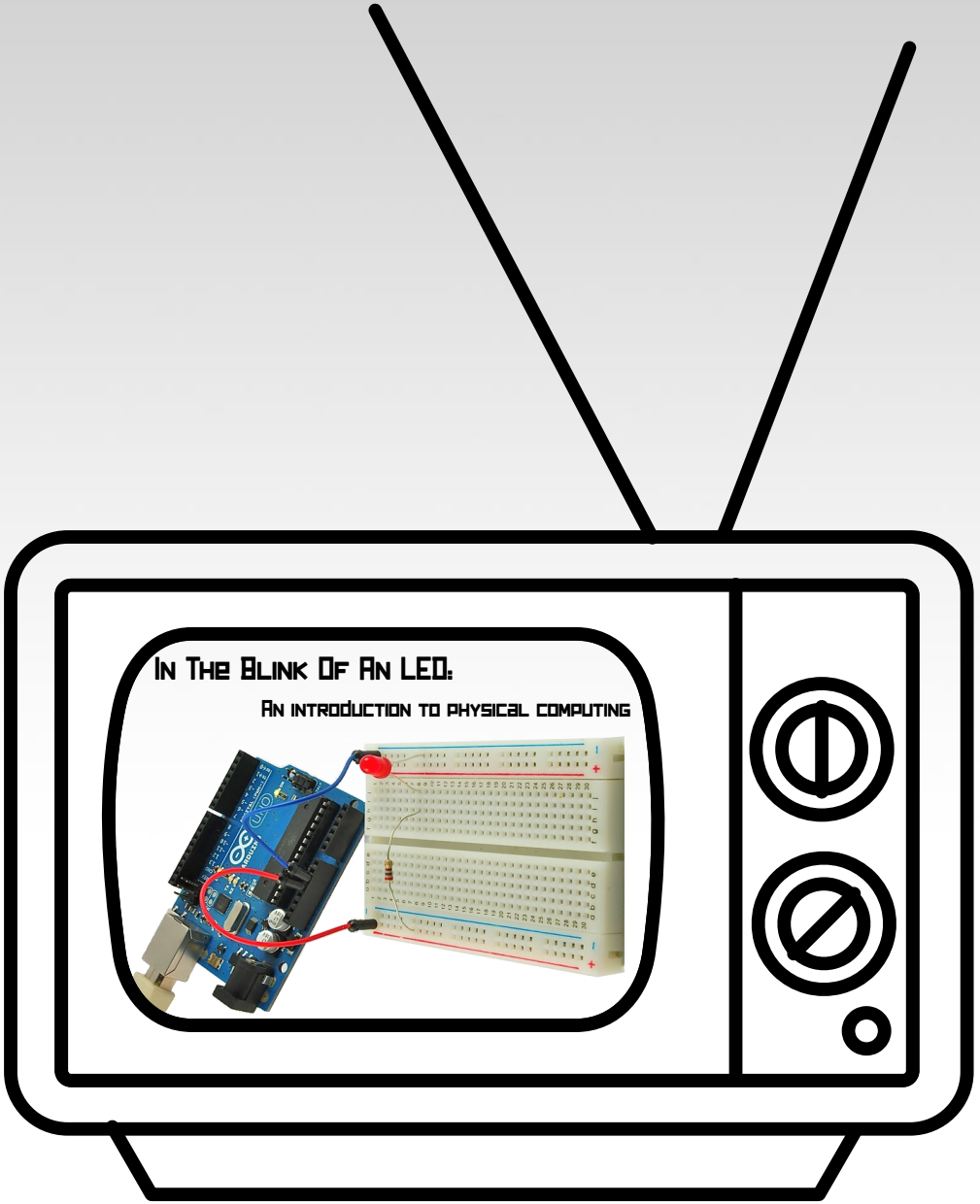
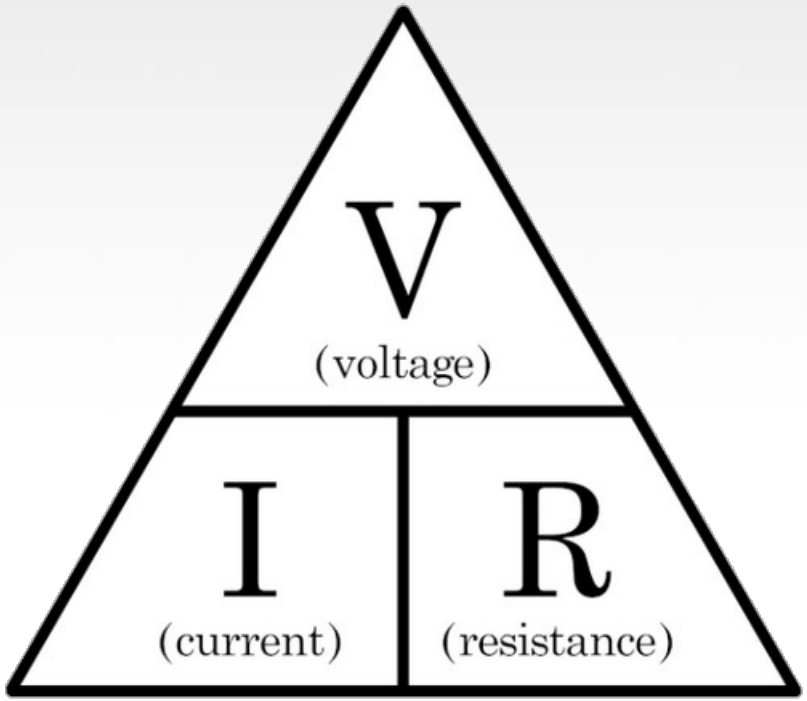
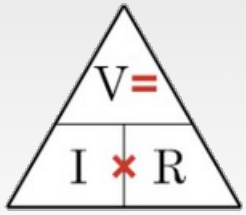
Previously On...

- Voltage, Current and Resistance

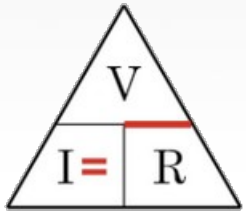


Ohm's Law

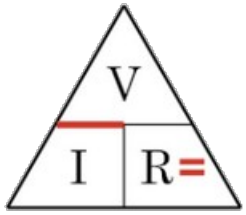




$$V = I \times R$$



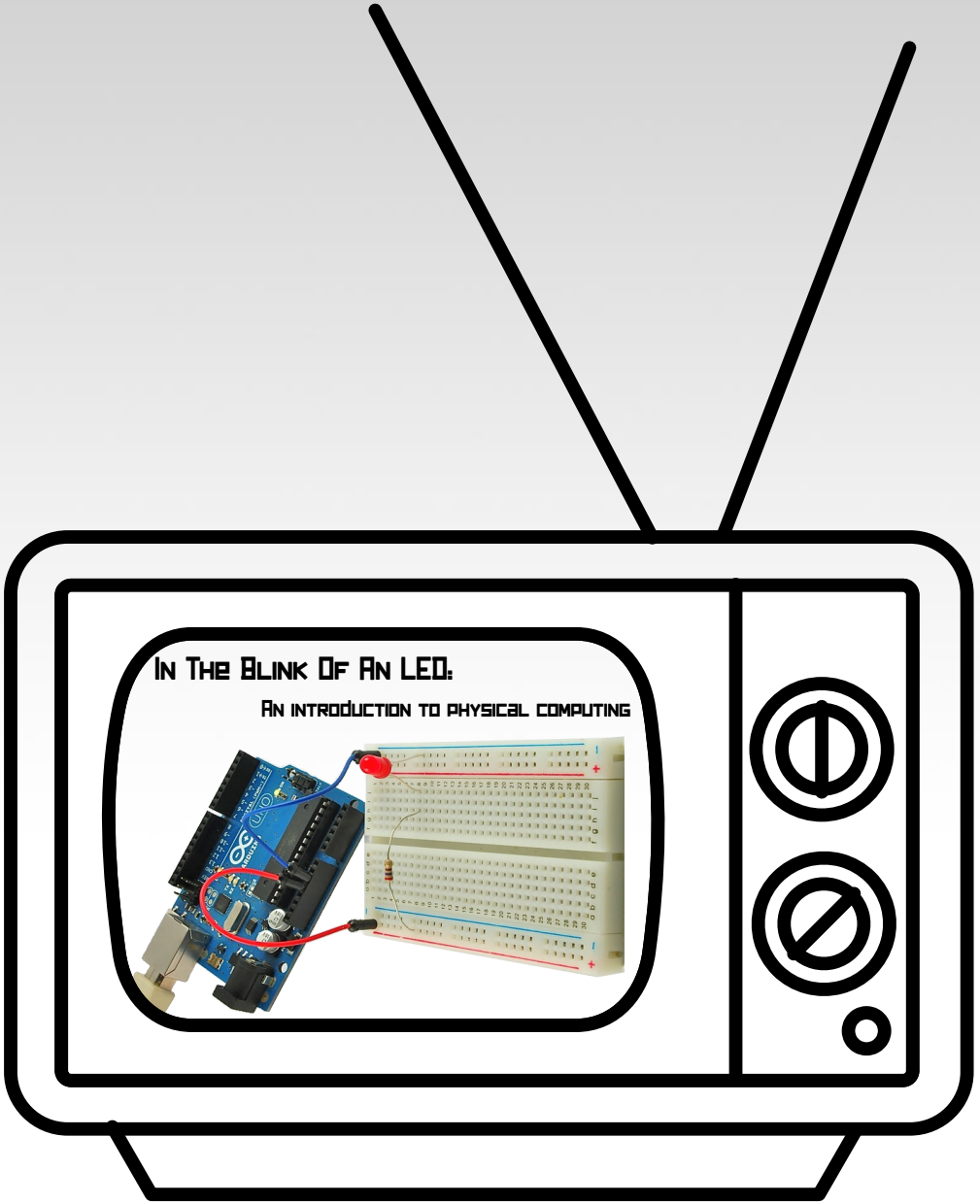
$$I = V \div R$$



$$R = V \div I$$

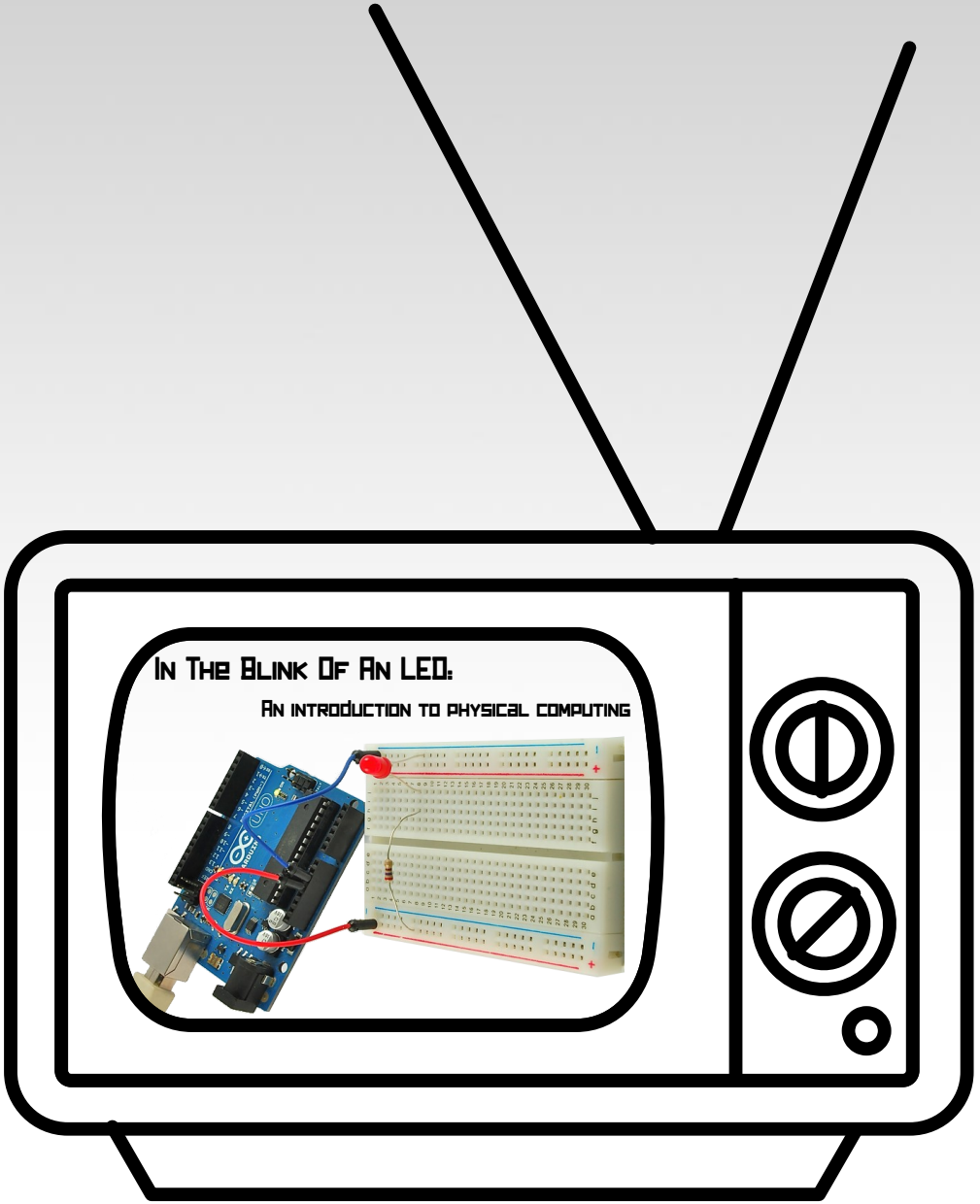
Previously On...

- Voltage, Current and Resistance



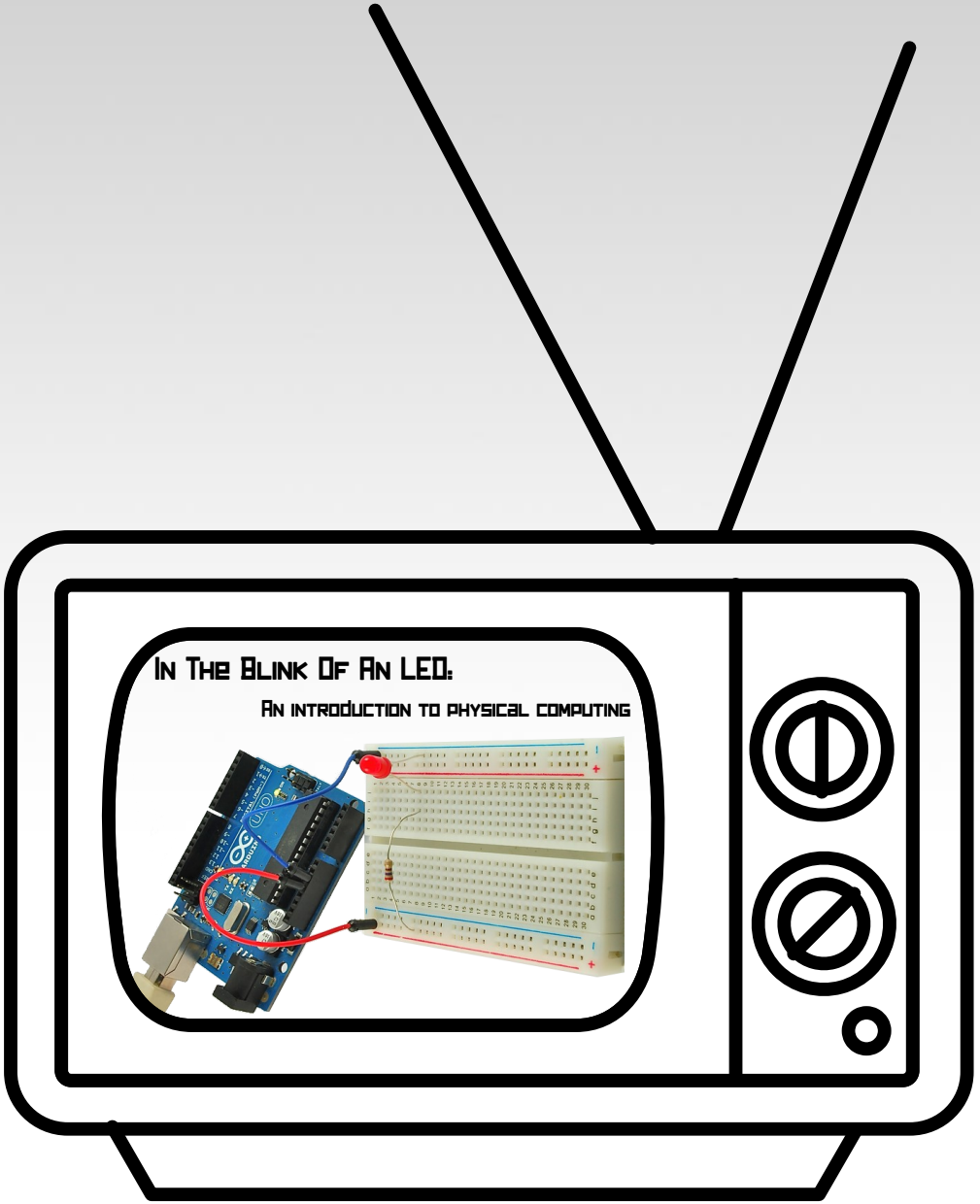
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs



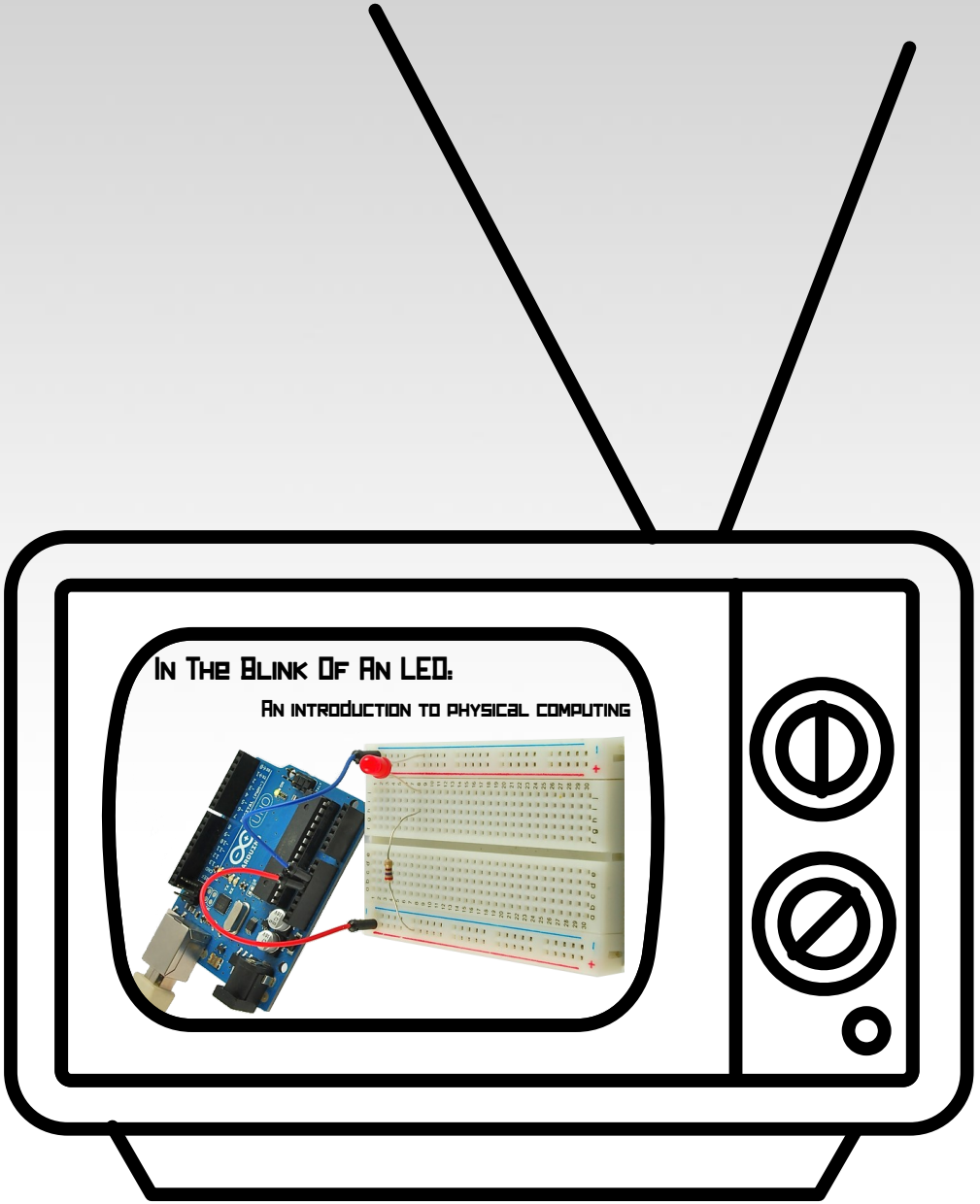
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog



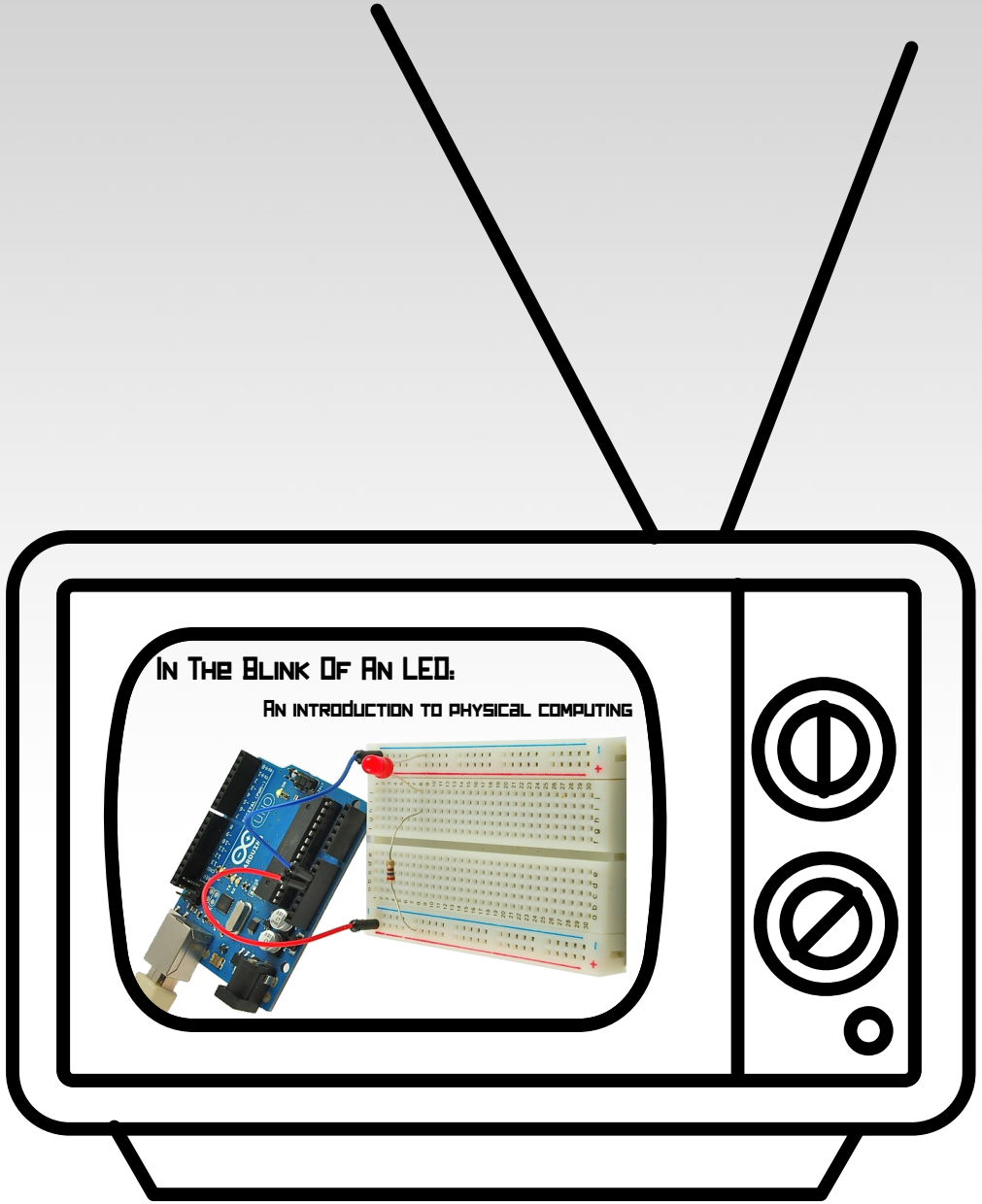
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands



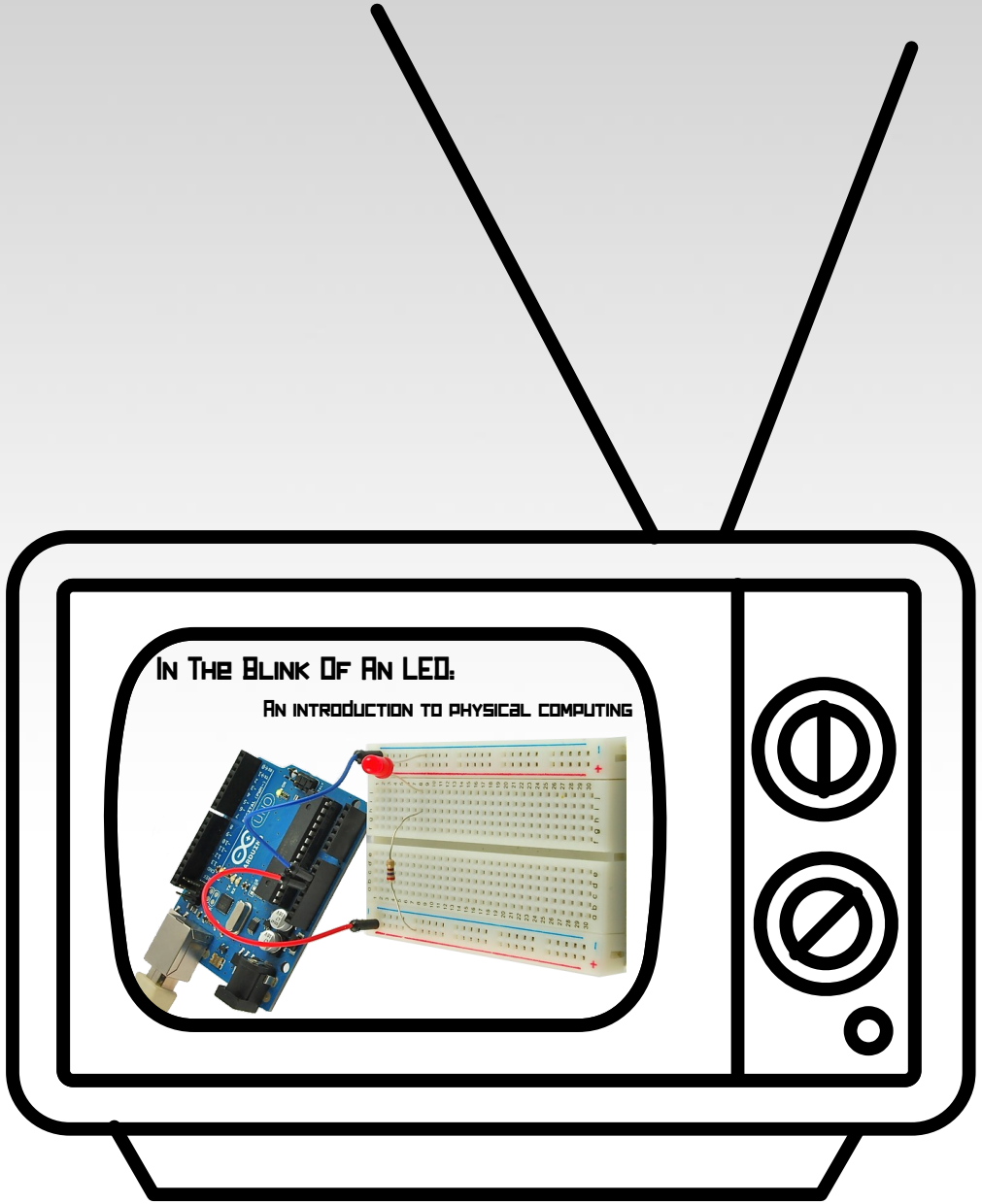
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands
  - `pinMode(pin, INPUT/OUTPUT);`
  - `digitalWrite(pin, HIGH/LOW);`
  - `delay(ms);`
  - `analogWrite(pin, 0-255);`



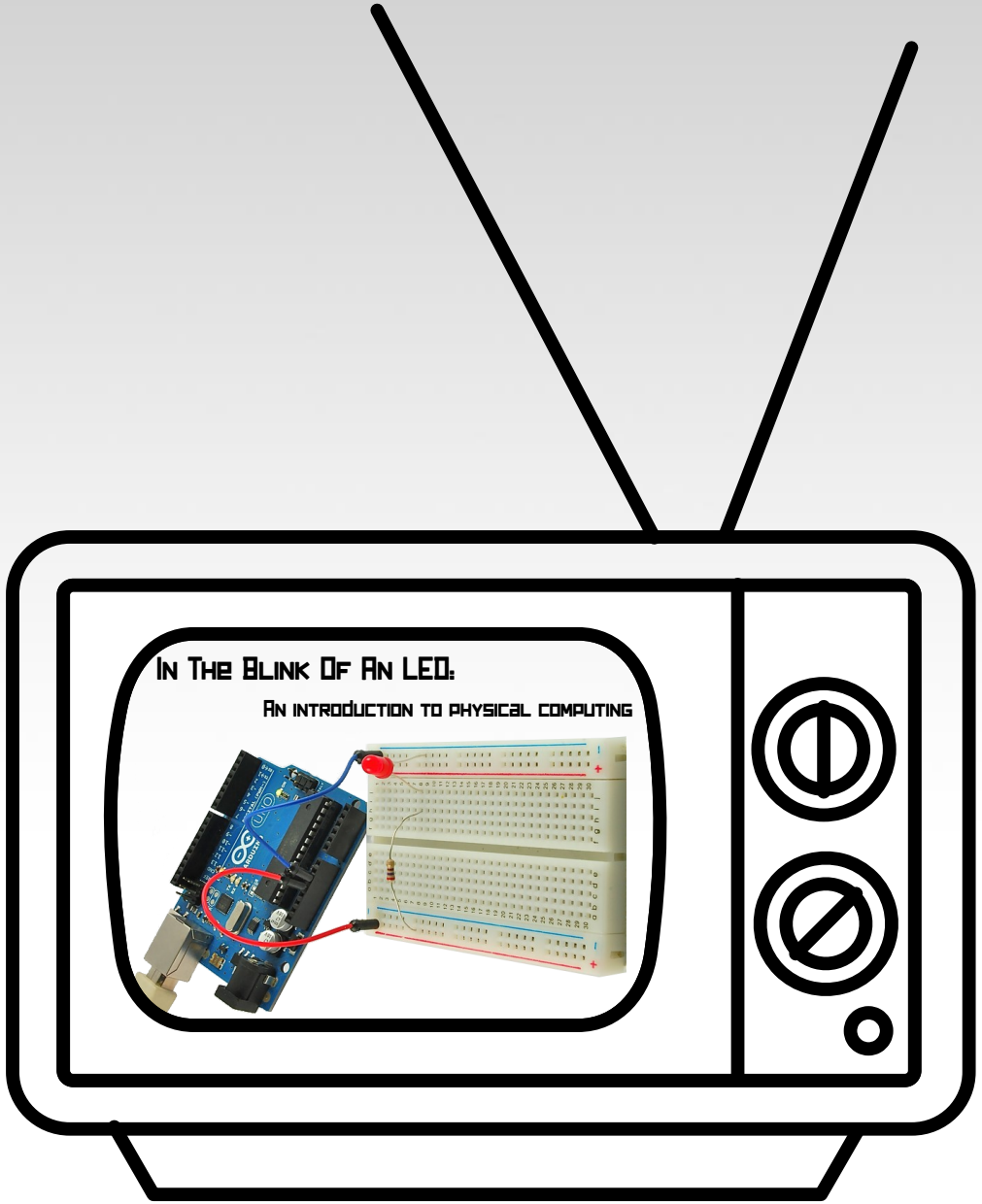
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands
  - `pinMode(pin, INPUT/OUTPUT);`
  - `digitalWrite(pin, HIGH/LOW);`
  - `delay(ms);`
  - `analogWrite(pin, 0-255);`
- Functions



