An Arduino-Based RGB Color Sensor

David R. Brooks, *Institute for Earth Science Research and Education*

For environmental monitoring, especially from space, the color of natural surfaces contains information about the nature and possibly the health of that surface. For example, the greenness of a vegetated surface as viewed from above (often from space) may be related to the health of that vegetation. Is it possible to get useful information about surfaces with a ground-based Arduino system that senses light separated into its component colors?

There are two basic kinds of red/green/blue (RGB) color sensors for Arduinos. One uses a white light source (an LED) to illuminate a nearby surface and measures the light reflected from that artificially lit surface; see, for example, the TCS34725 breakout board from adafruit.com. The other kind simply responds to light reflected naturally from a surface. For a system that analyzes the color of natural or manmade surfaces, typically as observed from above, the second kind of sensor is what's needed.

The ISL29125 I2C sensor is available on a breakout board from sparkfun.com (SEN-12829, $8.50). The sensor's operating current is only 56 µA. It's a 3.3 V power/logic board, so it needs to be used with a 3.3 V Arduino. (An UNO or other 5-V Arduino will have a 3.3 V power output pin but would still need a separate logic level converter board to communicate with the sensor.) For this project I used a 3.3 V Pro Mini with a 3.3 V FTDI microB USB board from Sparkfun (DEV-09873, $15.50), which is needed because Pro Minis don't have an on-board USB connector. Pro Minis are available from Sparkfun (DEV-11114, $10, without headers) and from many other sources (often including headers) for much less than that. Pro Minis are supported as part of a standard Arduino IDE installation, so there's no need to install additional board support. Make sure you have a 3.3 V board and that you specify it correctly when you use Tools to select the board:

```
Board: "Arduino Pro or Pro Mini"
Processor: "ATmega328P (3.3V, 8 MHz)"
```

Pro Minis are low-power boards that can be powered with an external DC supply up to about 12 V through the RAW/GND pins; 3 or 4 AA or AAA batteries in series and even 3.7 V LiPo batteries which produce about 4.2 V when fully charged will work.

Here's a pin-out diagram for the Pro Mini with an FTDI board shown in the upper right-hand corner; its female header strip is on the back of the board. Note that the SDA/SCL pins required for I2C devices aren't available from the header pins when installed on a breadboard; separate wires must be soldered onto the board (yellow for SDA and blue for SCL for the layout shown below). Some versions will label these connectors (labeled 1 and 2 in the pin-out

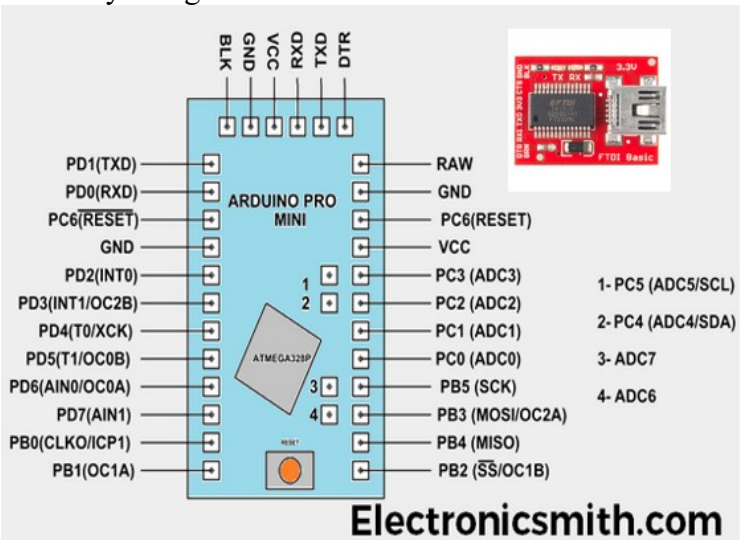
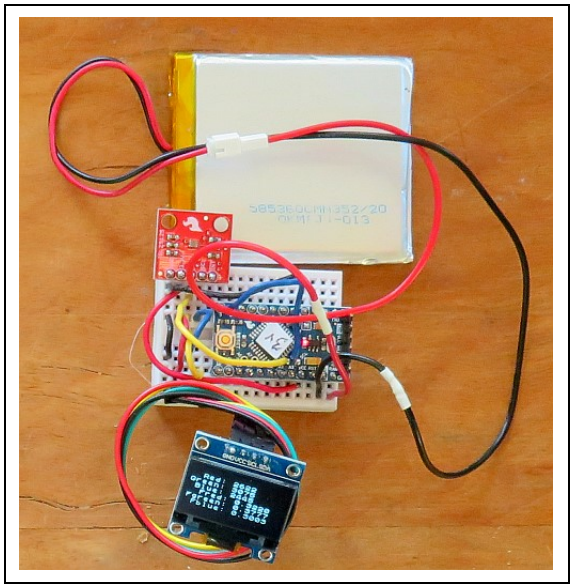


