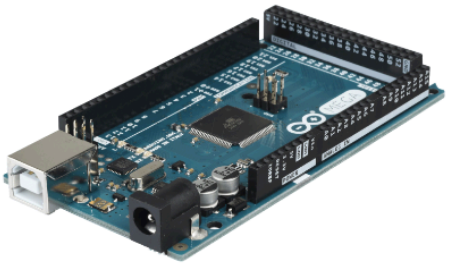
I2E101: Introduction to Electronics & Programming

by doing hands-on **fun** projects

No prior knowledge of electronics is required. The course starts with basics and builds up on concepts with every class. By the end of this course, students will be able to program Arduino and do amazing projects!

Sessions: 8 Classes
Requirement: Laptop for programming



Material Cost: $120 (Approximate)

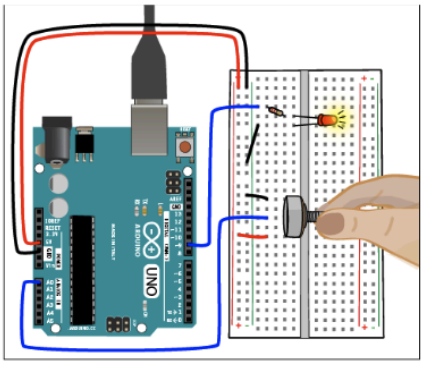
- Arduino Kit: https://amzn.to/2HfEIck
- Multimeter: https://amzn.to/2qIvn6p
- Presentation Board: https://amzn.to/2J9ZnyH
- Project material, etc.

Topics covered:

- Start with basics: Charge, Electricity, Ohm's Law, Circuits
- Learn all different building blocks: Resistors, Diodes, LEDs, Capacitors, Potentiometer, Photoresistor, Transistors (NPN, PNP), Electromagnets, Relays and more. Learn to use a digital Multimeter.
- Introduction to Arduino Microcontroller and Programming.
- Use all different types of sensors and create your own amazing projects

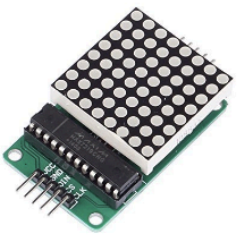
Curriculum & Projects

- Chapter 1: Introduction to Electricity, Ohm's Law. I.V.R.
- Chapter 2: Understanding Electromagnets. Multimeter
- Chapter 3: Learn PNP and NPN Transistors
- Chapter 4: Introduction to Relays
- Chapter 5: Introduction to Arduino PLC
- Chapter 6: Programming with Arduino
- Chapter 7: Microphone / Sound Sensor, PIR Motion Sensor
- Chapter 8: Burglar Alarm / Buzzer, LED Display



Optional Purchases:

- Programming with Arduino: https://amzn.to/2vrHc5F
- More Sensor Modules: https://amzn.to/2EXZESZ
- OLED Display: https://amzn.to/2qIF053 (http://www.diymalls.com/files/IIC_OLED.zip)



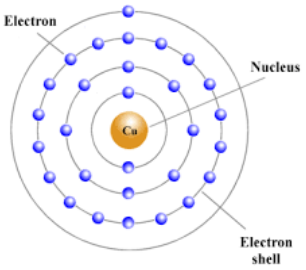
I2E101 Introduction to Electronics and Programming

Course Overview

- Chapter 1: Introduction to Electricity, Ohm's Law. I.V.R.
  - Project 1: Create an electronic circuit **Light with On/Off switch**
  - Project 2: Create a **Dimmable light** using potentiometer
- Chapter 2: Understanding Electromagnets. Multimeter
  - Project 3: Create an **Electromagnet**
  - Project 4: Create a **Serial Circuit** and a **Parallel Circuit**
- Chapter 3: Learn PNP and NPN Transistors
  - Project 5: Water level indicator (Using **PNP Transistor**)
  - Project 6: Water level indicator (Using **NPN Transistor**)
- Chapter 4: Introduction to Relays
  - Project 7: Light a bulb when water level = HIGH using **Relay**
- Chapter 5: Introduction to Arduino PLC
  - Project 8: First **Arduino Program**: Using Digital output
  - Project 9: Using **Potentiometer** (Analog Input)
- Chapter 6: Programming with Arduino
  - Project 10: Using **Photoresistor** (Analog Input)
  - Project 11: Create a **Dimmer** using Arduino PWM
- Chapter 7: Microphone / Sound Sensor, PIR Motion Sensor
  - Project 12: Create a **Clapper** using **Microphone** (Digital)
  - Project 13: Create a **Clapper** using **Microphone** (Analog)
  - Project 14: Create a Motion driven **Night light**
- Chapter 8: Burglar Alarm / Buzzer, LED Display
  - Project 15: Create a **Burglar Alarm**
  - Project 16: Draw Darth Vader on 8x8 **LED Display**

Electricity

Electricity is a **form of energy**. Electricity is the **flow of electrons**. All matter is made up of atoms, and an atom has a center, called a nucleus. The nucleus contains positively charged particles called protons and uncharged particles called neutrons. The nucleus of an atom is surrounded by negatively charged particles called electrons. The negative charge of an electron is equal to the positive charge of a proton, and the number of electrons in an atom is usually equal to the number of protons.



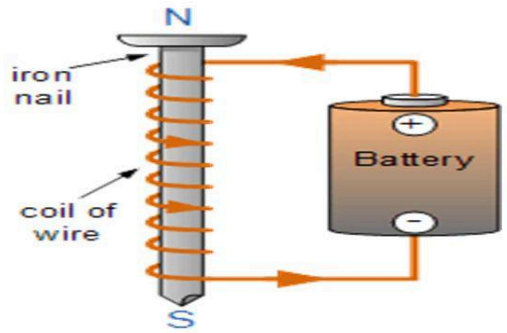
Atom

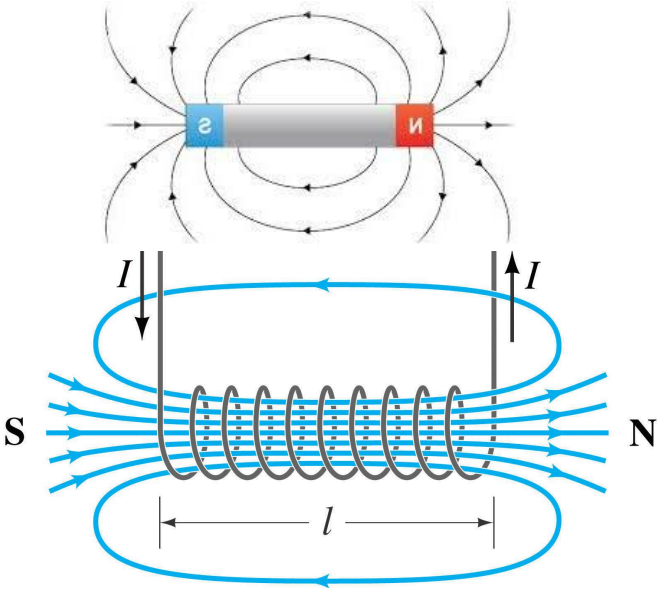
When the balancing force between protons and electrons is upset by an outside force, an atom may gain or lose an electron. When electrons are "lost" from an atom, the free movement of these electrons constitutes an electric current.

It is the set of physical phenomena associated with the presence and **motion of electric charge**. Although initially considered a phenomenon separate from **magnetism**, both are recognized as part of a single phenomenon: **electromagnetism**. Various common phenomena are related to electricity, including lightning, static electricity, electric heating, electric discharges and many others.

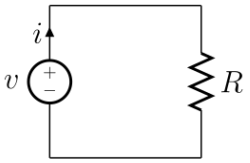
The presence of an electric charge, which can be either positive or negative, produces an **electric field**. The movement of electric charges is an electric current and produces a **magnetic field**.

Electromagnetism





Current



An electric **current** is a flow of electric charge. In electric circuits this charge is often carried by moving electrons in a wire. It can also be carried by ions in an electrolyte, or by both ions and electrons such as in an ionised gas (plasma).

The unit for measuring an electric current is the **ampere**, which is the flow of electric charge across a surface at the rate of one coulomb per second. Electric current is measured using a device called an ammeter. The coulomb (symbol: C) is unit of electric charge.

Voltage

Voltage, electric potential difference, electric pressure or electric tension (formally denoted ΔV or ΔU, but more often simply as V or U, for instance in the context of Ohm's laws) is the difference in electric potential between two points. The voltage between two points is equal to the work done per unit of charge against a static electric field to move a test charge between two points. This is measured in units of **volts**.

Electric potential differences between points can be caused by static electric fields, by electric current through a magnetic field, by time-varying magnetic fields, or some combination of these three. A voltmeter can be used to measure the voltage (or potential difference) between two points in a system; often a common reference potential such as the ground of the system is used as one of the points.

Ohm’s Law

Ohm's law states that the **current** through a conductor between two points is directly proportional to the **voltage** across the two points. Introducing the constant of proportionality, the **resistance**, one arrives at the usual mathematical equation that describes this relationship:

I = V / R

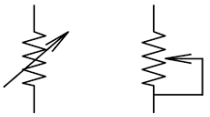
where I is the current through the conductor in units of **Amperes**, V is the voltage measured across the conductor in units of **Volts**, and R is the resistance of the conductor in units of **ohms**. More specifically, Ohm's law states that the R in this relation is constant, independent of the current.

Resistor

A **resistor** is a passive electrical component to create resistance in the flow of electric current. In almost all electrical networks and electronic circuits they can be found. The resistance is measured in **ohms** (Ω). An ohm is the resistance that occurs when a current of one ampere passes through a resistor with a one volt drop across its terminals. The current is proportional to the voltage across the terminal ends.

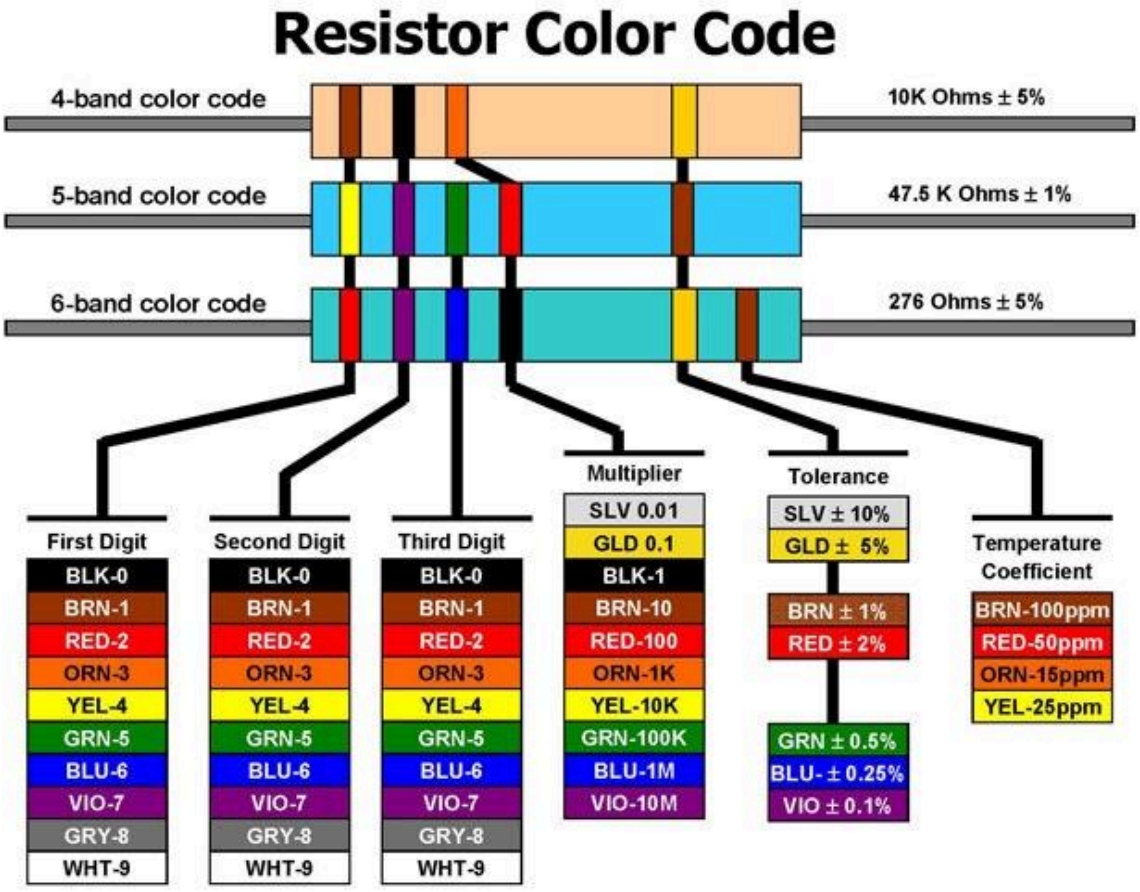
Symbols

resistor



variable resistor (potentiometer)

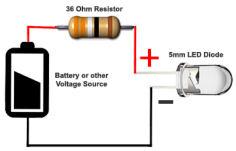
How to read a Resistor?

