Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

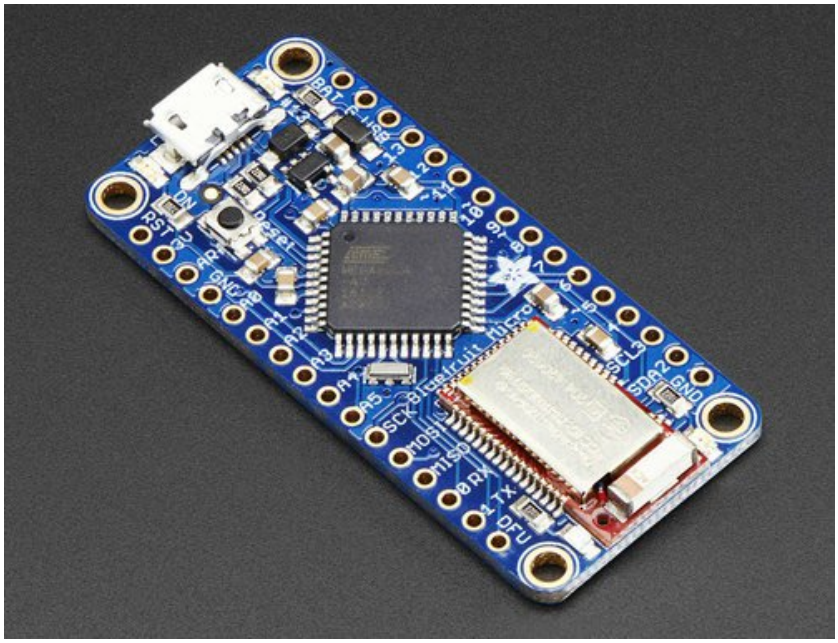
We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Introducing Bluefruit LE Micro

Created by lady ada



Last updated on 2016-09-21 10:01:42 PM UTC

Guide Contents

Guide Contents 2
Overview 9
- Program over USB with the Arduino IDE 9
- The Power of Bluefruit LE 10
- Use the Bluefruit App to get your project started 11
- You can do a lot more too! 11

Pinouts 13
Major Parts 13
Power Pins 14
LEDs 15
BLE Use Pins 15
Available GPIO 16
Other Pins 16
Using with Arduino IDE 17
Install Drivers (Windows Only) 17
Install Bluefruit Micro board support 17
- Ubuntu & Linux Issue Fix 17

Install the Adafruit nRF51 BLE Library 17
Run first example 18
Uploading to the Bluefruit Micro 19
- Uploading to a brand new board/Upload failures 20

Run the sketch 21
AT command testing 22
Configuration! 24
Which board do you have? 24
- Bluefruit Micro or Feather 32u4 Bluefruit 24
- Feather M0 Bluefruit LE 24
- Bluefruit LE SPI Friend 25
- Bluefruit LE UART Friend or Flora BLE 25

Configure the Pins Used 25
- Common settings: 26
- Software UART 26
- Hardware UART 26

Mode Pin 26
SPI Pins 26
Software SPI Pins 26
Select the Serial Bus 26
UART Based Boards (Bluefruit LE UART Friend & Flora BLE) 26
SPI Based Boards (Bluefruit LE SPI Friend) 27
BLEUart 28
Opening the Sketch 28
Configuration 28
Running the Sketch 29
ATCommand 33
Opening the Sketch 33
Configuration 33
Running the Sketch 34
HIDKeyboard 36
Opening the Sketch 36
Configuration 36
Running the Sketch 37
Bonding the HID Keyboard 37
Android 38
iOS 39
OS X 40
Controller 42
Opening the Sketch 42
Configuration 42
Running the Sketch 43
Using Bluefruit LE Connect in Controller Mode 43
Streaming Sensor Data 44
Control Pad Module 45
Color Picker Module 46
HeartRateMonitor 48
Opening the Sketch 48
Configuration 48

If Using Hardware or Software UART 49
Running the Sketch 49
nRF Toolbox HRM Example 50
CoreBluetooth HRM Example 51
UriBeacon 53
Opening the Sketch 53
Configuration 53
Running the Sketch 54
HALP! 55
AT Commands 56
Test Command Mode '=?' 56
Write Command Mode '=xxx' 56
Execute Mode 56
Read Command Mode '?' 57
Standard AT 58
AT 58
ATI 58
ATZ 58
ATE 58
+++ 59
General Purpose 60
AT+FACTORYRESET 60
AT+DFU 60
AT+HELP 60
AT+NVMWRITE 60
AT+NVMREAD 61
Hardware 62
AT+BAUDRATE 62
AT+HWADC 62
AT+HWGETDIETEMP 62
AT+HWGPIO 63
AT+HWGPIOMODE 63

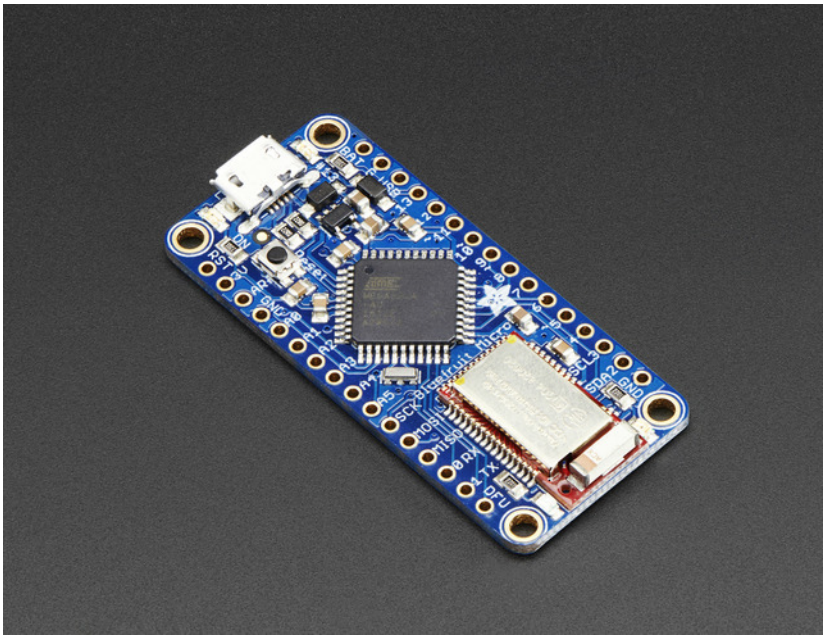
AT+HWI2CSCAN 64
AT+HWVBAT 64
AT+HWRANDOM 64
AT+HWMODELED 64
AT+UARTFLOW 65
Beacon 66
AT+BLEBEACON 66
AT+BLEURIBEACON 67
Deprecated: AT+EDDYSTONEENABLE 68
AT+EDDYSTONEURL 68
AT+EDDYSTONECONFIGEN 68
AT+EDDYSTONESERVICEEN 69
AT+EDDYSTONEBROADCAST 69
BLE Generic 70
AT+BLEPOWERLEVEL 70
AT+BLEGETADDRTYPE 70
AT+BLEGETADDR 70
AT+BLEGETPEERADDR 71
AT+BLEGETRSSI 71
BLE Services 72
AT+BLEUARTTX 72
TX FIFO Buffer Handling 72
AT+BLEUARTRX 73
AT+BLEUARTFIFO 73
AT+BLEKEYBOARDEN 73
AT+BLEKEYBOARD 74
AT+BLEKEYBOARDCODE 74
Modifier Values 74
AT+BLEHIDEN 75
AT+BLEHIDMOUSEMOVE 75
AT+BLEHIDMOUSEBUTTON 76
AT+BLEHIDCONTROLKEY 76
AT+BLEHIDGAMEPAD 77

