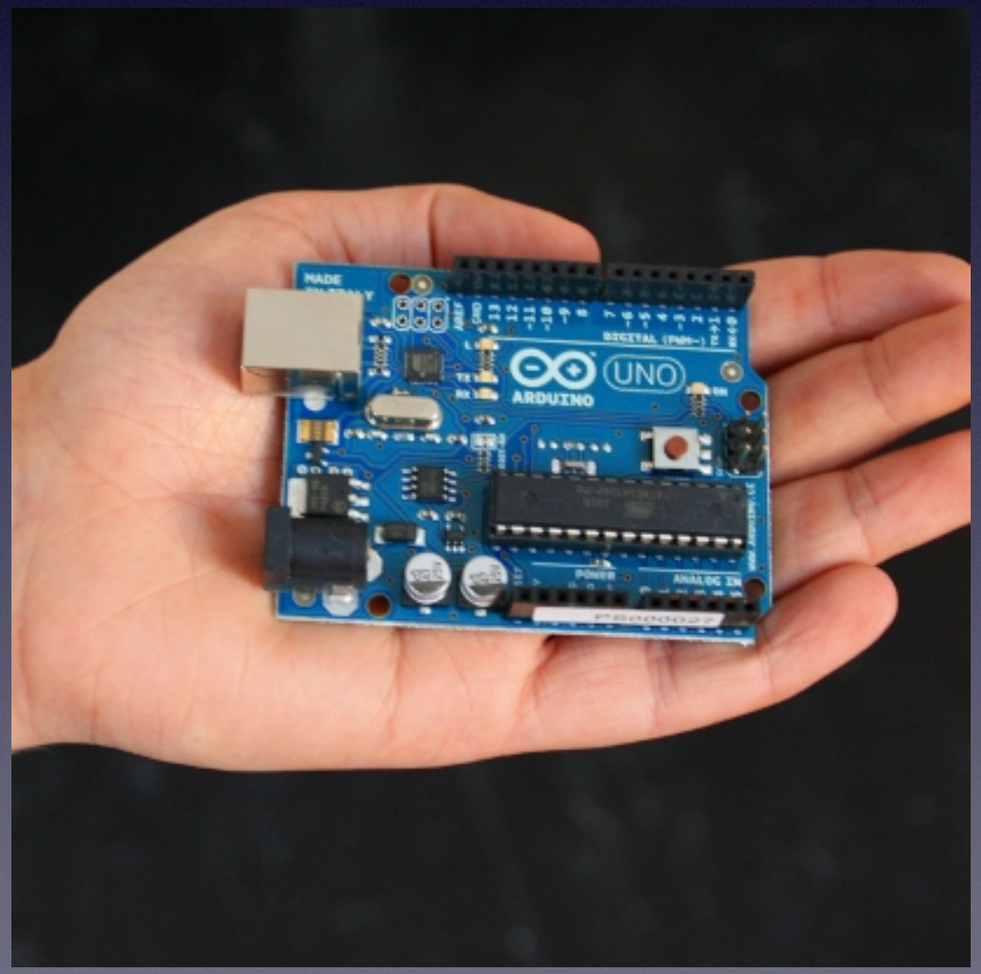
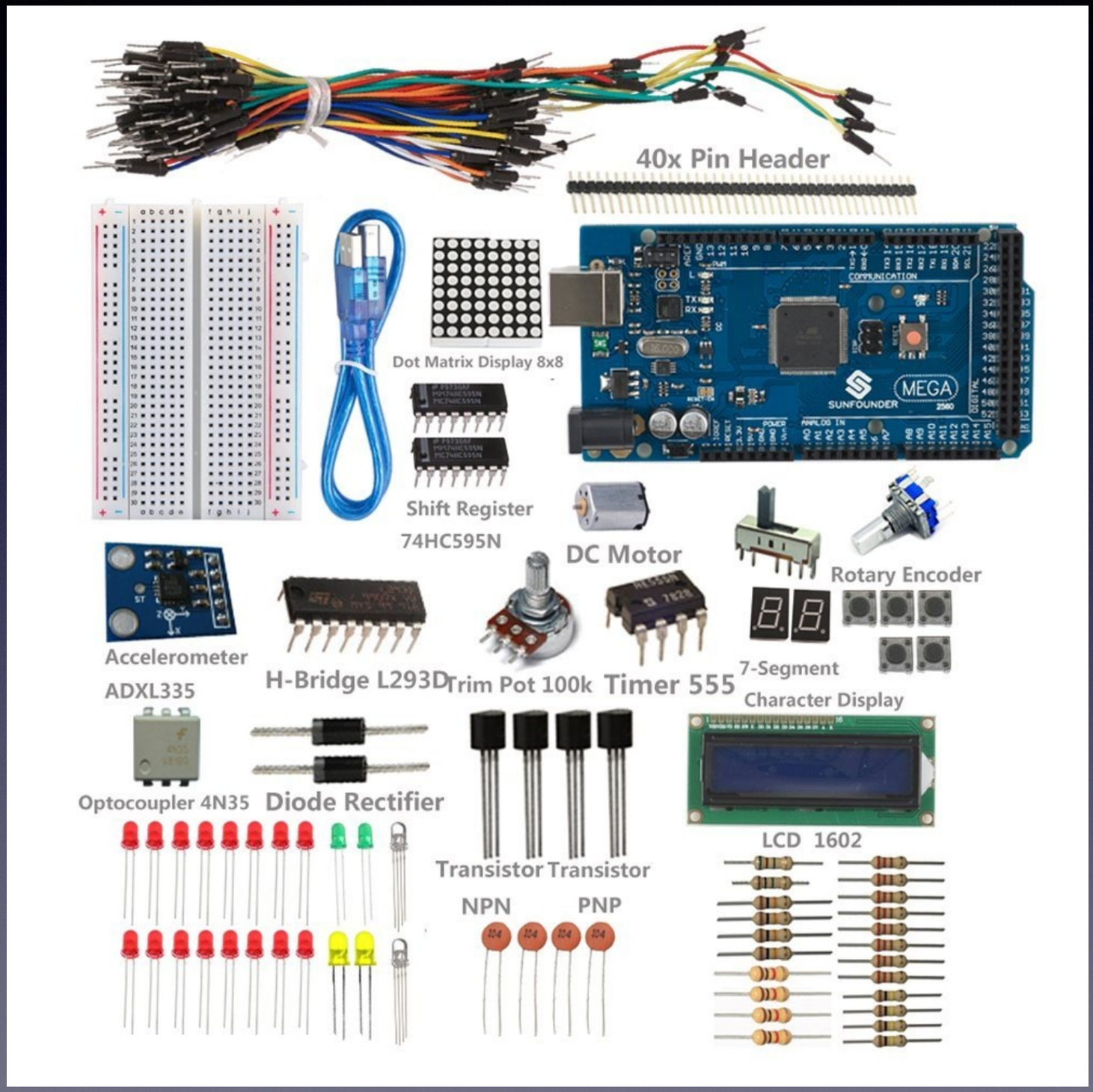
Arduino Programming

Using Open Source Hardware and Software.

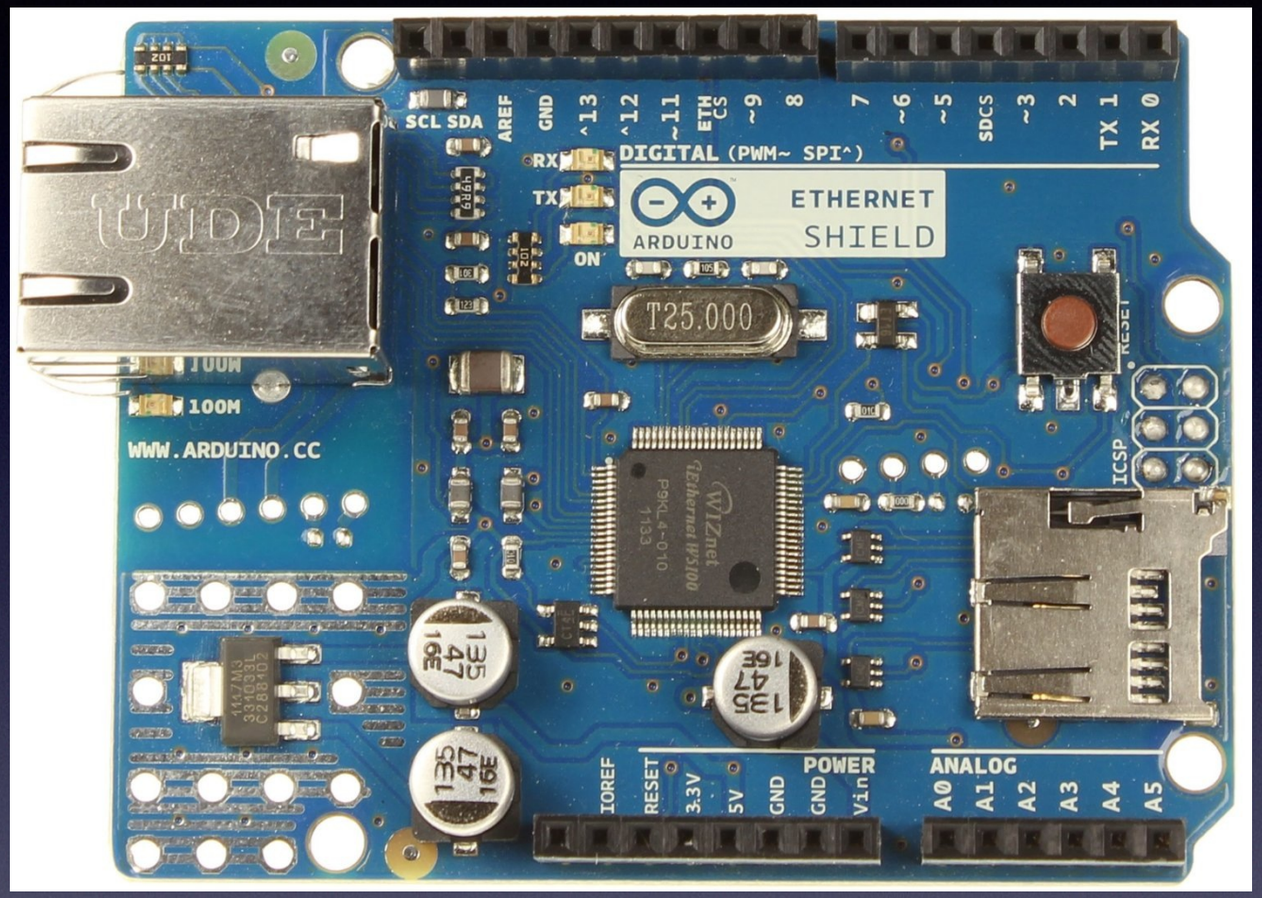


by Matt DePorter
Manager, Advanced Project Support
School of Information:
Technology, Science, and Arts
University of Arizona

Arduino Kit

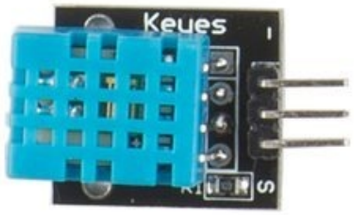

40x Pin Header
Dot Matrix Display 8x8
Shift Register
74HC595N
DC Motor
Rotary Encoder
Accelerometer
ADXL335
H-Bridge L293D
Trim Pot 100k
Timer 555
7-Segment
Character Display
Optocoupler 4N35
Diode Rectifier
LCD 1602
Transistor
NPN
Transistor
PNP

Arduino Kit


SCL SDA
AREF
GND
~13
~12
~11
ETH CS
~9
8
7
~6
~5
SDCS
~3
2
TX 1
RX 0
DIGITAL (PWM~ SPI^)
RX
TX
ON
ARDUINO
ETHERNET
SHIELD
T25.000
100M
WWW.ARDUINO.CC
ICSP
RESET
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
ANALOG
A0
A1
A2
A3
A4
A5

