pulse in bootloader mode, time out and then press reset to restart the bootloader. The board shown is a Trinket, which uses the same upload system as Gemma.

Setting up with Arduino IDE

Chances are, you picked up a Gemma because it is programmable with the Arduino IDE. Note that the Gemma is not a full Flora or Arduino-compatible, it uses a different (smaller) chip than the Flora, Uno, Mega, Leonardo or Due. However, there are many small sketches and libraries that will work just fine. Some may not even need anything other than pin number changes.

Even though Gemma has a USB connector, it does not have a "Serial Console" capability, so you cannot use Serial to send and receive data to/from a computer!

Arduino IDE Setup

Just follow the steps in the steps in the [Adafruit Arduino IDE setup guide \(http://adafru.it/eUF\)](http://adafru.it/eUF) to easily install a pre-configured Arduino IDE to program Gemma!

If you are running Arduino IDE 1.6.4 or greater, you can also use [this quickstart guide to add in the Gemma plugin \(http://adafru.it/f7Y\)](http://adafru.it/f7Y)

When you're finished installing the IDE come back to this page to continue the Gemma guide.

Blink!

After installing the Arduino IDE with support for Adafruit's boards you can load a simple blinking LED example to test uploading to Gemma works as expected. Open the Arduino IDE and replace the sketch code with the following blink code:

```

/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.

This example code is in the public domain.

To upload to your or Trinket:
1) Select the proper board from the Tools->Board Menu (Arduino Gemma if
   teal, Adafruit Gemma if black)
2) Select the uploader from the Tools->Programmer ("Arduino Gemma" if teal,
   "USBtinyISP" if black Gemma)
3) Plug in the Gemma into USB, make sure you see the green LED lit
4) For windows, make sure you install the right Gemma drivers
5) Press the button on the Gemma/Trinket - verify you see
   the red LED pulse. This means it is ready to receive data
6) Click the upload button above within 10 seconds
*/

int led = 1; // blink 'digital' pin 1 - AKA the built in red LED

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}

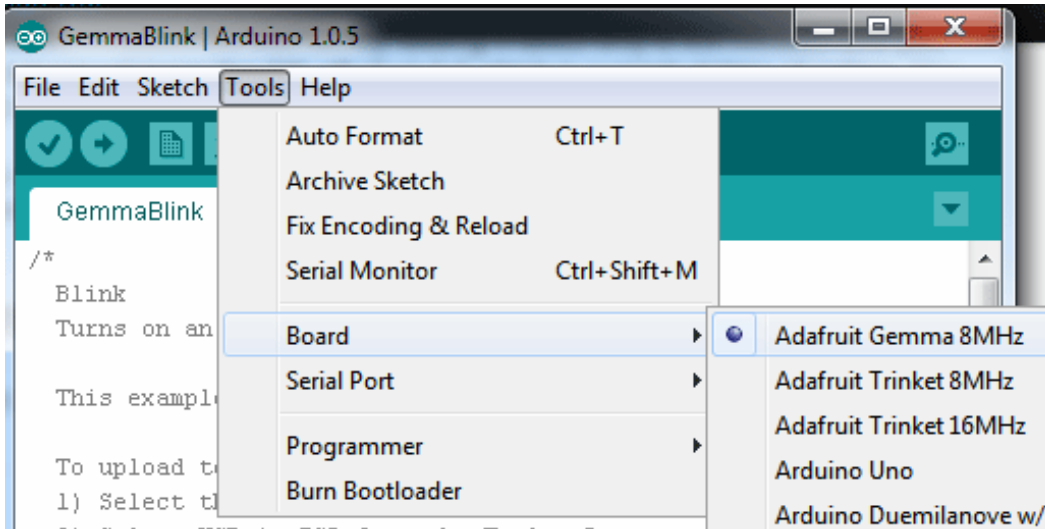
```

Make sure to pick the right version of the Gemma board in the IDE! If you're using the Adafruit Gemma it won't program correctly when selecting the Arduino Gemma option and vice versa! Carefully read the instructions below to pick the right board.

