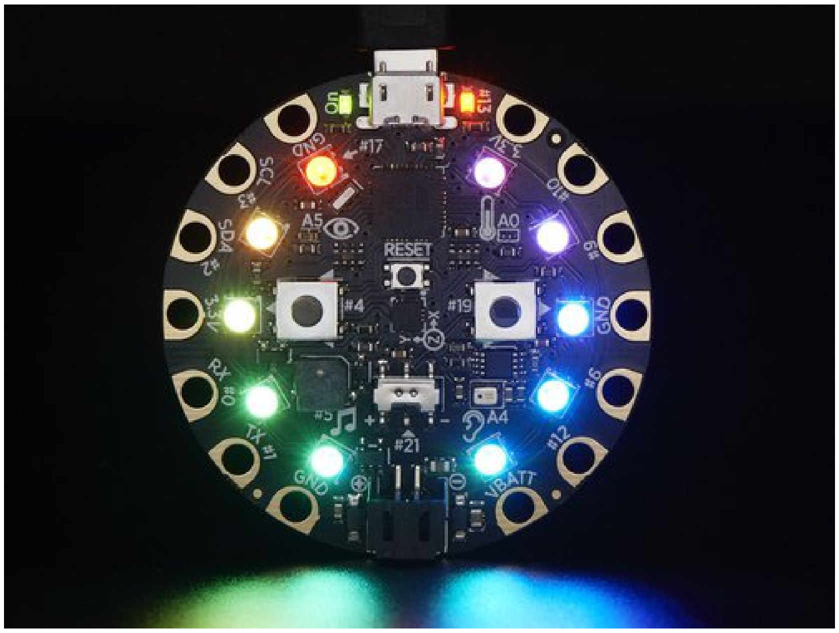
Introducing Circuit Playground

Created by lady ada



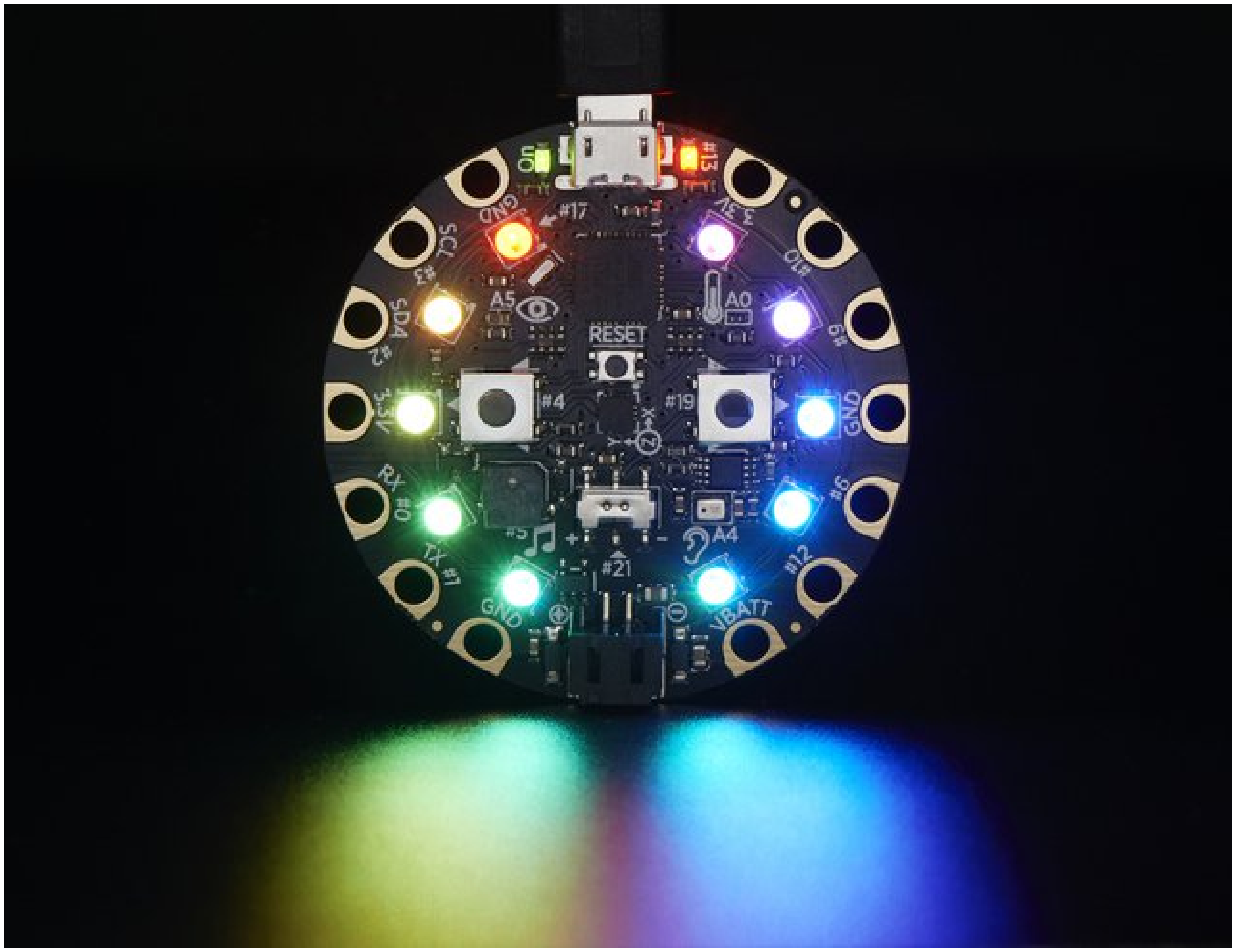
Last updated on 2016-09-20 07:25:12 PM UTC

Guide Contents

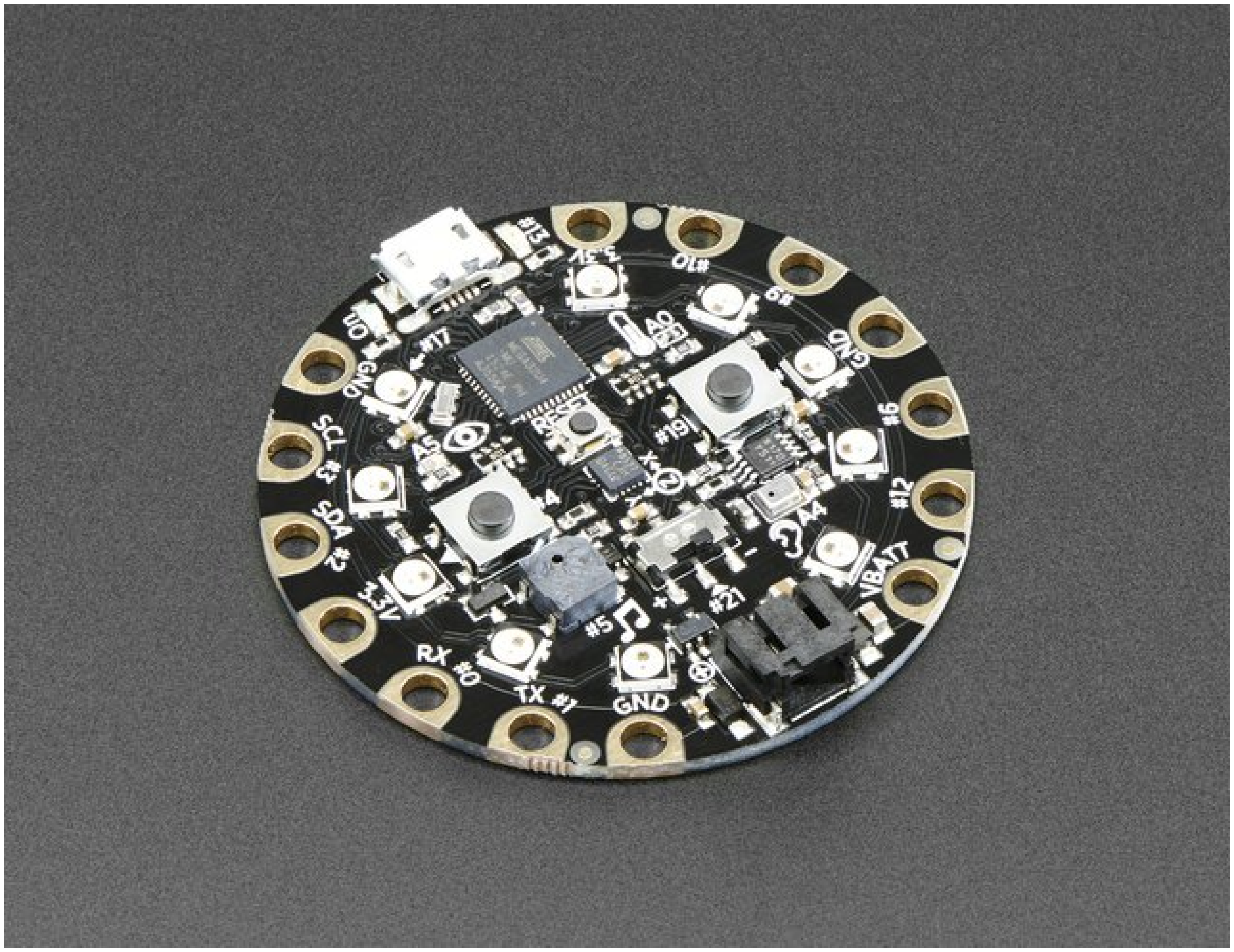
Overview

Would you like to learn electronics, with an all-in-one board that has sensors and LEDs built in? Circuit Playground is here, and it's the best way to practice programming on real hardware. No soldering or sewing required!

Circuit Playground features an ATmega32u4 micro-processor, just like our popular Flora. It also is round and has alligator-clip pads around it. You can power it from USB, a AAA battery pack (http://adafru.it/727), or Lipoly (for advanced users). Program your code into it, then take it on the go.



- ATmega32u4 Processor, running at 3.3V and 8MHz
- MicroUSB port for programming and debugging with Arduino IDE
- USB port can act like serial port, keyboard, mouse, joystick or MIDI



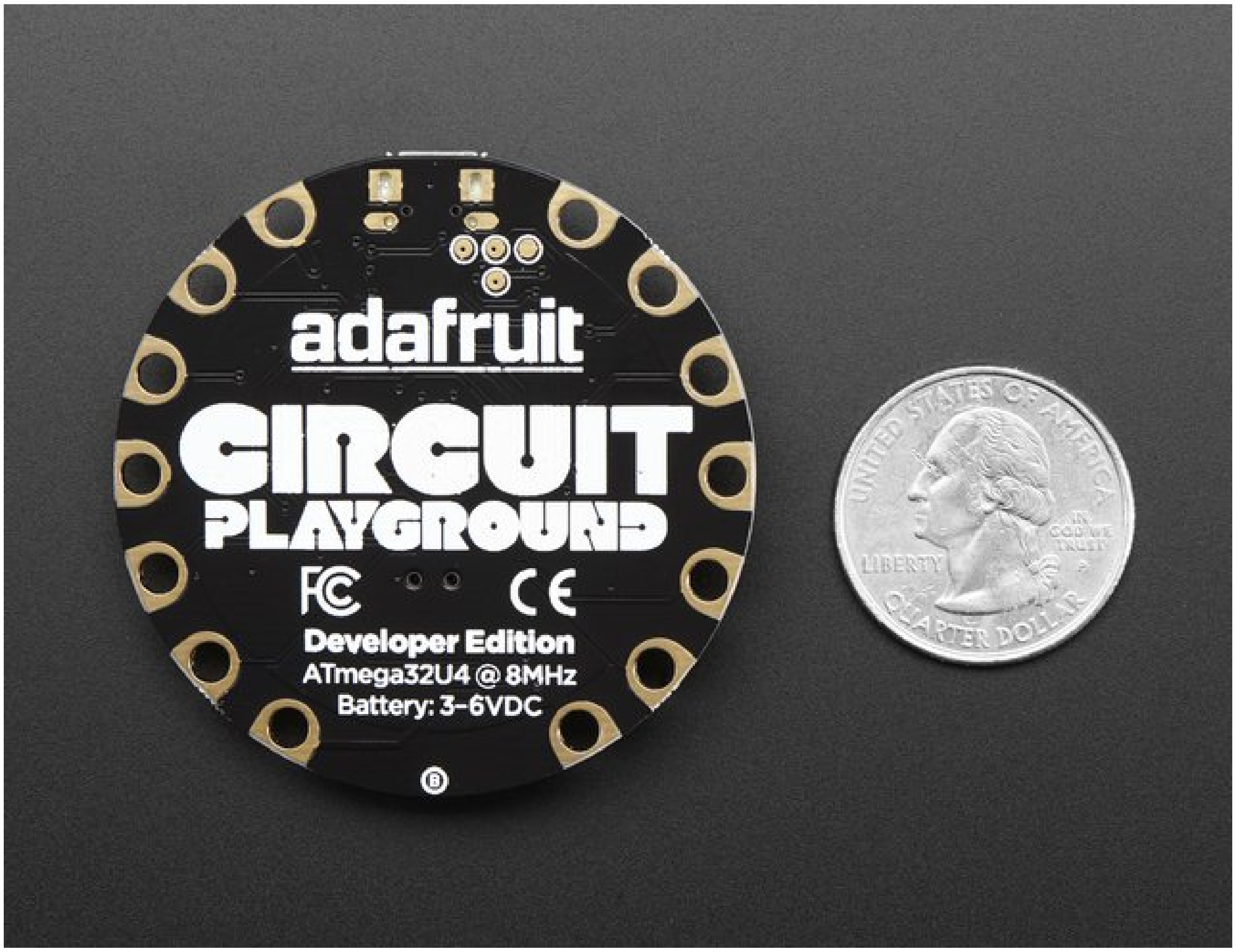
Circuit Playground has built-in USB support. Built in USB means you plug it in to program it, it just shows up - all you need is a Micro-B USB cable, no additional purchases are needed! With the new 1.6.4+ Arduino IDE, it takes only a few seconds to add support (http://adafru.it/nb4). The Circuit Playground has USB HID support, so it can act like a mouse or keyboard to attach directly to computers.