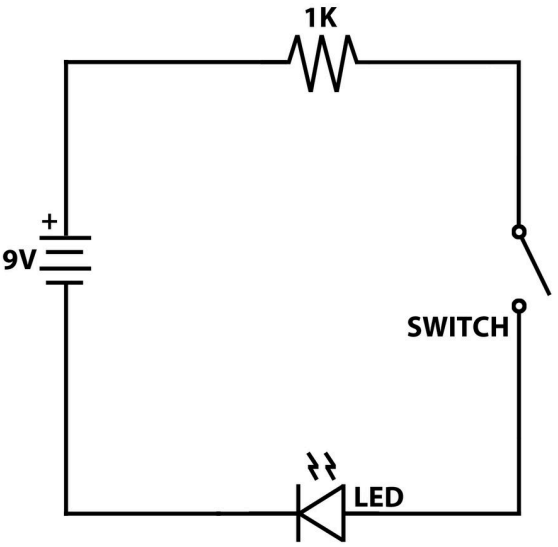


Project 1: Light with an On/Off switch

Hardware Required

- Breadboard
- 9V Battery
- 5V Power Supply
- Resistor
- LED
- Switch





Steps

1. Calculate the amount of current that a 9V battery when connected with 1K resistor will generate. (Hint: V = IR or I = V/R)

2. Before connecting the LED, measure the amount of current that a 9V battery with connected with 1K resistor will generate using **multimeter**.

3. Using multimeter, measure the resistance of the LED. Connect the LED. Pay attention to the direction. (Hint: smaller leg is negative)

4. Repeat the above by replacing 9V battery with a Power Supply board (set to 5V)

5. Measure the Voltage across the Power Supply and make sure it's 5V

6. Is the LED brighter or dimmer? Why?

7. Calculate the current draw again with 5V supply

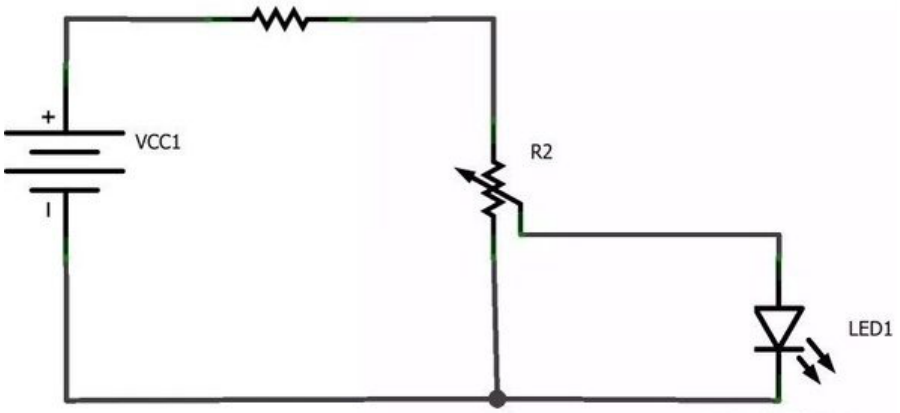
8. (Optional) Try changing power supply to 3.3V and try again

9. If LED is not bright, try lowering the resistor to ~ 300 Ω

Project 2: Dimmable light using potentiometer

Hardware Required

- Breadboard
- Battery
- 5V Power Supply
- Resistor (300Ω)
- Potentiometer
- LED



Steps

1. Connect multimeter to the outside and the center pin of the potentiometer

2. Measure the resistance. Rotate the potentiometer knob and see if the multimeter changes reading

3. Add the potentiometer to Project 1 as shown in the circuit above.

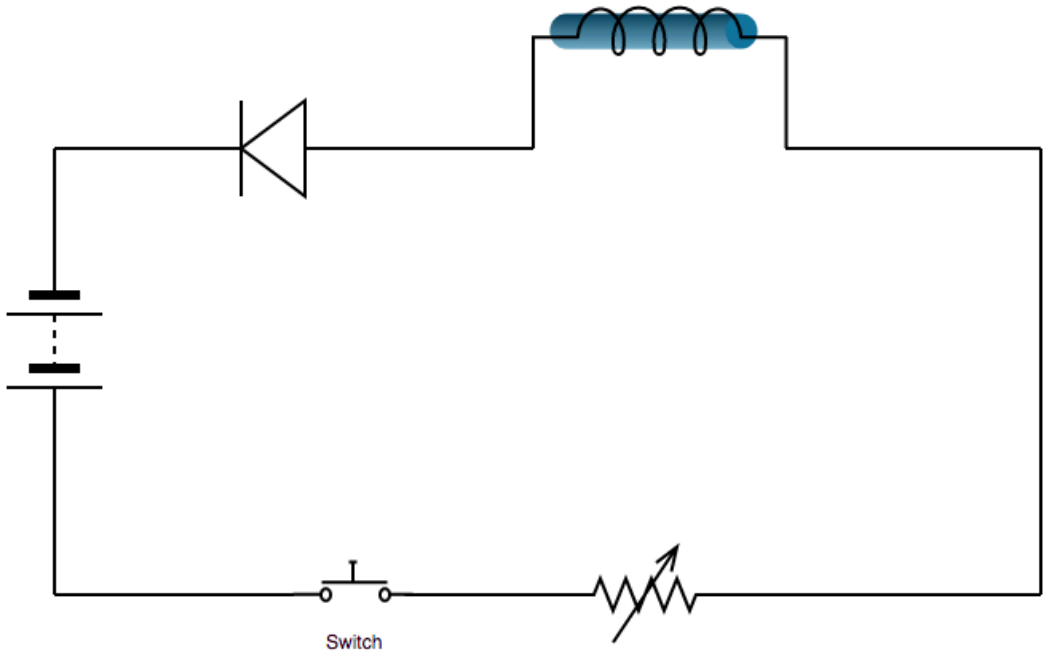
4. Rotate the potentiometer knob and check the LED brightness change as a result

5. Can you think of 5 applications of potentiometer?

6. Can you create a circuit where one LED goes brighter and another goes dimmer when you turn the potentiometer knob?

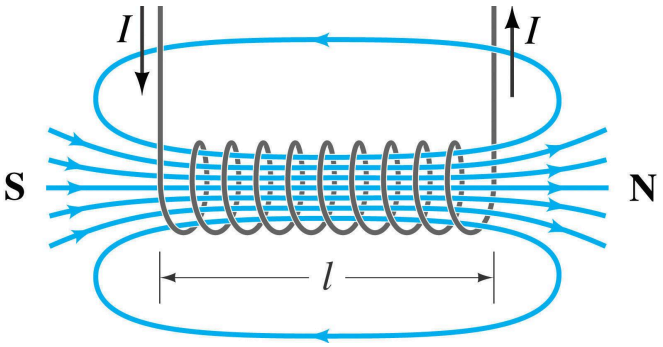
Project 3: Create an Electromagnet

An electromagnet is a type of magnet in which the **magnetic field** is produced by an **electric current**. The magnetic field disappears when the current is turned off. A current through the wire creates a magnetic field which is concentrated in the hole in the center of the coil. The wire turns are often wound around a magnetic core made from a ferromagnetic or ferrimagnetic material such as iron; the magnetic core concentrates the magnetic flux and makes a more powerful magnet.



Hardware Required

- Breadboard
- Soft Iron Rod, James Clip
- Electric Wire (>24 gauge)
- LED, Switch and 400 Ohms resistor
- 9V or higher battery



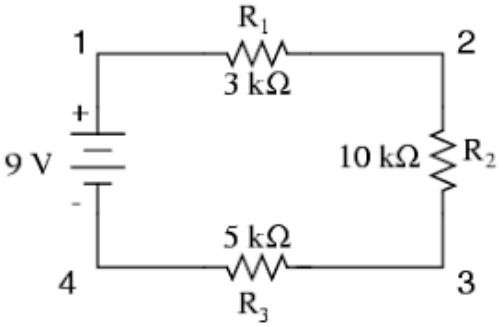
Steps

1. Wrap the wire around the Iron Rod. The soft iron inside the coil makes the magnetic field stronger because it becomes a magnet itself when the current is flowing. Soft iron is used because it loses its magnetism as soon as the current stops flowing.
2. Connect the circuit as shown.
3. Bring the James Clip closer to the iron rod and you will experience no magnetic field. Try again after turning the switch ON.

Project 4: Serial vs Parallel Circuits

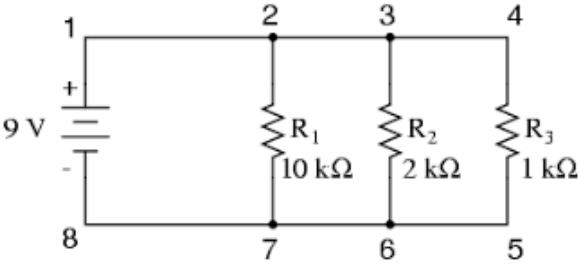
Connecting in Serial

- Same current flows through each resistor
- The total voltage of the circuit is the sum of the voltages across each resistor
- If one resistor fails, the entire circuit won't work
- Series circuits are easier to wire and troubleshoot
- Varying voltages across each resistor is okay



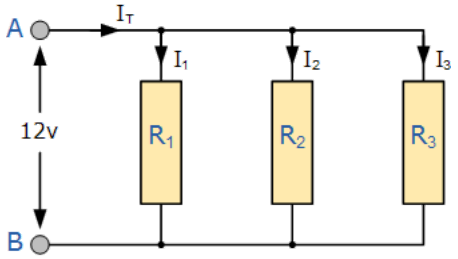
Connecting in Parallel

- The voltage across each resistor is same
- The total current is the sum of currents through each resistor
- The total output current is shared through each parallel string
- Exact voltages are required in each parallel string to help avoid current hogging



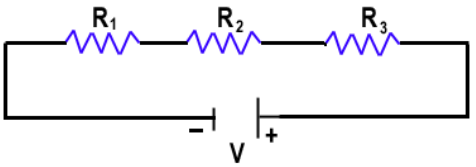
Q1. In the circuit showm, if R1=R2=R3 = 300Ω, what is the value of I1, I2, I3 and I

V = I * R
⇒ 12 = I_1 * R_1 = I_2 * R_2 = I_3 * R_3
⇒ 12 = I_1 * 300
⇒ I_1 = 12 / 300 = 0.040 A = 40 mA
⇒ I_1 = I_2 = I_3 = **40 mA**
I = I_1 + I_2 + I_3 = 40 + 40 + 40 = **120 mA**



Q2. If R1 = R2 = R3 = 300Ω, V = 12V, what is the value of total current through the circuit?

