



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

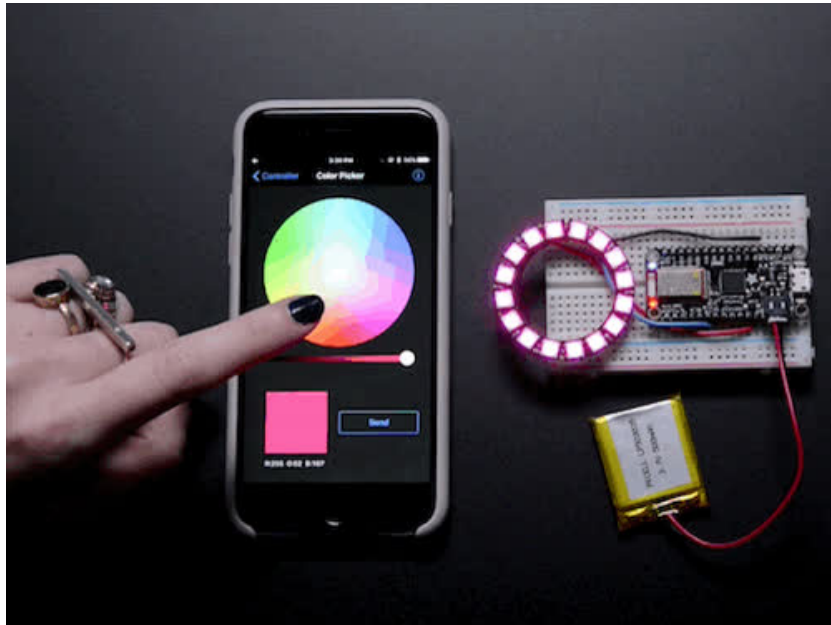
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



□

Adafruit Feather M0 Bluefruit LE

Created by lady ada



Last updated on 2017-01-26 06:43:50 PM UTC

Guide Contents

Guide Contents	2
Overview	10
Pinouts	14
Power Pins	14
Logic pins	15
Bluefruit LE Module + Indicator LEDs	15
Other Pins!	16
SWD Pins	16
Assembly	18
Header Options!	18
Soldering in Plain Headers	20
Prepare the header strip:	20
Add the breakout board:	20
And Solder!	21
Soldering on Female Header	22
Tape In Place	22
Flip & Tack Solder	23
And Solder!	23
Power Management	25
Battery + USB Power	25
Power supplies	25
Measuring Battery	26
ENable pin	26
Arduino IDE Setup	27
https://adafruit.github.io/arduino-board-index/package_adafruit_index.json	28
Using with Arduino IDE	29
Install SAMD Support	29
Install Adafruit SAMD	30
Install Drivers (Windows Only)	30
Blink	31
Sucessful Upload	32
Compilation Issues	32

Manually bootloading	33
Ubuntu & Linux Issue Fix	33
Adapting Sketches to M0	34
Analog References	34
Pin Outputs & Pullups	34
Serial vs SerialUSB	34
AnalogWrite / PWM	34
Missing header files	34
Bootloader Launching	35
Aligned Memory Access	35
Floating Point Conversion	35
How Much RAM Available?	35
Storing data in FLASH	36
Installing BLE Library	37
Install the Adafruit nRF51 BLE Library	37
Run first example	37
Uploading to the Feather Bluefruit LE	38
Compilation Issues	40
Manually bootloading	40
Run the sketch	40
AT command testing	41
Configuration!	43
Which board do you have?	43
Bluefruit Micro or Feather 32u4 Bluefruit	43
Feather M0 Bluefruit LE	43
Bluefruit LE SPI Friend	44
Bluefruit LE UART Friend or Flora BLE	44
Configure the Pins Used	44
Common settings:	45
Software UART	45
Hardware UART	45
Mode Pin	45
SPI Pins	45
Software SPI Pins	45

Select the Serial Bus	45
UART Based Boards (Bluefruit LE UART Friend & Flora BLE)	45
SPI Based Boards (Bluefruit LE SPI Friend)	46
ATCommand	47
Opening the Sketch	47
Configuration	47
Running the Sketch	48
BLEUart	50
Opening the Sketch	50
Configuration	50
Running the Sketch	51
HIDKeyboard	55
Opening the Sketch	55
Configuration	55
Running the Sketch	56
Bonding the HID Keyboard	56
Android	57
iOS	58
OS X	59
Controller	61
Opening the Sketch	61
Configuration	61
Running the Sketch	62
Using Bluefruit LE Connect in Controller Mode	62
Streaming Sensor Data	63
Control Pad Module	64
Color Picker Module	65
HeartRateMonitor	67
Opening the Sketch	67
Configuration	67
If Using Hardware or Software UART	68
Running the Sketch	68
nRF Toolbox HRM Example	69

CoreBluetooth HRM Example	70
UriBeacon	72
Opening the Sketch	72
Configuration	72
Running the Sketch	73
HALP!	74
AT Commands	75
Test Command Mode '=?'	75
Write Command Mode '=xxx'	75
Execute Mode	75
Read Command Mode '?'	76
Standard AT	77
AT	77
ATI	77
ATZ	77
ATE	77
+++	78
General Purpose	79
AT+FACTORYRESET	79
AT+DFU	79
AT+HELP	79
AT+NVMWRITE	79
AT+NVMREAD	80
AT+MODESWITCHEN	80
Hardware	81
AT+BAUDRATE	81
AT+HWADC	81
AT+HWGETDIETEMP	81
AT+HWGPIO	82
AT+HWGPIOMODE	82
AT+HWI2CSCAN	83
AT+HWVBAT	83

AT+HWRANDOM	83
AT+HWMODELED	83
AT+UARTFLOW	84
Beacon	85
AT+BLEBEACON	85
AT+BLEURIBEACON	86
Deprecated: AT+EDDYSTONEENABLE	87
AT+EDDYSTONEURL	87
AT+EDDYSTONECONFIGEN	87
AT+EDDYSTONESEVICEEN	88
AT+EDDYSTONEBROADCAST	88
BLE Generic	89
AT+BLEPOWERLEVEL	89
AT+BLEGETADDRTYPE	89
AT+BLEGETADDR	89
AT+BLEGETPEERADDR	90
AT+BLEGETRSSI	90
BLE Services	91
AT+BLEUARTTX	91
TX FIFO Buffer Handling	91
AT+BLEUARTTXF	92
AT+BLEUARTRX	92
AT+BLEUARTFIFO	93
AT+BLEKEYBOARDEN	93
AT+BLEKEYBOARD	93
AT+BLEKEYBOARDCODE	94
Modifier Values	94
AT+BLEHIDEN	94
AT+BLEHIDMOUSEMOVE	95
AT+BLEHIDMOUSEBUTTON	95
AT+BLEHIDCONTROLKEY	96
AT+BLEHIDGAMEPADEN	96
AT+BLEHIDGAMEPAD	97

