

AoM IoT Bit Setup Guide

Author: Mark Hofmeister, (716) 261-0079, mah473@pitt.edu

Assembly v2, Electronics v2, Code v3.0

Last modified: 9/9/2022

You've just picked up one of the most sophisticated and useful LittleBits in the AoM repository of snap electronics. You've made a **fantastic** decision. This is the opportunity of a lifetime – and I can't wait to hear what you think. Below will detail how to make the IoT bit come to life and a few basic example applications.

The IoT bit can either upload (“post”) or download (“get”) digital values from a cloud on the web. This means that it will be posting/getting values that are either HIGH or LOW, i.e. ON or OFF. This means that the Littlebits that you connect to the IoT Bit should be operated **digitally**. For example, you **will** be able to turn an LED ON or OFF with IoT bit. You will not, however, be able to adjust the brightness of the LED with a PWM signal. This means that you may need to make use of the “[threshold](#)” and “[latch](#)” bits to digitize analog signals, which can be explored both through the links and through hands-on experimentation.

Table of Contents:

[Table of Contents:](#)

[Initialization](#)

[Create an account on the Adafruit IO Cloud](#)

[Load your WiFi credentials and Adafruit IO information onto the SD card](#)

[Connect](#)

[Here's a diagram of the interface's functionality:](#)

[“Hello, World” Applications](#)

[Post to IO \(and IFTTT!\)](#)

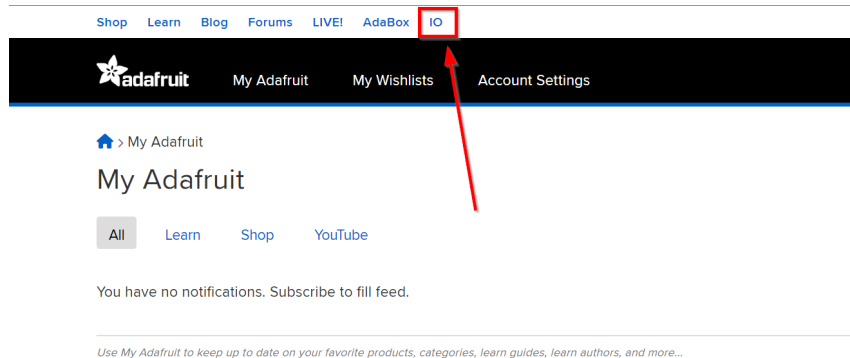
[Get from IO](#)

Initialization

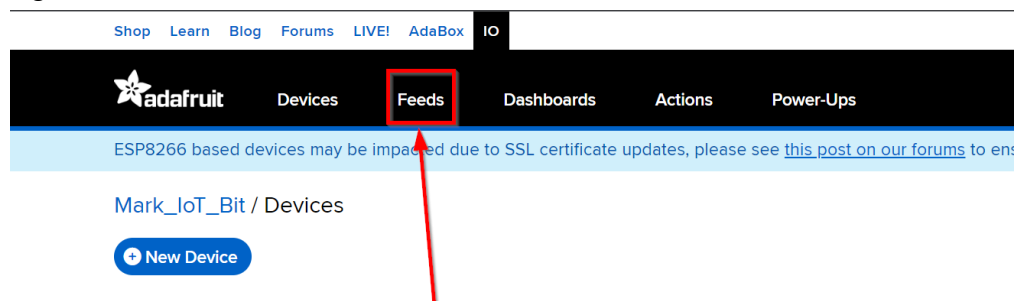
1. Create an account on the Adafruit IO Cloud

We'll start by creating an account on [Adafruit IO](#), which is a **web-based cloud service** where the IoT bit will upload and download data from.

Once you create an account, navigate to the “IO” tab.



Next, navigate to the “Feeds” tab in IO.