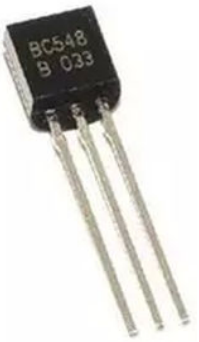
V = I * R
⇒ 12 = I * (R_1 + * R_2 + * R_3)
⇒ 12 = I * 900
⇒ I = 12 / 900 = .01333 A = **13.33 mA**



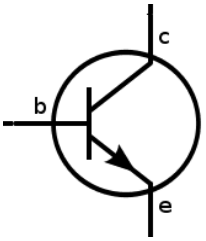
3. Revisit Project 2 to analyse how potentiometer is splitting the circuit into a parallel circuit

Transistors

The transistor is like an electronic switch (and can also act as an amplifier). It can turn a current on and off. A very common one is the "bipolar junction transistor" or "BJT". And it usually looks like this →

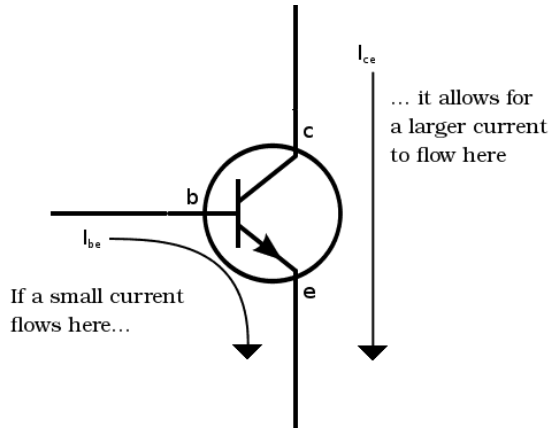


It has three pins: Base (b), collector (c) and emitter (e). And it comes in two versions: NPN and PNP. The schematic symbol for the NPN looks like this:



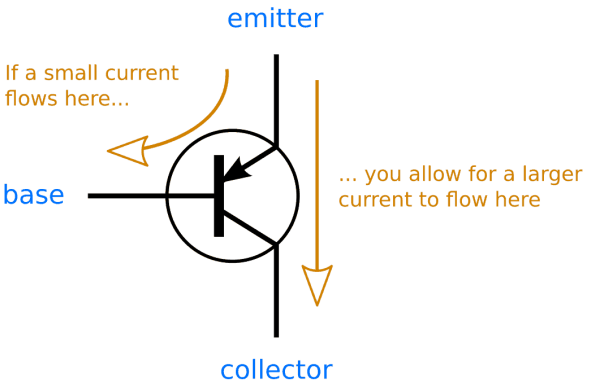
How does Transistors work?

The transistor works because of something called a semiconducting material. A current flowing from the base to the emitter "opens" the flow of current from the collector to the emitter



In a standard **NPN** transistor, you need to apply a very small voltage (of about 0.7V) between the base and the emitter to get the current flowing from base to emitter. When you apply approximately 0.7V from base to emitter you will turn the transistor ON and allow a current to flow from collector to emitter

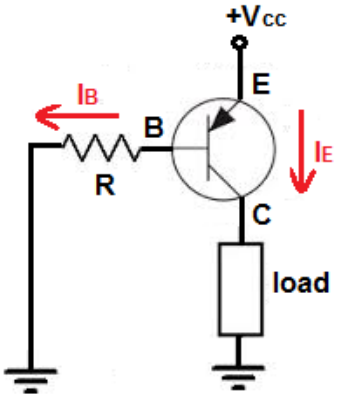
A **PNP** transistor will "turn on" when you have a small current running from emitter to base of the transistor. When I say "turn on", I mean that the transistor will open up a channel between emitter and collector. And this channel can carry a much larger current.



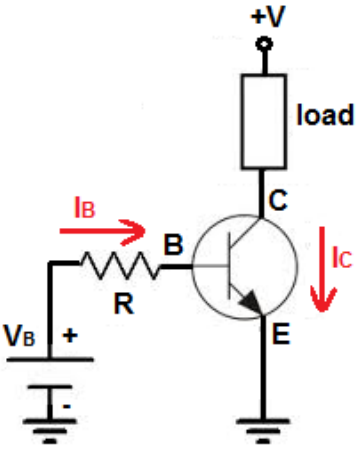
To get current running from emitter to base, you need a voltage difference of about 0.7V. Since the current goes from emitter to base, the base needs to be 0.7V lower than the emitter.

By setting the base voltage of a PNP transistor to 0.7V lower than the emitter, you "turn the transistor on" and allow for current to flow from emitter to collector

PNP vs NPN transistor



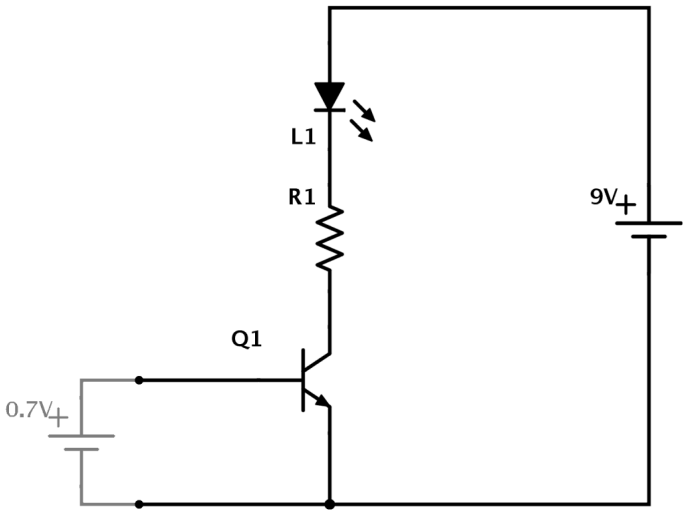
In a PNP transistor, positive voltage is given to the emitter terminal and current flows from the emitter to the collector, given there is sufficient negative current flow from the base



In an NPN transistor, positive voltage is given to the collector terminal and current flows from the collector to the emitter, given there is sufficient base current

Transistor Example

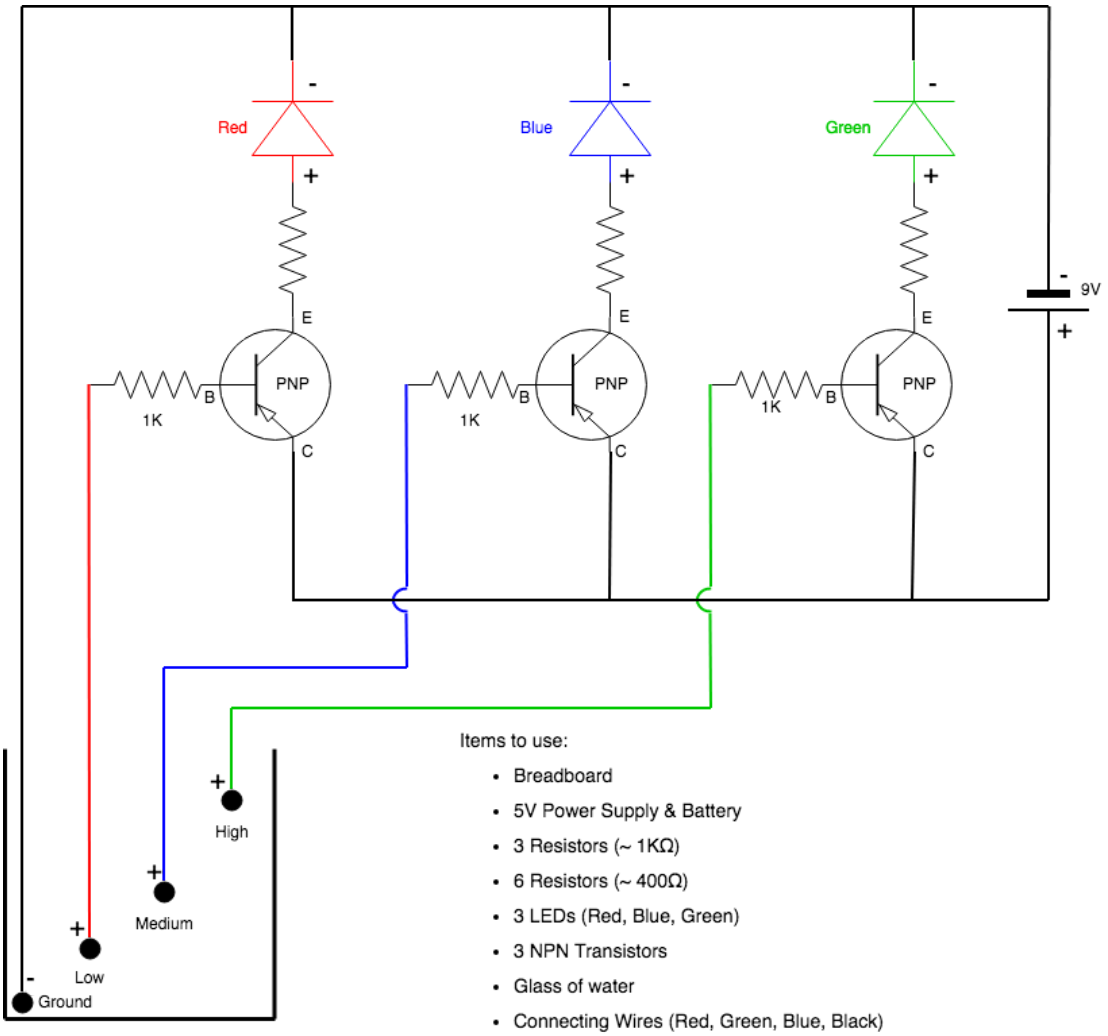
In this example you can see how transistors work. A 9V battery connects to an LED and a resistor. But it connects through the transistor. This means that no current will flow in that part of the circuit until the transistor turns ON.



To turn the transistor ON you need to apply 0.7V from base to emitter of the transistor. Imagine you have a small 0.7V battery. (In a practical circuit you would use resistors to get the correct voltage from whatever voltage source you have)

When you apply a very small current (0.7V battery) from base to emitter, the transistor turns ON. This allows current to flow from the collector to the emitter. And thereby turning the LED ON!

