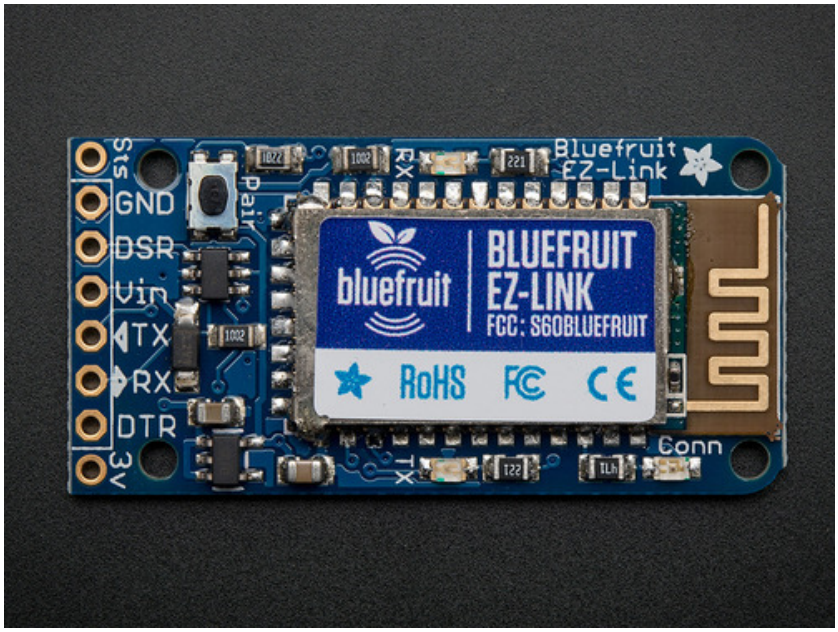
adafruit learning system

Introducing Bluefruit EZ-Link Breakout

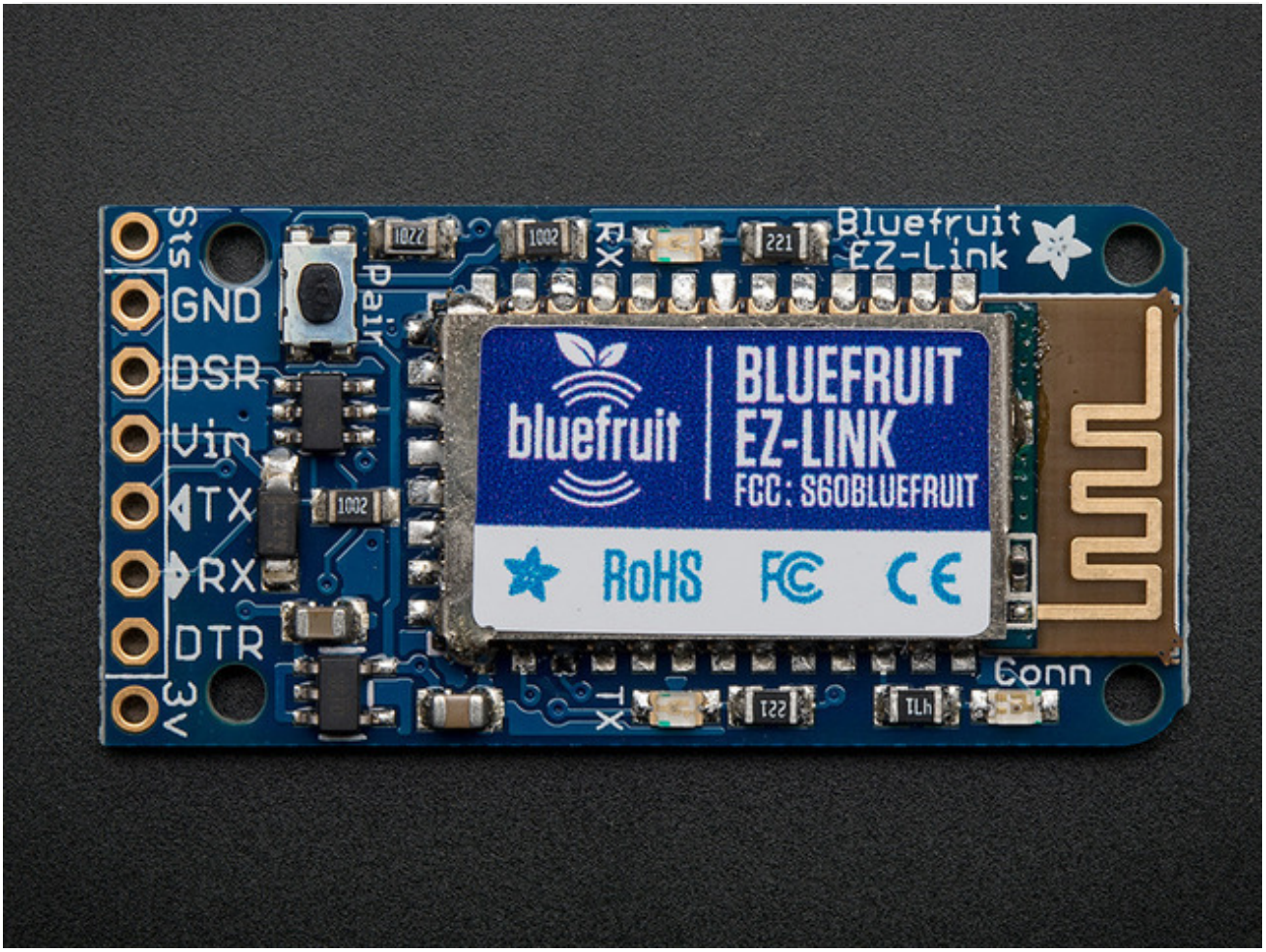
Created by lady ada



Last updated on 2014-11-27 11:00:19 AM EST

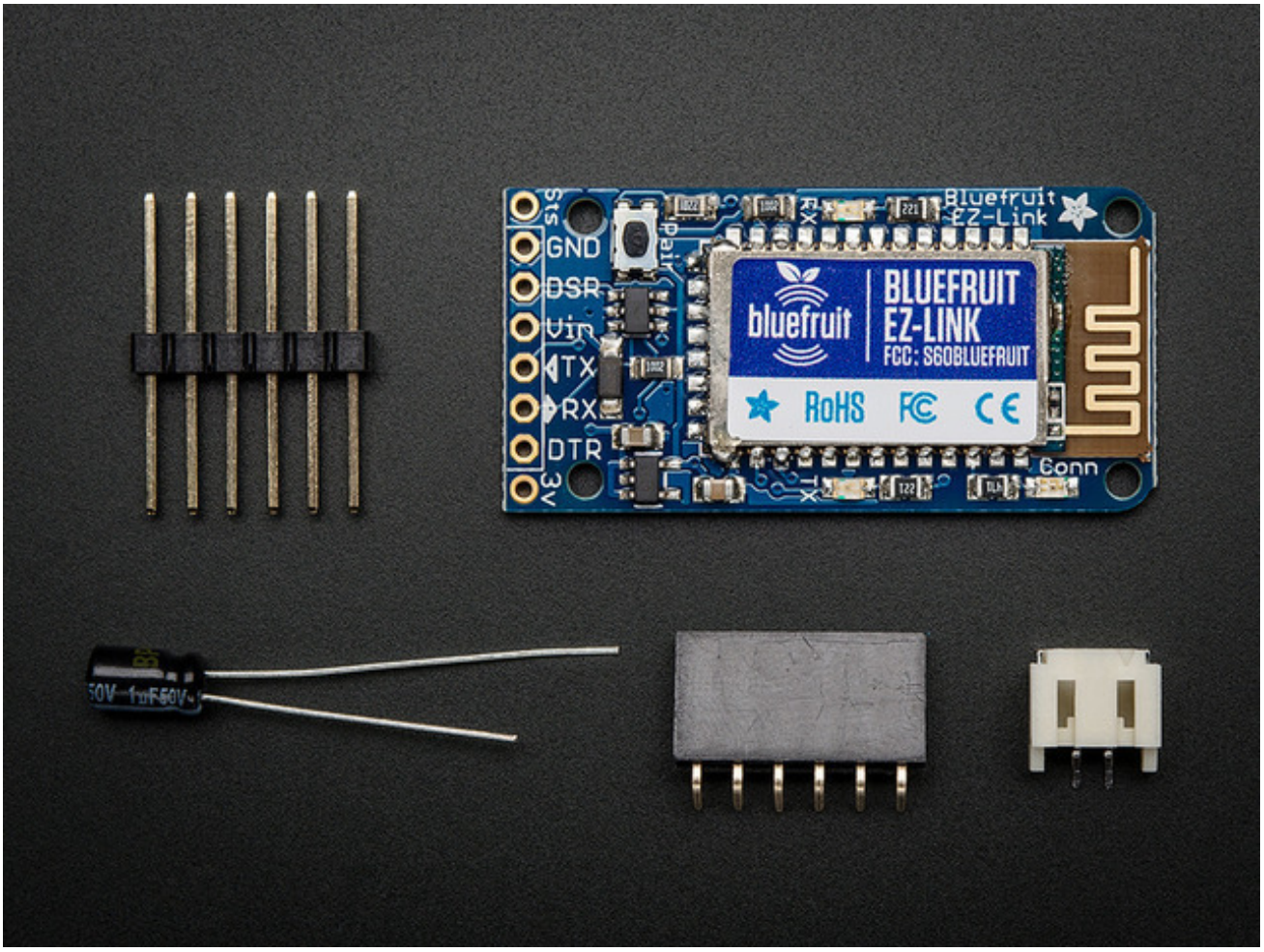
Guide Contents

Overview



We are excited to add another product to our growing Adafruit Bluefruit line, this time its the Bluefruit EZ-Link: the best Bluetooth Serial Link device ever made. Like you, we have purchased all sorts of Bluetooth serial link modules, with high expectations - we just wanted something that *worked*! But nothing ever did exactly what we wanted: there was always some configuration modes to wade through, and using one of those other modules to reprogram an Arduino is impossible.

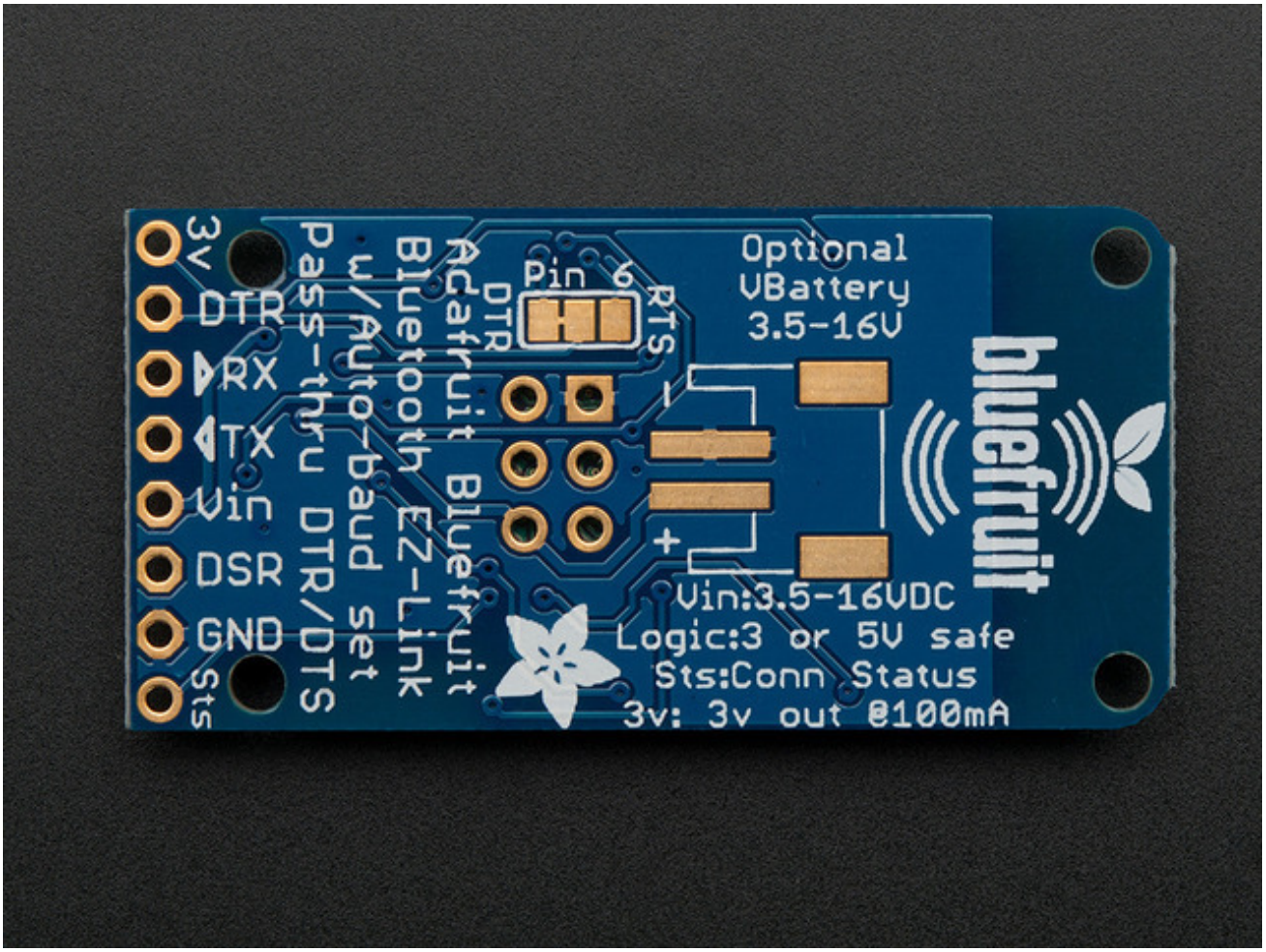
So we did what we always do, we went in and engineered something better. Something that *works!*