AT+BLEMIDIEN	97
AT+BLEMIDIRX	97
AT+BLEMIDITX	98
AT+BLEBATTEN	98
AT+BLEBATTVAL	98
BLE GAP	99
AT+GAPCONNECTABLE	99
AT+GAPGETCONN	99
AT+GAPDISCONNECT	99
AT+GAPDEVNAME	99
AT+GAPDELBONDS	100
AT+GAPINTERVALS	100
AT+GAPSTARTADV	101
AT+GAPSTOPADV	101
AT+GAPSETADVDATA	101
BLE GATT	103
GATT Limitations	103
AT+GATTCLEAR	103
AT+GATTADDSERVICE	103
AT+GATTADDCHAR	104
AT+GATTCHAR	105
AT+GATTLIST	106
AT+GATTCHARRAW	107
Debug	108
AT+DBGMEMRD	108
AT+DBGNVMRD	108
AT+DBGSTACKSIZE	108
AT+DBGSTACKDUMP	108
History	111
Version 0.7.7	111
Version 0.7.0	111
Version 0.6.7	112

Version 0.6.6	112
Version 0.6.5	113
Version 0.6.2	113
Version 0.5.0	113
Version 0.4.7	114
Version 0.3.0	114
Command Examples	115
Heart Rate Monitor Service	115
Python Script	115
SDEP (SPI Data Transport)	118
SDEP Overview	118
SPI Setup	118
SPI Hardware Requirements	118
IRQ Pin	118
SDEP Packet and SPI Error Identifier	118
Sample Transaction	118
SDEP (Simple Data Exchange Protocol)	119
Endianness	119
Message Type Indicator	119
SDEP Data Transactions	119
Message Types	119
Command Messages	119
Response Messages	120
Alert Messages	121
Standard Alert IDs	121
Error Messages	121
Standard Error IDs	122
Existing Commands	122
SDEP AT Wrapper Usage	122
GATT Service Details	124
UART Service	124
UART Service	125
Characteristics	125
TX (0x0002)	125
RX (0x0003)	125
Software Resources	126
Bluefruit LE Client Apps and Libraries	126

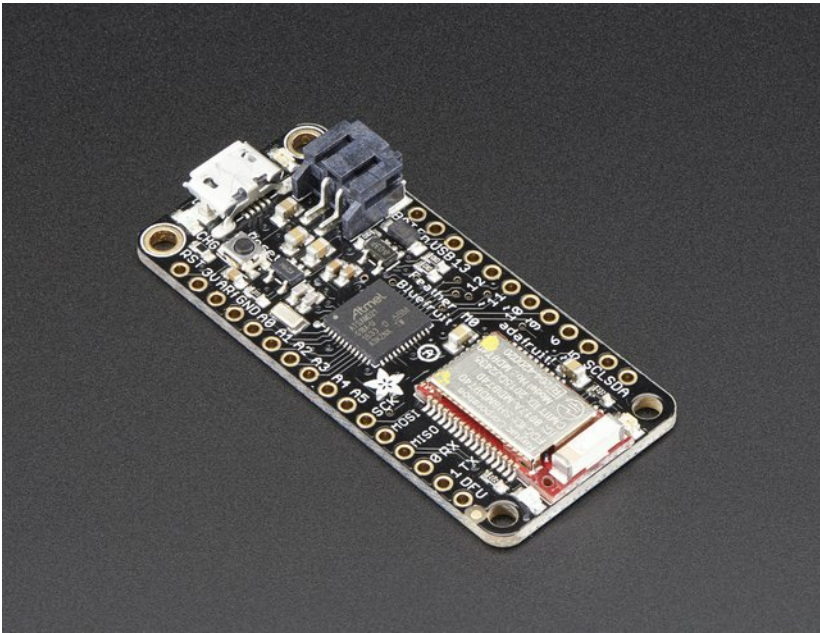
Bluefruit LE Connect (http://adafru.it/f4G) (Android/Java)	126
Bluefruit LE Connect (http://adafru.it/f4H) (iOS/Swift)	126
Bluefruit LE Connect for OS X (http://adafru.it/o9F) (Swift)	126
Bluefruit LE Command Line Updater for OS X (http://adafru.it/pLF) (Swift)	127
Deprecated: Bluefruit Buddy (http://adafru.it/mCn) (OS X)	127
ABLE (http://adafru.it/ijB) (Cross Platform/Node+Electron)	128
Bluefruit LE Python Wrapper (http://adafru.it/fQF)	128
Debug Tools	129
AdaLink (http://adafru.it/fPq) (Python)	129
Adafruit nRF51822 Flasher (http://adafru.it/fVL) (Python)	129
BLE FAQ	130
Bluefruit LE Connect (Android)	131
Nordic nRF Toolbox	131
Adafruit_nRF51822_Flasher	131
DFU Bluefruit Updates	136
Downloads	137
Datasheets	137
Schematic	137
Fab Print	137
Device Recovery	139
How to Recover a Bluefruit Board	139
1. Force DFU Mode at Startup	139
2. Update the Bluefruit Firmware	139
BLEFRIEND32 Firmware (UART, 32KB SRAM)	139
BLESPIFRIEND Firmware (SPI)	139
3. Flash a Test Sketch	139
4. Perform a Factory Reset	139
Still Having Problems?	140
Feather HELP!	141