Project 5: Water level indicator (Using PNP Transistor)



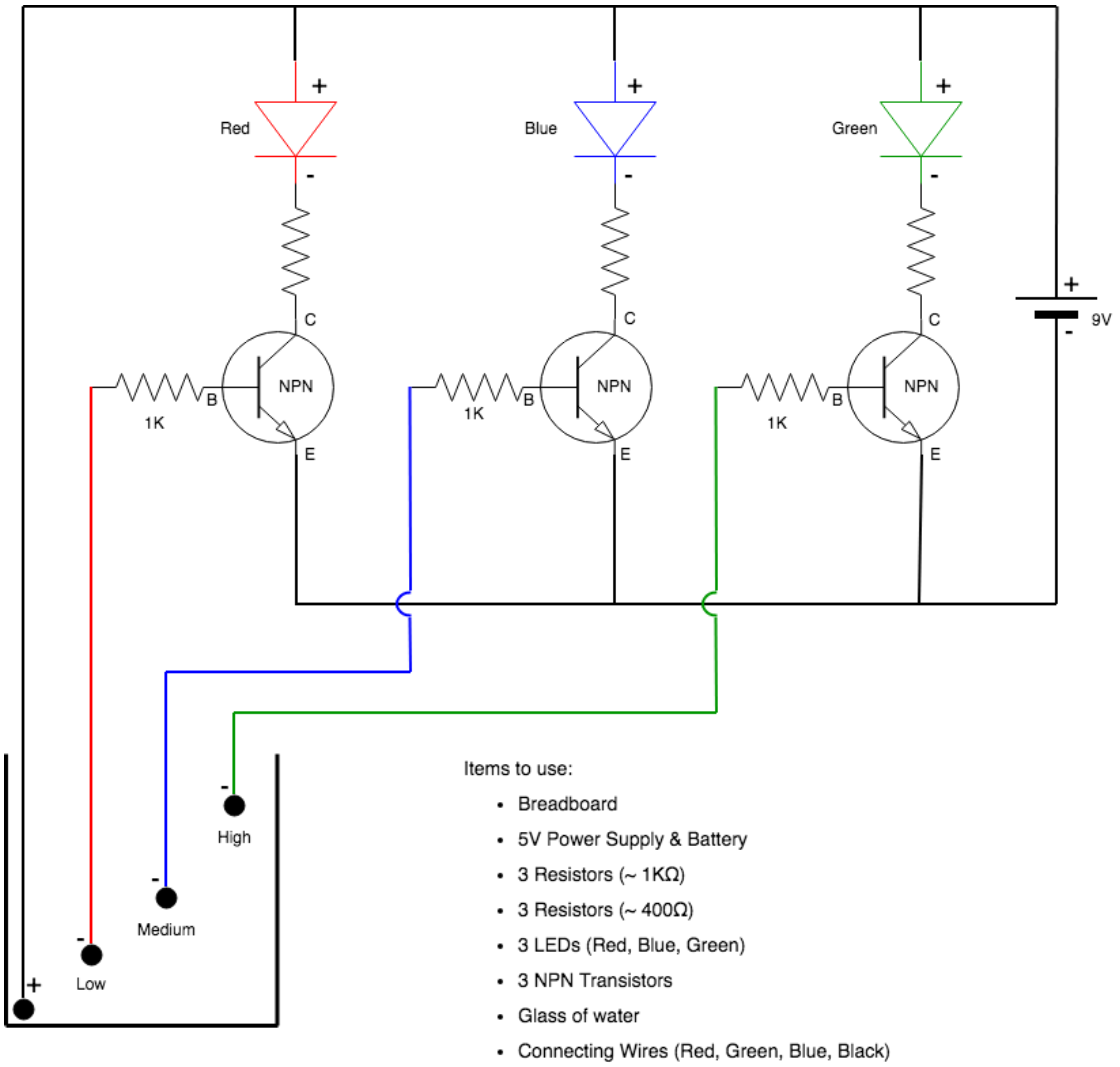
Steps

1. Using multimeter, select 3 resistors of ~400Ω and 3 of 1K
2. Connect PNP Transistor in series with LED Diode and Resistor. Pay close attention to LEDs positive and negative side (Hint: small leg is negative)
3. Use color coding to make it easy to understand the circuit. For example: Use Red LED with Red wire for Low indicator.
4. Pour water very slowly and see different LEDs light up as water level increases

Observation

Transistor is acting as a switch in this experiment, which is controlled by the water level. As the water level rises, the circuit is completed and a small amount of current flows to the base of the transistor. By setting the base voltage of a PNP transistor to lower than the emitter, we turn the transistor ON and allow for current to flow from emitter to collector and thereby turning the LED ON!

Project 6: Water level indicator (Using NPN Transistor)



Steps

1. Note the difference when using NPN transistors vs PNP transistors
2. Base is connected to Positive (+) in NPN as opposed to Negative (-) in PNP
3. Note the direction of the LEDs and Battery

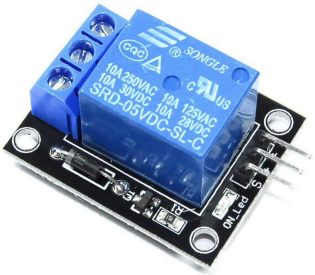
Observation

Transistor is acting as a switch in this experiment, which is controlled by the water level. As the water level rises, the circuit is completed and a small amount of current flows to the base of the transistor. When we apply a small current from base to emitter, the transistor turns ON. This allows current to flow from the collector to the emitter and thereby turning the LED ON!

Challenge Exercise (Self)

- *Measure the resistance of water using multimeter.*
- *Try replacing 3 LEDs with a single RGB LED in the circuit*

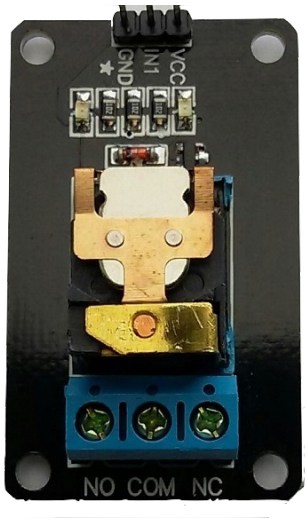
Relays

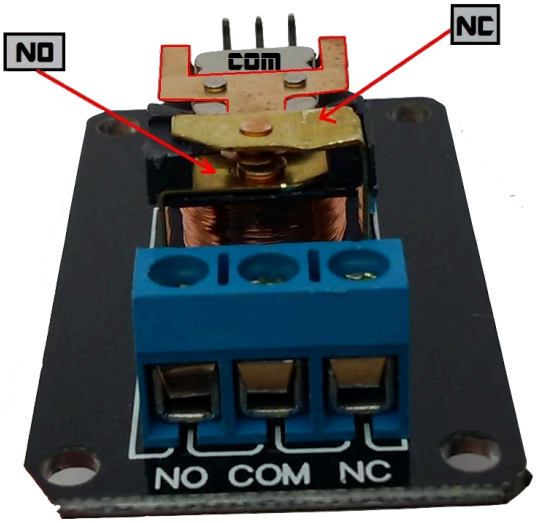


A relay is an **electrically operated switch**. Many relays use an **electromagnet** to mechanically operate a switch.

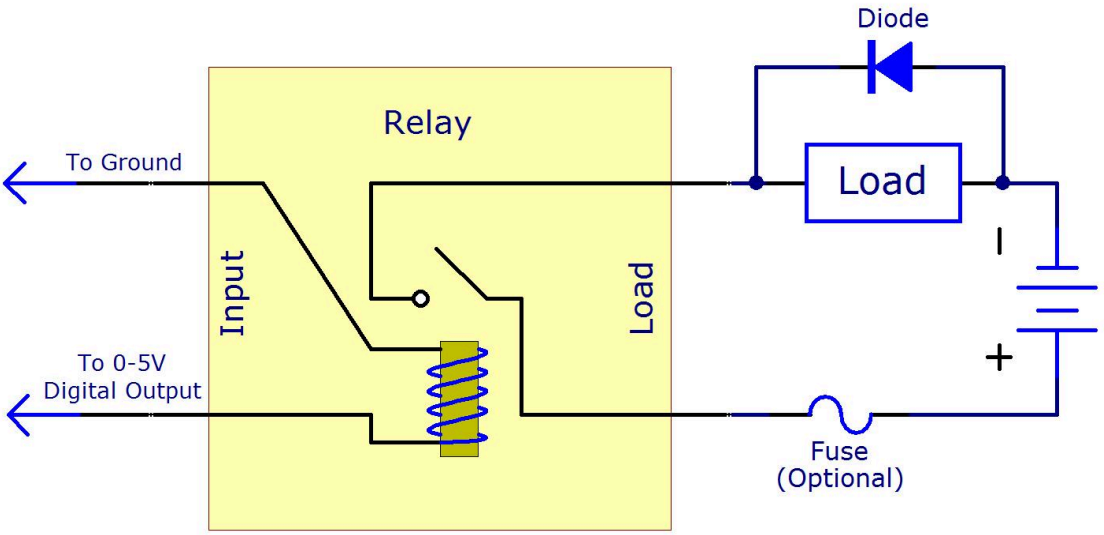
Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal.

Internals of a Relay



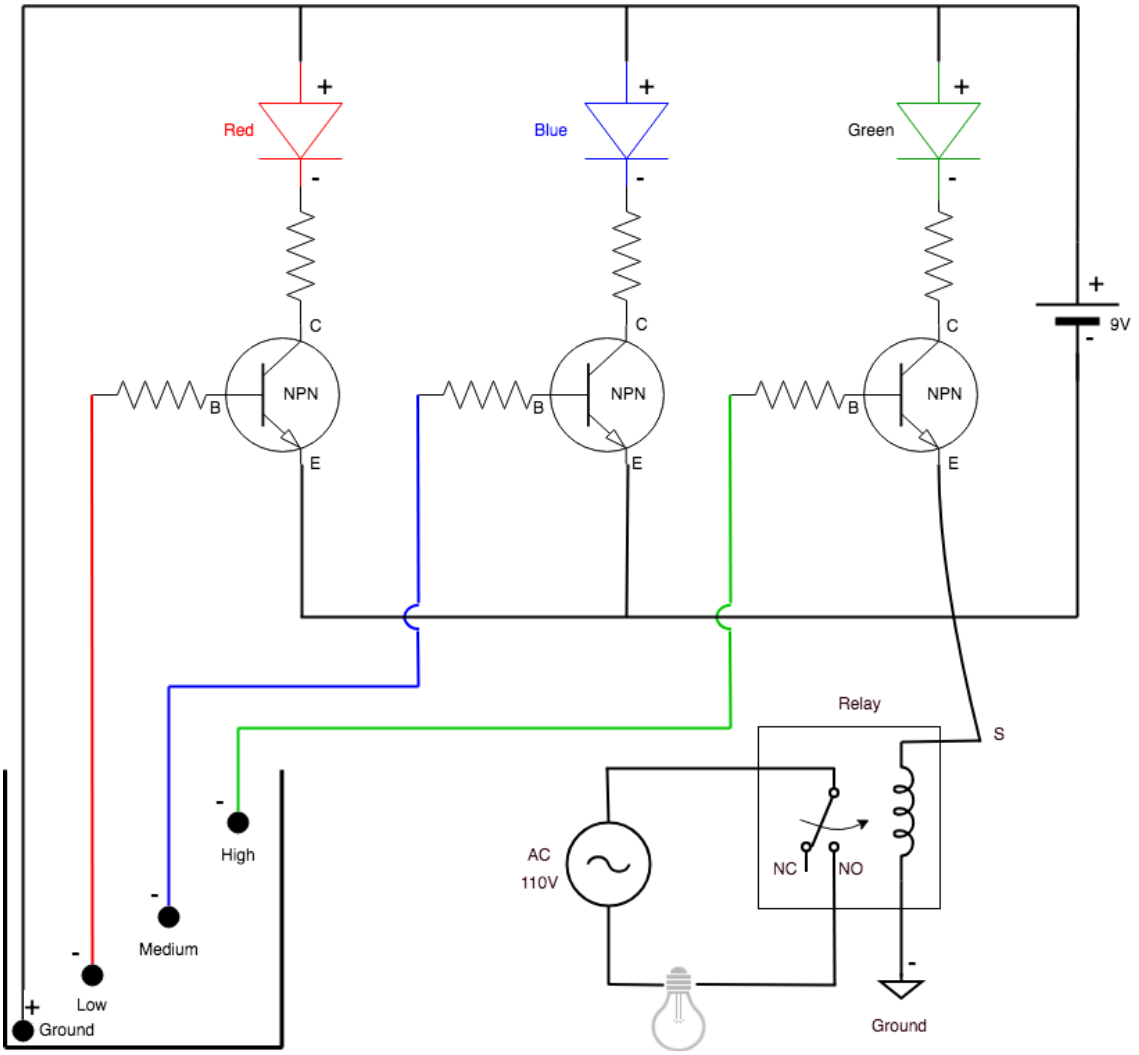


Using Relay in a circuit



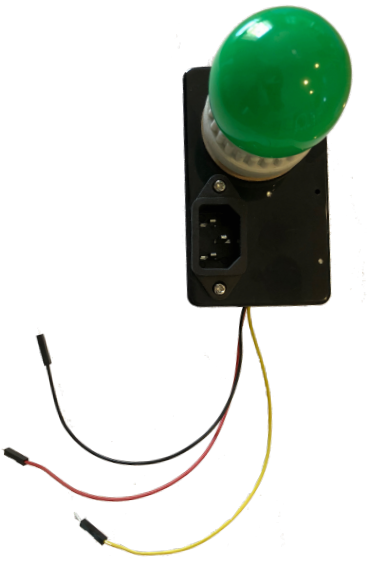
Project 7: Light a bulb when water level = HIGH

Note: Since we are dealing with AC, do this project in Adult supervision



Additional Hardware to use

- Everything from Project #4
- Relay, Bulb with socket, Power Cable, Wires or Use
- Since we are dealing with High Voltage AC, use the Relay Box shown in the picture to the right.



