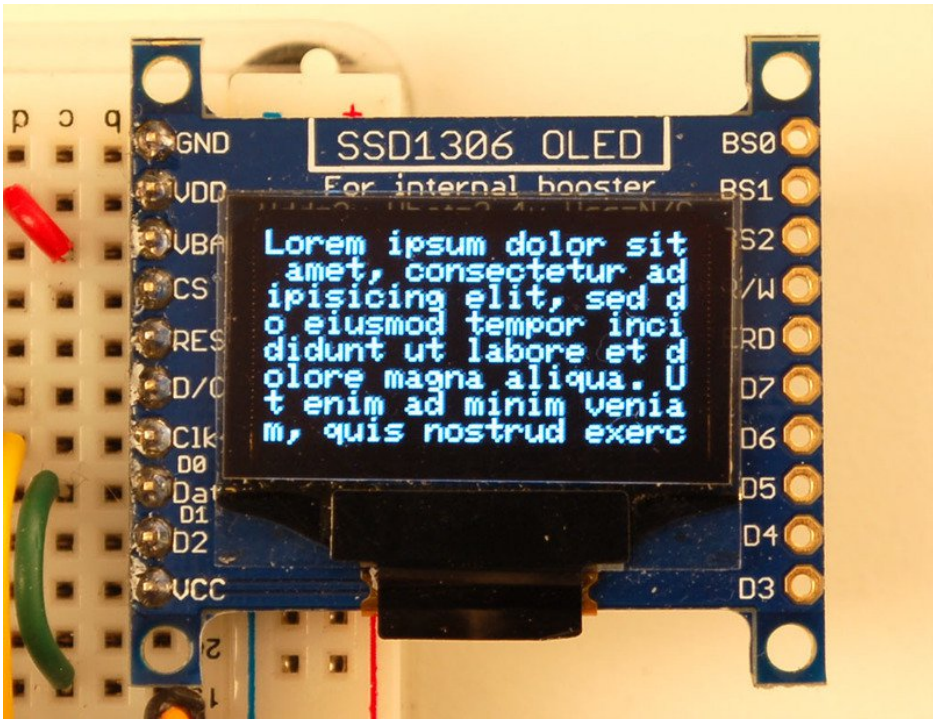
adafruit learning system

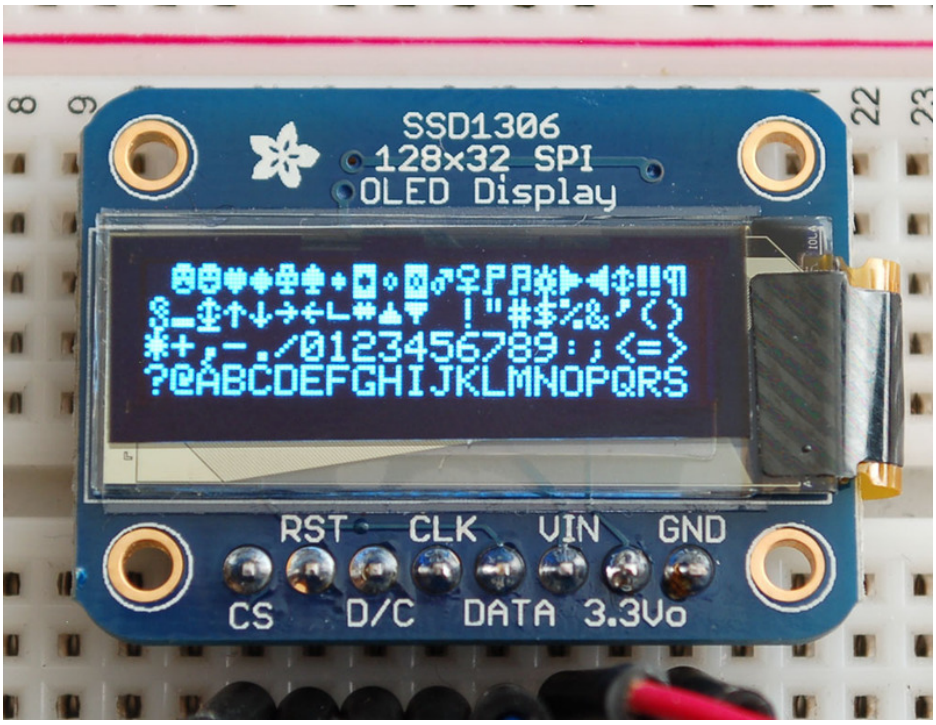
Monochrome OLED Breakouts

Created by lady ada

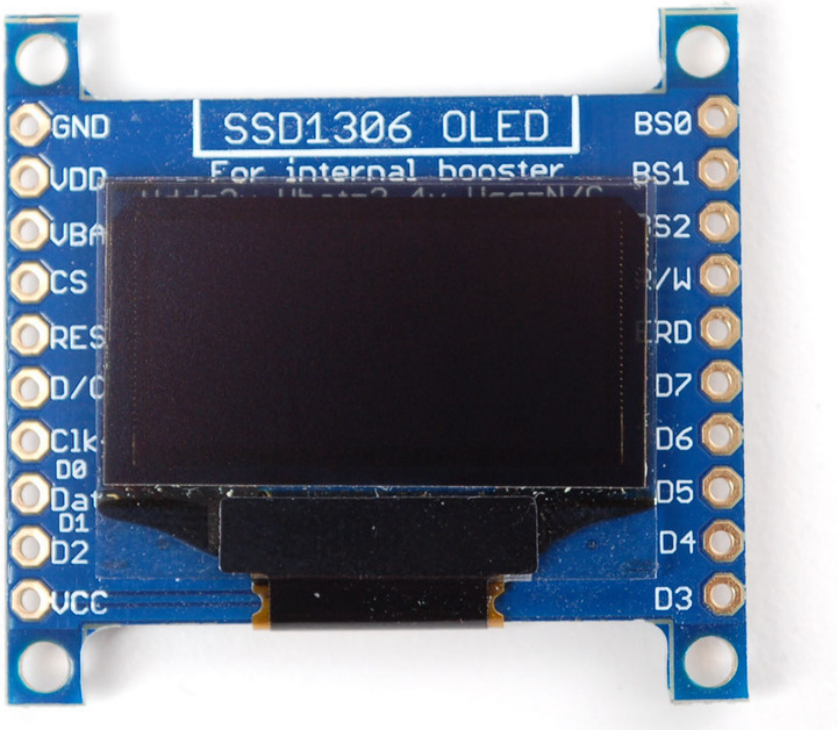
Last updated on 2019-09-18 07:27:06 PM UTC

Overview





This is a quick tutorial for our 128x64 and 128x32 pixel monochrome OLED displays. These displays are small, only about 1" diagonal, but very readable due to the high contrast of an OLED display. Each OLED display is made of 128x64 or 128x32 individual white OLEDs, each one is turned on or off by the controller chip. Because the display makes its own light, no backlight is required. This reduces the power required to run the OLED and is why the display has such high contrast; we really like this miniature display for its crispness!



The driver chip, **SSD1306** can communicate in multiple ways including **I2C**, **SPI** and **8-bit parallel**. However, only the 128x64 display has all these interfaces available. For the 128x32 OLED, only SPI is available. Frankly, we prefer SPI since its the most flexible and uses a small number of I/O pins so our example code and wiring diagram will use that.

Power Requirements

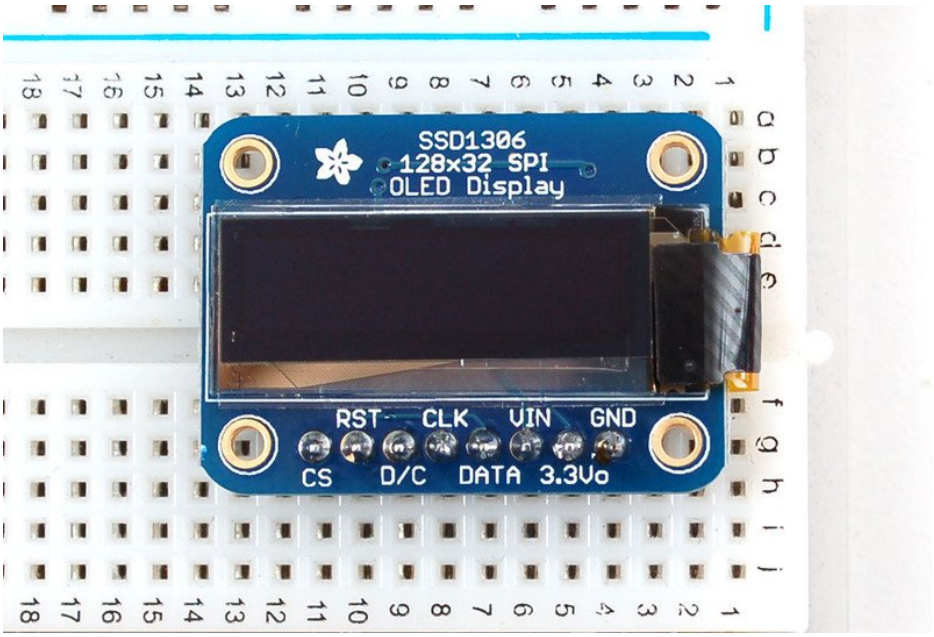
OLED Power Requirements

The OLED and driver require a 3.3V power supply and 3.3V logic levels for communication. The power requirements depend a little on how much of the display is lit but on average the display uses about 20mA from the 3.3V supply. Built into the OLED driver is a simple switch-cap charge pump that turns 3.3v-5v into a high voltage drive for the OLEDs. You can run the entire display off of one 3.3V supply or use 3.3V for the chip power and up to 4.5V for the OLED charge pump or 3.3V for the chip power and a 7-9V supply directly into the OLED high voltage pin.

5V- ready 128x64 and 128x32 OLEDs

Unless you have the older v1 128x64 OLED, you can rest assured that your OLED is 5V ready. All 1.3" 128x64 and the small 128x32 SPI and I2C are 5V ready, if you have a v2 0.96" 128x64 OLED with the 5V ready mark on the front, it's also 5V safe. If you have an older 0.96" OLED (see below) you'll need to take extra care when wiring it to a 5V micontroller. The OLED is designed to be 5V compatible so you can power it with 3-5V and the onboard regulator will take care of the rest.