Overview

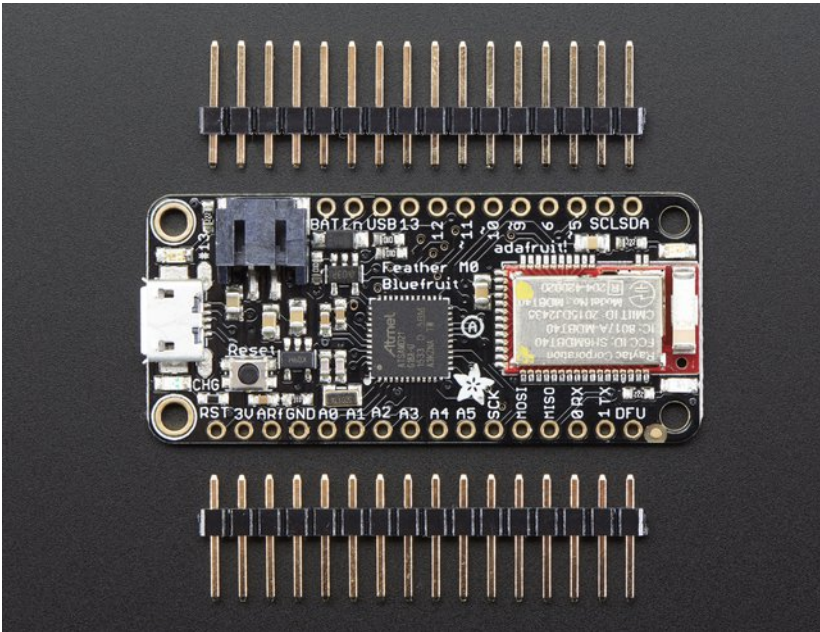
Feather is the new development board from Adafruit, and like it's namesake it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller cores.

This is the **Adafruit Feather M0 Bluefruit** - our take on an 'all-in-one' Cortex M0+ Arduino-compatible + Bluetooth Low Energy with built in USB and battery charging. Its an Adafruit Feather M0 with a BTLE module, ready to rock! [We have other boards in the Feather family, check'em out here \(http://adafru.it/jAQ\)](http://adafru.it/jAQ)

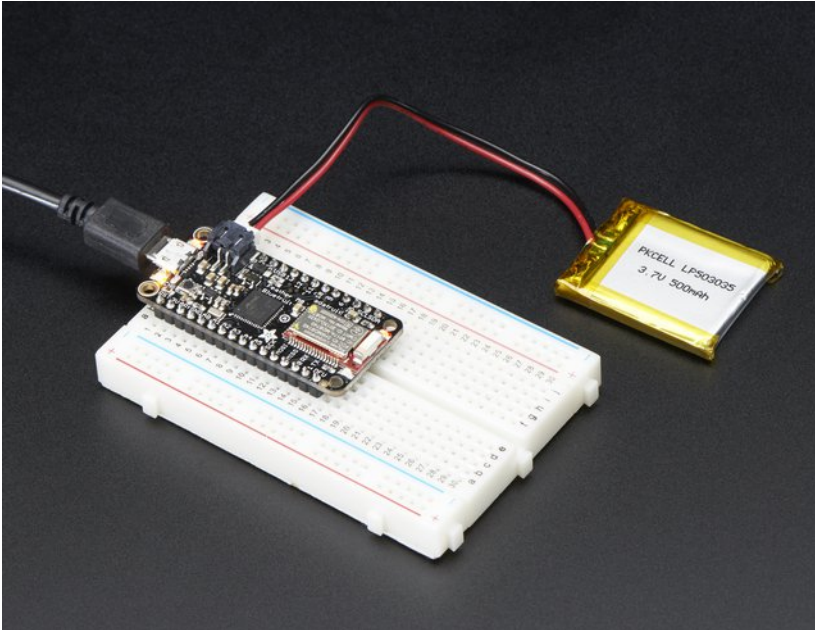
Bluetooth Low Energy is the hottest new low-power, 2.4GHz spectrum wireless protocol. In particular, its the only wireless protocol that you can use with iOS without needing special certification and it's supported by all modern smart phones. This makes it excellent for use in portable projects that will make use of an iOS or Android phone or tablet. It also is supported in Mac OS X and Windows 8+



At the Feather M0's heart is an ATSAM D21G18 ARM Cortex M0 processor, clocked at 48 MHz and at 3.3V logic, the same one used in the new [Arduino Zero \(http://adafru.it/2843\)](http://adafru.it/2843). This chip has a whopping 256K of FLASH (8x more than the Atmega328 or 32u4) and 32K of RAM (16x as much)! This chip comes with built in USB so it has USB-to-Serial program & debug capability built in with no need for an FTDI-like chip.

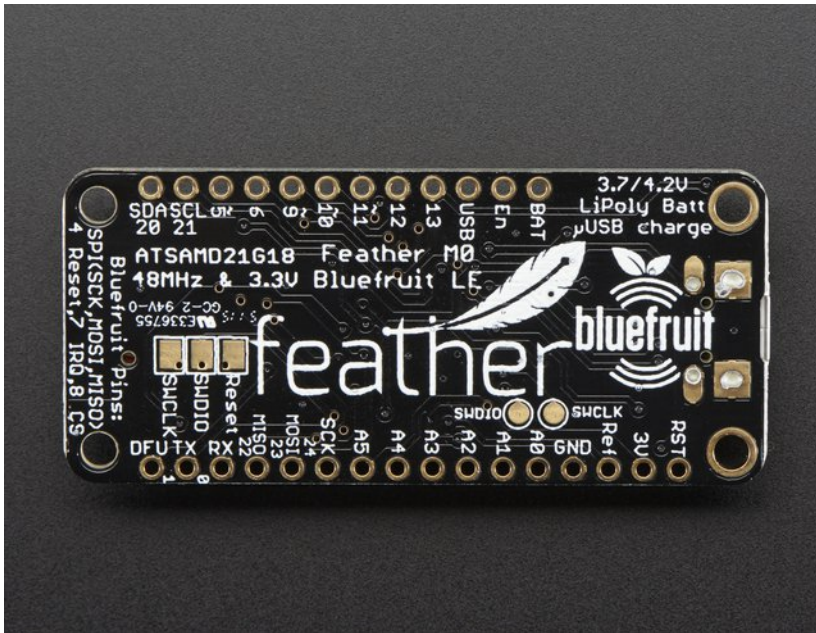


To make it easy to use for portable projects, we added a connector for any of our 3.7V Lithium polymer batteries and built in battery charging. You don't need a battery, it will run just fine straight from the micro USB connector. But, if you do have a battery, you can take it on the go, then plug in the USB to recharge. The Feather will automatically switch over to USB power when its available. We also tied the battery thru a divider to an analog pin, so you can measure and monitor the battery voltage to detect when you need a recharge.