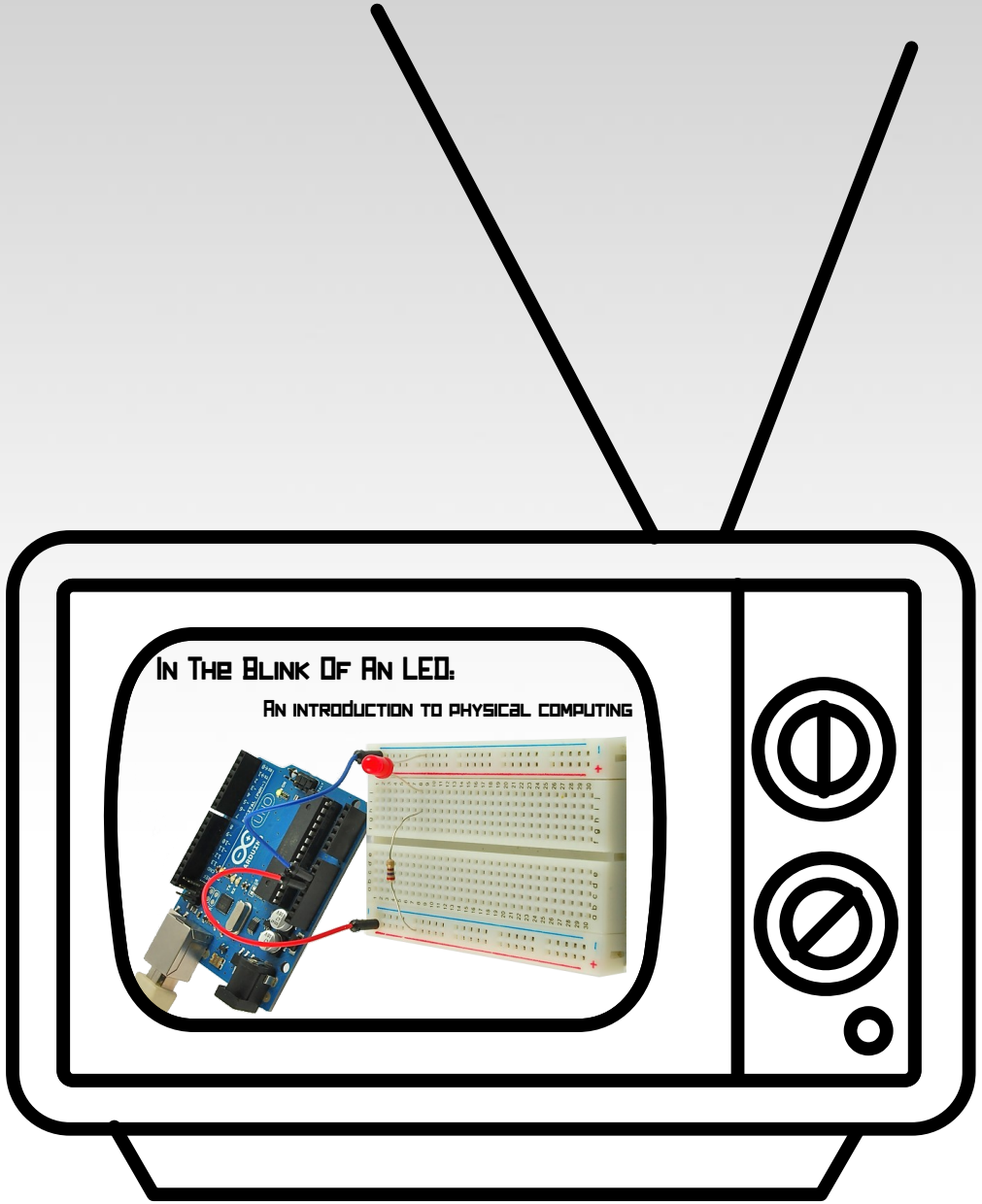
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands
  - `pinMode(pin, INPUT/OUTPUT);`
  - `digitalWrite(pin, HIGH/LOW);`
  - `delay(ms);`
  - `analogWrite(pin, 0-255);`
- Functions
  - `void setup() {}`
  - `void loop() {}`



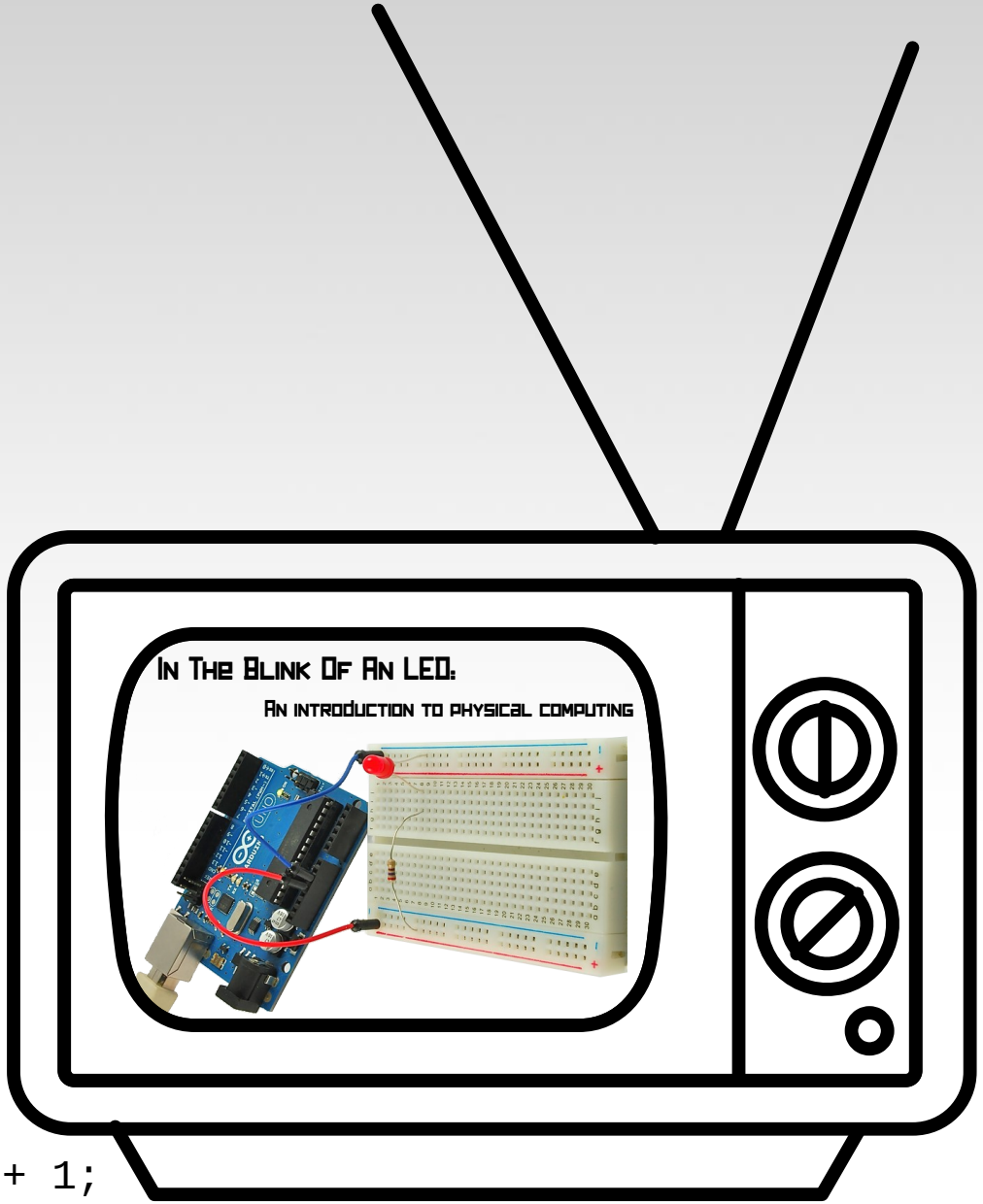
Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands
  - `pinMode(pin, INPUT/OUTPUT);`
  - `digitalWrite(pin, HIGH/LOW);`
  - `delay(ms);`
  - `analogWrite(pin, 0-255);`
- Functions
  - `void setup() {}`
  - `void loop() {}`
- Comments
  - `// Keep things clear`

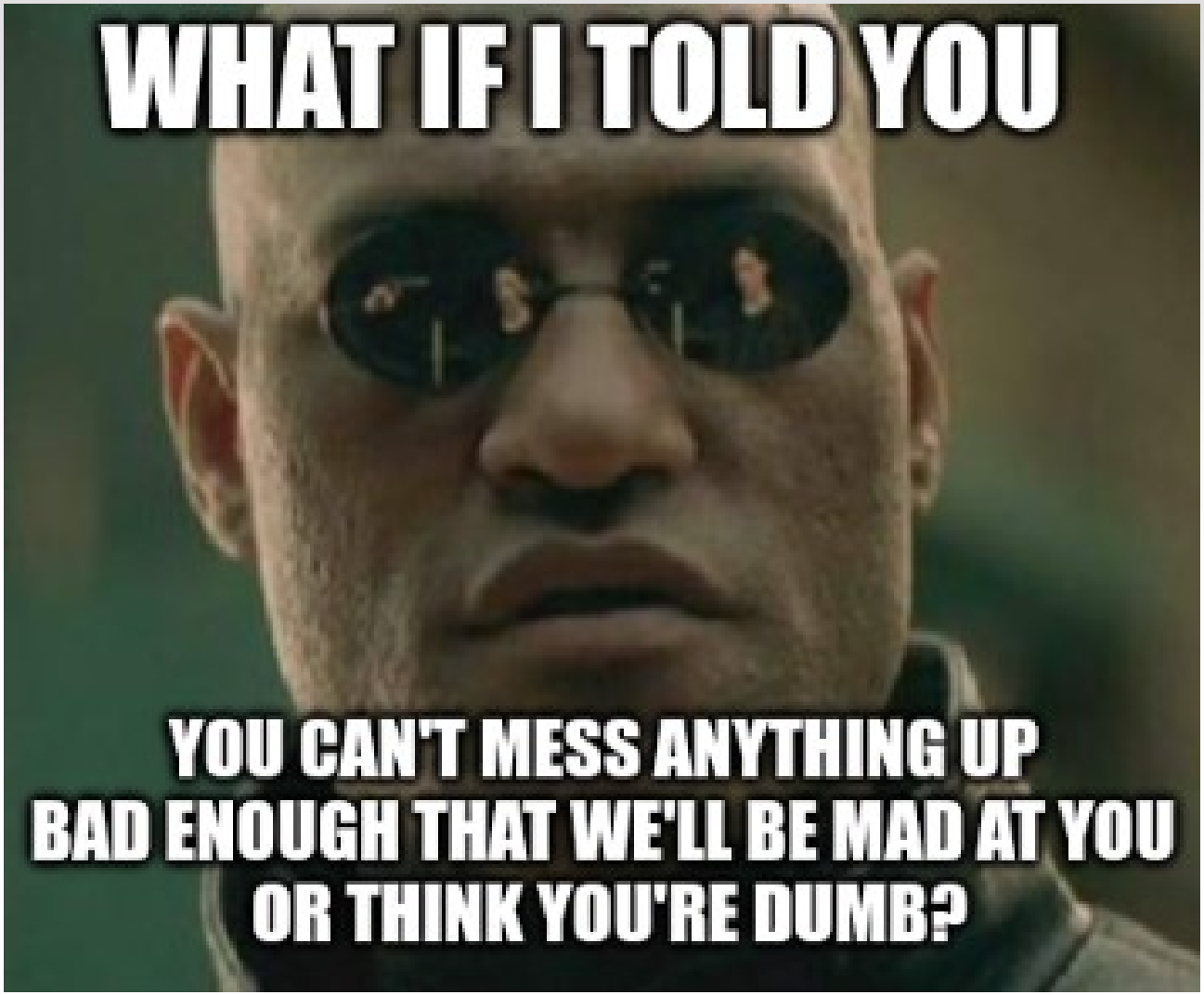


Previously On...

- Voltage, Current and Resistance
- Inputs and Outputs
- Digital Vs. Analog
- Commands
  - `pinMode(pin, INPUT/OUTPUT);`
  - `digitalWrite(pin, HIGH/LOW);`
  - `delay(ms);`
  - `analogWrite(pin, 0-255);`
- Functions
  - `void setup() {}`
  - `void loop() {}`
- Comments
  - `// Keep things clear`
- Variables
  - `int led = 9;`
  - `int brightness = brightness + 1;`



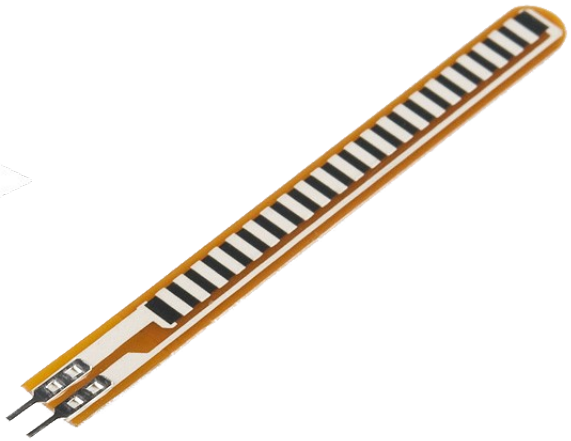
The Most Important Lesson



Inputs!

Input is any signal entering an electrical system

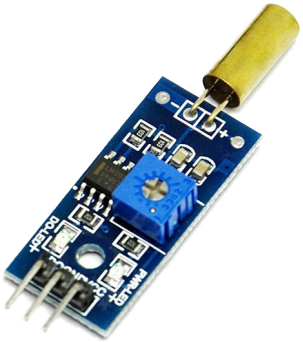
- Both digital and analog sensors are forms of input
- Input can also take many other forms: a keyboard, a mouse, infrared sensors, biometric sensors, touch sensors, cameras or just plain voltage from a circuit
- This is how you can make a computer react to the world around it



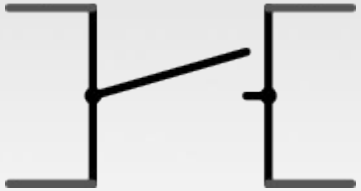
Digital Input

Like digital output, can only sense if a signal is high or low

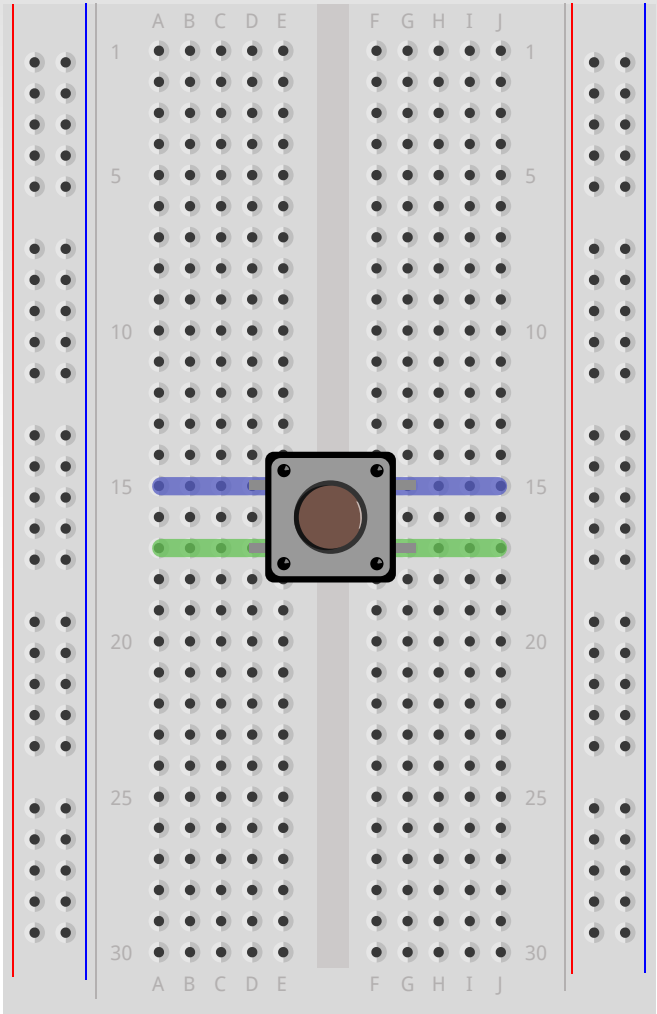
- For Arduino
  - High is voltage above 3 V
  - Low is voltage below 1.5 V
- General category is “switches”
- Includes push buttons (momentary switches), toggle switches, knife switches (maintained switches), capacitive/touch switches, and many other sensors

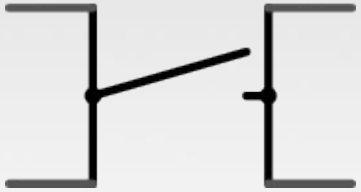


Component Spotlight: The Push Button / Momentary Switch



Push Button and Breadboard Connections

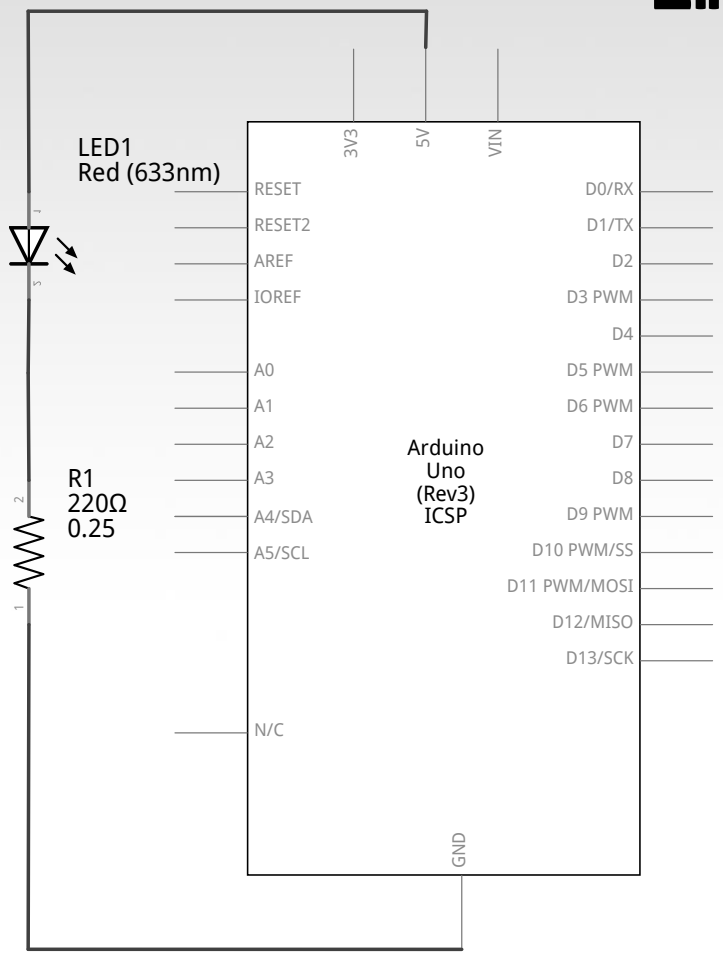




Our buttons have to bridge the gap on our breadboards

The top pins are connected to each other and so are the bottom ones

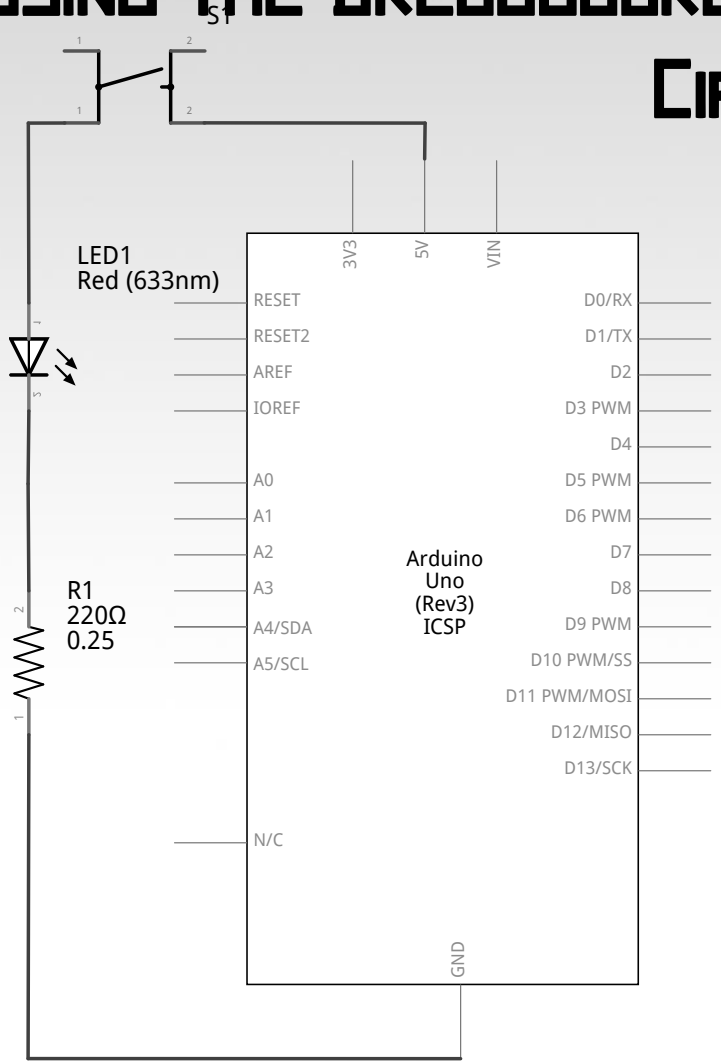
Using the Breadboard to Built a Simple Input Circuit



Remember our original circuit?

fritzing

Using the Breadboard to Built a Simple Input Circuit

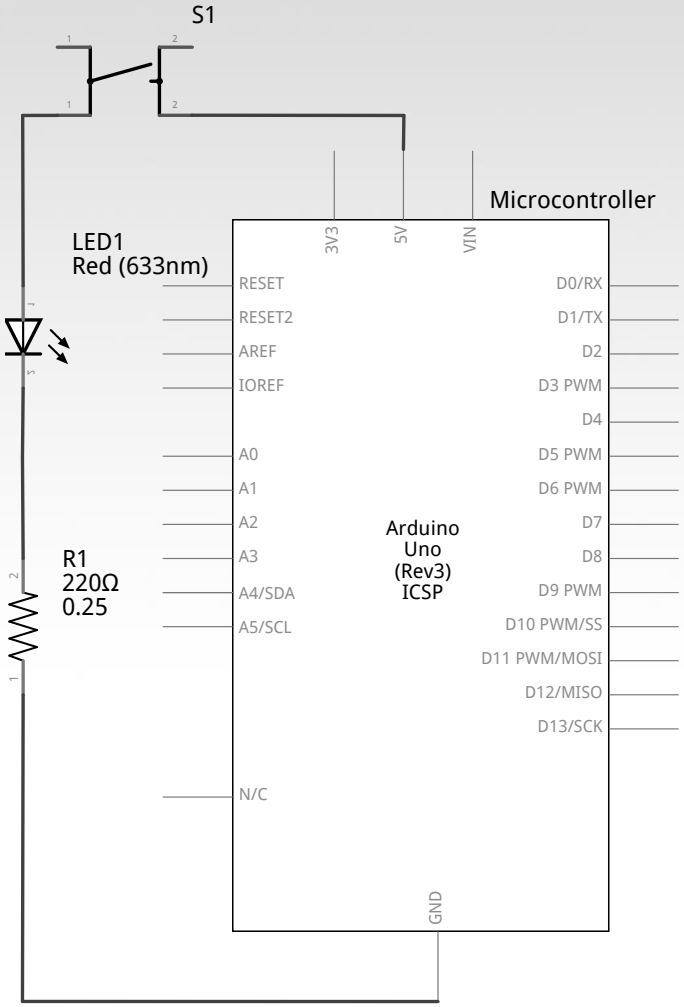


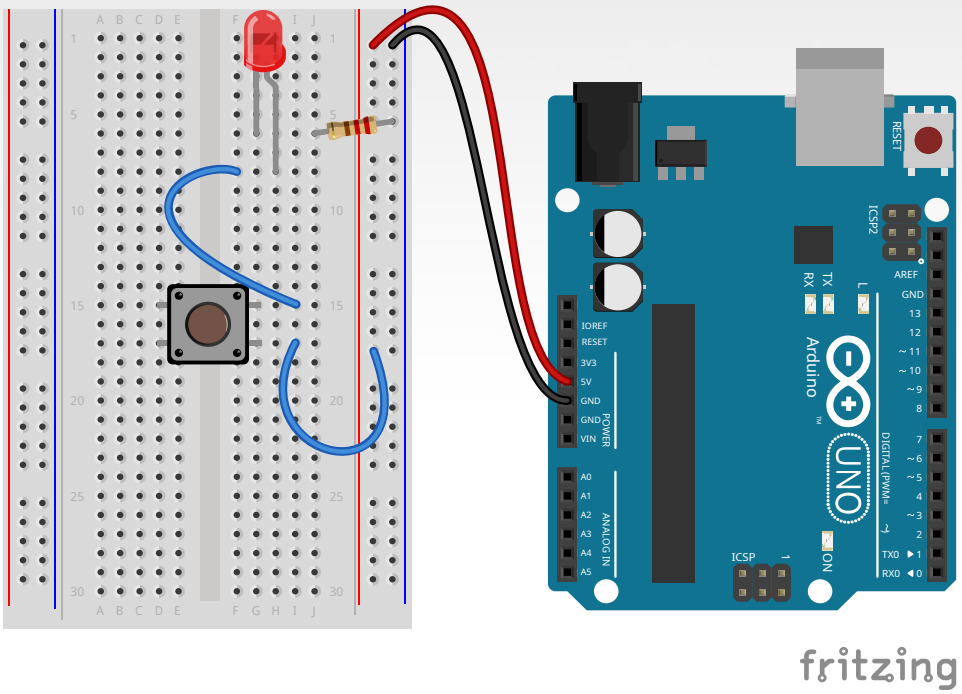
Remember our original circuit?

We're going to wire that up again, but this time with a switch

fritzing

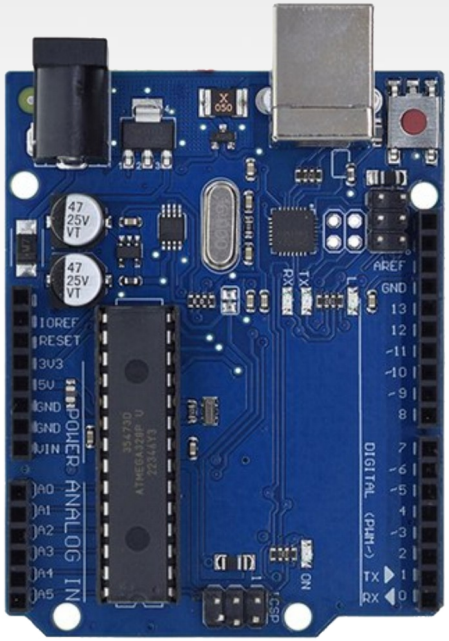
Using the Breadboard to Built a Simple Input Circuit



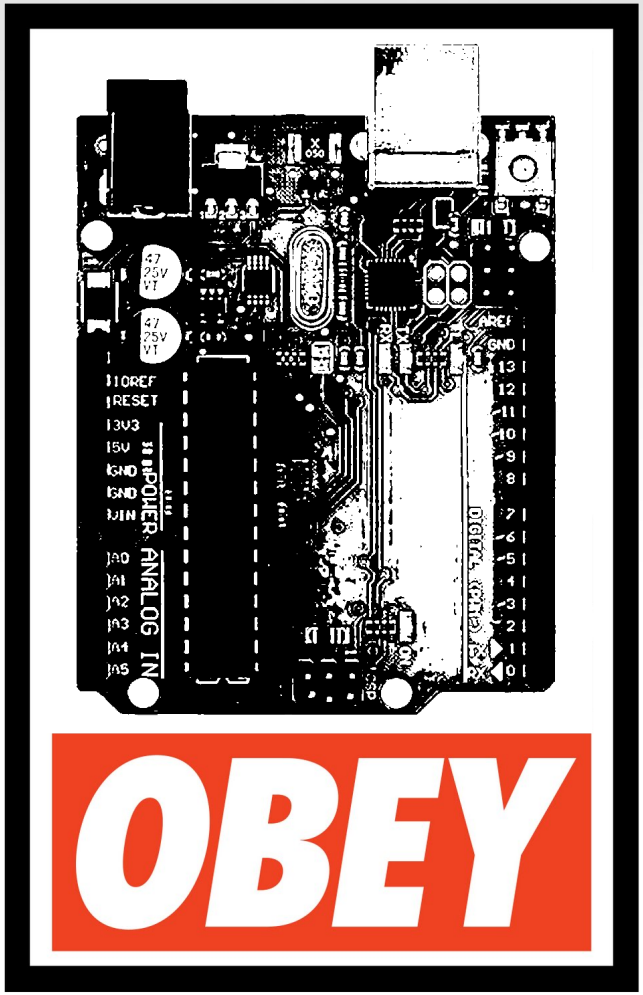


fritzing

Go ahead and plug your board in!