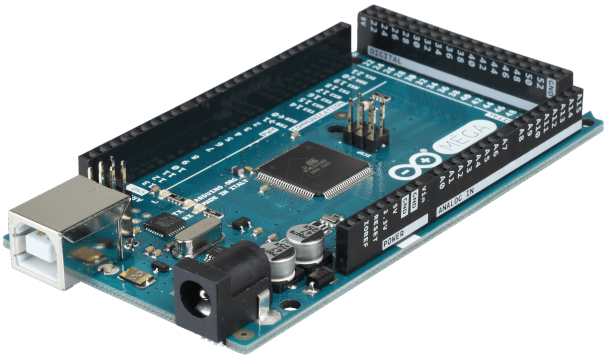
Steps

1. Modify Project 4 and replace Green LED with the relay box
2. Relay box wires: Black = -ve, Red = +ve, Yellow = Signal
3. Fill the bucket with water slowly and see the Green Bulb turn on when the Water level reaches HIGH

Arduino Microcontroller

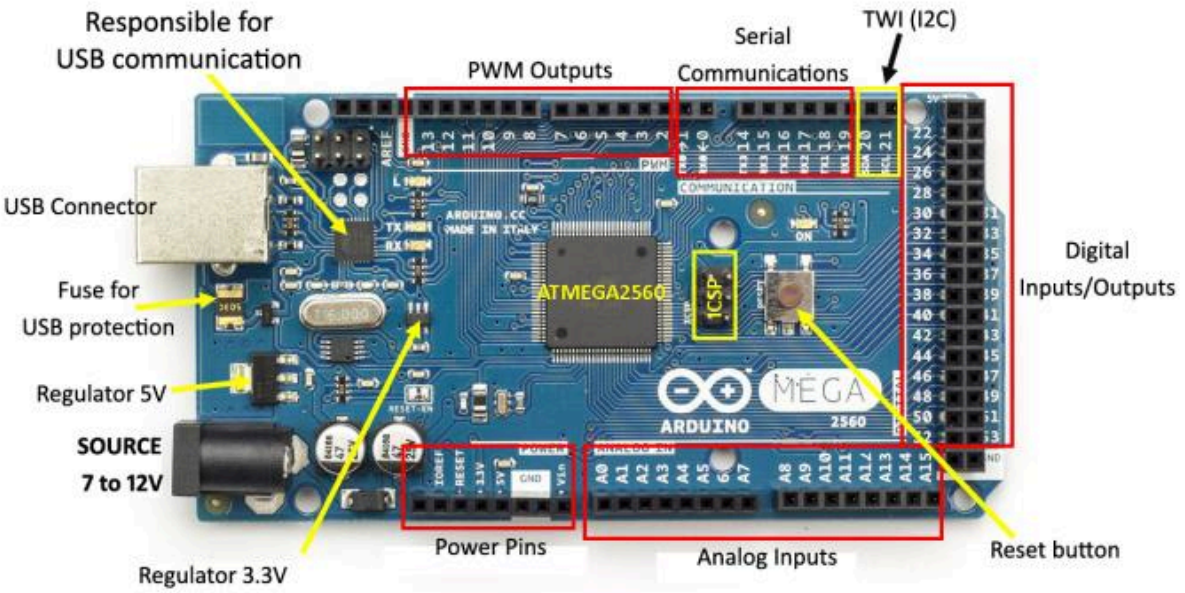
Arduino is an open-source electronics platform based on easy-to-use **hardware** and **software**. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online.

Technically, the Arduino is a **programmable logic controller (PLC).** PLC is nothing more than a computer with a processor, except that the architecture is created in a way that is focused on interacting with the outside world. It gets information from the outside world through **inputs** – digital and analog sensors, relays and other assorted gadgets. It interacts with the real world through **outputs** – motors, valves, conveyor belts, actuators and much more.



In between all of the inputs and outputs is the PLC – the heart of the beast and the brains behind the entire operation. PLC programming makes the decisions based on **input from the real world**, and then immediately interacts with the real world through the outputs – all in fractions of a second. These are essentially robots. There are several types of Arduino boards. We will be using **Arduino MEGA 2560 R3** for all our projects.

Know your Arduino



Getting Started

Read an introduction on what is Arduino and why you'd want to use it.
https://www.arduino.cc/en/Guide/Introduction

Arduino Software IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus.

Download from: https://www.arduino.cc/en/Main/Software

It connects to the Arduino hardware to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called **sketches**

Libraries

Extend the ability of your Arduino with additional libraries
https://www.arduino.cc/en/Guide/Libraries

