Adafruit Circuit Playground Tri-Color E-Ink Gizmo

Created by Kattni Rembor

https://learn.adafruit.com/adafruit-circuit-playground-tri-color-e-ink-gizmo

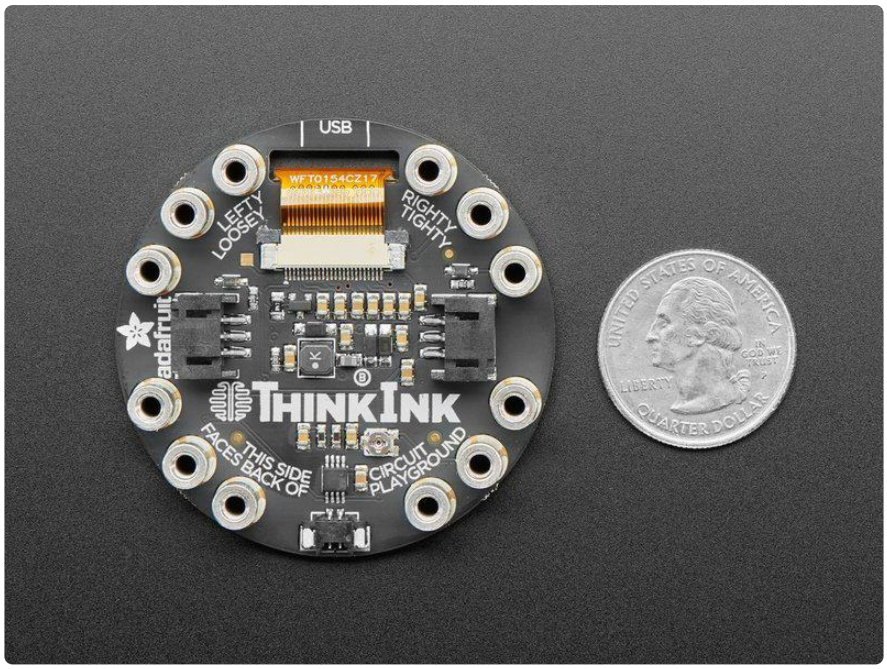
Last updated on 2022-12-01 03:44:59 PM EST

Table of Contents

Overview

Extend and expand your Circuit Playground projects with a bolt on E-Ink Gizmo that lets you add a lovely tri-color e-Ink display in a sturdy and reliable fashion. This PCB looks just like a round E-Ink breakout but has permanently affixed M3 standoffs that act as mechanical and electrical connections.

What is e-Ink? Chances are you've seen one of those new-fangled 'e-readers' like the Kindle or Nook. They have gigantic electronic paper 'static' displays - that means the image stays on the display even when power is completely disconnected. The image is also high contrast and very daylight readable. It really does look just like printed paper!



Once attached you'll get a 1.54" 152x152 display (black and red ink pixels and a white-ish background), two 3-pin STEMMA connectors for attaching NeoPixel strips () or servos (), and a Class D audio amplifier with a Molex PicoBlade connector that can plug one of our lil speakers ().

Using our CircuitPython or Arduino library, you can create a 'frame buffer' with what pixels you want to have activated and then write that out to the display. The library we wrote does all the work for you (), you can just interface with it as if it were an Adafruit_GFX compatible display (). This E-Ink breakout does not contain an extra SRAM chip (not enough pins are available) so you'll need ~7K of SRAM to spare for a

display buffer, the Adafruit Circuit Playground Express (CPX) has 32KB and the Adafruit Circuit Playground Bluefruit (CPB) has 256KB, so there's plenty of RAM.

This is a great companion for our Circuit Playground Express or Bluefruit boards thanks to their fast SPI hardware speeds and plenty of RAM, and works in Arduino and CircuitPython. If you're using this with Circuit Playground Express and CircuitPython, you won't be able to do a lot because there isn't a ton of memory - mostly just displaying the REPL and maybe running an image slideshow. For CircuitPython use, the Bluefruit is recommended and works really great!

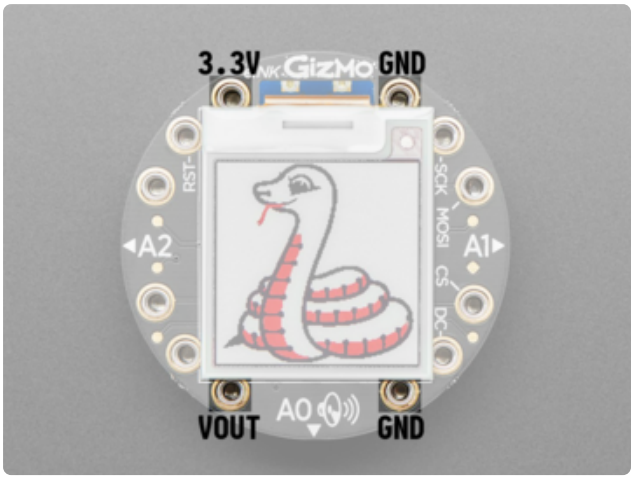
You cannot use it with the Circuit Playground Classic in Arduino, there's not enough RAM!

This comes with a PCB that has pre-soldered standoffs attached, and 12x M3 screws for attachment. It fits all Circuit Playgrounds, but like we mentioned earlier, the Express and Bluefruit are recommended.

