Arduino Workshop

ARDUINO

Project 6

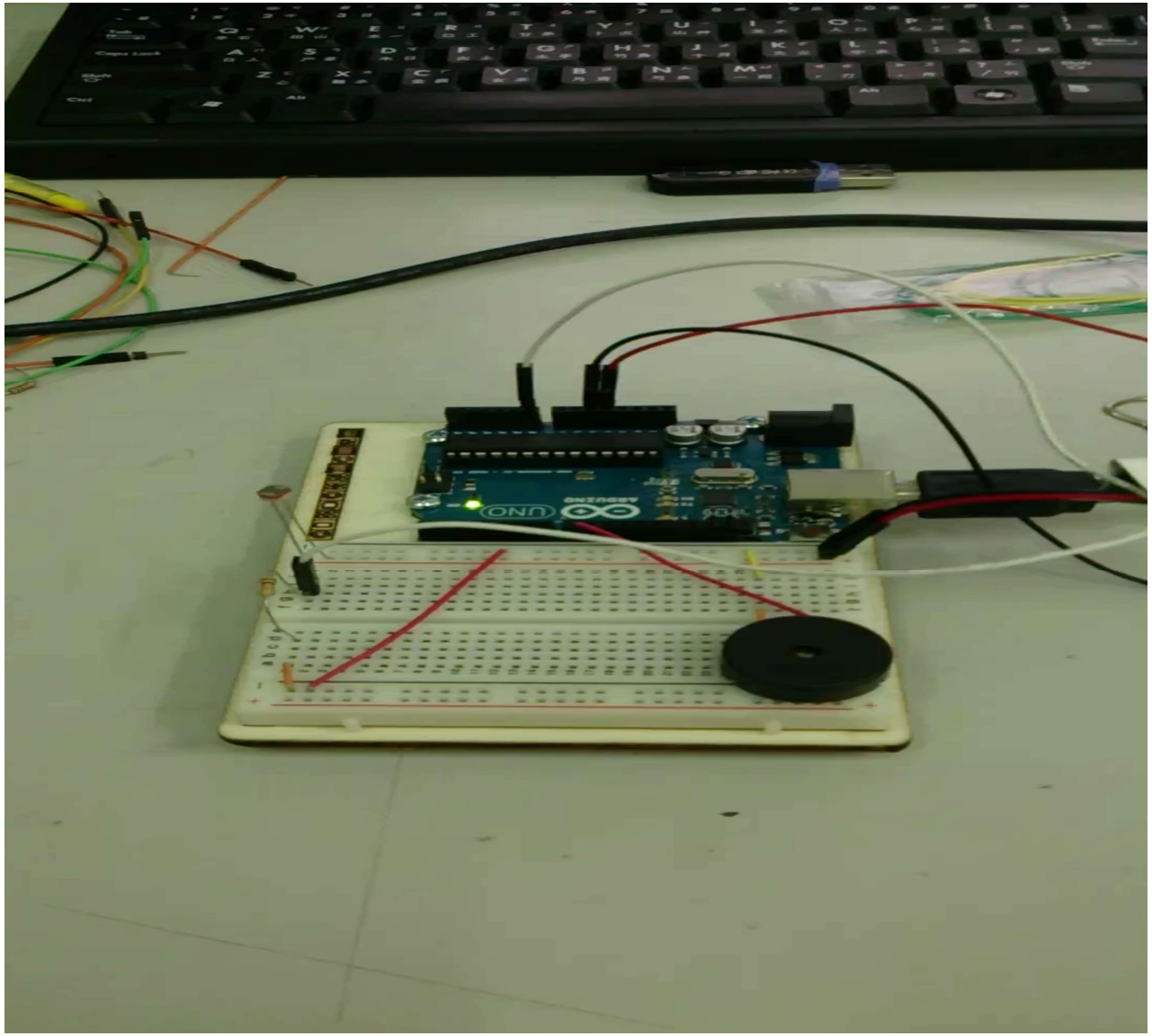
Light Theremin (特雷門琴)

Discover making sound with the tone() function,
calibrating analog sensors

Introduction

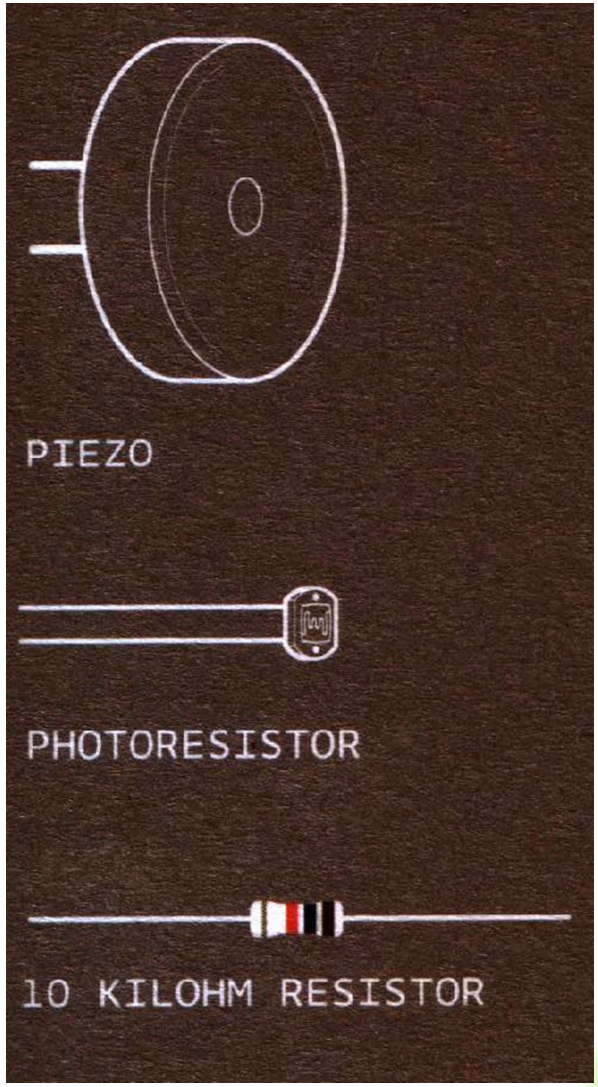
- A theremin is an instrument that makes sounds based on the **movements of a musician's hands around** the instrument.
- The theremin detects where a performer's hands are in relation to **two antennas** by reading the **capacitive change** on the antennas.
  - One antenna controls the **frequency** of the sound and the other controls **volume**.
  - While the Arduino can't exactly replicate the mysterious sounds from this instrument, it is possible to emulate them using the **tone ()** function.