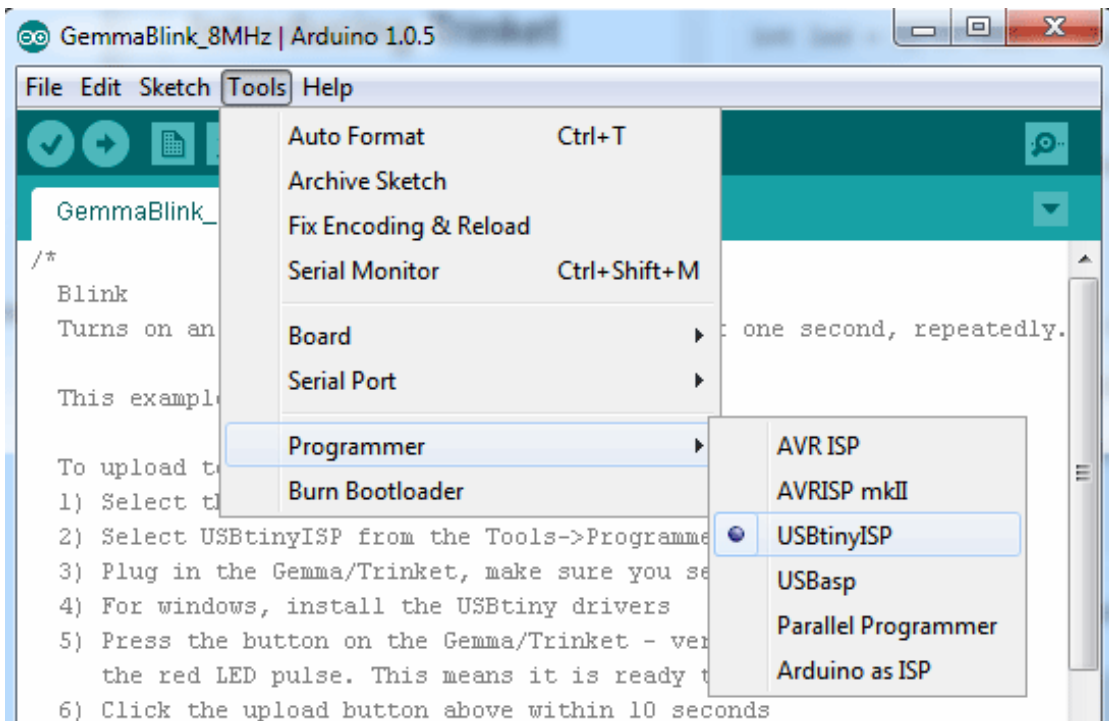
Adafruit Gemma (Black Gemma)

If you're using the **Adafruit Gemma** (with a **black** PCB board) select the **Adafruit Gemma 8MHz**

board from the **Tools->Board** menu.

