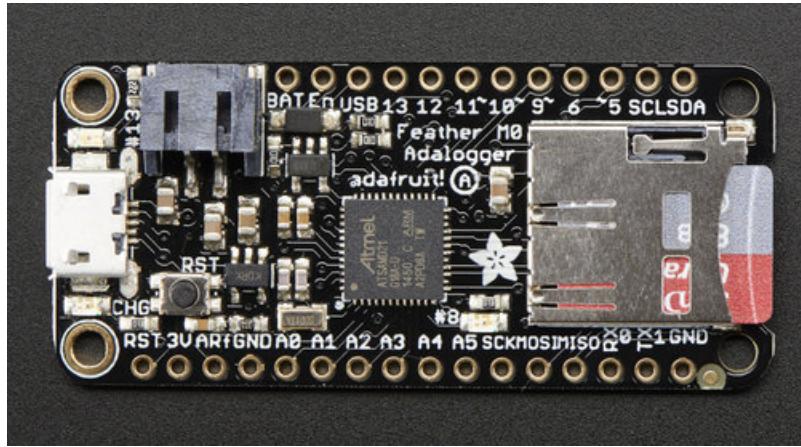




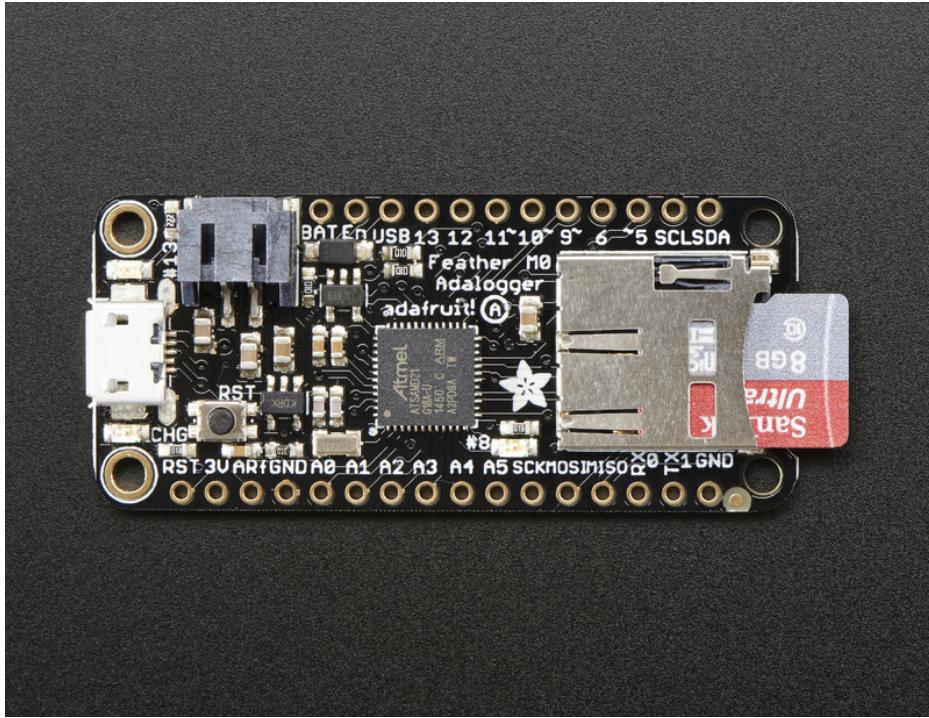
## Adafruit Feather M0 Adalogger

Created by lady ada



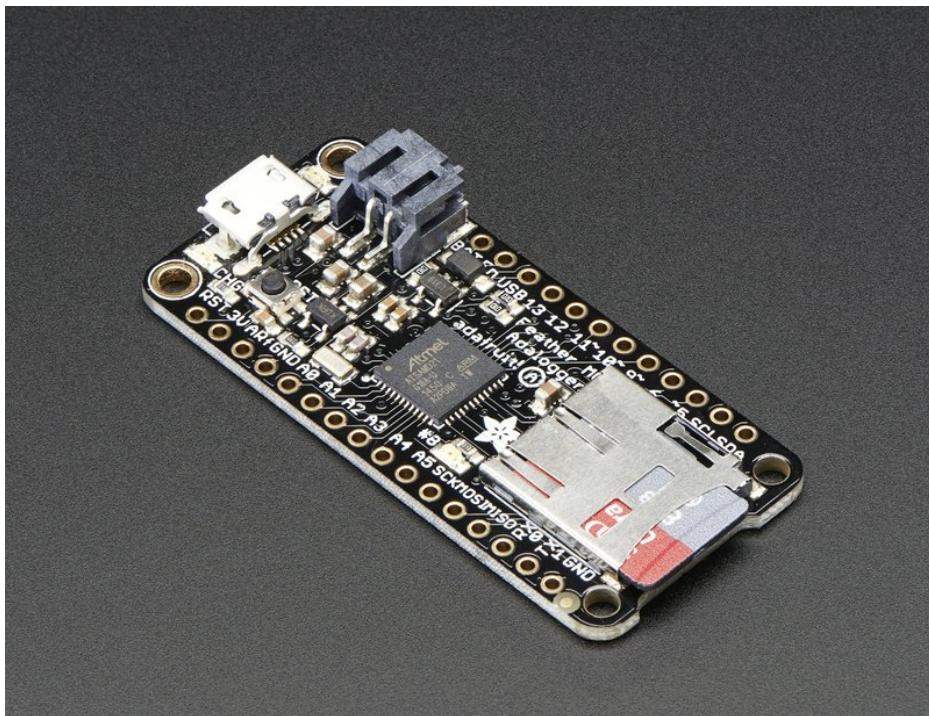
Last updated on 2020-09-30 11:09:13 AM EDT

# Overview

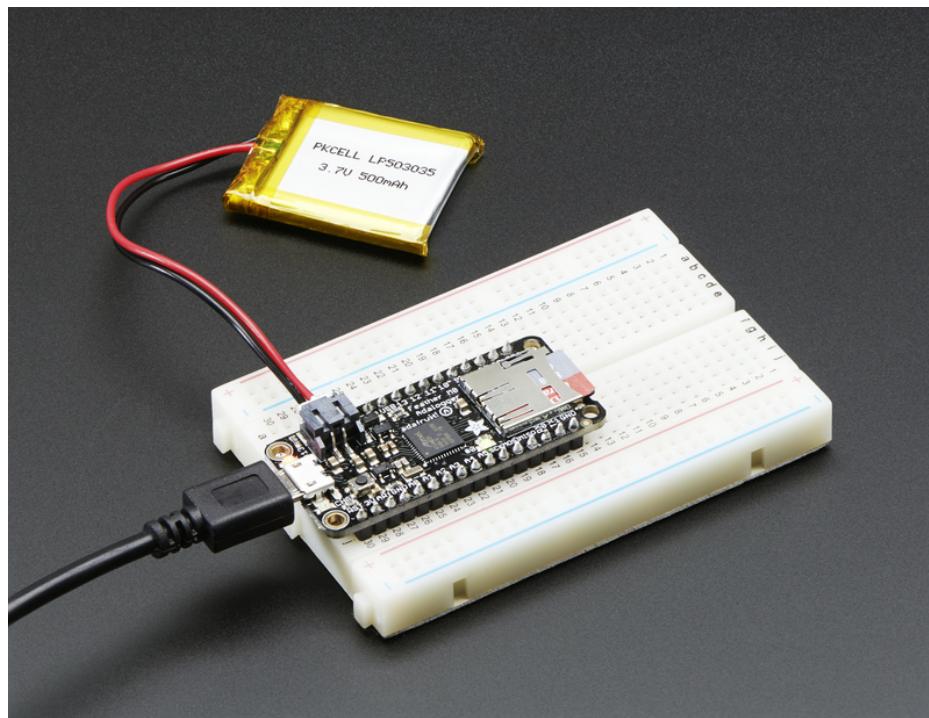


Feather is the new development board from Adafruit, and like its namesake it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller cores.

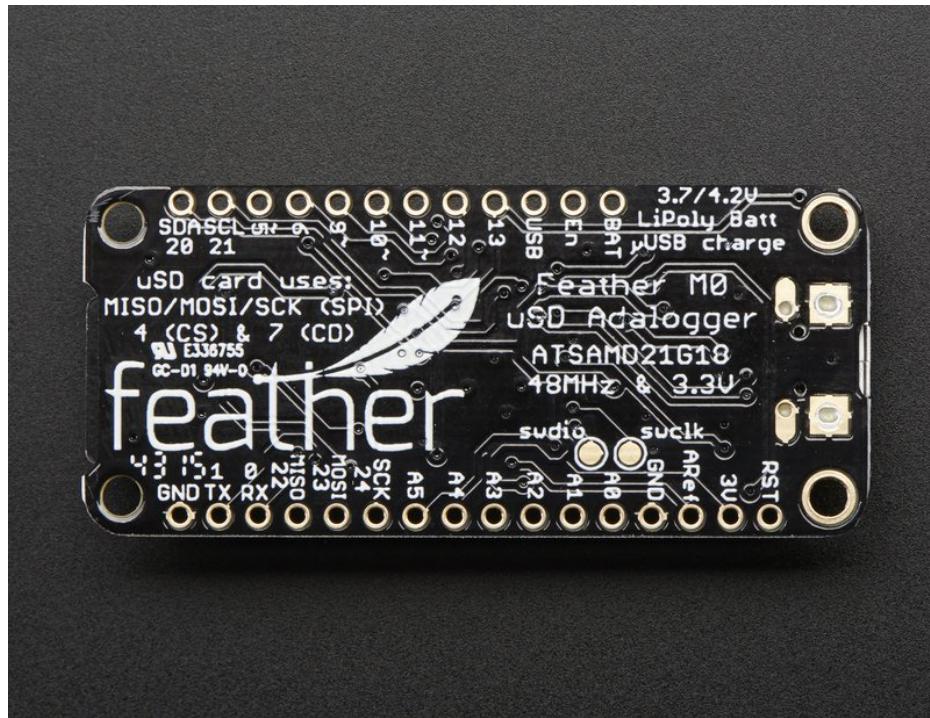
This is the **Adafruit Feather M0 Adalogger** - our take on an 'all-in-one' Cortex M0 datalogger (or data-reader) with built in USB and battery charging. Its an Adafruit Feather M0 with a microSD holder ready to rock! [We have other boards in the Feather family, check'em out here \(<https://adafru.it/jAQ>\)](https://adafru.it/jAQ)



At the Feather M0's heart is an ATSAMD21G18 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.

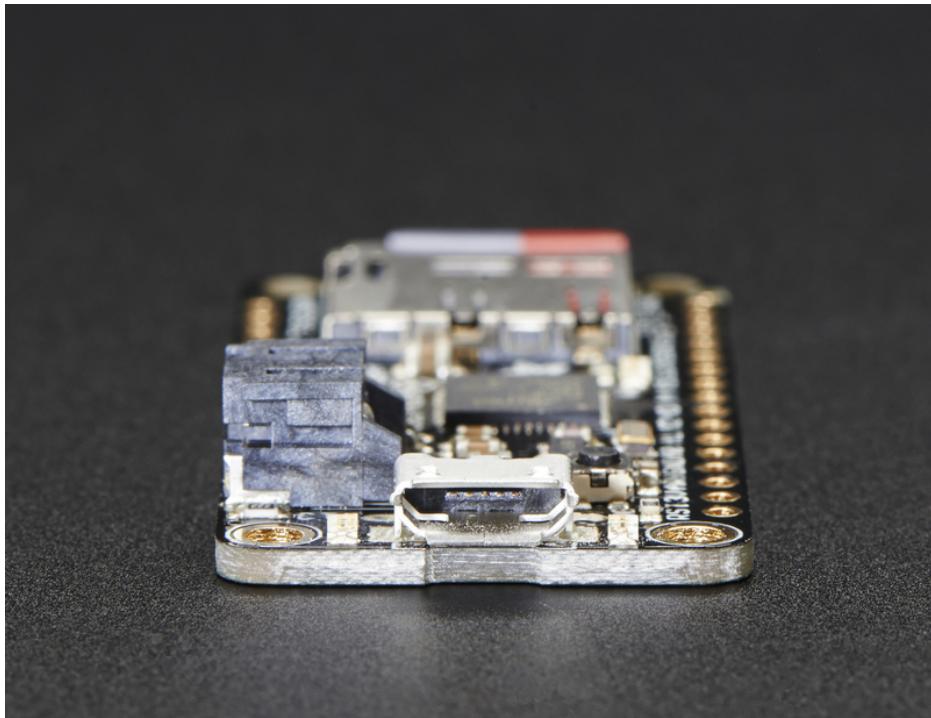


Here's some handy specs! Like all Feather M0's you get:

- Measures 2.0" x 0.9" x 0.28" (51mm x 23mm x 8mm) without headers soldered in
- Light as a (large?) feather - 5.3 grams
- ATSAMD21G18 @ 48MHz with 3.3V logic/power
- 256KB of FLASH + 32KB of RAM
- 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 Adalogger** uses the extra space left over to add MicroSD + a green LED:

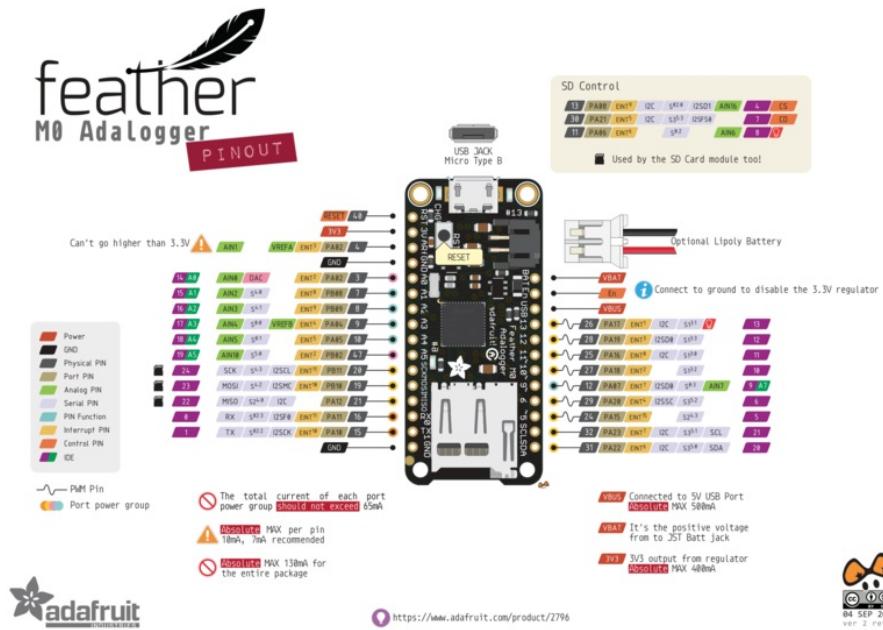
- Pin #8 green LED for your blinking pleasure
- MicroSD card holder for adding as much storage as you could possibly want, for reading or writing.



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, MicroSD card and USB cable not included** (but we do have lots of options in the shop if you'd like!)

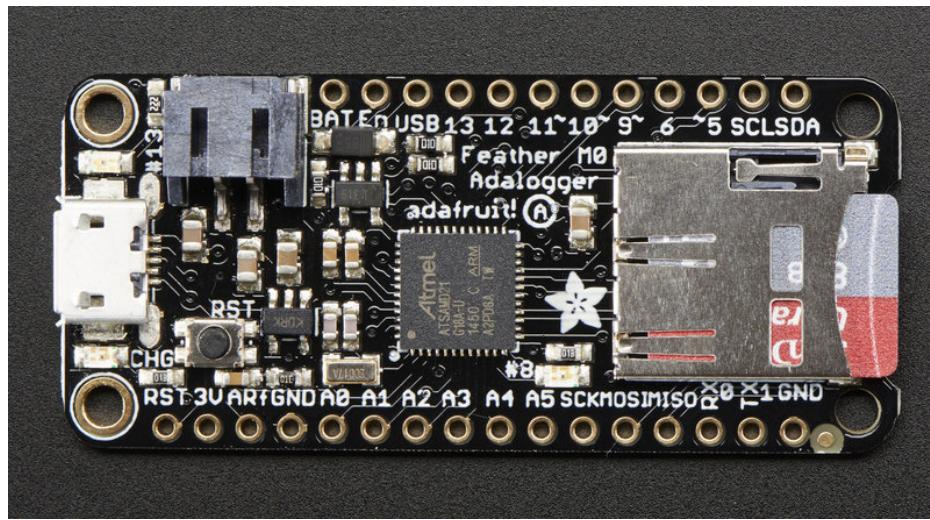
Check out our tutorial for all sorts of details, including schematics, files, IDE instructions, and more!

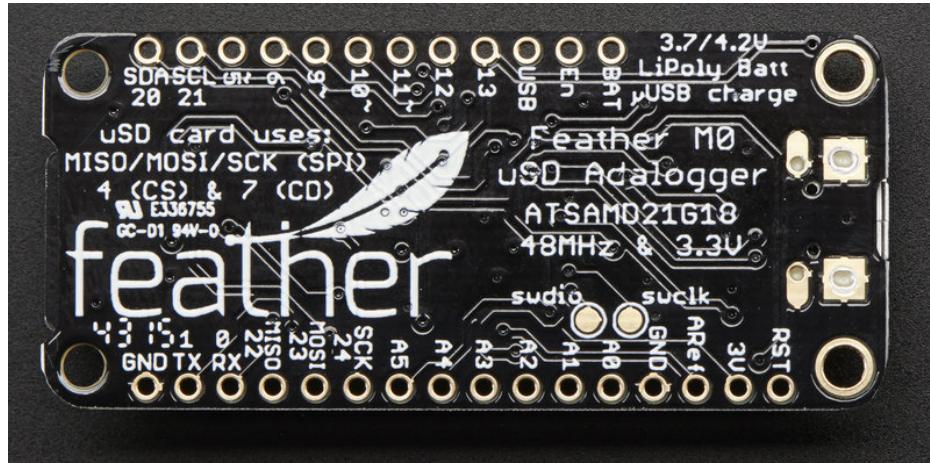
# Pinouts



Note AREF in the diagram should be marked PA03 not PA02

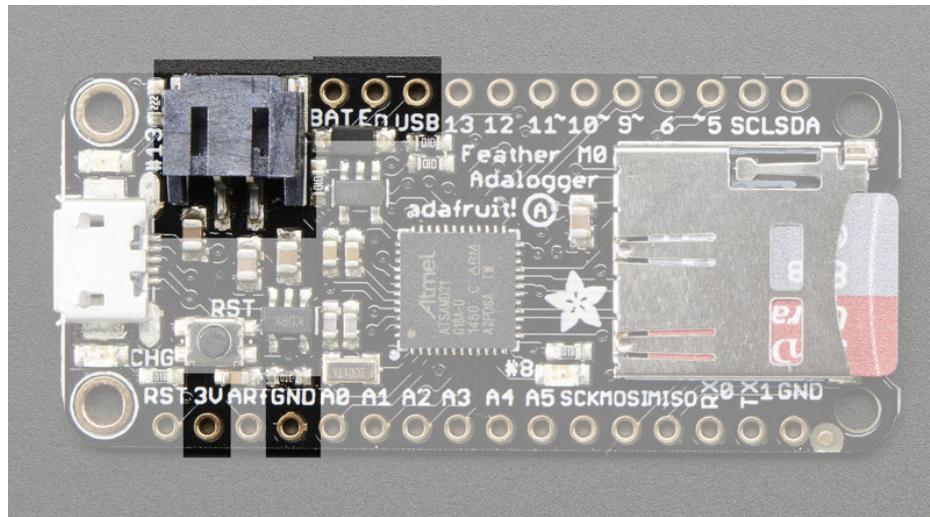
The Feather M0 Adalogger is chock-full of microcontroller goodness. There's also a lot of pins and ports. We'll take you a tour of them now!





