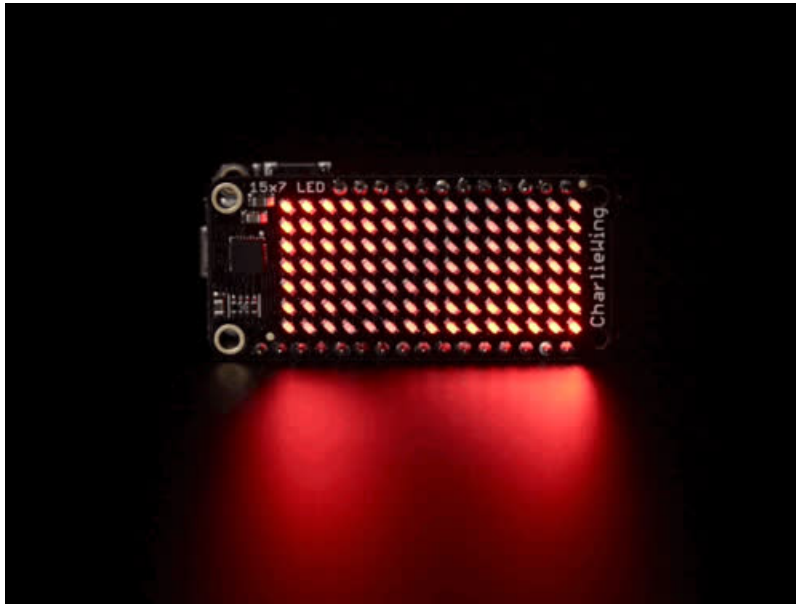




Adafruit 15x7 CharliePlex FeatherWing

Created by lady ada

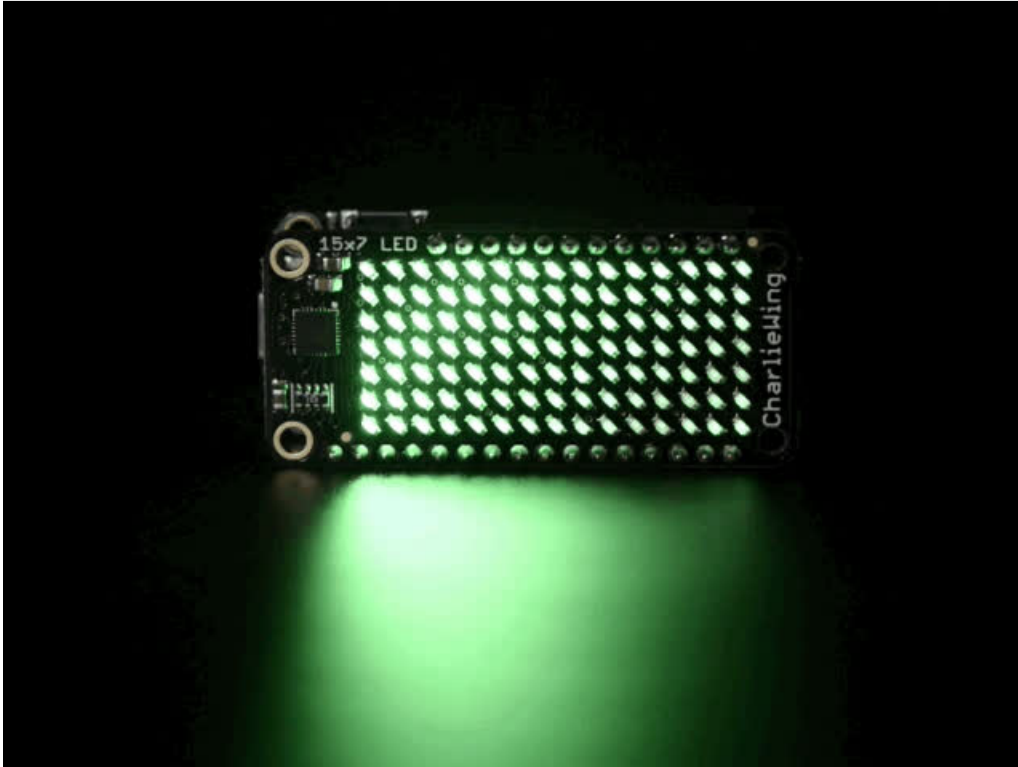


Last updated on 2018-08-22 03:54:03 PM UTC

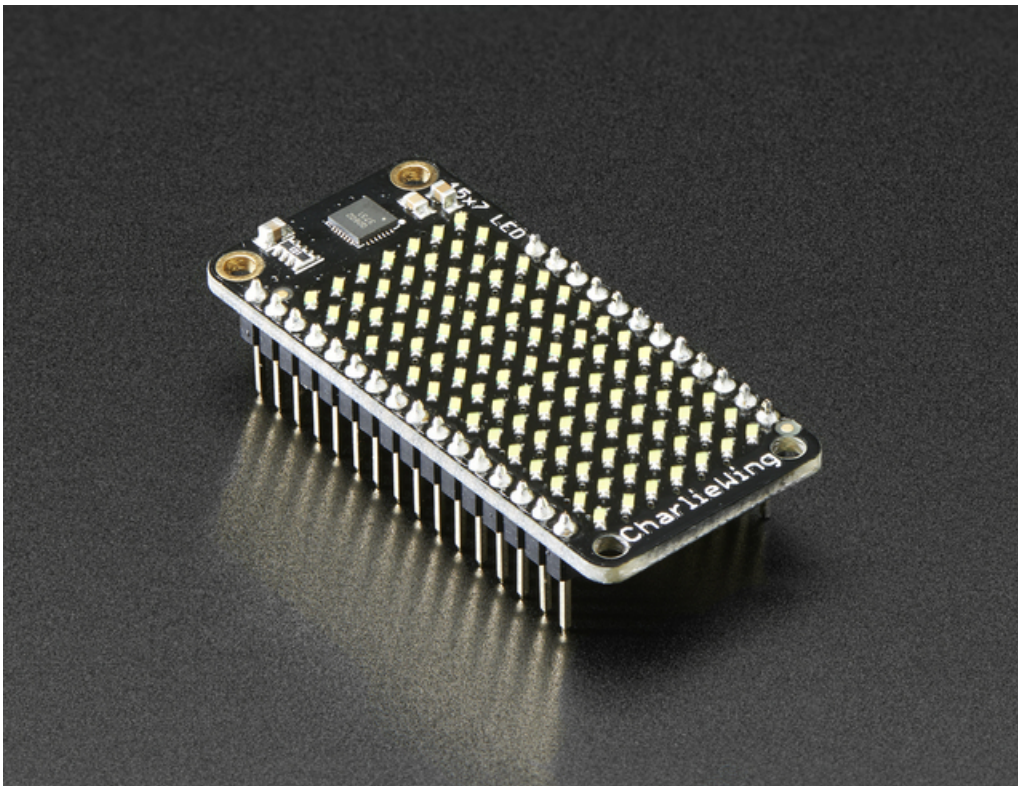
Guide Contents

Guide Contents	2
Overview	3
Pinouts	6
Power Pins	6
I2C pins	6
Usage	8
Download Adafruit_IS31FL3731 library	8
Install Adafruit GFX	8
Load Demo	8
Library Reference	10
Initialize	10
Drawing	10
Adafruit GFX	10
Multiple Buffers	10
Downloads	12
Datasheets & Files	12
Schematic	12
Fabrication Print	12