Here's some of the great goodies baked in:

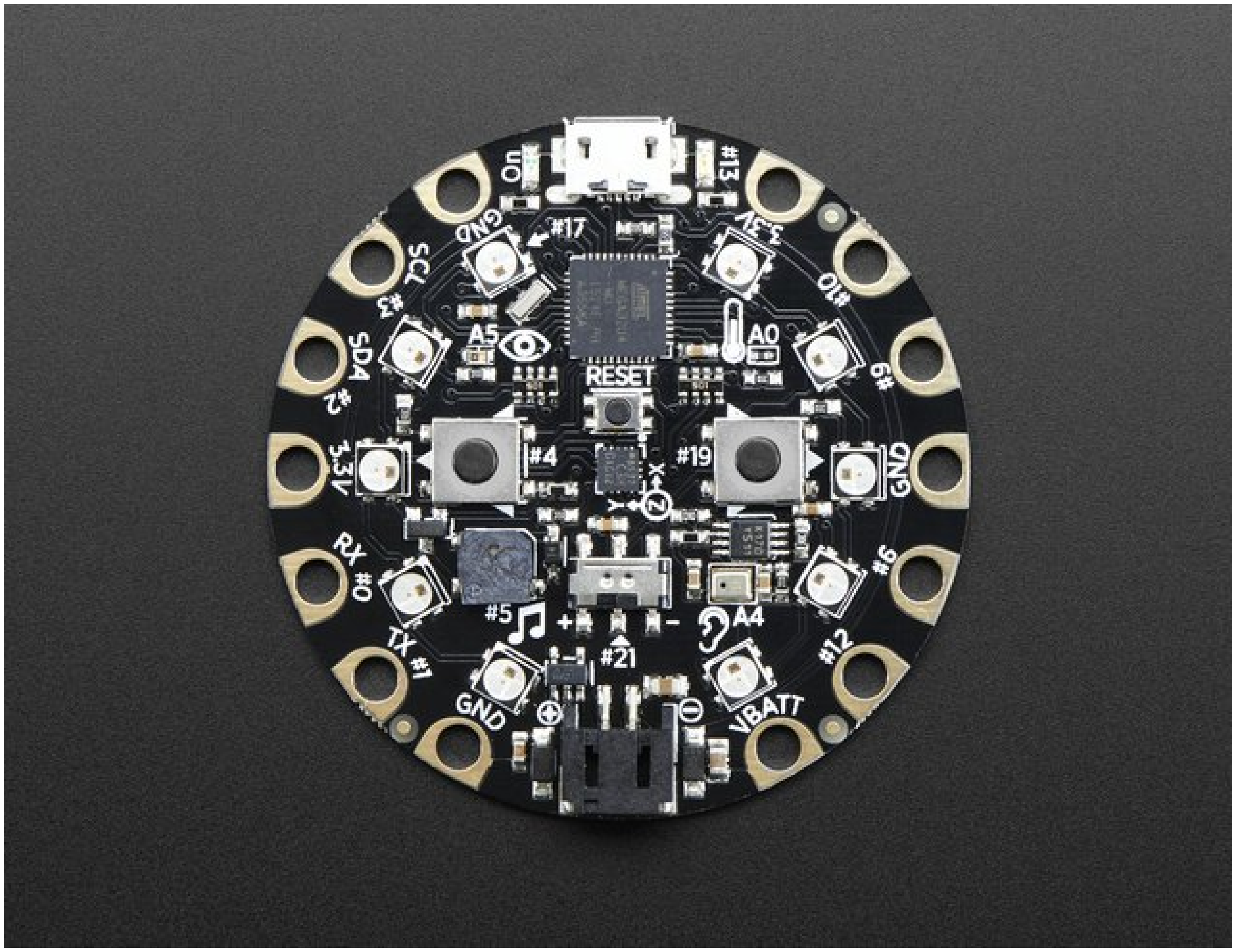
- 10 x mini NeoPixels, each one can display any rainbow color
- 1 x Motion sensor (LIS3DH triple-axis accelerometer with tap detection, free-fall detection)
- 1 x Temperature sensor (thermistor)
- 1 x Light sensor (phototransistor)
- 1 x Sound sensor (MEMS microphone)
- 1 x Mini speaker (magnetic buzzer)
- 2 x Push buttons, left and right
- 1 x Slide switch
- 8 x alligator-clip friendly input/output pins

Includes I2C, UART, and 4 pins that can do analog inputs/PWM output
- All 8 pads can act as capacitive touch inputs
- Green "ON" LED so you know its powered
- Red "#13" LED for basic blinking
- Reset button

We've started out with a **Developer Edition** of Circuit Playground. This version is designed for people who have a little experience with Arduino already, who want to help build & document projects. There might be minor hardware or software bugs. Once we feel like the design is really solid we'll revise/re-release it into a universal edition for anyone to use!

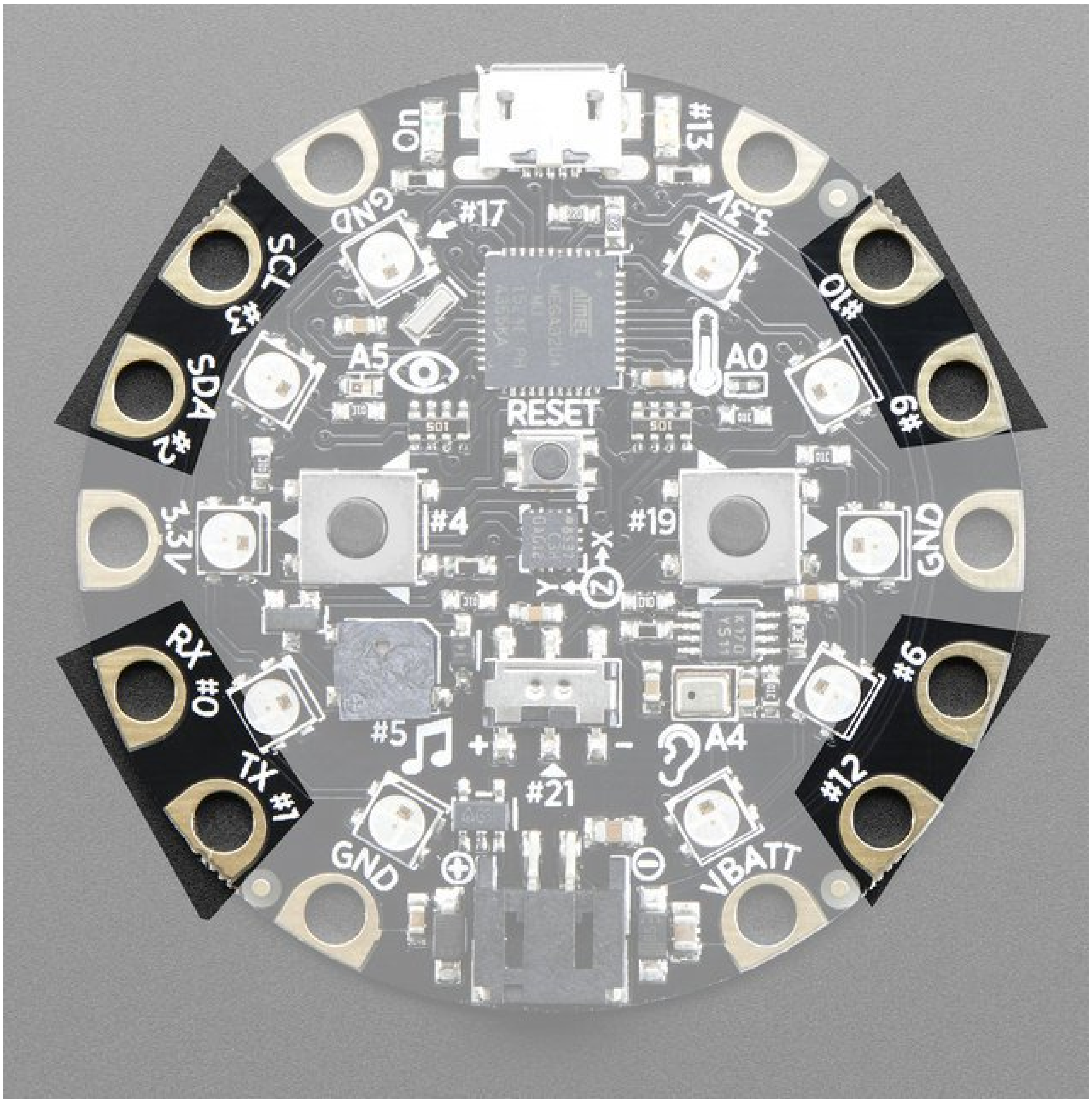


Pinouts



Circuit Playground is chock-full of blinkies, sensors and electronic goodies. Since there's a lot going on, we're going to look at each piece one by one and talk about what it is and how to use it. This is a technically-oriented description for the **Developer Edition** and the people who are planning on writing code for Circuit Playground

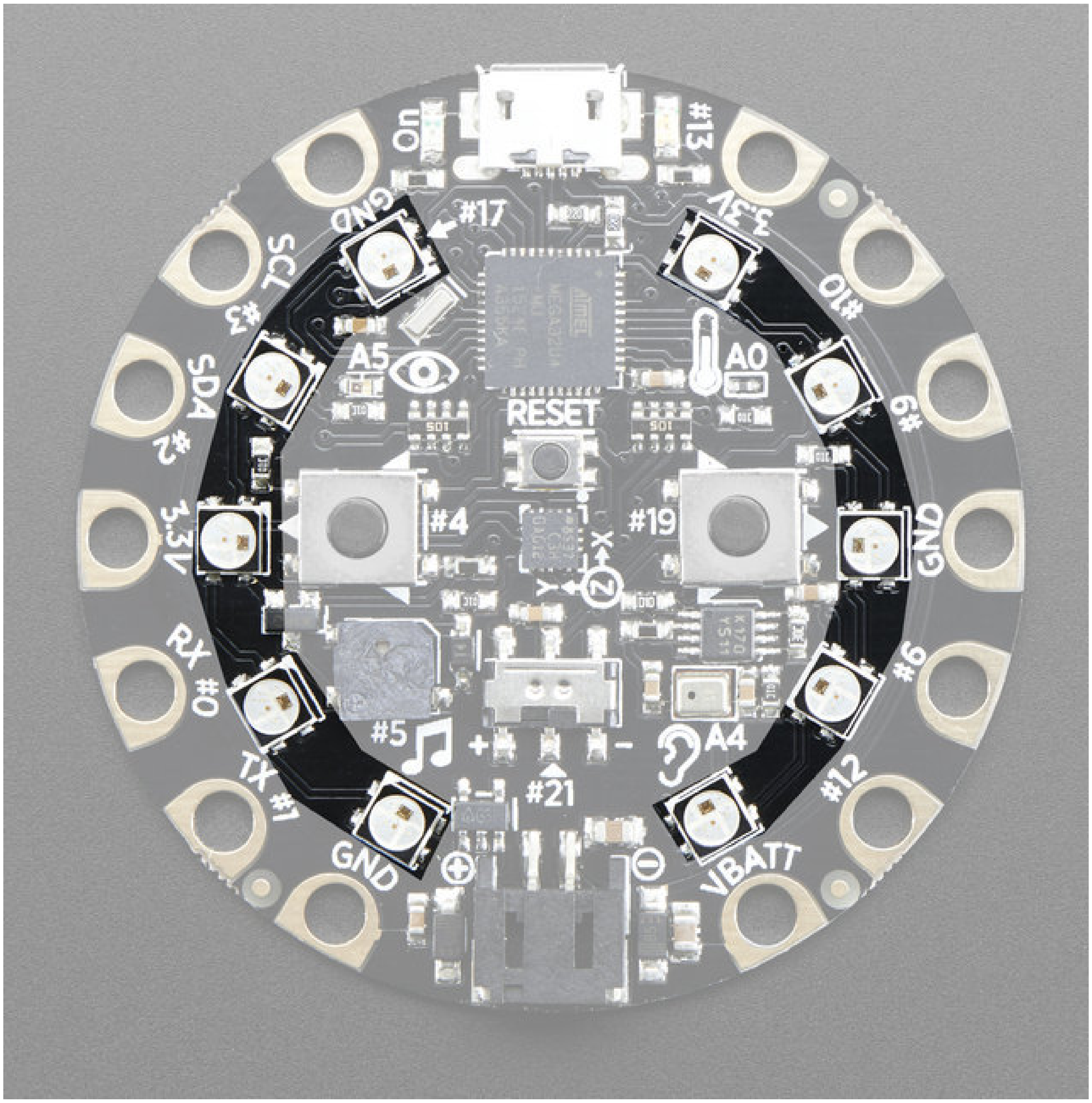
GPIO + Capacitive Touch Pads



All 8 non-power pads around the circuit playground have the ability to act as capacitive touch pads. Each pad has a 1Mohm resistor between it and digital pin #30. You can toggle this pin to control whether the resistor is a pullup or pulldown or floating. Note that this means that all the pads have a 2Mohm resistance between them.

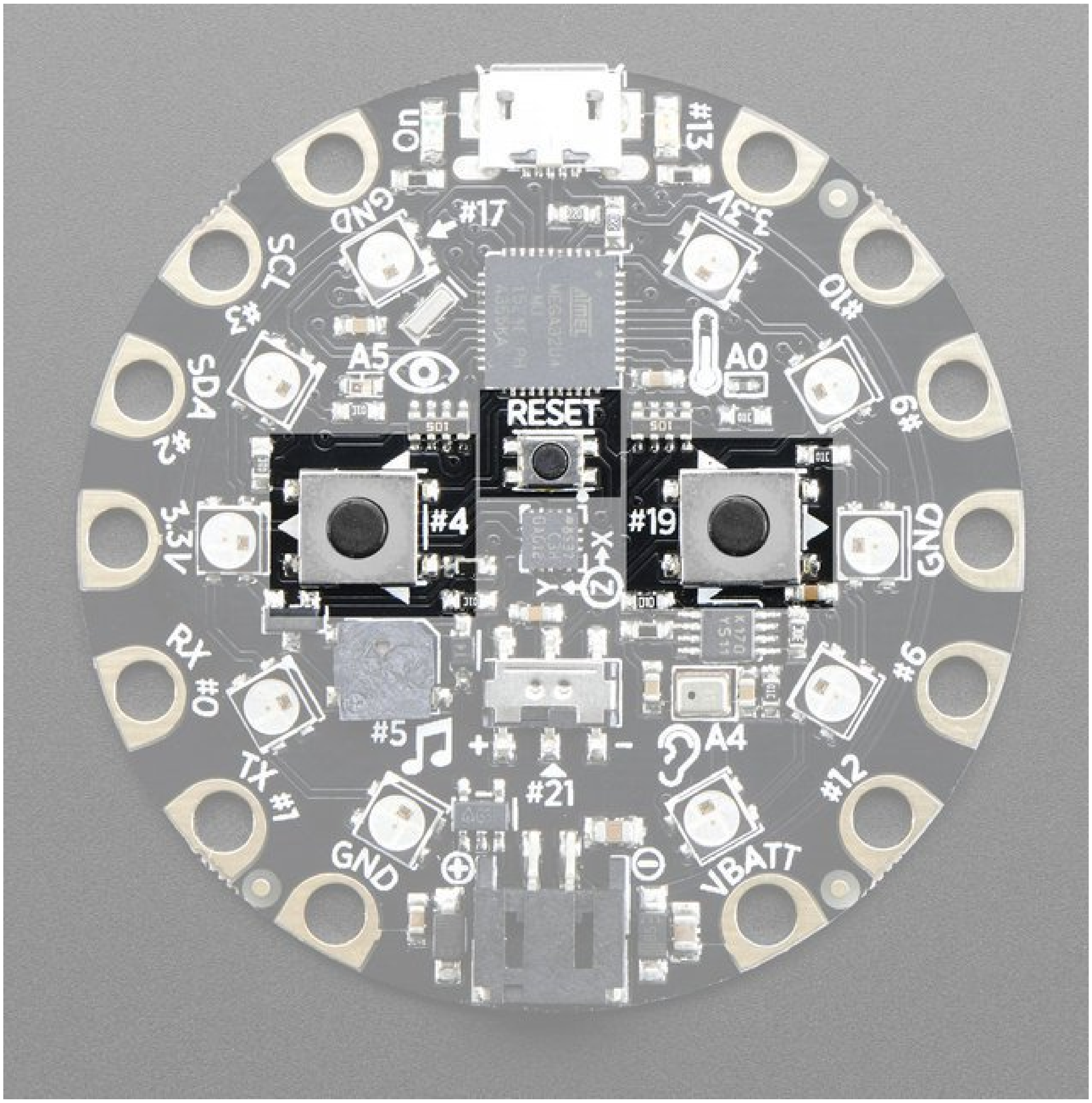
You can also of course just use those pads for GPIO, we expose the hardware Serial (TX + RX), hardware I2C (SDA + SCL) and 4 gpio pins that can also do analog readings. They are the same exact pins as those on the Flora