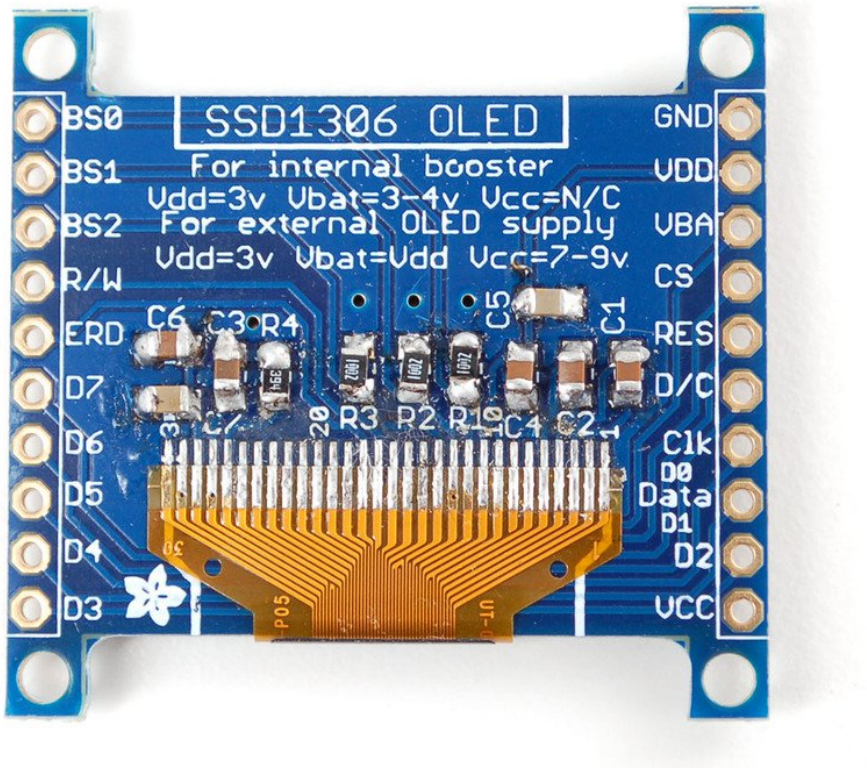
All OLEDs are safe to use with 3.3V logic and power.



Simply connect **GND** to ground, and **Vin** to a 3 to 5V power supply. There will be a 3.3V output on the **3Vo** pin in case you want a regulated 3.3V supply for something else.

0.96" 128x64 OLED

The older 0.96" 128x64 OLED is a little more complex to get running as it is not 5V compatible by default, so you have to provide it with 3.3V power.



- **VDD** is the 3.3V logic power. This must be 3 or 3.3V
- **VBAT** is the input to the charge pump. If you use the charge pump, this must be 3.3V to 4.2V
- **VCC** is the high voltage OLED pin. If you're using the internal charge pump, this must be left unconnected. If you're not using the charge pump, connect this to a 7-9V DC power supply.

For most users, we suggest connecting **VDD** and **VBAT** together to 3.3V and then leaving **VCC** unconnected.

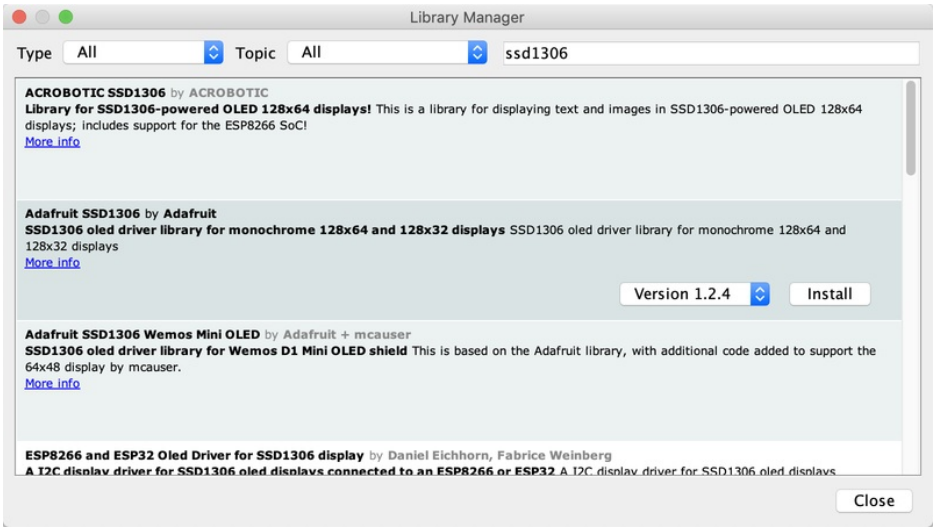
Arduino Library & Examples

For all of the different kinds of small OLED monochrome displays, you'll need to install the Arduino libraries. The code we have is for any kind of Arduino, if you're using a different microcontroller, the code is pretty simple to adapt, the interface we use is basic bit-twiddling SPI or I2C

Install Arduino Libraries

Using these OLEDs with Arduino sketches requires that two libraries be installed: **Adafruit_SSD1306**, which handles the low-level communication with the hardware, and **Adafruit_GFX**, which builds atop this to add graphics functions like lines, circles and text.

In recent versions of the Arduino IDE software (1.6.2 and later), this is most easily done through the Arduino Library Manager, which you'll find in the "Sketch" menu: **Sketch→Include Library→Manage Libraries...**

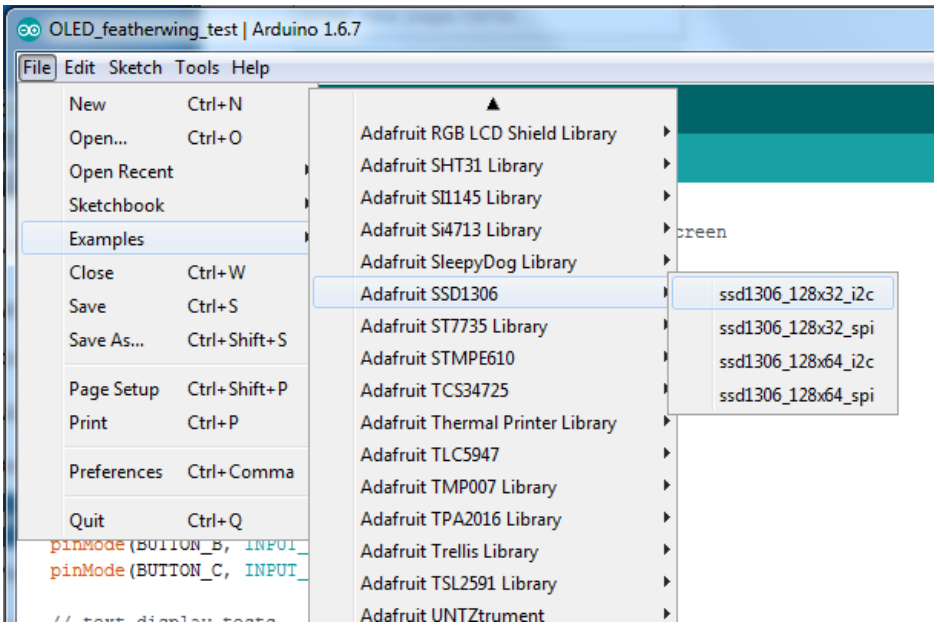


Enter "ssd1306" in the search field, locate the Adafruit SSD1306 library and select "Install" (or "Upgrade" if you have an older version). Then repeat the same for "gfx" and the Adafruit GFX library.

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

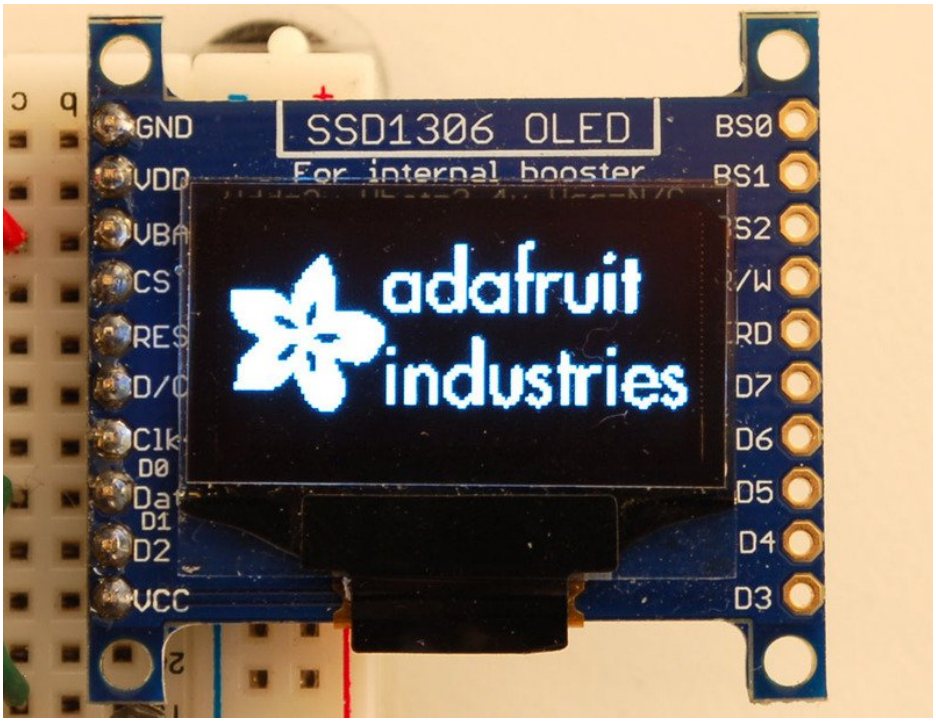
Run Demo!