## 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

Nearly all pins can do PWM output

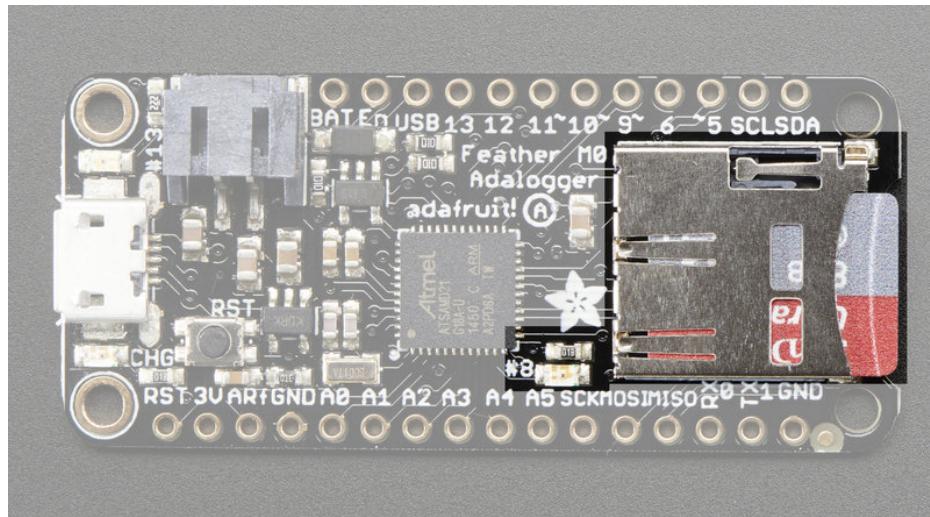
All pins can be interrupt inputs

- **#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).

## Micro SD Card + Green LED

---



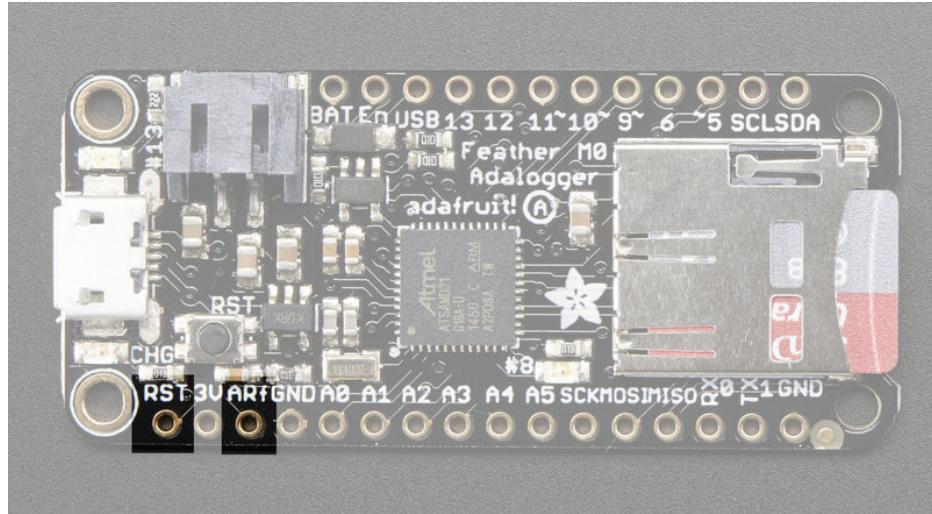
Since not all pins can be brought out to breakouts, due to the small size of the Feather, we use these to control the SD card!

- #4 - used as the MicroSD card **CS** (chip select) pin
- #7 - used as the MicroSD card **CD** (card detect) pin. If you want to detect when a card is inserted/removed, configure this pin as an input with a pullup. When the pin reads low (0V) then there is no card inserted. When the pin reads high, then a card is in place. It will not tell you if the card is valid, its just a mechanical switch
- #8 - This pin was also left over, so we tied it to a green LED, its next to the SD card. It might be handy to blink this LED when writing / reading valid data or some other user-alert!

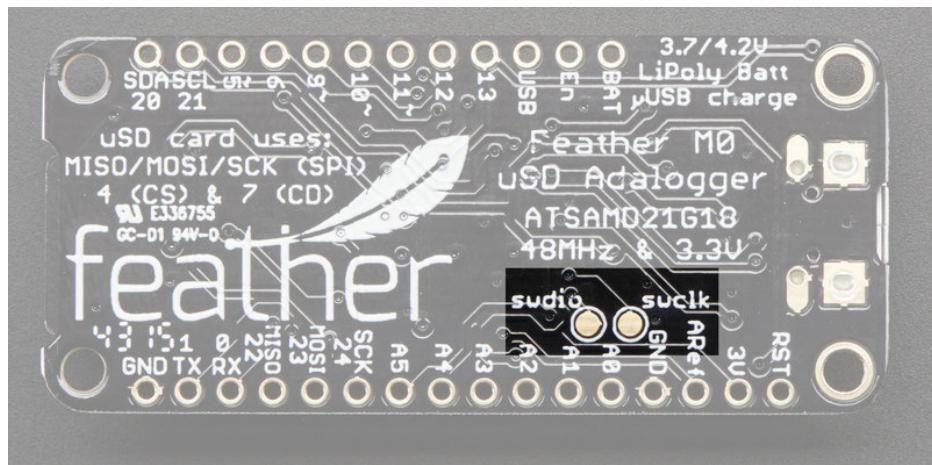
## Other Pins!

---

- **RST** - this is the Reset pin, tie to ground to manually reset the AVR, 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!



**SWCLK & SWDIO** - These pads on the bottom are used to program the chip. They can also be connected to an SWD debugger.



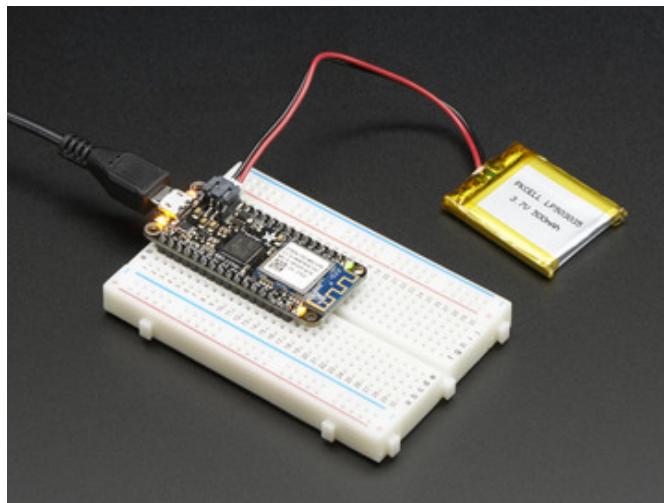
# Assembly

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

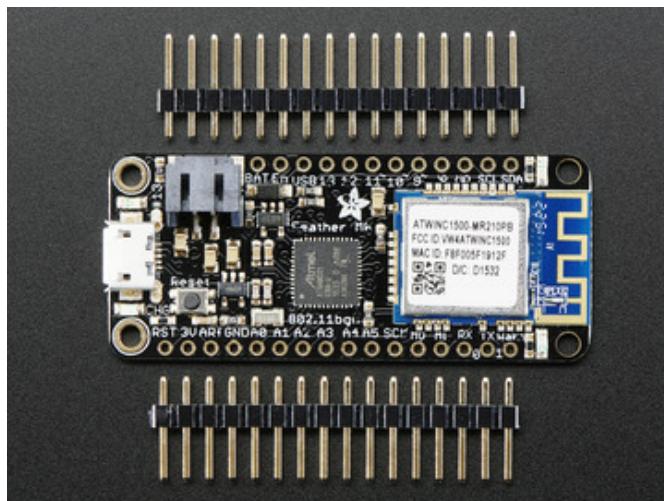
## Header Options!

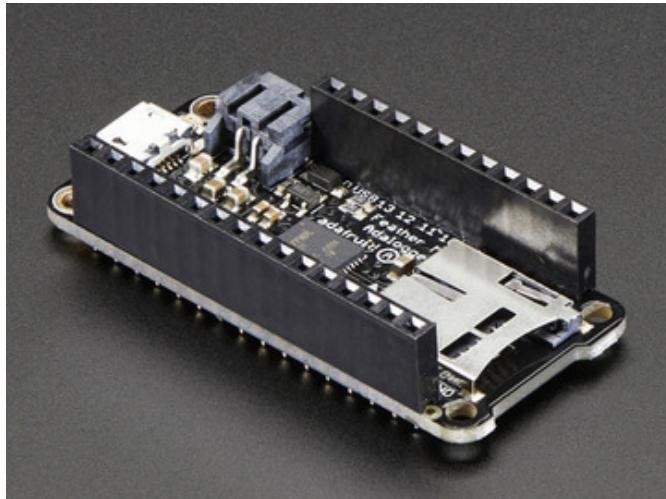
---

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

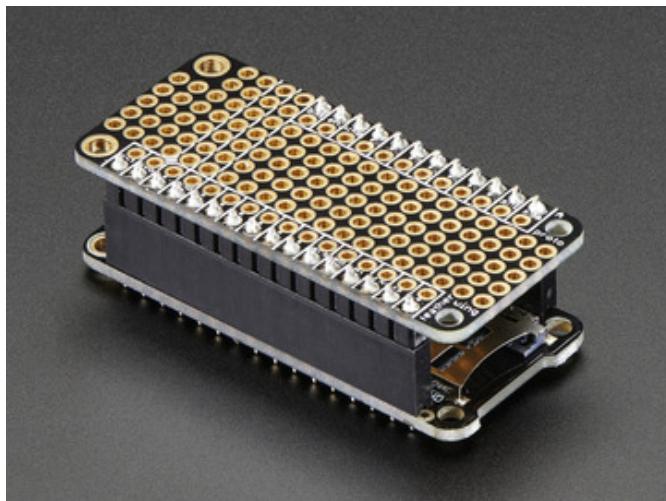


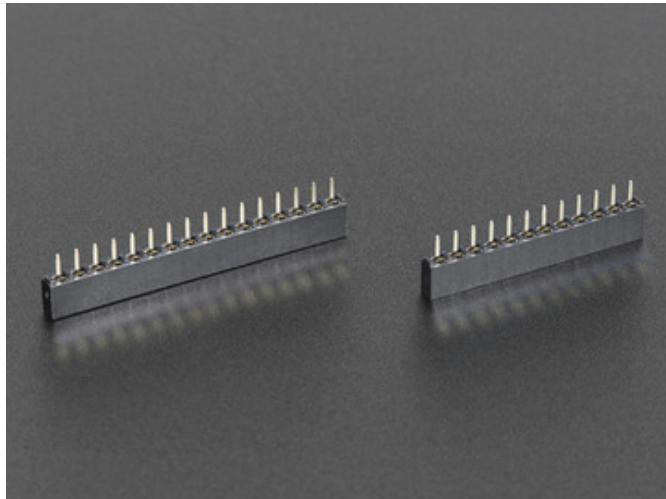
The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard



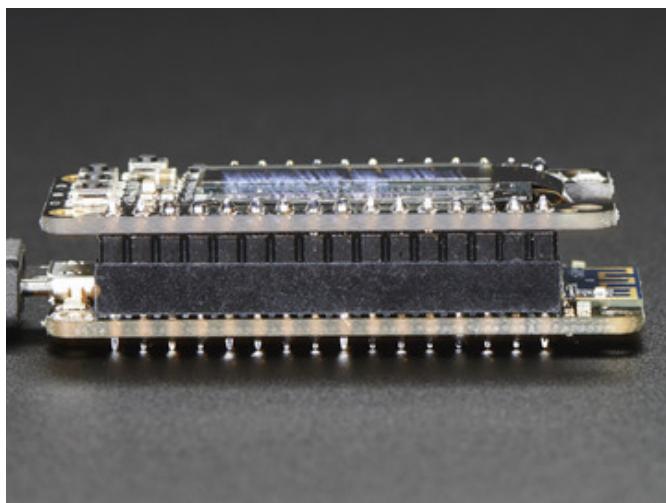


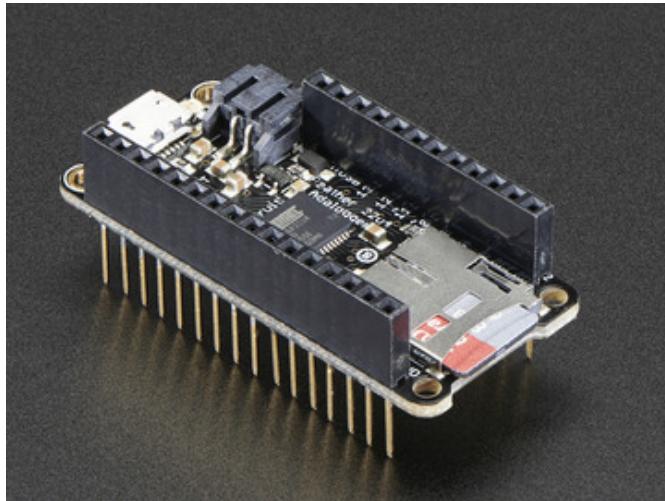
Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily



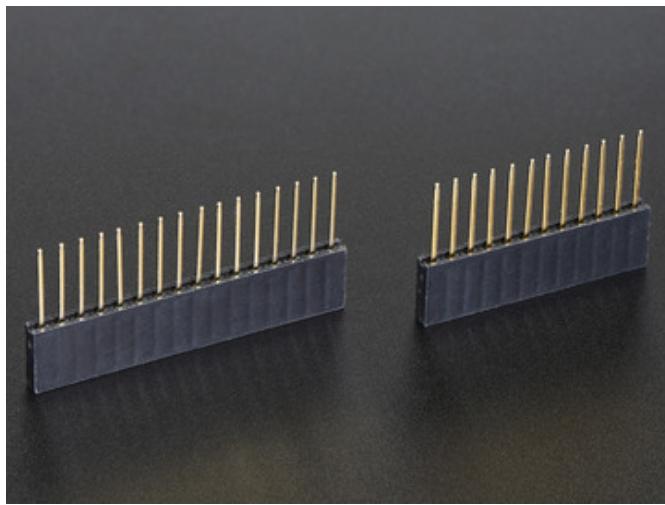


We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape

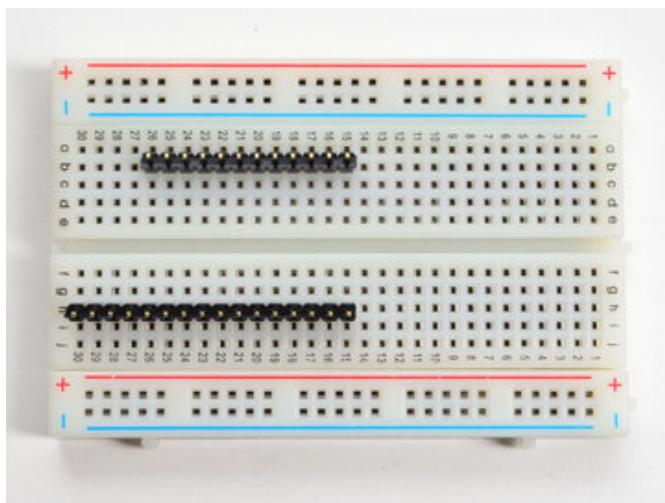




Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a solderless breadboard *and* plug a featherwing on top. But it's a little bulky



## Soldering in Plain Headers

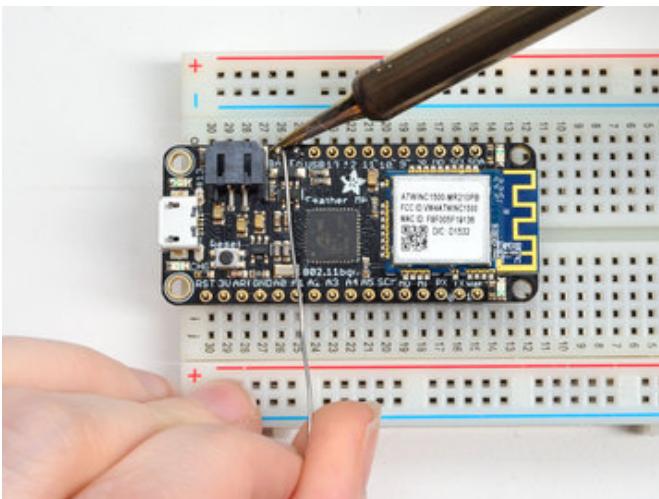


Prepare the header strip:

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

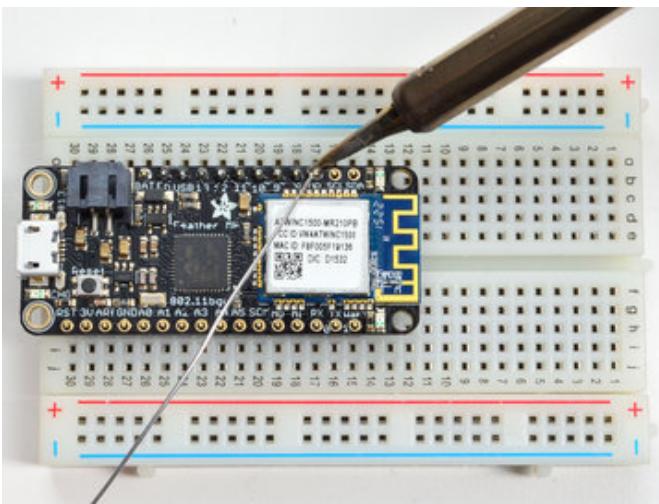
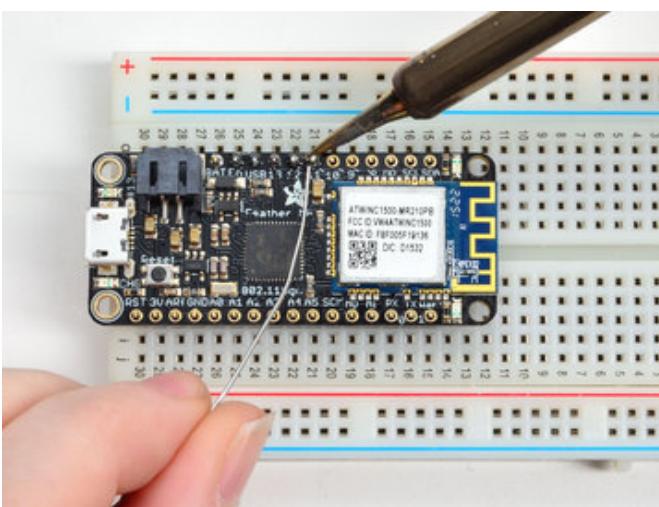


pins poke through the breakout pads

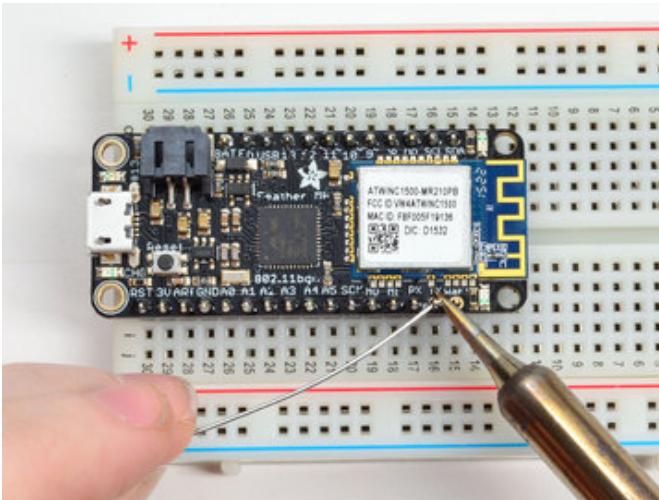
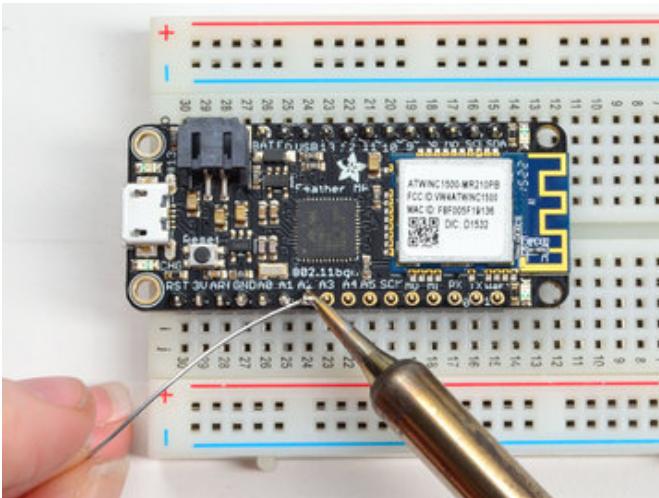
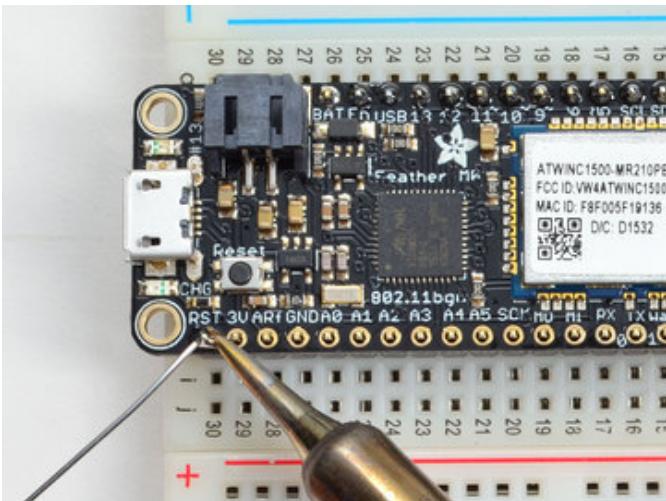
And Solder!

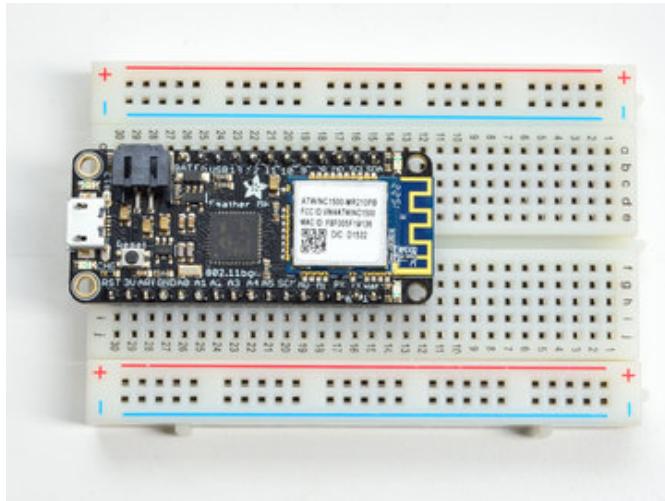
Be sure to solder all pins for reliable electrical contact.

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



Solder the other strip as well.





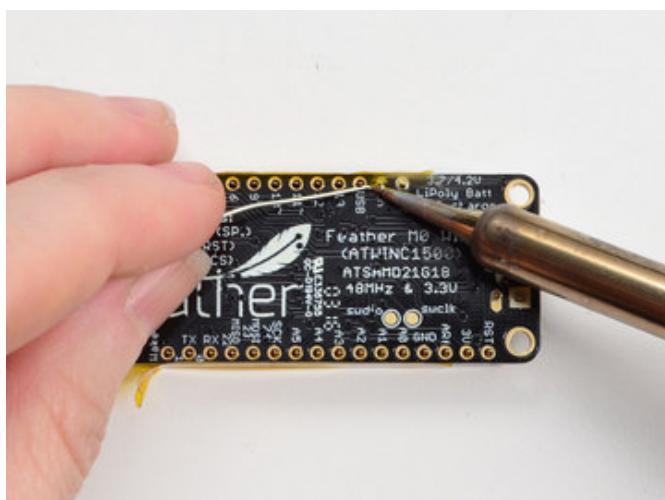
You're done! Check your solder joints visually and continue onto the next steps

## Soldering on Female Header



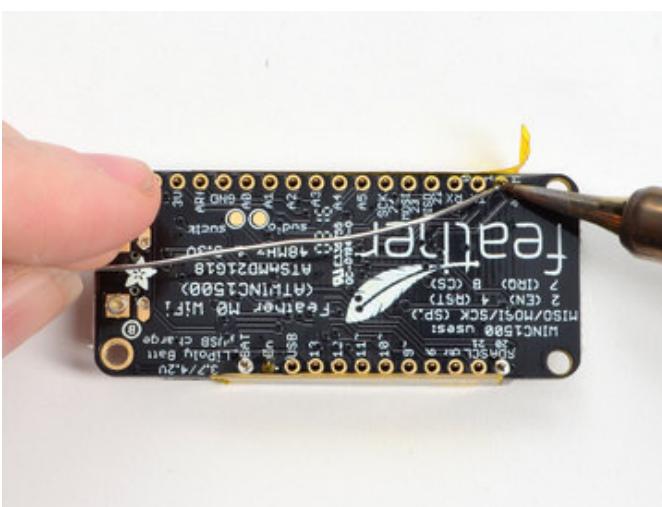
### Tape In Place

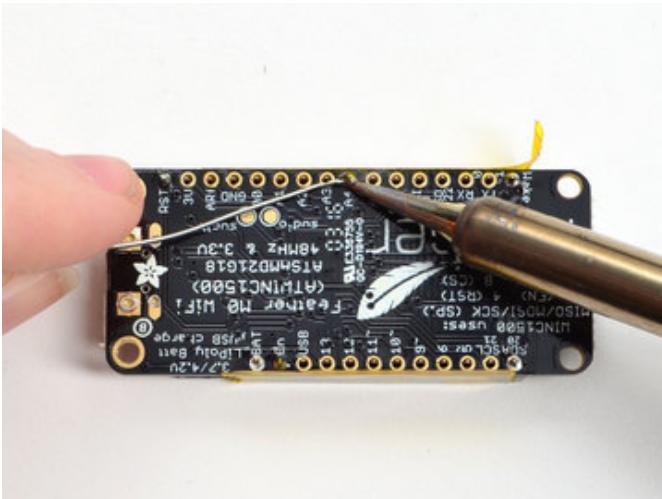
For sockets you'll want to tape them in place so when you flip over the board they don't fall out



### Flip & Tack Solder

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

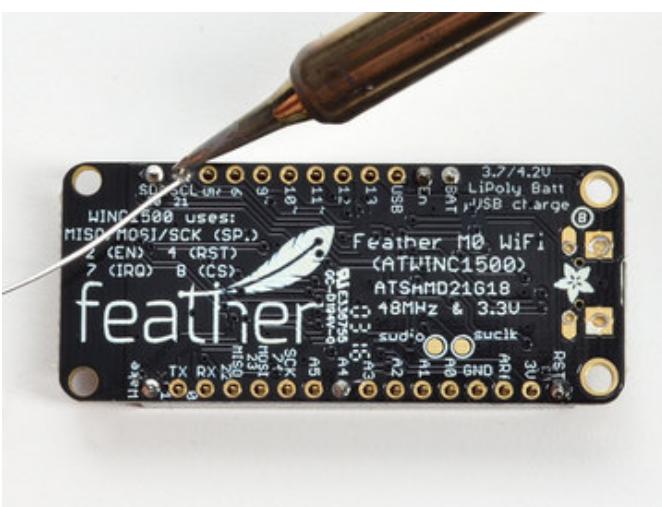




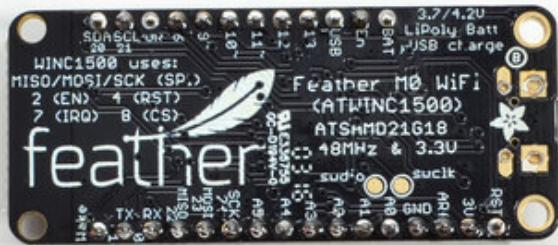
And Solder!

Be sure to solder all pins for reliable electrical contact.

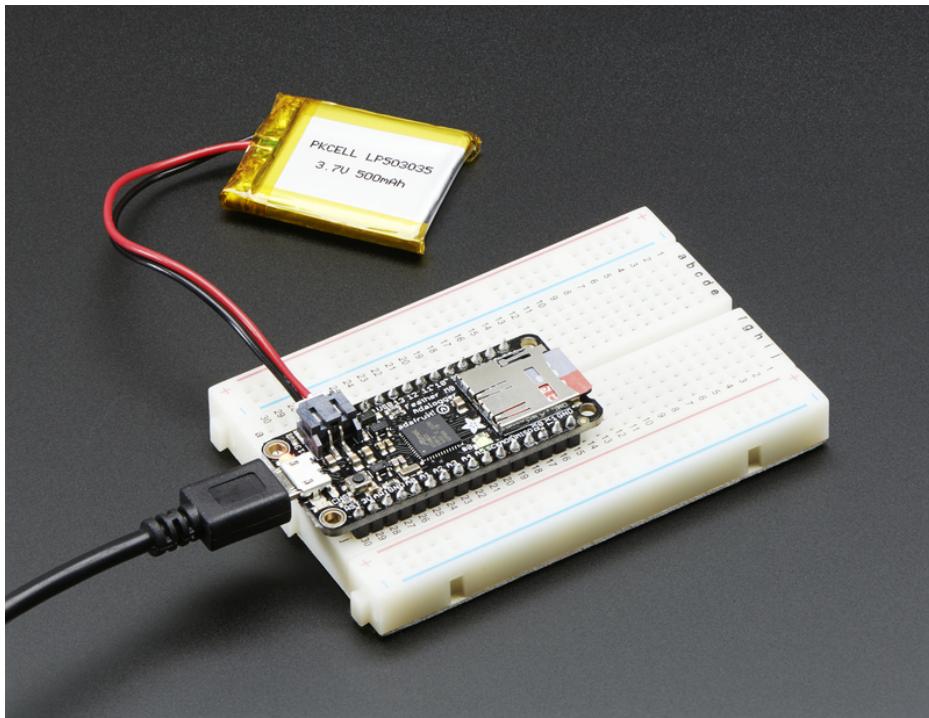
(For tips on soldering, be sure to check out our [Guide to Excellent Soldering](#) (<https://adafru.it/aTk>)).



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



# Power Management

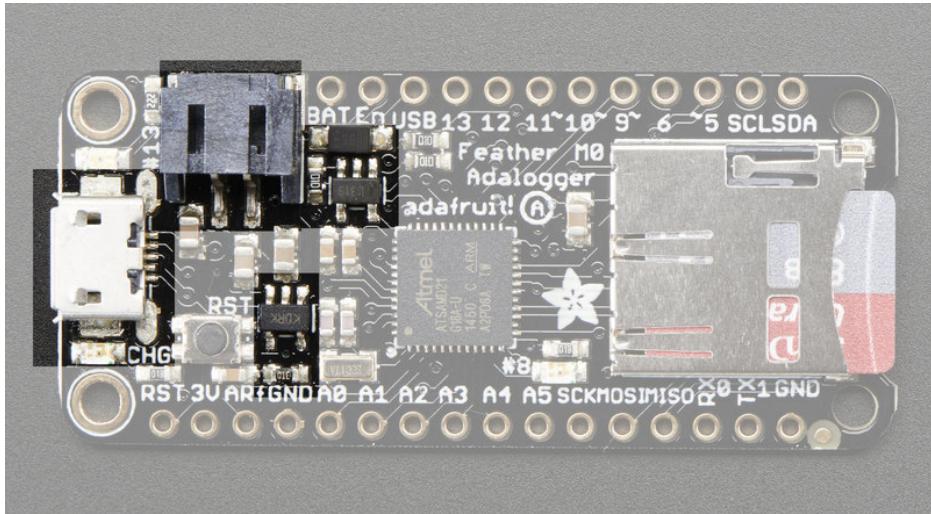


## Battery + USB Power

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



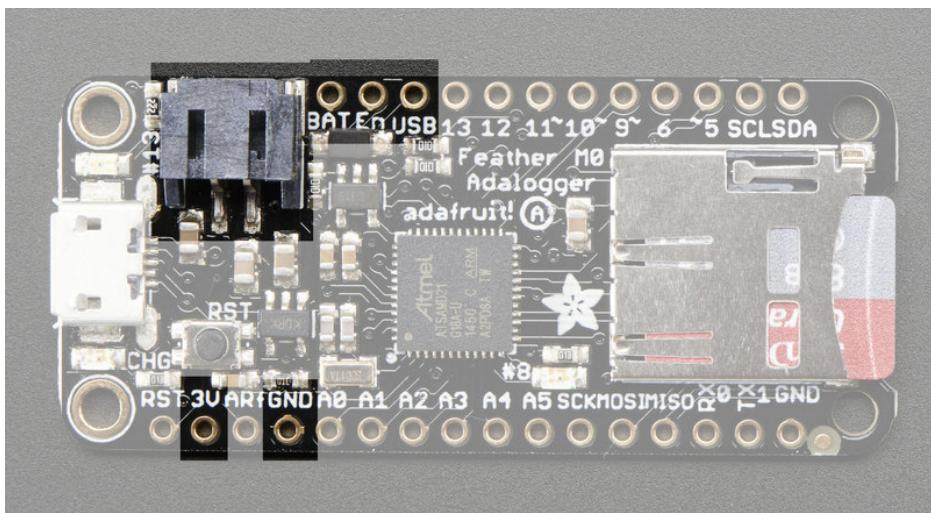
The above shows the Micro USB jack (left), Lipoly JST jack (top left), as well as the 3.3V regulator and changeover diode (just to the right of the JST jack) and the Lipoly charging circuitry (to the right of the Reset button). There's also a **CHG** LED, which will light up while the battery is charging.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



The charge LED is automatically driven by the Lipoly charger circuit. It will try to detect a battery and is expecting one to be attached. If there isn't one it may flicker once in a while when you use power because it's trying to charge a (non-existent) battery. It's not harmful, and it's totally normal!

