An Introduction to Arduino
a low cost digital prototyping platform

Clive Barry, Mogue Carpenter, Aileen Drohan, David Kirwan, Anton Krug, and Martin Walshe

South East Makerspace,
Old Printworks, Thomas Hill,
Waterford City, X91 TW63
info@southeastmakerspace.org
https://www.southeastmakerspace.org

Abstract. This workshop offers an introduction to the Arduino prototyping platform, useful for artists, hobbiests and the wider maker community. Each participant receives a pack containing an Arduino Uno and components to perform each of 5 simple experiments. No prior coding experience is needed as all code will be provided. The only requirement is the Arduino IDE, and a packed lunch!

Keywords: arduino, digital electronics, prototyping, introduction

Table of Contents

Introduction

What is an Arduino?

"Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike." [Arduino, 2015a]

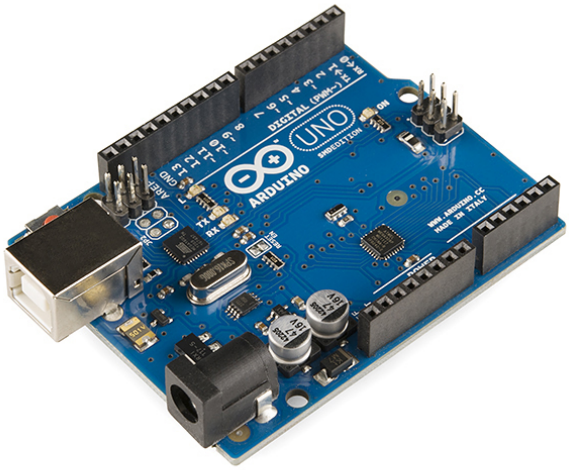


Fig. 1: Arduino Uno R3 [Wikipedia, 2013]

Open Source

Open Source can be described as resources and software that can be used, redistributed, rewritten and in some circumstances sold free of charge. The schematics and component list for hardware can also be open source as is the case with the Arduino platform [Culkin, 2011].

Electronics
: Technology which makes use of the controlled motion of electrons through different media [Culkin, 2011].

Prototype
: An original form that can serve as a basis or standard for other things [Culkin, 2011].

Platform
: Hardware architecture with software framework on which other software can run [Culkin, 2011].

Where might an Arduino be used?

A few contrived examples of where one might use an Arduino in order to automate some process:

Automatic Dog's Water Bowl

An dog owner wants to ensure her pet is never left without water. She attaches a system for measuring the water level in the dog's bowl. Her Arduino is programmed to measure this value every 5 minutes. If this level falls below a certain value, a valve is opened and a water pump is activated to fill up the water bowl. It then sends an SMS to the owner to let her know that the dog is in safe hands.

- Read inputs from a water level sensor
- Control a valve which lets water flow
- Control speed of a water pump
- Send SMS to owner about giving the dog water

Bird Table Camera

A rising social media ornithologist wishes to share pictures from all the visitors to the bird table in his garden. He mounts an infra-red movement sensor on the bird table attached to an Arduino which is configured to record an image and send it to Twitter. His neighbours marvel at how many crows he's feeding.

- Read inputs from a movement sensor
- Control a camera shutter
- Transmit image back to PC
- Send tweet with picture of the bird table visitor

Fingerprint Door Lock

A student is sick of forgetting his keys and being locked out of his house. He uses a fingerprint scanner and an Arduino to make a biometric fingerprint door lock. He needs only scan his thumb print now and the door will unlock.

- Read inputs from a fingerprint sensor
- Compares the finger print against an authorised fingerprint
- Records the time and date a finger was pressed on the scanner
- Makes audio error tone if the fingerprint was invalid
- If valid fingerprint it unlocks the door

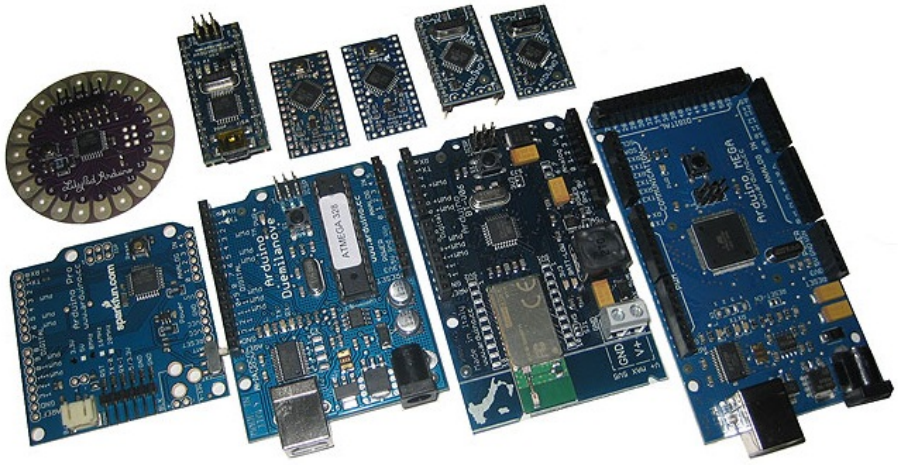


Fig. 2: Arduino Family

There are multiple different flavours of Arduino, each with different capability levels and or specialised features. It is through this variety which makes it is possible to use the Arduino in a wide range of situations from home automation to wearable technology.

