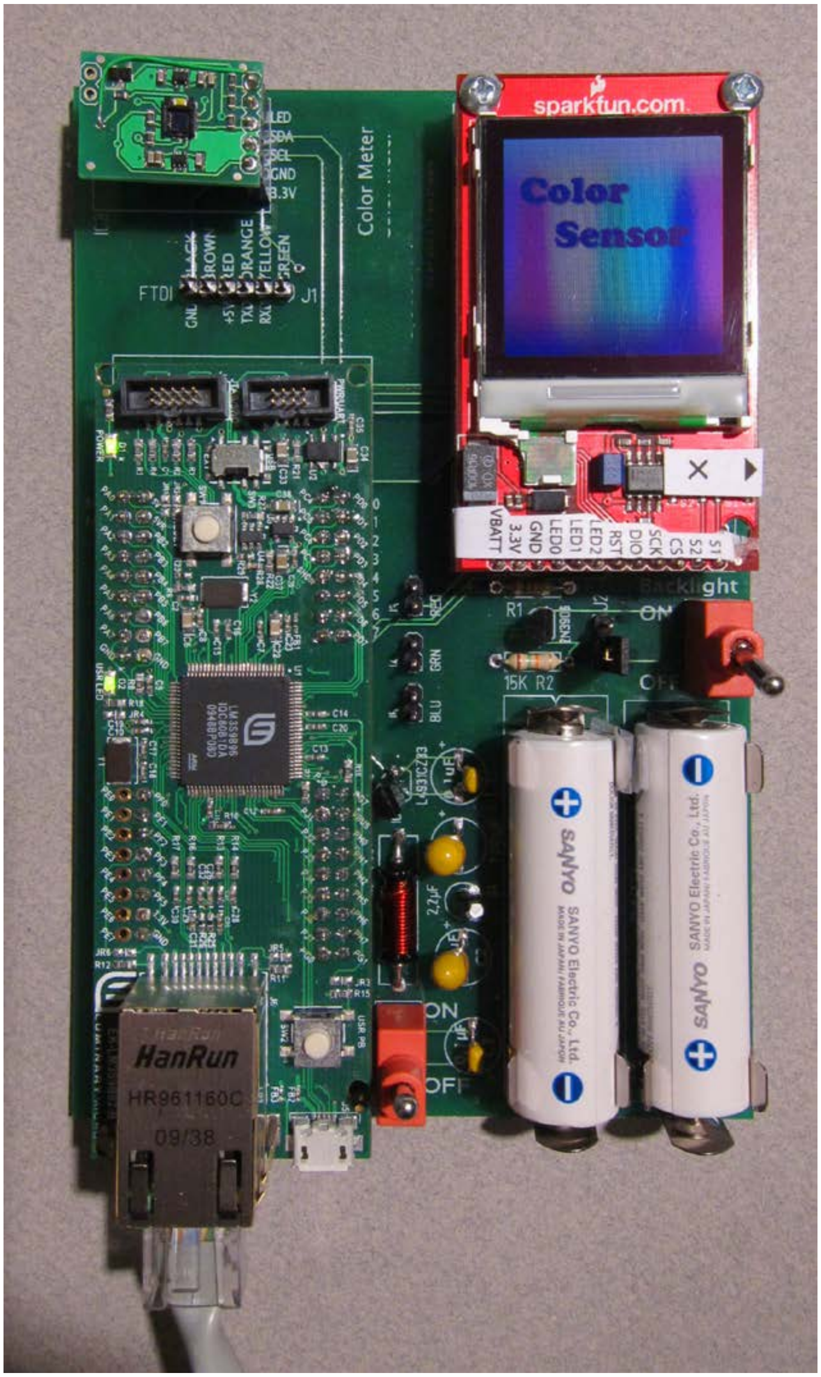
The Color Sensor

John Peterson
June 2010



Color

Color is a notoriously subjective experience. While people can be very particular about the colors they use for lighting, designs, or environments, human perception of color is very subjective.   This makes it difficult to accurately measure and understand color.

