*Projects and Stuff*

Chameleon

Project Log

Contents

[2012/05/06 3](#_Toc324170558)

[2012/05/07 3](#_Toc324170559)

# 2012/05/06

I began setting up the bare-bones of the project, starting with creating a new Git repository. I also started simple documentation about what the project goals are.

This project will be based upon the work that I’ve done on other capacitive-sensing projects. The main benefit of capacitive sensors over other methods, like IR LEDs and Diodes/transistors, is that the sensor itself has zero additional cost. While other sensor types require an actual sensor and maybe even an emitter of some sort, which costs money, capacitive sensors are just traces on a PCB. The argument could be made that the resistors and capacitors used with the capacitive sensors also cost money, but in production quantities, it’s an order of magnitude cheaper. All of the external components for a capacitive sensor typically cost less than one cent total in production quantities.

I also began developing the schematics today. Because much of this project is based upon my previous work on capacitive sensors, much of the hard work and research is already done. So placing the major components was simple.

I’m considering which LED driver to use. Since there are 5 RGB LEDs (15 LEDs total), I’m thinking a 16-bit LED driver would be perfect. It will have to be a chip that can adjust the current or PWM duty cycle of each LED separately. I can also add a status LED to the bottom of the board with the remaining output.

# 2012/05/07

I’ve continued working through the schematic today, on a plane ride to Arizona. I’ve switched the display driver from the AS1107 to the PCA9635, which is better suited to optimally driving the smaller number of LEDs. This may not be the final decision on the driver, though, depending on price.

Each LED needs a current limiting resistor. Here’s how I determined the value for each.

**RED**

***Input Voltage = 5V***

***Forward Voltage = 1.8V*** (actually, it ranges from 1.8V to 2.6V, but a drop of only 1.8V means greater current, so I want to choose a resistor value based on this. If I used the 2.6V value, and the actual drop were 1.8V, then I might burn up my LED because the current limiting resistor wasn’t of a high enough value)

***My Desired Maximum Current = 15mA*** (or 0.015A for our calculation) (each LED can handle 30mA max, but 15mA should provide plenty of brightness. I may change this to 20mA if the prototype isn’t bright enough)

As you may recall from Ohm’s law,

To find the value of the resistance needed, you can deduce that

But you have to recall that E in this case is reduced by the forward voltage of the LED, so

And now we can plug everything in:

Since 213.3 Ω isn’t a common resistor value, we’ll go with 220 Ω, which is slightly higher, but inconsequentially so.

**GREEN & BLUE (same forward voltage)**

***Input Voltage = 5V***

***Forward Voltage = 2.8V*** (Actual values are 2.8V-3.6V)

***My Desired Maximum Current = 15mA*** (or 0.015A for our calculation)

Following the same as above, we get

And now we can plug everything in:

Since 146.6 Ω isn’t a common resistor value, we’ll go with 150 Ω, which is slightly higher, but inconsequentially so.

I also added the ATTiny44 as the brains for this device. The ATTiny44 is completely swappable with the ATTiny84 if you need more memory (8k vice 4k). It’s a microcontroller I’ve used in past projects as well. It is compatible with Atmel’s QTouch library, and can control up to 4 QMatrix capacitive sensors. In this case, I only need to control 1.

While I’ve currently got the SPX29150T-L-5-0/TR in the schematic as the regulator, I’m going to have to change this. Since the battery will be providing a nominal 4.2V, I’m considering using a boost converter to operate at 5V.

One thing you’ll notice looking at the schematics is that there are several capacitors associated with the LEDs. The 0.1uF Capacitors are intended to keep the LEDs from interfering with the capacitive sensor when they’re switching on and off. These may not be required, because the PCA9635 “LED outputs programmable to logic 1, logic 0 or ‘high-impedance’ (default at power-up) when ~~OE~~ is HIGH”. I’ll have to read more into this, but as long as the outputs are pulled high or low (and not high-impedance), it shouldn’t have negative effects on the sensor. The other key thing is to keep the LEDs as far away from the sensor as possible.