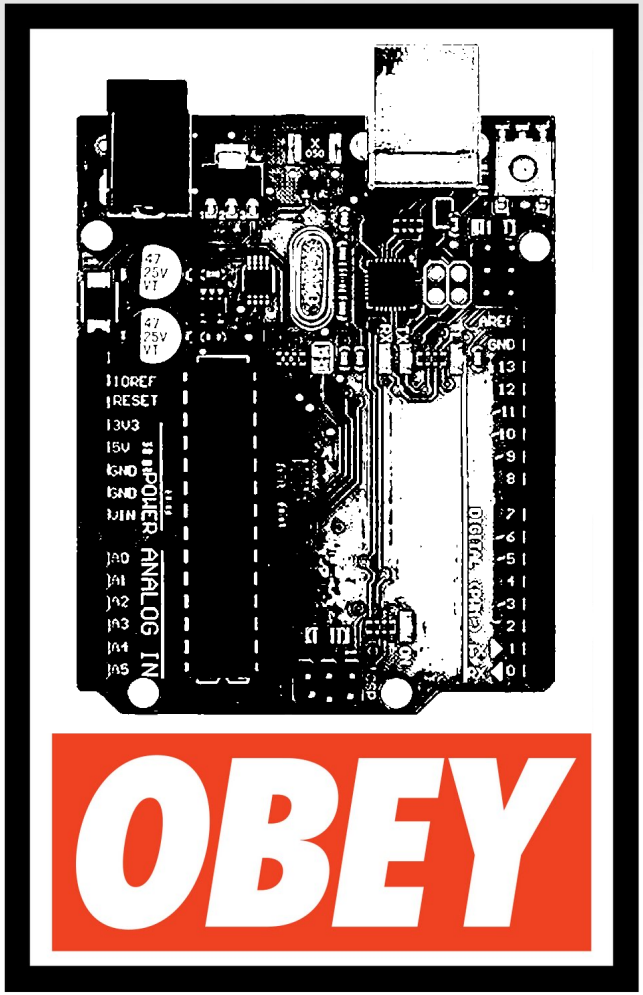


Adding Control



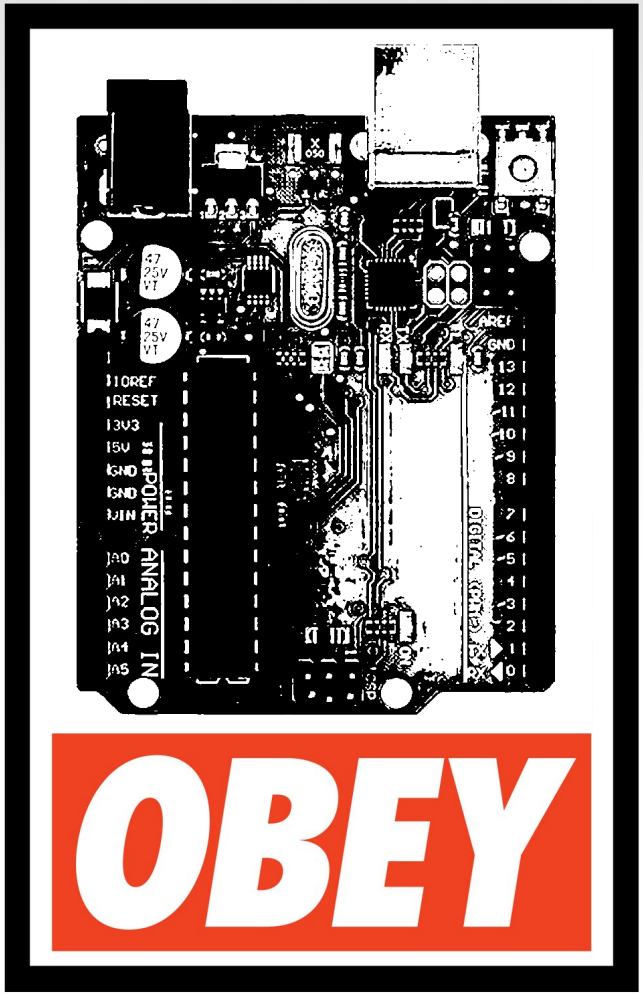
Adding Control

Any circuit that has meaningful input has some degree of control

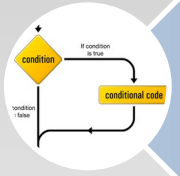


Adding Control

Any circuit that has meaningful input has some degree of control



Microcontrollers allow us to assign different relationships between inputs and outputs

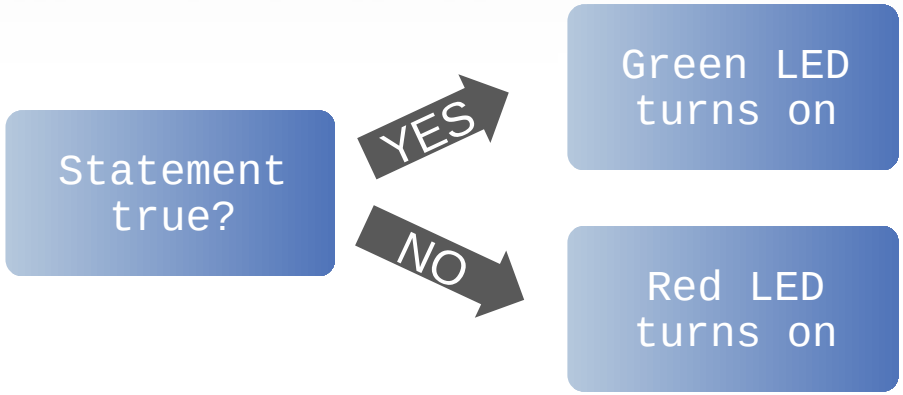


if() statements / Boolean logic

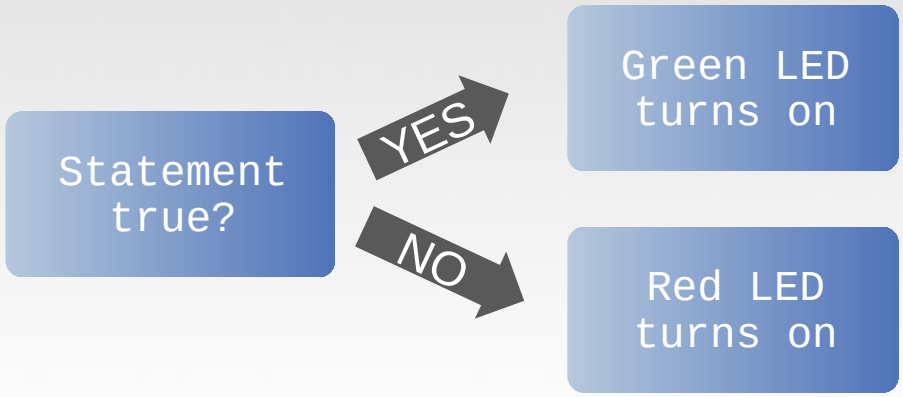
Project # 3 – Lie Detector

Just give me 3 clock cycles with the perp...

Psuedo-code – how should this work?

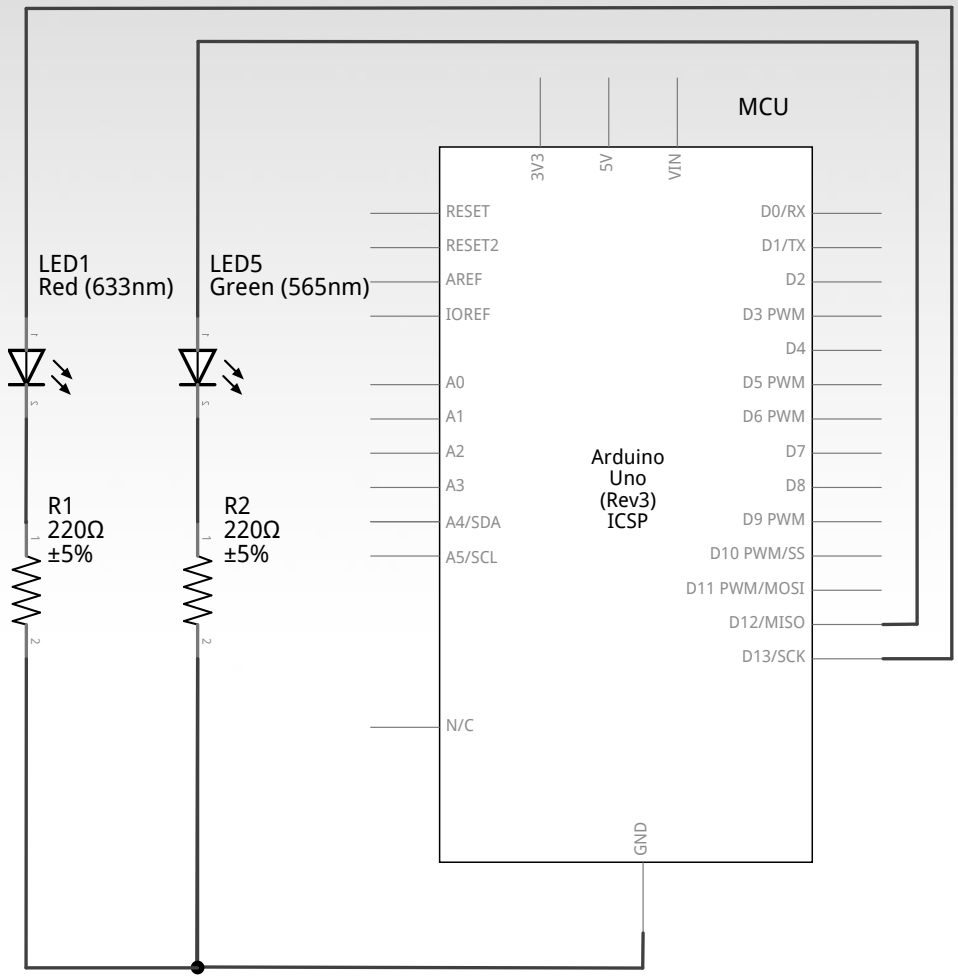


Project # 3 – Lie Detector
Inputs and Outputs



Inputs	Outputs
None	Green LED
	Red LED

Project # 3 – Lie Detector Schematic

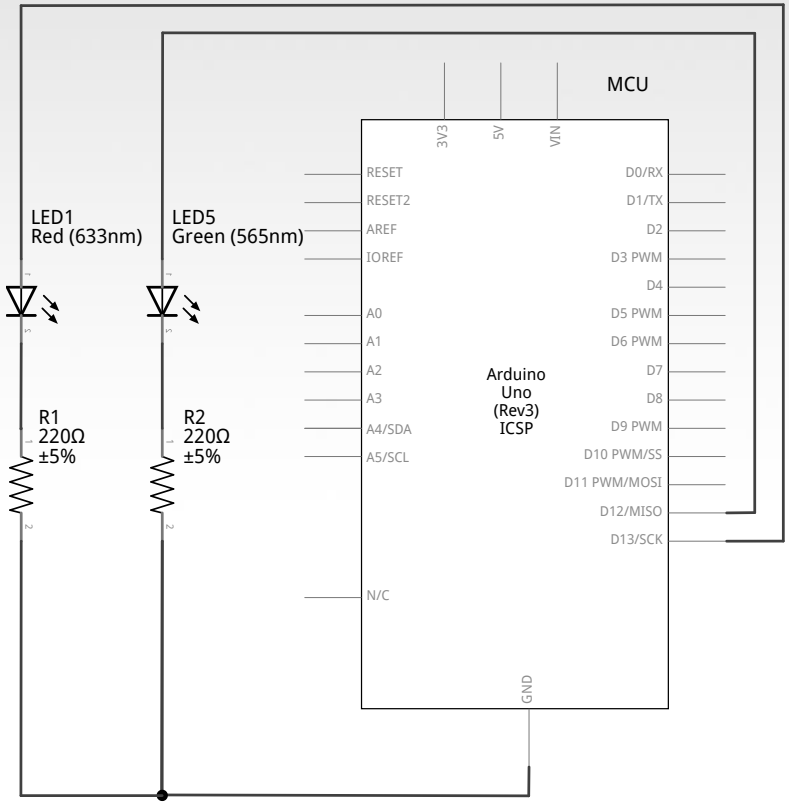


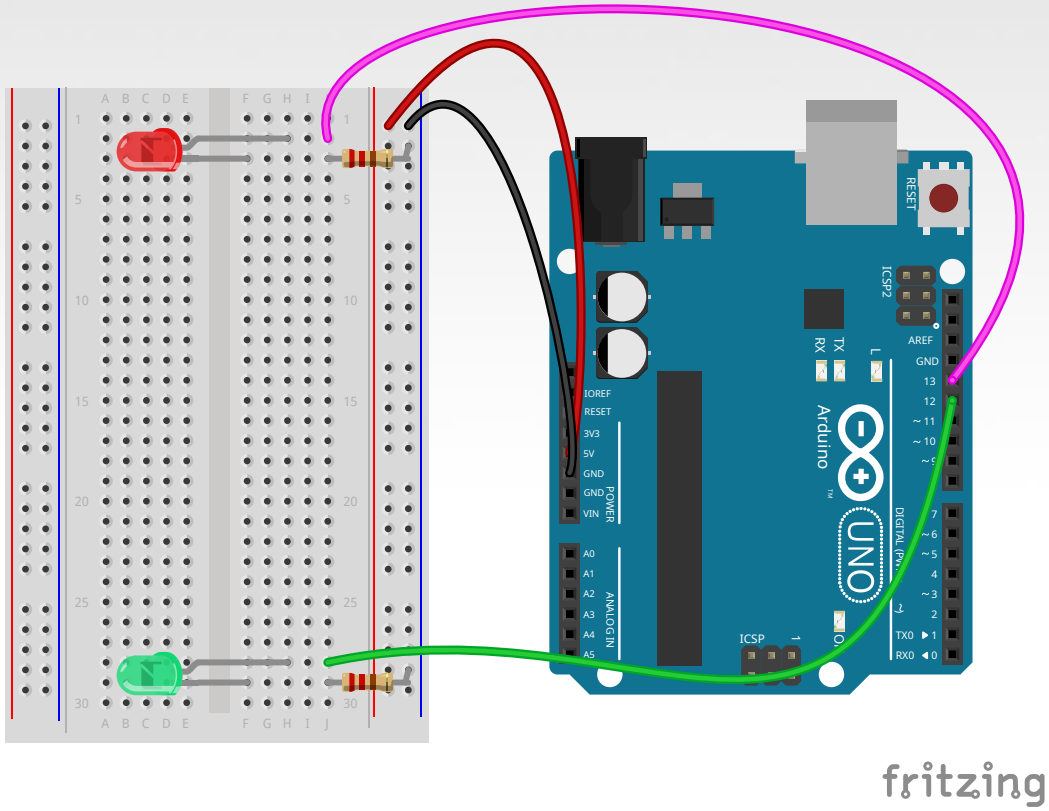
You can leave the button on the breadboard- we'll be using it again soon

fritzing

Project # 3 – Lie Detector

Wiring Diagram





Project # 3 – Lie Detector
Conditional Statements

the if() statement

```
if (varA > varB) {
  analogWrite(led, varA);
}
else {
  analogWrite(led, varB);
}
```

Project # 3 – Lie Detector
Conditional Statements

```
if (varA > varB) {
  analogWrite(led, varA);
}
else {
  analogWrite(led, varB);
}
```

If this is true

Project # 3 – Lie Detector
Conditional Statements

```
if (varA > varB) {
  analogWrite(led, varA);
}
else {
  analogWrite(led, varB);
}
```

Project # 3 – Lie Detector
Conditional Statements

```
if (varA > varB) {
  analogWrite(led, varA);
}
else {
  analogWrite(led, varB);
}
```

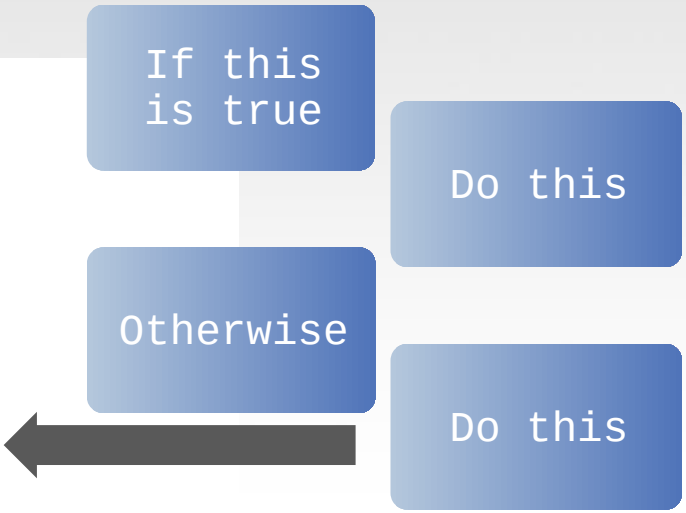
If this is true

Do this

Otherwise

Project # 3 – Lie Detector
Conditional Statements

```
if (varA > varB) {
  analogWrite(led, varA);
}
else {
  analogWrite(led, varB);
}
```



Boolean Operators

<Boolean>	True if:
`(a)`	a is true, HIGH or does not equal zero
`(!a)`	a is false, LOW or equals zero
`(a) == (b)`	a is equal to b
`(a) != (b)`	a is not equal to b
`(a) > (b)`	a is greater than b
`(a) >= (b)`	a is greater than or equal to b
`(a) < (b)`	a is less than b
`(a) <= (b)`	a is less than or equal to b
`(a) && (b)`	both a is true AND b is true
`(a) || (b)`	either a is true OR b is true

Déjà Vu

Select Board

fade.ino

```
/*
 * Gradually fade an LED on and off
 */

// assign pin based off circuit
int led = 9;

int brightness = 0;     // holds the value for how bright the led is
int fade_amount = 5;    // how much the brightness changes each loop

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 9 as an output.
  pinMode(led, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  analogWrite(led, brightness);               // set the brightness of pin 9

  brightness = brightness + fade_amount;      // update the brightness for the next loop

  if (brightness <= 0 || brightness >= 255) { // if the LED is all the way off or on
    fade_amount = fade_amount * -1;           // change the direction of the fade
  }
  delay(30);                                  // pause to see the dimming effect
}

```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = false;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (a) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = false;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (b) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = false;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (!b) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = false;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (!a) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = 9;
int b = 5;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (a > b) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = 9;
int b = 5;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (a < b) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = 5;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (b > 5 || a) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = 5;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (b > 5 && a) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

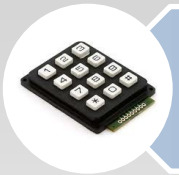
Project # 3 – Lie Detector

if_tester.ino

```
/*
 * Test our if() statements
 */

// assign pin based off circuit
int red_led = 13;
int green_led = 12;

// these are the variables we'll play around with
int a = true;
int b = 5;

// the setup function runs once when you press reset or power the board
void setup() {
  pinMode(red_led, OUTPUT);
  pinMode(green_led, OUTPUT);

  // change the expression inside the if() and look at the results
  if (b >= 5 && a) {
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, HIGH);
  }
  else {
    digitalWrite(green_led, LOW);
    digitalWrite(red_led, HIGH);
  }
}

// the loop function runs over and over again forever
void loop() {

}
```

Boolean Operators

<Boolean>	True if:
`(a)`	a is true, HIGH or does not equal zero
`(!a)`	a is false, LOW or equals zero
`(a) == (b)`	a is equal to b
`(a) != (b)`	a is not equal to b
`(a) > (b)`	a is greater than b
`(a) >= (b)`	a is greater than or equal to b
`(a) < (b)`	a is less than b
`(a) <= (b)`	a is less than or equal to b
`(a) && (b)`	both a is true AND b is true
`(a) || (b)`	either a is true OR b is true

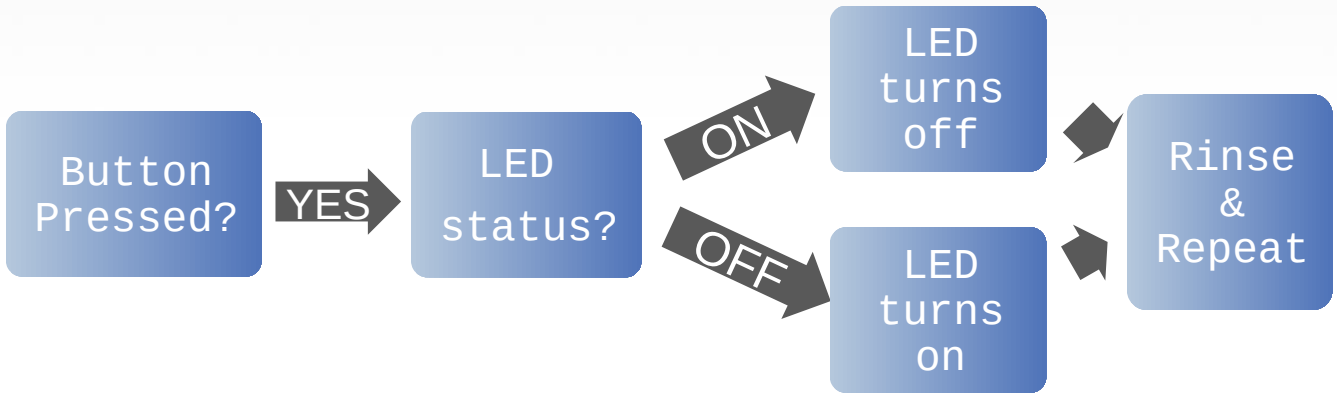


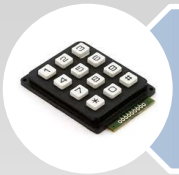
digitalRead()

Project # 4 – Button/Maintained Switch Switch

Who you callin' momentary?

Psuedo-code – how should this work?



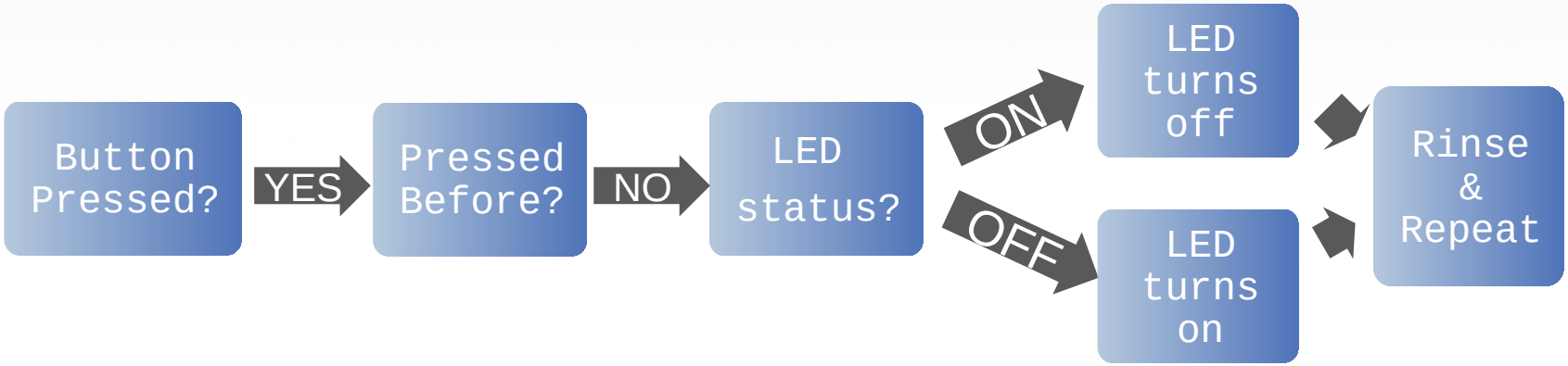


digitalRead()

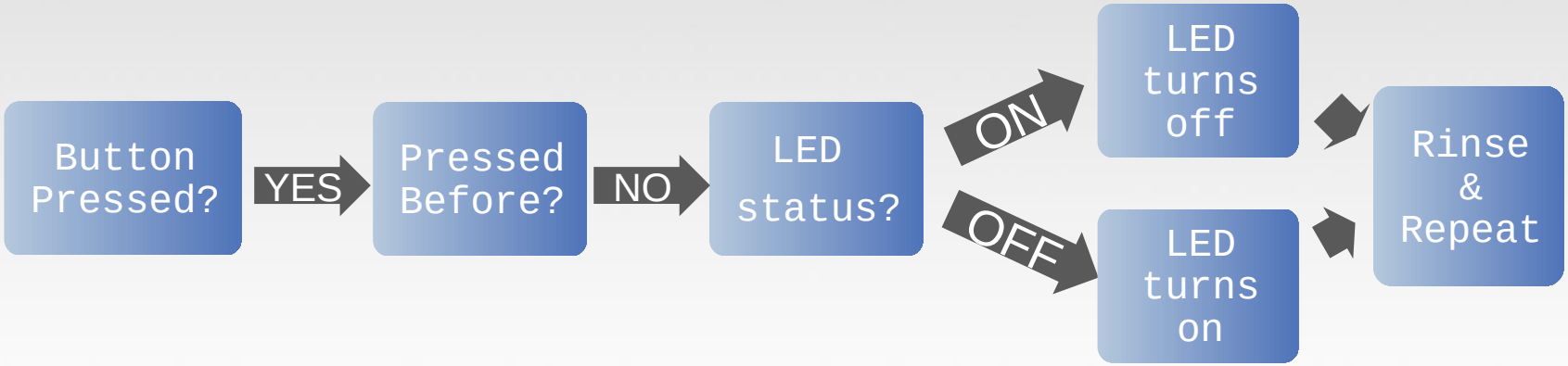
Project # 4 – Button/Maintained Switch Switch

Who you callin' momentary?

Psuedo-code – how should this work?