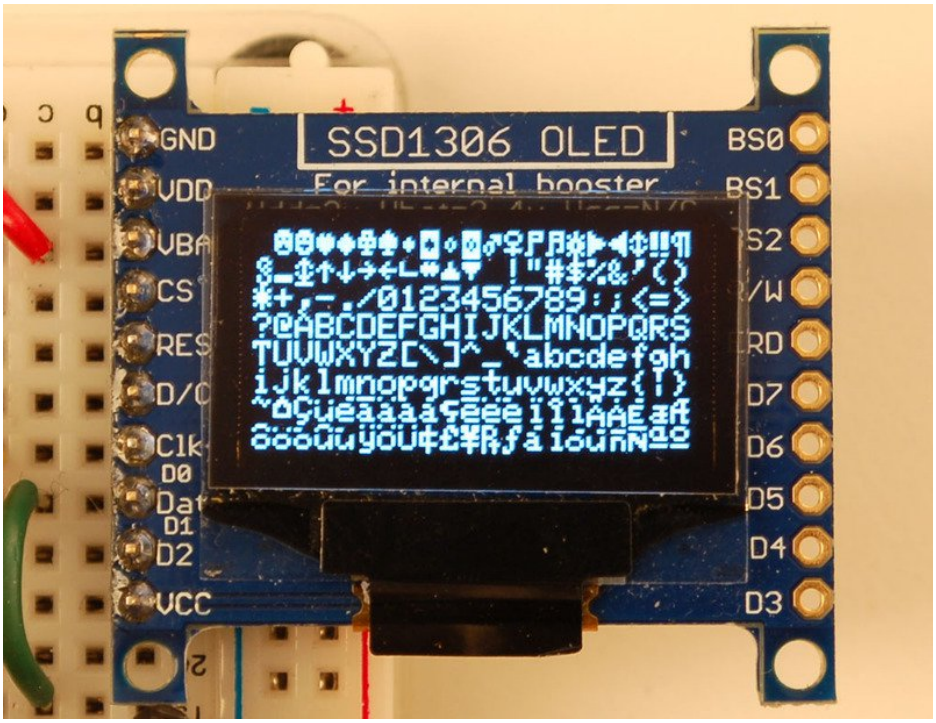
After installing the **Adafruit_SSD1306** and **Adafruit_GFX** library, restart the Arduino IDE. You should now be able to access the sample code by navigating through menus in this order:
File→Sketchbook→Libraries→Adafruit_SSD1306→SSD1306...



After you've finished wiring the display as indicated on the following pages, load the example sketch to demonstrate the capabilities of the library and display.

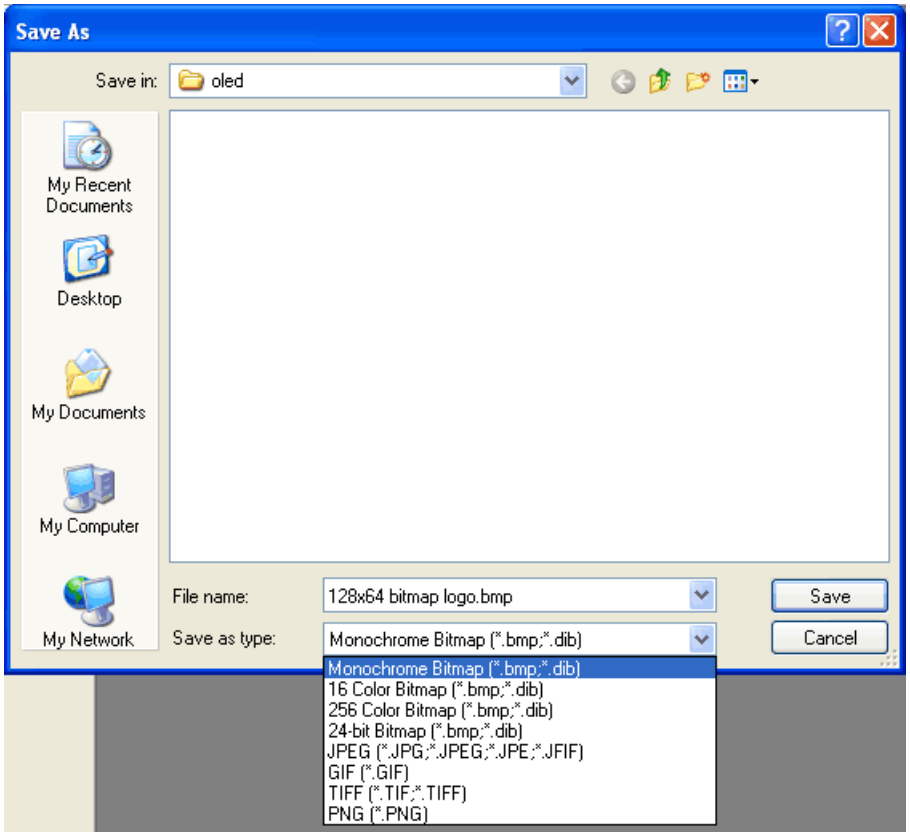
The OLED SSD1306 driver is based on the Adafruit GFX library which provides all the underlying graphics functions such as drawing pixels, lines, circles, etc. For more details about what you can do with the OLED check out the GFX library tutorial (https://adafru.it/aPx)



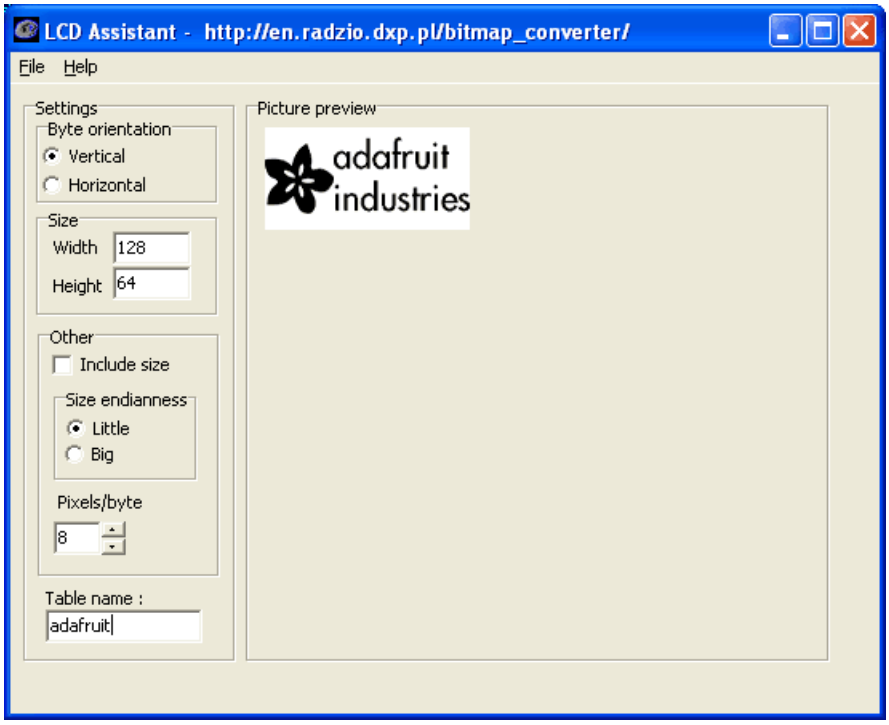


Create Bitmaps

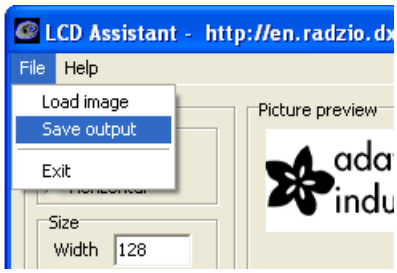
You can create bitmaps to display easily with the LCD assistant software (https://adafru.it/aPs). First make your image using any kind of graphics software such as photoshop or Paint and save as a **Monochrome Bitmap (bmp)**



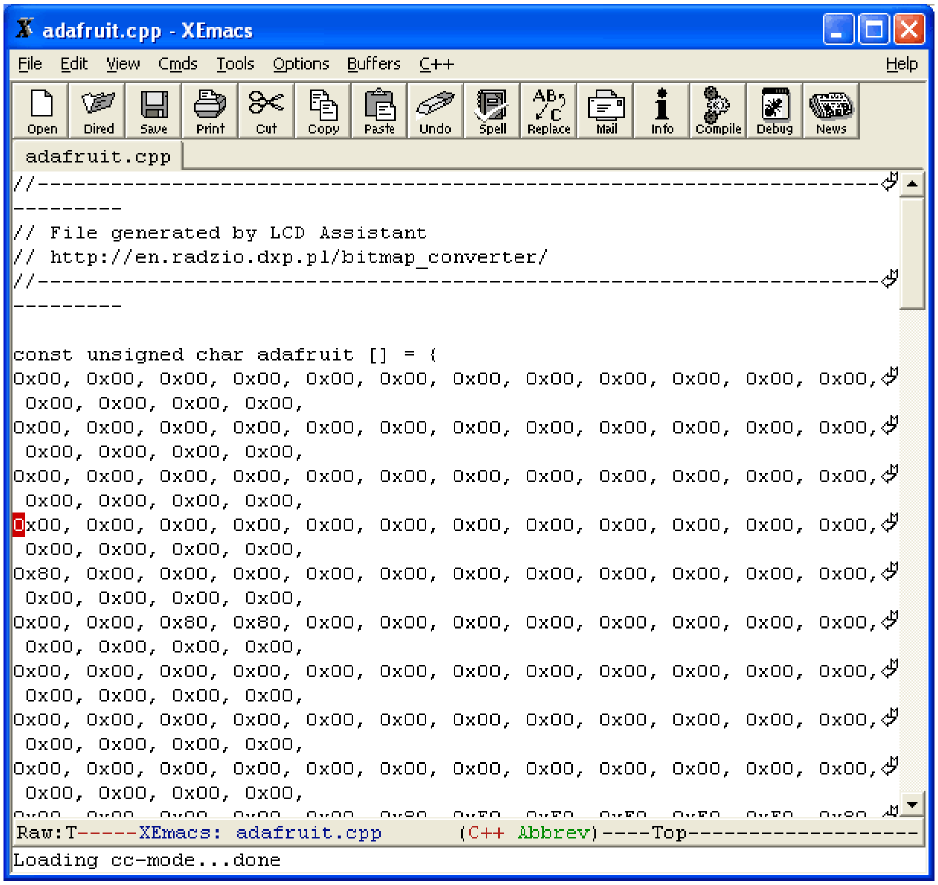
Select the following options (You might also want to try **Horizontal** if **Vertical** is not coming out right)



