Halloween Countdown Display Matrix

Created by John Park

https://learn.adafruit.com/halloween-countdown-display-matrix

Last updated on 2024-06-03 03:12:19 PM EDT

Table of Contents

Overview

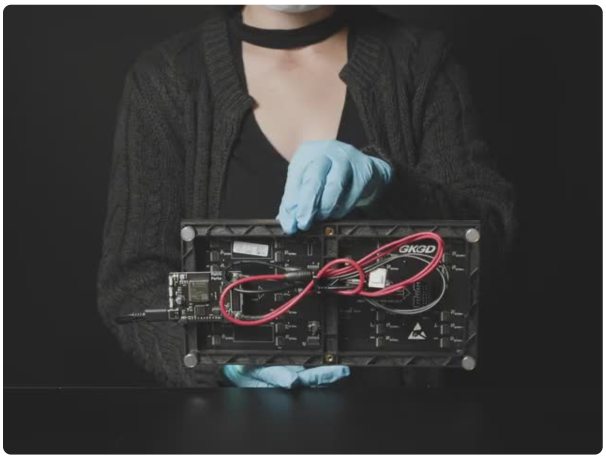
Don't lose track of the days between now and your favorite holiday -- Halloween!

This internet-connected LED matrix display running CircuitPython will tell you how many days, hours, and minutes left until Halloween! Uses Adafruit IO time server to keep in sync. And, it runs on either the Matrix Portal or a Metro M4 Airlift with RGB Matrix Shield.

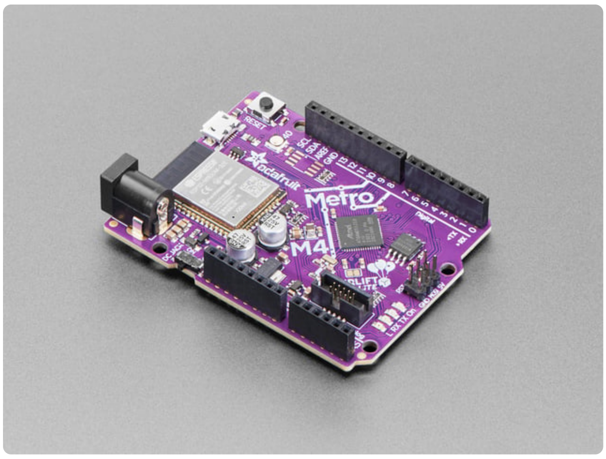
Custom bitmap images make it spooky and festive!

BUT THAT'S NOT ALL! You can adapt the code in this project to create a countdown display for ANY EVENT! Use any graphics and colors you like, too!

Parts



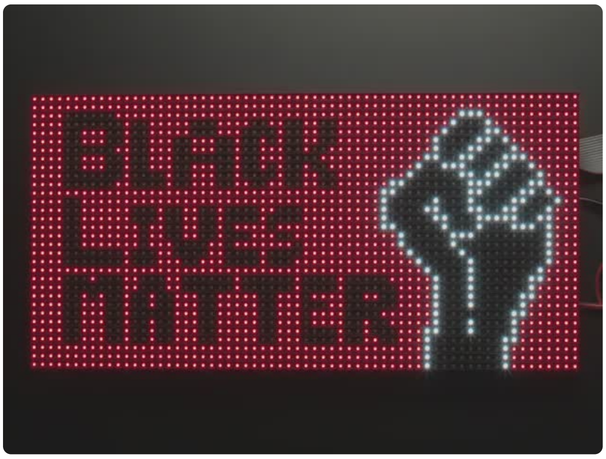
Adafruit Matrix Portal - CircuitPython Powered Internet Display
Folks love our wide selection of RGB matrices and accessories, for making custom colorful LED displays... and our RGB Matrix Shields...
https://www.adafruit.com/product/4745



Adafruit Metro M4 Express AirLift (WiFi) - Lite

Give your next project a lift with AirLift - our witty name for the ESP32 co-processor that graces this Metro M4. You already know about the Adafruit Metro...

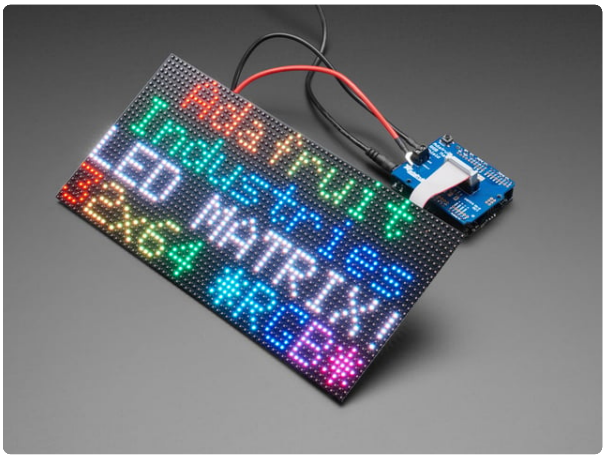
https://www.adafruit.com/product/4000



64x32 RGB LED Matrix - 4mm pitch

Bring a little bit of Times Square into your home with this sweet 64 x 32 square RGB LED matrix panel. These panels are normally used to make video walls, here in New York we see them...

https://www.adafruit.com/product/2278



Adafruit RGB Matrix Shield for Arduino

Our RGB matrices are dazzling, with their hundreds or even thousands of individual RGB LEDs. Compared to NeoPixels, they've got great density, power usage and the...

https://www.adafruit.com/product/2601

Black LED Diffusion Acrylic Panel 12" x 12" - 0.1" / 2.6mm thick

A nice whoppin' slab of some lovely black acrylic to add some extra diffusion to your LED Matrix project. This material is 2.6mm (0.1") thick and is made of special cast...

https://www.adafruit.com/product/4594

5V 2.5A Switching Power Supply with 20AWG MicroUSB Cable

Our all-in-one 5V 2.5 Amp + MicroUSB cable power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's...

https://www.adafruit.com/product/1995

USB cable - USB A to Micro-B

This here is your standard A to micro-B USB cable, for USB 1.1 or 2.0. Perfect for connecting a PC to your Metro, Feather, Raspberry Pi or other dev-board or...

https://www.adafruit.com/product/592

Micro B USB to USB C Adapter

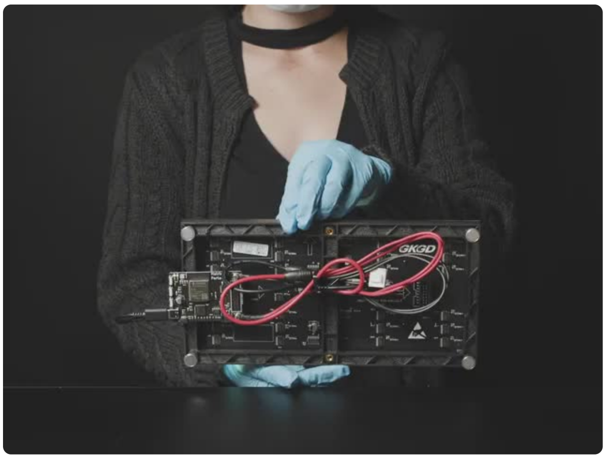
As technology changes and adapts, so does Adafruit, and speaking of adapting, this adapter has a Micro B USB jack and a USB C...

https://www.adafruit.com/product/4299

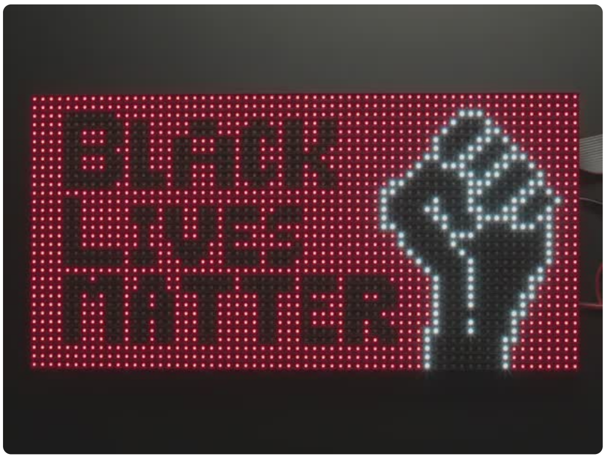
Using MatrixPortal

You can build this project with an all-in-one Matrix Portal board, its definitely the easiest and least-expensive way to go about it.

