ARDUINO DEVELOPMENT FOR BEGINNERS

PRESENTED BY RICHARD GOWEN (@alt_bier)

V3 – Updated for BSidesDFW 2020 HHV

This Slide Deck Is Available at https://altbier.us/arduino/

WHAT IS ARDUINO?

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

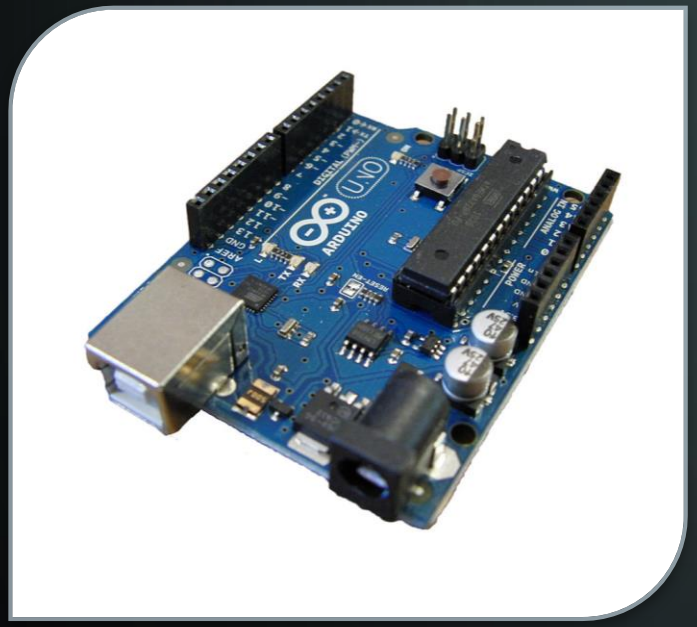
Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs.

ARDUINO BOARDS

Arduino boards come in many shapes and sizes to fit a wide variety of applications. Here is a list of just some of the Arduino boards available.

- Arduino Uno
- Arduino Leonardo
- Arduino Due
- Arduino Yun
- Arduino Tre
- Arduino Micro
- Arduino Robot
- Arduino Esplora
- Arduino Mega ADK
- Arduino Ethernet
- Arduino Mega 2560
- Arduino Mini
- LilyPad Arduino USB
- LilyPad Arduino Simple
- LilyPad Arduino SimpleSnap
- LilyPad Arduino
- Arduino Nano
- Arduino Pro Mini
- Arduino Pro
- Arduino Fio

ARDUINO BOARDS



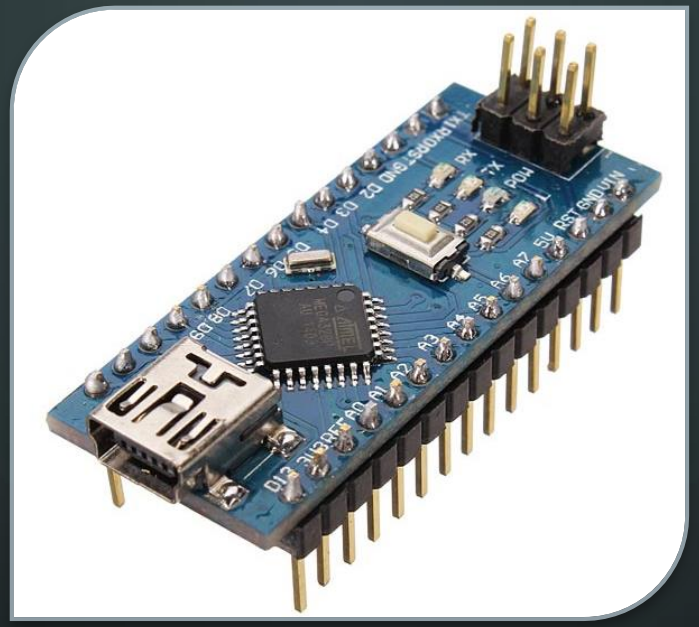
ARDUINO UNO

Processor: ATmega328P

Flash: 32 kB / SRAM: 2 kB

Digital IO Pins: 14 / Analog Input Pins: 6

Dimensions: 2.7 in × 2.1 in



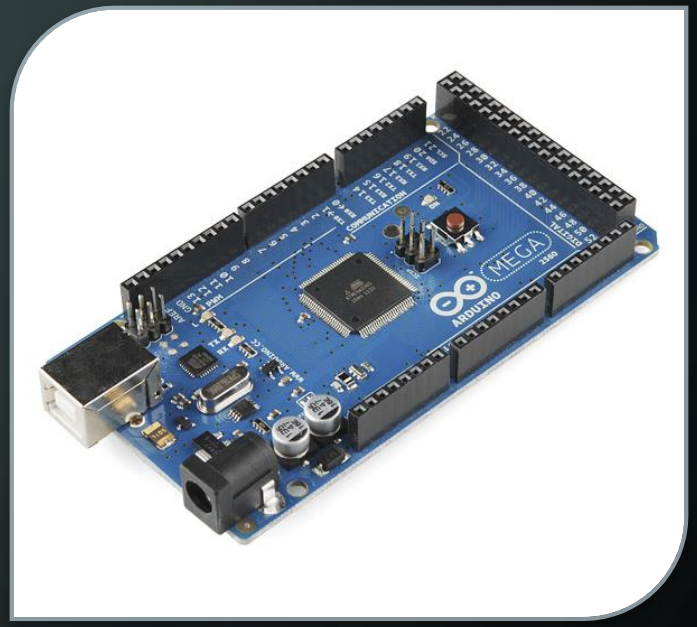
ARDUINO NANO

Processor: ATmega328

Flash: 16 kB / SRAM: 1 kB

Digital IO Pins: 14 / Analog Input Pins: 8

Dimensions: 1.70 in × 0.73 in



ARDUINO MEGA2560

Processor: ATmega2560

Flash: 256 kB / SRAM: 8 kB

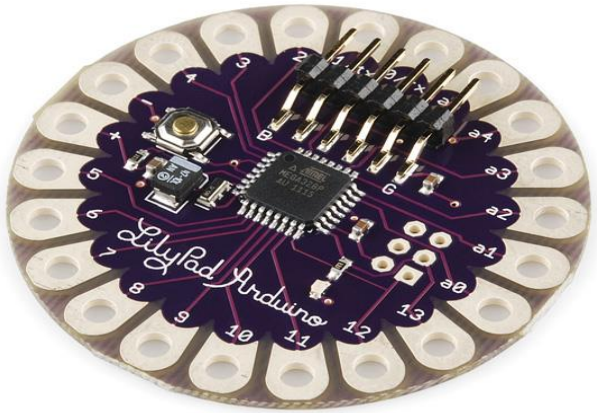
Digital IO Pins: 54 / Analog Input Pins: 18

Dimensions: 4 in × 2.1 in

ARDUINO BOARDS

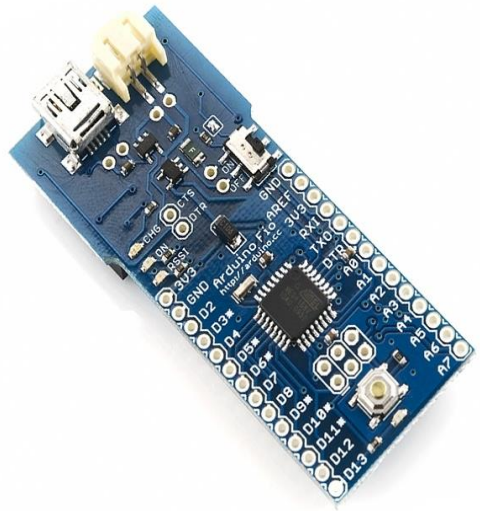
ARDUINO ETHERNET

Processor: ATmega328
Flash: 32 kB / SRAM: 2 kB
Digital IO Pins: 14 / Analog Input Pins: 6
Built-in Ethernet Adapter instead of USB
Dimensions: 2.7 in × 2.1 in



LILYPAD ARDUINO

Processor: ATmega168V
Flash: 16 kB / SRAM: 1 kB
Digital IO Pins: 14 / Analog Input Pins: 8
Wearable Design
Dimensions: 2 in Round

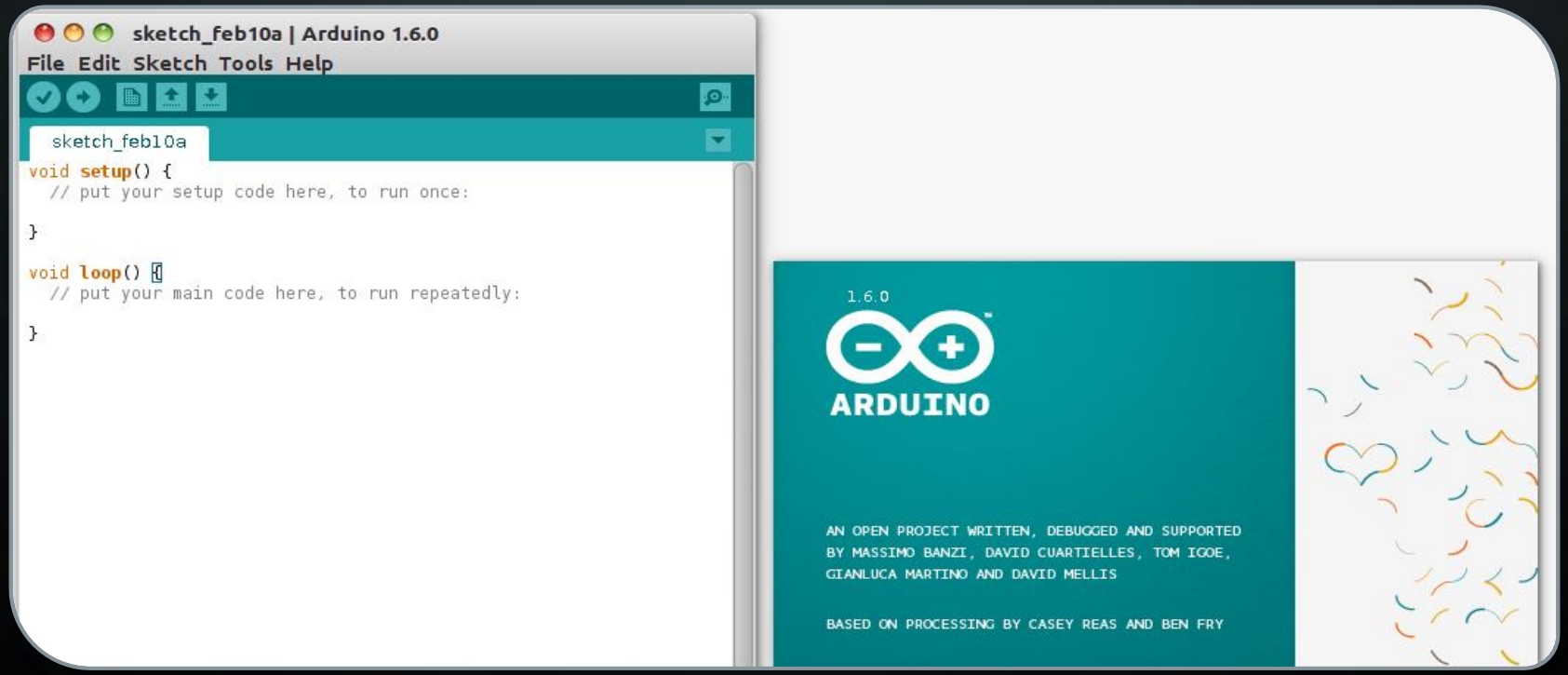


ARDUINO FIO

Processor: ATmega328P
Flash: 32 kB / SRAM: 2 kB
Digital IO Pins: 14 / Analog Input Pins: 8
Dimensions: 2.6 in × 1.1 in

ARDUINO IDE SOFTWARE

The Arduino integrated development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.