Workshop Aims

In this workshop the aim is to give you a crash course in digital electronics, and provide you with the basic skill-set to start using the Arduino or similar micro-controllers in your future projects.

Workshop Requirements

Each person will require the following:

- PC, either Linux, Mac or Windows can be used
- Arduino IDE pre-installed (Internet at the makerspace is flaky!)
- A sambo to keep you going

Learning Outcomes

Each person will leave with:

- Arduino starter kit
- Crash course in digital electronics
- Confidence to use Arduino in future projects

Arduino Starter Kit Contents

The Arduino starter kit contains the following components, which we will be making use of during the workshop.

- 1 × Arduino Compatible R3 Uno
- 1 × Breadboard
- 16 × jumper wires various colours
- 20 × 5mm LED's assorted colours
- 10 × 10kΩ resistors
- 10 × 330Ω resistors
- 1 × RGB LED
- 1 × photo resistor
- 2 × push buttons
- 1 × DFRobot MP3 Module
- 1 × 1GB SD Card
- 1 × 5Ω speaker

Basic Circuit Theory

In an electrical circuit there is a fundamental relationship between voltage, current and resistance and it is explained by Ohms Law [ElectronicsTutorials, 2015].

Voltage

Voltage, (SI Unit: V - Volts) is the potential energy of an electrical supply stored in the form of an electrical charge. Voltage can be thought of as the force that pushes electrons through a conductor and the greater the voltage the greater is its ability to push the electrons through a given circuit [ElectronicsTutorials, 2015].

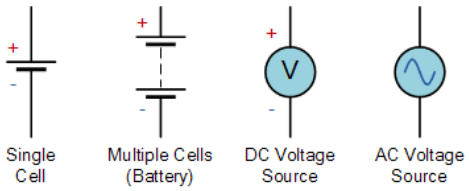


Fig. 3: Voltage Symbols [ElectronicsTutorials, 2015]

Current

Current, (SI Unit: A - Ampere) is the movement or flow of electrical charge and is measured in Amperes. It is the continuous and uniform flow (called a drift) of electrons (the negative particles of an atom) around a circuit that are being pushed by the voltage source [ElectronicsTutorials, 2015].

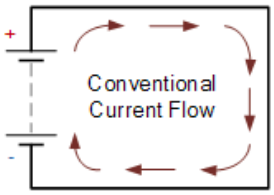


Fig. 4: Current Symbols [ElectronicsTutorials, 2015]

Resistance

Resistance, (SI Unit: Ω - Ohms) of a circuit is its ability to resist or prevent the flow of current (electron flow) through itself making it necessary to apply a greater voltage to the electrical circuit to cause the current to flow again. Note that Resistance cannot be negative in value only positive [ElectronicsTutorials, 2015].

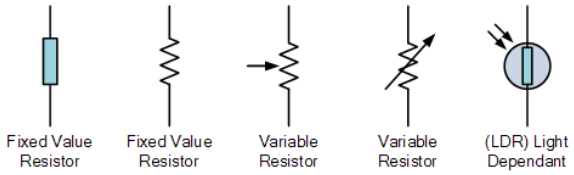


Fig. 5: Resistance Symbols [ElectronicsTutorials, 2015]

Ohm's Law

The following equation Ohm's Law explains the relationship between Voltage, Current and Resistance for an electrical circuit. For more information see: [ElectronicsTutorials, 2014].

$$V = I \times R \tag{1}$$

$$I = \frac{V}{R} \tag{2}$$

$$R = \frac{V}{I} \tag{3}$$

Fig. 6: Ohm's Law

Analogue vs Digital Signals

The world we live in is an analogue one. There are an infinite number of possible combinations of colours smells and sounds. The digital world however is not infinite. It is discrete and finite where we are limited by several factors, such as

memory and computational capabilities. In order to represent a real world thing digitally, it is by necessity that some information is lost.