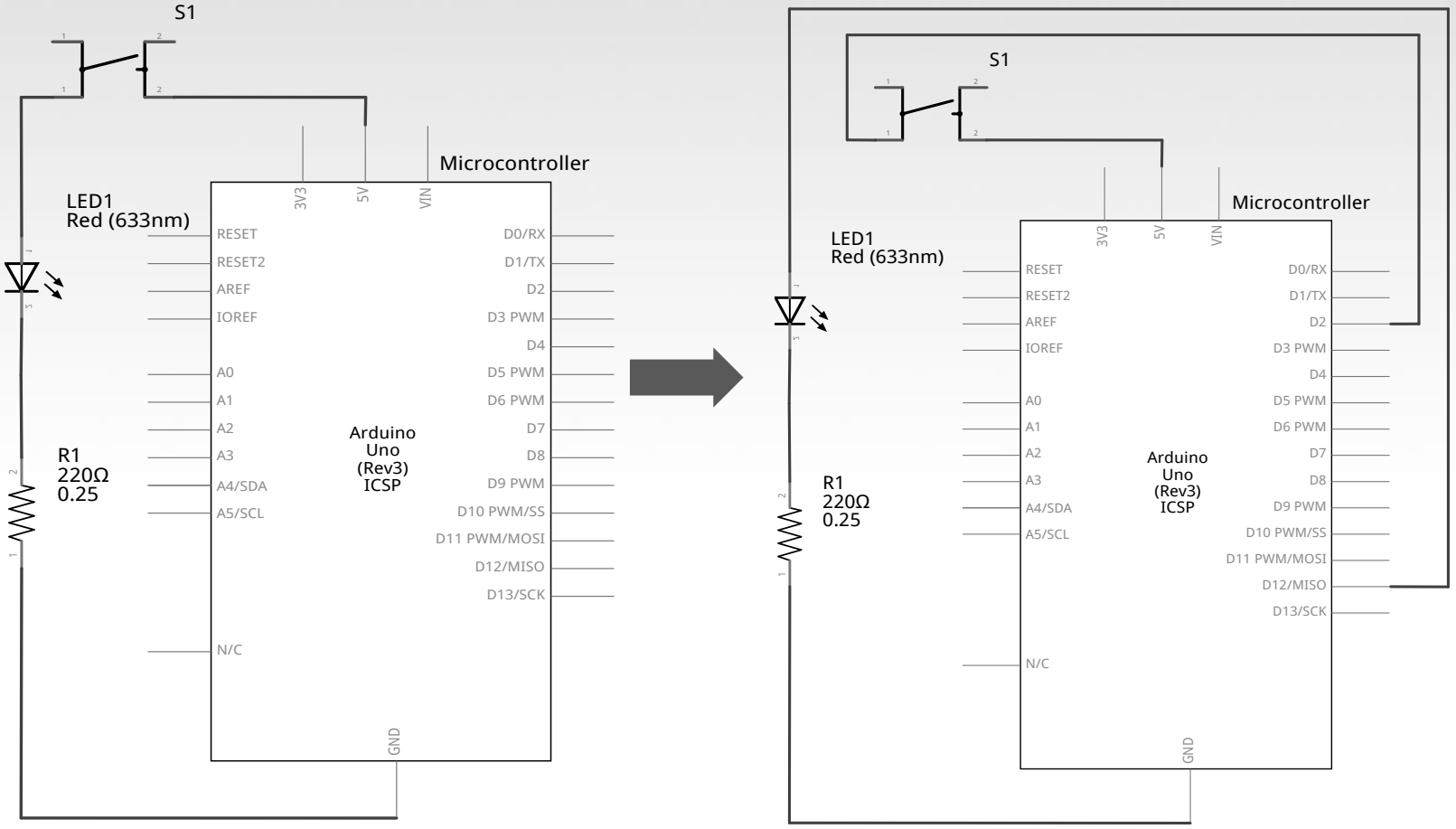


Project # 4 – Button/Maintained Switch Switch Inputs and Outputs

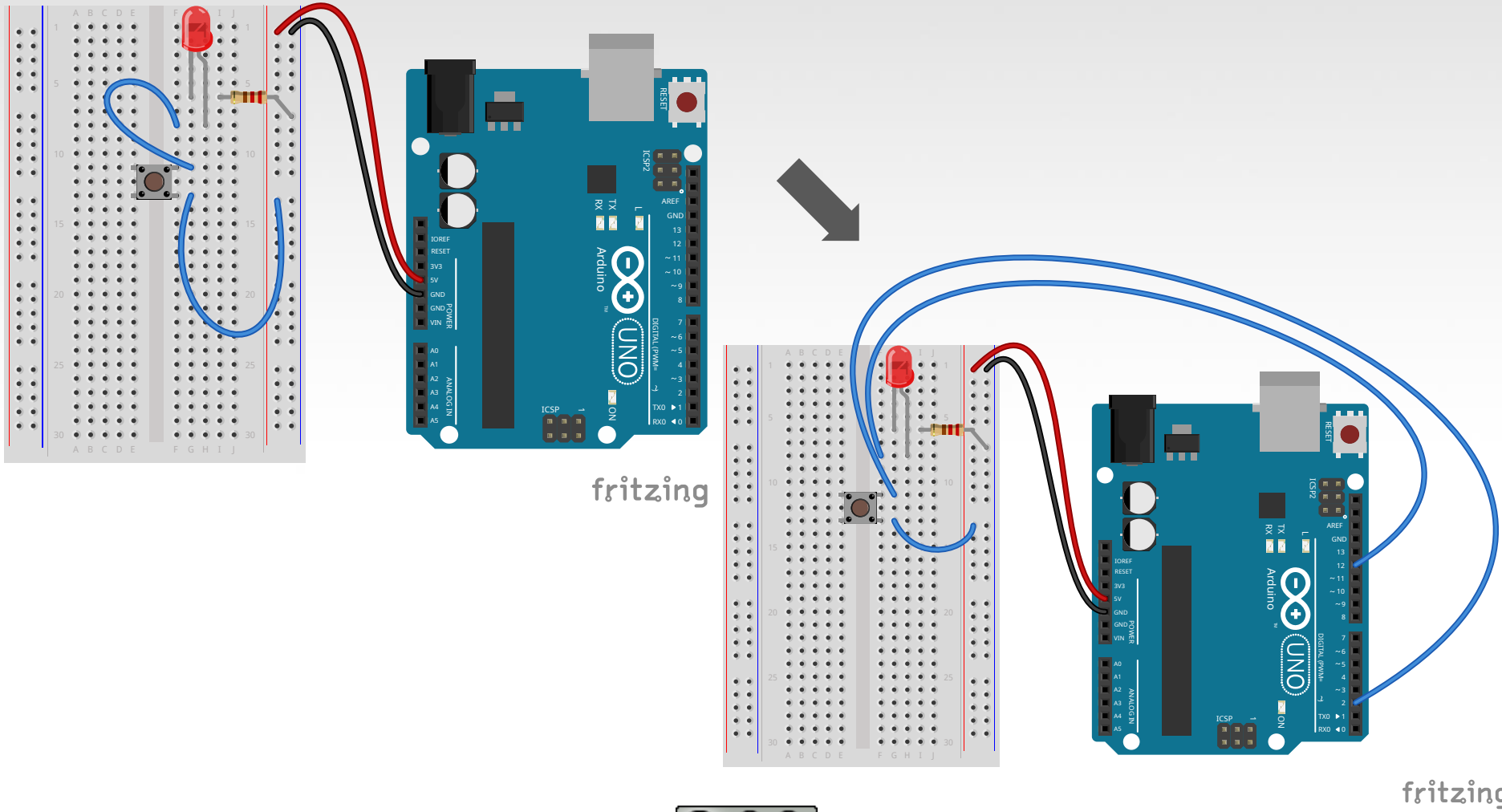


Inputs	Outputs
Push Button	LED

Transitioning to Micro-control Output & Input

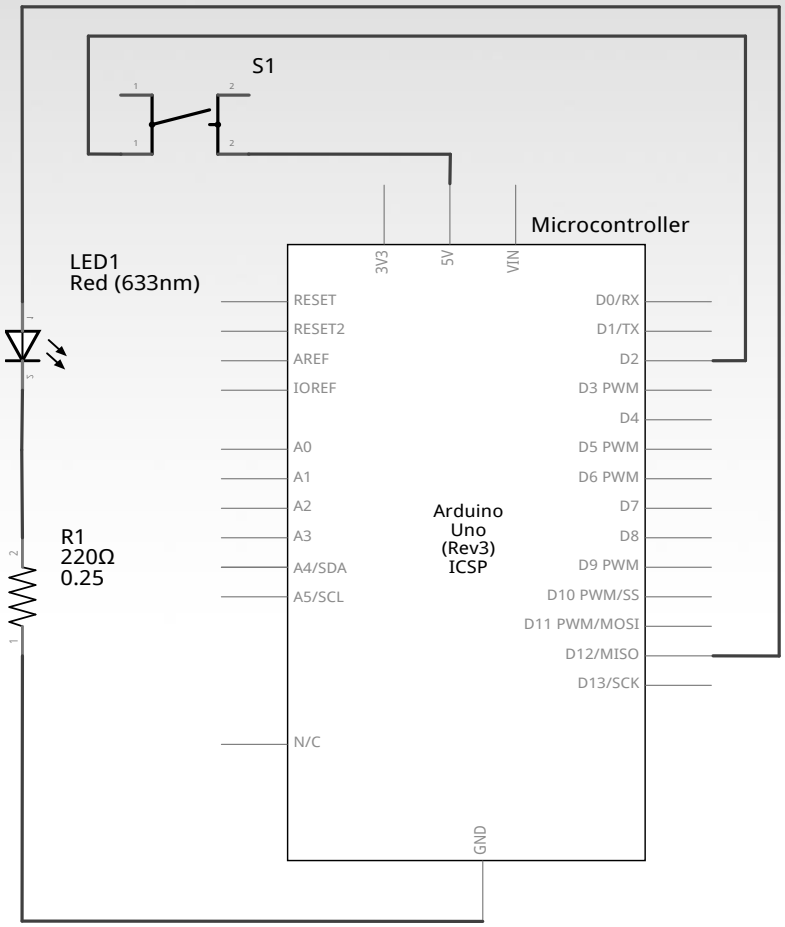


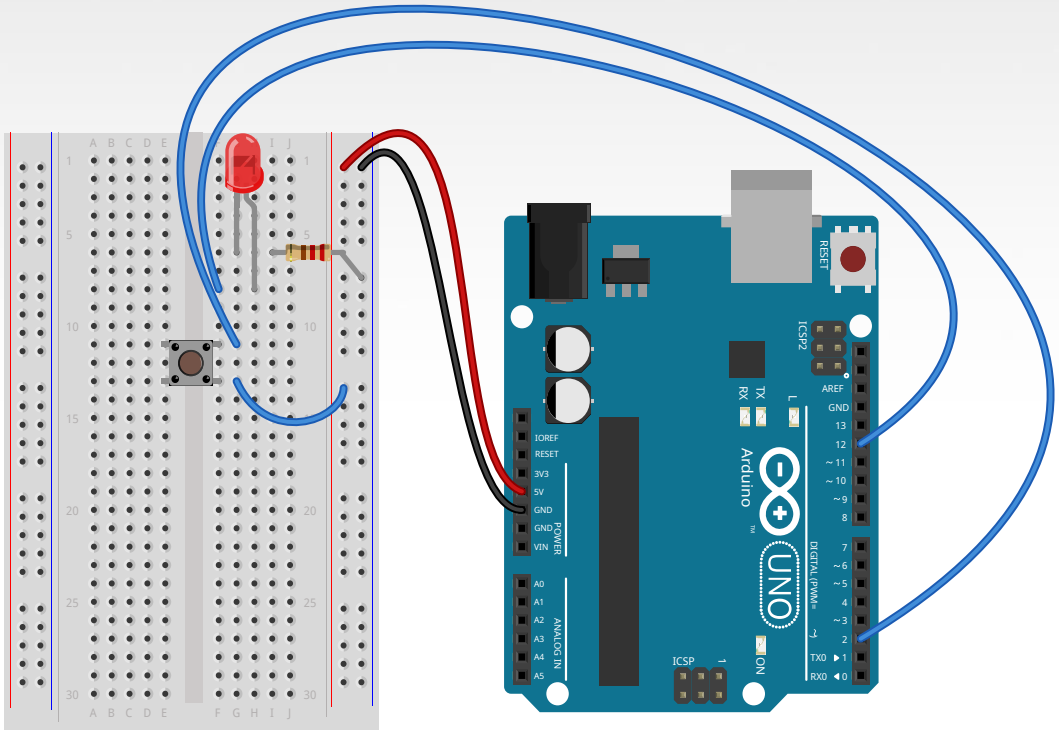
Transitioning to Micro-control Output & Input



Project # 4 – Button/Maintained Switch Switch

Wiring Diagram



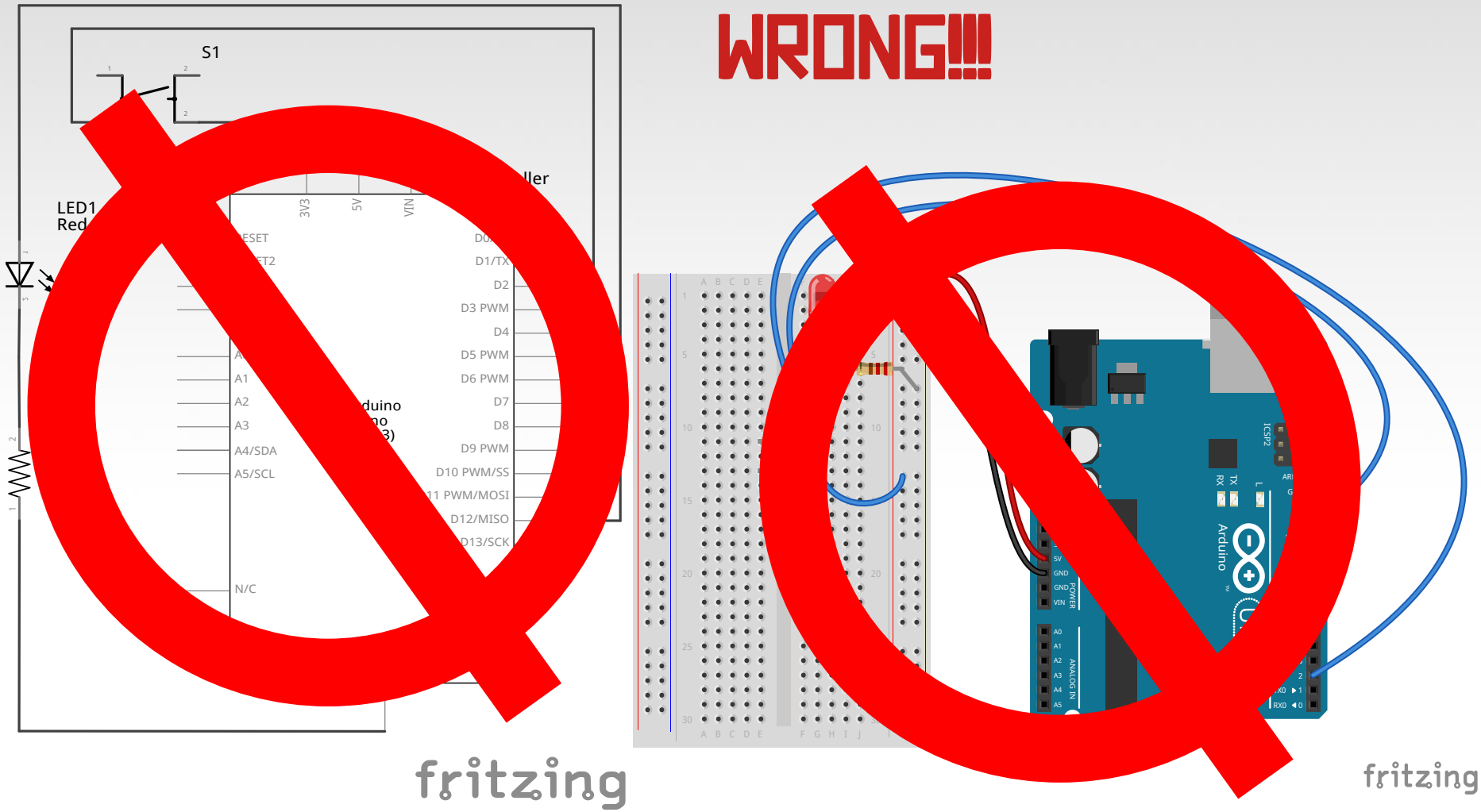


fritzing

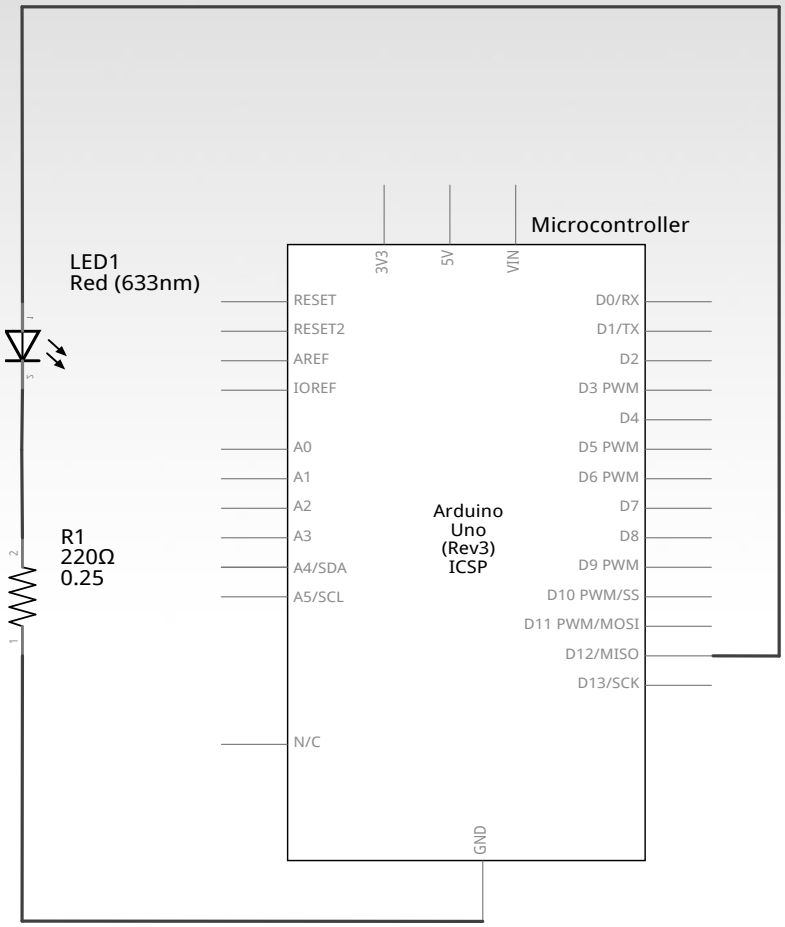
fritzing

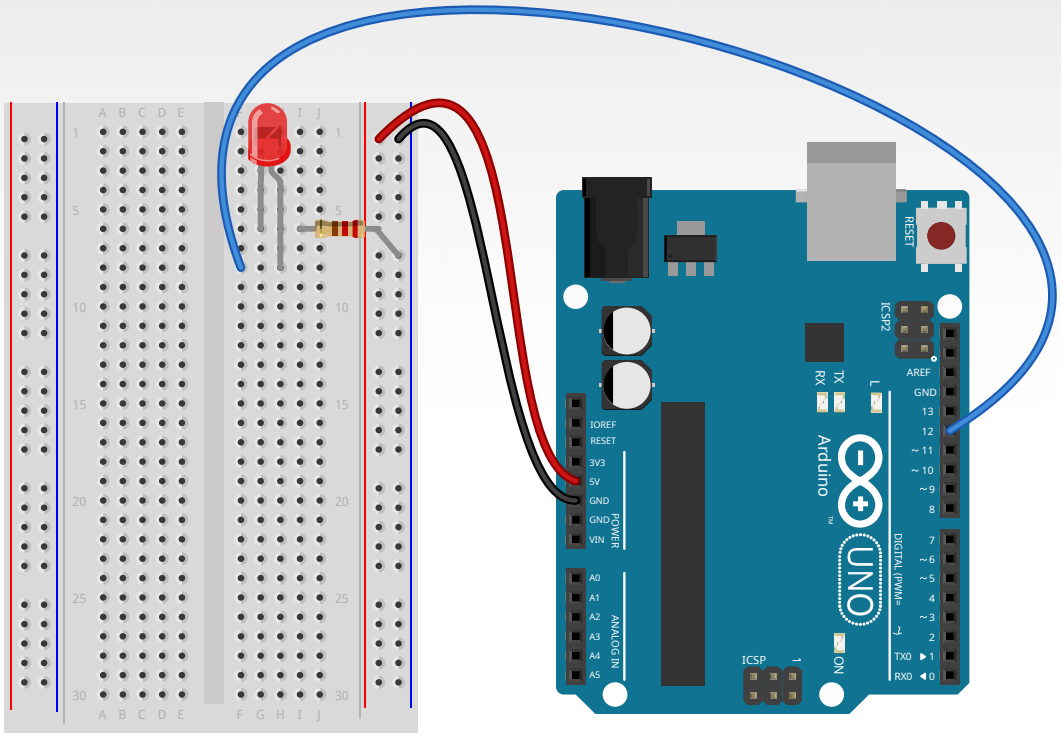
Project # 4 – Button/Maintained Switch Switch

Wiring Diagram

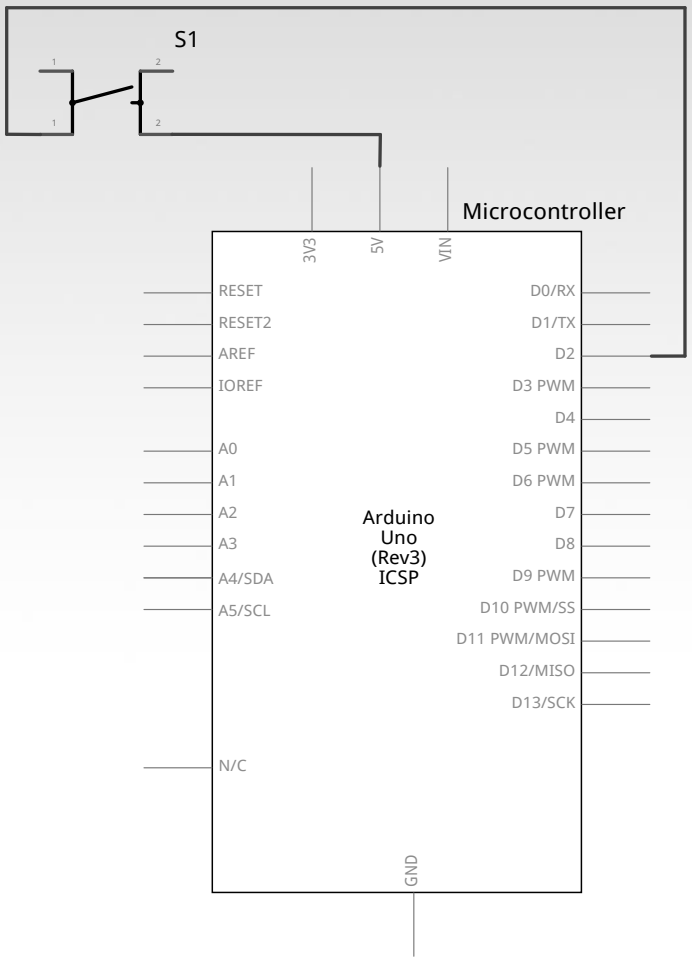


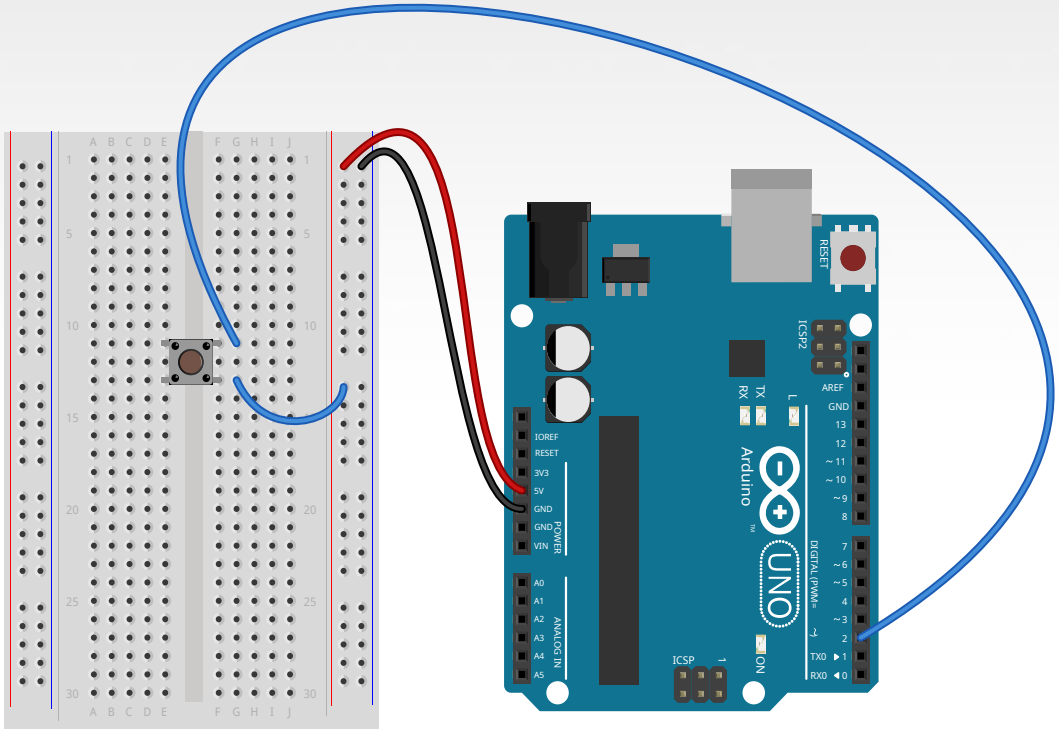
This Part Is Good





So This Must Be Bad...



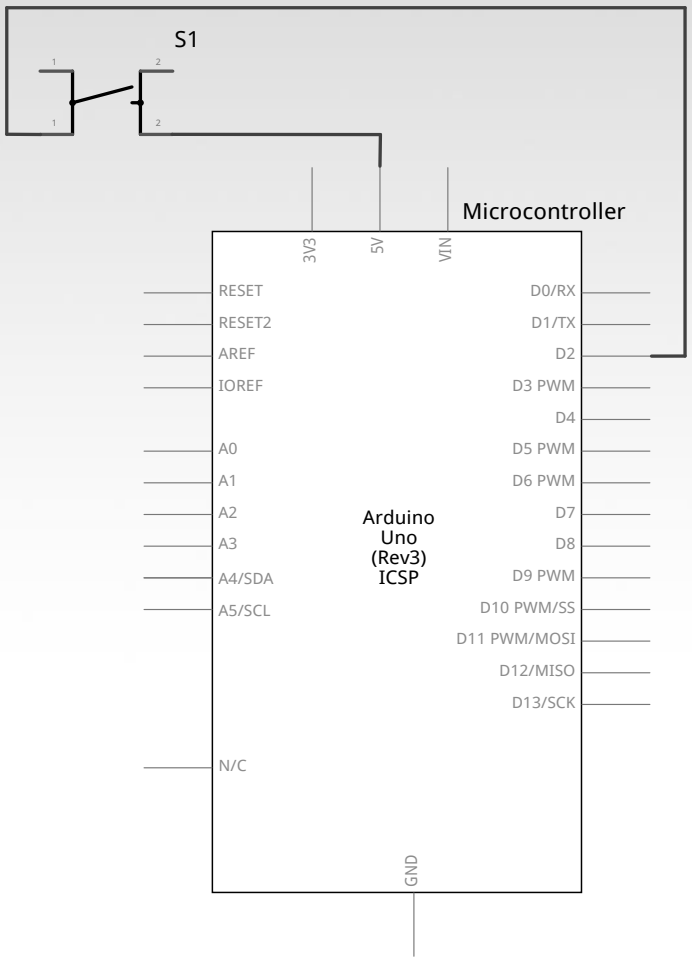


fritzing

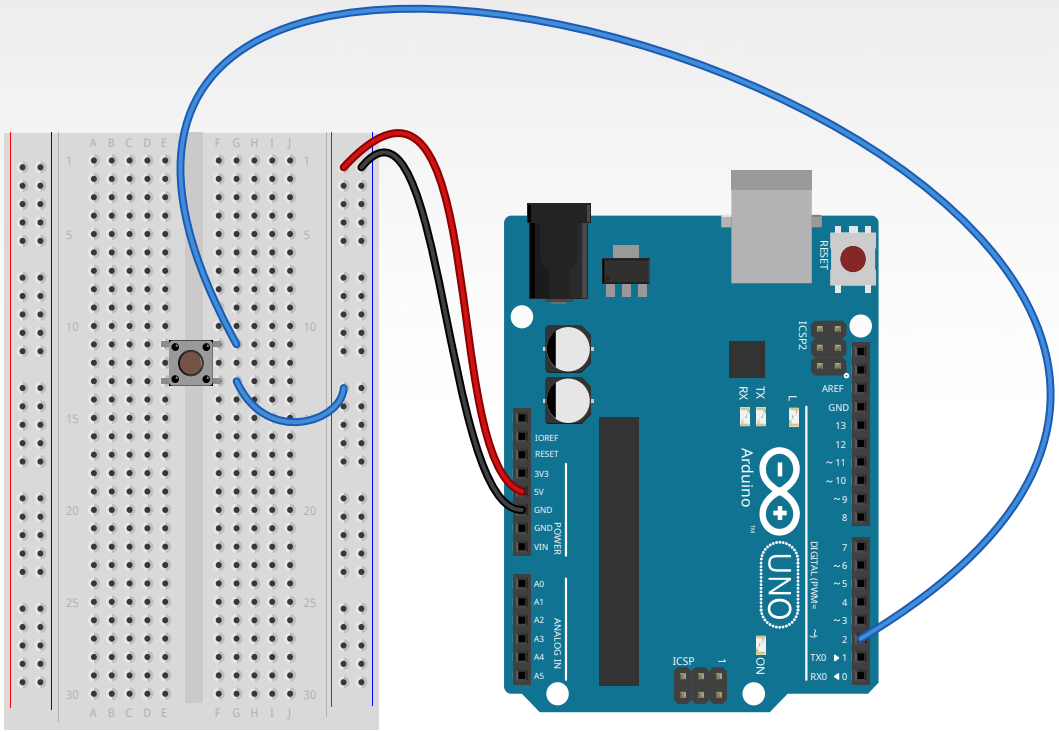
fritzing

So This Must Be Bad...

But Why?!