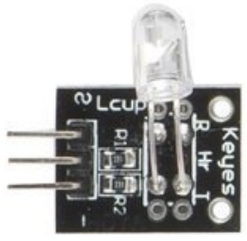
Arduino Kit

Flame sensor module



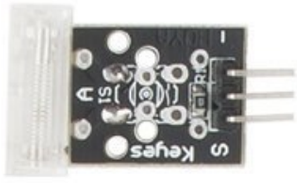
Humiture sensor module

Tracking sensor module

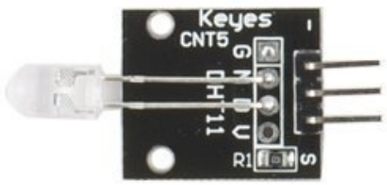


Finger-Pulse sensor module

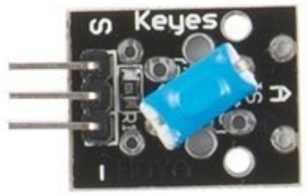
High-Sensitive voice sensor module



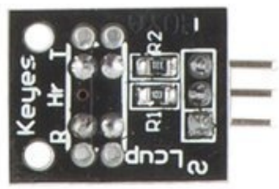
Knock sensor module



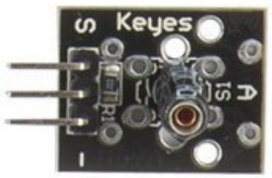
Colorful Auto-flash module



Tilt-Swith module



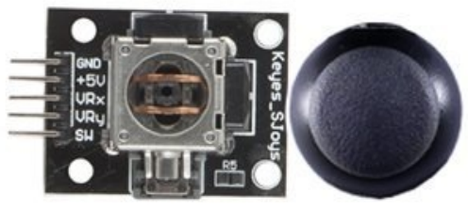
Light break sensor module



Shock-switch sensor module

Metal touch sensor module

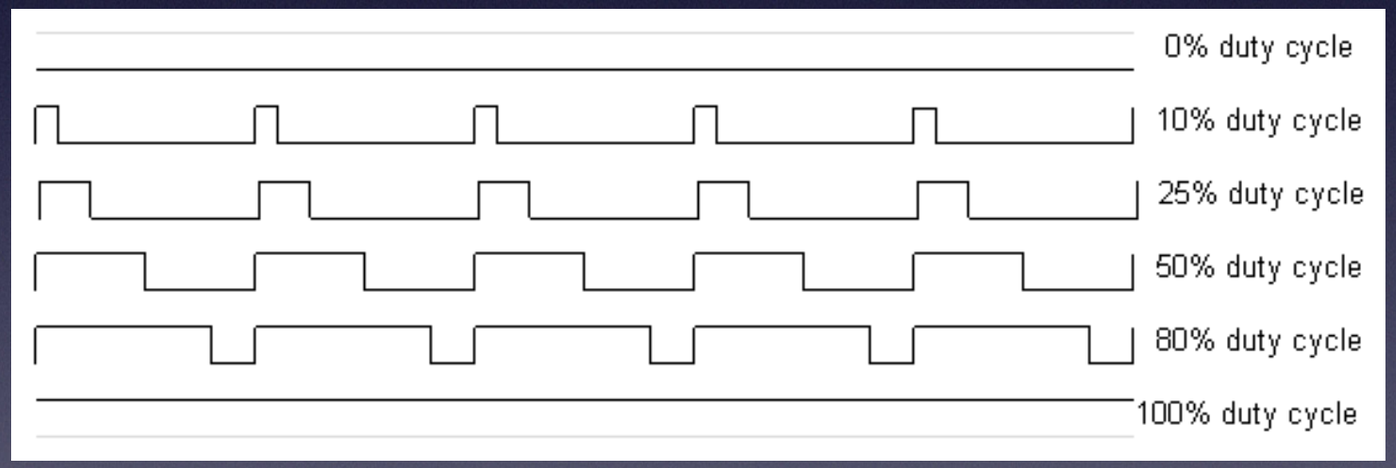
Linear-Hall sensor module



Joystick PS2 module

- The Arduino MEGA
  - Has 54 digital inputs and outputs, 15 of which are PWM, or pulse width modulation.
  - It also has 16 analog inputs
  - It can be run off of a 9V battery, but can take a varied range of input of 7V-12V
  - Each port can supply 5V at 40mA

- Microcontroller: ATmega1280
- Flash Memory: 128KB (ATmega1280)
- Clock Speed: 16 MHz
- Interface: USB



http://arduino.cc/en/Tutorial/SecretsOfArduinoPWM

- Pulse Width Modulation
  - This means that the port will give full power and no power over and over again in small pulses.
  - The gap between pulses in this case determines how bright the LED will be.
  - By adding more space, the LED will appear dim, but if you have less space, the LED will appear bright.
- So, 6 of these ports can give precise control to your LEDs. The rest can only signify either a HIGH or a LOW.
- High means that the port is pumping out 5V, and low means that the port is being switched to ground.

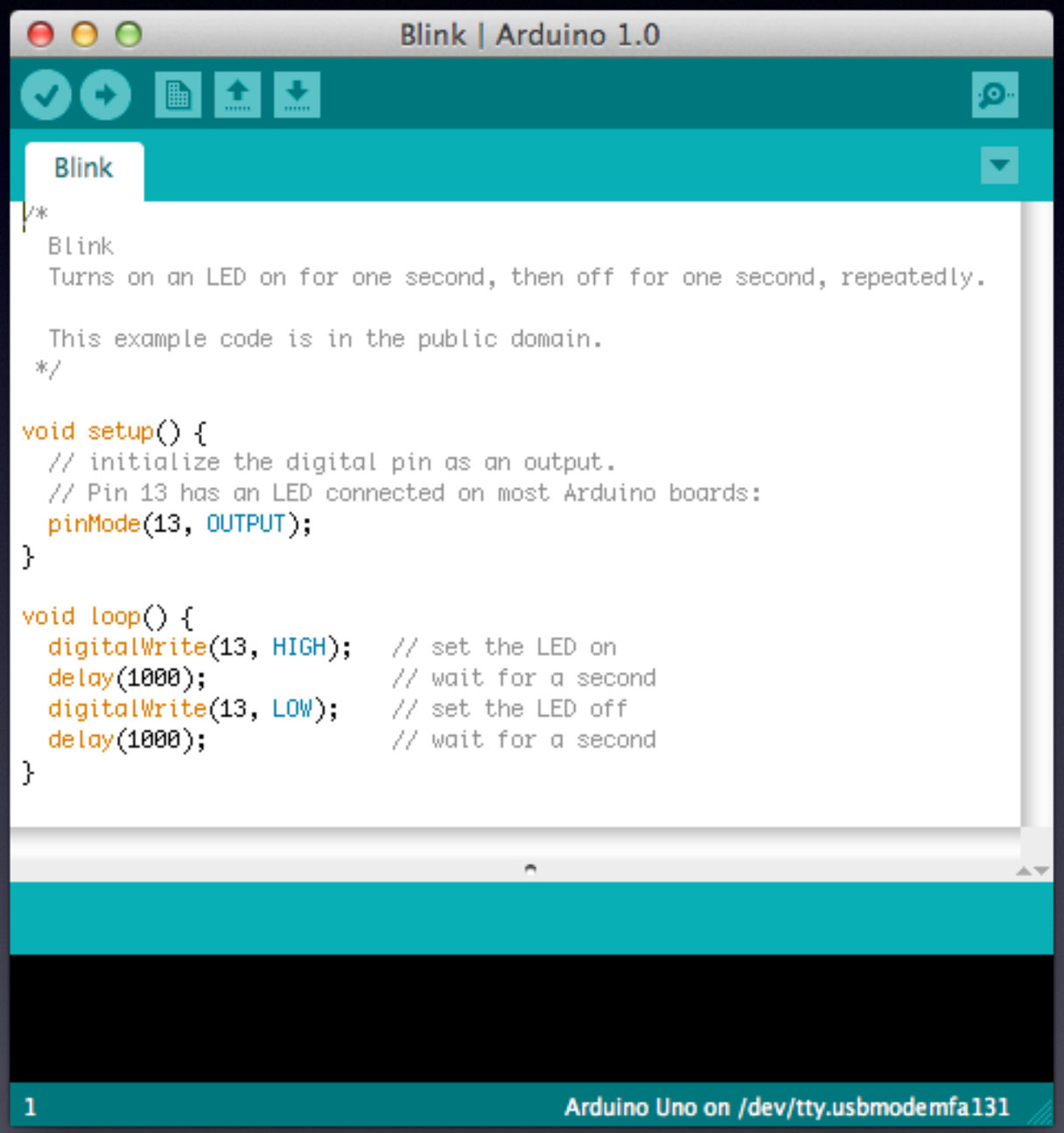
So how can you use the other ports?

The ports can either be used as is, or can use a method called Bit-banging Pulse Width Modulation.

This is a hacky way of getting PWM on a non-PWM port.

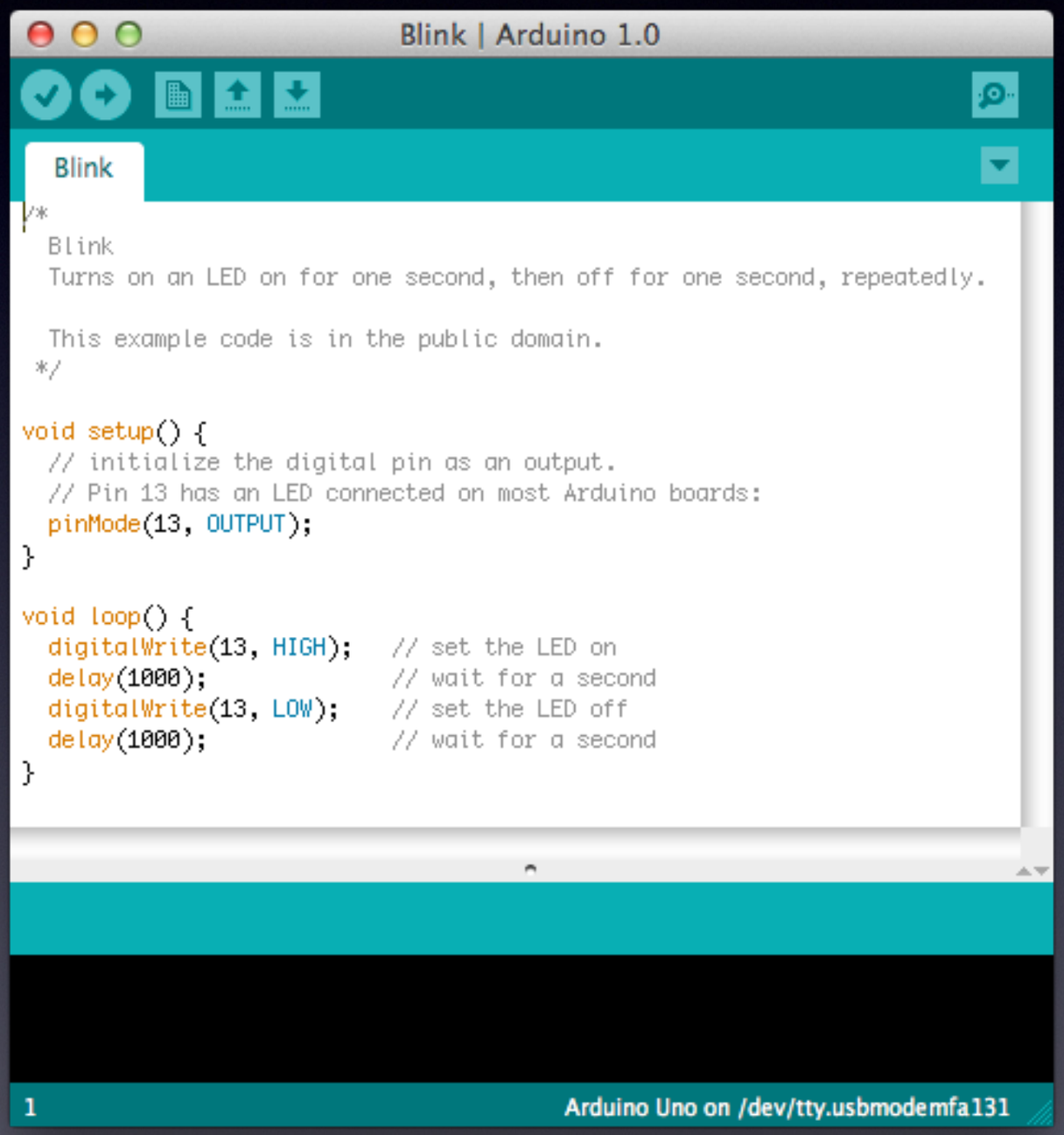
This can cause problems with timing, so one must be careful!

Programming your Arduino



- Code is written in C
  - Has an initializer block
  - Main execution loop
- Use Arduino Development software
- Cross Platform
  - Windows
  - Mac
  - Linux

Programming your Arduino



- Two components to Arduino Programming
  - void setup()
    - Used to initialize ports, variables, etc.
  - void loop()
    - Where main code execution occurs
- Usage of modified C libraries
- Can use standard C function calls

Resources

- Arduino Tutorials
  - http://arduino.cc/en/Tutorial/HomePage
- Arduino Libraries for Sensors
  - http://playground.arduino.cc/Main/LibraryList#Sensors
- Hardware resources
  - https://www.sparkfun.com/categories/103
  - http://www.adafruit.com/category/17
  - http://www.digikey.com/

Researching Sensors

1. Choose a sensor.
2. Do some searching on the sensor.
3. Determine what component is on the breakout board.
4. Use information to determine Arduino Library needed.
5. Download/Include library in code, try sample code.
6. Read sensor data.
7. Integrate sensor data into project.

Researching Sensors

Choose a sensor.

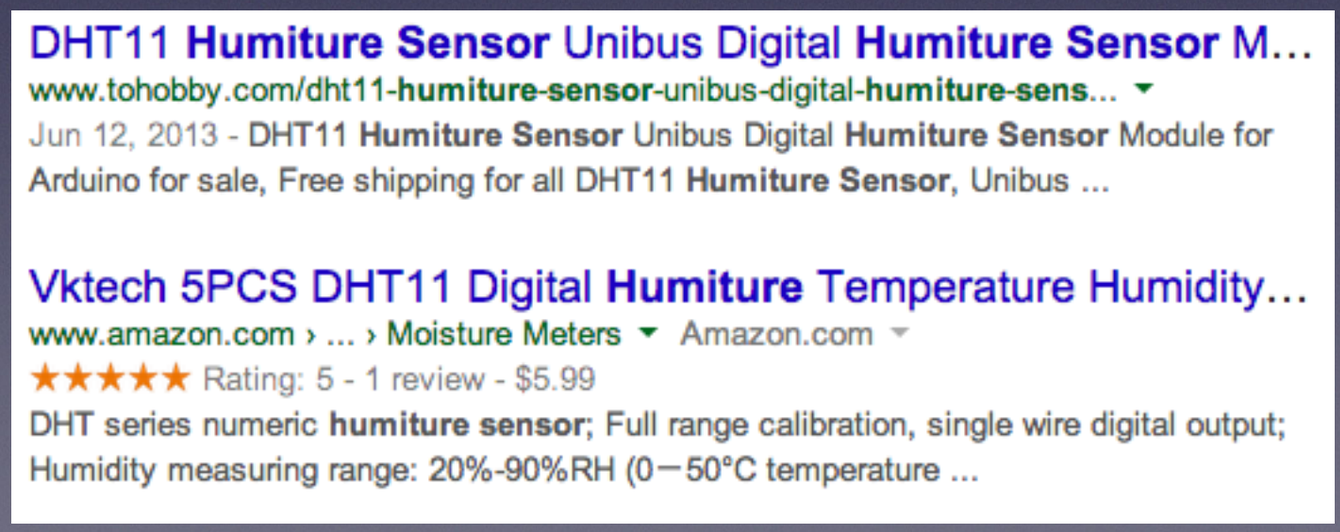
I chose the Humiture sensor module.

