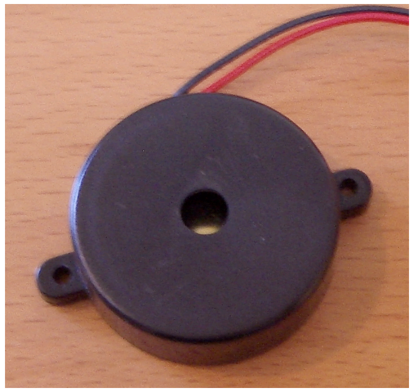
Piezoelectrics

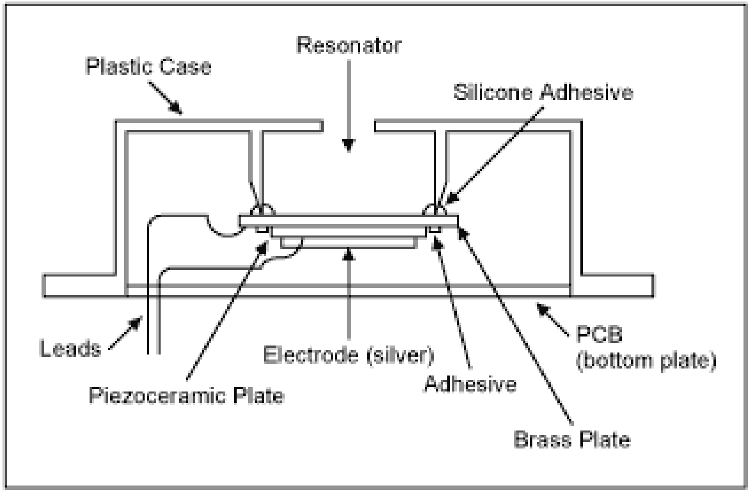
- Big word – *piezein* is greek for "squeeze"
- Some crystals, when squeezed, make a spark
- Turns out the process goes the other way too
- Spark a quartz crystal, and it flexes
- Piezo buzzers use this to make sound
  (flex something back and forth, it moves air)

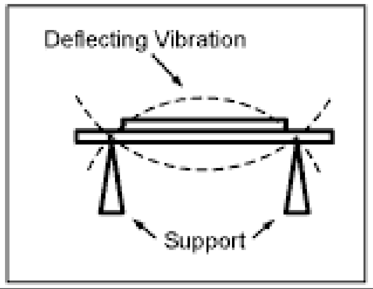
Piezo buzzers don't have quartz crystals, but instead a kind of ceramic that also exhibits piezoelectric properties.
I pronounce it "pie-zoh". Or sometimes "pee-ay-zoh".

Piezo Buzzers



- Two wires, red & black.
  Polarity matters: black=ground
- Apply an oscillating voltage to make a noise
- The buzzer case supports the piezo element and has resonant cavity for sound





Oscillating voltage alternately squeezes and releases the piezo element.
Must apply flucuating voltage, a steady HIGH or LOW won't work.

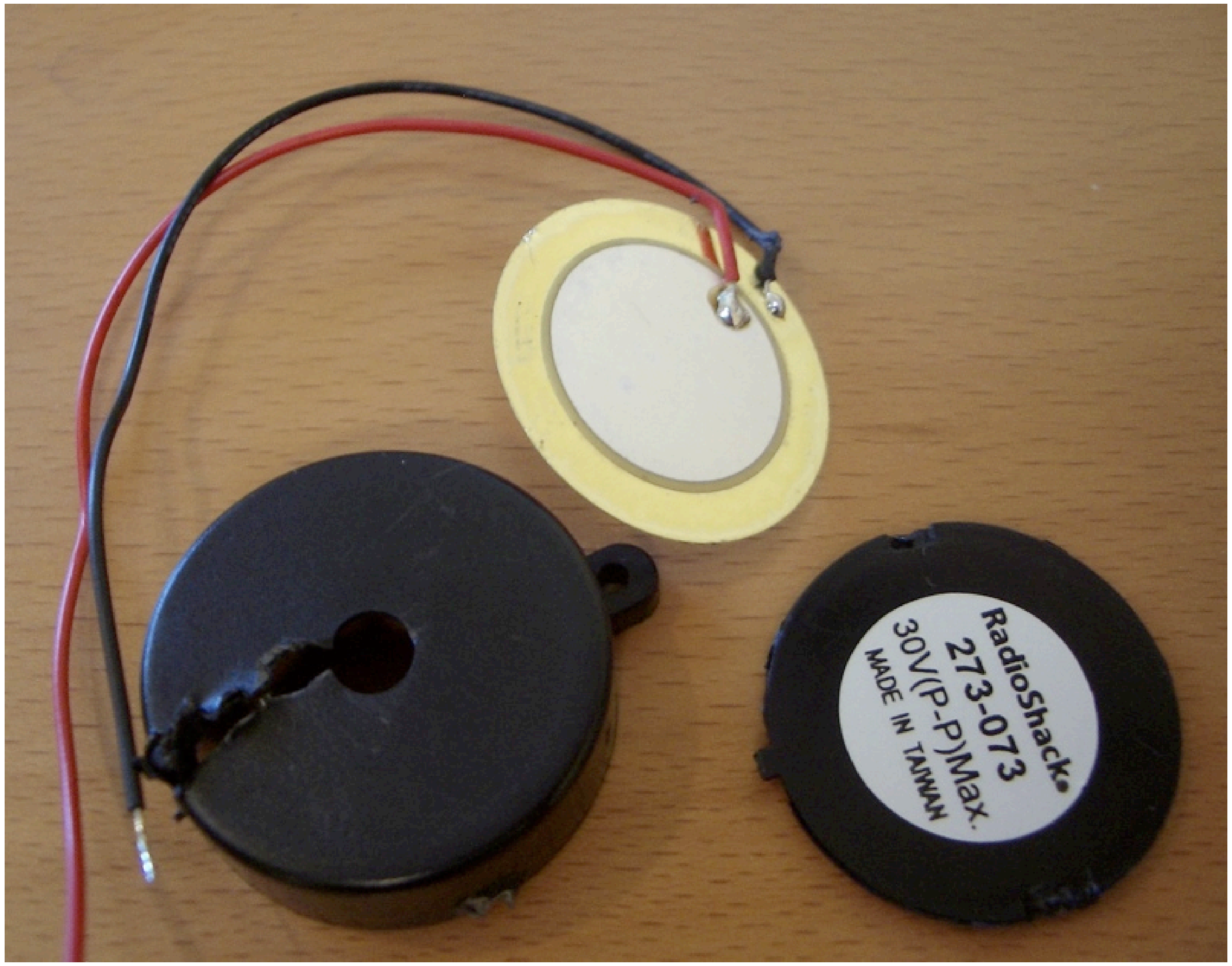
diagrams from: http://www.maxim-ic.com/appnotes.cfm/appnote_number/988

What's in a Piezo Buzzer?