AT+BLEMIDIEN 77
AT+BLEMIDIRX 78
AT+BLEMIDITX 78
AT+BLEBATTEN 78
AT+BLEBATTVAL 78
BLE GAP 80
AT+GAPCONNECTABLE 80
AT+GAPGETCONN 80
AT+GAPDISCONNECT 80
AT+GAPDEVNAME 80
AT+GAPDELBONDS 81
AT+GAPINTERVALS 81
AT+GAPSTARTADV 82
AT+GAPSTOPADV 82
AT+GAPSETADVDATA 82
BLE GATT 84
GATT Limitations 84
AT+GATTCLEAR 84
AT+GATTADDSERVICE 84
AT+GATTADDCHAR 85
AT+GATTCHAR 86
AT+GATTLIST 87
AT+GATTCHARRAW 88
Debug 89
AT+DBGMEMRD 89
AT+DBGNVMRD 89
AT+DBGSTACKSIZE 89
AT+DBGSTACKDUMP 89
History 92
Version 0.7.0 92
Version 0.6.7 93
Version 0.6.6 93

Version 0.6.5 94
Version 0.6.2 94
Version 0.5.0 94
Version 0.4.7 94
Version 0.3.0 95
SDEP (SPI Data Transport) 96
SDEP Overview 96
SPI Setup 96
SPI Hardware Requirements 96
IRQ Pin 96
SDEP Packet and SPI Error Identifier 96
Sample Transaction 96
SDEP (Simple Data Exchange Protocol) 97
Endianness 97
Message Type Indicator 97
SDEP Data Transactions 97
Message Types 97
Command Messages 97
Response Messages 98
Alert Messages 99
Standard Alert IDs 99
Error Messages 99
Standard Error IDs 100
Existing Commands 100
SDEP AT Wrapper Usage 100
GATT Service Details 102
UART Service 102
UART Service 103
Characteristics 103
TX (0x0002) 103
RX (0x0003) 103
Software Resources 104
Bluefruit LE Client Apps and Libraries 104
Bluefruit LE Connect (http://adafru.it/f4G) (Android/Java) 104
Bluefruit LE Connect (http://adafru.it/f4H) (iOS/Swift) 104
Bluefruit LE Connect for OS X (http://adafru.it/o9F) (Swift) 104
Bluefruit LE Command Line Updater for OS X (http://adafru.it/pLF)

(Swift) 105
Deprecated: Bluefruit Buddy (http://adafru.it/mCn) (OS X) 105
ABLE (http://adafru.it/ijB) (Cross Platform/Node+Electron) 106
Bluefruit LE Python Wrapper (http://adafru.it/fQF) 106
Debug Tools 107
AdaLink (http://adafru.it/fPq) (Python) 107
Adafruit nRF51822 Flasher (http://adafru.it/fVL) (Python) 107
BLE FAQ 108
Bluefruit LE Connect (Android) 109
Nordic nRF Toolbox 109
Adafruit_nRF51822_Flasher 109
DFU Bluefruit Updates 112
Downloads 113
Datasheets 113
Drivers 113
Schematic 113
Fabrication Print 113

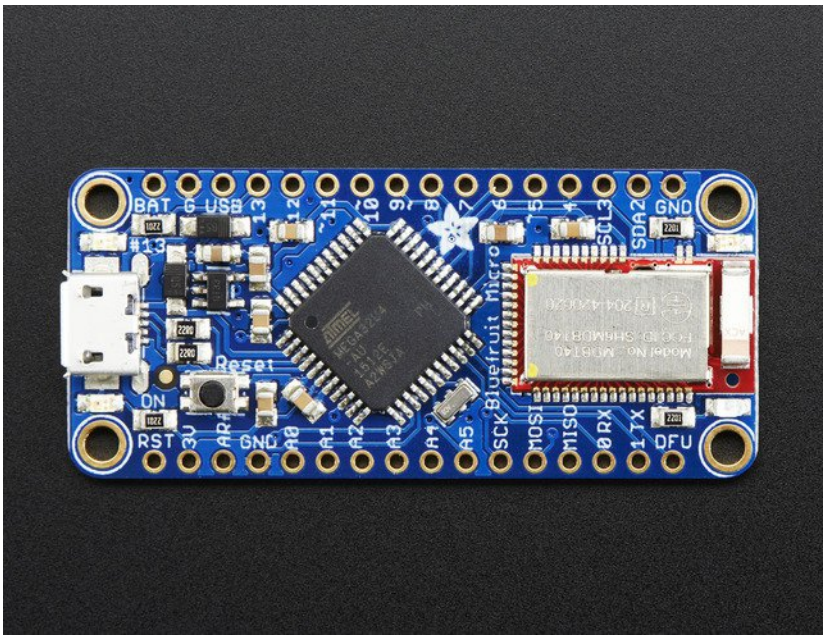
Overview



Would you like to add powerful and easy-to-use Bluetooth Low Energy to your robot, art or other electronics project? Heck yeah! With BLE now included in modern smart phones and tablets, its fun to add wireless connectivity. So what you really need is the new Adafruit Bluefruit LE Micro - it's an Atmega32u4 chip (same as used as in the Adafruit Flora (http://adafru.it/dVl) or Arduino Leonardo (http://adafru.it/dy8)) plus our SPI Bluefruit LE Friend (http://adafru.it/fLp) in one.

Bluetooth Low Energy is the hottest new low-power, 2.4GHz spectrum wireless protocol. It's on all the talk shows, it's featured on the cover of Vogue, it even has it's very own perfume! OK, maybe that stuff isn't completely true. But it **is** a very exciting way to have your electronic projects communicate with a BLE-capable computer, tablet or phone.

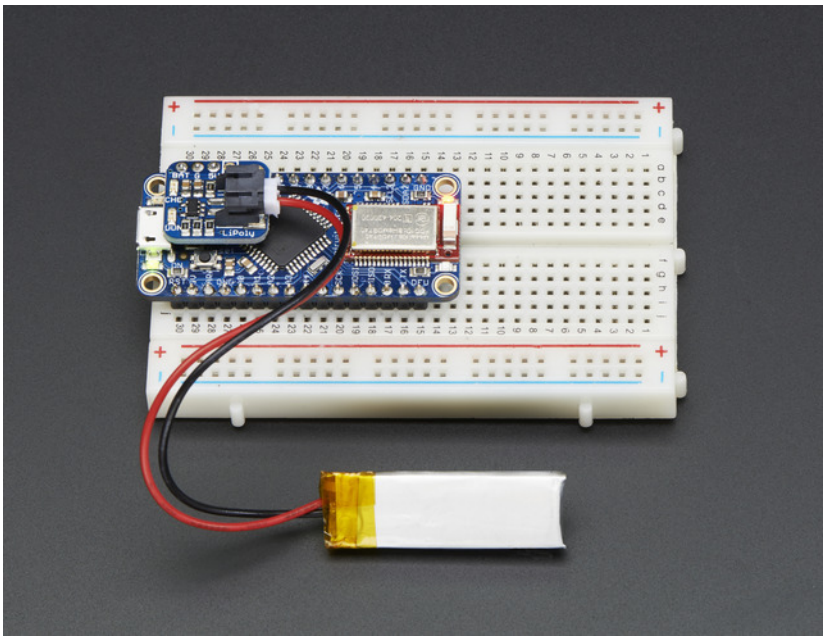
In particular, its the only wireless protocol that you can use with iOS without needing special certification and it's supported by all modern smart phones. This makes it excellent for use in portable projects that will make use of an iOS or Android phone or tablet.



Program over USB with the Arduino IDE

The Bluefruit LE Micro makes this easier than ever, by combining an ATmega32u4 microcontroller and our Bluefruit LE module. You can program the ATmega32u4 over USB using the built-in USB bootloader, either directly with avrdude or using the Arduino IDE (it's the same as used as in the Adafruit Flora (http://adafru.it/dVl) or Arduino Leonardo (http://adafru.it/dy8). This microcontroller runs at 8MHz clock speed, and 3.3V logic for compatibility with the vast majority of sensors available on the market. It has 20+ GPIO pins, including I2C, SPI, a UART, and 6 analog inputs. It has 28KB of FLASH available for your program, and 2KB of RAM. And, of course, native USB for programming and communication.