ARDUINO IDE SOFTWARE

You can download the Arduino IDE Software here: https://arduino.cc

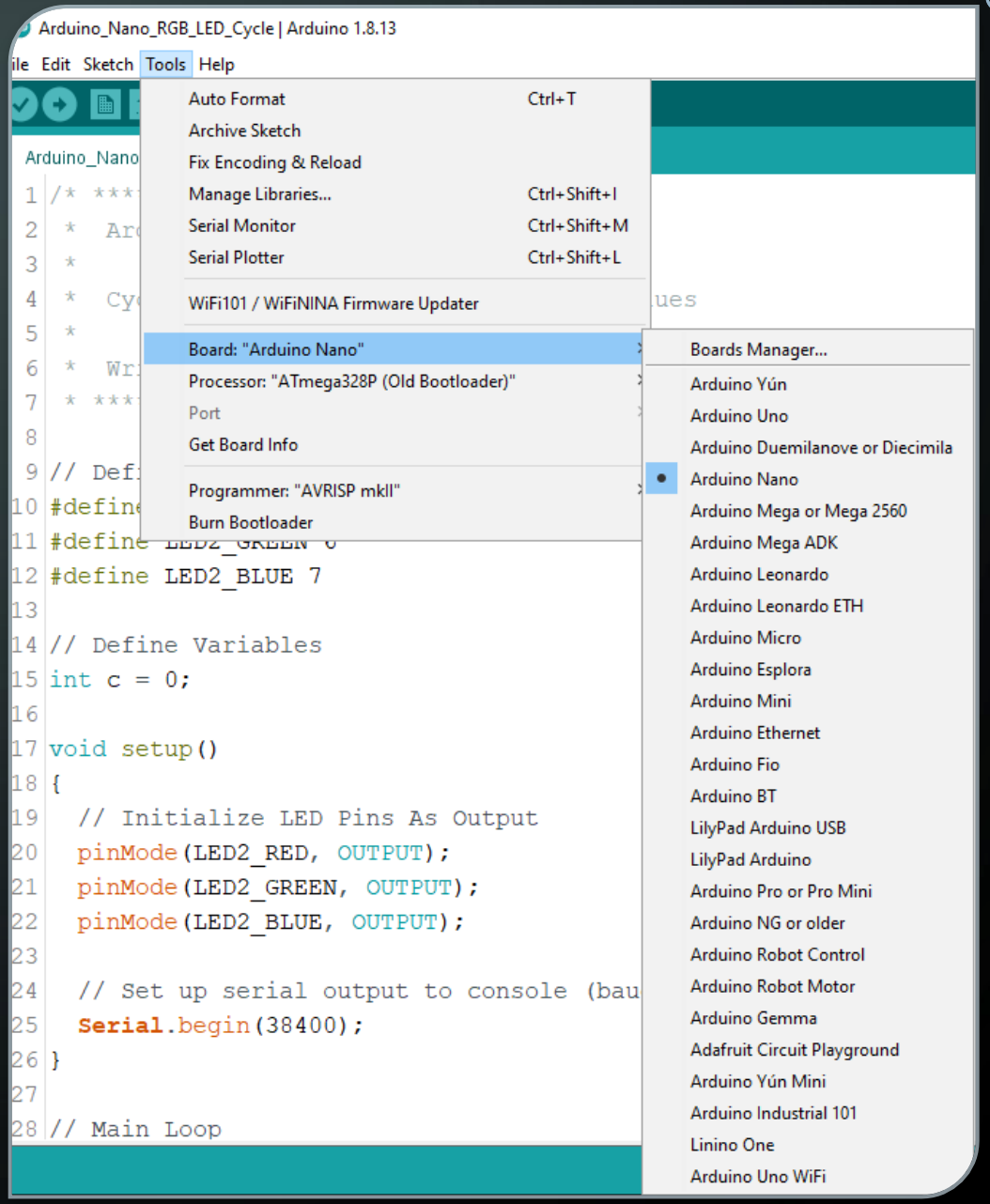
Click on Software > Downloads which will present you with options based on your OS.

There are versions for Windows, Linux, and Mac OS.

ARDUINO IDE SOFTWARE

Once you install the Arduino Software on your workstation simply choose the Arduino board that you wish to interface with and the processor type. It should auto detect the COM port on your workstation it is physically connected to, but if not you can manually select that as well.

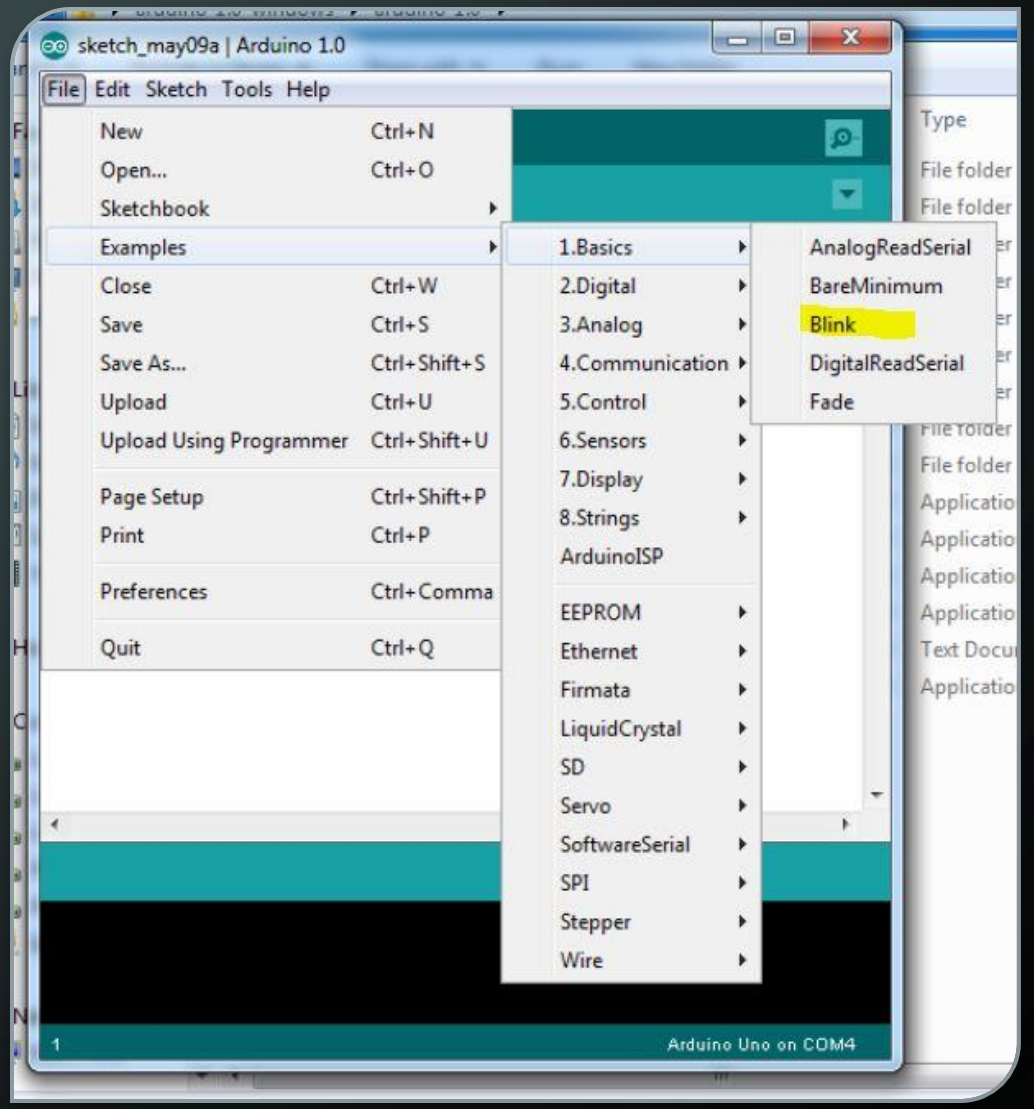
At this point you can upload code to program the Arduino or interface with the Arduino's console to view its output.



ARDUINO IDE SOFTWARE

The Arduino Software comes with several example programs that can be used to get started quickly programming your Arduino

There are also several online resources you could use to download Arduino code that will allow you to program your Arduino to do a variety of things.



ELECTRONICS 101

The next few slides will provide a basic review of electronics.

A working knowledge of some basic electronics concepts is necessary when working with the Arduino or any other hardware development board.

A more detailed electronics overview is available here:

https://altbier.us/electronics/

ELECTRIC CURRENT TYPES

There are two types of electric current types:
Alternating Current (AC), and **Direct Current** (DC).

In AC, the direction electricity flows throughout the circuit is constantly reversing.
The frequency rate of this reversal is measured in Hertz (reversals per second).
So, when they say that the US power supply is 60 Hz, what they mean is that it is reversing 120 times per second (twice per cycle).

