adafruit learning system

eInk FeatherWing Display Stand

Created by Ruiz Brothers

Last updated on 2019-04-03 08:31:58 PM UTC

3D Printing

E-Ink/E-Paper Display

Make a custom name tag with Adafruit's 2.13in tri-color e-Ink FeatherWing! The 2.13in e-paper display features red, black and white. 3D print a stand and display it on your desk. Load crispy clear tricolored bitmap images easily using CircuitPython or Arduino!

3D Printed Stand

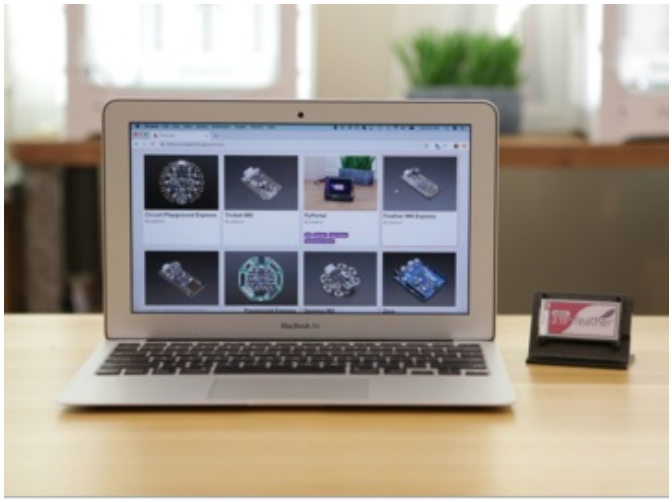
Designed to house the Adafruit 2.13in e-ink/e-Paper FeatherWing, this stand is 3D printed without any support material. Use M2.5 standoffs and fasteners to secure the FeatherWing to the 3D printed stand. Download the CAD source files to print or modify the design.

Assembly

Start by inserting four M2.5 x 6mm standoffs into the mounting holes in the 3D printed stand. Use four hex nuts to secure the standoffs. Place the FeatherWing on top of the standoffs and line up the holes. Insert and fasten four M2.5 screws into the mounting tabs on the FeatherWing.

https://adafru.it/ErS

https://adafru.it/ErS



CircuitPython Setup

Display bitmap images and text on the 2.13in eInk/epaper display FeatherWing using CircuitPython libraries and example code. Download the UF2 file for your hardware.

https://adafru.it/Em8

https://adafru.it/Em8

Prerequisite Guides

If you're new to Adafruit Feather, CircuitPython or soldering, take a moment to walk through the following guides to get you started.

- Adafruit eink epaper Displays (https://adafru.it/ErT)
- Welcome to CircuitPython (https://adafru.it/cpy-welcome)
- Adafruit Feather M4 Express (https://adafru.it/ErU)

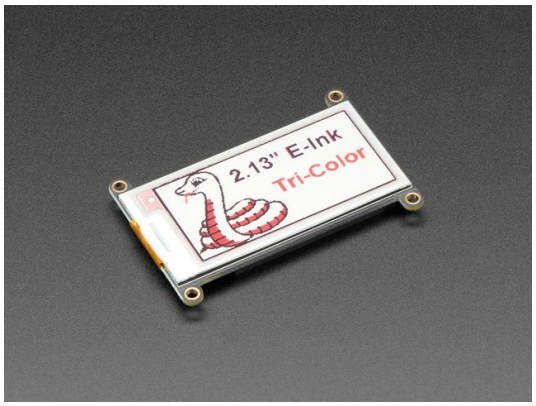
Displaying Bitmaps

Bitmap images (.BMP) can be loaded from either an SD card or the internal flash. Images must be **24-bit, 212px x 104px.**

Parts

- Adafruit 2.13in Tri-Color eInk Display FeatherWing ()
- Adafruit Feather M4 Express ()