You can get at the piezo element pretty easily.

Be careful not to crack the white disc that is the actual piezo

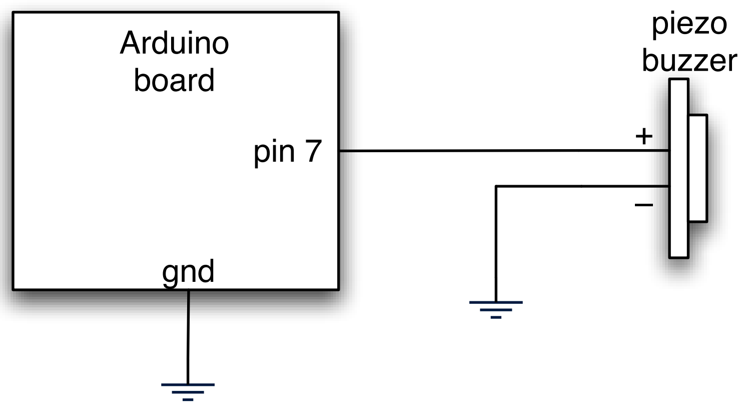
Only take it out of its case to use it as a sensor

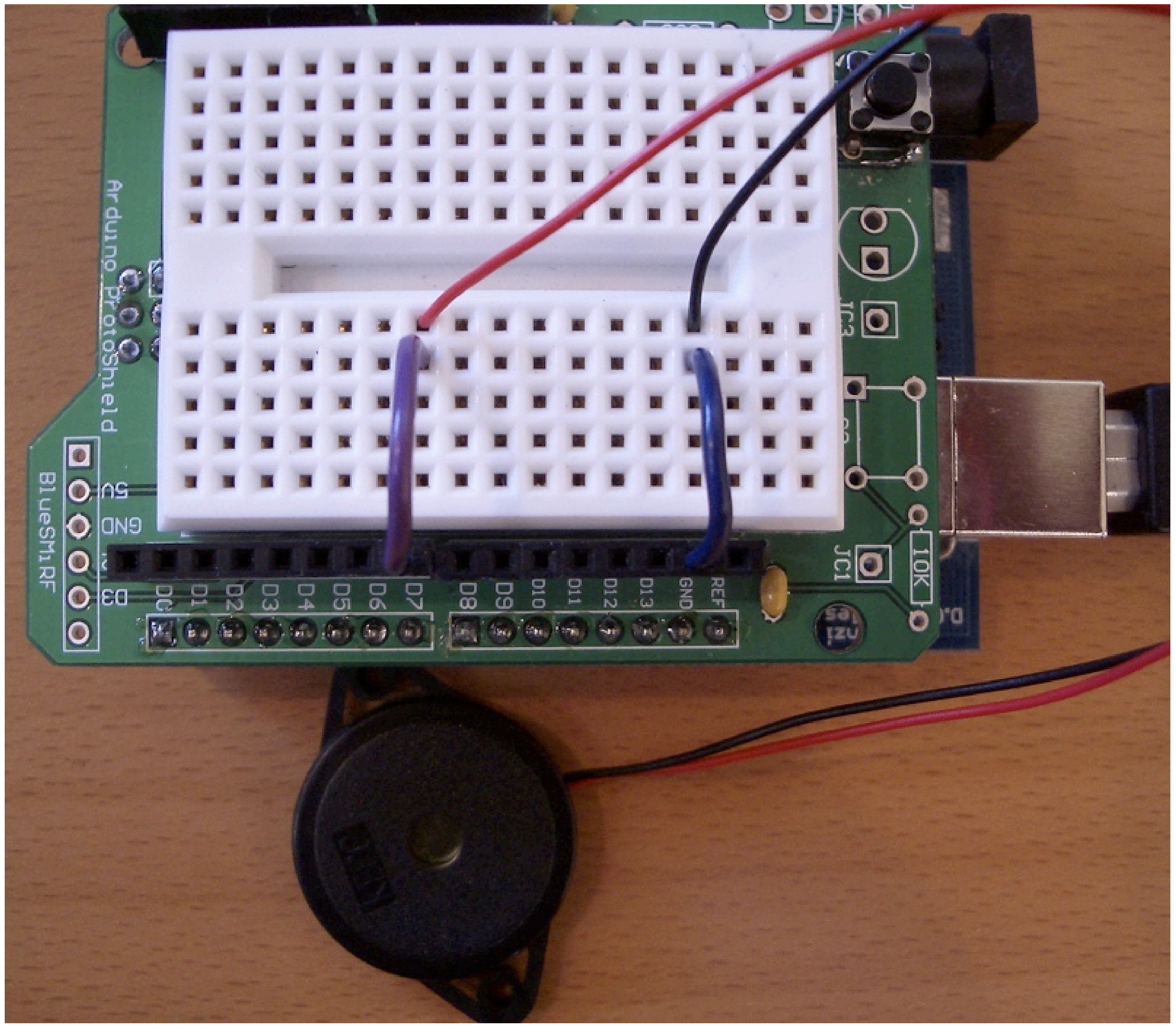


another $1.99 I won't be getting back from Radio Shack

Of course, you usually destroy the enclosure to get at the element.
And it's the enclosure that has the proper support and resonant cavity to make a loud sound

