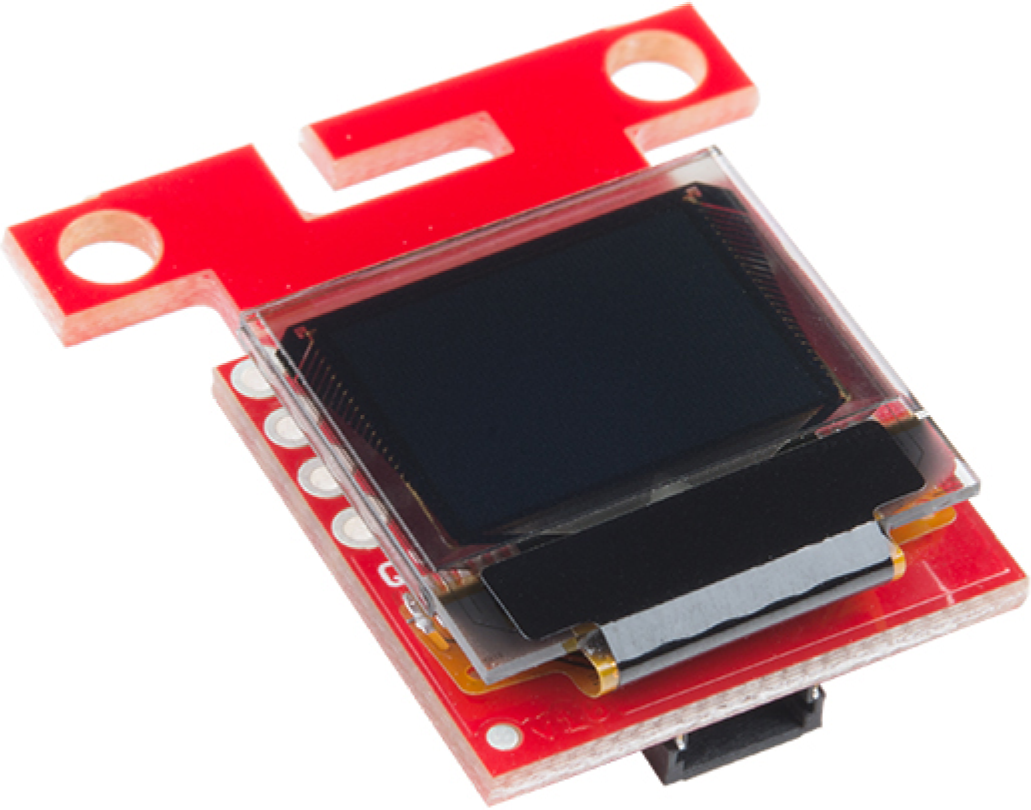
Qwiic Micro OLED Hookup Guide

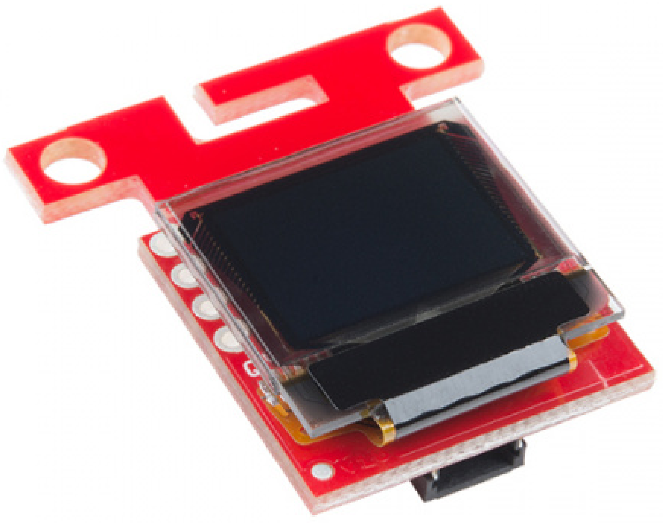
LCD-14532



Qwiic Micro OLED Hookup Guide

Introduction

The Qwiic Micro OLED is a Qwiic enabled version of our micro OLED display! This small monochrome, blue-on-black OLED display displays incredibly clear images



SparkFun Micro OLED Breakout (Qwiic)
LCD-14532

Required Materials

To get started, you'll need a microcontroller to, well, control everything.

- SparkFun ESP32 Thing
- SparkFun RedBoard - Programmed with Arduino
- Particle Photon (Headers)
- Raspberry Pi 3

Now to get into the Qwiic ecosystem, the key will be one of the following Qwiic shields to match your preference of microcontroller:

- SparkFun Qwiic HAT for Raspberry Pi
- SparkFun Qwiic Shield for Arduino
- SparkFun Qwiic Shield for Photon

You will also need a Qwiic cable to connect the shield to your OLED, choose a length that suits your needs.