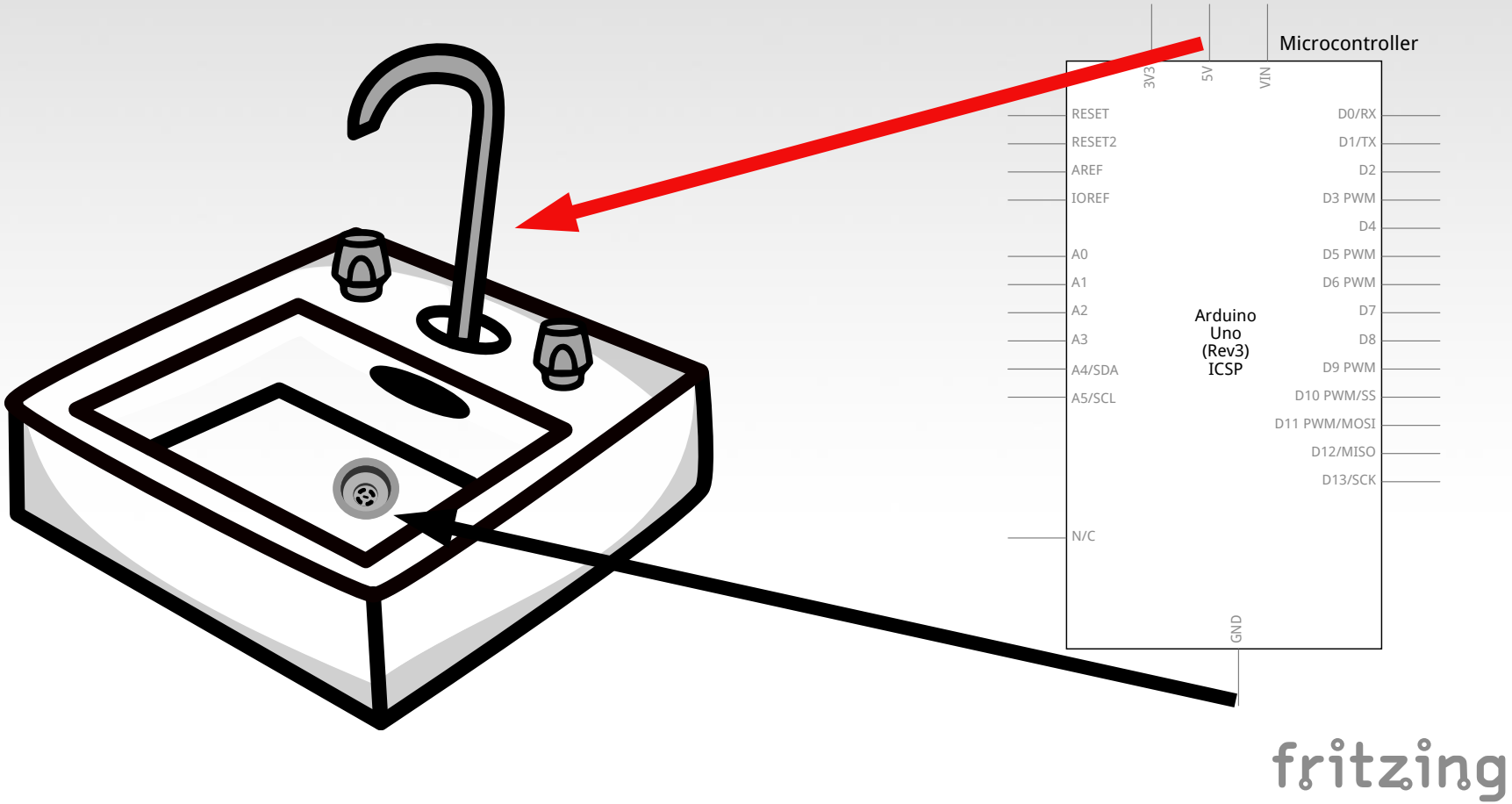


fritzing

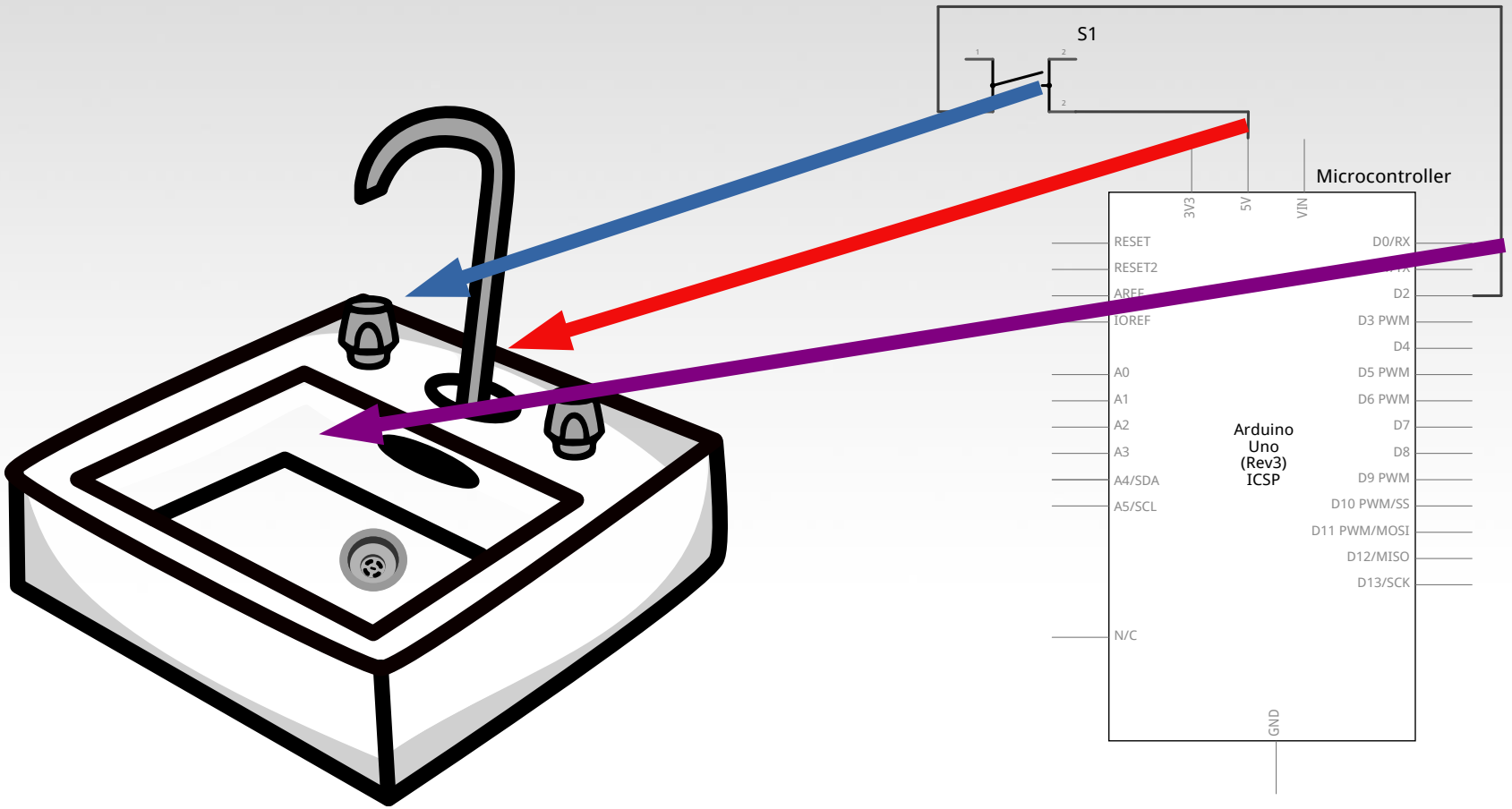


fritzing

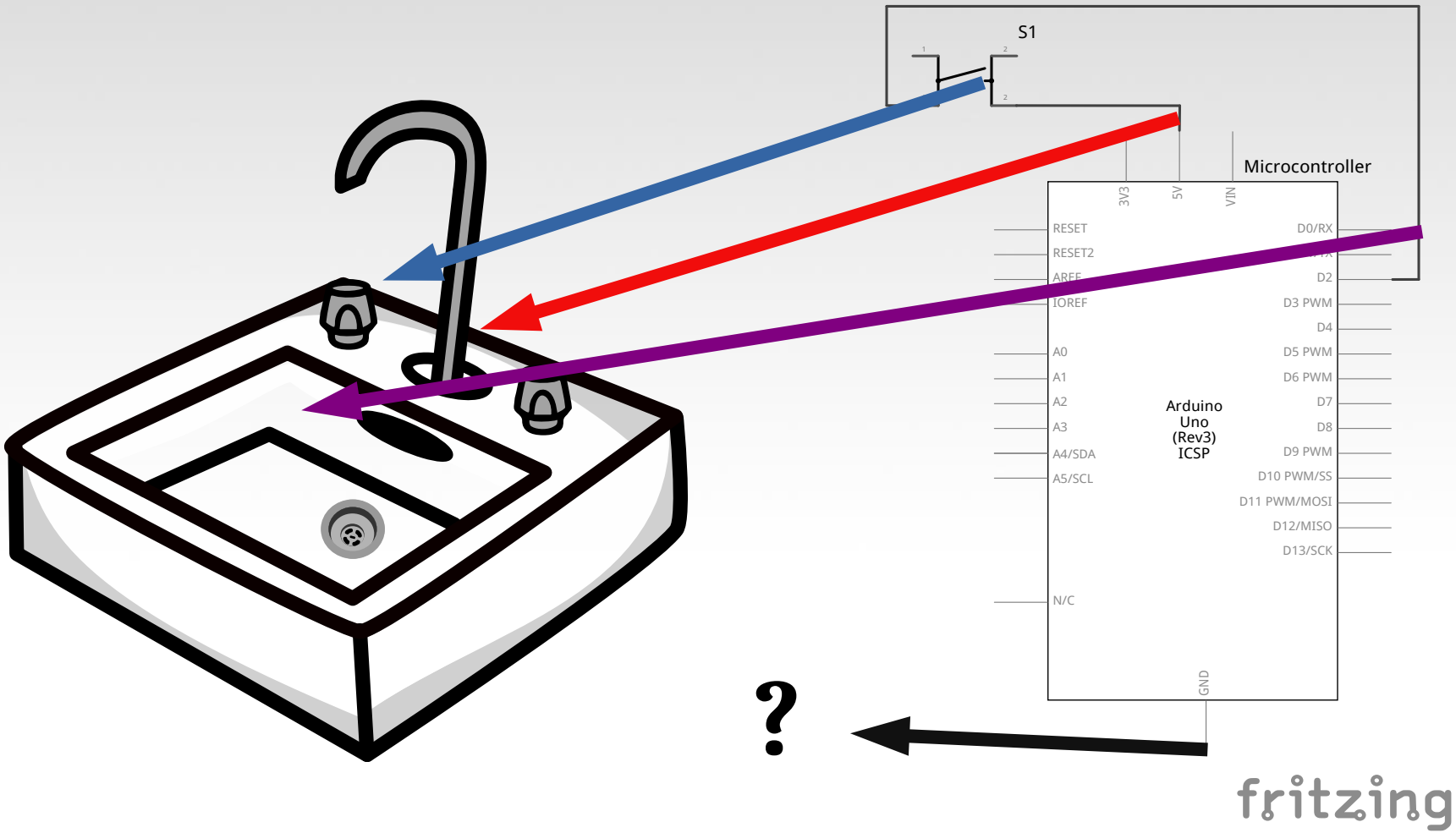
Another Water Analogy



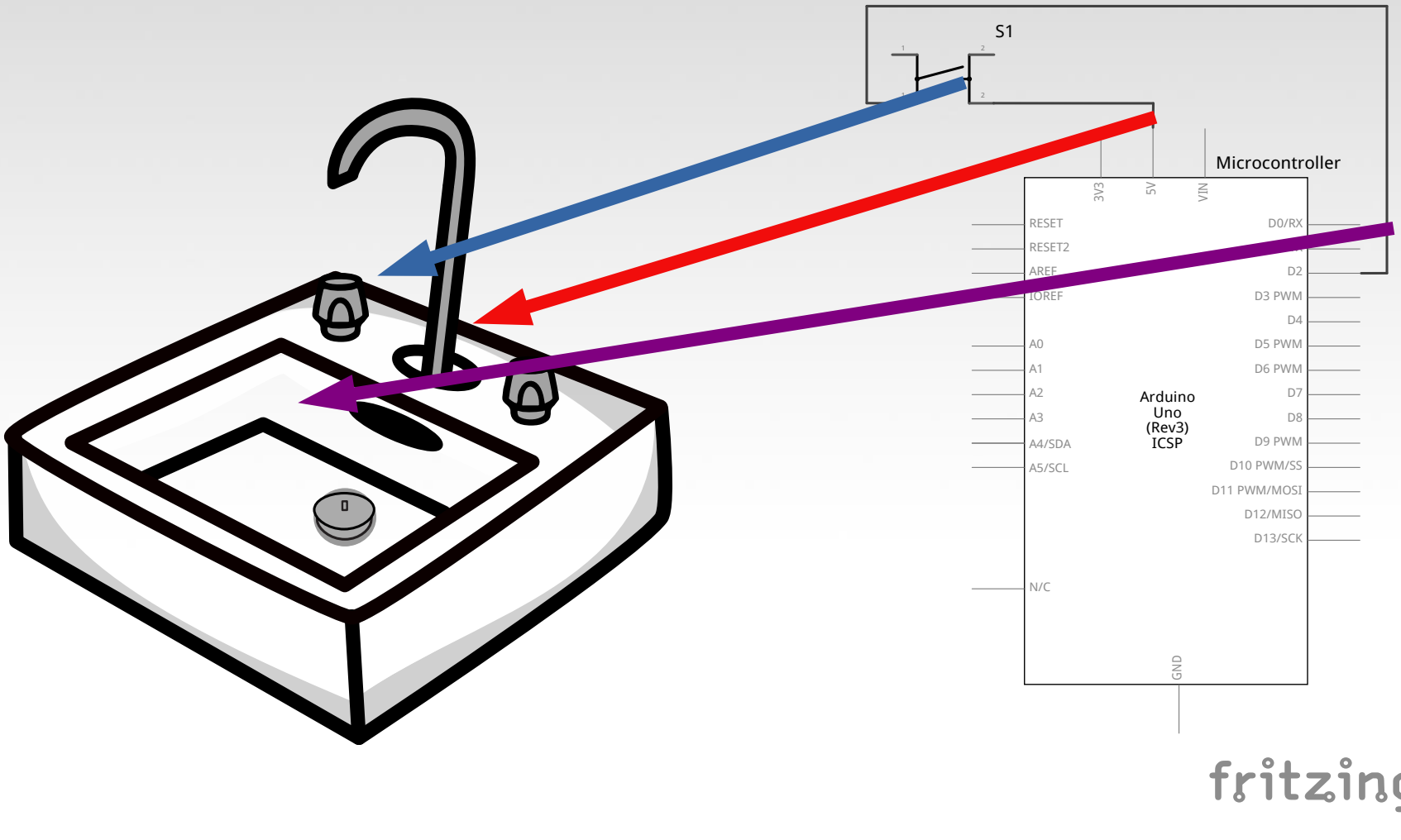
Analogy Continued...



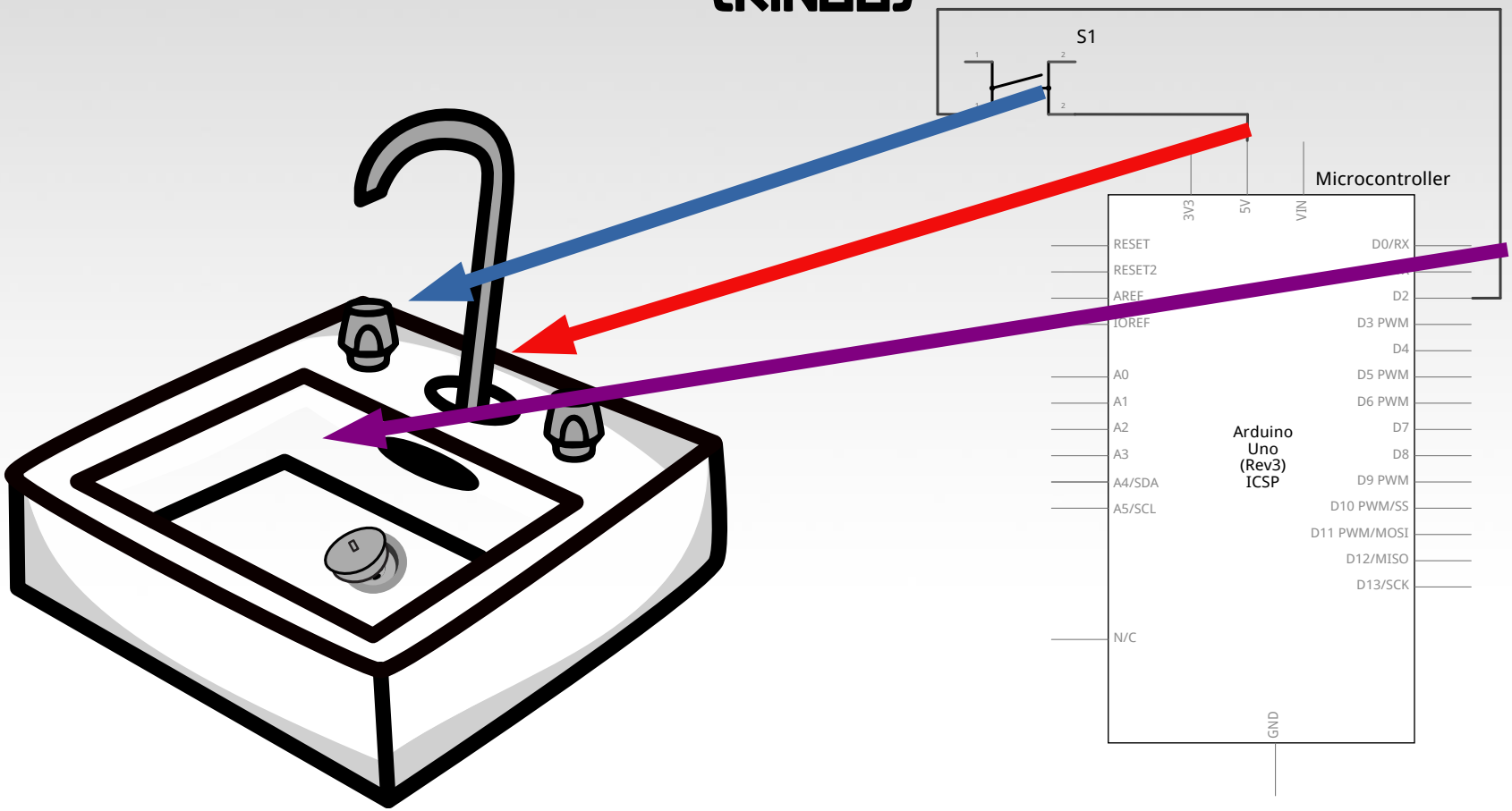
Analogy Continued...



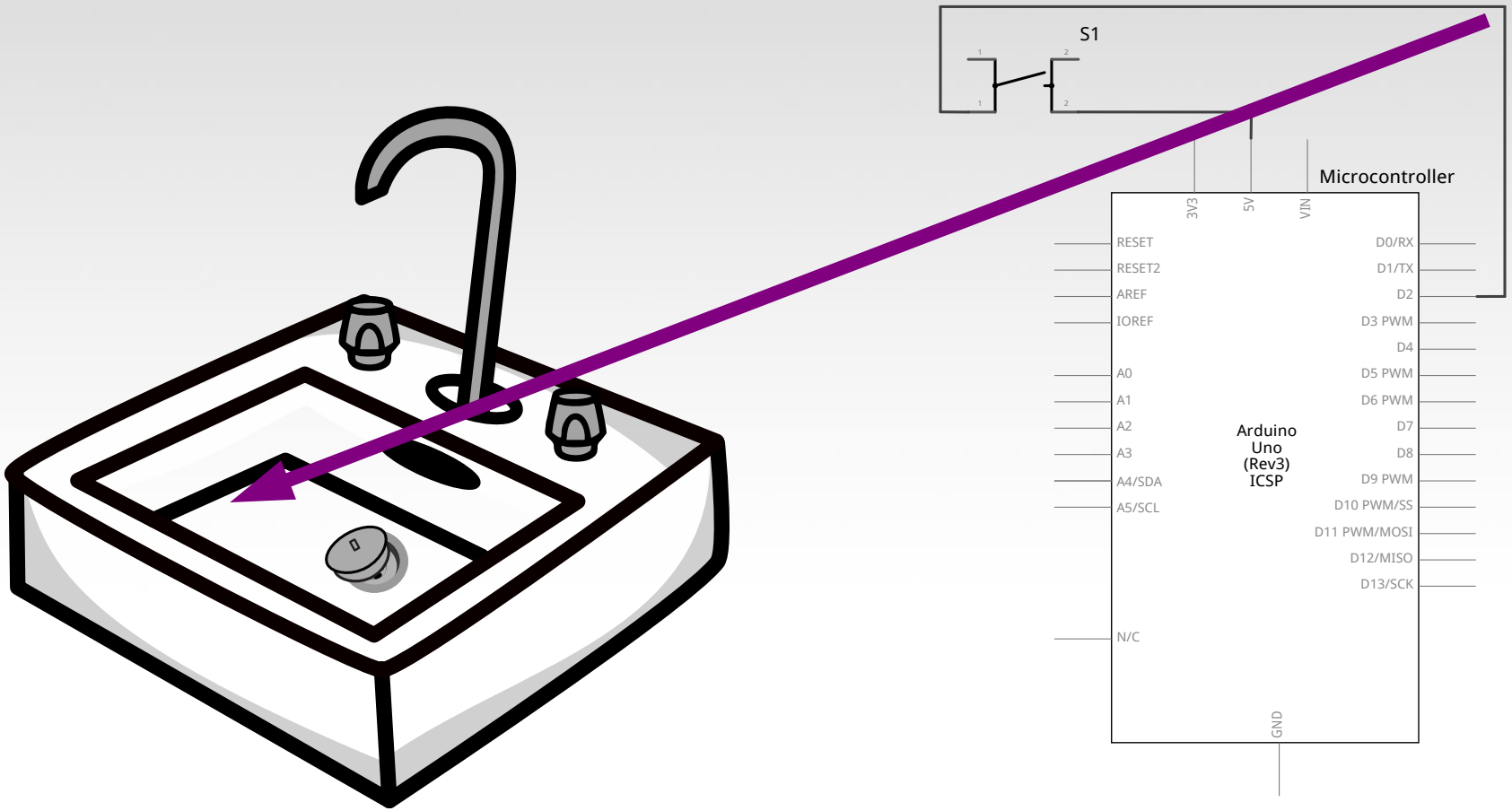
All Stopped Up



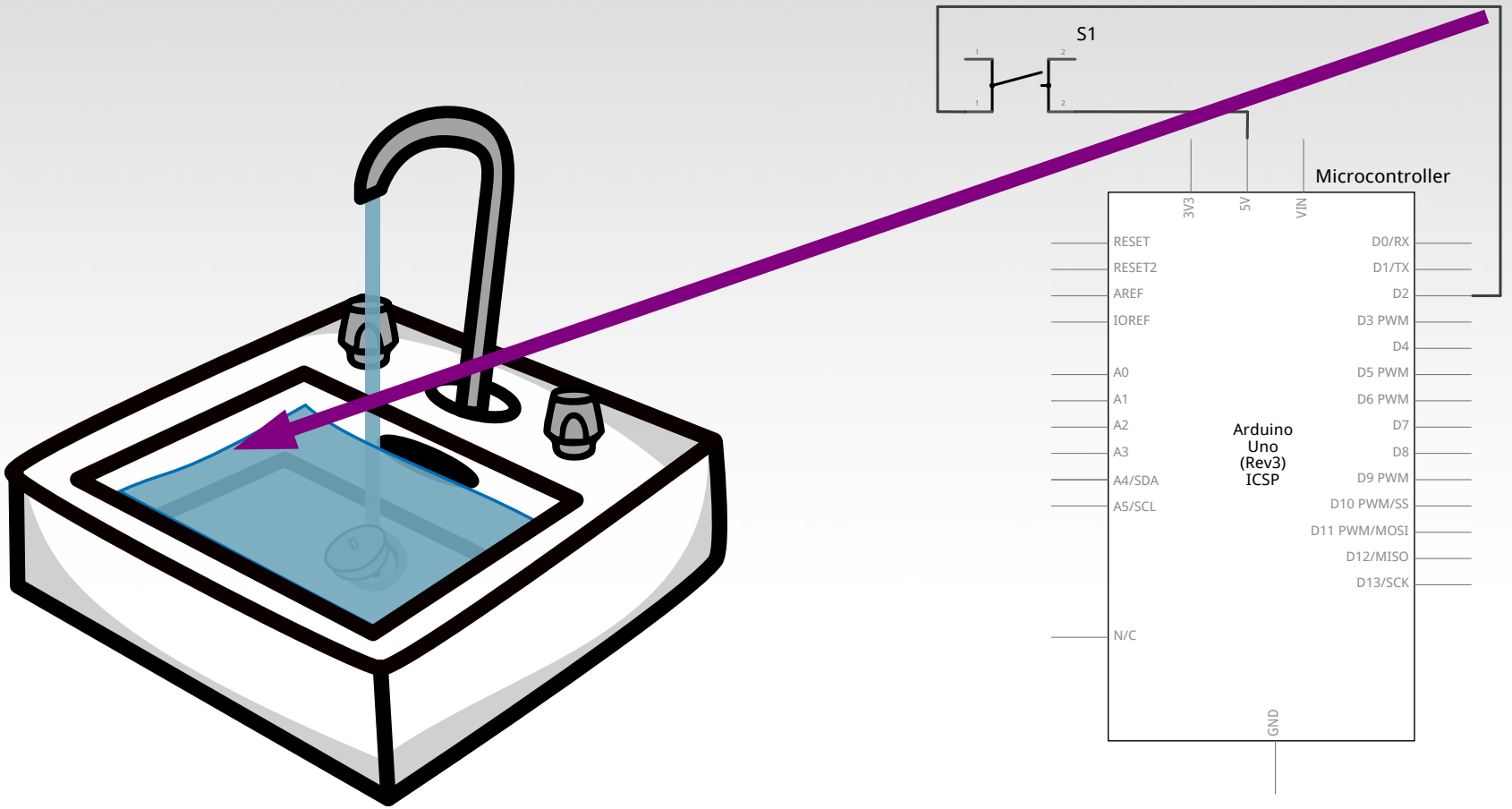
All Stopped Up (kinda)



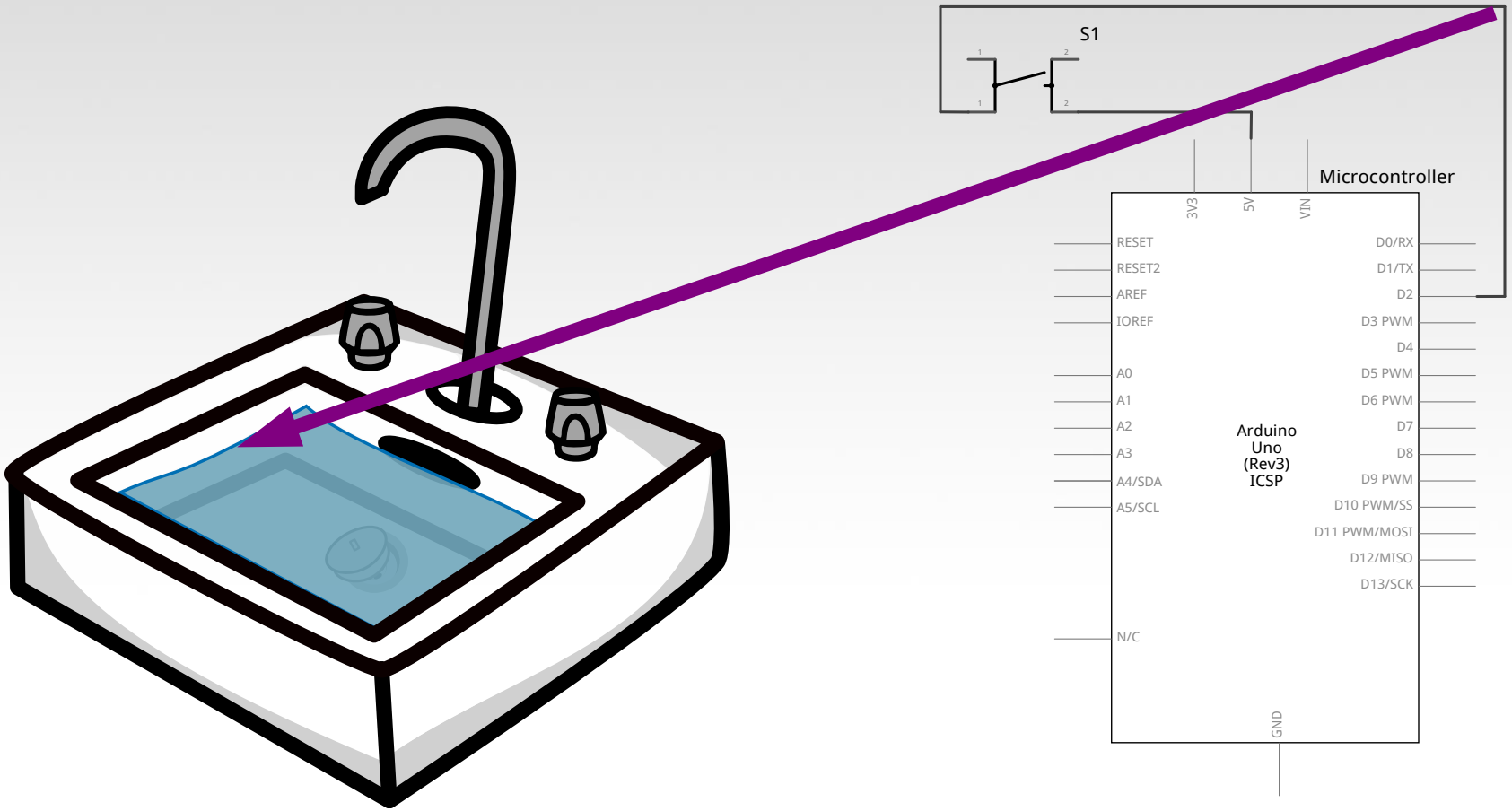
Switch Off



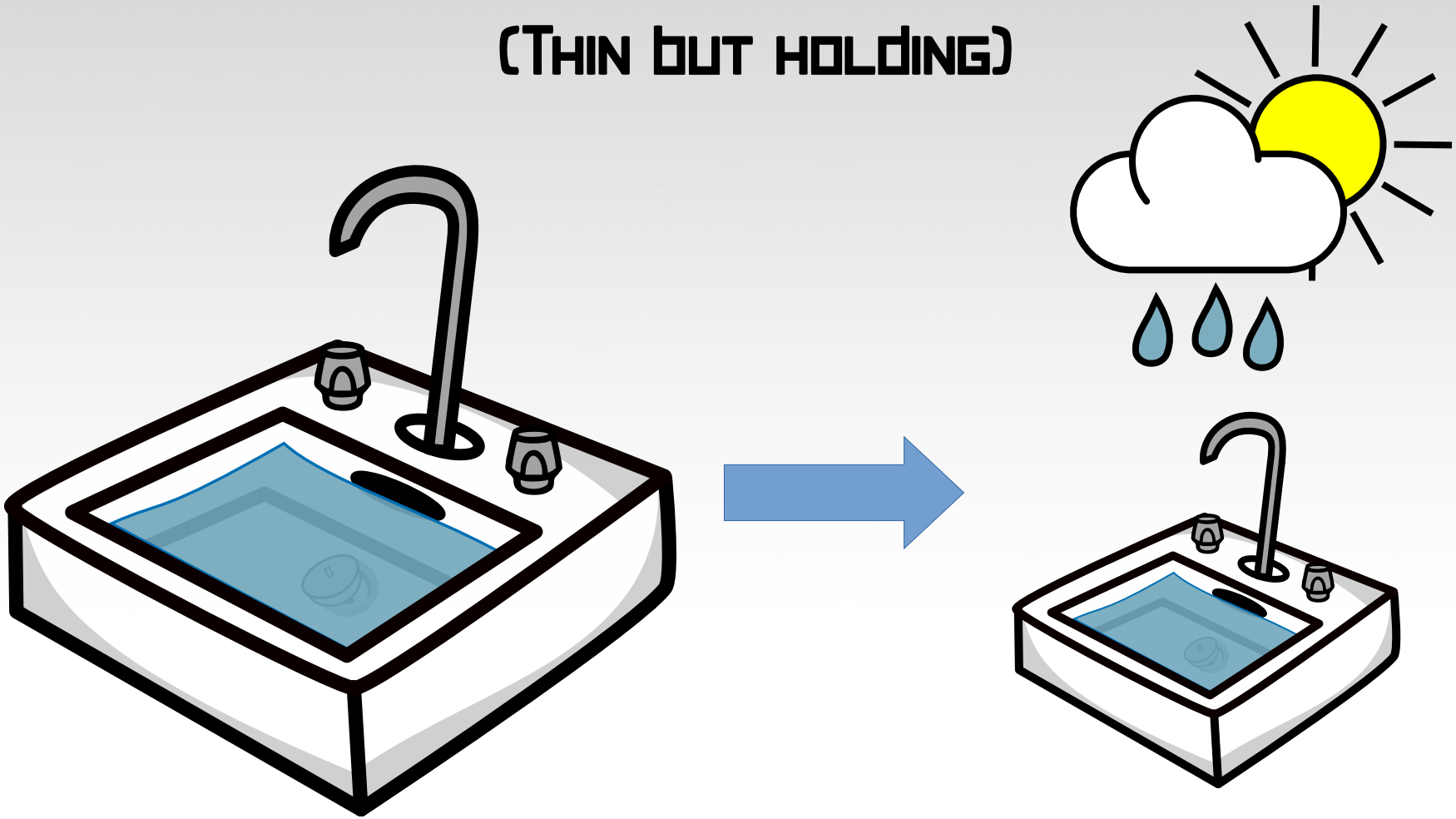
Switch On



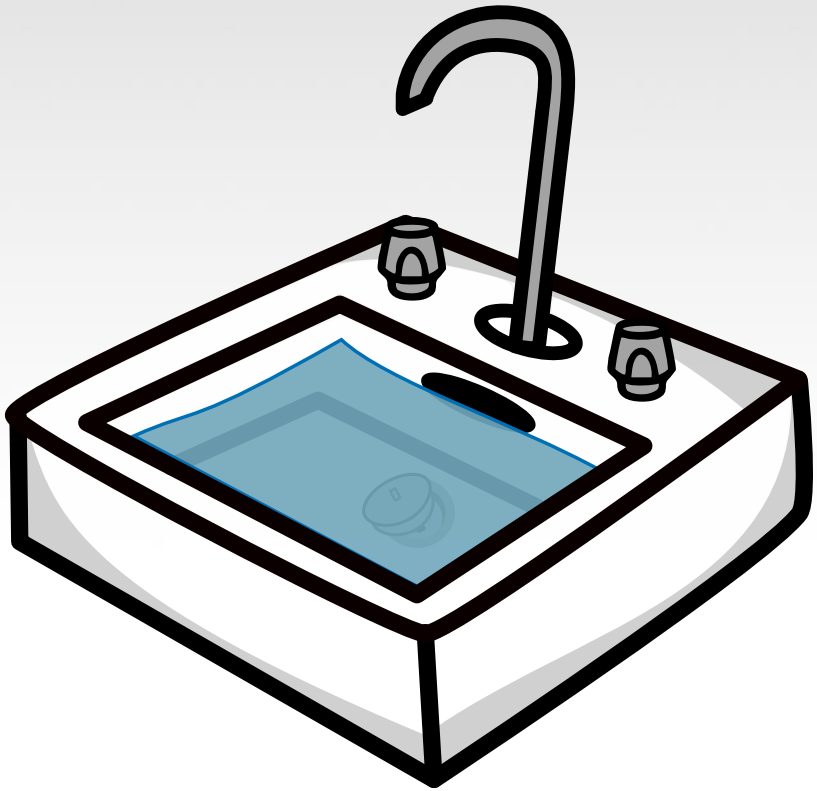
Switch Off?!

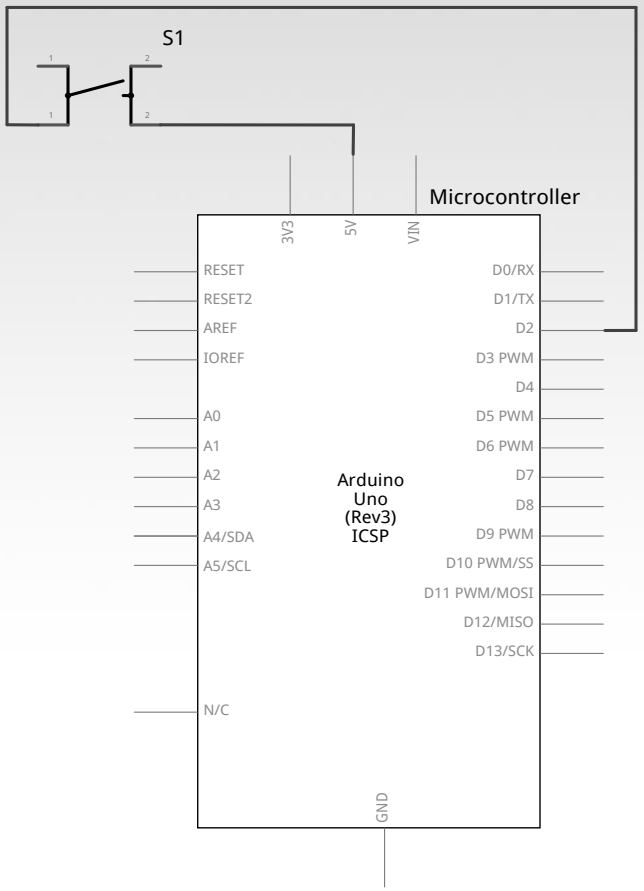


Analogy Extended
(Thin but holding)

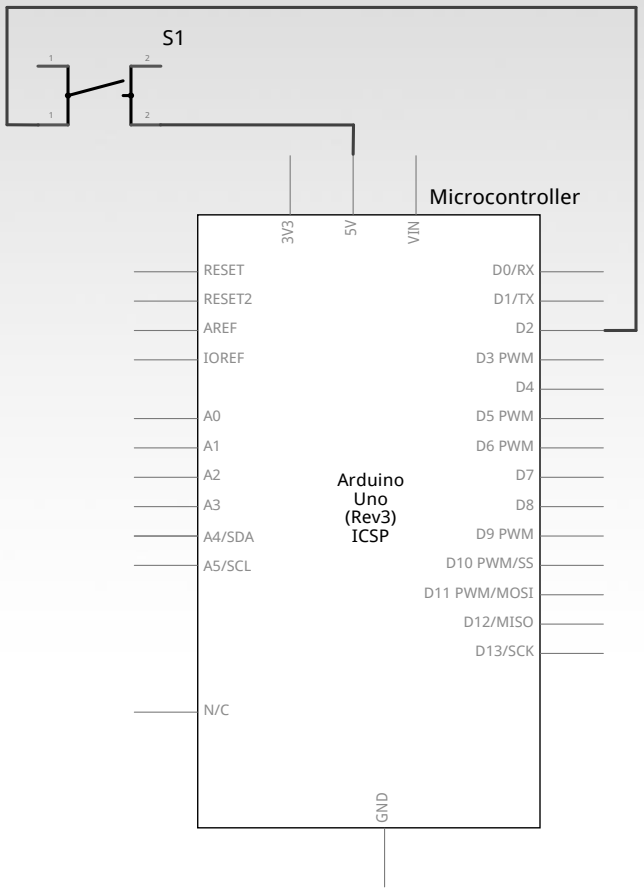


So This Approach...