Piezo Buzzer





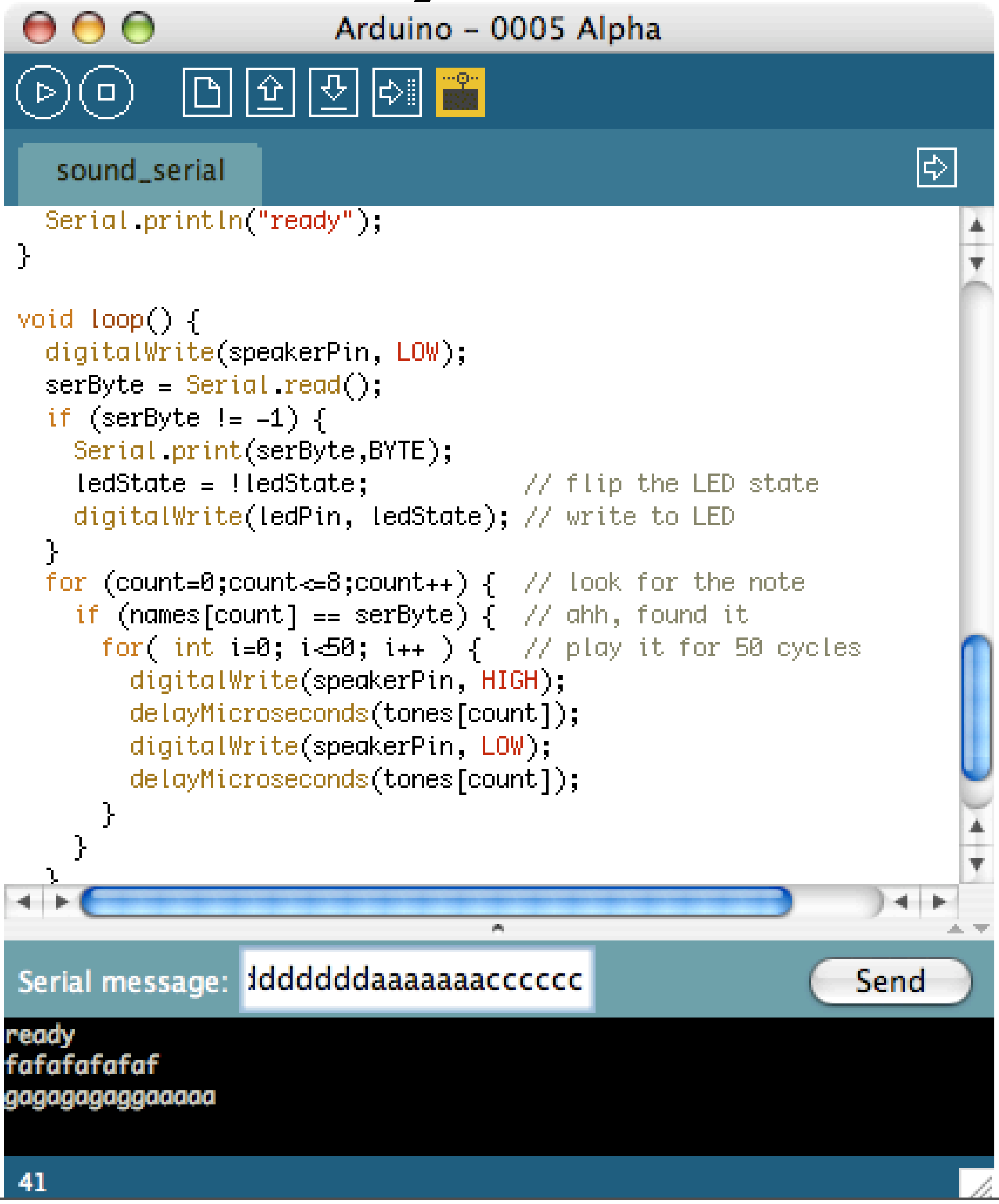
Piezo leads are very thin. The breadboard holes grab them better than the header sockets, which is why the jumper leads are used.

Play a Melody

“sound_serial”

Play the piezo beeper with the Serial Monitor

Type multiple letters from “cdefgabC” to make melodies



```
  Serial.println("ready");
}

void loop() {
  digitalWrite(speakerPin, LOW);
  serByte = Serial.read();
  if (serByte != -1) {
    Serial.print(serByte,BYTE);
    ledState = !ledState;              // flip the LED state
    digitalWrite(ledPin, ledState); // write to LED
  }
  for (count=0;count<=8;count++) {  // look for the note
    if (names[count] == serByte) {  // ahh, found it
      for( int i=0; i<50; i++ ) {   // play it for 50 cycles
        digitalWrite(speakerPin, HIGH);
        delayMicroseconds(tones[count]);
        digitalWrite(speakerPin, LOW);
        delayMicroseconds(tones[count]);
      }
    }
```

```
ddddddaaaaaaacccccc
```

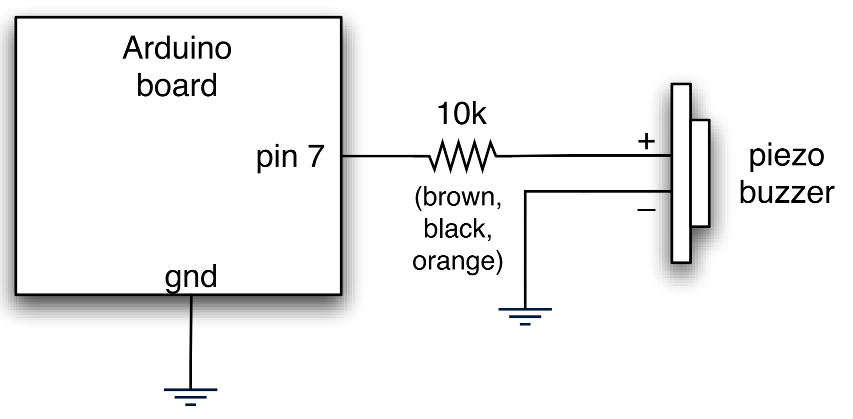
```
ready
fafafafafaf
gagagagaggaaaaa
```