Demo



Ingredients

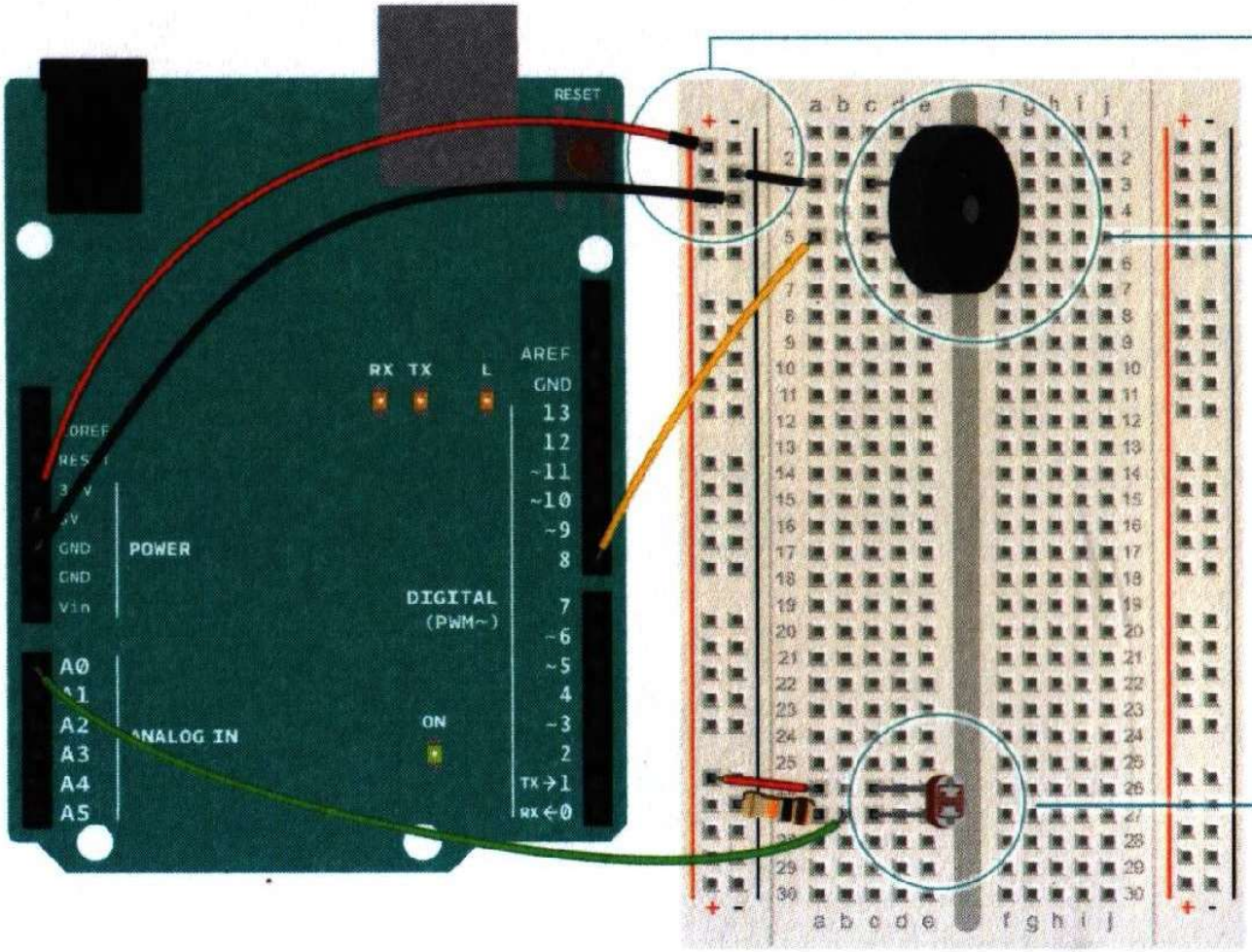
- 1 x 壓電(Piezo)
- 1 x 光敏電阻
- 1 x 10K電阻



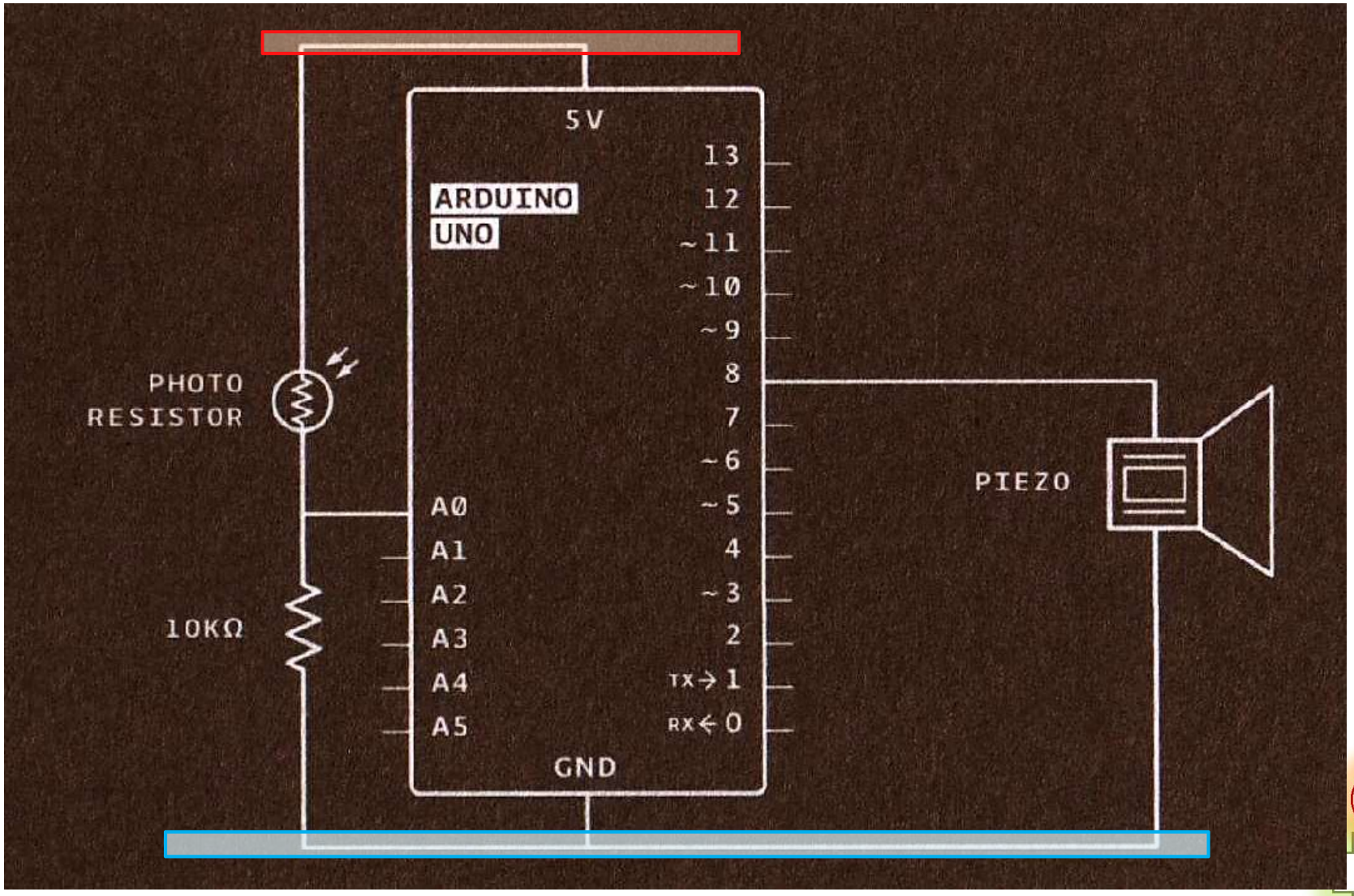
Piezo (壓電/蜂鳴器)

- is a small element that vibrates when it receives electricity.
  - When it moves, it displaces air around it, creating sound waves.

The Top View of the Circuit



Schematic Diagram



Setting Calibration

- analogRead() returns readings didn't range all the way from 0 to 1023.
  - The **fixed resistor** connecting to ground limits the low end of the range
  - The **brightness of your light** limits the high end.
- Instead of settling for a limited range, you'll calibrate the sensor readings getting the high and low values
  - mapping them to sound frequencies using the **map( )** function to get as much range out of your theremin as possible.

PWM vs. tone()

Notice how the signal is low most of the time, but the frequency is the same as PWM 200.

Notice how the voltage is high most of the time, but the frequency is the same as PWM 50.

The duty cycle is 50% (on half the time, off half the time), but the frequency changes.

Same duty cycle as Tone 440; but twice the frequency.