NeoPixels



Each Circuit Playground comes with 10 'NeoPixels' (technically, SK6812-3535 chips). These are connected to digital pin **#17** and are powered by the 3.3V regulator. This is technically undervolting but we test them at this voltage and they work fine, if slightly tinted

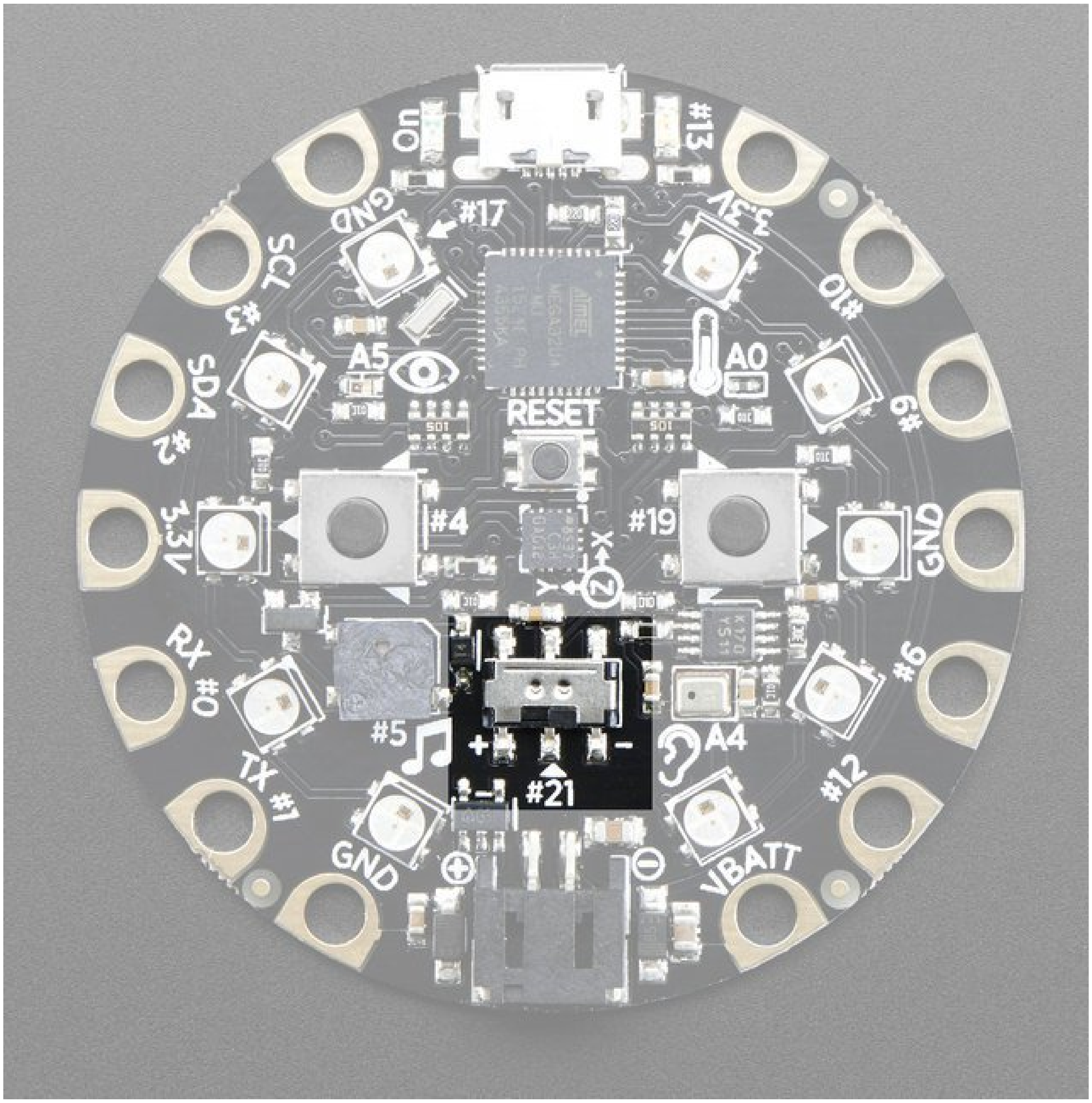
Pushbuttons



There are three tactile pushbutton switches. One is the Reset button. Press this button once to reset, double-click to enter the bootloader manually.

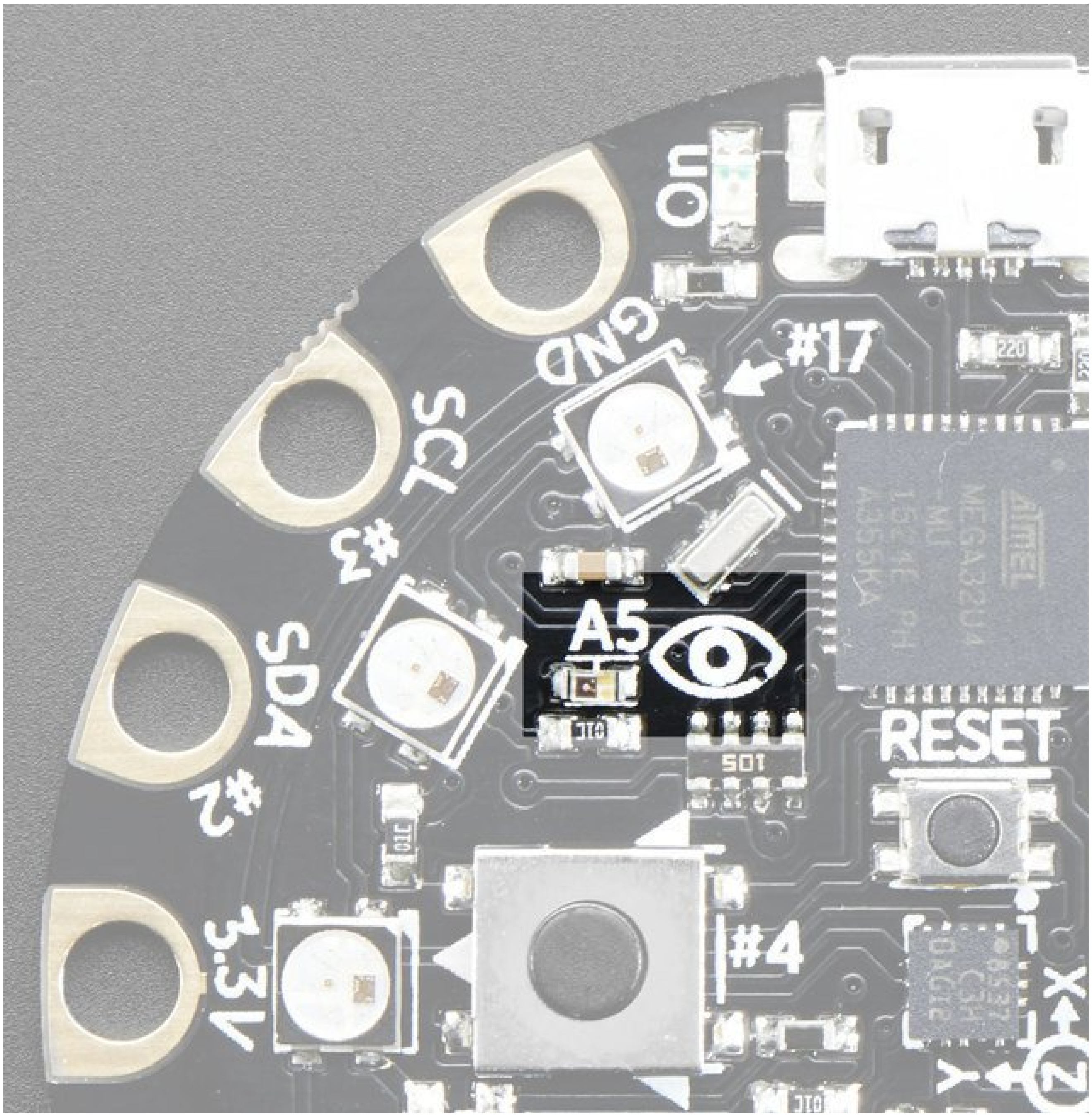
The other two buttons are the Left and Right buttons, connected to digital **#4** (Left) and **#19** (Right) each. These have *pull-down* resistors installed so are, by default, LOW and when pressed read HIGH. This is to make if-then logic a little easier to read for beginners

Slide Switch



There is a single slide switch near the center of the Circuit Playground. It is connected to digital **#21** and will read LOW when slid to the left, and HIGH when in the right hand position

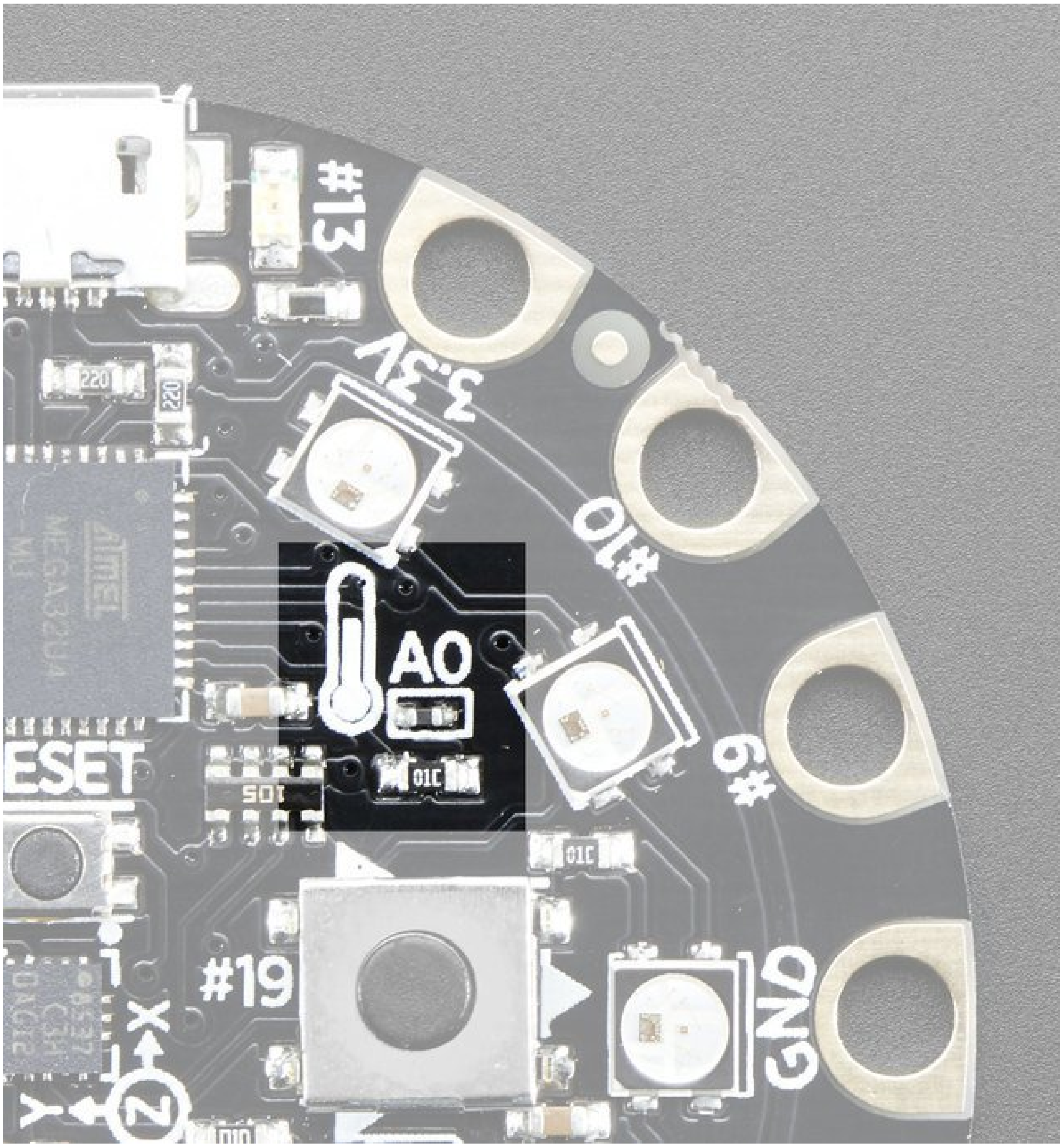
Light Sensor



There is an analog light sensor, part number ALS-PT19 (http://adafru.it/n8F), in the top left part of the board. This can be used to detect ambient light, with similar spectral response to the human eye.

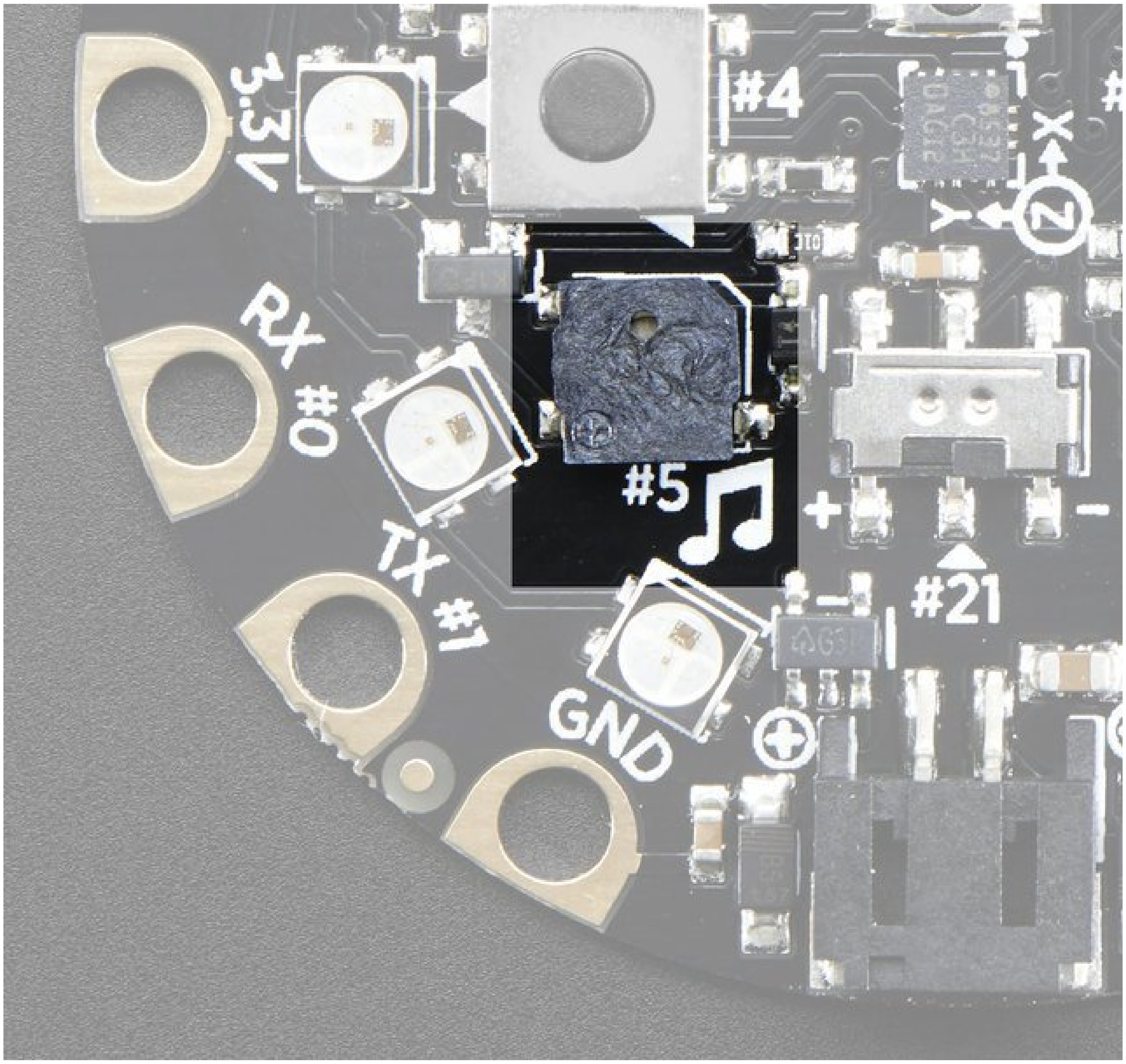
This sensor is connect to analog pin **#A5** and will read between 0 and 1023 with higher values corresponding to higher light levels. A reading of about 300 is common for most indoor light levels.