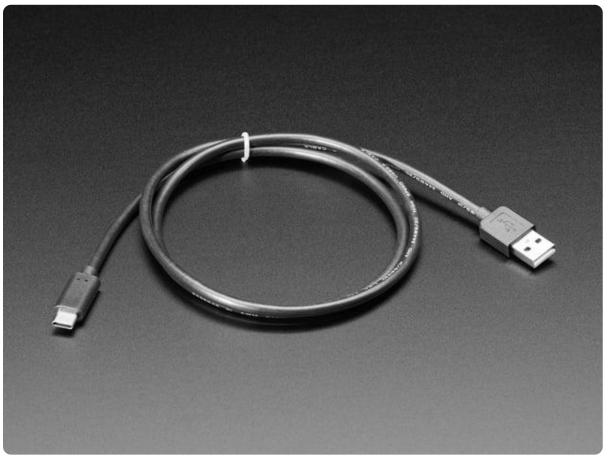
You will need a matrix portal, matrix, and USB C power/data cable



Adafruit Matrix Portal - CircuitPython Powered Internet Display
Folks love our wide selection of RGB matrices and accessories, for making custom colorful LED displays... and our RGB Matrix Shields...
https://www.adafruit.com/product/4745



64x32 RGB LED Matrix - 4mm pitch
Bring a little bit of Times Square into your home with this sweet 64 x 32 square RGB LED matrix panel. These panels are normally used to make video walls, here in New York we see them...
https://www.adafruit.com/product/2278



USB Type A to Type C Cable - approx 1 meter / 3 ft long
As technology changes and adapts, so does Adafruit. This USB Type A to Type C cable will help you with the transition to USB C, even if you're still...
https://www.adafruit.com/product/4474

Install CircuitPython

CircuitPython (https://adafru.it/tB7) is a derivative of MicroPython (https://adafru.it/BeZ) designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the **CIRCUITPY** drive to iterate.

Set up CircuitPython Quick Start!

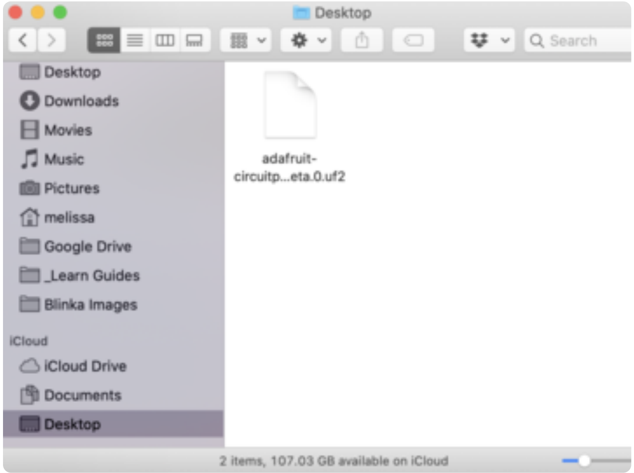
Follow this quick step-by-step for super-fast Python power :)

Download the latest version of CircuitPython for this board via circuitpython.org

https://adafru.it/Nte

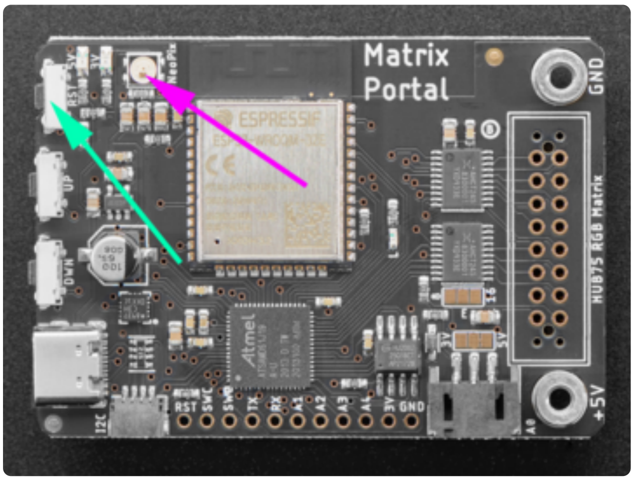
Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafru.it/Amd).



Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).

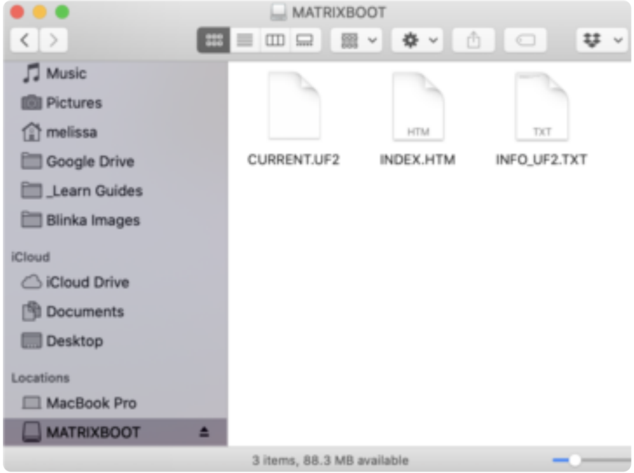


Plug your MatrixPortal M4 into your computer using a known-good USB cable.

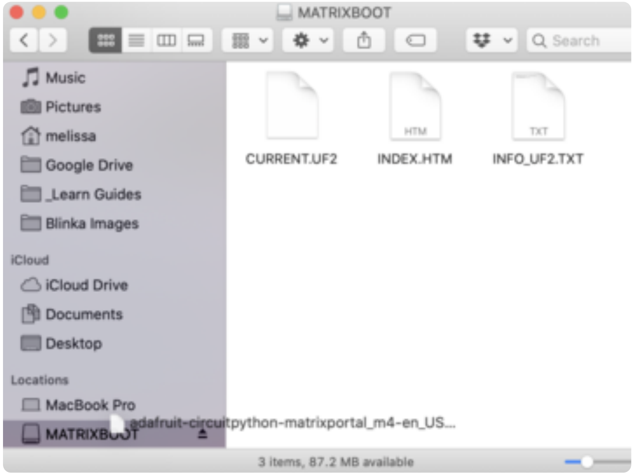
A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Double-click the **Reset** button (indicated by the green arrow) on your board, and you will see the NeoPixel RGB LED (indicated by the magenta arrow) turn green. If it turns red, check the USB cable, try another USB port, etc.

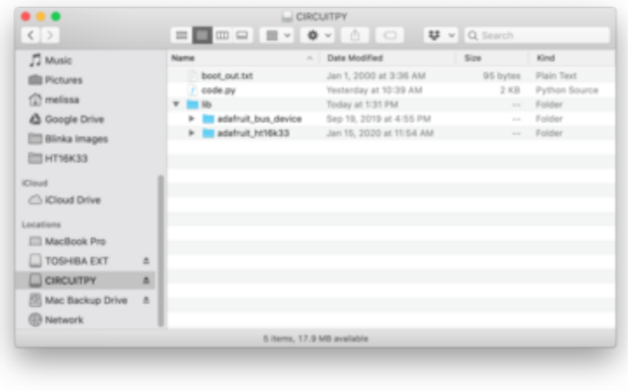
If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!



You will see a new disk drive appear called **MATRIXBOOT**.



Drag the **adafruit_circuitpython_etc.uf2** file to **MATRIXBOOT.**



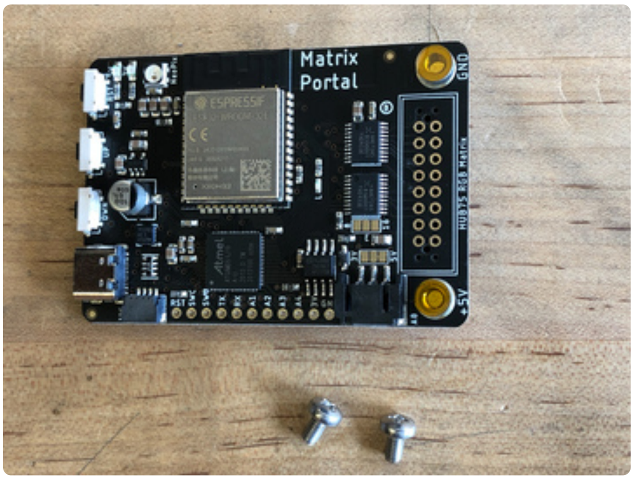
Name	Date Modified	Size	Kind
boot_out.txt	Jan 1, 2000 at 3:36 AM	95 bytes	Plain Text
code.py	Yesterday at 10:39 AM	2 KB	Python Source
lib	Today at 1:31 PM	--	Folder
adafruit_bus_device	Sep 19, 2019 at 4:55 PM	--	Folder
adafruit_ht16k33	Jan 15, 2020 at 11:54 AM	--	Folder

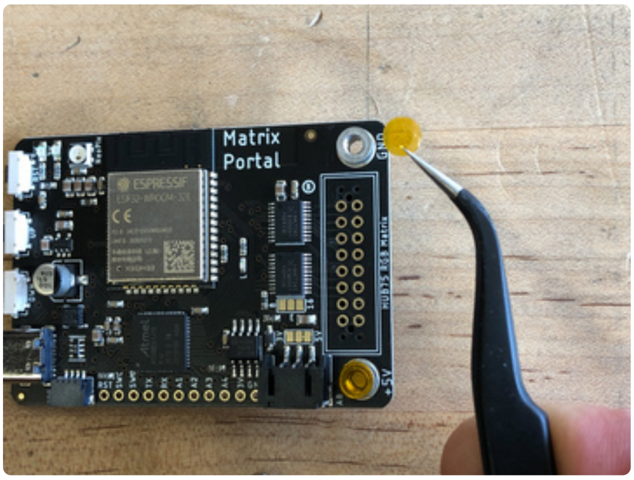
5 items, 17.9 MB available

The LED will flash. Then, the **MATRIXBOOT** drive will disappear and a new disk drive called **CIRCUITPY** will appear.