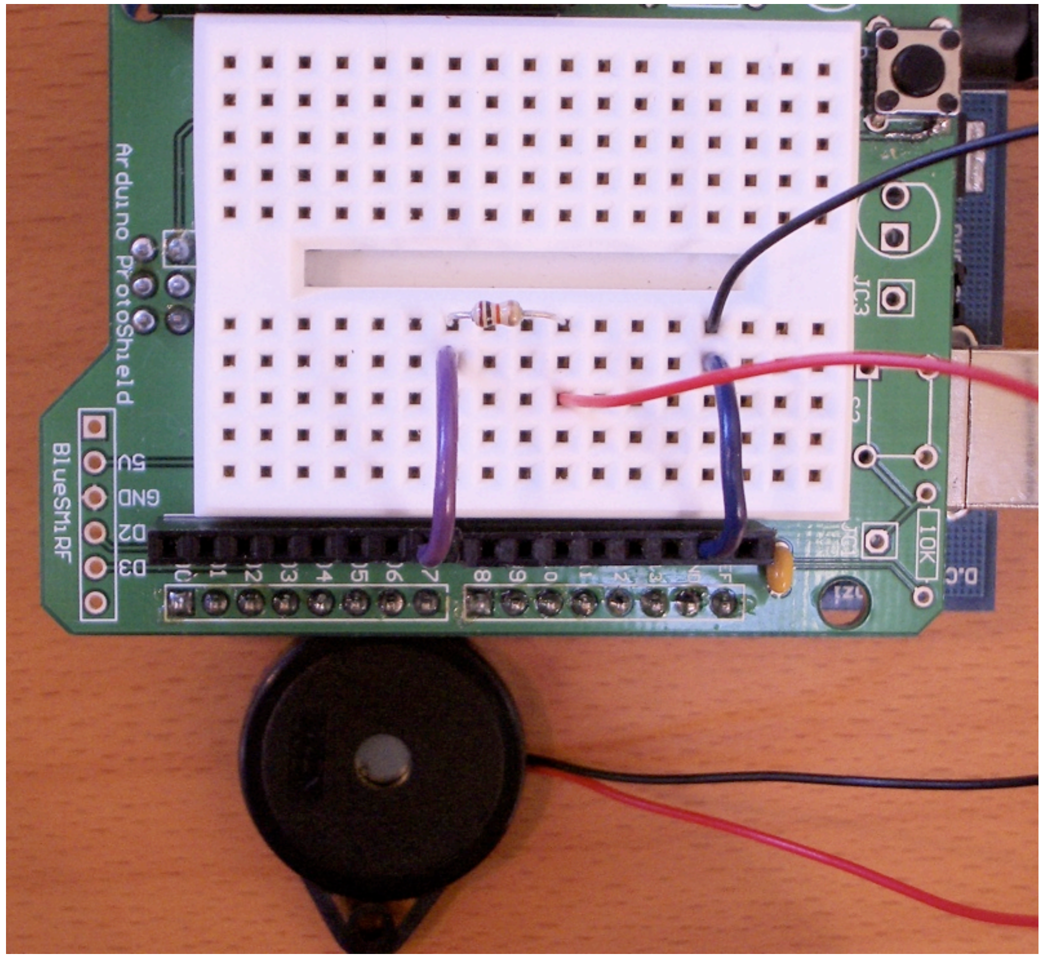
41

This sketch is in the handout, and is based on “Examples/pwm_sound/keyboard_serial”
Notice the problem with this sketch?
Different notes play for different amounts of time.
50 cycles of low C isn’t the same amount of time as 50 cycles of high B

Making it Quieter

Easiest way: add a resistor



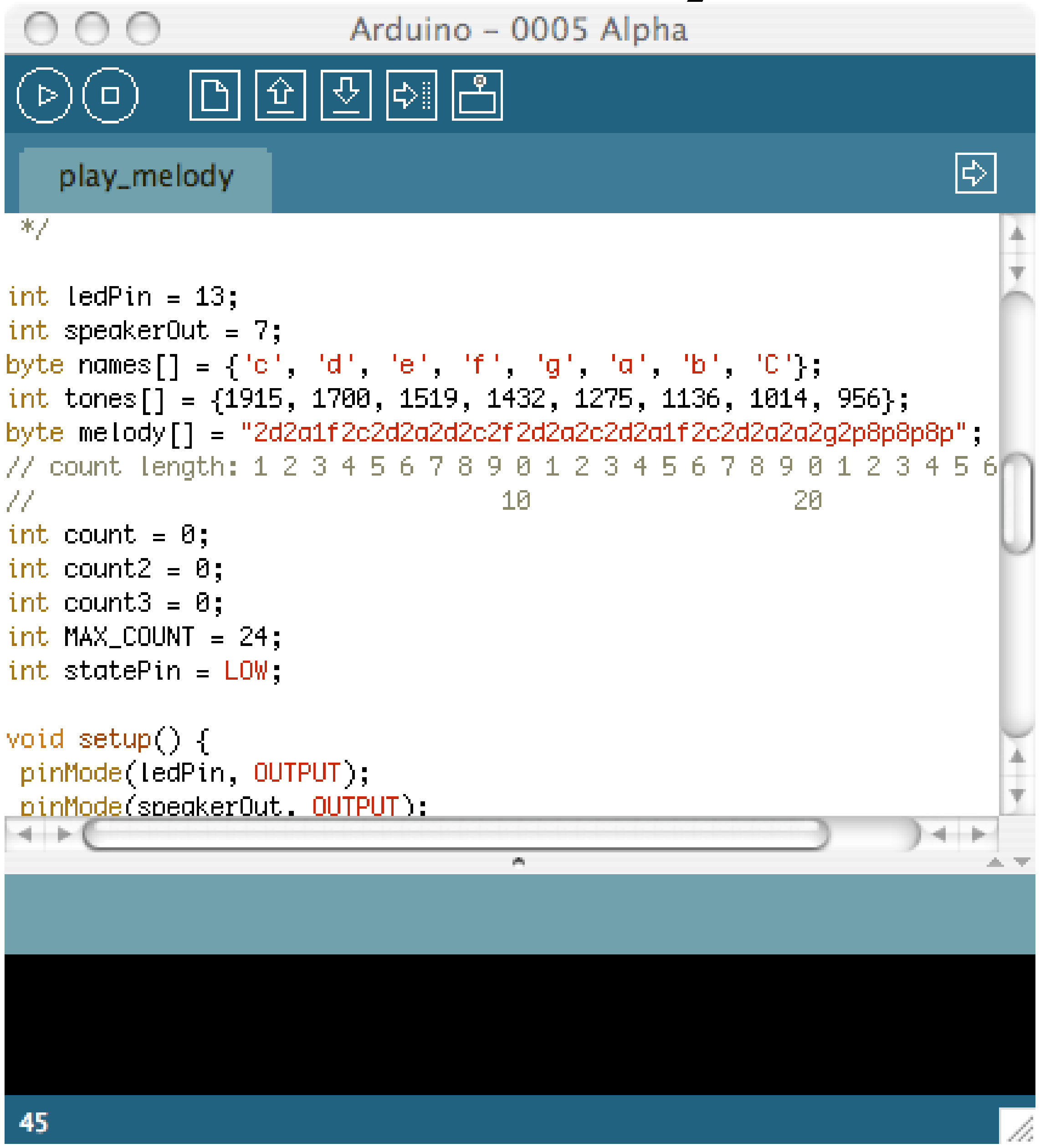


Like most things in electronics, if you want less of something, add a resistor.
A better value would probably be 1k, but we don’t have that on hand.
This may not seem important now, but wait for the next project.

Play a Stored Melody

“play_melody”

Plays a melody stored in the Arduino



```
 */

int ledPin = 13;
int speakerOut = 7;
byte names[] = {'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C'};
int tones[] = {1915, 1700, 1519, 1432, 1275, 1136, 1014, 956};
byte melody[] = "2d2a1f2c2d2a2d2c2f2d2a2c2d2a1f2c2d2a2a2g2p8p8p8p";
// count length: 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
//                                 10                  20
int count = 0;
int count2 = 0;
int count3 = 0;
int MAX_COUNT = 24;
int statePin = LOW;

void setup() {
 pinMode(ledPin, OUTPUT);
 pinMode(speakerOut, OUTPUT);
```

This is in the handout, but is also in “Examples/pwm_sound/play_melody” (pin changed)
Melody definition is sort of like the old cell ringtone style
Melody playing logic is hard to follow.

Make a Theremin

"ooo-weee-ooooo"

The original spooky sound machine

Works by measuring your body's electric field

No touching needed!

