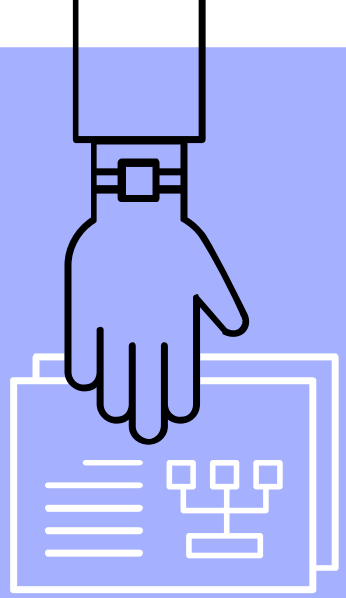
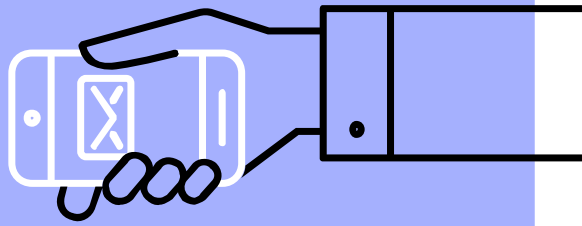
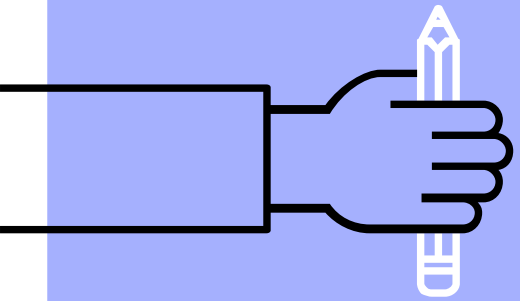
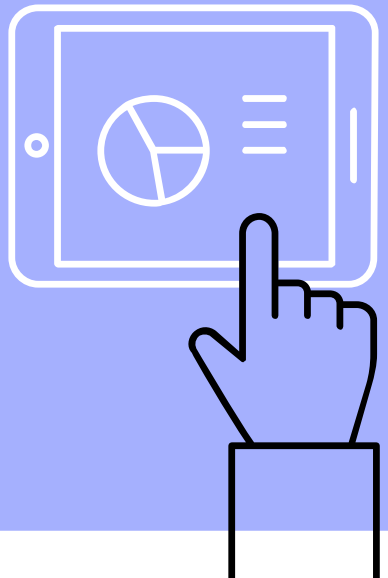