That's it, you're done! :)

Prep the MatrixPortal

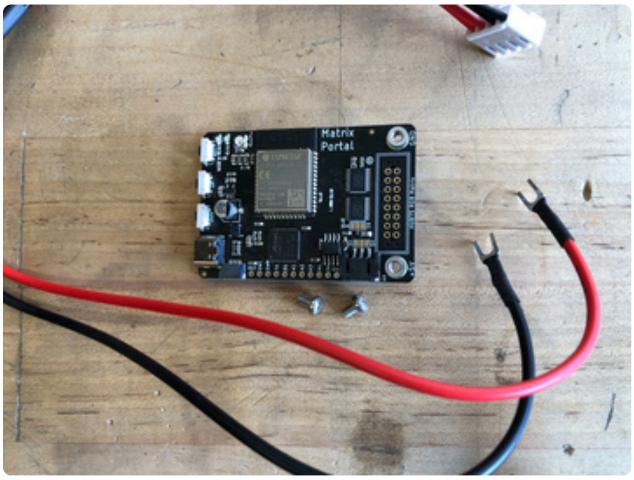


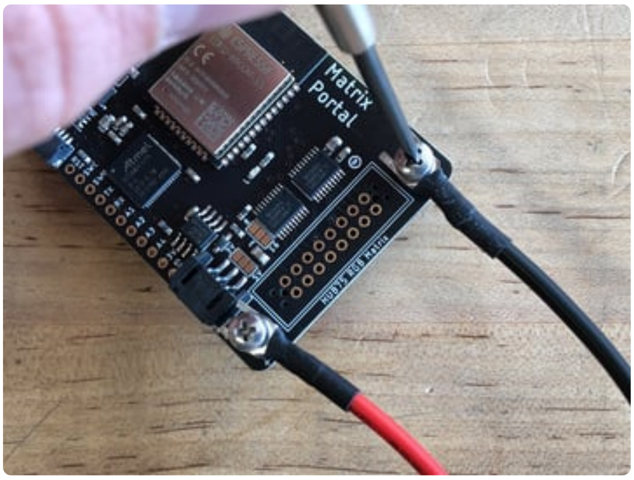


Power Prep

The MatrixPortal supplies power to the matrix display panel via two standoffs. These come with protective tape applied (part of our manufacturing process) which MUST BE REMOVED!

Use some tweezers or a fingernail to remove the two amber circles.

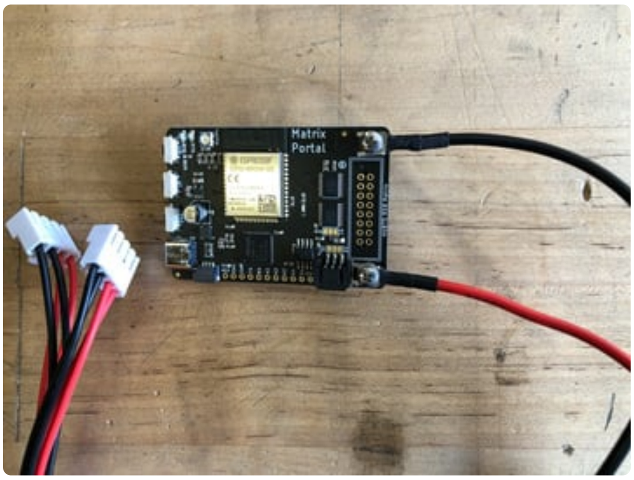


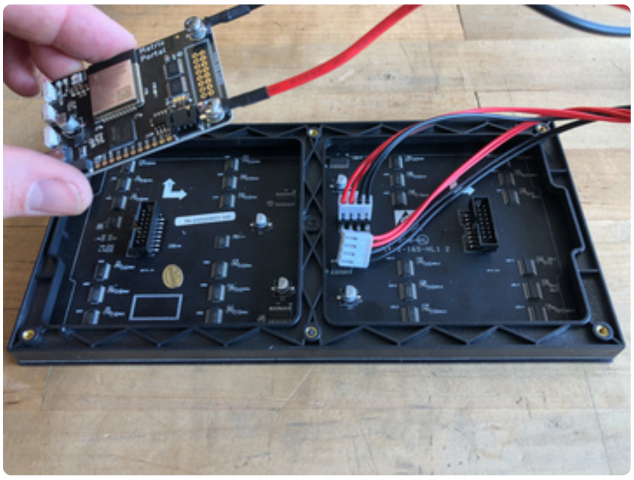


Power Terminals

Next, screw in the spade connectors to the corresponding standoff.

red wire goes to +**5V**
black wire goes to **GND**



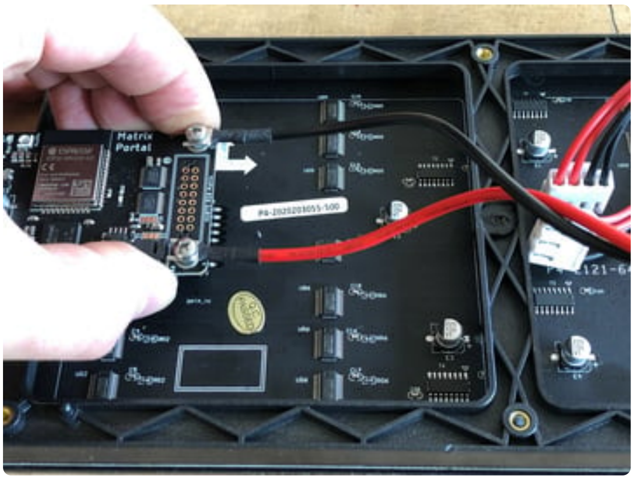


Panel Power

Plug either one of the four-conductor power plugs into the power connector pins on the panel. The plug can only go in one way, and that way is marked on the board's silkscreen.

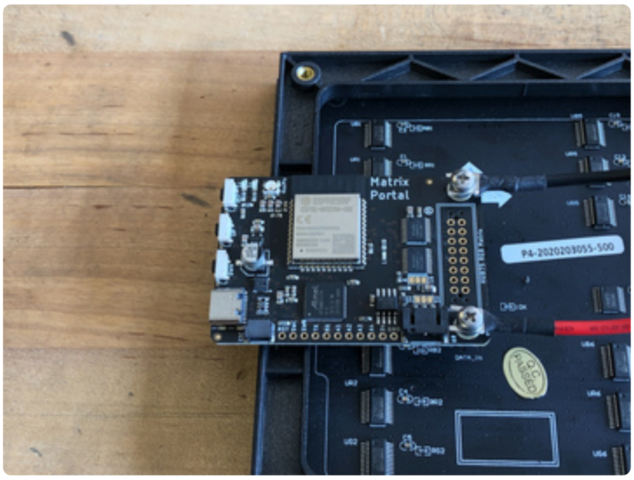
Dual Matrix Setup

If you're planning to use a 64x64 matrix, follow these instructions on soldering the Address E Line jumper (https://adafru.it/OdJ).

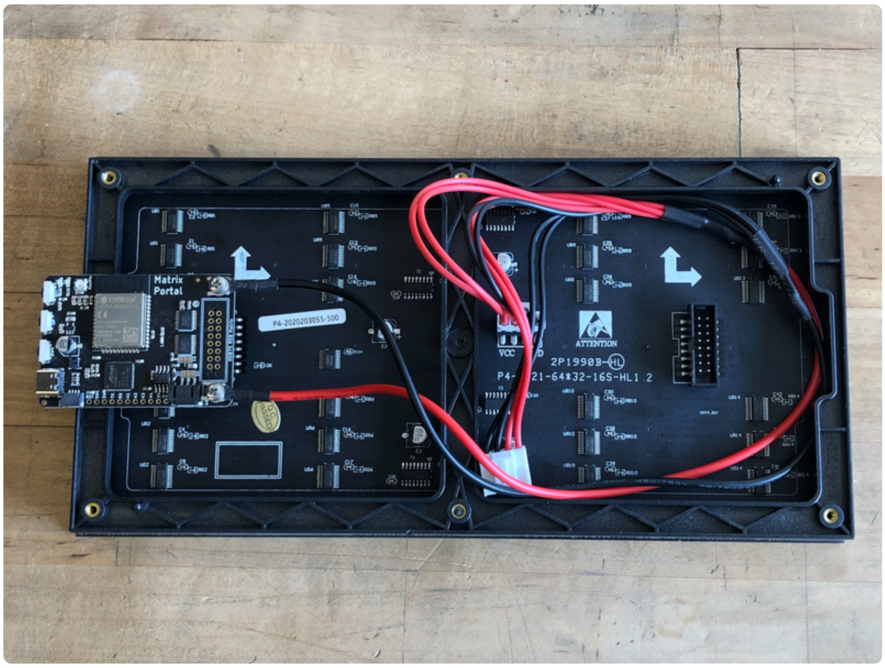


Board Connection

Now, plug the board into the left side shrouded 8x2 connector as shown. The orientation matters, so take a moment to confirm that the **white indicator arrow on the matrix panel is oriented pointing up and right** as seen here and the MatrixPortal overhangs the edge of the panel when connected. This allows you to use the edge buttons from the front side.



