Open Created 2 days ago by **Lingdong Huang** Developer Close issue

OLED Tips & Tricks

I got my OLED display working well but it seems that there were quite a few interesting problems I had to figure out. I thought I'd share the info here in case it might be helpful for others using the OLED.

Footprint mismatch:

If you look at Neil's OLED on the course website:
http://academy.cba.mit.edu/classes/output_devices/OLED/hello.SSD1306.45.jpg

The pins go **VCC** GND SCL SDA.

But if you look at the OLED's available in the CBA section, they go GND **VCC** SCL SDA

The internet says that due to an inventory problem, either versions exist in the world:
https://github.com/squix78/esp8266-fritzing-parts/issues/6

If you're soldering the OLED directly onto the board (instead of using jumpers), maybe look at the physical OLED first.

SDA and SCL pins

Due to my ignorance, I didn't know before that there're dedicated pins on microcontrollers for hardware I2C (I thought they're always software emulated). You can check the Microcontrollers data sheet to figure out.

On SAMD21E one set is PA16,PA17, the other set is PA08,PA09.

Telling Arduino Wire library to use another set of pins

The Wire library in Arduino will default to one set of SDA and SCL pins available on the microcontroller. If you're unfortunate like me and made a board that connects to the other set, you can force Arduino to use that, by doing:

```
#define HARDWARE_I2C_USE_THE_OTHER_SET_OF_PINS

#include <Wire.h>
#ifdef HARDWARE_I2C_USE_THE_OTHER_SET_OF_PINS
    TwoWire WIRE(&PERIPH_WIRE1,PIN_WIRE1_SDA, PIN_WIRE1_SCL);
#else
    #define WIRE Wire
#endif
```

Then, throughout the rest of your program, refer to `Wire` with `WIRE`, e.g. `WIRE.beginTransmission()`. To change back to the default pins, comment out the first line.

Software I2C

If you connected the OLED to non-dedicated SDA/SCL pins, you can use software simulated I2C. If I'm not mistaken, Neil's C code seems to do that:
http://academy.cba.mit.edu/classes/output_devices/OLED/hello.SSD1306.45.c

In Arduino IDE, you can use the SoftWire library that does it for you:
https://www.arduino.cc/reference/en/libraries/softwire/

You also need a dependency called AsyncDelay, both can be installed from the IDE menu GUI.

You can find my examples using SoftWire to control the OLED at the bottom of this issue.

Adafruit SSD1306 Library Incompatibility

I believe that the Adafruit SSD1306 library does not work for the OLED's in cba section. You can see the failure in action in issue 102 (only visible to cba section, sorry).

So I fixed up the library, and the patch can be downloaded below:

Adafruit_SSD1306.cpp

Place the file in the Arduino library folder, replacing the original. On Macs, the path is `~/Documents/Arduino/libraries/Adafruit_SSD1306/Adafruit_SSD1306.cpp`

I don't know what the original library got wrong. I just deleted sections that seemed problematic and replaced it with my own implementation.

You can run their default example now to see it working. There might be some glitches with scrolling feature, which I might fix that later.

Also if you're using other brands of OLED's or different resolutions than the 128x64 in the cba section, my patch will probably break (make a backup/duplicate first).

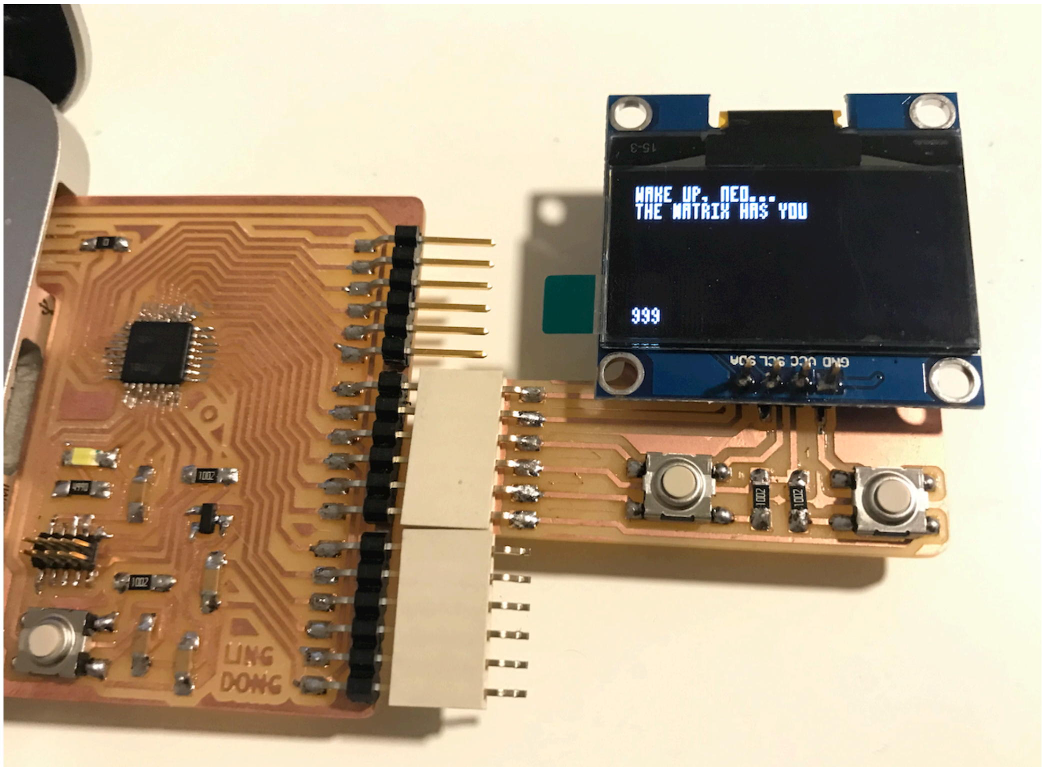
Drawing stuff old fashioned way

If you're like me and don't like libraries (especially broken ones), you can draw stuff by directly sending the commands via I2C. The SSD1306 data sheet (linked on course website) has detailed documentation.