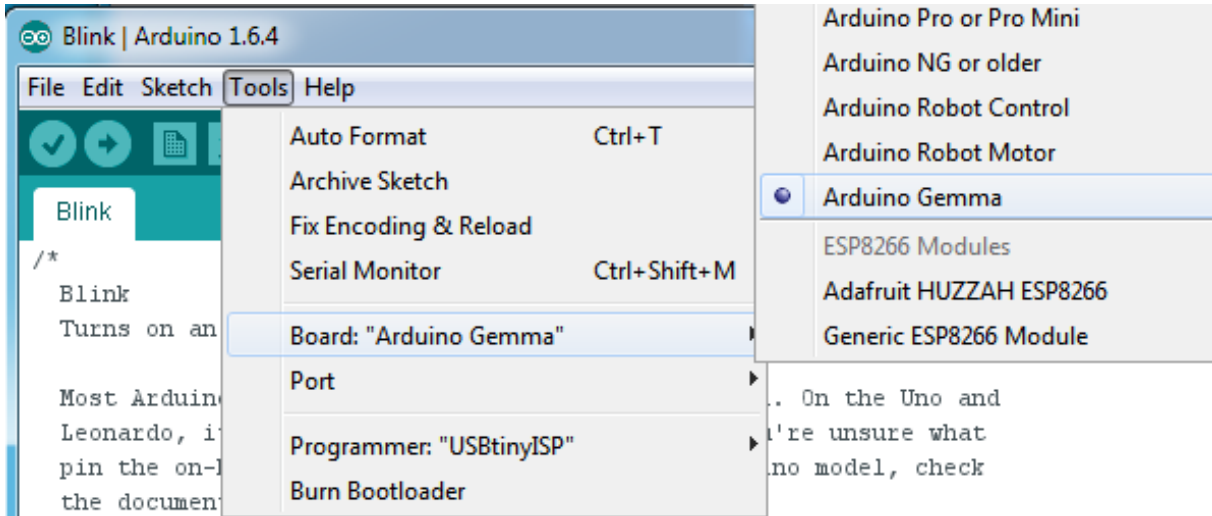


Then, select **USBtinyISP** from the **Tools->Programmer** sub-menu

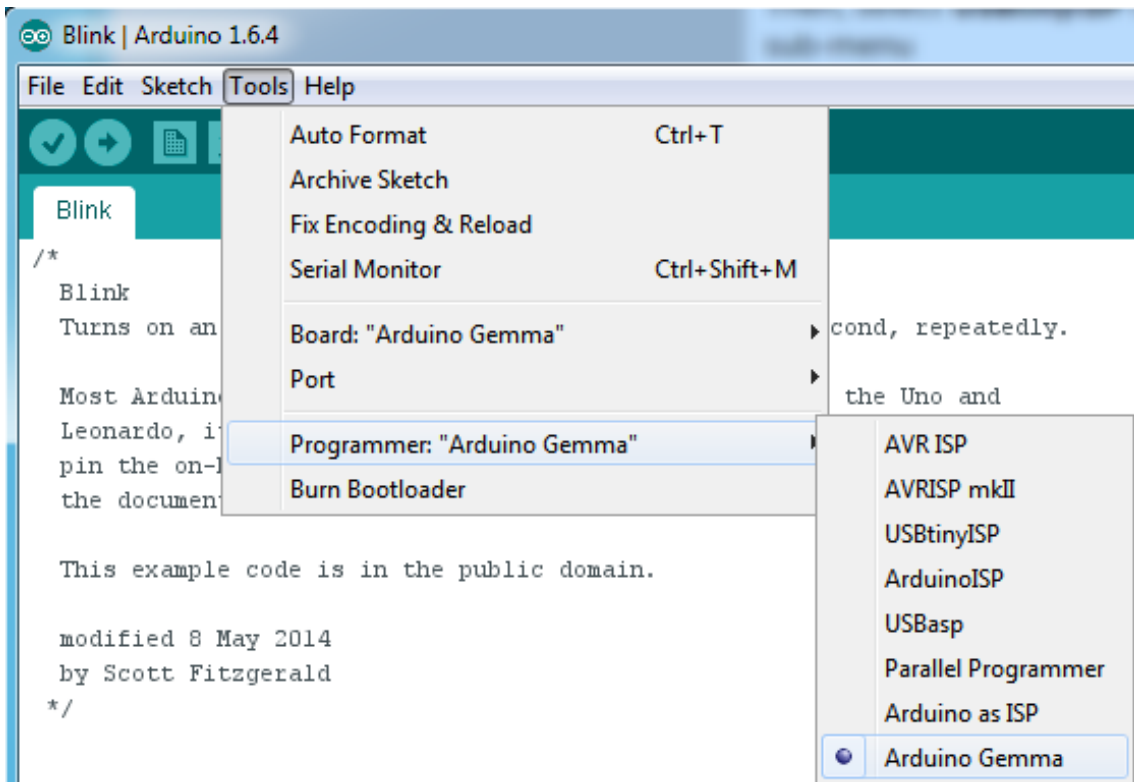


Arduino Gemma (Teal Gemma)

However if you're using the newer **Arduino Gemma** (with a **teal** PCB board) select the Arduino Gemma board from the **Tools->Boards** menu.