Analogue Signal Examples

The following are some examples of analogue signals:

- Temperature
- Sound
- EM Radiation

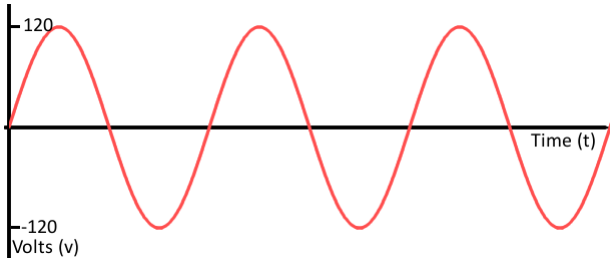


Fig. 7: Analogue Signal [Lindblom, 2015]

Digital Signal Examples

The following are some examples of digital signals:

- Morse Code
- WIFI
- Binary

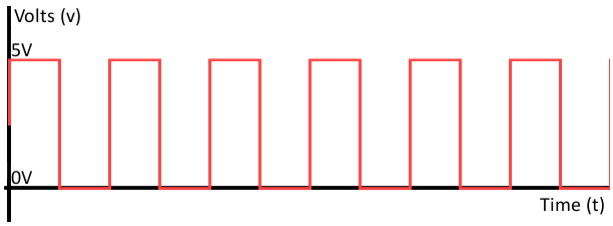


Fig. 8: Digital Signal [Lindblom, 2015]

Electronic Components

The following section contains more information on some of the various components included in the *Introduction to Arduino* workshop.

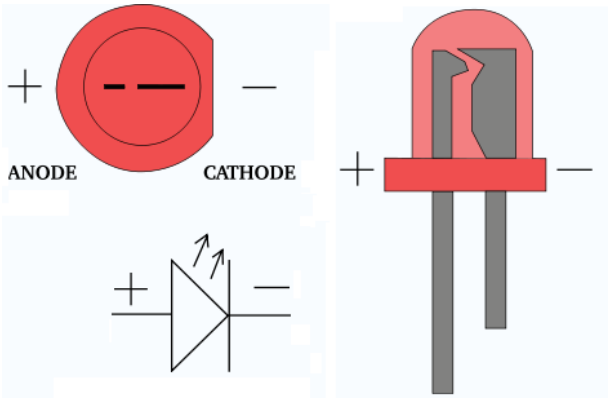


Fig. 9: LED Component [Sapkota, 2010]

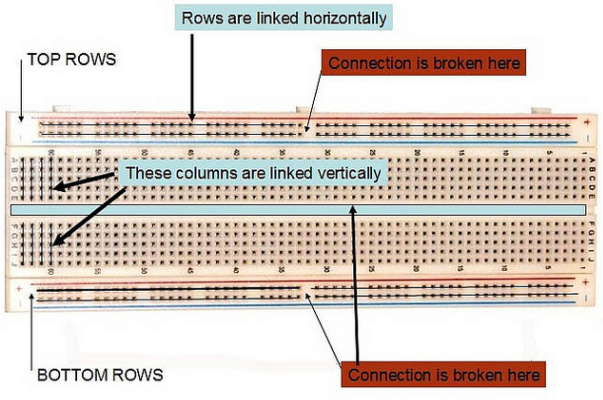
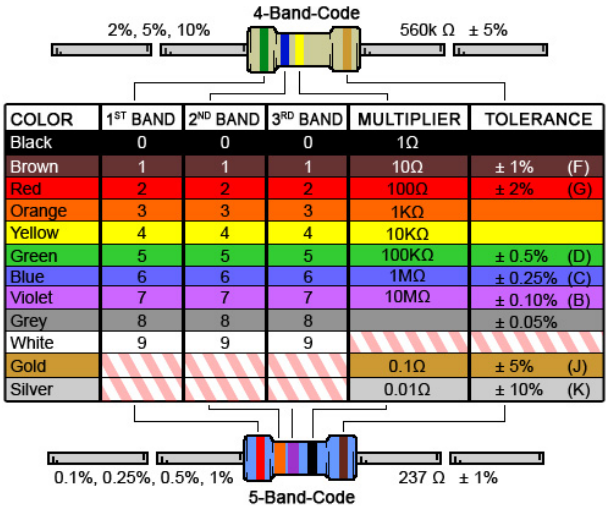


Fig. 10: Breadboard [Sapkota, 2010]



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		± 0.05%
White	9	9	9		
Gold				0.1Ω	± 5% (J)
Silver				0.01Ω	± 10% (K)

Fig. 11: Resistor Chart [Digikey, 2016]

Basic Arduino Coding Concepts

Variables

A variable is a way of naming and storing a value for later use by the program, such as data from a sensor or an intermediate value used in a calculation [Arduino, 2015b]. Table: 1 shows some of the basic variable

Type	Example
char	'a'
string	'Hello there'
byte	0 to 255
int	-32,768 to 32,767
unsigned int	0 to 65,535
long	-2,147,483,648 to 2,147,483,647
unsigned long	0 to 4,294,967,295
float	-3.4028235E38 to 3.4028235E38