Overview

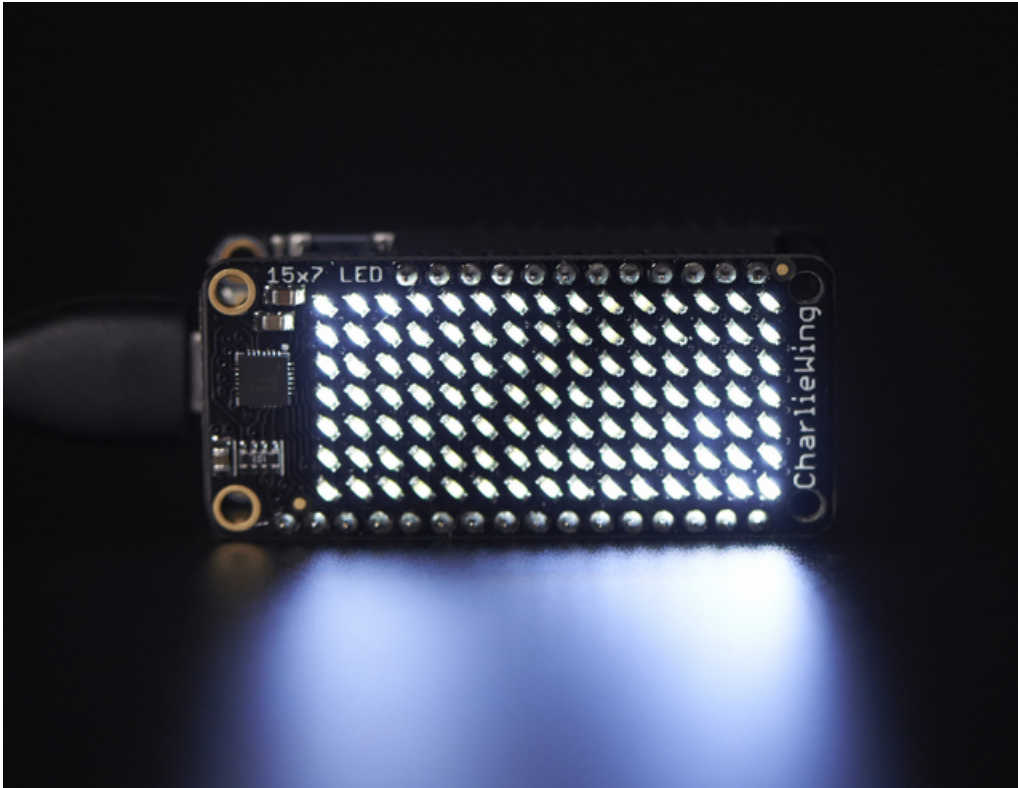


You won't be able to look away from the mesmerizing patterns created by this **Adafruit 15x7 CharliePlex LED Matrix Display FeatherWing**. This 15x7 LED display can be paired with any of our [Feather boards \(https://adafruit.it/nEt\)](https://adafruit.it/nEt) for a beautiful, bright grid of 105 charlieplexed LEDs. It even comes with a built-in charlieplex driver that is run over I2C.