In DC, electricity flows in one direction between power and ground.
In this arrangement there is always a positive source of voltage and ground (0V) source of voltage.

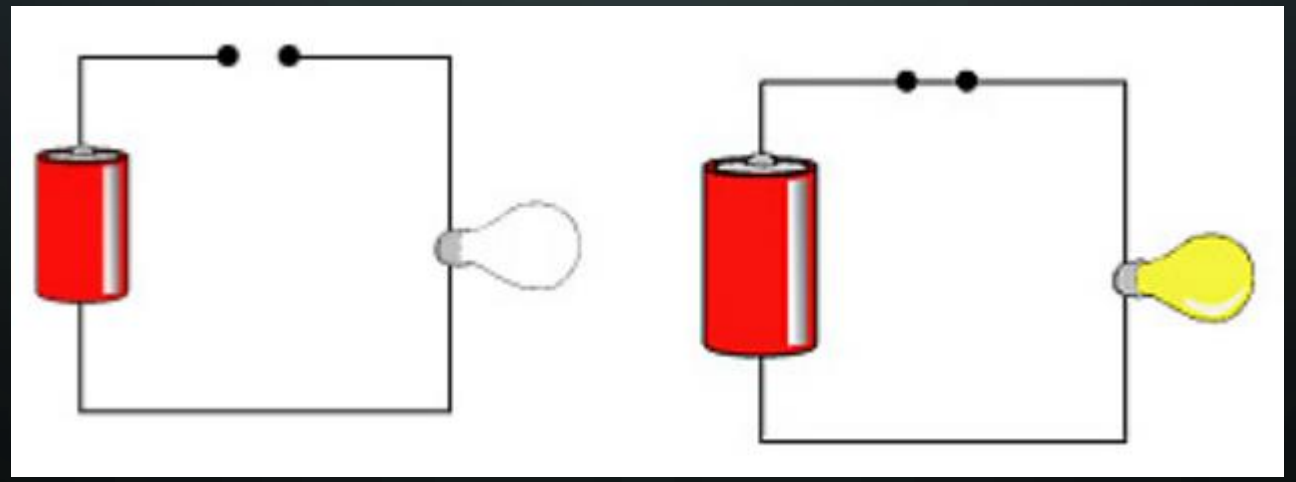
Generally, any modern electronic device with computational ability uses DC.
The reason is that they use specific voltage levels to indicate binary/logical states.

OPEN AND CLOSED CIRCUITS

Electrons will not flow through an **open circuit** (left diagram)

Electrons will flow freely through a **closed circuit** (right diagram).

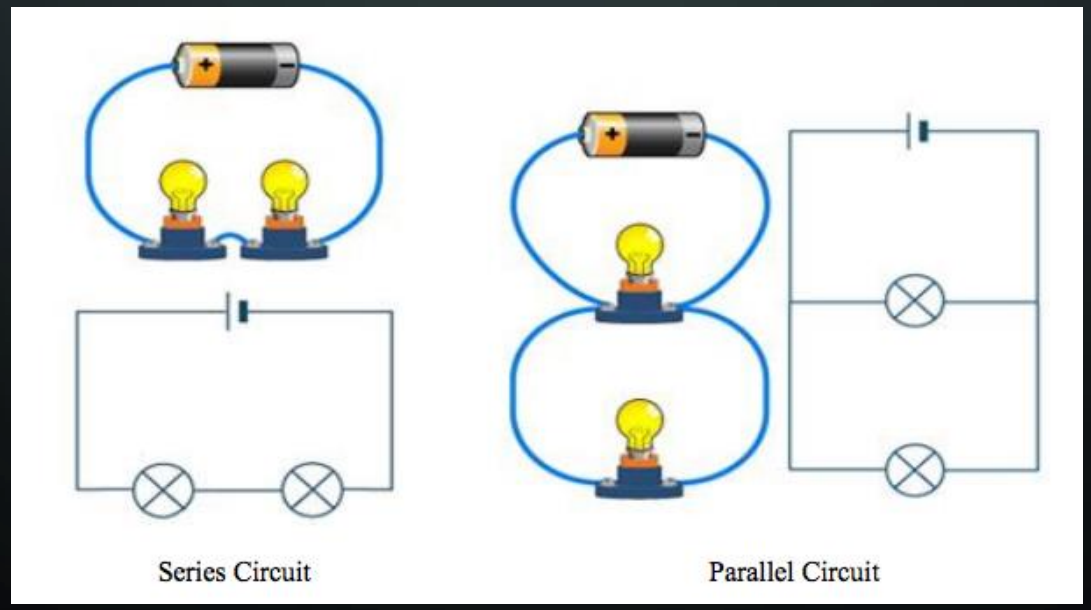
A **switch** is a device that allows for a circuit to be toggled between its open and closed states.



PARALLEL VS. SERIES CIRCUITS

In a **series circuit**, the current through each of the components is the same, and the voltage across the circuit is the sum of the voltages across each component.

In a **parallel circuit**, the voltage across each of the components is the same, and the total current is the sum of the currents through each component.



RESISTORS

A **Resistor** is a passive two-terminal electrical component that implements resistance as a circuit element.

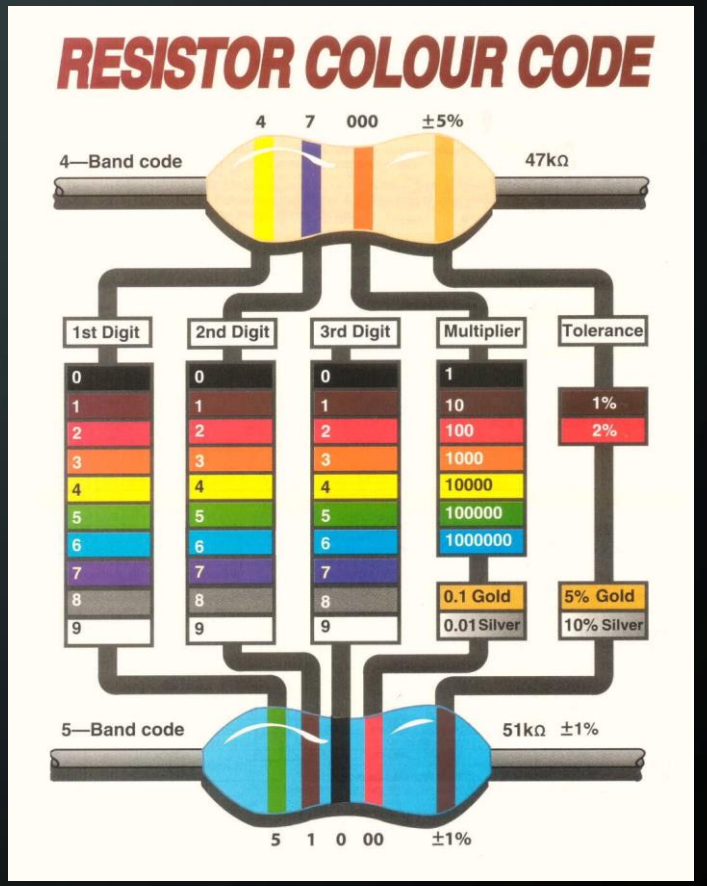
Resistors may be used to reduce current flow, and, at the same time, may act to lower voltage levels within circuits.

The amount of resistance a resistor adds to a circuit as measured in **ohms (Ω)**.

Resistors are **non-polar** components (i.e. can be placed in a circuit in either direction).

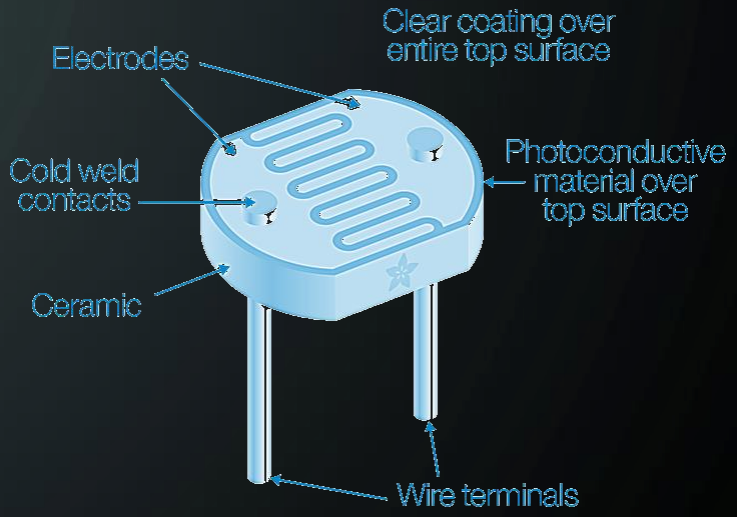
The resistance value is usually printed onto the component in the form of colored stripes. Some resistors have four stripes, and some have five.

The resistor color chart (right) can be used to determine the value.



PHOTORESISTORS

A **Photoresistor** also known as a Light Dependent Resistor (LDR) is a passive component that decreases resistance with respect to receiving luminosity (light) on the component's sensitive surface. The resistance of a photoresistor decreases with increase in light



Its resistance depends on the intensity of light incident upon it:

- Under dark conditions, resistance is very high (M Ω).
- Under bright condition, resistance is lowered (~100 - 200 Ω).

Photoresistors are **non-polar** components (i.e. can be placed in a circuit in either direction).

A photoresistor can be applied in light-sensitive detector circuits and light-activated and dark-activated switching circuits acting as a resistance semiconductor.

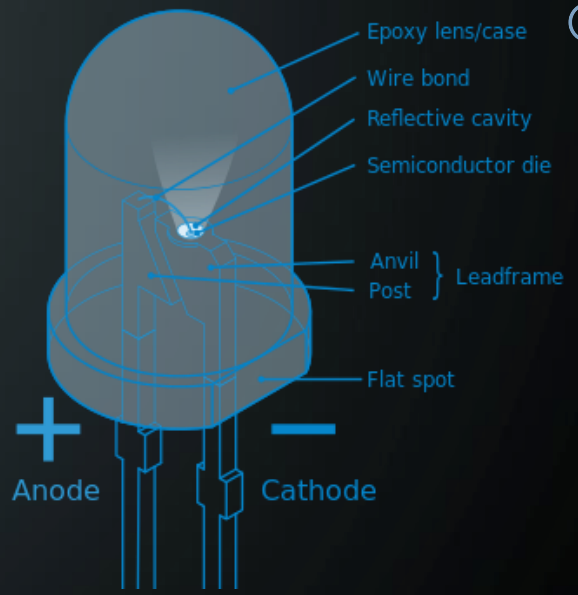
DIODES AND LIGHT EMITTING DIODES

A **Diode** is a two-terminal electronic component that conducts primarily in one direction; has low (ideally zero) resistance to current in one direction, and high (ideally infinite) resistance in the other.