and import your monochrome bitmap image. Save the output to a **cpp** file



You can use the output directly with our example code

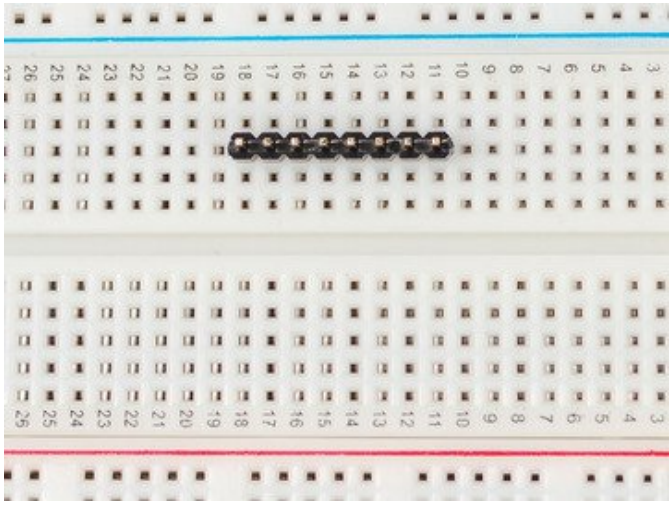

adafruit.cpp - XEmacs
File Edit View Cmds Tools Options Buffers C++ Help
Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News
adafruit.cpp
//--

// File generated by LCD Assistant
// http://en.radzio.dxp.pl/bitmap_converter/
//--
----
const unsigned char adafruit [] = {
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, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x80, 0x80, 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,
Raw:T-----XEmacs: adafruit.cpp (C++ Abbrev)----Top-------------------
Loading cc-mode...done

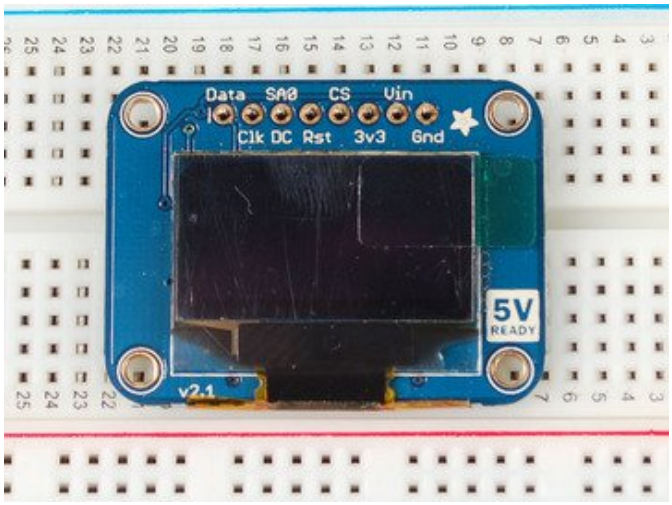
Wiring 128x64 OLEDs

Solder Header

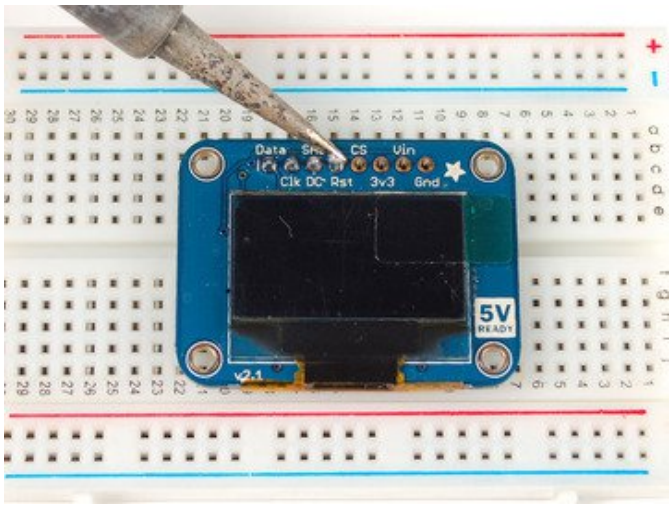
Before you start wiring, a strip of header must be soldered onto the OLED. It is not possible to "press-fit" the header, it must be attached!



Start by placing an 8-pin piece of header with the **long** ends down into a breadboard for stability



Place the OLED on top so all the short ends of the header stick thru the header pads



Finish by soldering each of the 8 pins to the 8 pads!

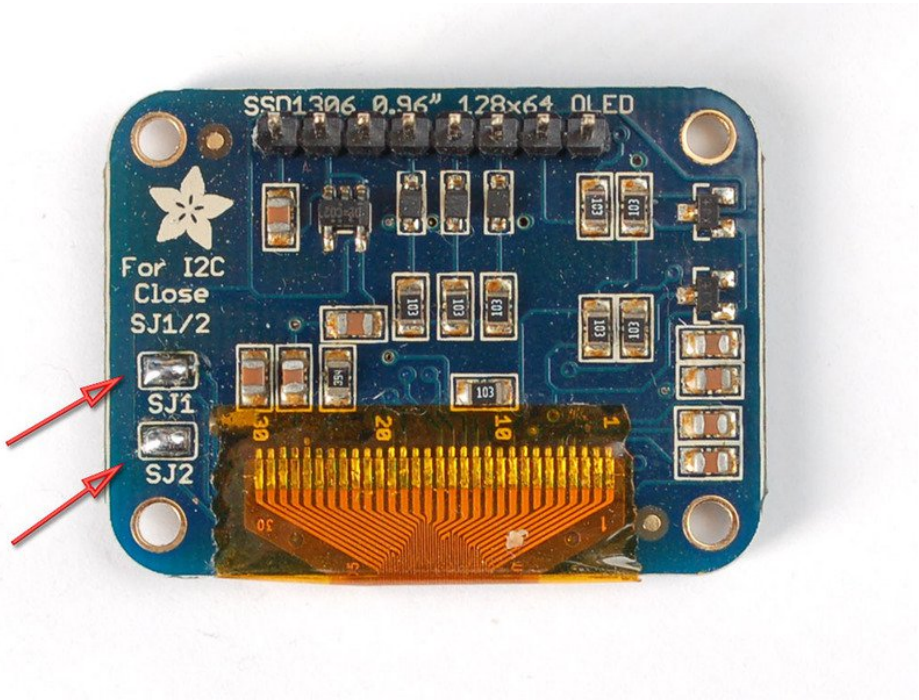
I2C or SPI

The nice thing about the 128x64 OLEDs is that they can be used with I2C (+ a reset line) or SPI. SPI is generally faster than I2C but uses more pins. It's also easier for some microcontrollers to use SPI. Anyways, you can use either one with this display

Using with I2C

The display can be used with any I2C microcontroller. Because the I2C interface is for 'writing' to the display only, you'll still have to buffer the entire 512 byte frame in the microcontroller RAM - you can't read data from the OLED (even though I2C is a bidirectional protocol).

To start, you'll need to solder the two jumpers on the back of the OLED. **Both** must be soldered 'closed' for I2C to work!



Finally, connect the pins to your Arduino

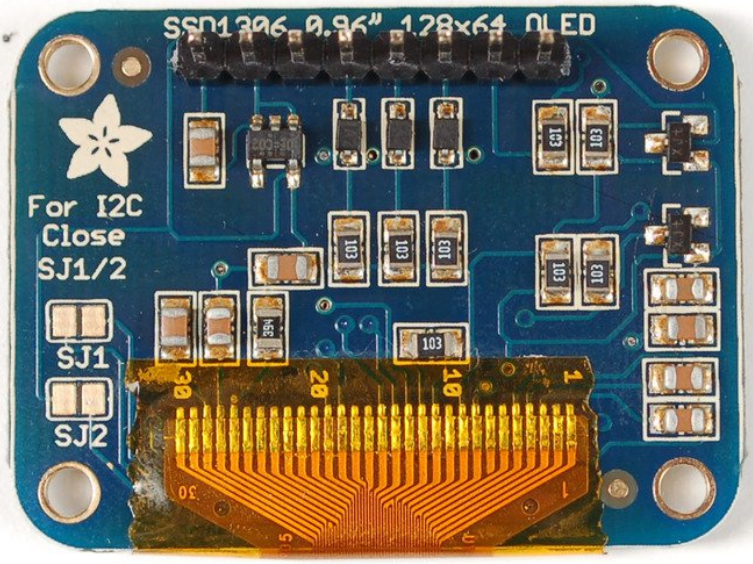
- **GND** goes to ground
- **Vin** goes to 5V
- **Data** to I2C SDA (on the Uno, this is **A4** on the Mega it is **20** and on the Leonardo digital **2**)
- **Clk** to I2C SCL (on the Uno, this is **A5** on the Mega it is **21** and on the Leonardo digital **3**)
- **RST** to digital **4** (you can change this pin in the code, later)

This matches the example code we have written. Once you get this working, you can try a different Reset pin (you can't change the SDA and SCL pins).

Finally you can run the **File→Sketchbook→Libraries→Adafruit_SSD1306→SSD1306_128x64_i2c example**

Using with SPI

The breakouts are ready for SPI by default, but if you used them for I2C at some point, you'll need to remove the solder jumpers. Use wick or a solder sucker to make sure both are clear!



