



Light Meter

Created by Dan Malec



Last updated on 2018-08-22 03:32:14 PM UTC

Guide Contents

Guide Contents	2
Overview & Parts	3
Code & Wiring	4
Downloads	7

Overview & Parts

I recently needed to measure how different materials affect light transmission for a gardening project. This seemed like a perfect fit for an Arduino project based around the TSL2561 Light sensor.

It can display the raw "Visible light" count from the sensor and then the calculated Lux.

I used the following parts in this project:

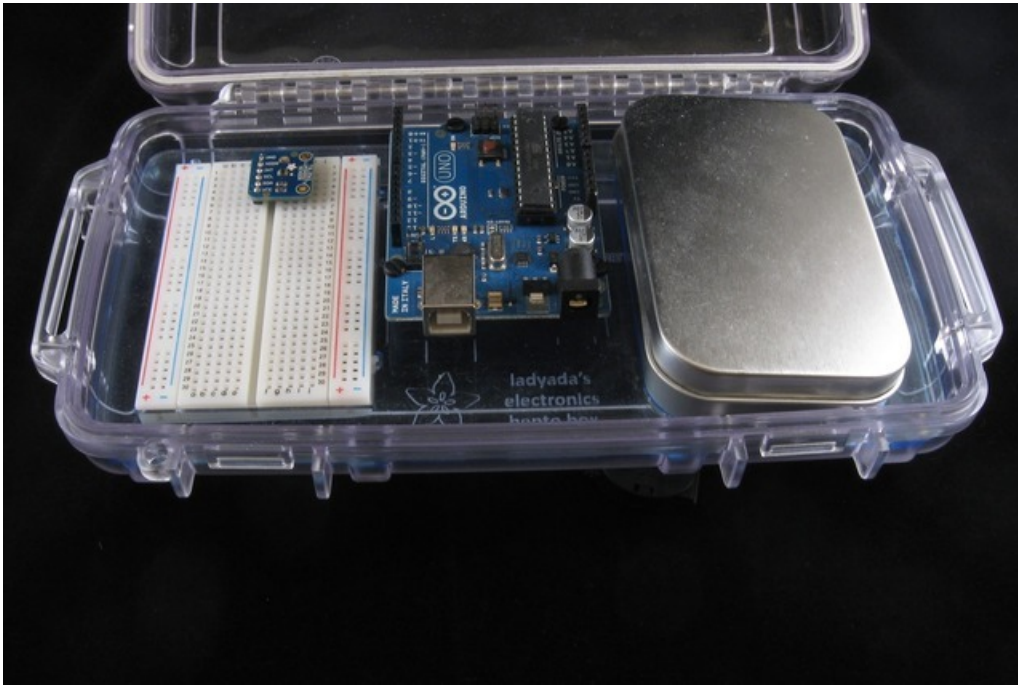
- [Arduino Uno \(http://adafru.it/50\)](http://adafru.it/50)
- [TSL2561 Digital Light Sensor \(http://adafru.it/439\)](http://adafru.it/439)
- [16×2 LCD \(http://adafru.it/181\)](http://adafru.it/181)
- [LCD I2C / SPI Backpack \(http://adafru.it/292\)](http://adafru.it/292)

Optional (but definitely helps keep everything tidy):

- [Lady Ada's Bento Box \(http://adafru.it/765\)](http://adafru.it/765)