- Qwiic Cable - 100mm
- Qwiic Cable - 500mm
- Qwiic Cable - 50mm
- Qwiic Cable - 200mm

Suggested Reading

If you aren't familiar with the Qwiic system, we recommend reading here for an overview.

Qwiic Connect System

We would also recommend taking a look at the following tutorials if you aren't familiar with them.

- **Introduction to I2C:** One of the main embedded communications protocols in use today.
- **Qwiic Shield for Arduino & Photon Hookup Guide** Get started with our Qwiic ecosystem with the Qwiic shield for Arduino or Photon.

Hardware Overview

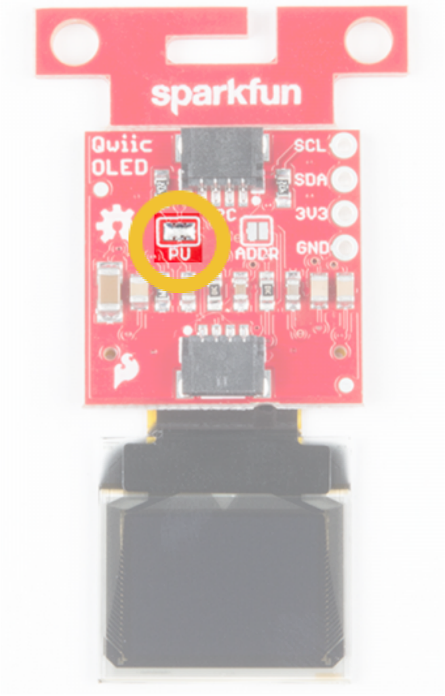
Listed below are some of the operating ranges and characteristics of the Qwiic Micro OLED.

Characteristic	Range
Voltage	**3.3V**
Temperature	-40°C to 85°C
I2C Address	0X3D (Default) or 0X3C (Closed Jumper)

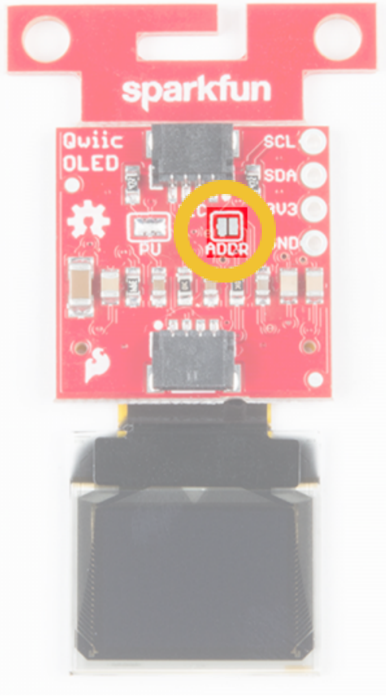
Pins

Pin	Description	Direction
GND	Ground	In
3.3V	Power	In
SDA	Data	In
SCL	Clock	In

Optional Features



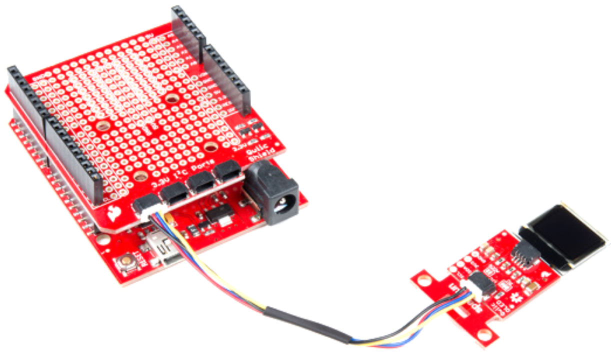
There are several jumpers on board that can be changed to facilitate several different functions. The first of which is the I2C pull-up jumper, highlighted below. If multiple boards are connected to the I2C bus, the equivalent resistance goes down, increasing your pull up strength. If multiple boards are connected on the same bus, make sure only one board has the pull-up resistors connected.



The ADDR jumper (highlighted at left) can be used to change the I2C address of the board. The default jumper is open by default, pulling the address pin high and giving us an I2C address of 0X3D. Closing this jumper will ground the address pin, giving us an I2C address of 0X3C.