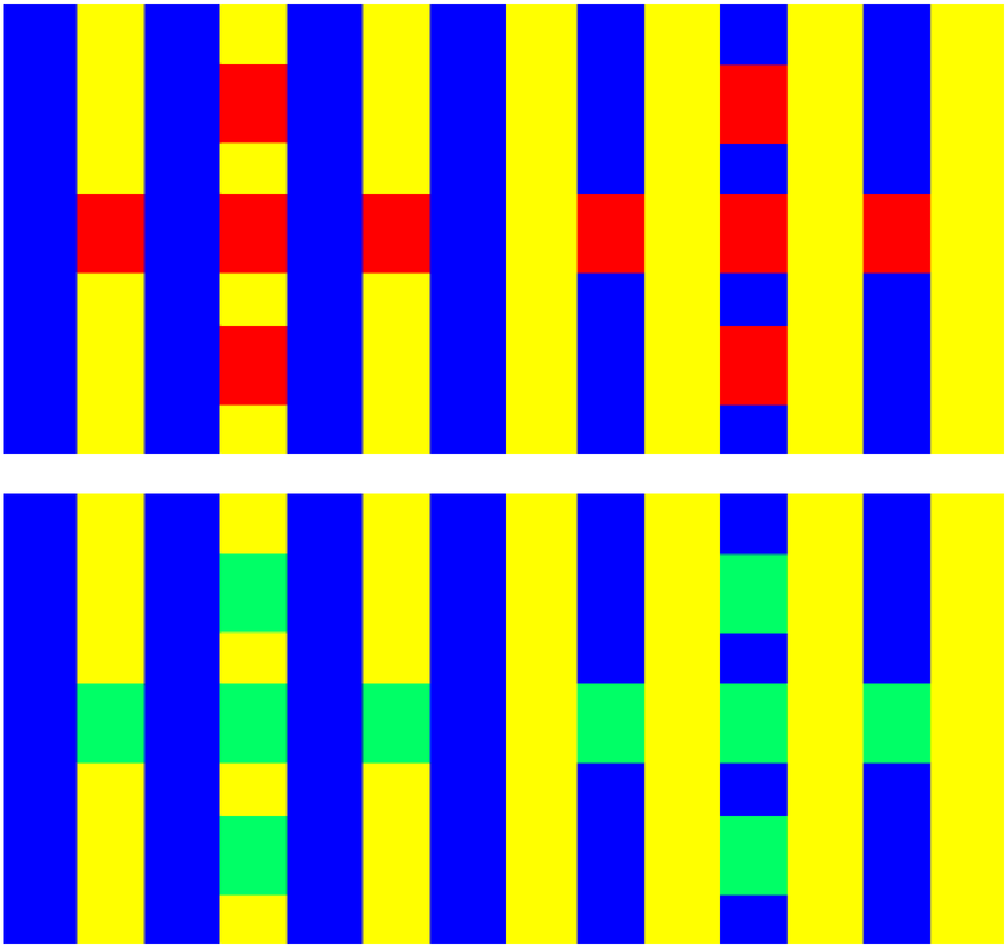


Figure 1 - Color Illusions

A good example is the illusion in Figure 1. In the top, the red squares are all the same, on the bottom, the green squares are all the same. Yet because of the surrounding context, your eyes are fooled into thinking they're different colors.

The problem is even more significant as more environments turn to LED lighting for energy savings. LEDs often produce white by mixing red, green and blue primary colors. It's difficult for people to accurately judge the color balance objectively without resorting to measuring tools like the Color Sensor.

Color Sensor Design

The Color Sensor is a simple to use, low cost tool for getting accurate color readings from both reflected and emitted light. It gives accurate readings in multiple intuitive formats, both on a built-in LCD display, and also on a larger web page available over a local network. The Color Sensor is lightweight and portable, making it easy to use in a variety of environments.