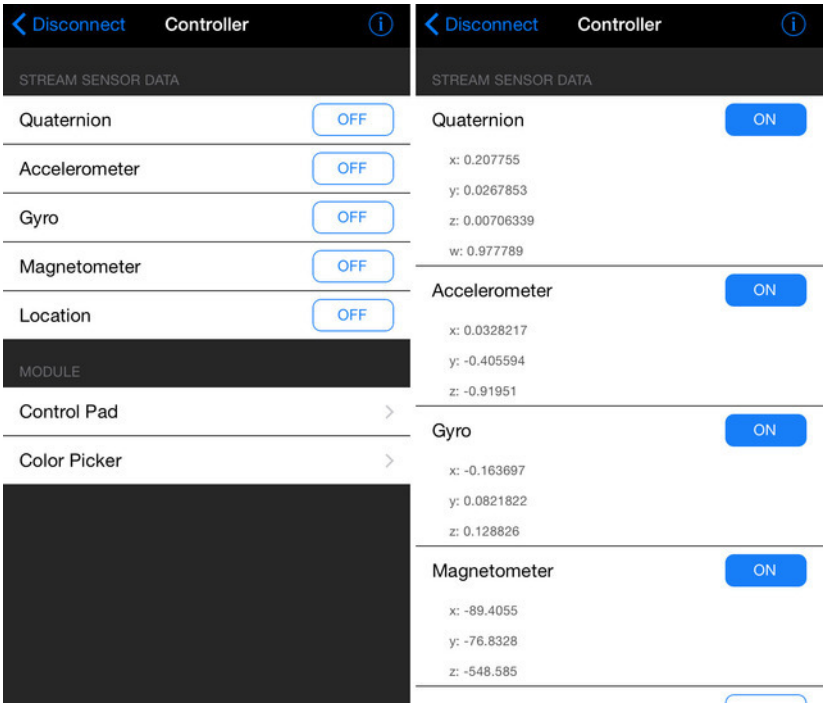
You can also add a rechargable Lithium Polymer battery by picking up a LiPoly backpack(http://adafru.it/e0w). Solder it on top of the Bluefruit LE Micro and it will let you charge the battery over the micro USB connector. When the USB is unplugged, it will run off the battery.



The Power of Bluefruit LE

The Bluefruit LE module is an nRF51822 chipset from Nordic, programmed with multi-function code that can do quite a lot! For most people, they'll be very happy to use the standard Nordic UART RX/TX connection profile. In this profile, the Bluefruit acts as a data pipe, that can 'transparently' transmit back and forth from your iOS or Android device. You can use our iOS App (http://adafru.it/iCi) or Android App (http://adafru.it/f4G), or write your own to communicate with the UART service (http://adafru.it/iCF).

The board is capable of much more than just sending strings over the air! Thanks to an easy to learn AT command set (http://adafru.it/iCG), you have full control over how the device behaves, including the ability to define and manipulate your own GATT Services and Characteristics (http://adafru.it/iCH), or change the way that the device advertises itself for other Bluetooth Low Energy devices to see. You can also use the AT commands to query the die temperature, check the battery voltage, and more, check the connection RSSI or MAC address, and tons more. Really, way too long to list here!



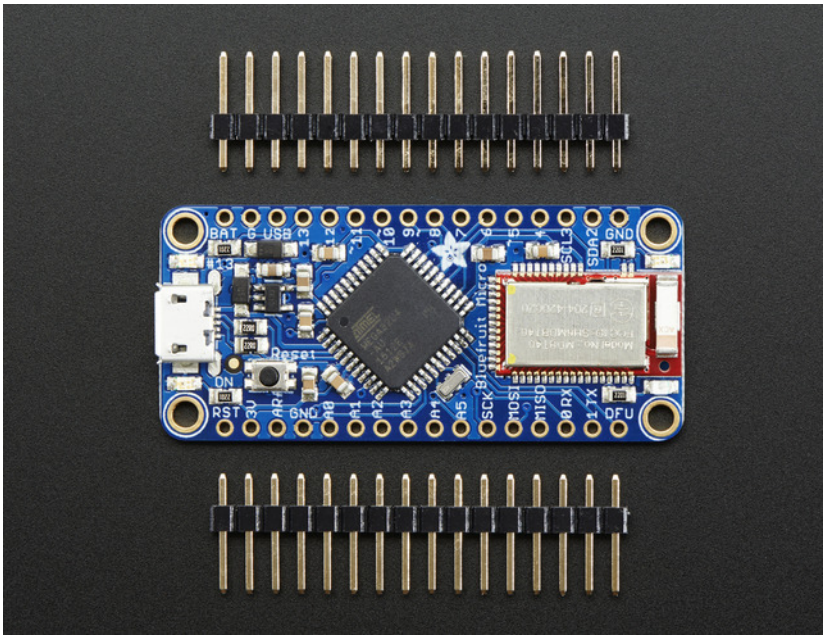
Use the Bluefruit App to get your project started

Using our Bluefruit iOS App (http://adafru.it/iCi) or Android App (http://adafru.it/f4G), you can quickly get your project prototyped by using your iOS or Android phone/tablet as a controller. We have a color picker (http://adafru.it/iCI), quaternion/accelerometer/gyro/magnetometer or location (GPS) (http://adafru.it/iCI), and an 8-button control game pad (http://adafru.it/iCI). This data can be read over BLE and piped into the ATmega32u4 chip for processing & control

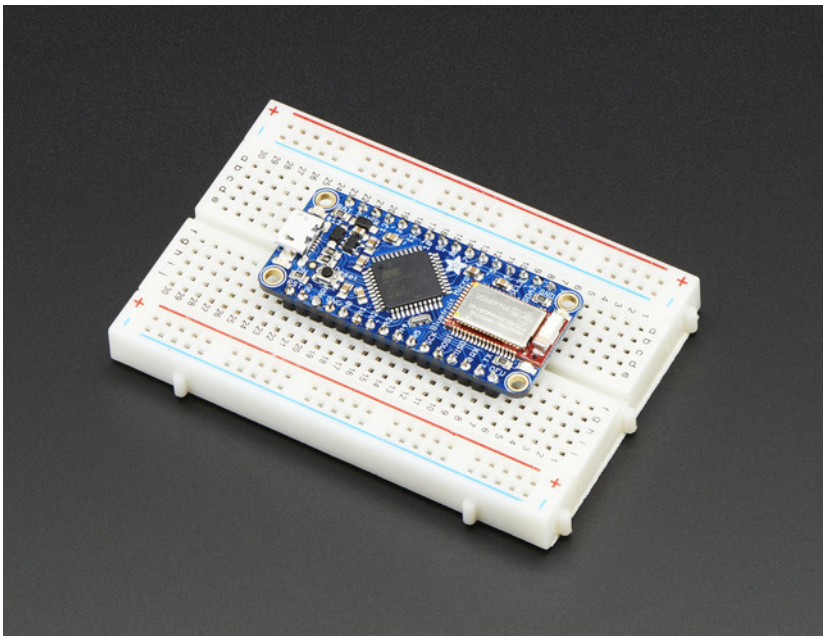
Once you get the hang of our using the Bluefruit app, you can experiment with making your own custom AppOne option is using Cordova - you still have to do a little programming but its much easier than doing it from scratch (http://adafru.it/iOf)

You can do a lot more too!

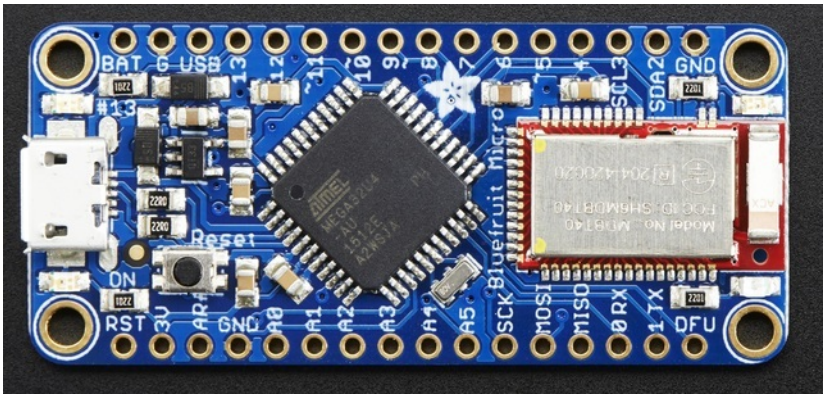
- The Bluefruit can also act like an HID Keyboard (http://adafru.it/iOA), Mouse or Media Controller (for devices that support BLE HID)
- Can become a BLE Heart Rate Monitor (http://adafru.it/iOB) (a standard profile for BLE) - you just need to add the pulse-detection circuitry
- Turn it into a UriBeacon (http://adafru.it/iOC) (now called Eddystone (http://adafru.it/fSA)), the Google standard for Bluetooth LE beacons. Just power it and the 'Friend will bleep out a URL to any nearby devices with the UriBeacon app installed.
- Built in over-the-air bootloading capability so we can keep you updated with the hottest new firmware (http://adafru.it/iOD). Use any Android or iOS device to get updates and install them. This will update the native code on the BLE module, to add new wireless capabilities, not program the ATmega chip.