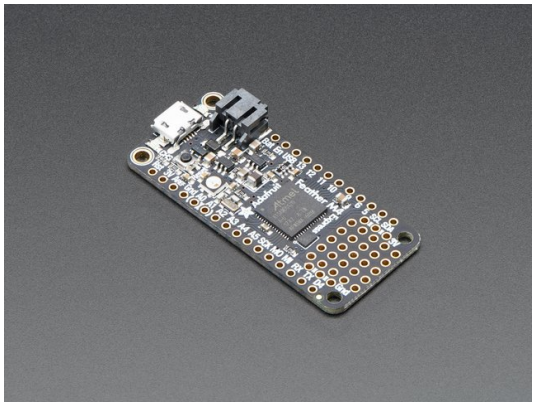
- M2.5 Black Nylon Standoff Kit ()
- Flashforge Inventor 3D Printer ()
- Filament for 3D printers ()



Adafruit 2.13" Tri-Color eInk / ePaper Display FeatherWing

$24.95
IN STOCK

ADD TO CART



Adafruit Feather M4 Express - Featuring ATSAMD51

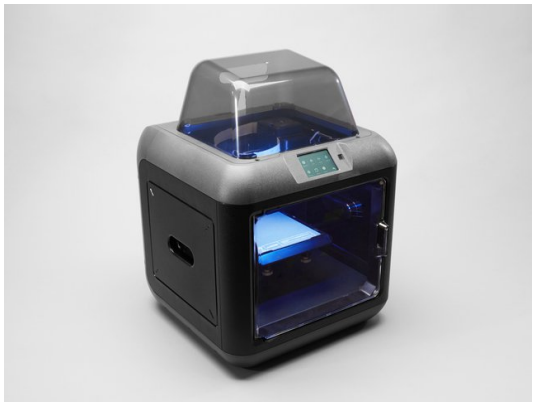
$22.95
IN STOCK

ADD TO CART

Black Nylon Screw and Stand-off Set – M2.5 Thread

$16.95
IN STOCK

ADD TO CART



Monoprice Inventor II 3D Printer with Touchscreen and WiFi

$650.00
IN STOCK

ADD TO CART

Filament for 3D Printers in Various Colors and Types

$0.00
OUT OF STOCK

OUT OF STOCK

CircuitPython Code

Library Installation

You'll need to install the Adafruit CircuitPython EPD (https://adafru.it/BTd) library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/zdx). Our introduction guide has a great page on how to install the library bundle (https://adafru.it/ABU) for both express and non-express boards.

You'll need to manually install the necessary libraries from the bundle:

- **adafruit_epd**
- **adafruit_bus_device**
- **adafruit_framebuf.mpy**

Before continuing, make sure your board's **lib** folder or root filesystem has the library files and folders listed above copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz)so you are at the CircuitPython >>> prompt.

Adafruit EPD Bitmap Example

Here's the complete example of how to display a bitmap image on your display. Note that any .bmp image you want to display must be exactly **212 pixels** by **104 pixels** and **24-bit**. We will be using the image below. Click the button below to download the image and save it as **blinka.bmp** on your **CIRCUITPY** drive.

https://adafru.it/ErW

https://adafru.it/ErW

Upload The Code

Copy and paste the code below into a new text document (we recommend using Mu (https://adafru.it/ANO) as your editor, which is designed for CircuitPython). Save the file to the **CIRCUITPY** drive and name it as **code.py.**