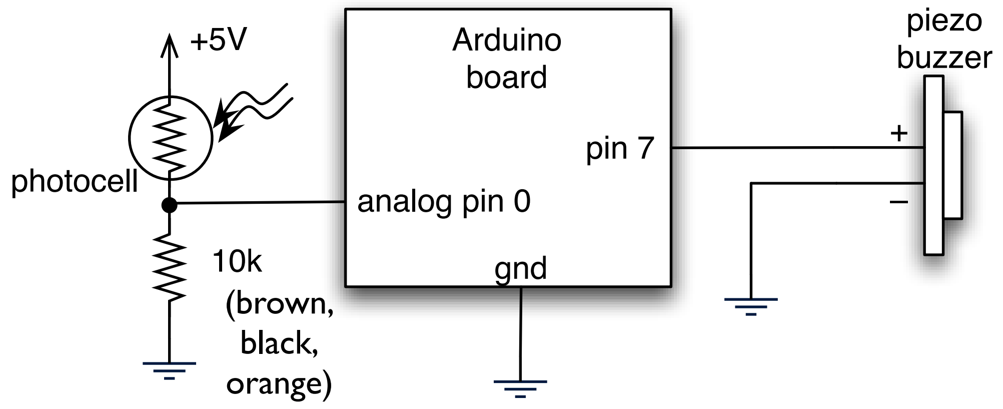
We'll use light in lieu of RF

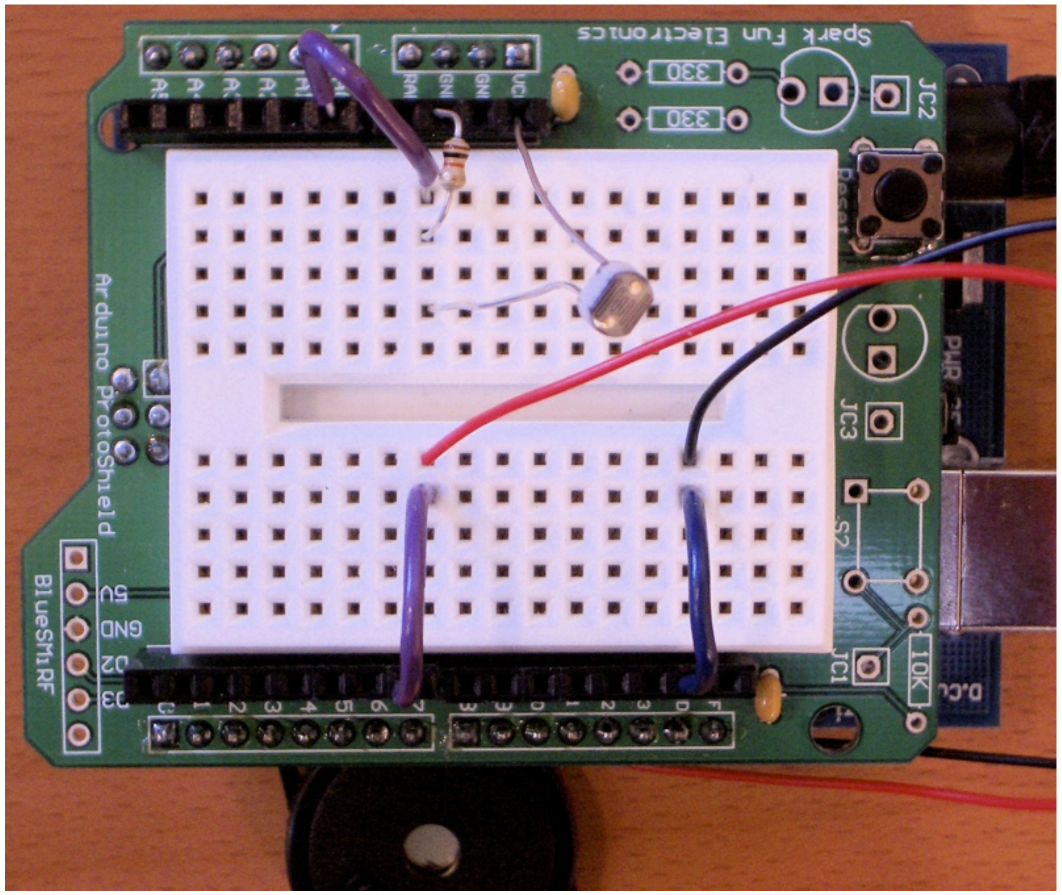
Leon Theremin

As heard on Star Trek, Beach Boys, horror movies, Mars Attacks!, and bad New Age songs.
Works sorta like those touch switches, but no touching here.
That is, your body becomes a variable capacitor.

Make a Theremin

Take photocell circuit from before, bolt it on





This is a light-to-sound converter, if you will.