Temperature Sensor



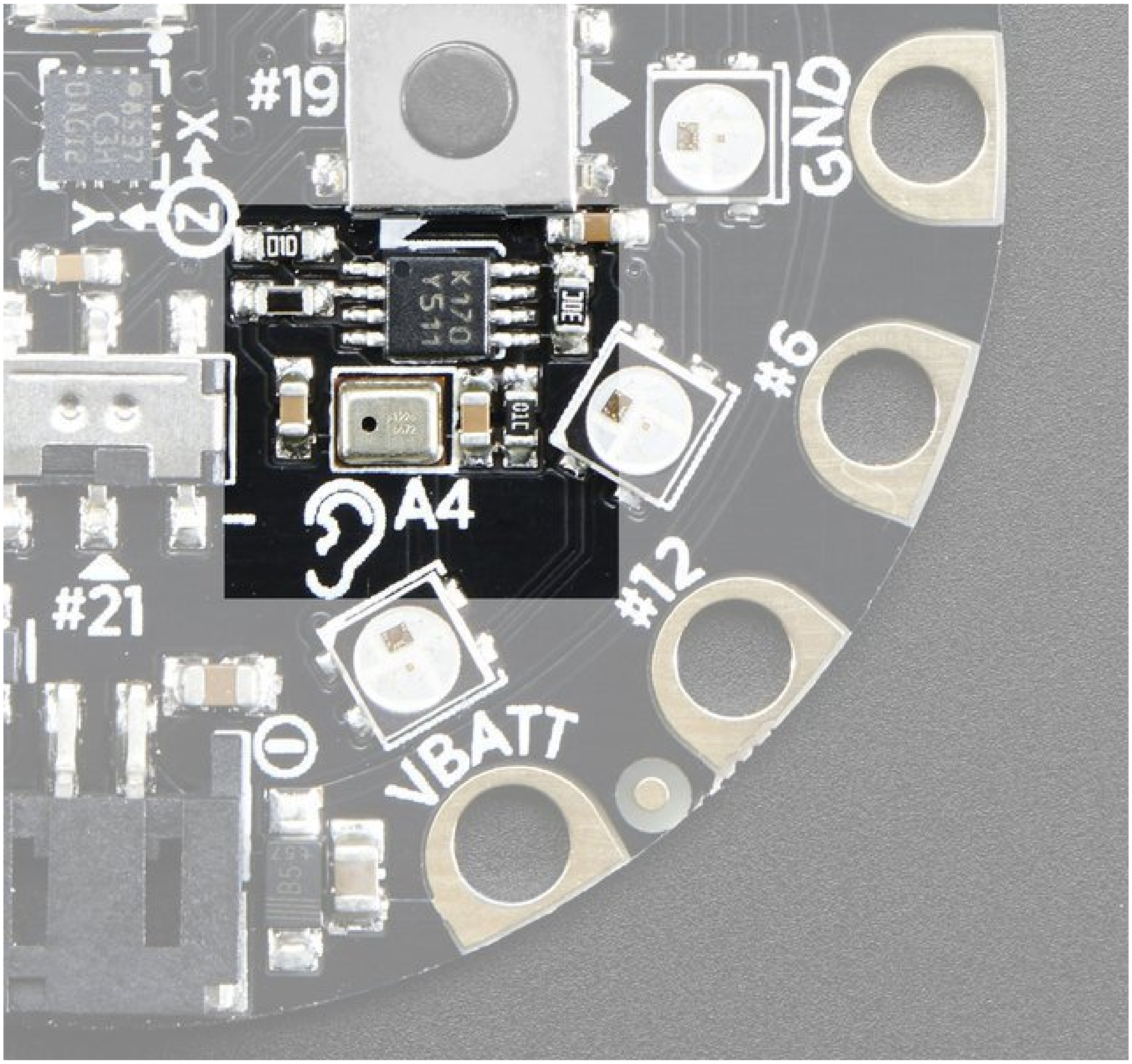
There is an NTC thermistor (Murata NCP15XH103F03RC) that we use for temperature sensing. While it isn't an all-in-one temperature sensor, with linear output, it's easy to calculate the temperature based on the analog voltage on analog pin **#A0**. There's a 10K resistor connected to it as a pull down.

Speaker Buzzer



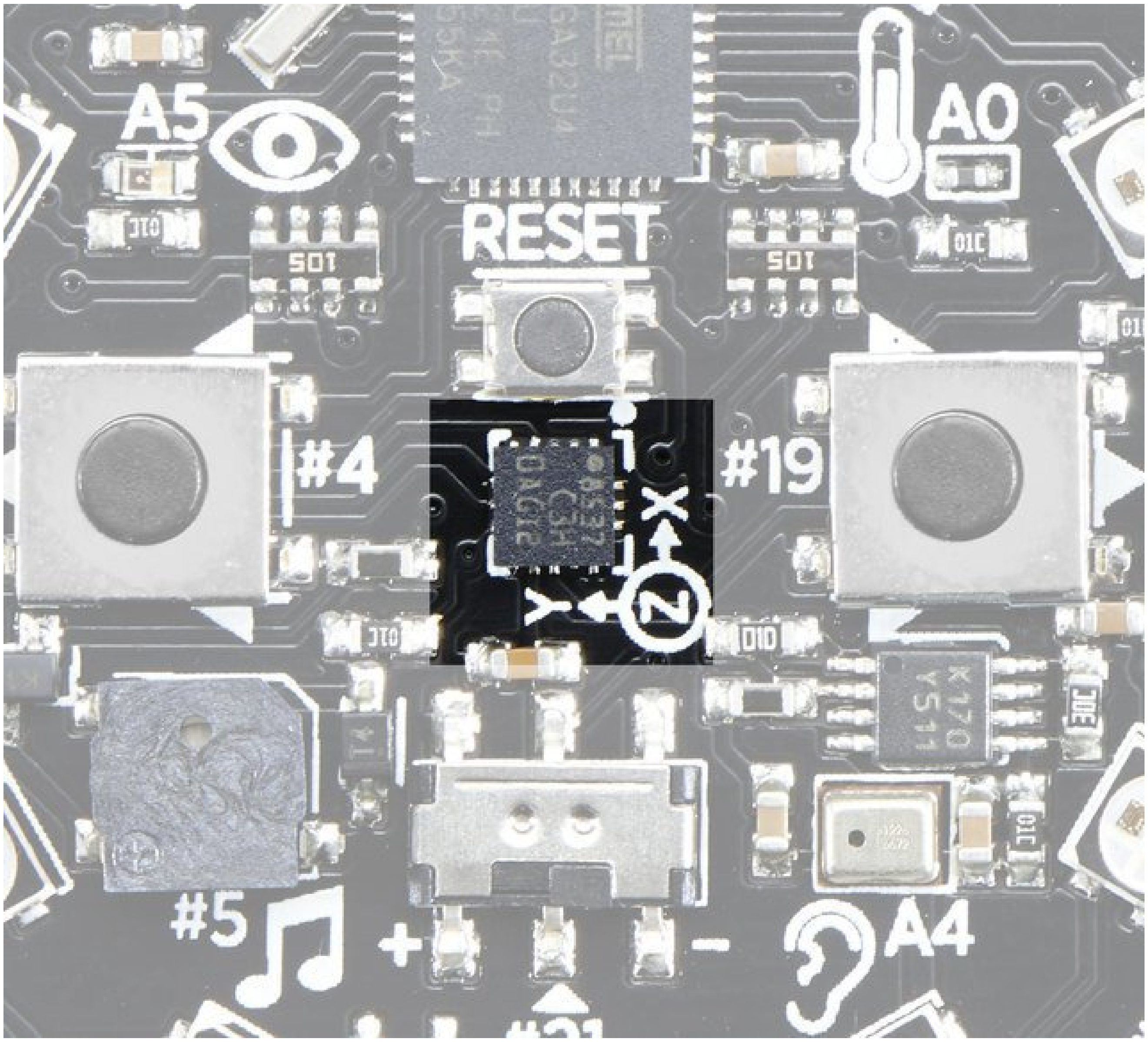
You can make your circuit playground sing with the built in buzzer. This is a miniature magnetic speaker connected to digital pin **#5** with a transistor driver. You can use PWM at varying frequencies to make basic tones.

Microphone Audio Sensor



A MEMS microphone can be used to detect audio levels and even perform basic FFT functions. You can read the analog voltage corresponding to the audio on analog pin **#A4**. Note that this is the raw analog audio waveform! When it's silent there will be a reading of ~330 and when loud the audio will read between 0 and 800 or so. Averaging and smoothing must be done to convert this to sound-pressure-level.

Triple-Axis Accelerometer



A LIS3DH 3-axis XYZ accelerometer is in the dead center of the board and you can use it to detect tilt, gravity, motion, as well as 'tap' and 'double tap' strikes on the board. The LIS3DH is connected to the hardware SPI pins (to leave the I2C pins free) and has the CS pin on digital pin **#8** and an optional interrupt output on digital pin**#7** (also known as IRQ #4)

Windows Driver Installation

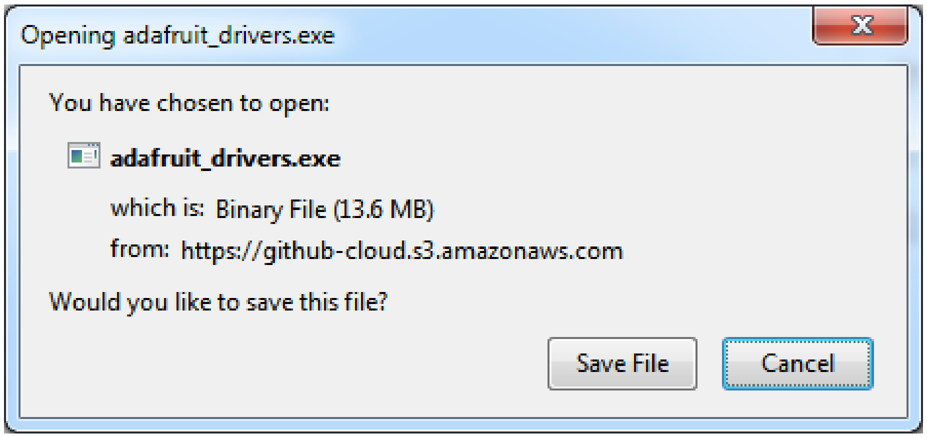
Mac and Linux do not require drivers, only Windows folks need to do this step

Before you plug in your board, you'll need to possibly install a driver!

Click below to download our Driver Installer

Download Adafruit Windows Driver Installer
http://adafru.it/mai

Download and run the installer


Opening adafruit_drivers.exe

You have chosen to open:

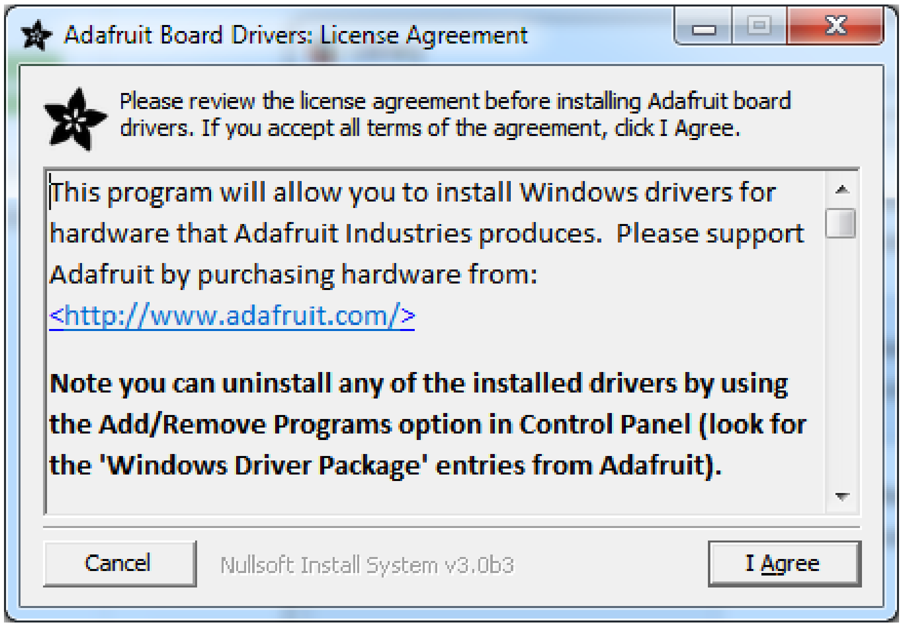
adafruit_drivers.exe

which is: Binary File (13.6 MB)

from: https://github-cloud.s3.amazonaws.com

Would you like to save this file?

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

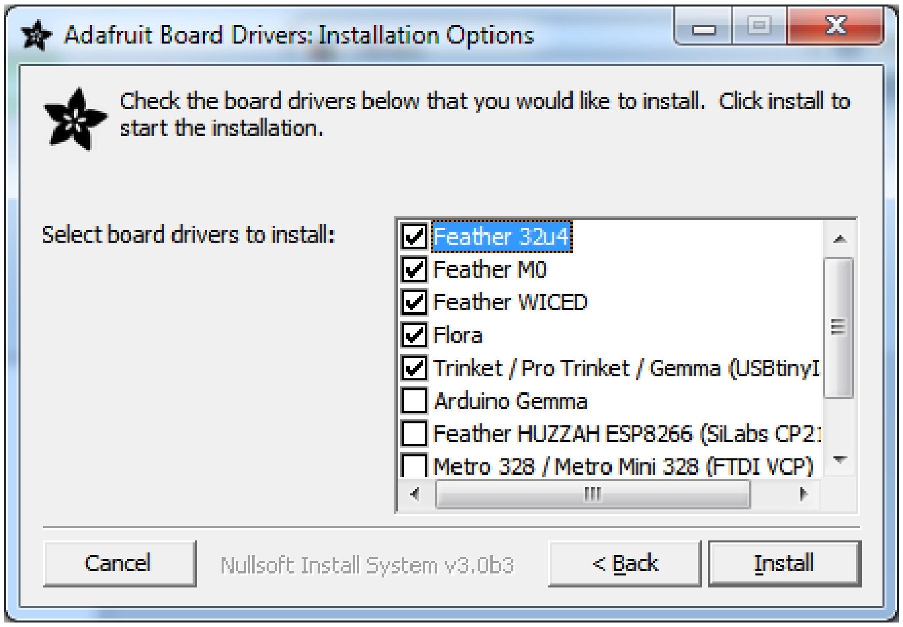

Adafruit Board Drivers: License Agreement

Please review the license agreement before installing Adafruit board drivers. If you accept all terms of the agreement, click I Agree.