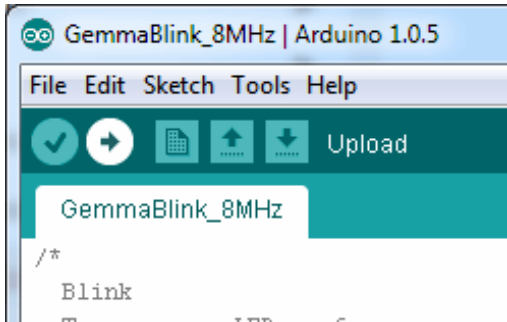
Then select **Arduino Gemma** as the Programmer type



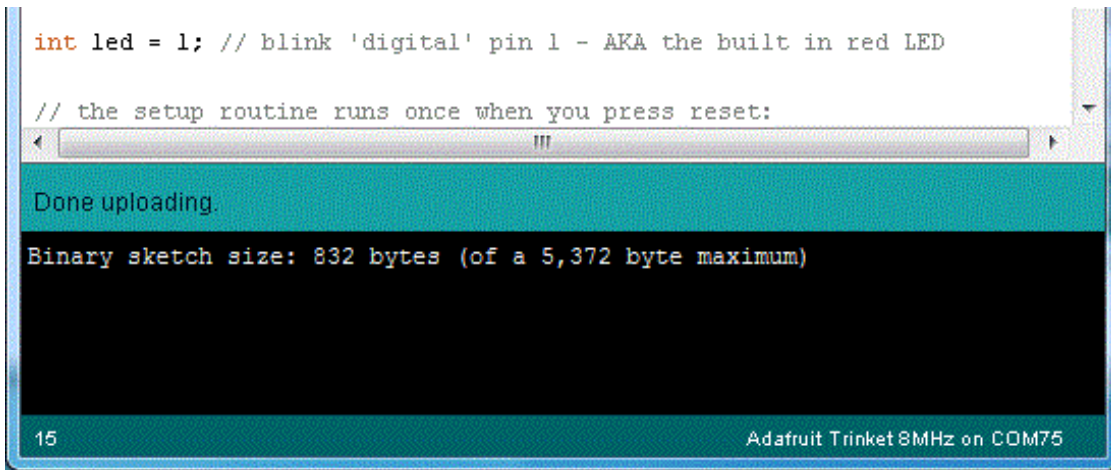
Get Into Bootloader Mode

Plug in the Gemma, make sure you see the green LED lit (power good) and the red LED pulsing. Press the button if the red LED is not pulsing, to get into bootloader mode.

Click the **Upload** button (or select **File->Upload**)



If everything goes smoothly you should see the following (no red error messages) and of course, the red LED on the Gemma will blink on/off once a second



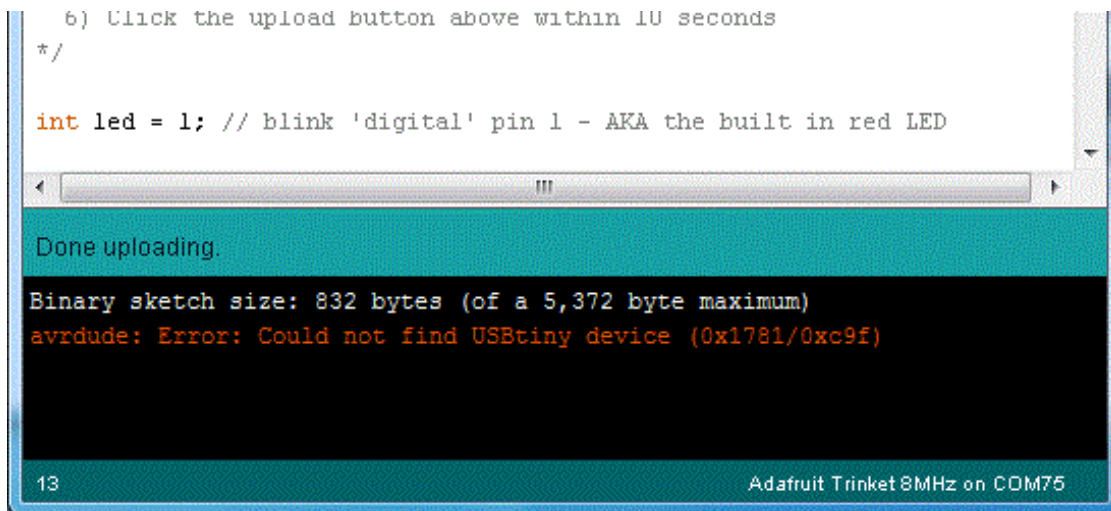
Something Went Wrong!