Researching Sensors

Do some searching on the sensor.

https://www.google.com/search?q=keyes+humiture+sensor&oq=keyes+humiture+sensor&aqs=chrome..69i57.9724j0j1&sourceid=chrome&espv=210&es_sm=119&ie=UTF-8

This search yields results for a DHT11 Sensor.



Researching Sensors

Determine what component is on the breakout board.

See if it looks anything like what you have.

http://www.tohobby.com/dht11-humiture-sensor-unibus-digital-humiture-sensor-module-for-arduino.html

Looks good to me!

(Apparently I was “Feeling Lucky”, as it was my first result)

Researching Sensors

Use information to determine Arduino Library needed.

Search for "DHT11" at:

http://playground.arduino.cc/Main/LibraryList#Sensors

Which yields results:

http://playground.arduino.cc/main/DHT11Lib

A DHT11 Class for Arduino.

Last Modified: December 14, 2013, at 03:57 AM
By: robtillaart
Platform: UNO (others not tested)

---

remarks & comments

Intro

The DHT11 is a relatively cheap sensor for measuring temperature and humidity. This article describes a small library for reading both from the sensor. The DHT22 is similar to the DHT11 and has greater accuracy. However, this library is not suitable for the DHT21 or DHT22 as they have a different data format. Check DHTlib for support of these sensors.

Researching Sensors

Download/Include library in code, try sample code.

Follow instructions at:

https://learn.adafruit.com/dht/using-a-dhtxx-sensor

Download code at:

https://github.com/adafruit/DHT-sensor-library

Instructions:

Check out

http://www.instructables.com/

https://learn.adafruit.com/

Researching Sensors

Connect sensor to MEGA

See Datasheet for DHT11

http://www.micro4you.com/files/sensor/DHT11.pdf

Find tutorials on how to connect. E.g.

https://learn.adafruit.com/dht

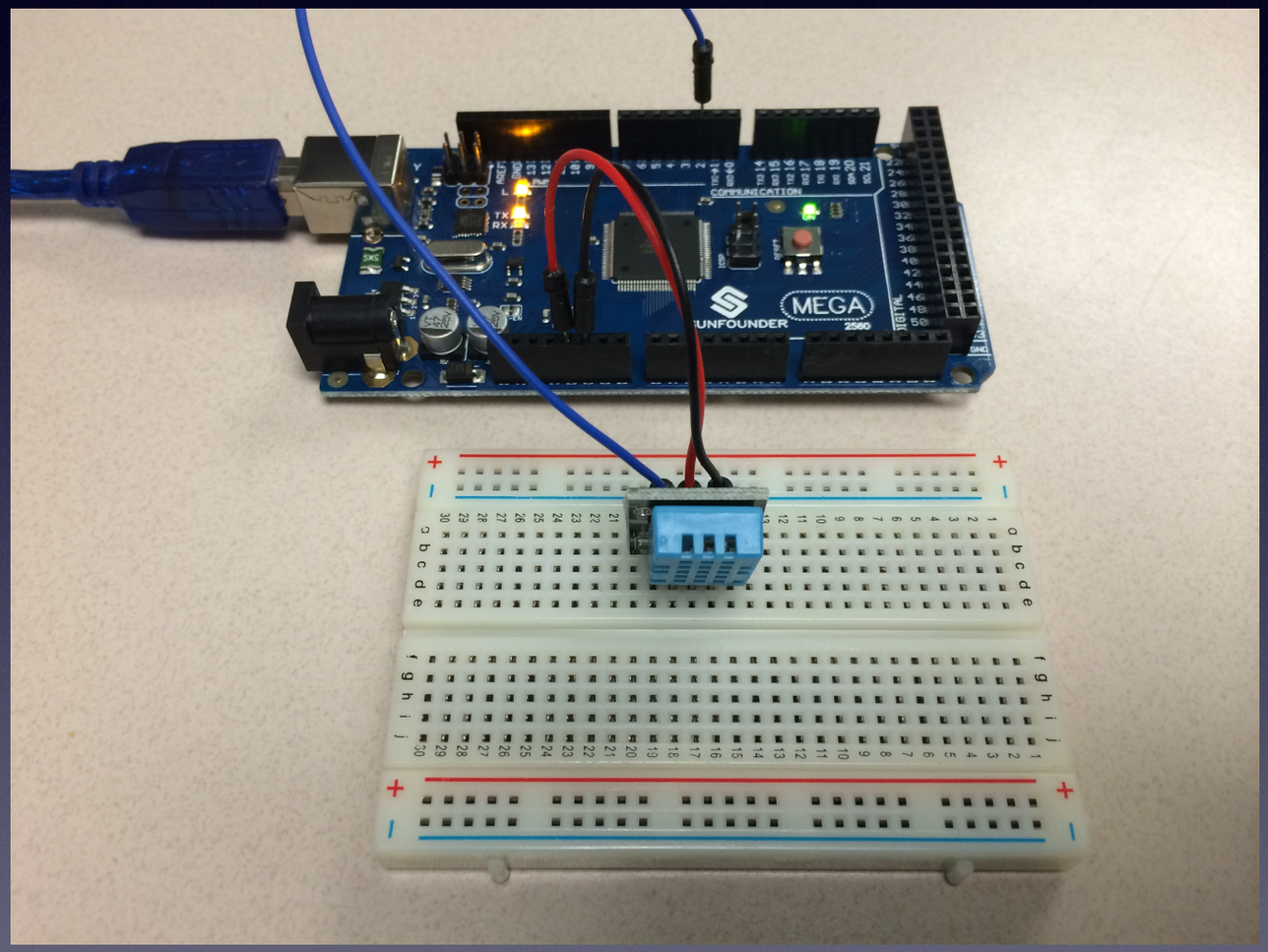
Make sure to pay very close attention to wiring!

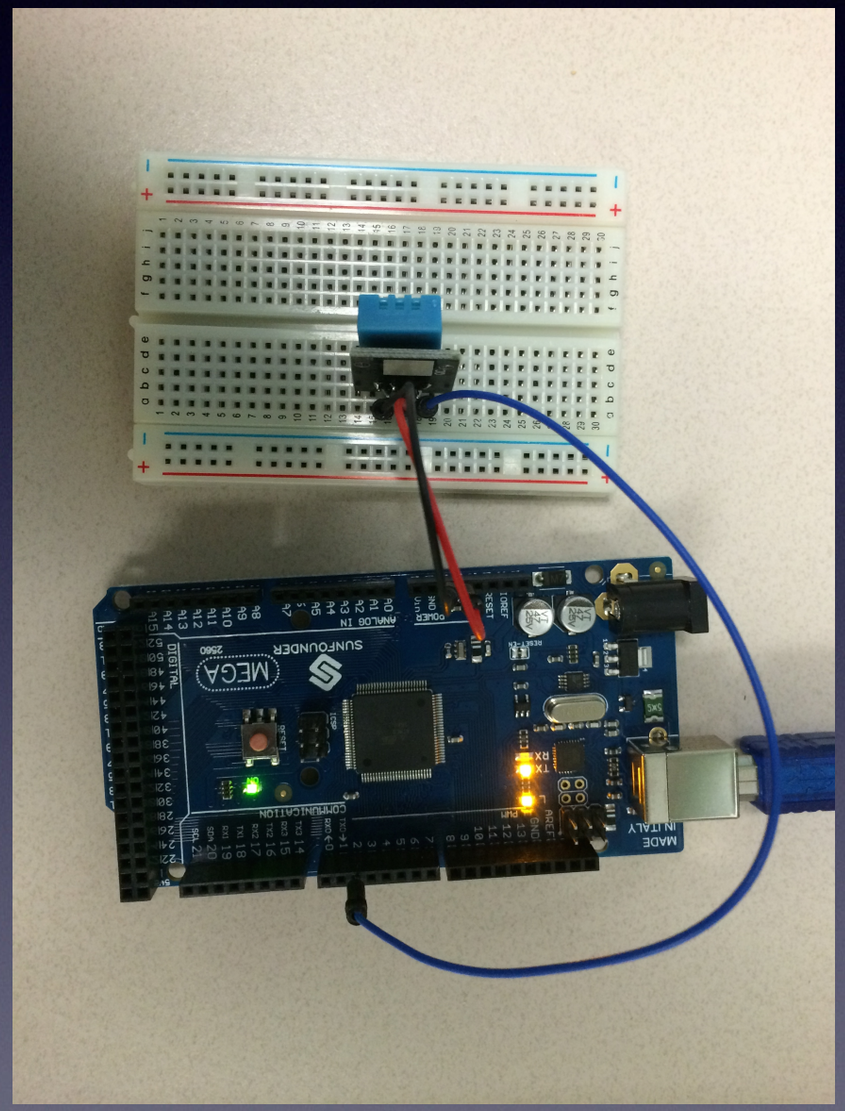
Connecting wrong wires can destroy the MEGA or the sensor itself.

Use Extreme Caution and triple check your wiring before attaching to the MEGA.

Researching Sensors

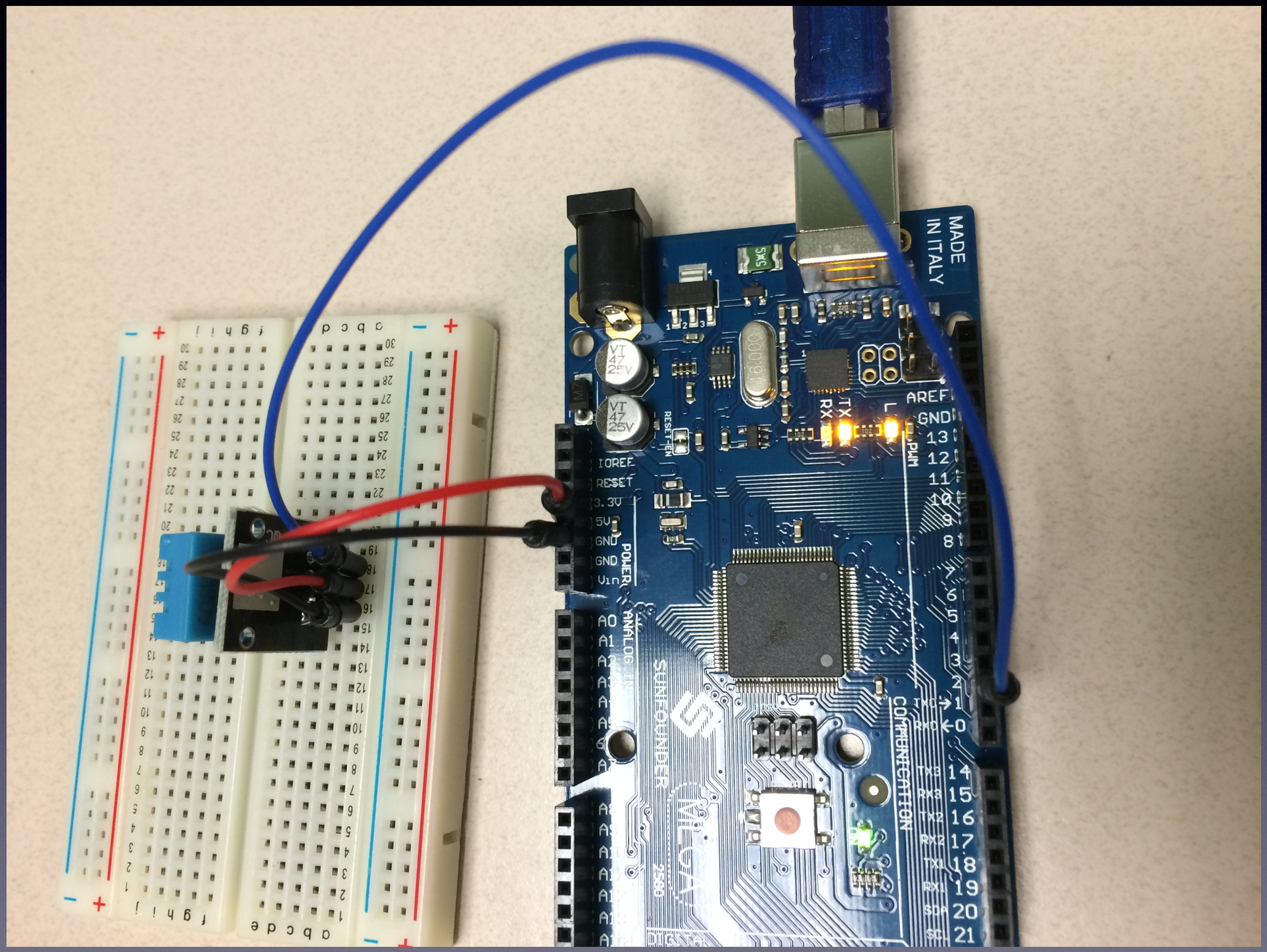
Hardware connections





Researching Sensors

Hardware connections



Researching Sensors

Read sensor data

DHTtester | Arduino 1.0.5

DHTtester §

```
// Example testing sketch for various DHT humidity/temperature sensors
// Written by ladyada, public domain

#include "DHT.h"

#define DHTPIN 2     // what pin we're connected to

// Uncomment whatever type you're using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT22   // DHT 22  (AM2302)
//#define DHTTYPE DHT21   // DHT 21 (AM2301)

// Connect pin 1 (on the left) of the sensor to +5V
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 4 (on the right) of the sensor to GROUND
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the se

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println("DHTxx test!");

  dht.begin();
}

void loop() {
```

Binary sketch size: 7,506 bytes (of a 258,048 byte maximum)

9 Arduino Mega 2560 or Mega ADK on /dev/tty.usbmodemfd1411

/dev/tty.usbmodemfd1411

Send

```
Hu: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
DHTxx test!
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
Humidity: 33.00 %       Temperature: 25.00 *C
```

Autoscroll | No line ending | 9600 baud

Researching Sensors

Integrate sensor data into project.