This directionality indicates that diodes are **polar** components (i.e. must be placed in a circuit in the proper direction).

A diodes two electrodes are called Anode and Cathode.

- The **Anode** is the positively charged electrode
- The **Cathode** is the negatively charged electrode



A light-emitting diode (LED) emits light when activated.

When a suitable voltage is applied, electrons that pass through the device release energy in the form of photons. This effect is called electroluminescence, and the color of the light corresponds to the energy of the photons released.

ARDUINO SIMPLE PROJECTS

Two very simple Arduino projects are outlined in this section.

- Making an LED Blink
- Making an RGB LED Cycle through colors

These projects are Arduino introduction staples. They have been presented by many others and well documented on the Internet because they are so effective.

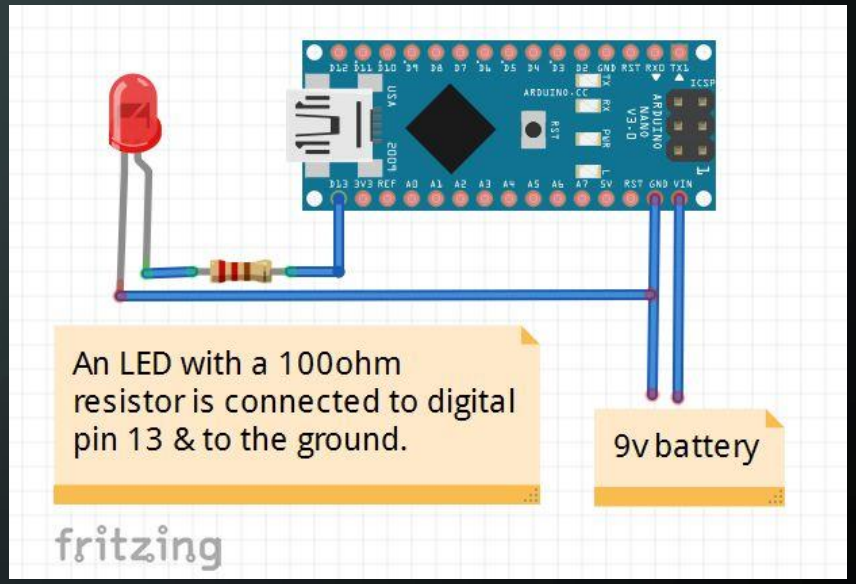
These will be outlined here as typically presented and then again in the intermediate project section that follows as we learn to do more with these same simple circuits.

Feel free to skip ahead to the intermediate project section where these projects will be outlined in a slightly different way.

ARDUINO NANO MAKING ONE LED BLINK

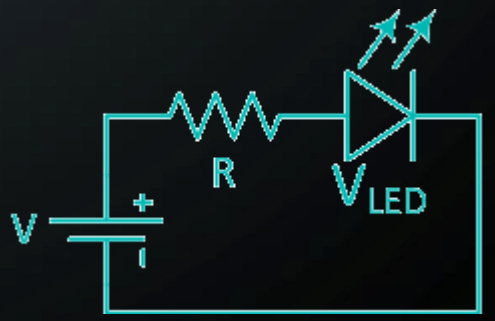
Things you will need:

- Arduino Nano
- A breadboard
- An LED
- A resistor 100 ohm
- Jumper wires



Wiring:

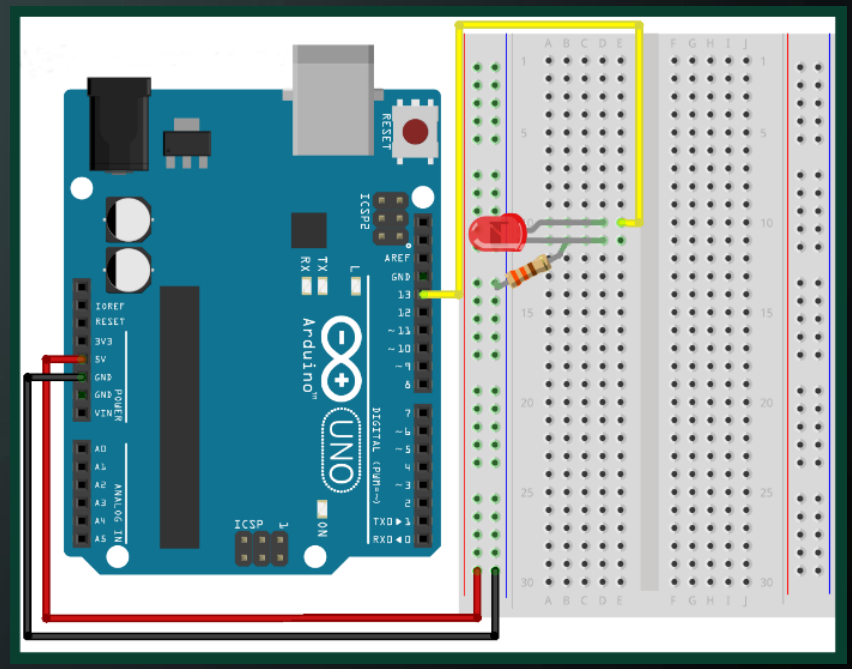
- You have to attach the LED to the Nano in a circuit as shown in the example image.
- Connect the LED's positive end to the resistor and the negative end to the Nano’s ground. Connect the other end of the resistor to the Nano's digital pin 13 to complete a circuit.
- To power the Nano board, you can use a USB cable or connect an external battery positive to VIN and negative to ground.



ARDUINO UNO MAKING ONE LED BLINK

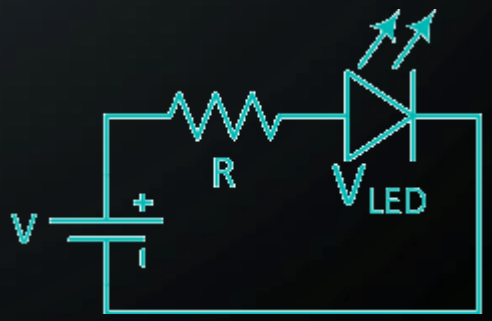
Things you will need:

- Arduino Uno
- A breadboard
- An LED
- A resistor 100 ohm
- Jumper wires



Wiring:

- You have to attach the LED to the Uno in a circuit as shown in the example image.
- Connect the LED's positive end to the resistor and the negative end to the Uno’s ground. Connect the other end of the resistor to the Uno's digital pin 13 to complete a circuit.
- To power the Uno board, you can use a USB cable or connect an external battery.



ARDUINO MAKING ONE LED BLINK

Programming the Arduino

(This code works for the Nano & Uno):

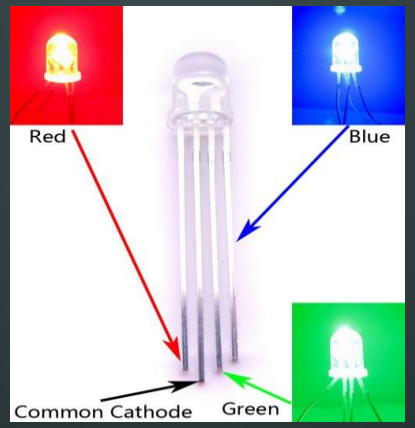
- Connect the Arduino to your workstation via USB
- The LED blink code should be in your Arduino IDE software examples.
- Open the example blink code in
  File -> Examples -> Basics -> Blink
- Upload this code to the Arduino by clicking on the upload button
- You should see the LED start blinking on and off in 1 second intervals

This Code Is Available Here: https://altbier.us/arduino/BlinkOneLED.txt

ARDUINO UNO MAKING AN RGB LED CYCLE

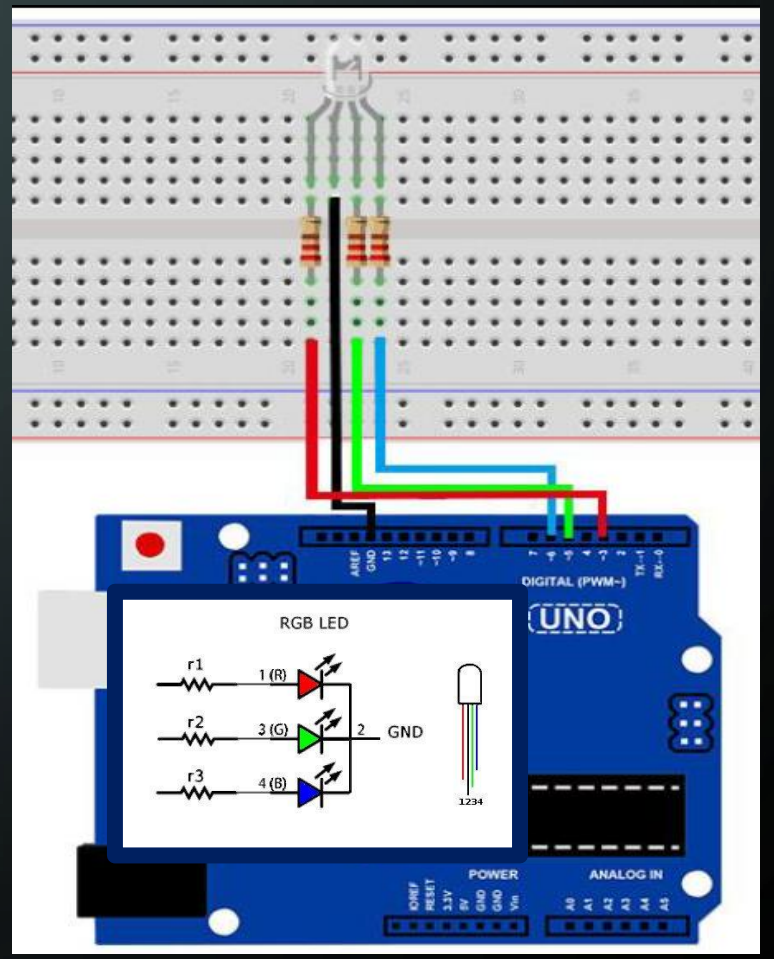
Things you will need:

- Arduino Uno
- A breadboard
- An RGB LED
- Three resistors 100 ohm
- Jumper wires



Wiring:

- You have to attach the LED to the Uno in a circuit as shown in the example image.
- Connect the LED's positive ends to the resistors and the negative end to the Uno's ground. Connect the other ends of the resistors to the Uno's digital pins 3, 5, and 6 as shown to complete a circuit.
- To power the Uno board, you can use a USB cable or connect an external battery.