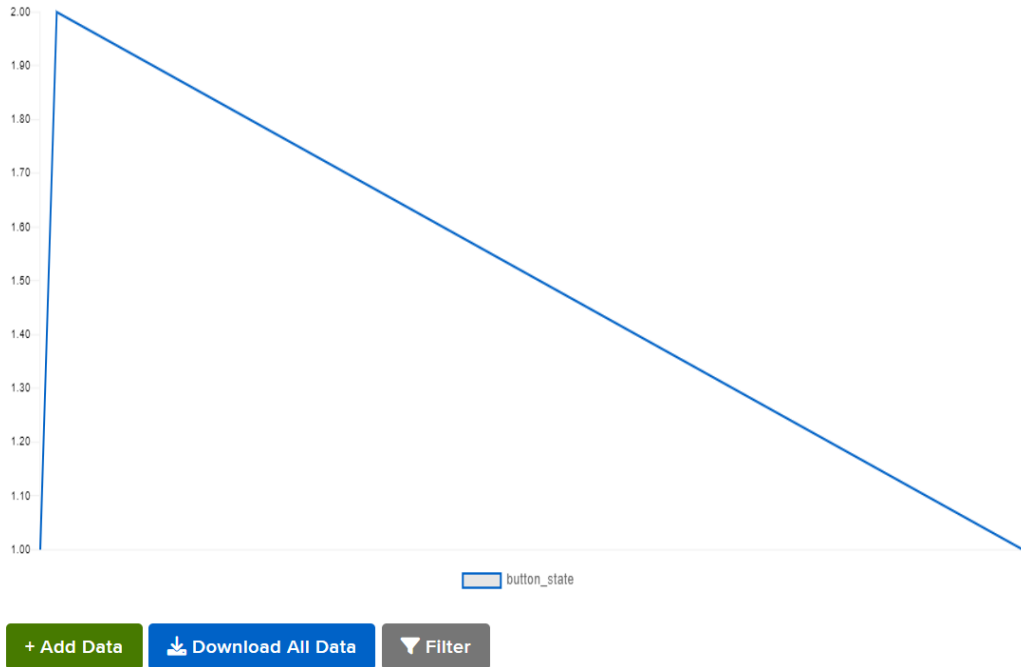
New to WipperSnapper?
[Follow this guide to get connected!](#)

Next, Create a New feed by selecting the [+ New Feed](#) button. “Feeds” are specific places that the IoT Bit can send data to. Make sure to name your feed so that it corresponds to the data that you’ll be sending it. For example, if the feed will be

receiving data about whether a button is on or off, you might name the feed “button_state.”

The feed will be empty because the IoT bit hasn’t uploaded any data to it yet. To get the “ball rolling,” you can manually add data by pressing the [+ Add Data](#) button. You’ll also notice that the feed has a graph to show the history of values in the feed.

[Mark_IoT_Bit](#) / [Feeds](#) / button_state



2. Load your WiFi credentials and Adafruit IO information onto the SD card

To connect the IoT bit to the feed in Adafruit IO, we'll have to provide the IoT bit with WiFi credentials and some feed information.

Remove the MicroSD card from the IoT Bit and connect it to your computer (via an adapter, if necessary. If you do not have an adapter, ask a TA for one.) When you open the secrets.txt file on the SD card, you'll see the contents of 7 variables, separated by semicolons:

```
REQUEST_RATE_SEC;SECRET_SSID;SECRET_PASS;IO_USERNAME;IO_GROUP;IO_FEED_KEY;IO_KEY
```

Here's what all of those variables mean:

REQUEST_RATE_SEC	The number of seconds in between each request the IoT Bit makes to the cloud. There is a request limit per minute with the free version, so leave at 2 seconds.
SECRET_SSID	The SSID of your WiFi network, i.e. the name. If you are using a phone as a mobile hotspot, it will be the name of your phone.
SECRET_PASS	The password of your WiFi network/hotspot. Keep the SSID and password a secret!
IO_USERNAME	The username of your Adafruit IO account, not necessarily the name of your general Adafruit account.
IO_GROUP	The name of the group that the feed you will post to/get from resides in. Feeds will typically be created in the "default" group.
IO_FEED_KEY	<p>The feed's key is the "password" that gives the IoT bit permission to get data from and post data to the feed.</p> <p>The feed key can be found in the "feed info" widget.</p>