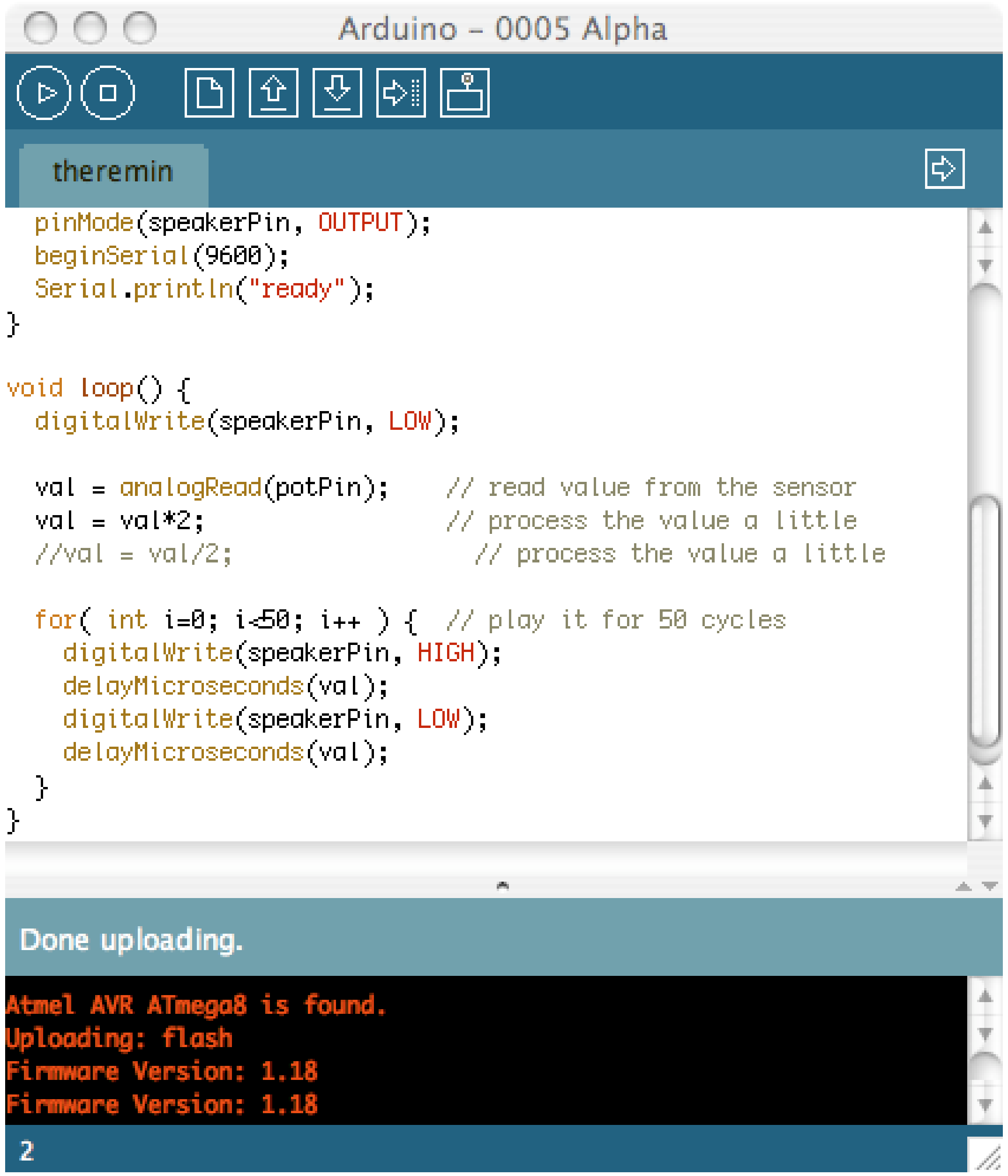
Make a Theremin

"theremin"

Move hand over photocell to change pitch

Play with val processing & cycles count to alter sensitivity, pitch and timbre

This is *frequency modulation,* since you're changing the frequency



```
  pinMode(speakerPin, OUTPUT);
  beginSerial(9600);
  Serial.println("ready");
}

void loop() {
  digitalWrite(speakerPin, LOW);

  val = analogRead(potPin);    // read value from the sensor
  val = val*2;                 // process the value a little
  //val = val/2;                 // process the value a little

  for( int i=0; i<50; i++ ) {  // play it for 50 cycles
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(val);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(val);
  }
}
```

Done uploading.

```
Atmel AVR ATmega8 is found.
Uploading: flash
Firmware Version: 1.18
Firmware Version: 1.18
```

2

Okay so maybe it sounds more like a bad video game than a spooky movie
The glitchy sound is cause because of the time it takes to read the sensor
There are ways around such stuff, but requires more complex programming using timers & interrupts
The sound can get annoying quick

Piezo Buzzer as Sensor

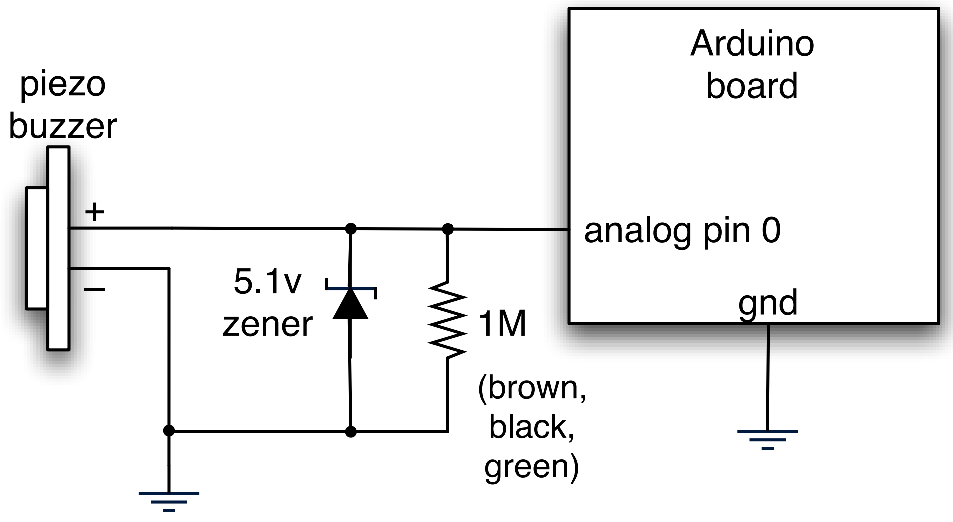
- Piezo buzzers exhibit the *reverse* piezoelectric effect.
- The normal piezoelectric effect is generating electricity from squeezing a crystal.
- Can get several thousand volts, makes a spark
- You probably have seen a big example of this already:

fireplace lighter

I have a demo piezo igniter from one of these lighters. It's fun to shock yourself.
Puts out several thousand volts. (ionization voltage of air =~ 30kV/cm)

Piezo Read

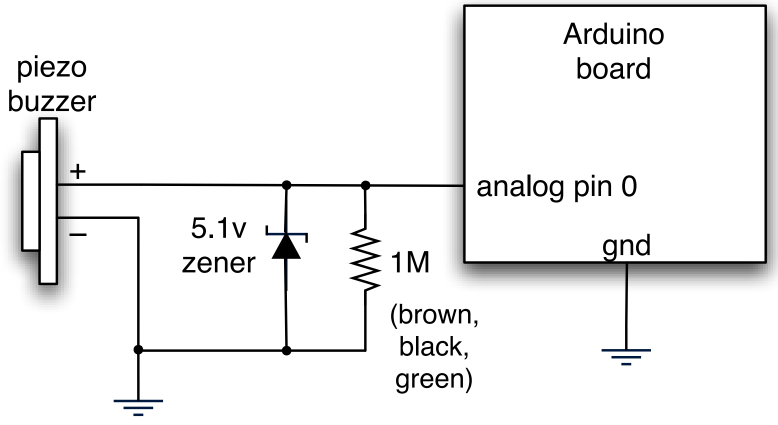
- To read a piezo you can just hook it into an analog input, but:
- You need to drain off any voltage with a resistor, or it just builds up
- You should have a protection diode to limit big voltages, else fry your inputs