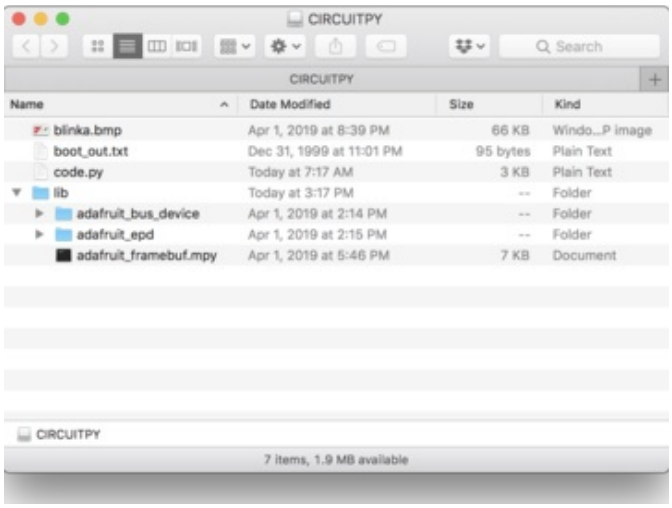
```
import digitalio
import busio
import board
from adafruit_epd.epd import Adafruit_EPD
```

```
from adafruit_epd.il0373 import Adafruit_IL0373

# create the spi device and pins we will need
spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
ecs = digitalio.DigitalInOut(board.D9)
dc = digitalio.DigitalInOut(board.D10)
srcs = None
rst = None
busy = None

# give them all to our driver
print("Creating display")
display = Adafruit_IL0373(104, 212, spi,         # 2.13" Tri-color display
                          cs_pin=ecs, dc_pin=dc, sramcs_pin=srcs,
                          rst_pin=rst, busy_pin=busy)

display.rotation = 3

FILENAME = "blinka.bmp"

def read_le(s):
    # as of this writting, int.from_bytes does not have LE support, DIY!
    result = 0
    shift = 0
    for byte in bytearray(s):
        result += byte << shift
        shift += 8
    return result

class BMPError(Exception):
    pass

def display_bitmap(epd, filename):
    # pylint: disable=too-many-locals, too-many-branches
    try:
        f = open("/" + filename, "rb")
    except OSError:
        print("Couldn't open file")
        return

    print("File opened")
    try:
        if f.read(2) != b'BM':  # check signature
            raise BMPError("Not BitMap file")

        bmpFileSize = read_le(f.read(4))
        f.read(4)  # Read & ignore creator bytes

        bmpImageoffset = read_le(f.read(4))  # Start of image data
        headerSize = read_le(f.read(4))
        bmpWidth = read_le(f.read(4))
        bmpHeight = read_le(f.read(4))
        flip = True

        print("Size: %d\nImage offset: %d\nHeader size: %d" %
              (bmpFileSize, bmpImageoffset, headerSize))
        print("Width: %d\nHeight: %d" % (bmpWidth, bmpHeight))

        if read_le(f.read(2)) != 1:
```

```
            raise BMPError("Not singleplane")
        bmpDepth = read_le(f.read(2))  # bits per pixel
        print("Bit depth: %d" % (bmpDepth))
        if bmpDepth != 24:
            raise BMPError("Not 24-bit")
        if read_le(f.read(2)) != 0:
            raise BMPError("Compressed file")

        print("Image OK! Drawing...")

        rowSize = (bmpWidth * 3 + 3) & ~3  # 32-bit line boundary

        for row in range(bmpHeight):  # For each scanline...
            if flip:  # Bitmap is stored bottom-to-top order (normal BMP)
                pos = bmpImageoffset + (bmpHeight - 1 - row) * rowSize
            else:  # Bitmap is stored top-to-bottom
                pos = bmpImageoffset + row * rowSize

            # print ("seek to %d" % pos)
            f.seek(pos)
            rowdata = f.read(3*bmpWidth)
            for col in range(bmpWidth):
                b, g, r = rowdata[3*col:3*col+3]  # BMP files store RGB in BGR
                if r < 0x80 and g < 0x80 and b < 0x80:
                    epd.pixel(col, row, Adafruit_EPD.BLACK)
                elif r >= 0x80 and g >= 0x80 and b >= 0x80:
                    pass  # epd.pixel(row, col, Adafruit_EPD.WHITE)
                elif r >= 0x80:
                    epd.pixel(col, row, Adafruit_EPD.RED)
    except OSError:
        print("Couldn't read file")
    except BMPError as e:
        print("Failed to parse BMP: " + e.args[0])
    finally:
        f.close()
    print("Finished drawing")

# clear the buffer
display.fill(Adafruit_EPD.WHITE)
display_bitmap(display, FILENAME)
display.display()
```

https://adafru.it/ANO

https://adafru.it/ANO



CIRCUITPY USB Drive

Reference the screenshot image for the files and folders that should be on your **CIRCUITPY** drive.