Each order comes with one fully assembled and tested Bluefruit LE Micro board. The ATmega32u4 on board comes with a USB bootloader for quick programming. The BLE module comes with our latest firmware. We also include some headers which you can solder in to use the module in a solderless breadboard.



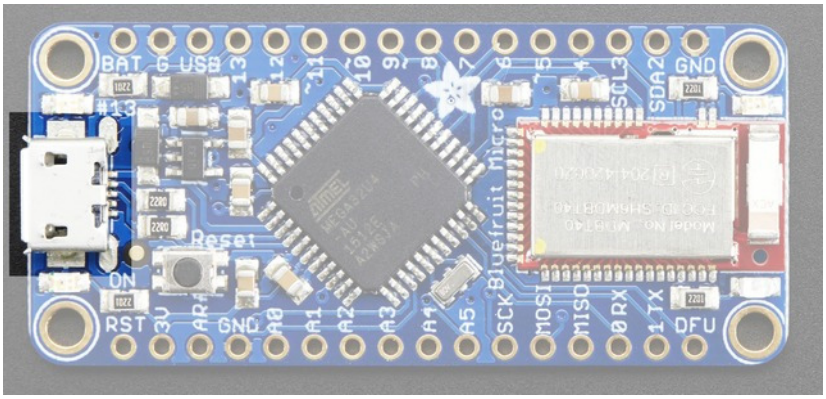
Pinouts



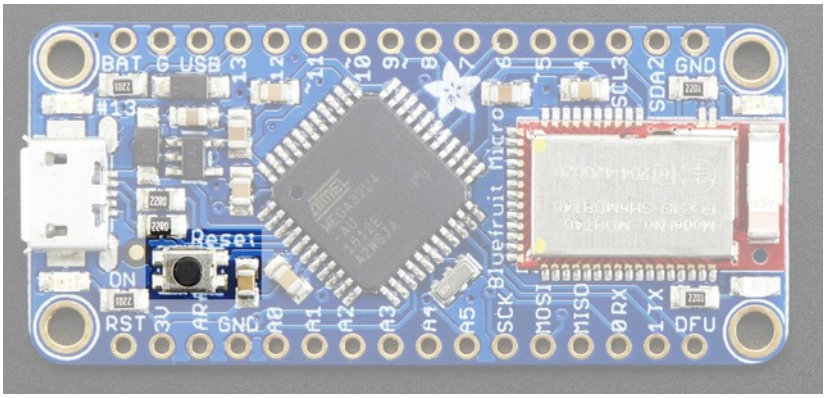
There's a lot going on in this board so lets do a high-level overview of major components, then dig into the individual pinouts

Major Parts

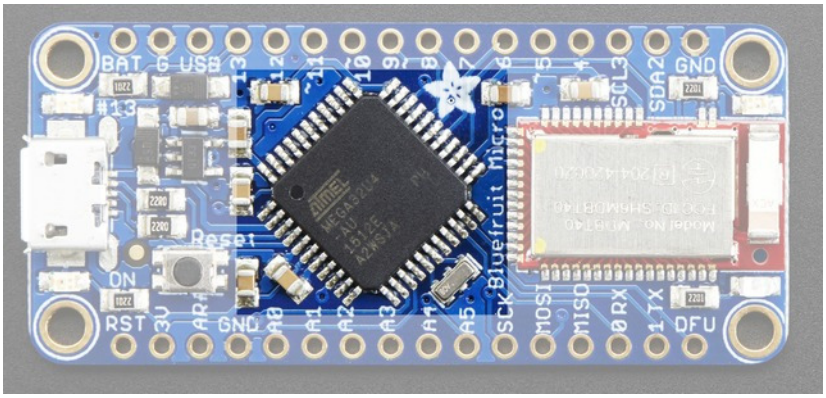
There's a few major components you need to know about.



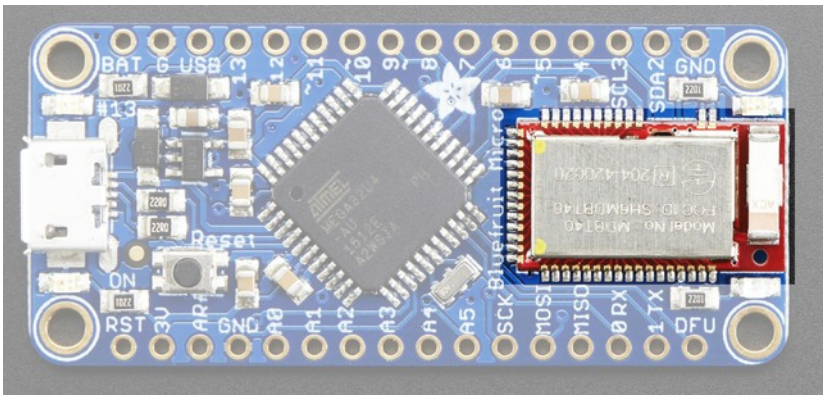
Starting from the left, there's the Micro USB connector. This is how you power and program the Bluefruit Micro. If you added on the LiPoly backpack you can also recharge the battery this way.



To the right is a small tactile switch. This is the reset button for the ATmega32u4. You can use this to reset the microcontroller and/or push it into bootloader mode if the bootloader gets 'stuck' during programming



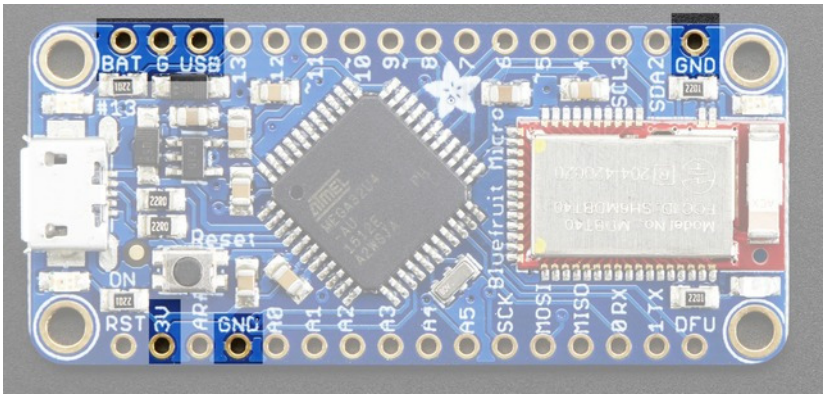
This is the main microcontroller, an ATmega32u4 chip. This chip has 32KB of flash (4KB taken by bootloader, 28K available for user code), 2.5KB of RAM, and lots of GPIO. It runs at 8MHz and 3.3V logic. It is the same chip used as in the Adafruit Flora (http://adafru.it/dVl) or Arduino Leonardo (http://adafru.it/dy8) so if you have used those, then this chip will be very familiar. You can use the Arduino IDE to program this chip once you add the board with the Board Manager and many (although not 100%) of Arduino libraries will work out-of-the-box.



Finally there's the BLE module. This is the radio-handling part. It is an FCC/CE-certified emittermodule with a ceramic antenna. The ATmega32u4 talks to this module over SPI using our Arduino library to transmit and receive data. It is based on the nRF51822 chipset from Nordic Semiconductor.