Check nothing is impeding the board from plugging in firmly. If there's a plastic nub on the matrix that's keeping the Portal from sitting flat, cut it off with diagonal cutters

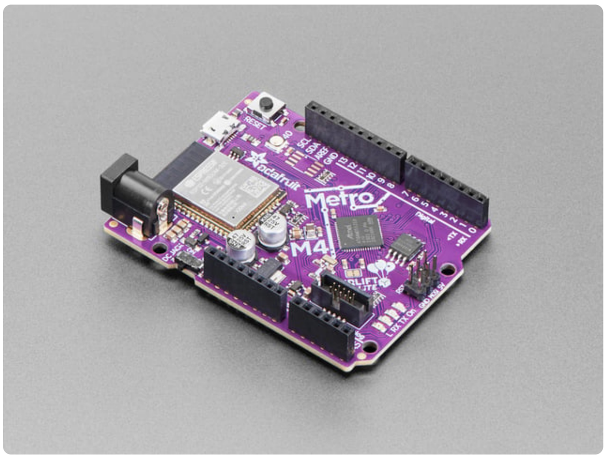


For info on adding LED diffusion acrylic, see the page LED Matrix Diffuser.

Using M4 Airlift

If you have a Metro M4 AirLift, you can build this project easily - you just need an RGB Matrix shield to help connect!

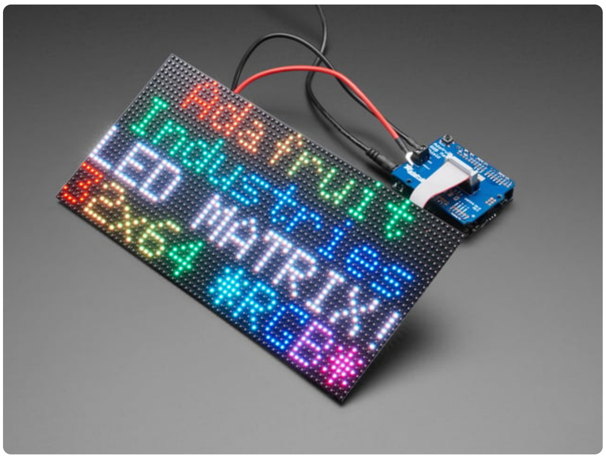
You will need a Metro M4 Airlift, matrix shield and matrix



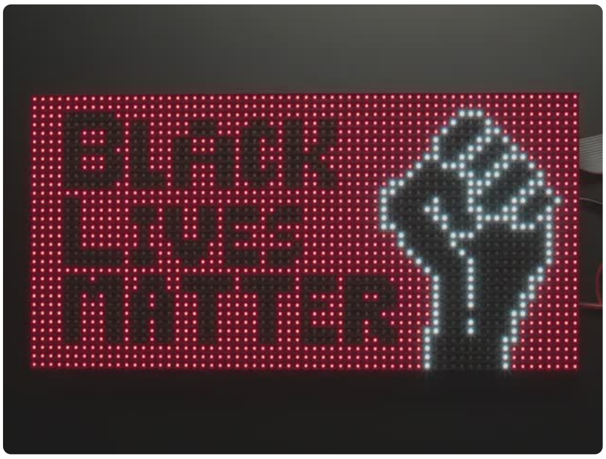
Adafruit Metro M4 Express AirLift (WiFi) - Lite

Give your next project a lift with AirLift - our witty name for the ESP32 co-processor that graces this Metro M4. You already know about the Adafruit Metro...

https://www.adafruit.com/product/4000



Adafruit RGB Matrix Shield for Arduino
Our RGB matrices are dazzling, with their hundreds or even thousands of individual RGB LEDs. Compared to NeoPixels, they've got great density, power usage and the...
https://www.adafruit.com/product/2601



64x32 RGB LED Matrix - 4mm pitch
Bring a little bit of Times Square into your home with this sweet 64 x 32 square RGB LED matrix panel. These panels are normally used to make video walls, here in New York we see them...
https://www.adafruit.com/product/2278

Install CircuitPython

CircuitPython (https://adafru.it/tB7) is a derivative of MicroPython (https://adafru.it/BeZ) designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the **CIRCUITPY** drive to iterate.

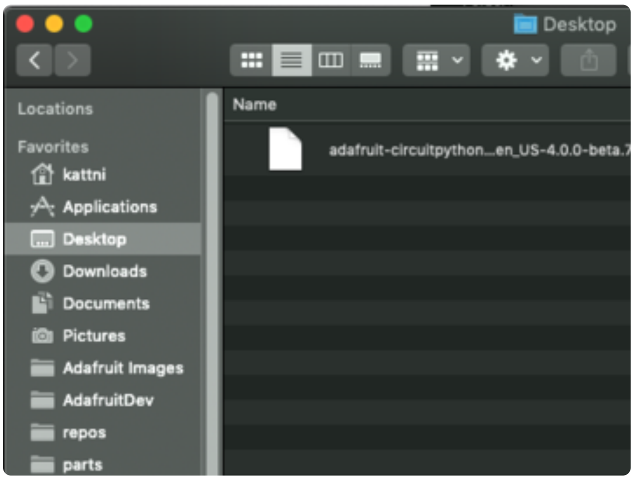
Set up CircuitPython Quick Start!

Follow this quick step-by-step for super-fast Python power :)

Download the latest version of CircuitPython for this board via circuitpython.org
https://adafru.it/EDA

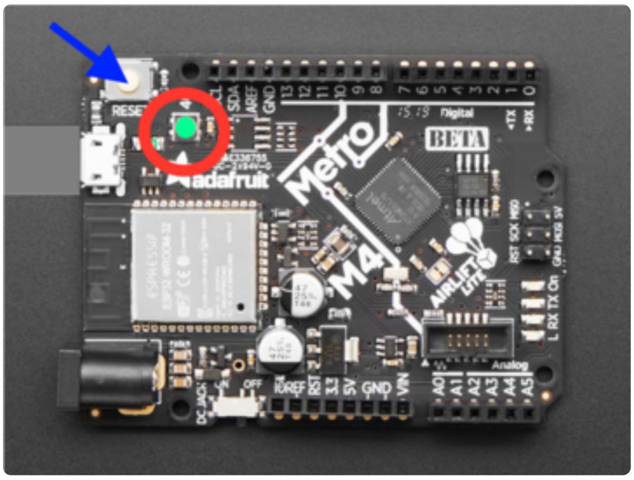
Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafru.it/Amd).



Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).

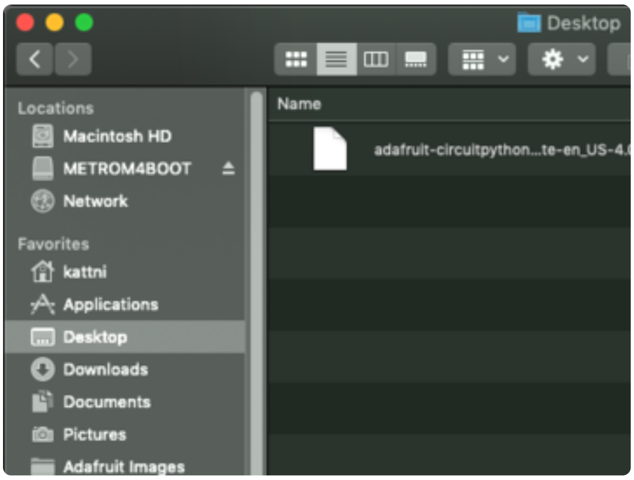


Plug your Metro M4 into your computer using a known-good USB cable.

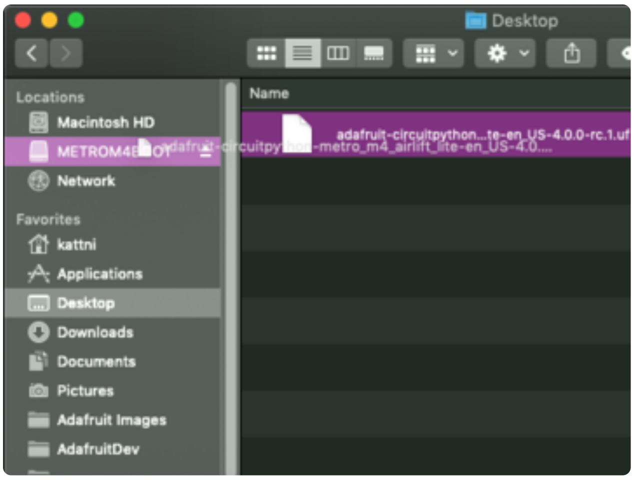
A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Double-click the **Reset** button next to the USB connector on your board (blue arrow), and you will see the NeoPixel RGB (LED circled in red) turn green. If it turns red, check the USB cable, try another USB port, etc.

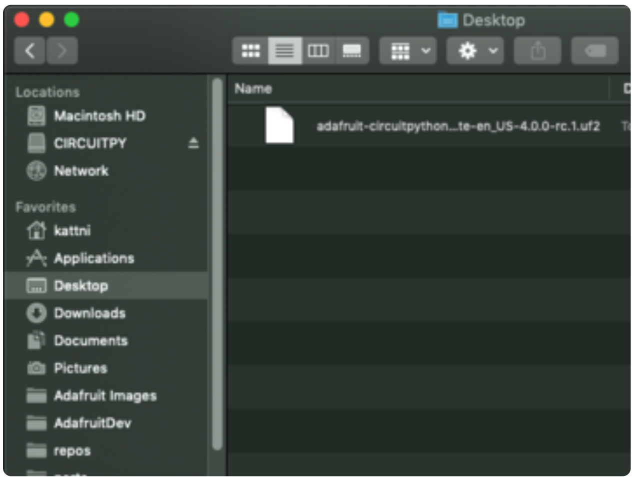
If double-clicking the Reset button doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!