ARDUINO UNO MAKING AN RGB LED CYCLE

Programming the Arduino Uno:

- Connect the Arduino Uno to your workstation via USB
- Type in the example code (right) into your Arduino IDE software as a new project
- Upload this code to the Arduino by clicking on the upload button
- You should see the RGB LED start transitioning between colors pausing on red, green, and blue for one second before transitioning to the next color

Project_RGB_LED

```
// Define Pins
#define RED 3
#define GREEN 5
#define BLUE 6

void setup()
{
  // Initialize Pins As Output
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
}

// Main Loop
void loop()
{
  // Define Color Level Variables
  int redValue = 0;    // Valid Values Between 0 and 255
  int greenValue = 0;
  int blueValue = 0;
  for (int c = 1; c < 4; c += 1) // Loop 3 times for the 3 colors
  {
    for (int i = 0; i < 256; i += 1) // Fades one color to the next
    {
      if (c == 1) { redValue = (255 - i); greenValue = i; }
      if (c == 2) { greenValue = (255 - i); blueValue = i; }
      if (c == 3) { blueValue = (255 - i); redValue = i; }
      analogWrite(RED, redValue);
      analogWrite(GREEN, greenValue);
      analogWrite(BLUE, blueValue);
      delay(10);
    }
    // Fade complete, pause on single full color for 1 second
    analogWrite(RED, redValue);
    analogWrite(GREEN, greenValue);
    analogWrite(BLUE, blueValue);
    delay(1000);
  }
}
```

This Code Is Available Here: https://altbier.us/arduino/RGB_LED.txt

ARDUINO INTERMEDIATE PROJECTS

This next section will outline some intermediate Arduino projects.

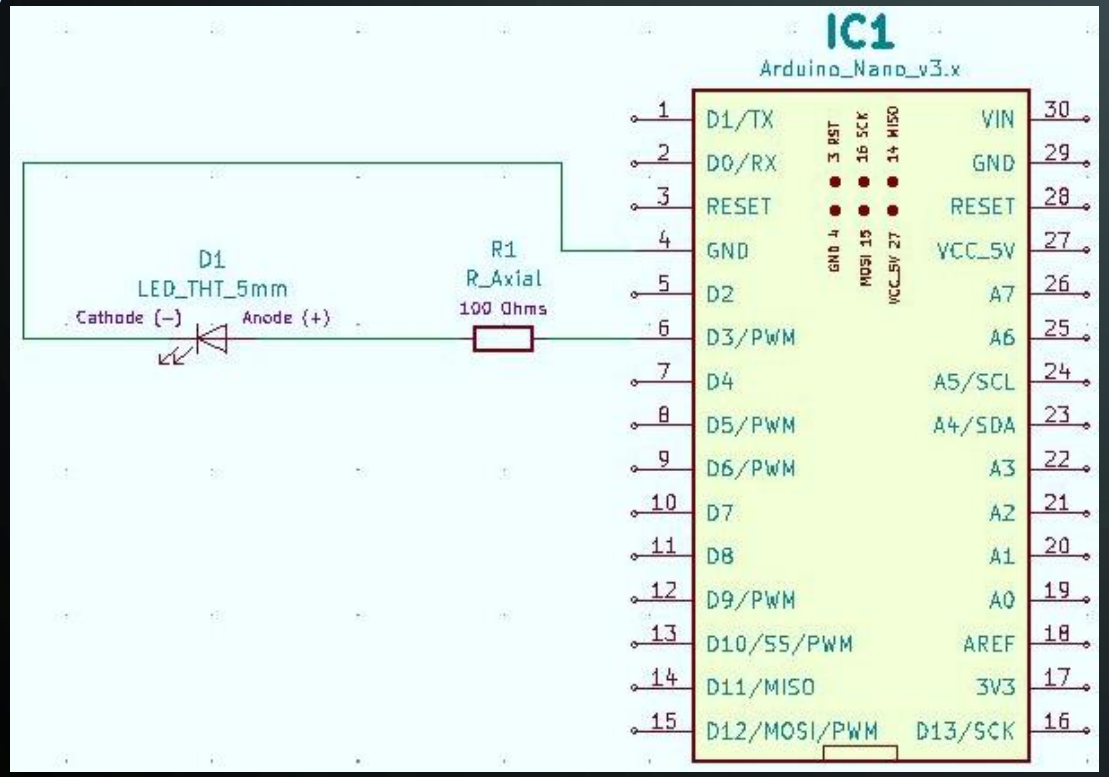
These projects will be centered around three different physical circuit layouts with several Arduino code blocks for each.

- LED connected to Arduino Nano PWM – Lab HHV2020_01
  - Digital Blink LED
  - Analog Fade LED
  - Analog Step LED
- RGB LED and Tactile Switch connected to Arduino Nano – Lab HHV2020_02
  - Color Cycle RGB LED
  - Use Tactile Switch (Button) to Color Change RGB LED
- LED and Photoresistor connected to Arduino Nano – Lab HHV2020_03
  - Use Light via a Photoresistor to Control LED state (on/off)
  - Use a Photoresistor to display Luminosity levels

The Lab reference numbers refer to the BSidesDFW Hardware Hacking Village Videos which can be accessed here: **https://altbier.us/bsidesdfwHHV2020/**

LED CONNECTED TO ARDUINO NANO PWM

Schematic

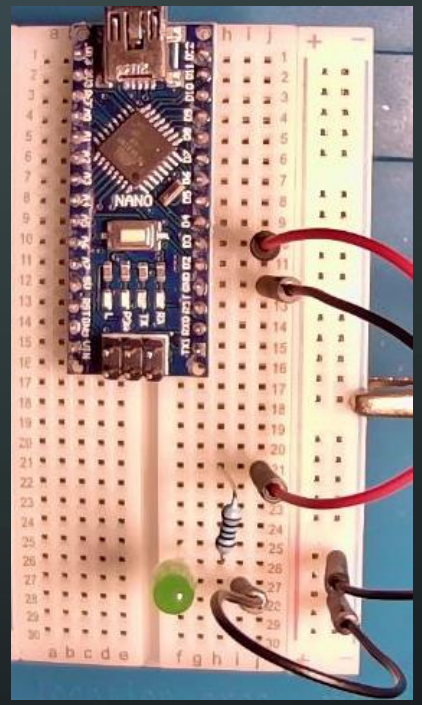


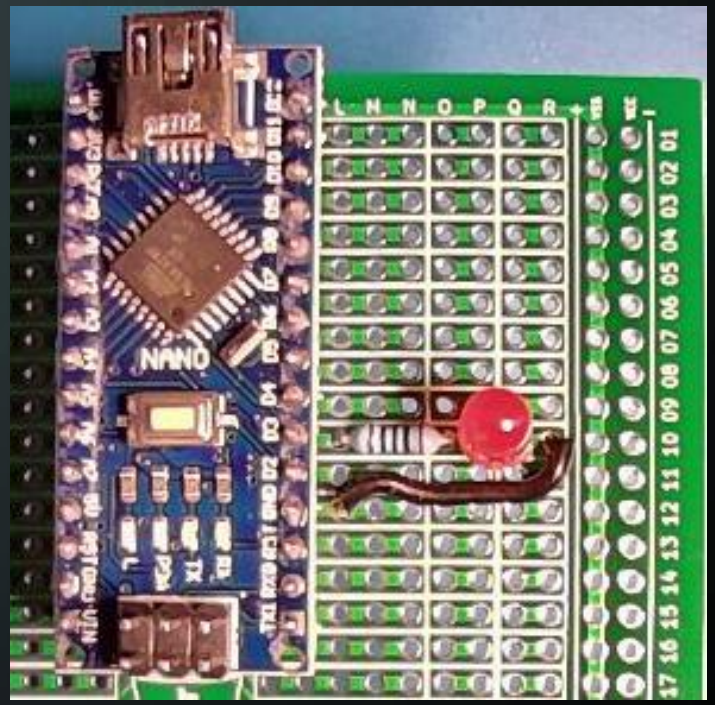
Wire up a circuit as shown in the schematic and physical layout.

Components:

- 1x Resistor 100 Ohm
- 1x LED 5mm

Physical Layout





Strip Board Connection Details

- Arduino Nano – `I1-15` and `K1-15`
- Resistor 100 Ohm – `L10` and `O10`
- LED – `P10` (Anode) and `Q10` (Cathode)
- Wire – `R10` and `L12`

LED CONNECTED TO ARDUINO NANO PWM

Blink LED (digital) Code

Use "**digitalWrite**" to turn the LED on / off by setting the voltage to HIGH (1) / LOW (0).

Note that "#define LED1 3" is equivalent to "int LED1 3" that was used in the simple project example.

```
/* *************************
*  Arduino Nano Single LED Blink
*
*  Use digitalWrite to Blink an LED
*
*  Written by @alt_bier
*
*************************
*/

// Define Pins
#define LED1 3

void setup() {
  // Initialize LED Pins As Output
  pinMode(LED1, OUTPUT);
}

// Main Loop
void loop() {
  digitalWrite(LED1, HIGH);  // turn the LED on
  delay(2000);               // wait for 2 seconds
  digitalWrite(LED1, LOW);   // turn the LED off
  delay(2000);               // wait for 2 seconds
}
```

This Code Is Available Here: https://github.com/gowenrw/BSidesDFW_2020_HHV/

LED CONNECTED TO ARDUINO NANO PWM

Fade LED (analog) Code

Use "**analogWrite**" to cycle the LED through all values between 0 (LOW) and 255 (HIGH) .

We can use these analog values because the digital pin the LED is connected to supports PWM (Pulse Width Modulation). Not all digital pins support PWM and those without would be limited to binary values 0 (LOW) and 1 (HIGH).