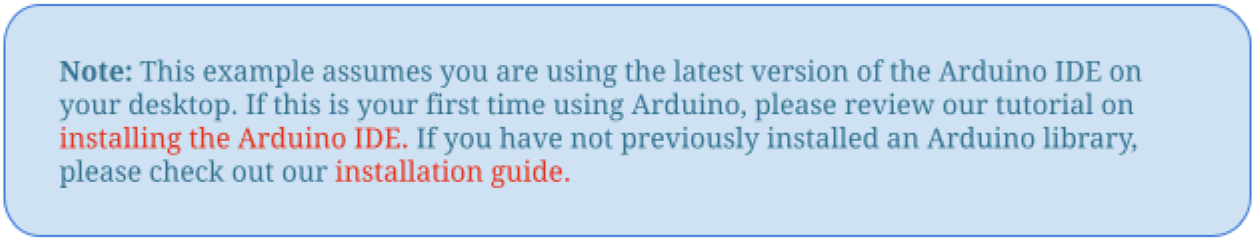
Hardware Assembly

If you haven't yet [assembled your Qwiic Shield](), now would be the time to head on over to that tutorial. With the shield assembled, Sparkfun's new Qwiic environment means that connecting the screen could not be easier. Just plug one end of the Qwiic cable into the OLED display, the other into the Qwiic Shield and you'll be ready to start displaying images on your little display.



The OLED screen itself is loosely attached to the breakout board initially, so be careful handling it! You can either use your own enclosure for the OLED display, or you can use some double sided foam tape for a less permanent solution.

Library Overview



Note: This example assumes you are using the latest version of the Arduino IDE on your desktop. If this is your first time using Arduino, please review our tutorial on installing the Arduino IDE. If you have not previously installed an Arduino library, please check out our installation guide.

First, you'll need to download and install the **SparkFun Micro OLED library.** You can install this library automatically in the Arduino IDE's Library Manager by searching for **"Micro OLED Breakout"**. Or you can manually download it from the GitHub repository. Also, make sure to download the Qwiic example sketches, which we will be reviewing in this tutorial.

- SparkFun Micro LED Library
- Micro OLED Example Sketches

Before we get started developing a sketch, let's look at the available functions of the library.

- `void command(uint8_t c);` — Sends the display a command byte.
- `void data(uint8_t c);` — Sends the display a data byte.
- `void setColumnAddress(uint8_t add);` — Sets the column address.
- `void setPageAddress(uint8_t add);` — Sets the page address.

LCD Drawing Functions

- `void clear(uint8_t mode):` — Clears the screen buffer in the OLED's memory, pass in mode = ALL to clear GDRAM in the OLED controller. Pass in mode = PAGE to clear the screen page buffer.
- `void clear(uint8_t mode, uint8_t c);` — clears the screen buffer in the OLED's memory, replaces it with a character 'c'.
- `void invert(boolean inv);` — Turns every black pixel white, turns all white pixels black.
- `void contrast(uint8_t contrast);` — Changes the contrast value anywhere between 0 and 255.
- `void display(void);` — Moves display memory to the screen to draw the image in memory.
- `void setCursor(uint8_t x, uint8_t y);` — Set cursor position to (x, y).
- `void pixel(uint8_t x, uint8_t y);` — Draw a pixel using the current fore color and current draw mode in the screen buffer's x,y position.

- `void pixel(uint8_t x, uint8_t y, uint8_t color, uint8_t mode);` — Draw a pixel with NORM or XOR draw mode in the screen buffer's x,y position.
- `void line(uint8_t x0, uint8_t y0, uint8_t x1, uint8_t y1);` — Draw line using current fore color and current draw mode from x0,y0 to x1,y1 of the screen buffer.
- `void line(uint8_t x0, uint8_t y0, uint8_t x1, uint8_t y1, uint8_t color, uint8_t mode);` — Draw line using color and mode from x0,y0 to x1,y1 of the screen buffer.
- `void lineH(uint8_t x, uint8_t y, uint8_t width);` — Draw horizontal line using current fore color and current draw mode from x,y to x+width,y of the screen buffer.
- `void lineH(uint8_t x, uint8_t y, uint8_t width, uint8_t color, uint8_t mode);` — Draw horizontal line using color and mode from x,y to x+width,y of the screen buffer.
- `void lineV(uint8_t x, uint8_t y, uint8_t height);` — Draw vertical line using current fore color and current draw mode from x,y to x,y+height of the screen buffer.
- `void lineV(uint8_t x, uint8_t y, uint8_t height, uint8_t color, uint8_t mode);` — Draw vertical line using color and mode from x,y to x,y+height of the screen buffer.
- `void rect(uint8_t x, uint8_t y, uint8_t width, uint8_t height);` — Draw rectangle using current fore color and current draw mode from x,y to x+width,y+height of the screen buffer.
- `void rect(uint8_t x, uint8_t y, uint8_t width, uint8_t height, uint8_t color , uint8_t mode);` —Draw rectangle using color and mode from x,y to x+width,y+height of the screen buffer.
- `void rectFill(uint8_t x, uint8_t y, uint8_t width, uint8_t height);` — Draw filled rectangle using current fore color and current draw mode from x,y to x+width,y+height of the screen buffer.
- `void rectFill(uint8_t x, uint8_t y, uint8_t width, uint8_t height, uint8_t color , uint8_t mode);` — Draw filled rectangle using color and mode from x,y to x+width,y+height of the screen buffer.
- `void circle(uint8_t x, uint8_t y, uint8_t radius);` — Draw circle with radius using current fore color and current draw mode with center at x,y of the screen buffer.
- `void circle(uint8_t x, uint8_t y, uint8_t radius, uint8_t color, uint8_t mode);` — Draw circle with radius using color and mode with center at x,y of the screen buffer.
- `void circleFill(uint8_t x0, uint8_t y0, uint8_t radius);` — Draw filled circle with radius using current fore color and current draw mode with center at x,y of the screen buffer.
- `void circleFill(uint8_t x0, uint8_t y0, uint8_t radius, uint8_t color, uint8_t mode);` — Draw filled circle with radius using color and mode with center at x,y of the screen buffer.