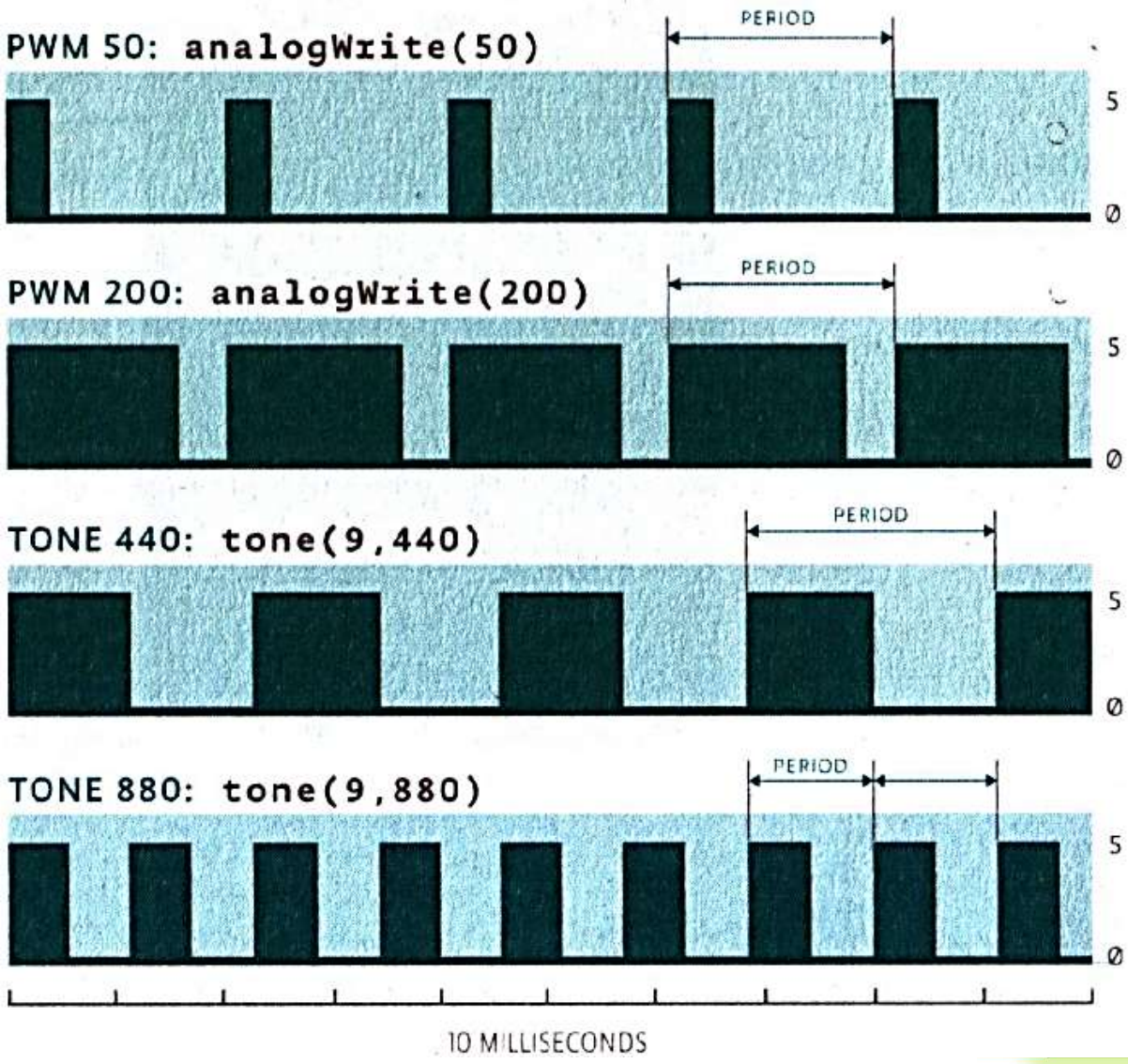


Fig. 1

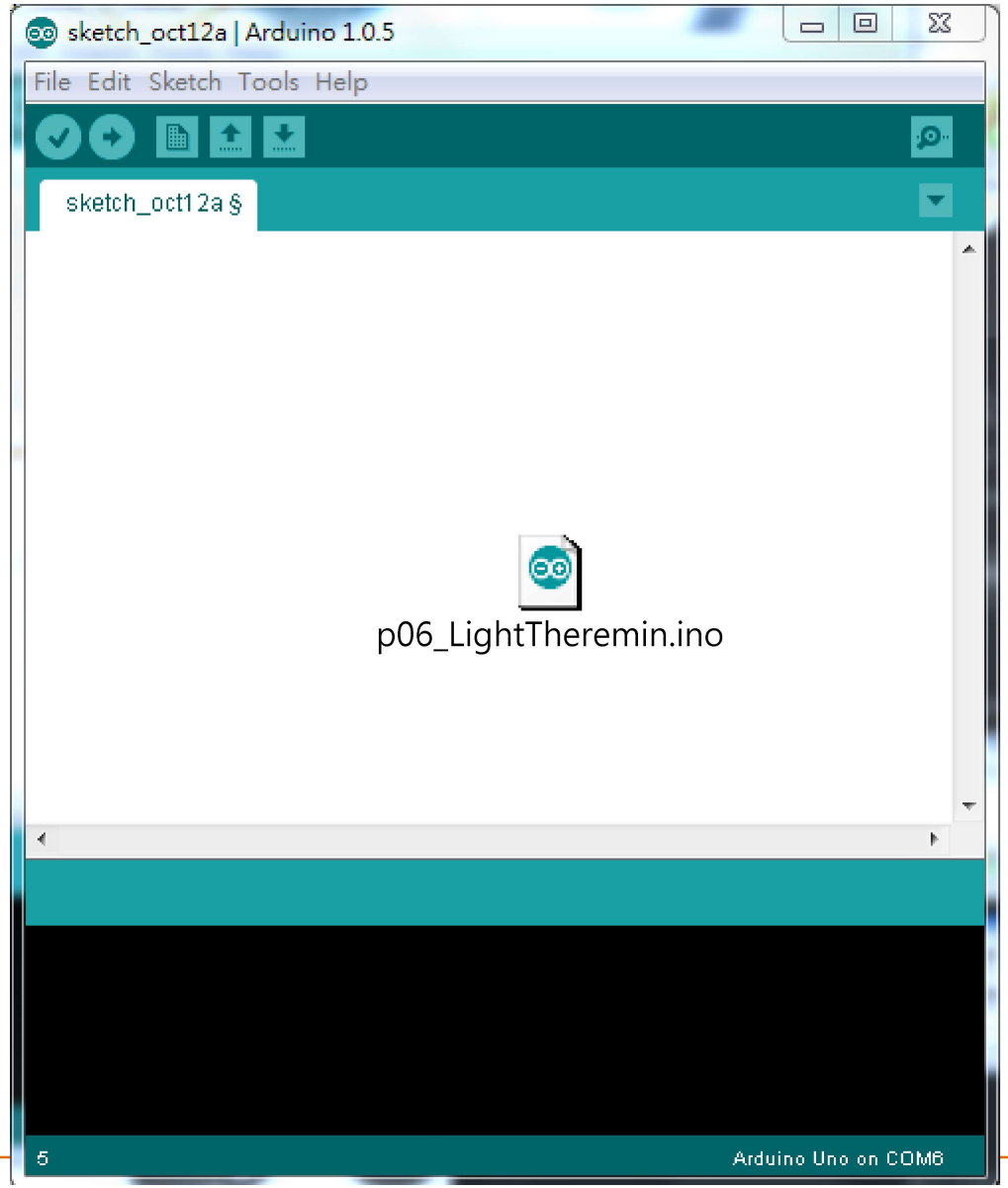
PWM vs. tone()

- **PWM** with analogWrite( )
  - the frequency is fixed
  - you change the ratio of the pulses in that period of time to vary the duty cycle.
- **tone** ( )
  - you're still sending pulses, but changing the frequency of them.
  - tone ( ) always pulses at a **50% duty cycle**
    - (half the time the pin is high,the other half the time it is low).

tone() function

- Description
  - Generates a square wave of the specified frequency (and 50% duty cycle) on a pin. A duration can be specified, otherwise the wave continues until a call to noTone().
- Syntax
  - tone(pin, frequency) or tone(pin, frequency, **duration**)
    - Parameters
      - pin: the pin on which to generate the tone
      - frequency: the frequency of the tone in hertz - unsigned int
      - duration: the duration of the tone in milliseconds (optional) - unsigned long
- Returns
  - nothing

Codes


sketch_oct12a | Arduino 1.0.5
File Edit Sketch Tools Help
sketch_oct12a §
p06_LightTheremin.ino
5
Arduino Uno on COM6

Project 7

Keyboard Instrument

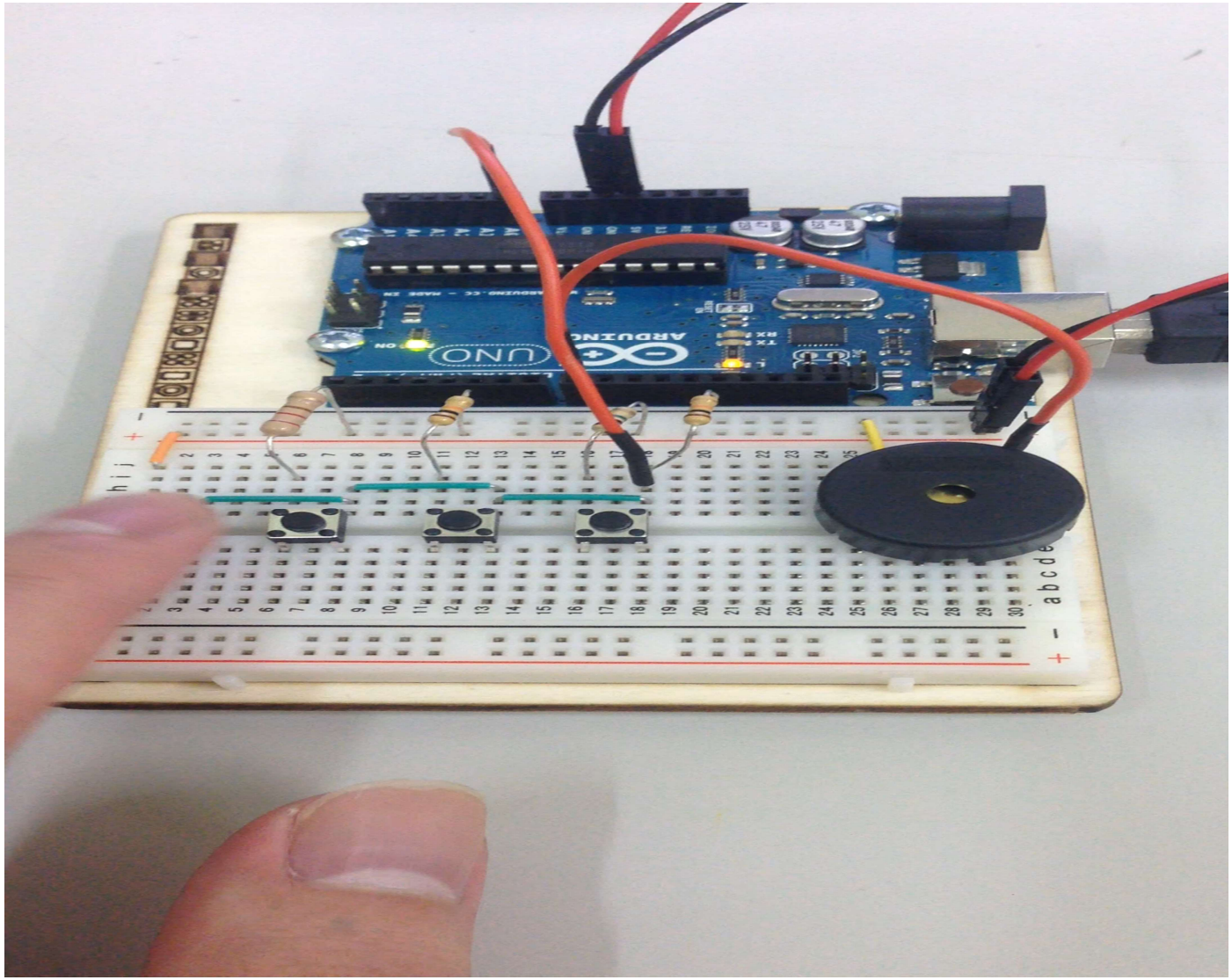
Discover: resistor ladders, arrays

Introduction

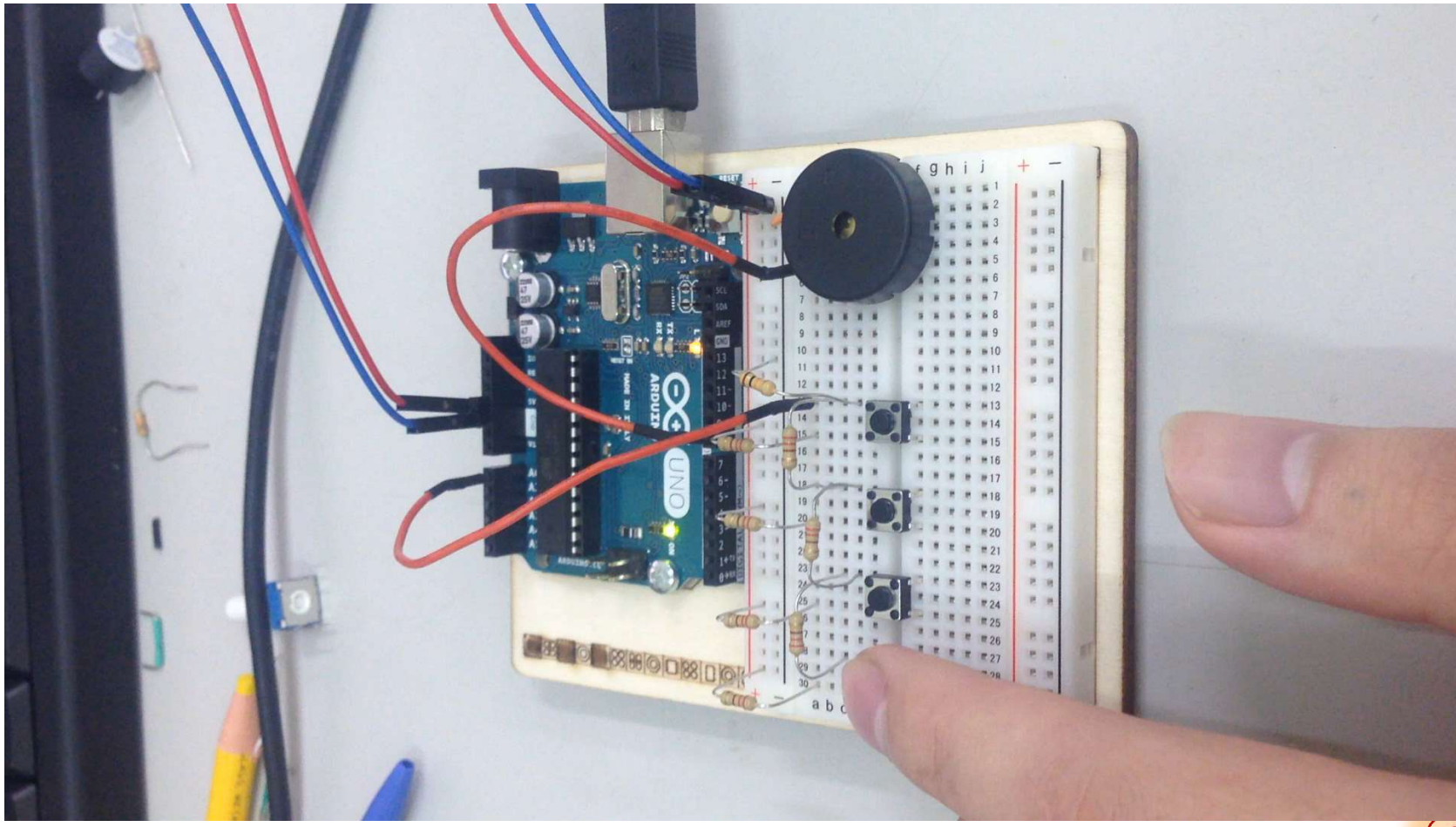
- Resistor Ladder(電阻梯形電路).
  - is an electrical circuit made of repeating unit of resistor
    - this time we built a R2R(區域對區域) Ladder which is a simple way to perform a **Digital to Analog conversion**.