Table 1: Variable Types in Wiring

types available to us for use in the Arduino. There are other more advanced data types such as Arrays and Pointers, which we may use, but

won't explain them in this introduction. More information can be found regarding these data types online. See the Arduino online reference library at: https://www.arduino.cc/en/Reference/HomePage.

setup() function

The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup function will only run once, after each power-up or reset of the Arduino board [Arduino, 2015c].

```
void setup(){ // setup initializes serial
  Serial.begin(9600);
}
void loop(){ // write to serial, then wait 2 seconds
  Serial.write("Hello World");
  delay(2000);
}
```

loop() function

After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board [Arduino, 2015d].

```
// Create int variable to represent
// a physical pin on the Arduino
int buttonPin = 3;

// setup initializes serial and the button pin
void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

// loop checks the button pin each time,
// and will send serial if it is pressed
void loop()
{
  if (digitalRead(buttonPin) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');

  delay(1000);
}
```

Experiment 1 - Blink

This example is what one might call the *hello world* example for Arduino sketches. We wire up the experiment as shown in the diagram fig: 12. And upload the Sketch code in the next section on page: 15.

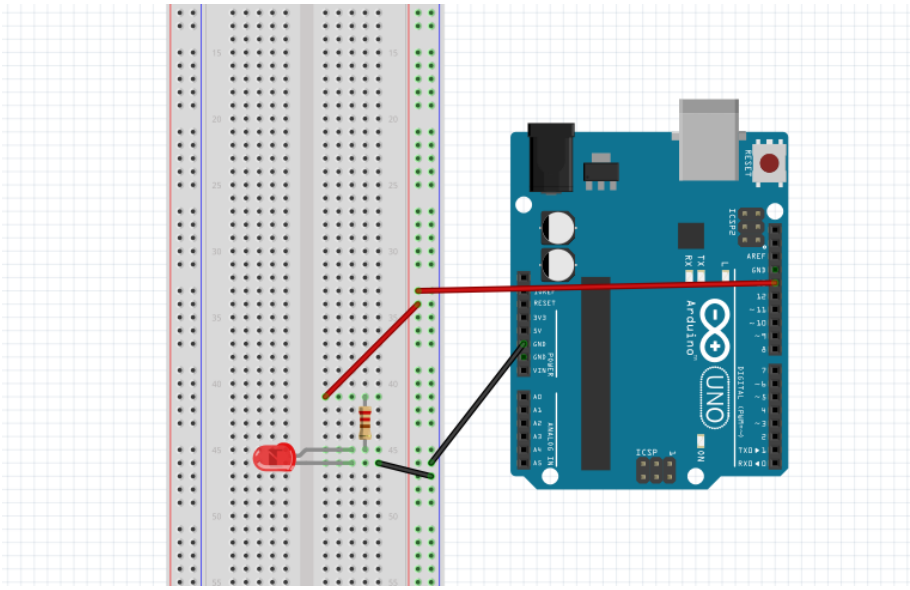


Fig. 12: LED Blink [Fritzing, 2015]

When the Arduino boots, the led should flash on for a second, then off for a second and repeat.

Sketch Code

```
/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.

This example code is in the public domain.
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  // turn the LED on
  // (HIGH is the voltage level)
  digitalWrite(13, HIGH);

  //wait for 1000 milliseconds
  delay(1000);

  // turn the LED off by making
  // the voltage LOW
  digitalWrite(13, LOW);

  // wait for 1000 milliseconds
  delay(1000);
}
```

Experiment 2 - Button

We wire up the experiment as shown in the diagram fig: 13. And upload the sketch code in the next section on page: 17.

