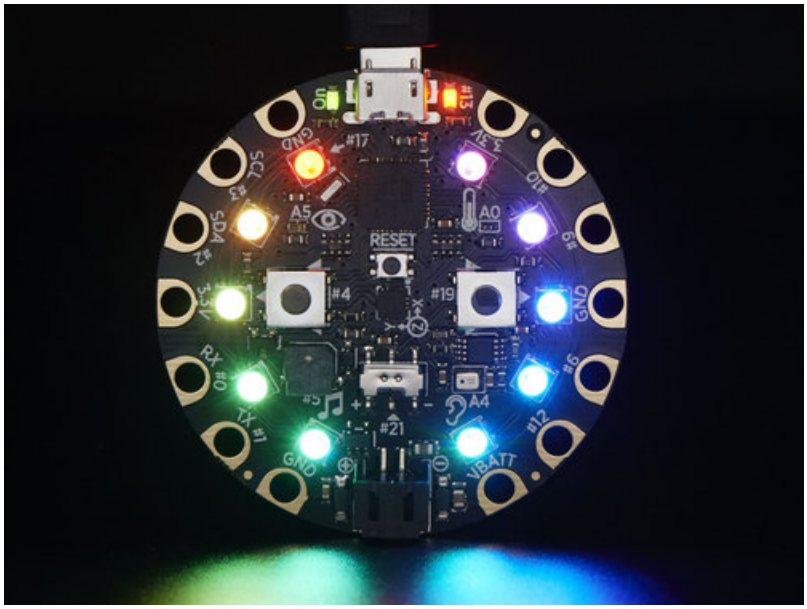
adafruit learning system

# Introducing Circuit Playground

Created by lady ada

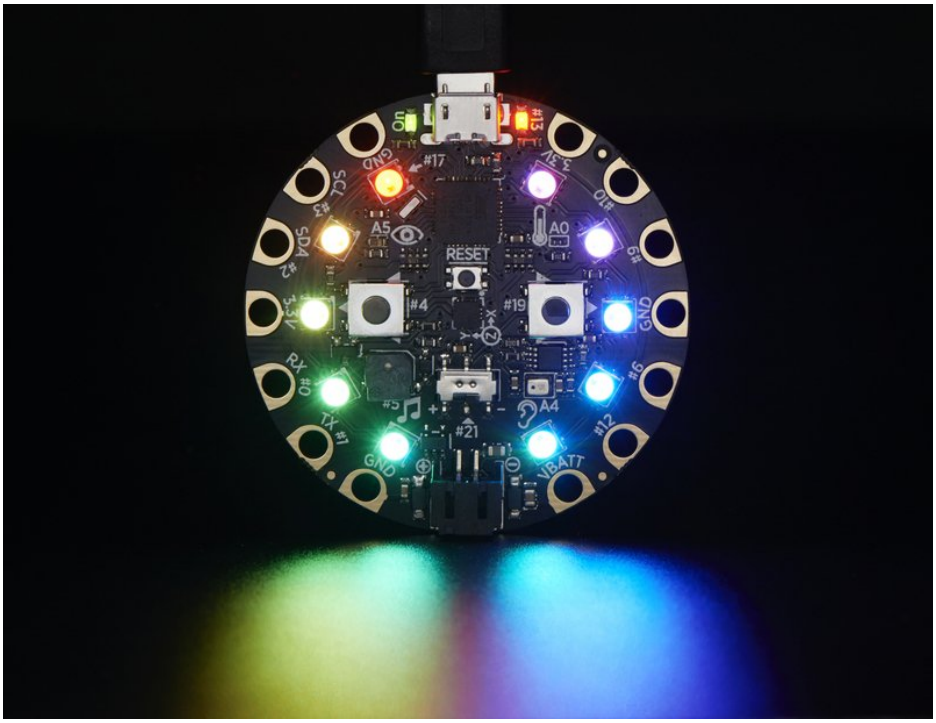



Last updated on 2020-01-09 07:51:30 PM UTC

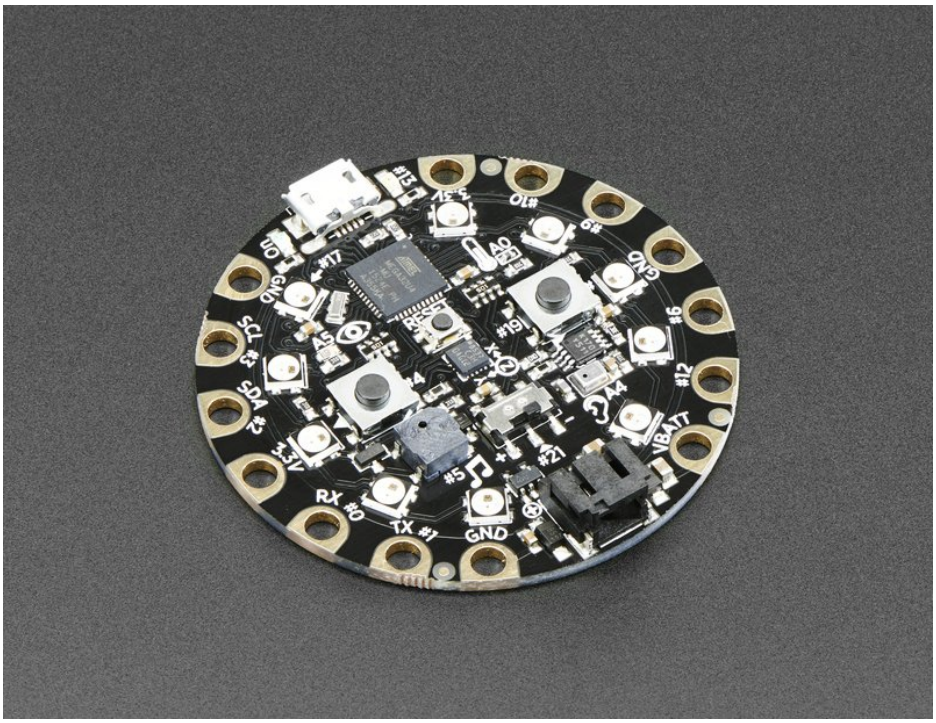
# Overview

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

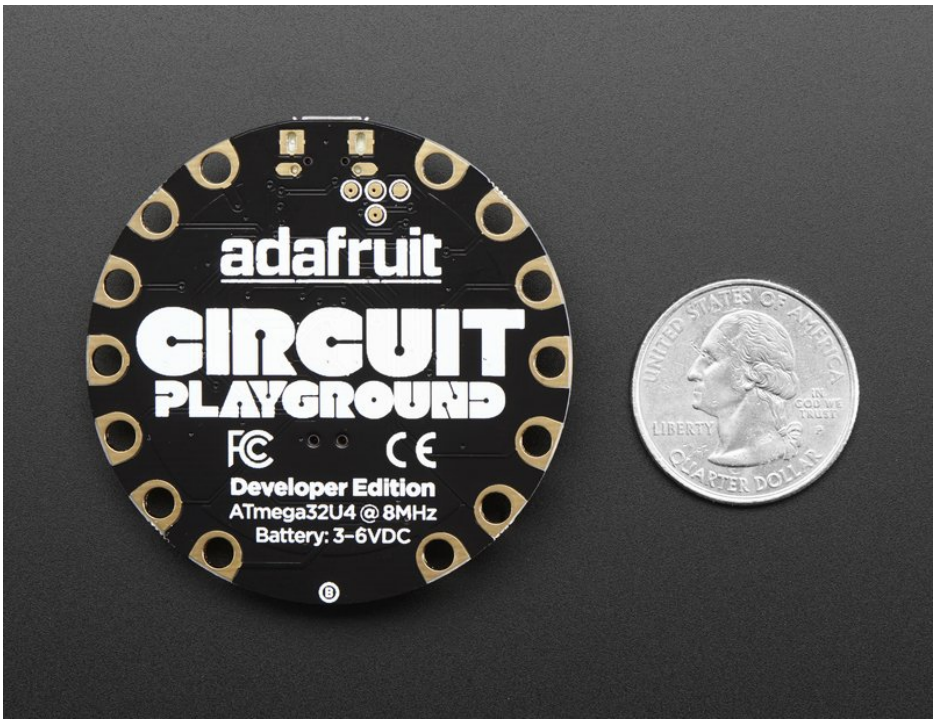



Circuit Playground has built-in USB support. Built in USB means you plug it in to program it, it just shows up - all you need is a Micro-B USB cable, no additional purchases are needed! With the new 1.6.4+ Arduino IDE, it takes only a few seconds to add support (https://adafru.it/nb4). The Circuit Playground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


adafruit
CIRCUIT
PLAYGROUND
Developer Edition
ATmega32U4 @ 8MHz
Battery: 3-6VDC
LIBERTY
IN GOD WE TRUST
QUARTER DOLLAR

# Classic vs. Express

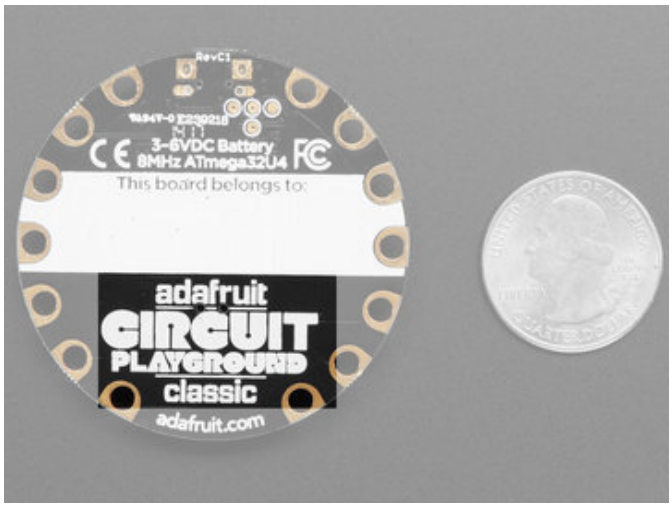
Circuit Playground started its life as a board with simple requirements - just work with Arduino IDE and Code.org. But since it's initial launch in 2015 we've learned a lot and improved the board greatly!