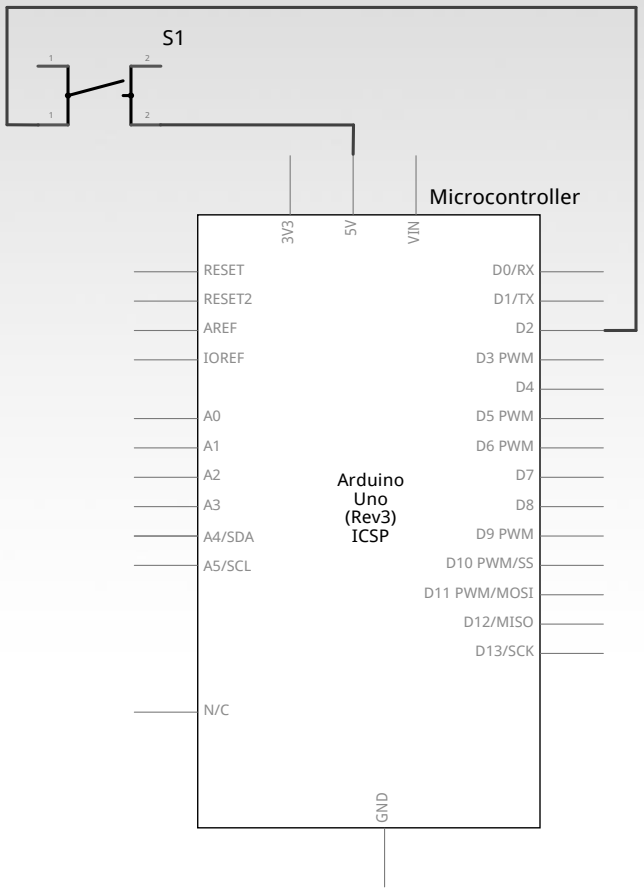




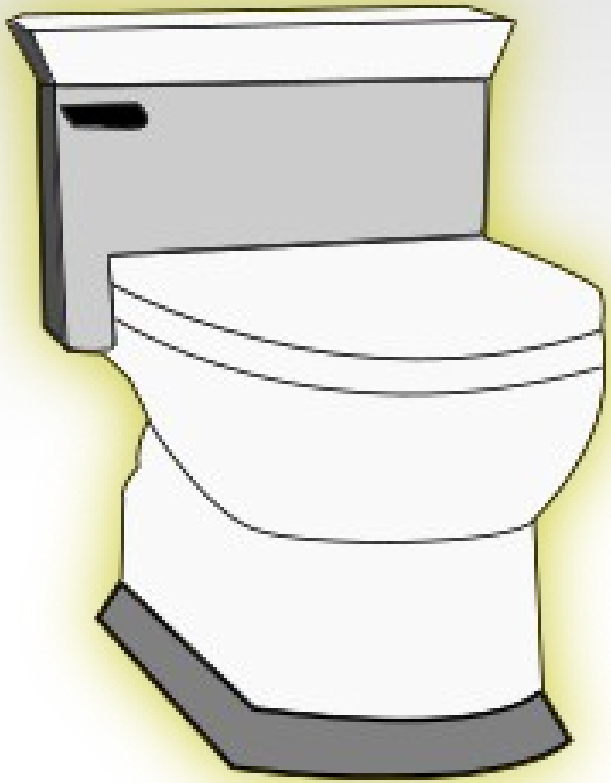
Doesn't Work!

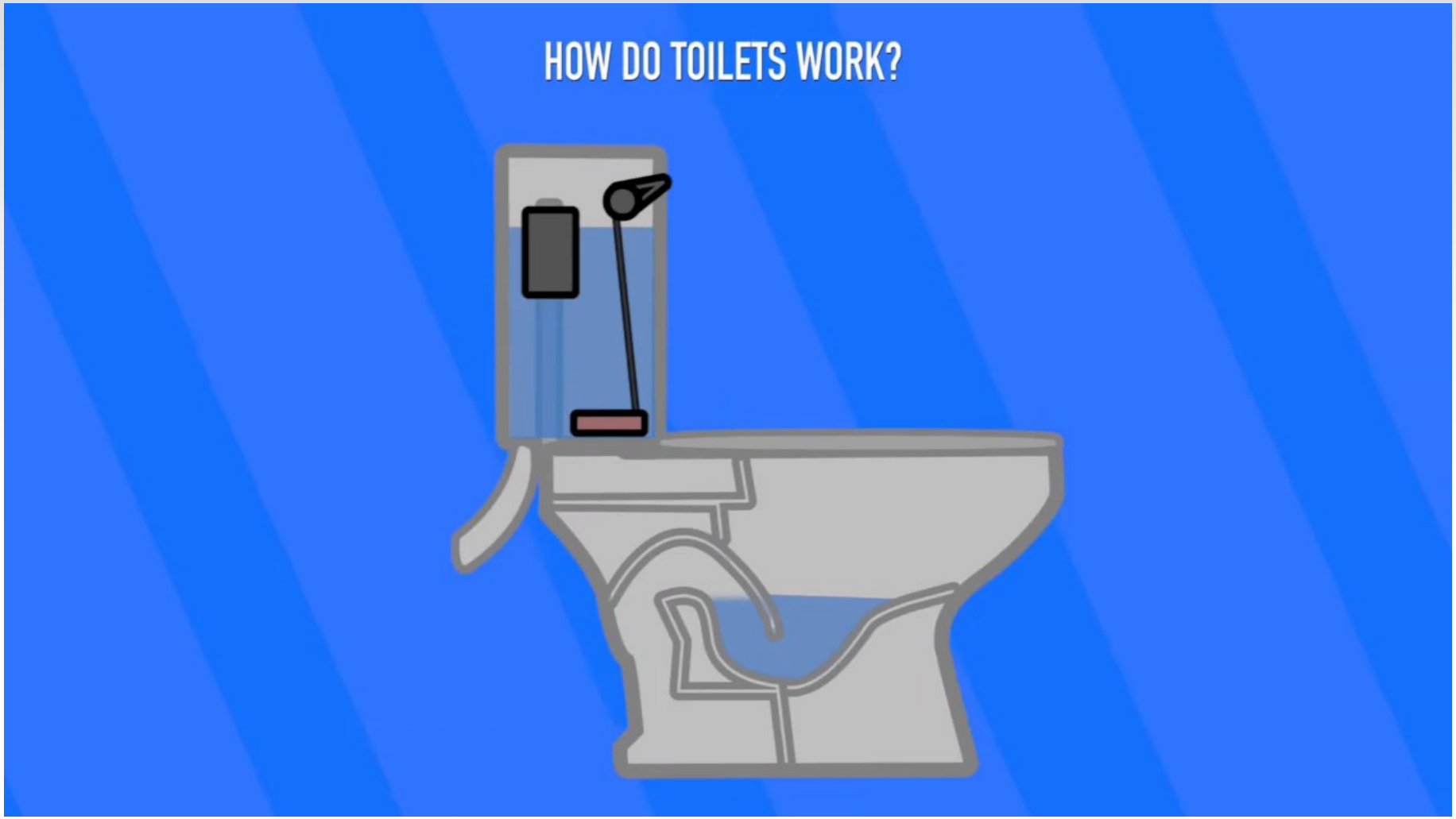


What About...

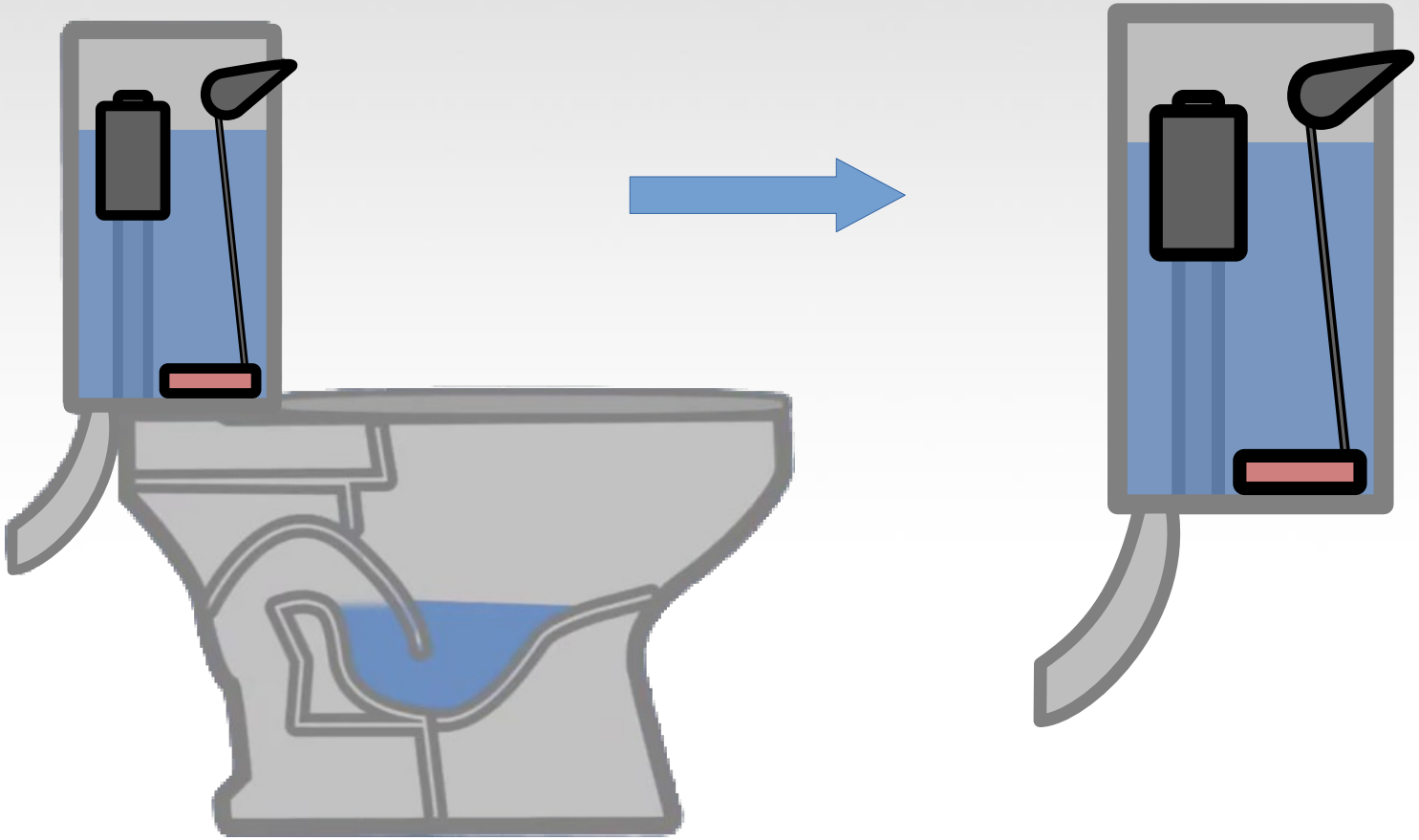


Changing Bathroom Fixtures?