Code & Wiring

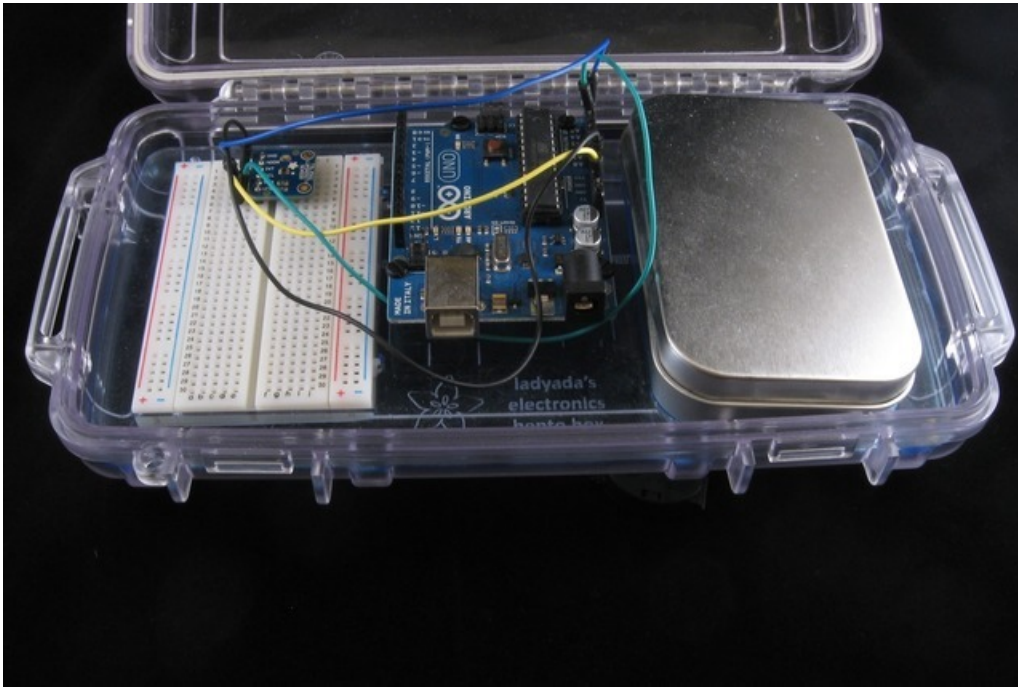
First up is plugging the light sensor into the breadboard.



The next step is connecting the sensor to the Arduino:

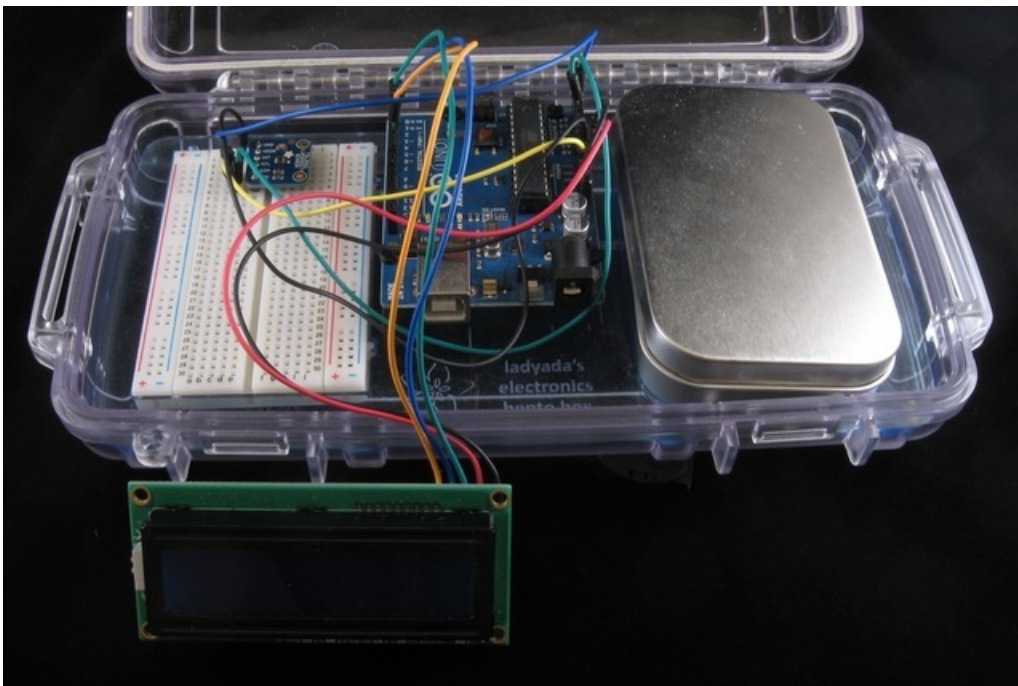
- Sensor **GND pin** - black wire – Arduino **ground pin**
- Sensor **VCC pin** - yellow wire – Arduino **3.3v pin**
- Sensor **SCL pin** - green wire – Arduino **analog pin 5** (i2c clock line)
- Sensor **SDA pin** - blue wire – Arduino **analog pin 4** (i2c data line)

If you're using only the TSL2561 on your Arduino and no other I2C devices, its fine to connect it directly. If you have other I2C devices that run on 5V connected at the same time [you should use a i2c-compatible level shifter such as this one \(http://adafruit.it/757\)](http://adafruit.it/757) to shift both the SCL and SDA data. Connect the **HV** pin to 5V, the **LV** pin to 3.3V, grounds to ground, and connect **A1** and **B1** channels to the TSL SDA/SCL pins and **A2** and **B2** to the matching Arduino I2C pins