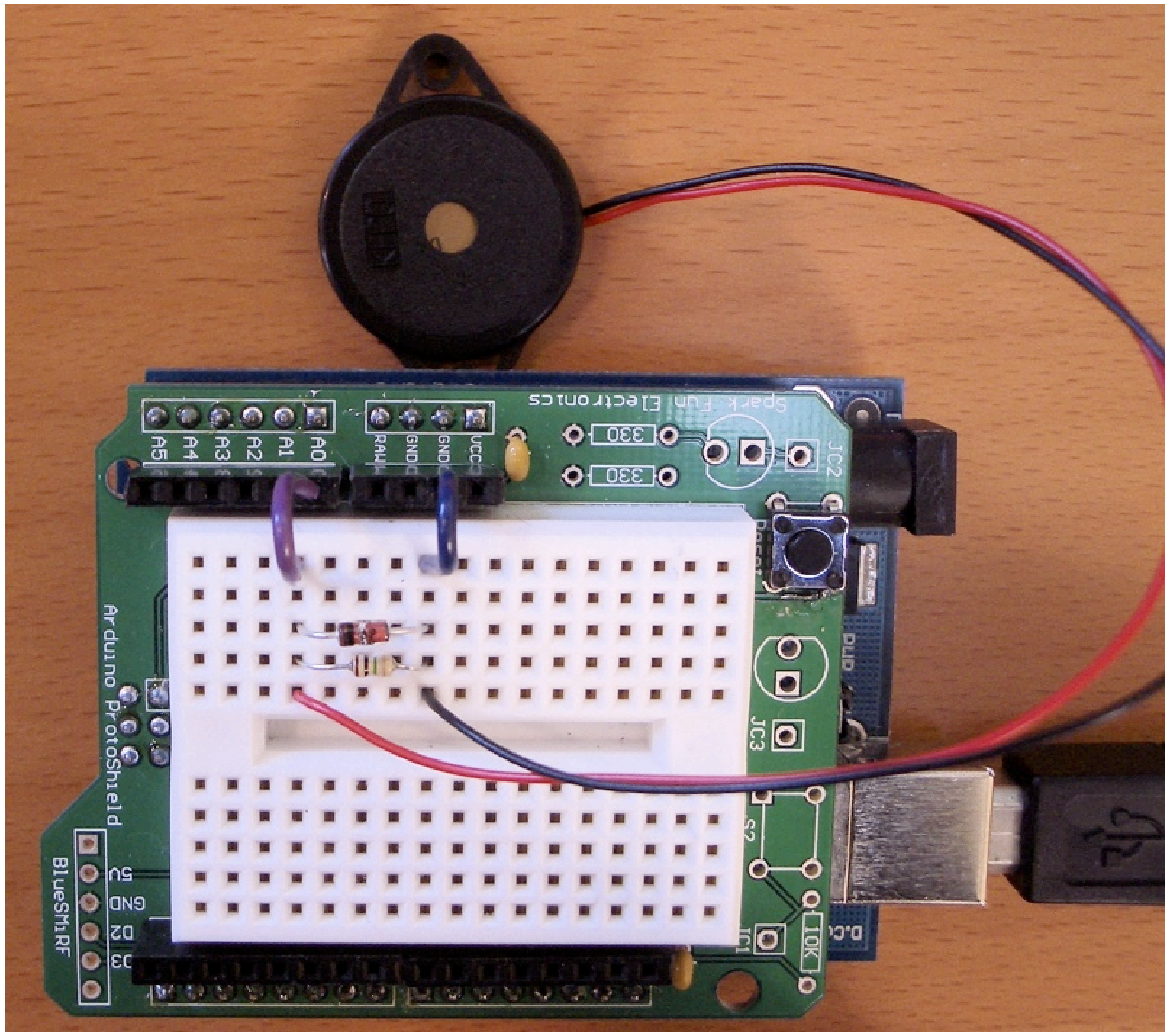


piezo input schematic

Note polarity of piezo still matters.
The protection diode is a special kind of diode called a “zener diode”. It acts invisible until the voltage gets over its designed value (5.1 volts in this case), then it acts like a short circuit.

Piezo Read





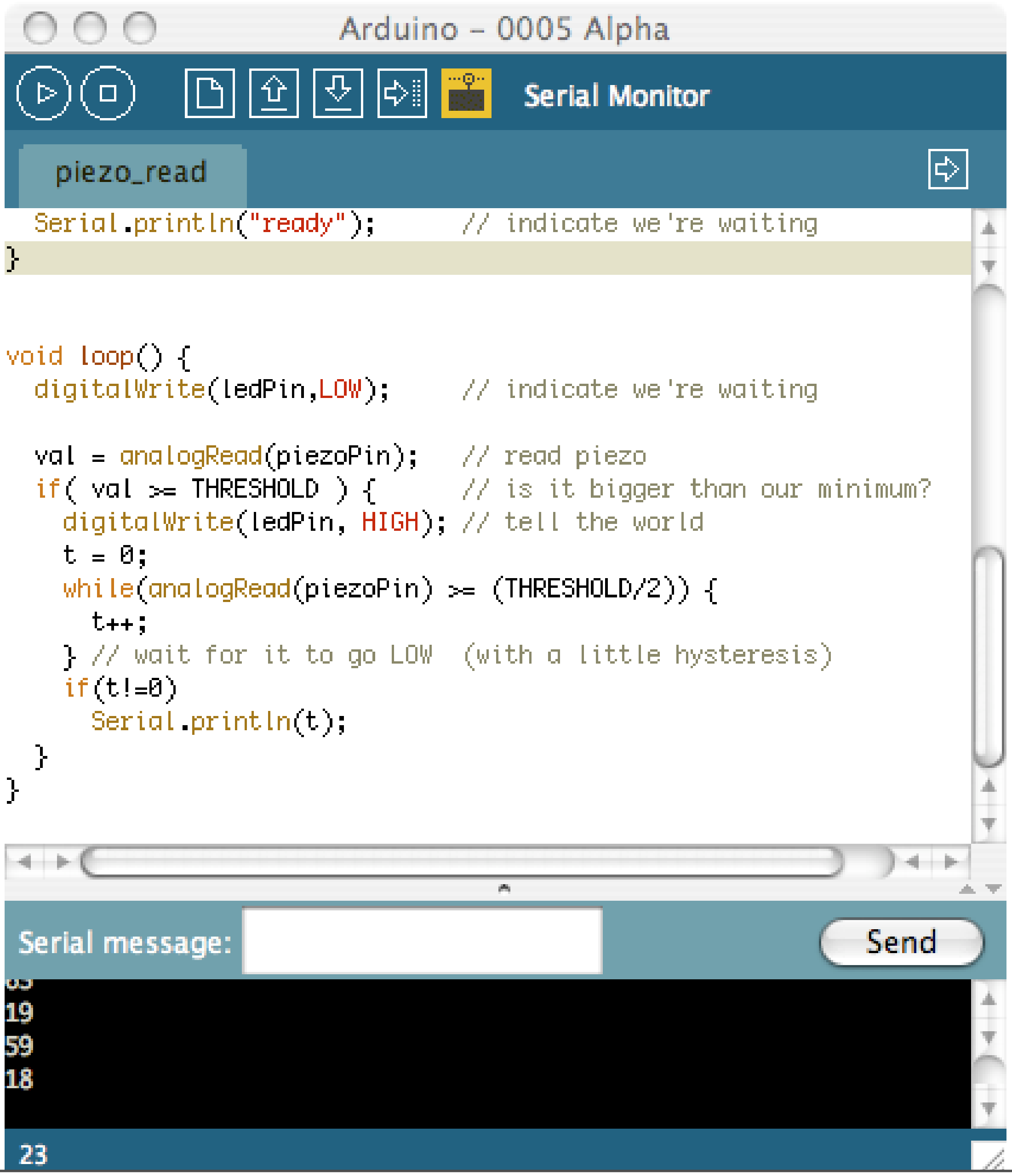
Create two little busses for GND and A0, and hook components across it.
Black bar on diode indicates “bar” of diode.