There are **TWO** Circuit Playgrounds - one **Classic** and one **Express**.

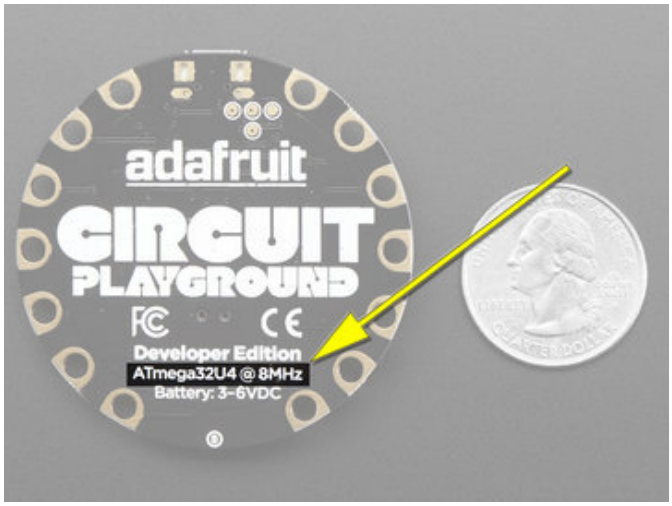
The Classic version can run Arduino and Code.org

The Express version can run MakeCode, CircuitPython, Arduino, and Code.org CS Discoveries.

## How to tell if you have a Classic

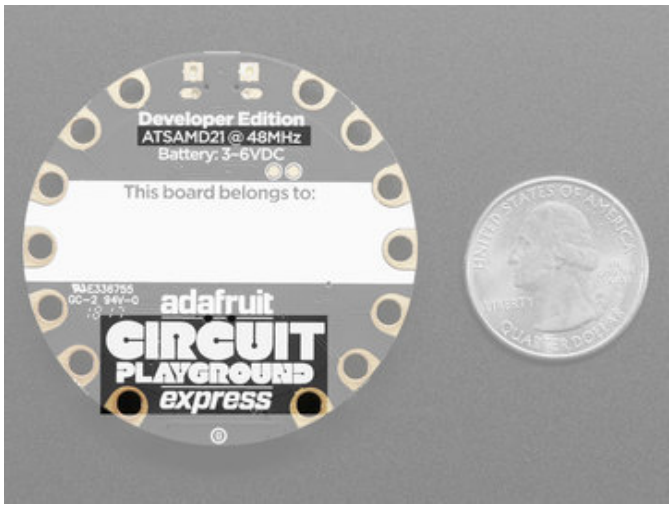



Current Circuit Playground Classic boards have **Classic** written on the back lower half



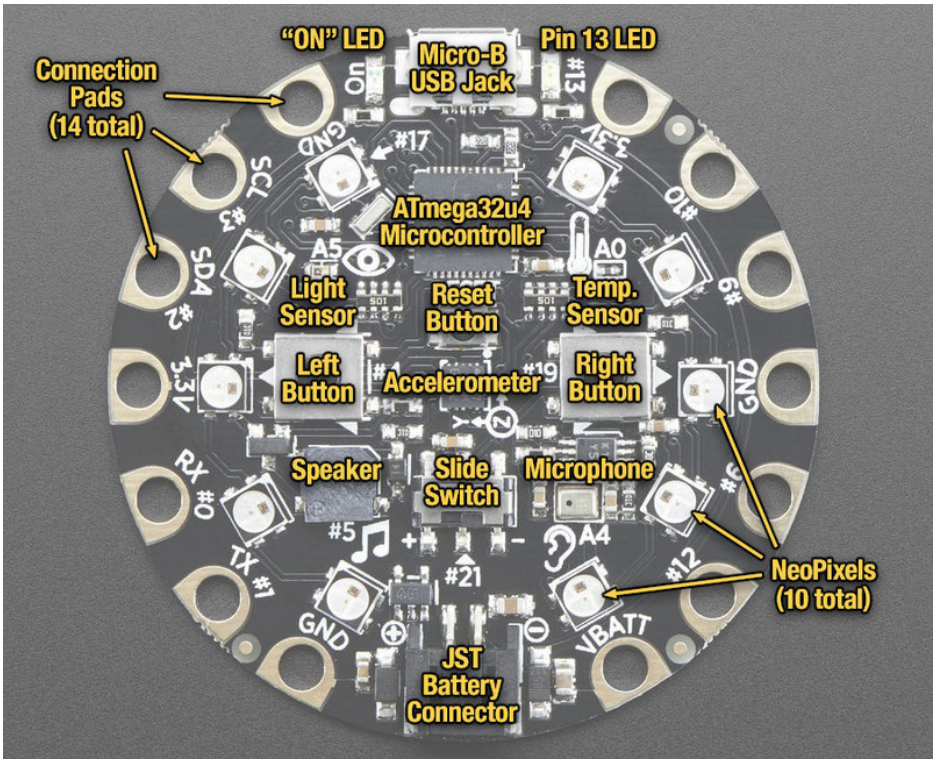

Older Circuit Playground Classics didn't have the word **Classic** on the back, but they do have text that says the chip type, **ATmega32U4**

## How to tell if you have an Express



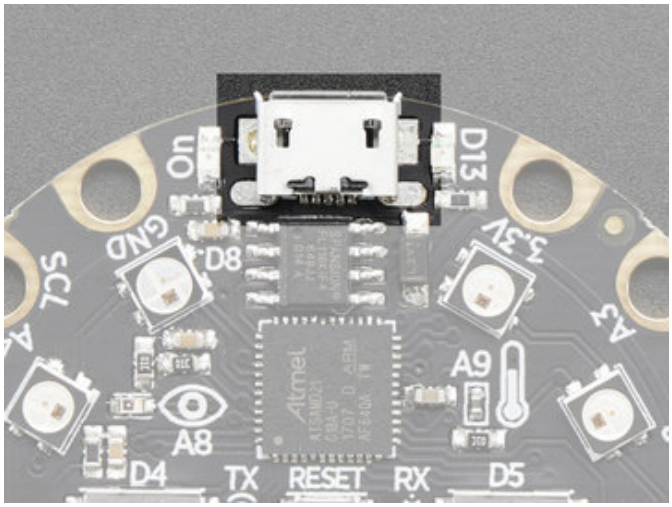

Express boards have **CIRCUIT PLAYGROUND EXPRESS** on the back lower half. They'll also note the chip is an **ATSAMD21**

## Guided Tour



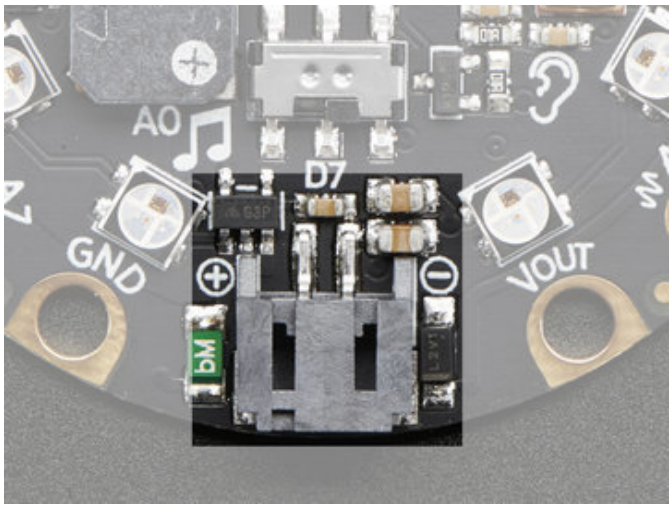

Let me take you on a tour of your Circuit Playground Classic (we'll shorten that to **CPC**). Each CPC comes chock-full of good design to make it a joy to use.

## Power and Data



### Micro B USB connector

This is at the top of the board. We went with the tried and true micro-B USB connector for power and/or USB communication (bootloader, serial, HID, etc). Use with any computer with a standard data/sync cable.

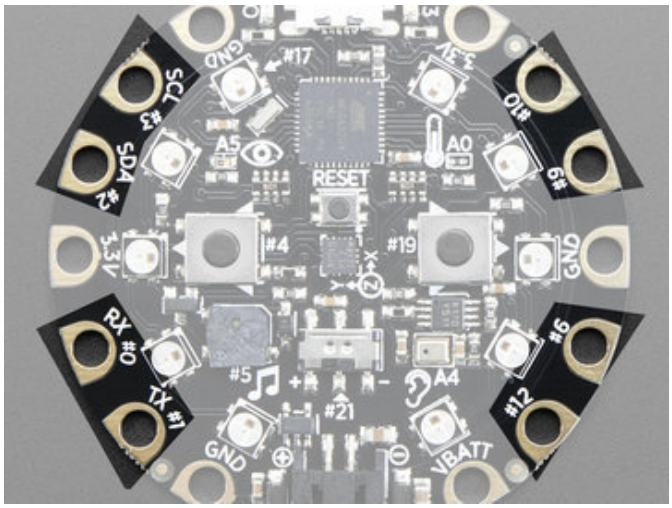



### JST Battery Input

This is at the bottom of the board. You can take your CPC anywhere and power it from an external battery. This pin can take up 6V DC input, and has reverse-polarity, over-current and thermal protections. The circuitry inside will use either the battery input power or USB power, safely switching from one to the other. If both are connected, it will use whichever has the higher voltage. Works great with a Lithium Polymer battery or our 3xAAA battery packs with a JST connector on the end. There is no built in battery charging (so that you can use Alkaline *or* Lithium batteries safely)

## Alligator/Croc Clip Pads

To make it super-easy to connect to the microcontroller, we have 14 connection pads. You can solder to them, use alligator/croc clips, sew with conductive thread, even use small metal screws!