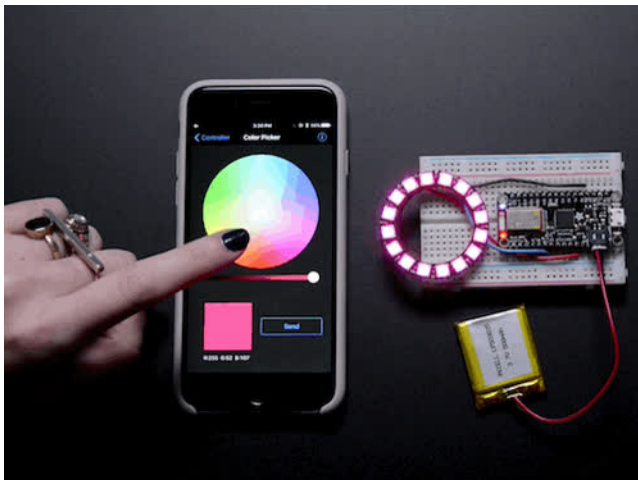


Here's some handy specs! Like all Feather M0's you get:

- Measures 2.0" x 0.9" x 0.28" (51mm x 23mm x 8mm) without headers soldered in
- Light as a (large?) feather - 5.7 grams
- ATSAM21G18 @ 48MHz with 3.3V logic/power
- 256KB of FLASH + 32KB of RAM
- No EEPROM
- 3.3V regulator with 500mA peak current output
- USB native support, comes with USB bootloader and serial port debugging
- You also get tons of pins - 20 GPIO pins
- Hardware Serial, hardware I2C, hardware SPI support
- 8 x PWM pins
- 10 x analog inputs
- Built in 100mA lipoly charger with charging status indicator LED
- Pin #13 red LED for general purpose blinking
- Power/enable pin
- 4 mounting holes
- Reset button



The **Feather M0 Bluefruit** uses the extra space left over to add our excellent Bluefruit BTLE module + two status indicator LEDs



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 \(http://adafruit.it/iCi\)](http://adafruit.it/iCi) or [Android App \(http://adafruit.it/f4G\)](http://adafruit.it/f4G), or [write your own to communicate with the UART service \(http://adafruit.it/iCF\)](http://adafruit.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 \(http://adafruit.it/iCG\)](http://adafruit.it/iCG), you have full control over how the device behaves, including the ability to define and manipulate your own [GATT Services and Characteristics \(http://adafruit.it/iCH\)](http://adafruit.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 \(http://adafruit.it/iCi\)](http://adafruit.it/iCi) or [Android App \(http://adafruit.it/f4G\)](http://adafruit.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 \(http://adafruit.it/iCI\)](http://adafruit.it/iCI), [quaternion/accelerometer/gyro/magnetometer or location \(GPS\) \(http://adafruit.it/iCI\)](http://adafruit.it/iCI), and an 8-button [control game pad \(http://adafruit.it/iCI\)](http://adafruit.it/iCI). This data can be read over BLE and piped into the ATmega32u4 chip for processing & control

You can do a lot more too!

- [The Bluefruit can also act like an HID Keyboard \(http://adafruit.it/iOA\)](http://adafruit.it/iOA) (for devices that support BLE HID)
- [Can become a BLE Heart Rate Monitor \(http://adafruit.it/iOB\)](http://adafruit.it/iOB) (a standard profile for BLE) - you just need to add the pulse-detection circuitry

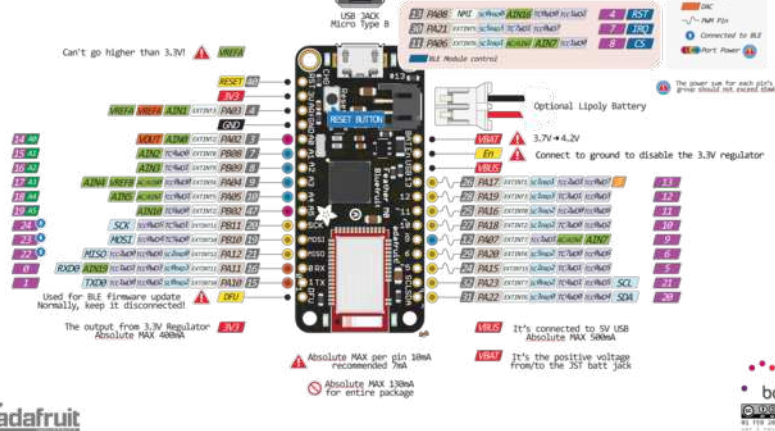
- [Turn it into a UriBeacon \(http://adafru.it/iOC\)](http://adafru.it/iOC), the Google standard for Bluetooth LE beacons. Just power it and the 'Friend will bleep out a URL to any nearby devices with the UriBeacon app installed.
- [Built in over-the-air bootloading capability so we can keep you updated with the hottest new firmware\(http://adafru.it/iOD\)](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.

Comes fully assembled and tested, with a USB bootloader that lets you quickly use it with the Arduino IDE. We also toss in some header so you can solder it in and plug into a solderless breadboard. **Lipoly battery, breadboard and USB cable not included** (but we do have lots of options in the shop if you'd like!)