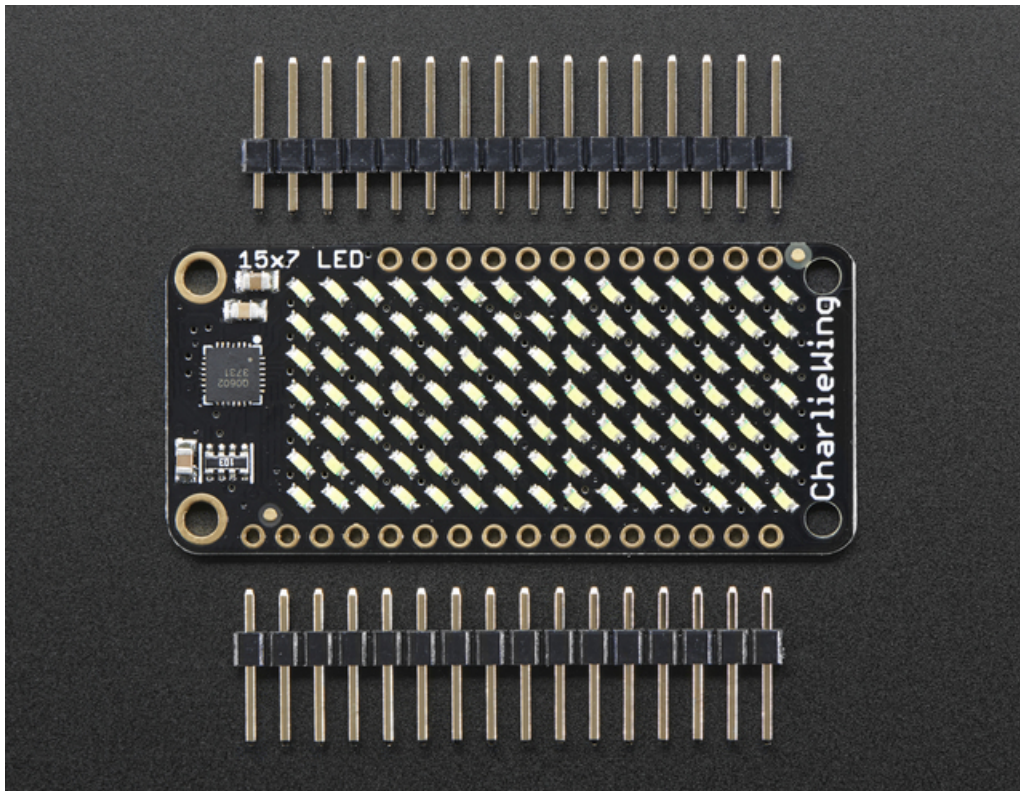


We carry these FeatherWings in [five vivid colors](http://adafru.it/297320) (<http://adafru.it/297320>)

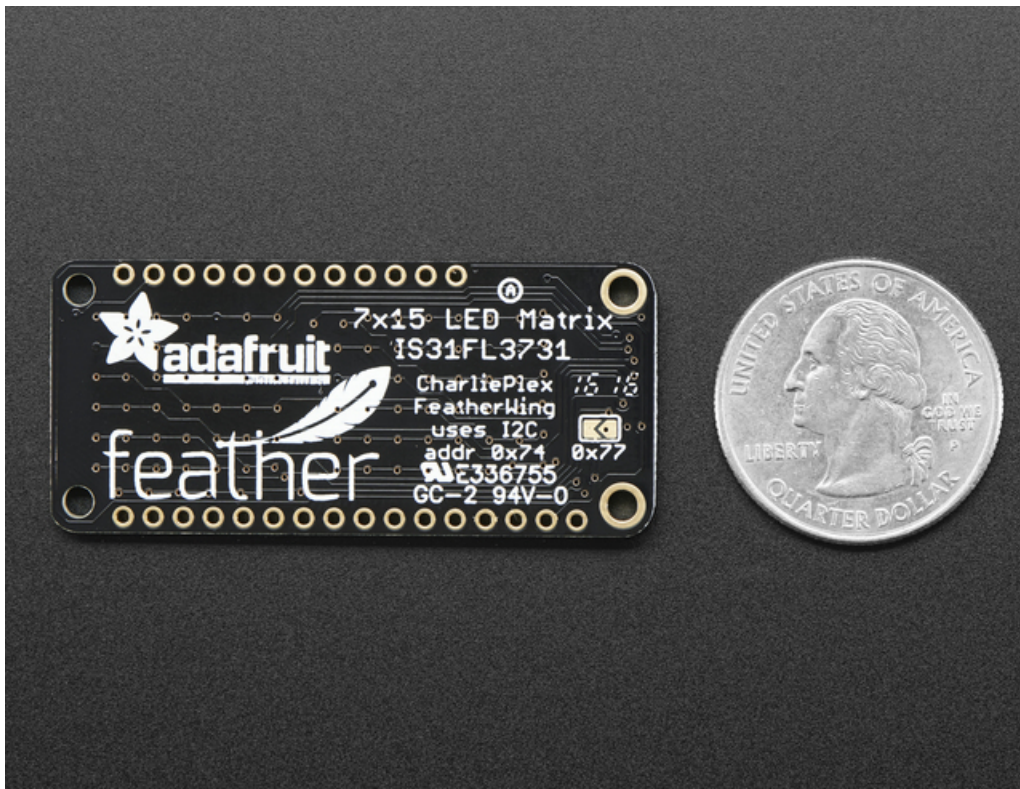
What is particularly nice about this Wing is the I2C LED driver chip has the ability to PWM each individual LED in a 15x7 grid so you can have beautiful LED lighting effects, without a lot of pin twiddling. Simply tell the chip which LED on the grid you want lit, and what brightness and it's all taken care of for you. You get 8-bit (256 level) dimming for each individual LED.