diagram) as SCL and SDA, but others will label them just as ADC5/ADC4. Note that in addition to headers to insert the Pro Mini into a breadboard, a 6-pin male header is also required at the top end of the board as shown here, for attaching the 3.3 VFTDI board required for communicating with your computer. When you connect the FTDI board, make sure that the BLK mark on the FTDI corresponds with the BLK mark on the Pro Mini. Finally, note that some "off shore" Pro Minis may have a different pin arrangement, so be careful.



Here's a mini breadboard layout for this system, running with a LiPo battery. The OLED is connected between its header pins and the breadboard with M/F jumpers so you can orient the sensor board (the red square) to point at whatever you like and still be able to read the display. The little 3V tag on the Pro Mini is to mark it as a 3.3 V version rather than a 5 V version, identical in appearance.

The ISL29125 has a standard I2C interface and a library with code examples available from Sparkfun. For the IZOKEE OLED you will also need to install the U8g2 library from the IDE's Sketch→Include Library→Library Manager, of which U8x8lib is a text-only subset.

The basic ISL29125 default output is in the form of three unsigned integer values for the R, G, and B sensors. Here's some code that displays these values, adds the values for all three sensors, calculates the fraction of the total from each of the three sensors, and displays the outputs on the OLED. What can these data tell us about a surface? Will this sensor work outdoors where surfaces are illuminated by bright sunlight? The datasheet at https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/isl29125.pdf claims that this sensor *"is designed to reject IR in light sources allowing the device to operate in environments from sunlight to dark rooms."* The upper limit on incident light is given as 10,000 lux, which isn't nearly enough for direct noontime summer sunlight (~100,000 lux) but might still be adequate for light reflected from vegetated surfaces. Are there other ways to interpret relationships among the three sensors? This project is just a starting point and these questions need further investigation!

```
/*ISL29125_OLED.ino, D. Brooks, January 2022
  A basic test sketch is available in the Sparkfun library.
  This code adds some additional calculations and
  an IZOKEE 0.96" I2C OLED display for output.
*/
#include <Wire.h>
#include "SparkFunISL29125.h"
SFE_ISL29125 RGB_sensor;
#include "U8x8lib.h"
U8X8_SSD1306_128X64_NONAME_HW_I2C u8x8(U8X8_PIN_NONE);
unsigned int red,green,blue,total;
float Fred,Fgreen,Fblue;
const long int DelayTime=2000L;
void setup(){
  Serial.begin(9600); u8x8.begin();
```

```
  u8x8.setFont(u8x8_font_chroma48medium8_r);
  if (RGB_sensor.init()) {
    Serial.println("Sensor Initialization Successful\n\r");
  }
}
void loop() {
  red=RGB_sensor.readRed(); green=RGB_sensor.readGreen();
  blue=RGB_sensor.readBlue(); total=red+blue+green;
  Fred=(float)red/total; Fgreen=(float)green/total;
  Fblue=(float)blue/total;
  Serial.print("Red: "); Serial.println(red);
  Serial.print("Green: "); Serial.println(green);
  Serial.print("Blue: "); Serial.println(blue);
  Serial.print("Fred: "); Serial.println(Fred,4);
  Serial.print("Fgreen: "); Serial.println(Fgreen,4);
  Serial.print("Fblue: "); Serial.println(Fblue,4);
  u8x8.setCursor(0,1);
  u8x8.print("  Red: "); u8x8.println(red);
  u8x8.print("Green: "); u8x8.println(green);
  u8x8.print(" Blue: "); u8x8.println(blue);
  u8x8.print("  Fred: "); u8x8.println(Fred,4);
  u8x8.print("Fgreen: "); u8x8.println(Fgreen,4);
  u8x8.print(" Fblue: "); u8x8.println(Fblue,4);
  delay(DelayTime);
  u8x8.clearDisplay();
}
```