Of the 14 pads, you get a wide range of power pins, I2C, UART, Analog In, Digital In/Out and PWMt.

Some of them can even sense the touch of your finger!

See the next pinouts page for more details!

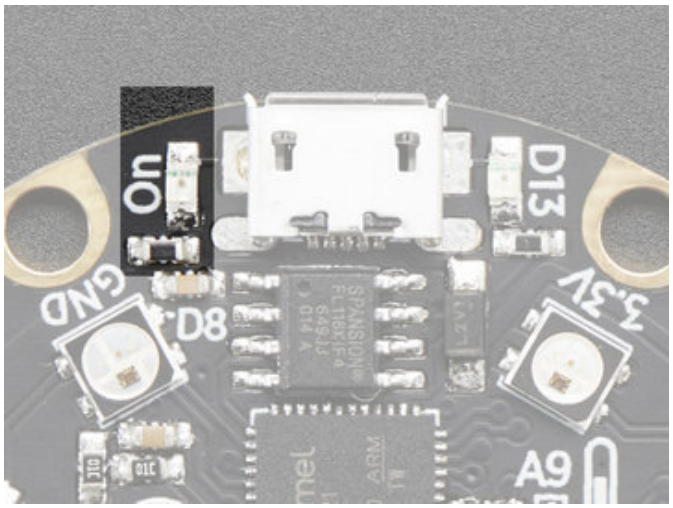
All 8 non-power pads around the circuit playground have the ability to act as capacitive touch pads. Each pad has a 1Mohm resistor between it and digital pin #30. You can toggle this pin to control whether the resistor is a pullup or pulldown or floating. Note that this means that all the pads have a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

## Microchip

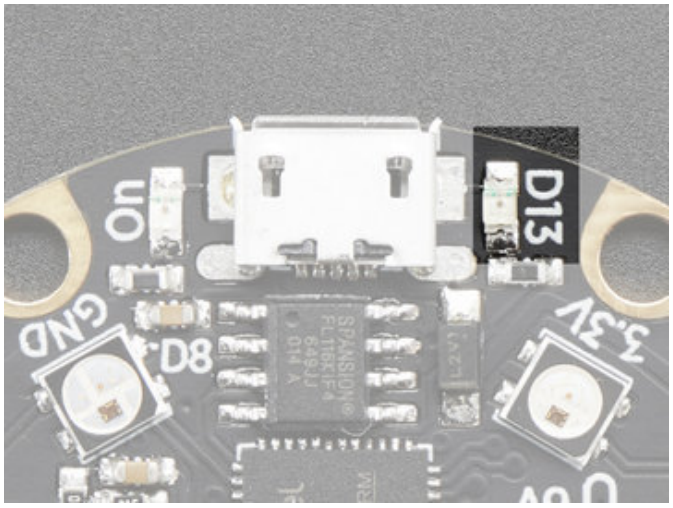
The **brains** of the operation here is the **ATMEGA32u4** an 8-bit AVR microcontroller.  It sits in the top center, and is what you use to run Arduino!

## LEDs



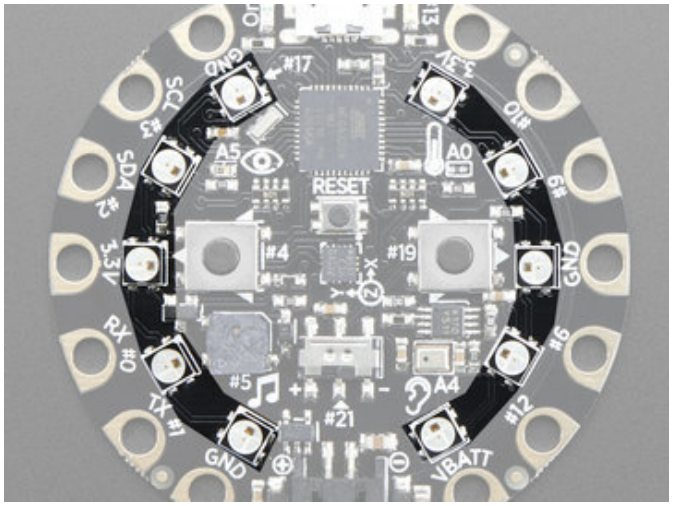

## Green ON LED

To the left of the USB connector. This LED lets you know that the CPC is powered on. If it's lit, power is good! If it's dim, flickering or off, there's a power problem and you will have problems. You can't disable this light, but you *can* cover it with electrical tape if you want to make it black.




## Red #13 LED

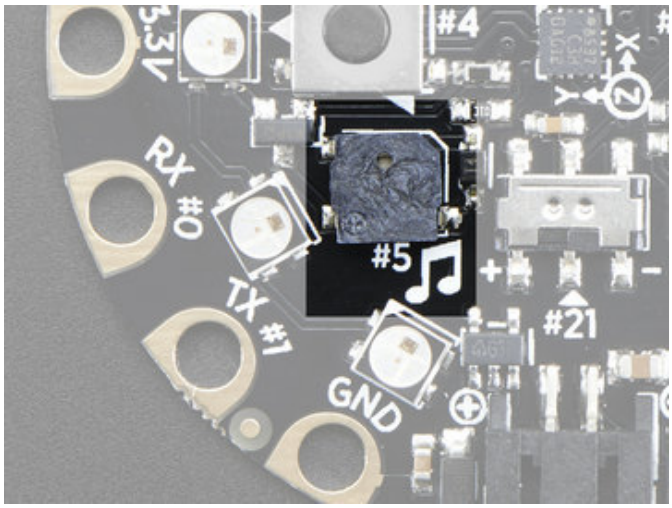
To the right of the USB connector. This LED does double duty. Its connected with a series resistor to the digital #13 GPIO pin. It pulses nicely when the CPC is in bootloader mode, and its also handy for when you want an indicator LED. Many first projects blink this LED to prove that programming worked.




## 10 x Color NeoPixel LED

The ten LEDs surrounding the outer edge of the boards are all full color, RGB LEDs, each one can be set to any color in the rainbow. Great for beautiful lighting effects! The NeoPixels will also help you know when the bootloader is running (they will turn green) or if it failed to initialize USB when connected to a computer (they will turn red).

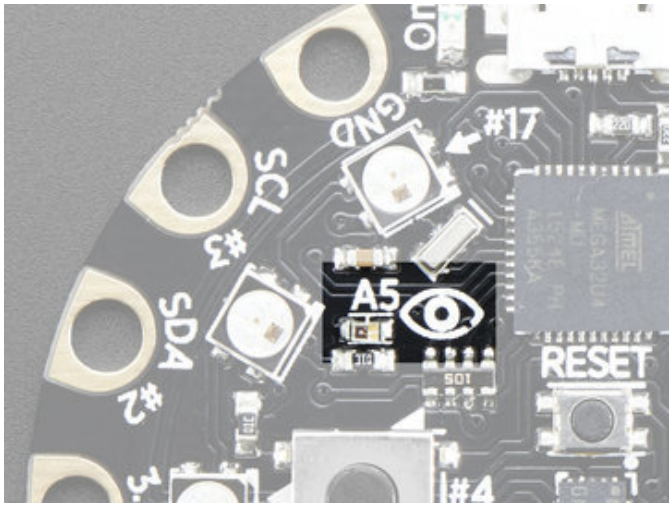
# Speaker




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.

## Sensors

The Circuit Playground Classic has a large number of sensor **inputs** that let you add all sorts of interactivity to your project.

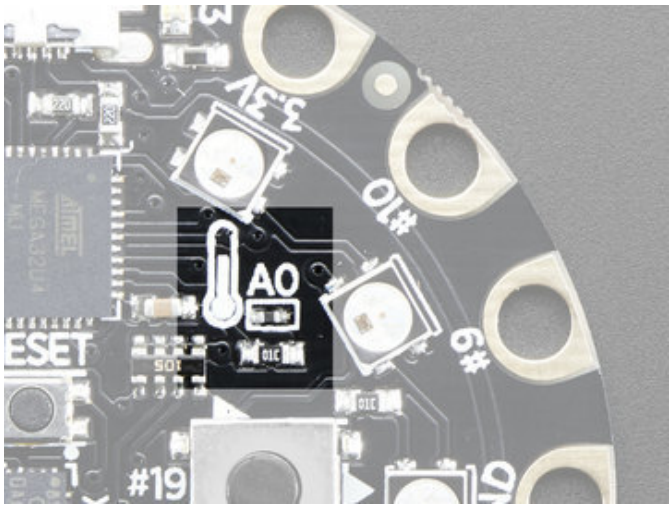



### Light Sensor

There is an analog light sensor, part number ALS-PT19 (https://adafru.it/tC2), in the top left part of the board. This can be used to detect ambient light, with similar spectral response to the human eye.

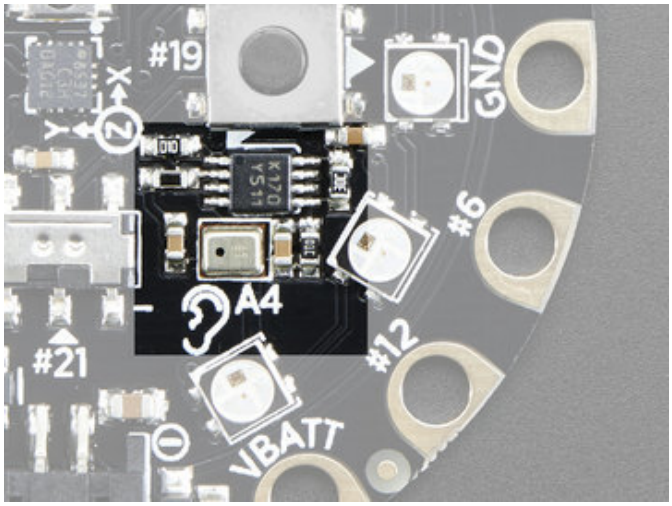
This sensor is connect to analog pin **A5** and will read between 0 and 1023 with higher values corresponding to higher light levels. A reading of about 300 is common for most indoor light levels.