This program will allow you to install Windows drivers for hardware that Adafruit Industries produces. Please support Adafruit by purchasing hardware from: <http://www.adafruit.com/>

Note you can uninstall any of the installed drivers by using the Add/Remove Programs option in Control Panel (look for the 'Windows Driver Package' entries from Adafruit).

Select which drivers you want to install, we suggest selecting all of them so you don't have to do this again!


Adafruit Board Drivers: Installation Options

Check the board drivers below that you would like to install. Click install to start the installation.

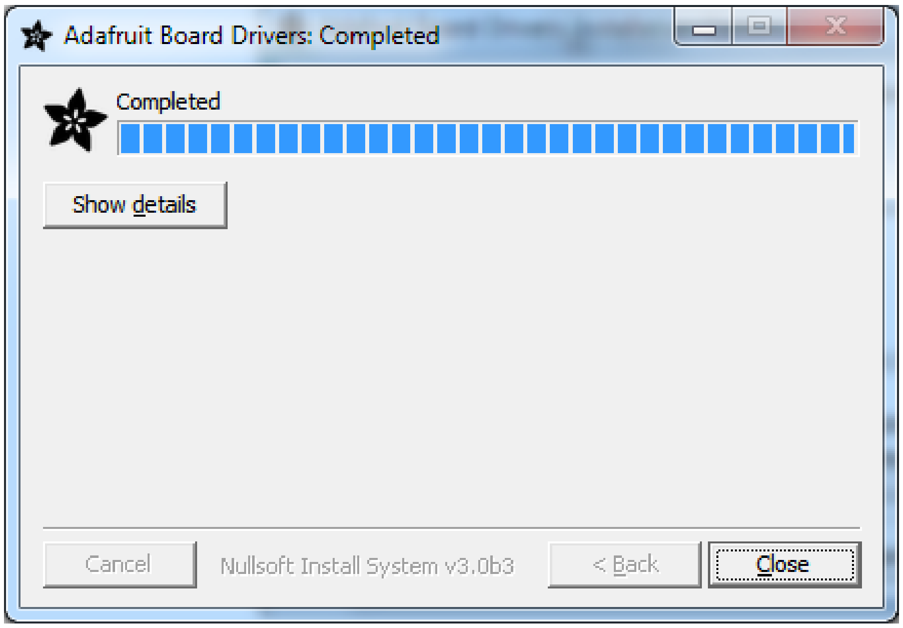
Select board drivers to install:

- [✓] Feather 32u4
- [✓] Feather M0
- [✓] Feather WICED
- [✓] Flora
- [✓] Trinket / Pro Trinket / Gemma (USBtinyI
- [ ] Arduino Gemma
- [ ] Feather HUZZAH ESP8266 (SiLabs CP21
- [ ] Metro 328 / Metro Mini 328 (FTDI VCP)

By default, we install the **Feather 32u4** , **Feather M0**, **Flora** and **Trinket** / **Pro Trinket** / **Gemma** / **USBtinyISP** drivers.

You can also, optionally, install the **Arduino Gemma** (different than the Adafruit Gemma!), Huzzah and Metro drivers

Click **Install** to do the installin'


Adafruit Board Drivers: Completed
Completed
Show details
Cancel
Nullsoft Install System v3.0b3
< Back
Close

Arduino 1.6.x IDE

The Arduino IDE version 1.6.x is the latest version of Arduino and recommended for most users. Follow the steps on this page to learn how to install a version of the Arduino 1.6.x IDE with support for Adafruit's boards.

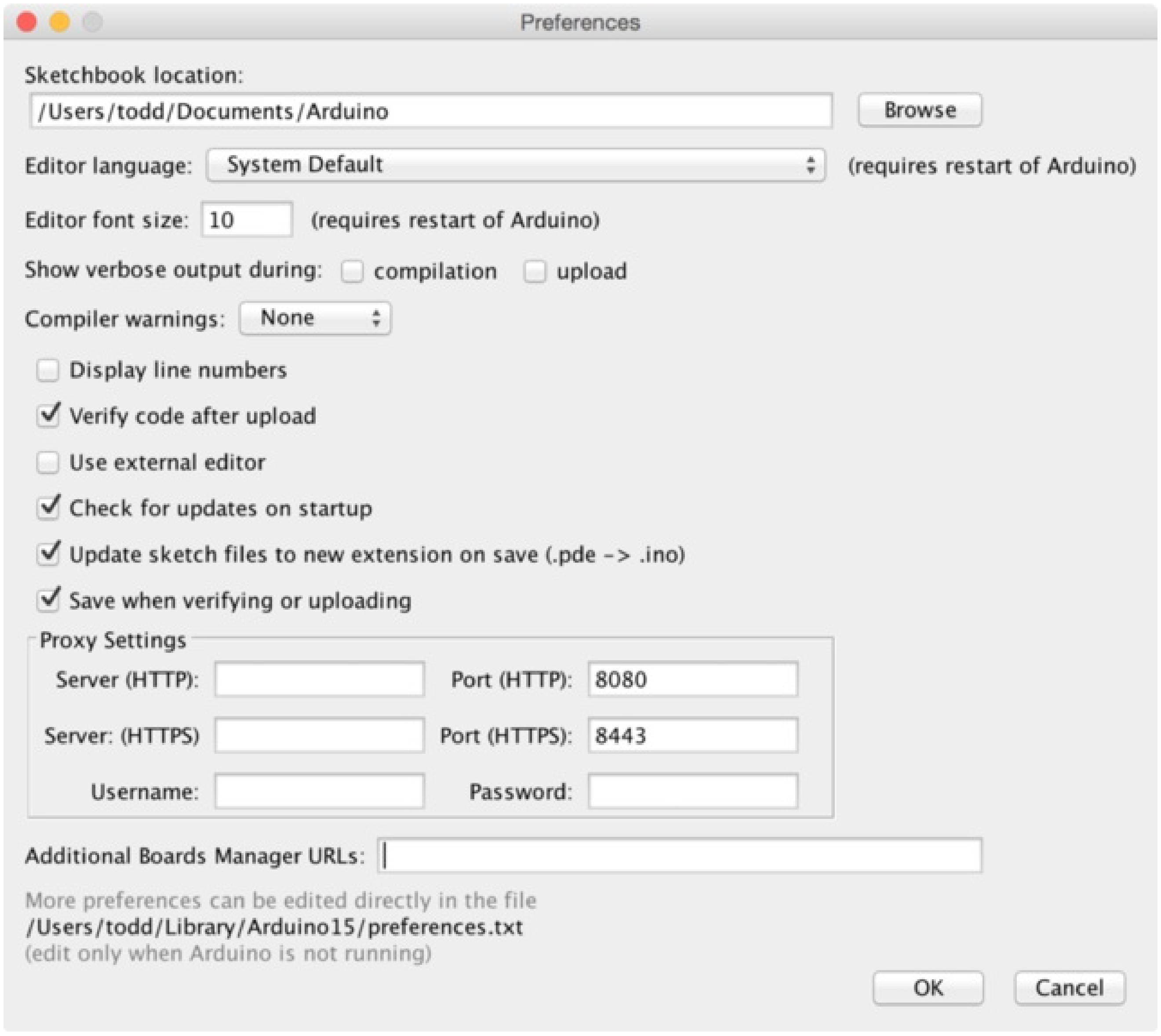
Super Easy Installation (Recommended)

With the Arduino IDE version 1.6.4 and greater you can now add 3rd party boards directly from the stock IDE! 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 1.6.4** or higher:

Latest Arduino IDE Download
http://adafru.it/f1P

After you have downloaded and installed the latest version (remember it**must** be 1.6.4 or higher), 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.


Preferences

Sketchbook location: /Users/todd/Documents/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 10 (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☐ Display line numbers

✓ Verify code after upload

☐ Use external editor

✓ Check for updates on startup

✓ Update sketch files to new extension on save (.pde -> .ino)

✓ Save when verifying or uploading

Proxy Settings

Server (HTTP):		Port (HTTP):	8080
Server: (HTTPS)		Port (HTTPS):	8443
Username:		Password:	

Additional Boards Manager URLs:

More preferences can be edited directly in the file
/Users/todd/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)

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 ofthird party board URLs on the Arduino IDE wiki (http://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.

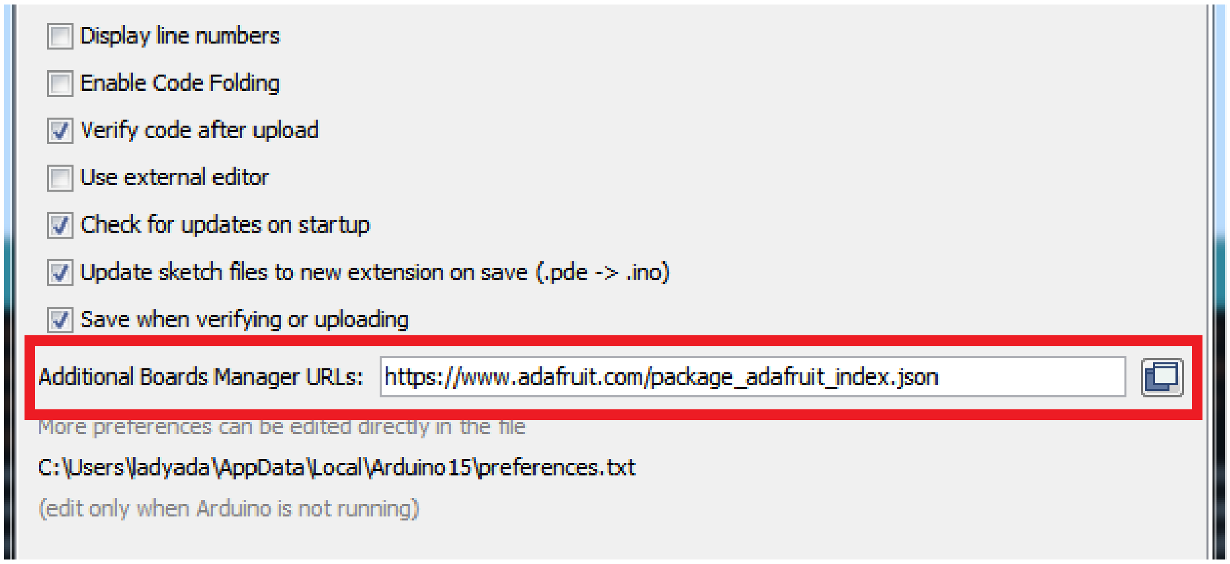
If you don't see the Additional Boards Manager URLs box, make sure you downloaded the Arduino IDE from arduino.cc! Older versions and derivatives of the IDE may not have it

Add the Adafruit Board Support package!

Paste

https://adafruit.github.io/arduino-board-index/package_adafruit_index.json

Into the "Additional Board Managers URLS" box



☐ Display line numbers

☐ Enable Code Folding

☑ Verify code after upload

☐ Use external editor

☑ Check for updates on startup

☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: https://www.adafruit.com/package_adafruit_index.json

More preferences can be edited directly in the file

C:\Users\ladyada\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

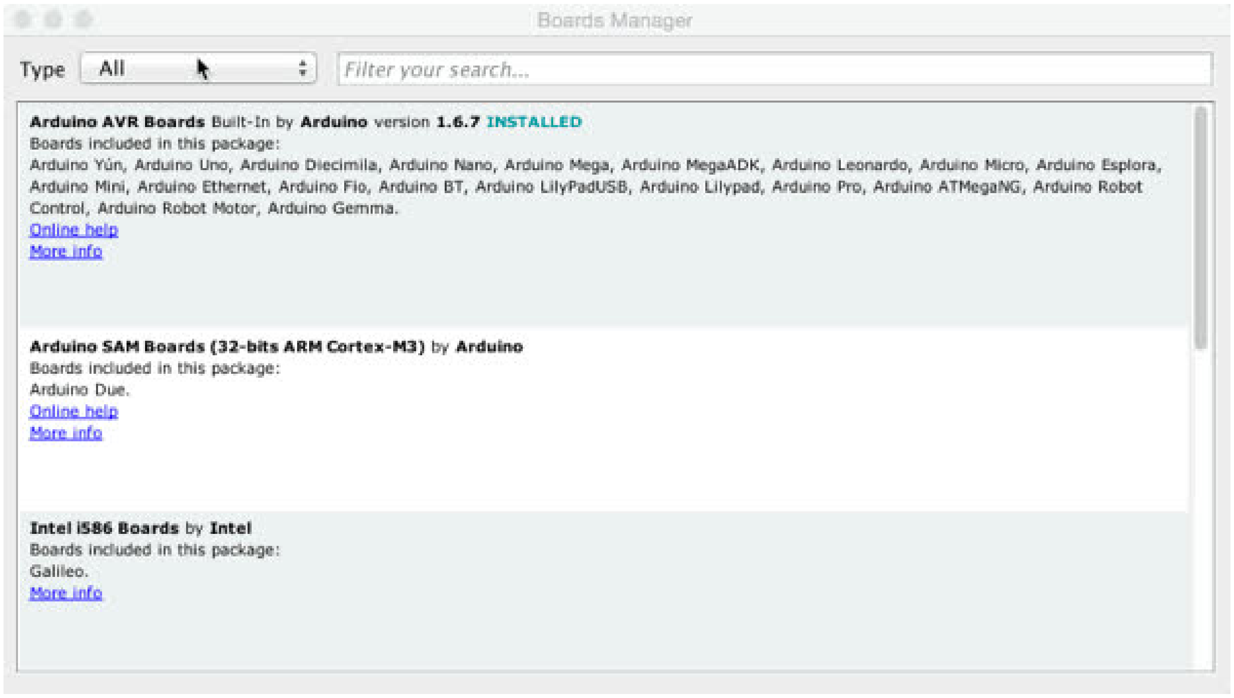
Click **OK** to save the new preference settings. Next we will look at installing boards with the Board Manager.

Manage Board Support

Adding the link to the Adafruit board support package does not actually install anything, it only tells the Arduino IDE where to find the software.

Now that you have added the appropriate URLs to the Arduino IDE preferences, 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 **Contributed**. You will then be able to select and install the boards supplied by the URLs added to the prefrences. In the example below, we are installing support for ***Adafruit AVR Boards***, but the same applies to all boards installed with the Board Manager.



Arduino AVR Boards Built-In by **Arduino** version **1.6.7** INSTALLED
Boards included in this package:
Arduino Yún, Arduino Uno, Arduino Diecimila, Arduino Nano, Arduino Mega, Arduino MegaADK, Arduino Leonardo, Arduino Micro, Arduino Esplora, Arduino Mini, Arduino Ethernet, Arduino Fio, Arduino BT, Arduino LilyPadUSB, Arduino Lilypad, Arduino Pro, Arduino ATMegaNG, Arduino Robot Control, Arduino Robot Motor, Arduino Gemma.
Online help
More info

Arduino SAM Boards (32-bits ARM Cortex-M3) by **Arduino**
Boards included in this package:
Arduino Due.
Online help
More info

Intel i586 Boards by **Intel**
Boards included in this package:
Galileo.
More info

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

Finally follow the steps below for your platform to finish the installation - basically installing drivers and permissions management

- Windows Setup (http://adafru.it/jAa)
- Mac OSX Setup (http://adafru.it/mak)
- Linux Setup (http://adafru.it/iOE)

Also check out the troubleshooting section (http://adafru.it/mal) for some advice on common errors. Have fun using Adafruit's boards!

All-in-one Installation (not recommended!)