Arduino 2



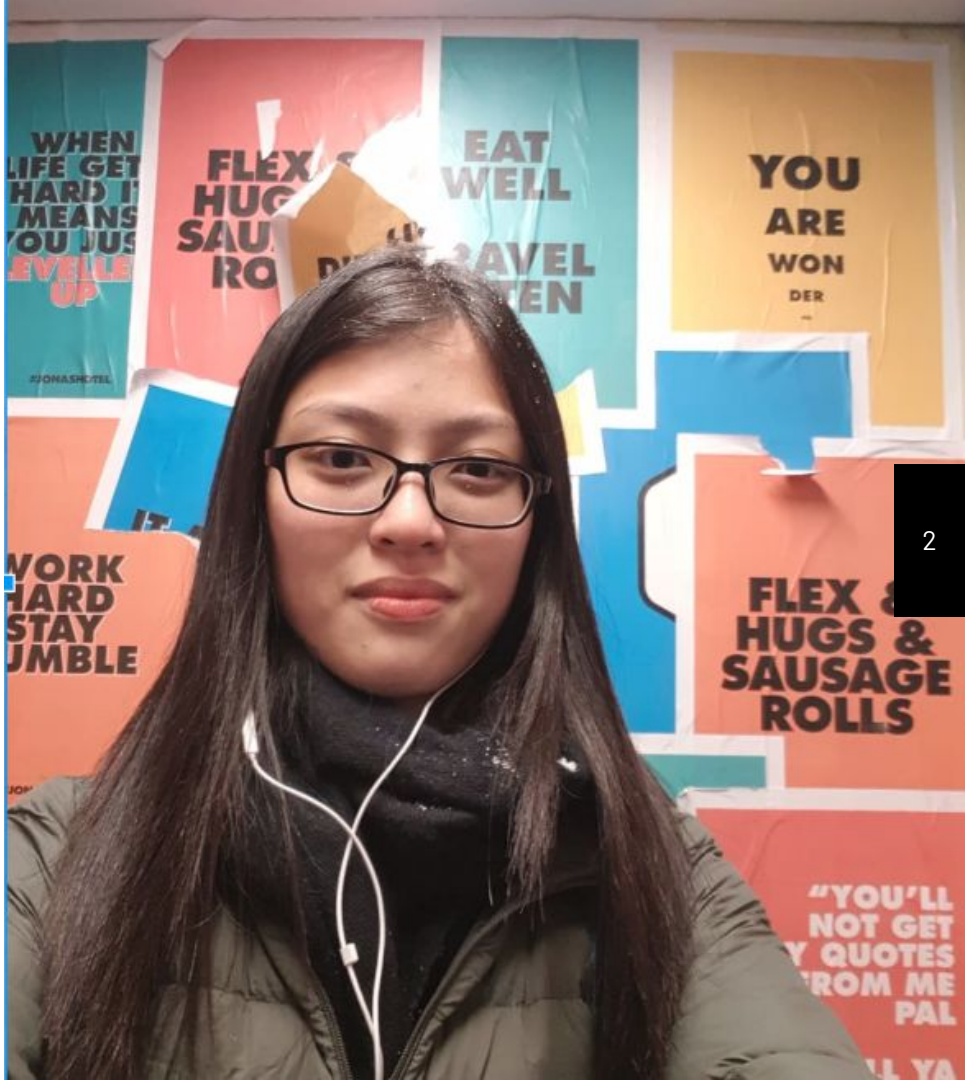


Hello World!

Katherine Kee, Wan Ting

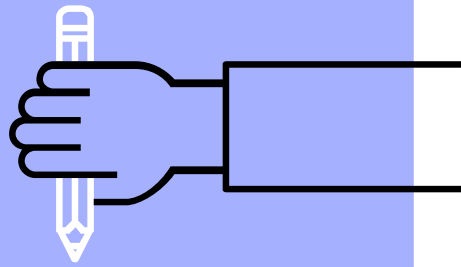
CS Yr 1

MSP, Automation Engineer Intern @PBA

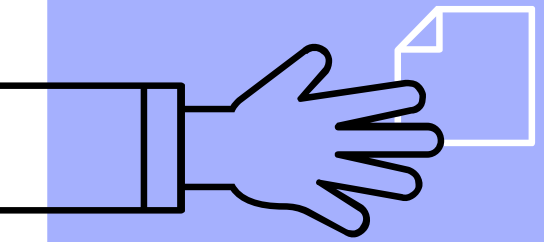


Topics we are covering

- Libraries
- Pullup
- Tones in Buzzer
- Troubleshooting Methods
- Ultrasonic Sensor
- Fritzing (Maybe)



Libraries



how it provides functionality
in the Arduino environment

A library is a collection of ***non-volatile*** *resources, for* ***software development****. Including configuration data, documentation, help data, message* ***templates****,* ***pre-written code*** *and subroutines, classes, values or type specifications.*

Library

- Collection of Pre-Compiled Routines (aka Modules)
- Stores frequently used routines
- Automatic linking (easy to add to programs)

Guide to write your own libraries:
https://www.arduino.cc/en/Reference/APIStyleGuide

E.g: Morse Code Library

Download:

https://www.arduino.cc/en/uploads/Hacking/Morse.zip

```
/*
  Morse.cpp - Library for flashing Morse code.
  Created by David A. Mellis, November 2, 2007.
  Released into the public domain.
*/

#include "Arduino.h"
#include "Morse.h"

Morse::Morse(int pin)
{
  pinMode(pin, OUTPUT);
  _pin = pin;
}

void Morse::dot()
{
  digitalWrite(_pin, HIGH);
  delay(250);
  digitalWrite(_pin, LOW);
  delay(250);
}

void Morse::dash()
{
  digitalWrite(_pin, HIGH);
  delay(1000);
  digitalWrite(_pin, LOW);
  delay(250);
}
```

Libraries

Using custom libraries helps to reduce the amount of code you have to write.

Best used for codes that you will write frequently across various programs!

```
#include <Morse.h>

Morse morse(13);

void setup()
{
}

void loop()
{
  morse.dot(); morse.dot(); morse.dot();
  morse.dash(); morse.dash(); morse.dash();
  morse.dot(); morse.dot(); morse.dot();
  delay(3000);
}
```

Standard Libraries

Ready to download/Pre-installed into your arduino

EEPROM
Ethernet
Firmata
GSM
LiquidCrystal
SD
Servo
SPI
SoftwareSerial
Stepper
TFT
WiFi
Wire

Ethernet - for connecting to the internet using the Arduino Ethernet Shield, Arduino Ethernet Shield 2 and Arduino Leonardo ETH

LiquidCrystal - for controlling liquid crystal displays (LCDs)