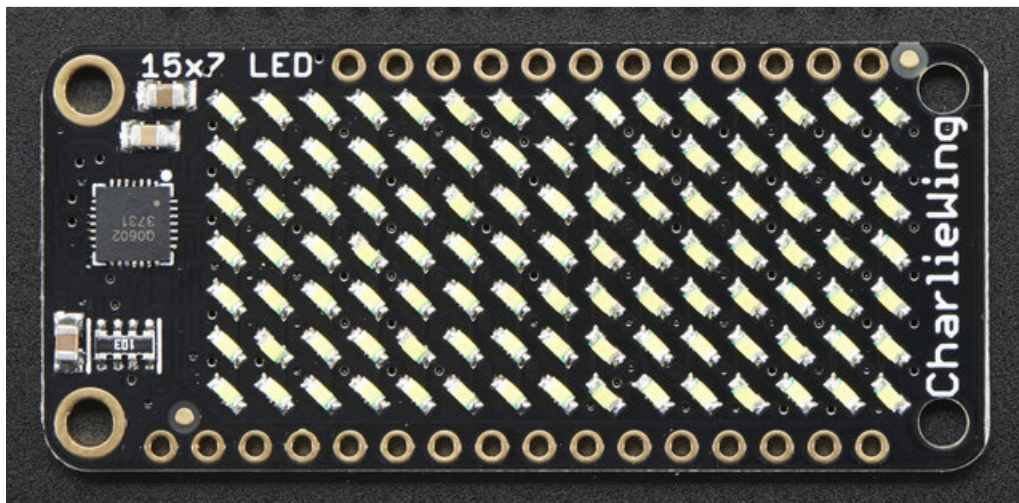
The IS31FL3731 is a nice little chip - and it runs happily over 3.3V power. Inside is enough RAM for 8 separate frames of display memory so you can set up multiple frames of an animation and flip them to be displayed with a single command. Since it uses I2C, it takes up only the SDA/SCL pins on your Feather and can share those pins with other I2C devices and sensors.



Our graphics library makes it easy to draw shapes, text and bitmaps for some strikingly beautiful LED displays.