Pinouts

The E-Ink Gizmo adds an e-ink display to you Circuit Playground, but it also adds some other extras. Let's take a look!

Power Pins



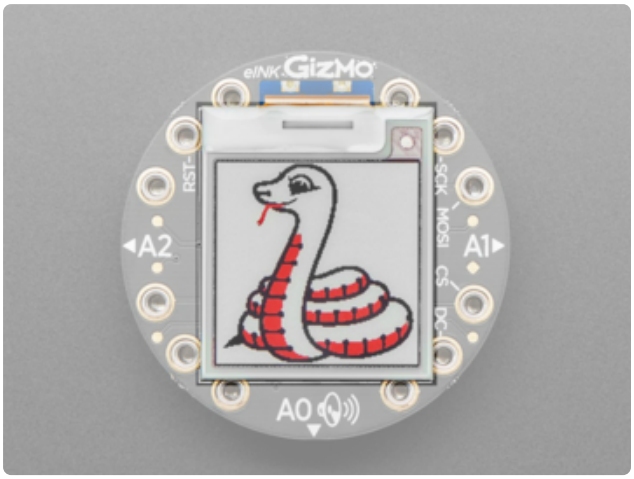
VOUT - power input, connect to 3-5VDC.
GND - power and signal ground. Connect to your power supply and microcontroller ground.
3.3V is the output from the onboard 3.3V regulator. If you have a need for a clean 3.3V output, you can use this! It can provide at least 100mA output.

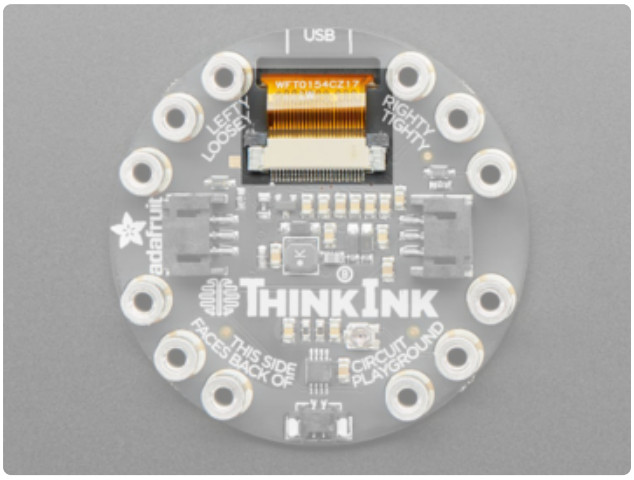
SPI Display Pins

The display used on the E-Ink Gizmo is a 1.54" tri-color e-Ink display.

SCK - this is the SPI clock pin, its an input to the chip. This is connected to A4 or SCL.
MOSI - this is the Microcontroller Out Serial In pin, for data sent from your processor to the e-ink display. This is connected to A5 or SDA.
CS - this is the chip select pin, drop it low to start an SPI transaction. Its an input to the chip. This is connected to A6 or RX.
DC - this is the e-ink SPI data or command selector pin. This is connected to A7 or TX.
RST - this is the E-Ink ReSeT pin. It is connected to A3.

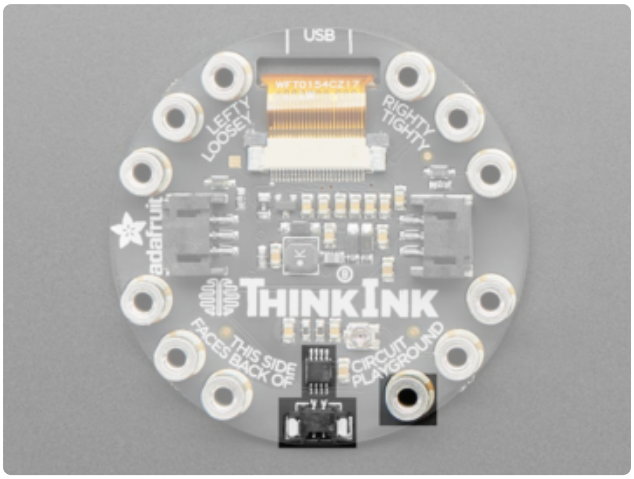
E-Ink Display





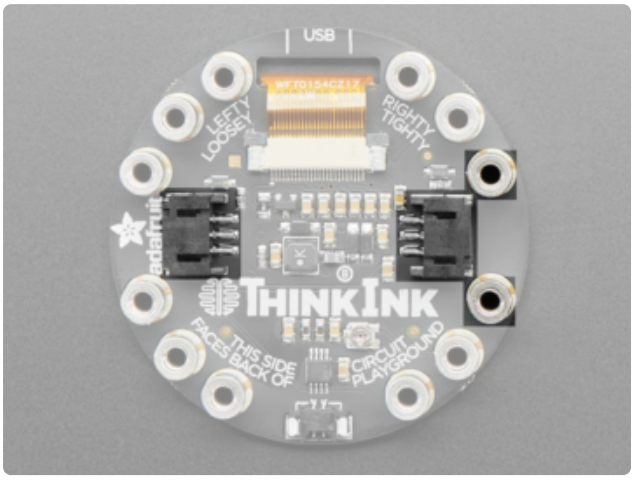
Front and center is a 1.54" 152x152 tri-color e-ink display, with black and red ink pixels and a white-ish background.
The display cable goes through to the back and connects to the display connector towards the top in the middle of the back.