Servo - for servo motors

Stepper - for stepper motors

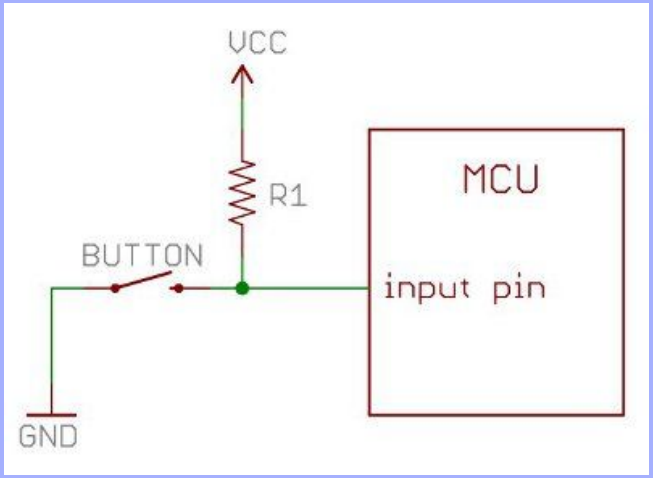
WiFi - for connecting to the internet using the Arduino WiFi shield

Pullup

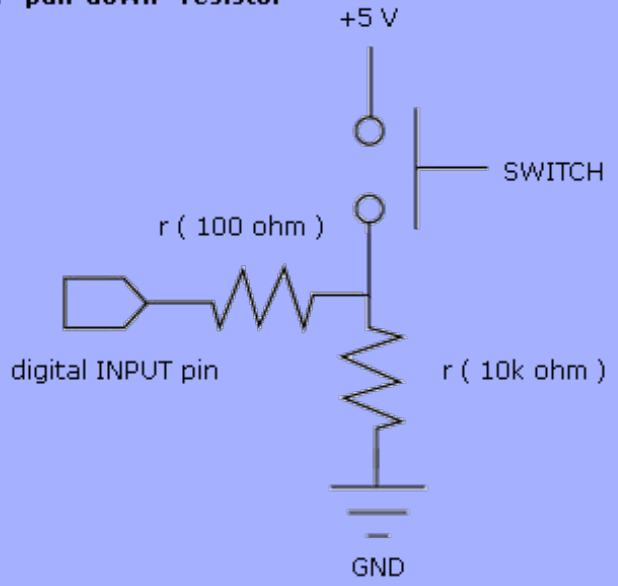
If there is **nothing connected** to the pin and your program **reads the state** of the pin, will it be high (pulled to VCC) or low (pulled to ground)?

This phenomena is referred to as floating.

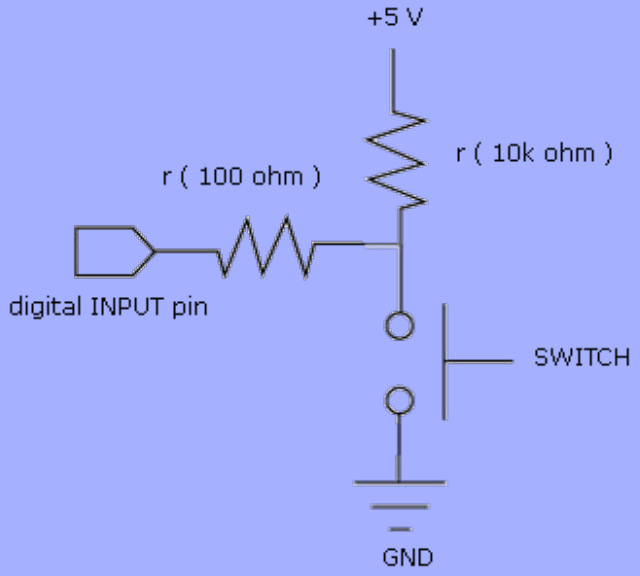
To prevent this unknown state, a **pull-up** or **pull-down** resistor will ensure that the pin is in either a high or low state, while also using a low amount of current.



Switch with "pull-down" resistor



Switch with "pull-up" resistor



Pull-up/down Resistor

	Input = 0 (Not Pressed)	Input = 1 (Pressed)
Not Pulled	May short circuit!	
Pull-up Resistor	HIGH	LOW
Pull-Down Resistor	LOW	HIGH

Internal Resistors

Many MCUs, like the ATmega328, have internal pull-ups that can be enabled and disabled.

pinMode(5, INPUT_PULLUP);

// Enable internal pull-up resistor on pin 5

//An internal 20K-ohm resistor is pulled to 5V.

//This configuration causes the input to read HIGH when the switch is open, and LOW when it is closed.