Power Pins

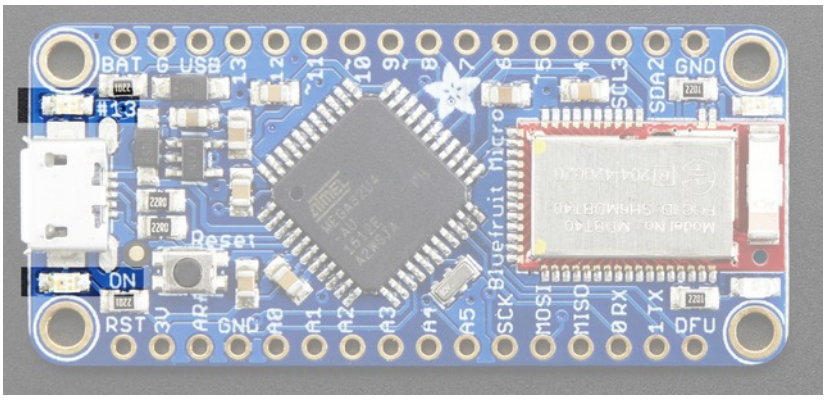
These are the pins you'll be connecting to for power input and output.



- **BAT** is a battery input port. If you are using the LiPoly backpack, this connects to the lithium ion/polymer batter. You can also inject any other kind of battery in here, from 3-16VDC. It has a Schottkey diode for polarity protection
- **G & GND** - There are 3 ground pins available
- **USB** is the 'raw' 5V power from the USB port when power is coming in via USB. You can use this if you need up to 500mA current @ 5V but its only live when USB is connected
- **3V** This is the output from the onboard regulator, 3.3V output at 150mA

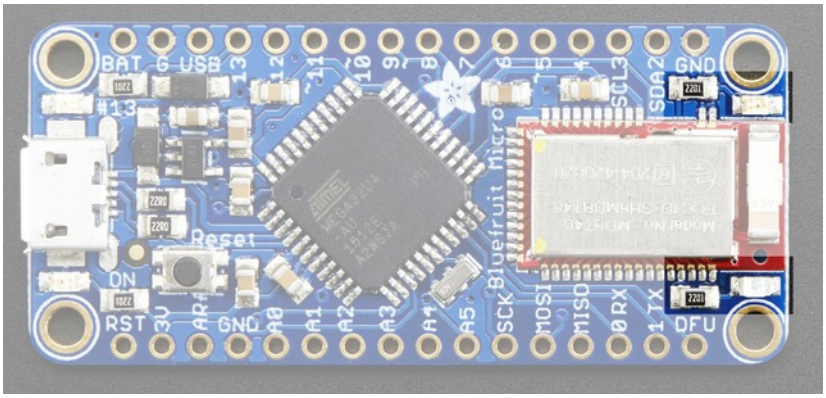
LEDs

There are four indicators for telling you what is up with your Bluefruit Micro



The two LEDs on the left next to the MicroUSB jack are

- **Red #13 LED** which is connected to digital #13 / PC7 of the ATmega32u4. You can use it to signal or blink. It will also 'pulse' on and off when the bootloader is active
- **Green ON LED** this LED is always on when the board is powered.

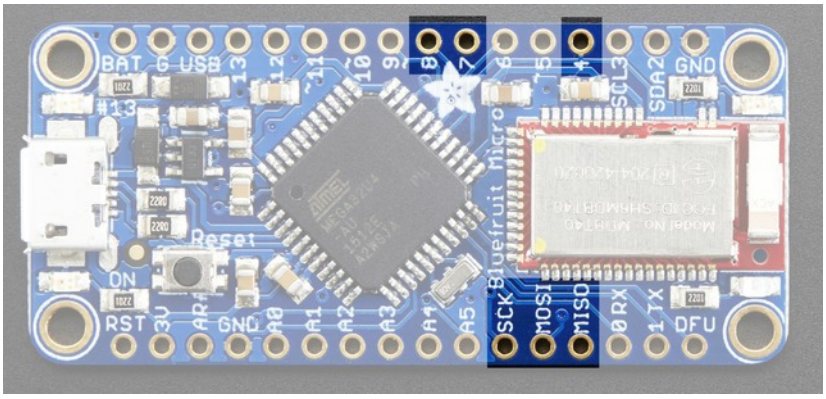


The two LEDs on the right next to the BLE module are

- **Red BLE status LED** This LED will blink 3 times every few seconds to let you know that the BLE module is alive and running/waiting for connection. It's also used when doing a DFU update on the module.
- **Blue BLE connection LED** lights up when a device is connected via BLE.

BLE Use Pins

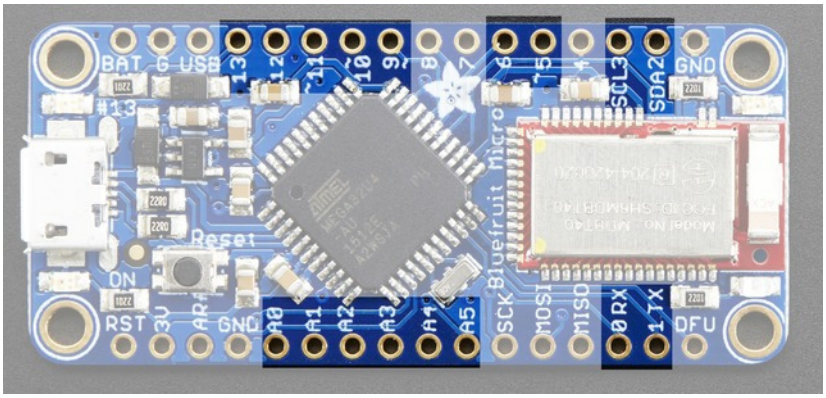
These pins are used to talk to the module. You can use the MOSI/MISO/SCK pins for other SPI devices (we use SPI transactions to avoid data collisions) but keep the 3 other pins unused for anything else.



- **Digital 4** - connected to the Bluefruit Reset pin, resets the BLE module when set to low.
- **Digital 7** - the IRQ from the Bluefruit to the microcontroller. Used for the Bluefruit to tell the microcontroller when new data arrived/is ready to read
- **Digital 8** - the Bluefruit CS pin, used to select the module for data transfer.
- **MOSI/MISO/SCK** - SPI pins used to send/receive data and commands between the microcontroller and BLE

Available GPIO

There's a ton of pins available!



All of these pins are digital input/output pins, with 3.3V logic.Some have extra capabilities

- **#0** - Hardware UART RX pin
- **#1** - Hardware UART TX pin
- **#2** - I2C SDA (data) pin
- **#3** - I2C SCL (clock) pin
- **#5** - PWM (analogWrite) output
- **#9** - PWM (analogWrite) output
- **#10** - PWM (analogWrite) output
- **#11** - PWM (analogWrite) output
- **#12** - GPIO
- **#13** - Connected to a red LED next to the micro USB jack
- **A0 thru A5** - Analog inputs.

You can also use the **SCK/MOSI/MISO** pins with an SPI device as long as it is on a Chip Select line and you disable it when you talk to the BLE module.

Other Pins

- **ARef** - This is the analog reference input for the ATmega32u4, you can use this to change the analog reference for **analogRead()** however do not set higher than 3.3V (which is the default, also)
- **DFU** - This is used when updating the firmware on the BLE module itself. Don't use unless you need/want to update the BLE device runcode. It will not reprogram the ATmega32u4

Using with Arduino IDE

Install Drivers (Windows Only)

Start out by installing the serial/CDC drivers for the Bluefruit LE Micro. We'll be using PJRC's awesome generic CDC drivers, works with all Windows and all CDC devices. (Thanks Paul!) (http://adafru.it/fLA)

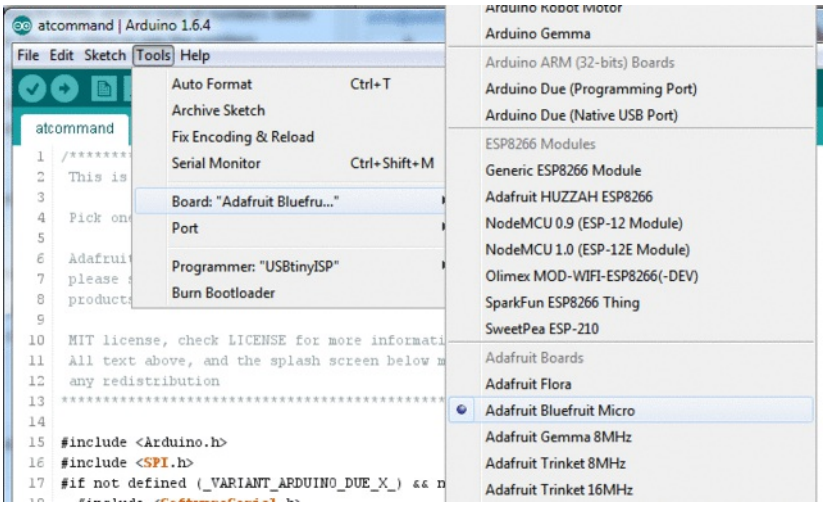
Mac/Linux does not need a driver, continue as is!