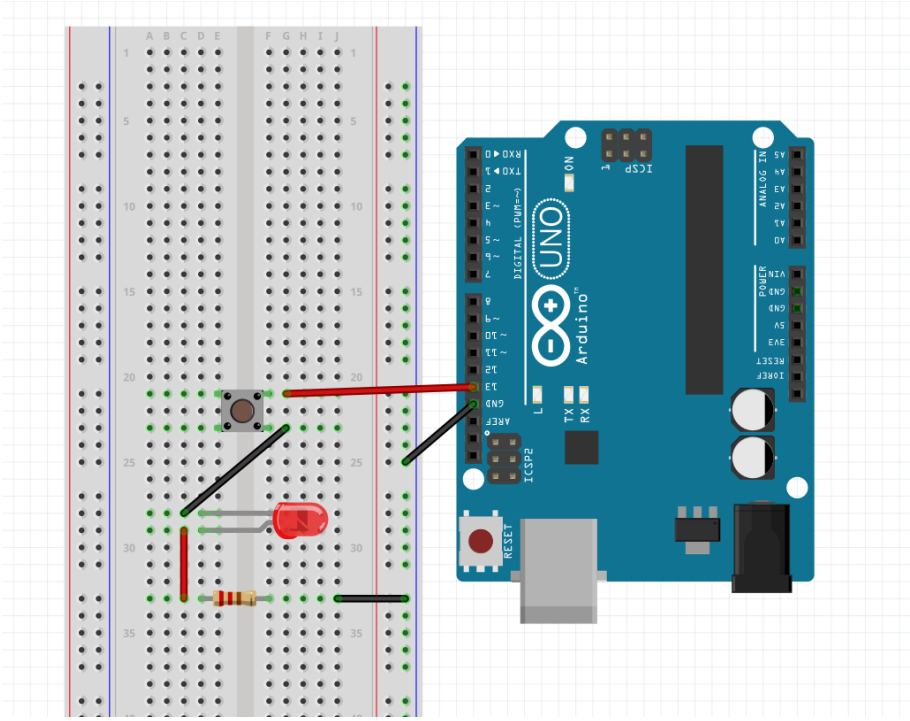


Fig. 13: Button activated LED [Fritzing, 2015]

The LED will light up when the button is pressed.

Sketch Code

```
/*
Button activated LED

This example code is in the public domain.
*/


// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
}
```

Experiment 3 - Generating Music

We wire up the experiment as shown in the diagram fig: 14. And upload the sketch code in the next section on page: 19.

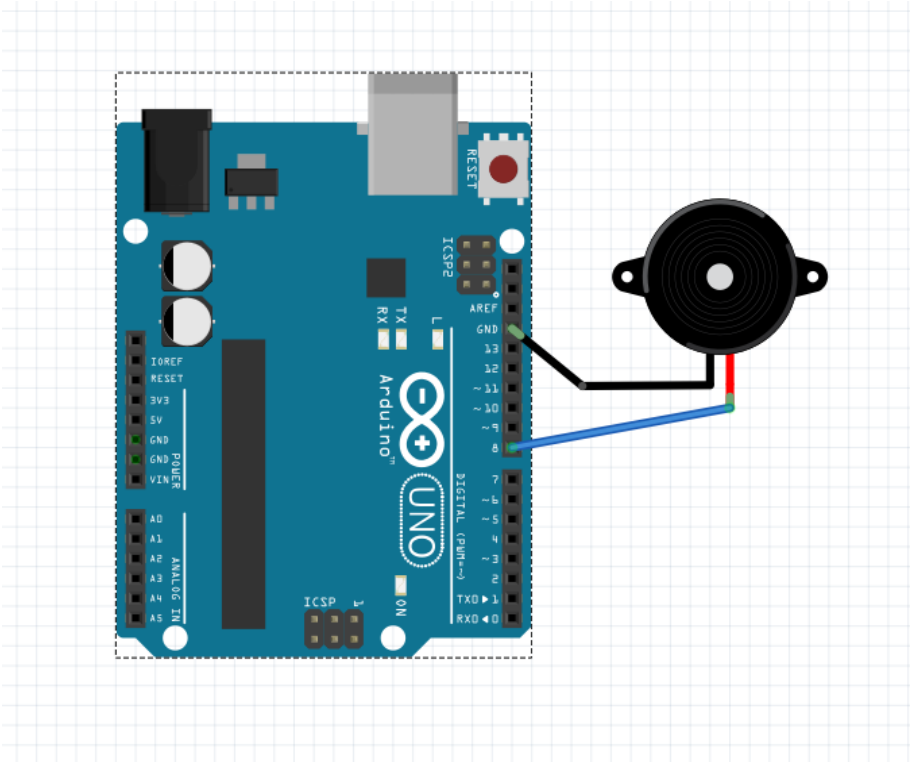


Fig. 14: Play musical notes through a speaker [Fritzing, 2015]

As soon as the Arduino boots it should play a series of musical notes and then stop. For more information see the online reference page for the functions used in this experiment here: https://www.arduino.cc/en/Reference/Tone

Sketch Code

```
/*
Melody - Plays a melody
This example code is in the public domain.
*/
#include "pitches.h"

int melody[] = { // notes in the melody:
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000 / noteDurations[thisNote];
    tone(8, melody[thisNote], noteDuration);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

```
// ********************* pitches.h ************************

/************************************************
* Public Constants
************************************************/