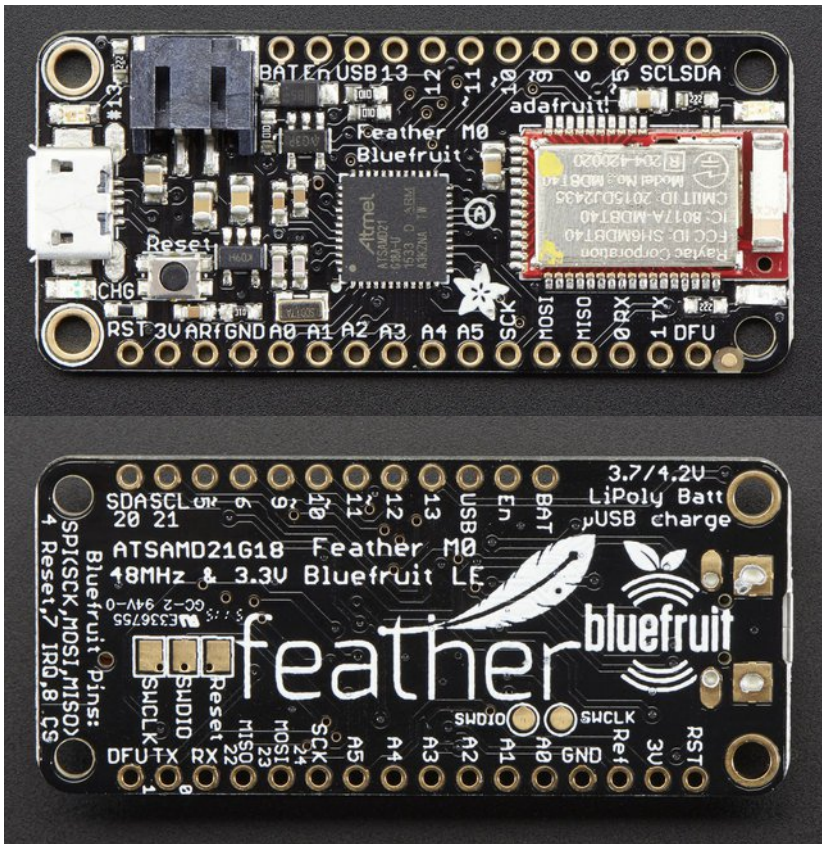
Pinouts



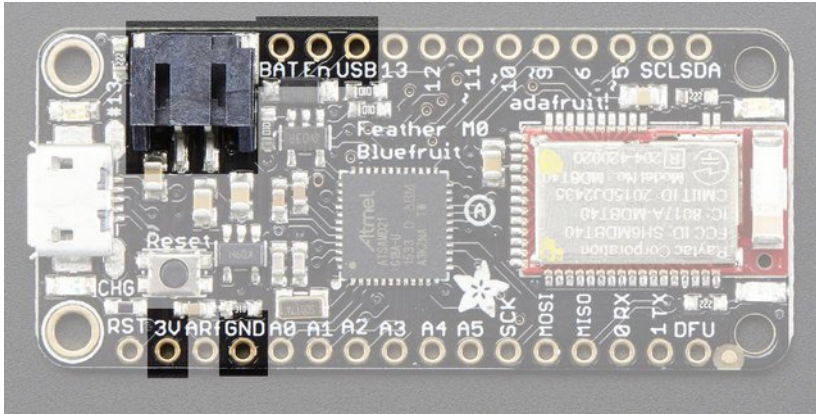
https://www.adafruit.com/products/2995



The Feather M0 Bluefruit is chock-full of microcontroller goodness. There's also a lot of pins and ports. We'll take you a tour of them now!



Power Pins



- **GND** - this is the common ground for all power and logic
- **BAT** - this is the positive voltage to/from the JST jack for the optional Lipoly battery
- **USB** - this is the positive voltage to/from the micro USB jack if connected
- **EN** - this is the 3.3V regulator's enable pin. It's pulled up, so connect to ground to disable the 3.3V regulator
- **3V** - this is the output from the 3.3V regulator, it can supply 500mA peak

Logic pins

This is the general purpose I/O pin set for the microcontroller.

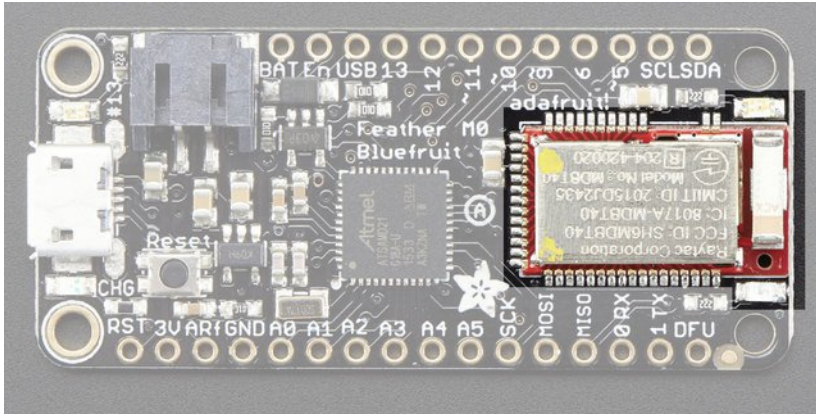
All logic is 3.3V

All pins can do PWM output

All pins can be interrupt inputs

- **#0 / RX** - GPIO #0, also receive (input) pin for **Serial1** (hardware UART), also can be analog input
- **#1 / TX** - GPIO #1, also transmit (output) pin for **Serial1**, also can be analog input
- **#20 / SDA** - GPIO #20, also the I2C (Wire) data pin. There's no pull up on this pin by default so when using with I2C, you may need a 2.2K-10K pullup.
- **#21 / SCL** - GPIO #21, also the I2C (Wire) clock pin. There's no pull up on this pin by default so when using with I2C, you may need a 2.2K-10K pullup.
- **#5** - GPIO #5
- **#6** - GPIO #6
- **#9** - GPIO #9, also analog input **A7**. This analog input is connected to a voltage divider for the lipoly battery so be aware that this pin naturally 'sits' at around 2VDC due to the resistor divider
- **#10** - GPIO #10
- **#11** - GPIO #11
- **#12** - GPIO #12
- **#13** - GPIO #13 and is connected to the **red LED** next to the USB jack
- **A0** - This pin is analog *input* **A0** but is also an analog *output* due to having a DAC (digital-to-analog converter). You can set the raw voltage to anything from 0 to 3.3V, unlike PWM outputs this is a true analog output
- **A1 thru A5** - These are each analog input as well as digital I/O pins.
- **SCK/MOSI/MISO** (GPIO 24/23/22) - These are the hardware SPI pins, you can use them as everyday GPIO pins (but recommend keeping them free as they are best used for hardware SPI connections for high speed and are shared with the BLE)

Bluefruit LE Module + Indicator LEDs



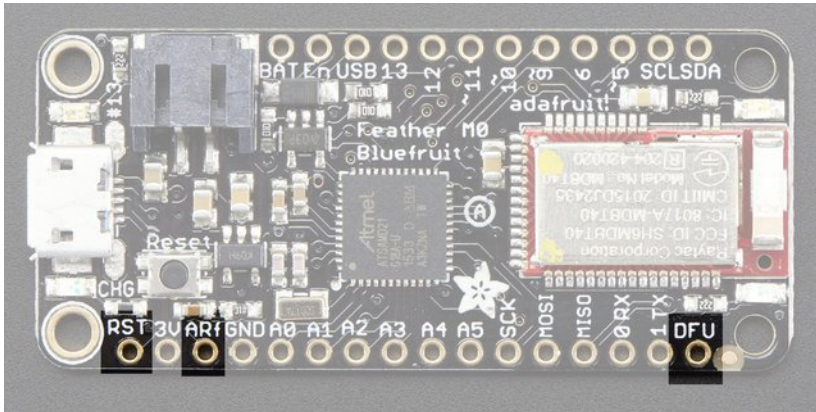
Since not all pins can be brought out to breakouts, due to the small size of the Feather, we use these to control the BLE module

- **#8** - used as the Bluefruit **CS** (chip select) pin
- **#7** - used as the Bluefruit **IRQ** (interrupt request) pin.
- **#4** - used as the Bluefruit **Reset** pin

Since these are not brought out there should be no risk of using them by accident!