- `void drawChar(uint8_t x, uint8_t y, uint8_t c);` — Draws a character at position (x, y).
- `void drawChar(uint8_t x, uint8_t y, uint8_t c, uint8_t color, uint8_t mode);` — Draws a character using a color and mode at position (x, y)
- `void drawBitmap(uint8_t * bitArray);` — Draws a preloaded bitmap.
- `uint8_t getLCDWidth(void);` — Gets the width of the LCD as a byte.
- `uint8_t getLCDHeight(void);` — Gets the height of the LCD as a byte.

Font Settings

- `uint8_t getFontWidth(void);` — Gets the current font width as a byte.
- `uint8_t getFontHeight(void);` — Gets the current font height as a byte.
- `uint8_t getTotalFonts(void);` — Return the total number of fonts loaded into the MicroOLED's flash memory.
- `uint8_t getFontType(void);` — Returns the font type number of the current font (Font types shown below).
- `uint8_t setFontType(uint8_t type);` — Sets the font type (Font types shown below).

Font Type	Maximum Columns	Maximum Rows	Description
0	10	6	Smallest, 5x7-pixel characters.
1	6	3	Medium, 8x16-pixel characters.
2	5	3	7-segment display style characters, 10x16-pixels each.
3	5	1	Large, 12x48 (the entire screen height) characters.

- `uint8_t getFontStartChar(void);` — Returns the starting ASCII character of the current font.
- `uint8_t getFontTotalChar(void);` — Return the total characters of the current font.

Rotation and Scrolling

The following functions will scroll the screen in the various specified directions of each function. Start and Stop indicate the range of rows/columns that will be scrolling.

- `void scrollRight(uint8_t start, uint8_t stop);` — Scrolls right
- `void scrollLeft(uint8_t start, uint8_t stop);`
- `void scrollVertRight(uint8_t start, uint8_t stop);`
- `void scrollVertLeft(uint8_t start, uint8_t stop);`
- `void scrollStop(void);` The following two functions are pretty self explanatory, they will flip the graphics if `flip` is true.
- `void flipVertical(boolean flip);`
- `void flipHorizontal(boolean flip);`

Example: Feature Demo

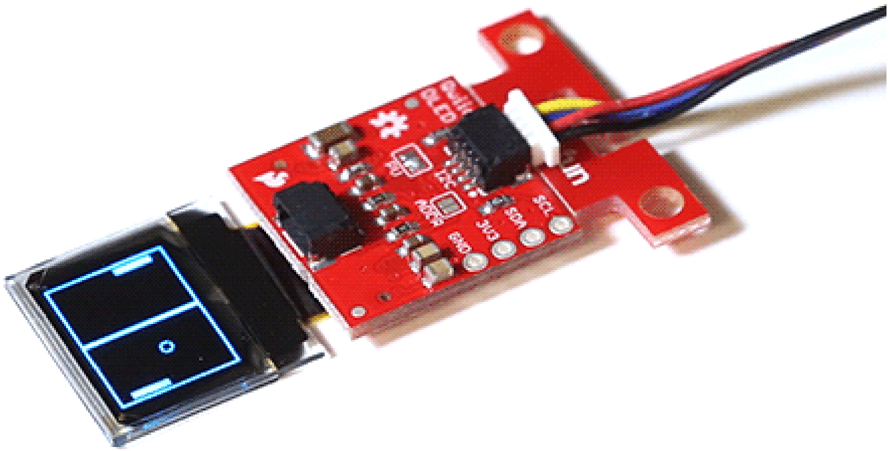
This first example demonstrates many of the available features of the screen through several applications. Keep in mind when looking at this example that drawing anything takes two steps. You must first write what you want the screen to display into the screens memory, then you must tell the screen to display what is in its memory. To begin, we must include our Wire library to use I2C, and the SFE_MicroOLED library to control the screen. Then the code initializes the OLED using DC_JUMPER = 1. If you have closed the jumper on the breakout board, use DC_JUMPER = 0.

