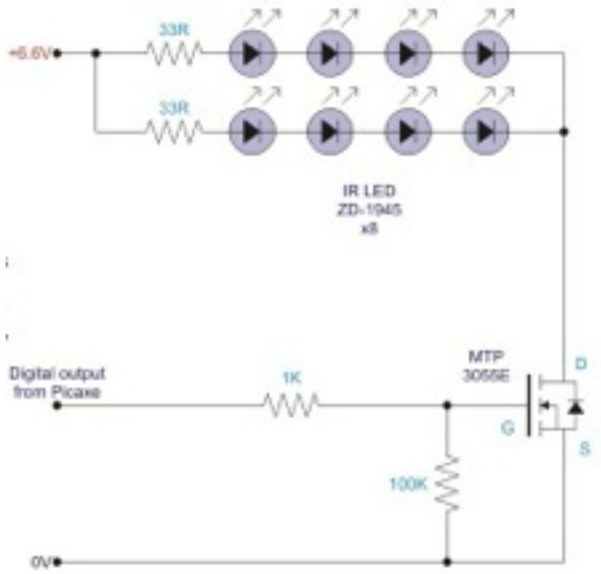
Fashioning Circuits
Oct 31

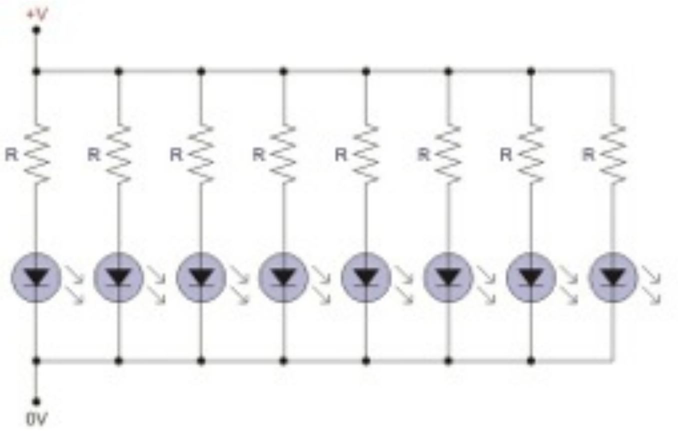
serial vs. parallel LEDs
tri-color LEDs
fade and forloopiteration

Serial vs. Parallel Wiring



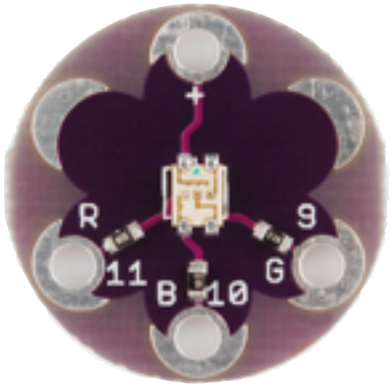
- Serial
    - wired positive to negative to positive to negative, in a series
    - when one goes out, no current flows
    - voltage drop calculated across all of the series - all LEDs share the voltage; LEDs will likely be more dim
    - Advantages - drains battery less slowly; can connect different types of components

Serial vs. Parallel Wiring



- Parallel
  - wired all positives together, and all negatives together
  - when one goes out, current still flows
  - LEDs receive full voltage; no dimming
  - Advantages - drains battery more quickly; have to take care when connecting components - have to be the same to be wired in parallel

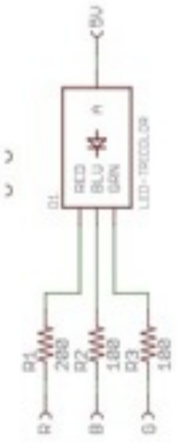
Tri-color LED



- Three colors - Red, Blue, Green
  - “color changing” = 1 LED with 3 petals.

- 1 connection from the battery and three to ground

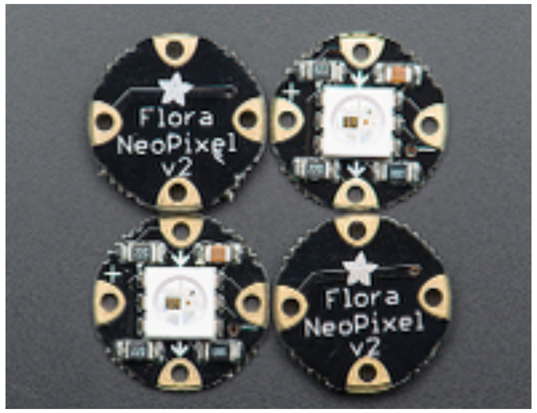
Tri Color LEDs



- Common anode LED
- one anode but there are three cathodes and three resistors
    - anode / cathode
        - current flows from anode to cathode
            - Note the notation for diode –
        - (may not get into this) uses “current sinking”
            - as opposed to common cathode LEDs which use current sourcing
            - <my brain hurts>
    -