#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
```

```
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
```

```
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
```

Experiment 4 - Thermistor

We wire up the experiment as shown in the diagram fig: 15. And upload the sketch code in the next section on page: 24.

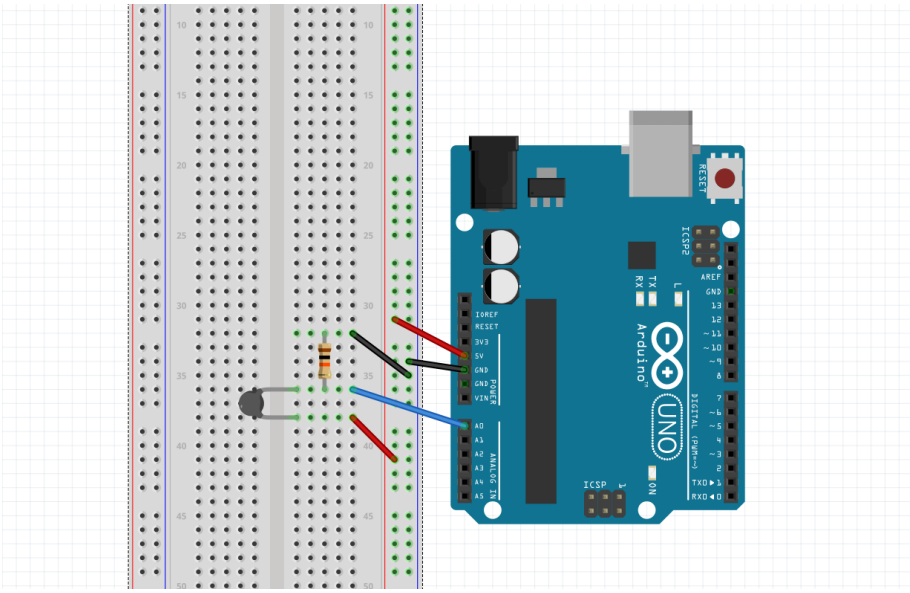


Fig. 15: Read a Thermistor Value

The thermistor is an electrical resistor whose resistance is greatly reduced by heating. Its often used for measurement and control applications. The following graph fig: 16 on page: 24. demonstrates the fall of resistance as the temperature rises.

To view the values being read from the thermistor, simply open the Arduino serial monitor.

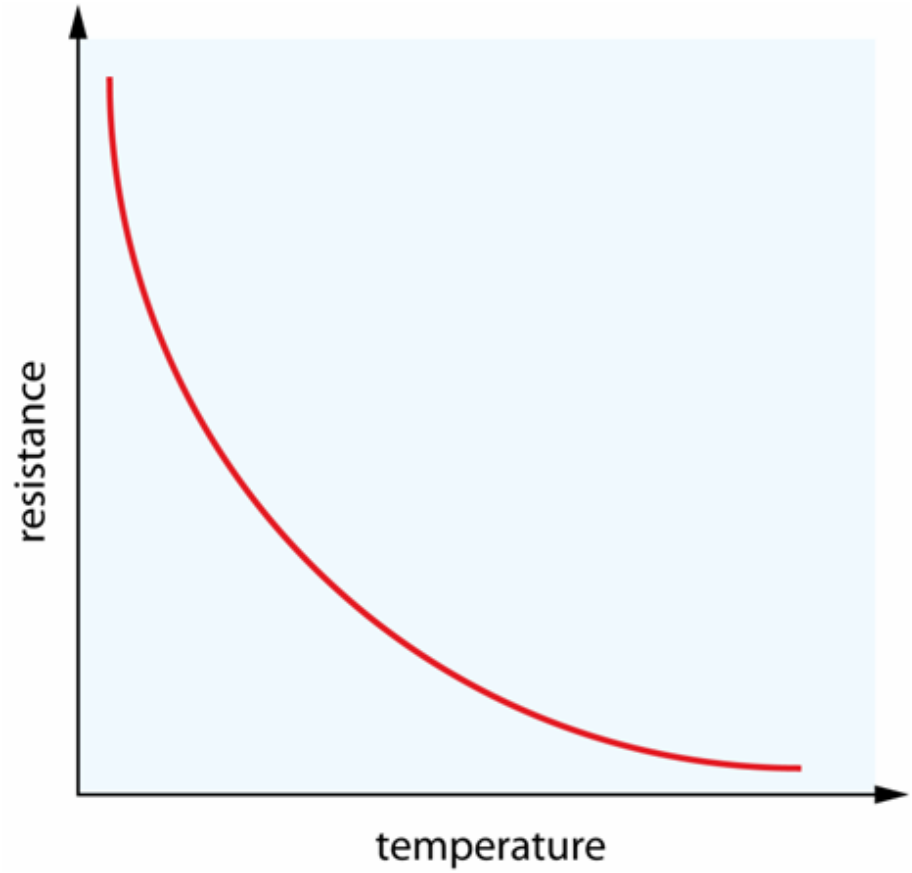


Fig. 16: Thermistor Plot: Resistance vs Temperature

Sketch Code