You will see a new disk drive appear called **METROM4BOOT**.



Drag the **adafruit_circuitpython_etc.uf2** file to **METROM4BOOT.**



The LED will flash. Then, the **METROM4BOOT** drive will disappear and a new disk drive called **CIRCUITPY** will appear.

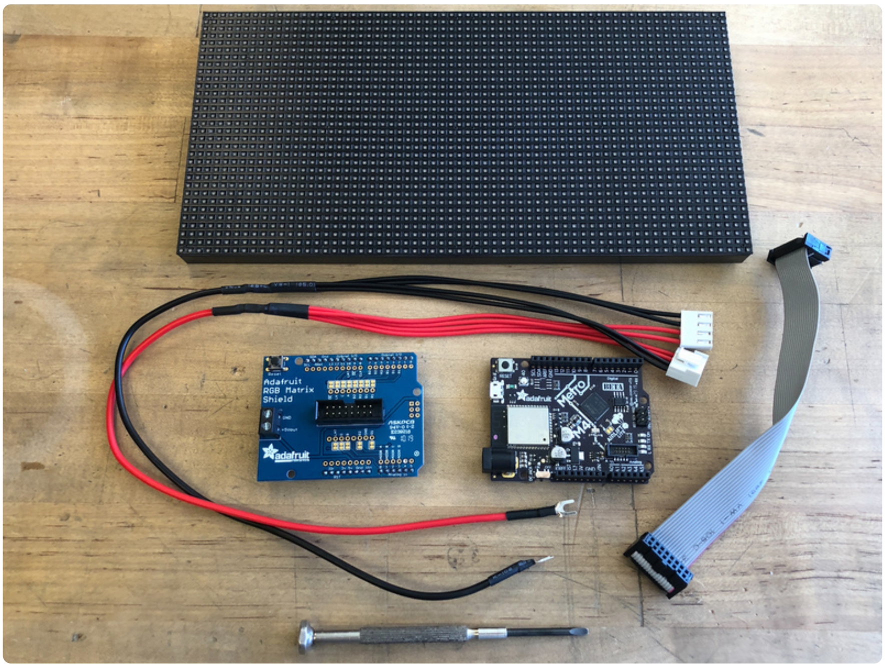
That's it, you're done! :)

Build Metro M4 Airlift Matrix Display

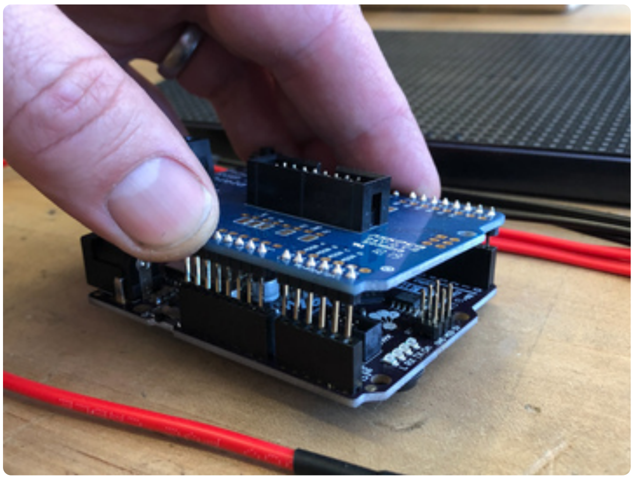
Assembly

Talking to an LED matrix display can be tricky! The 64 x 32 LED used here has a whopping 2,048 pixels, and each can display RGB colors, which makes for a whole lot of data to sling around. Thankfully, our RGB Matrix shield paired with the Metro M4 Airlift does most of the heavy lifting.

Let's assemble the boards and the display so we can get things running!



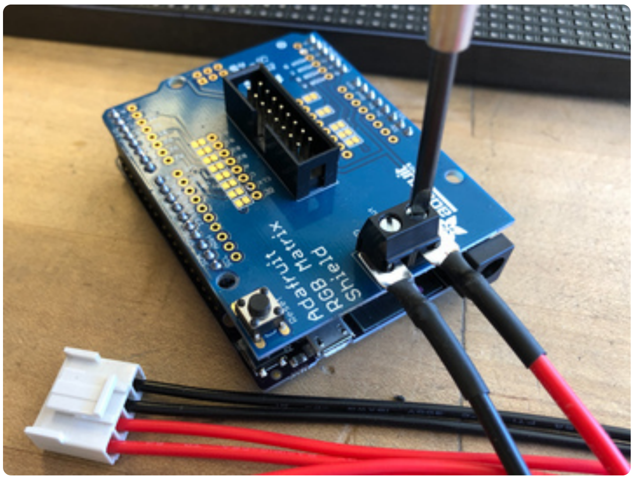
Shields Up

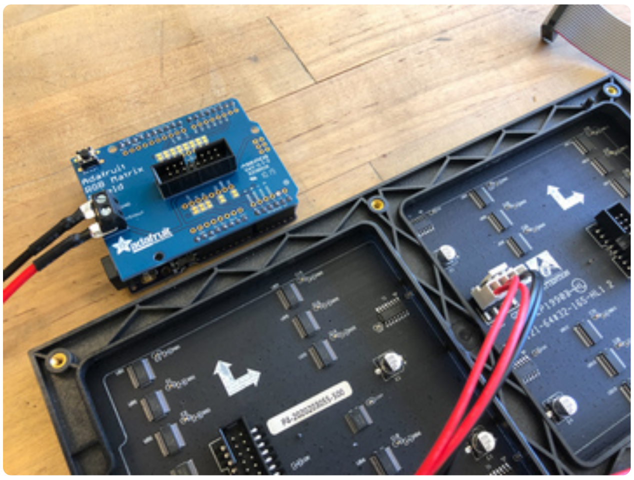


First, add the male headers, screw terminal block, and the 8x2-pin socket to the Matrix shield, by following this guide (https://adafru.it/FBU). Be careful to match the socket polarity to the silkscreen image on the board.

Be sure to also perform the clock pin mod as shown here (https://adafru.it/FBU).

Then plug the shield into the Metro M4 Airlift.

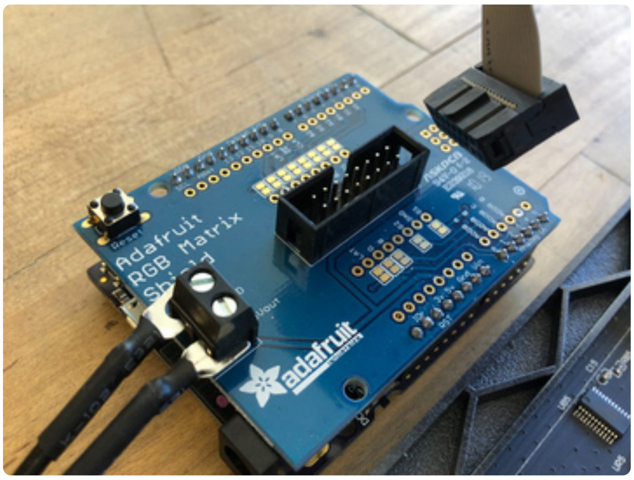


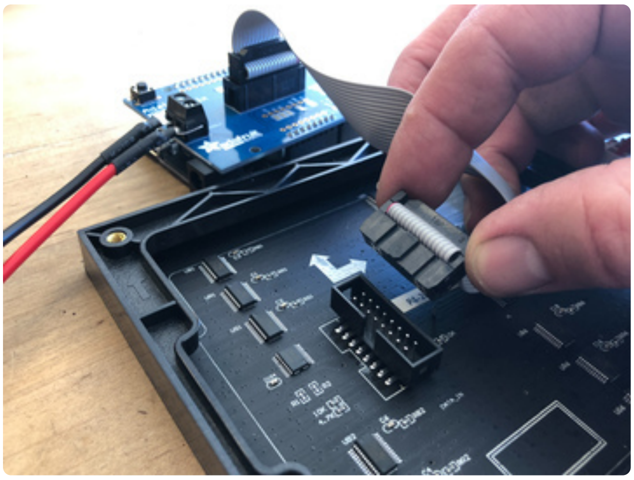


Power Connections

To provide power, we'll screw the wiring harness connectors to the screw terminal blocks of the shield. Be sure to match the **black** wire to **GND** and the **red** wire to +**5Vout.**