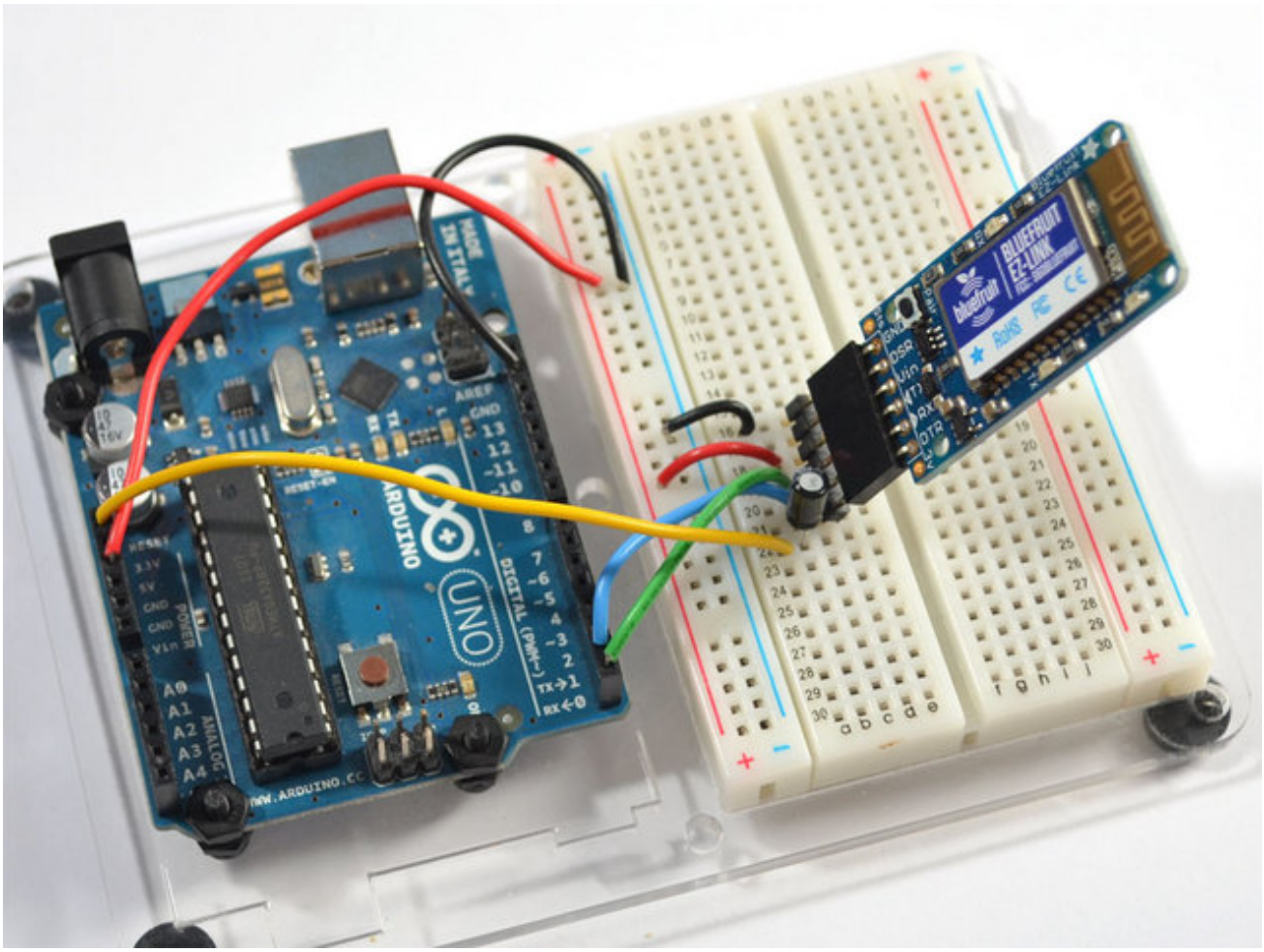
The Bluefruit EZ-Link is a regular 'SPP' serial link client device, that can pair with any computer or tablet and appear as a serial/COM port (except iOS as iOS does not permit SPP pairing). But here is where it gets exciting: unlike any other BT module, the EZ-Link can **automatically detect and change the serial baud rate**. That means if you open up the COM port on your computer at 9600 baud, the output is 9600, 57600? 57600. Yep even 2400. All the most common baud rates are supported: 2400, 4800, 9600, 19200, 38400, 57600, 115200 and 230400. You never have to configure or customize the module by hand - it all happens completely automatically inside the RF module.

Now if we stopped there, you'd probably think "wow that is pretty nice" but we didn't stop there! The EZ-Link has another impressive feature: **the DTR/RTS/DSR flow control pins are automatically synced to the computer serial port**. What this means that if the computer sets the hardware flow control DTR or RTS pins the pins on the bluetooth module will follow. If the DSR input pin is brought high or low on the EZ-Link, the computer can detect that as well. Every other Bluetooth SPP device we've ever seen, if they even have the RTS/DTR pins brought out, do not sync back to the computer, instead the flow control is for the module serial buffer itself.

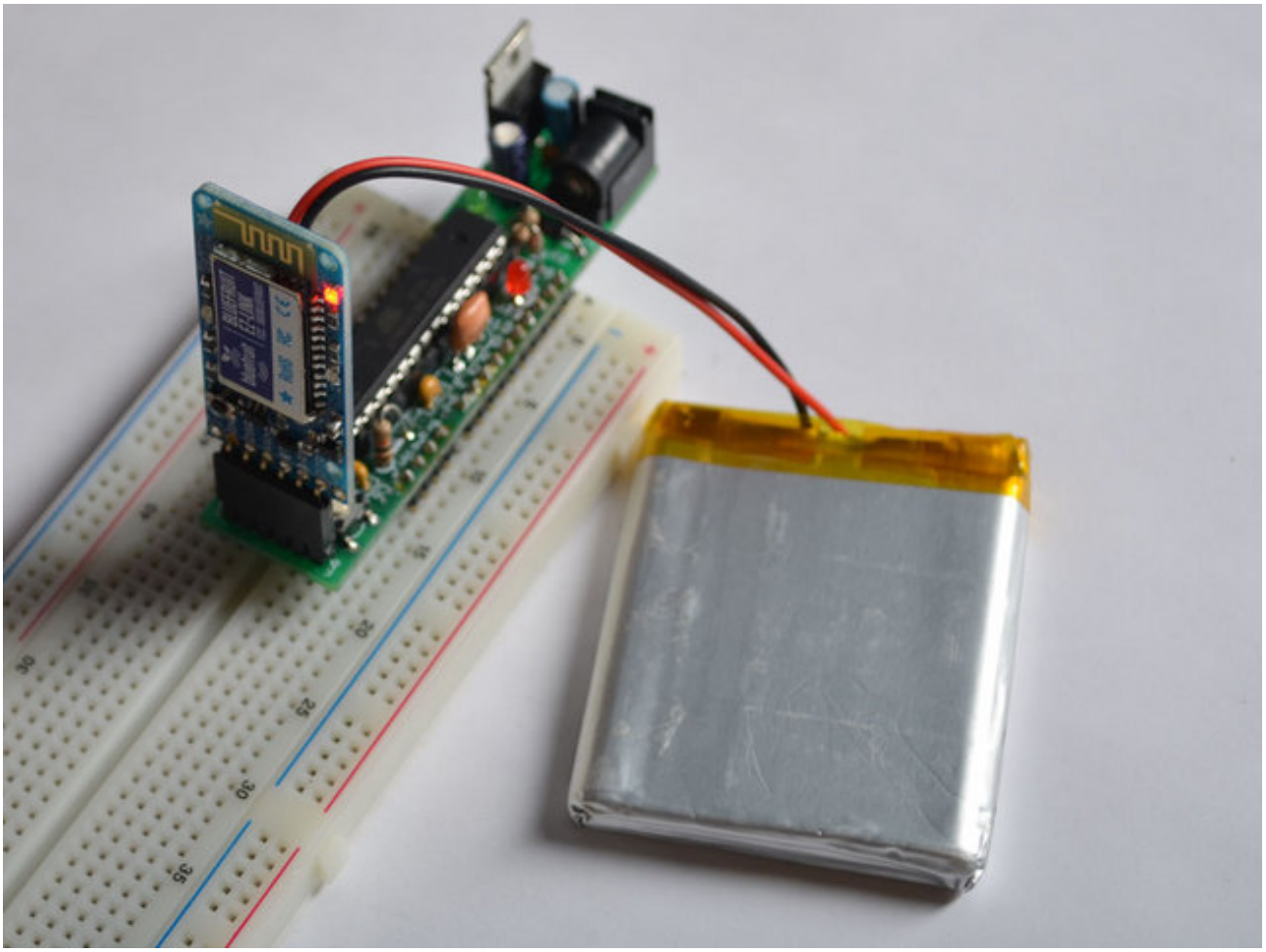
Works great with Mac and Windows computers. Linux is not supported at this time.



Together, this creates something pretty amazing: a Bluetooth module that can change baud rates on the fly and toggle the DTR pin as desired. What we've got here now is a way to program an Arduino (or compatible) from 10 meters away, completely wirelessly, with no extra software, custom hardware, odd firmware hacks or modified firmware. In fact, you can use the Bluefruit as a sort of 'wire free' FTDI-like cable with any device that has an FTDI re-programming port. It works great, and you can use the serial console as well. This package includes the necessary 1uF capacitor between the DTR and reset pin.

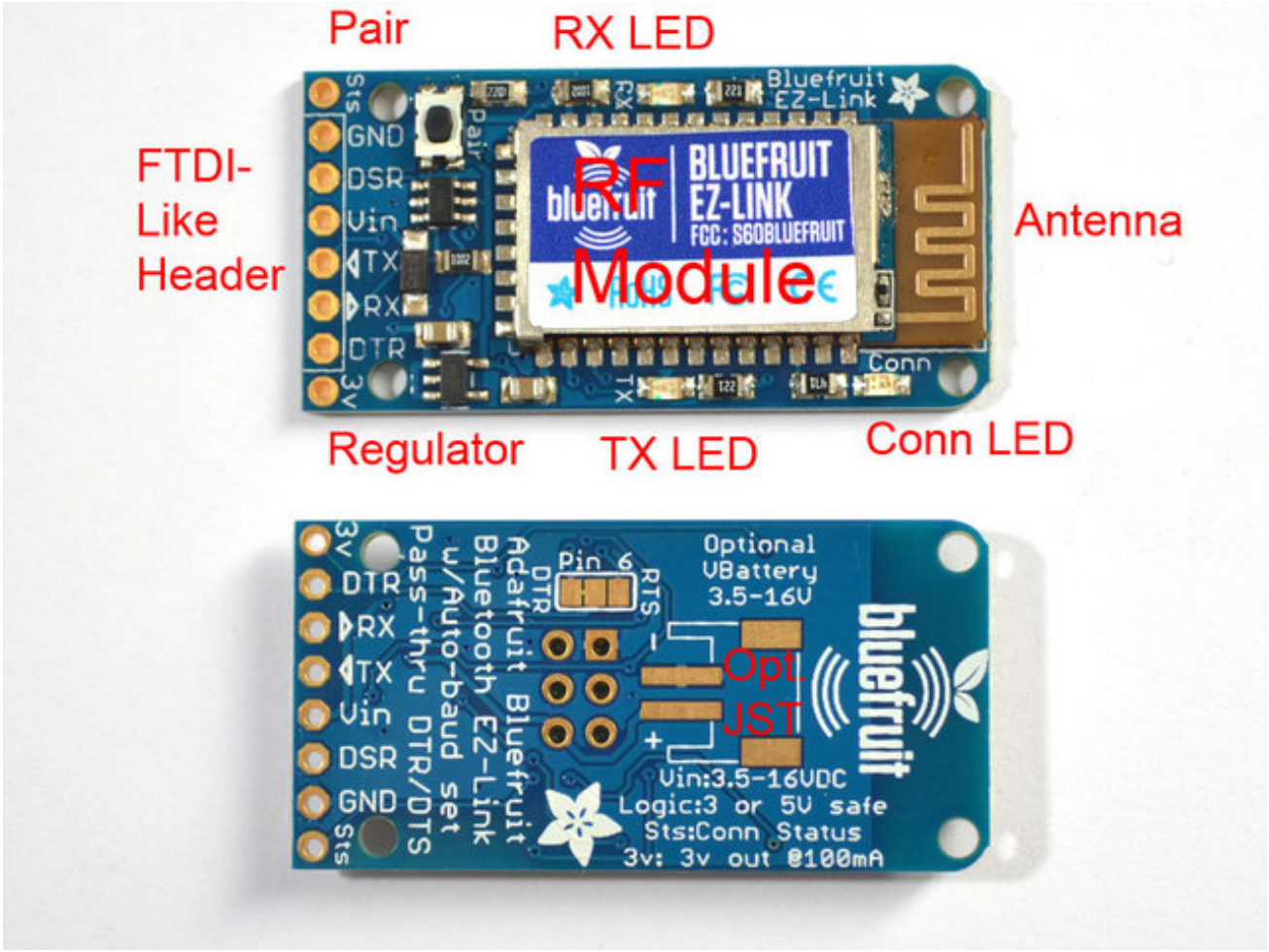


Each order of this EZ-Link + Extras pack includes one ready to go and assembled EZ-Link board. We also include a 6-pin right angle female header and a stick of 6-pin extra-long male header. Chances are you will want to solder the female header in to the center of the board so you can use it like an FTDI-cable but we left you the option of wires, or other kind of header. We also toss in a JST connector. The JST connector can be soldered on the back (optional) to connect a Lipoly battery for portable projects. If the JST is used, the battery can also power the microcontroller that the EZ-Link is plugged into.



Tour

Bluefruit EZ-Link is the ultimate bluetooth module! Lets take a tour of all the great stuff we packed onto this device.



RF Module

In the center is the bluetooth radio module. This module does all the heavy lifting of creating the RF connection and sending data back and forth. It's a custom module made for Adafruit and is FCC and CE certified.

LEDs

There are three LEDs, one red and two blue.

The red Connection LED is used to indicate the state of the bluetooth connection/link. If it is slow-blinking, there is no pairing for the module. If it is fast blinking, it's paired to a computer and the wireless connection is 'live' - the computer has opened a UART connection and is sending/receiving data.

The two blue LEDs are for monitoring the data received & sent. When data is sent from the

computer to the module, the **TX** LED will blink. When data is sent from the module to the paired computer, the **RX** LED will blink. You can use this to debug your wiring and connection.

Pair Button

The pair button is used to reset the computer pairing for Bluetooth link. If you want to re-pair the module, press the button for 5 seconds, you'll see the red LED fast-blink to indicate its ready for pairing! You can now use your computer to Bluetooth-scan for the EZ-Link device.

Level shifting and Regulator Circuitry

There is an on-board regulator that can take 3-16VDC and convert it to 3V to power the RF module. The regulator is also reverse-polarity protected.
Onboard level-shifting circuitry converts any input data signal to be 3V safe, so you can use a 5V microcontroller.

Signal I/O (FTDI-Like Header)

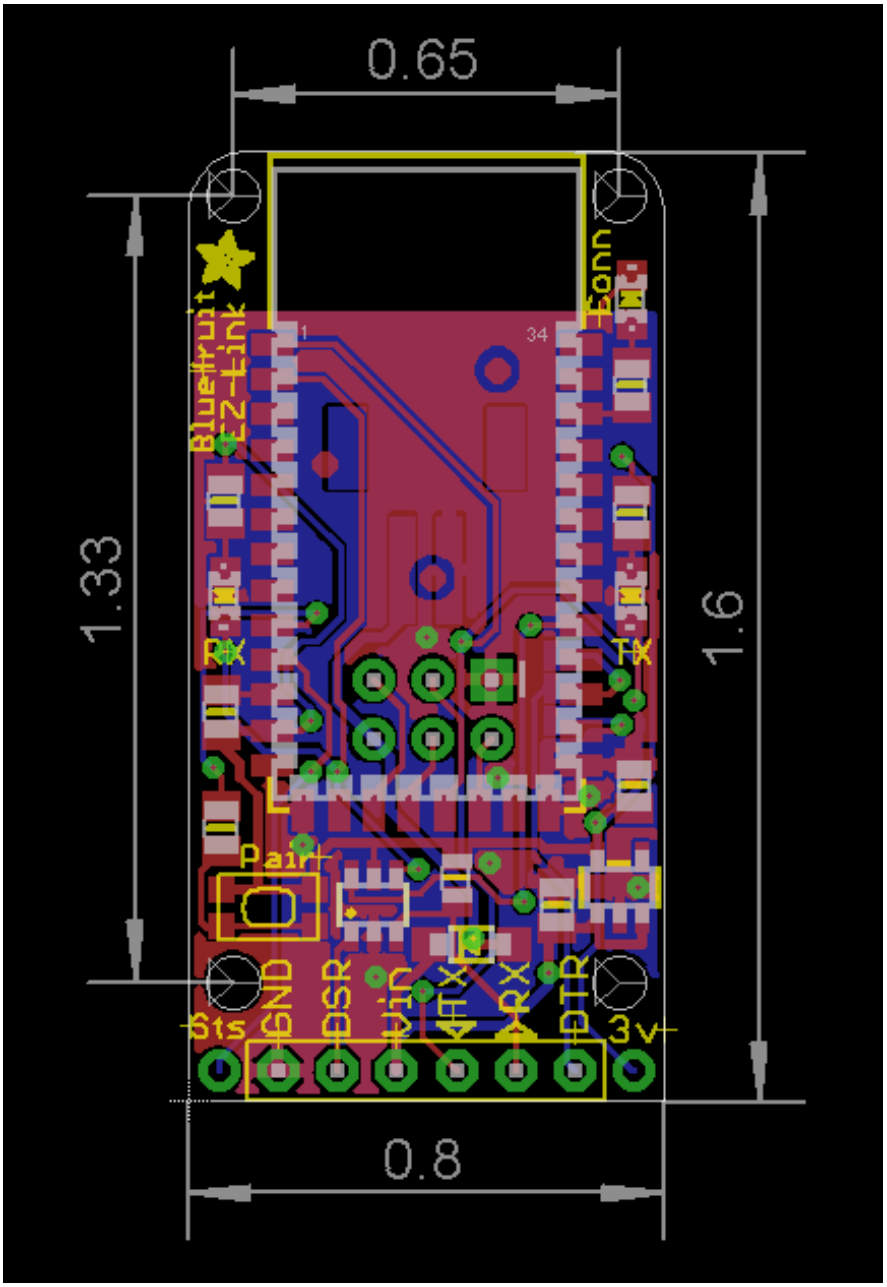
The row of 8 0.1" spaced holes at the bottom are used for signal rx/tx and power. For more details, see the **Pinout** page

Optional JST for Battery

On the back is an optional JST connection that can be soldered with a JST PH-2 socket for powering the module with a battery such as one of Lithium Polymer cells (http://adafru.it/cFB). When a battery is plugged in, the **Vin** breakout pin will act as a power output and can be used to power your microcontroller.