If you get the error message `avrdude: Error: Could not find USBtiny device (0x1781/0xc9f)`

That means the bootloader wasn't active. Make sure to press the button on the Gemma to activate the bootloader *before* clicking the Upload button.

```
6) Click the upload button above within 10 seconds
*/

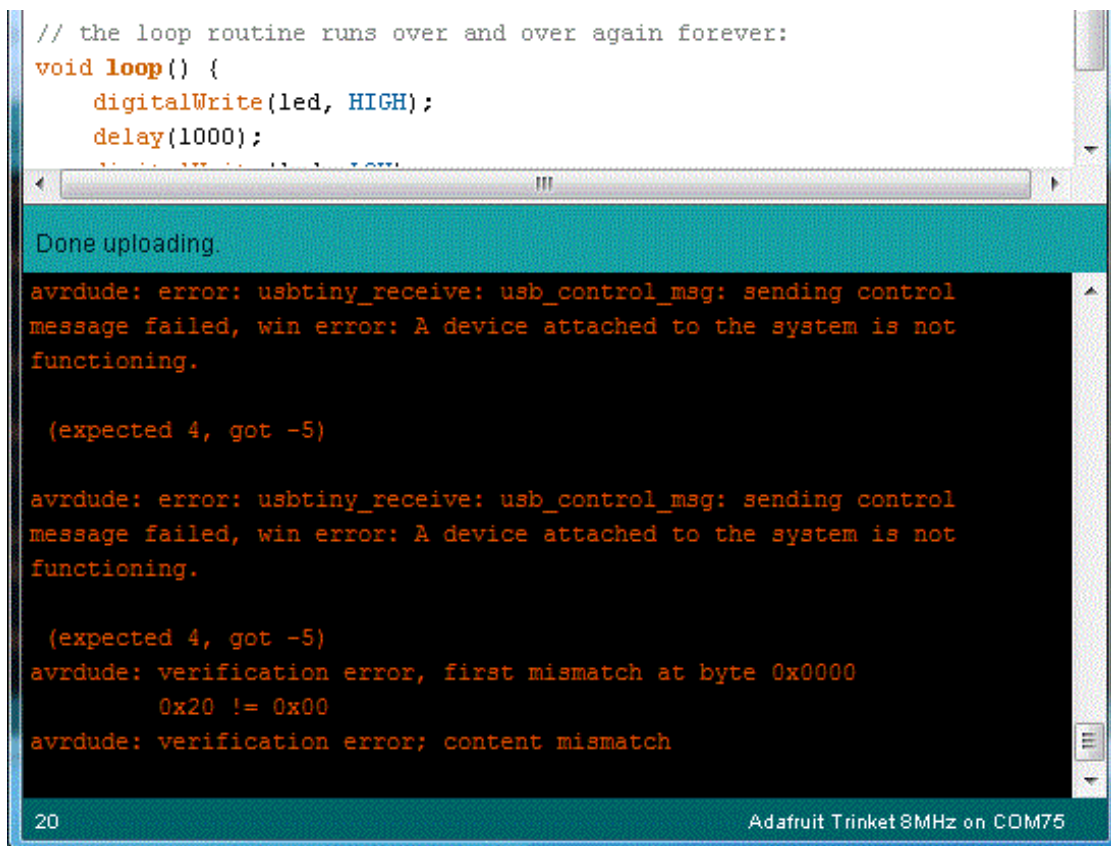
int led = 1; // blink 'digital' pin 1 - AKA the built in red LED
```