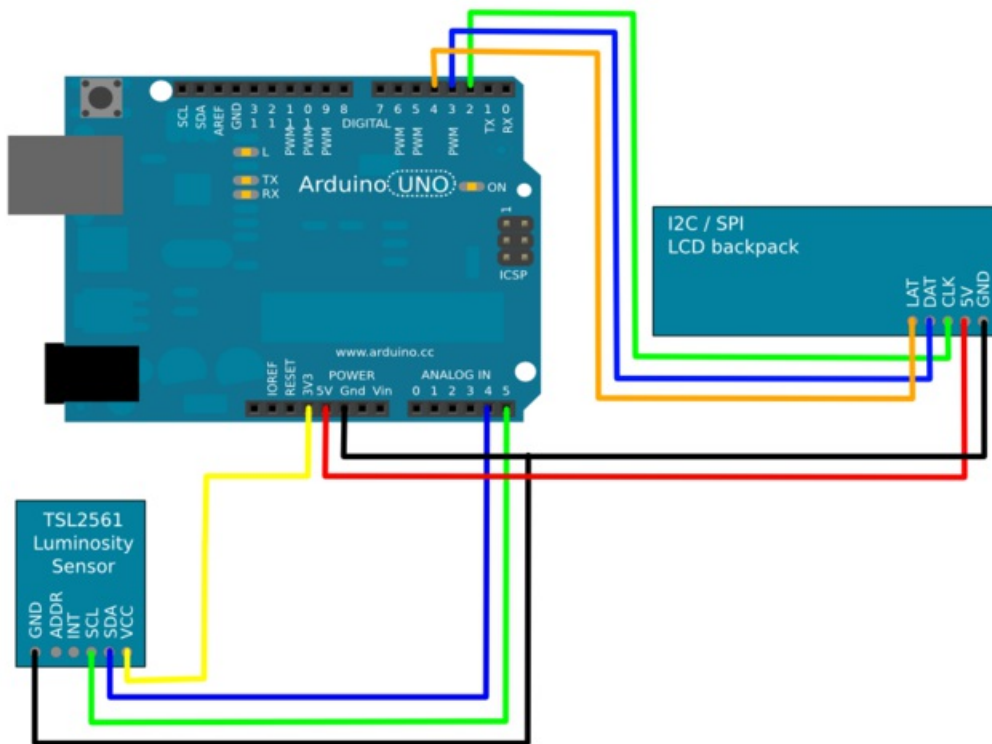


The LCD can now be added to the circuit. Since this project uses SPI mode to talk to the LCD, make sure the SPI solder enable jumper has been soldered.

- LCD backpack **GND pin** - black wire – Arduino **ground pin**
- LCD backpack **5V pin** - red wire – Arduino **A5V pin**
- LCD backpack **LAT pin** - orange wire – Arduino **digital pin 4** (SPI latch pin)
- LCD backpack **DAT pin** - blue wire – Arduino **digital pin 3** (SPI data pin)
- LCD backpack **CLK pin** - green wire – Arduino **digital pin 2** (SPI clock pin)



With all components added, the block diagram of the circuit looks like this:



Arduino Uno image courtesy of Fritzing

For my project, I'm gathering data at reasonably high light levels; so, I went with the following settings:

```
tsl.setGain(TSL2561_GAIN_0X);
tsl.setTiming(TSL2561_INTEGRATIONTIME_13MS);
```

If you are measuring low light levels, you may want to adjust these two lines accordingly:

```
tsl.setGain(TSL2561_GAIN_16X);
tsl.setTiming(TSL2561_INTEGRATIONTIME_402MS);
```

An additional area to highlight is how the sketch is printing the light level values:

```
snprintf_P(output_buffer, 6, PSTR("%5d"), (full_spectrum - ir_spectrum));
```

snprintf_P is a variant of **sprintf** that adds a couple of nice features. The 'n' indicates that you can specify a maximum number of bytes to write into the buffer; this helps protect against accidental buffer overruns. The '_P' indicates that the format string is read from program memory; this helps conserve RAM. In the invocation above, I'm using the companion macro **PSTR()** to keep the format string parameter in program memory.

Downloads

Download the latest code on [GitHub \(https://adafru.it/aUJ\)](https://adafru.it/aUJ)