With some clever code, you can use this as a color sensor or even a pulse sensor!



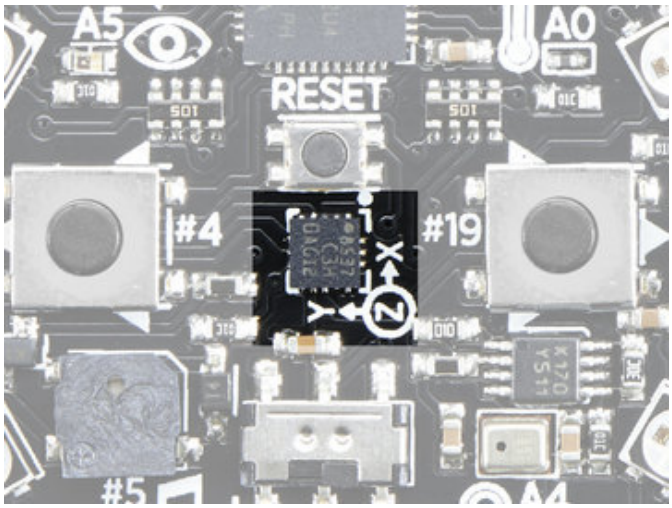

## Temperature Sensor

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.




## Microphone Audio Sensor

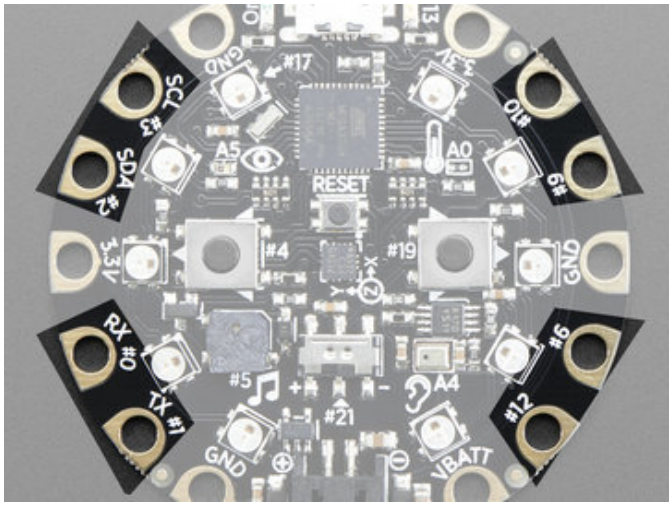
A MEMS microphone can be used to detect audio levels and even perform basic FFT functions. You can read the analog voltage corresponding to the audio on analog pin **#A4**. Note that this is the raw analog audio waveform! When it's silent there will be a reading of ~330 and when loud the audio will read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




## Motion Sensor

We can sense motion with an accelerometer. This sensor detects *acceleration* which means it can be used to detect when its being moved around, as well as gravitational pull in order to detect orientation.

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)

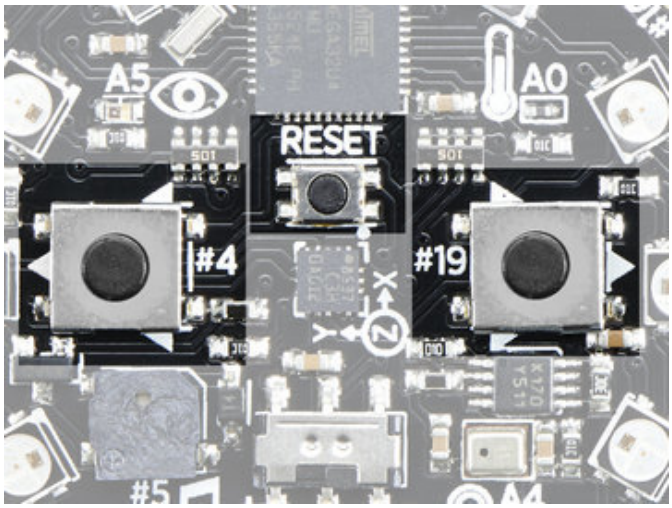



## Capacitive Touch

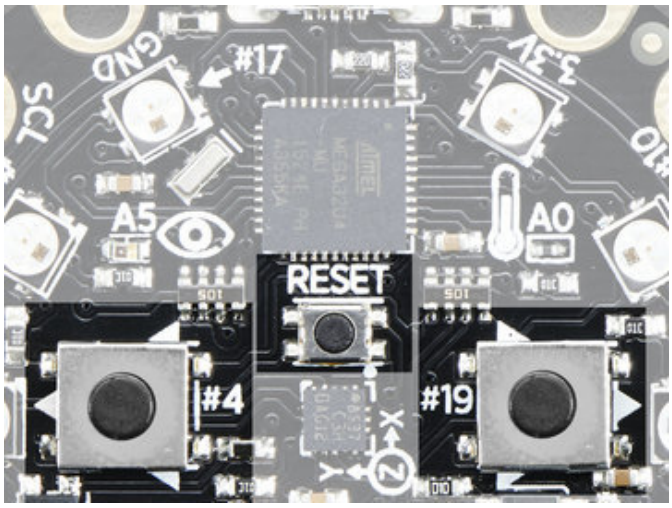
The CPC has some resistor pullups and an extra pin that gives it the ability to perform capacitive touch readings. This is a great way to sense human touch without additional components. Even animals will work if its directly touching their skin!

On the Classic you get **eight** capacitive touch pads (all GPIO pads)

# Switches & Buttons

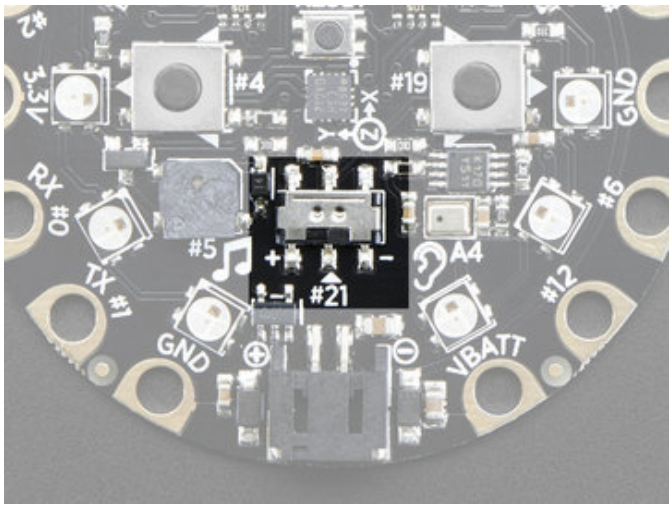



There are two large buttons, connected to digital **#19** (Left) and **#5** (Right) each. These are pulled to ground when not pressed, and connected to 3.3V when pressed, so they read HIGH.




This small button in the center of the board is for **Resetting** the board. You can use this button to restart or reset the CPC.

Press this button **once** to reset, **double-click** to enter the bootloader manually.



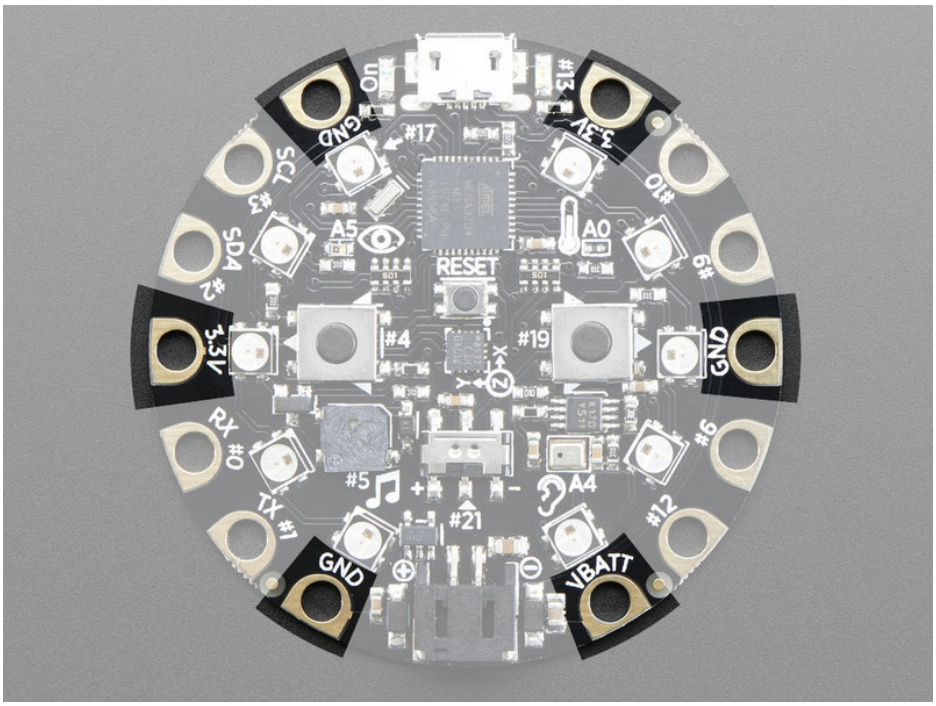

There is a single slide switch near the center of the Circuit Playground. It is connected to digital **#21** and will read LOW when slid to the left, and HIGH when in the right hand position

# Pinouts

Despite having only 14 pads with 8 general purpose I/O pins available, there are a *lot* of possibilities with Circuit Playground Classic. We went over all the internals in the last page. On this page we'll go through each pin/pad to explain what you can do with it.