An example project:

http://www.instructables.com/id/Arduino-TempHumidity-with-LCD-and-Web-Interface/

Note:

This can be done using what you have in your kit.

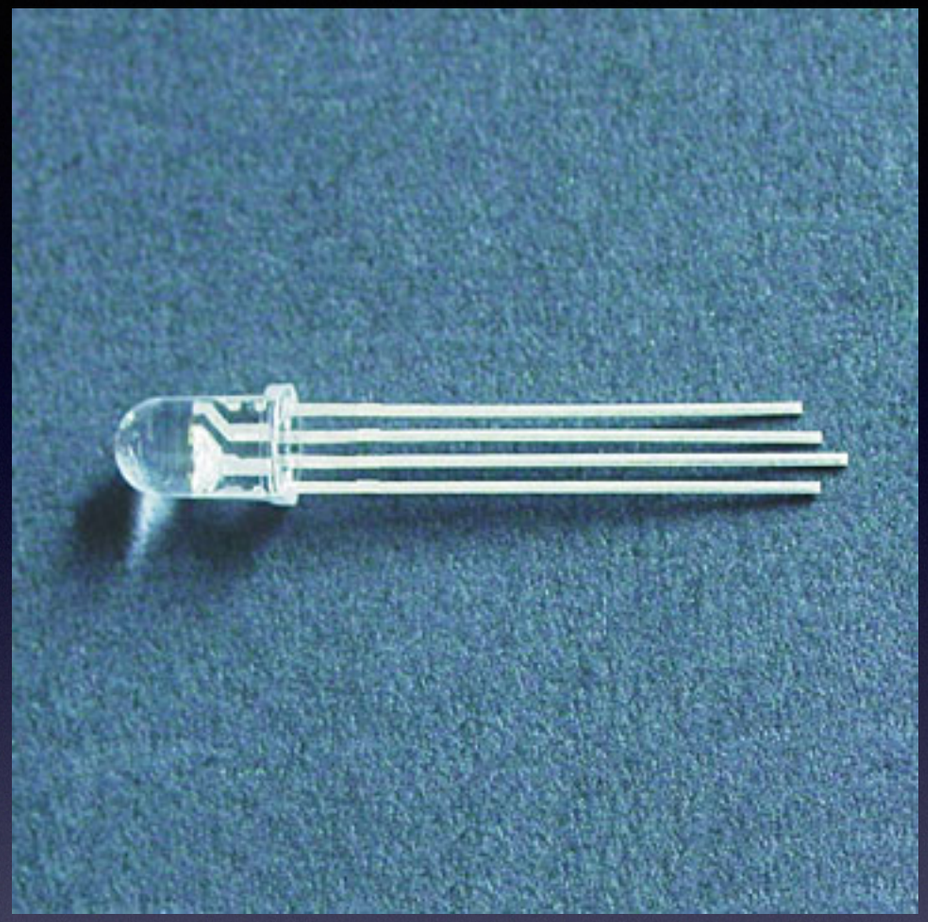
Other examples:

RGB LED

to

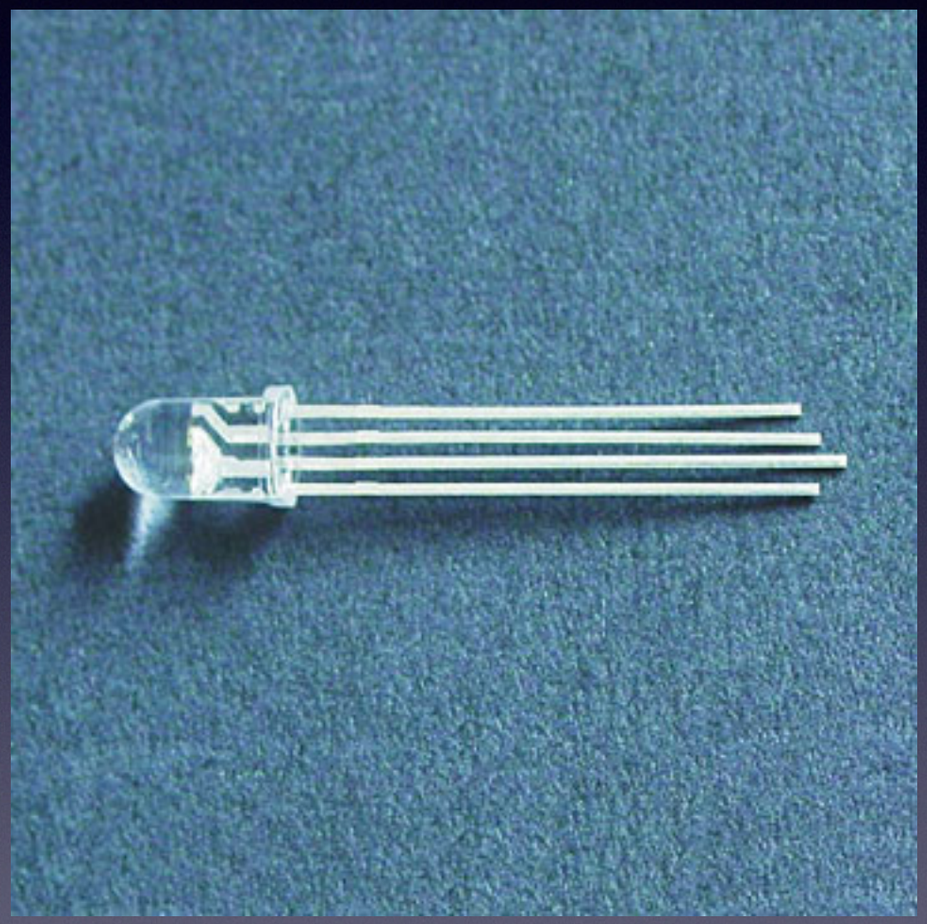
Add colored light to your projects

RGB LEDs



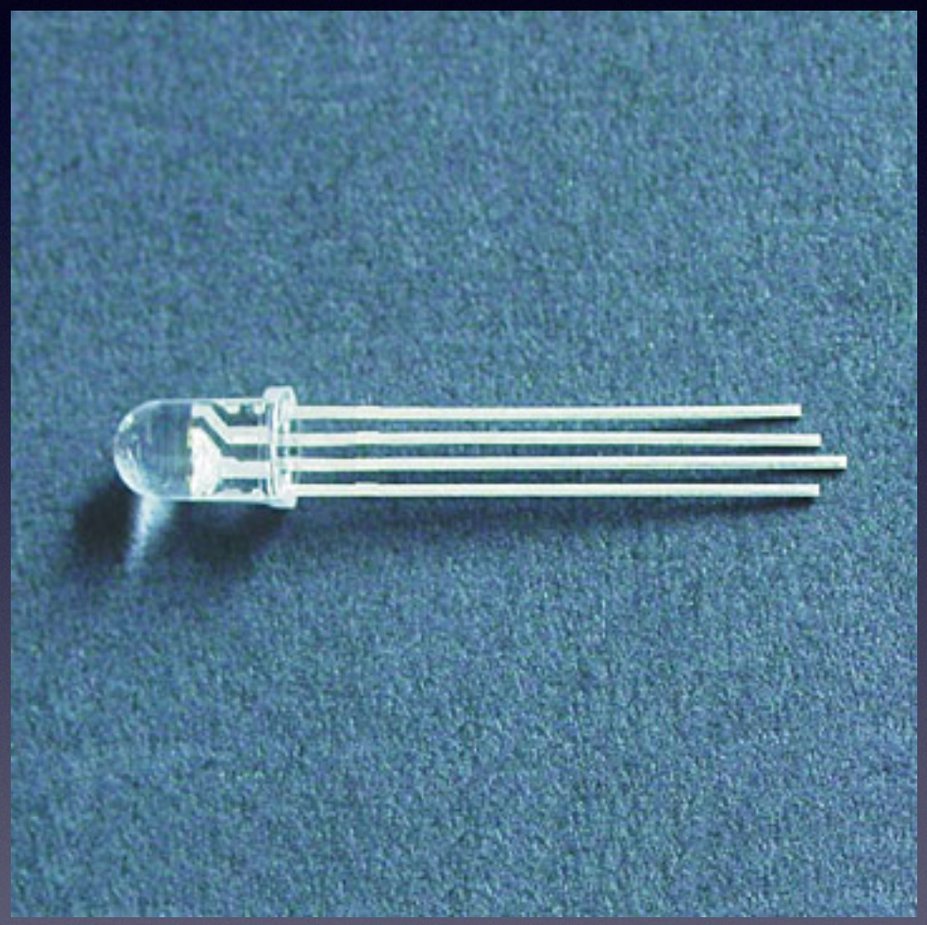
- Two types
  - Common Anode
  - Common Cathode

RGB LEDs



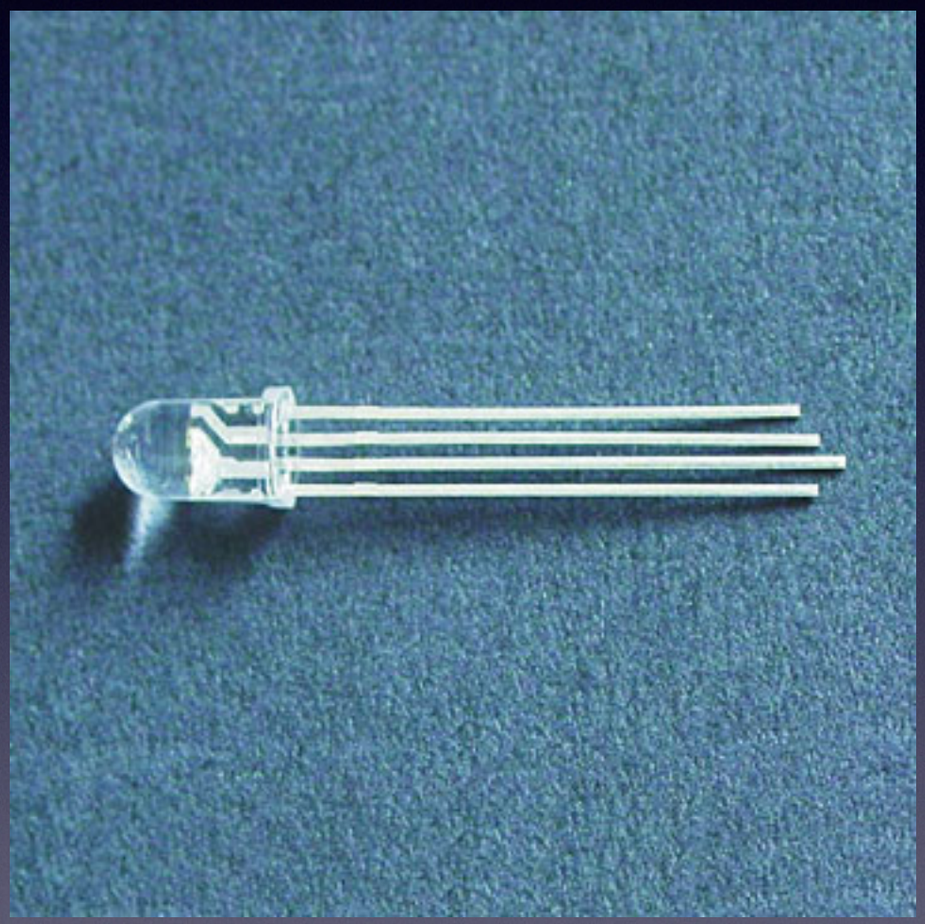
- Common Anode
  - +5V Anode (Single node)
  - Connect resistor to each wire and connect to output pin
- Write a LOW to turn the LED ON
- Write HIGH to turn it OFF
- Method: Current Sinking
  - (Preferred method for micro-controllers)

RGB LEDs



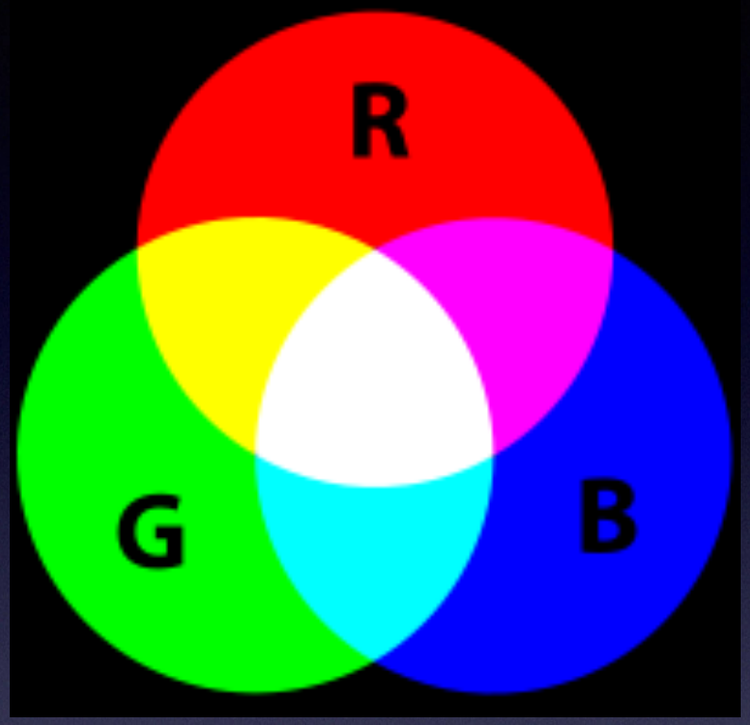
- Common Cathode
  - Cathode to ground
  - Each LED's anode to resistor and output pin
  - HIGH turns LED on.
  - (Method: Current Sourcing)

RGB LEDs



- We have RGB Common Anode LEDs
  - They work opposite of a normal LED.
  - Sending a PWM 0 value, will turn the corresponding color to full brightness, or ON.
  - Sending a PWM 255 value, will set the specified color to OFF or zero brightness..