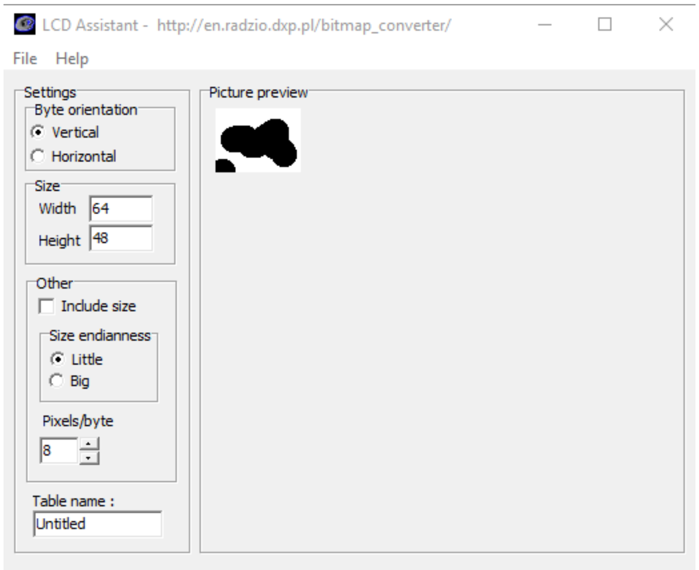
```
#include <Wire.h>  // Include Wire if you're using I2C
#include <SFE_MicroOLED.h>  // Include the SFE_MicroOLED library
#define PIN_RESET 9
#define DC_JUMPER 1
MicroOLED oled(PIN_RESET, DC_JUMPER);    // I2C declaration

void setup()
{
  delay(100);
  Wire.begin();
  oled.begin();    // Initialize the OLED
  oled.clear(ALL); // Clear the display's internal memory
  oled.display();  // Display what's in the buffer (splashscreen)
  delay(1000);     // Delay 1000 ms
  oled.clear(PAGE); // Clear the buffer.

  randomSeed(analogRead(A0) + analogRead(A1));
}
```

The code below tells the microcontroller how to print each example. This is a good place to look to see examples of how the functions discussed earlier are implemented.

```
void loop()
{
  lineExample();   // Then the line example function
  shapeExample();  // Then the shape example
  textExamples();  // Finally the text example
}
```

The example code will look something like the GIF here.