```
/*
Sample code to work with a Thermistor

Author: David Kirwan
Licence: Apache 2.0
*/

int sensorPin = A0;
int sensorValue = 0;
```

```
void setup() {
  // declare the ledPin as an OUTPUT:
  Serial.begin(9600);
  pinMode(sensorPin, INPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println(sensorValue);
  delay(sensorValue);
}
```

Experiment 5 - clap4Led

We wire up the experiment as shown in the diagram fig: 17. And upload the sketch code in the next section on page: 27.

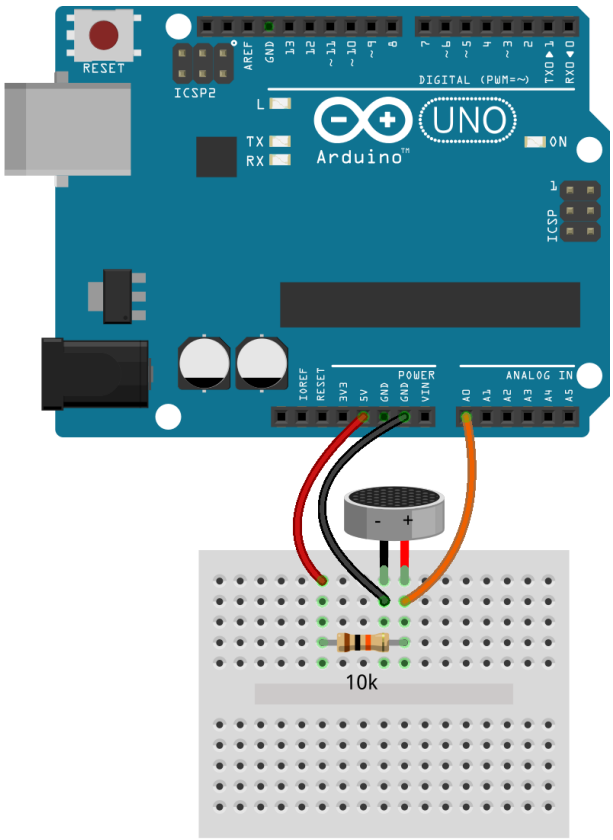


Fig. 17: Microphone Activated Circuit

The microphone module is polarised, which means it has a positive and negative terminal which must be correctly placed in the circuit in order to function as expected. Please examine the diagram fig: 18 on page: 27 to correctly identify which leg is positive and which is negative.

When this circuit is complete, the LED onboard the Arduino will flash when the microphone detects sound.

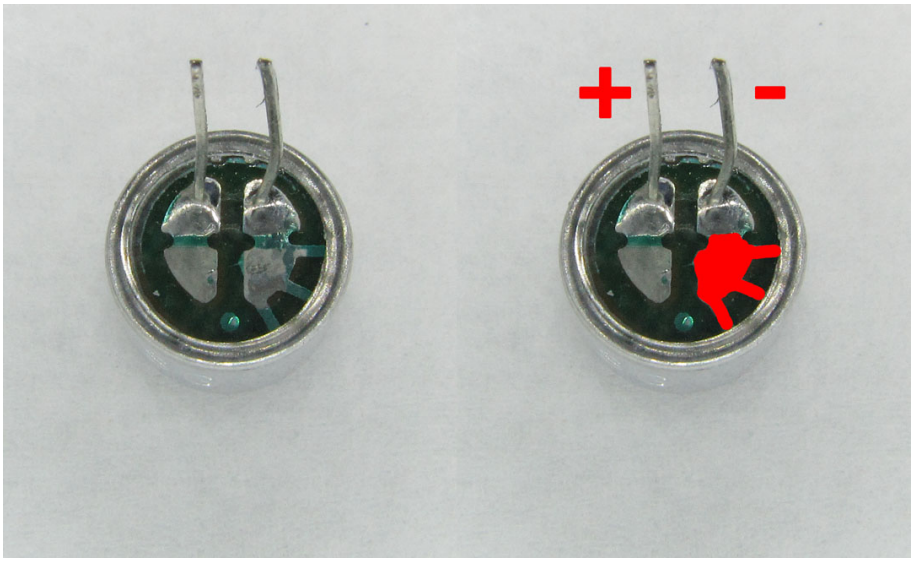


Fig. 18: Microphone Module Polarity

Sketch Code

```
/*
Sample code to work with a Thermistor

Author: Anton Krug
Licence: Apache 2.0
*/

//global variables
int readValue;
int maximumValue;
int minimumValue;


// will reset maximum and minimum values only when we call it
void resetMaximumAndMinimum()
{
  maximumValue = 0;             // set maximum to lowest value possible
  minimumValue = 32767;         // set minimum to highest value possible
```

```
}


// this runs only once on the startup
void setup()
{
  // initialize built-in LED light at digital pin 13 as an output
  pinMode(13, OUTPUT);
  resetMaximumAndMinimum();
}