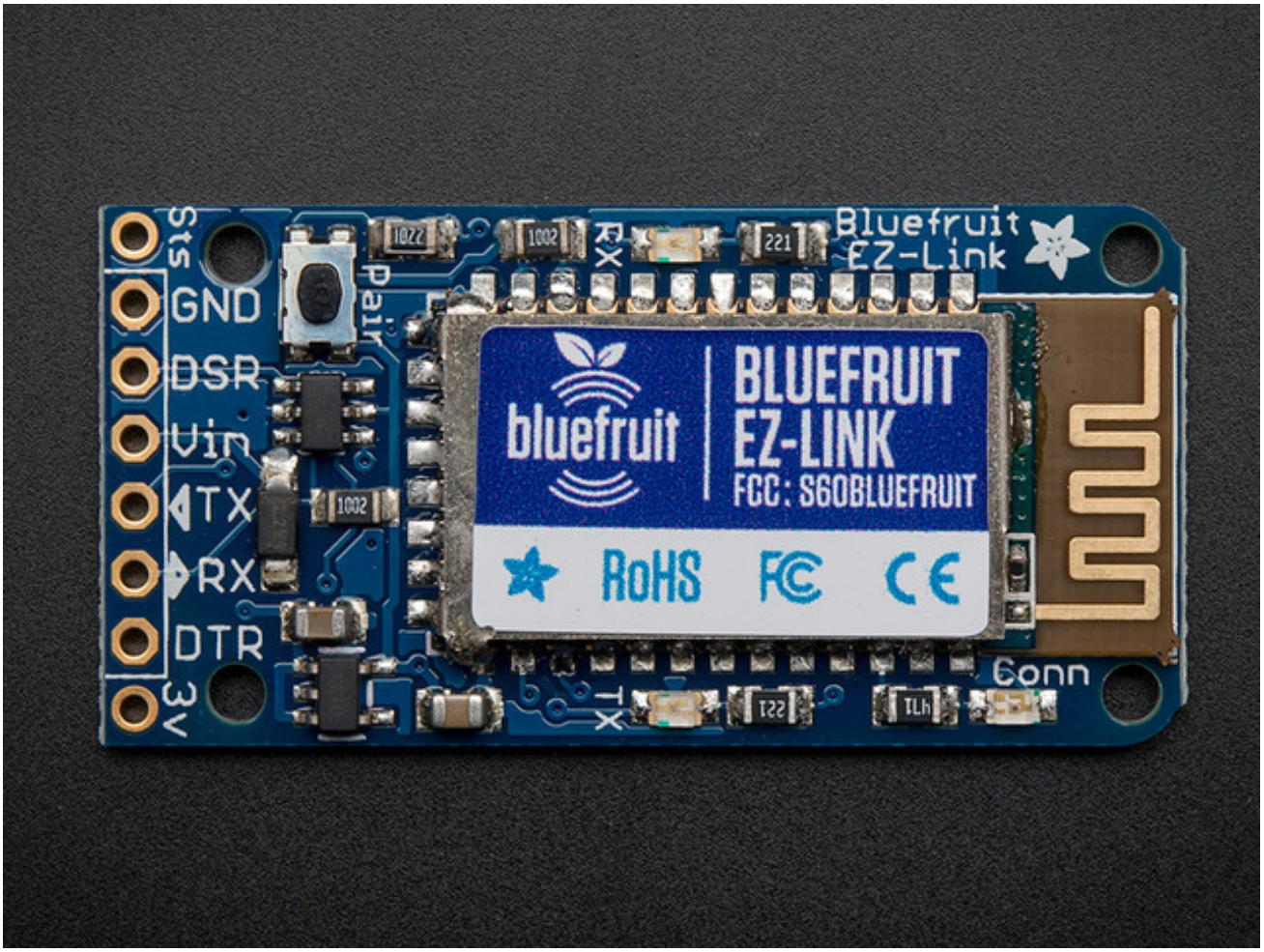
Mounting Holes

Four 0.09" can be used to attach the EZ-Link, see below!


0.65
1.33
1.6
0.8
Bluefruit
EZ-Link
Conn
Pair
RX
TX
Sts
GND
BSR
Vin
TX
RX
DTR
3V

Pinout

Bluefruit EZ-Link is a serial link with an "FTDI"-like header so you can use it with any product that has an FTDI header already. You can also plug it into a breadboard or wire it directly depending on your needs. However, there are a few minor changes that make it worth spending a few minutes understanding what each pin is for.



Main "FTDI" 6-pin header (middle 6 pins)

1. **GND** - this it the common ground pin, used for power and signal reference ground. Make sure in your system all grounds between microcontroller, battery/power and the EZ-Link are connected
2. **DSR** - this is the "Data Signal Ready" hardware flow control pin that is transmitted from the Microcontroller, through the EZ-Link to the paired computer. If you want to send a signal outside of the UART back to the computer, this pin can do it. The computer can then read the terminal-status lines. This isn't a high-speed line, expect up to 100ms delay from when the pin toggles to when the signal is read on the computer. If not used, tie to ground.

On FTDI cables this is often labeled CTS, for a different signal line. Bluetooth does not have support for sending this signal back to the computer so we substituted DSR instead.

3. **Vin** - This is the power pin *into* the Bluefruit module, it passes through a 3.3V regulator so this voltage can be from 3.3V to 16VDC and is reverse-polarity protected. If a battery is connected via the optional JST connector on the back, this pin becomes an *output* so you can power your microcontroller/project via the battery.
   If you plan to use the optional JST connector on the back, don't provide power to this pin from the microcontroller
4. **TX** - This is the serial UART output pin that is transmitted from the paired computer, wirelessly to the Bluefruit and out this pin. It is 3V logic level so you can use it with 3 or 5V microcontrollers.
5. **RX**- This is the serial UART input pin that is transmitted from the microcontroller into the EZ-Link then sent wirelessly to the computer. It is level shifted so you can use 3 or 5V logic microcontrollers/signal
6. **DTR** - This is the "Data Terminal Ready" hardware flow control pin that is sent from the paired computer to the microcontroller. Often this is used for telling the client that the server is ready to send data. However, this pin tends to be used for reprogramming Arduino microcontrollers by connecting it to the reset line. The Arduino IDE toggles this pin up and down to reset the Arduino to start the bootloader before uploading code. This pin is 3V output so you can use it with 3 or 5V

Extra Pins

On either side of the 'classic FTDI' header, we added two additional pins you may find useful

- **3V** - this is the 3.3V output from the onboard voltage regulator. If you power the module from the optional JST jack or Vin pin, and that voltage is 3.5V or higher, you can grab a nice regulated 3.3V up to 100mA from this pin
- **Sts -** this is the Connection Status pin, when it is 0V, the UART is not 'open' on the other side. When it is 3V, that means the bluetooth UART is 'open' on the computer. Normally the DTR pin would be used for this kind of status but since the DTR pin is used for resetting Arduino's, we have a different pin for this purpose.

Pairing

Before you start this step you should make sure your Bluefruit EZ-Link is powered up. You can do this by either applying 3-16VDC to the **Vin** & **GND** breakout pins or by soldering in the JST connector on the back and plugging in a power supply there.

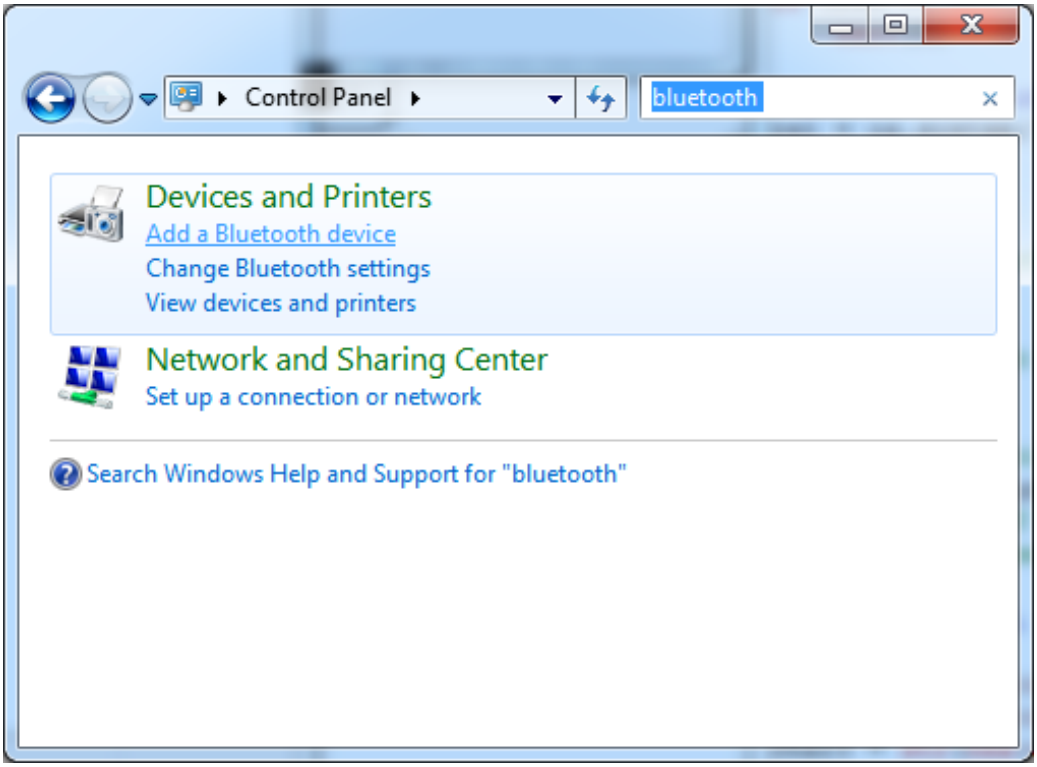
Make sure the power connection is solid & any wires or connectors properly soldered in. **Do not try to 'press fit' or 'twist' wires onto the power pins without soldering, as it will not work well and cause a lot of difficulty with pairing and usage.**

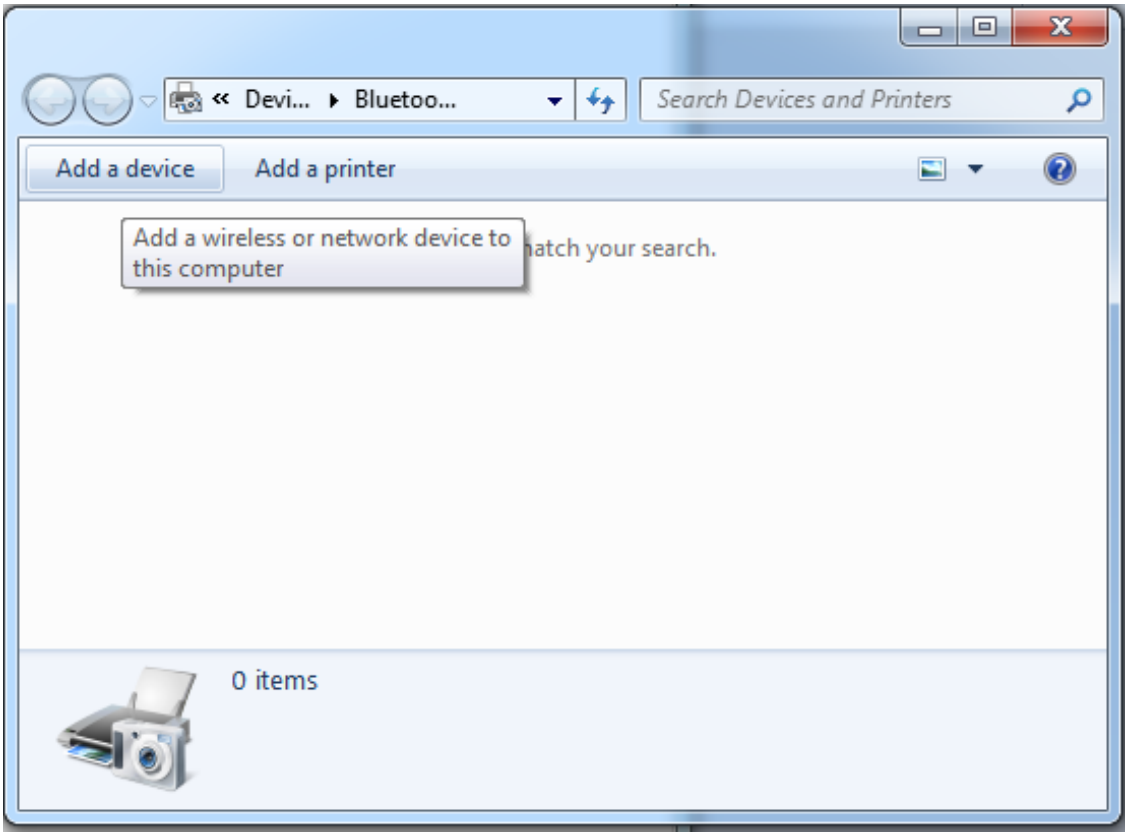
You'll also need to make sure that your computer that you'll be pairing with has Bluetooth v2.1 or great hardware installed. Nearly every Mac computer/laptop, Windows laptop and Android tablet has this already. If not, a USB Bluetooth module such as this one is ideal. (http://adafru.it/1327)Watch out for "cheap $5" bluetooth USB modules, they are often v2.0 only, and won't work with Bluefruit due to the advanced firmware requirements.