Finally, connect the pins to your Arduino -

- **GND** goes to ground
- **Vin** goes to 5V
- **DATA** to digital **9**
- **CLK** to digital **10**
- **D/C** to digital **11**
- **RST** to digital **13**
- **CS** to digital **12**

*(**Note**: If using the display with other SPI devices, D/C, CLK and DAT may be shared, but CS must be unique for each device.)*

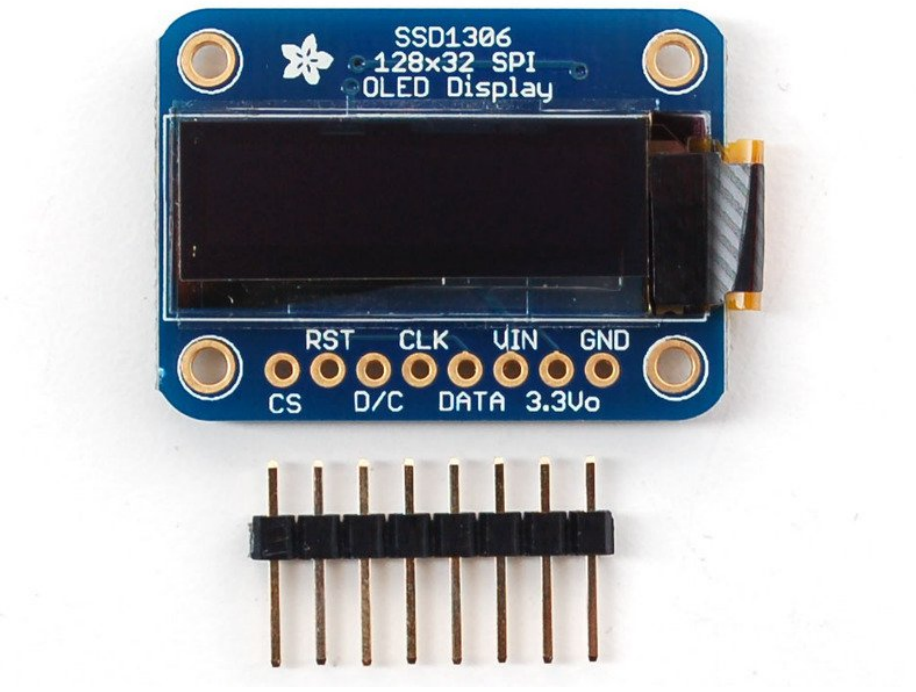
This matches the example code we have written. Once you get this working, you can try another set of pins.

Finally you can run the **File→Sketchbook→Libraries→Adafruit_SSD1306→SSD1306_128x64_spi example**

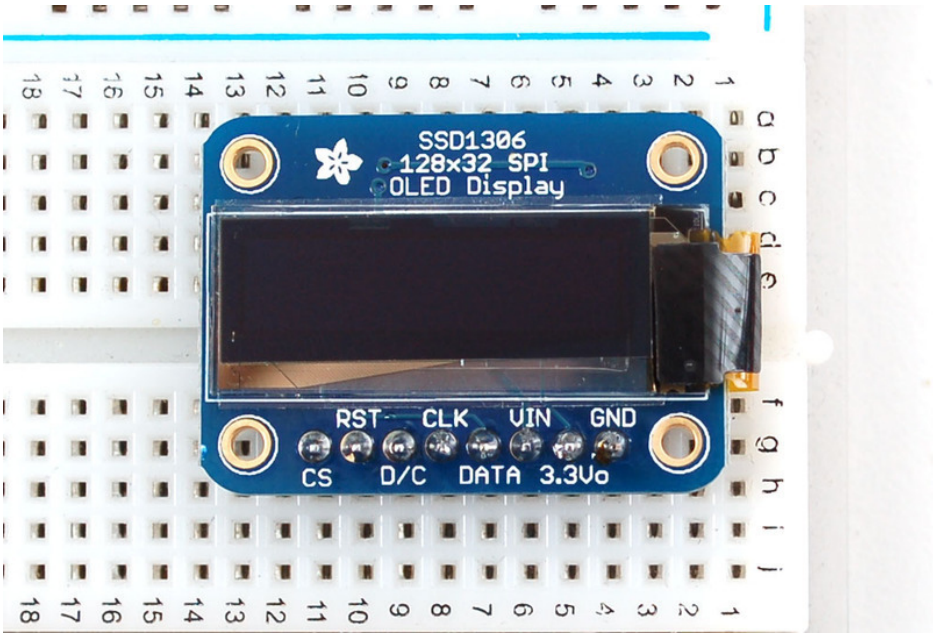
Wiring 128x32 SPI OLED display

128x32 SPI OLED

The 128x32 SPI OLED is very easy to get up and running because it has built in level shifting. First up, take a piece of 0.1" header 8 pins long.



Plug the header long end down into a breadboard and place the OLED on top. Solder the short pins into the OLED PCB.



Finally, connect the pins to your Arduino - **GND** goes to ground, **Vin** goes to 5V, **DATA** to digital **9**, **CLK** to digital **10**, **D/C** to digital **11**, **RST** to digital **13** and finally **CS** to digital **12.**

*(**Note**: If using the display with other SPI devices, D/C, CLK and DAT may be shared, but CS must be unique for each device.)*

This matches the example code we have written. Once you get this working, you can try another set of pins.

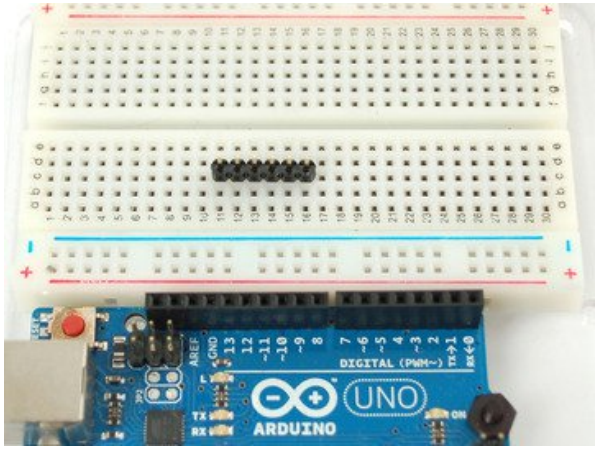
Finally you can run the **File→Sketchbook→Libraries→Adafruit_SSD1306→SSD1306_128x32_SPI example**

If you're using the 128x32 OLED, be sure to uncomment the "#define SSD1306_128_32" in the top of Adafruit_SSD1306.h to change the buffer size

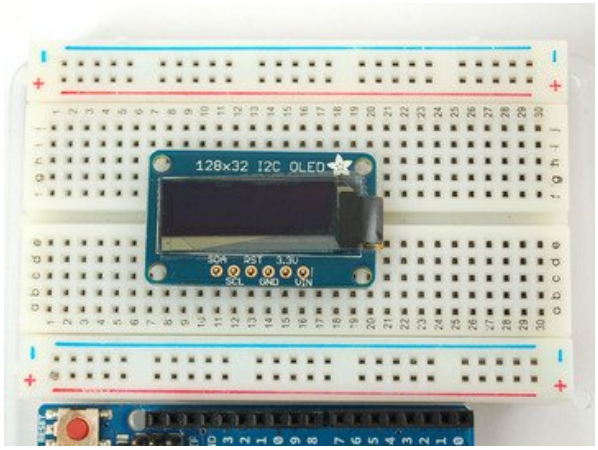
Wiring 128x32 I2C Display

128x32 I2C OLED

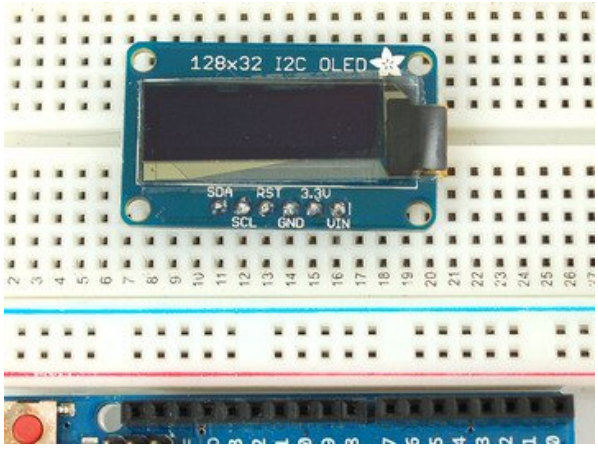
The 128x32 I2C OLED is very easy to get up and running because it has built in level shifting and regulator. First up, take a piece of 0.1" header 6 pins long.



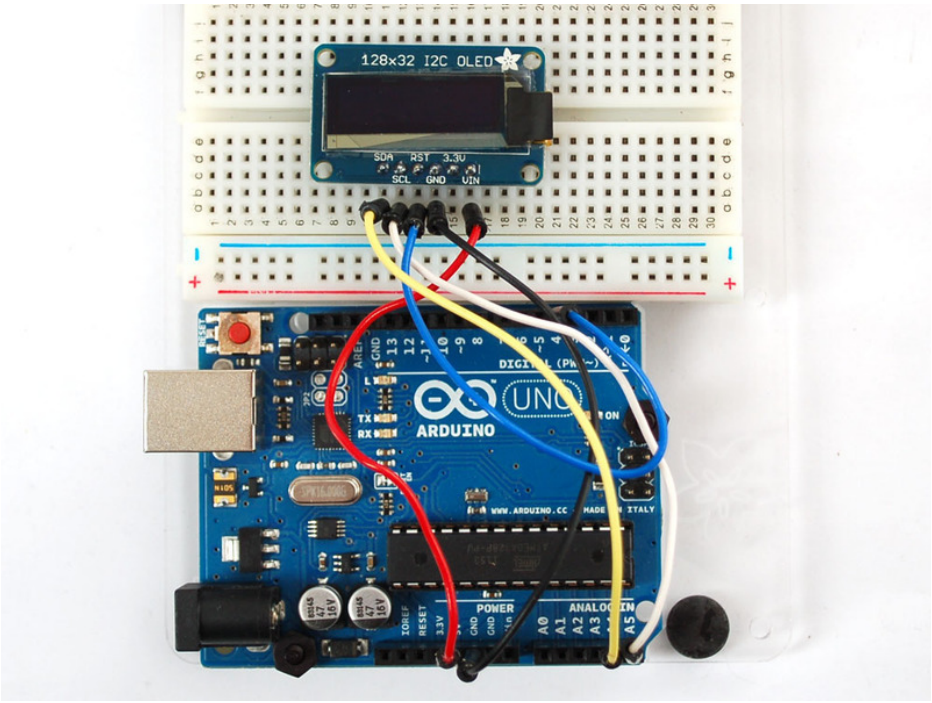
Plug the header long end down into a breadboard



Place the OLED on top



Solder the short pins into the OLED PCB.