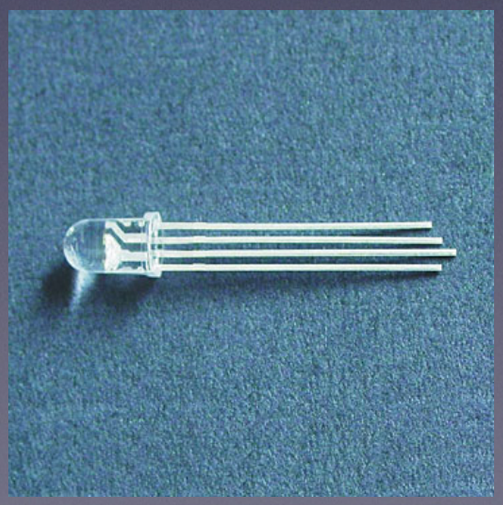
Ways to control RGB LEDs



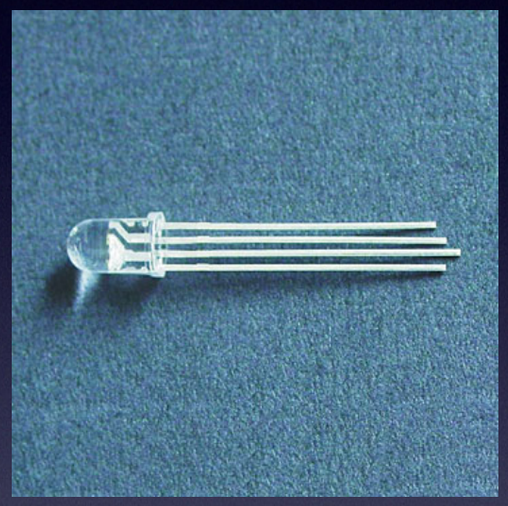
First way

- Use any digital port and send a HIGH or LOW value.
- Can only achieve Red, Green, Blue, Cyan, Magenta, Yellow

http://en.wikipedia.org/wiki/RGB_color_model

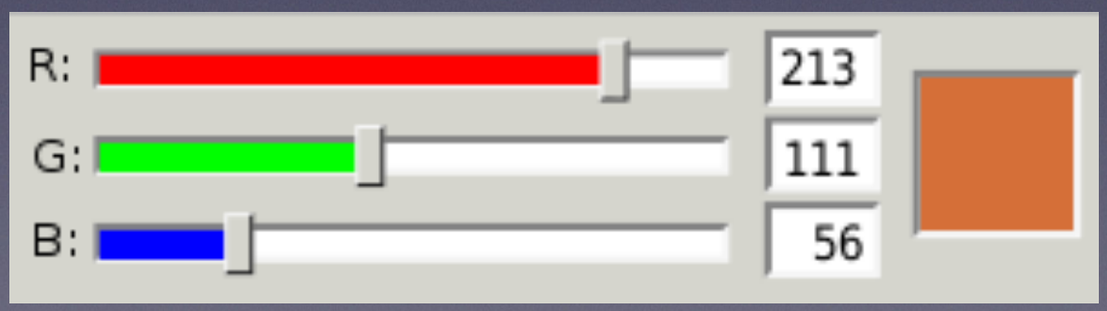


Ways to control RGB LEDs



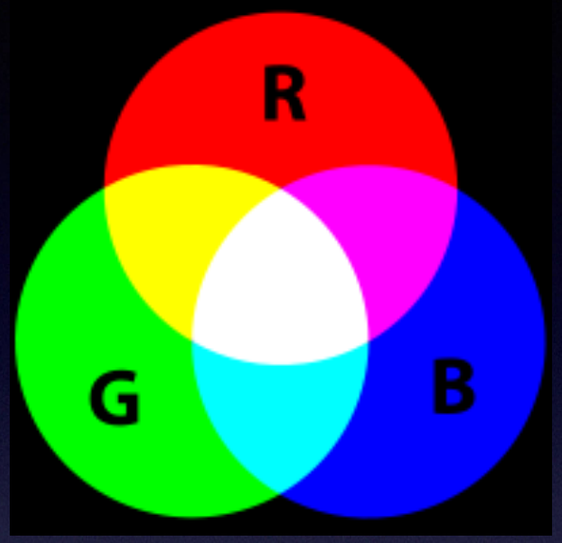
Second Way

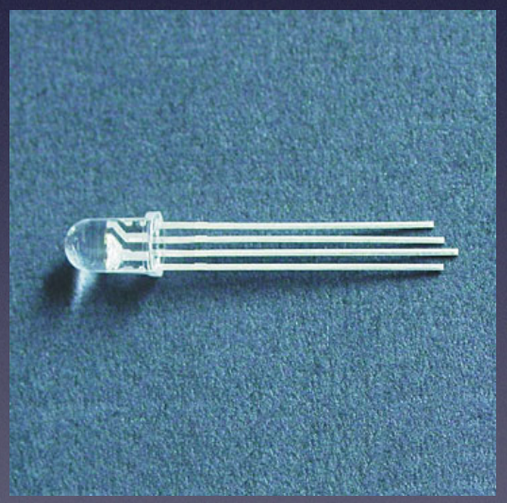
- Use any PWM port to specify a color intensity from 0 to 255
- Can display just about any color (8 bit color)



http://en.wikipedia.org/wiki/RGB_color_model

First way



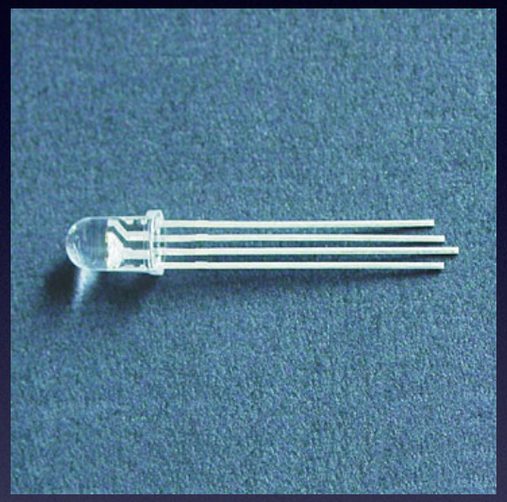


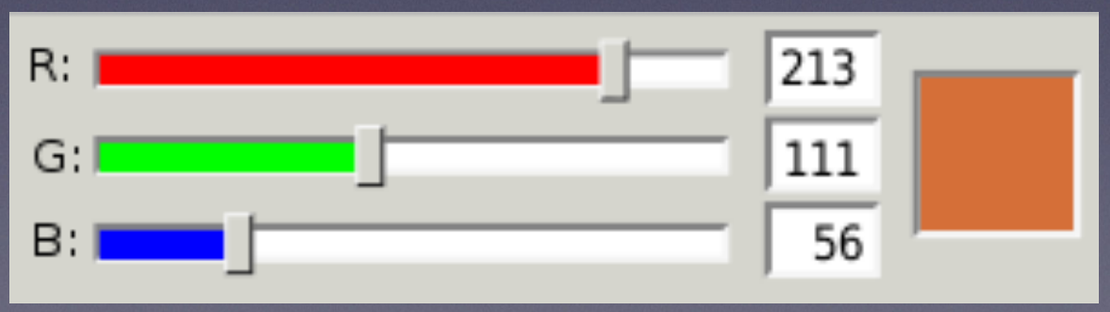
RGBDigitalCycle | Arduino 1.0

RGBDigitalCycle

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(10, OUTPUT); // green
  pinMode(9, OUTPUT);  // Blue
  pinMode(8, OUTPUT);  // red
}

void loop() {
  digitalWrite(10, HIGH);
  digitalWrite(9, HIGH);
  digitalWrite(8, HIGH);
  digitalWrite(10, LOW);  // green
  delay(1200);
  digitalWrite(10, HIGH);
  digitalWrite(9, LOW);   // blue
  delay(1200);
  digitalWrite(9, HIGH);
  digitalWrite(8, LOW);   // red
  delay(1200);
  digitalWrite(10, LOW);  // yellow
  delay(1200);
  digitalWrite(8, HIGH);
  digitalWrite(9, LOW);   // Cyan (Light Blue)
  delay(1200);
  digitalWrite(10, HIGH);
  digitalWrite(8, LOW);   // Magenta (purple)
  delay(1200);
  digitalWrite(10, LOW);  // White
  delay(1200);
}
```

Second Way





RGBAnalogCycleWithSerial | Arduino 1.0

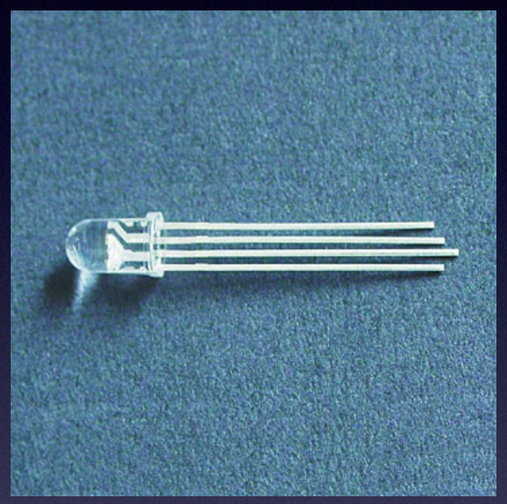
RGBAnalogCycleWithSerial

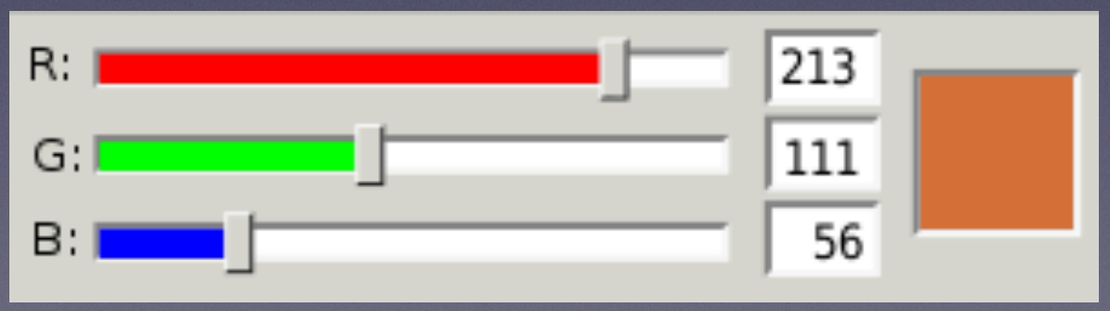
```
int pwm_a = 9;
int pwm_b = 10;
int pwm_c = 11;
int i;

void setup()
{
pinMode(pwm_a, OUTPUT);
pinMode(pwm_b, OUTPUT);
pinMode(pwm_c, OUTPUT);

Serial.begin(9600);
}

void loop()
{
//
//fade all channels up and down.
//
Serial.println("Fading all pwm channels up to max.");
for (i=0; i<=255; i++)
{
analogWrite(pwm_a, i);
analogWrite(pwm_b, i);
analogWrite(pwm_c, i);

delay(10);
}

Serial.println("All pwm channels at max.");
delay(1000);
Serial.println("Fading all channels to 0");

for (i=255; i>=0; i--)
{
analogWrite(pwm_a, i);
analogWrite(pwm_b, i);
analogWrite(pwm_c, i);

delay(10);
}

Serial.println("All pwm channels at zero.");
delay(1000);