If you get a lot of red text, errors and also a warning about Verification Failed

Check that you updated the avrdude.conf file above - if you don't update the description of the Attiny85 in the configure file by replacing it, the IDE wont know to be patient with the Gemma bootloader and will have many upload errors

```
// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```



On Linux if you get the error message "usbtiny_receive: error sending

On Linux if you get the error message "usbtiny_receive: error sending control message: Protocol error (expected 4, got -71)"

These can generally be ignored and should not interfere with the program upload. Unfortunately Linux's USB core is a little flakey communicating with the ATtiny85 processor on the Trinket/Gemma and can cause these errors. If an upload does fail, try it again as it is likely an intermittent issue.

Programming with Arduino IDE

Once you've gotten the basic Blink example to work, you can try some of the other Arduino functions and libraries. We'll be filling out this section with more example code and links to tutorials - this is just to get you started!

pinMode() & digitalWrite() & digitalRead()

You can use pinMode() to make inputs and outputs on any of digital pins #0 thru #20. digitalWrite also works well, and you can also use it with pinMode(INPUT) to activate the internal pull-up resistor on a pin.

For example, to set up digital #0 as an input, with an internal pullup, and then check if it is being pulled to ground via a button or switch and turn on the red LED when it is pressed:

```

/*
Button
Turns on an LED when a switch connected from #0 to ground is pressed

This example code is in the public domain.

To upload to your Gemma or Trinket:
1) Select the proper board from the Tools->Board Menu
2) Select USBtinyISP from the Tools->Programmer
3) Plug in the Gemma/Trinket, make sure you see the green LED lit
4) For windows, install the USBtiny drivers
5) Press the button on the Gemma/Trinket - verify you see
   the red LED pulse. This means it is ready to receive data
6) Click the upload button above within 10 seconds
*/

#define SWITCH 0
#define LED 1

// the setup routine runs once when you press reset:
void setup() {
  // initialize the LED pin as an output.
  pinMode(LED, OUTPUT);
  // initialize the SWITCH pin as an input.
  pinMode(SWITCH, INPUT);
  // ...with a pullup
  digitalWrite(SWITCH, HIGH);
}

// the loop routine runs over and over again forever:
void loop() {
  if (!digitalRead(SWITCH)) { // if the button is pressed
    digitalWrite(LED, HIGH); // light up the LED
  } else {
    digitalWrite(LED, LOW); // otherwise, turn it off
  }
}

```

analogRead()

You can read an analog voltage from digital #2 (called **A1**)

For example, to read an analog voltage on pin #2, you would call **analogRead(A1)**

analogWrite()

There are a few PWM outputs on the Trinket, you can call `analogWrite()` on digital #0 and #1

For example, to pulse the built-in LED slowly, upload this code:

```
/*
  Pulse
  Pulses the internal LED to demonstrate the analogWrite function

  This example code is in the public domain.

  To upload to your Gemma or Trinket:
  1) Select the proper board from the Tools->Board Menu
  2) Select USBtinyISP from the Tools->Programmer
  3) Plug in the Gemma/Trinket, make sure you see the green LED lit
  4) For windows, install the USBtiny drivers
  5) Press the button on the Gemma/Trinket - verify you see
     the red LED pulse. This means it is ready to receive data
  6) Click the upload button above within 10 seconds
*/