Speaker Connector



On the bottom of the back, there is a Molex Picoblade speaker connector and a class D amplifier connected to pin A0. The location is labeled on the front as A0 with a speaker symbol.

STEMMA Connectors



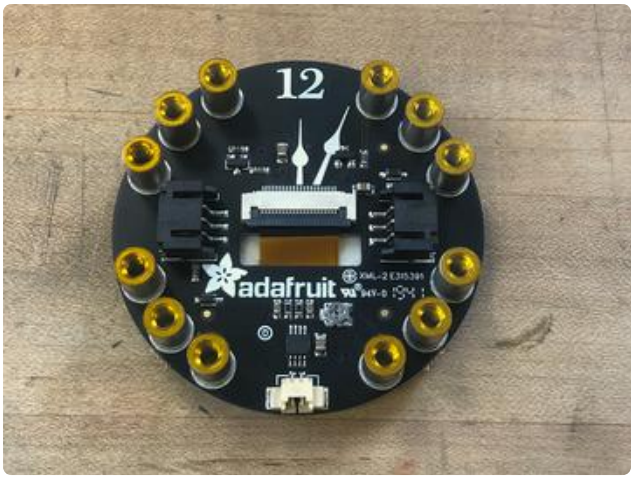
On the sides of the back are two 3-pin STEMMA connectors for hooking up NeoPixels or servos connected to pins A1 and A2.
The locations are labeled on the front as A1 and A2.

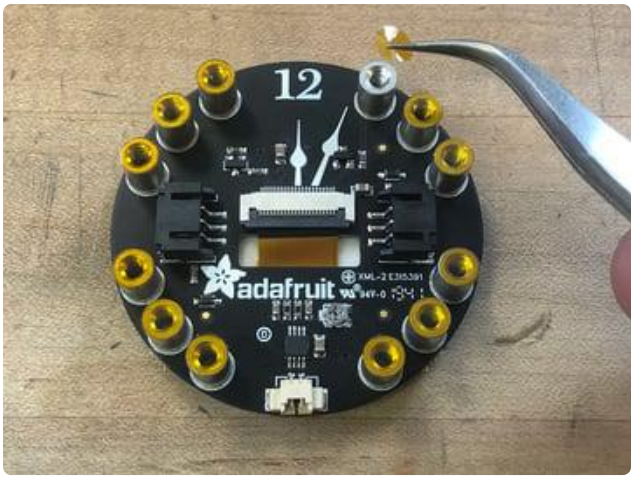
Assembly

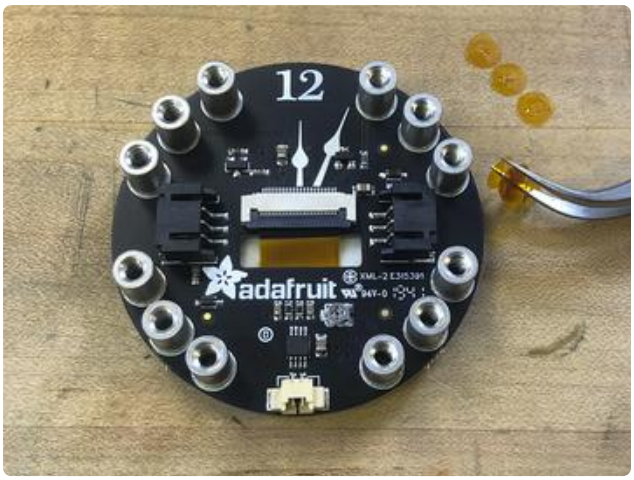
This page shows assembling the Circuit Playground TFT Gizmo, but the process is identical for the E-Ink Gizmo.

Placing the Circuit Playground TFT Gizmo on the Circuit Playground Express or Circuit Playground Bluefruit is pretty straightforward. All you need is a #2 Phillips screwdriver.

There may be plastic covers over the screw holes on the TFT Gizmo, which you will need to remove before assembly.