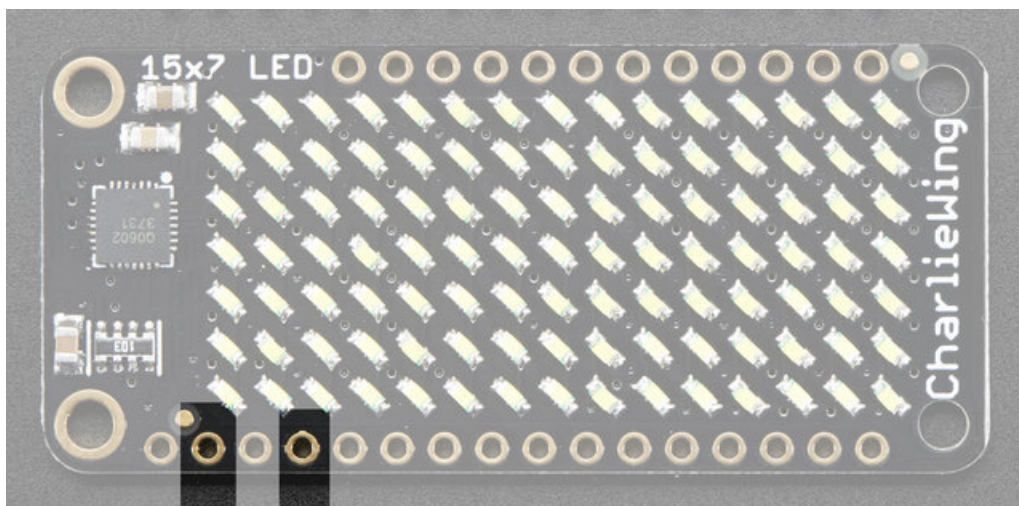
Pinouts



The 7x15 backpack makes it really easy to add a 105-LED display

The LEDs themselves do not connect to the Feather. Instead, a matrix driver chip (IS31FL3731) does the multiplexing for you. The Feather simply sends i2c commands to the chip to tell it what LEDs to light up & how bright, and it is handled for you. This takes a lot of the work and pin-requirements off the Feather. Since it uses only I2C for control, it works with any Feather and can share the I2C pins for other sensors or displays.

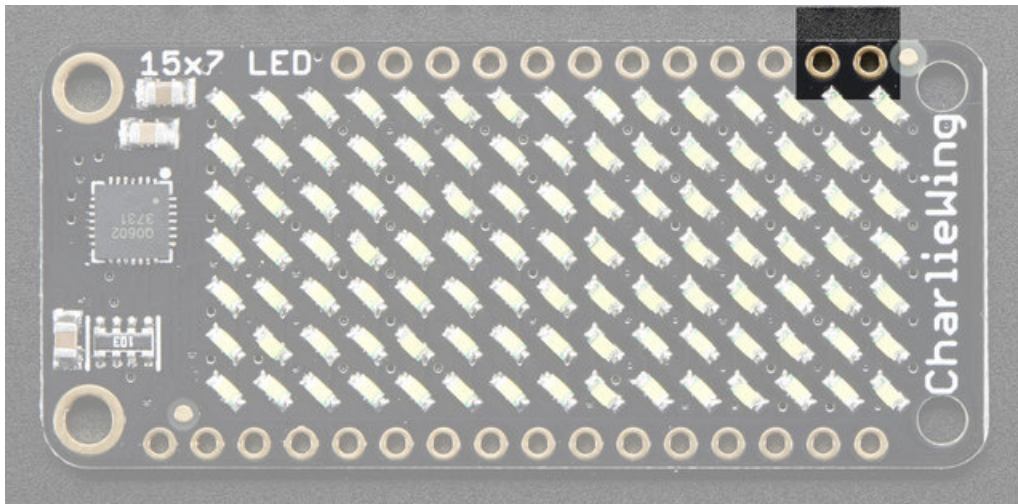
Power Pins



The LED matrix uses only the **3V** (second from left) and **GND** (fourth from left) pins for power and logic. Current draw depends on how many LEDs are lit but you can approximate it as about 40mA for most uses.

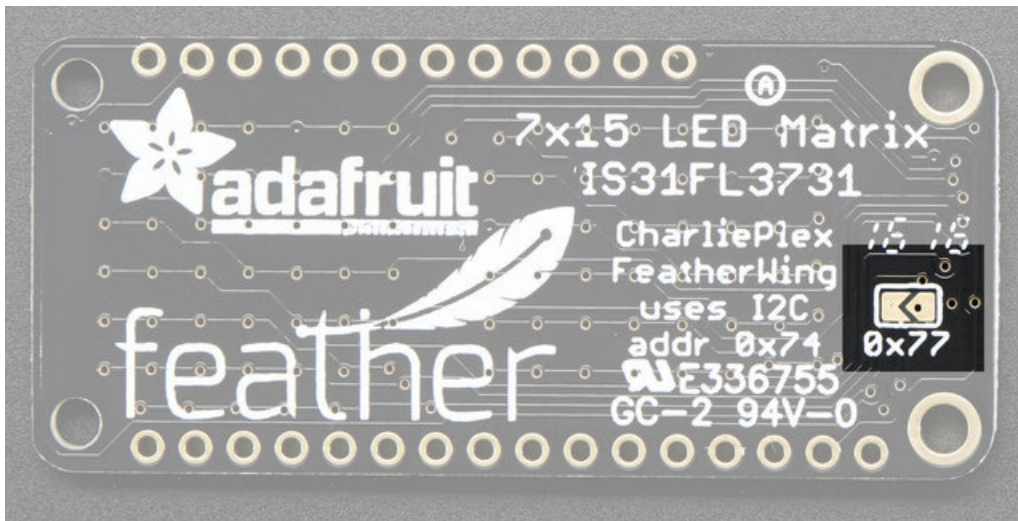
Note that the 3.3V power supply is a tiny bit lower than the forward voltage for the pure green, blue and white LED matrices but we didn't find any significant degradation in brightness. Really, they're still very bright.