Tri-Color LED

Sparkfun Pixelboard

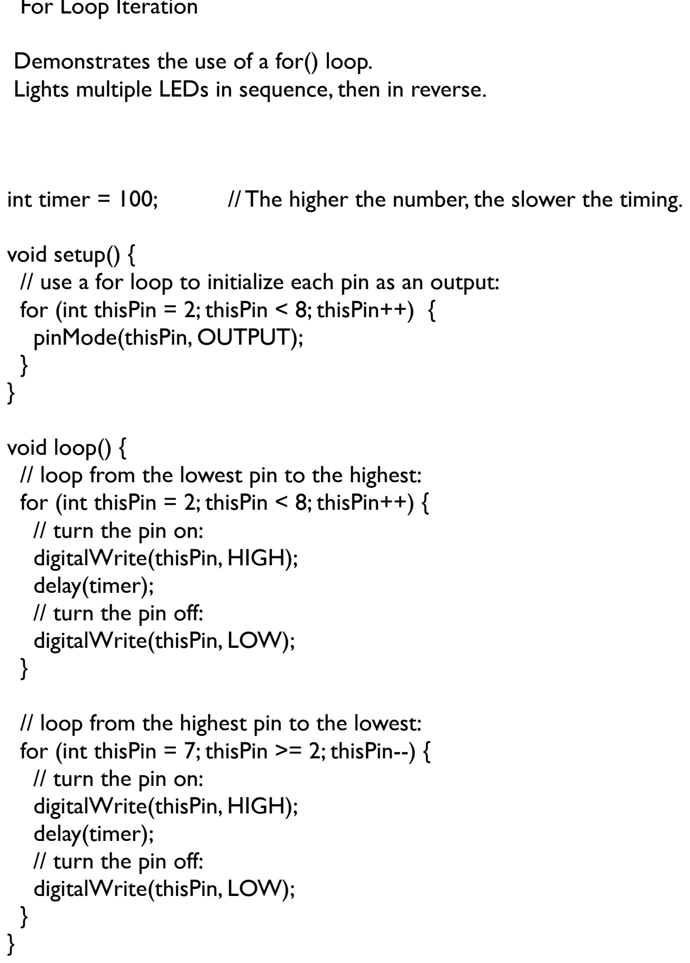


Adafruit Smart NeoPixel

- These RBG LEDs have a chip on them that means that you can connect multiple together with one connection from the battery, one connection to ground, and one connection for data.
- They are addressable when linked – in other words, you can control each individually by addressing the number in the chain

```
/*
 Sketch "FadeExternalLED"


int ledPin = 5;        // the pin that the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup()  {
  // declaring led pin to be an output is optional for analogWrite():
  pinMode(ledPin, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()  {
  // set the brightness of led pin:
  analogWrite(ledPin, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

- Upload
- Go over the code to see what is causing the fading
- Pulse Width Modulation
  - digital pin – only on / off
  - pulse width modulation emulates analog by varying the length of time the signal stays on (the pulse width) – simulates voltages between 0 and 5
  - scale 0 – 255



```
  For Loop Iteration

 Demonstrates the use of a for() loop.
 Lights multiple LEDs in sequence, then in reverse.



int timer = 100;           // The higher the number, the slower the timing.

void setup() {
  // use a for loop to initialize each pin as an output:
  for (int thisPin = 2; thisPin < 8; thisPin++)  {
    pinMode(thisPin, OUTPUT);
  }
}

void loop() {
  // loop from the lowest pin to the highest:
  for (int thisPin = 2; thisPin < 8; thisPin++) {
    // turn the pin on:
    digitalWrite(thisPin, HIGH);
    delay(timer);
    // turn the pin off:
    digitalWrite(thisPin, LOW);
  }

  // loop from the highest pin to the lowest:
  for (int thisPin = 7; thisPin >= 2; thisPin--) {
    // turn the pin on:
    digitalWrite(thisPin, HIGH);
    delay(timer);
    // turn the pin off:
    digitalWrite(thisPin, LOW);
  }
}
```

- examples, control, foorloopiteration
- Uses math
- need LEDS connected in a row – let’s change to work with RGB

Exercises

- Connect RGB LED to a button or switch
- Play with Fade with RGB