- **An Array** is a way to store different values that are related to each other using only one name.

Demo (one key version)

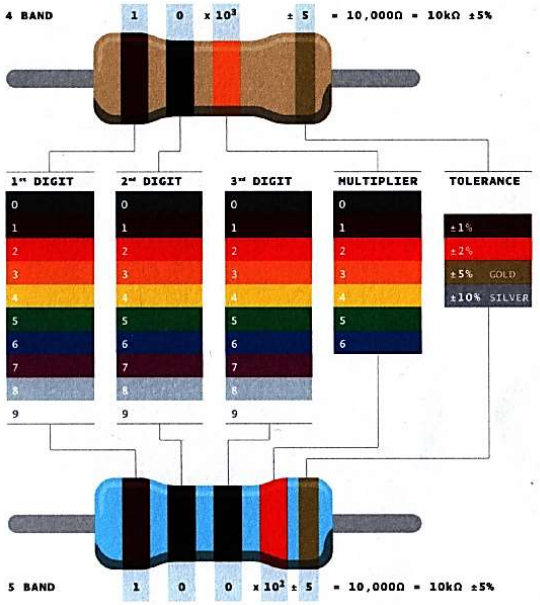


Demo (two keys version)

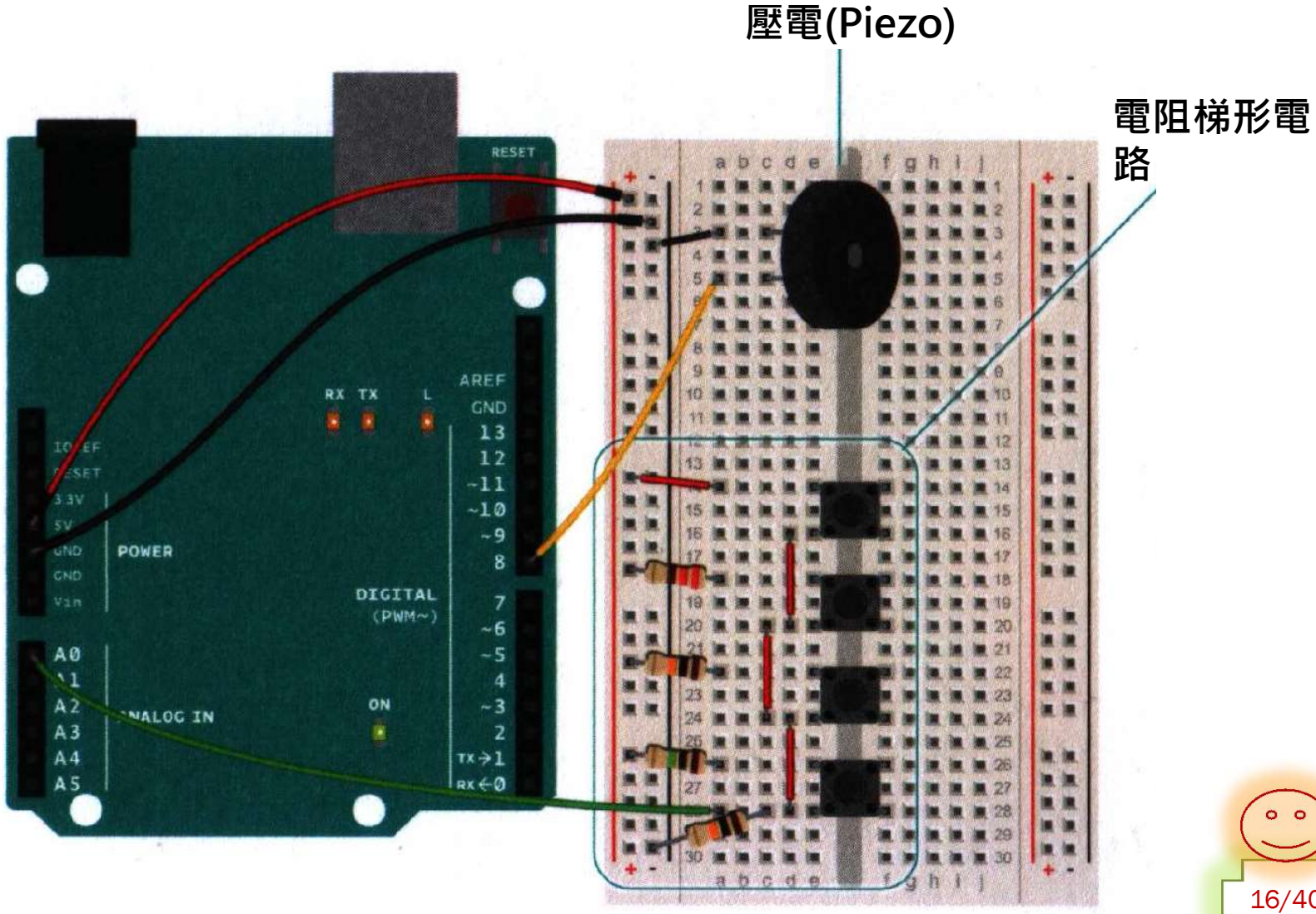


Ingredients

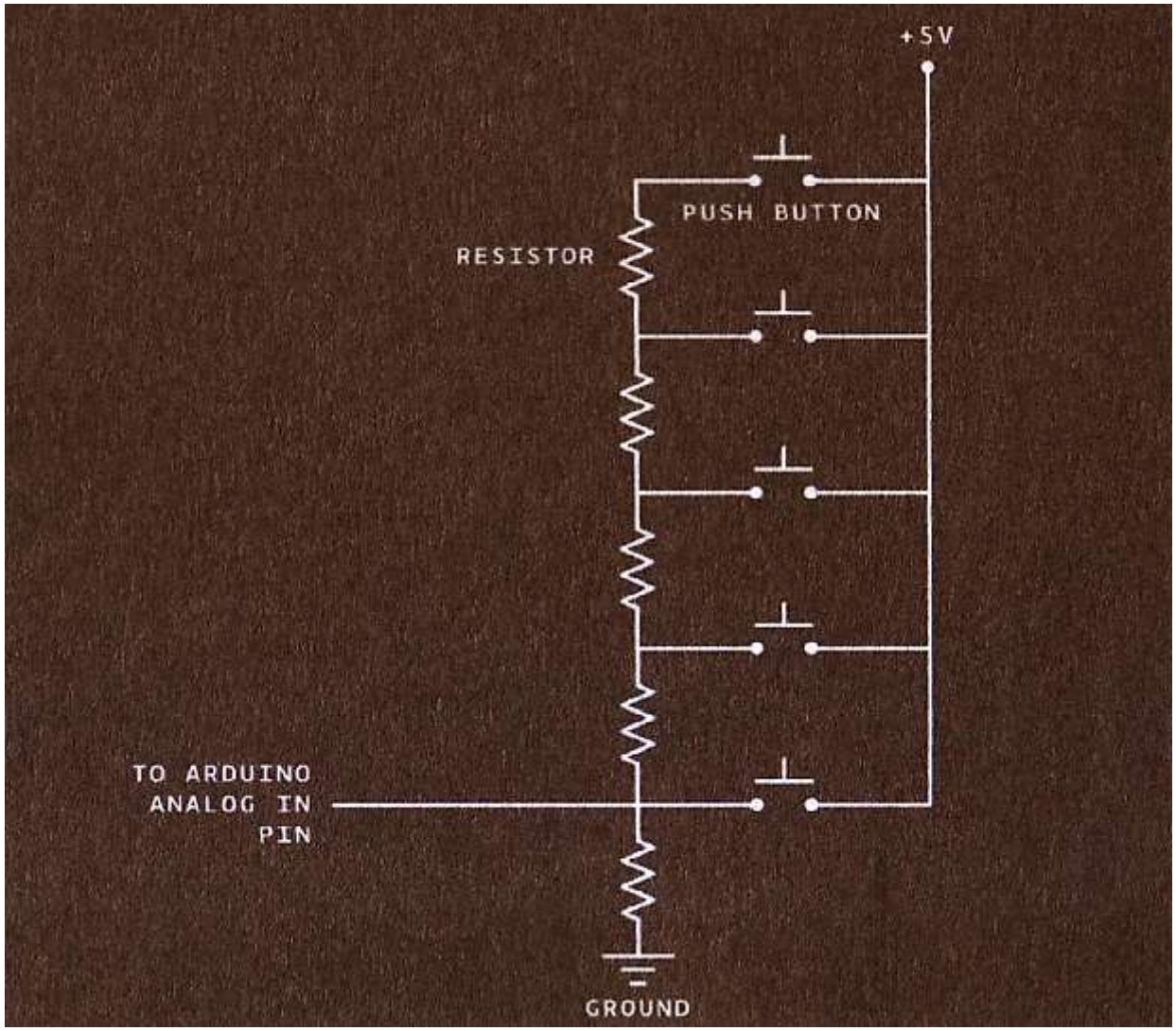
- 4個開關(Switch)
- 1個壓電(Piezo)
- 2個10K Ω電阻
- 1個1M Ω電阻
- 1個220 Ω電阻