fade_channel(pwm_a);
fade_channel(pwm_b);
fade_channel(pwm_c);
}
```

Done Saving.

Binary sketch size: 3848 bytes (of a 32256 byte maximum)

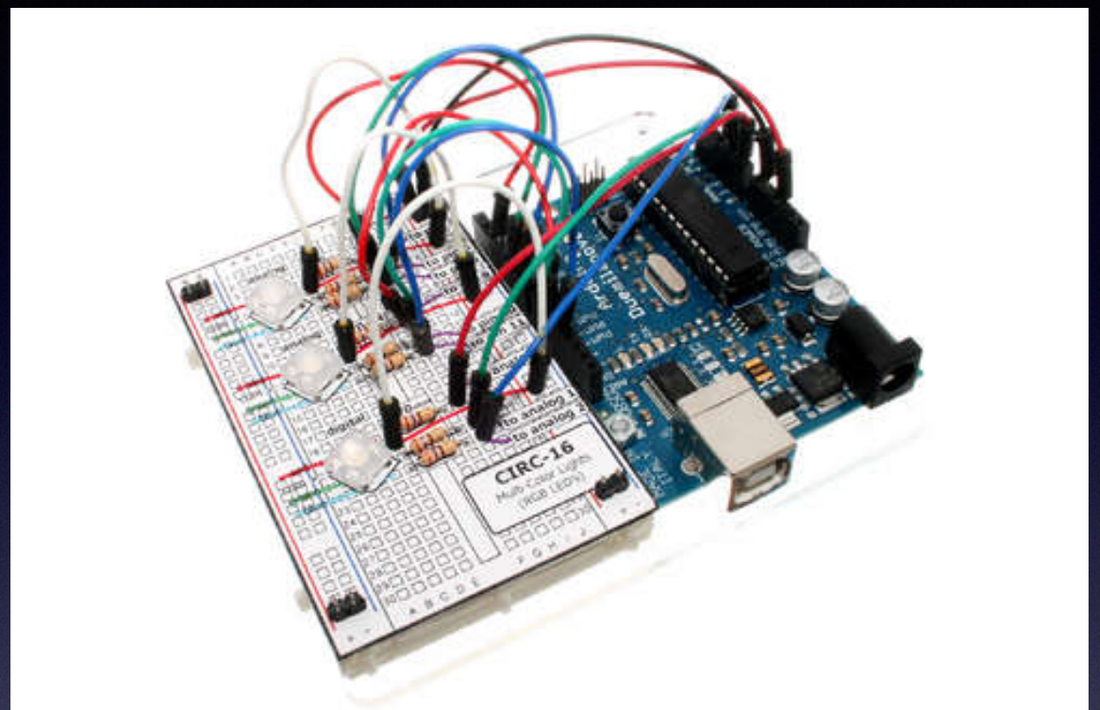
Second Way

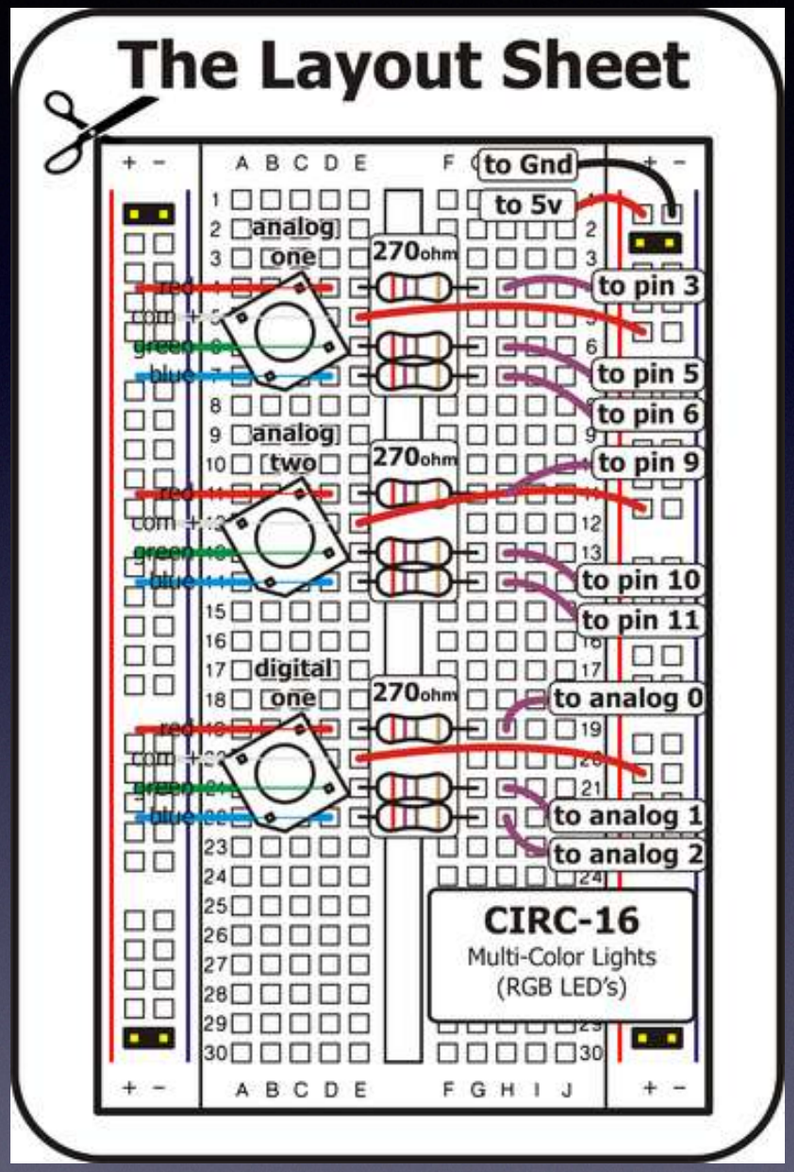


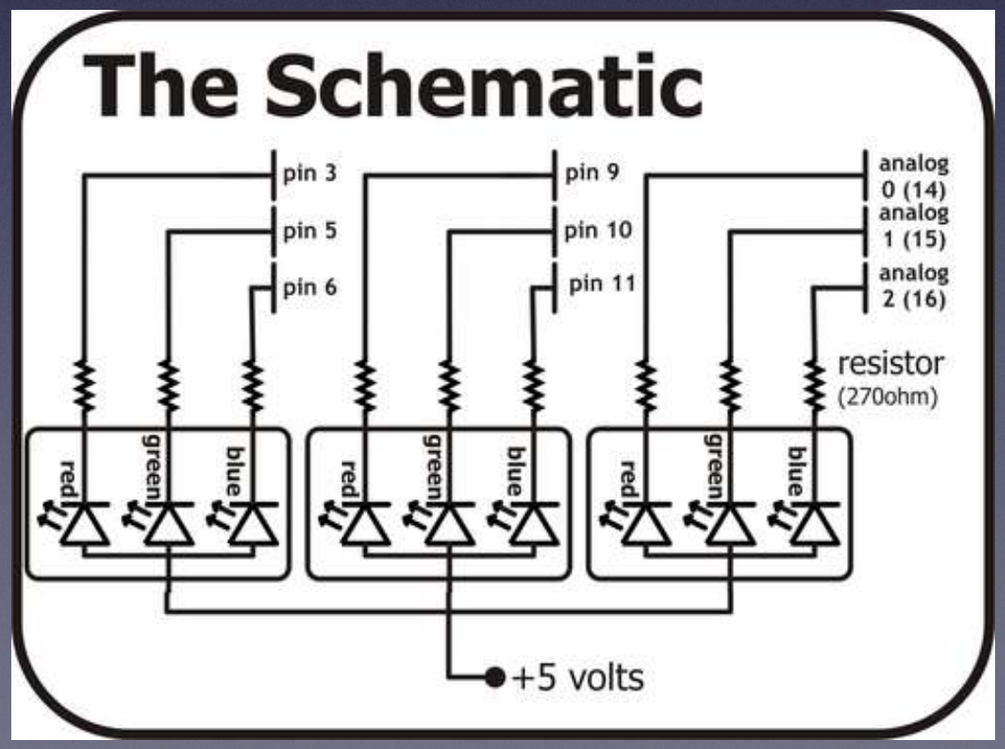



```
int pwm_red = 9;
int pwm_blue = 10;
int pwm_green = 11;
int i;

void setup()
{
pinMode(pwm_red, OUTPUT);
pinMode(pwm_blue, OUTPUT);
pinMode(pwm_green, OUTPUT);