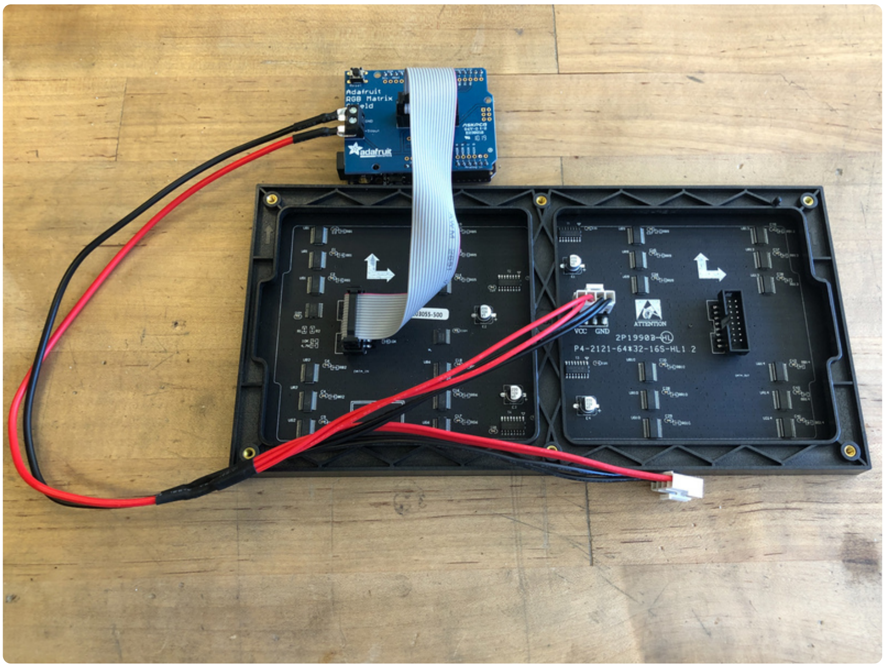
Now, simply plug the other end into the panel's power header. It can only go in one way.

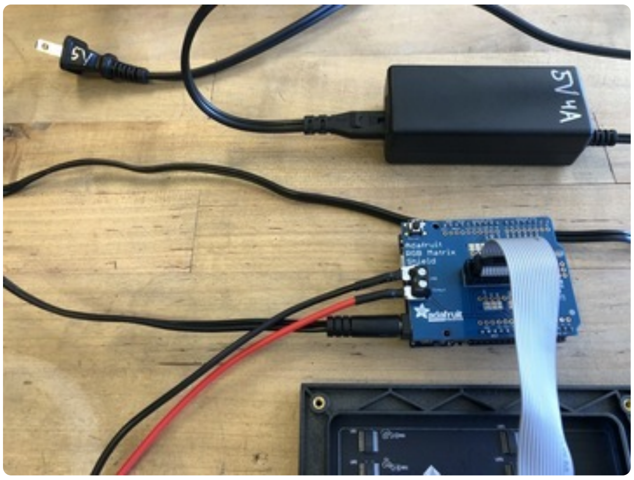


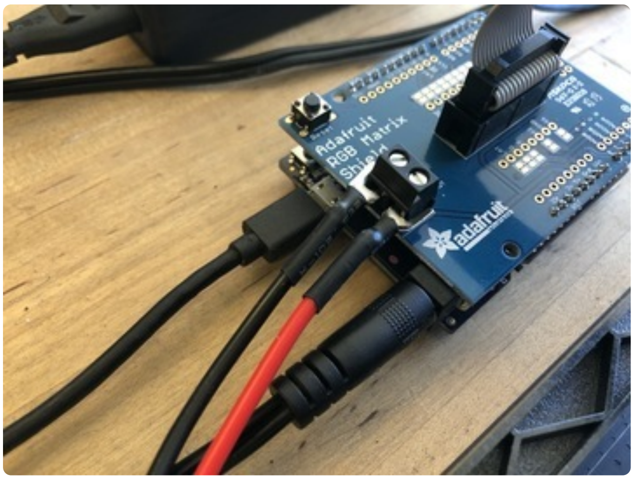


Data Cable

Plug in the two ends of the ribbon cable, note that the connectors are keyed to only fit in the correct orientation.







Wall Adapter

We'll power the Metro M4 from the 5V 2.5 (or a 4A) DC wall adapter plugged into the barrel jack. Even though USB can provide power to the board, the current isn't adequate for lighting up hundreds and thousands of LEDs!

For info on adding LED diffusion acrylic, see the page LED Matrix Diffuser.

Code the Halloween Countdown

Libraries

We'll need to make sure we have these libraries installed. (Check out this link (https://adafru.it/ABU) on installing libraries if needed.)

- **adafruit_bitmap_font**
- **adafruit_bus_device**
- **adafruit_display_shapes**
- **adafruit_display_text**
- **adafruit_esp32spi**
- **adafruit_io**
- **adafruit_matrixportal**
- **adafruit_requests.mpy**
- **neopixel.mpy**

Connect to the Internet

Once you have CircuitPython setup and libraries installed we can get your board connected to the Internet. The process for connecting can be found here (https://adafru.it/NFK).

Text Editor

Adafruit recommends using the Mu editor for editing your CircuitPython code. You can get more info in this guide (https://adafru.it/ANO).

Alternatively, you can use any text editor that saves simple text files.

Code

Click the Download: Project Zip File link below in the code window to get a zip file with all the files needed for the project. Copy **code.py** from the zip file and place on the **CIRCUITPY** drive.

You'll also need to copy the following files to the **CIRCUITPY** drive. See the graphic at the top of the page as to filenames and where they go):

- **fonts** directory, which contains a .bdf font
- **bmp** directory, which contains the graphics

- **settings.toml** (after you edit to put your WiFi and AIO credentials in the file)

```
# SPDX-FileCopyrightText: 2020 John Park for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time
import board
from adafruit_matrixportal.matrixportal import MatrixPortal

EVENT_YEAR = 2021
EVENT_MONTH = 10
EVENT_DAY = 31
EVENT_HOUR = 17
EVENT_MINUTE = 0

FRAME_DURATION = 3
FRAMES = (
    "bmps/jack.bmp",
    "DAYS",
    "bmps/ghost.bmp",
    "HOURS",
    "bmps/bats.bmp",
    "MINUTES",
    "bmps/skull.bmp",
    "bmps/halloween.bmp",
)

EVENT_DAY_IMAGE = "bmps/happy_halloween.bmp"
SYNCHRONIZE_CLOCK = True

# --- Display setup ---
matrixportal = MatrixPortal(status_neopixel=board.NEOPIXEL, debug=True)

current_frame = None

# Create a new label with the color and text selected
matrixportal.add_text(
    text_font="fonts/Arial-12.bdf",
    text_position=(4, (matrixportal.graphics.display.height // 2) - 1),
    text_color=0xEF7F31,
)


def set_time_until(unit=None):
    event_time = time.struct_time(
        (
            EVENT_YEAR,
            EVENT_MONTH,
            EVENT_DAY,
            EVENT_HOUR,
            EVENT_MINUTE,
            0,  # we don't track seconds
            -1,
            -1,
            False,
        )
    )
    remaining = time.mktime(event_time) - time.mktime(time.localtime())
    if remaining <= 0:
        # oh, its event time!
        matrixportal.set_background(EVENT_DAY_IMAGE)
        return
    remaining //= 60
    mins_remaining = remaining % 60
    remaining //= 60
    hours_remaining = remaining % 24
    remaining //= 24
    days_remaining = remaining

    if unit == "DAYS":
        text = "{} day".format(days_remaining)
        if days_remaining != 1:
            text += "s"
    if unit == "HOURS":
        text = "{} hour".format(hours_remaining)
        if hours_remaining != 1:
            text += "s"
    if unit == "MINUTES":
        text = "{} min".format(mins_remaining)
        if mins_remaining != 1:
            text += "s"
    matrixportal.set_text(text)
    matrixportal.set_background(0)


def set_next_frame():
    # pylint: disable=global-statement
    global current_frame

    # Advance to next frame if we already have one
    if current_frame is not None:
        current_frame += 1

    # Loop back or set initial frame
    if current_frame is None or current_frame >= len(FRAMES):
        current_frame = 0

    # Check if Picture or Text
    print(FRAMES[current_frame])
    if FRAMES[current_frame][-4:] == ".bmp":
        matrixportal.set_background(FRAMES[current_frame])
        matrixportal.set_text("")
    else:
        set_time_until(FRAMES[current_frame])


# Simulate the delay in case fetching time is fast
set_next_frame()
start_time = time.monotonic()
if SYNCHRONIZE_CLOCK:
    matrixportal.get_local_time()
while time.monotonic() < start_time + FRAME_DURATION:
    pass

while True:
    set_next_frame()
    time.sleep(FRAME_DURATION)
```

Adafruit IO Time Server

In order to get the precise time, our project will query the Adafruit IO Internet of Things service for the time. Adafruit IO is absolutely free to use, but you'll need to log in with your Adafruit account to use it. If you don't already have an Adafruit login, create one here (https://adafru.it/dAQ).

If you haven't used Adafruit IO before, check out this guide for more info (https://adafru.it/Ef8).

Once you have logged into your account, there are two pieces of information you'll need to place in your **settings.toml** file: **Adafruit IO username**, and **Adafruit IO key**. Head to io.adafruit.com (https://adafru.it/fsU) and simply click the **View AIO Key** link on the left hand side of the Adafruit IO page to get this information.

More on settings.toml here. (https://adafru.it/18f9)

How it Works

Libraries

Here's how the code works. First we import the `time`, board, and `adafruit_matrixportal` libraries.