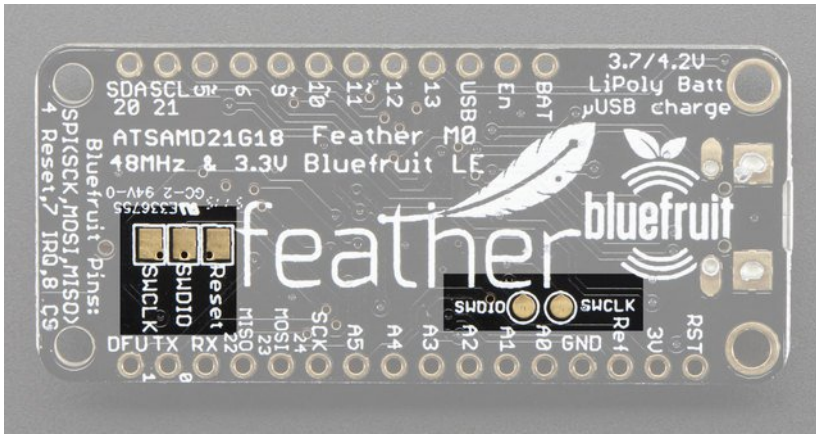
Other Pins!

- **RST** - this is the Reset pin, tie to ground to manually reset the ATSAM, as well as launch the bootloader manually
- **AREf** - the analog reference pin. Normally the reference voltage is the same as the chip logic voltage (3.3V) but if you need an alternative analog reference, connect it to this pin and select the external AREF in your firmware. Can't go higher than 3.3V!
- **DFU** - this is the force-DFU (device firmware upgrade) pin for over-the-air updates to the Bluefruit module. You probably don't need to use this but its available if you need to upgrade! Check out the **DFU Bluefruit Upgrades** page for how to use it. Otherwise, keep it disconnected.



SWD Pins

There's **two** sets of SWD pins. These are used for program/debug of the two processors on the Feather.



The round pads on the right are for the ATSAMD21G18 (main processor). The rectangular pads to the left are for the nrf51822 inside the BLE module.

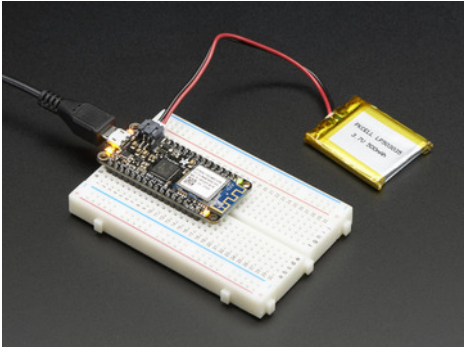
You cannot connect these together to debug both at the same time!

Assembly

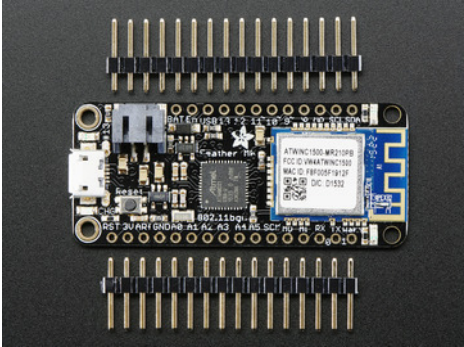
We ship Feathers fully tested but without headers attached - this gives you the most flexibility on choosing how to use and configure your Feather

Header Options!

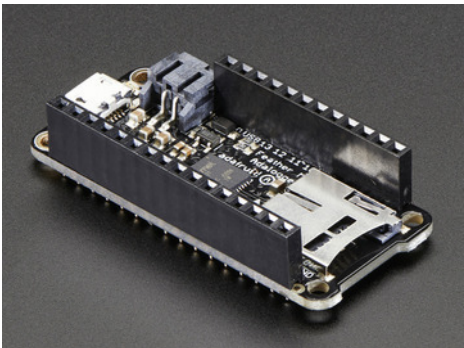
Before you go gung-ho on soldering, there's a few options to consider!



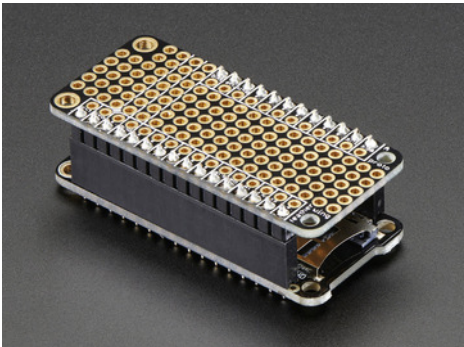
- The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard



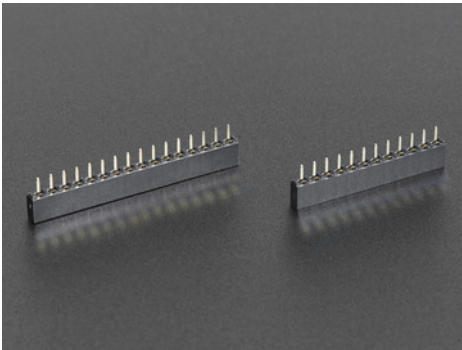
-



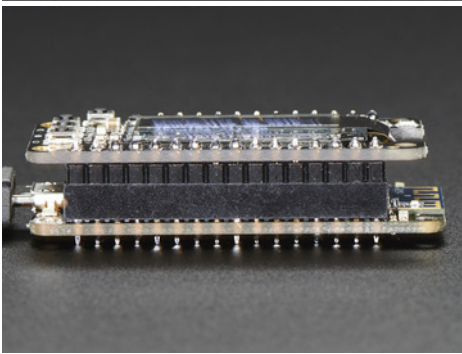
- Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily



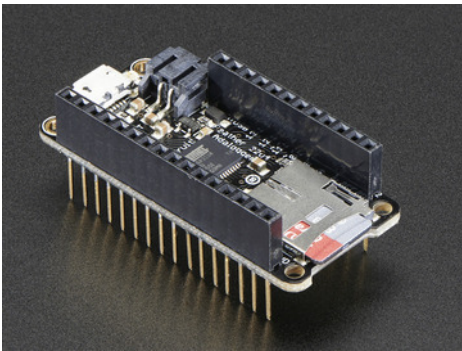
-



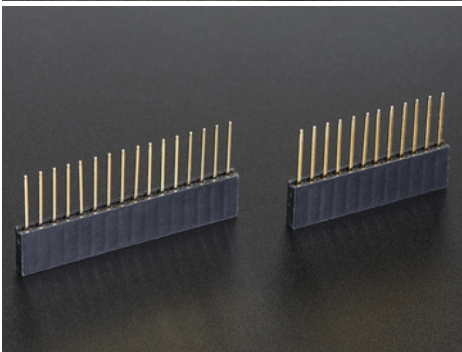
- We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape



-

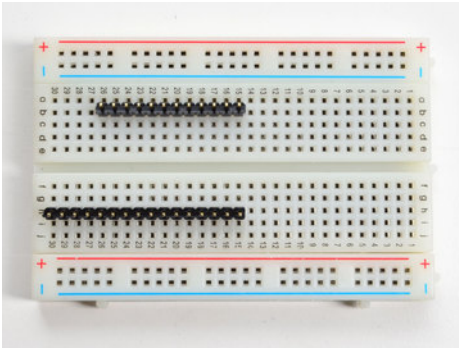


-



- Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a solderless breadboard *and* plug a featherwing on top. But its a little bulky

Soldering in Plain Headers

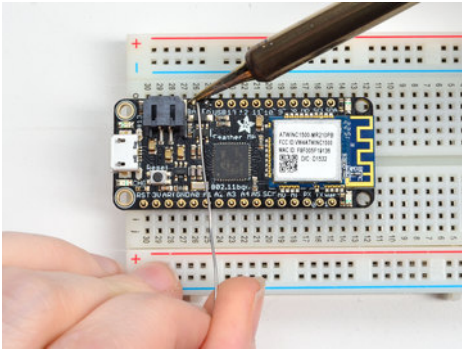


Prepare the header strip:

Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**

Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout

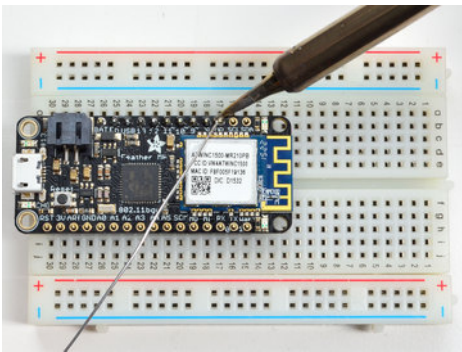
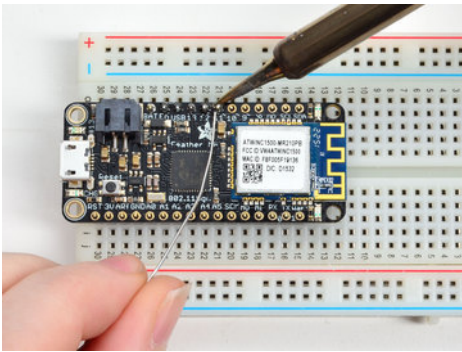


pads

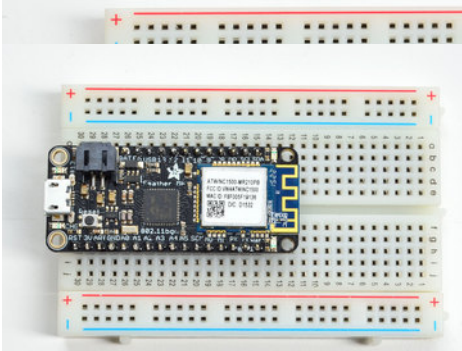
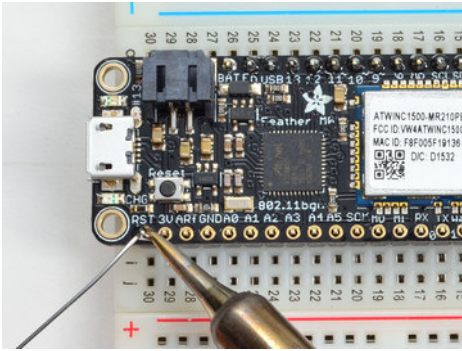
And Solder!

Be sure to solder all pins for reliable electrical contact.

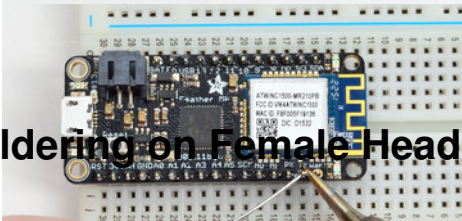
(For tips on soldering, be sure to check out our [Guide to Excellent Soldering](http://adafru.it/aTk) (<http://adafru.it/aTk>)).



Solder the other strip as well.