int led = 1; // pulse 'digital' pin 1 - AKA the built in red LED

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  for (int i=0; i<256; i++) {
    analogWrite(led, i); // PWM the LED from 0 to 255 (max)
    delay(5);
  }
  for (int i=255; i>=0; i--) {
    analogWrite(led, i); // PWM the LED from 255 (max) to 0
    delay(5);
  }
}
```

More...

We also know the following libraries work:

- [Adafruit NeoPixel \(http://adafru.it/aZU\)](http://adafru.it/aZU) - control up to ~150 Neopixels via a Trinket!
- SoftwareSerial - the built in SoftSerial library can (at least) transmit data on any digital pin.
- More as we do more testing and verification!

Downloads

Datasheets

Datasheet for the onboard regulator used (MIC5225 3.3V)

MIC5225-3.3

<http://adafru.it/cE4>

[Webpage for the ATtiny85, the microcontroller used in the Gemma \(http://adafru.it/cE5\)](http://adafru.it/cE5)

Source code

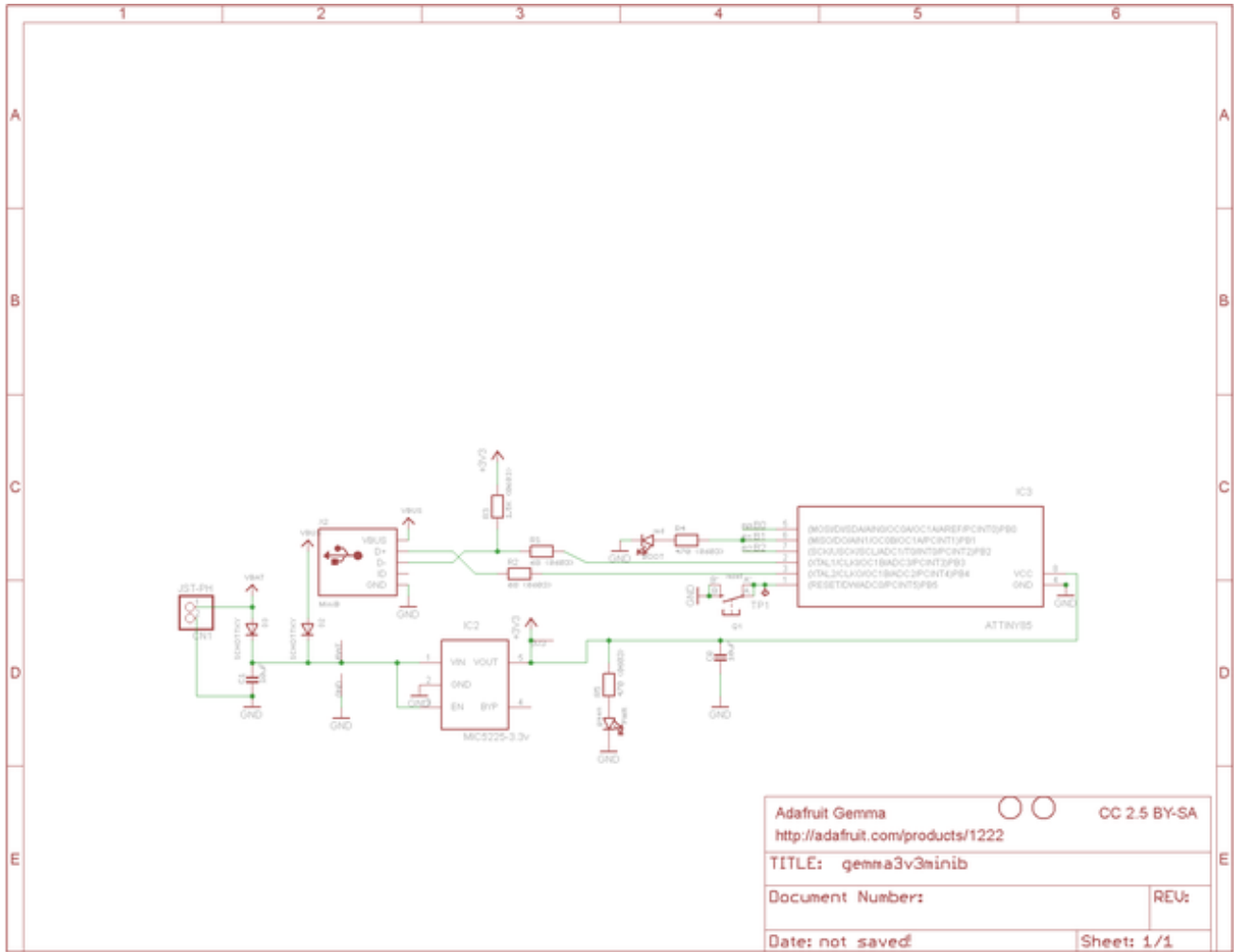
[Original code for the Trinket/Gemma bootloader on github \(http://adafru.it/cE6\)](http://adafru.it/cE6)

We do not offer any support for this code, it is released as-is!

Please note: you cannot use the Adafruit USB VID/PID for your own non-Trinket/Gemma products or projects. Purchase a USB VID for yourself at <http://www.usb.org/developers/vendor/>

Schematics

Gemma Schematic:



FAQ

When uploading with the Arduino IDE, I get a lot of "(expected 4, got -5)" warnings and then "avrdude: verification error; content mismatch"

[Check that you followed the instructions for updating the Arduino IDE, including replacing the old avrdude.conf \(http://adafru.it/cEY\)](#)- this step is not optional!

Hmm I'm still having problems with Arduino/Avrdude - and I definitely did the required updates
One fix that works for some people is to edit **avrdude.conf** and set the