Calculating pull-up resistor using Ohm's Law

$$V = I \cdot R$$

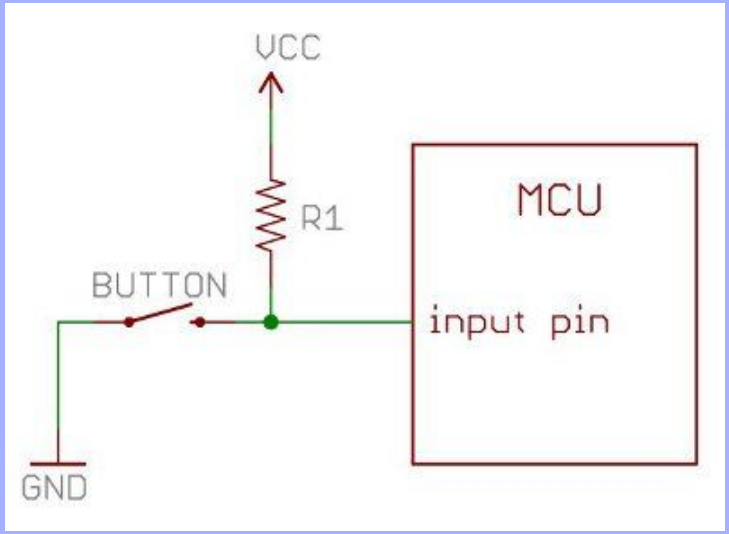
Referring to the schematic above, Ohm's Law now is:

$$V_{CC} = (current\ through\ R1) \cdot R1$$

Rearrange the above equation with some simple algebra to solve for the resistor:

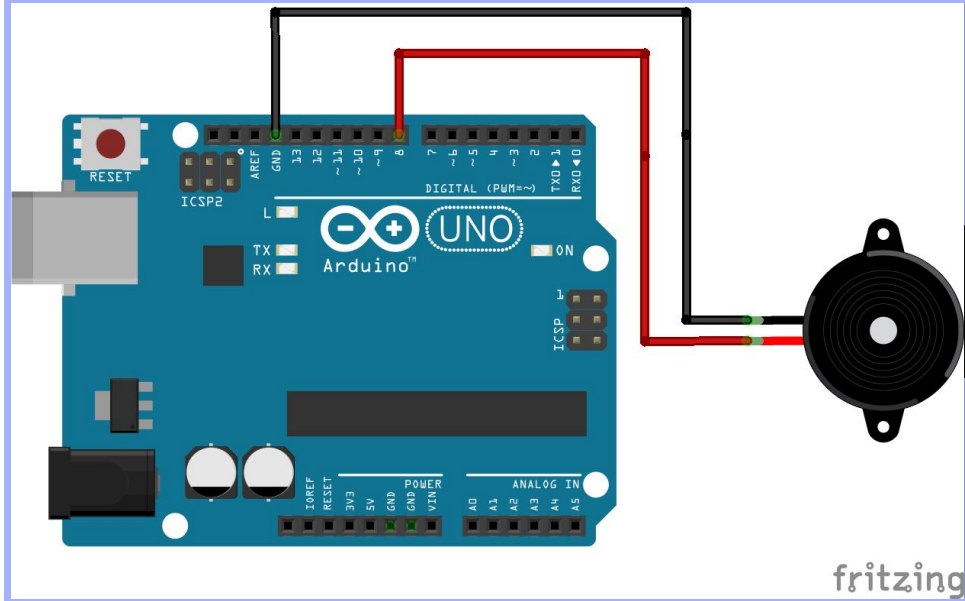
$$R1 = \frac{V_{CC}}{current\ through\ R1} = \frac{5\text{V}}{0.001\text{A}} = 5\text{k}\,Ohms$$

Remember to convert all of your units into volts, amps and Ohms before calculating (e.g. 1mA = 0.001 Amps). The solution is to use a 5kΩ resistor.



Now for some fun!

Please take our your buzzers!



Tones

tone(pin, frequency, duration)

https://www.princetronics.com/supermariothemesong/

https://github.com/AbhishekGhosh/Arduino-Buzzer-Tone-Codes/blob/master/star-wars.ino

TroubleShooting

Troubleshooting Methods

Beginning

-Driver

-Access to Serial Port/Physical Connection

Midway

-Look at **error messages**

-Syntax errors are the most common ['/' -> '\']

Tips:

-Check Serial Monitor

https://www.arduino.cc/en/Guide/Troubleshooting

Serial Monitor

Serial_01 | Arduino 1.6.7

File Edit Sketch Tools Help

Serial_01

```
int i = 0;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

}

void loop() {
   // put your main code here, to run repeatedly:
      if( i < 10) {
        Serial.println("Hello World !");
        delay(1000);
        i++;
      }
}
```

Done Saving.

```
Sketch uses 1,994 bytes (6%) of program storage space. Maximum is 32,
Global variables use 216 bytes (10%) of dynamic memory, leaving 1,832
```

5 Arduino/Genuino Uno on COM3

COM3 (Arduino/Genuino Uno)