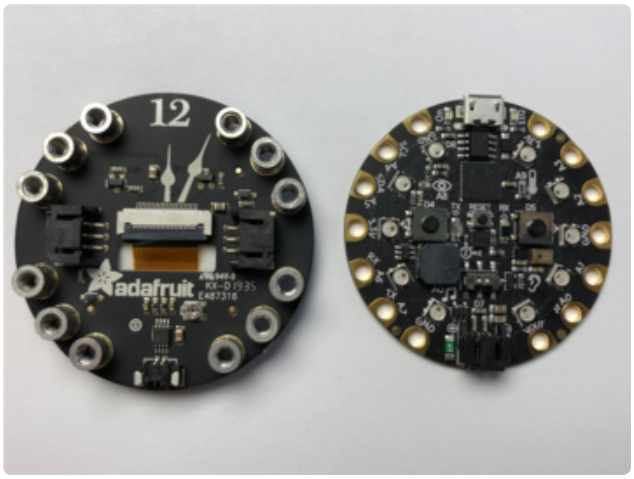




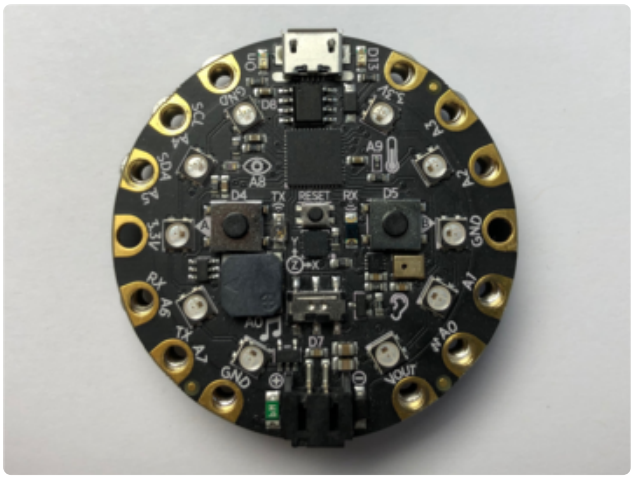


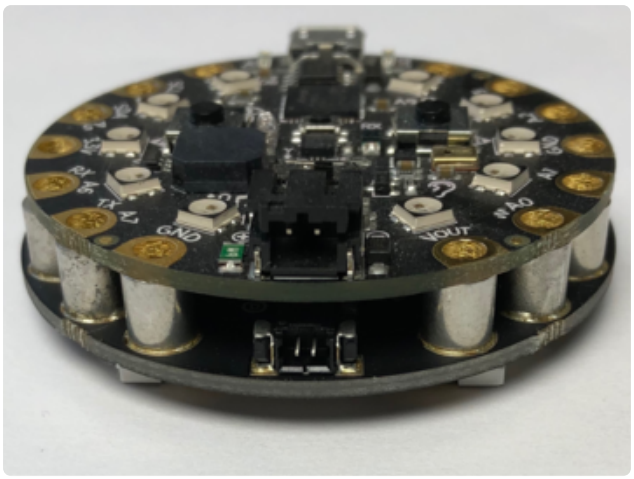
The amber colored Kapton tape dots must be removed from each of the twelve standoffs before assembling the boards. These are electrically insulating and will prevent the Gizmo from working properly if left in place.

You can use your fingernails or some tweezers or a pin to poke and lift each dot as shown here.

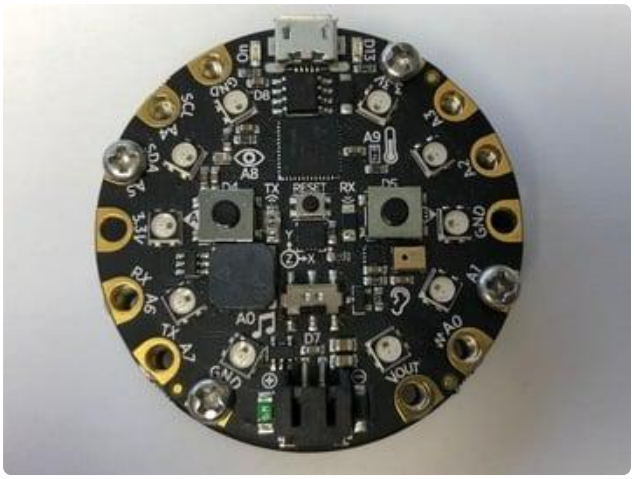


Start by aligning the two boards side by side like in the photo with the black plastic speaker connector and battery connectors pointing in the same direction.

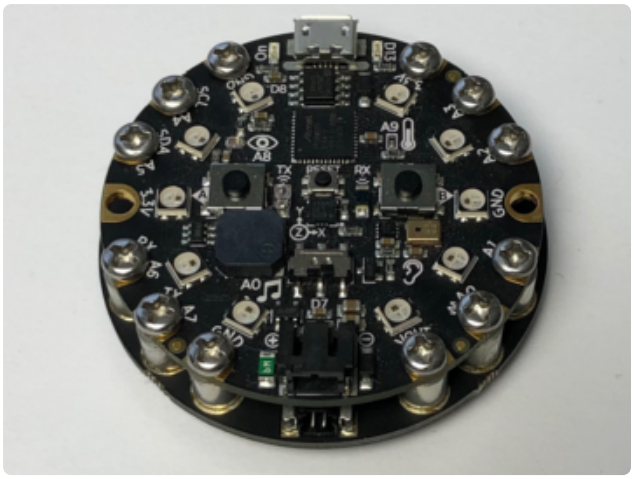




Place the Circuit Playground board on top of the Gizmo being sure that the connectors mentioned in the previous step are still aligned.



Install a few screws loosely, so that all of the holes are still aligned, before tightening them down.



Finish installing the remaining screws. After that, you're done!