I2C pins



All LED control is done over I2C using the IS31FL3731 interface library. This means **SDA** (rightmost) and **SCL** (to the left of SDA) must be connected, see above for those pins.

The default address is **0x74** but you can change the address to **0x77** by bridging solder onto the address jumper



Usage

Download Adafruit_IS31FL3731 library

To begin reading sensor data, you will need to [download Adafruit_IS31FL3731 from our github repository \(https://adafru.it/IWA\)](https://adafru.it/IWA). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

<https://adafru.it/IWB>

<https://adafru.it/IWB>

Rename the uncompressed folder **Adafruit_IS31FL3731** and check that the **Adafruit_IS31FL3731** folder contains **Adafruit_IS31FL3731.cpp** and **Adafruit_IS31FL3731.h**

Place the **Adafruit_IS31FL3731** library folder your **arduinofolder/libraries/** folder.
You may need to create the **libraries** subfolder if its your first library. Restart the IDE.

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

Install Adafruit GFX

You will need to do the same for the **Adafruit_GFX** library available here (<https://adafru.it/aJa>)

<https://adafru.it/cBB>

<https://adafru.it/cBB>

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

Place the **Adafruit_GFX** library folder your **arduinofolder/libraries/** folder like you did with the IS31FL library

Load Demo

Open up **File->Examples->Adafruit_IS31FL3731->featherswirl demo** and upload to your Feather + CharliePlex Matrix Feather

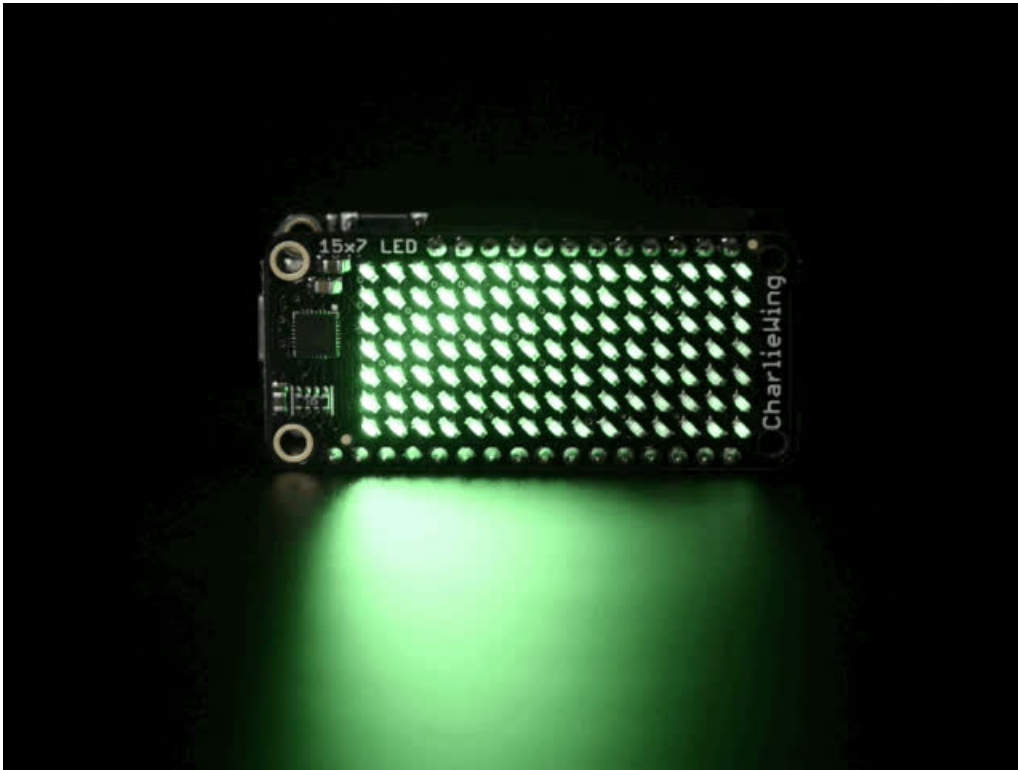
You must update the code before uploading!