## 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 'spiky' & sporadic.



## Measuring Battery

If you're running off of a battery, chances are you wanna know what the voltage is at! That way you can tell when the

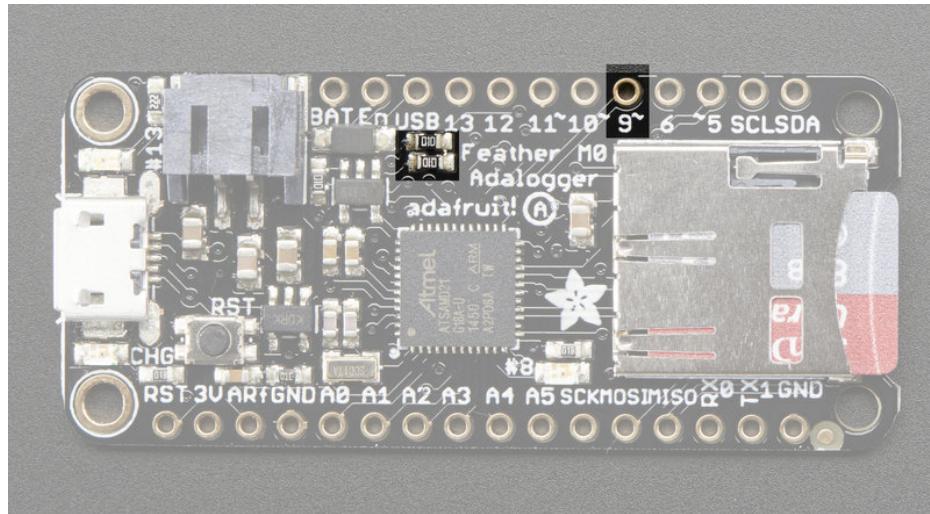
battery needs recharging. Lipoly batteries are 'maxed out' at 4.2V and stick around 3.7V for much of the battery life, then slowly sink down to 3.2V or so before the protection circuitry cuts it off. By measuring the voltage you can quickly tell when you're heading below 3.7V

To make this easy we stuck a double-100K resistor divider on the **BAT** pin, and connected it to **D9** (a.k.a analog #7 **A7**). You can read this pin's voltage, then double it, to get the battery voltage.

```
#define VBATPIN A7

float measuredvbat = analogRead(VBATPIN);
measuredvbat *= 2;      // we divided by 2, so multiply back
measuredvbat *= 3.3;    // Multiply by 3.3V, our reference voltage
measuredvbat /= 1024;   // convert to voltage
Serial.print("VBat: "); Serial.println(measuredvbat);
```

This voltage will 'float' at 4.2V when no battery is plugged in, due to the lipoly charger output, so its not a good way to detect if a battery is plugged in or not (there is no simple way to detect if a battery is plugged in)



## Average Power Draw w/SD Card

The average power draw of the ATSAMD21 + regulator circuitry is 11mA. Both the red and green LED each draw 1mA if you light them up.

Say you are running this sample sketch which logs the analog voltage on A0 to an SD card file once a second.

```
1 #include <SPI.h>
2 #include <SD.h>
3
4 // Set the pins used
5 #define cardSelect 4
6
7 File logfile;
8
9 // blink out an error code
10 void error(uint8_t errno) {
11     while(1) {
12         uint8_t i;
13         for(i=0; i<errno; i++)
14             if(i>0)
15                 if(i>1)
16                     if(i>2)
17                         if(i>3)
18                             if(i>4)
19                                 if(i>5)
20                                     if(i>6)
21                                         if(i>7)
22                                             if(i>8)
23                                                 if(i>9)
24                                                     if(i>10)
25             if(i>0)
26                 if(i>1)
27                     if(i>2)
28                         if(i>3)
29                             if(i>4)
30                                 if(i>5)
31                                     if(i>6)
32                                         if(i>7)
33                                             if(i>8)
34                                                 if(i>9)
35                                                     if(i>10)
36             if(i>0)
37                 if(i>1)
38                     if(i>2)
39                         if(i>3)
40                             if(i>4)
41                                 if(i>5)
42                                     if(i>6)
43                                         if(i>7)
44                                             if(i>8)
45                                                 if(i>9)
46                                                     if(i>10)
47             if(i>0)
48                 if(i>1)
49                     if(i>2)
50                         if(i>3)
51                             if(i>4)
52                                 if(i>5)
53                                     if(i>6)
54                                         if(i>7)
55                                             if(i>8)
56                                                 if(i>9)
57                                                     if(i>10)
58             if(i>0)
59                 if(i>1)
60                     if(i>2)
61                         if(i>3)
62                             if(i>4)
63                                 if(i>5)
64                                     if(i>6)
65                                         if(i>7)
66                                             if(i>8)
67                                                 if(i>9)
68                                                     if(i>10)
69             if(i>0)
70                 if(i>1)
71                     if(i>2)
72                         if(i>3)
73                             if(i>4)
74                                 if(i>5)
75                                     if(i>6)
76                                         if(i>7)
77                                             if(i>8)
78                                                 if(i>9)
79                                                     if(i>10)
80             if(i>0)
81                 if(i>1)
82                     if(i>2)
83                         if(i>3)
84                             if(i>4)
85                                 if(i>5)
86                                     if(i>6)
87                                         if(i>7)
88                                             if(i>8)
89                                                 if(i>9)
90                                                     if(i>10)
91             if(i>0)
92                 if(i>1)
93                     if(i>2)
94                         if(i>3)
95                             if(i>4)
96                                 if(i>5)
97                                     if(i>6)
98                                         if(i>7)
99                                             if(i>8)
100                                         if(i>9)
101                                         if(i>10)
102                                         if(i>11)
103                                         if(i>12)
104                                         if(i>13)
105                                         if(i>14)
106                                         if(i>15)
107                                         if(i>16)
108                                         if(i>17)
109                                         if(i>18)
110                                         if(i>19)
111                                         if(i>20)
112                                         if(i>21)
113                                         if(i>22)
114                                         if(i>23)
115                                         if(i>24)
116                                         if(i>25)
117                                         if(i>26)
118                                         if(i>27)
119                                         if(i>28)
120                                         if(i>29)
121                                         if(i>30)
122                                         if(i>31)
123                                         if(i>32)
124                                         if(i>33)
125                                         if(i>34)
126                                         if(i>35)
127                                         if(i>36)
128                                         if(i>37)
129                                         if(i>38)
130                                         if(i>39)
131                                         if(i>40)
132                                         if(i>41)
133                                         if(i>42)
134                                         if(i>43)
135                                         if(i>44)
136                                         if(i>45)
137                                         if(i>46)
138                                         if(i>47)
139                                         if(i>48)
140                                         if(i>49)
141                                         if(i>50)
142                                         if(i>51)
143                                         if(i>52)
144                                         if(i>53)
145                                         if(i>54)
146                                         if(i>55)
147                                         if(i>56)
148                                         if(i>57)
149                                         if(i>58)
150                                         if(i>59)
151                                         if(i>60)
152                                         if(i>61)
153                                         if(i>62)
154                                         if(i>63)
155                                         if(i>64)
156                                         if(i>65)
157                                         if(i>66)
158                                         if(i>67)
159                                         if(i>68)
160                                         if(i>69)
161                                         if(i>70)
162                                         if(i>71)
163                                         if(i>72)
164                                         if(i>73)
165                                         if(i>74)
166                                         if(i>75)
167                                         if(i>76)
168                                         if(i>77)
169                                         if(i>78)
170                                         if(i>79)
171                                         if(i>80)
172                                         if(i>81)
173                                         if(i>82)
174                                         if(i>83)
175                                         if(i>84)
176                                         if(i>85)
177                                         if(i>86)
178                                         if(i>87)
179                                         if(i>88)
180                                         if(i>89)
181                                         if(i>90)
182                                         if(i>91)
183                                         if(i>92)
184                                         if(i>93)
185                                         if(i>94)
186                                         if(i>95)
187                                         if(i>96)
188                                         if(i>97)
189                                         if(i>98)
190                                         if(i>99)
191                                         if(i>100)
192                                         if(i>101)
193                                         if(i>102)
194                                         if(i>103)
195                                         if(i>104)
196                                         if(i>105)
197                                         if(i>106)
198                                         if(i>107)
199                                         if(i>108)
200                                         if(i>109)
201                                         if(i>110)
202                                         if(i>111)
203                                         if(i>112)
204                                         if(i>113)
205                                         if(i>114)
206                                         if(i>115)
207                                         if(i>116)
208                                         if(i>117)
209                                         if(i>118)
210                                         if(i>119)
211                                         if(i>120)
212                                         if(i>121)
213                                         if(i>122)
214                                         if(i>123)
215                                         if(i>124)
216                                         if(i>125)
217                                         if(i>126)
218                                         if(i>127)
219                                         if(i>128)
220                                         if(i>129)
221                                         if(i>130)
222                                         if(i>131)
223                                         if(i>132)
224                                         if(i>133)
225                                         if(i>134)
226                                         if(i>135)
227                                         if(i>136)
228                                         if(i>137)
229                                         if(i>138)
230                                         if(i>139)
231                                         if(i>140)
232                                         if(i>141)
233                                         if(i>142)
234                                         if(i>143)
235                                         if(i>144)
236                                         if(i>145)
237                                         if(i>146)
238                                         if(i>147)
239                                         if(i>148)
240                                         if(i>149)
241                                         if(i>150)
242                                         if(i>151)
243                                         if(i>152)
244                                         if(i>153)
245                                         if(i>154)
246                                         if(i>155)
247                                         if(i>156)
248                                         if(i>157)
249                                         if(i>158)
250                                         if(i>159)
251                                         if(i>160)
252                                         if(i>161)
253                                         if(i>162)
254                                         if(i>163)
255                                         if(i>164)
256                                         if(i>165)
257                                         if(i>166)
258                                         if(i>167)
259                                         if(i>168)
260                                         if(i>169)
261                                         if(i>170)
262                                         if(i>171)
263                                         if(i>172)
264                                         if(i>173)
265                                         if(i>174)
266                                         if(i>175)
267                                         if(i>176)
268                                         if(i>177)
269                                         if(i>178)
270                                         if(i>179)
271                                         if(i>180)
272                                         if(i>181)
273                                         if(i>182)
274                                         if(i>183)
275                                         if(i>184)
276                                         if(i>185)
277                                         if(i>186)
278                                         if(i>187)
279                                         if(i>188)
280                                         if(i>189)
281                                         if(i>190)
282                                         if(i>191)
283                                         if(i>192)
284                                         if(i>193)
285                                         if(i>194)
286                                         if(i>195)
287                                         if(i>196)
288                                         if(i>197)
289                                         if(i>198)
290                                         if(i>199)
291                                         if(i>200)
292                                         if(i>201)
293                                         if(i>202)
294                                         if(i>203)
295                                         if(i>204)
296                                         if(i>205)
297                                         if(i>206)
298                                         if(i>207)
299                                         if(i>208)
300                                         if(i>209)
301                                         if(i>210)
302                                         if(i>211)
303                                         if(i>212)
304                                         if(i>213)
305                                         if(i>214)
306                                         if(i>215)
307                                         if(i>216)
308                                         if(i>217)
309                                         if(i>218)
310                                         if(i>219)
311                                         if(i>220)
312                                         if(i>221)
313                                         if(i>222)
314                                         if(i>223)
315                                         if(i>224)
316                                         if(i>225)
317                                         if(i>226)
318                                         if(i>227)
319                                         if(i>228)
320                                         if(i>229)
321                                         if(i>230)
322                                         if(i>231)
323                                         if(i>232)
324                                         if(i>233)
325                                         if(i>234)
326                                         if(i>235)
327                                         if(i>236)
328                                         if(i>237)
329                                         if(i>238)
330                                         if(i>239)
331                                         if(i>240)
332                                         if(i>241)
333                                         if(i>242)
334                                         if(i>243)
335                                         if(i>244)
336                                         if(i>245)
337                                         if(i>246)
338                                         if(i>247)
339                                         if(i>248)
340                                         if(i>249)
341                                         if(i>250)
342                                         if(i>251)
343                                         if(i>252)
344                                         if(i>253)
345                                         if(i>254)
346                                         if(i>255)
347                                         if(i>256)
348                                         if(i>257)
349                                         if(i>258)
350                                         if(i>259)
351                                         if(i>260)
352                                         if(i>261)
353                                         if(i>262)
354                                         if(i>263)
355                                         if(i>264)
356                                         if(i>265)
357                                         if(i>266)
358                                         if(i>267)
359                                         if(i>268)
360                                         if(i>269)
361                                         if(i>270)
362                                         if(i>271)
363                                         if(i>272)
364                                         if(i>273)
365                                         if(i>274)
366                                         if(i>275)
367                                         if(i>276)
368                                         if(i>277)
369                                         if(i>278)
370                                         if(i>279)
371                                         if(i>280)
372                                         if(i>281)
373                                         if(i>282)
374                                         if(i>283)
375                                         if(i>284)
376                                         if(i>285)
377                                         if(i>286)
378                                         if(i>287)
379                                         if(i>288)
380                                         if(i>289)
381                                         if(i>290)
382                                         if(i>291)
383                                         if(i>292)
384                                         if(i>293)
385                                         if(i>294)
386                                         if(i>295)
387                                         if(i>296)
388                                         if(i>297)
389                                         if(i>298)
390                                         if(i>299)
391                                         if(i>300)
392                                         if(i>301)
393                                         if(i>302)
394                                         if(i>303)
395                                         if(i>304)
396                                         if(i>305)
397                                         if(i>306)
398                                         if(i>307)
399                                         if(i>308)
400                                         if(i>309)
401                                         if(i>310)
402                                         if(i>311)
403                                         if(i>312)
404                                         if(i>313)
405                                         if(i>314)
406                                         if(i>315)
407                                         if(i>316)
408                                         if(i>317)
409                                         if(i>318)
410                                         if(i>319)
411                                         if(i>320)
412                                         if(i>321)
413                                         if(i>322)
414                                         if(i>323)
415                                         if(i>324)
416                                         if(i>325)
417                                         if(i>326)
418                                         if(i>327)
419                                         if(i>328)
420                                         if(i>329)
421                                         if(i>330)
422                                         if(i>331)
423                                         if(i>332)
424                                         if(i>333)
425                                         if(i>334)
426                                         if(i>335)
427                                         if(i>336)
428                                         if(i>337)
429                                         if(i>338)
430                                         if(i>339)
431                                         if(i>340)
432                                         if(i>341)
433                                         if(i>342)
434                                         if(i>343)
435                                         if(i>344)
436                                         if(i>345)
437                                         if(i>346)
438                                         if(i>347)
439                                         if(i>348)
440                                         if(i>349)
441                                         if(i>350)
442                                         if(i>351)
443                                         if(i>352)
444                                         if(i>353)
445                                         if(i>354)
446                                         if(i>355)
447                                         if(i>356)
448                                         if(i>357)
449                                         if(i>358)
450                                         if(i>359)
451                                         if(i>360)
452                                         if(i>361)
453                                         if(i>362)
454                                         if(i>363)
455                                         if(i>364)
456                                         if(i>365)
457                                         if(i>366)
458                                         if(i>367)
459                                         if(i>368)
460                                         if(i>369)
461                                         if(i>370)
462                                         if(i>371)
463                                         if(i>372)
464                                         if(i>373)
465                                         if(i>374)
466                                         if(i>375)
467                                         if(i>376)
468                                         if(i>377)
469                                         if(i>378)
470                                         if(i>379)
471                                         if(i>380)
472                                         if(i>381)
473                                         if(i>382)
474                                         if(i>383)
475                                         if(i>384)
476                                         if(i>385)
477                                         if(i>386)
478                                         if(i>387)
479                                         if(i>388)
480                                         if(i>389)
481                                         if(i>390)
482                                         if(i>391)
483                                         if(i>392)
484                                         if(i>393)
485                                         if(i>394)
486                                         if(i>395)
487                                         if(i>396)
488                                         if(i>397)
489                                         if(i>398)
490                                         if(i>399)
491                                         if(i>400)
492                                         if(i>401)
493                                         if(i>402)
494                                         if(i>403)
495                                         if(i>404)
496                                         if(i>405)
497                                         if(i>406)
498                                         if(i>407)
499                                         if(i>408)
500                                         if(i>409)
501                                         if(i>410)
502                                         if(i>411)
503                                         if(i>412)
504                                         if(i>413)
505                                         if(i>414)
506                                         if(i>415)
507                                         if(i>416)
508                                         if(i>417)
509                                         if(i>418)
510                                         if(i>419)
511                                         if(i>420)
512                                         if(i>421)
513                                         if(i>422)
514                                         if(i>423)
515                                         if(i>424)
516                                         if(i>425)
517                                         if(i>426)
518                                         if(i>427)
519                                         if(i>428)
520                                         if(i>429)
521                                         if(i>430)
522                                         if(i>431)
523                                         if(i>432)
524                                         if(i>433)
525                                         if(i>434)
526                                         if(i>435)
527                                         if(i>436)
528                                         if(i>437)
529                                         if(i>438)
530                                         if(i>439)
531                                         if(i>440)
532                                         if(i>441)
533                                         if(i>442)
534                                         if(i>443)
535                                         if(i>444)
536                                         if(i>445)
537                                         if(i>446)
538                                         if(i>447)
539                                         if(i>448)
540                                         if(i>449)
541                                         if(i>450)
542                                         if(i>451)
543                                         if(i>452)
544                                         if(i>453)
545                                         if(i>454)
546                                         if(i>455)
547                                         if(i>456)
548                                         if(i>457)
549                                         if(i>458)
550                                         if(i>459)
551                                         if(i>460)
552                                         if(i>461)
553                                         if(i>462)
554                                         if(i>463)
555                                         if(i>464)
556                                         if(i>465)
557                                         if(i>466)
558                                         if(i>467)
559                                         if(i>468)
560                                         if(i>469)
561                                         if(i>470)
562                                         if(i>471)
563                                         if(i>472)
564                                         if(i>473)
565                                         if(i>474)
566                                         if(i>475)
567                                         if(i>476)
568                                         if(i>477)
569                                         if(i>478)
570                                         if(i>479)
571                                         if(i>480)
572                                         if(i>481)
573                                         if(i>482)
574                                         if(i>483)
575                                         if(i>484)
576                                         if(i>485)
577                                         if(i>486)
578                                         if(i>487)
579                                         if(i>488)
580                                         if(i>489)
581                                         if(i>490)
582                                         if(i>491)
583                                         if(i>492)
584                                         if(i>493)
585                                         if(i>494)
586                                         if(i>495)
587                                         if(i>496)
588                                         if(i>497)
589                                         if(i>498)
590                                         if(i>499)
591                                         if(i>500)
592                                         if(i>501)
593                                         if(i>502)
594                                         if(i>503)
595                                         if(i>504)
596                                         if(i>505)
597                                         if(i>506)
598                                         if(i>507)
599                                         if(i>508)
600                                         if(i>509)
601                                         if(i>510)
602                                         if(i>511)
603                                         if(i>512)
604                                         if(i>513)
605                                         if(i>514)
606                                         if(i>515)
607                                         if(i>516)
608                                         if(i>517)
609                                         if(i>518)
610                                         if(i>519)
611                                         if(i>520)
612                                         if(i>521)
613                                         if(i>522)
614                                         if(i>523)
615                                         if(i>524)
616                                         if(i>525)
617                                         if(i>526)
618                                         if(i>527)
619                                         if(i>528)
620                                         if(i>529)
621                                         if(i>530)
622                                         if(i>531)
623                                         if(i>532)
624                                         if(i>533)
625                                         if(i>534)
626                                         if(i>535)
627                                         if(i>536)
628                                         if(i>537)
629                                         if(i>538)
630                                         if(i>539)
631                                         if(i>540)
632                                         if(i>541)
633                                         if(i>542)
634                                         if(i>543)
635                                         if(i>544)
636                                         if(i>545)
637                                         if(i>546)
638                                         if(i>547)
639                                         if(i>548)
640                                         if(i>549)
641                                         if(i>550)
642                                         if(i>551)
643                                         if(i>552)
644                                         if(i>553)
645                                         if(i>554)
646                                         if(i>555)
647                                         if(i>556)
648                                         if(i>557)
649                                         if(i>558)
650                                         if(i>559)
651                                         if(i>560)
652                                         if(i>561)
653                                         if(i>562)
654                                         if(i>563)
655                                         if(i>564)
656                                         if(i>565)
657                                         if(i>566)
658                                         if(i>567)
659                                         if(i>568)
660                                         if(i>569)
661                                         if(i>570)
662                                         if(i>571)
663                                         if(i>572)
664                                         if(i>573)
665                                         if(i>574)
666                                         if(i>575)
667                                         if(i>576)
668                                         if(i>577)
669                                         if(i>578)
670                                         if(i>579)
671                                         if(i>580)
672                                         if(i>581)
673                                         if(i>582)
674                                         if(i>583)
675                                         if(i>584)
676                                         if(i>585)
677                                         if(i>586)
678                                         if(i>587)
679                                         if(i>588)
680                                         if(i>589)
681                                         if(i>590)
682                                         if(i>591)
683                                         if(i>592)
684                                         if(i>593)
685                                         if(i>594)
686                                         if(i>595)
687                                         if(i>596)
688                                         if(i>597)
689                                         if(i>598)
690                                         if(i>599)
691                                         if(i>600)
692                                         if(i>601)
693                                         if(i>602)
694                                         if(i>603)
695                                         if(i>604)
696                                         if(i>605)
697                                         if(i>606)
698                                         if(i>607)
699                                         if(i>608)
700                                         if(i>609)
701                                         if(i>610)
702                                         if(i>611)
703                                         if(i>612)
704                                         if(i>613)
705                                         if(i>614)
706                                         if(i>615)
707                                         if(i>616)
708                                         if(i>617)
709                                         if(i>618)
710                                         if(i>619)
711                                         if(i>620)
712                                         if(i>621)
713                                         if(i>622)
714                                         if(i>623)
715                                         if(i>624)
716                                         if(i>625)
717                                         if(i>626)
718                                         if(i>627)
719                                         if(i>628)
720                                         if(i>629)
721                                         if(i>630)
722                                         if(i>631)
723                                         if(i>632)
724                                         if(i>633)
725                                         if(i>634)
726                                         if(i>635)
727                                         if(i>636)
728                                         if(i>637)
729                                         if(i>638)
730                                         if(i>639)
731                                         if(i>640)
732                                         if(i>641)
733                                         if(i>642)
734                                         if(i>643)
735                                         if(i>644)
736                                         if(i>645)
737                                         if(i>646)
738                                         if(i>647)
739                                         if(i>648)
740                                         if(i>649)
741                                         if(i>650)
742                                         if(i>651)
743                                         if(i>652)
744                                         if(i>653)
745                                         if(i>654)
746                                         if(i>655)
747                                         if(i>656)
748                                         if(i>657)
749                                         if(i>658)
750                                         if(i>659)
751                                         if(i>660)
752                                         if(i>661)
753                                         if(i>662)
754                                         if(i>663)
755                                         if(i>664)
756                                         if(i>665)
757                                         if(i>666)
758                                         if(i>667)
759                                         if(i>668)
760                                         if(i>669)
761                                         if(i>670)
762                                         if(i>671)
763                                         if(i>672)
764                                         if(i>673)
765                                         if(i>674)
766                                         if(i>675)
767                                         if(i>676)
768                                         if(i>677)
769                                         if(i>678)
770                                         if(i>679)
771                                         if(i>680)
772                                         if(i>681)
773                                         if(i>682)
774                                         if(i>683)
775                                         if(i>684)
776                                         if(i>685)
777                                         if(i>686)
778                                         if(i>687)
779                                         if(i>688)
780                                         if(i>689)
781                                         if(i>690)
782                                         if(i>691)
783                                         if(i>692)
784                                         if(i>693)
785                                         if(i>694)
786                                         if(i>695)
787                                         if(i>696)
788                                         if(i>697)
789                                         if(i>698)
790                                         if(i>699)
791                                         if(i>700)
792                                         if(i>701)
793                                         if(i>702)
794                                         if(i>703)
795                                         if(i>704)
796                                         if(i>705)
797                                         if(i>706)
798                                         if(i>707)
799                                         if(i>708)
800                                         if(i>709)
801                                         if(i>710)
802                                         if(i>711)
803                                         if(i>712)
804                                         if(i>713)
805                                         if(i>714)
806                                         if(i>715)
807                                         if(i>716)
808                                         if(i>717)
809                                         if(i>718)
810                                         if(i>719)
811                                         if(i>720)
812                                         if(i>721)
813                                         if(i>722)
814                                         if(i>723)
815                                         if(i>724)
816                                         if(i>725)
817                                         if(i>726)
818                                         if(i>727)
819                                         if(i>728)
820                                         if(i>729)
821                                         if(i>730)
822                                         if(i>731)
823                                         if(i>732)
824                                         if(i>733)
825                                         if(i>734)
826                                         if(i>735)
827                                         if(i>736)
828                                         if(i>737)
829                                         if(i>738)
830                                         if(i>739)
831                                         if(i>740)
832                                         if(i>741)
833                                         if(i>742)
834                                         if(i>743)
835                                         if(i>744)
836                                         if(i>745)
837                                         if(i>746)
838                                         if(i>747)
839                                         if(i>748)
840                                         if(i>749)
841                                         if(i>750)
842                                         if(i>751)
843                                         if(i>752)
844                                         if(i>753)
845                                         if(i>754)
846                                         if(i>755)
847                                         if(i>756)
848                                         if(i>757)
849                                         if(i>758)
850                                         if(i>759)
851                                         if(i>760)
852                                         if(i>761)
853                                         if(i>762)
854                                         if(i>763)
855                                         if(i>764)
856                                         if(i>765)
857                                         if(i>766)
858                                         if(i>767)
859                                         if(i>768)
860                                         if(i>769)
861                                         if(i>770)
862                                         if(i>771)
863                                         if(i>772)
864                                         if(i>773)
865                                         if(i>774)
866                                         if(i>775)
867                                         if(i>776)
868                                         if(i>777)
869                                         if(i>778)
870                                         if(i>779)
871                                         if(i>780)
872                                         if(i>781)
873                                         if(i>782)
874                                         if(i>783)
875                                         if(i>784)
876                                         if(i>785)
877                                         if(i>786)
878                                         if(i>787)
879                                         if(i>788)
880                                         if(i>789)
881                                         if(i>790)
882                                         if(i>791)
883                                         if(i>792)
884                                         if(i>793)
885                                         if(i>794)
886                                         if(i>795)
887                                         if(i>796)
888                                         if(i>797)
889                                         if(i>798)
890                                         if(i>799)
891                                         if(i>800)
892                                         if(i>801)
893                                         if(i>802)
894                                         if(i>803)
895                                         if(i>804)
896                                         if(i>805)
897                                         if(i>806)
898                                         if(i>807)
899                                         if(i>808)
900                                         if(i>809)
901                                         if(i>810)
902                                         if(i>811)
903                                         if(i>812)
904                                         if(i>813)
905                                         if(i>814)
906                                         if(i>815)
907                                         if(i>816)
908                                         if(i>817)
909                                         if(i>818)
910                                         if(i>819)
911                                         if(i>820)
912                                         if(i>821)
913                                         if(i>822)
914                                         if(i>823)
915                                         if(i>824)
916                                         if(i>825)
917                                         if(i>82
```

```

15    }
14    digitalWrite(13, HIGH);
15    delay(100);
16    digitalWrite(13, LOW);
17    delay(100);
18  }
19  for (i=errno; i<10; i++) {
20    delay(200);
21  }
22 }
23 }
24
25 // This line is not needed if you have Adafruit SAMD board package 1.6.2+
26 // #define Serial SerialUSB
27
28 void setup() {
29   // connect at 115200 so we can read the GPS fast enough and echo without dropping chars
30   // also spit it out
31   Serial.begin(115200);
32   Serial.println("\r\nAnalog logger test");
33   pinMode(13, OUTPUT);
34
35
36   // see if the card is present and can be initialized:
37   if (!SD.begin(cardSelect)) {
38     Serial.println("Card init. failed!");
39     error(2);
40   }
41   char filename[15];
42   strcpy(filename, "/ANALOG00.TXT");
43   for (uint8_t i = 0; i < 100; i++) {
44     filename[7] = '0' + i/10;
45     filename[8] = '0' + i%10;
46     // create if does not exist, do not open existing, write, sync after write
47     if (! SD.exists(filename)) {
48       break;
49     }
50   }
51
52   logfile = SD.open(filename, FILE_WRITE);
53   if( ! logfile ) {
54     Serial.print("Couldnt create ");
55     Serial.println(filename);
56     error(3);
57   }
58   Serial.print("Writing to ");
59   Serial.println(filename);
60
61   pinMode(13, OUTPUT);
62   pinMode(8, OUTPUT);
63   Serial.println("Ready!");
64 }
65
66 uint8_t i=0;
67 void loop() {
68   digitalWrite(8, HIGH);
69   logfile.print("A0 = "); logfile.println(analogRead(0));