Finally, connect the pins to your Arduino

- **GND** goes to ground
- **Vin** goes to 5V
- **SDA** to I2C Data (on the Uno, this is **A4** on the Mega it is **20** and on the Leonardo digital **2**)
- **SCL** to I2C Clock(on the Uno, this is **A5** on the Mega it is **21** and on the Leonardo digital **3**)
- **RST** to digital **4** (you can change this pin in the code, later)

This matches the example code we have written. Once you get this working, you can change the RST pin. You cannot change the I2C pins, those are 'fixed' in hardware

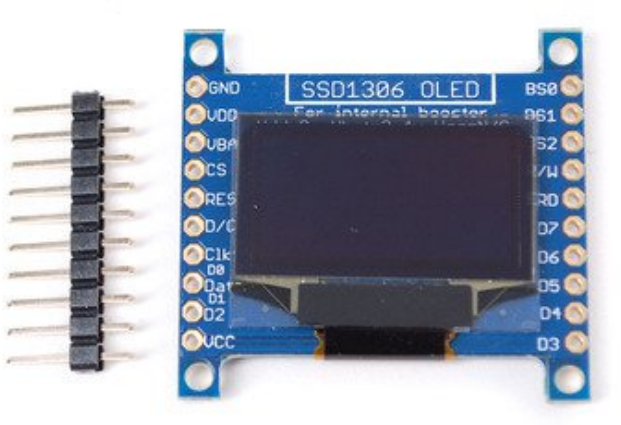
Finally you can run the **File→Sketchbook→Libraries→Adafruit_SSD1306→SSD1306_128x32_i2c** example

Wiring OLD 0.96" 128x64 OLED

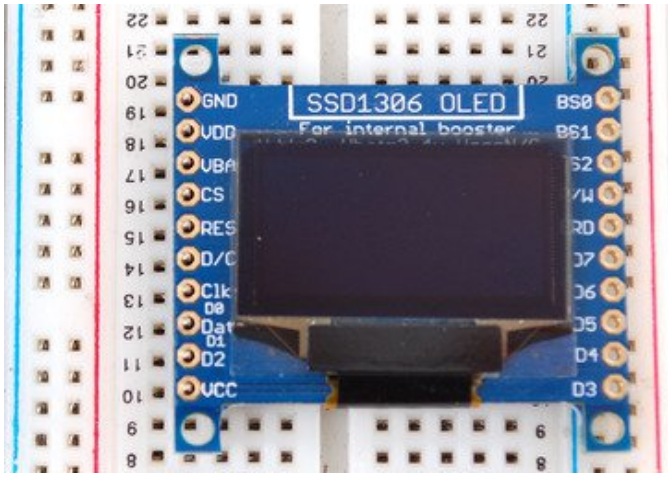
This wiring diagram is only for the older 0.96" OLED that comes with a level shifter chip. If you did not get a level shifter chip, you have a V2.0 so please check out the other wiring tutorial!

128x64 Version 1.0 OLED

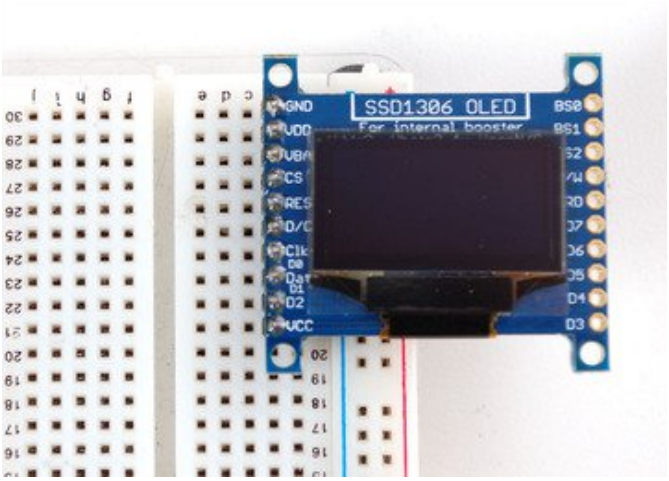
The version 1 128x64 OLED runs at 3.3V and does not have a built in level shifter so you'll need to use a level shifting chip to use with a 5V microcontroller. The following will assume that is the case. If you're running a 3.3V microcontroller system, you can skip the level shifter.



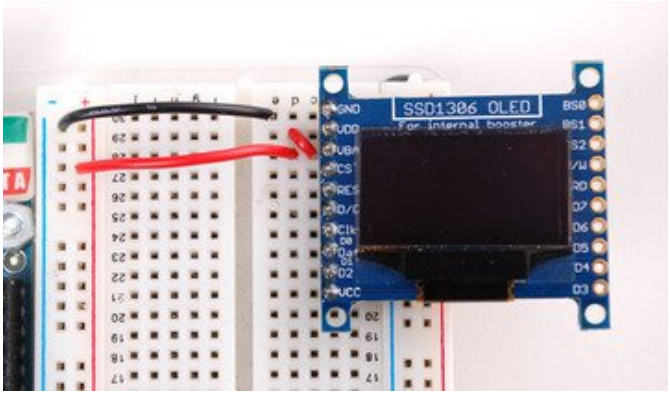
We'll assume you want to use this in a breadboard, take a piece of 0.1" header 10 pins long.



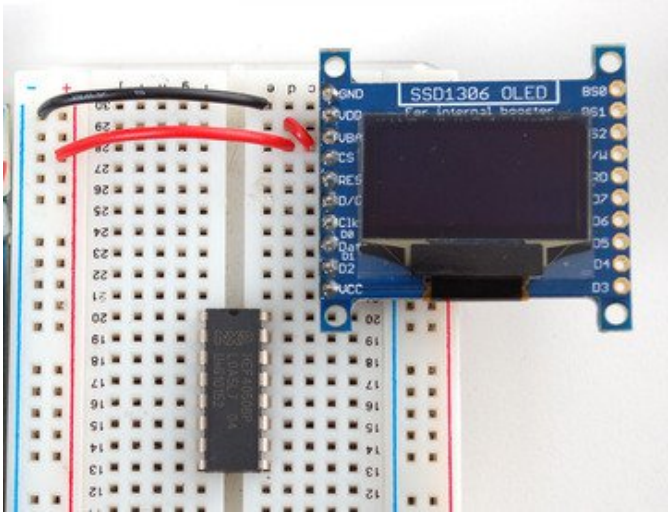
Place the header in a breadboard and then place the left hand side of the OLED on top.



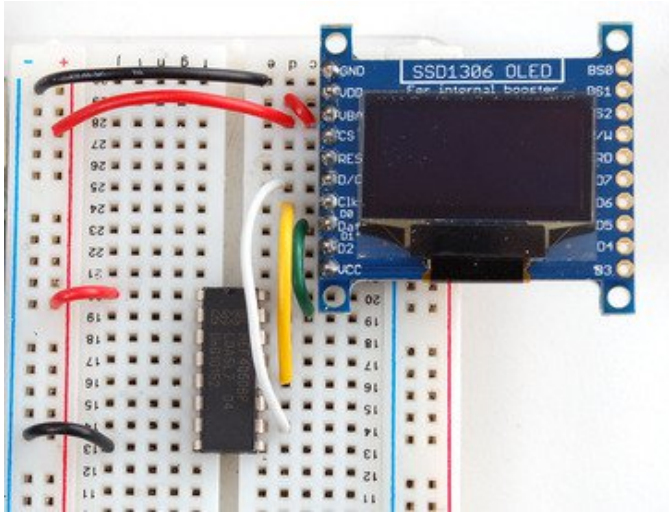
And solder the pins



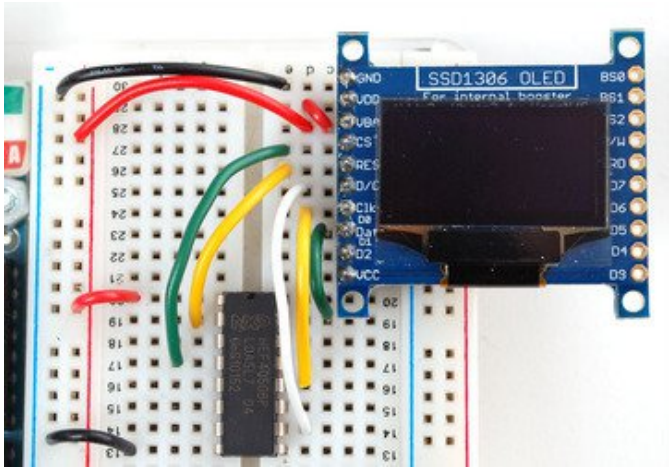
We'll be using the internal charge pump so connect **VDD** and **VBAT** together (they will connect to 3.3V)**.** GND goes to ground.



Place a CD4050 level shifter chip so pin one is at the top.

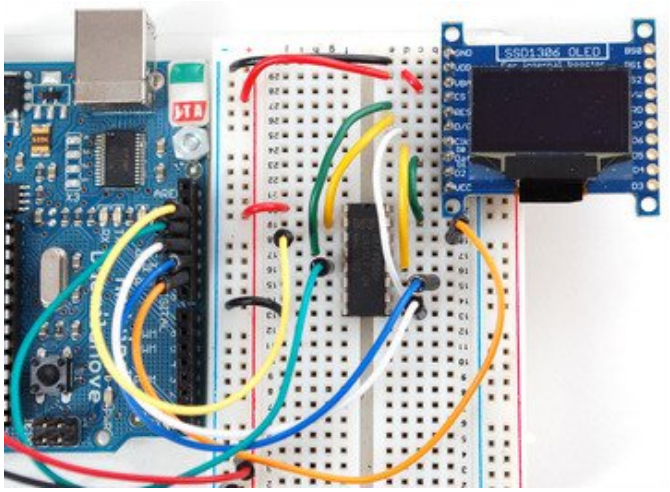


Connect pin 10 to **D/C** pin 12 to **CLK** (SPI clock) and pin 15 to **DAT** (SPI data).