Send

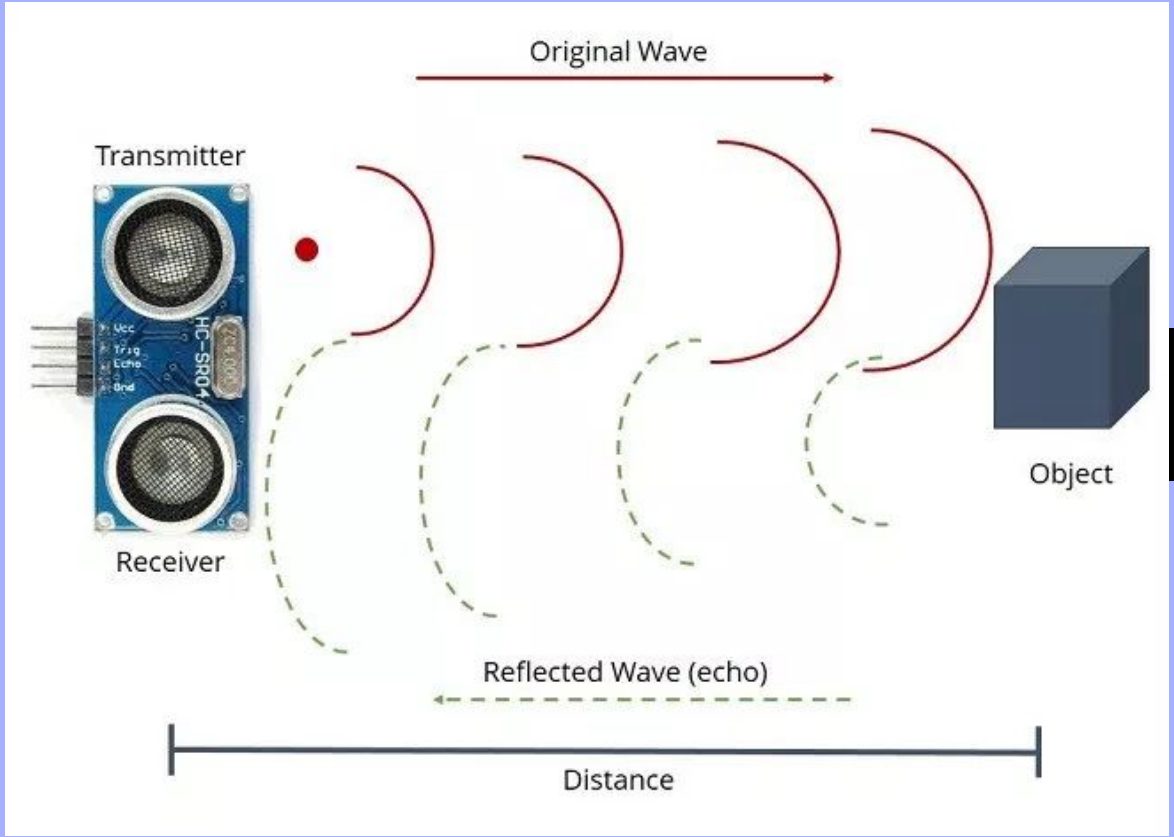
```
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
```

Autoscroll No line ending 9600 baud

Ultra-sonic Sensor

Ultrasound is sound waves with frequencies **higher** than the upper audible limit of human hearing.

Distance measured by time.