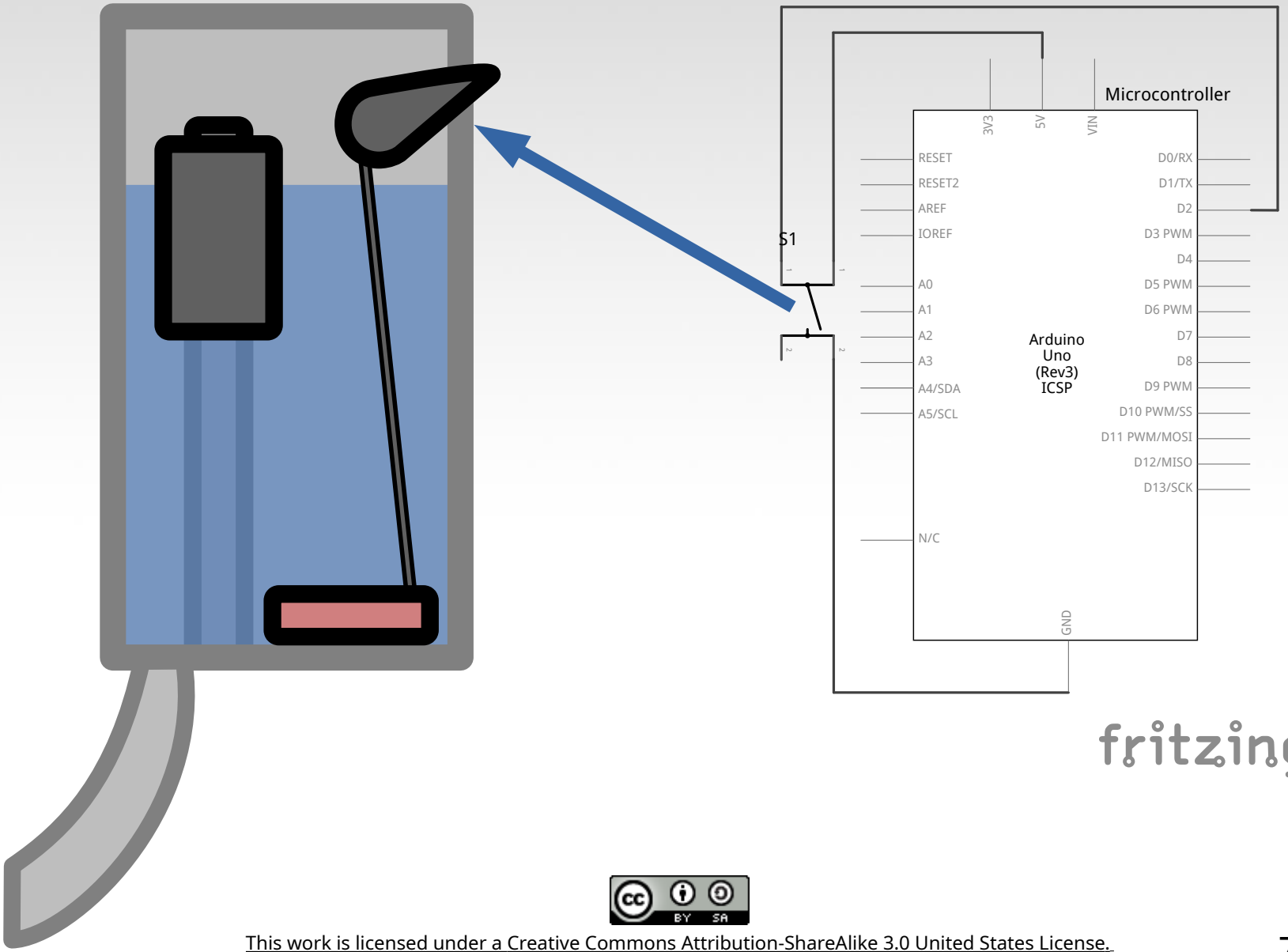




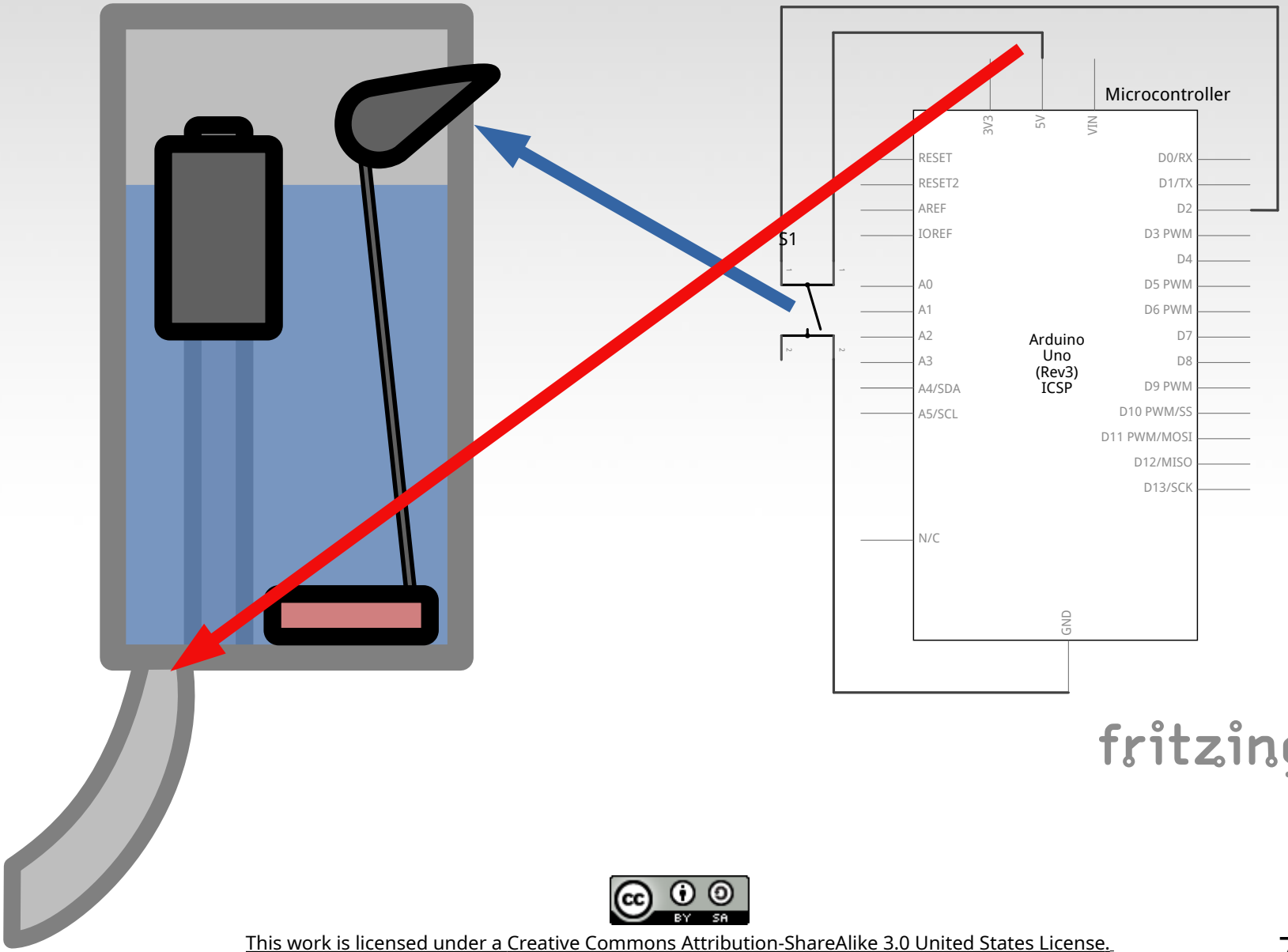
Getting Rid of the Unnecessary Crap



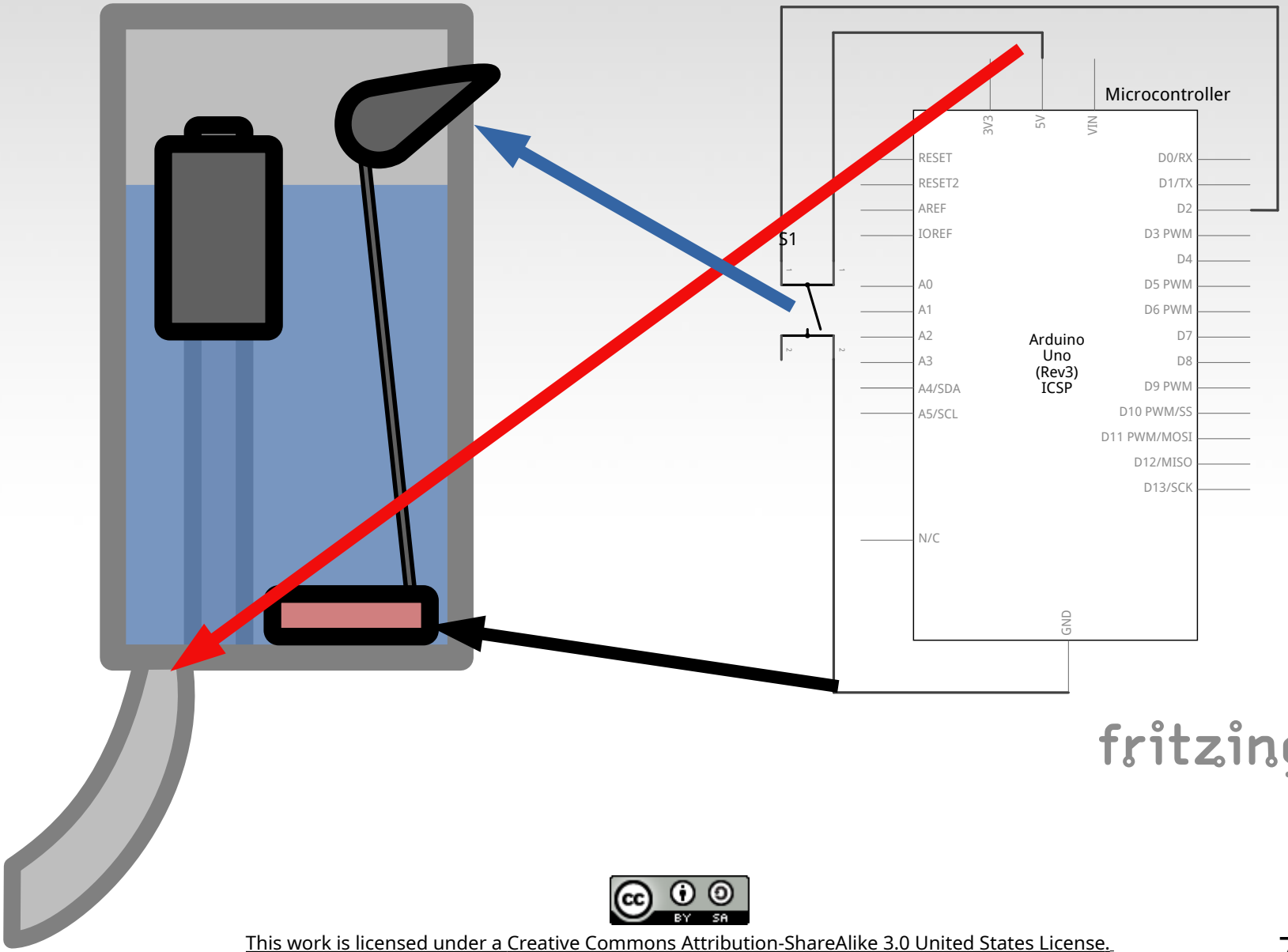
Changing the Viewpoint



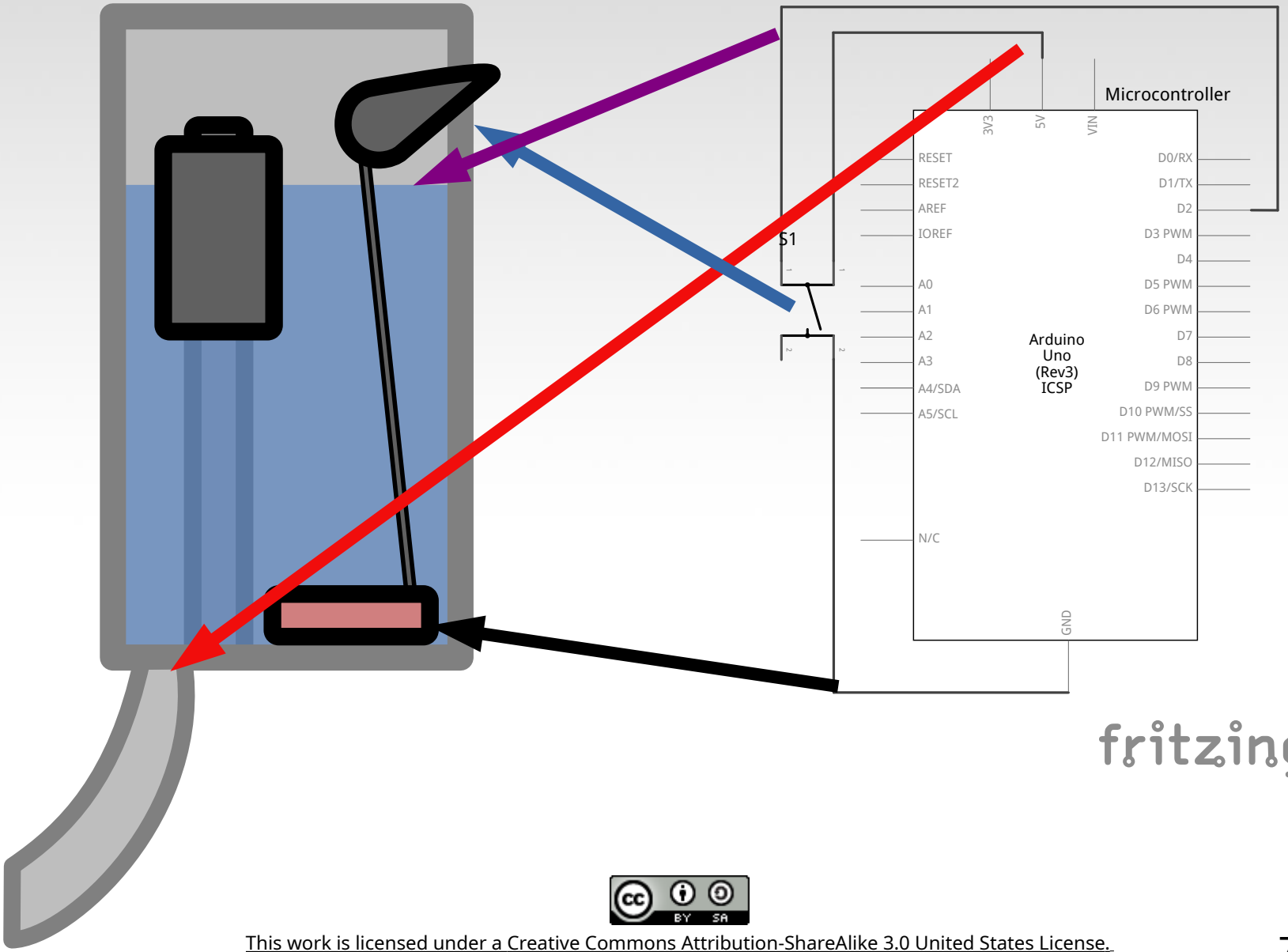
Changing the Viewpoint



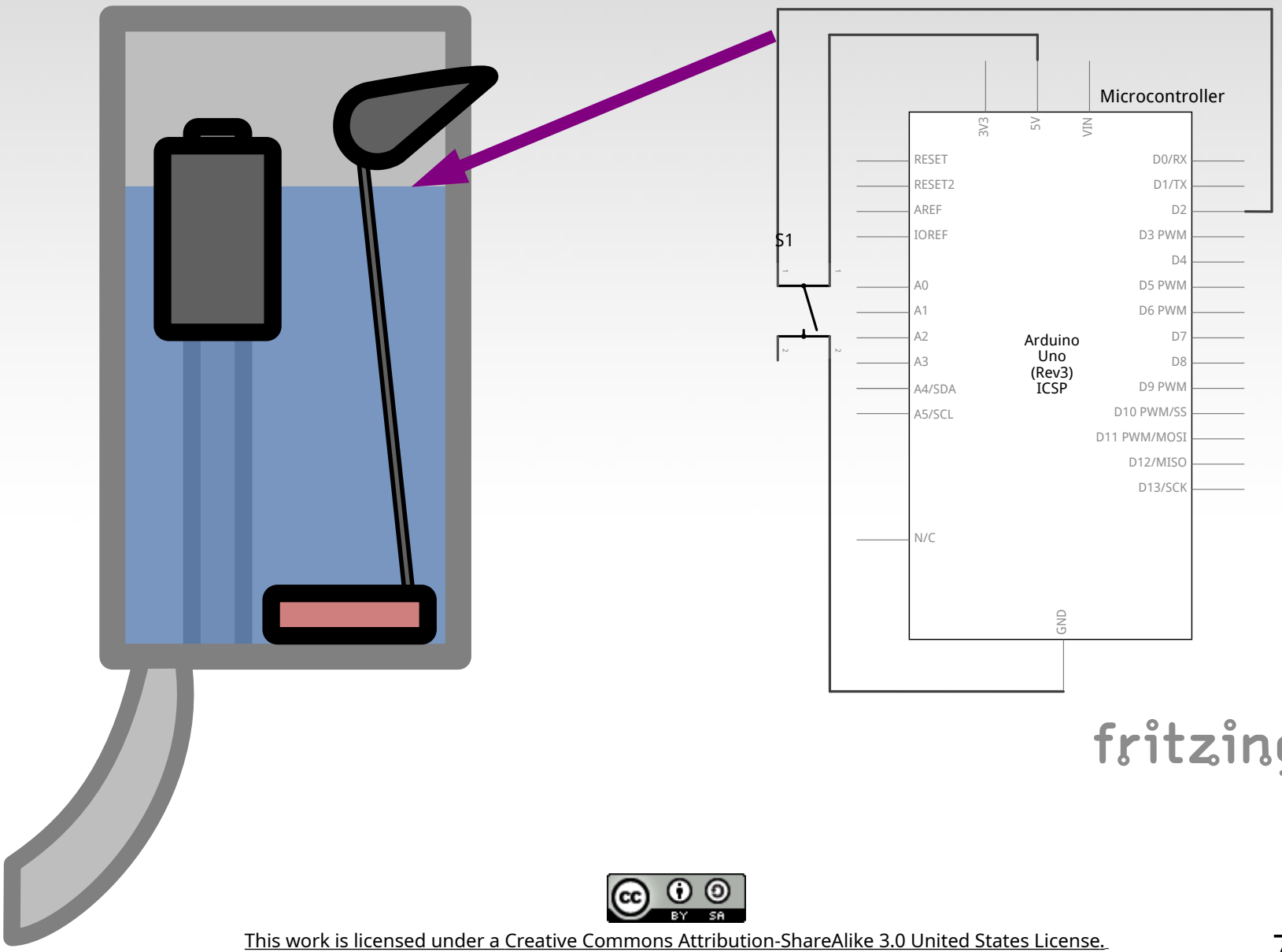
Changing the Viewpoint



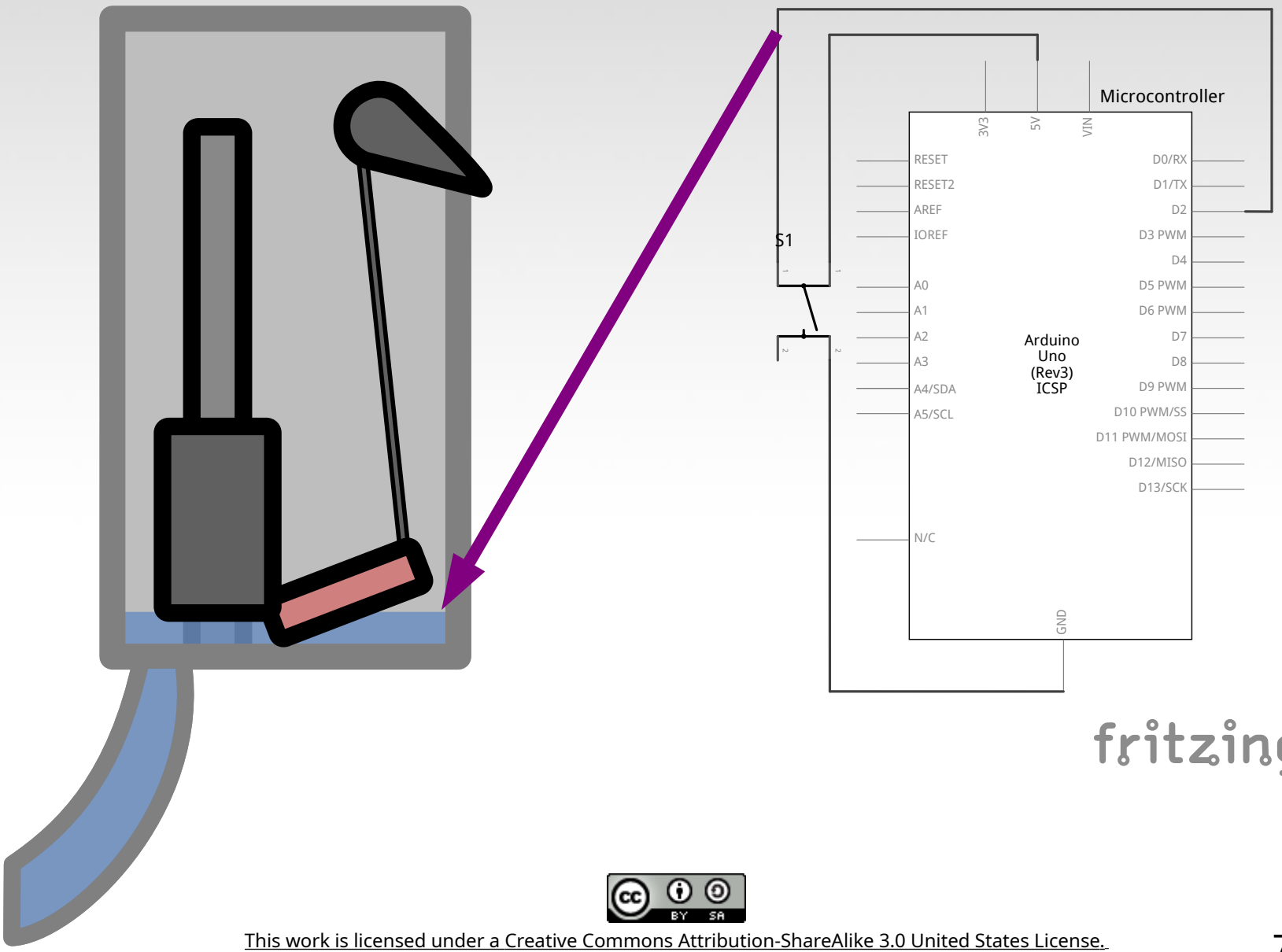
Changing the Viewpoint



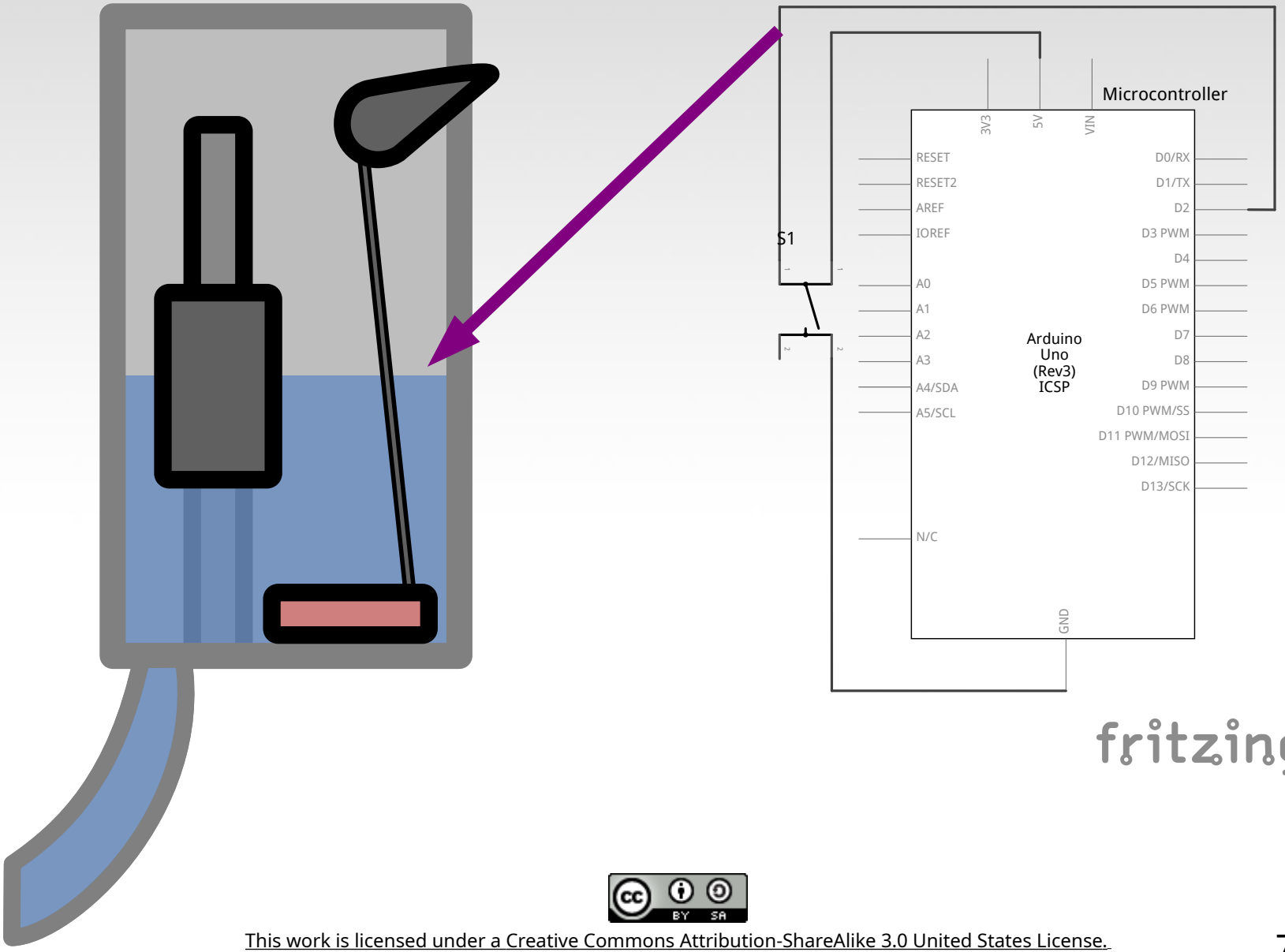
Switch Off



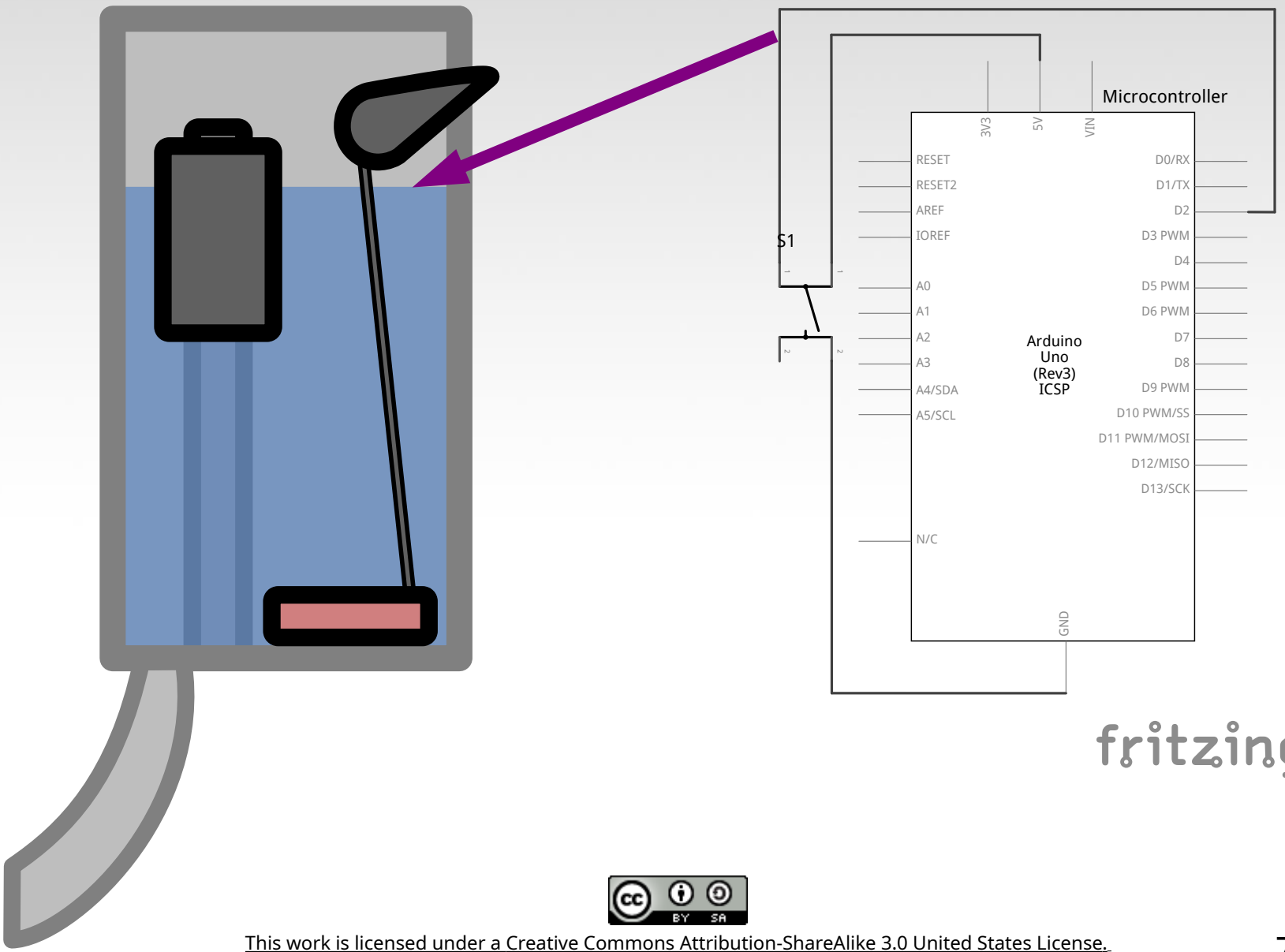
Switch On



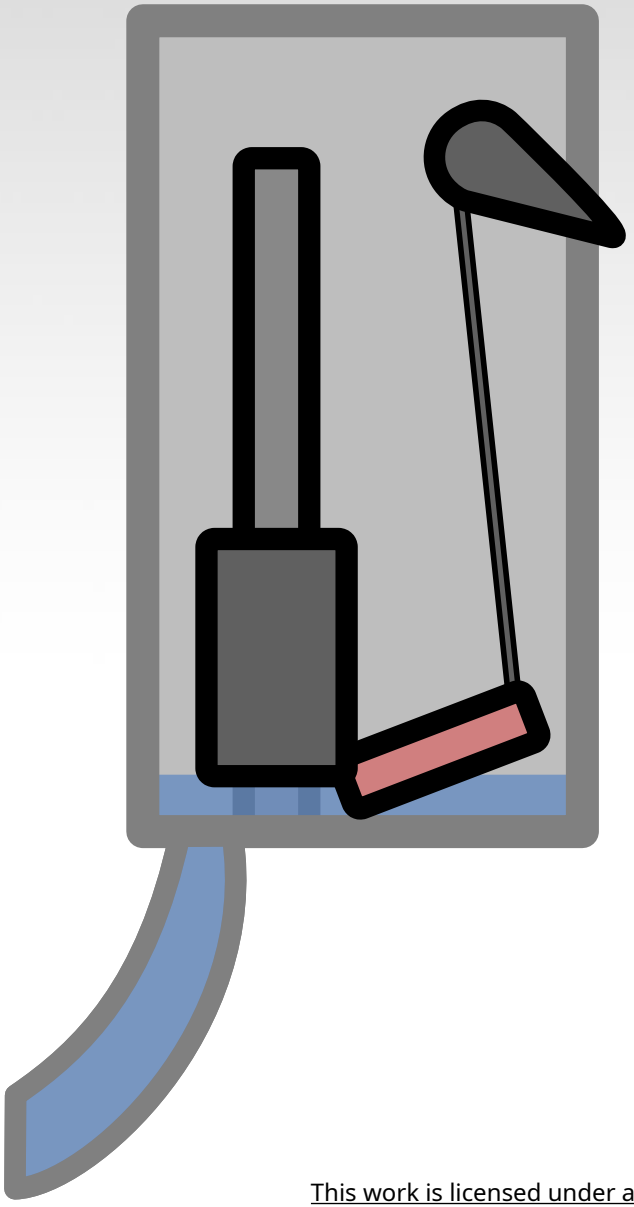
Switch Right After Off

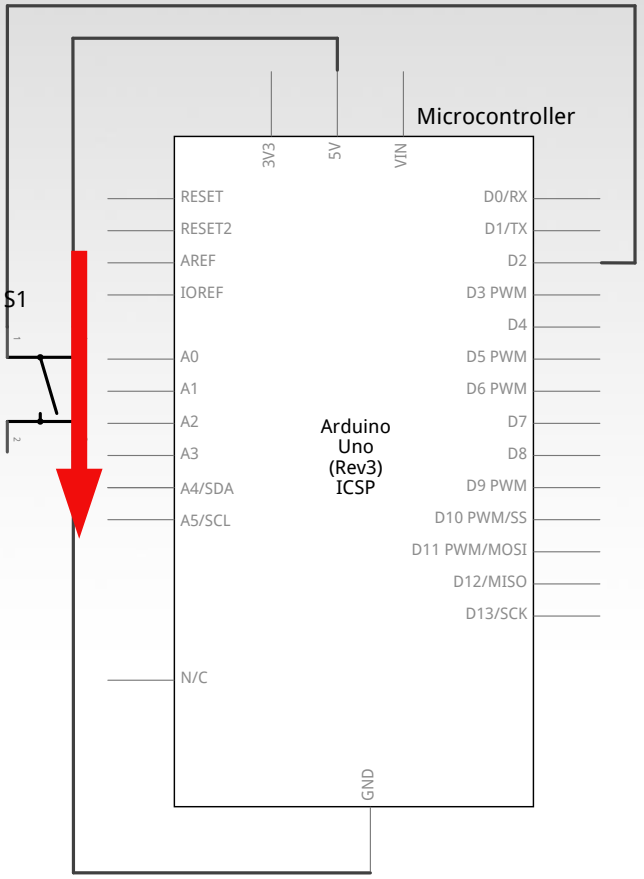


Switch Off

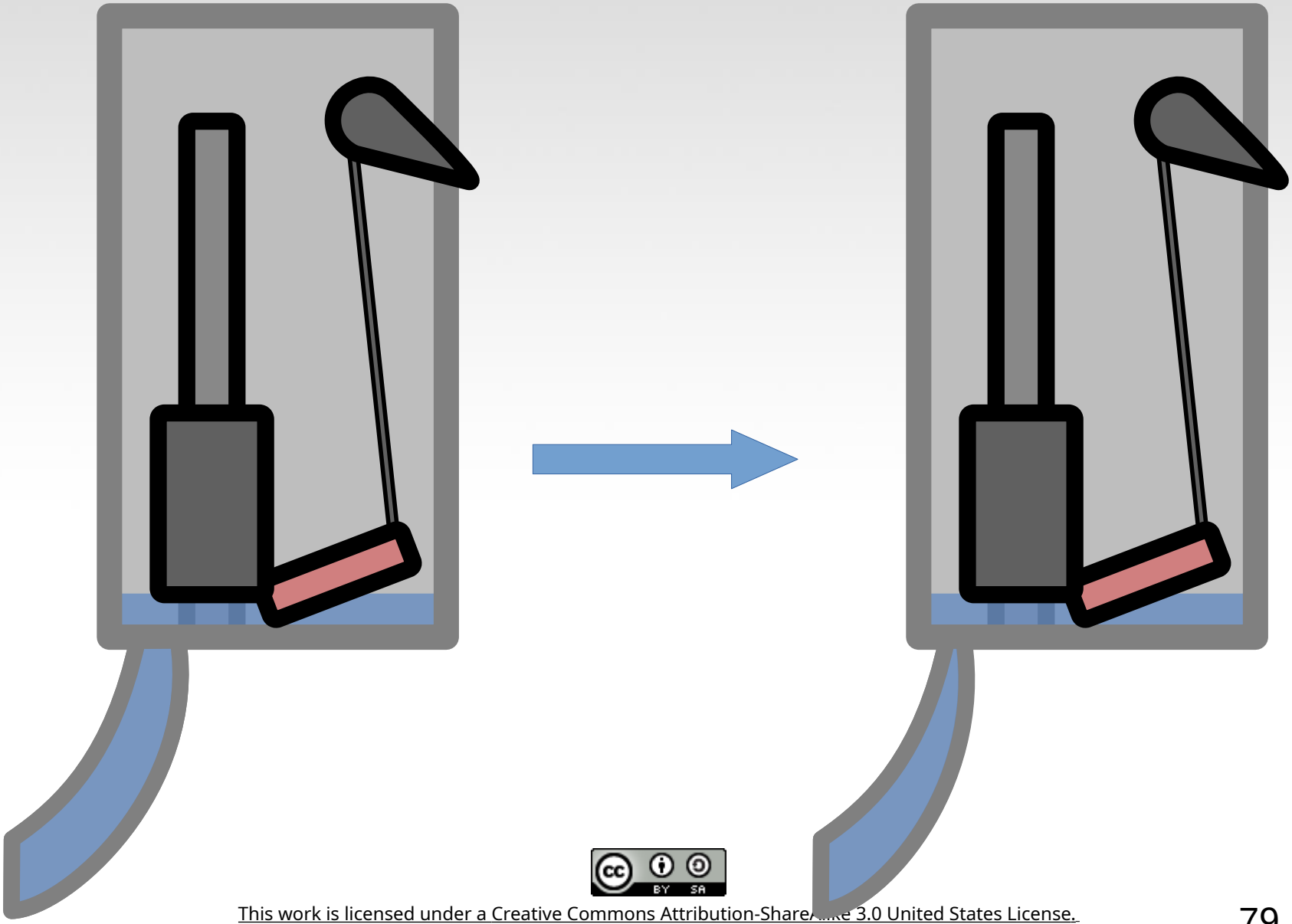


Stop Playin' With the Commode!!

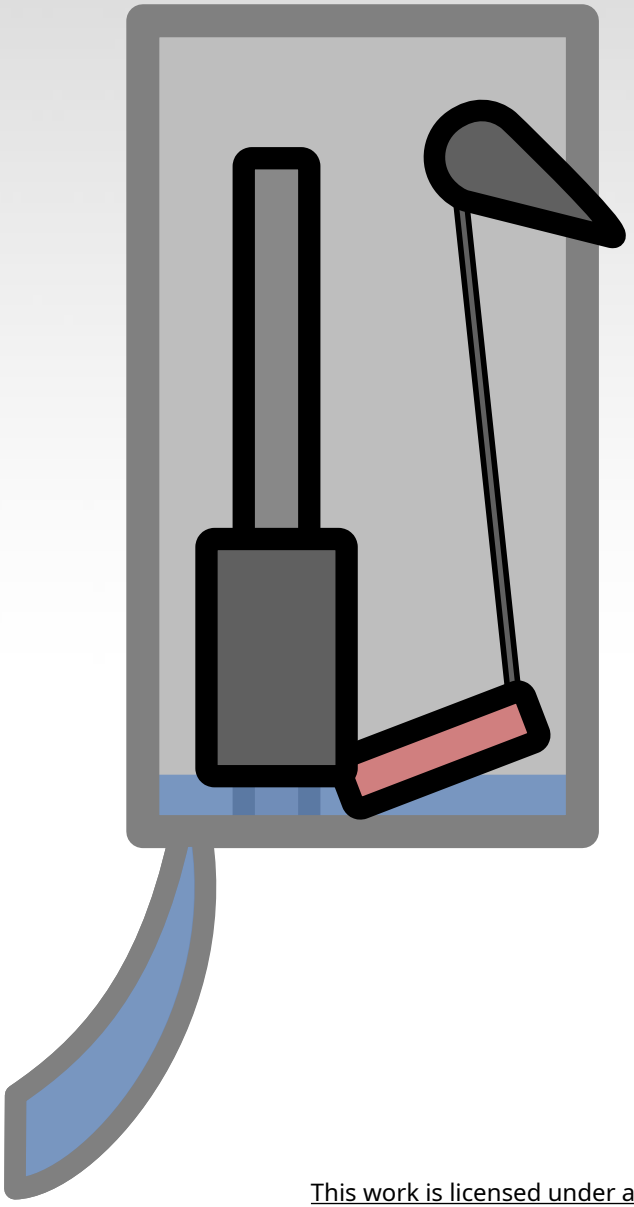


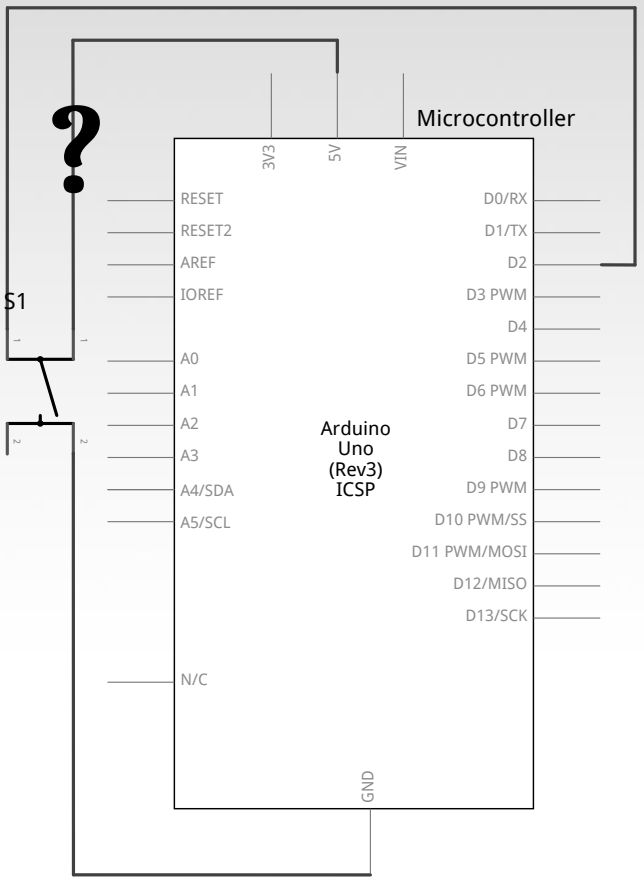


Possible Solution



How Do We Get Here?

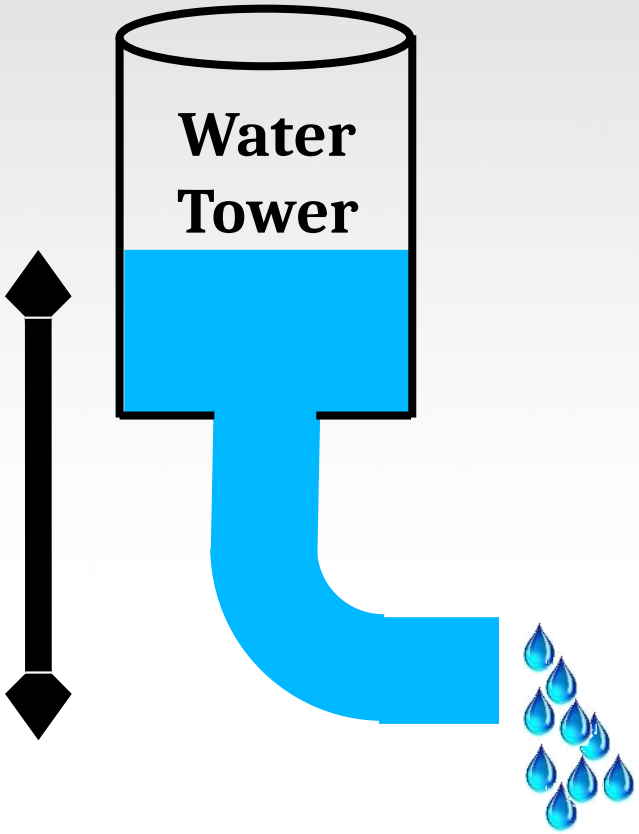




fritzing

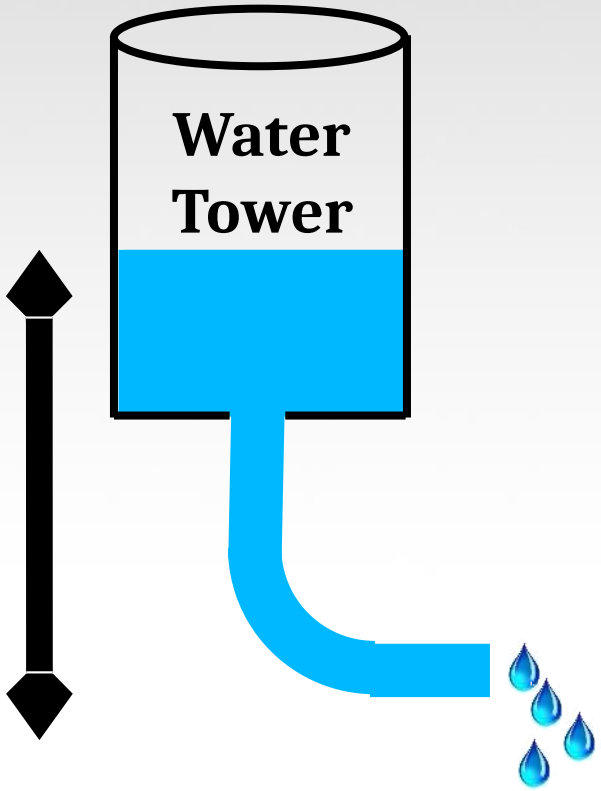
$$V = I R$$

Resistance Analogy



Big Pipe == Lower Resistance

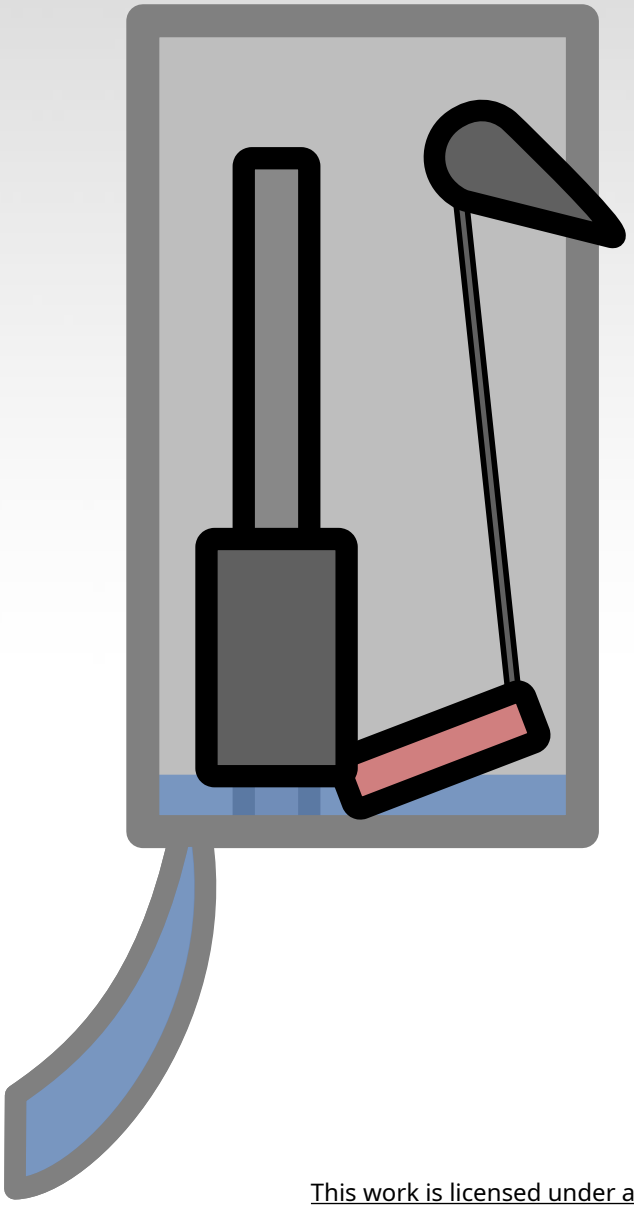
$$V = I R$$

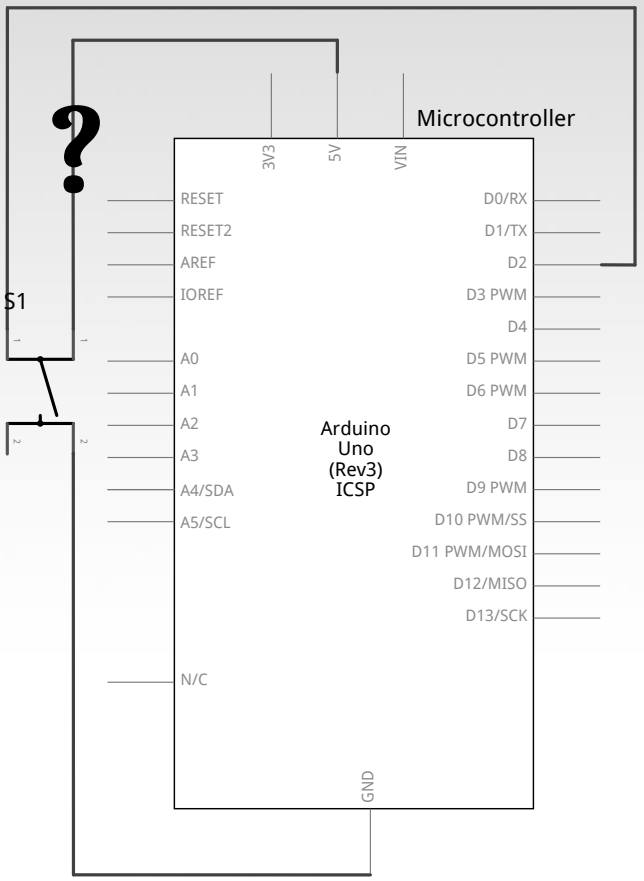


Small Pipe == Higher Resistance

$$V = I R$$

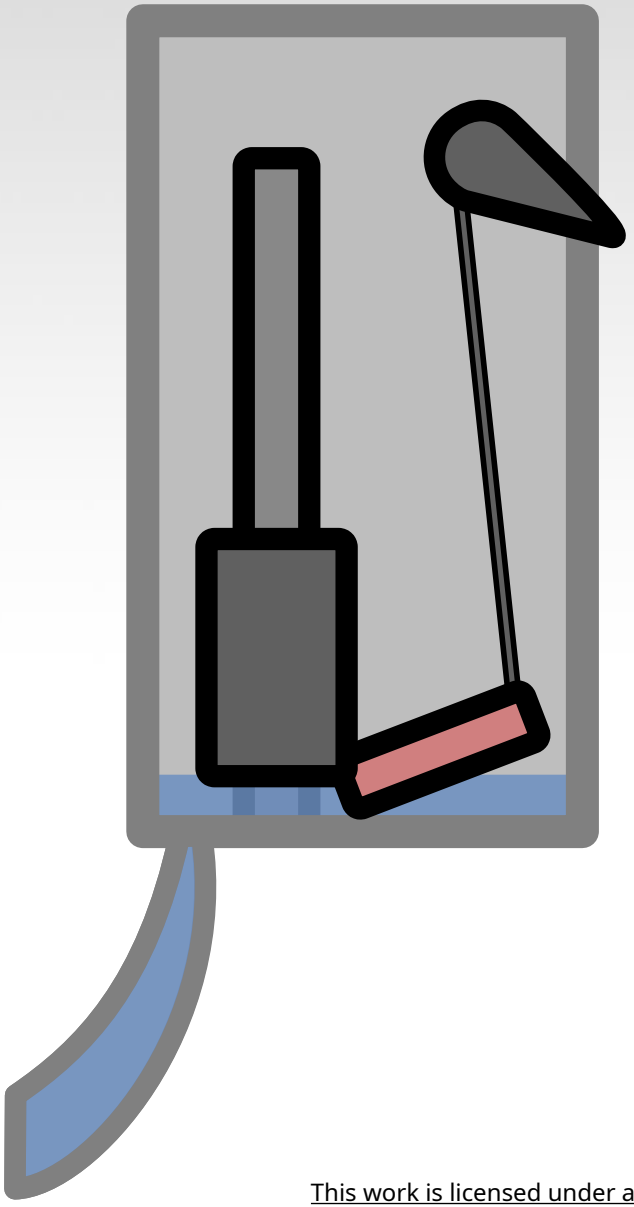
How Do We Get Here?