Install Bluefruit Micro board support

Chances are you want to jump right in and start programming the Bluefruit Micro! The fastest/easiest way to do that is to use the Arduino IDE. You'll need v1.6.4 or greater, which has a Board Manager support.

Check out that guide first, to add Adafruit Boards support!(http://adafru.it/f7X)

OK now that you are back, launch the IDE and select Bluefruit Micro from the**Boards** menu



Ubuntu & Linux Issue Fix

Note if you're using Ubuntu 15.04 (or perhaps other more recent Linux distributions) there is an issue with the modem manager service which causes the Bluefruit LE micro to be difficult to program. If you run into errors like "device or resource busy", "bad file descriptor", or "port is busy" when attempting to program then you are hitting this issue. (http://adafru.it/fP6)

The fix for this issue is to make sure Adafruit's custom udev rules are applied to your system. One of these rules is made to configure modem manager not to touch the Bluefruit Micro board and will fix the programming difficulty issue. Follow the steps for installing Adafruit's udev rules on this page. (http://adafru.it/iOE)

Install the Adafruit nRF51 BLE Library

In order to try out our demos, you'll need to download the Adafruit BLE library for the nRF51 based modules such as this one (a.k.a. Adafruit_BluefruitLE_nRF51)

You can check out the code here at github,(http://adafru.it/f4V) but its likely easier to just download by clicking:

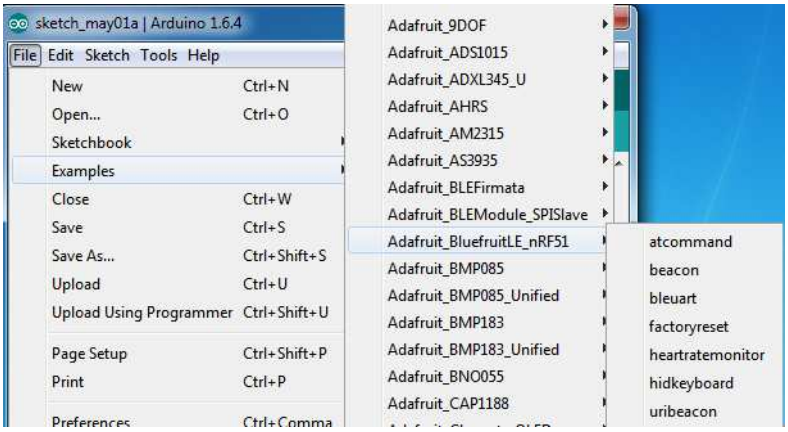
Download the Adafruit nRF51 Bluetooth Library
http://adafru.it/f4W

Rename the uncompressed folder**Adafruit_BluefruitLE_nRF51** and check that the **Adafruit_BluefruitLE_nRF51** folder contains **Adafruit_BLE.cpp** and **Adafruit_BLE.h** (as well as a bunch of other files)

Place the **Adafruit_BluefruitLE_nRF51** library folder your ***arduinosketchfolder*/libraries/** folder.
You may need to create the**libraries** subfolder if its your first library. Restart the IDE.

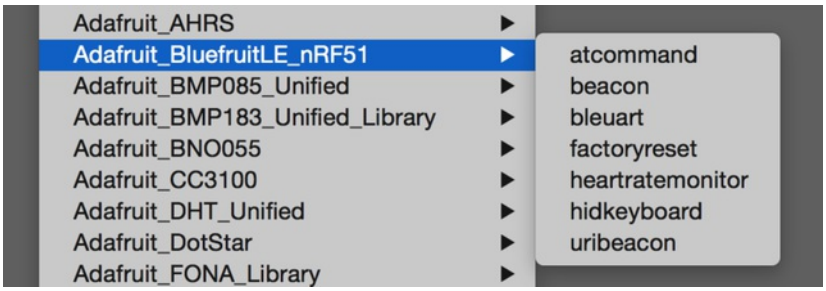
We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (http://adafru.it/aYM)

After restarting, check that you see the library folder with examples:

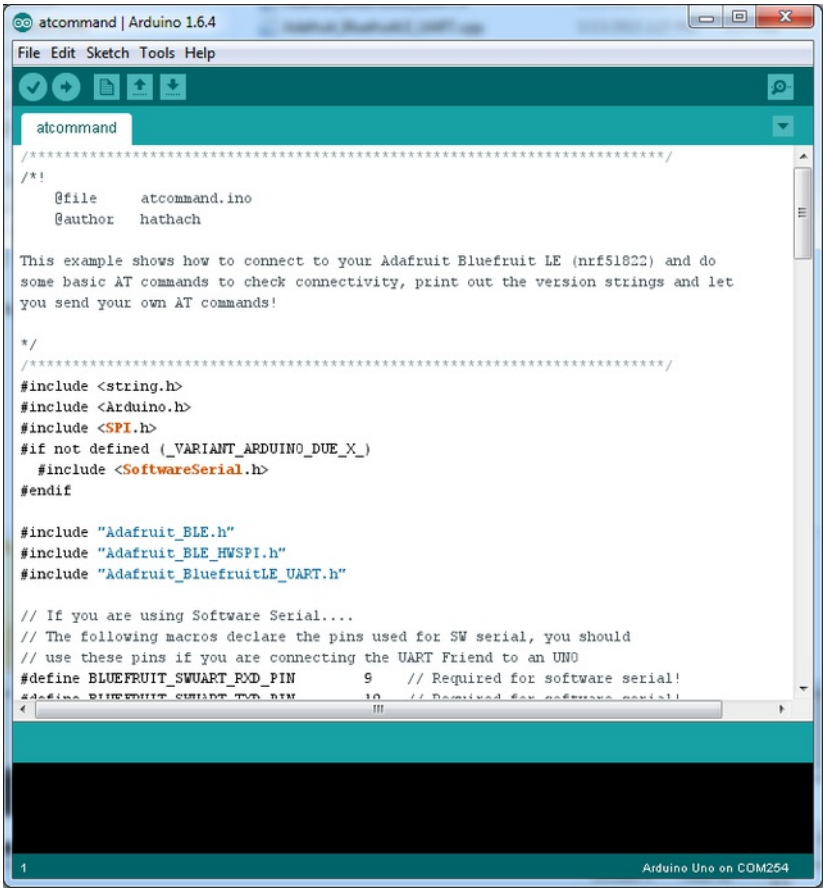


Run first example

Lets begin with the beginner project, which we can use to do basic tests. To open the ATCommand sketch, click on the **File > Examples > Adafruit_BluefruitLE_nRF51** folder in the Arduino IDE and select **atcommand**:



This will open up a new instance of the example in the IDE, as shown below:



Don't upload the sketch yet! You will have to begin by changing the configuration.