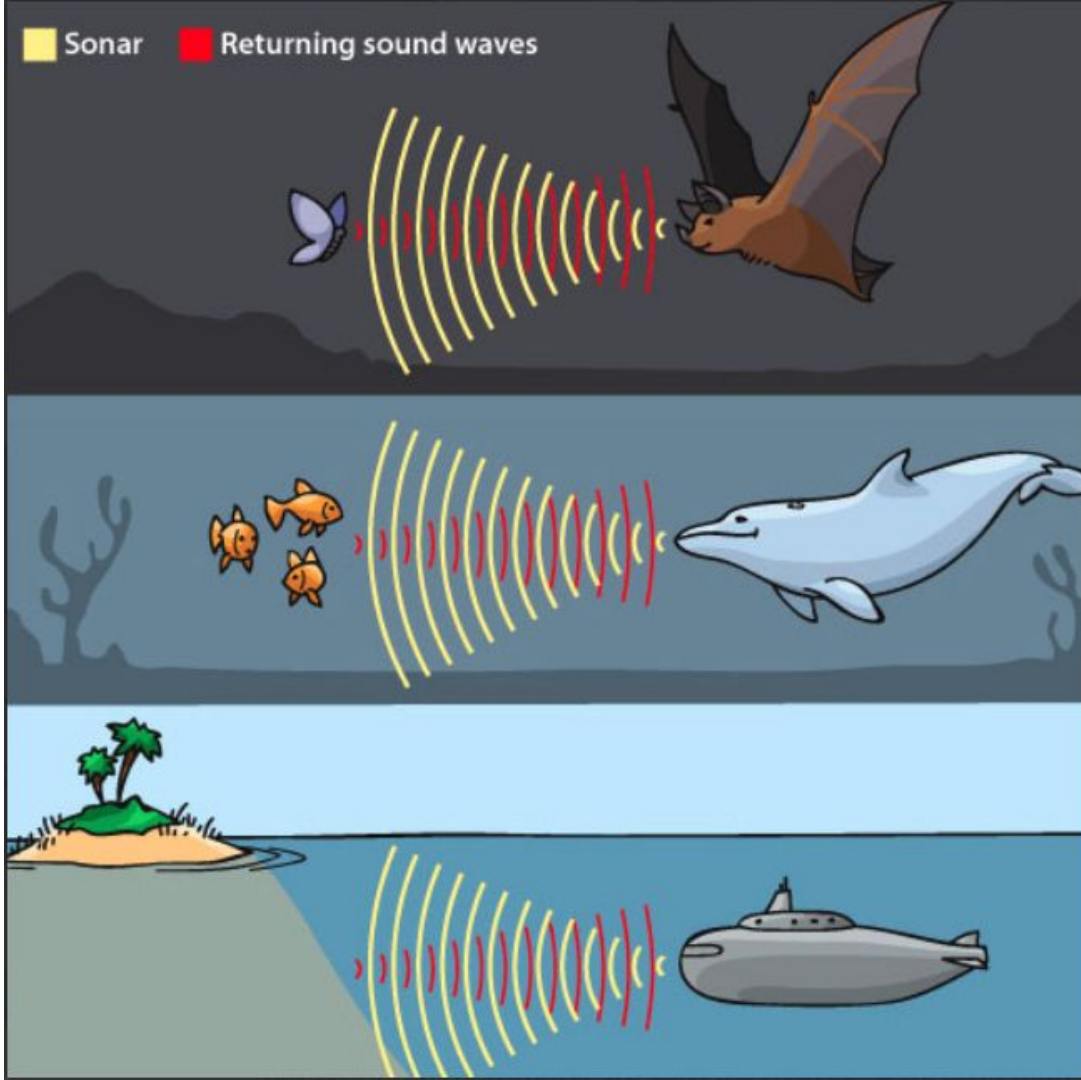


Uses in “Nature”

Detects objects by measuring the time it takes for the returning sound waves to reach back to it after bouncing of it.