Serial.begin(9600);
}

void loop()
{

// RED
Serial.println("Fading Red pwm channel up to max.");
analogWrite(pwm_blue, 255);
analogWrite(pwm_green, 255);
for (i=255; i>=0; i--)
{
analogWrite(pwm_red, i);
delay(10);
}
delay(100);
// BLUE
Serial.println("Fading Blue pwm channel up to max.");
analogWrite(pwm_red, 255);
analogWrite(pwm_green, 255);
for (i=255; i>=0; i--)
{
analogWrite(pwm_blue, i);
delay(10);
}
delay(100);
// GREEN
Serial.println("Fading Green pwm channel up to max.");
analogWrite(pwm_red, 255);
analogWrite(pwm_blue, 255);
for (i=255; i>=0; i--)
{
analogWrite(pwm_green, i);
delay(10);
}
```

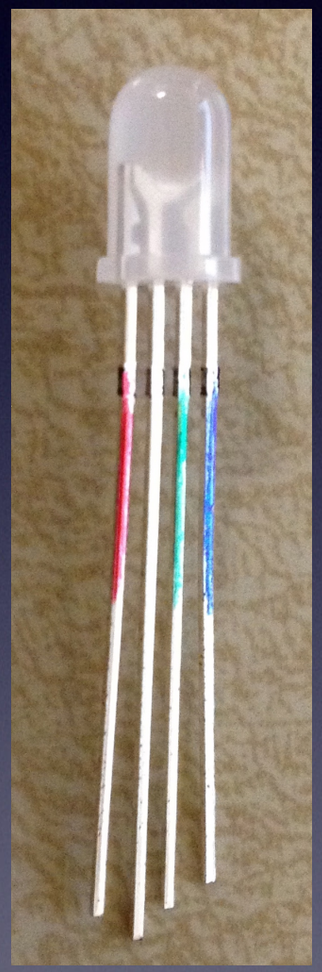
Connecting up the hardware

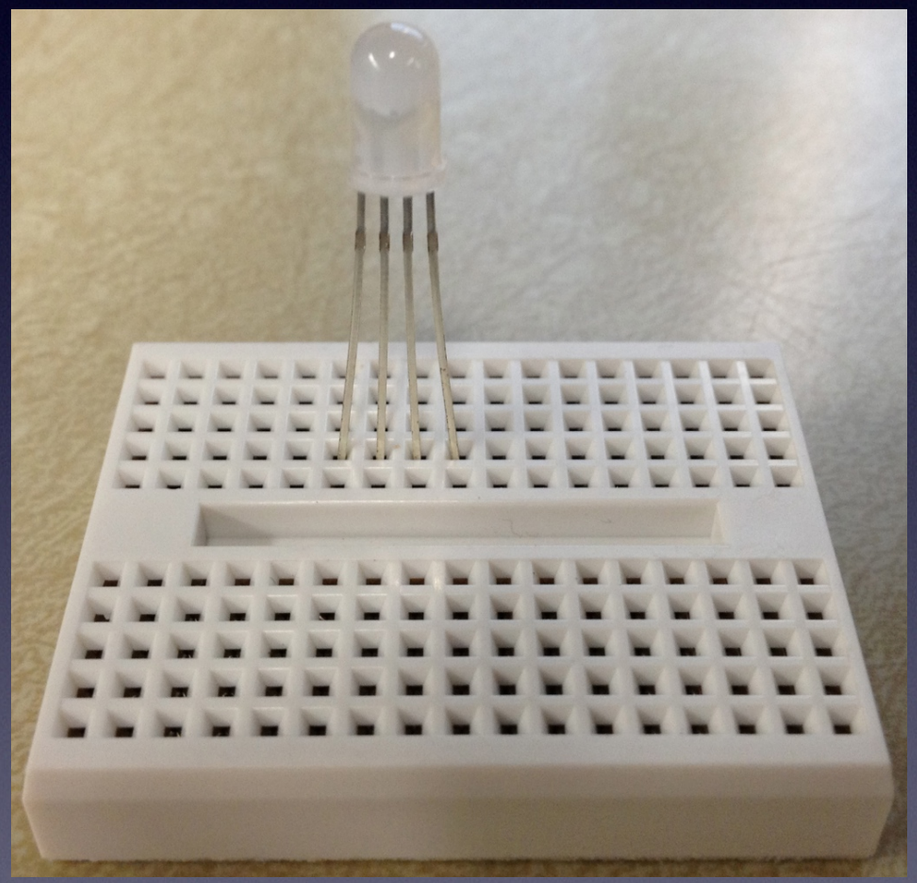


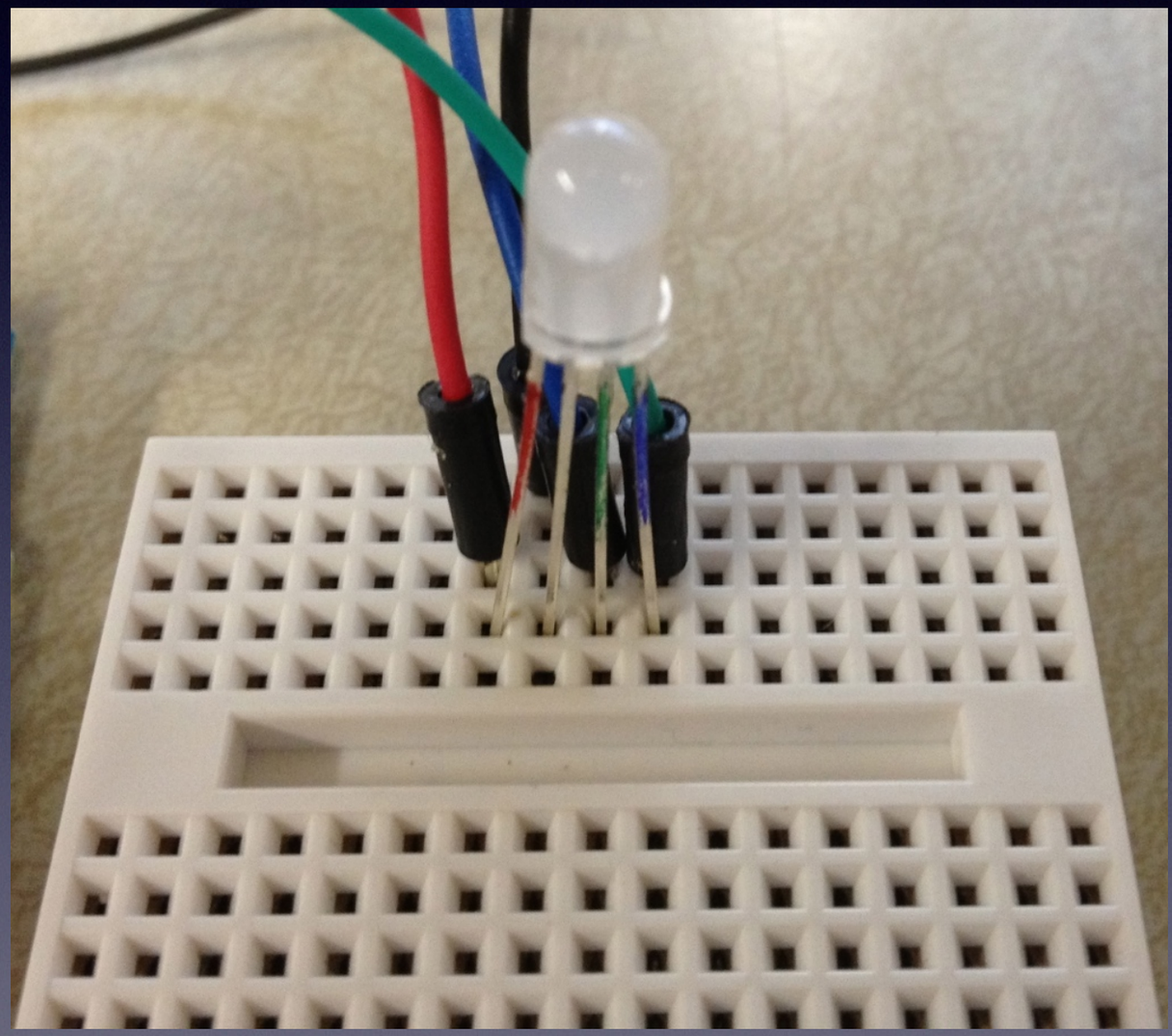




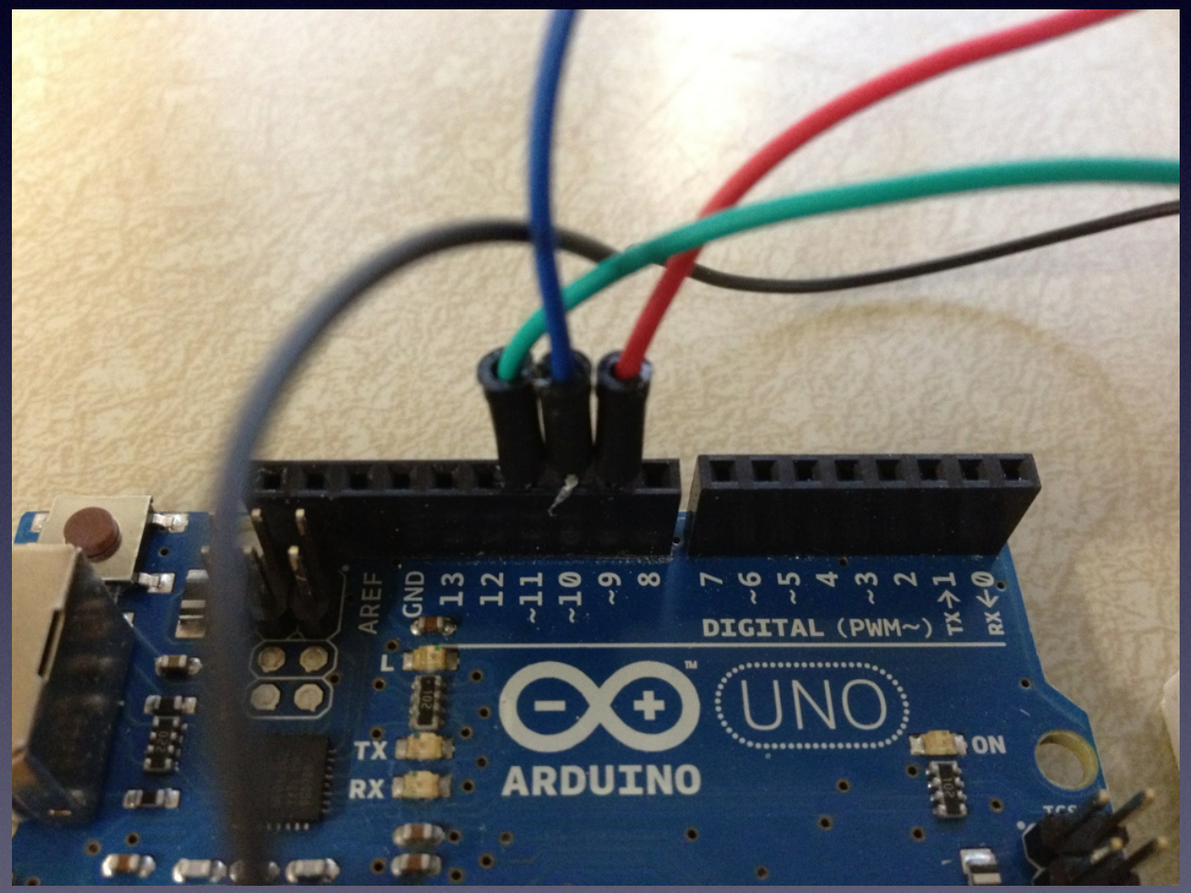
Connecting up the hardware

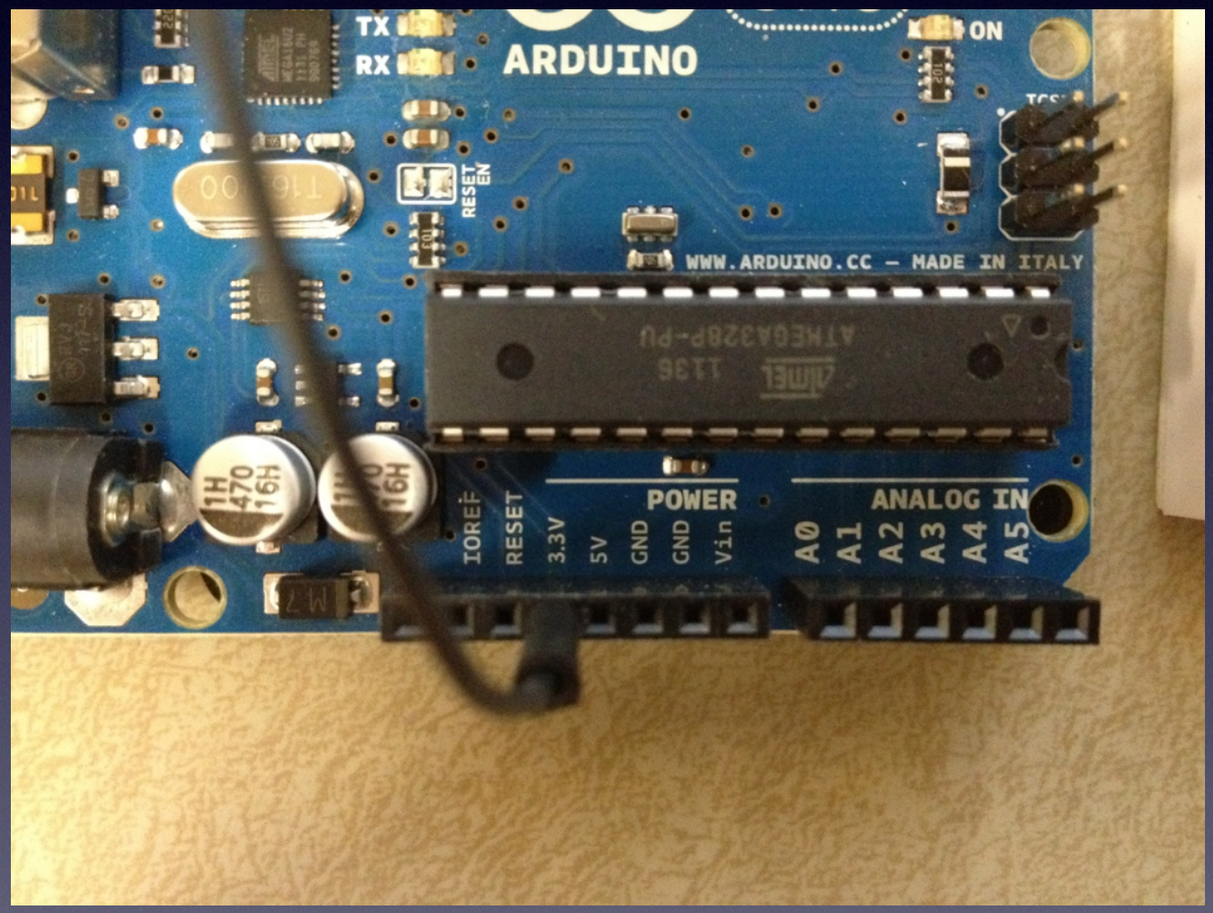




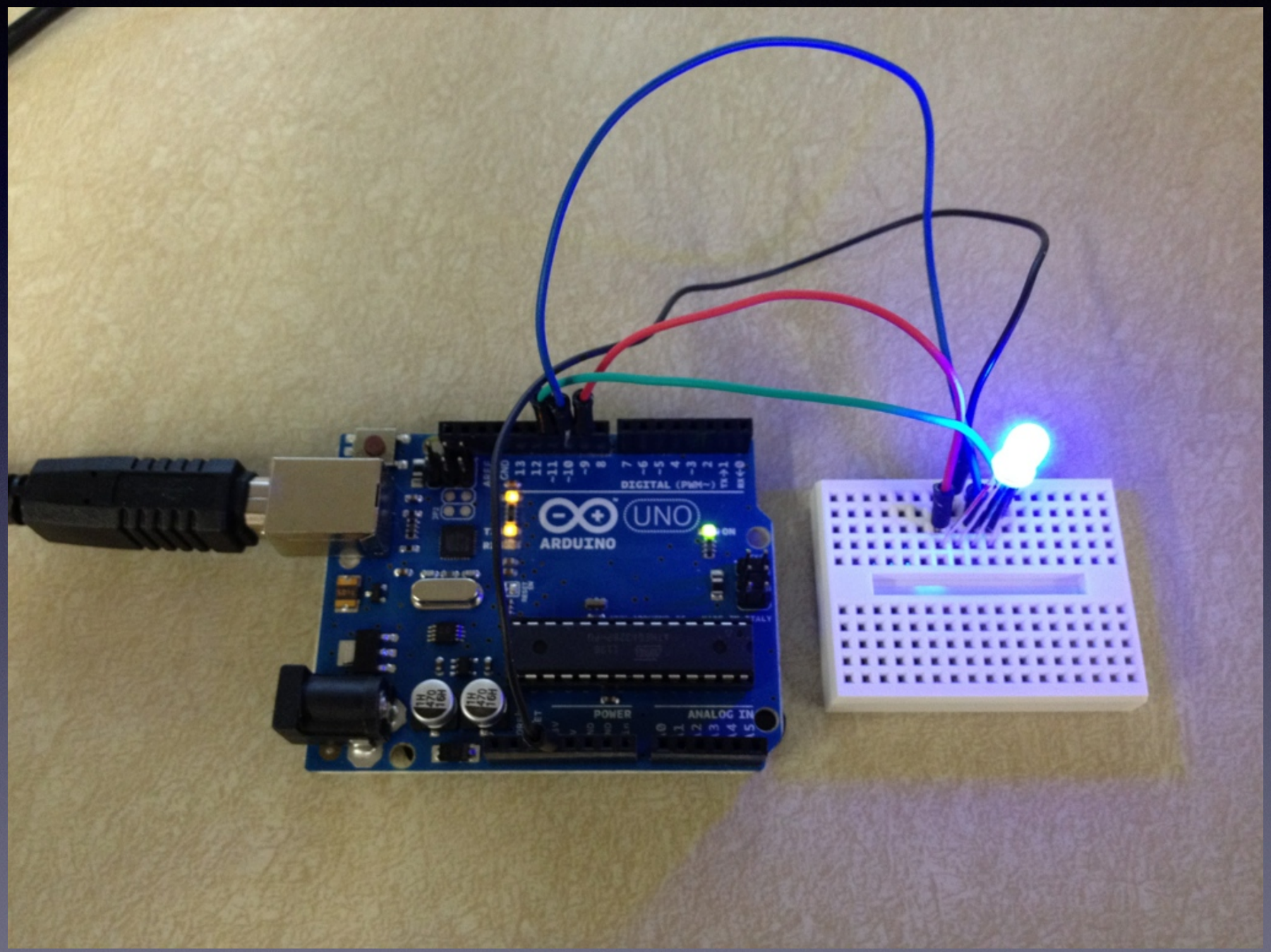


Connecting up the hardware





Connecting up the hardware

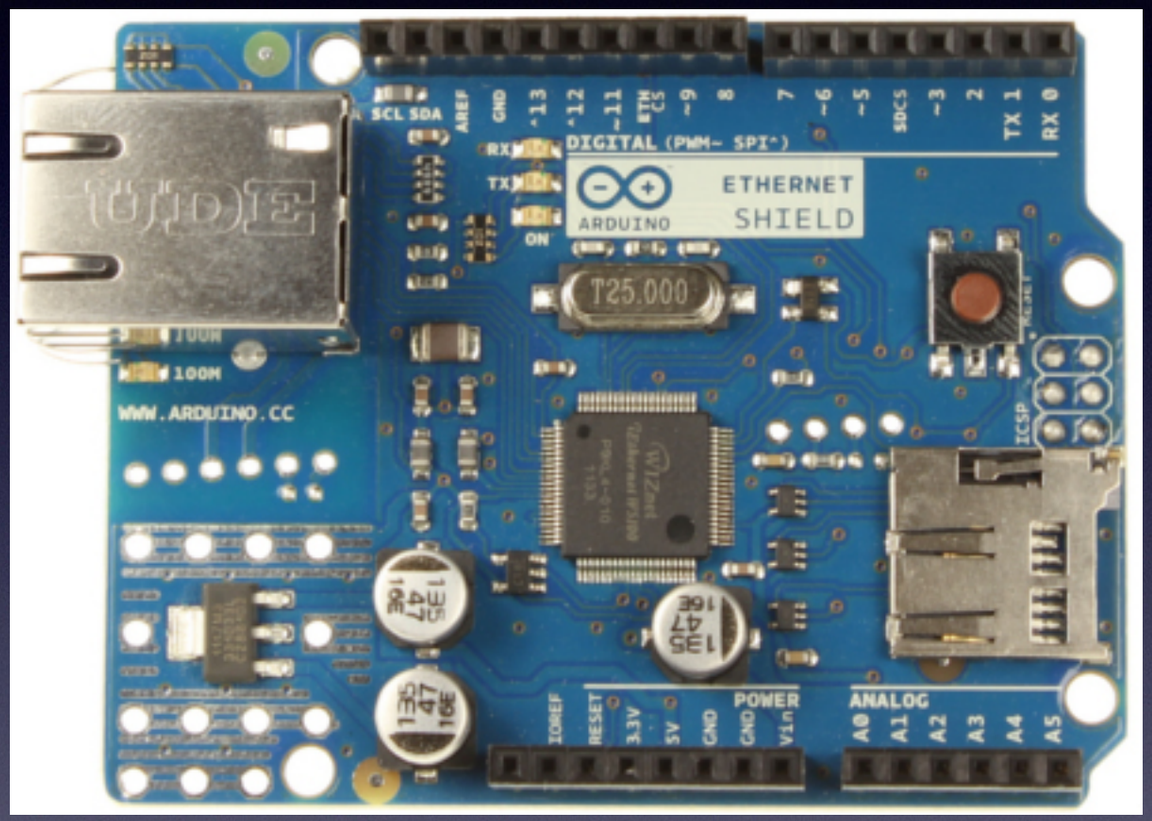


Arduino Ethernet Shield

for

Communication

Arduino Ethernet Shield



- Requires Arduino board
- Operates at 5V
- Ethernet Controller: W5100 with internal 16K buffer
- Connection speed: 10/100Mb
- Connection with Arduino on SPI port

Arduino Ethernet Shield

Purpose

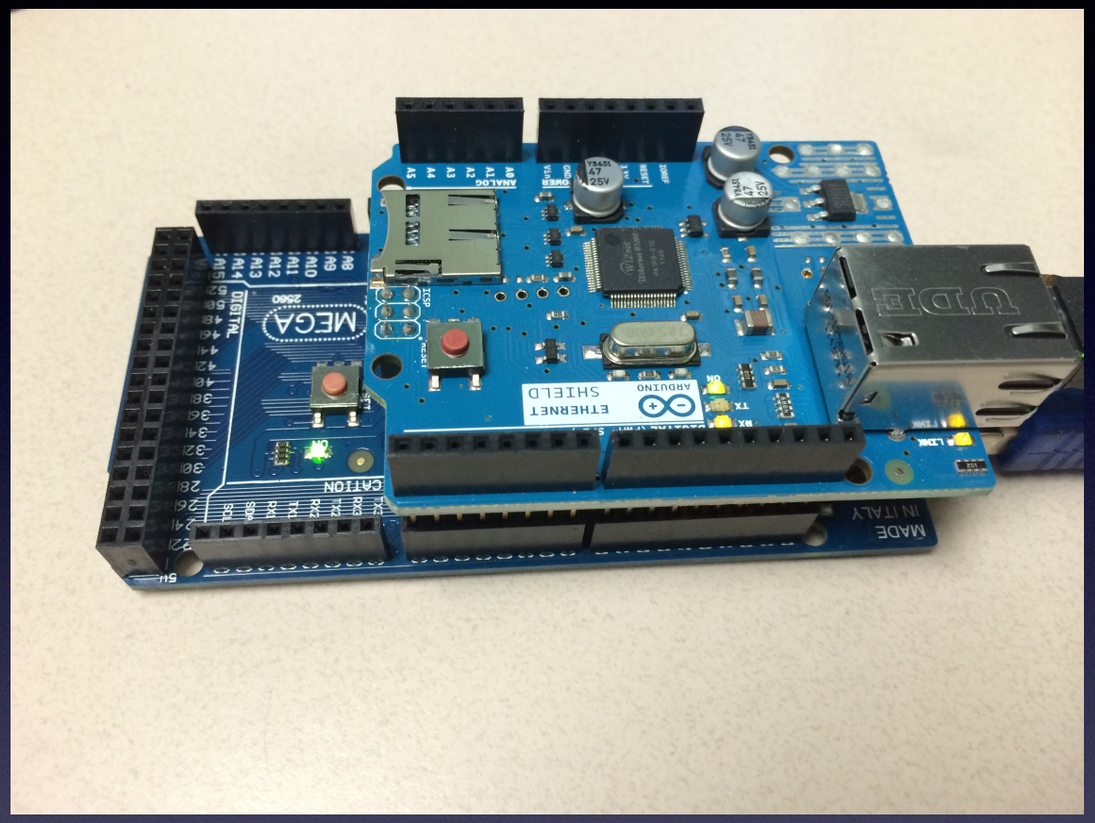
- Run web server
- Run Web client
- Communicate with other computers, Arduinos or the internet

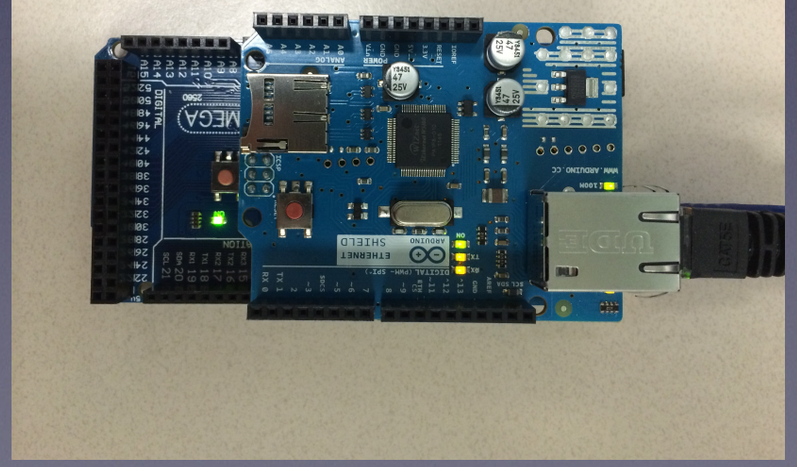