The Color Sensor uses the TI/Stellaris LM3S9B96 MCU to control all operations. The MCU communicates with the color sensor over using the built-in I2C interface, the LCD display using a modified SPI protocol, the Ethernet using the built-in Ethernet controller, serial IO for system debugging and number of switches and indicators via parallel I/O.

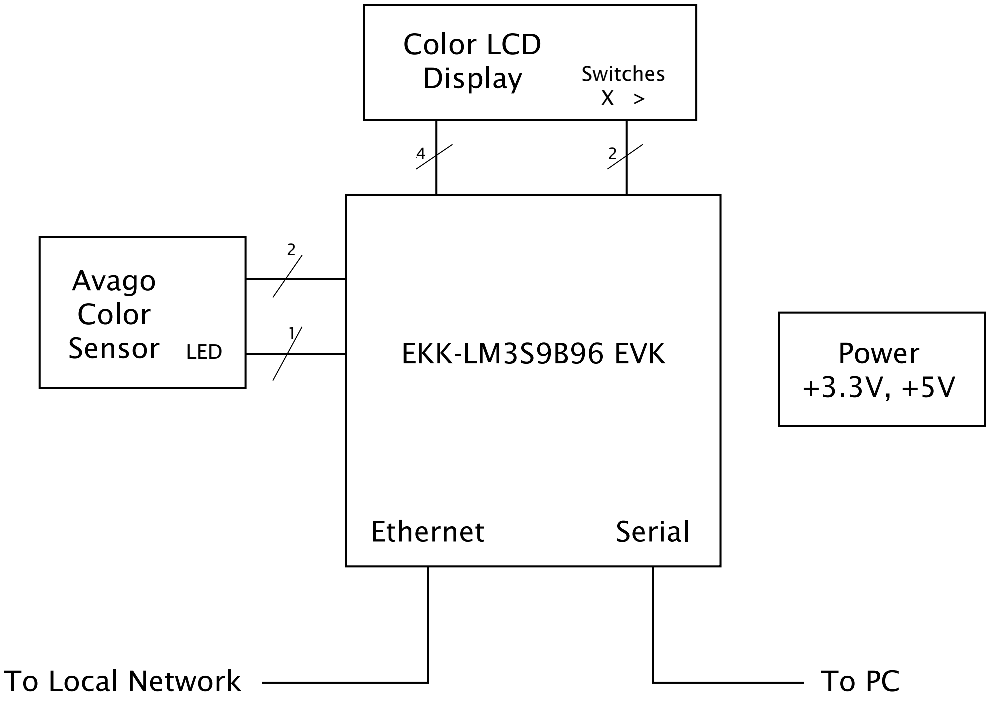


Figure 2 - Block Diagram

All of these communication channels are available simultaneously thanks to the built-in SafeRTOS operating system. The OS handles communication channels with independent concurrent tasks, so they're all "live" at the same time. There's no need to switch communication modes.

Hardware Design and Construction

The prototype for the Color Sensor was constructed from numerous modules available as breakout boards, including the sensor itself, the LCD display, and the EKK-LM3S9B96 Evaluation Kit. These are mounted on a custom-made PCB that also provides battery power using two AA cells and a step-up boost converter to provide the 5V required by the EKK-LM3S9B96.

The color LCD display is similar to those found in low-cost cell phones. It's based on an Epson controller, and mounted on a breakout board by Sparkfun electronics. Having an LCD display allows you to use the Color Sensor as a portable instrument, and get immediate readings from it. The MCU communicates with the LCD using a modified SPI interface. Unfortunately the interface used by the Epson controller is non-standard enough that it was necessary to "bit-bang" the protocol. Fortunately the display controller can handle the MCU at full speed this way. The LCD supplies two pushbutton switches, these are routed to the MCU's GPIO lines for user interface.