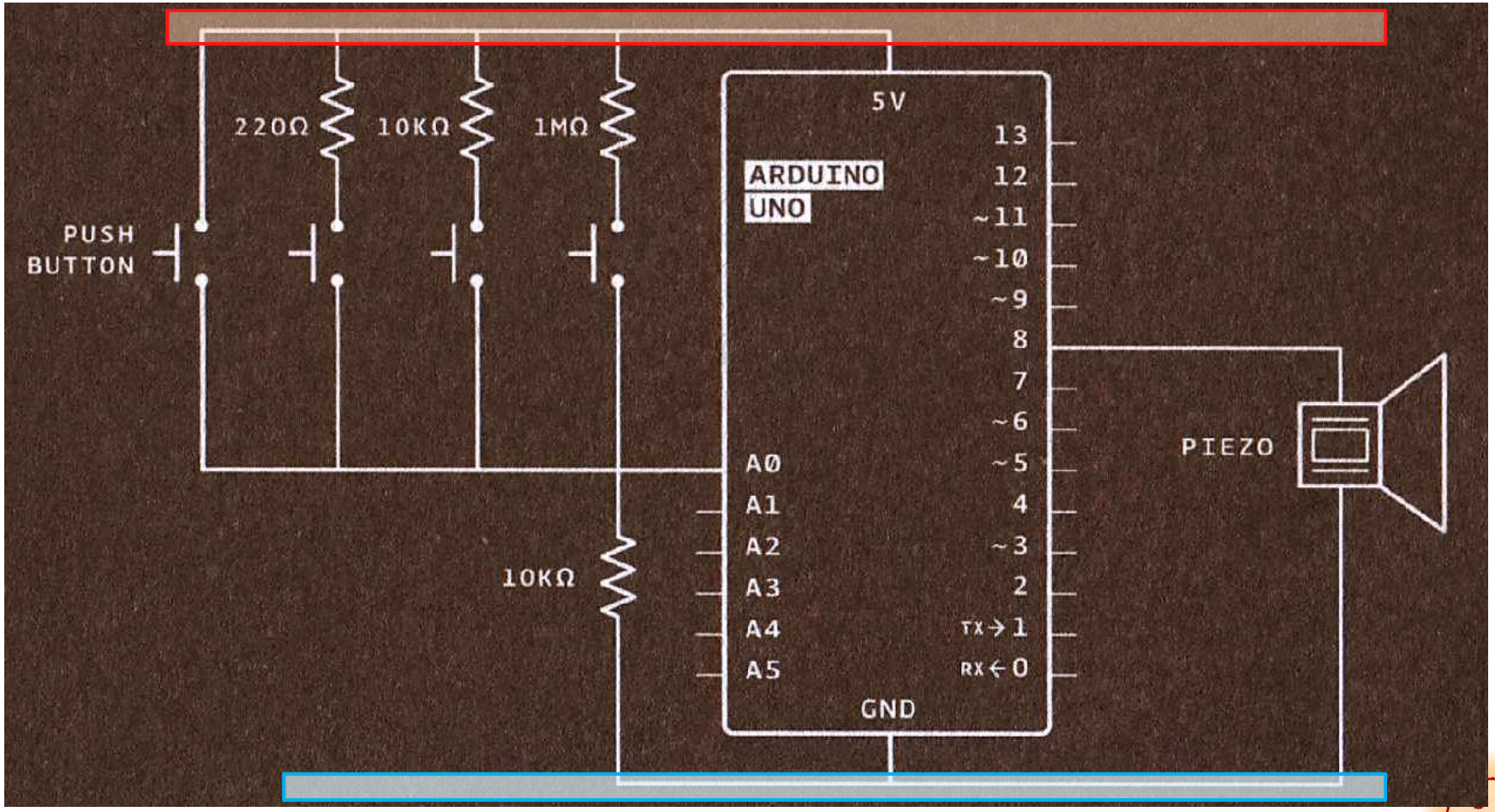
The Top View of the Circuit



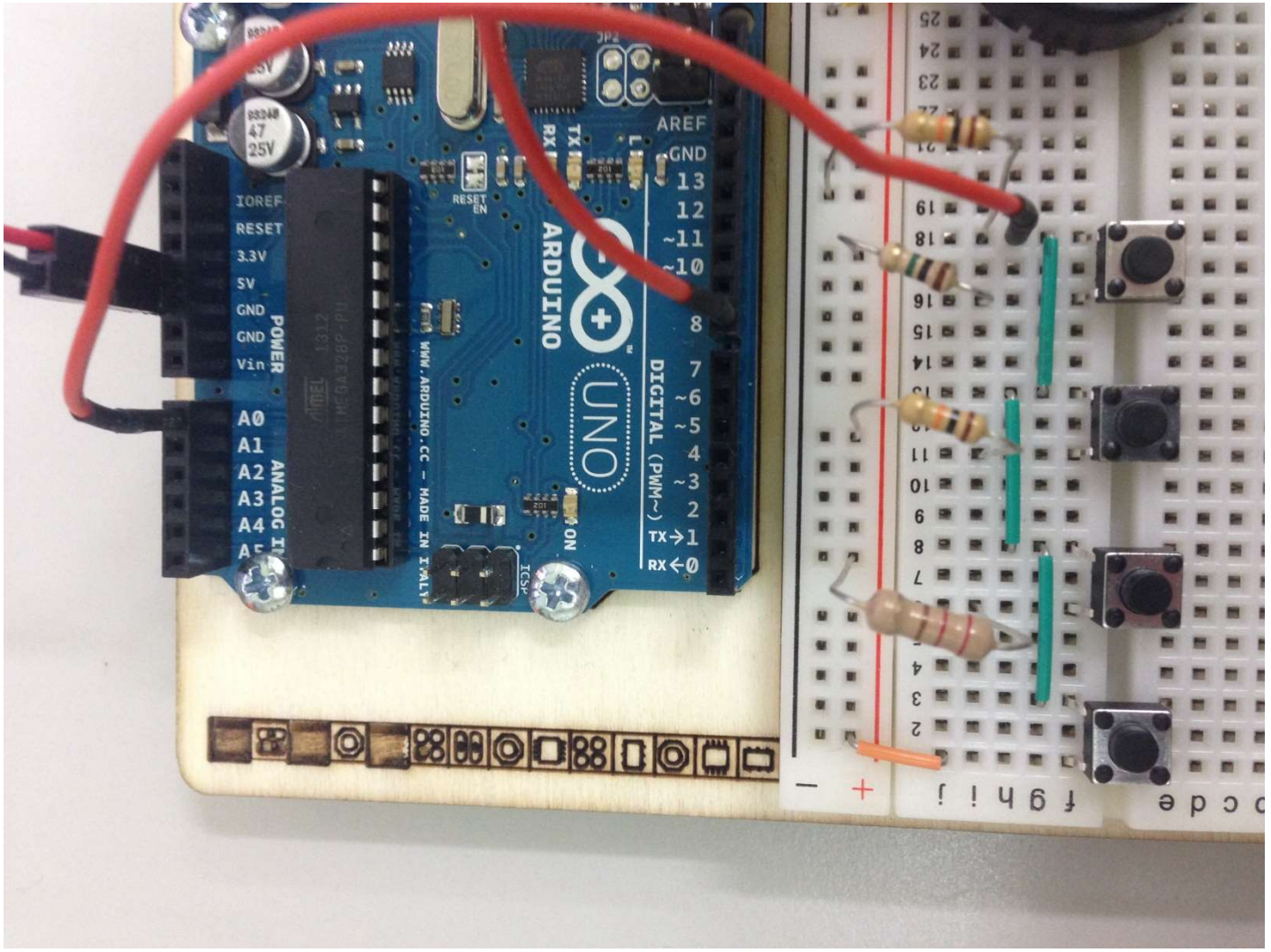
Resister Ladder



Schematic Diagram



The Circuit (cont.)



The Codes

```
int buttons[6];
// set up an array with 6 integers

int buttons[0] = 2;
// give the first element of the array the value 2
```

```
1 int notes[] = {262,294,330,349};
```

The Codes (cont.)

```
2  void setup() {
3    Serial.begin(9600);
4  }
```

```
5  void loop() {
6    int keyVal = analogRead(A0);
7    Serial.println(keyVal);
```

```
8    if(keyVal == 1023){
9      tone(8, notes[0]);
10   }
```

The Codes (cont.)

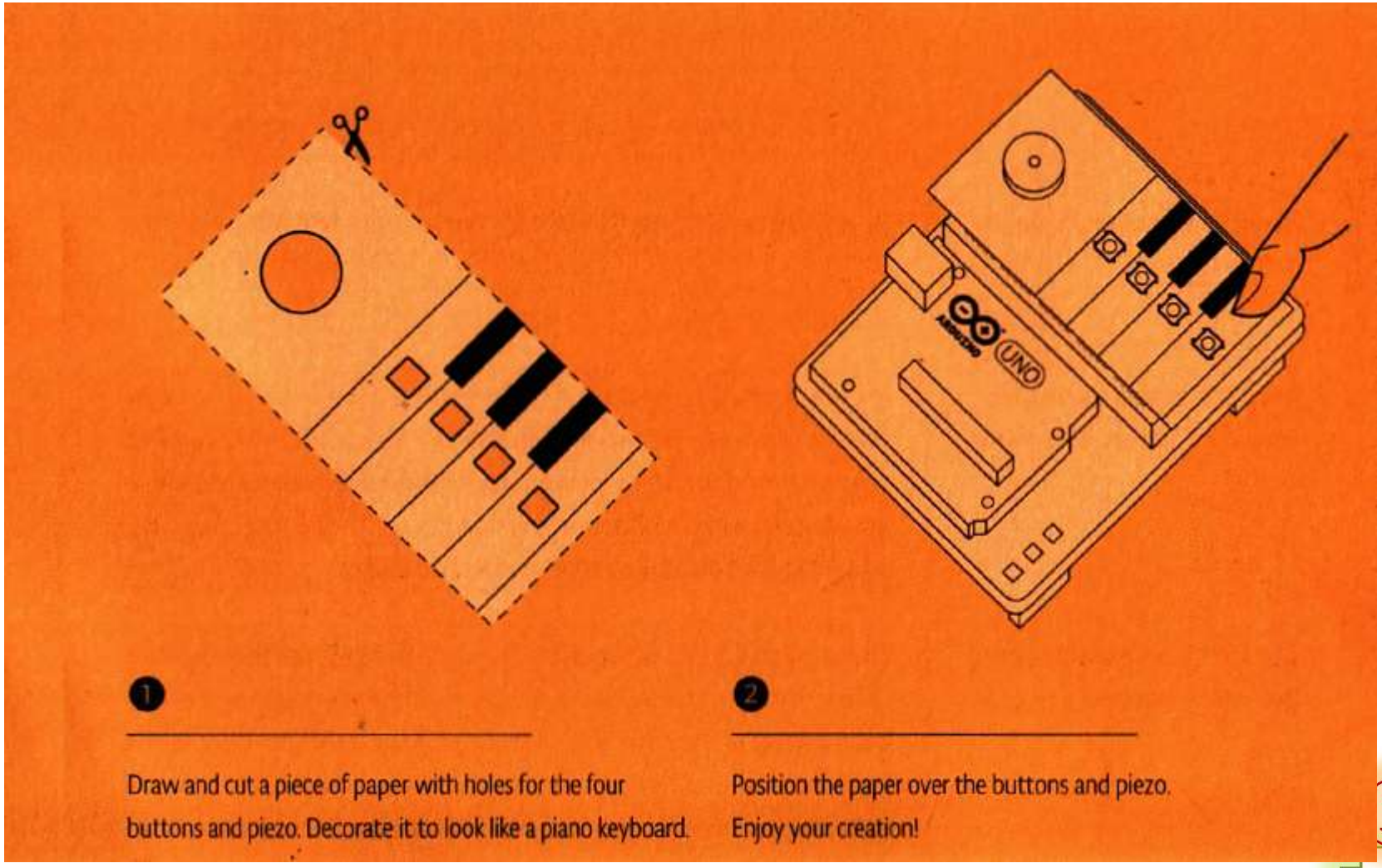
```
11    else if(keyVal >= 990 && keyVal <= 1010){
12      tone(8, notes[1]);
13    }
14    else if(keyVal >= 505 && keyVal <= 515){
15      tone(8, notes[2]);
16    }
17    else if(keyVal >= 5 && keyVal <= 10){
18      tone(8, notes[3]);
19    }
```

```
20    else{
21      noTone(8);
22    }
23  }
```