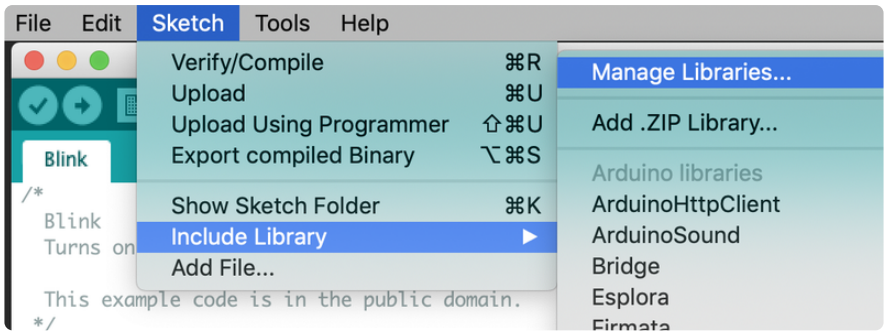
Arduino Code

Do not update more than once every 180 seconds or you may permanently damage the display

Install Adafruit_EPD & GFX libraries

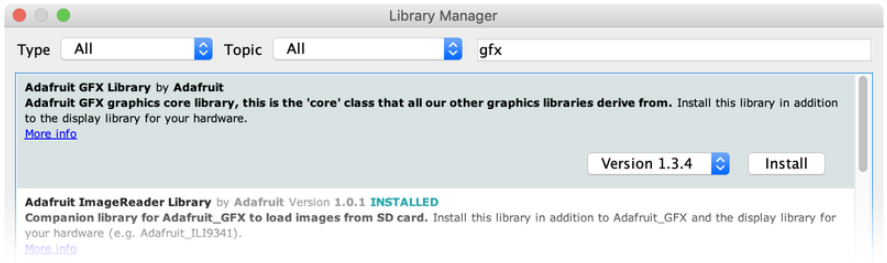
To begin reading sensor data, you will need to install the Adafruit_EPD library (code on our github repository) (). It is available from the Arduino library manager so we recommend using that.

From the IDE open up the library manager...



And type in Adafruit EPD to locate the library. Click Install

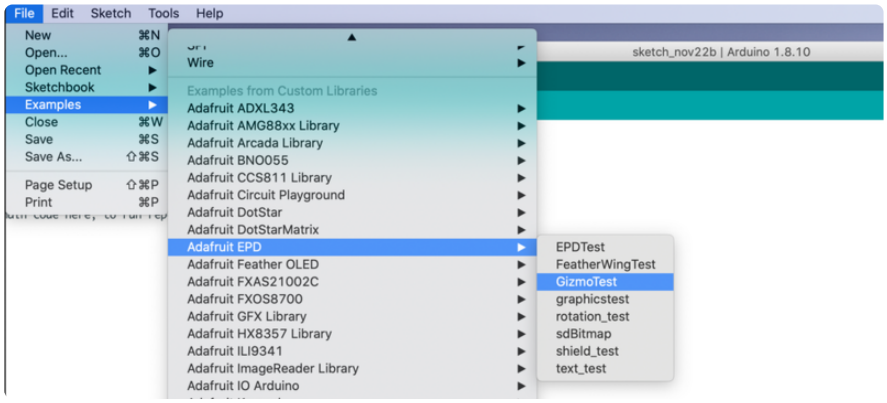
Do the same to install the latest Adafruit GFX library, click Install



While we're in the Library Manager, we need to install a few more libraries. Search for and install Adafruit BusIO, Adafruit ImageReader Library, Adafruit SPIFlash, and SdFat - Adafruit Fork.

Load E-Ink Gizmo Demo

Open up File→Examples→Adafruit EPD→GizmoTest



For Circuit Playground Express, be sure to use the Adafruit board definition rather than the Arduino one.

Configure Display Type & Size

Find the part of the script where you can pick which display is going to be used. The eInk displays are made up a combination of a Chipset and a Film in different sizes. We have narrowed it down to just a few choices between the size of the display, chipset, and film based on available combinations. In the sketch, we have sorted it by size, so it's easy to find your display.

You will need to uncomment the appropriate initializer and and leave any other type commented.

For the 1.54" 152x152 Tri-Color Gizmo (), you will use the `ThinkInk_154_Tricolor_Z17` display initializer.

For the 1.54" 200x200 Tri-Color Gizmo (), you will use the `ThinkInk_154_Tricolor_Z90` display initializer.

For example, for the 200x200 Gizmo, uncomment this line, and comment any other line that is creating a ThinkInk display object

```
// 1.54" 200x200 Tricolor EPD with SSD1681 chipset
ThinkInk_154_Tricolor_Z90 display(EPD_DC, EPD_RESET, EPD_CS, SRAM_CS, EPD_BUSY);
```

For the older 152x152 Gizmo, uncomment this line, and comment any other line that is creating a ThinkInk display object

```
// 1.54" 152x152 Tricolor EPD with ILI0373 chipset
ThinkInk_154_Tricolor_Z17 display(EPD_DC, EPD_RESET, EPD_CS, SRAM_CS, EPD_BUSY);
```

Upload the example to your Circuit Playground Express or Circuit Playground Bluefruit. You should now see a series of demos running. Note that the demos are set to pause for 15 seconds between each demo which is ok for short term usage, but it is strongly recommended to use 180 between refreshes when used over a longer period of time.

Drawing Bitmaps

This will not work for the Circuit Playground Classic because it does not have SPI flash.

Not only can you draw shapes but you can also load images from QSPI flash, perfect for static images!