Event Time

Then, we'll set the variables for the event year, month, day, hour, and minute. These will be used to calculate the countdown remaining based on the current time.

```
import time
import board
from adafruit_matrixportal.matrixportal import MatrixPortal

EVENT_YEAR = 2020
EVENT_MONTH = 10
EVENT_DAY = 31
EVENT_HOUR = 17
EVENT_MINUTE = 0
```

Frames

Next, we set the `FRAME_DURATION = 3` which means each image or text screen will hold for three seconds. You can adjust this as you like.

Then, we'll define the order in which the images and text will be displayed. This is pretty cool, as it is essentially making a list of which items you want to see appear sequentially. You can mix and match this as you see fit!

```
FRAME_DURATION = 3
FRAMES = (
    "bmps/jack.bmp",
    "DAYS",
    "bmps/ghost.bmp",
    "HOURS",
    "bmps/bats.bmp",
    "MINUTES",
    "bmps/skull.bmp",
    "bmps/halloween.bmp",
)
```

We'll also set one image aside as the `EVENT_DAY_IMAGE = "bmps/happy_halloween.bmp"`.

UNTIL
HALLOWEEN
HAPPY
HALLOWEEN

MatrixPortal Setup

Next, we'll set up the matrixportal object for the display, and then create a text label.

```
matrixportal = MatrixPortal(status_neopixel=board.NEOPIXEL, debug=True)

current_frame = None

# Create a new label with the color and text selected
matrixportal.add_text(
    text_font="fonts/Arial-12.bdf",
    text_position=(4, (matrixportal.graphics.display.height // 2) - 1),
    text_color=0xEF7F31,
)
```

Set Time Until Function

This function is used to derive the time until the event based up one the start variables and the current time.

```
def set_time_until(unit=None):
    event_time = time.struct_time(
        (
            EVENT_YEAR,
            EVENT_MONTH,
            EVENT_DAY,
            EVENT_HOUR,
            EVENT_MINUTE,
            0,  # we don't track seconds
            -1,
            -1,
            False,
        )
    )
    remaining = time.mktime(event_time) - time.mktime(time.localtime())
    if remaining &lt;= 0:
        # oh, its event time!
        matrixportal.set_background(EVENT_DAY_IMAGE)
        return
    # secs_remaining = remaining % 60
    remaining //= 60
    mins_remaining = remaining % 60
    remaining //= 60
    hours_remaining = remaining % 24
    remaining //= 24
    days_remaining = remaining

    if unit == "DAYS":
        text = "{} day".format(days_remaining)
        if days_remaining != 1:
            text += "s"
    if unit == "HOURS":
        text = "{} hour".format(hours_remaining)
        if hours_remaining != 1:
            text += "s"
    if unit == "MINUTES":
        text = "{} min".format(mins_remaining)
        if mins_remaining != 1:
            text += "s"
    matrixportal.set_text(text)
    matrixportal.set_background(0)
```

Set Next Frame Function

We'll use this function to set the next frame to either a bitmap graphic or a text label depending on where we are in the list order.

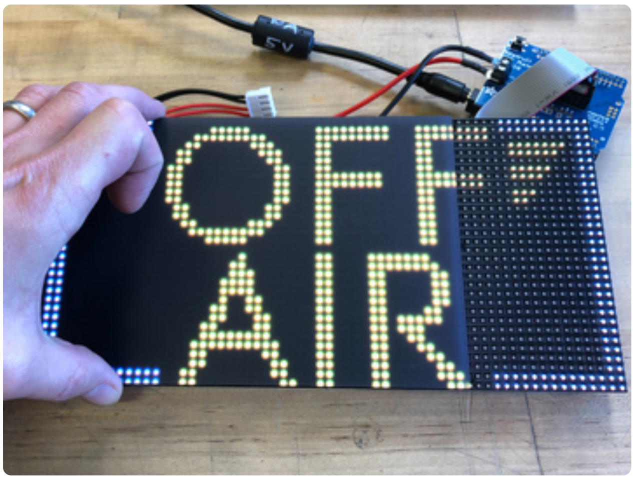
```
ef set_next_frame():
    global current_frame

    # Advance to next frame if we already have one
    if current_frame is not None:
        current_frame += 1

    # Loop back or set initial frame
    if current_frame is None or current_frame &gt;= len(FRAMES):
        current_frame = 0

    # Check if Picture or Text
    print(FRAMES[current_frame])
    if FRAMES[current_frame][-4:] == ".bmp":
        matrixportal.set_background(FRAMES[current_frame])
        matrixportal.set_text("")
    else:
        set_time_until(FRAMES[current_frame])
```

Main Loop

The main loop of the program is simple -- it calls the `set_next_frame()` function and then pauses for the frame duration!

```
while True:
    set_next_frame()
    time.sleep(FRAME_DURATION)
```

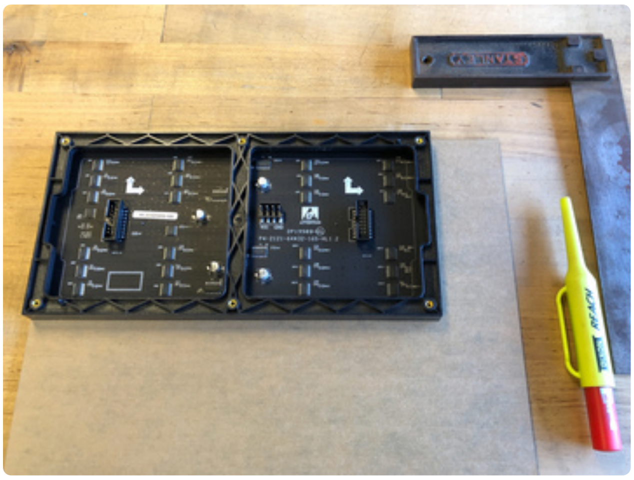
LED Matrix Diffuser

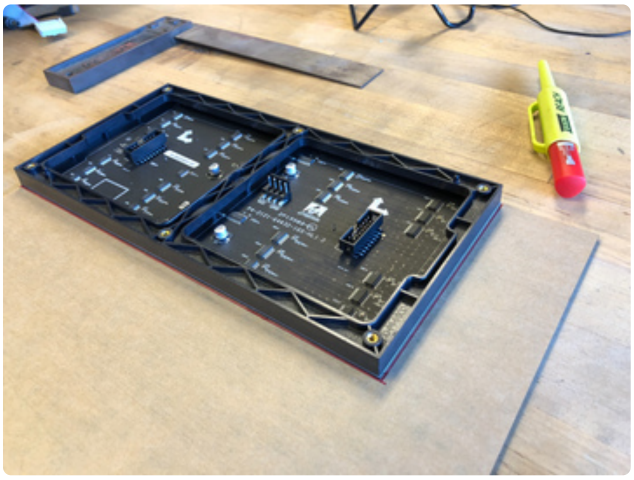


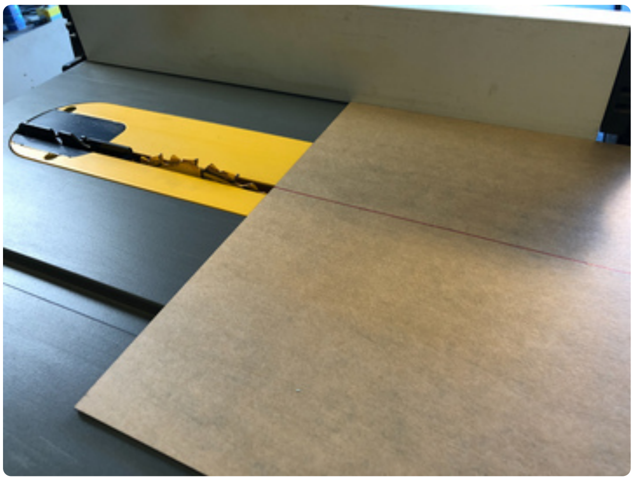
LED Diffusion Acrylic

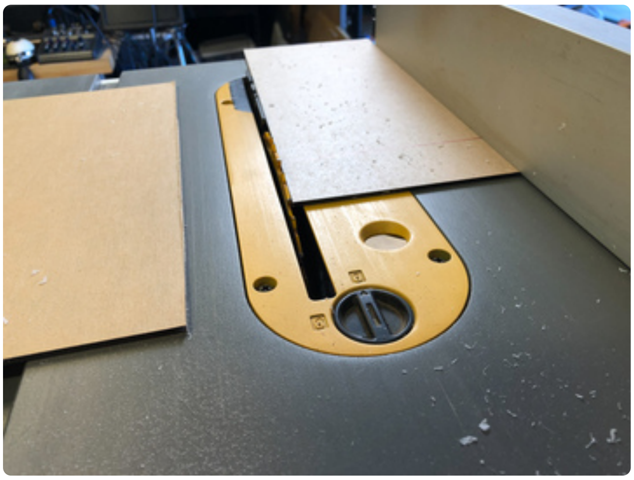
You can add an LED diffusion acrylic faceplate (http://adafru.it/4594) to the your LED matrix display. (Pictured here with the ON AIR project (https://adafru.it/MPE))

This can help protect the LEDs as well as enhance the look of the sign both indoors and out by reducing glare and specular highlights of the plastic matrix grid.