If you have trouble using the super easy installation method above another way to install support for Adafruit's boards is with the following preconfigured Arduino IDE downloads. It isn't recommended because you wont have the latest IDE!

Just grab the right file for your platform and use it like the normal Arduino IDE. Adafruit's boards like Trinket, Pro Trinket, Gemma, and Flora are configured to show up in the board list automatically

Arduino 1.6.4 with Adafruit Boards for Windows
http://adafru.it/f83

Arduino 1.6.4 with Adafruit Boards for Mac OSX (10.7+)
http://adafru.it/f84
Arduino 1.6.4 with Adafruit Boards for Linux (32-bit)
http://adafru.it/f85
Arduino 1.6.4 with Adafruit Boards for Linux (64-bit)
http://adafru.it/f86

Once you've downloaded and installed the IDE follow the steps below for your platform to finish the installation:

- Windows Setup (http://adafru.it/jAa)
- Mac OSX Setup (http://adafru.it/mak)
- Linux Setup (http://adafru.it/iOE)

Also check out the troubleshooting section (http://adafru.it/mal) for some advice on common errors. Have fun using Adafruit's boards!

Manual Installation (super advanced!)

If you have your own version of the Arduino IDE or would like to install the Adafruit boards yourself then follow the steps below. However it is **highly recommended** that most users stick to the initial methods

With the 1.6.x version of the Arduino IDE the process of adding support for new boards is much simpler than previous versions. First start by downloading a version of the Arduino 1.6.x IDE and installing it as normal. **Note that these instructions are only tested to work against the Arduino 1.6.4 IDE, earlier or later versions may or may not work!**

Once you've installed the stock Arduino IDE download and unzip the following file which contains Adafruit's board definitions (http://adafru.it/eTY):

Adafruit Board Definitions
http://adafru.it/eTZ

Unzip the file and navigate inside the **Adafruit_Arduino_Boards** folder to find a **hardware** subfolder with a small hiearchy of files, and a **drivers** folder with Flora drivers specific to Windows.

Next find your installed Arduino IDE's **hardware** subfolder. For Windows and Linux the hardware subfolder should be directly beneath the folder where Arduino's IDE was installed.

For Mac OSX the folder is hidden inside the application bundle. Right click on the Arduino application and click '**Show Package Contents**', then navigate to the**Content -> Java -> Hardware** folder to find the hardware subfolder.

Once you've found Arduino's hardware subfolder carefully merge in the contents of the hardware folder from Adafruit's board definitions, being sure to overwrite any file that conflicts (only avrdude.conf should conflict). When you're done the Arduino hardware folder should have the following hiearchy:

- **hardware**
    - **adafruit**
        - **avr**
            - The contents of the hardware/adafruit/avr folder from Adafruit's board definitions. There will be a boards.txt, platform.txt and variants subfolder.
    - **arduino**
        - No changes to this hiearchy, it is exactly as provided by the Arduino IDE.
    - **tools**
        - **avr**
            - **etc**
                - **avrdude.conf** - This version of avrdude.conf should be from Adafruit's board definitions and completely overwrites the version provided by the Arduino IDE.
            - ... all other files as provided by the Arduino IDE.
        - ... all other files as provided by the Arduino IDE.

Finally on Windows only, copy the contents of the drivers folder from the board definitions download into the Arduino IDE's drivers folder (it's a sibling of the hardware folder you found earlier).

That's all you need to do to manually install Adafruit's boards with the 1.6.x version of Arduino! Now follow the steps below for your platform to finish the installation:

- Windows Setup (http://adafru.it/jAa)
- Mac OSX Setup (http://adafru.it/mak)
- Linux Setup (http://adafru.it/iOE)

Also check out the troubleshooting section (http://adafru.it/mal) for some advice on common errors. Have fun using Adafruit's boards!

HELP!

I just plugged it in and I can't seem to connect to my Circuit Playground with Arduino!

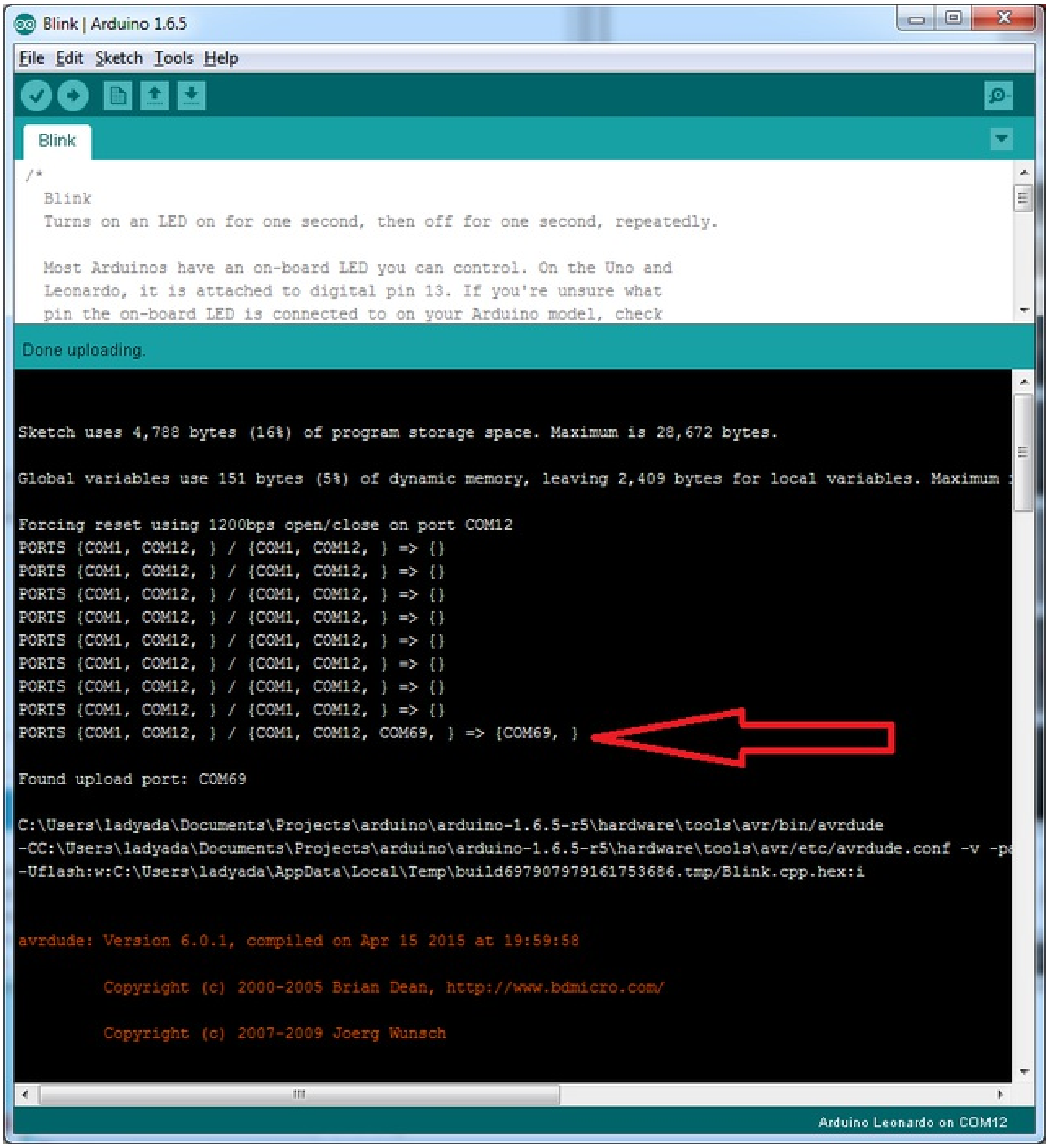
99% of initial problems with circuit playground are due to having**charge USB cables** instead of sync cables. Do not use a cable you've only used for charging a phone. Make sure its a cable that can pass data as well as power. Lately, there's been a lot of products shipped with charging only cables and it's very confusing because the Circuit Playground lights up but does not show up in the Arduino IDE!

So, please, try multiple USB cables, and if you find a charge-only cable, cut it in half and throw it away so you will not make the mistake again!

Ack! I "did something" and now when I plug in the Circuit Playground it doesn't show up as a device anymore so I cant 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 Circuit Playground

1. Turn on **verbose upload** in the Arduino IDE preferences
2. Plug in Circuit Playground, 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.**Circuit Playground (make sure you select the correct board)**
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 Cplay will show up as the Bootloader COM/Serial port
9. The IDE should see the bootloader COM/Serial port and upload properly



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the Uno and
  Leonardo, it is attached to digital pin 13. If you're unsure what
  pin the on-board LED is connected to on your Arduino model, check
```

```
Sketch uses 4,788 bytes (16%) of program storage space. Maximum is 28,672 bytes.

Global variables use 151 bytes (5%) of dynamic memory, leaving 2,409 bytes for local variables. Maximum

Forcing reset using 1200bps open/close on port COM12
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, } => {}
PORTS {COM1, COM12, } / {COM1, COM12, COM69, } => {COM69, }

Found upload port: COM69

C:\Users\ladyada\Documents\Projects\arduino\arduino-1.6.5-r5\hardware\tools\avr/bin/avrdude
-CC:\Users\ladyada\Documents\Projects\arduino\arduino-1.6.5-r5\hardware\tools\avr/etc/avrdude.conf -v -p
-Uflash:w:C:\Users\ladyada\AppData\Local\Temp\build6979079791617536 86.tmp/Blink.cpp.hex:i


avrdude: Version 6.0.1, compiled on Apr 15 2015 at 19:59:58

         Copyright (c) 2000-2005 Brian Dean, http://www.bdmicro.com/

         Copyright (c) 2007-2009 Joerg Wunsch
```

I can't get the Circuit Playground 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. Make sure you select **Circuit Playground!** Do not use anything else, do not use the 32u4 breakout board line.

Use the 'repair' technique above to fix it.

Libraries

We wrapped up everything you need to run Arduino code on your Circuit Playground is wrapped up into a tidy library that integrates all the sensing and lighting.

Installing Via Library Manager

The Circuit Playground library is available on the Adafruit GitHub website (http://adafru.it/naF). The library is installed in versions of the Arduino IDE greater than 1.6.7 as follows:

- In the menu click "Sketch", then "Include Library"
- At the top, click "Manage Libraries. . ."
- Type "Circuit Playground" in the search box. You should see Adafruit Circuit Playground listed.

Download Adafruit_CircuitPlayground 'Manually'

To begin working with Circuit Playground, you will need to download Adafruit_CircuitPlayground from our github repository (http://adafru.it/naF). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

Download Adafruit CircuitPlayground Library
http://adafru.it/naB

Rename the uncompressed folder **Adafruit_CircuitPlayground** and check that the **Adafruit_CircuitPlayground** folder contains **Adafruit_CircuitPlayground.cpp** and **Adafruit_CircuitPlayground.h**

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

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

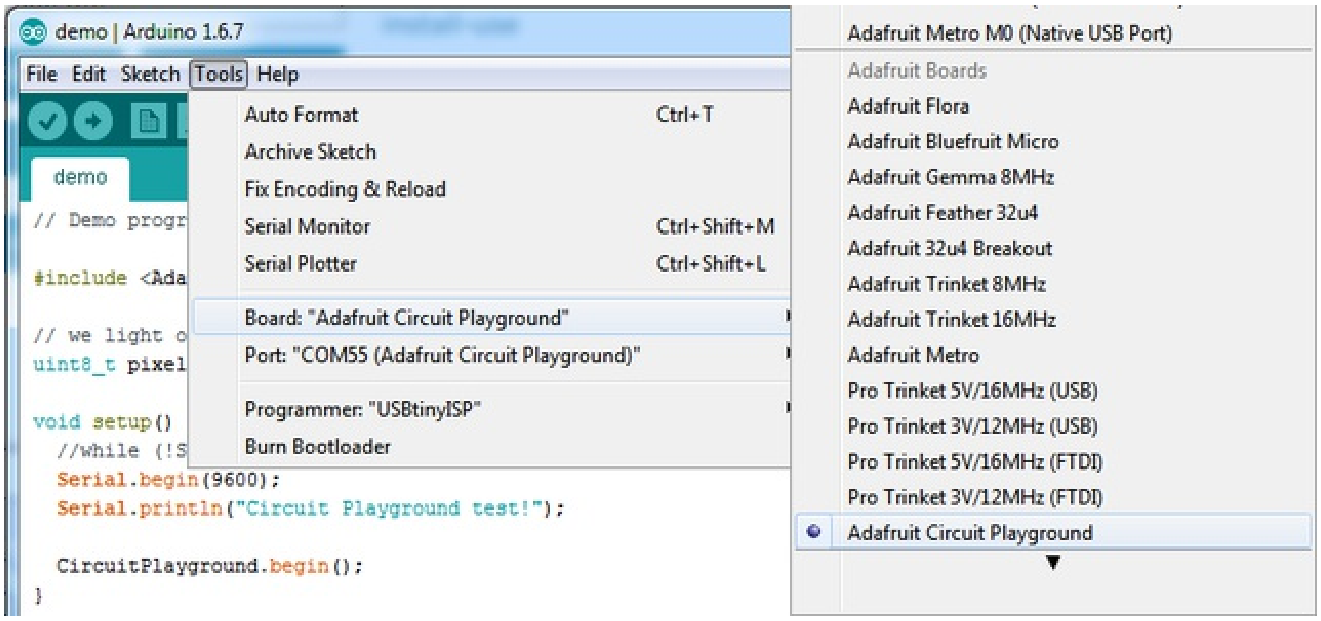
Run the Demo

Plug in your Circuit Playground. Under Windows you'll need to make sure you have installed the driver package (http://adafru.it/naC). Do that if you haven't!

Next, you'll also need to make sure you have installed the Adafruit Board Support package. That's required no matter what computer you're using.

Select the Circuit Playground Board

Under the **Tools -> Board** submenu, scroll down to **Adafruit Boards** and pick **Adafruit Circuit Playground**

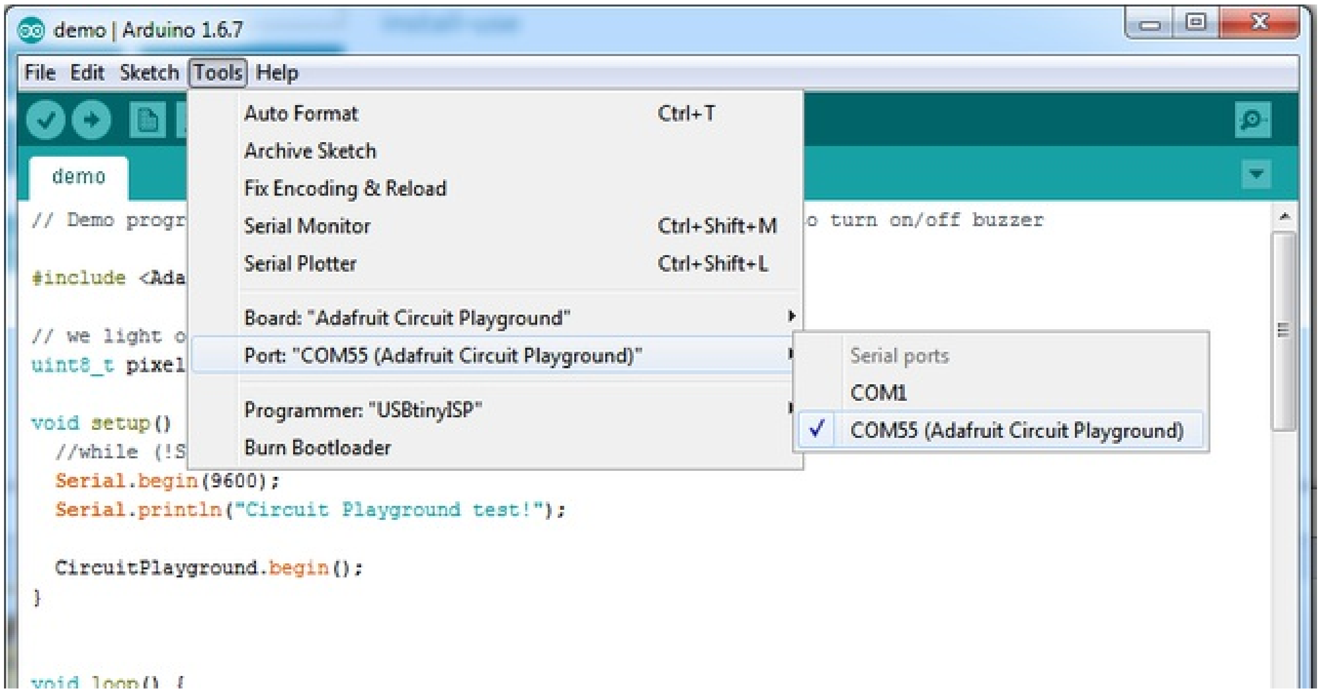