Connect pin 2 to **RES** (reset) and pin 4 to **CS** (chip select). Pin 1 goes to 3.3V and pin 8 to ground.

*(**Note**: If using the display with other SPI devices, D/C, CLK and DAT may be shared, but CS must be unique for each device.)*



You can connect the inputs of the level shifter to any pins you want but in this case we connected digital I/O **13** to pin 3 of the level shifter, **12** to pin 5, **11** to pin 9, **10** to pin 11 and **9** to pin 14. This matches the example code we have written. Once you get this working, you can try another set of pins.

Troubleshooting

Display does not work on initial power but does work after a reset.

The OLED driver circuit needs a small amount of time to be ready after initial power. If your code tries to write to the display too soon, it may not be ready. It will work on reset since that typically does not cycle power. If you are having this issue, try adding a small amount of delay before trying to write to the OLED.

In Arduino, use **delay()** to add a few milliseconds before calling **oled.begin()**. Adjust the amount of delay as needed to see how little you can get away with for your specific setup.

Display is showing burn in on some pixels.

The display can have image burn in for any pixels left on over a long period of time - many days. Try to avoid having the display on constantly for that length of time.

Downloads

Software

You can download our SSD1306 OLED display Arduino library from github (https://adafru.it/aHq) which comes with example code. The library can print text, bitmaps, pixels, rectangles, circles and lines. It uses 1K of RAM since it needs to buffer the entire display but its very fast! The code is simple to adapt to any other microcontroller. You'll also have to install the Adafruit GFX graphics core library at this github repo (https://adafru.it/aJa) and install it after you've gotten the OLED driver library.

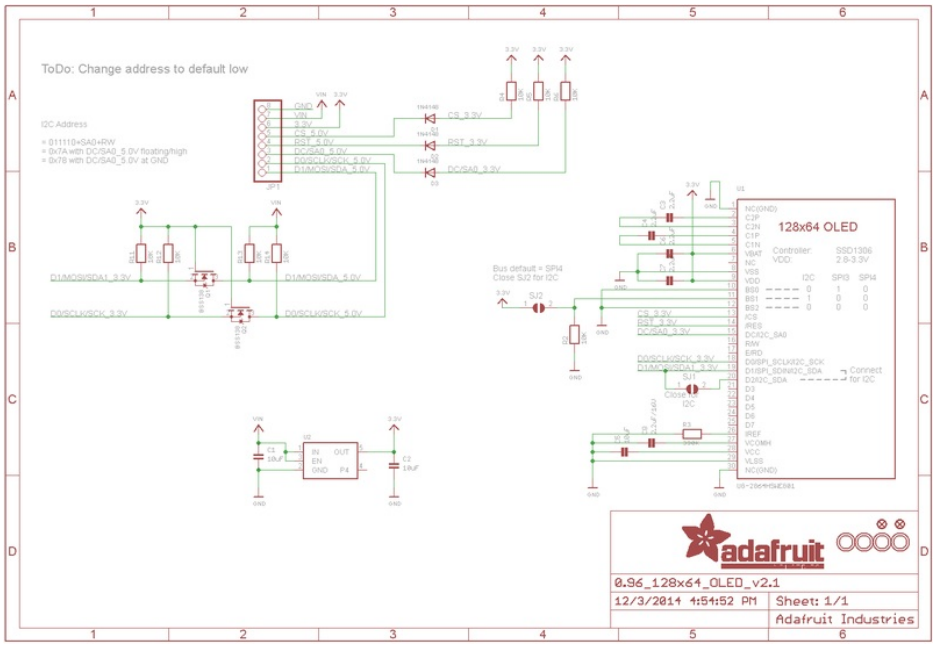
Datasheets

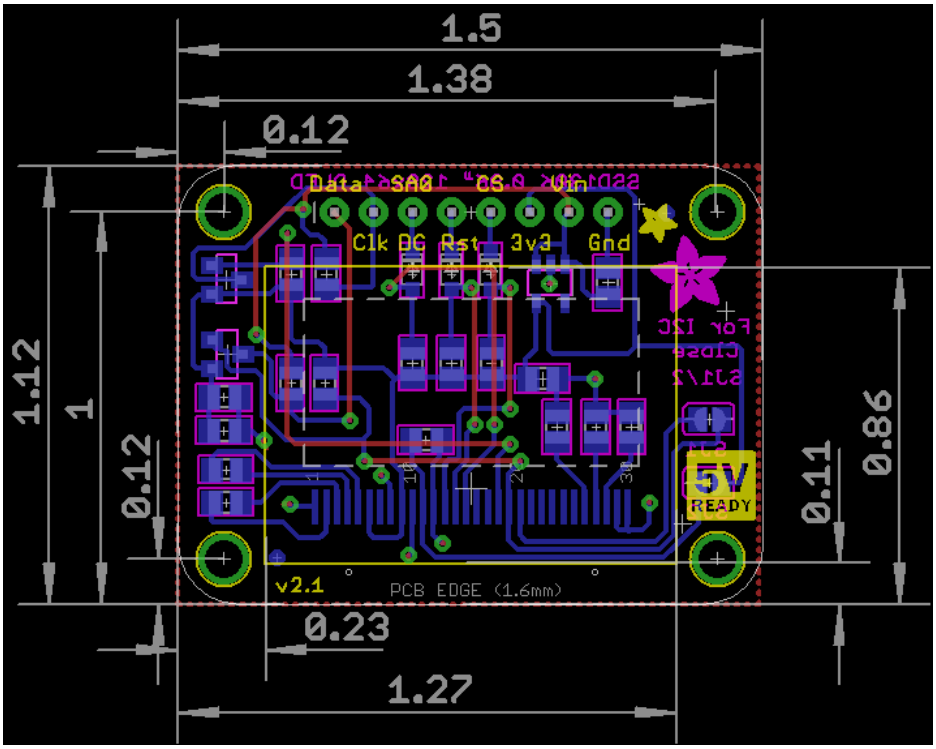
- UG-2864HSWEG01 (https://adafru.it/aJI) Datasheet
- UG-2832HSWEG02 (https://adafru.it/qrf) Datasheet
- UG-2864HSWEG01 (https://adafru.it/wWD) User Guide
- UG-2832HSWEG04 (https://adafru.it/qVA) Datasheet
- SSD1306 (https://adafru.it/aJK) Datasheet

Files

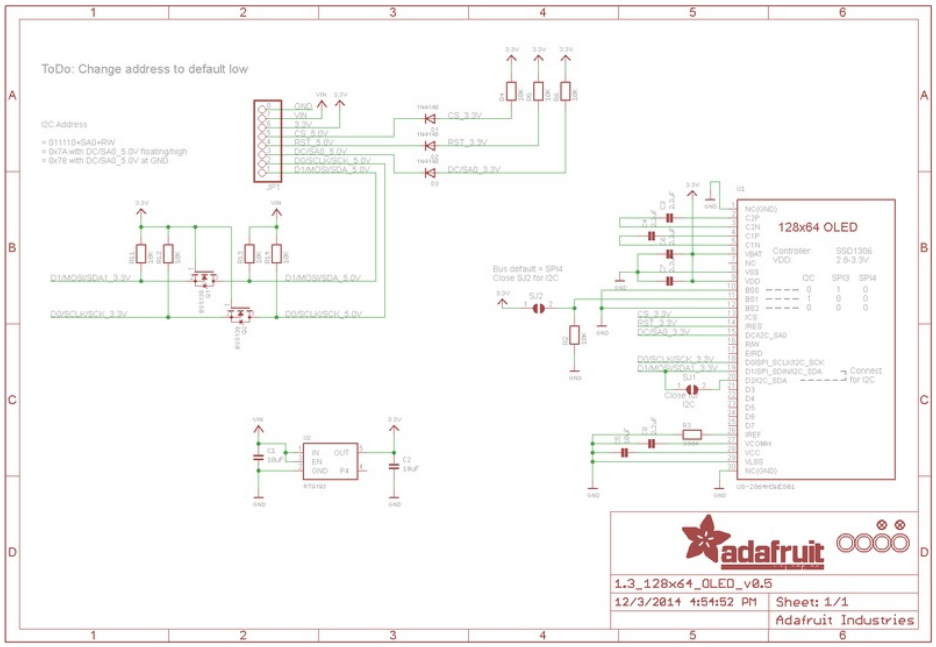
- EagleCAD PCB files for 128x32 0.91" SPI display PCB (https://adafru.it/aJL)
- EagleCAD PCB files for 128x32 0.91" I2C display on GitHub (https://adafru.it/rPF)
- EagleCAD PCB files for 128x64 0.96" display on GitHub (https://adafru.it/aJM)
- EagleCAD PCB files for 128x64 1.3" display on GitHub (https://adafru.it/rJe)
- Fritzing objects available in the Adafruit Fritzing Library (https://adafru.it/aP3)