The 1.54" display can show a max of 152x152 pixels. Lets use this Blinka bitmap as our demo:

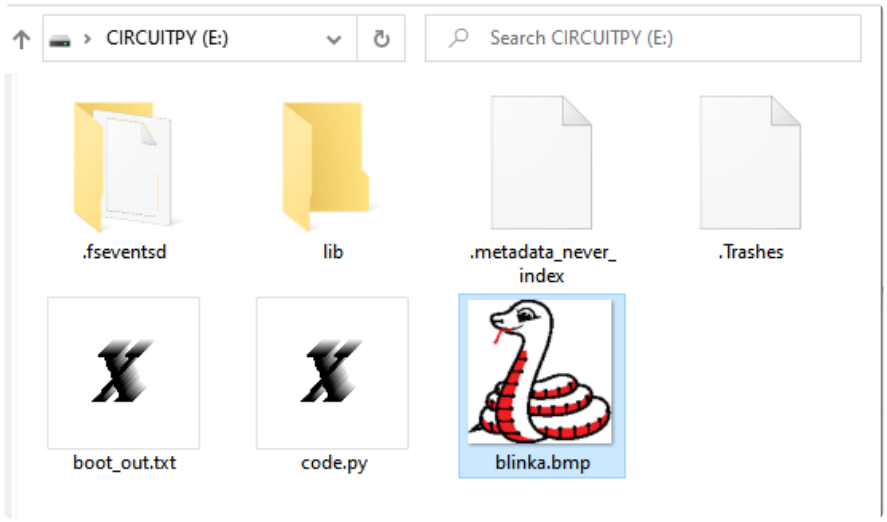
Click here to download 152x152 image of blinka

Click here to download 200x200 image of blinka

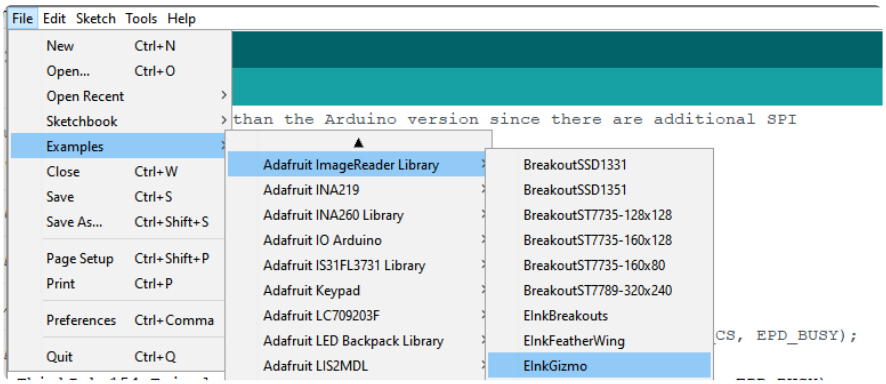
In order to copy the files to QSPI Flash, the easiest way is to load CircuitPython onto your device. If you are not sure how, you can check out our Welcome to CircuitPython guide ().

After you have installed CircuitPython on your device, you should see a CIRCUITPY drive appear in the list of drives.

Copy over and rename the file to blinka.bmp to the root of that drive. Once that is copied, you can continue.



Now in Arduino, open the File→Examples→Adafruit ImageReader Library→EInkGizmo example.



Upload to the Circuit Playground and you should see Blinka appear!

Arduino Library Documentation

Arduino Library Documentation ()

Adafruit GFX Library

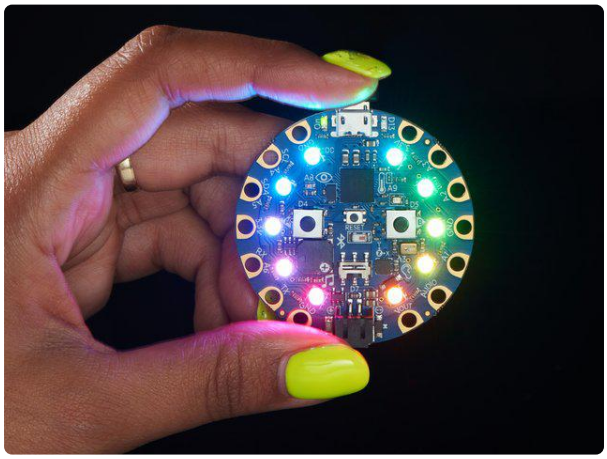
Adafruit GFX Library ()

CircuitPython

It's super simple to get started with the Adafruit Circuit Playground E-Ink Gizmo and CIrcuitPython using the Adafruit CircuitPython Gizmo () module and CircuitPython's built in `displayio`.

You will need a board capable of running CircuitPython such as the Circuit Playground Express or Circuit Playground Bluefruit. The Circuit Playground Classic will only run Arduino sketches.

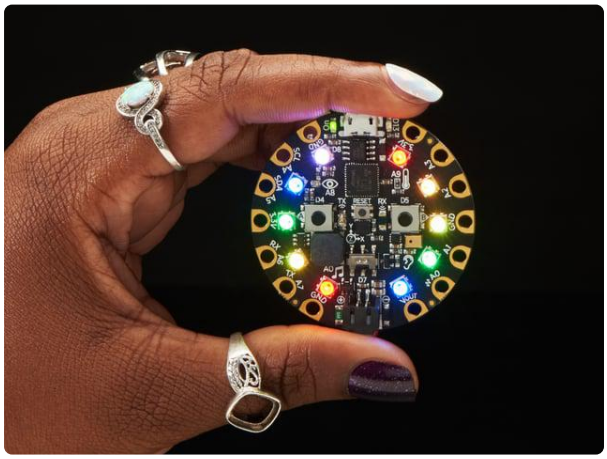
The Circuit Playground Express, which has a SAMD21 chip, may not be able to run the full example due to memory constraints. This is why we recommend using the Circuit Playground Bluefruit.