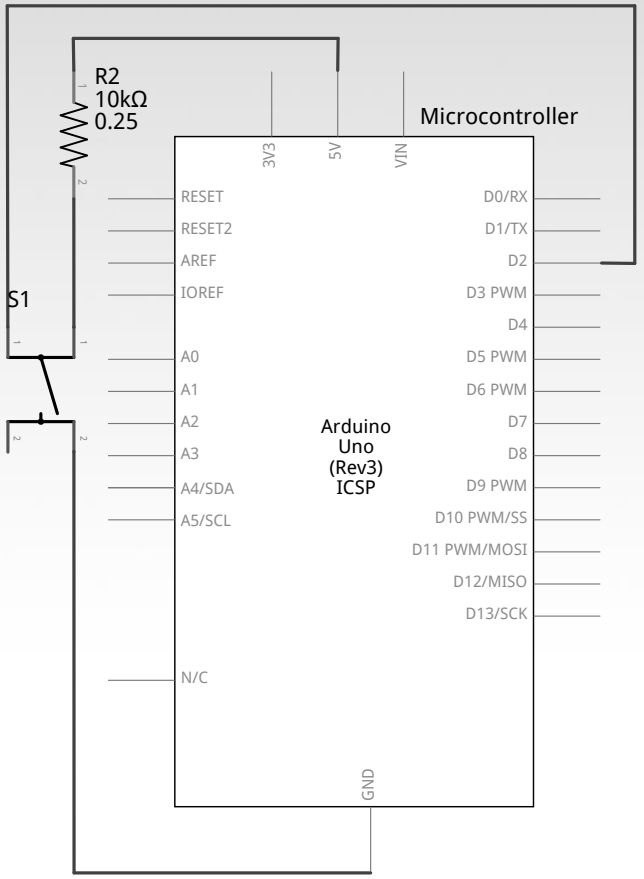




fritzing

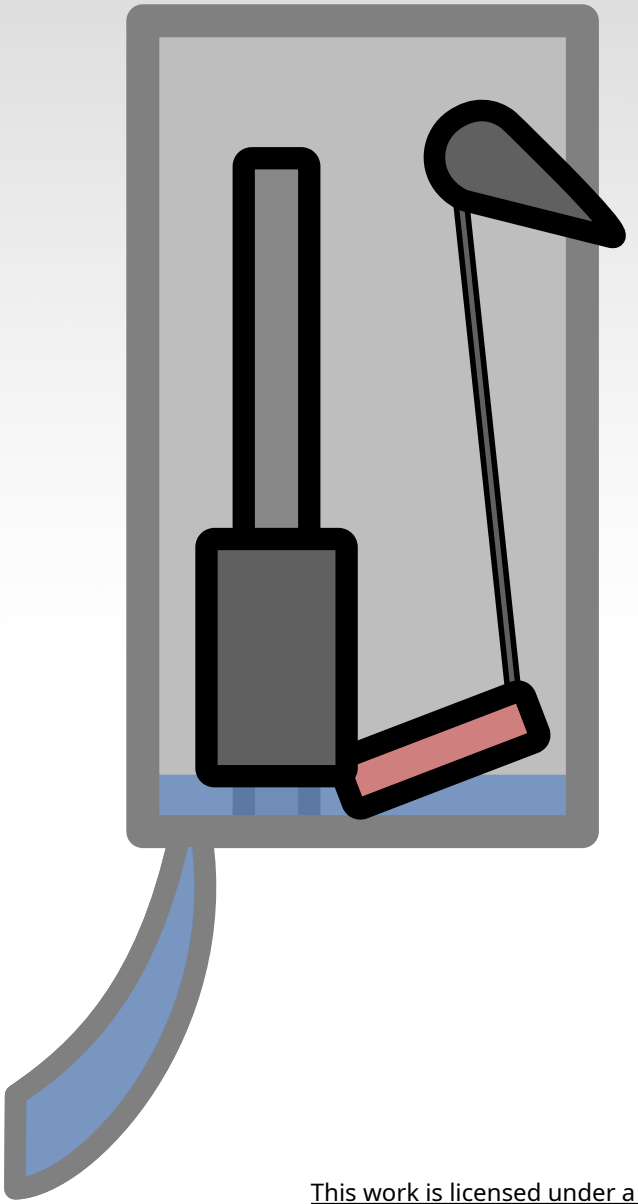
The Pull-Up Resistor!

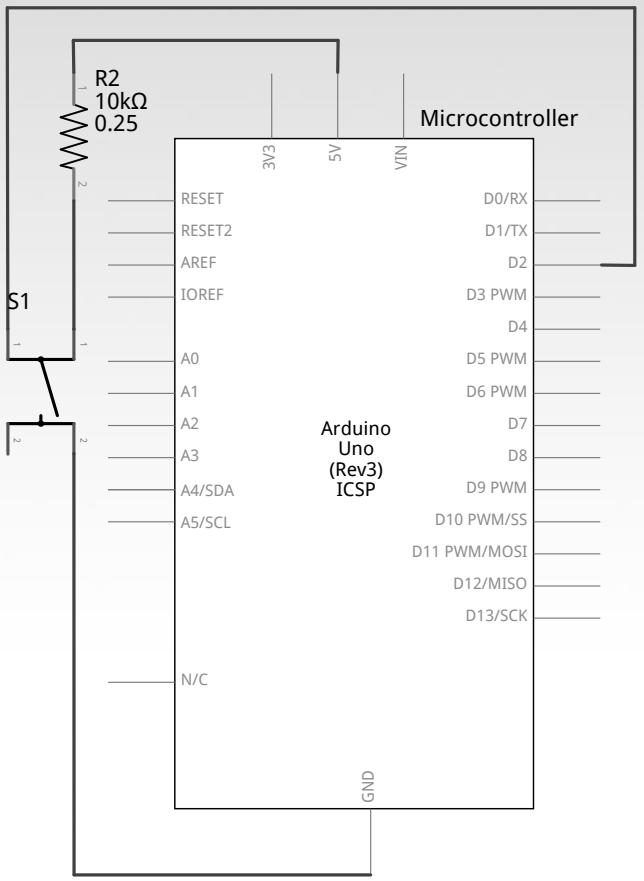




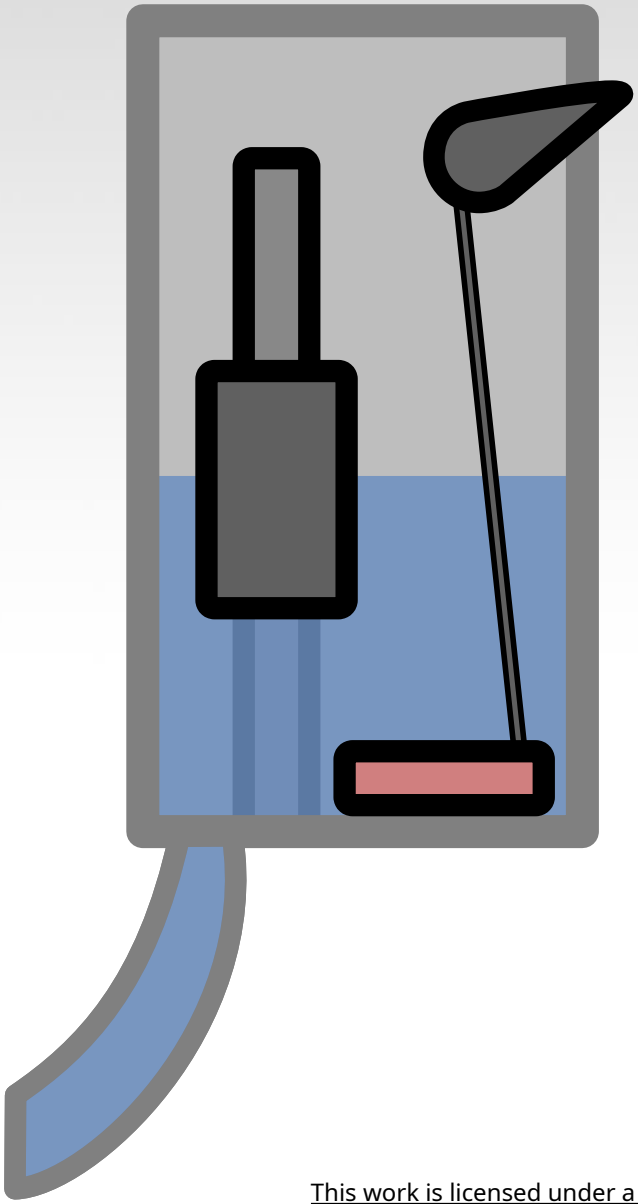
fritzing

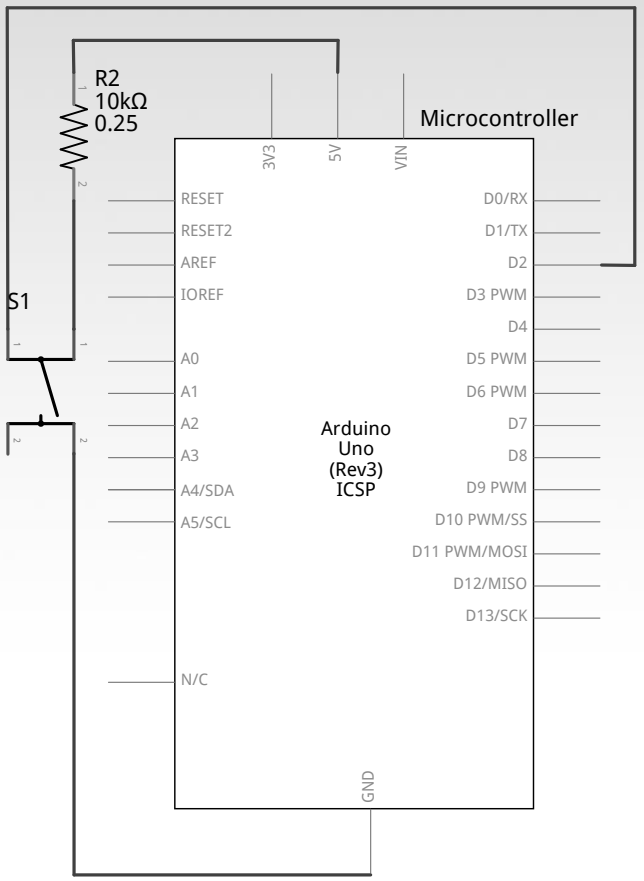
But There's A Trade-Off Between This...





...And This





fritzing

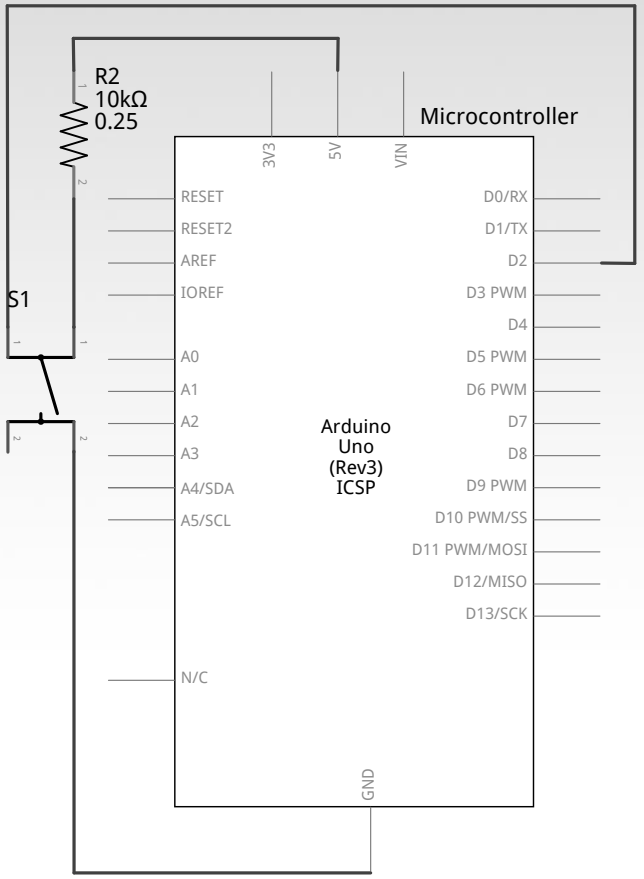
It's All About Compromise

10kΩ is a good value

(brown-black-black-red-gold)

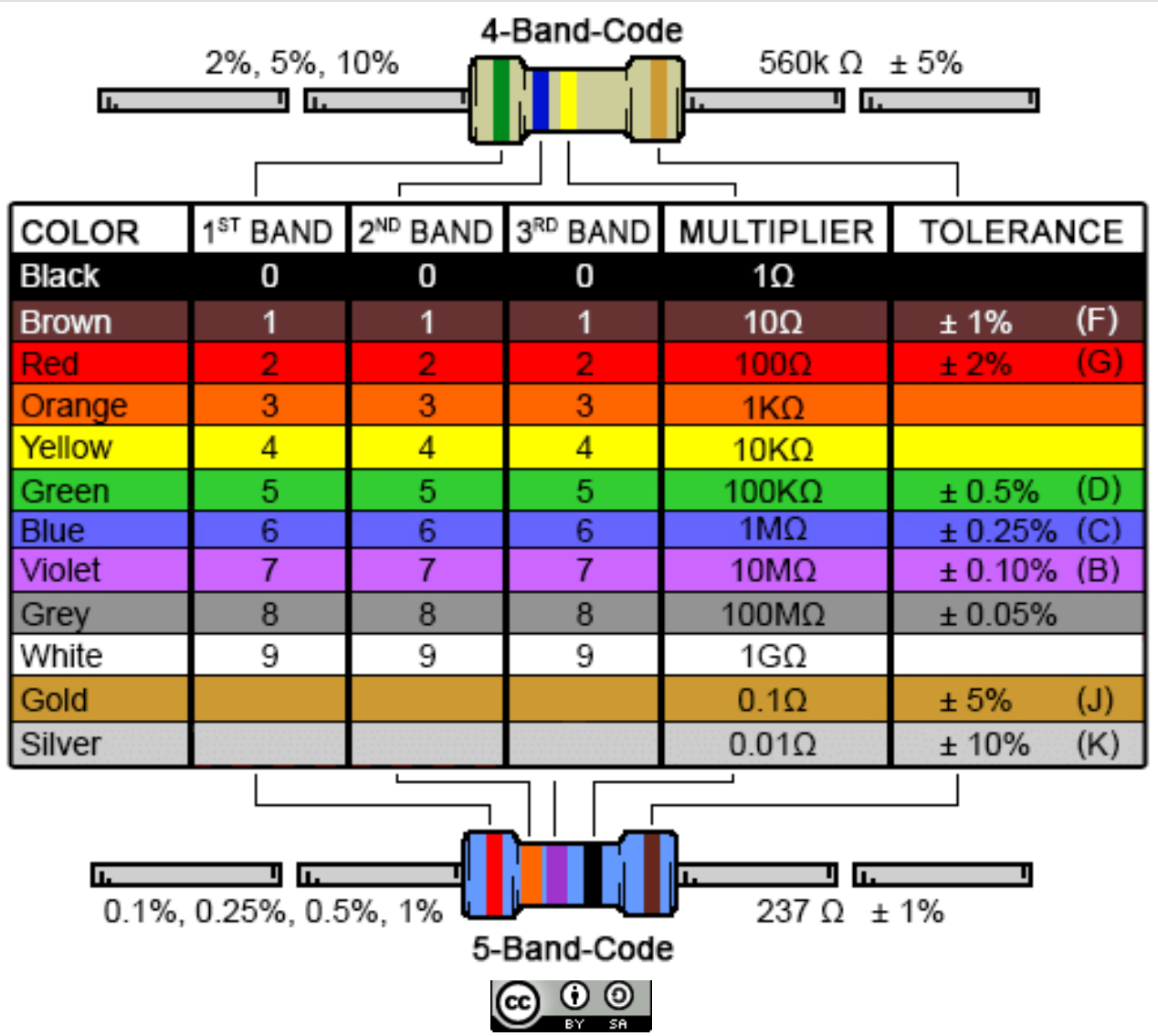
or

(brown-black-orange-gold)



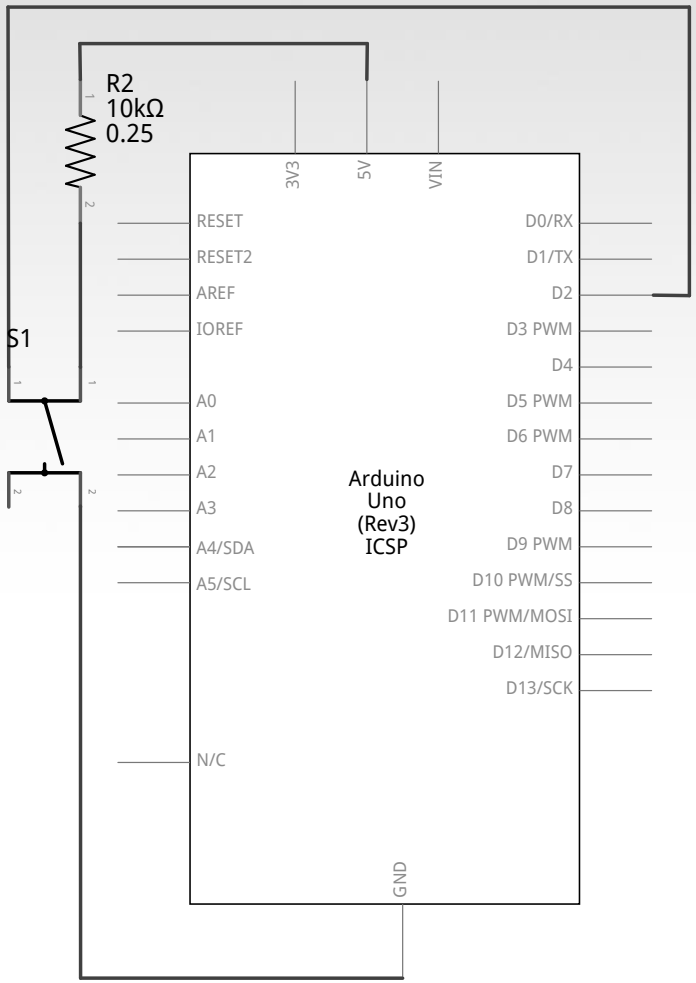
fritzing

Decoding Resistor Values



COLOR	1ST BAND	2ND BAND	3RD BAND	MULTIPLIER	TOLERANCE
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1% (F)
Red	2	2	2	100Ω	± 2% (G)
Orange	3	3	3	1KΩ	
Yellow	4	4	4	10KΩ	
Green	5	5	5	100KΩ	± 0.5% (D)
Blue	6	6	6	1MΩ	± 0.25% (C)
Violet	7	7	7	10MΩ	± 0.10% (B)
Grey	8	8	8	100MΩ	± 0.05%
White	9	9	9	1GΩ	
Gold				0.1Ω	± 5% (J)
Silver				0.01Ω	± 10% (K)

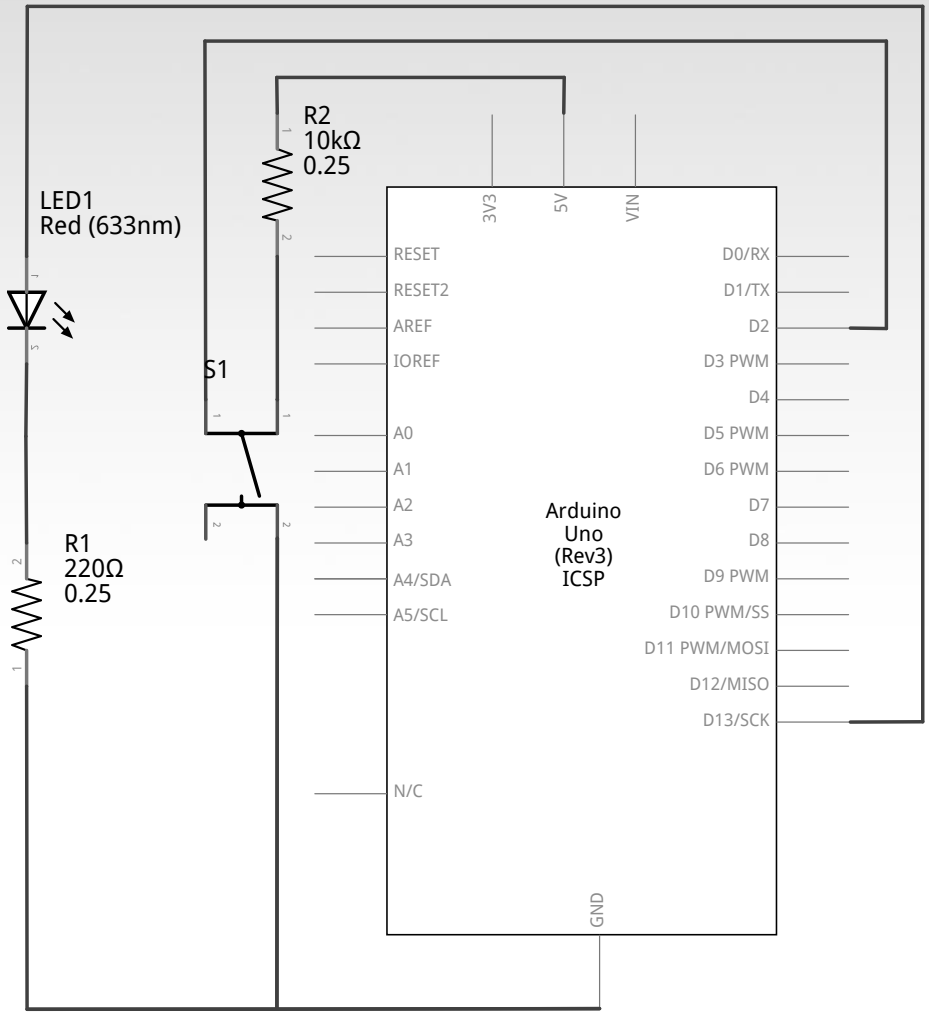
Project # 4 – Button/Maintained Switch Switch Input Schematic



Schematic of Pull-Up Resistor and Switch

fritzing

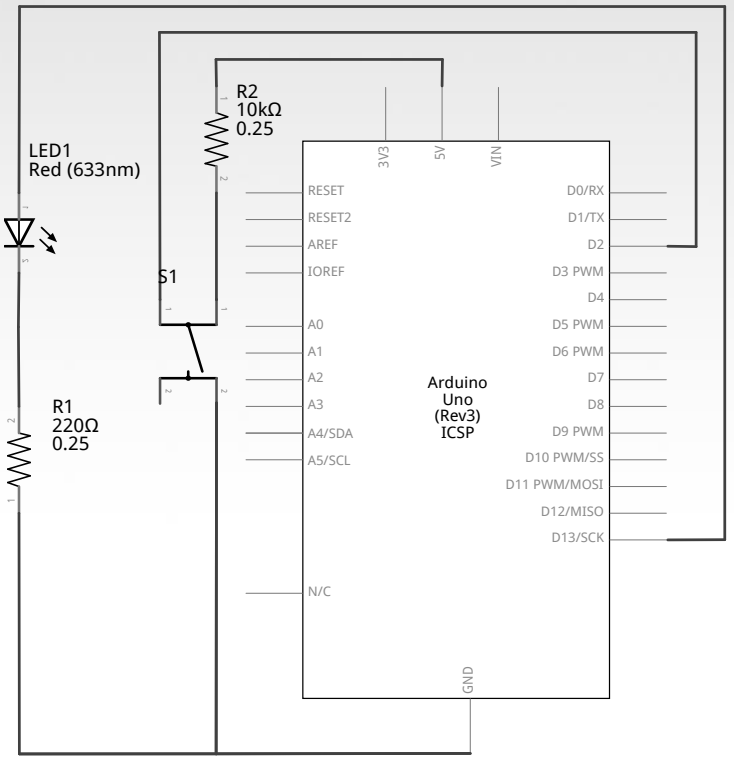
Project # 4 – Button/Maintained Switch Switch Full Schematic

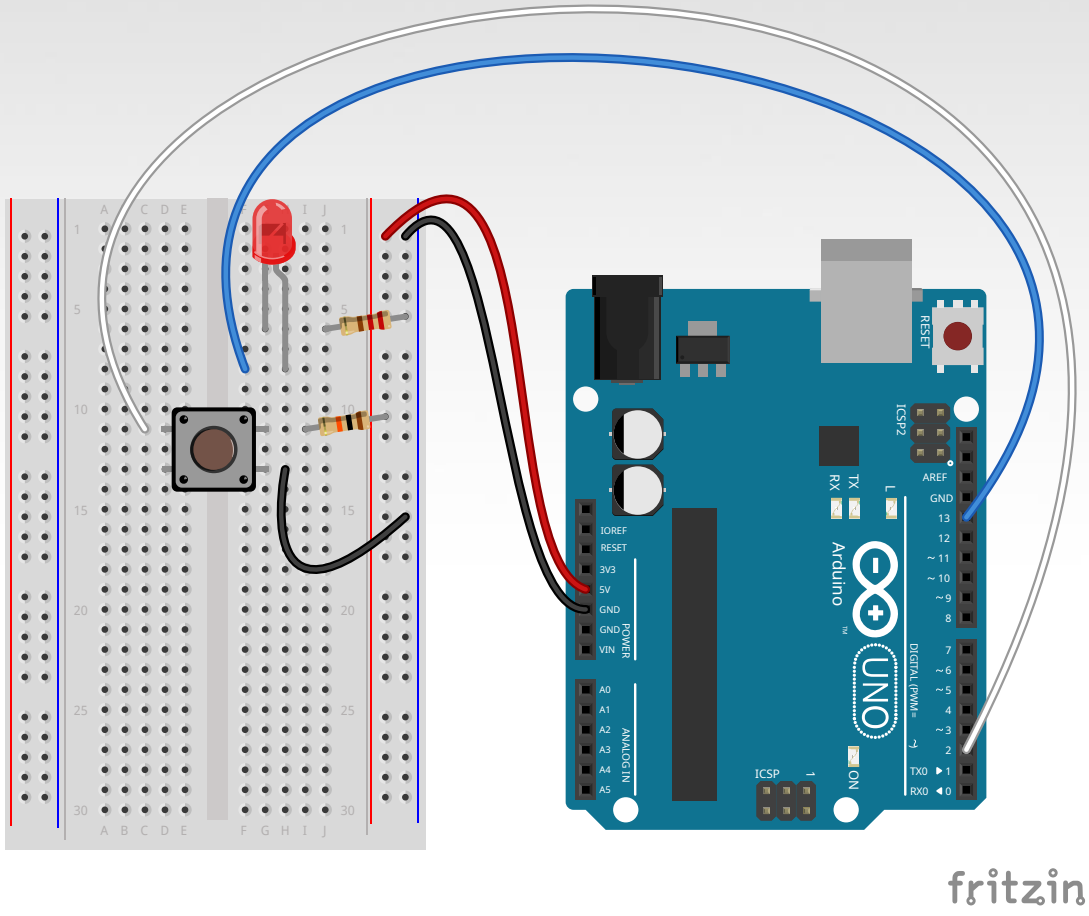


And the output is still the same as Project 1!

fritzing

Project # 4 – Button/Maintained Switch Switch Wiring Diagram





Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
```

Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
  ex: pinMode(2, INPUT);
```

Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
  ex: pinMode(2, INPUT);

digitalRead(pin);
```

Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
  ex: pinMode(2, INPUT);

digitalRead(pin);
  ex: digitalRead(2);
```

Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
  ex: pinMode(2, INPUT);

digitalRead(pin);
  ex: digitalRead(2);
  ex: int button_state = digitalRead(2);
```

Project # 4 – Button/Maintained Switch Switch
Our First Input Commands!

```
pinMode(pin, INPUT/OUTPUT);
  ex: pinMode(2, INPUT);

digitalRead(pin);
  ex: digitalRead(2);
  ex: int button_state = digitalRead(2);
```

button_state will either be HIGH or LOW

HIGH if switch is off, LOW if switch is on

Project # 4 – Button/Maintained Switch Switch Code Review

button_switch.ino

```
/*
 * make a push button behave like a switch
 */

// assign pins based off circuit
int button = 2;
int led = 13;

int button_val = HIGH;         // the button starts off not pressed
int last_button_val = HIGH;    // and wasn't pressed before the sketch started
int pressed = false;           // will tell us if the button was just pressed
int led_val = LOW;             // LED starts turned off

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize pin 2 as an input and pin 13 as an output
  pinMode(button, INPUT);
  pinMode(led, OUTPUT);
}

```

Code Review Continued

button_switch.ino

```
// the loop function runs over and over again forever
void loop() {
  button_val = digitalRead(button);                    // read the value of pin 2

  if(button_val == LOW && last_button_val == HIGH) {   // if the button is pressed but
    pressed = true;                                    // wasn't pressed last check,
  }                                                    // set the variable 'pressed' to true
  else {
    pressed = false;                                   // otherwise set it to false
  }

  if(pressed) {                                        // if 'pressed' is true
    if(led_val == LOW) {                               // and led is off,
      led_val = HIGH;                                  // turn led on
    }                                                  // if 'pressed is true
    else {                                             // and led is on,
      led_val = LOW;                                   // turn led off
    }
  }

  digitalWrite(led, led_val);                          // write the LED state to the LED

  if(button_val != last_button_val) {                  // if the button has been pressed or
    delay(50);                                         // released, give the circuit time
  }                                                    // to settle

  last_button_val = button_val;                        // update last_button_val for next loop
}
```

Code Review Continued

button_switch.ino

```
/*
 * make a push button beha
 */

// assign pins based off c
int button = 2;
int led = 13;

int button_val = HIGH;
int last_button_val = HIGH
int pressed = false;
int led_val = LOW;

// the setup function runs
void setup() {
  // initialize pin 2 as a
  pinMode(button, INPUT);
  pinMode(led, OUTPUT);
}

```

button_switch.ino

```
// the loop function runs over and over again forever
void loop() {
  button_val = digitalRead(button);

  if(button_val == LOW && last_button_val == HIGH) {
    pressed = true;
  }
  else {
    pressed = false;
  }

  if(pressed) {
    if(led_val == LOW) {
      led_val = HIGH;
    }
    else {
      led_val = LOW;
    }
  }

  digitalWrite(led, led_val);

  if(button_val != last_button_val) {
    delay(50);
  }

  last_button_val = button_val;
}
```

Project # 4 – Button/Maintained Switch Switch Puzzles

Challenge 4a – Wire another LED. Make the button turn one off and the other on

Challenge 4b – Make an LED blink when you press the button (hint: look at File>Examples>Digital>Blink Without Delay)

Challenge 4c – Make an LED alternate between off, on and blink by pressing the button

Special Thanks:

roto

SenseICs