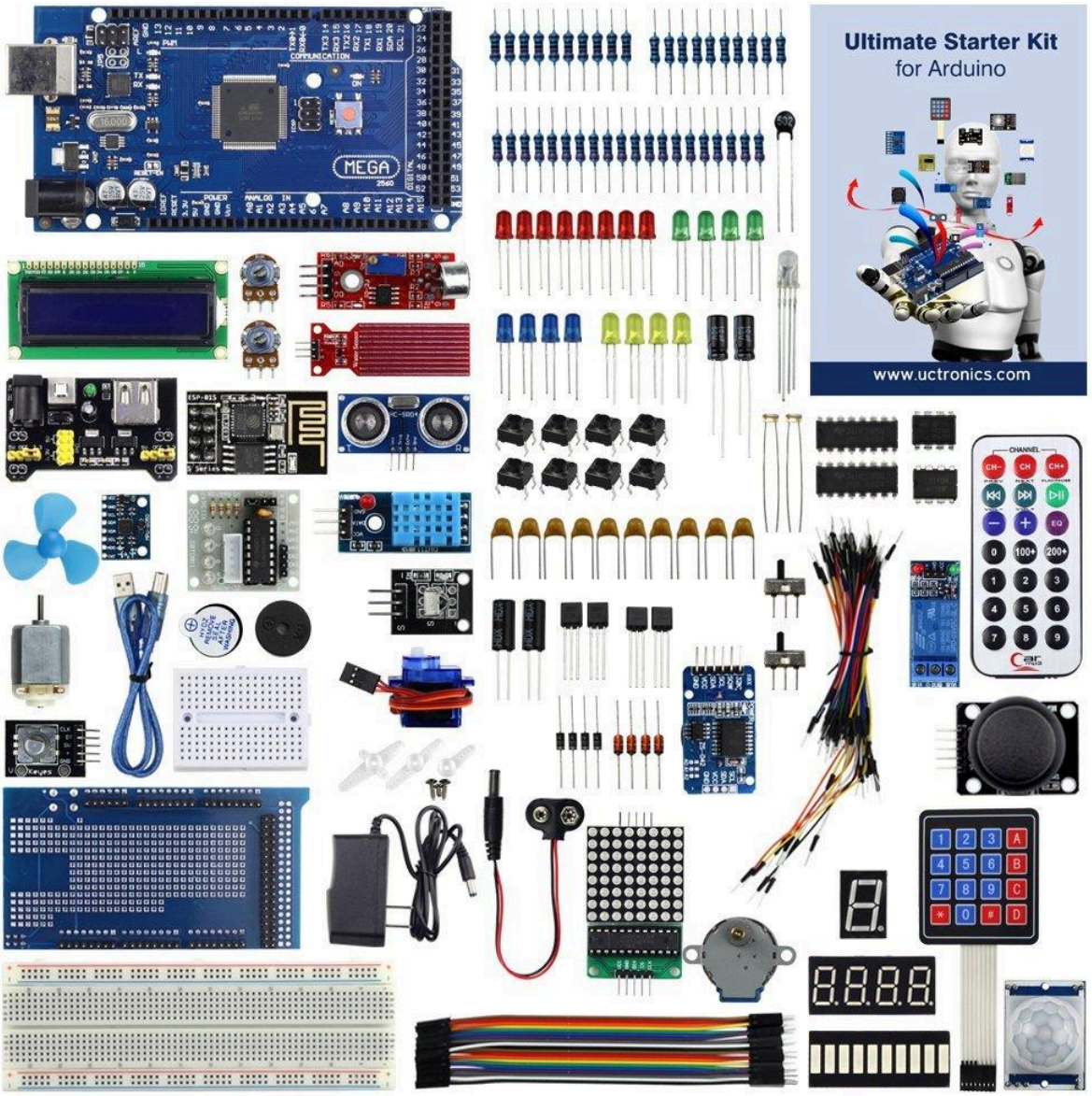
Built-in Examples

https://www.arduino.cc/en/Tutorial/BuiltInExamples

Programming reference

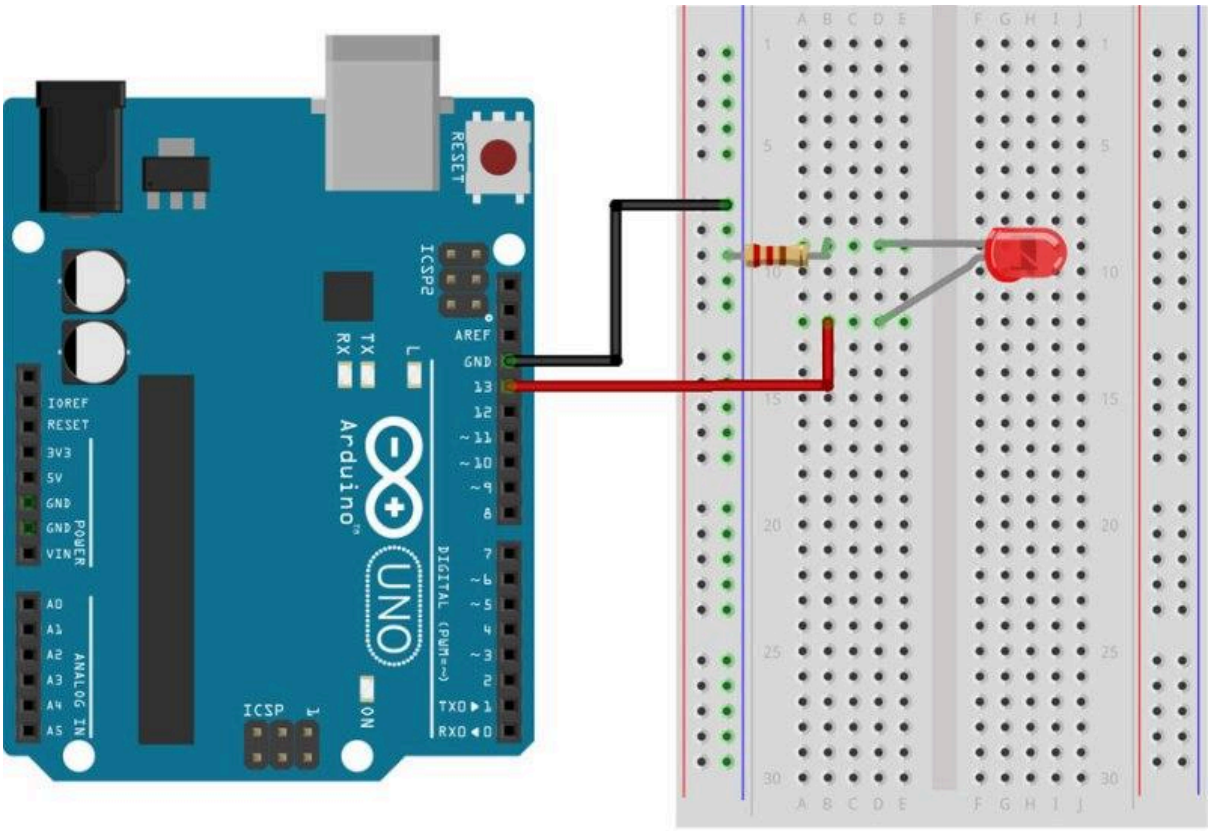
https://www.arduino.cc/reference/en/

Arduino MEGA 2560 R3, Servo Motor, Sensors

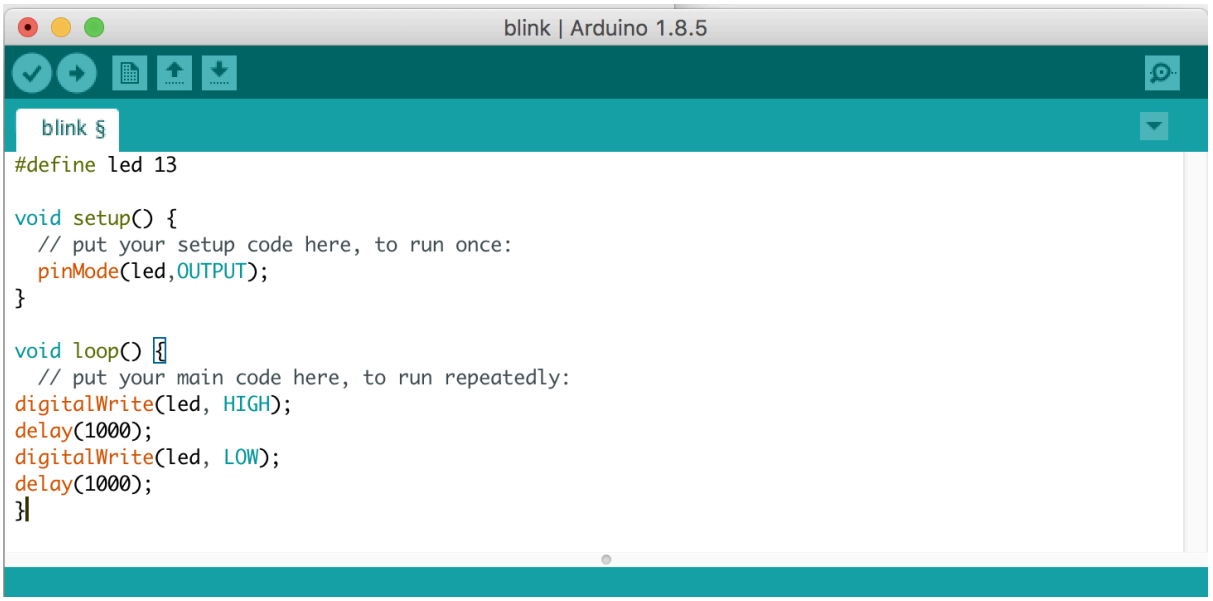


Project 8: First Arduino Program: Using digital output

Circuit



Sketch (File ==> Examples ==> Basic -> Blink)



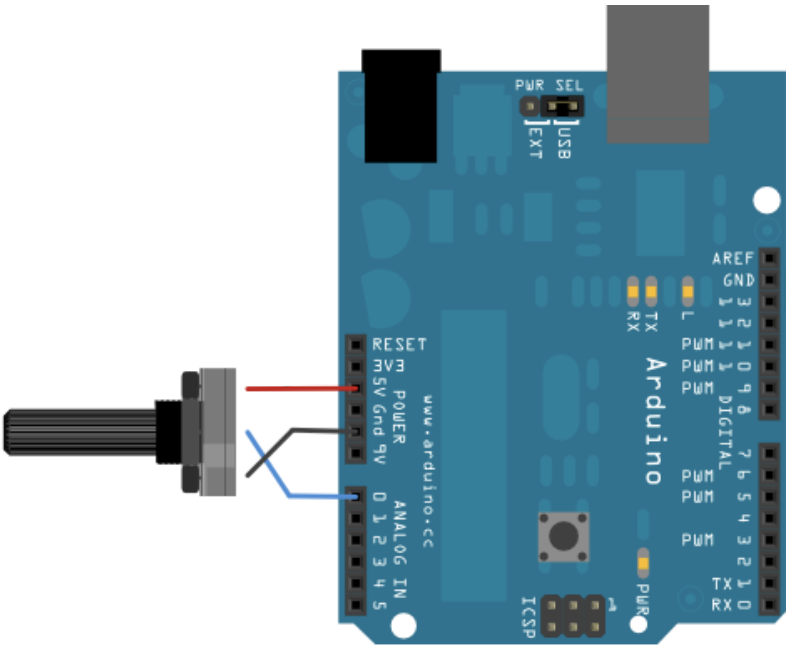
1. Make sure Arduino MEGA is selected in the boards and Port is also selected.
2. Upload the sketch to Ardunio

3. LED will light up with led pin 13 go HIGH and will turn off when it goes LOW
4. **digitalWrite()** sends 0V (LOW) or 5V (HIGH) to the specified PIN

Project 9: Using potentiometer (Analog Input)

Hardware Required

- Arduino Mega Board
- 10K Potentiometer
- Built-in LED on pin 13



Sketch

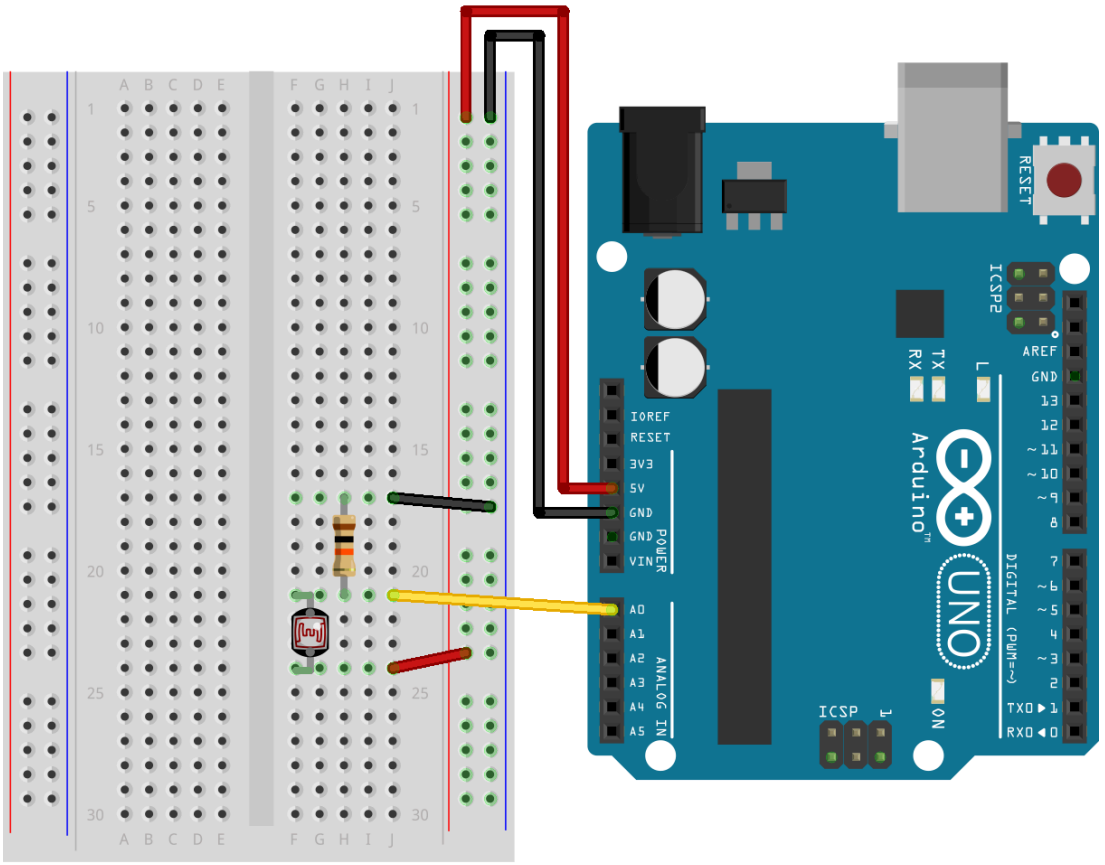
AnalogInput §

```
/*
  Demonstrates analog input by reading an analog sensor on analog pin 0 and
  turning on and off a light emitting diode(LED) connected to digital pin 13.
  The amount of time the LED will be on and off depends on the value obtained
  by analogRead().

  The circuit:
  - potentiometer: center pin of the potentiometer to the analog input 0
    one side pin (either one) to ground, the other side pin to +5V
  - LED: anode (long leg) attached to digital output 13
    cathode (short leg) attached to ground
*/
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;      // select the pin for the LED
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

Photoresistor

A **photoresistor** (or light-dependent resistor, **LDR**, or **photo-conductive cell**) is a light-controlled **variable resistor**. The resistance of a photoresistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity. A photoresistor can be applied in light-sensitive detector circuits, and light-activated and dark-activated switching circuits

Project 10: Using Photoresistor (Analog Input)



You can replace the potentiometer in previous project with a photoresistor and a resistor (as shown in the circuit above).

Vout (on pin A0) **= Vin*(R2/(R1+R2))**

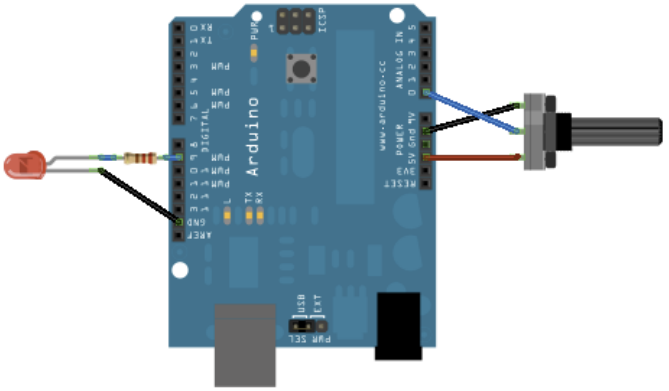
where Vin is 5V, R2 is 10k ohm and R1 is the photoresistor value that ranges from 1M ohm in darkness to 10k ohm in daylight (10 lumen) and less than 1k ohm in bright light or sunlight (>100 lumen)

Project 11: Create a Dimmer using Arduino PWM

Read an analog input pin, map the result to a range from 0 to 255, use that result to set the **pulse width modulation** (PWM) of an output pin to dim or brighten an LED

Hardware Required

- Arduino Mega Board
- 10K Potentiometer
- Red LED
- 220Ω Resistor



Sketch

AnalogInOutSerial §