LCD Assistant - http://en.radzio.dxp.pl/bitmap_converter/

Byte orientation

● Vertical
○ Horizontal

Size

Width: 64
Height: 48

Other

□ Include size

Size endianness

● Little
○ Big

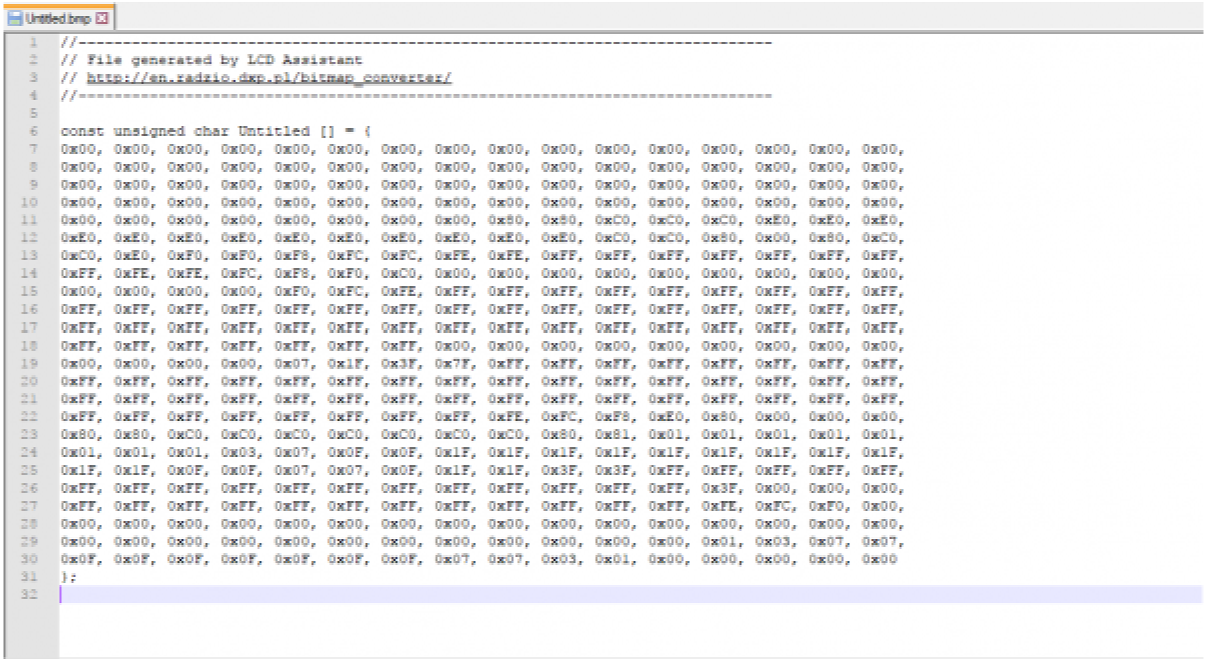
Pixels/byte: 8

Table name : Untitled

Example: Drawing Bitmaps

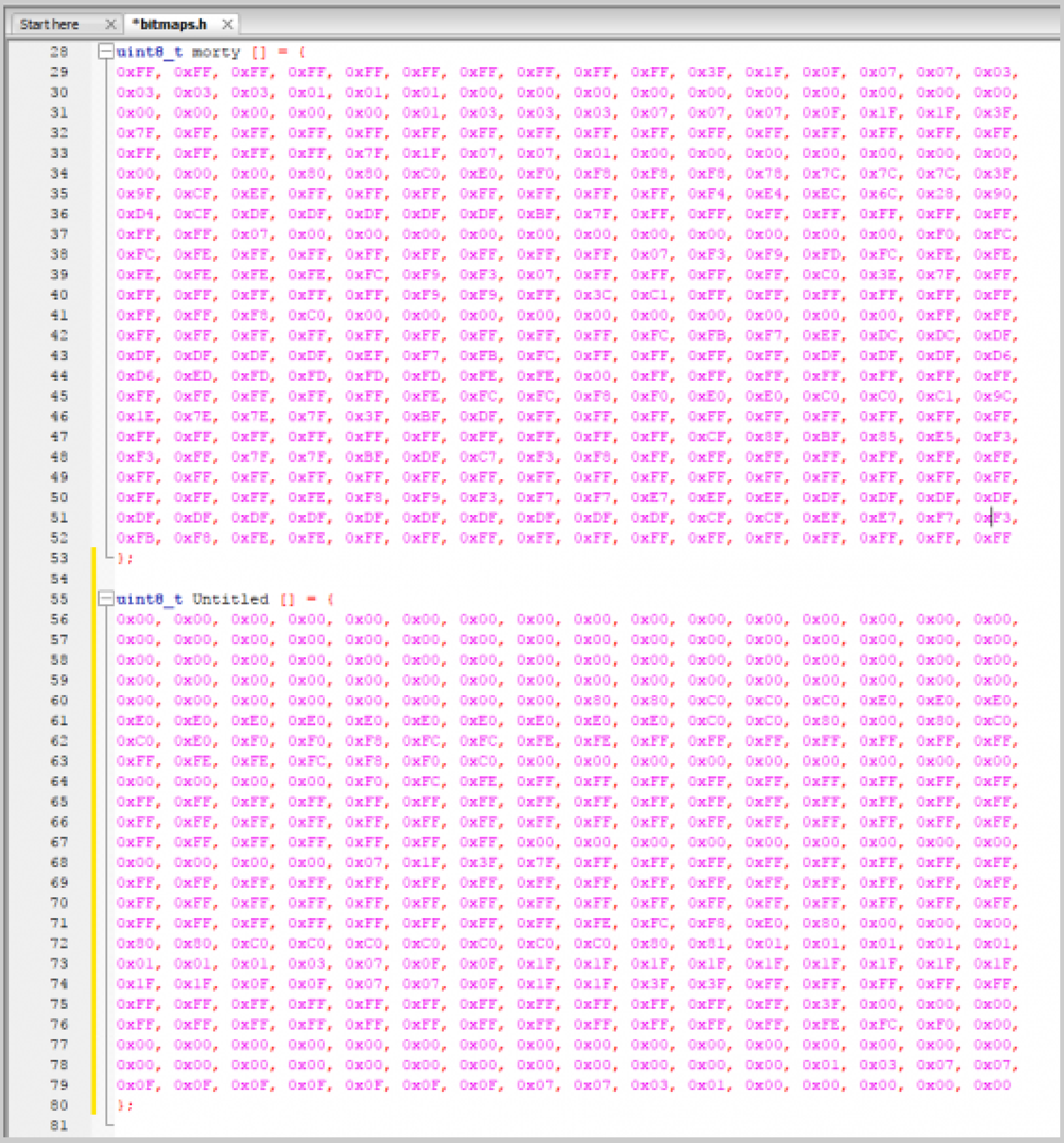
It is also possible to load bitmaps of your own custom images into the screen. This can be done using this Bitmap generator. The tool is pretty self explanatory, just load in an image, tell the tool that your screen is 64x48, go to File, and Save the output.