// the loop runs over and over again forever
void loop()
{
  // read current value from the microphone
  readValue = analogRead(A0);

  // if read value is smaller than minimum then set it as the new minimum
  if (readValue < minimumValue)
  {
    minimumValue = readValue;
  }

  // if read value is bigger than maximum then set it as the new maximum
  if (readValue > maximumValue)
  {
    maximumValue = readValue;
  }

  // Change the 10 constant to adjust the sensitivity.
  //   To 20 if you want the light triggered on louder claps
  //   Ti 5  if you want the light triggered on quieter noises
  //   Feel free to experiment with the values.
  if ( (maximumValue - minimumValue) > 10 )
  {
```

```
    digitalWrite(13, HIGH);         // turn the LED on (HIGH is the voltage lev
    delay(2 * 1000);                // wait for 2 seconds (2 * 1000ms)
    digitalWrite(13, LOW );         // turn the LED off by making the voltage LO

    // if we wouldn't clear the max & min then after first trigger it would g
    // triggered every single time no matter what read values were
    resetMaximumAndMinimum();
  }
}
```

Thanks for taking part!

Do you want to know more? Of course you do! Fried [2012] has developed a number of tutorials suitable for the beginner wishing to learn more about the Arduino system. These tutorials can be accessed at the following URI: http://www.ladyada.net/learn/arduino/

Would you like to work on or with technology like this in the future? Would you like to change profession? Then you might also be interested in investigating the newly formed *Internet of Things* degree course being run at the Waterford Insstitute of Technology: https://www.wit.ie/courses/school/science/department_of_computing_maths_physic
hons-in-the-internet-of-things

I hope you enjoyed this workshop and your time spent here. The South East Makerspace has developed several workshops in the IoT space and hope to continue creating more in the future. Be sure to sign up to the mailing list, and follow us on social media to keep informed about our events and workshops! We would be delighted to welcome new faces, however full membership is only open to those over 18 years old (insurance reasons)! For information on SEMS see:

Facebook http://www.facebook.com/SouthEastMakerSpace
FAQ https://wiki.southeastmakerspace.org/faq
Google+ http://plus.google.com/u/0/108025738894009906004/
How to join https://wiki.southeastmakerspace.org/how_to_join_sems
Mailing List http://lists.southeastmakerspace.org/mailman/listinfo/sems-general
Secure Contact https://www.southeastmakerspace.org/contact
Twitter http://twitter.com/SEMakerSpace

Glossary

Arduino an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects. https://www.arduino.cc. 3–5, 14, 18, 29

. 26

Sketch a sketch is the name that Arduino uses for a program. Once compiled the resulting firmware can be installed on the Arduino device. https://www.arduino.cc/en/Tutorial/Sketch. 14

Bibliography

Arduino. What is arduino?, 2015a. URL https://www.arduino.cc/en/Guide/Introduction. Accessed: 2015-10-18.

Arduino. Variables, 2015b. URL https://www.arduino.cc/en/Reference/VariableDeclaration. Accessed: 2015-10-18.

Arduino. setup(), 2015c. URL https://www.arduino.cc/en/Reference/Setup. Accessed: 2015-10-18.

Arduino. loop(), 2015d. URL https://www.arduino.cc/en/Reference/Loop. Accessed: 2015-10-18.

Jody Culkin. Arduino, 2011. URL http://www.jodyculkin.com/wp-content/uploads/2011/09/arduino-comic-latest3.pdf. Accessed: 2015-10-24.

Digikey. 4 band resistor color code calculator, 2016. URL http://www.digikey.ie/en/resources/conversion-calculators/conversion-calculator-resistor-color-code-4-band. Accessed: 2016-02-11.

ElectronicsTutorials. Resistors in series, 2014. URL http://www.electronics-tutorials.ws/resistor/res_3.html. Accessed: 2015-10-18.

ElectronicsTutorials. Relationship between voltage, current and resistance, 2015. URL http://www.electronics-tutorials.ws/dccircuits/dcp_1.html. Accessed: 2015-10-18.

Limor Fried. Arduino, 2012. URL http://www.ladyada.net/learn/arduino/. Accessed: 2015-10-24.

Fritzing. Fritzing - electronics made easy, 2015. URL http://fritzing.org/home/. Accessed: 2015-10-23.

Jim Lindblom. Analog vs. digital, 2015. URL https://learn.sparkfun.com/tutorials/analog-vs-digital. Accessed: 2015-10-18.

Sagar Sapkota. Building a circuit on a breadboard, 2010. URL http://www.buildcircuit.com/building-a-circuit-on-breadboard/. Accessed: 2016-02-10.

Wikipedia. Arduino uno r3, 2013. URL https://commons.wikimedia.org/wiki/File:Arduino_Uno_-_R3.jpg. Accessed: 2015-10-18.