	<div><div>Mark_IoT_Bit / Feeds / button_state</div><div><div>1.0 0.9 0.8 0.7 0.6 0.5 0.4 0.3 0.2 0.1 0</div><div>Sep 12<div>button_state</div>Sep 13</div></div><div><div>Feed Info</div><div>Manage feed name, key, description, and tags.</div><div>Privacy</div><div>This feed is: private Only you can see it.</div><div>Sharing</div><div>Not shared yet</div><div>Feed History</div><div>Feed history is ON Value size is limited</div></div><div><div>Name</div><div>button_state</div><div>Maximum length: 128 characters. Used: 12</div><div>Key</div><div>button-state</div><div>Changing the key will change API URLs and MQTT subscription topics. The only characters we permit are lower case english letters ("a" to "z"), numbers, and dash ("-"). See our guide to naming things in Adafruit IO for more information about how we handle the formatting of names and keys.</div></div></div>								
IO_KEY	<div><div>Your account's IO key is like the feed key, but associated with your entire Adafruit IO key, as opposed to a single feed. It can be found within the yellow key icon:</div><div><div>DevicesFeedsDashboardsActionsPower-Ups</div><div><div>ESP8266 based devices may be impacted due to SSL certificate updates, please see this post on our forums to ensure you can continue to connect to IO.</div><div>Mark_IoT_Bit / Feeds</div><div>New FeedNew Group</div><div>Default</div><div><table><tr><th>Feed Name</th><th>Key</th><th>Last value</th><th>Recorded</th></tr><tr><td><input type="checkbox"/> button_state</td><td>button-state</td><td></td><td>6 minutes ago</td></tr></table></div><div>Loaded in 0.13 seconds.</div></div></div></div>	Feed Name	Key	Last value	Recorded	<input type="checkbox"/> button_state	button-state		6 minutes ago
Feed Name	Key	Last value	Recorded						
<input type="checkbox"/> button_state	button-state		6 minutes ago						

YOUR ADAFRUIT IO KEY

Your Adafruit IO Key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your Adafruit IO Key can view all of your data, create new feeds for your account, and manipulate your active feeds.

If you need to regenerate a new Adafruit IO Key, all of your existing programs and scripts will need to be manually changed to the new key.

Username

Mark_IoT_Bit

Active Key

aio_zfel7118EMztX8g6omtckAZacx1N

REGENERATE KEY

[Hide Code Samples](#)

Arduino

```
#define IO_USERNAME  "Mark_IoT_Bit"
#define IO_KEY       "aio_zfeI7118EMztX8g6omtckAZacx1N"
```

Linux Shell

If your key becomes compromised, you can regenerate a character string to keep the hackers off of your trail.

Here's an example of what a secrets.txt file might look like with your data replacing the variable names:

2;Mark_iPhone;MarkPassword;AOMUser1;AOMGroupA;AOMGroupAFeed1;AoMloKeYsEcReT123

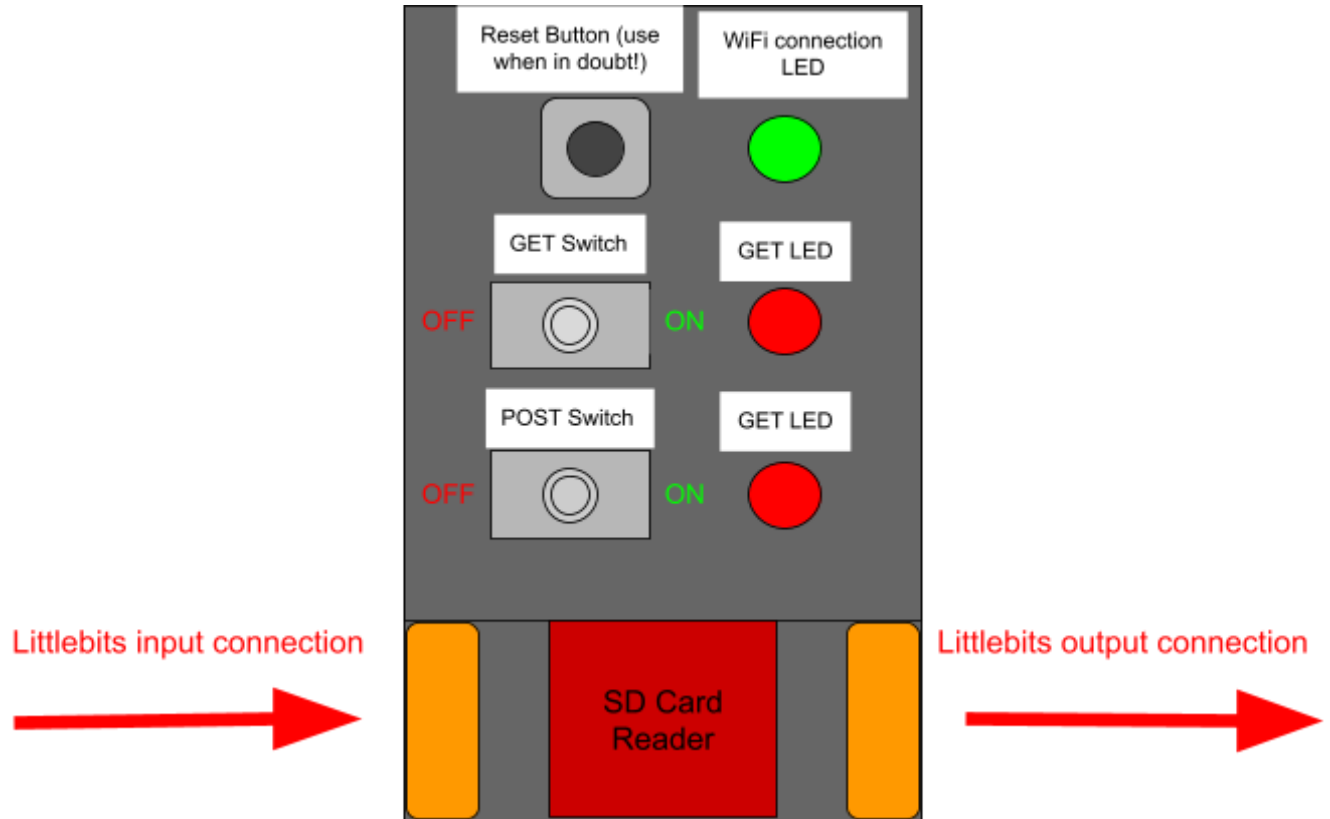
Important: If you are using a mobile hotspot, be sure to turn on the “Maximize Compatibility” option in the hotspot configuration, or the IoT bit will not be able to connect.

3. Connect

You're ready to let your IoT Bit talk to the world! Snap your IoT bit in series with (powered) Littlebits and let the IoT Bit configure itself. Here are the steps that you will see the bit go through:

1. **All 3 LEDs will blink 3 times quickly** - This indicates that the IoT bit is going to attempt to read from the SD card. This flashing will repeat every ~5 seconds until the IoT Bit reads the SD card's data successfully.
2. **All 3 LEDs will blink for a full second** - This indicates that the IoT bit has successfully read from the SD card and is ready to connect to the cloud.
3. From here, the IoT bit will enter a loop **depending on the state of the switches**. Below are the 4 operations:
 - a. **No switches are turned on** - disconnect from WiFi and remain idle.
 - b. **The GET Switch is turned on** - if the IoT bit is disconnected from WiFi, it will attempt to connect and illuminate the blue LED when it has succeeded. It will then "get" the latest value from the cloud. If the latest value in the cloud is "HIGH," the IoT bit will output a HIGH electrical signal. If the latest value in the cloud is "LOW," the IoT bit will output a LOW electrical signal.
 - c. **The POST Switch is turned on** - if the IoT bit is disconnected from WiFi, it will attempt to connect and illuminate the blue LED when it has succeeded. It will then detect the digital input signal to the IoT bit and "post" it to the cloud every time there is a change in the input value. If the IoT Bit's electrical input signal value changes to "HIGH," the IoT bit will post a HIGH value to the cloud. If the IoT Bit's electrical input signal value changes to "LOW," the IoT bit will post a LOWvalue to the cloud.
 - d. **The GET switch and POST switch are both turned on** - the IoT bit will both POST the incoming signal to Adafruit IO **and** GET the signal back from the cloud. Use this if you want to upload a value to the cloud as well as send that signal through the IoT bit to sequential LittleBits.

Here's a diagram of the interface's functionality:



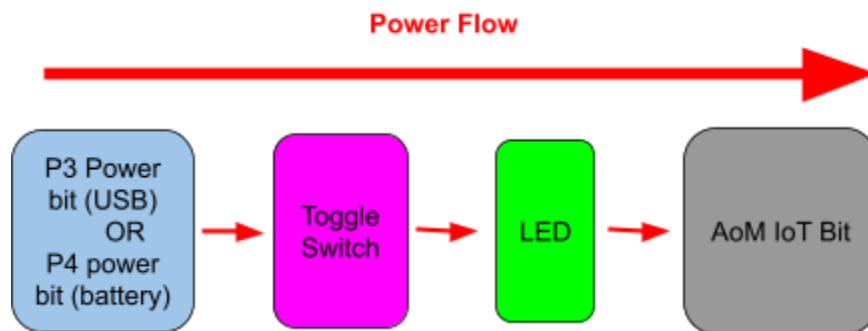
“Hello, World” Applications

1. Post to IO (and IFTTT!)

You will need:

- Littlebits p4 power bit (1)
- Littlebits toggle switch, flipped to the off position (1)
- Littlebits LED (1)
- AoM IoT Bit (1)

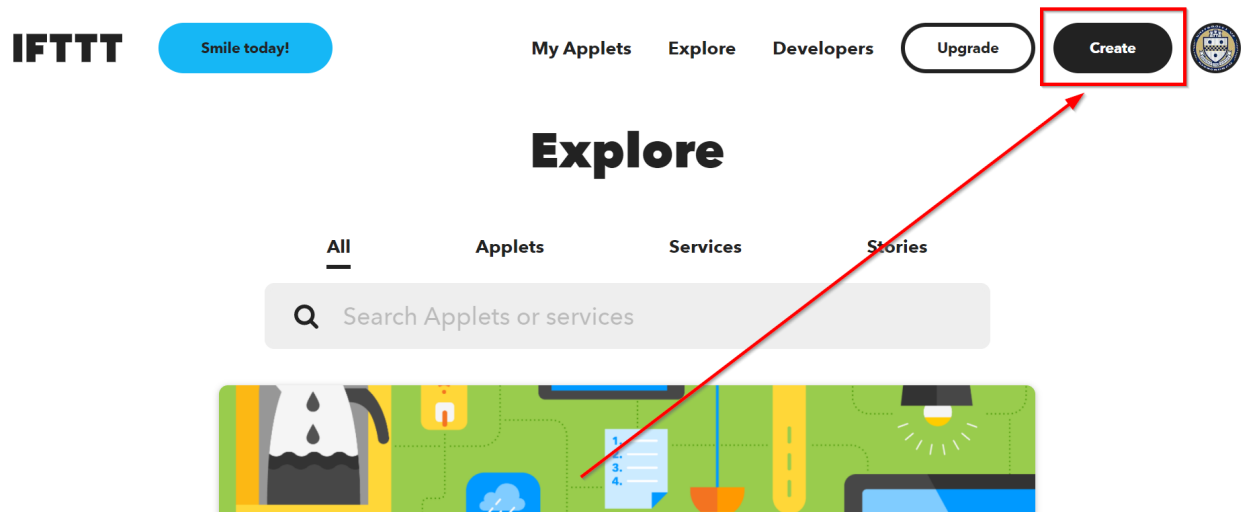
a. Build the circuit as shown below:



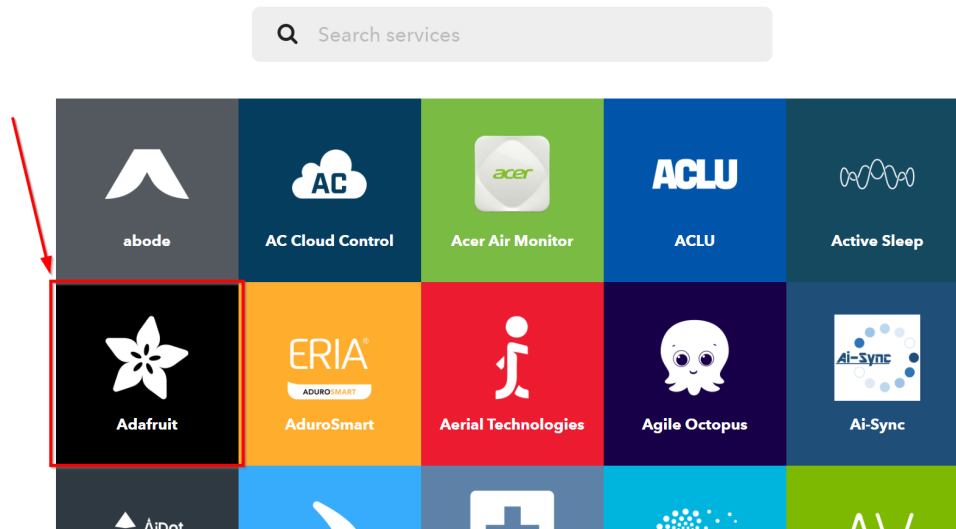
- b. Once the IoT Bit is powered, flip the “post” switch from left to right to the “ON” position.
- c. Once the green LED on the IoT Bit illuminates to indicate a WiFi connection, flip the Littlebits toggle switch state from off to on. The littlebits LED should illuminate, and the post LED will pulse to indicate a successful upload. Check the data stream on your Adafruit IO feed - you should see a new “HIGH” value uploaded to the feed. The IoT Bit will upload the IoT bit’s input signal to the cloud every time that input signal changes from LOW to HIGH or HIGH to LOW.

d. **Bonus - Connect your IoT bit to IFTTT**

Create an account on [IFTTT](https://ifttt.com) and navigate to the “create” button:



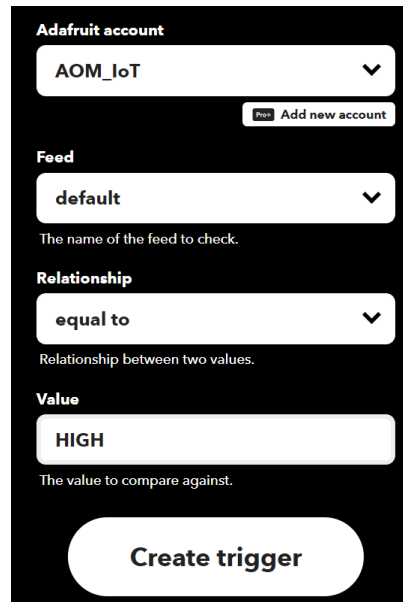
This will prompt you to create an “applet.” In the “If this” field of the applet, select “Add” and select the Adafruit IO service, as shown:



Select “Monitor a feed on Adafruit IO” as your trigger. Add your Adafruit IO username, feed name, set the relationship and “equal to,” and select “HIGH” as the value. This will trigger the “If this” portion of the widget any time the IoT bit uploads a HIGH value to Adafruit IO.