```

```

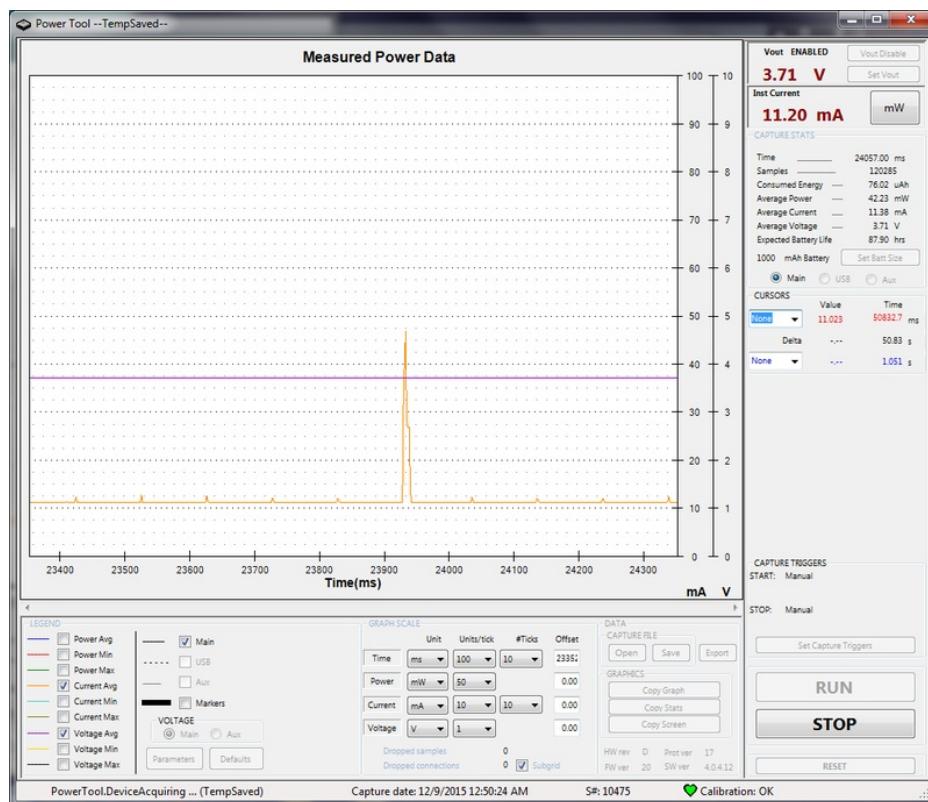
70 Serial.print("A0 = "); Serial.println(analogRead(0));
71 digitalWrite(8, LOW);
72
73 delay(100);
74 }

```

**adalogger.ino** hosted with ❤ by [GitHub](#)

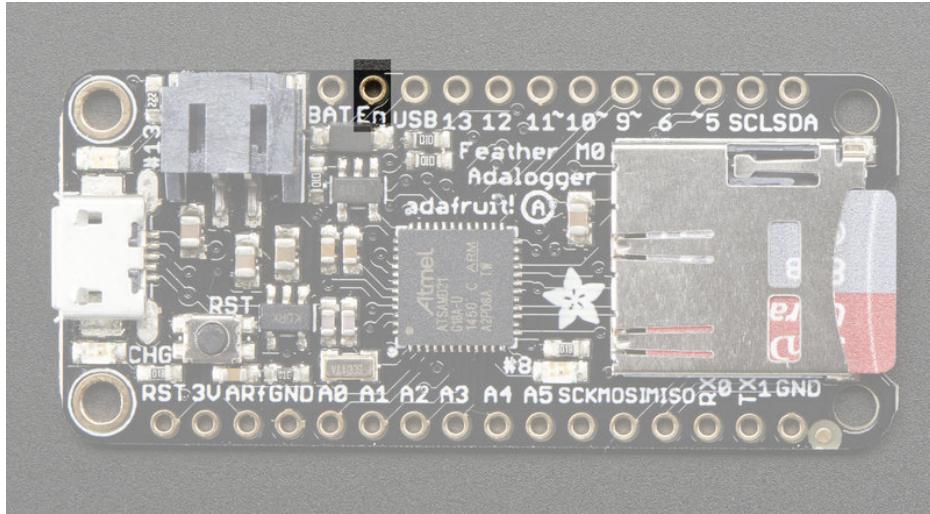
[view raw](#)

You'll draw 11mA or so for 100ms during the `delay(100)` line. Since we blink the pin #8 LED, we'll also get a 11mA blip for about 10ms. The data for the SD card is *buffered* which means that whenever we reach 512 bytes of log file that needs to be written, the SD card will actually write the data. When this happens you'll see a 50mA pulse for 10ms. If you use `flush()` to write the log file, this pulse will be much longer, as you have to write the file as well as the file table sectors. Your 50mA spike could end up being 500ms or longer. So basically, keep your file writes to a minimum if you can avoid it!



## ENable pin

If you'd like to turn off the 3.3V regulator, you can do that with the **EN(able)** pin. Simply tie this pin to **Ground** and it will disable the 3V regulator. The **BAT** and **USB** pins will still be powered



## Alternative Power Options

---

The two primary ways for powering a feather are a 3.7/4.2V LiPo battery plugged into the JST port or a USB power cable.

If you need other ways to power the Feather, here's what we recommend:

- For permanent installations, a [5V 1A USB wall adapter](https://adafru.it/duP) (<https://adafru.it/duP>) will let you plug in a USB cable for reliable power
- For mobile use, where you don't want a LiPoly, [use a USB battery pack!](https://adafru.it/e2q) (<https://adafru.it/e2q>)
- If you have a higher voltage power supply, [use a 5V buck converter](https://adafru.it/DHs) (<https://adafru.it/DHs>) and wire it to a [USB cable's 5V and GND input](https://adafru.it/DHu) (<https://adafru.it/DHu>)

Here's what you cannot do:

- **Do not use alkaline or NiMH batteries** and connect to the battery port - this will destroy the LiPoly charger and there's no way to disable the charger
- **Do not use 7.4V RC batteries on the battery port** - this will destroy the board

The Feather *is not designed for external power supplies* - this is a design decision to make the board compact and low cost. It is not recommended, but technically possible:

- **Connect an external 3.3V power supply to the 3V and GND pins.** Not recommended, this may cause unexpected behavior and the EN pin will no longer. Also this doesn't provide power on BAT or USB and some Feathers/Wings use those pins for high current usages. You may end up damaging your Feather.
- **Connect an external 5V power supply to the USB and GND pins.** Not recommended, this may cause unexpected behavior when plugging in the USB port because you will be back-powering the USB port, which *could* confuse or damage your computer.

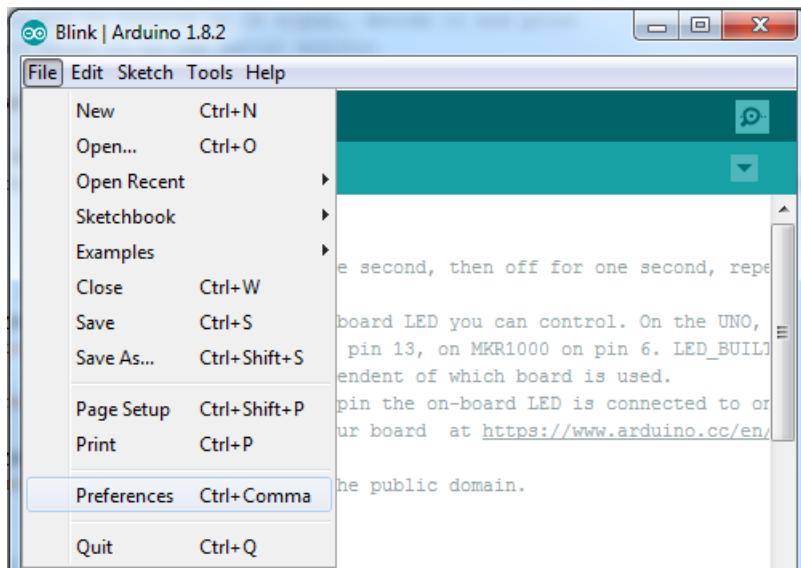
# Arduino IDE Setup

The first thing you will need to do is to download the latest release of the Arduino IDE. You will need to be using **version 1.8** or higher for this guide

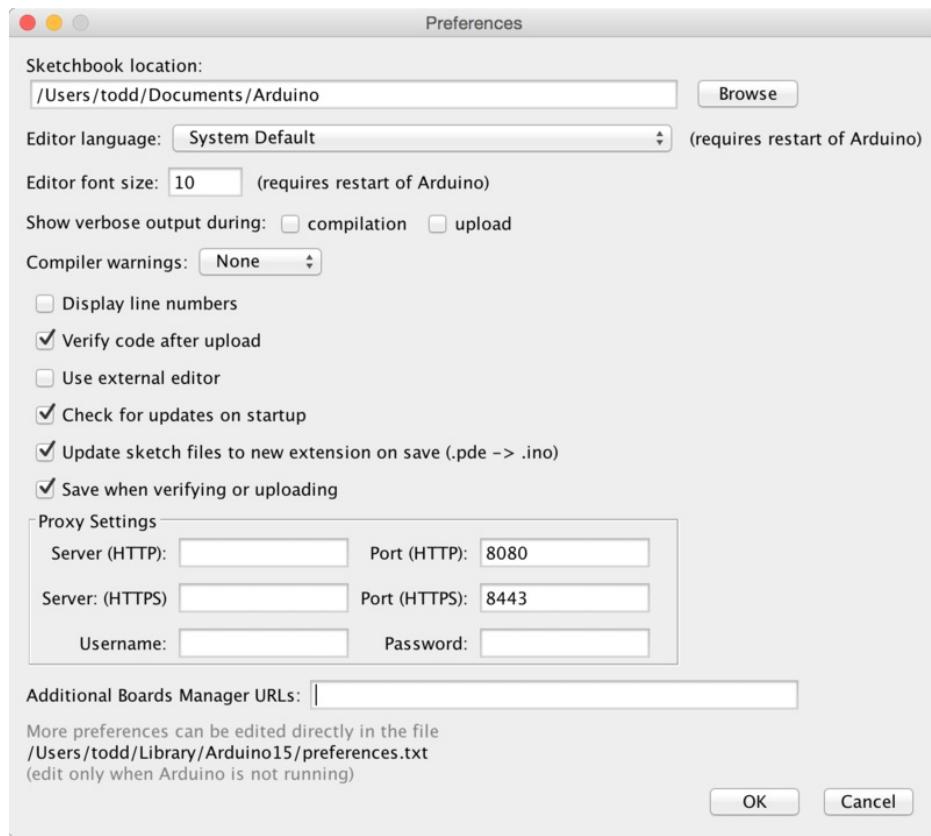
<https://adafru.it/f1P>

<https://adafru.it/f1P>

After you have downloaded and installed **the latest version of Arduino IDE**, you will need to start the IDE and navigate to the **Preferences** menu. You can access it from the **File** menu in *Windows* or *Linux*, or the **Arduino** menu on *OS X*.