There is code that prints values to serial. Open the **serial monitor** with a rate of **38400 baud**. You should see the LED values being displayed as they are set in real time.

```
/* ************************
 *  Arduino Nano Single LED with PWM Fade
 *
 *  Control an LED connected to a Digital PWM (Pulse Width Modulation) Pin
 *  Use analogWrite to set LED values between 0 (LOW) and 255 (HIGH)
 *  Create a Fade effect by using all values 0-255 then 255-0
 *
 *  Digital Pins without PWM use digitalWrite with values of 0 (LOW) or 1 (HIGH)
 *
 *  Written by @alt_bier
 * ************************ */

// Define Pins
#define LED1 3

void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED1, OUTPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}

// Main Loop
void loop()
{
  // Set Delay Time [in ms]
  int DelayTime = 5;

  // Fade LED LOW to HIGH
  for(int i=0; i<255; i++){
    // Set the LED value using analogWrite
    // analogWrite is only usable on Analog pins or Digital PWM pins
    analogWrite(LED1, i);
    // Print current LED value to serial console for troubleshooting
    Serial.print(" LED_VALUE=");
    Serial.println(i);
    // Pause the loop to display LED
    delay(DelayTime);
  }

  // Fade LED HIGH to LOW
  for(int i=255; i>0; i--){
    // Set the LED value using analogWrite
    analogWrite(LED1, i);
    // Print current LED value
    Serial.print(" LED_VALUE=");
    Serial.println(i);
    // Pause the loop to display LED
    delay(DelayTime);
  }
}
```

This Code Is Available Here: **https://github.com/gowenrw/BSidesDFW_2020_HHV/**

LED CONNECTED TO ARDUINO NANO PWM

Step LED (analog) Code

Use "**analogWrite**" to cycle the LED through a set of predefined values between 0 (LOW) and 255 (HIGH) .

Just like with the previous example, we can use these analog values because the digital pin the LED is connected to supports PWM (Pulse Width Modulation).

In this example we are using a set of predefined values rather than cycling through every possible value. Notice the LED light intensity changing and how these transitions are not as smooth as the fade code.

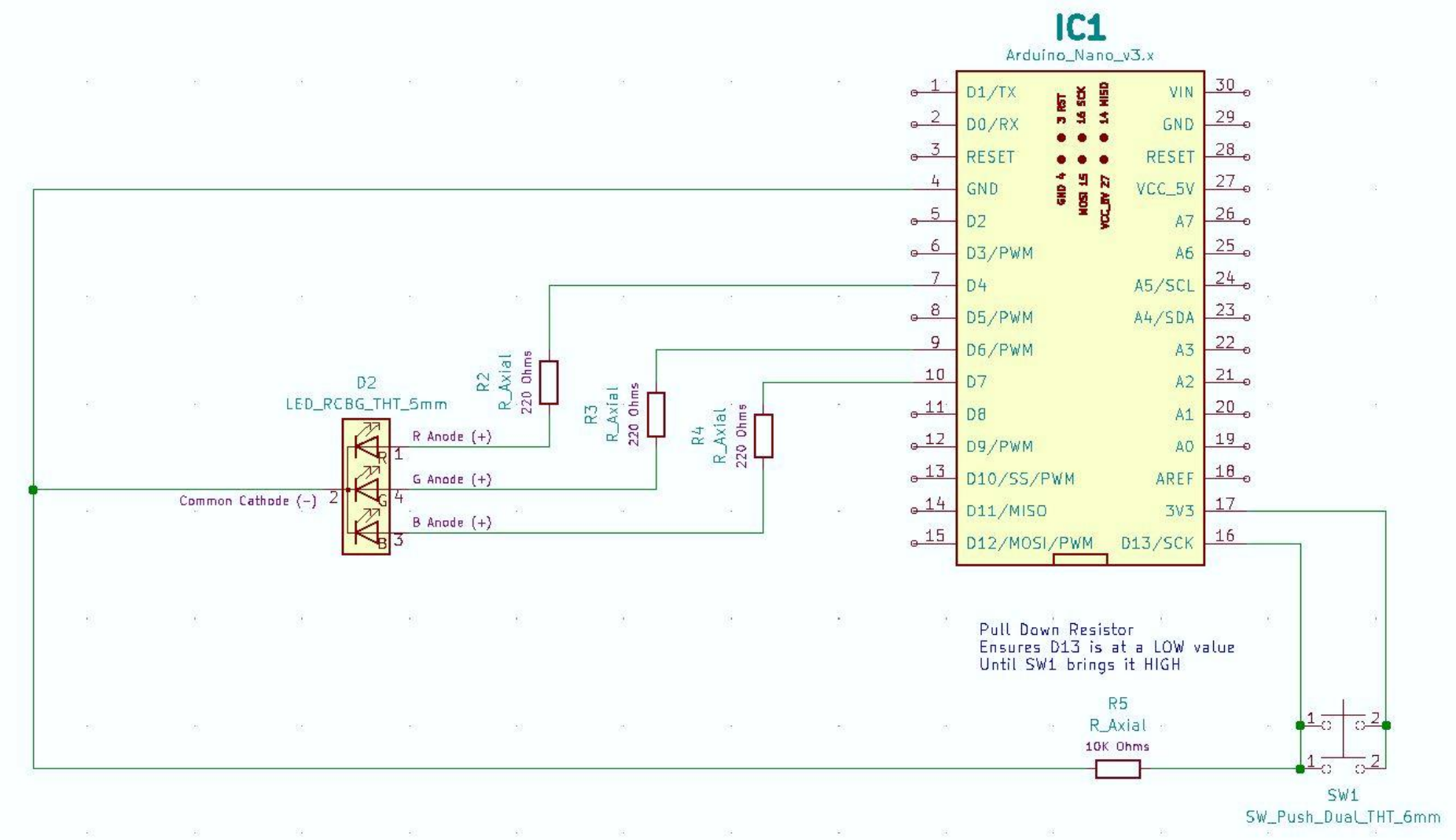
There is code that prints values to serial. Open the **serial monitor** with a rate of **38400 baud**. You should see the LED values being displayed as they are set in real time.

```
/* ************************
 *  Arduino Nano Single LED with PWM Step
 *
 *  Control an LED connected to a Digital PWM (Pulse Width Modulation) Pin
 *  Use analogWrite to set LED values between 0 (LOW) and 255 (HIGH)
 *  Step through a list of predefined values
 *
 *  Digital Pins without PWM use digitalWrite with values of 0 (LOW) or 1 (HIGH)
 *
 *  Written by @alt_bier
 * ************************ */

// Define Pins
#define LED1 3

// Define Variables
int step = 0;

void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED1, OUTPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}

// Main Loop
void loop()
{
  // Set Delay Time [in ms]
  int DelayTime = 100;

  // Step Value Array
  int StepValue[10] = {0, 50, 100, 150, 200, 255, 200, 150, 100, 50};

  // Set the LED value using analogWrite using the step and StepValue variables
  // analogWrite is only usable on Analog pins or Digital PWM pins
  analogWrite(LED1, StepValue[step]);

  // Print current LED value to serial console for troubleshooting
  Serial.print(" LED_VALUE=");
  Serial.println(StepValue[step]);

  // Increment the Step Counter
  step = step + 1;
  if (step > 9) {
    // Reset step counter if it exceeds the num of StepValue array elements
    step = 0;
  }

  // Pause the loop keeping the LED lit with current values before resuming
  delay(DelayTime);

}
```

This Code Is Available Here: **https://github.com/gowenrw/BSidesDFW_2020_HHV/**

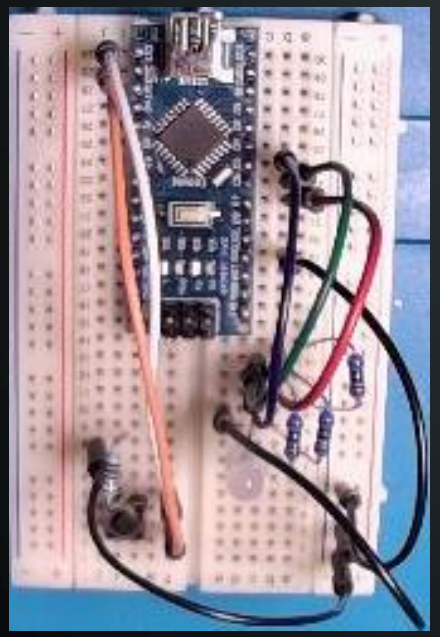
RGB LED AND TACTILE SWITCH ON ARDUINO NANO

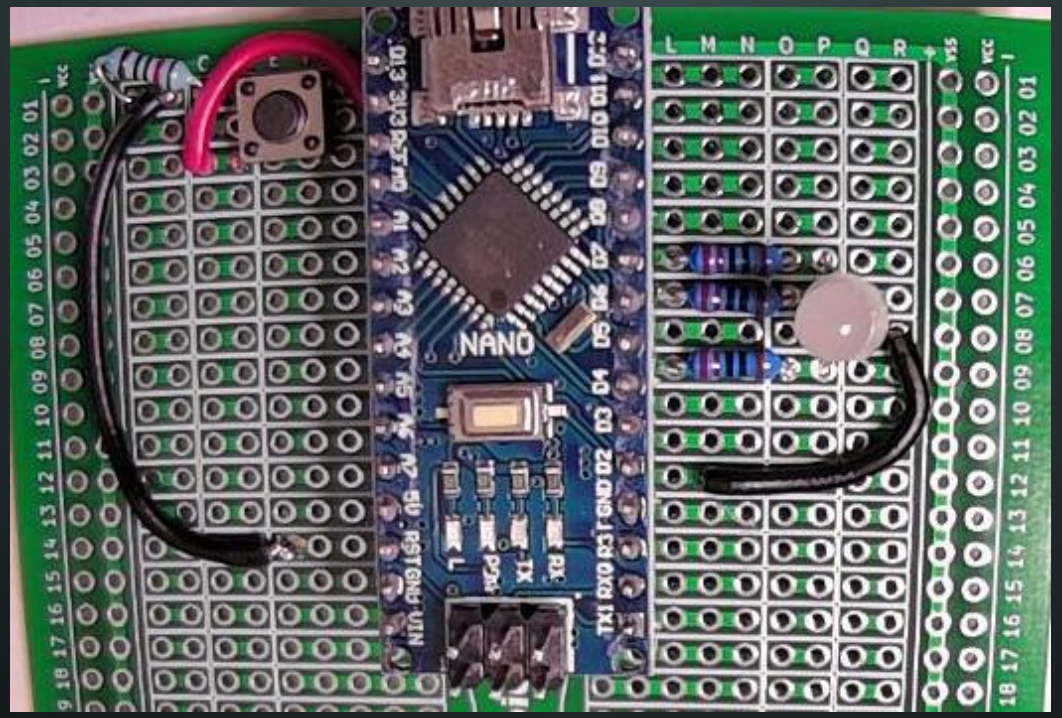
Schematic



RGB LED AND TACTILE SWITCH ON ARDUINO NANO

Physical Layout

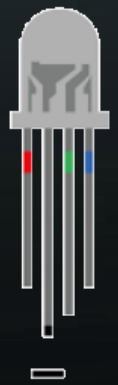


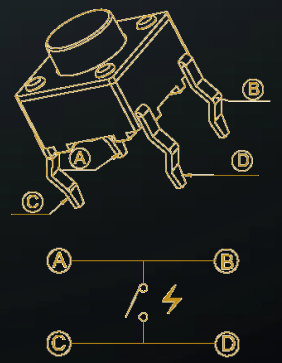


Wire up a circuit as shown in the schematic and physical layout.