**No external I/O pads are shared with internal sensors/devices, so you do not need to worry about 'conflicting' pins or interactions!**

## Power Pads

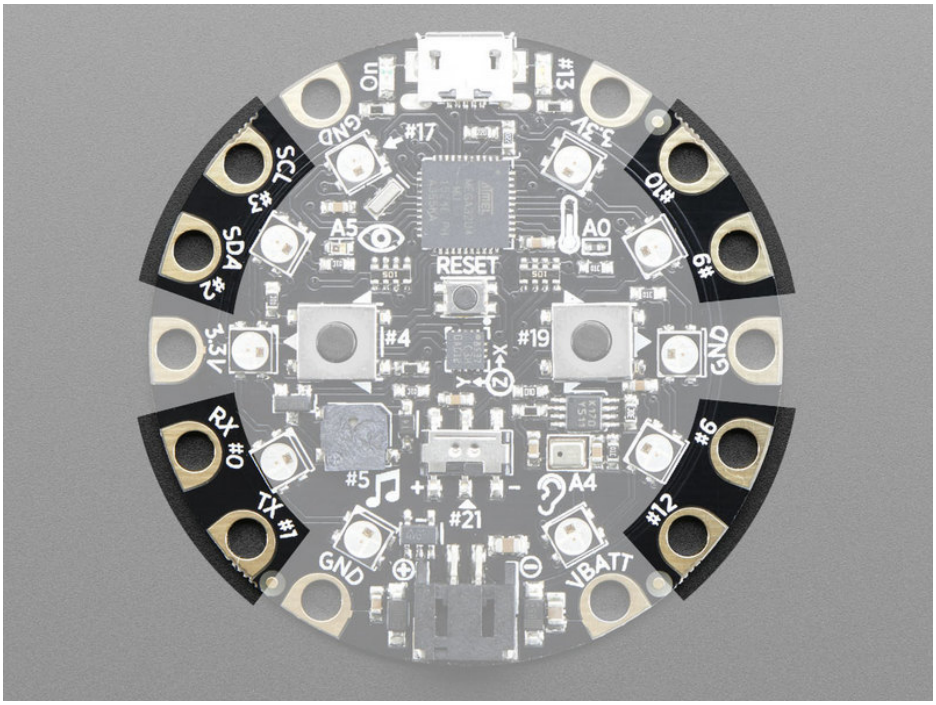



There are 6 power pads available, equally spaced around the perimeter.

- **GND** - there are 3 x **Ground** pads. They are all connected together, and are all the signal/power ground connections
- **3.3V** - there are two **3.3 Volt output** pads. They are connected to the output of the onboard regulator. The regulator can provide about 500mA max, but that includes all the built in parts too! So you should roughly budget about 300mA available for your usage (450mA if you are not using the onboard NeoPixels)
- **VBATT** - there is one **Voltage Output** pad. This is a special power pad,it will be connected to *either* the USB power or the battery input, whichever has the higher voltage. This output does not connect to the regulator so you can draw as much current as your USB port / Battery can provide.

## Input/Output Pads

Next we will cover the 8 GPIO (General Purpose Input Ouput) pins! For reference you may want to also check out the datasheet-reference in the downloads section for the core ATMEGA32U4 pin. We picked pins that have *a lot* of capabilities.




## Common to all pads

**All the GPIO pads can be used as digital inputs, digital outputs, for LEDs, buttons and switches.** In addition, all can be used as analog inputs (10-bit ADC). All pads can be used for hardware capacitive touch.

Each pad can provide up to ~20mA of current. Don't connect a motor or other high-power component directly to the pins! Instead, use a transistor to power the DC motor on/off (https://adafru.it/aUD)

All of the GPIO pads are 3.3V output level, and should not be used with 5V inputs. In general, most 5V devices are OK with 3.3V output though.

All of the pads are completely 'free' pins, they are not used by the USB connection, LEDs, sensors, etc so you never have to worry about interfering with them when programming.

- D6, D9, D10 and D11 can be **analog inputs**
- D3, D6, D9 and D10 can be **PWM outputs**
- D0, D1, D2 and D3 can be **hardware interrupt input**

## Each Pin!

Let's start with **#10** which is in the top right corner, and work our way clockwise

- **D10** / **A10** - This pin can be digital I/O, or Analog Input. This pin has PWM output
- **D9** / **A9** - This pin can be digital I/O, or Analog Input. This pin has PWM output.
- **D6** / **A7** - This pin can be digital I/O, or Analog Input. This pin has PWM output.
- **D12** / **A11** - This pin can be digital I/O, or Analog Input.
- **D1** - This pin can be digital I/O, it is also used for **Hardware Serial Transmit,** and can be an interrupt input.
- **D0** - This pin can be digital I/O, it is also used for **Hardware Serial Receive,** and can be an interrupt input.
- **D2** - This pin can be digital I/O, it is also the **I2C SDA** pin, and can be an interrupt input
- **D3** - This pin can be digital I/O or PWM output, it is also the **I2C SCL** pin, and can be an interrupt input

## Internally Used Pins!

These are the names of the pins that are used for built in sensors and such!

- **D4** - Left Button A
- **D5** - Speaker PWM output
- **D7** - Accelerometer interrupt
- **D13** - Red LED
- **D17** - Built-in 10 NeoPixels
- **D19** - Right Button B
- **D21** - Slide Switch
- **A0** - Temperature Sensor
- **A4 -** Microphone sound sensor
- **A5** - Light Sensor

# Windows Driver Installation

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

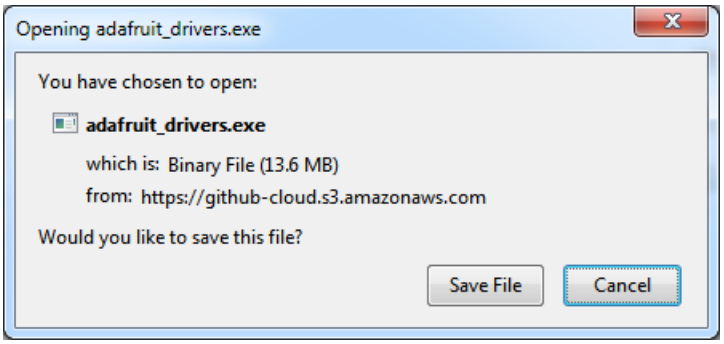
Before you plug in your board, you'll need to possibly install a driver!

Click below to download our Driver Installer.

https://adafru.it/AB0

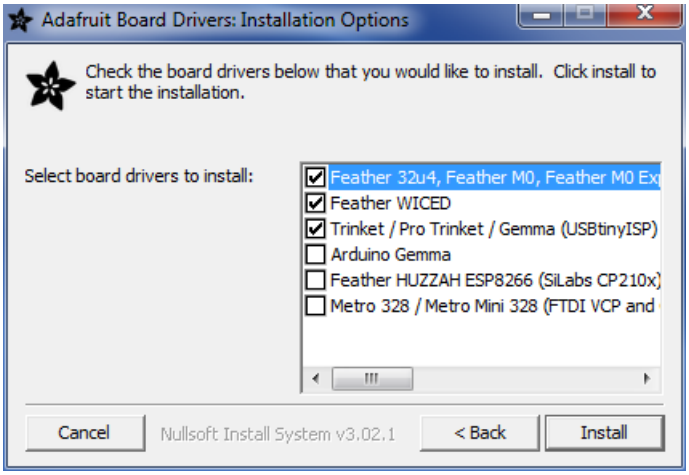
https://adafru.it/AB0

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, we suggest selecting all of them so you don't have to do this again!



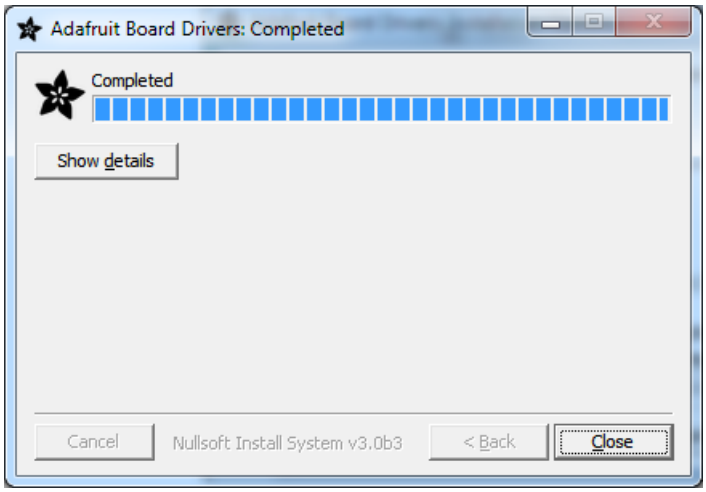

On Windows 7, by default, we install a single driver for most of Adafruit's boards, including the **Feather 32u4, the Feather M0, Feather M0, Express, Circuit Playground, Circuit Playground Express, Gemma M0, Trinket M0, Metro M0 Express**. On Windows 10 that driver is not necessary (it's built in to Windows) and it will not be listed.

The **Trinket / Pro Trinket / Gemma / USBtinyISP** drivers are also installed by default.

You can also, optionally, install the **Arduino Gemma** (different than the Adafruit Gemma!), **Huzzah and Metro 328** drivers.

Click **Install** to do the installin'.

Note that on Windows 10, support for many boards is built in. If you end up not checking any boxes, you don't need to run the installer at all!