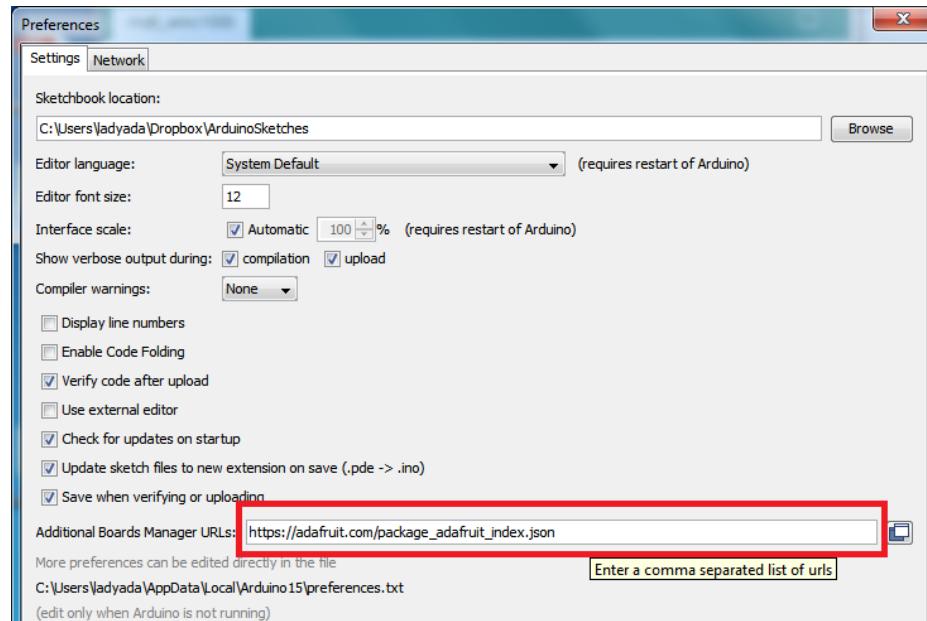
A dialog will pop up just like the one shown below.



We will be adding a URL to the new **Additional Boards Manager URLs** option. The list of URLs is comma separated, and *you will only have to add each URL once*. New Adafruit boards and updates to existing boards will automatically be picked up by the Board Manager each time it is opened. The URLs point to index files that the Board Manager uses to build the list of available & installed boards.

To find the most up to date list of URLs you can add, you can visit the list of [third party board URLs on the Arduino IDE wiki](#) (<https://adafru.it/f7U>). We will only need to add one URL to the IDE in this example, but *you can add multiple URLs by separating them with commas*. Copy and paste the link below into the **Additional Boards Manager URLs** option in the Arduino IDE preferences.

[https://adafruit.github.io/arduino-board-index/package\\_adafruit\\_index.json](https://adafruit.github.io/arduino-board-index/package_adafruit_index.json)



Here's a short description of each of the Adafruit supplied packages that will be available in the Board Manager when you add the URL:

- **Adafruit AVR Boards** - Includes support for Flora, Gemma, Feather 32u4, Trinket, & Trinket Pro.
- **Adafruit SAMD Boards** - Includes support for Feather M0 and M4, Metro M0 and M4, ItsyBitsy M0 and M4, Circuit Playground Express, Gemma M0 and Trinket M0
- **Arduino Leonardo & Micro MIDI-USB** - This adds MIDI over USB support for the Flora, Feather 32u4, Micro and Leonardo using the [arcore project \(https://adafru.it/eSI\)](https://adafru.it/eSI).

If you have multiple boards you want to support, say ESP8266 and Adafruit, have both URLs in the text box separated by a comma (,)

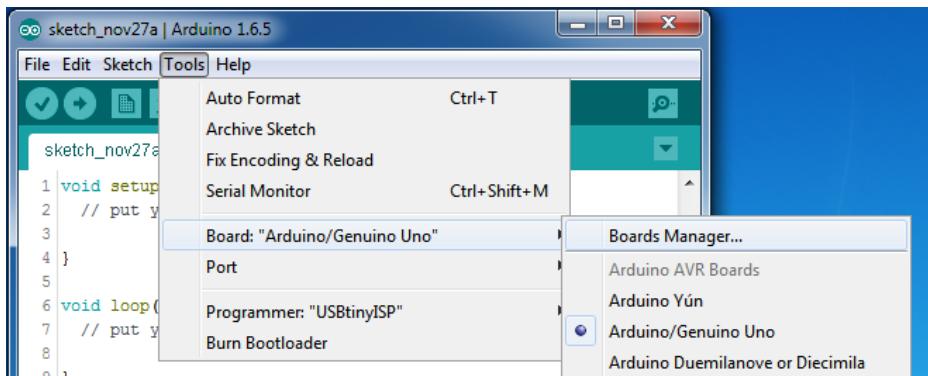
Once done click **OK** to save the new preference settings. Next we will look at installing boards with the Board Manager.

Now continue to the next step to actually install the board support package!

# Using with Arduino IDE

The Feather/Metro/Gemma/QTPy/Trinket M0 and M4 use an ATSAMD21 or ATSAMD51 chip, and you can pretty easily get it working with the Arduino IDE. Most libraries (including the popular ones like NeoPixels and display) will work with the M0 and M4, especially devices & sensors that use I2C or SPI.

Now that you have added the appropriate URLs to the Arduino IDE preferences in the previous page, you can open the **Boards Manager** by navigating to the **Tools->Board** menu.



Once the Board Manager opens, click on the category drop down menu on the top left hand side of the window and select **All**. You will then be able to select and install the boards supplied by the URLs added to the preferences.

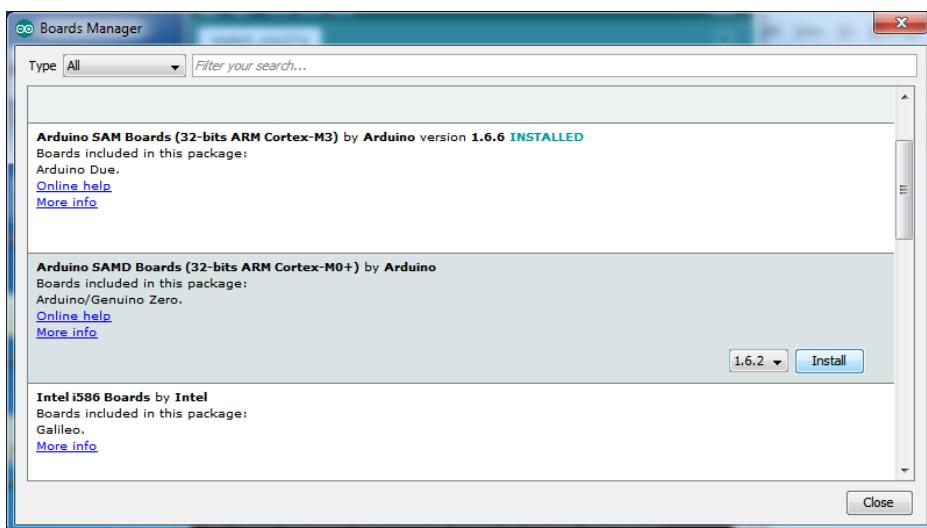


Remember you need SETUP the Arduino IDE to support our board packages - see the previous page on how to add adafruit's URL to the preferences

## Install SAMD Support

First up, install the latest **Arduino SAMD Boards** (version **1.6.11** or later)

You can type **Arduino SAMD** in the top search bar, then when you see the entry, click **Install**

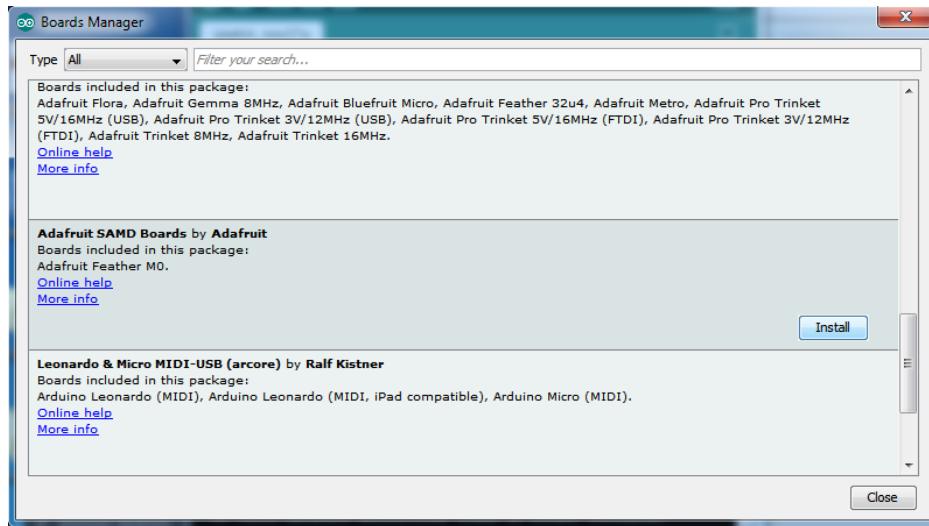


## Install Adafruit SAMD

Next you can install the Adafruit SAMD package to add the board file definitions

Make sure you have **Type All** selected to the left of the *Filter your search...* box

You can type **Adafruit SAMD** in the top search bar, then when you see the entry, click **Install**



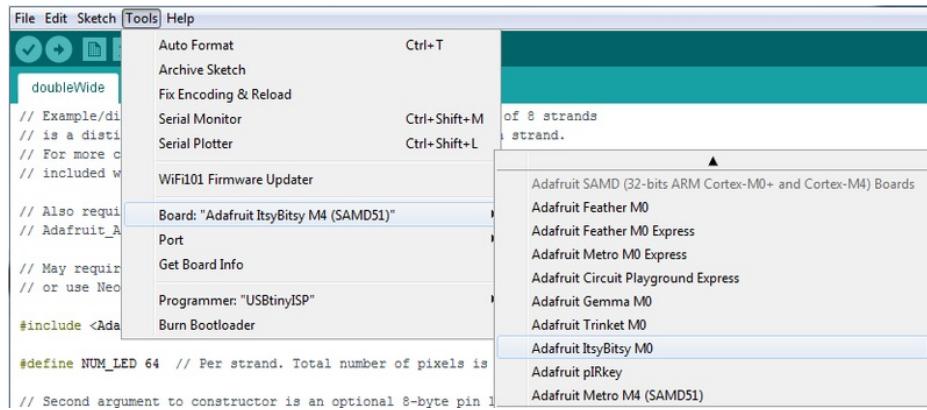
Even though in theory you don't need to - I recommend rebooting the IDE

**Quit and reopen the Arduino IDE** to ensure that all of the boards are properly installed. You should now be able to select and upload to the new boards listed in the **Tools->Board** menu.

Select the matching board, the current options are:

- **Feather M0** (for use with any Feather M0 other than the Express)
- **Feather M0 Express**
- **Metro M0 Express**
- **Circuit Playground Express**
- **Gemma M0**
- **Trinket M0**
- **QT Py M0**
- **ItsyBitsy M0**
- **Hallowing M0**
- **Crickit M0** (this is for direct programming of the Crickit, which is probably not what you want! For advanced hacking only)
- **Metro M4 Express**
- **Grand Central M4 Express**
- **ItsyBitsy M4 Express**
- **Feather M4 Express**
- **Trellis M4 Express**
- **PyPortal M4**
- **PyPortal M4 Titano**
- **PyBadge M4 Express**
- **Metro M4 Airlift Lite**

- PyGamer M4 Express
- MONSTER M4SK
- Hallowing M4
- MatrixPortal M4
- BLM Badge



## Install Drivers (Windows 7 & 8 Only)

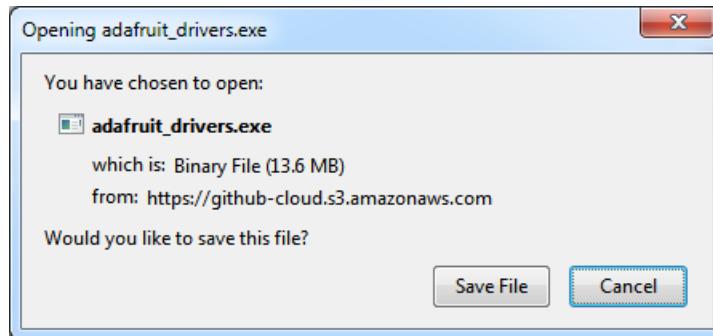
When you plug in the board, you'll need to possibly install a driver

Click below to download our Driver Installer

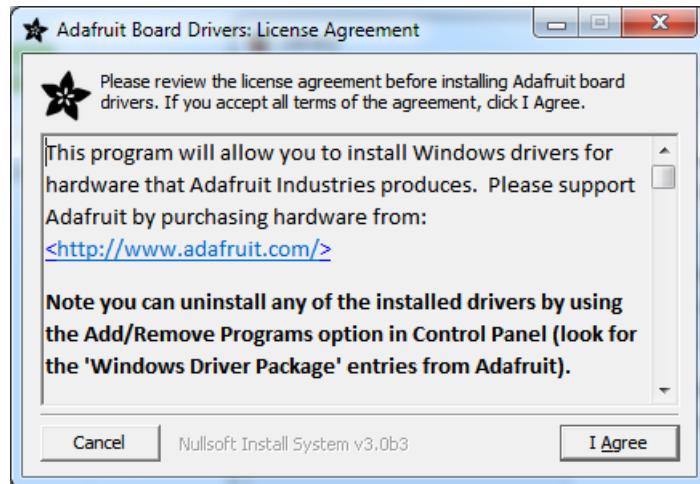
<https://adafru.it/EC0>

<https://adafru.it/EC0>

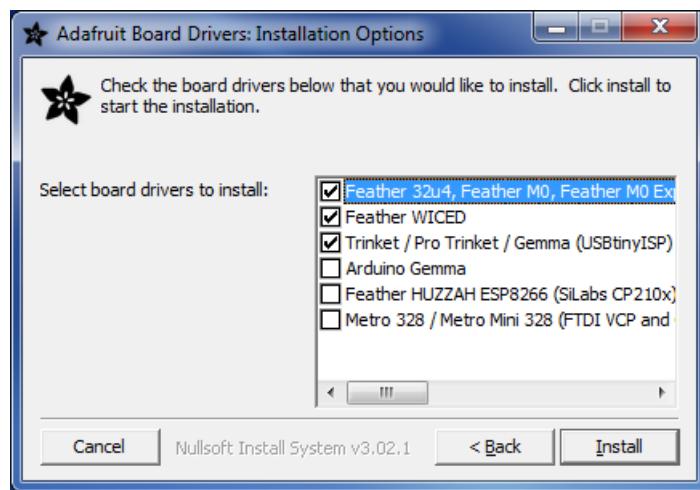
Download and run the installer



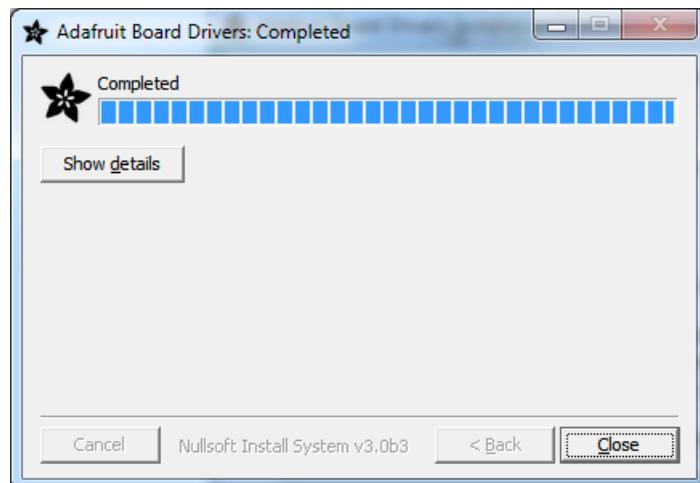
Run the installer! Since we bundle the SiLabs and FTDI drivers as well, you'll need to click through the license



Select which drivers you want to install, the defaults will set you up with just about every Adafruit board!



Click **Install** to do the installin'

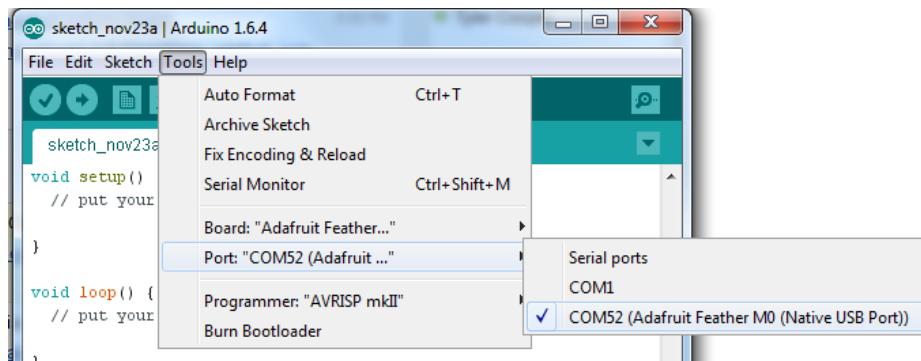


## Blink

Now you can upload your first blink sketch!