Components:

- 3x Resistor 220 Ohm
- 1x Resistor 10K Ohm
- 1x RGB LED 5mm
- 1x Tactile Switch SPST 4pin





Strip Board Connection Details

- Arduino Nano – I1-15 and K1-15
- Resistor 220 Ohm – L9 and O9
- Resistor 220 Ohm – L7 and O7
- Resistor 220 Ohm – L6 and O6
- RGB LED
  - R Anode – P9
  - Cathode – Q8 (bend forward)
  - G Anode – P7
  - B Anode – P6
- Wire – R8 and M12
- Switch SPST 4 Pin
  - Pin A(1) – D1
  - Pin B(1) – G1
  - Pin C(2) – D3
  - Pin D(2) – N/C (bend off)
- Resistor 10K Ohm – C1 and A1
- Wire – H2 and C3
- Wire – B1 and E14

RGB LED AND TACTILE SWITCH ON ARDUINO NANO

RGB LED Cycle Code

Using an array of RGB color values we will cycle through the array of colors on the LED.

Open the **serial monitor** with a rate of **38400 baud** to see the LED values being set.

```
/* *************************
 *  Arduino Nano RGB LED Cycle
 *
 *  Cycle between predefined RGB color values
 *
 *  Written by @alt_bier
 * ************************* */

// Define Pins
#define LED2_RED 4
#define LED2_GREEN 6
#define LED2_BLUE 7

// Define Variables
int c = 0;

void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED2_RED, OUTPUT);
  pinMode(LED2_GREEN, OUTPUT);
  pinMode(LED2_BLUE, OUTPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}
```

```
// Main Loop
void loop()
{
  // Set Delay for each transition of main loop [in ms]
  int DelayTime = 100;

  // Color Settings in R,G,B Binary Values
  // 0      1    2      3      4    5    6
  // White Red Yellow Green Cyan Blue Magenta
  int Color[7][3] = {
    {1, 1, 1}, {1, 0, 0}, {1, 1, 0}, {0, 1, 0}, {0, 1, 1}, {0, 0, 1}, {1, 0, 1}
  };

  // LED Pin Array
  int LED[3] = {
    LED2_RED, LED2_GREEN, LED2_BLUE
  };

  // Print to the serial console the current color selection for troubleshooting
  Serial.print(" C=");
  Serial.print(c);

  for (int i = 0; i < 3; i++) {          // Loop Through Each Color (MAX = 3 RGB)
    // Send the digital value (1 or 0) to the LED PIN
    digitalWrite(LED[i], Color[c][i]);
    // Print to the serial console the pin and value sent for troubleshooting
    Serial.print(" PIN=");
    Serial.print(LED[i]);
    Serial.print(" VALUE=");
    Serial.print(Color[c][i]);
  }

  // Increment the Current Color
  c = c + 1;
  if (c > 6) {
    c = 0;
  }

  // Print a new line to the serial console
  Serial.println();

  // Pause the loop keeping the LED lit with current values before resuming
  delay(DelayTime);

}
```

This Code Is Available Here: **https://github.com/gowenrw/BSidesDFW_2020_HHV/**

RGB LED AND TACTILE SWITCH ON ARDUINO NANO

RGB LED with Tactile Switch Code

Using an array of RGB color values and a tactile switch we will cycle through the array of colors on the LED one at a time when the switch is pushed.

Open the **serial monitor** with a rate of **38400 baud** to see the LED values being set.

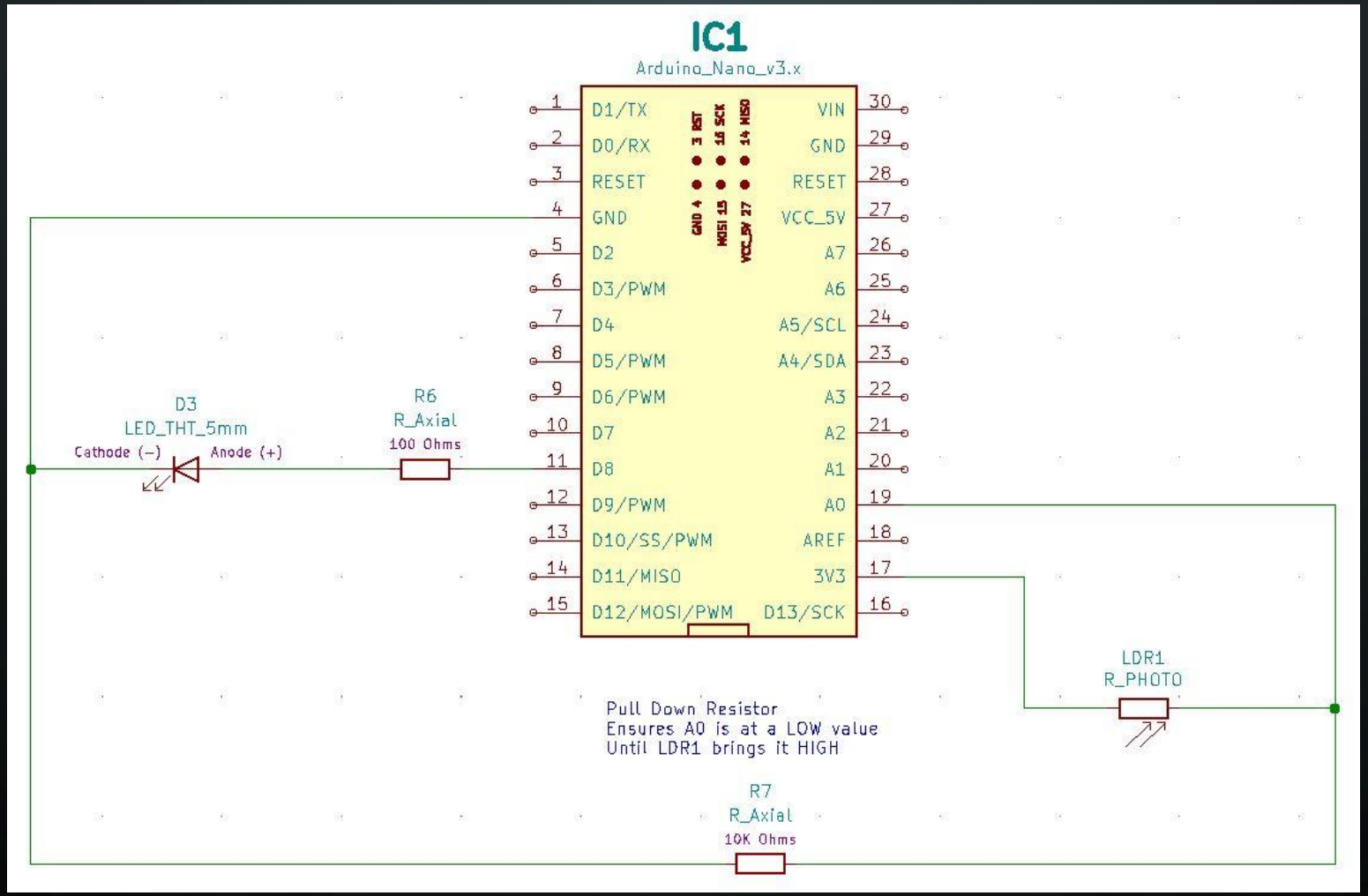
```
/* ************************
 *  Arduino Nano RGB LED with Tactile Switch
 *
 *  Control a Red Green Blue Common Cathode LED using a Tactile Switch
 *  Cycle between predefined RGB color settings by pressing the Tactile Switch
 *
 *  Written by @alt_bier
 * ************************ */

// Define Pins
#define LED2_RED 4
#define LED2_GREEN 6
#define LED2_BLUE 7
#define SW1 13

// Define Variables
int c = 0;
int SW1State = 0;

void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED2_RED, OUTPUT);
  pinMode(LED2_GREEN, OUTPUT);
  pinMode(LED2_BLUE, OUTPUT);

  // Initialize Tactile Switch Pin As Input:
  pinMode(SW1, INPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}
```

```
// Main Loop
void loop()
{
  // Set Delay for each transition of main loop [in ms]
  int DelayTime = 100;

  // Color Settings in R,G,B Binary Values
  // 0     1   2     3     4    5    6
  // White Red Yellow Green Cyan Blue Magenta
  int Color[7][3] = {
    {1, 1, 1}, {1, 0, 0}, {1, 1, 0}, {0, 1, 0}, {0, 1, 1}, {0, 0, 1}, {1, 0, 1}
  };

  // LED Pin Array
  int LED[3] = {
    LED2_RED, LED2_GREEN, LED2_BLUE
  };

  // Read the state of SW1
  SW1State = digitalRead(SW1);

  // Print to the serial console the state of SW1 for troubleshooting
  Serial.print(" SW1=");
  Serial.print(SW1State);

  // Check if SW1 is pressed.
  // If it is, the SW1State is HIGH:
  if (SW1State == HIGH) {
    // Increment the Current Color
    c = c + 1;
    if (c > 6) {
      c = 0;
    }
  }

  // Print to the serial console the current color selection for troubleshooting
  Serial.print(" C=");
  Serial.print(c);

  for (int i = 0; i < 3; i++) {         // Loop Through Each Color (MAX = 3 RGB)
    // Send the digital value (1 or 0) to the LED PIN
    digitalWrite(LED[i], Color[c][i]);
    // Print to the serial console the pin and value sent for troubleshooting
    Serial.print(" PIN=");
    Serial.print(LED[i]);
    Serial.print(" VALUE=");
    Serial.print(Color[c][i]);
  }

  // Print a new line to the serial console
  Serial.println();

  // Pause the loop keeping the LED lit with current values before resuming
  delay(DelayTime);

}
```