Arduino Ethernet Shield

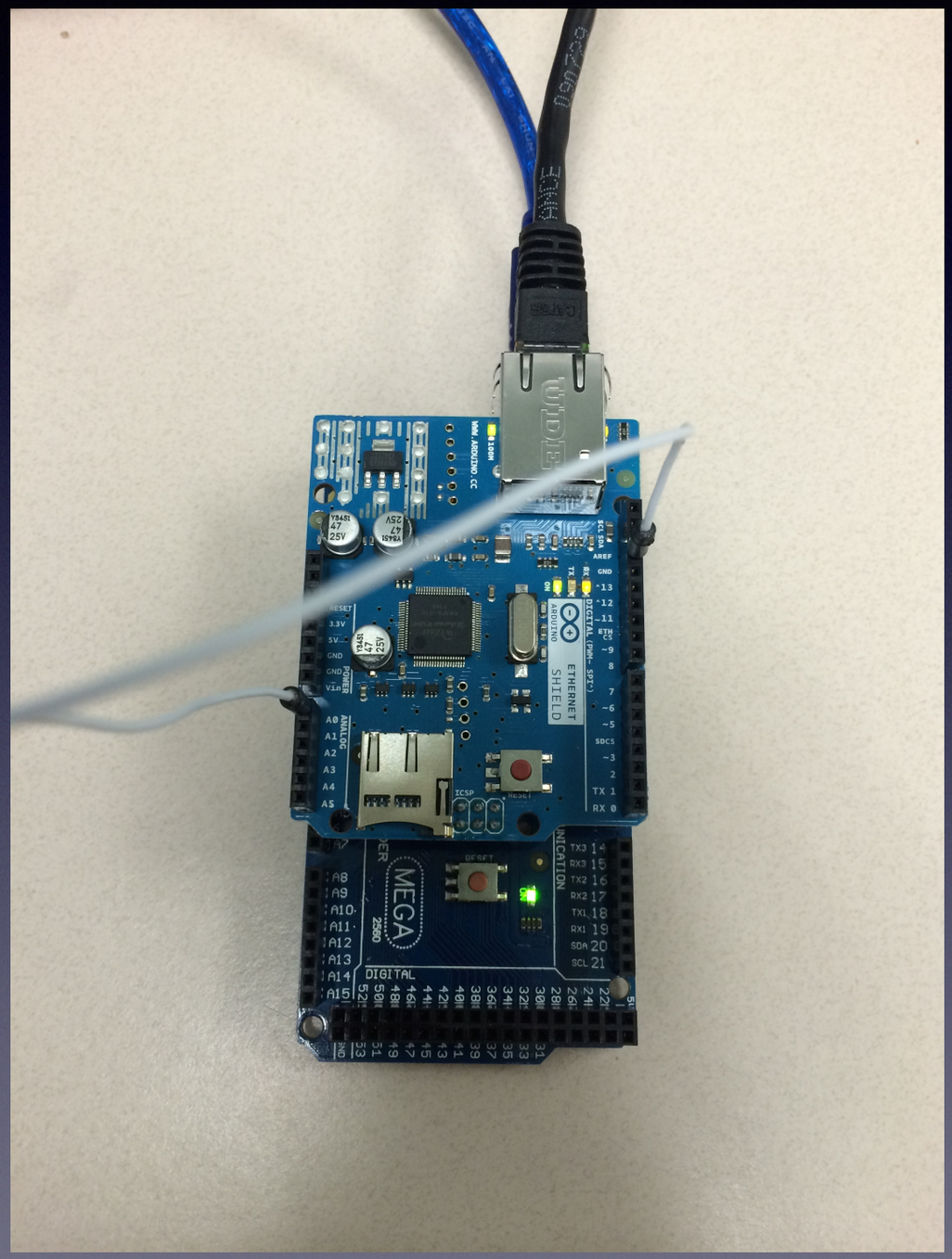
Run Arduino Example

- Select File, Examples, Ethernet, Web Server
- Set IP address for the server
- Upload code to Arduino

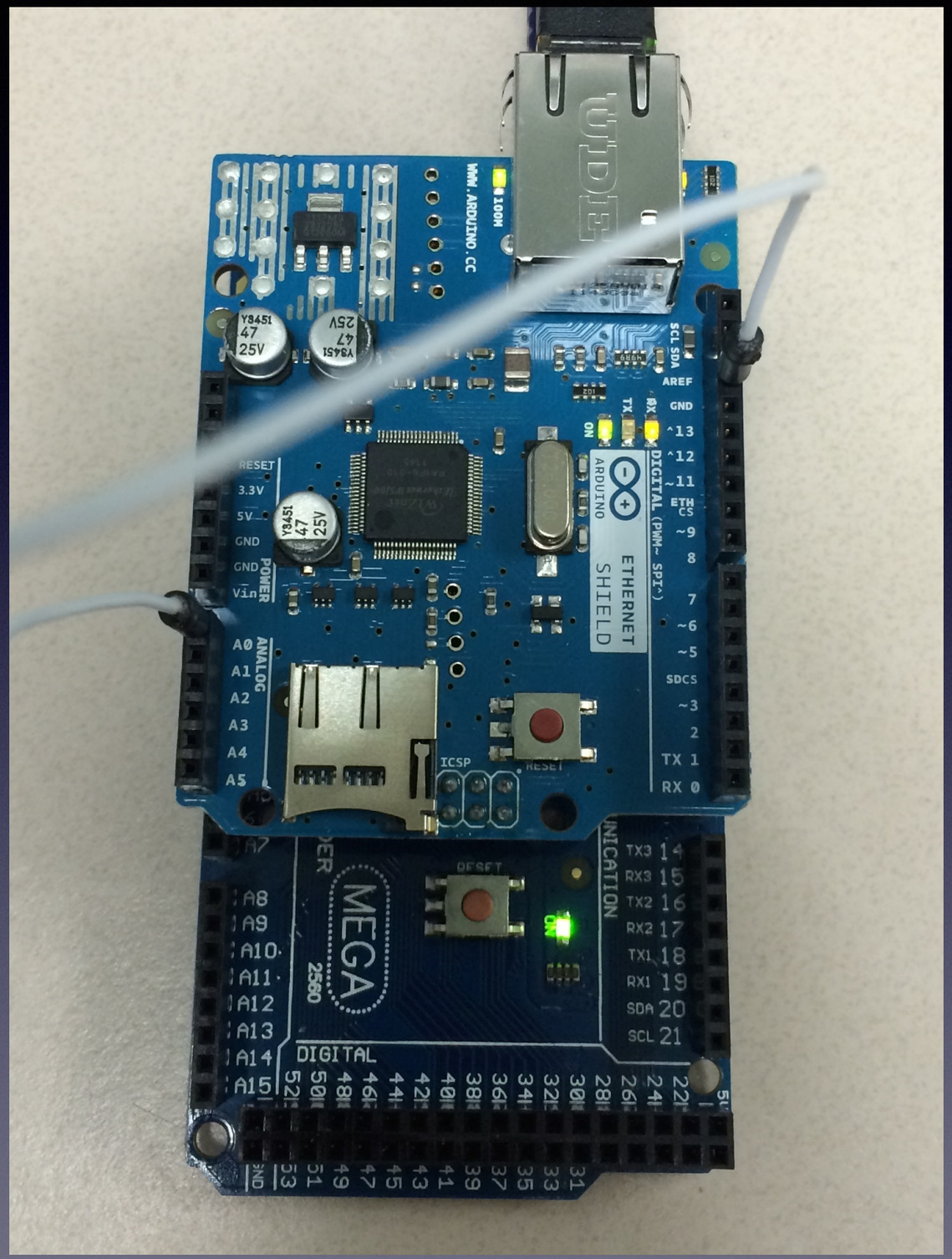
Connecting up the hardware







Ethernet Shield



Ethernet Shield

WebServer | Arduino 1.0.5

WebServer §

```
/*
  Web Server

 A simple web server that shows the value of the analog input pins.
 using an Arduino Wiznet Ethernet shield.

 Circuit:
 * Ethernet shield attached to pins 10, 11, 12, 13
 * Analog inputs attached to pins A0 through A5 (optional)

 created 18 Dec 2009
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe

 */

#include <SPI.h>
#include <Ethernet.h>

// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(10,192,92,199);

// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);

void setup() {
 // Open serial communications and wait for port to open:
  Serial.begin(9600);
   while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
```

Done uploading.

```
Binary sketch size: 13,382 bytes (of a 258,048 byte maximum)
```

75 Arduino Mega 2560 or Mega ADK on /dev/tty.usbmodemfd1411

Analog input 0 was grounded

Speed Dial | 10.192.92.199

10.192.92.199

analog input 0 is 0
analog input 1 is 171
analog input 2 is 230
analog input 3 is 304
analog input 4 is 348
analog input 5 is 345

Have fun!!!

There a tons of fun projects out there to get inspiration from!

Questions?