Circuit Playground Bluefruit - Bluetooth Low Energy

Circuit Playground Bluefruit is our third board in the Circuit Playground series, another step towards a perfect introduction to electronics and programming. We've...

https://www.adafruit.com/product/4333



Circuit Playground Express

Circuit Playground Express is the next step towards a perfect introduction to electronics and programming. We've taken the original Circuit Playground Classic and...

https://www.adafruit.com/product/3333

Circuit Playground Express with displayio

If you have a Circuit Playground Express board, you will need a special build that includes `displayio` to use the TFT Gizmo. Be sure to download the latest one. You can find it here:

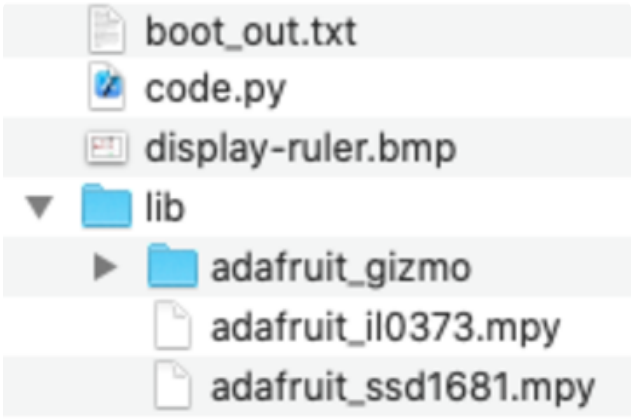
CircuitPython for Circuit Playground Express with displayio

CircuitPython Gizmo Library Installation

To use this display with `displayio`, you'll need to install the two required libraries onto your CircuitPython board.

First, make sure you are running the latest version of Adafruit CircuitPython () for your board.

Next, you'll need to install the necessary library to use the hardware Carefully follow the steps to find and install the library from Adafruit's CircuitPython library bundle. Our introduction guide has a great page on how to install the library bundle ().



You'll want to copy the following files and folders to the lib folder on your CIRCUITPY drive:

adafruit_il0373.mpy
adafruit_ssd1681.mpy
adafruit_gizmo
Before continuing make sure your board's lib folder has the adafruit_il0373.mpy, adafruit_ssd1681.mpy and adafruit_gizmo files and folder copied over.

CircuitPython Code

This example uses a bitmap image. Download display_ruler.bmp below and save it to your CIRCUITPY drive.

Download display-ruler.bmp

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import displayio
from adafruit_gizmo import eink_gizmo

display = eink_gizmo.EInk_Gizmo()
# Use the below line instead for the 200x200 E-Ink Gizmo
# display = eink_gizmo.EInk_HD_Gizmo()

# Create a display group for our screen objects
display_group = displayio.Group()
```

```
# Display a ruler graphic from the root directory of the CIRCUITPY drive
with open("/display-ruler.bmp", "rb") as file:
    picture = displayio.OnDiskBitmap(file)
    # Create a Tilegrid with the bitmap and put in the displayio group
    # CircuitPython 6 & 7 compatible
    sprite = displayio.TileGrid(
        picture,
        pixel_shader=getattr(picture, "pixel_shader", displayio.ColorConverter()),
    )
    # CircuitPython 7 compatible only
    # sprite = displayio.TileGrid(picture, pixel_shader=bitmap.pixel_shader)
    display_group.append(sprite)

    # Place the display group on the screen
    display.show(display_group)

    # Refresh the display to have it actually show the image
    # NOTE: Do not refresh eInk displays sooner than 180 seconds
    display.refresh()
    print("refreshed")

    time.sleep(180)
```

Let's take a look at the code!

We begin by importing `time`, `displayio` and the `eink_gizmo` helper.

```
import time
import displayio
from adafruit_gizmo import eink_gizmo
```

Next, we initialize the helper, which takes care of all the original e-ink Gizmo IL0373 driver initialization for us.

```
display = eink_gizmo.EInk_Gizmo()
```

If you have the 200x200 e-Ink Gizmo, you would instead create an `EInk_HD_Gizmo()` object, which takes care of the new HD e-ink Gizmo SSD1681 driver initialization:

```
display = eink_gizmo.EInk_HD_Gizmo()
```

Then we create a `group` to which we can add objects, such as the bitmap we want to display.

```
display_group = displayio.Group()
```

Next we open the bitmap file.

```
file = open("/display-ruler.bmp", "rb")
```

Then we convert the bitmap file into an object that `displayio` can work with, create a `TileGrid` with the bitmap object in it, and add it to the `group` we created.

```
picture = displayio.OnDiskBitmap(file)
sprite = displayio.TileGrid(picture, pixel_shader=getattr(picture, "pixel_shader",
displayio.ColorConverter()))
display_group.append(sprite)
```

Last, we use `show` to prepare the image to be displayed, and `refresh` to update the screen with the image. "`refreshed`" is printed to the serial console. Finally, we include a `180` second sleep due to the constraints of the e-ink display hardware - you should not update it more often than 180 second intervals!