It is not possible to pair the EZ-Link with iOS devices such as iPhones or iPads - Apple does not permit 'SSP' pairing! Check out the EZ-Key for a iOS compatible bluetooth keyboard module you may be able to use. (http://adafru.it/1535)

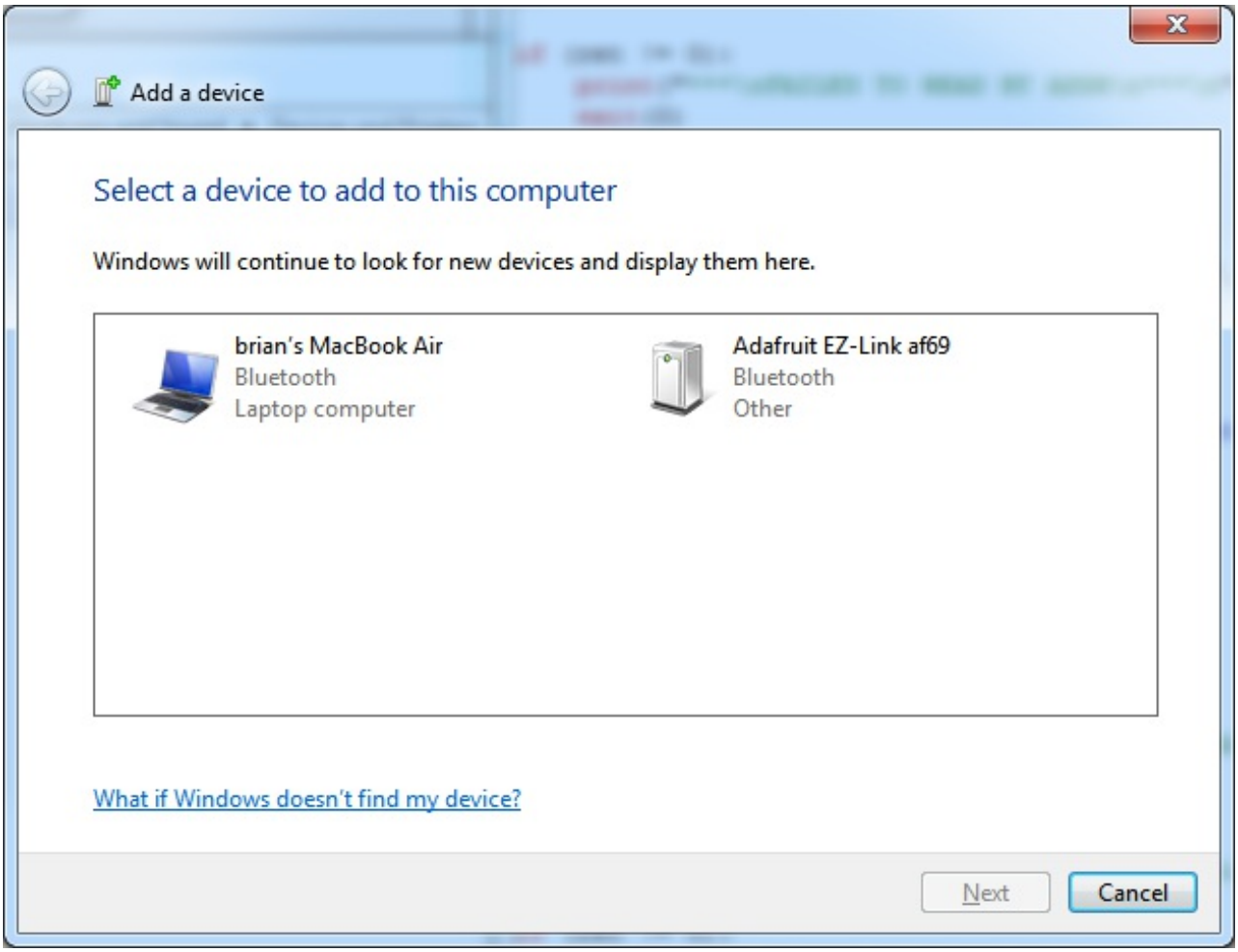
Windows

Open up the Bluetooth devices panel, on our computer we just clicked on the little Bluetooth icon in the task bar. You can also open the Control Panel and search for Bluetooth and then click Add Devices

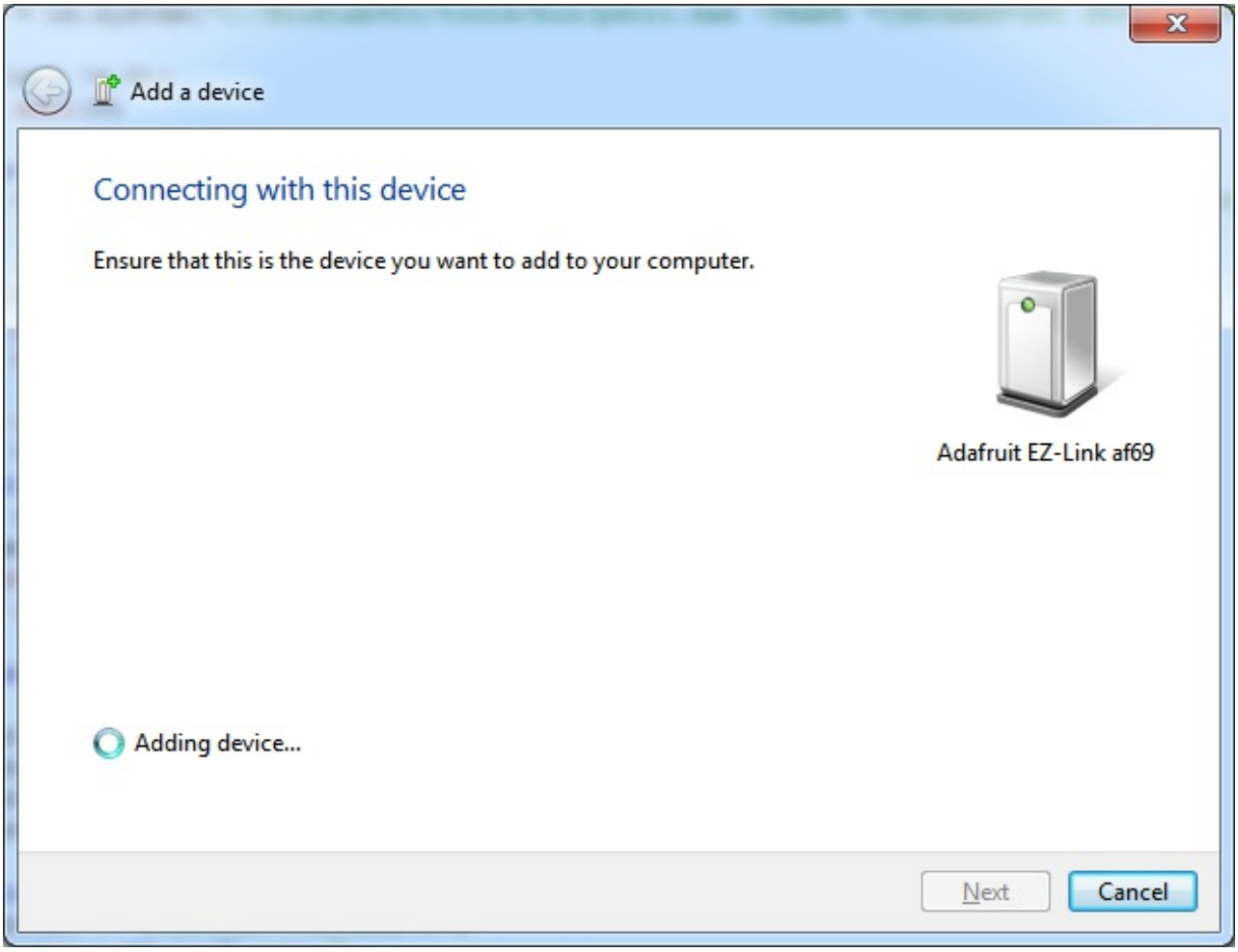


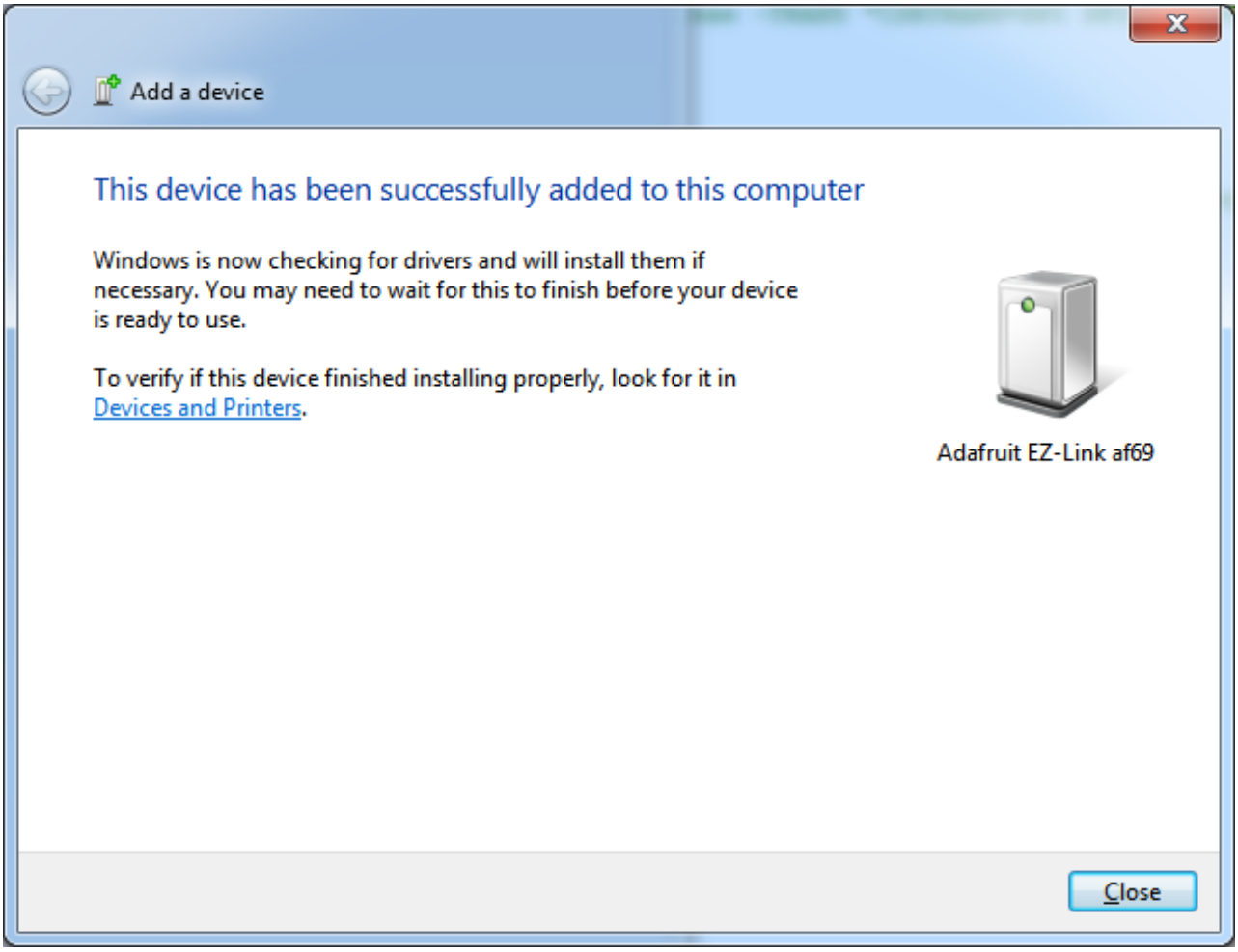


Either way, you'll get to a panel like this. Make sure the EZ-Link is plugged in and you see a once-every-two-seconds slow blink on the red LED. Within 30 seconds you should also see the **Adafruit EZ-Link xxxx** device appear. Now you can pair to this device.

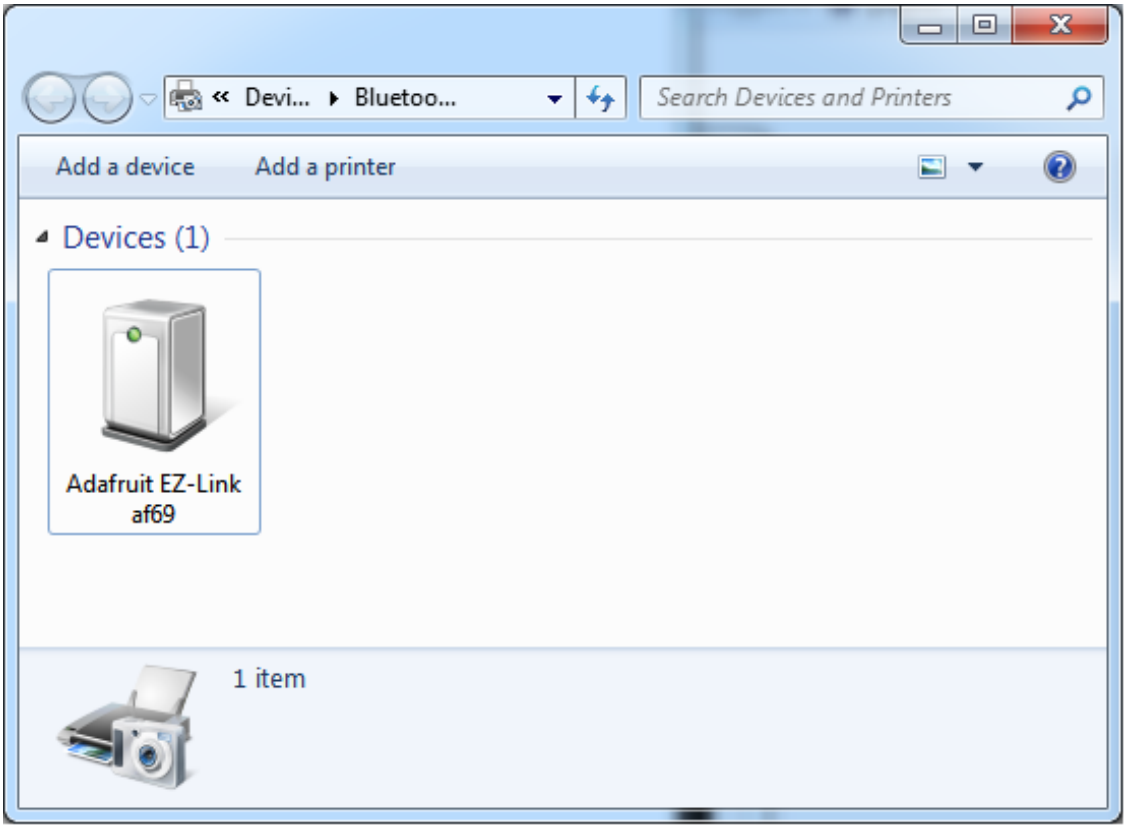


Click on it and select **Next** to add it


Add a device
Connecting with this device
Ensure that this is the device you want to add to your computer.
Adafruit EZ-Link af69
Adding device...
Next
Cancel

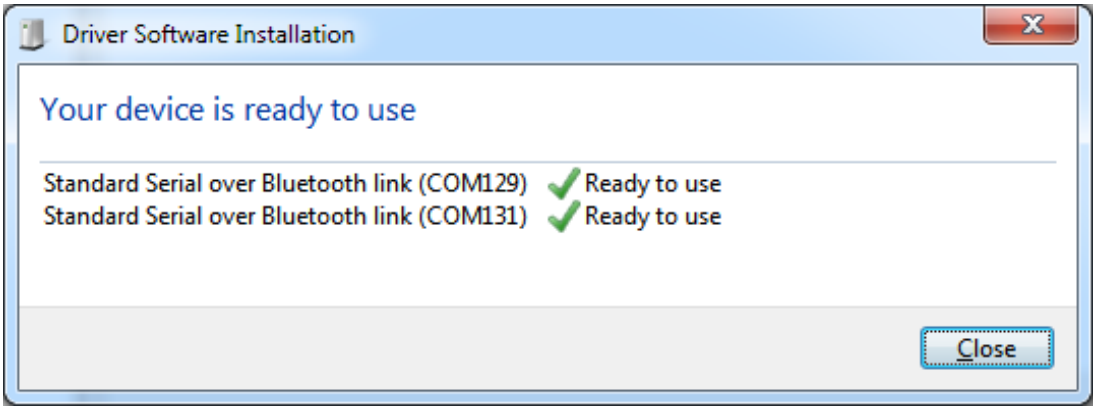


If you get asked for a pairing code, the code is **1234**
That's it, you're now paired!