The LCD has a separate control for the backlight. This draws substantial power (about 250mA), so it is switched via a transistor, allowing the MCU to turn it off to conserve battery life.

The actual color sensor is an Avago ADJD-S371-v11, again mounted on a breakout board by Sparkfun Electronics. The Avago has four separate sensing channels for red, green and blue, as well as a "clear" channel that sums the first three. The sensor is accessed via standard I2C protocol. In addition, the Avago sensor provides an LED to illuminate materials for reading surface colors. The Sparkfun board has this LED connected through a transistor to drive it, but I found the drop introduced by the transistor made the LED unacceptably dim. I modified the board to drive the LED directly from one of the LM3S9B96 GPIO ports, and this worked fine. The sensor is socketed, so it's possible to connect an extension cable to it for conveniently reading surface colors.

The UART port of the EKK-LM3S9B96 is routed to a header for connection to a serial to USB adapter. This allows direct serial communication with the device, which proved invaluable for debugging the software.

The device is powered by two AA batteries. A Max756 boost regulator steps up the 2.8V from the batteries to 5V used by the EKK-LM3S9B96 board. A 3.3V regulator provides power for the color sensor and the LCD controller.

Software

The driver code for the LCD was written mostly from scratch. Most of the functionality is simply copying bitmaps; the actual graphics were prepared offline using tools like Photoshop, or generated with a Python script.

The driver for the sensor simply reads the data off the Avago device using the I2C protocol. Avago recommends a calibration step, where a light (or white paper, if using the sensor illumination LED) is shown on the sensor. The gain and sense interval are adjusted until all three primary sensors register more than 1000 but less than the maximum, this puts them in the optimum range.

The project makes extensive use of the StellarisWare utilities and SafeRTOS supplied with the EKK-LM3S9B96. As mentioned previously, all I/O and interface are partitioned into separate tasks, allowing them to function independently.

The main tasks are

- The button task. This responds to the button pushes, and changes the UI state accordingly.
- The UART task. This listens for commands on the UART (the standard Stellaris utility was modified to be interrupt driven). This was used primarily for testing and debugging, but it can also be used to get readings from the project.
- The sensor task. This polls the sensor and updates the display.
- The Ethernet/IP task. This listens for requests for web pages on the Ethernet and responds to them.

All of these tasks run concurrently via SafeRTOS.

The RGB display simply displays the sensor output on the LCD. For the HSL display, the RGB values are converted to HSL. The "L" term (lightness, or brightness) is dropped, leaving hue (the horizontal axis) and saturation (the vertical axis). These are plotted on top of a stored image of the Hue/Saturation spectrum, giving a graphical representation of where the color falls. For the web page, this is plotted using a simple JavaScript routine.

The Web server listens for connections and produces a single web page, with the Hue/Saturation plot with a marker plotted in JavaScript.

Use

The Color Sensor is simple to use. Below the display, two controls are provided, ">", for "move to the next screen," and "X", for "take action on this screen". The first screen is for calibration. The Avago ADJD-S372-v11 sensor used has a wide dynamic range. By pointing a white light (or white object) at the sensor and hitting the "X" button, the sensor adjusts the gain and the sense interval to maximize the dynamic range of the actual color readings.

Pressing ">" takes you to the next screen, which allows you to toggle the sensor's built-in illumination LED for reading reflected light. Pressing ">" takes you to a screen with direct RGB readouts. This is useful if you're trying to, for example, calibrate two different light sources to produce the same output. Finally, the last ">" takes you to a hue / saturation color chart. The sensor's reading is plotted on the chart, providing an intuitive representation of the color (numerical results are also provided at the bottom). In both the RGB and HS screens, pressing "X" toggles the sensor updates. A final press of ">" takes you back to the calibration screen.

Figure 3 shows the Color Sensor in use. In the second photo, the sensor is pointed at the yellow-green "power" LED on the EKK-LM3S9B96. The red arrow shows the marker at the fully-saturated yellow green color on the hue/saturation display.