Piezo Read

“piezo_read”

Whack the piezo to generate a number based on force of whack

Waits for input to go over threshold, then to drop below threshold



```
  Serial.println("ready");       // indicate we're waiting
}


void loop() {
  digitalWrite(ledPin,LOW);      // indicate we're waiting

  val = analogRead(piezoPin);    // read piezo
  if( val >= THRESHOLD ) {       // is it bigger than our minimum?
    digitalWrite(ledPin, HIGH);  // tell the world
    t = 0;
    while(analogRead(piezoPin) >= (THRESHOLD/2)) {
      t++;
    } // wait for it to go LOW  (with a little hysteresis)
    if(t!=0)
      Serial.println(t);
  }
}
```

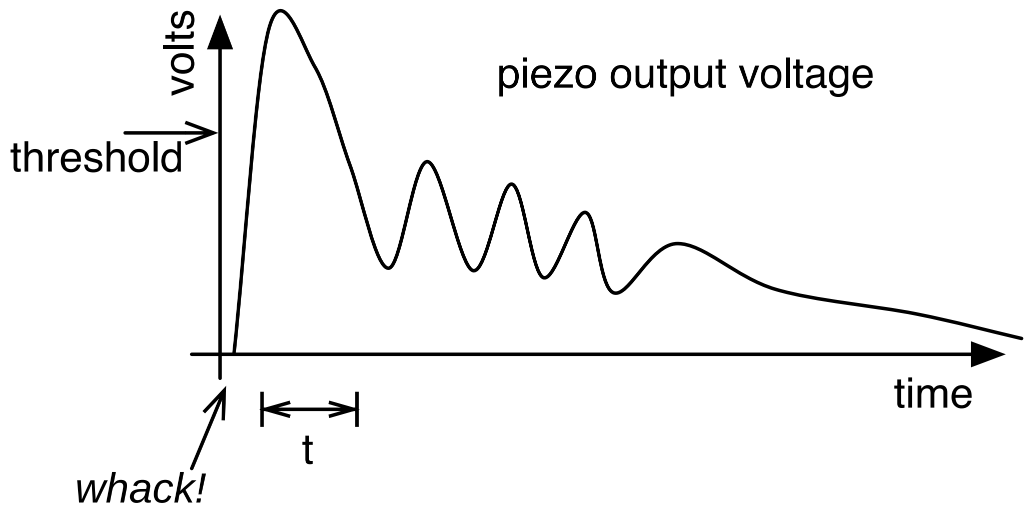
```
19
59
18
```

23

Number is “t”, the number of times it looped waiting for the value to drop below THRESHOLD/2.

How Does that Work?

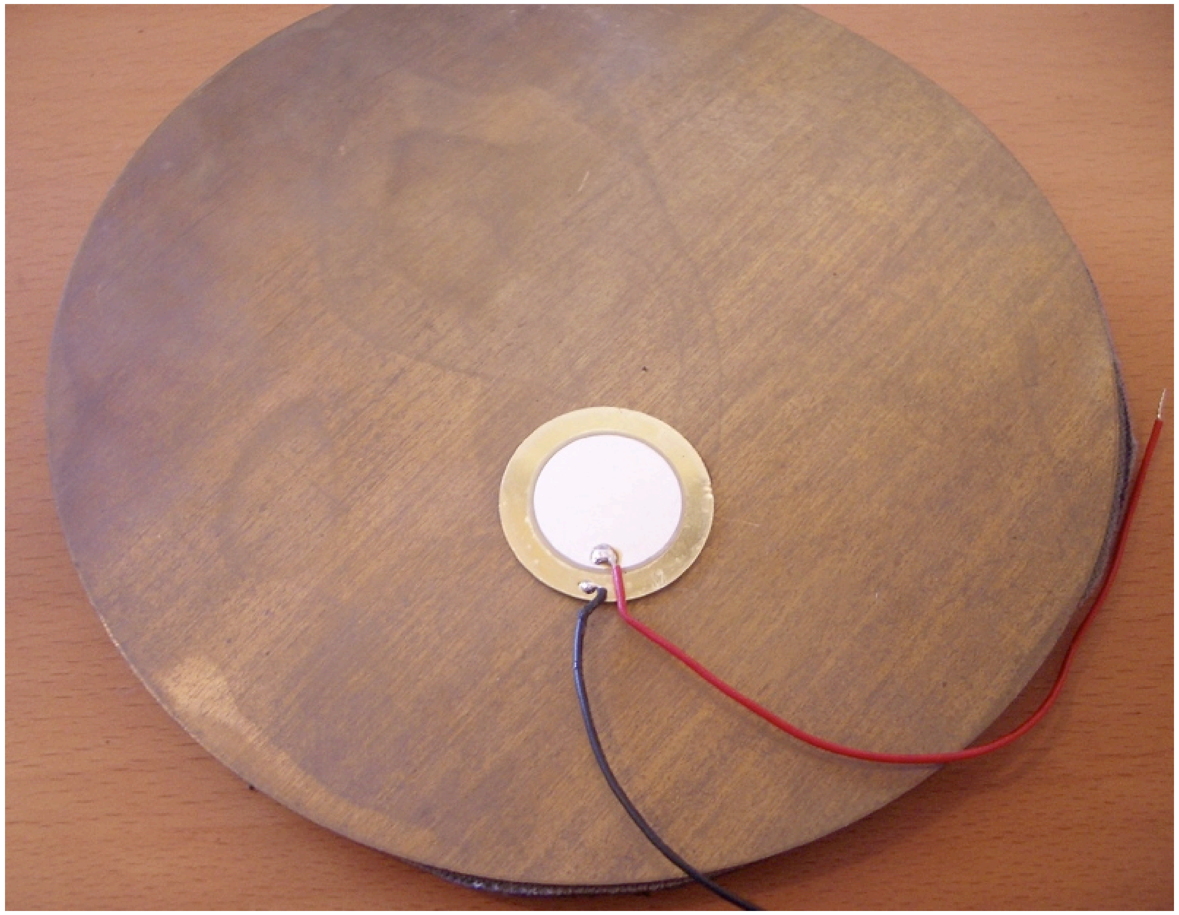
- When a piezo is struck, it "rings" like a bell
- But instead of sound, it outputs voltage
- The sketch measures *time* above a certain voltage, hoping to catch largest ring



Depending on how fast you can watch the input, this technique works either really well or not that well. There are much faster ways of watching inputs that loops with analogRead()
But for now it works okay

Custom Piezo Sensors

Can mount the element on anything
(floor mat, door, your body, etc.)



Here's one glued to a larger brass disc for a drum trigger