Plug in the M0 or M4 board, and wait for it to be recognized by the OS (just takes a few seconds). It will create a serial/COM port, you can now select it from the drop-down, it'll even be 'indicated' as Trinket/Gemma/Metro/Feather/ItsyBitsy/Trellis!

**Please note,** the QT Py and Trellis M4 Express are two of our very few boards that does not have an onboard pin 13 LED so you can follow this section to practice uploading but you wont see an LED blink!



Now load up the Blink example

```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}
```

And click upload! That's it, you will be able to see the LED blink rate change as you adapt the `delay()` calls.



If you are having issues, make sure you selected the matching Board in the menu that matches the hardware you have in your hand.

## Successful Upload

If you have a successful upload, you'll get a bunch of red text that tells you that the device was found and it was programmed, verified & reset

```
Done uploading.  
Write 11024 bytes to flash (173 pages)  
  
[=====] 36% (64/173 pages)  
[=====] 73% (128/173 pages)  
[=====] 100% (173/173 pages)  
  
done in 0.097 seconds  
  
Verify 11024 bytes of flash with checksum.  
  
Verify successful  
  
done in 0.049 seconds  
  
CPU reset.
```

The terminal window shows the progress of writing 11024 bytes to the flash, which takes approximately 0.097 seconds. It then verifies the data, which is successful and takes 0.049 seconds. Finally, it performs a CPU reset.

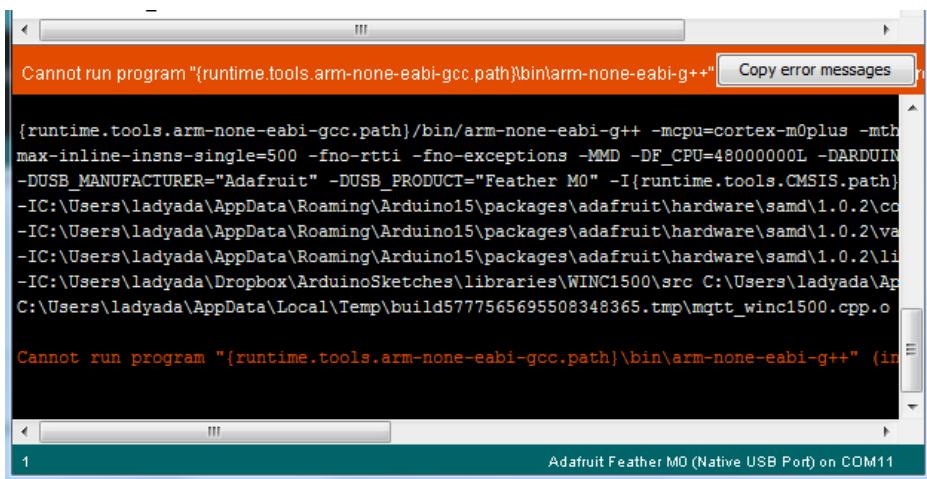
After uploading, you may see a message saying "Disk Not Ejected Properly" about the ...BOOT drive. You can ignore that message: it's an artifact of how the bootloader and uploading work.

## Compilation Issues

If you get an alert that looks like

Cannot run program "{runtime.tools.arm-none-eabi-gcc.path}\bin\arm-none-eabi-g++"

Make sure you have installed the **Arduino SAMD** boards package, you need *both* Arduino & Adafruit SAMD board packages

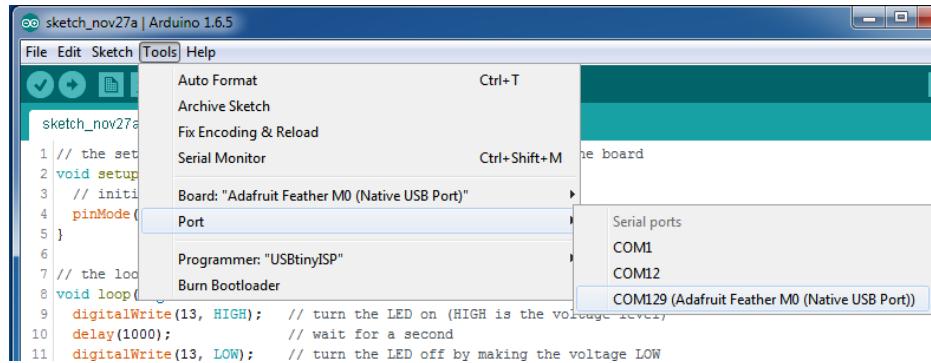


## Manually bootloading

If you ever get in a 'weird' spot with the bootloader, or you have uploaded code that crashes and doesn't auto-reboot into the bootloader, click the **RST** button **twice** (like a double-click) to get back into the bootloader.

The red LED will pulse and/or RGB LED will be green, so you know that its in bootloader mode.

Once it is in bootloader mode, you can select the newly created COM/Serial port and re-try uploading.



You may need to go back and reselect the 'normal' USB serial port next time you want to use the normal upload.

## Ubuntu & Linux Issue Fix

---

Follow the steps for installing Adafruit's udev rules on this page. (<https://adafru.it/iOE>)

# Feather HELP!



Even though this FAQ is labeled for Feather, the questions apply to ItsyBitsy's as well!

My ItsyBitsy/Feather stopped working when I unplugged the USB!

A lot of our example sketches have a

```
while (!Serial);
```

line in setup(), to keep the board waiting until the USB is opened. This makes it a lot easier to debug a program because you get to see all the USB data output. If you want to run your Feather without USB connectivity, delete or comment out that line

---

□ My Feather never shows up as a COM or Serial port in the Arduino IDE

**A vast number of Itsy/Feather 'failures' are due to charge-only USB cables**

We get upwards of 5 complaints a day that turn out to be due to charge-only cables!

Use only a cable that you **know** is for data syncing

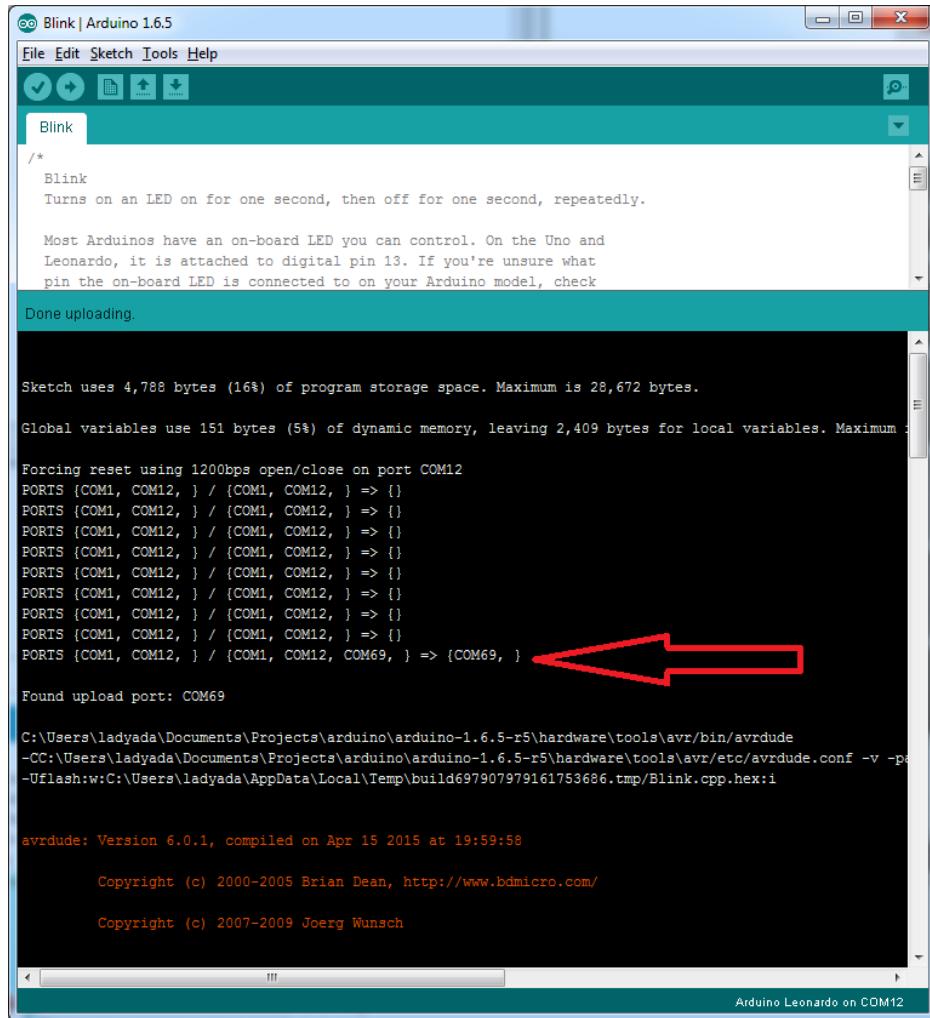
If you have any charge-only cables, cut them in half throw them out. We are serious! They tend to be low quality in general, and will only confuse you and others later, just get a good data+charge USB cable

---

□ Ack! I "did something" and now when I plug in the Itsy/Feather, it doesn't show up as a device anymore so I can't upload to it or fix it...

No problem! You can 'repair' a bad code upload easily. Note that this can happen if you set a watchdog timer or sleep mode that stops USB, or any sketch that 'crashes' your board

1. Turn on **verbose upload** in the Arduino IDE preferences
2. Plug in Itsy or Feather 32u4/M0, it won't show up as a COM/serial port that's ok
3. Open up the Blink example (Examples->Basics->Blink)
4. Select the correct board in the Tools menu, e.g. Feather 32u4, Feather M0, Itsy 32u4 or M0 (*physically check your board to make sure you have the right one selected!*)
5. Compile it (make sure that works)
6. Click Upload to attempt to upload the code
7. The IDE will print out a bunch of COM Ports as it tries to upload. **During this time, double-click the reset button, you'll see the red pulsing LED that tells you its now in bootloading mode**
8. The board will show up as the Bootloader COM/Serial port
9. The IDE should see the bootloader COM/Serial port and upload properly



I can't get the Itsy/Feather USB device to show up - I get "USB Device Malfunctioning" errors!

This seems to happen when people select the wrong board from the Arduino Boards menu.

If you have a Feather 32u4 (look on the board to read what it is you have) Make sure you select **Feather 32u4** for ATMega32u4 based boards! Do not use anything else, do not use the 32u4 breakout board line.

If you have a Feather M0 (look on the board to read what it is you have) Make sure you select **Feather M0** - do not

use 32u4 or Arduino Zero

If you have a ItsyBitsy M0 (look on the board to read what it is you have) Make sure you select **ItsyBitsy M0** - do not use 32u4 or Arduino Zero

---

□ I'm having problems with COM ports and my Itsy/Feather 32u4/M0

Theres **two** COM ports you can have with the 32u4/M0, one is the **user port** and one is the **bootloader port**. They are not the same COM port number!

When you upload a new user program it will come up with a user com port, particularly if you use Serial in your user

program.

If you crash your user program, or have a program that halts or otherwise fails, the user COM port can disappear.

When the user COM port disappears, Arduino will not be able to automatically start the bootloader and upload new software.

So you will need to help it by performing the click-during upload procedure to re-start the bootloader, and upload something that is known working like "Blink"

---

□ I don't understand why the COM port disappears, this does not happen on my Arduino UNO!

UNO-type Arduinos have a *seperate* serial port chip (aka "FTDI chip" or "Prolific PL2303" etc etc) which handles all serial port capability separately than the main chip. This way if the main chip fails, you can always use the COM port.

M0 and 32u4-based Arduinos do not have a separate chip, instead the main processor performs this task for you. It allows for a lower cost, higher power setup...but requires a little more effort since you will need to 'kick' into the bootloader manually once in a while

- 
- I'm trying to upload to my 32u4, getting "avrduude: butterfly\_recv(): programmer is not responding" errors

This is likely because the bootloader is not kicking in and you are accidentally **trying to upload to the wrong COM port**

The best solution is what is detailed above: manually upload Blink or a similar working sketch by hand by manually launching the bootloader

---

I'm trying to upload to my Feather M0, and I get this error "Connecting to programmer: .avrdude: butterfly\_recv(): programmer is not responding"

You probably don't have Feather M0 selected in the boards drop-down. Make sure you selected Feather M0.

---

□ I'm trying to upload to my Feather and i get this error "avrdude: ser\_recv(): programmer is not responding"

You probably don't have Feather M0 / Feather 32u4 selected in the boards drop-down. Make sure you selected Feather M0 (or Feather 32u4).

---

□ I attached some wings to my Feather and now I can't read the battery voltage!

Make sure your Wing doesn't use pin #9 which is the analog sense for the lipo battery!

---

□ The yellow LED Is flickering on my Feather, but no battery is plugged in, why is that?

The charge LED is automatically driven by the Lipoly charger circuit. It will try to detect a battery and is expecting one to be attached. If there isn't one it may flicker once in a while when you use power because it's trying to charge a (non-existant) battery.

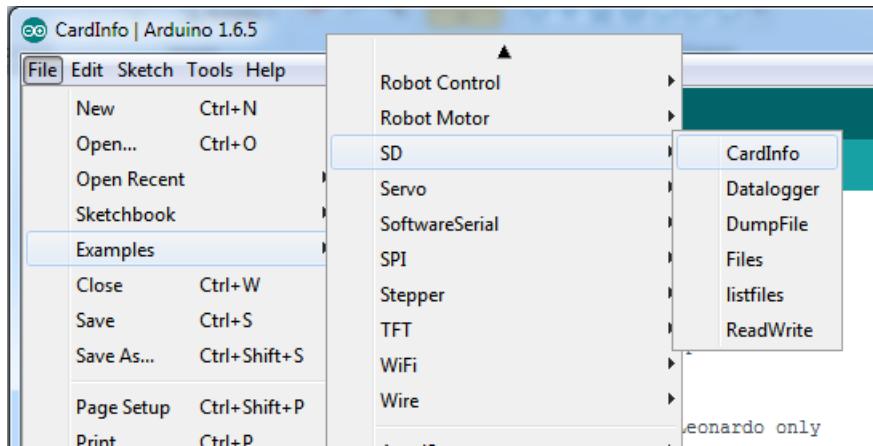
It's not harmful, and its totally normal!



# Using the SD Card

Once you have your Feather working, you probably want to rock out with some SD card reading and writing! Luckily, the Arduino IDE has an SD card library that works great, and it even comes with the IDE!

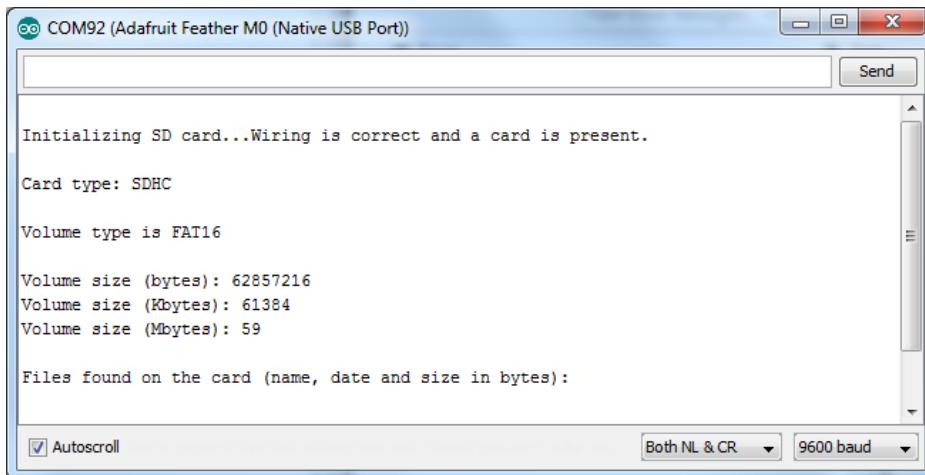
You can start with **CardInfo** which is very detailed



Luckily many of the default examples already have **chipSelect = 4** For other sketches, do check to make sure that CS is set to 4! The SD card uses hardware SPI for the remaining pins.

Make sure you have Adafruit SAMD board package version 1.6.2 or higher, so that Serial debug data goes out on Serial not SerialUSB!

One done, upload & open up the serial console and you'll get all this info including a list of files



Note that it may not print out the files, that's because the example `root.ls(LS_R | LS_DATE | LS_SIZE)` expects to print to Serial rather than SerialUSB.

If you want to list the files, use **listFiles** example

Once you have that working, check out the other examples, such the **Datalogger** example (saving analog data to SD

card) and **Dumpfile** example (reading back data from an SD card)

## Example logging sketch

---

If you want to try saving data to the SD card in the simplest sketch, try this example. You can adjust the **delay()** to set how often analog data is read from pin **A0** and saved to the SD card. The red LED will blink if there's an error, and the green LED will blink when data is written to the SD card.