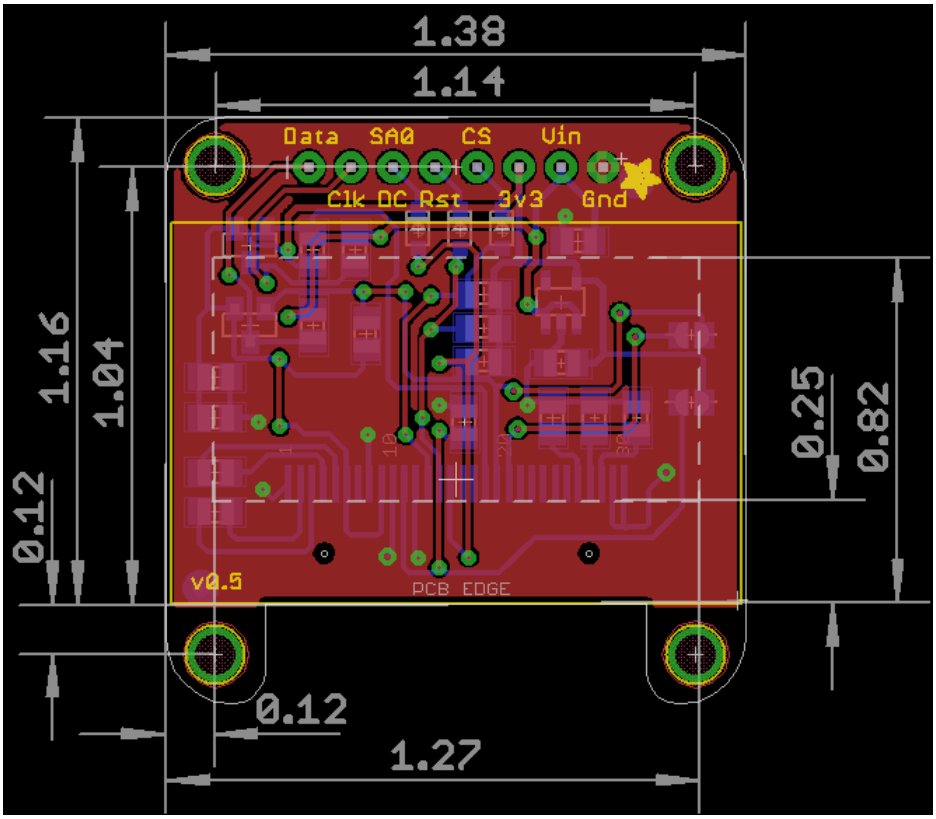
Schematic & Fabrication Print for 0.96" OLED



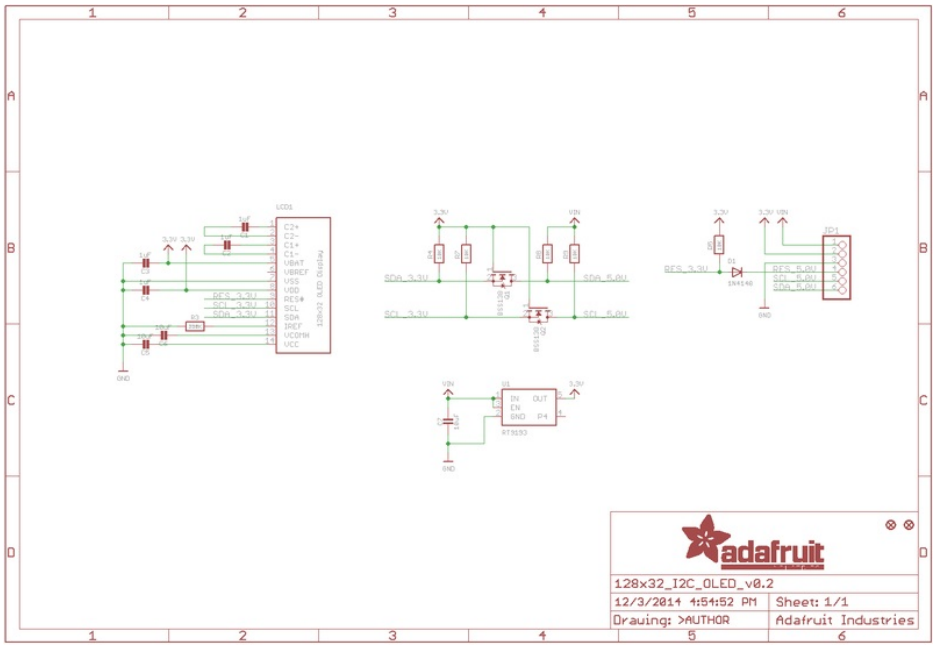


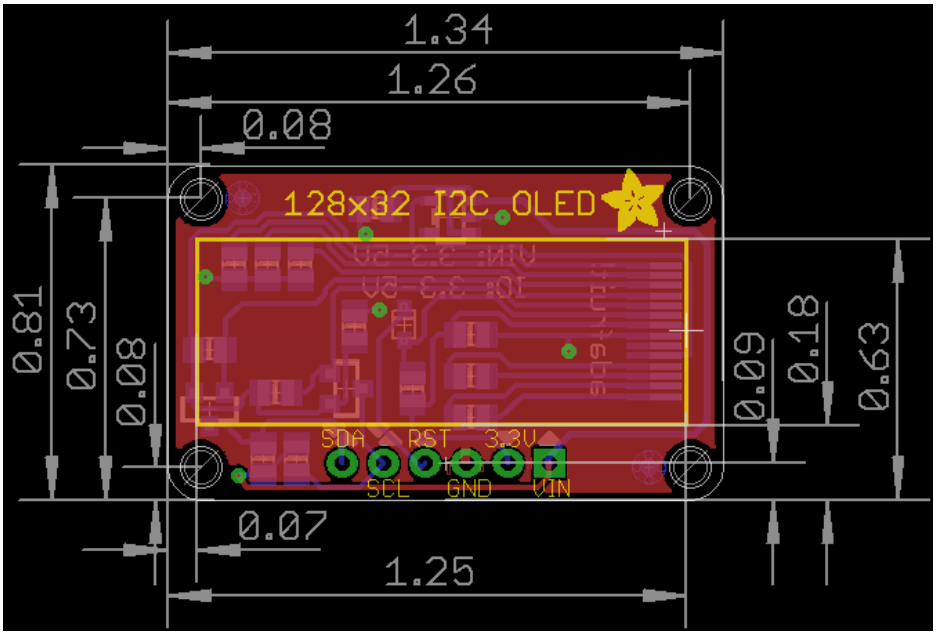
Schematic & Fabrication Print for 1.3" OLED



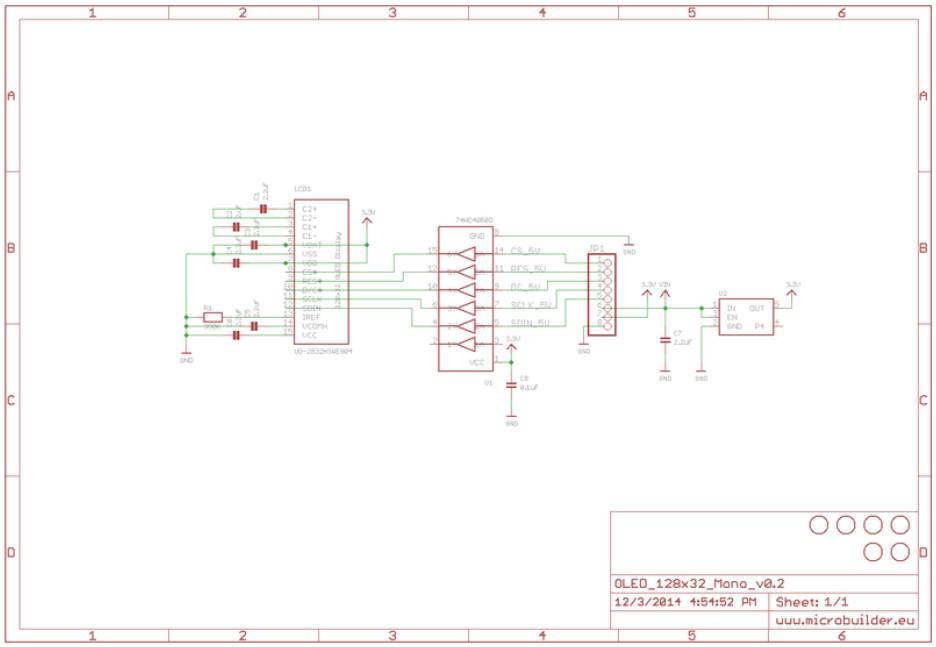


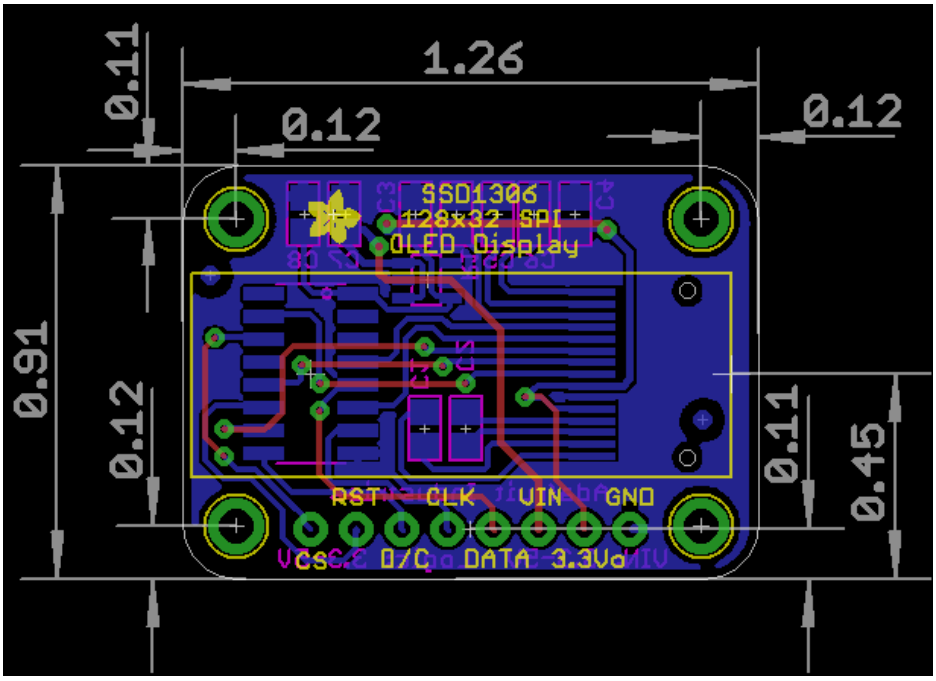
Schematic & Fabrication Print for 0.91" 128x32 I2C





Schematic & Fabrication Print for 0.91" 128x32 SPI




0.11
1.26
0.12
0.12
SSD1306
128x32 SPI
OLED Display
0.91
0.12
0.11
0.45
RST
CLK
VIN
GND
VCC
D/C
DATA
3.3Vo