## Manual Driver Installation

If windows needs the driver files (inf/cat) for some reason you can get all the drivers by downloading the source code zip file from this link:

https://adafru.it/AB0

And point windows to the **Drivers** folder when it asks for the driver location

# Arduino

> Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards (https://adafru.it/oVa) are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (https://adafru.it/oVb) (based on Wiring (https://adafru.it/oVc)), and the Arduino Software (IDE) (https://adafru.it/fvm), based on Processing (https://adafru.it/ddm).
>
> -- https://www.arduino.cc/en/Guide/Introduction (https://adafru.it/Bf6)

Arduino has over a decade of projects and history, so you'll find a lot of existing code that you can use with your Circuit Playground Classic.

Since Circuit Playground is now built into Arduino, it's great for beginners - and you can take advantage of the huge Arduino community.

For experts - Arduino is essentially C/C++ with a built in library of hardware interfaces. You can embed assembly, write ultra-fast code, and twiddle registers.

# Set Up & Test Arduino

The Circuit Playground Classic is 'natively' supported in the Arduino IDE so its **really easy** to set up!

## Download Latest Arduino IDE

Download the latest Arduino IDE, **version 1.8.5 or greater** is required!

You can also use Arduino Create, in which case the IDE is already the latest version

https://adafru.it/fvm

https://adafru.it/fvm

## Install Drivers (Windows 7 Only)

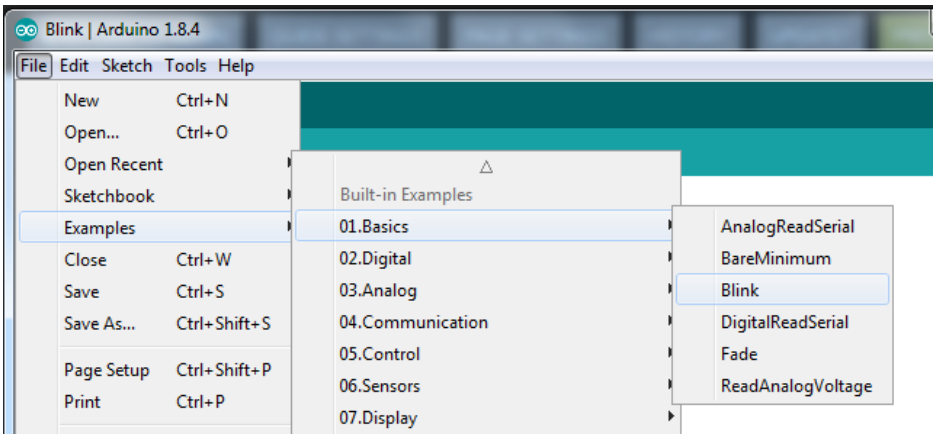
When you plug in the board, you'll need to possibly install a driver

Click here to download our Driver Installer (https://adafru.it/B-2)

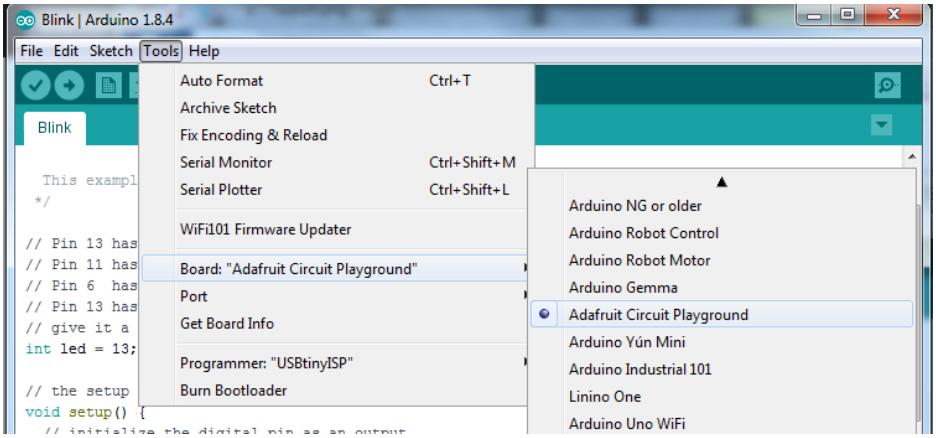
## Blink

Now you can upload your first blink sketch!

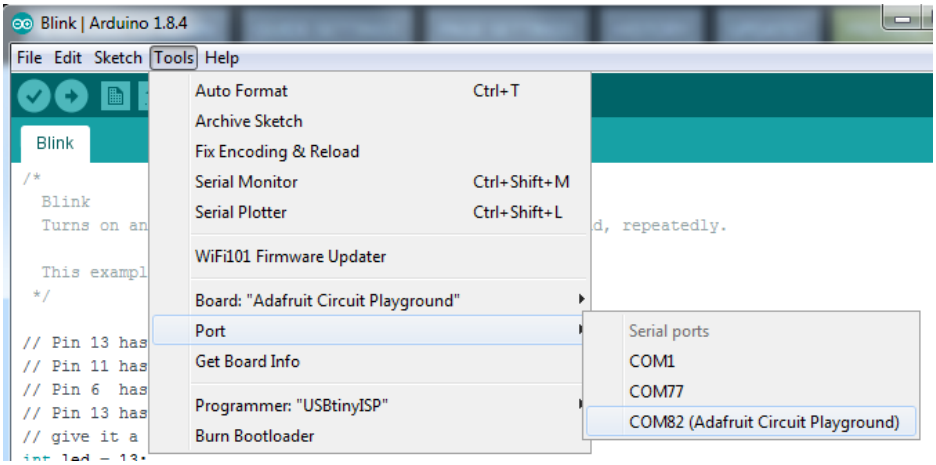
Open up the Blink example from the Arduino IDE




Select **Circuit Playground** from the Tools -> Board dropdown menul




Plug in the Circuit Playground 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 dropdown, it'll even be 'indicated' as a Circuit Playground board!




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.

## Manually bootloading

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

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 **RESET** button **twice** (like a double-click) to get back into the bootloader.

**The red LED will pulse so you know that its in bootloader mode.**

In the Arduino IDE, **re-select the Serial Port** to the new port that has been created for the bootloader.

Then upload **Blink** - make sure that works!

Once that works, go back and re-select the 'normal' USB serial port next time you want to use the normal upload.

## Ubuntu & Linux Issue Fix

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

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

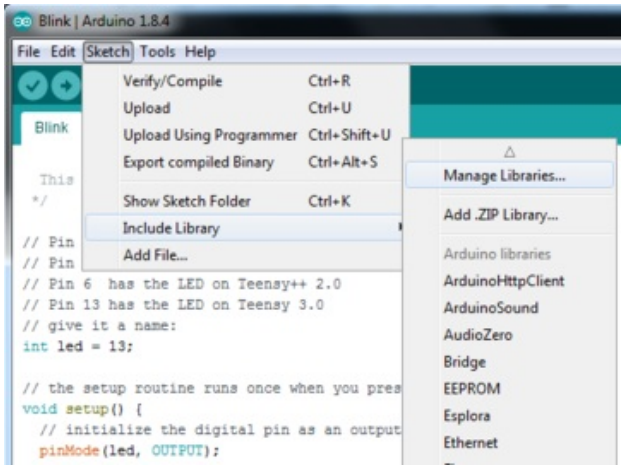
# Circuit Playground Library

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 (https://adafru.it/naF). but what's nice is that Arduino IDE comes with a version of the library and its super easy to update.

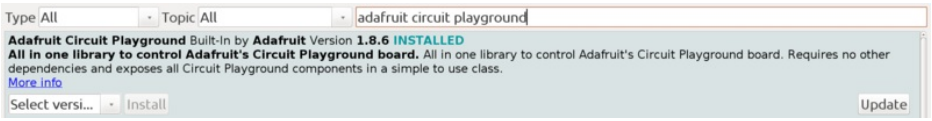
We recommend updating the library even if you just installed the Arduino IDE, we have constant updates!




In the menubar click "Sketch", then "Include Library"

At the top, click "Manage Libraries. . ."

Search for **Adafruit Circuit Playground** library and click **Update** to get the very latest version!



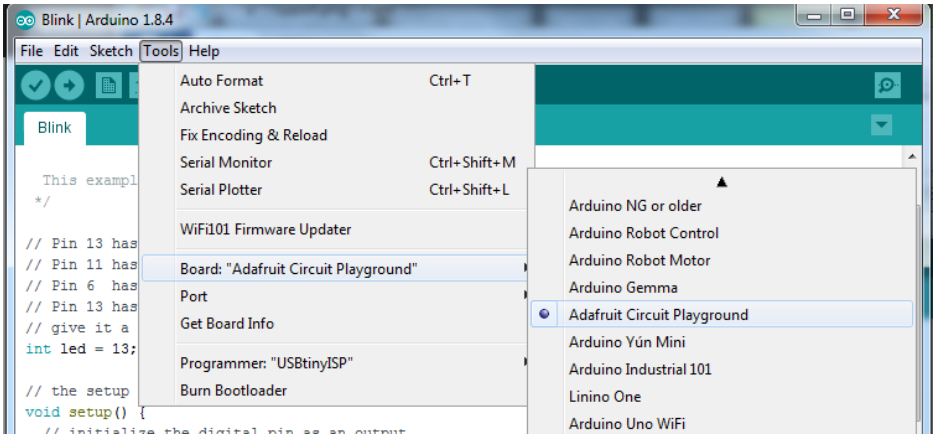

## Run the Demo

Now that you have the library installed you can continue.

Follow the Arduino Set Up guide page to make sure you can **Blink** upload. (https://adafru.it/B-3)Once that's known to work, come back here.