```
// These constants won't change. They're used to give names to the pins used:
const int analogInPin = A0;  // Analog input pin that the potentiometer is attached to
const int analogOutPin = 9; // Analog output pin that the LED is attached to

int sensorValue = 0;        // value read from the pot
int outputValue = 0;        // value output to the PWM (analog out)

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
}

void loop() {
  // read the analog in value:
  sensorValue = analogRead(analogInPin);
  // map it to the range of the analog out:
  outputValue = map(sensorValue, 0, 1023, 0, 255);
  // change the analog out value:
  analogWrite(analogOutPin, outputValue);

  // print the results to the Serial Monitor:
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);

  // wait 2 milliseconds before the next loop for the analog-to-digital
  // converter to settle after the last reading:
  delay(2);
}
```

In the loop() function, sensorValue stores the raw analog value from the potentiometer. Arduino has an **analogRead** range from 0 to 1023, and an **analogWrite** range only from 0 to 255, therefore the data from the potentiometer needs to be converted to fit into the smaller range before using it to dim the LED. That's what map() function is doing.

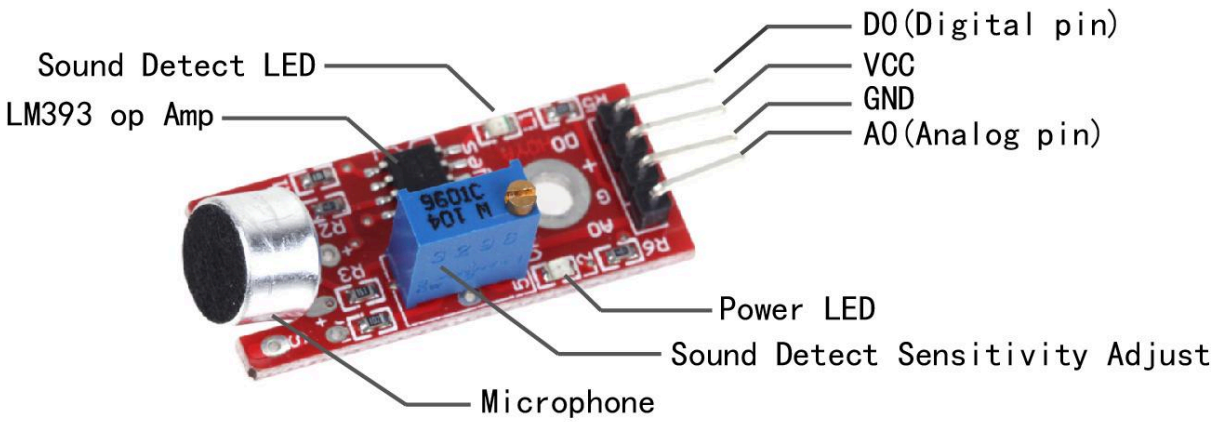
Serial.begin() function is used to pass data serially at the specified bit rate.

Microphone / Sound Sensor

Microphone is a **transducer**, which can sense sound and convert sound energy to electrical pulse. So microphone is a sensor too. **We can use sound to change voltage**. Microphones used in electronics works has two terminals – positive and negative. Polarity can be detected using a multimeter like diode testing mode.

Arduino Sound Detection Sensor

Sound is detected via a microphone and fed into an LM393 op amp. The sound level set point is adjusted via an on board potentiometer. When the sound level exceeds the set point, an LED on the module is illuminated and the output is sent low.



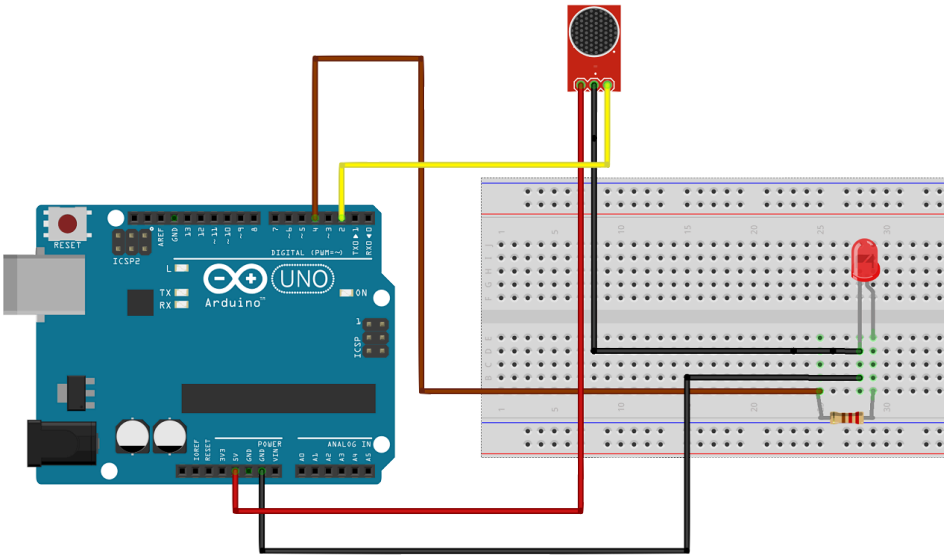
Project 12: Create a Clapper w/ Microphone (digital)

The clap detector listens for a clap (and lights up a LED). On a second clap, it turns off the LED. The program outputs all data to the serial port and adjusts input levels by measuring the ambient noise levels first.

Hardware Required

- Arduino Mega or UNO
- A Sound Sensor
- LED, 220 ohm Resistors

Circuit



Sketch

```
int soundSensor=2;
int LED=4;

void setup() {
 pinMode(soundSensor,INPUT);
 pinMode(LED,OUTPUT);
}

void loop() {
  int SensorData=digitalRead(soundSensor);
  if(SensorData==1){
        digitalWrite(LED,HIGH);
  } else {
        LEDStatus=false;
        digitalWrite(LED,LOW);
  }
}
```

(Optional) Advanced Clapper: https://coeleveld.com/arduino-microphone/

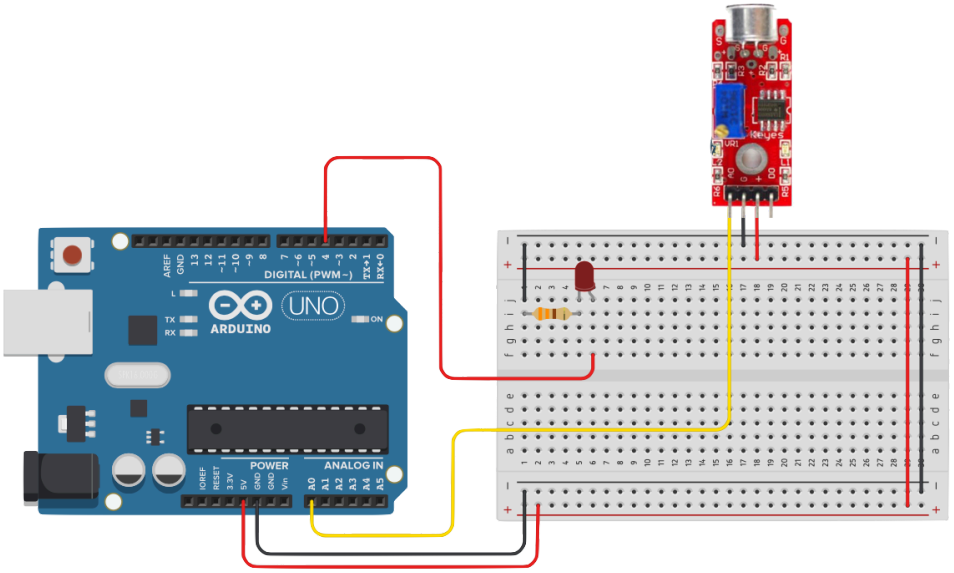
Project 13: Create a Clapper w/ Microphone (Analog)

In the previous example, you probably noticed that digital out (DO) always outputs 1 (HIGH) because of ambient noise. In this project, we will change it to use Analog output (AO) so we can account for ambient noise.

Hardware Required

- Arduino Mega or UNO
- A Sound Sensor
- LED, 220 ohm Resistors

Circuit

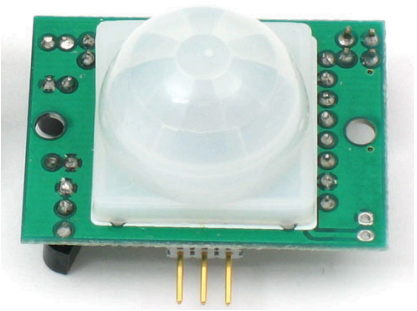


Sketch

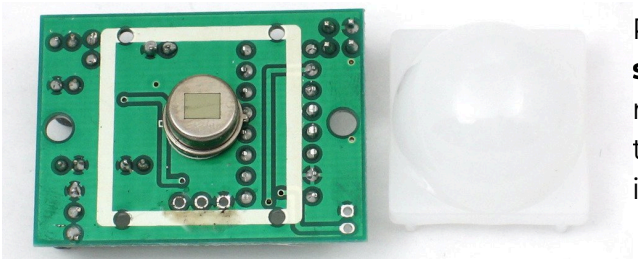
```
int soundSensor=A0;
int LED=4;

void setup() {
 pinMode(soundSensor,INPUT);
 pinMode(LED,OUTPUT);
 Serial.begin(9600);
}