Open the file generated as a text file, it should look something like the below image.



```
//------------------------------------------------------------------------------
// File generated by LCD Assistant
// http://en.radzio.dxp.pl/bitmap_converter/
//------------------------------------------------------------------------------

const unsigned char Untitled [] = {
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x80, 0xC0, 0xC0, 0xC0, 0xE0, 0xE0, 0xE0,
0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xC0, 0xC0, 0x80, 0x00, 0x80, 0xC0,
0xC0, 0xE0, 0xF0, 0xF0, 0xF8, 0xFC, 0xFC, 0xFE, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFE, 0xFE, 0xFC, 0xF8, 0xF0, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0xF0, 0xFC, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x07, 0x1F, 0x3F, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFC, 0xF8, 0xE0, 0x80, 0x00, 0x00, 0x00,
0x80, 0x80, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0x80, 0x81, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x03, 0x07, 0x0F, 0x0F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F,
0x1F, 0x1F, 0x0F, 0x0F, 0x07, 0x07, 0x0F, 0x1F, 0x1F, 0x3F, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x3F, 0x00, 0x00, 0x00,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFC, 0xF0, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x03, 0x07, 0x07,
0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x07, 0x07, 0x03, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00
};
```

This array is the image that will be displayed by the screen, so now we just have to paste it into the bitmaps.h header file as the correct data type so our compiler is able to find the image. Make sure you change the array to a uint8_t. The pasted bitmap should look something like the below image, with the variable type changed to uint8_t.



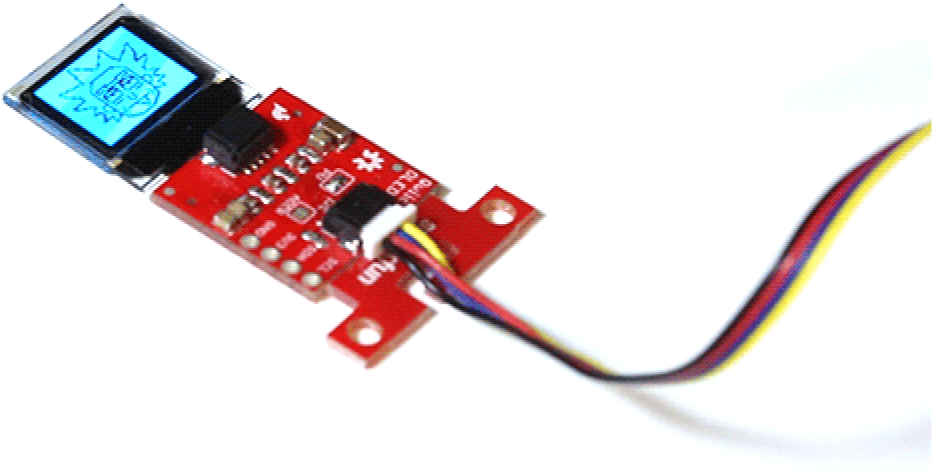
```
uint8_t morty [] = {
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x3F, 0x1F, 0x0F, 0x07, 0x07, 0x03,
    0x03, 0x03, 0x03, 0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x03, 0x03, 0x03, 0x07, 0x07, 0x07, 0x0F, 0x1F, 0x1F, 0x3F,
    0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0x7F, 0x1F, 0x07, 0x07, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x80, 0x80, 0xC0, 0xE0, 0xF0, 0xF8, 0xF8, 0xF8, 0x78, 0x7C, 0x7C, 0x7C, 0x3E,
    0x9F, 0xCF, 0xEF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF4, 0xE4, 0xEC, 0x6C, 0x28, 0x90,
    0xD4, 0xCF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xBF, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xF0, 0xFC,
    0xFC, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x07, 0xF3, 0xF9, 0xFD, 0xFC, 0xFE, 0xFE,
    0xFE, 0xFE, 0xFE, 0xFE, 0xFC, 0xF9, 0xF3, 0x07, 0xFF, 0xFF, 0xFF, 0xFF, 0xC0, 0x3E, 0x7F, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF9, 0xF9, 0xFF, 0x3C, 0xC1, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xF8, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC, 0xFB, 0xF7, 0xEF, 0xDC, 0xDC, 0xDF,
    0xDF, 0xDF, 0xDF, 0xDF, 0xEF, 0xF7, 0xFB, 0xFC, 0xFF, 0xFF, 0xFF, 0xFF, 0xDF, 0xDF, 0xDF, 0xD6,
    0xD6, 0xED, 0xFD, 0xFD, 0xFD, 0xFD, 0xFE, 0xFE, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFC, 0xFC, 0xF8, 0xF0, 0xE0, 0xE0, 0xC0, 0xC0, 0xC1, 0x9C,
    0x1E, 0x7E, 0x7E, 0x7F, 0x3F, 0xBF, 0xDF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xCF, 0x8F, 0xBF, 0x85, 0xE5, 0xF3,
    0xF3, 0xFF, 0x7F, 0x7F, 0xBF, 0xDF, 0xC7, 0xF3, 0xF8, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFE, 0xF8, 0xF9, 0xF3, 0xF7, 0xF7, 0xE7, 0xEF, 0xEF, 0xDF, 0xDF, 0xDF, 0xDF,
    0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xDF, 0xCF, 0xCF, 0xEF, 0xE7, 0xF7, 0xF3,
    0xFB, 0xF8, 0xFE, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF
};

uint8_t Untitled [] = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x80, 0xC0, 0xC0, 0xC0, 0xE0, 0xE0, 0xE0,
    0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0, 0xC0, 0xC0, 0x80, 0x00, 0x80, 0xC0,
    0xC0, 0xE0, 0xF0, 0xF0, 0xF8, 0xFC, 0xFC, 0xFE, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFE, 0xFE, 0xFC, 0xF8, 0xF0, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0xF0, 0xFC, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x07, 0x1F, 0x3F, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFC, 0xF8, 0xE0, 0x80, 0x00, 0x00, 0x00,
    0x80, 0x80, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0xC0, 0x80, 0x81, 0x01, 0x01, 0x01, 0x01, 0x01,
    0x01, 0x01, 0x01, 0x03, 0x07, 0x0F, 0x0F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F,
    0x1F, 0x1F, 0x0F, 0x0F, 0x07, 0x07, 0x0F, 0x1F, 0x1F, 0x3F, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x3F, 0x00, 0x00, 0x00,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFC, 0xF0, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x03, 0x07, 0x07,
    0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x0F, 0x07, 0x07, 0x03, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00
};
```