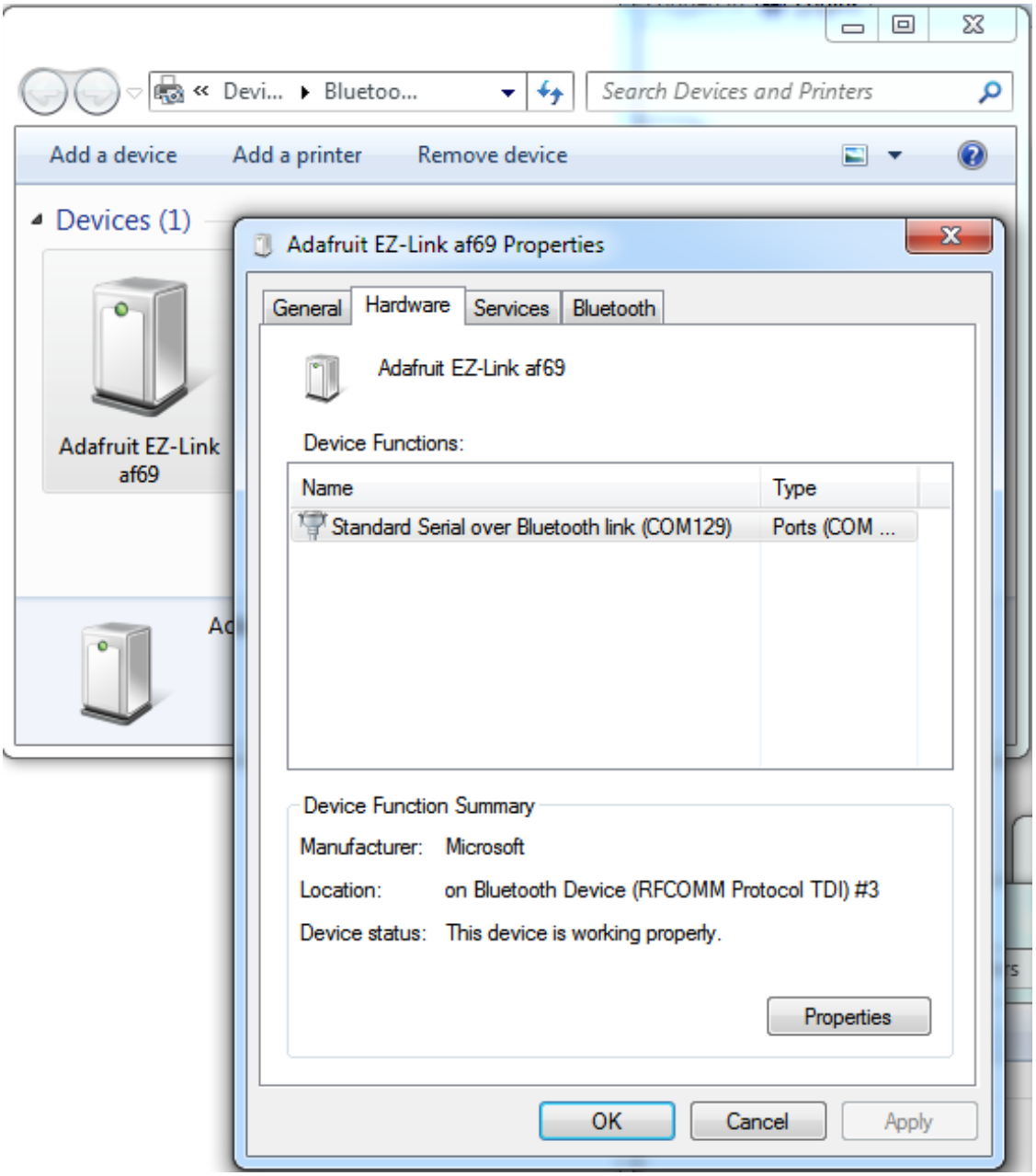


Next we will make sure we know the COM port to use. This is actually a little confusing because for some reason Windows creates *two* COM ports, but we only use one of them.

You may see the COM ports created by the Bluetooth driver, it'll look like this



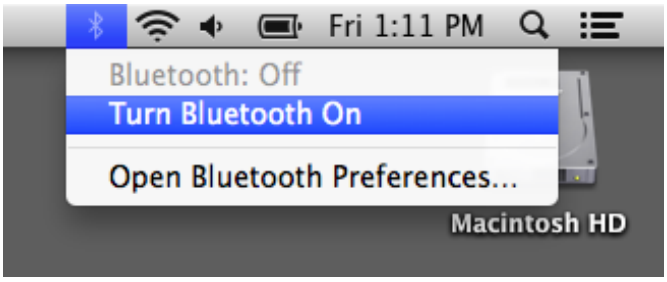
To figure out which COM port to use, go back to the Bluetooth device panel and right-click on the Adafruit EZ-Link and select **Properties** then the **Hardware** panel.



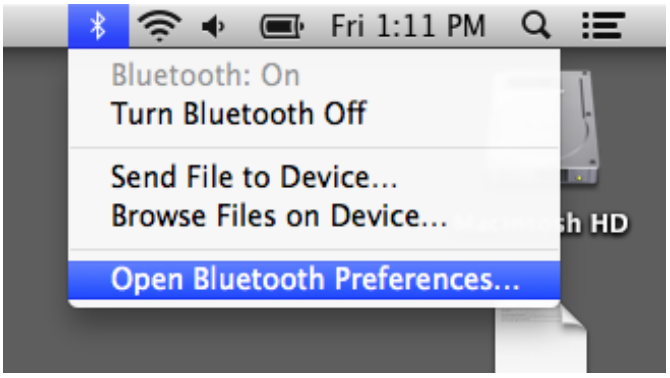
You'll see that only one COM port is mentioned here, in this case its **COM129**. That's the port you'll use when sending data to the EZ-Link

Mac OS X

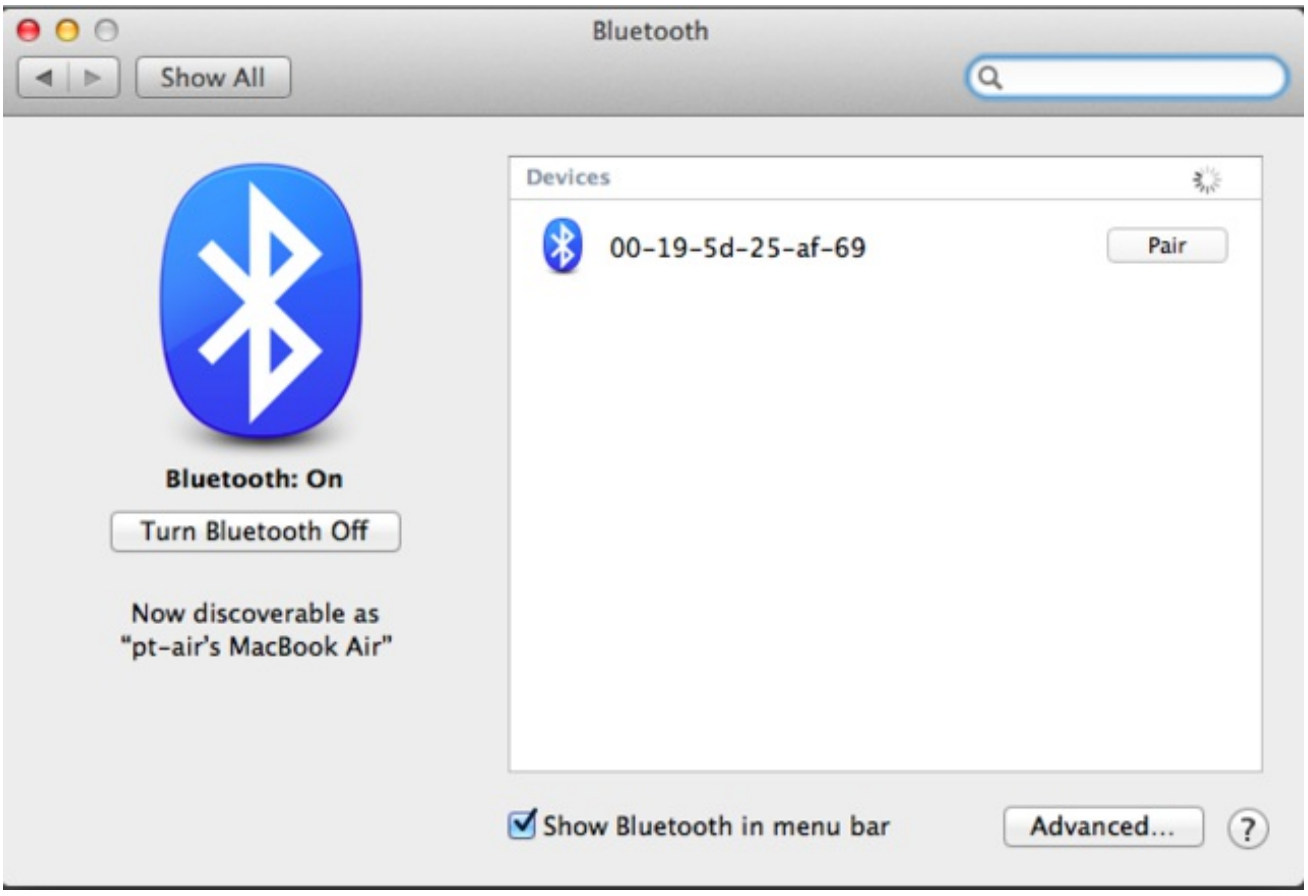
Pairing on a Mac is pretty easy. Start by turning on Bluetooth

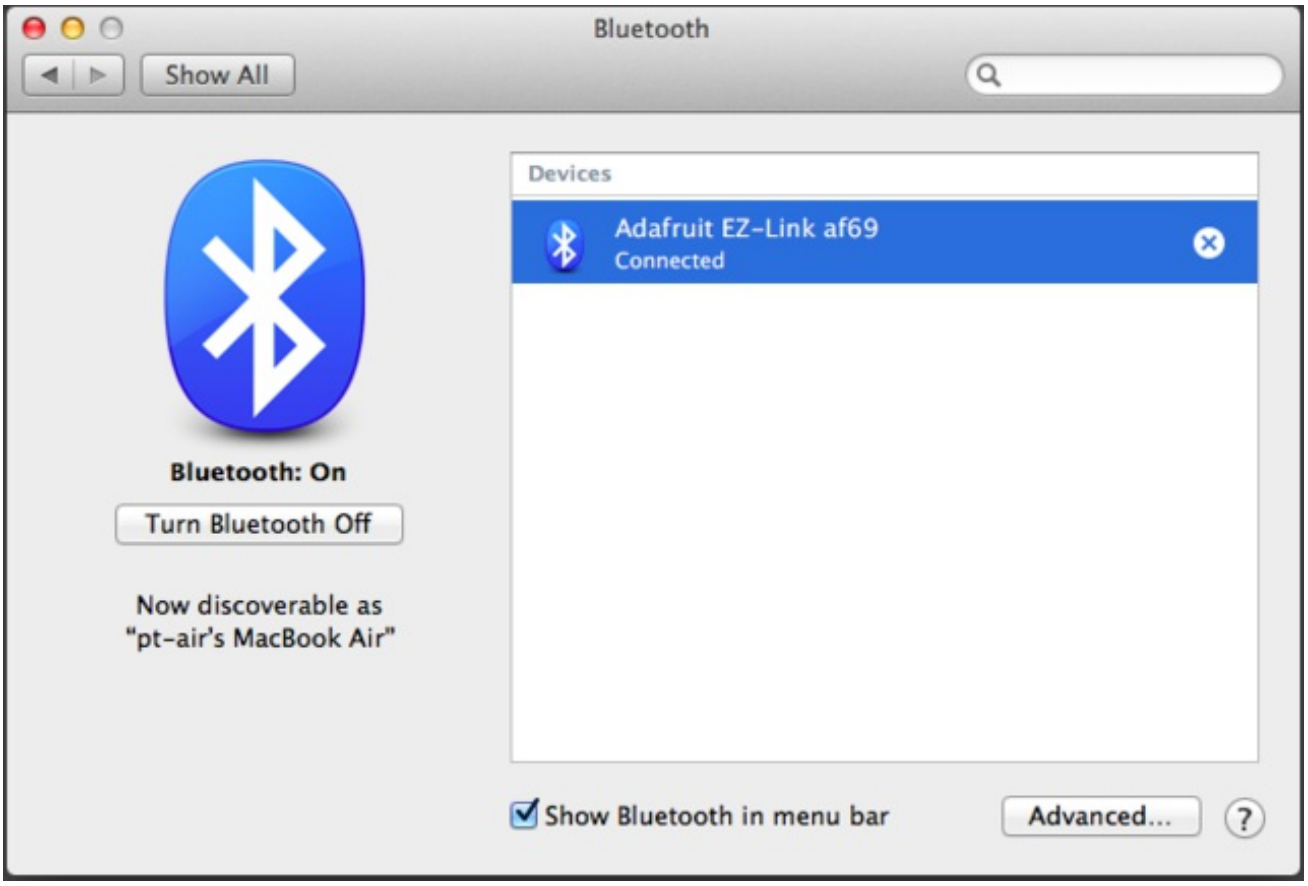


Then go back to the BT menu and open the **Preferences...**



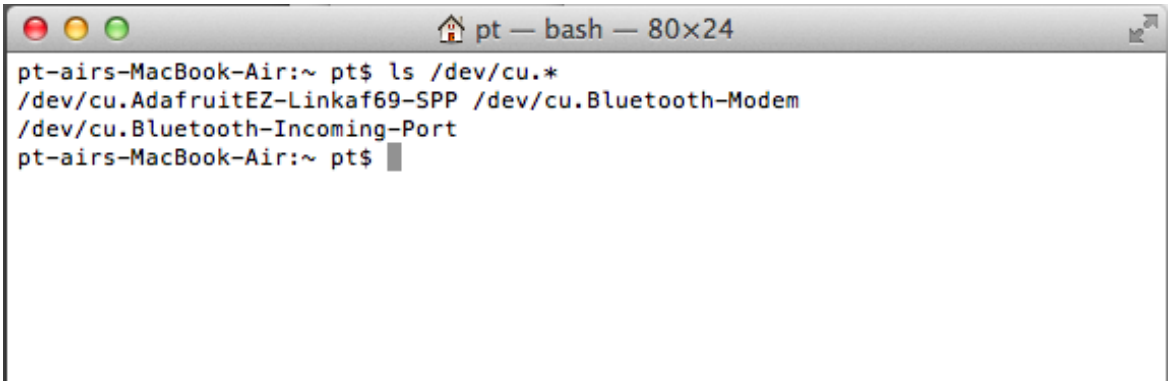
It will immediately start scanning for the device, you'll see the BT address first, then eventually it will turn into the EZ-Link name



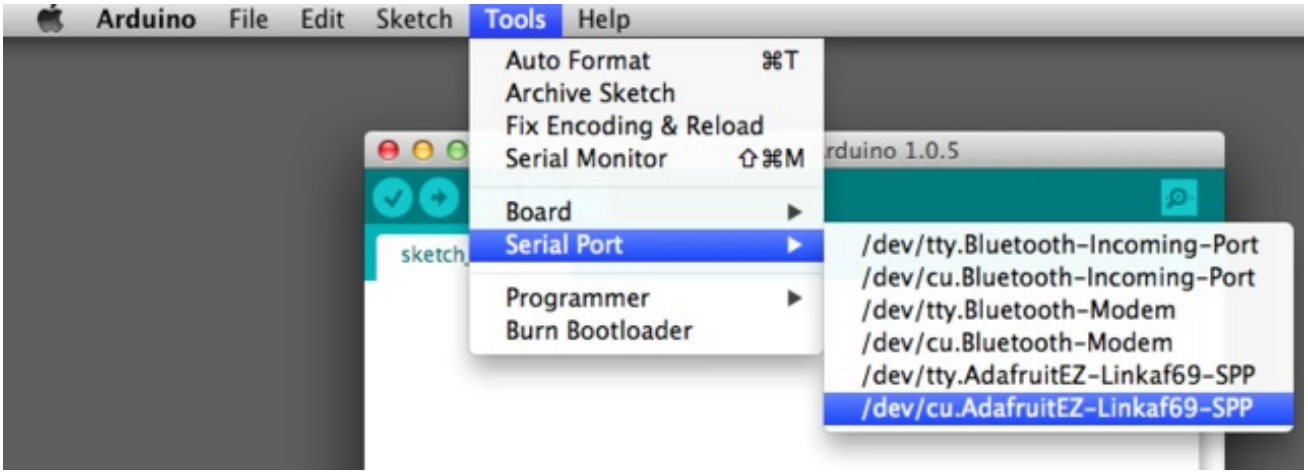


That's it! You're now paired to it. If you are asked for a code, the code is **1234**