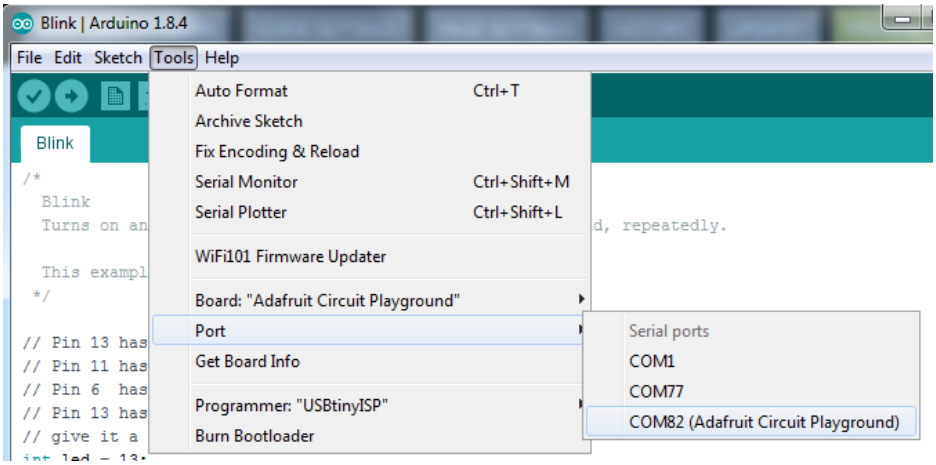
### Select the Circuit Playground Board

Under the **Tools -> Board** submenu, pick **Adafruit Circuit Playground**



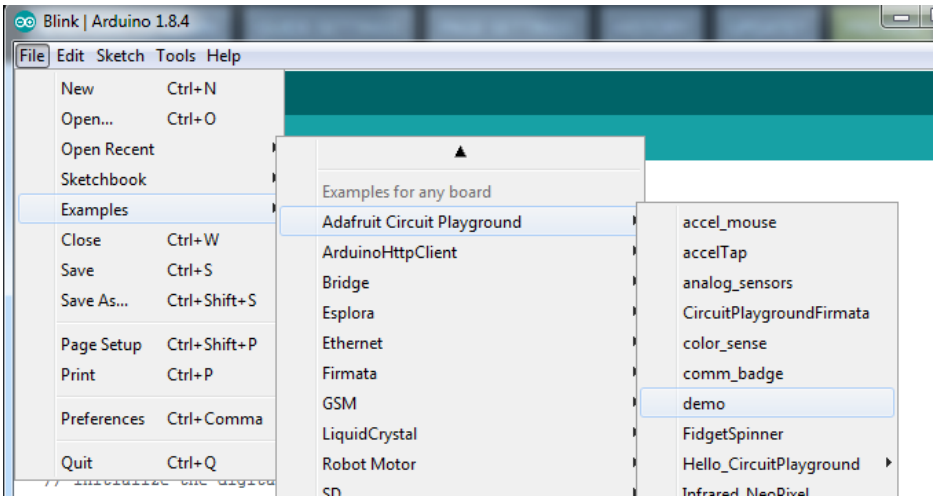

## Select the matching Port

Under **Tools->Port** select the port that is labeled (Circuit Playground)



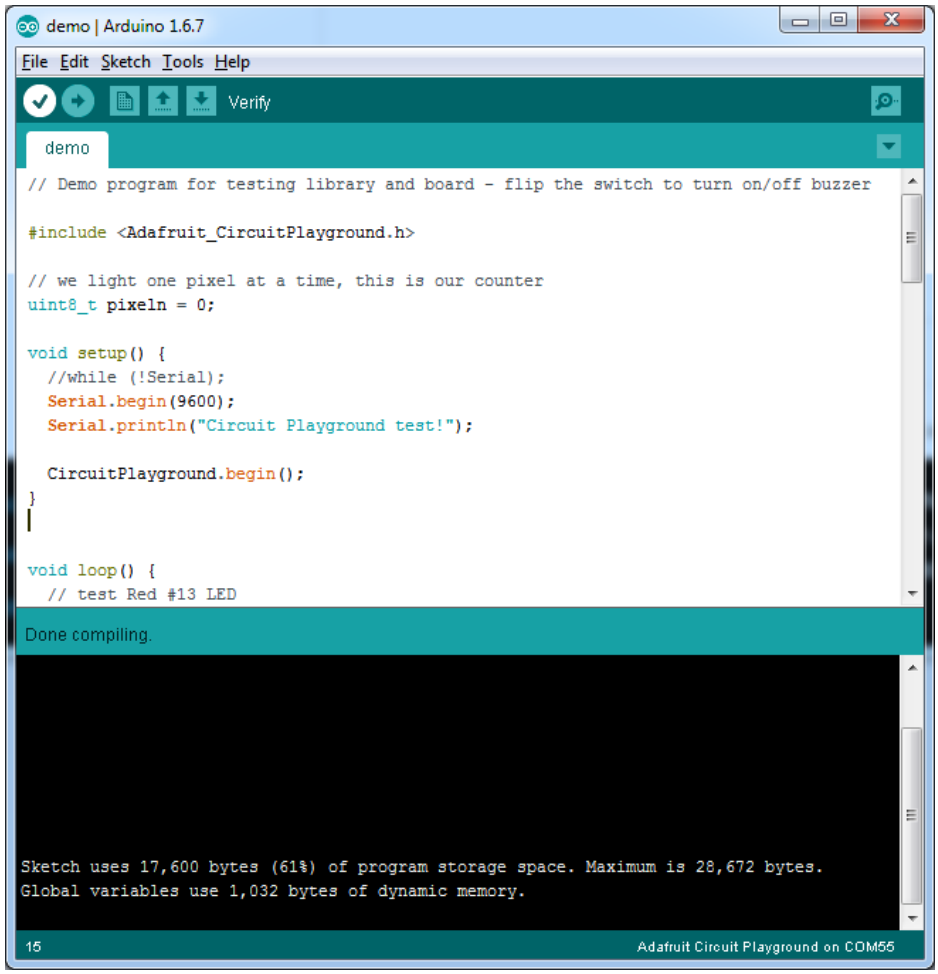

## Load the Demo Program

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



## Compile/Verify the Demo

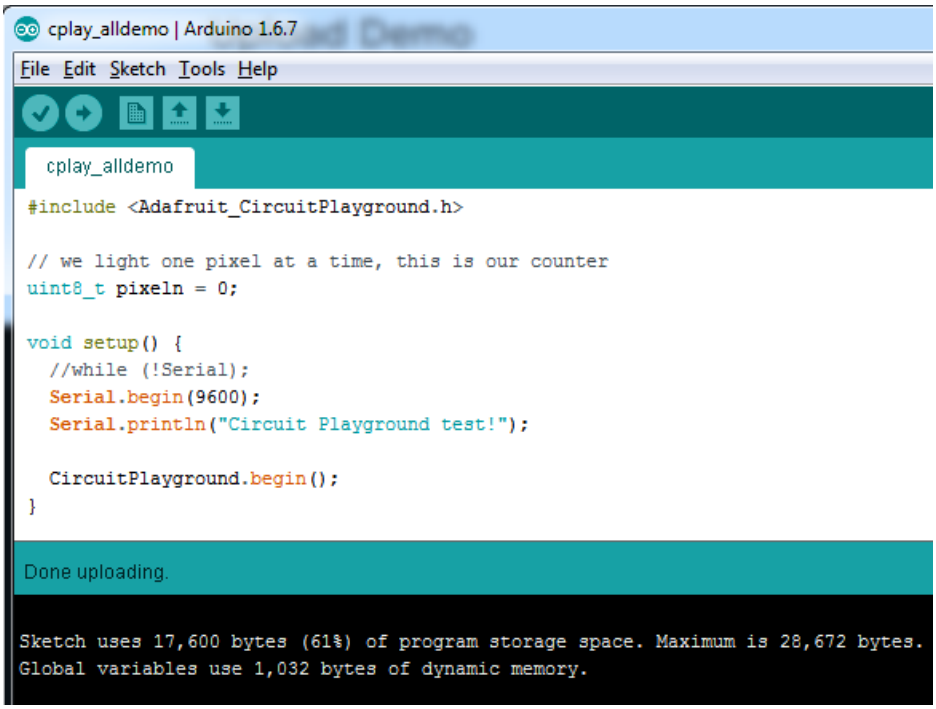
Click the **Verify** button (also the **Sketch->Verify** menu item) to compile/verify the demo. Make sure you get "**Done compiling.**" and no errors



