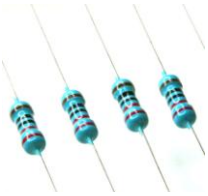
Let's play Global Thermonuclear War.

Parts:

1 x Pushbutton switch



2 x 220 Ohm Resistor
1 x 10k Ohm Resistor

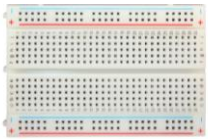
1 x USB Micro Cable

1 X Piezo Buzzer

8 x Jumper Wires

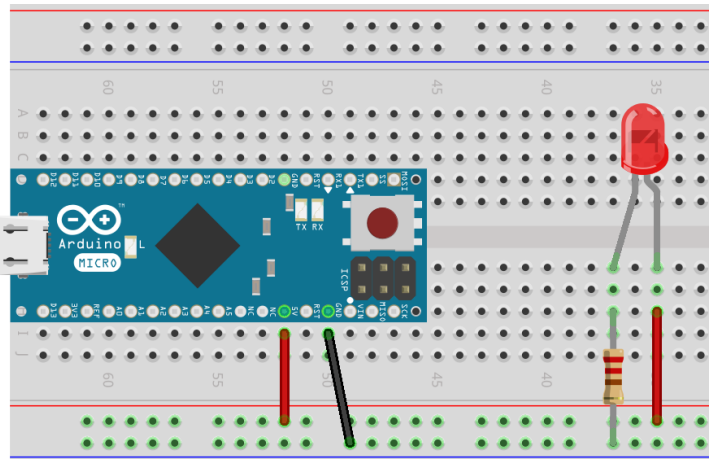
1 x Arduino Micro

1 x Red LED
1 x Green LED



1 x Breadboard

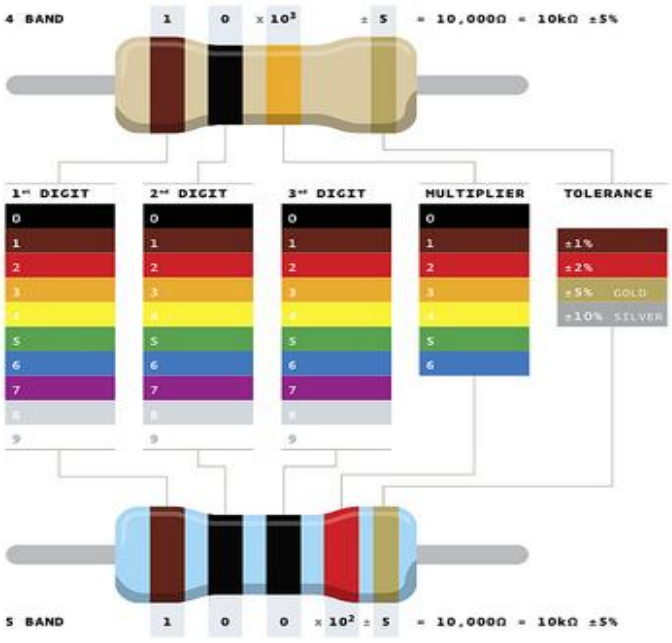
A Simple Circuit



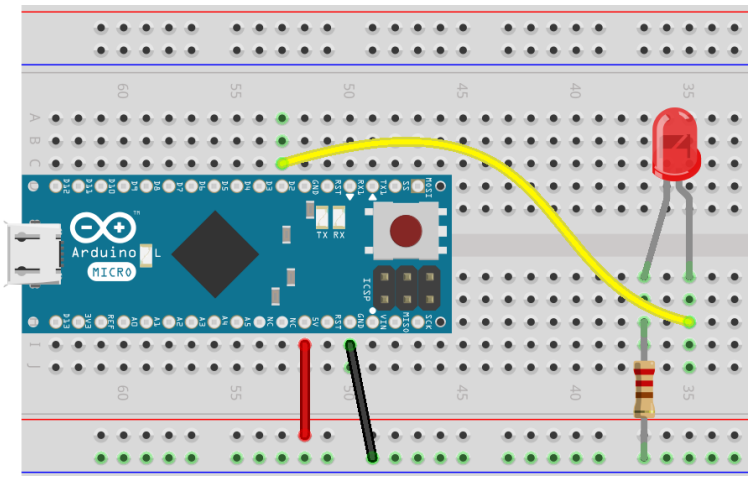
1. Connect the 5V pin on the Arduino (line 52) to the red power rail on the breadboard and the ground pin (line 50) to the blue ground rail
2. Connect the red LED to the breadboard, making sure the long leg, or anode, is furthest away from the Arduino
3. Connect the anode (long) end of the LED to the power rail and the cathode (short) end of the LED to the ground rail using a 220 Ohm resistor.
4. Plug the Arduino into the computer and you have a simple circuit! The light should be on.

What resistor is that?

In order to tell how many ohms a resistor is for your projects, just look at the colours. You might have a 4 band or a 5 band resistor. The chart to the right should help. Also, there are many resistor calculators online.