```
display.show(display_group)
display.refresh()
print("refreshed")
time.sleep(180)
```

After 180 seconds, the image will stop being displayed and you will see the latest serial output displayed. That's all there is to displaying a bitmap on the Circuit Playground E-Ink Gizmo with CircuitPython!

Further Information

Be sure to check out this excellent guide to CircuitPython Display Support Using displayio ()

Python Docs

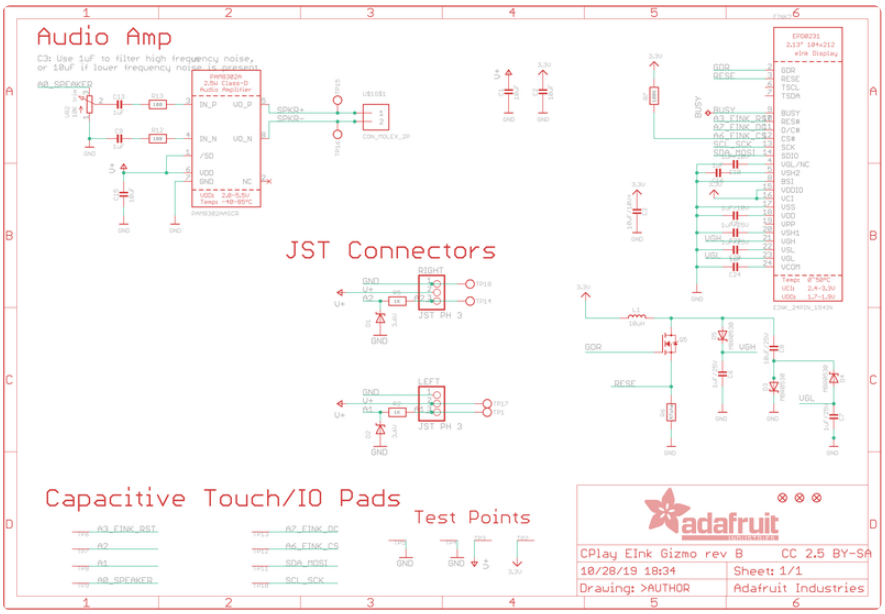
Python Docs ()

Downloads

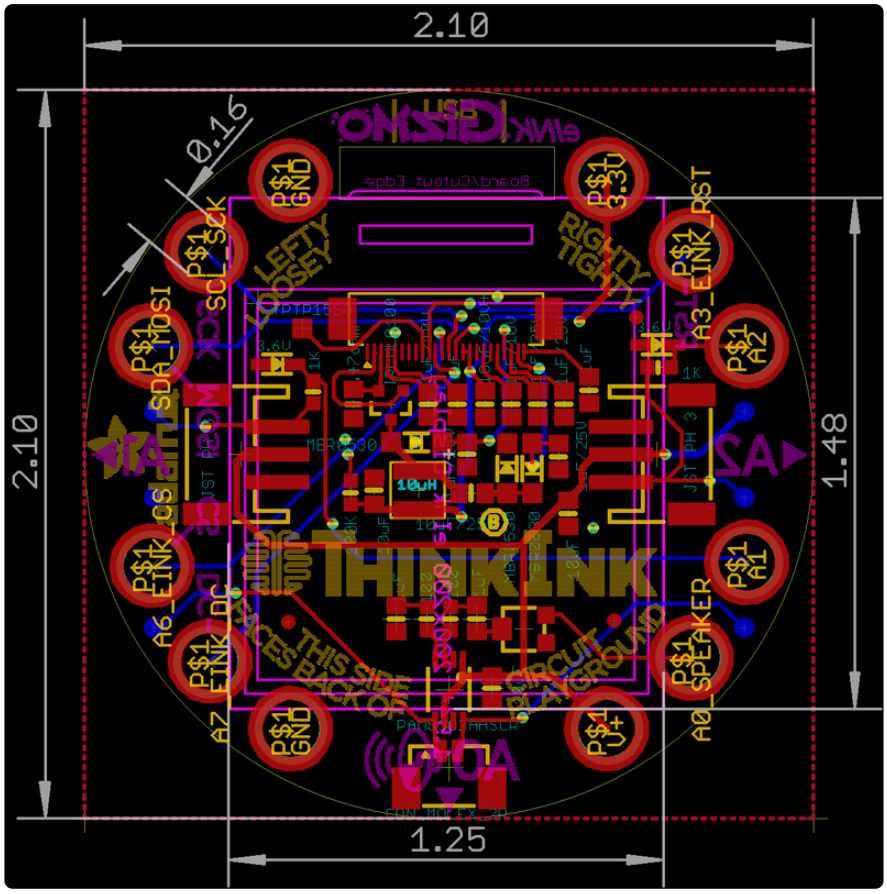
Files

- EagleCAD PCB files on GitHub ()
- Fritzing object in the Adafruit Fritzing Library ()

Schematic



Fab Print


2.10
0.16
2.10
1.48
1.25
P$1 GND
P$1 3.3V
P$1 SCK
P$1 MOSI
P$1 A3_EINK_RST
P$1 A2
P$1 A1
P$1 A0_SPEAKER
P$1 A4
P$1 A5
P$1 A6_EINK_DC
P$1 A7
P$1 GND
P$1 3.3V
THINKINK
10µH