Go to the second tab labeled **BluefruitConfig.h** and find these lines

```
// SHARED SPI SETTINGS
// -----------------------------------------------------------------------------------------
// The following macros declare the pins to use for HW and SW SPI communication.
// SCK, MISO and MOSI should be connected to the HW SPI pins on the Uno when
// using HW SPI.  This should be used with nRF51822 based Bluefruit LE modules
// that use SPI (Bluefruit LE SPI Friend).
// -----------------------------------------------------------------------------------------
#define BLUEFRUIT_SPI_CS               8
#define BLUEFRUIT_SPI_IRQ              7
#define BLUEFRUIT_SPI_RST              6    // Optional but recommended, set to -1 if unused
```

And change the last line to:

```
#define BLUEFRUIT_SPI_RST              4   // Optional but recommended, set to -1 if unused
```

(The Bluefruit Micro has the reset on digital #4 not #6)

Now go back to the main tab **atcommand** and look for this line of code

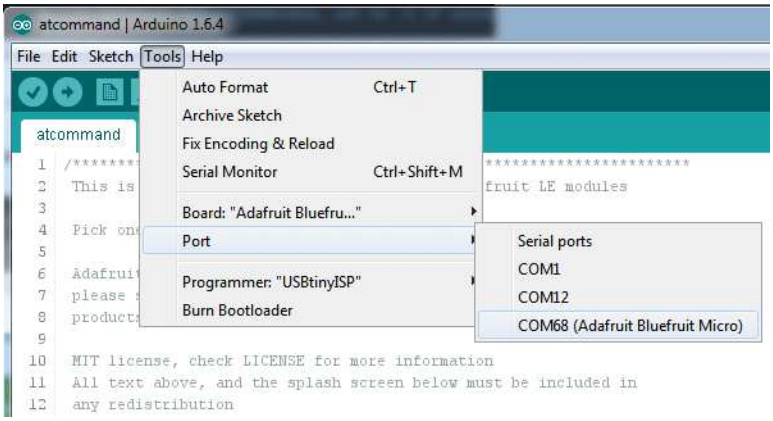
```
/* ...hardware SPI, using SCK/MOSI/MISO hardware SPI pins and then user selected CS/IRQ/RST */
Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ, BLUEFRUIT_SPI_RST);
```

Make sure that the second line is uncommented (it should be)

OK now you can upload to the Bluefruit Micro!
If you're using Ubuntu 15.04 or other Linux distributions and run into errors attempting to upload a program to the board, scroll up to the Ubuntu and Linux issue fix section above.

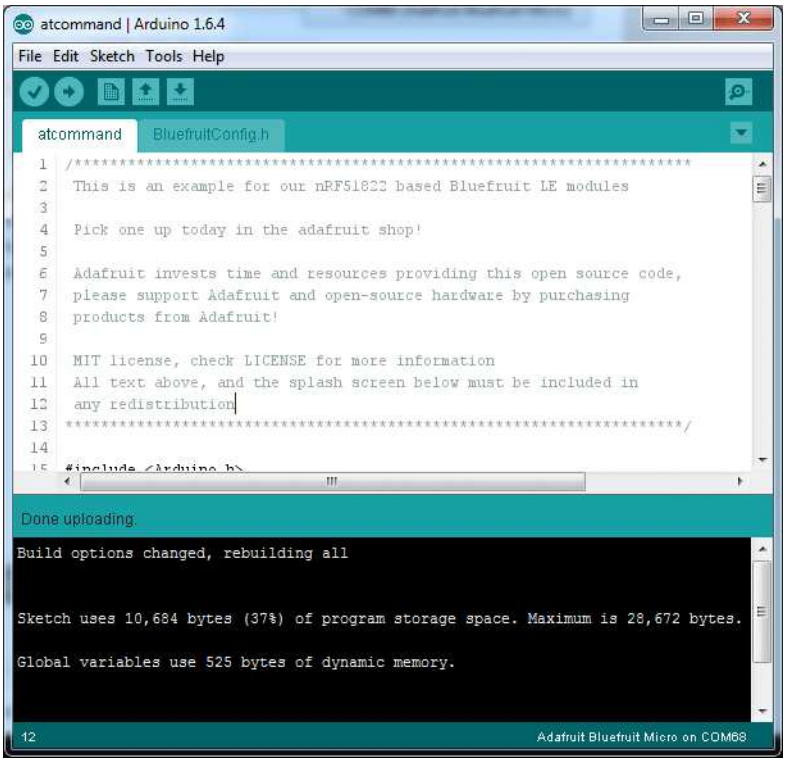
Uploading to the Bluefruit Micro

It's pretty easy to upload, first up make sure you have **Bluefruit Micro** selected on the boards dropdown as above. Also, in the **Ports** menu, look for the port labeled as such:



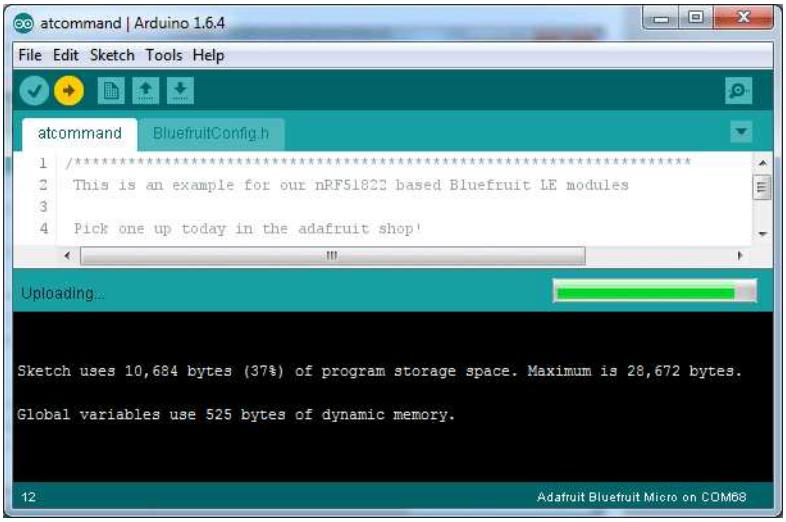
Now click the upload button on the Arduino IDE (or **File Menu -> Upload**)

If all is good you will see **Done Uploading** in the status bar



Uploading to a brand new board/Upload failures

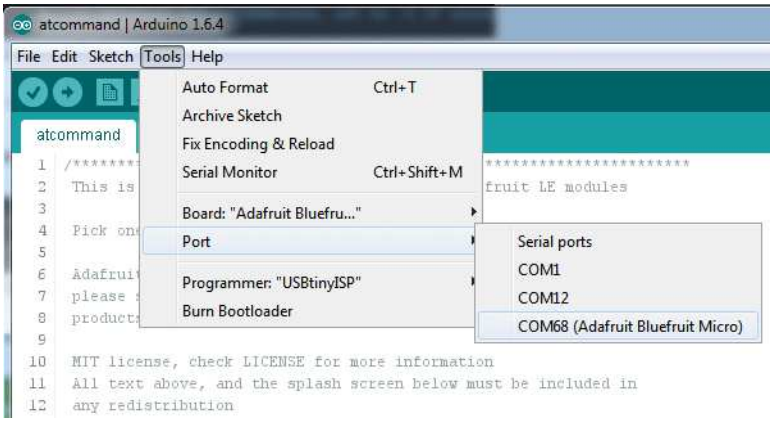
If you are uploading for the first time to a new board, or if upload fails, pres the **RESET** mini button on the Bluefruit Micro when you see the Yellow Arrow lit and the **Uploading...** text in the status bar. When you see the red LED pulsing on and off, you know the bootloader is running.



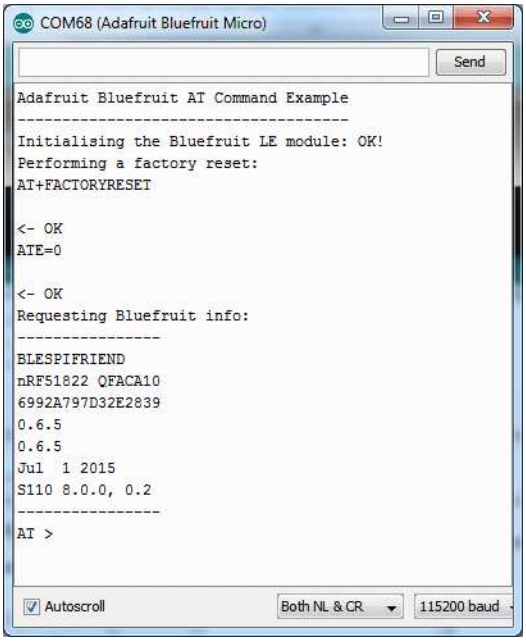
Don't click the reset button **before** uploading, unlike other bootloaders you want this one to run at the time Arduino is trying to upload