Next you can figure out the name of the device by opening up a **Terminal** and typing in **ls /dev/cu.*** and looking for something like **/dev/cu.AdafruitEZLinkxxxx-SPP**



If you're using the Arduino IDE, it will show up under that name, use the **cu.** version not the **tty.** version


Arduino
File
Edit
Sketch
Tools
Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⇧⌘M
Board
Serial Port
Programmer
Burn Bootloader
/dev/tty.Bluetooth-Incoming-Port
/dev/cu.Bluetooth-Incoming-Port
/dev/tty.Bluetooth-Modem
/dev/cu.Bluetooth-Modem
/dev/tty.AdafruitEZ-Linkaf69-SPP
/dev/cu.AdafruitEZ-Linkaf69-SPP

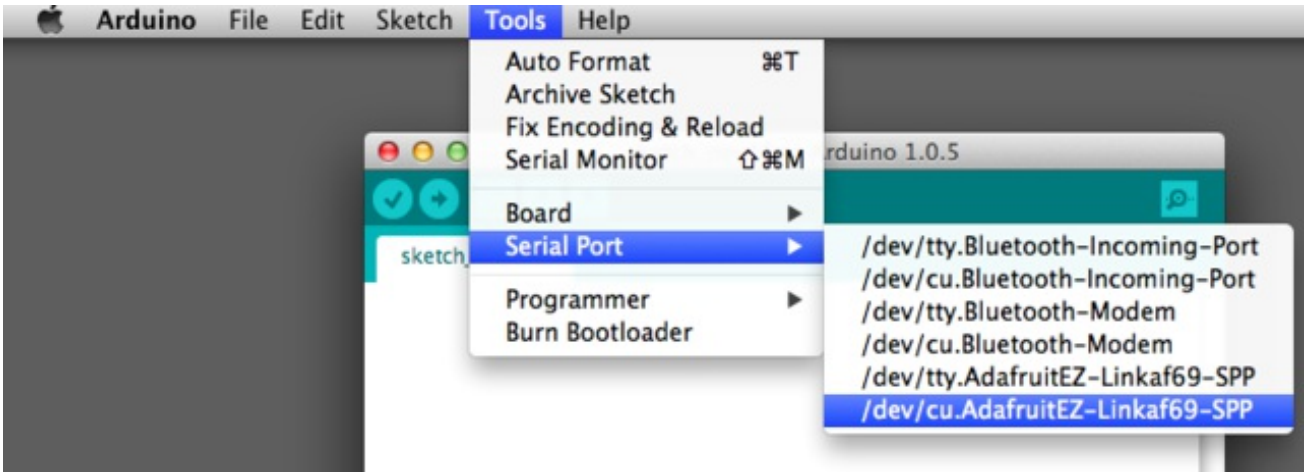
Testing

Its a good idea to test your Bluefruit EZ-Link, that way you can tell if there's something wrong with your microcontroller or wiring - you can at least rule out the EZ-Link from being the problem.

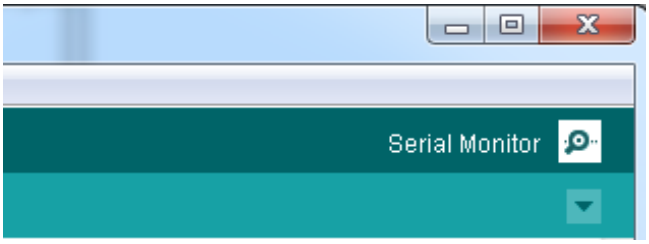
To test, you'll need a terminal program. Since most people have the Arduino IDE installed, I'll use that for the first 'loopback' test

Loop-back Test

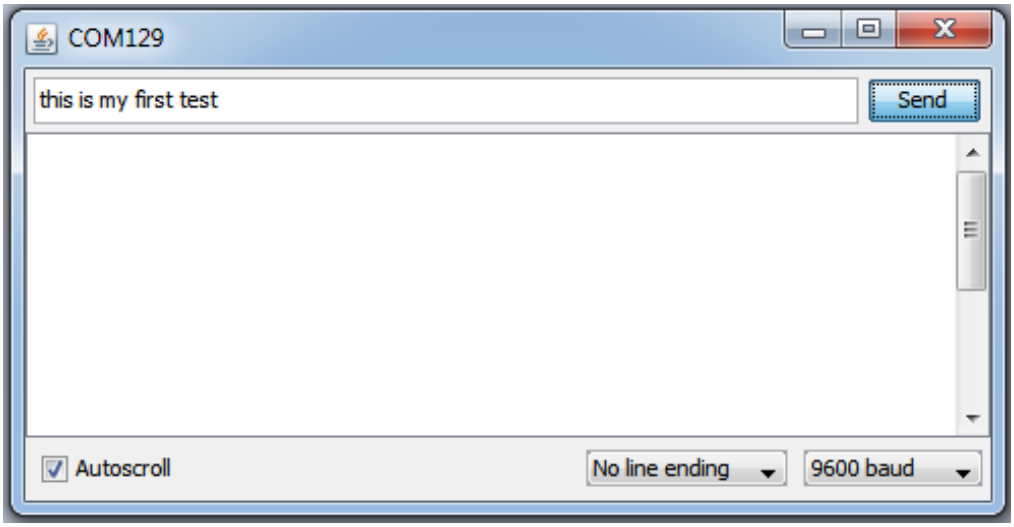
The loopback test is the easiest test and checks the RX/TX pins as well as the wireless link overall!
Start up the Arduino IDE and select the Bluetooth device you found in the Pairing step beforehand



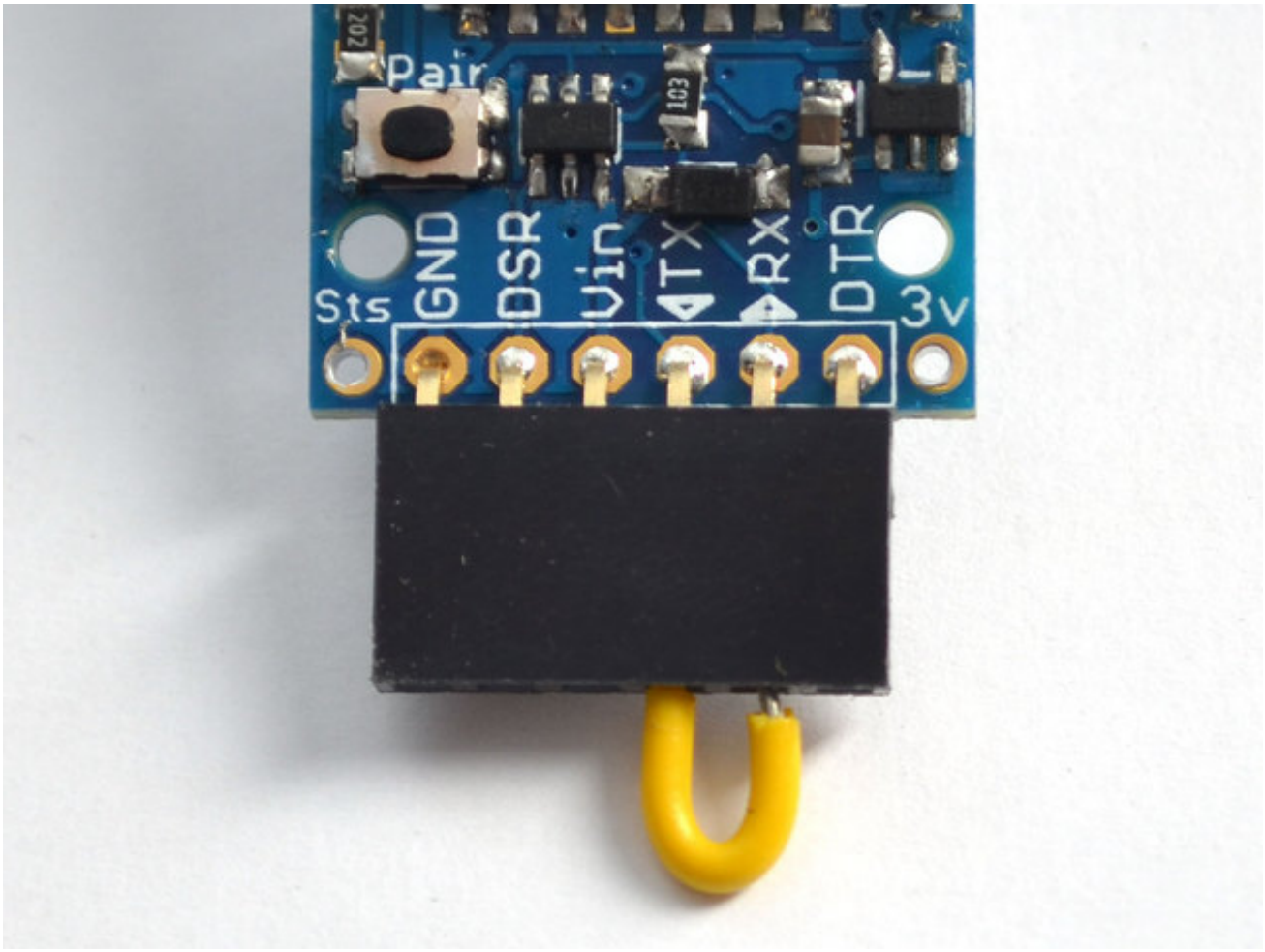
Then open up the Serial monitor. It may take a few seconds to make the link. You should see the red LED on the EZ-Link blink very fast now to indicate a connection is made



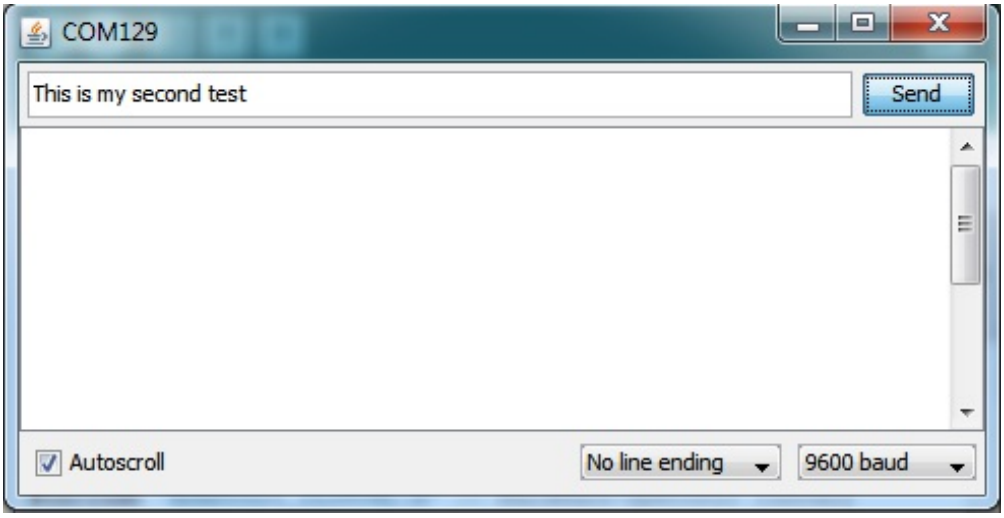
Type some text into the entry box and hit return, you should see nothing appear in the monitor



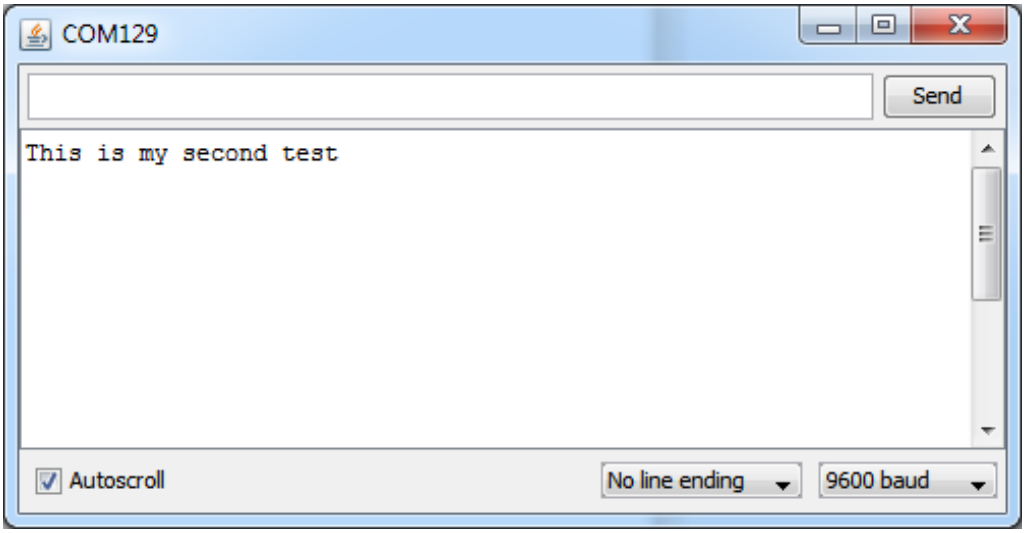
Now connect a wire from the RX to the TX pin on the EZ-Link



Now go back to the Serial Monitor and type in a new message & hit return



This time, you'll see whatever you send is *echoed* back to you - that way you have tested that data can be sent wirelessly to the EZ-Link, through the wire, and then back to your computer. If you watch closely you can also see the blue LEDs flicker