This Code Is Available Here: **https://github.com/gowenrw/BSidesDFW_2020_HHV/**

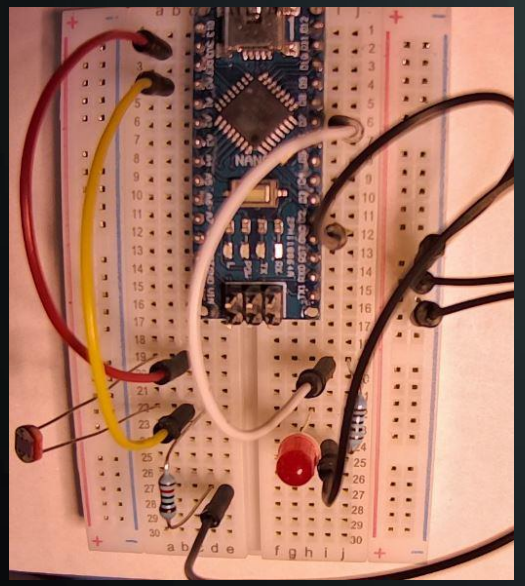
LED AND PHOTORESISTOR CONNECTED TO ARDUINO NANO

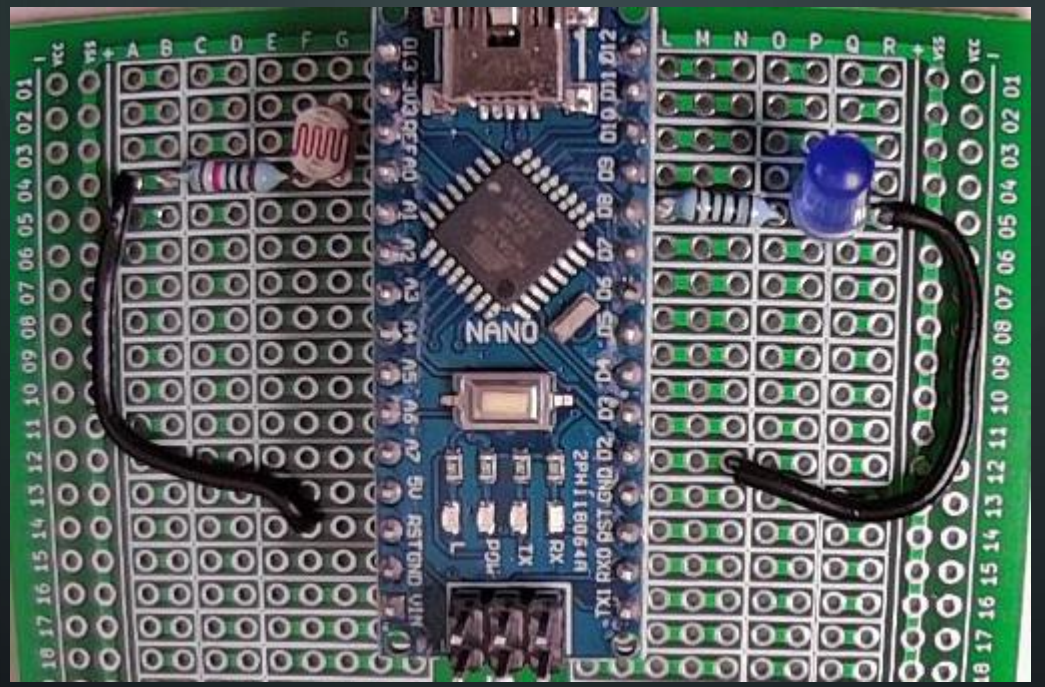
Schematic



LED AND PHOTORESISTOR CONNECTED TO ARDUINO NANO

Physical Layout





Strip Board Connection Details

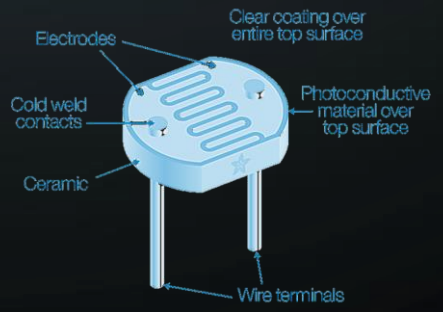
- Arduino Nano – **I1-15** and **K1-15**
- Resistor 100 Ohm – **L5** and **O5**
- LED – **P5** (Anode) and **Q5** (Cathode)
- Wire – **R5** and **N12**
- Photoresistor – **G2** and **G4**
- Resistor 10K Ohm – **B4** and **F4**
- Wire – **A4** and **F14**

Note how the pull-down resistor is connected to ground on a different Arduino pin than the LED even though the schematic shows them using the same pin.
This is possible because the ground pins on the Arduino are interconnected.

Wire up a circuit as shown in the schematic and physical layout.

Components:

- 1x Resistor 100 Ohm
- 1x Resistor 10K Ohm
- 1x LED 5mm
- 1x Photoresistor



LED AND PHOTORESISTOR CONNECTED TO ARDUINO NANO

Photoresistor LED Control Code

Use a photoresistor to control an LED turning it on when it is dark and off when it is light.

Cover the photoresistor to produce dark values which will turn on the LED

Open the **serial monitor** with a rate of **38400 baud** to see the photoresistor values being read and the LED values being set.

```
/* *************************
 *  Arduino Nano Photoresistor LED Control
 *
 *  Use the input from a Photoresistor to control an LED
 *  The Photoresistor will provide analog values indicating dark or light
 *  Set an LED to on (HIGH) or off (LOW) based on dark or light values
 *
 *  Written by @alt_bier
 * ************************* */

// Define Pins
#define LED3 8
#define PR1 A0

// Define Variables
int pval;

void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED3, OUTPUT);

  // Initialize Photoresistor Pin As Input:
  pinMode(PR1, INPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}
```

```
// Main Loop
void loop()
{
  // Set Delay Time [in ms]
  int DelayTime = 100;

  // Set the Photoresistor Threshold
  int pthr = 30;

  // Record the Photoresistor Value
  pval = analogRead(PR1);
  // Print current Photoresistor Value to serial console for troubleshooting
  Serial.print(" PVAL=");
  Serial.print(pval);

  // Set the LED value to HIGH (on) or LOW (off)
  // based on Photoresistor Value and Threshold
  if (pval > pthr) {
    // Turn LED OFF
    digitalWrite(LED3, LOW);
    // Print current LED state to serial console for troubleshooting
    Serial.print(" LED OFF");
  } else {
    // Turn LED ON
    digitalWrite(LED3, HIGH);
    // Print current LED state to serial console for troubleshooting
    Serial.print(" LED ON");
  }

  // Print a new line to the serial console
  Serial.println();

  // Pause the loop keeping the LED lit with current values before resuming
  delay(DelayTime);

}
```

This Code Is Available Here: https://github.com/gowenrw/BSidesDFW_2020_HHV/

LED AND PHOTORESISTOR CONNECTED TO ARDUINO NANO

Photoresistor Luminosity Code

Use a photoresistor to display the luminosity of its environment measured in lumens.

Cover the photoresistor to produce dark values which will show the decrease in luminosity.

Open the **serial monitor** with a rate of **38400 baud** to see the photoresistor values in raw analog form and converted lumens.

```
/* ************************
 *  Arduino Nano Photoresistor Luminosity
 *
 *  Use the input from a Photoresistor to measure
 *  the amount of light a.k.a. luminosity (Lux)
 *  in lumens it is receiving in real time
 *
 *  Written by @alt_bier
 * ************************ */

// Define Pins
#define LED3 8
#define PR1 A0

// Define Constants
#define VIN 3.3  // V power voltage
#define R7 10000 // Pull down resistor ohm value

// Define Variables
int pval;
float lux;
// Variable converts raw analog value to physical light value (lumen)
float sensorRawToPhys(int raw){
  // Conversion rule
  float Vout = float(raw) * (VIN / float(1023)); // Conversion analog to voltage
  float RLDR = (R7 * (VIN - Vout))/Vout;         // Conversion voltage to resistance
  float phys=500/(RLDR/1000);                    // Conversion resistance to lumen
  return phys;
}
```

```
void setup()
{
  // Initialize LED Pins As Output
  pinMode(LED3, OUTPUT);

  // Initialize Photoresistor Pin As Input:
  pinMode(PR1, INPUT);

  // Set up serial output to console (baud)
  Serial.begin(38400);
}

// Main Loop
void loop()
{
  // Set Delay Time [in ms]
  int DelayTime = 100;

  // Set the Photoresistor Threshold
  int pthr = 30;

  // Record the Photoresistor Value
  pval = analogRead(PR1);
  // Print current Photoresistor RAW Value to serial console for troubleshooting
  Serial.print(" PVAL=");
  Serial.print(pval);
  Serial.print(" raw    ");
  // Get the lumens based on pval
  lux = sensorRawToPhys(pval);
  // Print the Light Level
  Serial.print(" Light=");
  Serial.print(lux);
  Serial.print(" lumen    ");

  // Set the LED value to HIGH (on) or LOW (off)
  // based on Photoresistor Value and Threshold
  if (pval > pthr) {
    // Turn LED OFF
    digitalWrite(LED3, LOW);
    // Print current LED state to serial console for troubleshooting
    Serial.print(" LED OFF");
  } else {
    // Turn LED ON
    digitalWrite(LED3, HIGH);
    // Print current LED state to serial console for troubleshooting
    Serial.print(" LED ON");
  }

  // Print a new line to the serial console
  Serial.println();

  // Pause the loop keeping the LED lit with current values before resuming
  delay(DelayTime);

}
```

This Code Is Available Here: https://github.com/gowenrw/BSidesDFW_2020_HHV/

THANK YOU

I hope you enjoyed this presentation and learned something from it.

-- @alt_bier

This Slide Deck – https://altbier.us/arduino/

Code – https://github.com/gowenrw/BSidesDFW_2020_HHV/