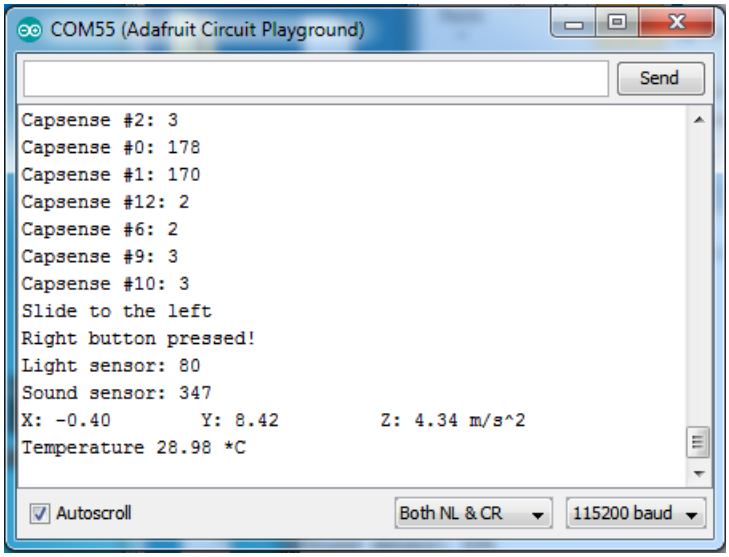

## Upload Demo

Click the Upload button to upload the code

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




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




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 Celsius

# HELP!

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

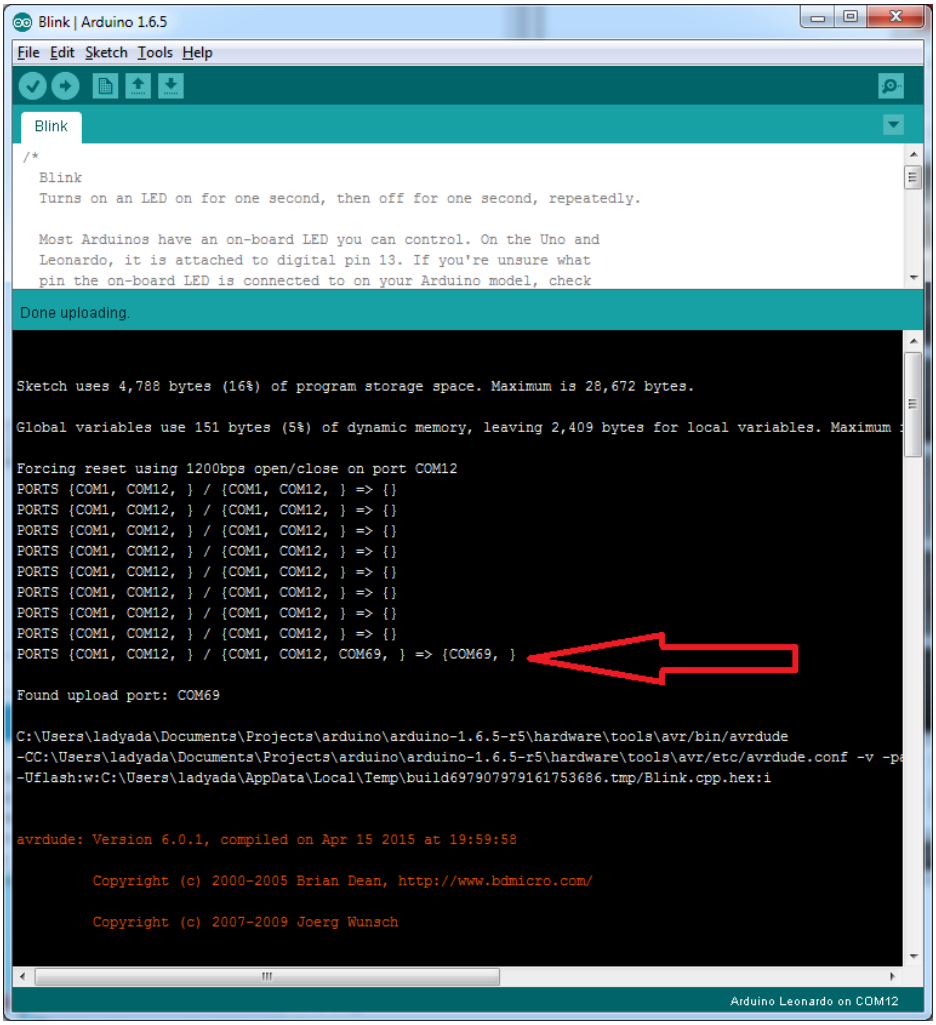
99% of initial problems with circuit playground are due to having **charge USB cables** instead of sync cables. Do not use a cable you've only used for charging a phone. Make sure its a cable that can pass data as well as power. Lately, there's been a lot of products shipped with charging only cables 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




## 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.

# Downloads

## Windows Driver Software

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

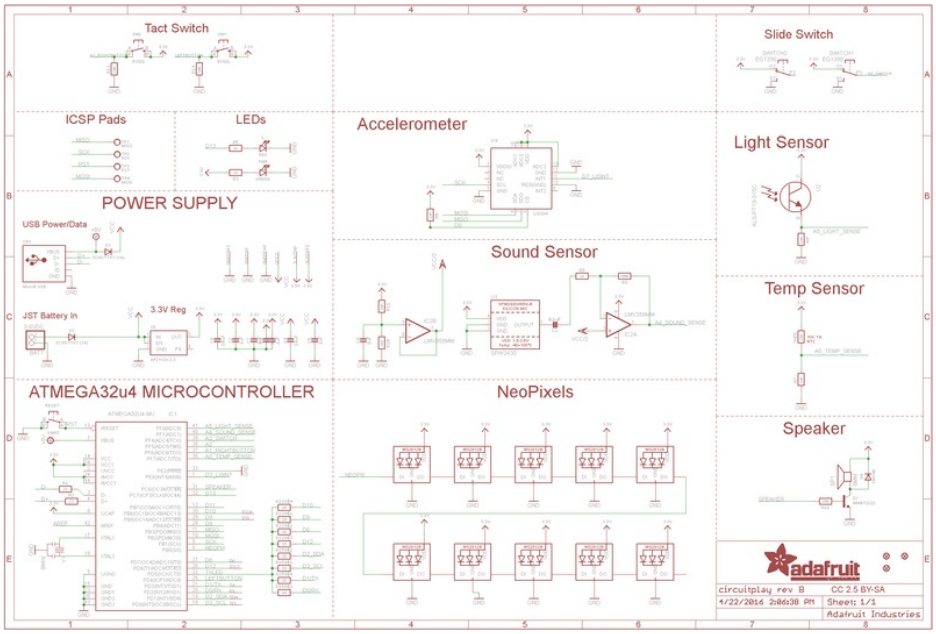
## Source

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

## Datasheets

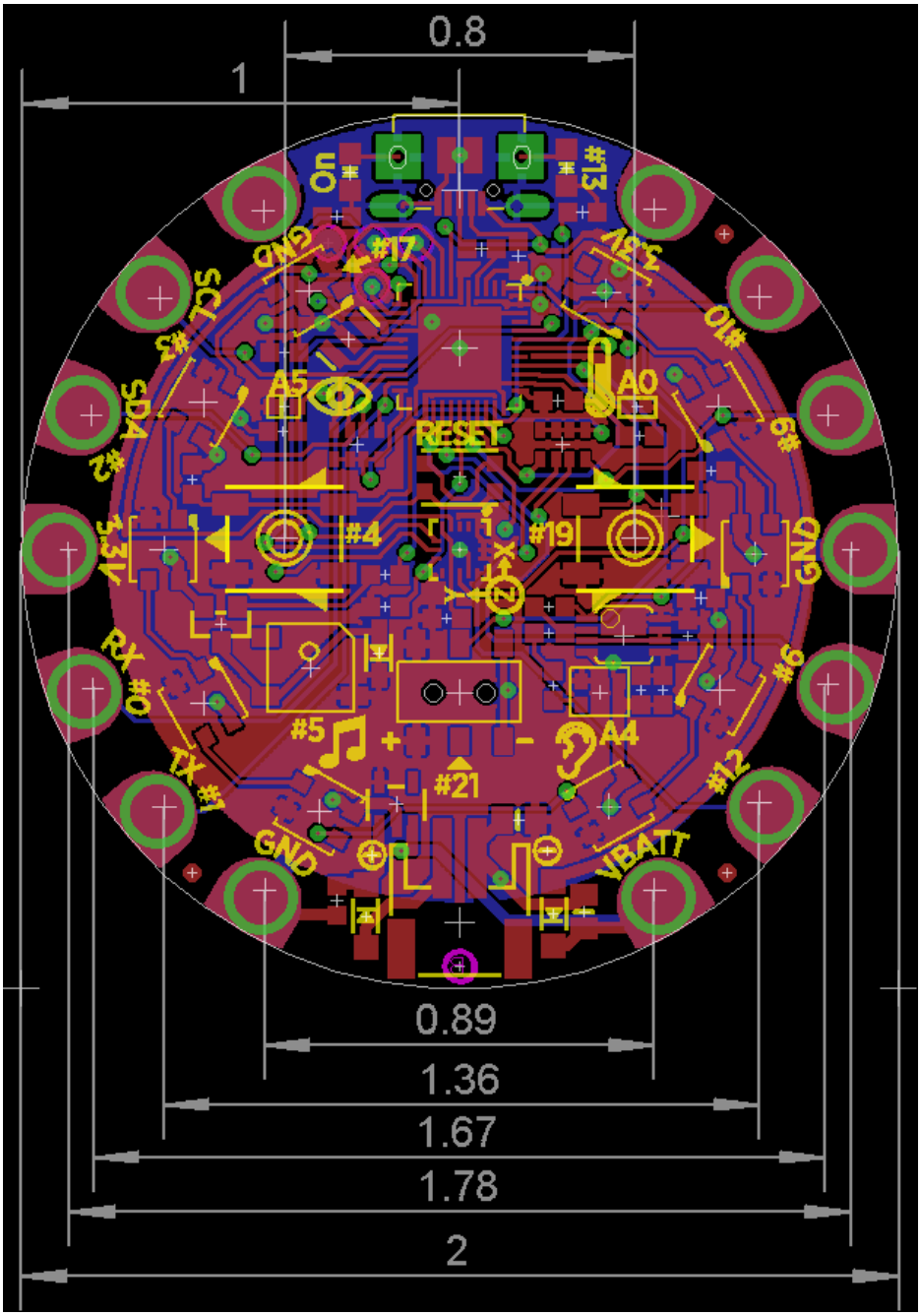
- Microcontroller datasheet (https://adafru.it/ecK)
- Buzzer datasheet (https://adafru.it/vCA)
- MEMS microphone datasheet (https://adafru.it/vCB)
- Thermistor datasheet (https://adafru.it/vCC)

## Schematic




## Fabrication Print

Dims in inches


0.8
1
SCL #3
SDA #2
3.3V
RX #0
TX #1
GND
#17
3.3V
#13
#10
#9
#6
#12
VBATT
GND
A5
A0
RESET
#4
#19
GND
#5
#21
A4
0.89
1.36
1.67
1.78
2