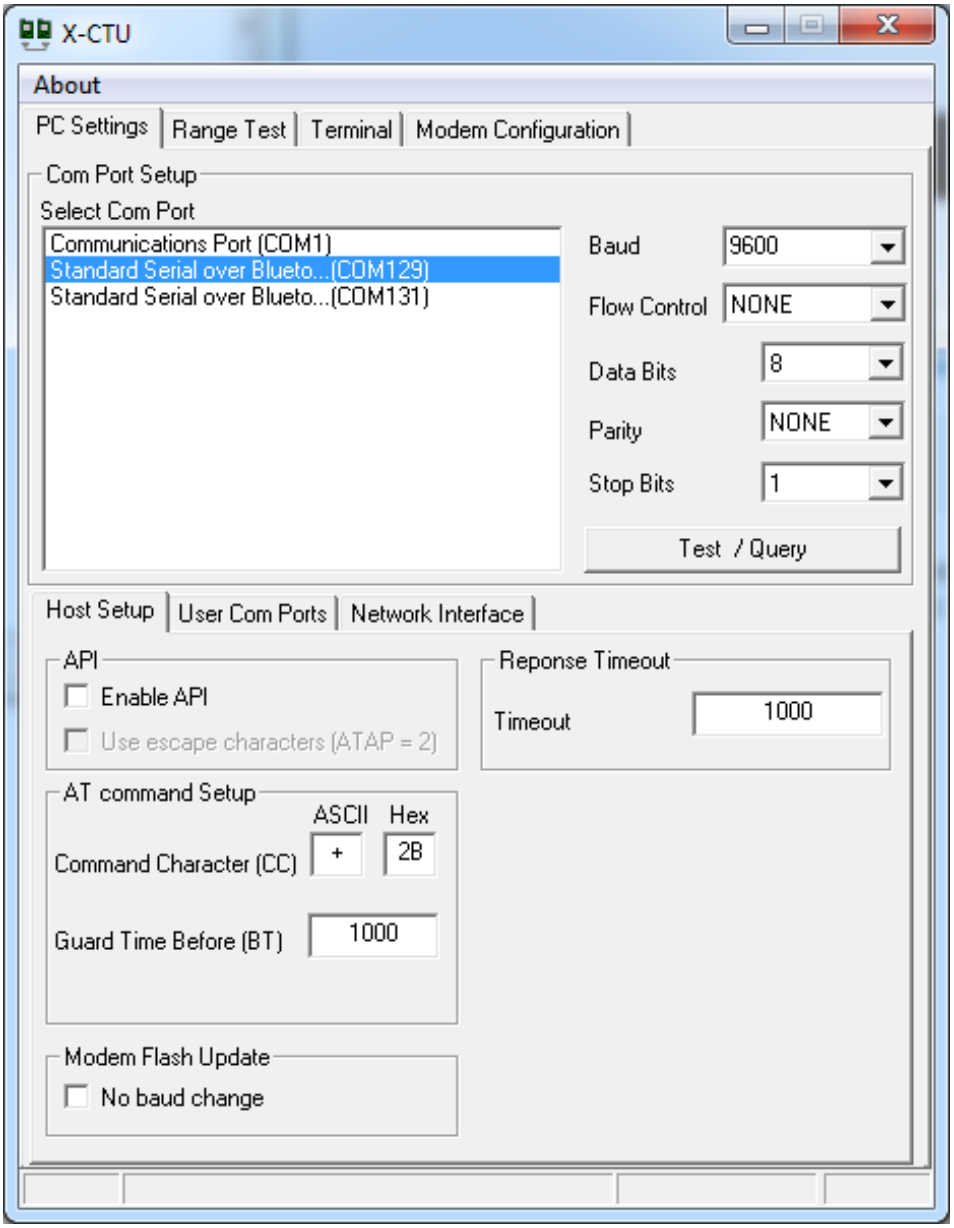


DTR/DSR Test

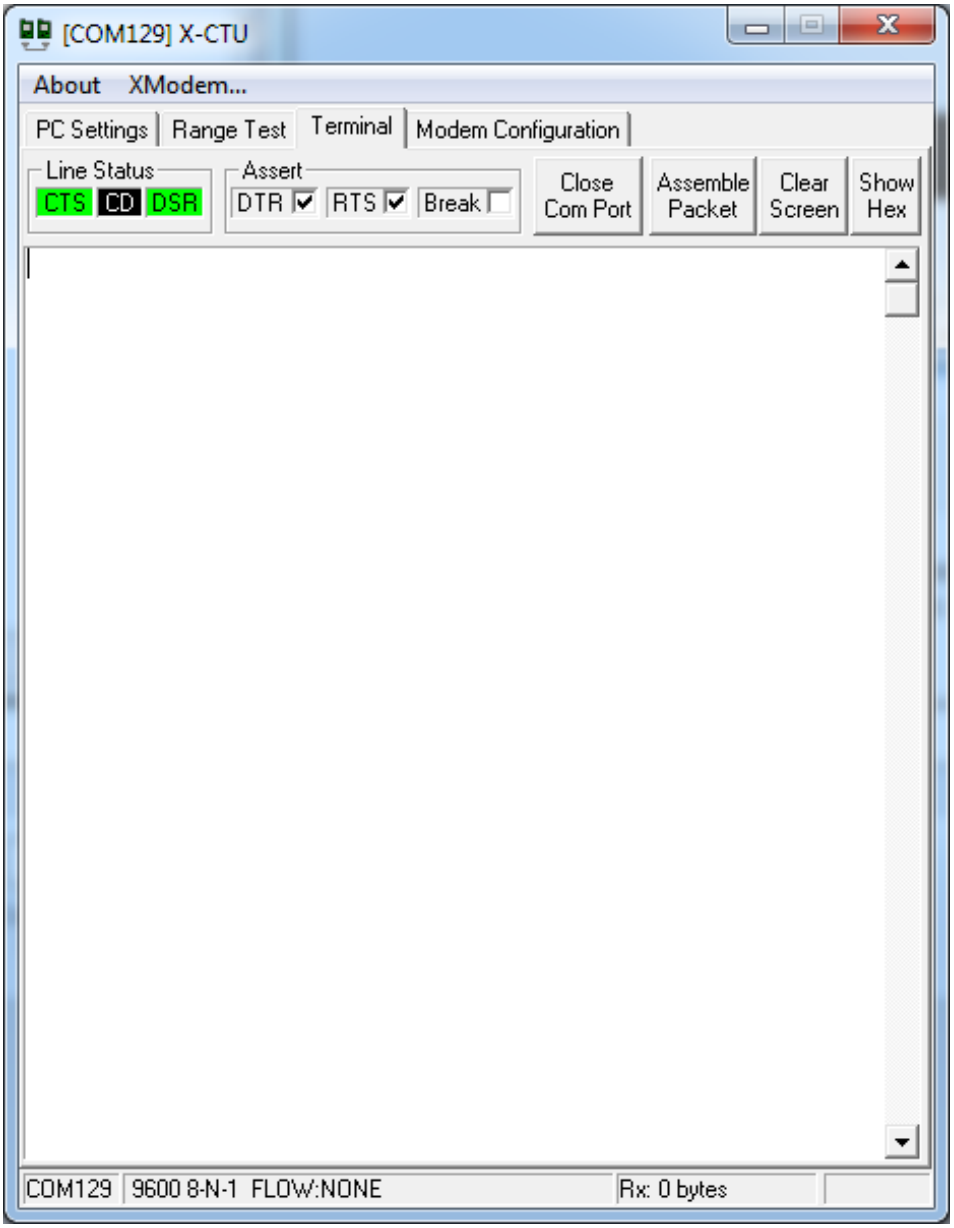
This is a more advanced test, for checking the DSR/DTR pins. For this, we need a terminal program that can control the flow lines. Unfortunately the Arduino IDE does not do this, but Digi's X-CTU program can.

If you're on a Mac, Serial Tools will let you perform the same tests as X-CTU is not available for Mac (http://adafru.it/cVK)

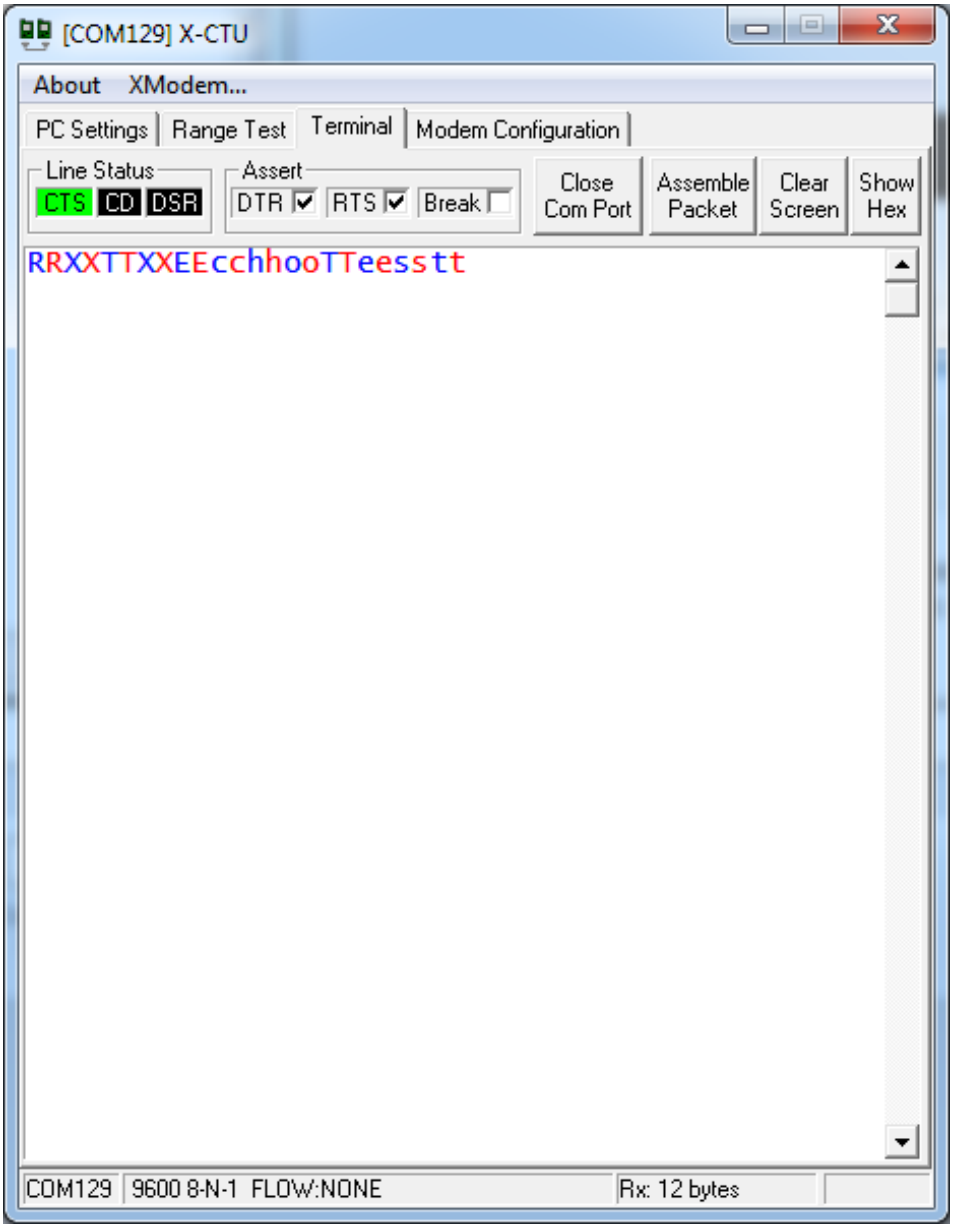
Select the COM port you identified before



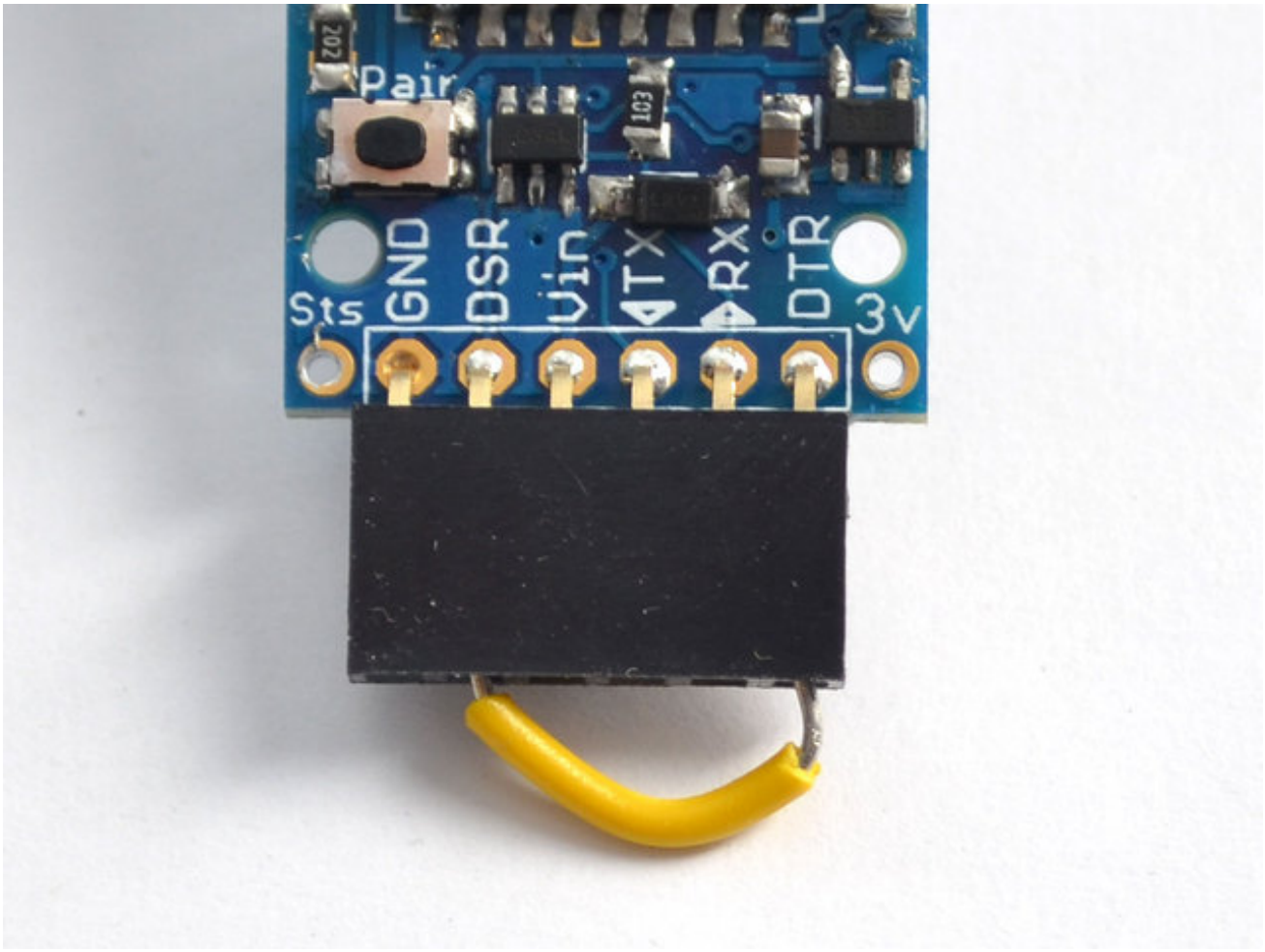
And click on **Terminal** to open up the serial monitor/entry window



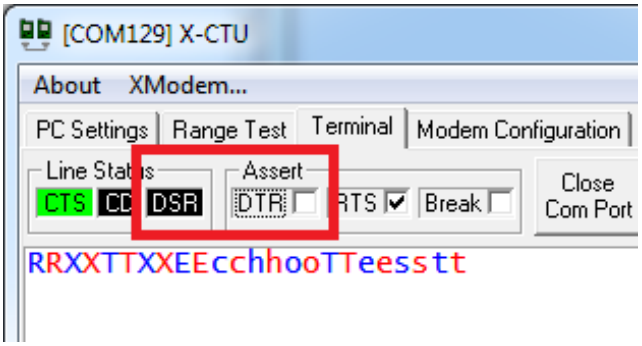
X-CTU has the nice ability to show you data sent and received at once, which makes it easy to do the RX-TX loopback test as above, you'll see each character echoed