```
chip_erase_delay = 900000;
```

under the **ATtiny85** heading to

```
chip_erase_delay = 400000;
```

That is, a shorter delay.

Can Gemma drive Neopixels (strips, squares, etc)? How many?

Yes! Gemma was designed to drive short segments of NeoPixels. There is enough RAM on the attiny85 to drive 100 pixels, but depending on program RAM usage you may have to scale back to 60 or 40.

To use with neopixels:

1. Connect the + power line of the strip to **Vout** on the Gemma, or to a separate 4-7VDC power source such as a 3 or 4 pack of AA batteries.
2. Connect the - common ground to the battery pack (if being used) and also to the Gemma **GND** pin
3. Connect the data in line to Gemma #1 - this will let you also see when data is being sent because the #1 red LED will flicker. You can use other pins but start with #1 since its easiest to debug and use
4. [Install the NeoPixel library as detailed in our Uber Guide \(http://adafru.it/cEz\)](#), and change the **PIN** to **1** (its 6 by default)
5. Upload and enjoy!

Can Gemma use the Flora Lux/Compass/Accelerometer/Color/GPS Sensors?

Maybe! We think we can get Gemma working with some of the basic sensors, but at this moment we don't have tutorials or examples, Flora code will not compile directly for the Gemma since the processors are different. Right now we think Gemma is best suited for basic buttons/LEDs/Neopixels type stuff

Can Gemma drive your Adafruit I2C LED Backpacks for 7-segment/matrix displays?

Short answer: yes! Check out <http://learn.adafruit.com/tap-tempo-trinket> (<http://adafru.it/cEA>) for

a tutorial on driving the 7-segment displays. Long answer: we think there's not enough space for all of the fonts for the 8x8 so you might be able to drive the 8x8 matrix in 'raw' mode (see the HT16K33 example sketch in the LEDBackpack Library) but unfortunately not with built-in font support.

That tutorial also shows how to use the TinyM I2C driver, which works great on the ATtiny85, and adapt other existing libraries for the Gemma/Trinket

Can Gemma drive a Servo?

Yup! In fact you can use 3 servos as long as they are powered by a good 5V supply, [check out this guide for more details \(http://adafru.it/cFC\)](http://adafru.it/cFC)

Gemma runs at 8MHz, but I really need it to run at 16 MHz, is this possible?

It is possible to run the Gemma at 16MHz, but the processor is not specified for 16MHz at 3.3V logic so it is considered overclocking!

However, the AVR series is pretty forgiving for overclocking, so *you may be able to run the 3V Gemma at 16 MHz*. Note that this is still overclocking, your code may run flakey or not at all!

Overclocking should not damage the AVR, but we still recommend sticking with 8 MHz only if you can get away with it!

[To run at 16Mhz, use the Trinket 16Mhz board definition and modify your sketch as described here. \(http://adafru.it/cFD\)](http://adafru.it/cFD)