At the top of the sketch, find the code that says

```
// If you're using the full breakout...
Adafruit_IS31FL3731 ledmatrix = Adafruit_IS31FL3731();
// If you're using the FeatherWing version
//Adafruit_IS31FL3731_Wing ledmatrix = Adafruit_IS31FL3731_Wing();
```

and uncomment the last line and recomment the second line so you have:


```
// If you're using the full breakout...  
//Adafruit_IS31FL3731 ledmatrix = Adafruit_IS31FL3731();  
// If you're using the FeatherWing version  
Adafruit_IS31FL3731_Wing ledmatrix = Adafruit_IS31FL3731_Wing();
```



Upload to your Arduino, you'll see the LED display swirl different brightnesses!

Library Reference

Now that you have the demo working, you can control the matrix directly.

Initialize

Start by creating a new matrix object with something like:

```
Adafruit_IS31FL3731 ledmatrix = Adafruit_IS31FL3731();
```

There's no arguments to the constructor

Then in your setup, call **begin(address)** to initialize the driver. Begin() will return false if the matrix was not found, and true if initialization worked out

```
if (! ledmatrix.begin()) {  
  Serial.println("IS31 not found");  
  while (1);  
}  
Serial.println("IS31 found!");
```

Drawing

You can then draw to the display. Note that since we write directly to the driver RAM, any pixels 'drawn' will appear immediately.

You can start with **drawPixel(x, y, brightness)** where x ranges between 0 and 15 inclusive, and y ranges between 0 and 8 inclusive. **Brightness** is the PWM of the LED, 0 is off, and 255 is all the way on.

This loop will light up every LED in increasing brightness:

```
int i = 0;  
for (uint8_t x=0; x<16; x++) {  
  for (uint8_t y=0; y<9; y++) {  
    ledmatrix.drawPixel(x, y, i++);  
  }  
}
```

Adafruit GFX

Once you get pixels drawing, you can use Adafruit GFX to draw lines, rectangles, circles, text, etc.

The Adafruit_GFX library for Arduino provides a common syntax and set of graphics functions for all of our LED, TFT, LCD and OLED displays. This allows Arduino sketches to easily be adapted between display types with minimal fuss... and any new features, performance improvements and bug fixes will immediately apply across our complete offering of color displays.

Check out our detailed tutorial here <http://learn.adafruit.com/adafruit-gfx-graphics-library> (<https://adafru.it/aPx>) It covers the latest and greatest of the GFX library!

Multiple Buffers

The IS31 has 8 full display frame buffers available. By default you draw and display to frame buffer #0

But! If you want to flip thru different images quickly, you can double buffer by writing to one buffer and then telling the IS31 to switch which one is visible.

To set which frame we are **drawing** to, use **setFrame(*n*)**
where *n* ranges from 0 to 7 inclusive

```
ledmatrix.setFrame(frame);
```

Then when you are ready to display it, to set which frame we are **displaying** to, use **displayFrame(*n*)**
where *n* ranges from 0 to 7 inclusive

```
ledmatrix.displayFrame(frame);
```

Downloads

Datasheets & Files

- [EagleCAD PCB Files on GitHub \(https://adafru.it/nEB\)](https://adafru.it/nEB)
- [Fritzing objects in Adafruit Fritzing library \(https://adafru.it/aP3\)](https://adafru.it/aP3)
- [IS31FL3731 Library \(https://adafru.it/IWA\)](https://adafru.it/IWA)
- [IS31FL3731 Datasheet \(https://adafru.it/IWC\)](https://adafru.it/IWC)

Schematic

Click to embiggen