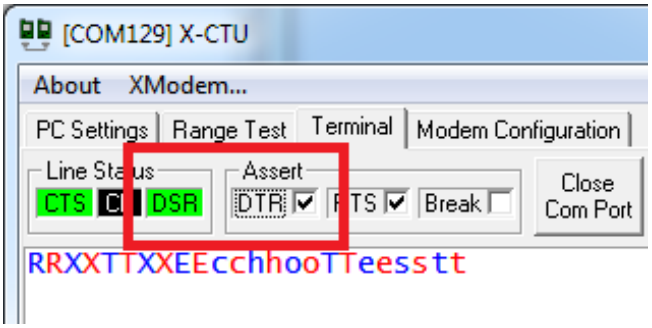


To do the DTR/DSR test, connect a wire from DSR to DTR line below

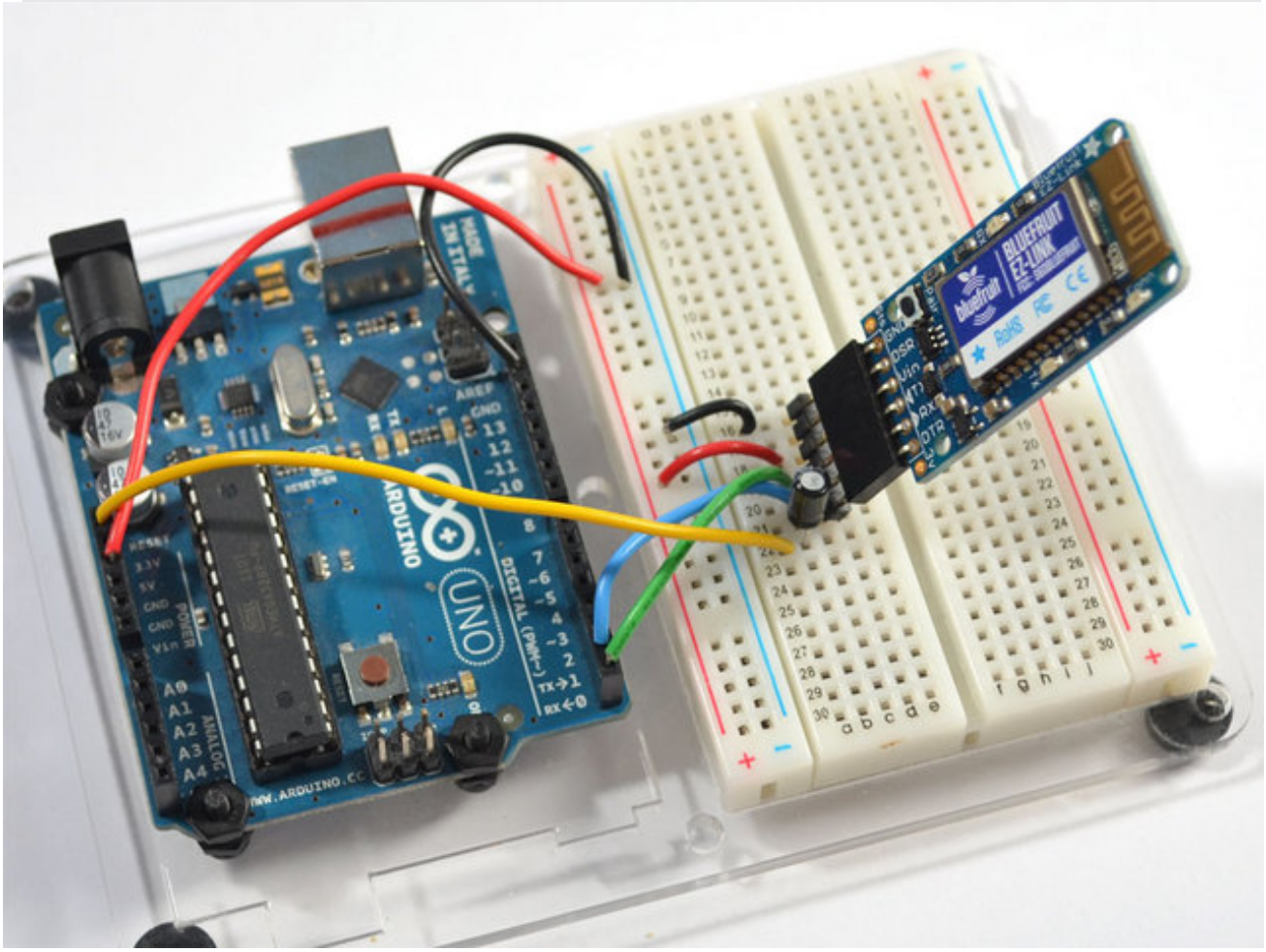


Then click on the **DTR** checkbox, this will toggle the DTR line on the EZ-Link, which will then feed back to the **DSR** line which will toggle the green DSR labeled box to the left. Make sure you see the green box turn on and off when you click the checkbox!




[COM129] X-CTU
About
XModem...
PC Settings
Range Test
Terminal
Modem Configuration
Line Status
CTS
DSR
Assert
DTR
Break
Close
Com Port
RRXXTTXXEEcchhooTTeesstt

Arduino Programming



Unlike any other Bluetooth module, you can use the EZ-Link to program an Arduino. The reason this is possible is because EZ-Link can dynamically change the baud rate and has the ability to send the DTR signal from the Arduino IDE wirelessly through the module. Both are required if you want to be able to quickly change to the 115.2K signal required for upload and toggle the DTR lines.

This techique can be used for Arduino Uno, Duemilanove, Mega and any other ATmega328/168/1280/2560 based Arduinos that use a USB->Serial converter. It won't work with "USB" Arduinos such as the Leonardo/Micro/Flora since they need a direct USB connection

This technique has been tested on Mac OS X and Windows 7, but should work identically on any other OS.

You can use **any** version of the Arduino IDE with this technique, the Bluetooth part is completely transparent to the software

Before you start

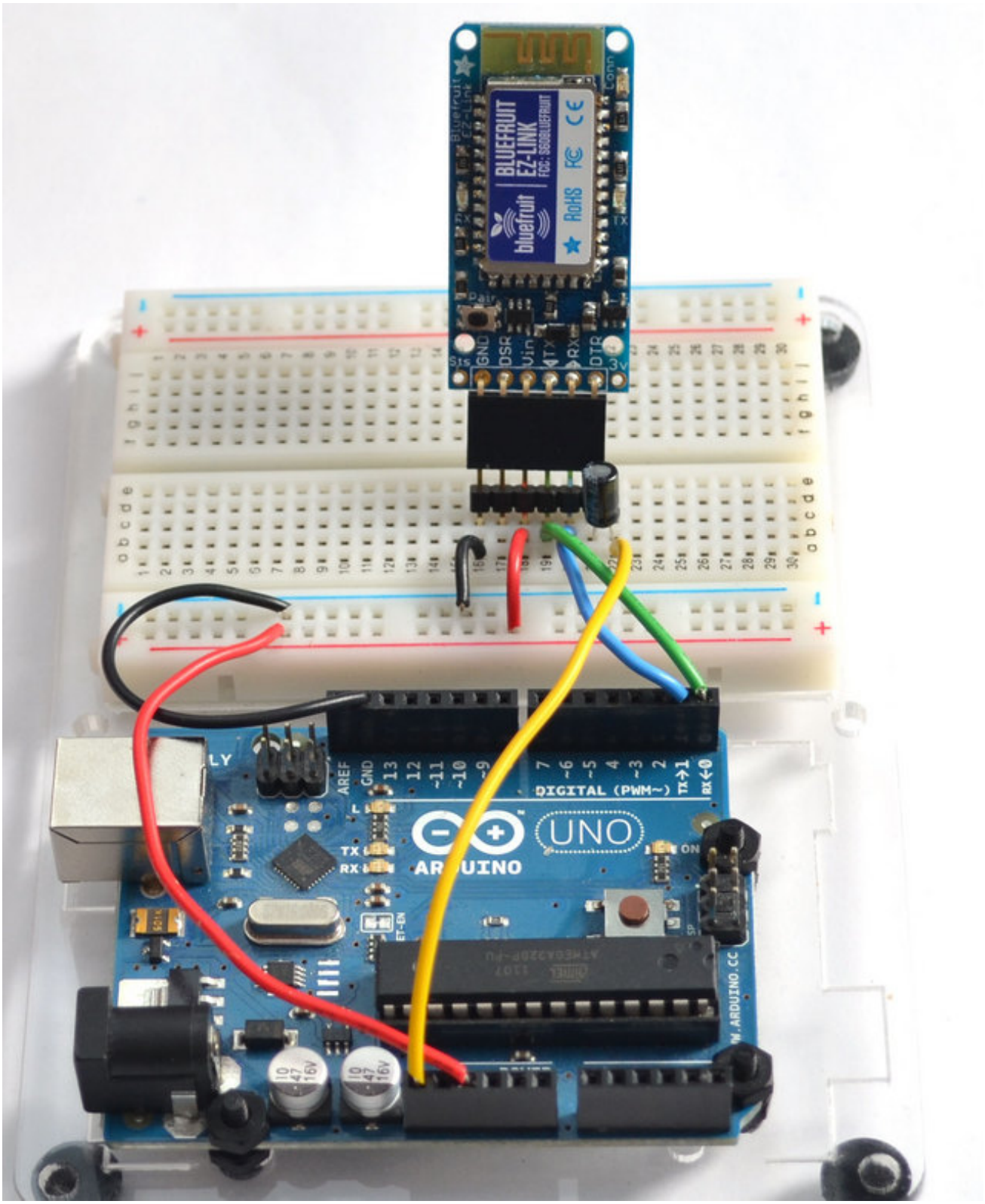
You will need to first be paired to the module, and ideally go through the loopback test so you know that's all working!

Wiring

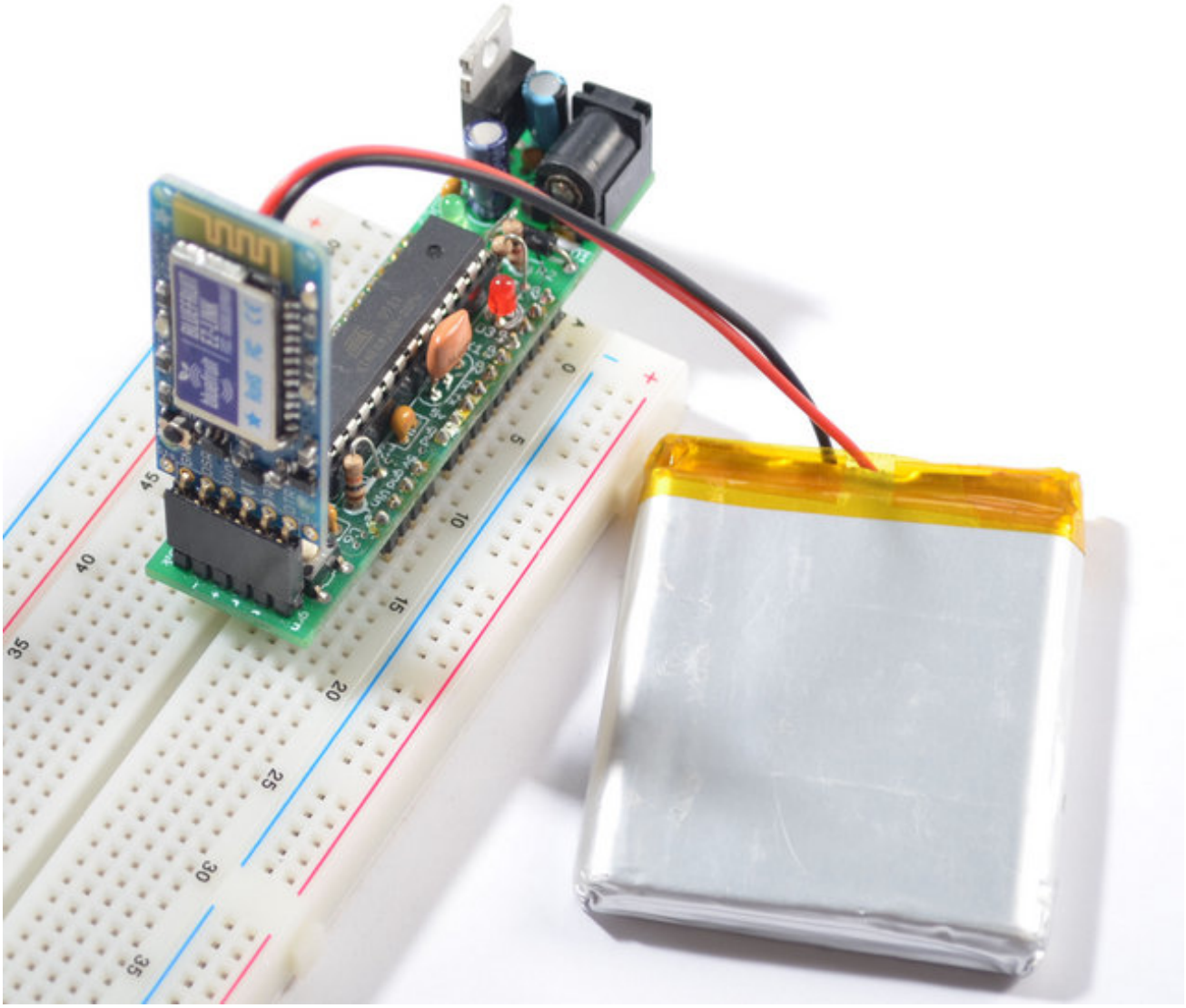
Wire up the EZ-Link to the Arduino as follows, we'll be using an Uno but the wiring is the same for another kind of Arduino/Compatible.

Connect:

- EZ-Link **GND** to Arduino **GND**
- EZ-Link **DSR** to **GND** (or no connect)
- EZ-Link **Vin** to Arduino **5V** (or any other 3-16V power input, 5V is ideal)
- EZ-Link **TX** to Arduino **#0** (RX)
- EZ-Link **RX** to Arduino **#1** (TX)
- EZ-Link **DTR** to a 1uF capacitor in series with Arduino **Reset** (The cap is in series, so **Reset** goes to the capacitor +, and capacitor - goes to **DTR**)



If you're using an Arduino compatible such as a Mini or Boarduino (or any of the other hundreds of Arduino-compat's), chances are it has an FTDI connector on it. You can plug the EZ-Link directly in, the **GND** pin lines up with the **Black** wire, and the **DTR** pin lines up with the **Green** wire. No other components are required



Upload

That's it! Now you can open up the Arduino IDE and select the COM/Serial port that you identified as the EZ-Link. If you open and close the Serial Monitor you should see the Arduino reset (the **L** pin #13 LED will blink)

If you're using Windows, the Arduino IDE might be a little sluggish while selecting the Bluetooth COM port, this is due to a bug in the underlying Java RX/TX library. You can fix it by following the instructions in this Arduino forum post (http://adafru.it/cVk)

Now make sure you have the right Arduino board select and upload as usual.

Since opening and closing a bluetooth connection takes a few seconds, there will be an extra 10 second delay when starting the upload process. This is due to the Bluetooth wireless protocol and how the Arduino IDE opens and closes the connection a few times to reset the Arduino. Please be patient!

F.A.Q.

Can I use AT commands to configure the EZ-Link?
There is no command mode for EZ-Link, it is designed to be used out of the box. Since it automatically detects the baud rates there is no need to set that up.
You can initiate pairing from any computer.

Can I change the BT name of my module?
At this time there is no way to change the BT name, it is hardcoded in and the last four digits match the lower two bytes of the MAC address

Can EZ-Link act as a BT 'master'?
No, it is a client-mode only device.

Hey! It's not working with my Android device!
Limit your Arduino sketch Serial communication speed to 9600 baud. This step is very important, if you try other baud rates the Android device will not be able to communicate with the Bluefruit EZ-link.

I'm sometimes getting odd errors on my Mac when uploading, says the port's busy?
The Mac bluetooth core sometimes doesn't fully release the bluetooth connection state as fast as we'd like. Wait a few seconds and try again!

Why is uploading sketches slower when using EZ-Link than using a USB cable?
Wireless just isn't as fast as wired. For example, with USB 'wired' data, the data is just 'sent' because the wire is trusted. With wireless/bluetooth, there has to be a lot more layers of checking and verification because there's no way to know if it was received and its suscepitble to noise. You can try to speed up by unchecking the Verify Firmware box in the Prefs of the Arduino IDE, that should cut it in half...but of course adds some risk!

I'm using Windows and unable to pair/connect to the Serial Port and/or the EZ-Link pairs but I cannot actually use it to send/receive data!
Make sure you do not have any extra Bluetooth drivers installed. In particular, make sure you don't have CSR/Harmony bluetooth drivers or any other non-Windows-native bluetooth stack. You should be using ONLY the native Windows bluetooth! Unpair the EZ-Link, uninstall any other drivers and reboot, then re-pair. For more details check out this forum thread:

https://forums.adafruit.com/viewtopic.php?f=53&t=63214&p=321810#p321622 (http://adafru.it/eca)