void loop() {
  int SensorData=analogRead(soundSensor);
  Serial.println(SensorData);
  if(SensorData > 300){ //Change the value to account for ambient noise
      digitalWrite(LED,HIGH);
      delay(5000);
  } else{
      digitalWrite(LED,LOW);
  }
 }
```

(Optional) Advanced Clapper: https://coeleveld.com/arduino-microphone/

PIR Motion Sensor



PIR sensors allow you to **sense motion**, almost always used to detect whether a human has moved in or out of the sensors range. They are often referred to as PIR, "**Passive Infrared**", "Pyroelectric", or "IR motion" sensors.



PIRs are basically made of a **pyroelectric sensor** (which you can see as the round metal can with a rectangular crystal in the center), which can detect levels of infrared radiation.

NOTE: Everything emits some low level radiation, and the hotter something is, the more radiation is emitted.

PIR Lens

Most of the real magic happens with the **optics**. To increase the detection area, simple lens (as in a Camera) is used. They condenses a large area (such as a landscape) into a small one (on film or a CCD sensor).

Specs

Output: Digital pulse high (3V) when triggered (motion detected) digital low when idle (no motion detected). Pulse lengths are determined by resistors and capacitors on the PCB and differ from sensor to sensor.

Sensitivity range: up to 20 feet (6 meters) 110° x 70° detection range

Power supply: 5V-12V input voltage for most modules (they have a 3.3V regulator), but 5V is ideal in case the regulator has different specs

Learn More

http://www.glolab.com/pirparts/infrared.html

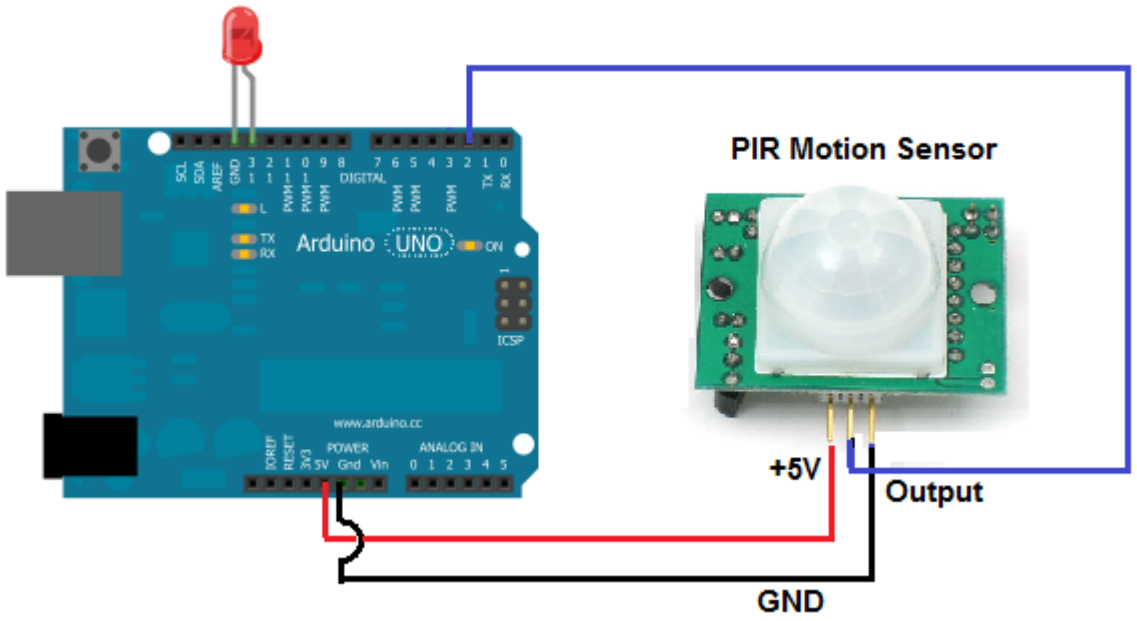
Project 14: Create a motion driven night light

With a PIR motion sensor integrated with an arduino, we can detect movement and program the arduino to turn a light on (or off) once this motion is detected.

Hardware Required

- Arduino Board
- PIR motion sensor
- LED

Circuit



Sketch

```
const int ledPin= 13;
const int inputPin= 2;

void setup(){
  pinMode(ledPin, OUTPUT);
  pinMode(inputPin, INPUT);
}

void loop(){
  int value= digitalRead(inputPin);
  if (value == HIGH) {
    digitalWrite(ledPin, HIGH);
    delay(60000);
    digitalWrite(ledPin, LOW);
  }
  else {
    digitalWrite(ledPin, LOW);
```

MotionSensor

```
const int ledPin= 13;
const int inputPin= 2;

void setup(){
  pinMode(ledPin, OUTPUT);
  pinMode(inputPin, INPUT);
}

void loop(){
  int value= digitalRead(inputPin);

  if (value == HIGH) {
    digitalWrite(ledPin, HIGH);
    delay(60000);
    digitalWrite(ledPin, LOW);
  }
  else {
    digitalWrite(ledPin, LOW);
  }
}
```

```
  }
}
```

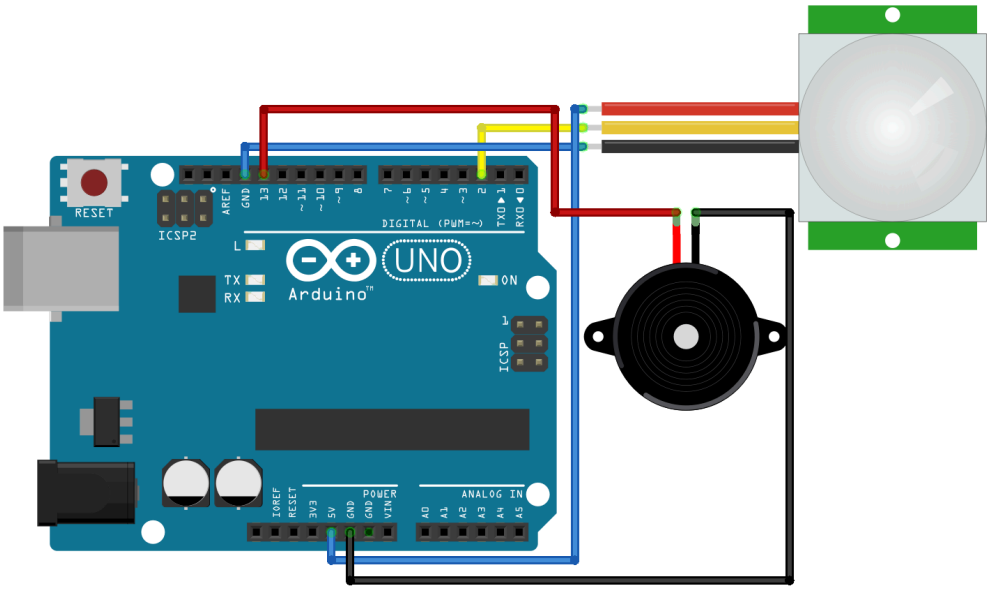
Project 15: Create a Burglar Alarm using Interrupt

With a PIR motion sensor integrated with an arduino, we can detect movement and program the arduino to trigger an alarm / buzzer when a motion is detected.

Hardware Required

- Arduino Board
- PIR motion sensor
- Buzzer

Circuit



Sketch

MotionBuzzer §

```
/*
Buzzer          - Digital Pin 13
Motion Sensor   - Digital Pin 02
*/
int motionPin = 2;
int buzzer = 13;
void setup() {
  pinMode(buzzer, OUTPUT);     // Buzzer as output
  pinMode(motionPin, INPUT_PULLUP);
  digitalWrite(buzzer, LOW);  //make Buzzer OFF
  //PIR Motion Sensor connected at digital pin 2 i.e. interrupt 0
  //attachInterrupt(0, alarmON, RISING); //Or call below recommended syntax
  attachInterrupt(digitalPinToInterrupt(motionPin), alarmON, RISING);
}
void loop() {
}
// Alarm ON
void alarmON() {
  digitalWrite(buzzer, HIGH);
}
```

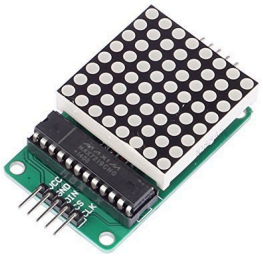
Note the use of **attachInterrupt()** instead of **loop()** function. We implemented the interrupt at rising edge of motion sensor input. Interrupt frees up microcontroller to get some other work done while not missing the input.

Learn more:
https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/

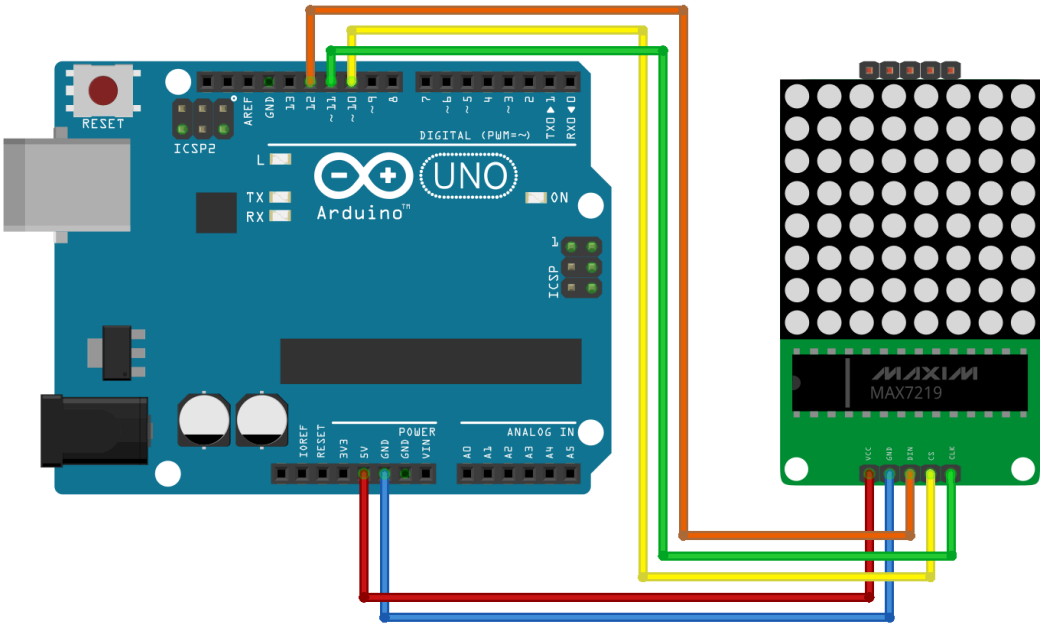
Project 16: Draw Darth Vader on 8x8 LED Display (MAX7219)

Hardware Required

- Arduino MEGA
- Breadboard
- MAX7219: 8x8 Matrix LED
- Wires

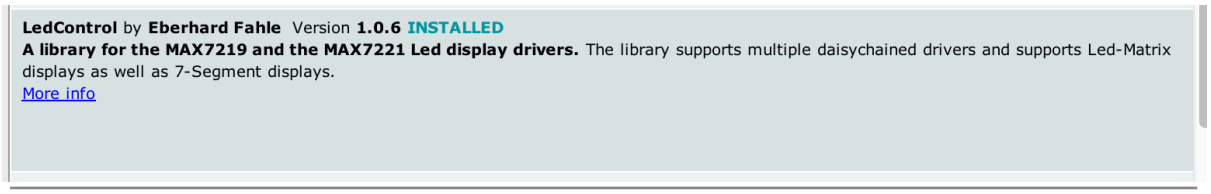


Circuit



Install Library for MAX7219

Go to Arduino => Sketch => Include Library => Manage Libraries and search for MAX7219. You will find the following library from Eberhard Fahle. Click on Install and restart IDE



Sketch

```
#include <LedControl.h>
LedControl lc=LedControl(12,11,10,1); //Pins: DIN,CLK,CS,# of Display(s) connected
unsigned long delayTime=200;  // Delay between Frames

// Put values in arrays
byte invader1a[] = {
   B00011000,              // First frame of invader #1
   B00111100,
   B01111110,
   B11011011,
   B11111111,
   B00100100,
   B01011010,
   B10100101
};

byte invader1b[] = {
  B00011000,               // Second frame of invader #1
  B00111100,
  B01111110,
  B11011011,
  B11111111,
  B00100100,
  B01011010,
  B01000010
};

void setup() {
  lc.shutdown(0,false);    // Wake up displays
  lc.setIntensity(0,5);    // Set intensity levels
  lc.clearDisplay(0);      // Clear Displays
}

// Take values in Arrays and Display them
void sinvader1a() {
  for (int i = 0; i < 8; i++) {
    lc.setRow(0,i,invader1a[i]);
  }
}

void sinvader1b() {
  for (int i = 0; i < 8; i++) {
    lc.setRow(0,i,invader1b[i]);
  }
}

void loop() {
    sinvader1a();          // Put #1 frame
    delay(delayTime);
    sinvader1b();          // Put #2 frame
    delay(delayTime);
```

```
}
```

References

https://www.arduino.cc/
http://www.resistorguide.com/
https://learn.adafruit.com/
https://en.wikipedia.org/wiki
https://learn.sparkfun.com/
https://www.thoughtco.com
https://www.ledsupply.com
https://www.build-electronic-circuits.com
https://www.makeuseof.com/