Distance = Speed * Time/2

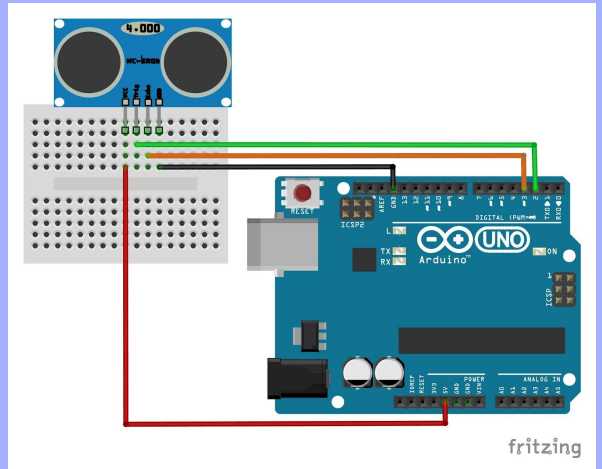
*This formula

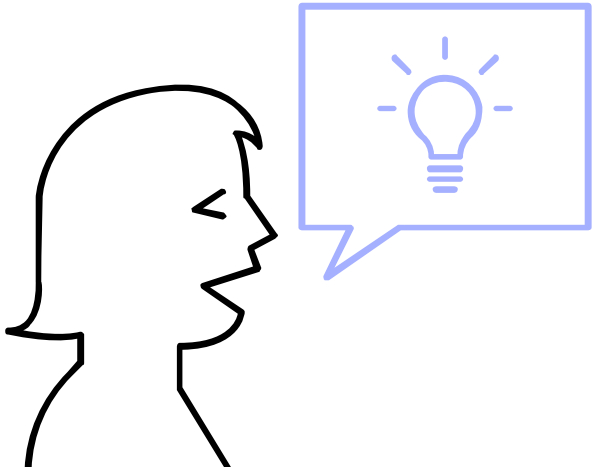


HC-SR04 - Ultrasonic Sensor

4 pins: 5V, trigger, echo, GND

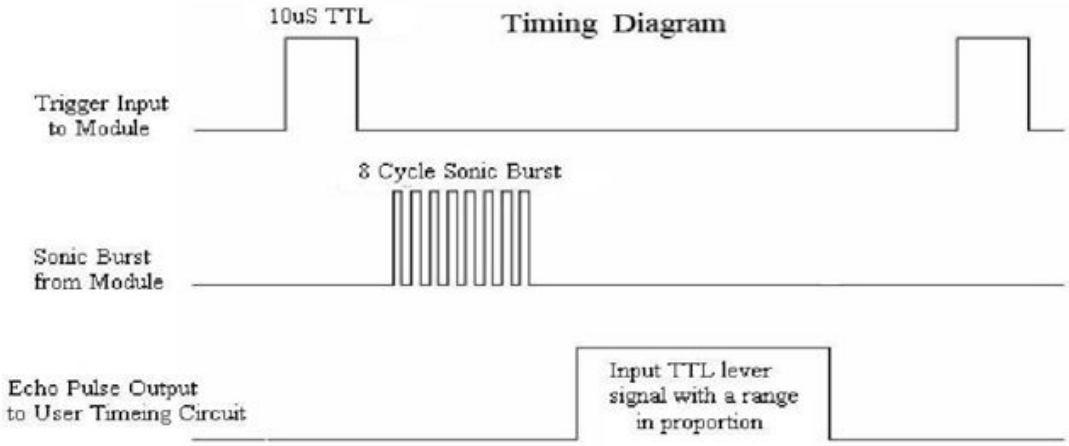
trigger pin > sound off a wave pulse,
echo pin > receive wave pulse

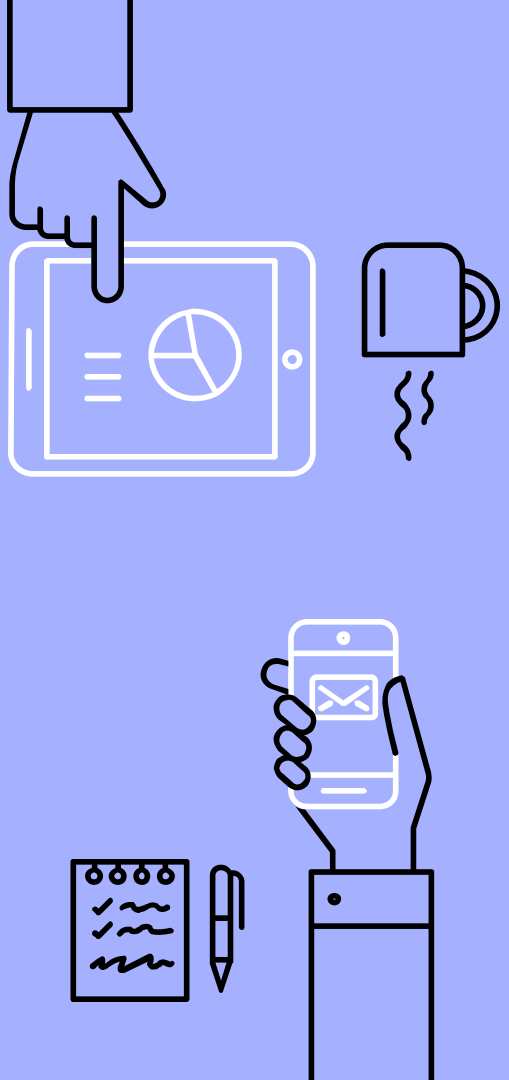




Timing

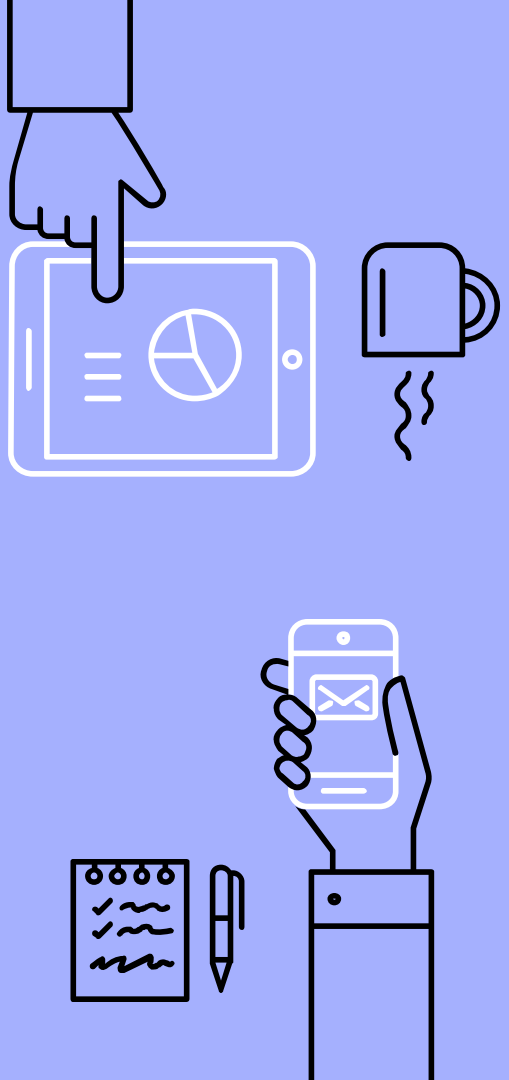
The value is divided by two since the wave travels forward and backwards covering the same distance