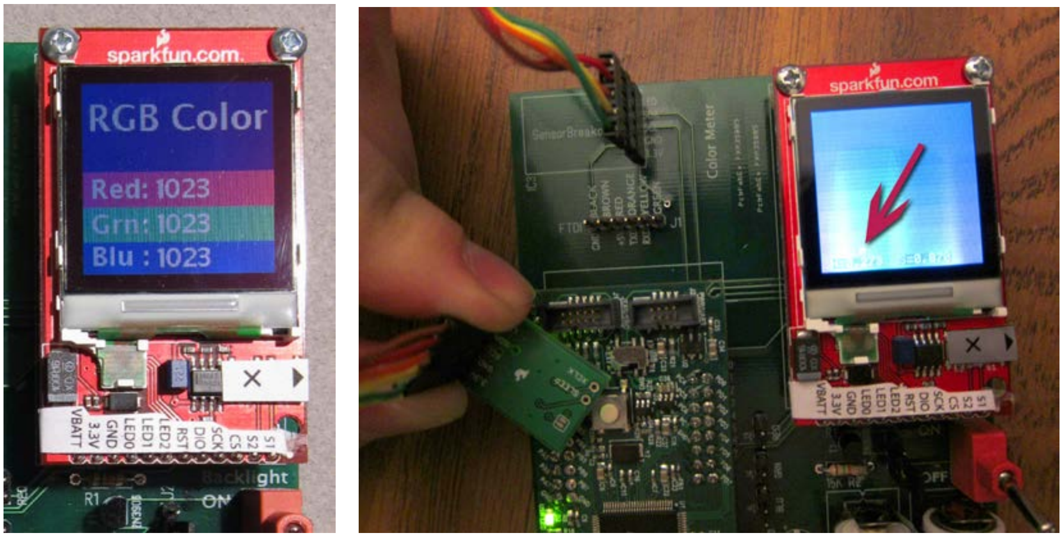


Figure 3 - Displays in use

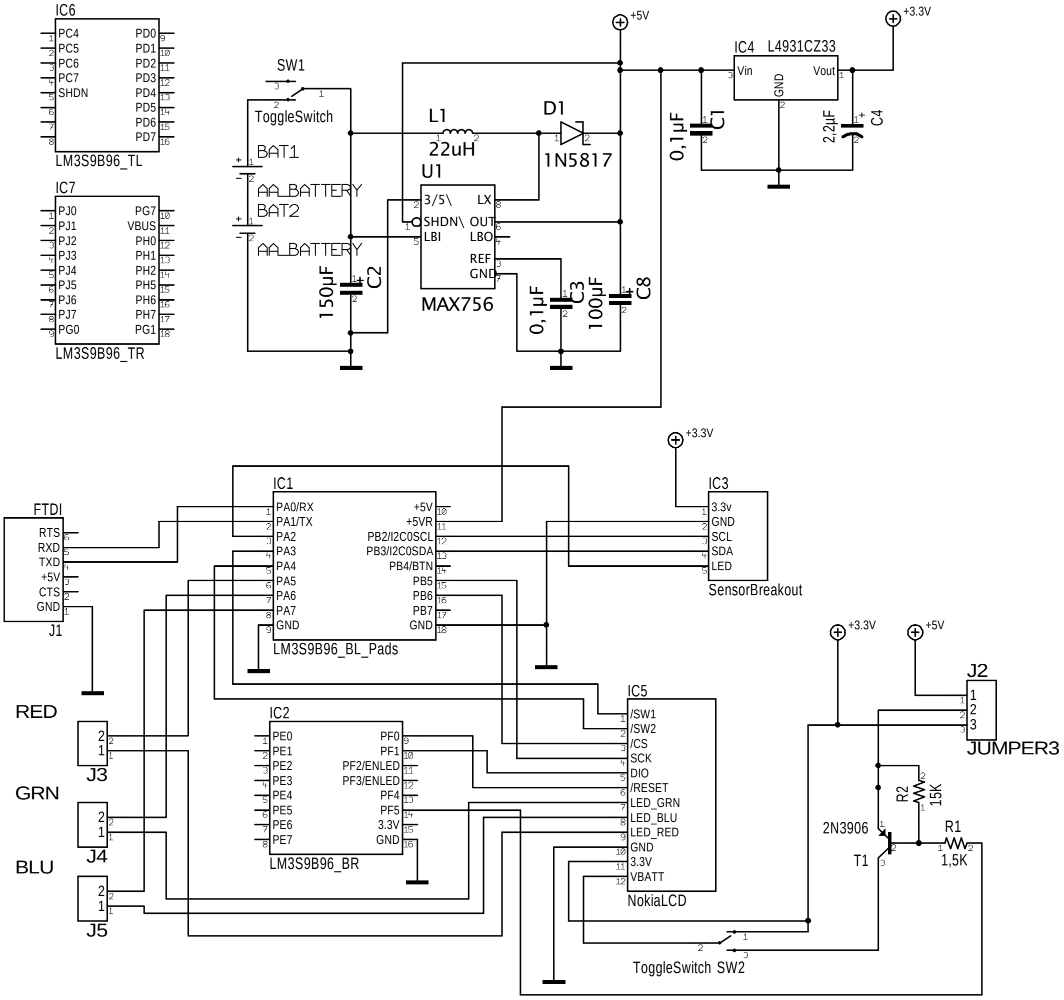
Accessing the color sensor over the network displays a page with the RGB and hue / saturation plots of the sensor's current reading. Controls are also provided on the web page for calibration and toggling the sensor's LED.