Note that to save power, we *buffer* the data, so you will only 'save' data truly every 50 datapoints (512 total characters written)

```
1 #include <SPI.h>
2 #include <SD.h>
3
4 // Set the pins used
5 #define cardSelect 4
6
7 File logfile;
8
9 // blink out an error code
10 void error(uint8_t errno) {
11     while(1) {
12         uint8_t i;
13         for (i=0; i<errno; i++) {
14             digitalWrite(13, HIGH);
15             delay(100);
16             digitalWrite(13, LOW);
17             delay(100);
18         }
19         for (i=errno; i<10; i++) {
20             delay(200);
21         }
22     }
23 }
24
25 // This line is not needed if you have Adafruit SAMD board package 1.6.2+
26 // #define Serial SerialUSB
27
28 void setup() {
29     // connect at 115200 so we can read the GPS fast enough and echo without dropping chars
30     // also spit it out
31     Serial.begin(115200);
32     Serial.println("\r\nAnalog logger test");
33     pinMode(13, OUTPUT);
34
35
36     // see if the card is present and can be initialized:
37     if (!SD.begin(cardSelect)) {
38         Serial.println("Card init. failed!");
39         error(2);
40     }
41     char filename[15];
42     strcpy(filename, "/ANALOG00.TXT");
43     for (uint8_t i = 0; i < 100; i++) {
44         filename[7] = '0' + i/10;
45         filename[8] = '0' + i%10;
```

```

46 // create if does not exist, do not open existing, write, sync after write
47 if (! SD.exists(filename)) {
48     break;
49 }
50 }
51
52 logfile = SD.open(filename, FILE_WRITE);
53 if( ! logfile ) {
54     Serial.print("Couldnt create ");
55     Serial.println(filename);
56     error(3);
57 }
58 Serial.print("Writing to ");
59 Serial.println(filename);
60
61 pinMode(13, OUTPUT);
62 pinMode(8, OUTPUT);
63 Serial.println("Ready!");
64 }
65
66 uint8_t i=0;
67 void loop() {
68     digitalWrite(8, HIGH);
69     logfile.print("A0 = "); logfile.println(analogRead(0));
70     Serial.print("A0 = "); Serial.println(analogRead(0));
71     digitalWrite(8, LOW);
72
73     delay(100);
74 }

```

**adalogger.ino** hosted with ❤ by [GitHub](#)

[view raw](#)

If you really want to make sure you save every data point, put a

`logfile.flush();`

right after the `logfile.print`'s however this will cause the adalogger to draw a log more power, maybe about 3x as much on average (30mA avg rather than about 10mA)

## Next steps!

---

Once you know the SD card works, check out the [SD card library](#) (<https://adafruit.it/ucu>) examples, [SD library documentation](#) (<https://adafruit.it/ucu>) and [Notes](#) (<https://adafruit.it/ucv>)!

# Adapting Sketches to M0 & M4

The ATSAMD21 and 51 are very nice little chips, but fairly new as Arduino-compatible cores go. **Most** sketches & libraries will work but here's a collection of things we noticed.

The notes below cover a range of Adafruit M0 and M4 boards, but not every rule will apply to every board (e.g. Trinket and Gemma M0 do not have ARef, so you can skip the Analog References note!).

## Analog References

---

If you'd like to use the **ARef** pin for a non-3.3V analog reference, the code to use is `analogReference(AR_EXTERNAL)` (it's AR\_EXTERNAL not EXTERNAL)

## Pin Outputs & Pullups

---

The old-style way of turning on a pin as an input with a pullup is to use

```
pinMode(pin, INPUT)  
digitalWrite(pin, HIGH)
```

This is because the pullup-selection register on 8-bit AVR chips is the same as the output-selection register.

For M0 & M4 boards, you can't do this anymore! Instead, use:

```
pinMode(pin, INPUT_PULLUP)
```

Code written this way still has the benefit of being *backwards compatible with AVR*. You don't need separate versions for the different board types.

## Serial vs SerialUSB

---

99.9% of your existing Arduino sketches use `Serial.print` to debug and give output. For the Official Arduino SAMD/M0 core, this goes to the Serial5 port, which isn't exposed on the Feather. The USB port for the Official Arduino M0 core is called **SerialUSB** instead.

In the Adafruit M0/M4 Core, we fixed it so that `Serial` goes to `USB` so it will automatically work just fine.

However, on the off chance you are using the official Arduino SAMD core and *not* the Adafruit version (which really, we recommend you use our version because it's been tuned to our boards), and you want your `Serial` prints and reads to use the `USB` port, use `SerialUSB` instead of `Serial` in your sketch.

If you have existing sketches and code and you want them to work with the M0 without a huge find-replace, put

```
#if defined(ARDUINO_SAMD_ZERO) && defined(SERIAL_PORT_USBVIRTUAL)  
// Required for Serial on Zero based boards  
#define Serial SERIAL_PORT_USBVIRTUAL  
#endif
```

right above the **first** function definition in your code. For example:



The screenshot shows the Arduino IDE interface with a sketch titled "datecalc". The code in the editor is as follows:

```
datecalc $  
1 // Simple date conversions and calculations  
2  
3 #include <Wire.h>  
4 #include "RTClib.h"  
5  
6 #if defined(ARDUINO_ARCH_SAMD)  
7 // for Zero, output on USB Serial console, remove line below if using programming port to program the Zero!  
8 #define Serial SerialUSB  
9 #endif  
10  
11 void ShowDate(const char* txt, const DateTime& dt) {  
12     Serial.print(txt);  
13     Serial.print(' ');
```

## AnalogWrite / PWM on Feather/Metro M0

After looking through the SAMD21 datasheet, we've found that some of the options listed in the multiplexer table don't exist on the specific chip used in the Feather M0.

For all SAMD21 chips, there are two peripherals that can generate PWM signals: The Timer/Counter (TC) and Timer/Counter for Control Applications (TCC). Each SAMD21 has multiple copies of each, called 'instances'.

Each TC instance has one count register, one control register, and two output channels. Either channel can be enabled and disabled, and either channel can be inverted. The pins connected to a TC instance can output identical versions of the same PWM waveform, or complementary waveforms.

Each TCC instance has a single count register, but multiple compare registers and output channels. There are options for different kinds of waveform, interleaved switching, programmable dead time, and so on.

The biggest members of the SAMD21 family have five TC instances with two 'waveform output' (WO) channels, and three TCC instances with eight WO channels:

- TC[0-4],WO[0-1]
- TCC[0-2],WO[0-7]

And those are the ones shown in the datasheet's multiplexer tables.

The SAMD21G used in the Feather M0 only has three TC instances with two output channels, and three TCC instances with eight output channels:

- TC[3-5],WO[0-1]
- TCC[0-2],WO[0-7]

Tracing the signals to the pins broken out on the Feather M0, the following pins can't do PWM at all:

- **Analog pin A5**

The following pins can be configured for PWM without any signal conflicts as long as the SPI, I2C, and UART pins keep their protocol functions:

- **Digital pins 5, 6, 9, 10, 11, 12, and 13**
- **Analog pins A3 and A4**

If only the SPI pins keep their protocol functions, you can also do PWM on the following pins:

- TX and SDA (Digital pins 1 and 20)

## analogWrite() PWM range

On AVR, if you set a pin's PWM with `analogWrite(pin, 255)` it will turn the pin fully HIGH. On the ARM cortex, it will set it to be 255/256 so there will be very slim but still-existing pulses-to-0V. If you need the pin to be fully on, add test code that checks if you are trying to `analogWrite(pin, 255)` and, instead, does a `digitalWrite(pin, HIGH)`

## analogWrite() DAC on A0

If you are trying to use `analogWrite()` to control the DAC output on **A0**, make sure you do **not** have a line that sets the pin to output. **Remove:** `pinMode(A0, OUTPUT)`.

## Missing header files

There might be code that uses libraries that are not supported by the M0 core. For example if you have a line with

```
#include <util/delay.h>
```

you'll get an error that says

```
fatal error: util/delay.h: No such file or directory
#include <util/delay.h>
^
compilation terminated.
Error compiling.
```

In which case you can simply locate where the line is (the error will give you the file name and line number) and 'wrap it' with `#ifdef`'s so it looks like:

```
#if !defined(ARDUINO_ARCH_SAM) && !defined(ARDUINO_ARCH_SAMD) && !defined(ESP8266) &&
!defined(ARDUINO_ARCH_STM32F2)
#include <util/delay.h>
#endif
```

The above will also make sure that header file isn't included for other architectures

If the `#include` is in the arduino sketch itself, you can try just removing the line.

## Bootloader Launching

For most other AVRs, clicking **reset** while plugged into USB will launch the bootloader manually, the bootloader will time out after a few seconds. For the M0/M4, you'll need to **double click** the button. You will see a pulsing red LED to let you know you're in bootloader mode. Once in that mode, it won't time out! Click reset again if you want to go back to launching code.

## Aligned Memory Access

This is a little less likely to happen to you but it happened to me! If you're used to 8-bit platforms, you can do this nice thing where you can typecast variables around. e.g.

```
uint8_t mybuffer[4];
```

```
float f = (float)mybuffer;
```

You can't be guaranteed that this will work on a 32-bit platform because **mybuffer** might not be aligned to a 2 or 4-byte boundary. The ARM Cortex-M0 can only directly access data on 16-bit boundaries (every 2 or 4 bytes). Trying to access an odd-boundary byte (on a 1 or 3 byte location) will cause a Hard Fault and stop the MCU. Thankfully, there's an easy work around ... just use `memcpy`!

```
uint8_t mybuffer[4];
float f;
memcpy(&f, mybuffer, 4)
```

## Floating Point Conversion

Like the AVR Arduinos, the M0 library does not have full support for converting floating point numbers to ASCII strings. Functions like `sprintf` will not convert floating point. Fortunately, the standard AVR-LIBC library includes the `dtostrf` function which can handle the conversion for you.

Unfortunately, the M0 run-time library does not have `dtostrf`. You may see some references to using `#include <avr/dtostrf.h>` to get `dtostrf` in your code. And while it will compile, it does **not** work.

Instead, check out this thread to find a working `dtostrf` function you can include in your code:

<http://forum.arduino.cc/index.php?topic=368720.0> (<https://adafru.it/IFS>)

## How Much RAM Available?

The ATSAMD21G18 has 32K of RAM, but you still might need to track it for some reason. You can do so with this handy function:

```
extern "C" char *sbrk(int i);

int FreeRam () {
    char stack_dummy = 0;
    return &stack_dummy - sbrk(0);
}
```

Thx to <http://forum.arduino.cc/index.php?topic=365830.msg2542879#msg2542879> (<https://adafru.it/m6D>) for the tip!

## Storing data in FLASH

If you're used to AVR, you've probably used **PROGMEM** to let the compiler know you'd like to put a variable or string in flash memory to save on RAM. On the ARM, its a little easier, simply add **const** before the variable name:

```
const char str[] = "My very long string";
```

That string is now in FLASH. You can manipulate the string just like RAM data, the compiler will automatically read from FLASH so you dont need special PROGMEM-knowledgeable functions.

You can verify where data is stored by printing out the address:

```
Serial.print("Address of str $"); Serial.println((int)&str, HEX);
```

If the address is \$2000000 or larger, its in SRAM. If the address is between \$0000 and \$3FFF Then it is in FLASH

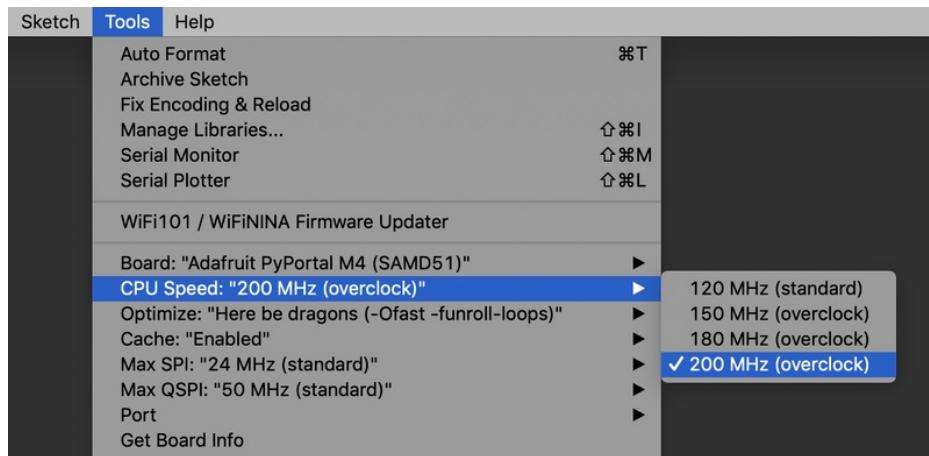
## Pretty-Printing out registers

There's a *lot* of registers on the SAMD21, and you often are going through ASF or another framework to get to them. So having a way to see exactly what's going on is handy. This library from drewfish will help a ton!

<https://github.com/drewfish/arduino-ZeroRegs> (<https://adafru.it/Bet>)

## M4 Performance Options

As of version 1.4.0 of the *Adafruit SAMD Boards* package in the Arduino Boards Manager, some options are available to wring extra performance out of M4-based devices. These are in the *Tools* menu.



All of these performance tweaks involve a degree of uncertainty. There's no guarantee of improved performance in any given project, and *some may even be detrimental*, failing to work in part or in whole. If you encounter trouble, select the default performance settings and re-upload.

Here's what you get and some issues you might encounter...

### CPU Speed (overclocking)

This option lets you adjust the microcontroller core clock...the speed at which it processes instructions...beyond the official datasheet specifications.

Manufacturers often rate speeds conservatively because such devices are marketed for harsh industrial environments...if a system crashes, someone could lose a limb or worse. But most creative tasks are less critical and operate in more comfortable settings, and we can push things a bit if we want more speed.

There is a small but nonzero chance of code **locking up** or **failing to run** entirely. If this happens, try **dialing back the speed by one notch and re-upload**, see if it's more stable.

Much more likely, **some code or libraries may not play well** with the nonstandard CPU speed. For example, currently the NeoPixel library assumes a 120 MHz CPU speed and won't issue the correct data at other settings (this will be worked on). Other libraries may exhibit similar problems, usually anything that strictly depends on CPU timing...you might encounter problems with audio- or servo-related code depending how it's written. **If you encounter such code or libraries, set the CPU speed to the default 120 MHz and re-upload.**

### Optimize

There's usually more than one way to solve a problem, some more resource-intensive than others. Since Arduino got its start on resource-limited AVR microcontrollers, the C++ compiler has always aimed for the **smallest compiled program size**. The “Optimize” menu gives some choices for the compiler to take different and often faster approaches, at the expense of slightly larger program size...with the huge flash memory capacity of M4 devices, that's rarely a problem now.

The “**Small**” setting will compile your code like it always has in the past, aiming for the smallest compiled program size.

The “**Fast**” setting invokes various speed optimizations. The resulting program should produce the same results, is slightly larger, and usually (but not always) noticeably faster. It's worth a shot!

“**Here be dragons**” invokes some more intensive optimizations...code will be larger still, faster still, but there's a possibility these optimizations could cause unexpected behaviors. *Some code may not work the same as before.* Hence the name. Maybe you'll discover treasure here, or maybe you'll sail right off the edge of the world.

Most code and libraries will continue to function regardless of the optimizer settings. If you do encounter problems, **dial it back one notch and re-upload**.

## Cache

This option allows a small collection of instructions and data to be accessed more quickly than from flash memory, boosting performance. It's enabled by default and should work fine with all code and libraries. But if you encounter some esoteric situation, the cache can be disabled, then recompile and upload.

## Max SPI and Max QSPI

**These should probably be left at their defaults.** They're present mostly for our own experiments and can cause serious headaches.

Max SPI determines the clock source for the M4's SPI peripherals. Under normal circumstances this allows transfers up to 24 MHz, and should usually be left at that setting. But...if you're using write-only SPI devices (such as TFT or OLED displays), this option lets you drive them faster (we've successfully used 60 MHz with some TFT screens). The caveat is, if using *any* read/write devices (such as an SD card), *this will not work at all*...SPI reads *absolutely* max out at the default 24 MHz setting, and anything else will fail. **Write = OK. Read = FAIL.** This is true even if your code is using a *lower bitrate setting*...just having the different clock source prevents SPI reads.

Max QSPI does similarly for the extra flash storage on M4 “Express” boards. *Very few* Arduino sketches access this storage at all, let alone in a bandwidth-constrained context, so this will benefit next to nobody. Additionally, due to the way clock dividers are selected, this will only provide some benefit when certain “CPU Speed” settings are active. Our [PyPortal Animated GIF Display](https://adafru.it/EkO) (<https://adafru.it/EkO>) runs marginally better with it, if using the QSPI flash.

## Enabling the Buck Converter on some M4 Boards

---

If you want to reduce power draw, some of our boards have an inductor so you can use the 1.8V buck converter instead of the built in linear regulator. If the board does have an inductor (see the schematic) you can add the line **SUPC->VREG.bit.SEL = 1;** to your code to switch to it. Note it will make ADC/DAC reads a bit noisier so we don't use it by default. [You'll save ~4mA](https://adafru.it/FOH) (<https://adafru.it/FOH>).

# Downloads

## Datasheets

- ATSAMD21 Datasheet (<https://adafru.it/kUf>) (the main chip on the Feather M0)

**Pinout note:** The RX and TX pins are known as Serial1

- PCB Files on GitHub (<https://adafru.it/obn>)
- Fritzing object in the Adafruit Fritzing Library (<https://adafru.it/aP3>)

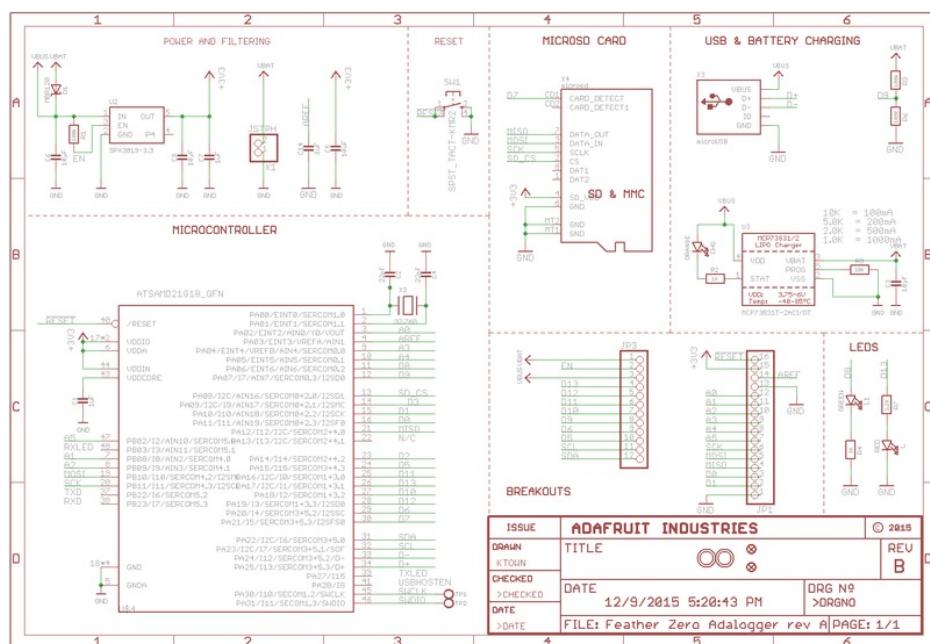
<https://adafru.it/z3e>

<https://adafru.it/z3e>

*Note AREF in the diagram should be marked PA03 not PA02*

## Schematic

Click to enlarge



## Fabrication Print

Dimensions in inches