Make the Light Blink with Code



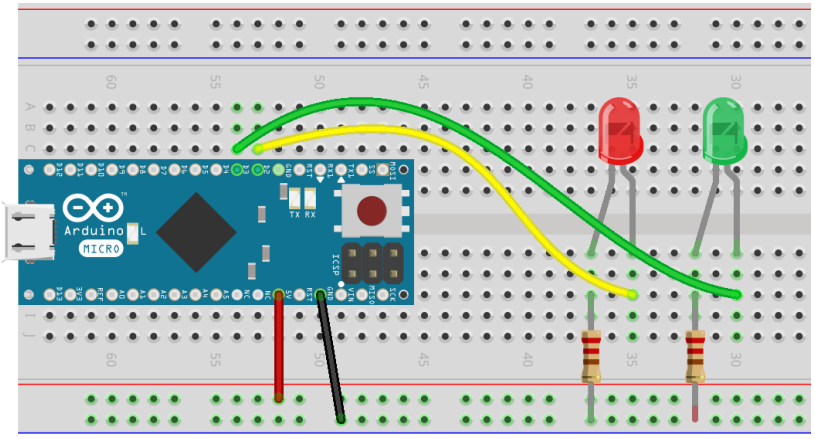
1. Disconnect the wire connected to the power rail and re-connect it to digital pin 2 on the Arduino (line 53).
2. Open up the Arduino software on your computer to open up a new sketch. Sketches are programs that we write to make the Arduino do stuff to our components.
3. Write the Following code:

```
void setup(){
  pinMode(2,OUTPUT);
}

void loop(){
  digitalWrite(2,LOW);
  delay(1000);
  digitalWrite(2,HIGH);
  delay(1000);
}
```

```
// Red LED attached
// to pin 2

// Red LED off
// Wait a second
// Red LED on
// Wait a second
```

4. Verify the code by clicking the tick mark
5. Connect your board
6. Go to Tools > Board and select the Arduino Micro
7. Go to Tools > Port and select anything but COM1. If you don't see another COM number, your board isn't connected properly
8. Upload the code using the right-pointing arrow
9. The light should now blink on and off

Add another Light



1. Connect the green LED into the breadboard, paying attention to the long and short legs
2. Connect the anode of the green LED to digital pin 3 (line 54) and the cathode into the ground rail with a 220 Ohm resistor
3. Make the following additions to the code you wrote last time:

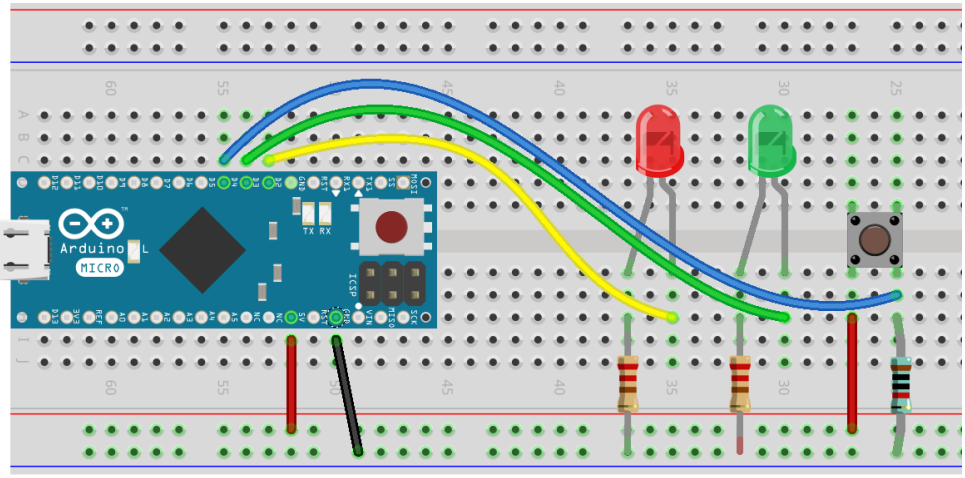
```
void setup(){
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
}
void loop(){
  digitalWrite(2,LOW);
  digitalWrite(3,HIGH);
  delay(1000);
  digitalWrite(2,HIGH);
  digitalWrite(3,LOW);
  delay(1000);
}
```

```
// Red LED attached
// to pin 2, Green
// LED to pin 3

// Red LED off
// Green LED on
// Wait a second
// Red LED on
// Green LED off
// Wait a second
```

4. Verify the code by clicking the tick mark
5. Connect your board
6. Upload the code using the right-pointing arrow
7. The lights should alternate on and off one second apart.

Can you make a more interesting pattern? What happens when you change the delays around? What about switching around the LOW and HIGH for the LEDs?

Add a Button



A button is a kind of sensor. When the button is pressed, it completes our circuit. When it is not pressed, the circuit is open.