Run the sketch

OK check again that the correct port is selected



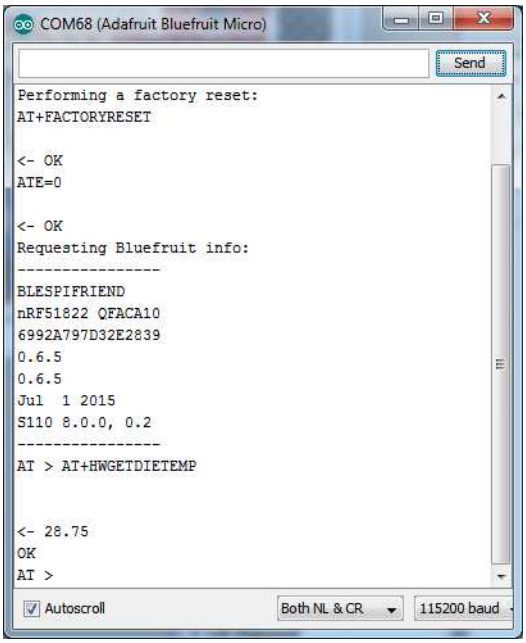
Then open up the Serial console. You will see the following:

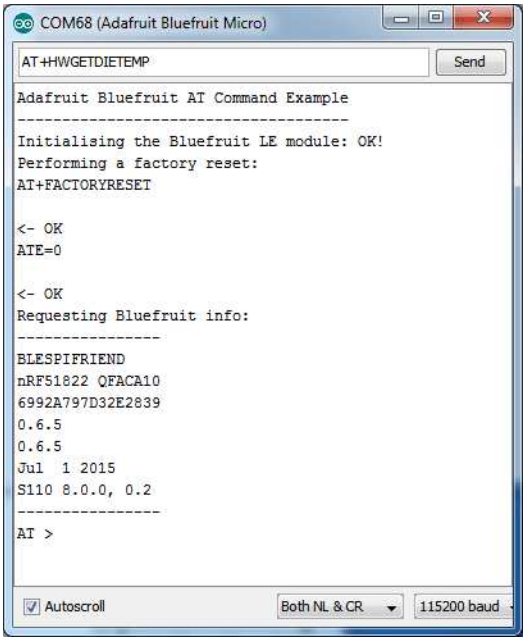


This sketch starts by doing a factory reset, then querying the BLE radio for details. These details will be useful if you are debugging the radio. If you see the information as above, you're working! (Note that the dates and version numbers may vary)

AT command testing

Now you can try out some **AT commands** - check the rest of the learn guide for a full list. We'll just start with **AT+HWGETDIETEMP** which will return the approximate ambient temperature of the BLE chipset





OK now you know how to upload/test/communicate with your Bluefruit Micro. Next up we have a bunch of tutorials who can follow for checking out the bluetooth le radio and apps.

For all the following examples, we share the same code between various modules so**don't forget to make sure you have the RESET pin set to 4 in BluefruitConfig.h for each sketch before uploading, and that Hardware SPI mode is selected by checking that**

```
/* ...hardware SPI, using SCK/MOSI/MISO hardware SPI pins and then user selected CS/IRQ/RST */
Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ, BLUEFRUIT_SPI_RST);
```

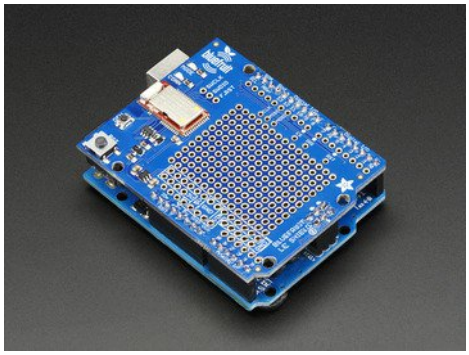
Is uncommented

Configuration!

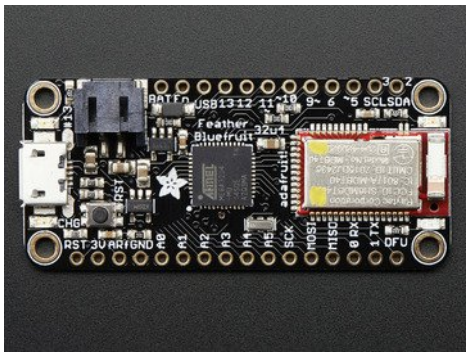
Before you start uploading any of the example sketches, you'll need to CONFIGURE the Bluefruit interface - there's a lot of options so pay close attention!

Which board do you have?

There's a few products under the Bluefruit name:

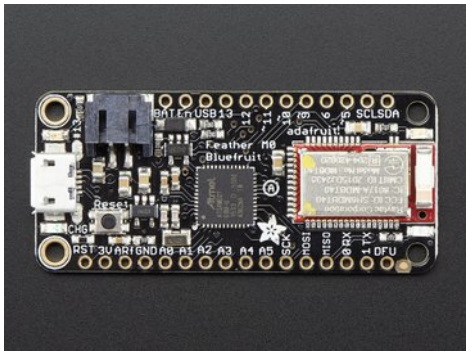
- 

  If you are using the Bluefruit LE Shield then you have an **SPI-connected NRF51822** module. You can use this with **Atmega328** (Arduino UNO or compatible)**, ATmega32u4** (Arduino Leonardo, compatible) or **ATSAMD21** (Arduino Zero, compatible) and possibly others.
  Your pinouts are **Hardware SPI, CS = 8, IRQ = 7, RST = 4**

- 

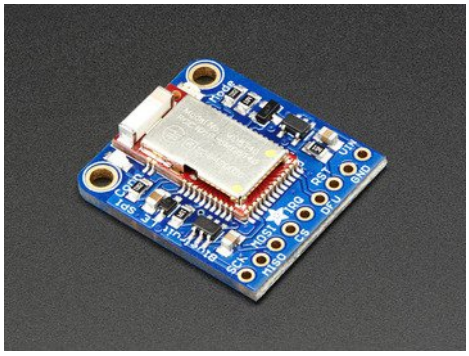
  ## Bluefruit Micro or Feather 32u4 Bluefruit

  If you have a Bluefruit Micro or Feather 32u4 Bluefruit LE then you have an **ATmega32u4** chip with **Hardware SPI**, CS = **8**, IRQ = **7**, RST = **4**

- 

  ## Feather M0 Bluefruit LE

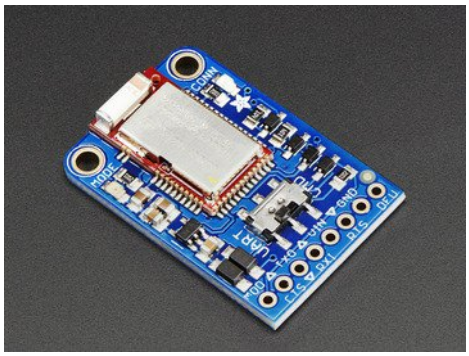
  If you have a Feather M0 Bluefruit LE then you have an **ATSAMD21** chip with **Hardware SPI**, CS = **8**, IRQ = **7**, RST = **4**

- 

  ### Bluefruit LE SPI Friend

  If you have a stand-alone module, you have a bit of flexibility with wiring however we strongly recommend **Hardware SPI**, CS = **8**, IRQ = **7**, RST = **4**

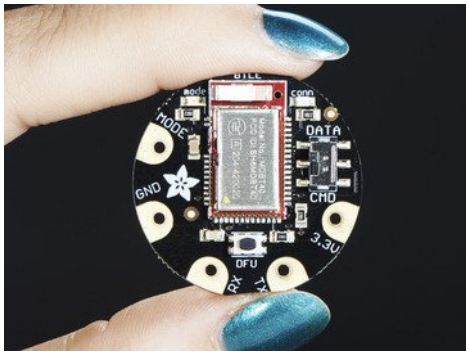
  You can use this with just about any microcontroller with 5 or 6 pins

- 

  ### Bluefruit LE UART Friend or Flora BLE

  If you have a stand-alone UART module you have some flexibility with wiring. However we suggest **hardware UART** if possible. You will likely need to use the flow control **CTS** pin if you are not using hardware UART. There's also a **MODE** pin

  You can use this with just about any microcontroller with at least 3 pins, but best used with a Hardware Serial/UART capable chip!

- 

Configure the Pins Used

You'll want to check the Bluefruit Config to set up the pins you'll be using for UART or SPI

Each example sketch has a secondary tab with configuration details. You'll want to edit and save the sketch to your own documents folder once set up.