Codes


sketch_oct12a | Arduino 1.0.5
File Edit Sketch Tools Help
sketch_oct12a §
p08_DigitalHourglass.ino
5
Arduino Uno on COM6

Finish Up


UNO
1
Draw and cut a piece of paper with holes for the four buttons and piezo. Decorate it to look like a piano keyboard.
2
Position the paper over the buttons and piezo. Enjoy your creation!

Project 8

Digital HourGlass

Discover: Long data type, creating a timer

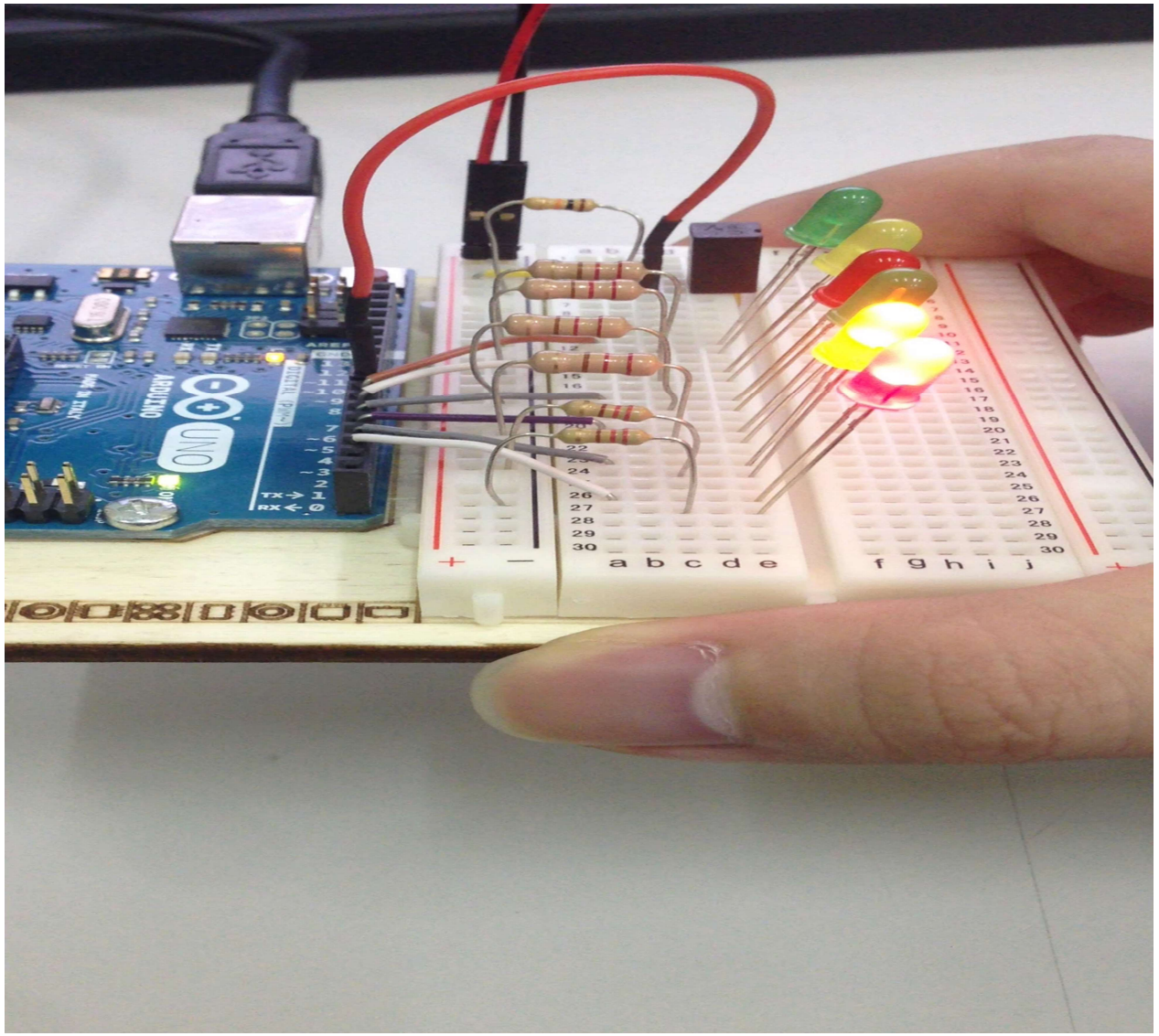
Introduction

- Up to now, when you've wanted something to happen at a specific time interval with the Arduino, you've used **delay**.
  - handy but a little confining.
- When the Arduino calls delay(), it **freezes its current state** for the duration of the delay.
  - That means there can be no other input or output while it's waiting.
  - Delay() is not very helpful for keeping track of time.
    - If you wanted to do something every 10 seconds, having a 10 second delay would be fairly cumbersome.

Introduction

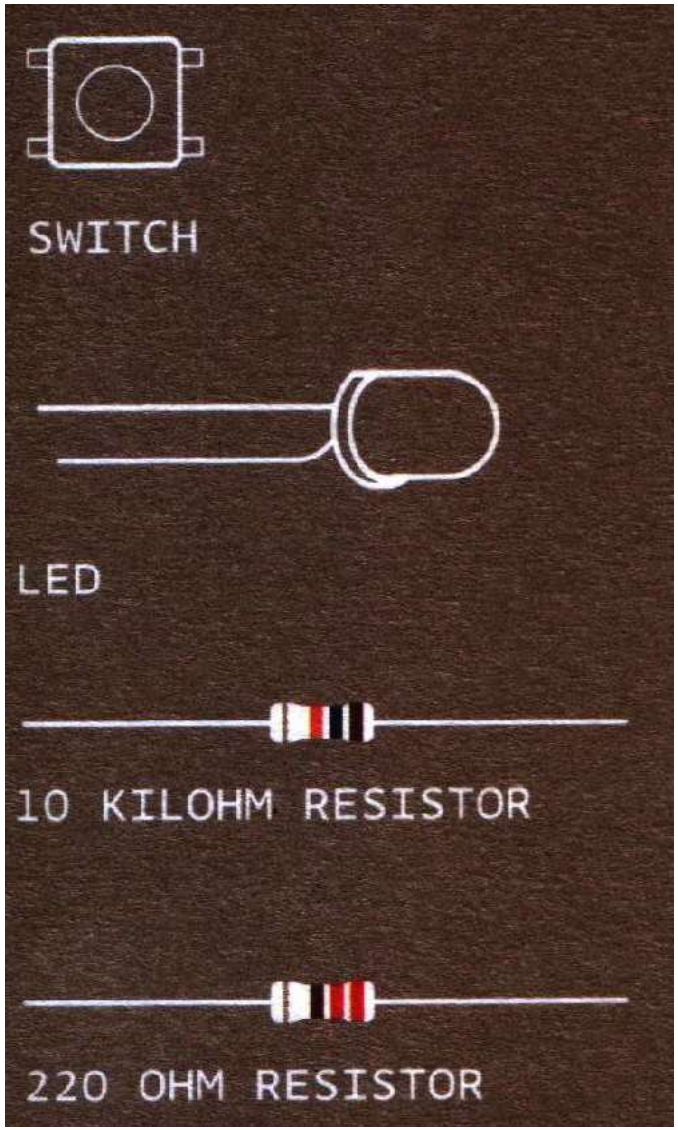
- The **millis**() function, keeps track of the time your arduino has been running in **milliseconds**.
  - Int:16-bit number (-32,768 and 32,768)
  - millis() needs long data type: a **32-bit number**(-2,147,483,648 and 2,147,483,648
    - his **unsigned version** holds number between 0 and 4,294,967,295.