```
// Demo progr
#include <Ada

// we light o
uint8_t pixel

void setup()
  //while (!S
  Serial.begin(9600);
  Serial.println("Circuit Playground test!");

  CircuitPlayground.begin();
}
```

demo | Arduino 1.6.7
File Edit Sketch Tools Help
demo
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
Board: "Adafruit Circuit Playground"
Port: "COM55 (Adafruit Circuit Playground)"
Programmer: "USBtinyISP"
Burn Bootloader
Adafruit Metro M0 (Native USB Port)
Adafruit Boards
Adafruit Flora
Adafruit Bluefruit Micro
Adafruit Gemma 8MHz
Adafruit Feather 32u4
Adafruit 32u4 Breakout
Adafruit Trinket 8MHz
Adafruit Trinket 16MHz
Adafruit Metro
Pro Trinket 5V/16MHz (USB)
Pro Trinket 3V/12MHz (USB)
Pro Trinket 5V/16MHz (FTDI)
Pro Trinket 3V/12MHz (FTDI)
● Adafruit Circuit Playground

Select the matching Port

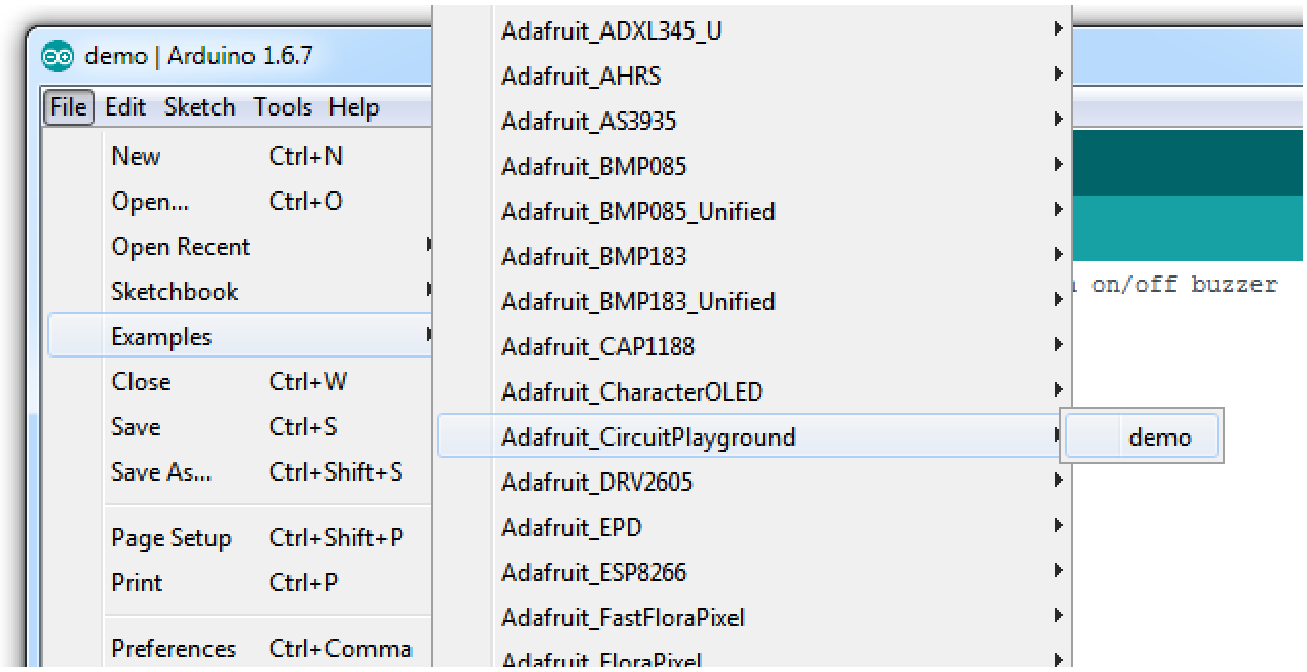
Under **Tools->Port** select the port that is labed (Circuit Plaground)


demo | Arduino 1.6.7
File Edit Sketch Tools Help
demo
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
Board: "Adafruit Circuit Playground"
Port: "COM55 (Adafruit Circuit Playground)"
Programmer: "USBtinyISP"
Burn Bootloader
Serial ports
COM1
✓ COM55 (Adafruit Circuit Playground)

```
// Demo progr                                          o turn on/off buzzer

#include <Ada

// we light o
uint8_t pixel

void setup()
  //while (!S
  Serial.begin(9600);
  Serial.println("Circuit Playground test!");

  CircuitPlayground.begin();
}


void loop() {
```

Load the Demo Program

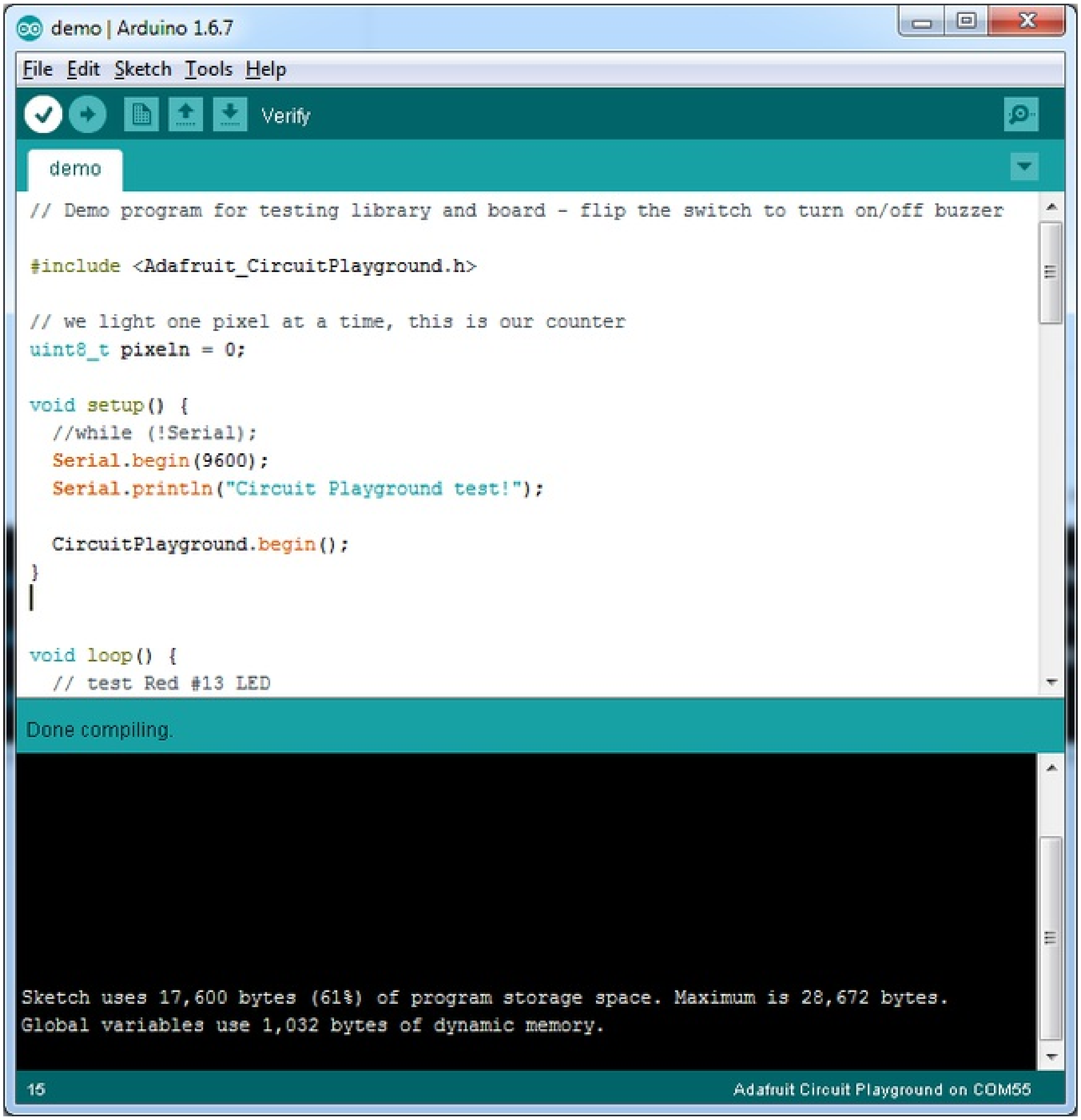
OK you're now ready to load the demo. Under**File->Examples** locate **Adafruit CircuitPlayground** and then select the **demo** program.



Compile/Verify the Demo

Click the **Verify** button (also the **Sketch->Verify** menu item) to compile/verify the demo.

Make sure you get "**Done compiling.**" and no errors



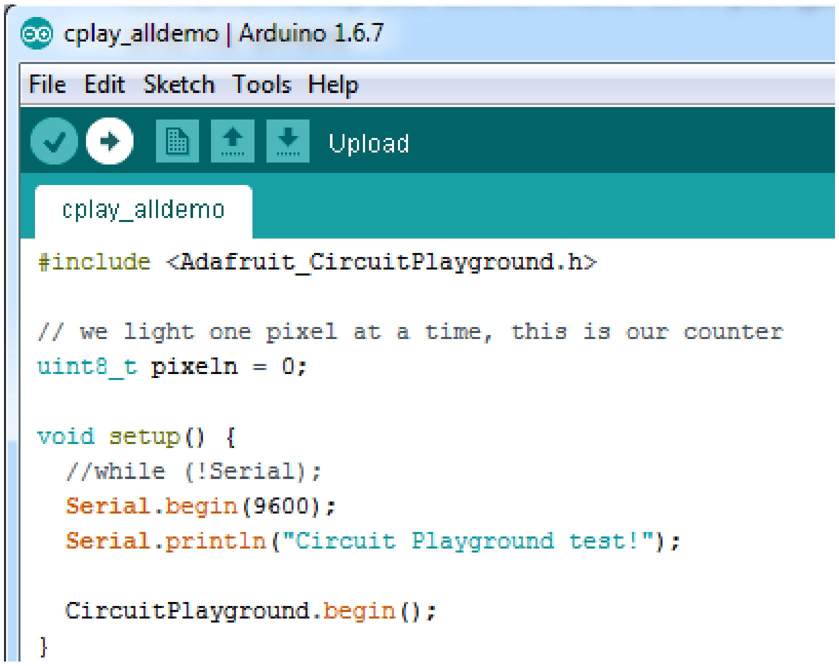
```
// Demo program for testing library and board - flip the switch to turn on/off buzzer

#include <Adafruit_CircuitPlayground.h>

// we light one pixel at a time, this is our counter
uint8_t pixeln = 0;

void setup() {
  //while (!Serial);
  Serial.begin(9600);
  Serial.println("Circuit Playground test!");

  CircuitPlayground.begin();
}


void loop() {
  // test Red #13 LED
```

Done compiling.

```
Sketch uses 17,600 bytes (61%) of program storage space. Maximum is 28,672 bytes.
Global variables use 1,032 bytes of dynamic memory.
```

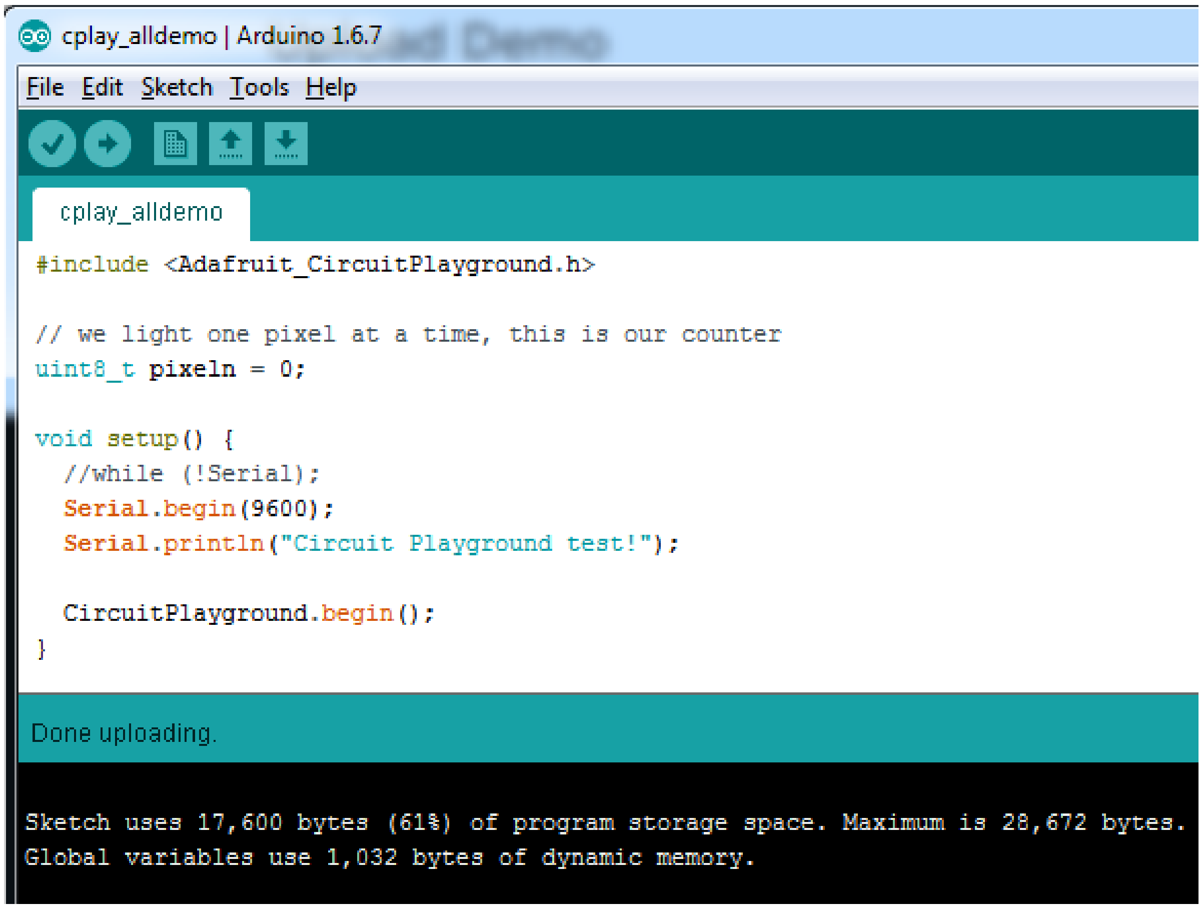
Upload Demo

Click the Upload button to upload the code