Now we will be able to call drawBitmap(Untitled) to draw our image. Some example code showing how to display some Rick and Morty bitmaps is shown below.

```
#include <Wire.h>  // Include Wire if you're using I2C
#include <SFE_MicroOLED.h>  // Include the SFE_MicroOLED library
#include "bitmaps.h"

//The library assumes a reset pin is necessary. The Qwiic OLED has RST hard-wired, so pick an arbitrarty IO pin that is not being used
#define PIN_RESET 9
//The DC_JUMPER is the I2C Address Select jumper. Set to 1 if the jumper is open (Default), or set to 0 if it's closed.
#define DC_JUMPER 0

MicroOLED oled(PIN_RESET, DC_JUMPER);    // I2C declaration

void setup()
{
  delay(100);
  oled.begin();     // Initialize the OLED
  oled.clear(ALL); // Clear the display's internal memory
  oled.display();  // Display what's in the buffer (splashscreen)
  delay(1000);     // Delay 1000 ms
  oled.clear(PAGE); // Clear the buffer.

}

void loop()
{
    drawRick();
    delay(5000);
    drawMorty();
    delay(5000);
}
```

```
void drawRick()
{
    oled.clear(ALL);
    oled.clear(PAGE);
    oled.drawBitmap(rick);//Display Logo
    oled.display();
}

void drawMorty()
{
    oled.clear(ALL);
    oled.clear(PAGE);
    oled.drawBitmap(morty);//Display Logo
    oled.display();
}
```

The output of this code will look something like this:



Resources and Going Further

Now that you've successfully got your OLED displaying things, it's time to incorporate it into your own project!

For more on the Qwiic Micro OLED, check out the links below:

- **Schematic (PDF)**
- **Eagle Files (ZIP)**
- **Datasheet (PDF)**
- **Bitmap Generator**
- **Qwiic System Landing Page**
- **SparkFun Micro OLED Arduino Library**
- **SparkFun Qwiic Micro OLED GitHub Repository** -- Board design files for the Qwiic Micro OLED.
- **Product Showcase: Qwiic Presence Sensor & OLED**

Need some inspiration for your next project? Check out some of these related tutorials:

Micro OLED Breakout Hookup Guide
Learn how to hook up the Micro OLED breakout to an Arduino. Then draw pixels, shapes, text and bitmaps all over it!

Photon OLED Shield Hookup Guide
The Photon OLED Shield has everything you need to add a small yet crisp OLED screen to your Photon projects. This hookup guide will show you how to get started.

MicroView Hookup Guide
A quick tutorial to get you up and running with your MicroView Development Board.

Or check out the following blogs for ideas.

Enginursday: Lightning Detector for the Trail
Enginursday: Upgrading the Lightning Detector
Perfecting Coffee Roasting with the Qwiic Photodetector Breakout