Demo



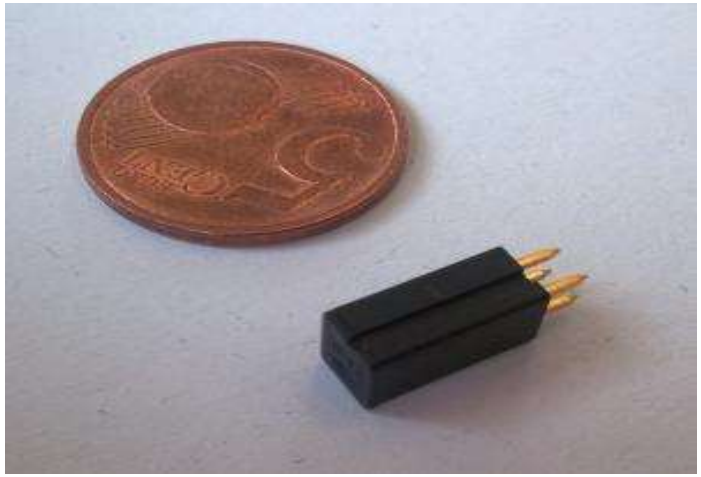
Ingredients

- Tilt Sensor (傾斜感測器)，或水銀開關
- 六個LED燈
- 六個220 Ω電阻
- 一個10K Ω電阻

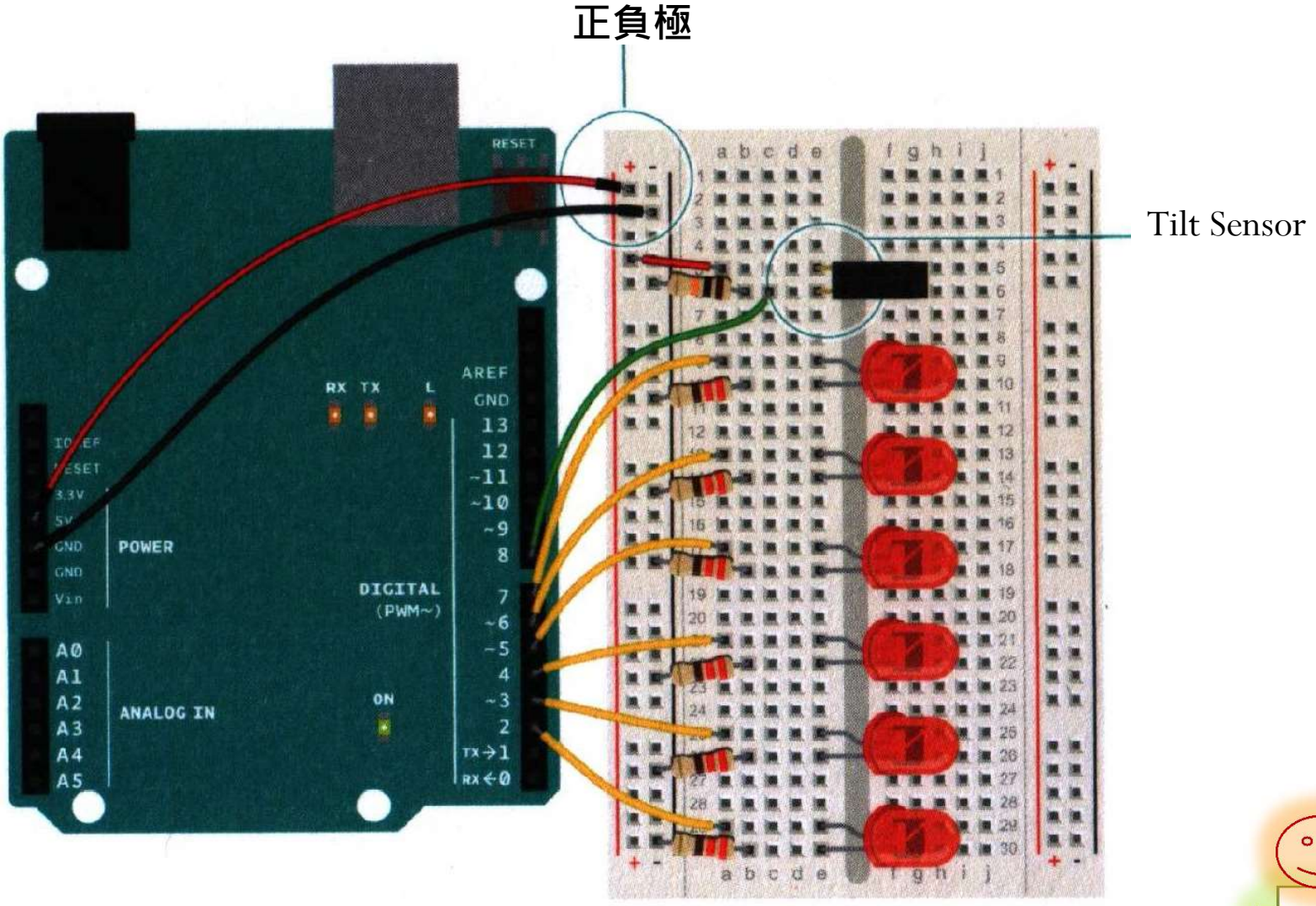


Tilt Sensor

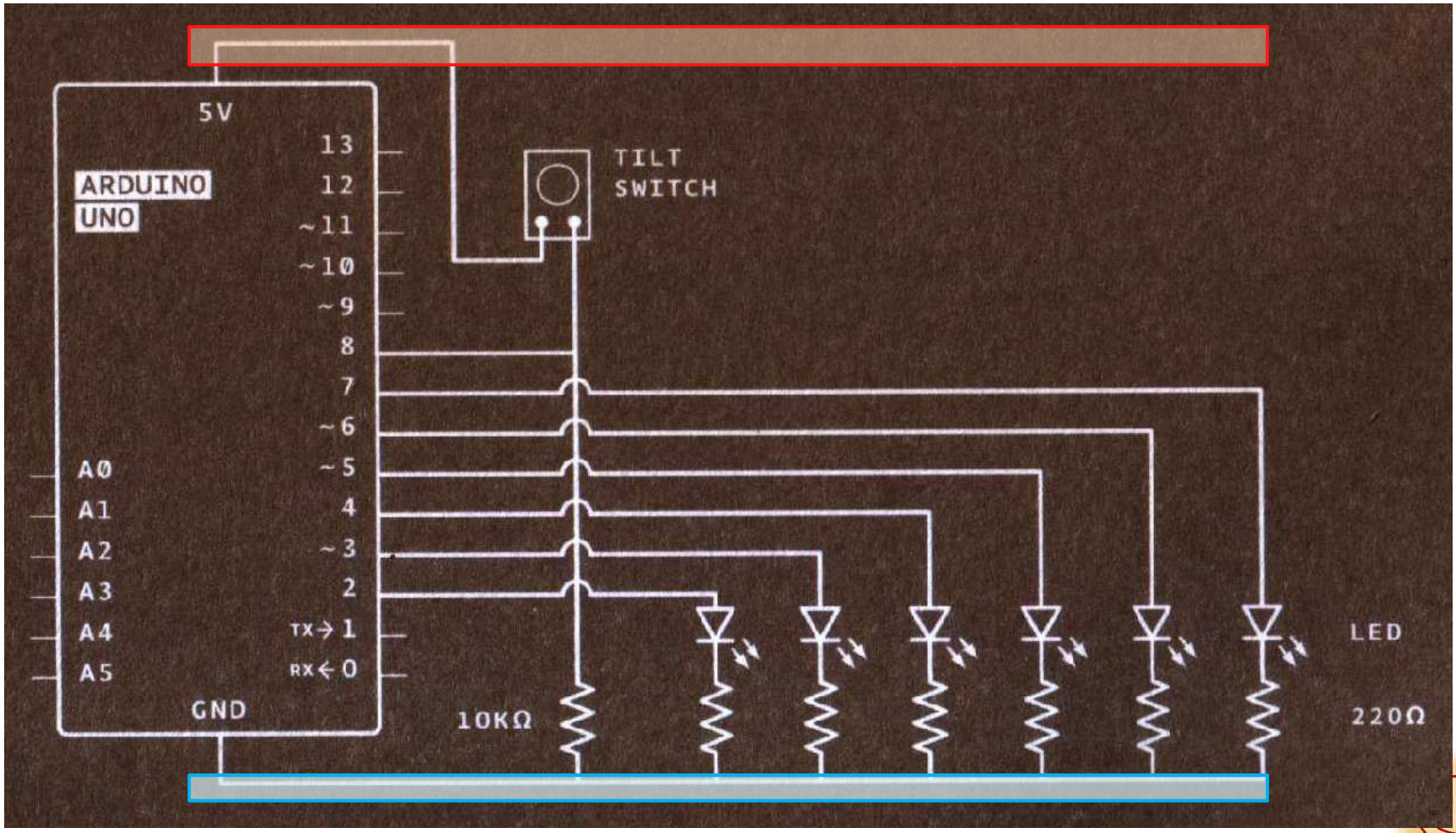
- The tilt sensor is a sensor with a **small metal ball** inside itself, and when you move it, he change his state.
  - like an accelerometer (a tilt sensor it self), but **less expensive** and give less information (only up/down).
  - can detect the tilting of an object.
- This type of sensor is the **environmental-friendly** version of a mercury-switch.