```
#include <Adafruit_CircuitPlayground.h>

// we light one pixel at a time, this is our counter
uint8_t pixeln = 0;

void setup() {
  //while (!Serial);
  Serial.begin(9600);
  Serial.println("Circuit Playground test!");

  CircuitPlayground.begin();
}
```

You should get a **Done uploading.** message in the blue statusbar

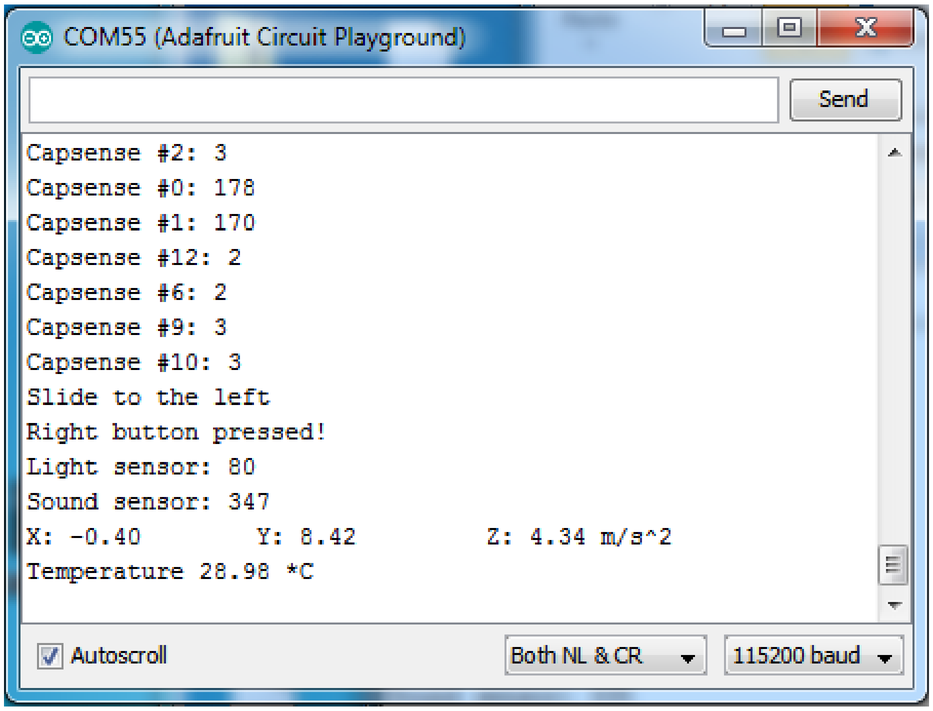


```
#include <Adafruit_CircuitPlayground.h>

// we light one pixel at a time, this is our counter
uint8_t pixeln = 0;

void setup() {
  //while (!Serial);
  Serial.begin(9600);
  Serial.println("Circuit Playground test!");

  CircuitPlayground.begin();
}
```

Done uploading.

```
Sketch uses 17,600 bytes (61%) of program storage space. Maximum is 28,672 bytes.
Global variables use 1,032 bytes of dynamic memory.
```

You can now run the serial console to get data output:



```
Capsense #2: 3
Capsense #0: 178
Capsense #1: 170
Capsense #12: 2
Capsense #6: 2
Capsense #9: 3
Capsense #10: 3
Slide to the left
Right button pressed!
Light sensor: 80
Sound sensor: 347
X: -0.40        Y: 8.42         Z: 4.34 m/s^2
Temperature 28.98 *C
```

Both NL & CR   115200 baud

You'll get information such as:

- "Capacitive touch" readings for all 8 outer pads (under 50 means not touched, over 100 usually means the pads are touched)
- Slide switch location (left or right)
- If the Right and Left buttons are pressed
- Light sensor readings, higher values mean more light
- Sound sensor readings
- X, Y and Z accelerometer readings
- Temperature in Celcius

Downloads

Windows Driver Software

- Available here (http://adafru.it/naC)

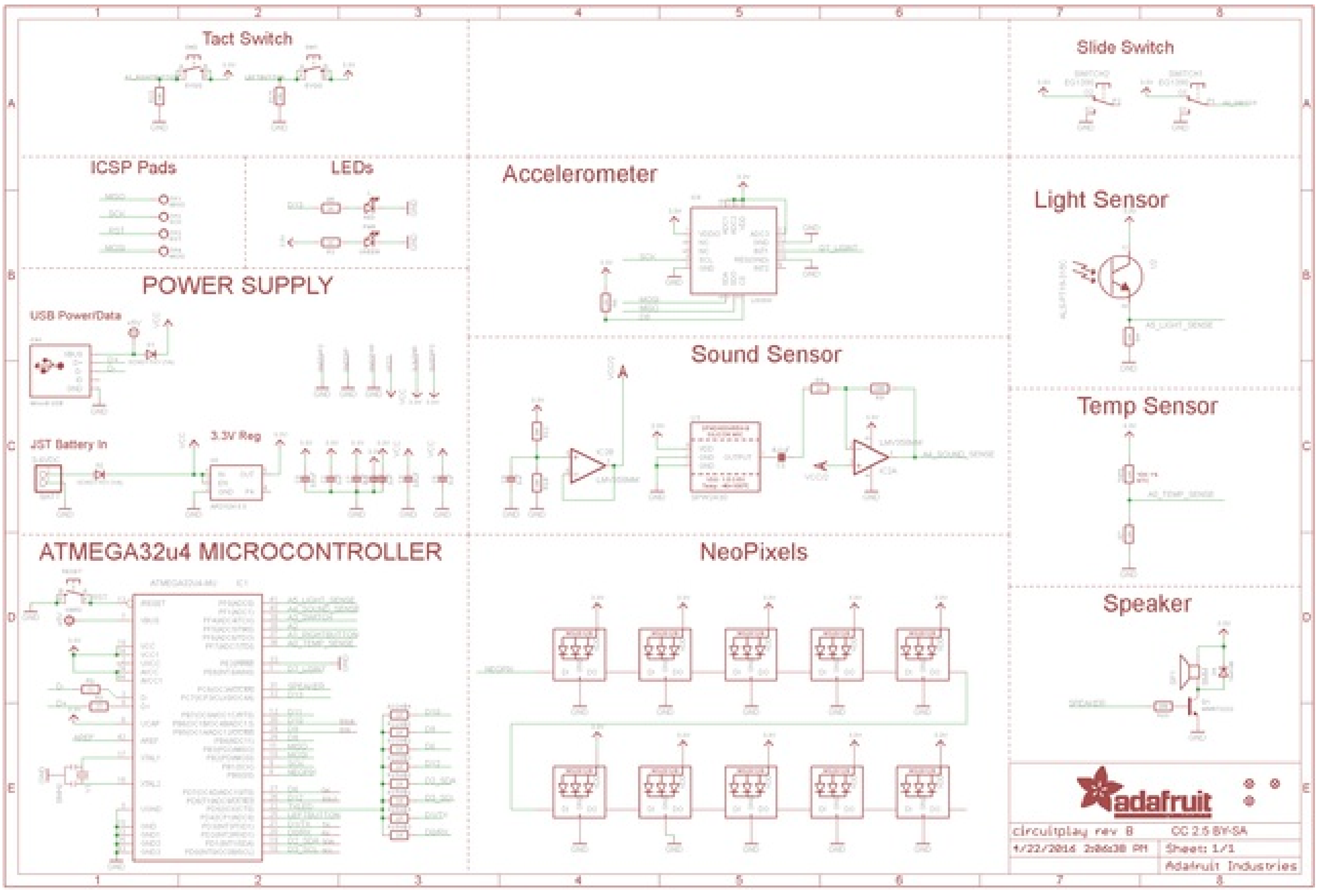
Source

- Arduino Circuit Playground interfacing library (http://adafru.it/naF)
- Adafruit Board Support Pkg (Should be installed via the Board Manager!) (http://adafru.it/eTY)
- PCB Files in EagleCAD format (http://adafru.it/nb5)
- Fritzing object available in the Adafruit Fritzing Library(http://adafru.it/aP3)

Datasheets

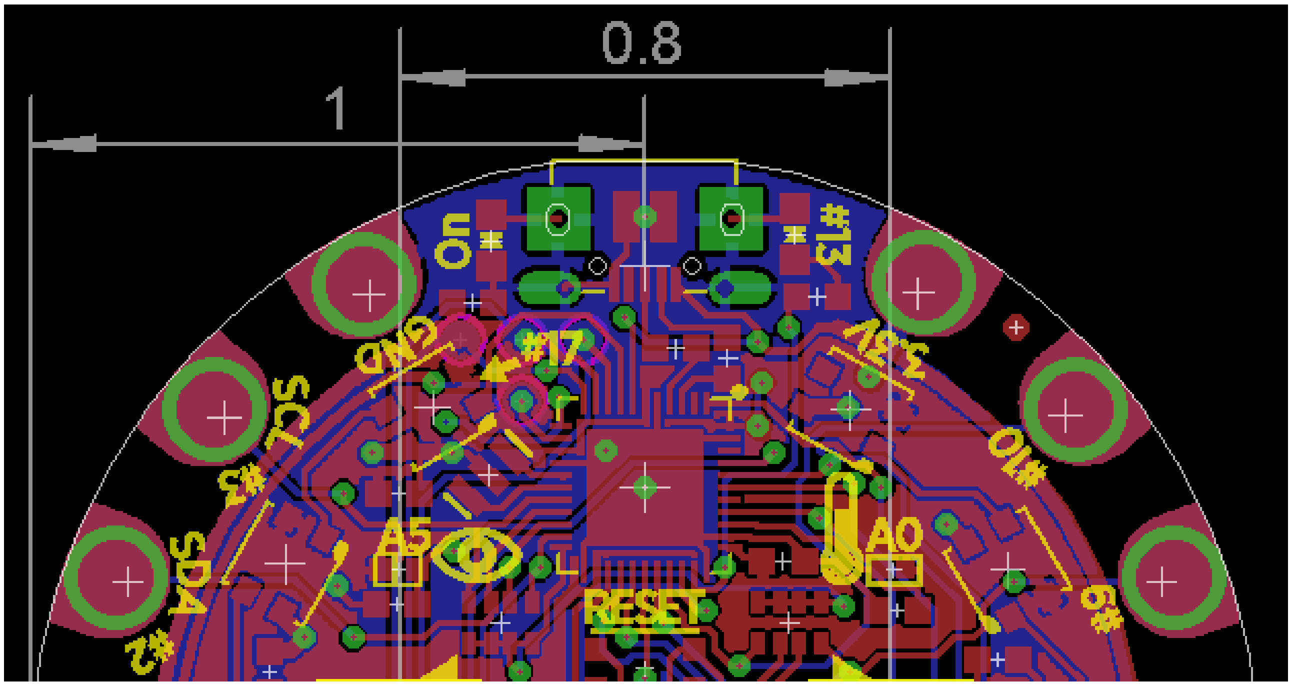
- Microcontroller datasheet (http://adafru.it/rnc)
- Buzzer datasheet (http://adafru.it/p3e)
- MEMS microphone datasheet (http://adafru.it/p3d)
- Thermistor datasheet (http://adafru.it/p3a)

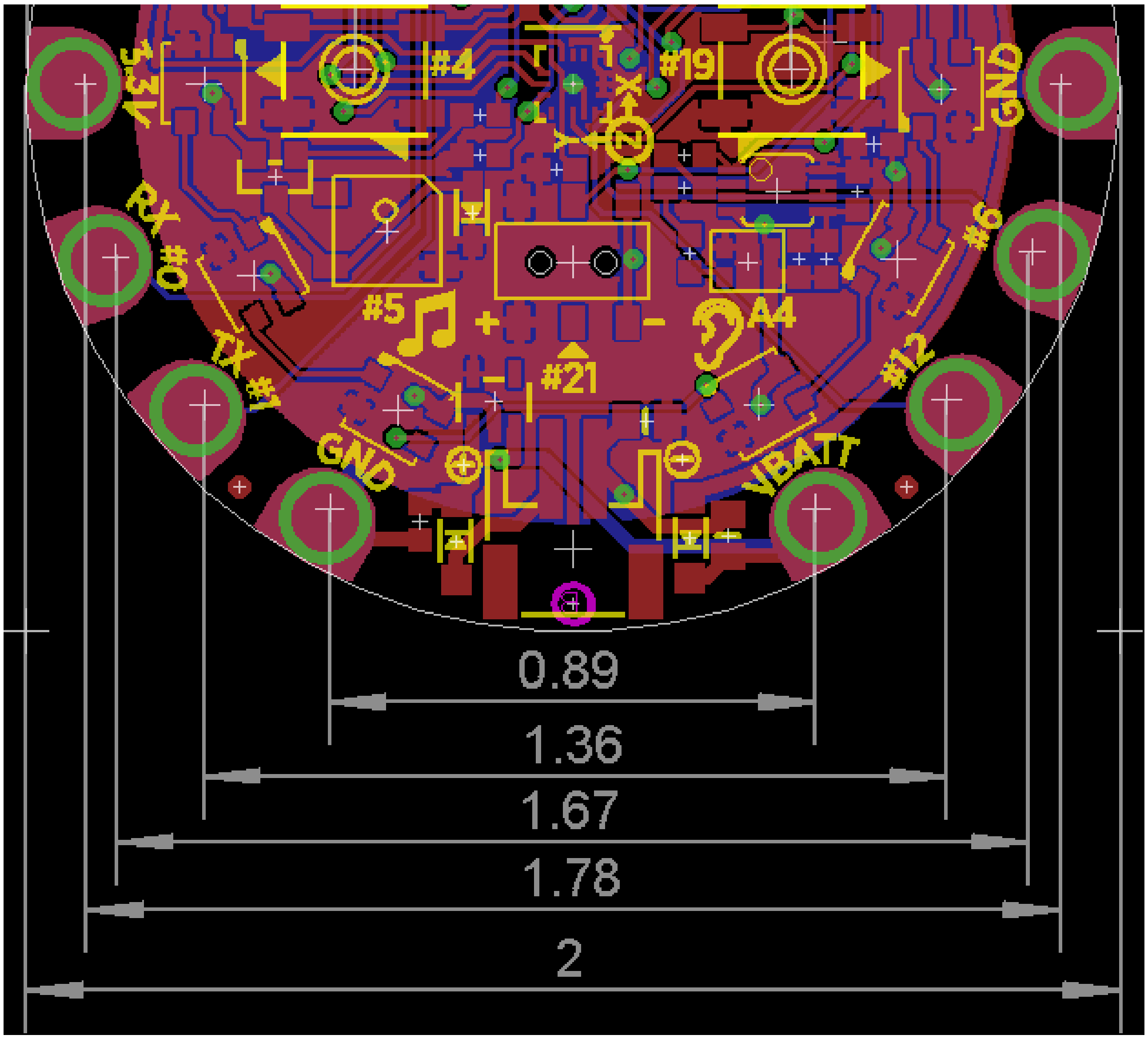
Schematic



Fabrication Print

Dims in inches




3.3V
#4
#19
GND
RX #0
TX #1
#5
#21
A4
#6
#12
GND
VBATT
X
Y
Z
0.89
1.36
1.67
1.78
2