Conclusions and Future Work

The Color Sensor is a very low cost instrument that could be very useful in a variety of situations where it's important to match or calibrate colors. The ability to access the data over the network also allows for applications where lighting is monitored over time.

The display could easily be extended with more color modes, such as Lab, XYZ, Yxy, etc for specialized applications.

Schematic


IC6
PC4
PC5
PC6
PC7
SHDN
PD0
PD1
PD2
PD3
PD4
PD5
PD6
PD7
LM3S9B96_TL
IC7
PJ0
PJ1
PJ2
PJ3
PJ4
PJ5
PJ6
PJ7
PG0
PG7
VBUS
PH0
PH1
PH2
PH5
PH6
PH7
PG1
LM3S9B96_TR
SW1
ToggleSwitch
BAT1
AA_BATTERY
BAT2
AA_BATTERY
150µF
C2
L1
22uH
D1
1N5817
U1
3/5\
SHDN\
LBI
LX
OUT
LBO
REF
GND
MAX756
0,1µF
C3
100µF
C8
+5V
IC4
L4931CZ33
Vin
GND
Vout
0,1µF
C1
2,2µF
C4
+3.3V
+3.3V
IC3
3.3v
GND
SCL
SDA
LED
SensorBreakout
FTDI
RTS
RXD
TXD
+5V
CTS
GND
J1
IC1
PA0/RX
PA1/TX
PA2
PA3
PA4
PA5
PA6
PA7
GND
+5V
+5VR
PB2/I2C0SCL
PB3/I2C0SDA
PB4/BTN
PB5
PB6
PB7
GND
LM3S9B96_BL_Pads
RED
J3
GRN
J4
BLU
J5
IC2
PE0
PE1
PE2
PE3
PE4
PE5
PE6
PE7
PF0
PF1
PF2/ENLED
PF3/ENLED
PF4
PF5
3.3V
GND
LM3S9B96_BR
IC5
/SW1
/SW2
/CS
SCK
DIO
/RESET
LED_GRN
LED_BLU
LED_RED
GND
3.3V
VBATT
NokiaLCD
ToggleSwitch SW2
+3.3V
+5V
J2
JUMPER3
R2
15K
2N3906
T1
R1
1,5K