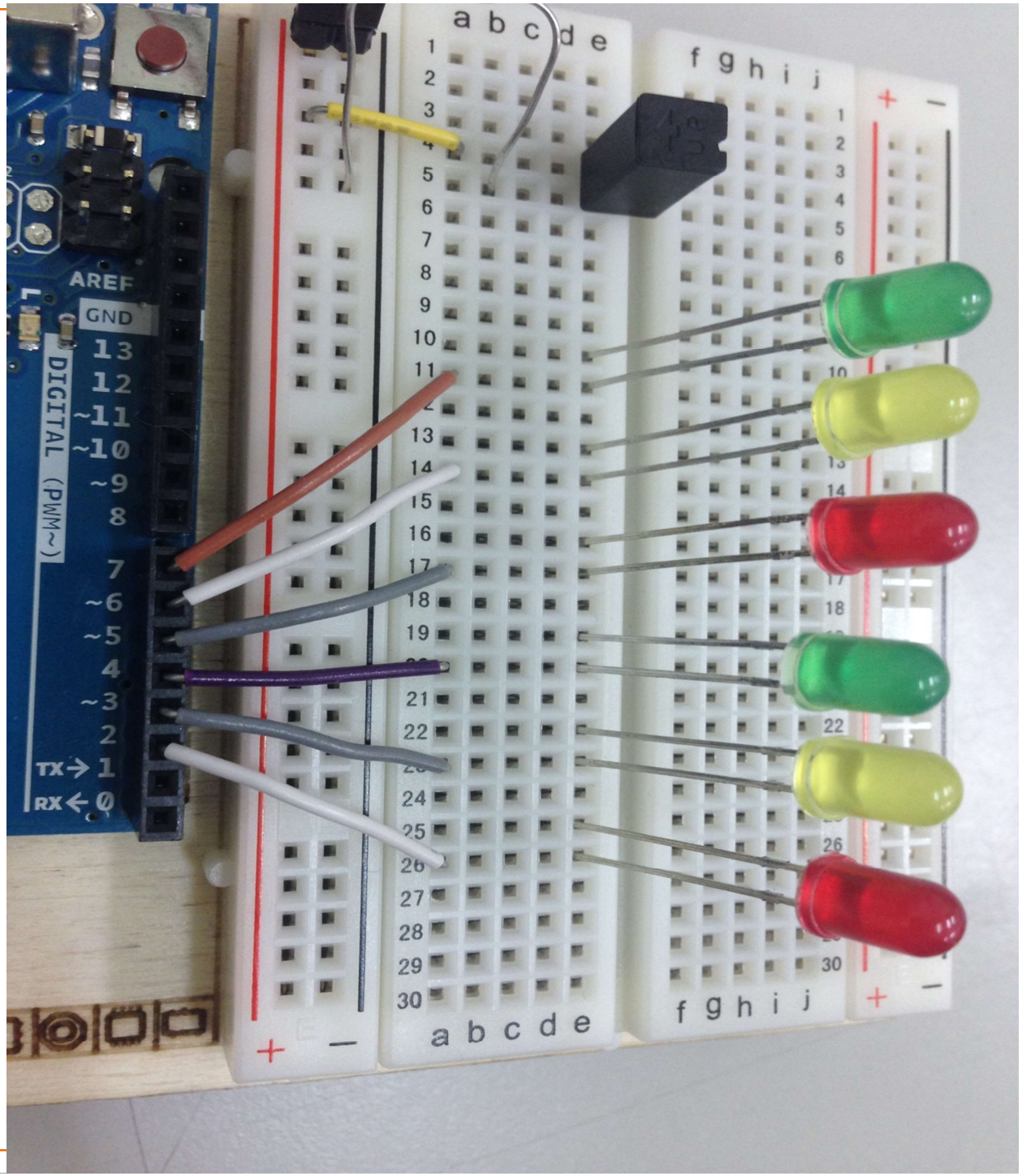
The Top View of the Circuit



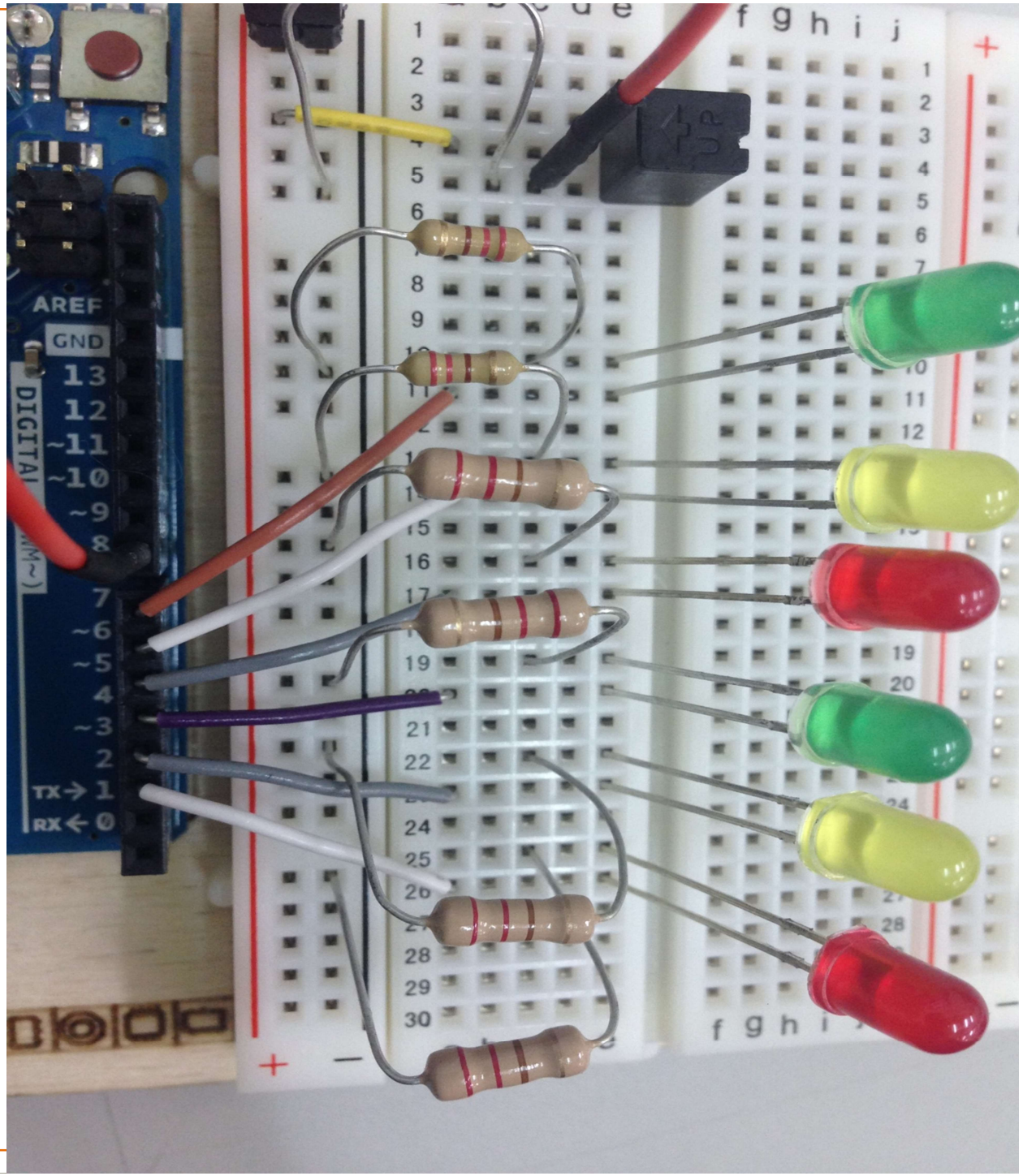
Schematic Diagram



The Circuit



The Circuit



The Codes

```
1 const int switchPin = 8;

2 unsigned long previousTime = 0;

3 int switchState = 0;
4 int prevSwitchState = 0;


5 int led = 2;
```

The Codes (cont.)

```
6 long interval = 600000;
```

```
7 void setup() {
8   for(int x = 2;x<8;x++){
9     pinMode(x, OUTPUT);
10  }

11  pinMode(switchPin, INPUT);
12 }
```

The Codes (cont.)

```
13 void loop(){
14   unsigned long currentTime = millis();

15   if(currentTime - previousTime > interval) {
16     previousTime = currentTime;

17     digitalWrite(led, HIGH);
18     led++;
```

The Codes (cont.)

```
19      if(led == 7){
20      }
21  }
```

```
22  switchState = digitalRead(switchPin);
```

The Codes (cont.)

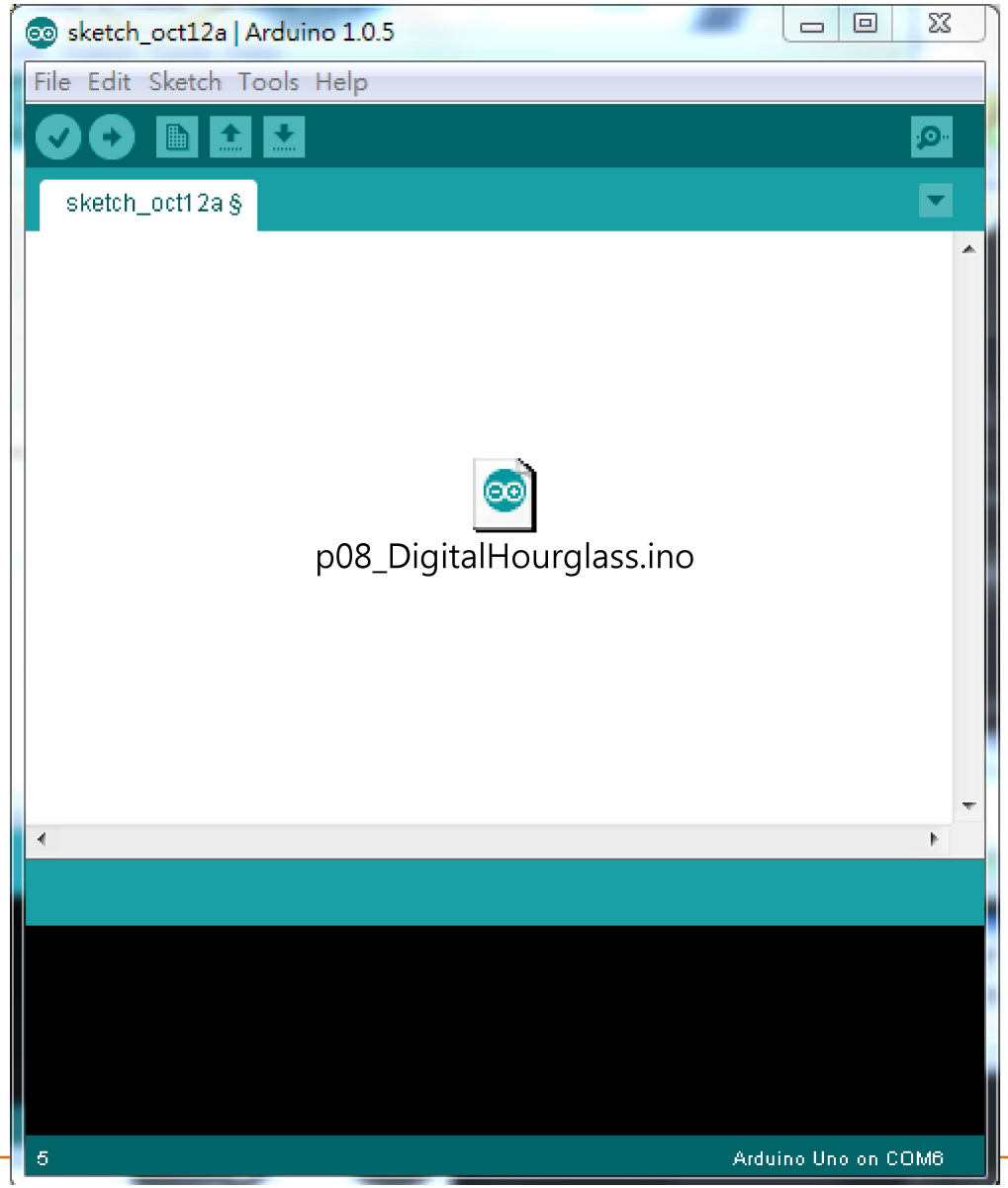
```
23   if(switchState != prevSwitchState){
24     for(int x = 2;x<8;x++){
25       digitalWrite(x, LOW);
26     }

27     led = 2;
28     previousTime = currentTime;
29   }
```

```
30   prevSwitchState = switchState;
31 }
```

Caution

- You need to change the time at which the led turns on
  - because the original code checks if every led is on every **600000** millisecond, i.e.,10 minute.
  - The responsiveness will be poor ➔ change the variable to 60000 which is only one minute, and a led turns on every 10 seconds.

Codes


sketch_oct12a | Arduino 1.0.5
File Edit Sketch Tools Help
sketch_oct12a §
p08_DigitalHourglass.ino
5
Arduino Uno on COM6