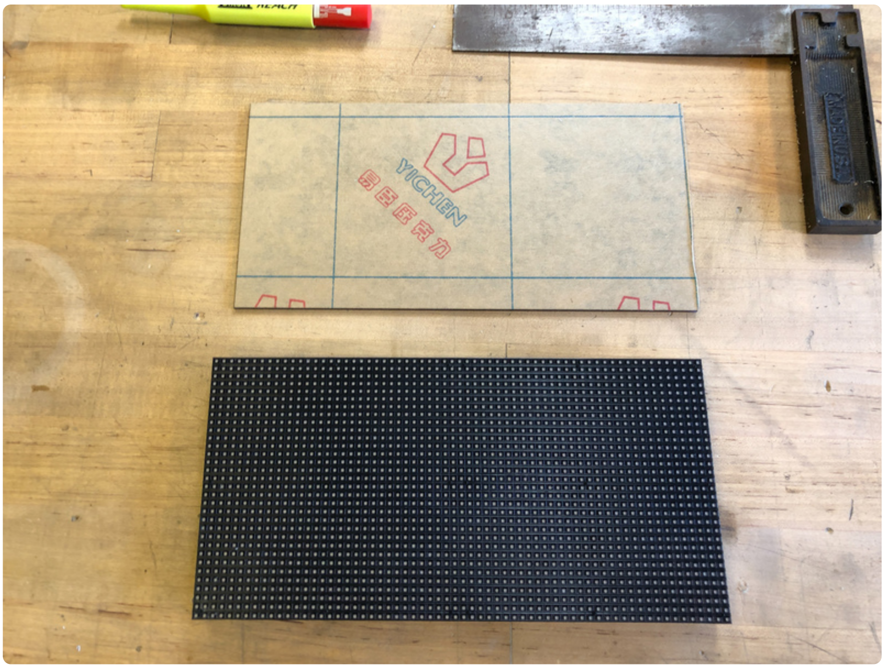


Measure and Cut the Plastic

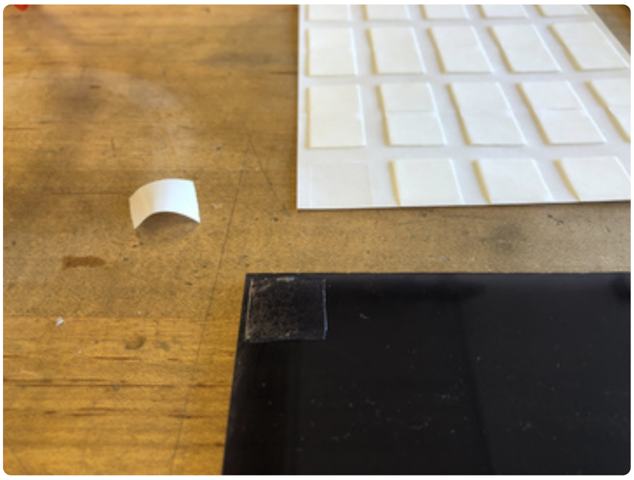
You can use the sign to measure and mark cut lines on the paper backing of the acrylic sheet.

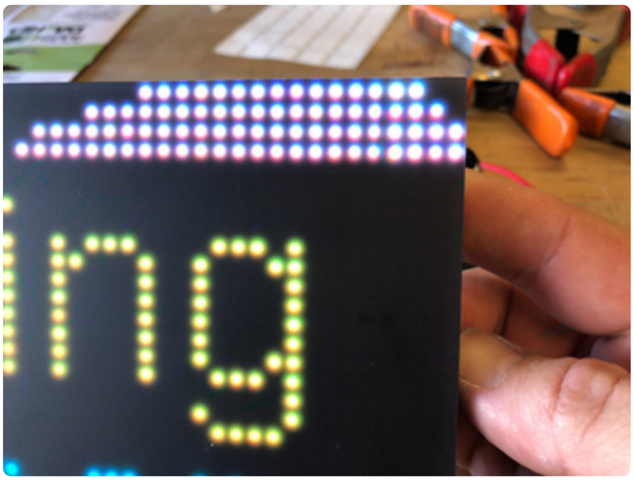
Then, use a tablesaw or bandsaw with a fine toothed blade and a guide or sled to make the cuts.

Note: it is possible to score and snap acrylic, but it can be very tricky to get an even snap without proper clamping.



Peel away the paper backing from both sides and set the acrylic onto your matrix display with the matte finished side facing out.



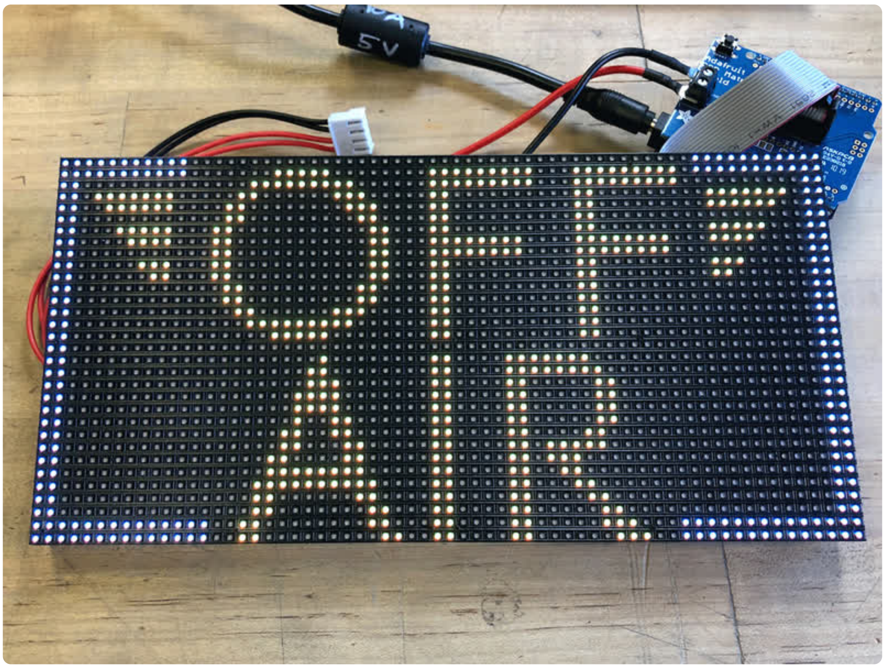


Uglu Dashes

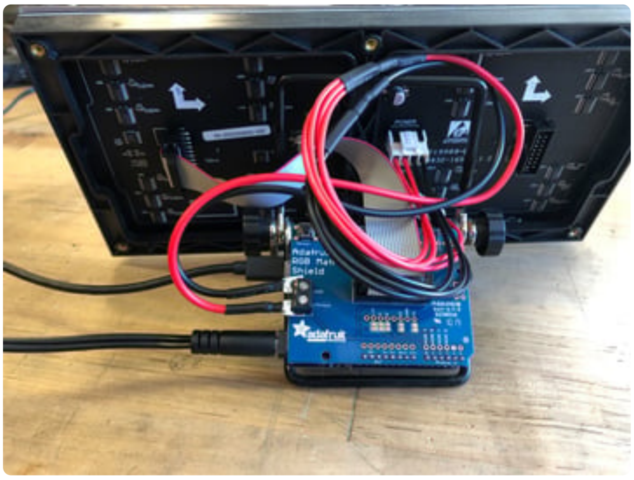
The best method we've found for adhering acrylic to the matrix display is to use Uglu Dashes clear adhesive rectangles from Pro Tapes (https://adafru.it/NcP). They are incredibly strong (although can be removed if necessary), easy to apply, and are invisible once attached.

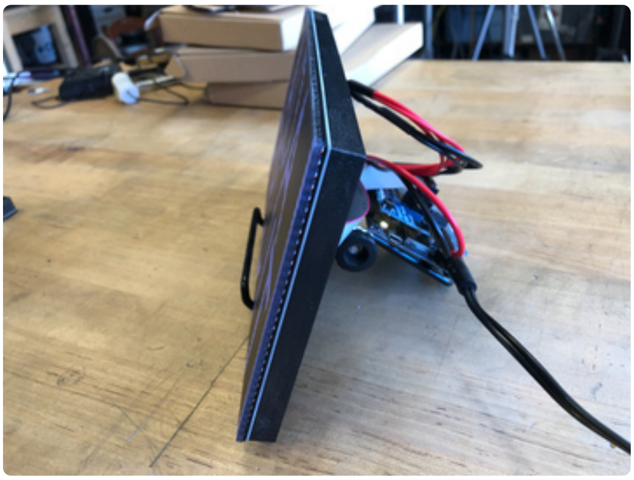
Use one at each corner and one each at the halfway point of the long edges, then press the acrylic and matrix panel together for about 20 seconds.

Here you can see the impact of using the diffusion acrylic. (Pictured here with the ON AIR sign project)



5V
OFF
AIR





Stand

A very simple and attractive way to display your matrix is with the adjustable bent-wire stand (http://adafru.it/1679).

Alternately, you can use a frame, 3D printed brackets (https://adafru.it/MZf), tape, glue, or even large binder clips to secure the acrylic to the sign and then mount it on on a wall, shelf, or display cabinet.

These mini-magnet feet (http://adafru.it/4631) can be used to stick the sign to a ferrous surface.