I modified Neil's C code so it works with Arduino Wire/Softwire. You can download a minimal example below:

OLED.MIN.ino

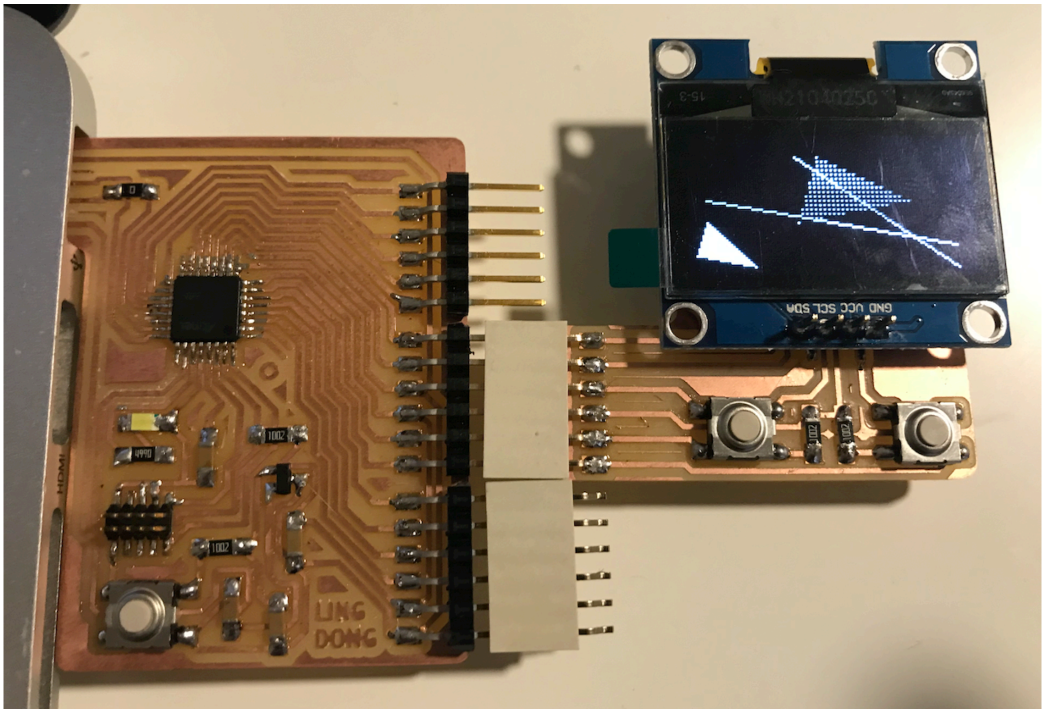
If you run it it just draws some simple text and a counter, pretty much like Neil's demo:



It includes options to use software/hardware I2C, and which I2C pins to use; just fiddle with the macros on the top of the file.

I also made another example, that rasterizes some lines and triangles. It uses two buffers to keep track of which regions are "dirty", but if your microcontroller doesn't have much storage one buffer also works.

📎 OLED.DRAW.ino



I also made some 3D renderers but the code gets quite cryptic, let me know if you're interested...

Add a button

Doing graphics involves dealing with a lot of loops and buffers, and it seems that it is possible to write code so buggy, that corrupts some data on the microcontroller or trap it in a limbo state. Hardware reset button makes the code restart, but if the bug is at the very beginning the microcontroller gets stuck immediately again. It happened to me, and when I tried to push new programs using Arduino IDE, the board was so busy getting stuck that it didn't respond. I had to time it perfectly: within one or two seconds of pressing "upload" in IDE, press the reset button, so the microcontroller restarted but hadn't reached the buggy code yet. It took multiple trials...

If you always have an Atmel-ICE handy I think you could just re-bootload it. But if not, I think that a good idea is to add an extra button, and run the main program logic only after the button is pressed.

UPDATE: Better solution from @ampennes :

> Depending on your D21 bootloader you might be able to force it to stay in the bootloader and not execute your "buggy program" Jake and I tend to use the Gemma M0 bootloader and if you double click the reset button or hold reset down while plugging it in then you'll stick in the bootloader and eliminate the guessing the timing issue. I'm not sure if that's implemented in the generic D21 bootloader but it's worth a shot.

> Adafruit bootloaders here and I'm pretty sure you want "bootloader-gemma_m0-v3.13.0.bin"

That's all the things I can think of now. Perhaps to people with prior experience these are too basic and not worth mentioning, but for me it took some work to figure out. So I hope it might be useful for some others in a similar situation. Thanks for reading.

📎 OLED.MIN.ino 📎 OLED.DRAW.ino

(Originally posted in the cba section, reposted here at request of @ampennes :)

Drag your designs here or click to upload.

Linked issues **0**

Lingdong Huang @lingdong mentioned in issue classes/863.21/CBA/cbasite#104
2 days ago