Monitor a feed on Adafruit IO

This Trigger fires anytime it validates the data that you send to your feed. Example: If Feed Temperature > 80, fire Trigger.



The screenshot shows the configuration interface for a trigger on the Adafruit IO platform. It features a dark background with white text and input fields. The sections are: 'Adafruit account' with a dropdown menu showing 'AOM_IoT' and an 'Add new account' button; 'Feed' with a dropdown menu showing 'default' and a description 'The name of the feed to check.'; 'Relationship' with a dropdown menu showing 'equal to' and a description 'Relationship between two values.'; and 'Value' with a text input field containing 'HIGH' and a description 'The value to compare against.' At the bottom is a large 'Create trigger' button.

Adafruit account

AOM_IoT

Add new account

Feed

default

The name of the feed to check.

Relationship

equal to

Relationship between two values.

Value

HIGH

The value to compare against.

Create trigger

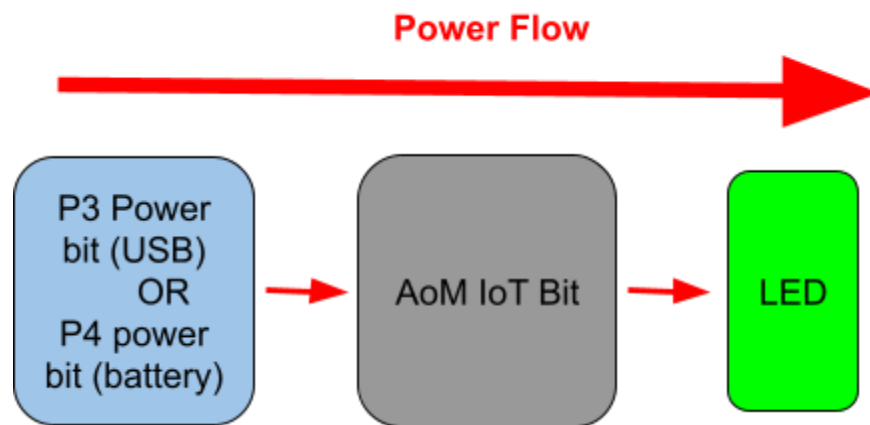
In the “Then That” field of the widget, select any web service that you want to interface to - it’s up to you! A simple “Hello World” test is the Google Tasks service, which allows you to create a task every time the Adafruit IO cloud receives a high value. But you’re only limited by your imagination - and the limits of the free IFTTT account, of course.

2. Get from IO

You will need:

- Littlebits p4 power bit (1)
- Littlebits LED (1)
- AoM IoT Bit (1)

a. Build the circuit as shown below:



- b. Once the IoT Bit is powered, flip the “get ” switch from left to right to the “ON” position.
- c. Once the green LED illuminates to indicate a WiFi connection, the GET switch will begin to pulse, indicating that the output of the IoT bit is reflecting the data in the cloud - i.e. if the most recent data point in the cloud is HIGH, the IoT bit will output a HIGH signal. If the most recent data point in the cloud is LOW, the IoT bit will output a LOW signal. This behavior will be reflected by the LED.

You’ve (hopefully) mastered the very basics of the proprietary AoM IoT bit. Congratulations - the control of an infinite number of objects is now possible with the flip of a switch, from **anywhere** that can connect to WiFi. Don’t let the power get to your head - but go out and make incredible devices. However, I simply must emphasize this once more:

This was created with design thinking and is a work in progress. The technology is snazzy and fancy, and that is all well and good, but what is most important is your experience. Which means that you should give us (honest!) user feedback - what works? What doesn’t? What’s awful? DON’T hesitate to reach out to me!