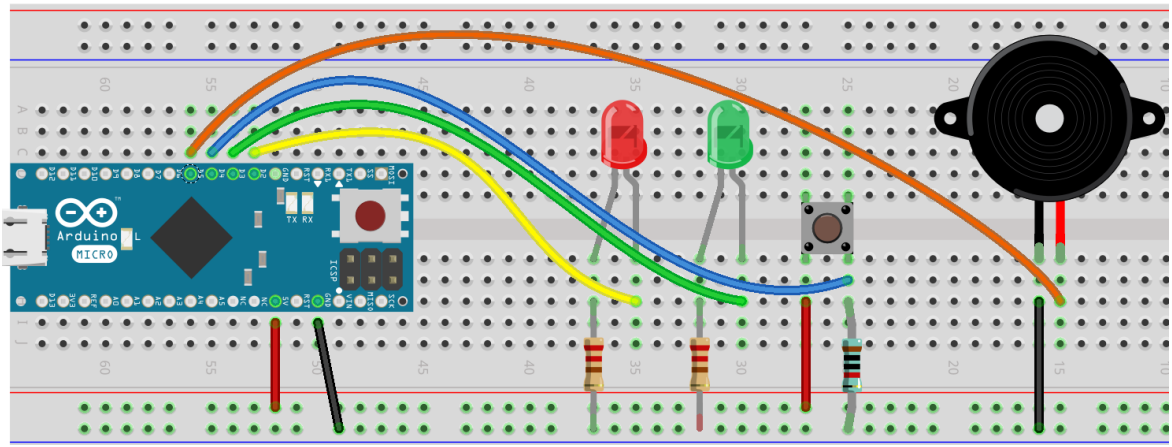
1. Put the button in the middle of the breadboard
2. Wire up one side of the button to power and the other side to ground with a 10K Ohm resistor.
3. Attach a wire from the ground side of the button, just before the resistor, to digital pin 4 (line 55)

4. Now that we have a switch, let's code it to do something. Re-write your code to look like the code to the right
5. Verify the code by clicking the tick mark
6. Connect your board
7. Upload the code using the right-pointing arrow
8. The green LED should be on if the button is not being pushed. When the button is pushed, it should turn on the red LED and the green LED should switch off

```
int switchState = 0;

void setup(){
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,INPUT);
}
void loop(){
  switchState = digitalRead(4);
  if (switchState == LOW) {
    digitalWrite(2,LOW);
    digitalWrite(3,HIGH);
  }
  else {
    digitalWrite(2,HIGH);
    digitalWrite(3,LOW);
  }
}
```

```
// Declare the variable for the
// button's state. Int stands for
// integer

// The button on pin 4 is an input


// Here we are saying that the
// variable, switchState, is
// reading digital pin 4. If the
// reading shows that the button
// is not being pressed, the red
// LED is off and the green light
// is on. Otherwise, if the button
// is pressed, the red LED light
// is on and the green light is
// off.
```

Let's add sound



1. Put the piezo to the right of the rest of the circuit
2. Wire the negative side (black) to the ground rail and the positive side (red) to digital pin 5 (line 56)
3. Add the following code to your previous code:

```
int switchState = 0;
int speakerPin = 5;

void setup(){
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,INPUT);
  pinMode(5,OUTPUT);
}
void loop(){
  switchState = digitalRead(4);
  if (switchState == LOW) {
    digitalWrite(2,LOW);
    digitalWrite(3,HIGH);
  }
  else {
    digitalWrite(2,HIGH);
    digitalWrite(3,LOW);
    tone(speakerPin, 50, 500);
    delay(1000);
    tone(speakerPin, 50, 500);
    delay(1000);
  }
}
```

```
// All we are adding is the
// variable speakerPin, which
// states that the piezo is on
// digital pin 5

// Here we are adding that digital
// pin 5 is an output

// Here we are adding the tone
// to the program. The tone is
// on digital pin 5, it is
// going to be at 50 hertz
// (frequency), and it will
// last for half a second (500
// milliseconds). This will
// repeat until the button is
// released
```

4. Verify the code by clicking the tick mark
5. Connect your board
6. Upload the code using the right-pointing arrow
7. When you push the button, the red light should turn on and the piezo should sound an alarm
8. Try changing the frequency. Can you make a tune?

C is 261 Hz
D is 294 Hz
E is 329 Hz
F is 349 Hz
G is 392 Hz
A is 440 Hz
B is 493 Hz
C is 523 Hz

Mary had a little lamb is
AGFGAAA
GGGACC
AGFGAAA
AGGAGF

More resources:

Code Redlands - Our very own website dedicated to coding.
http://www.coderedlands.org.au/

Arduino – The people who brought you the Arduino Micro that you used today.
http://arduino.cc/

Sparfun – A great place to learn about Arduinos and electronics.
https://learn.sparkfun.com/

Adafruit – Another great place to learn.
https://learn.adafruit.com/

Make – A source of information for the Do-It-Yourself type of person.
http://makezine.com/

Hackaday – Another source of information for the Do-It-Yourself type of person.
http://hackaday.com/

Fritzing – Make your own breadboard layouts and schematics.
http://fritzing.org/home/

Littlebird Electronics – A great Australian source for parts.
http://littlebirdelectronics.com.au/

Jaycar – Another great Australian source for parts.
http://www.jaycar.com.au/