You're done! Check your solder joints visually and continue onto the next steps

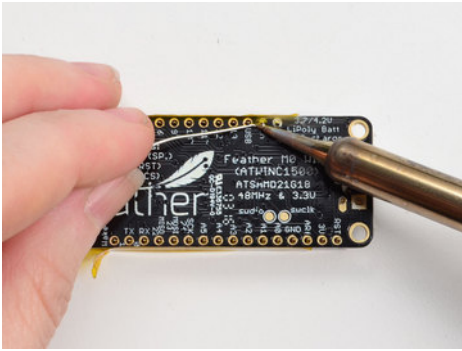


Soldering on Female Header



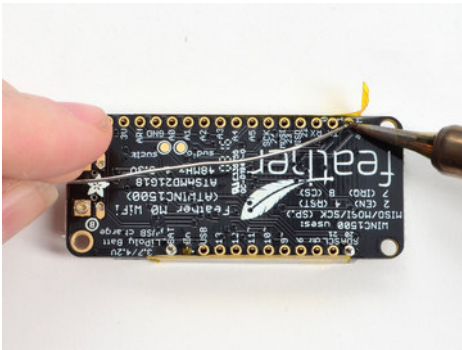
Tape In Place

For sockets you'll want to tape them in place so when you flip over the board they don't fall out



Flip & Tack Solder

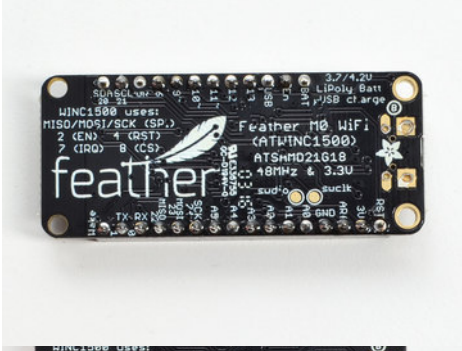
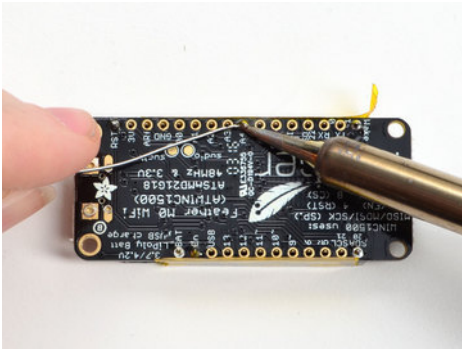
After flipping over, solder one or two points on each strip, to 'tack' the header in place



And Solder!

Be sure to solder all pins for reliable electrical contact.

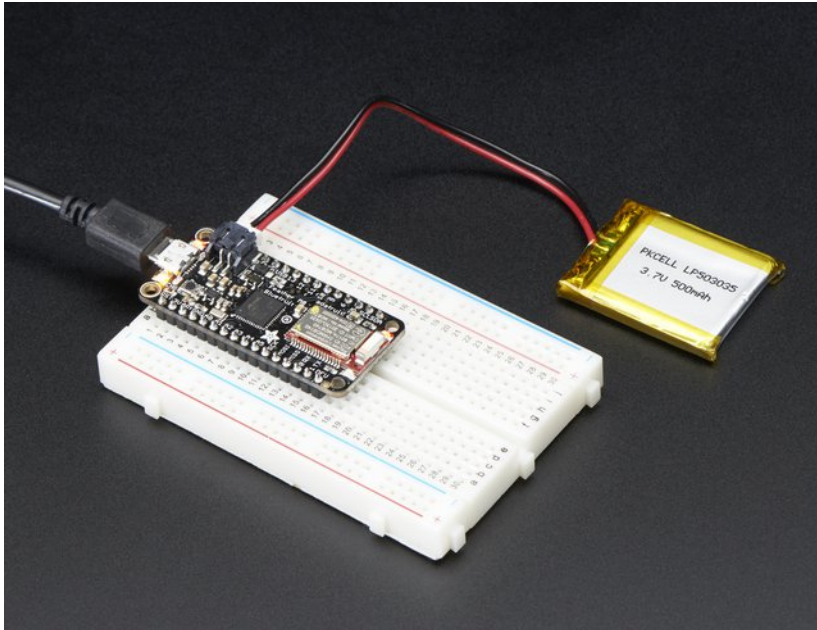
(For tips on soldering, be sure to check out our [Guide to Excellent Soldering](http://adafru.it/aTk) (<http://adafru.it/aTk>)).



You're done! Check your solder joints visually and continue onto the next steps



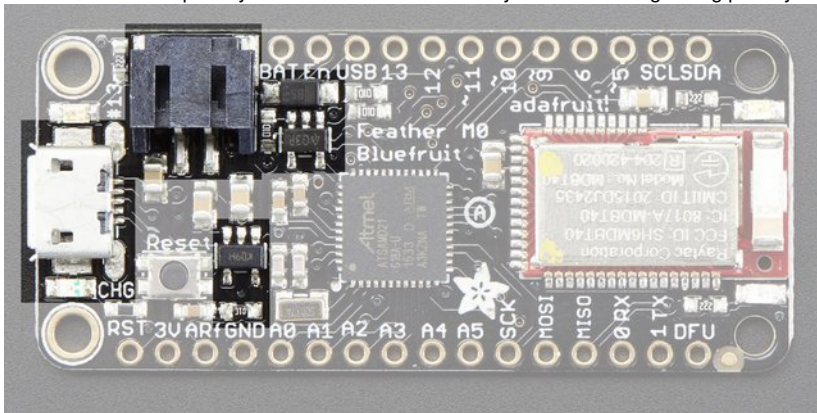
Power Management



Battery + USB Power

We wanted to make the Feather easy to power both when connected to a computer as well as via battery. There's **two ways to power** a Feather. You can connect with a MicroUSB cable (just plug into the jack) and the Feather will regulate the 5V USB down to 3.3V. You can also connect a 4.2/3.7V Lithium Polymer (Lipo/Lipoly) or Lithium Ion (Lilon) battery to the JST jack. This will let the Feather run on a rechargeable battery. **When the USB power is powered, it will automatically switch over to USB for power, as well as start charging the battery (if attached) at 100mA.** This happens 'hotswap' style so you can always keep the Lipoly connected as a 'backup' power that will only get used when USB power is lost.

The JST connector polarity is matched to Adafruit LiPoly batteries. Using wrong polarity batteries can destroy your Feather



The above shows the Micro USB jack (left), Lipoly JST jack (top left), as well as the 3.3V regulator and changeover diode (just to the right of the JST jack) and the Lipoly charging circuitry (to the right of the Reset button). There's also a **CHG** LED, which will light up while the battery is charging. This LED might also flicker if the battery is not connected.

Power supplies

You have a lot of power supply options here! We bring out the **BAT** pin, which is tied to the lipoly JST connector, as well as **USB** which is the +5V from USB if connected. We also have the **3V** pin which has the output from the 3.3V regulator. We use a 500mA peak regulator. While you can get 500mA from it, you can't do it continuously from 5V as it will overheat the regulator. It's fine for, say, powering an ESP8266 WiFi chip or XBee radio though, since the current draw is 'spikey' & sporadic.