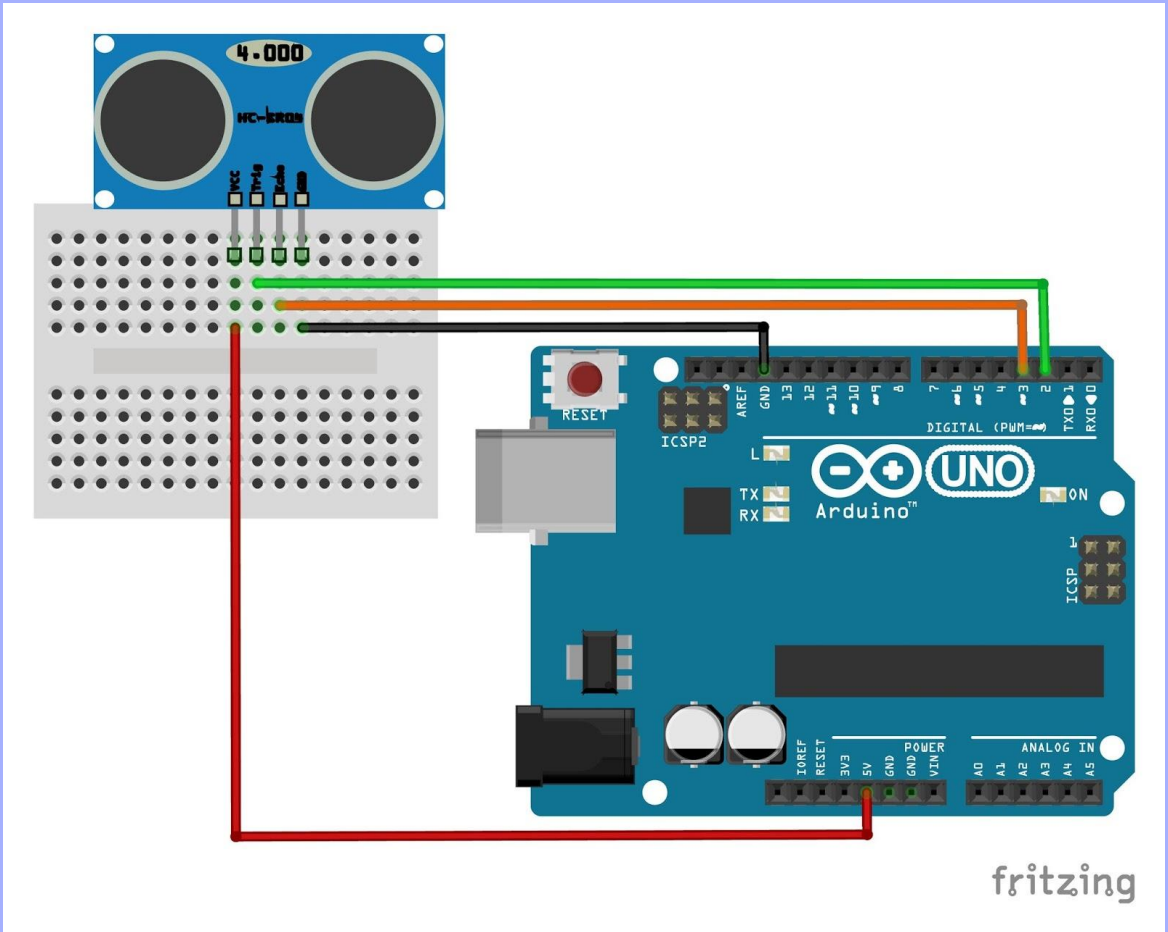




What we are doing

You need to **initialize serial communication** for you to see the distance that your ultrasonic sensor is measuring.
Establish variables for duration of the **ping (pulse that is produced by the trigger)** and the distance result in inches and centimeters.
The sensor is triggered by a **HIGH pulse of 10 or more microseconds.**
Before that, give a short LOW pulse to ensure a **clean** HIGH pulse.
Read the signal from the sensor: a HIGH pulse whose duration is the time (in microseconds) from the sending of the ping to the reception of its echo off an object.




4.000
HC-SR04
Vcc
Trig
Echo
GND
RESET
ICSP2
AREF
GND
13
12
11
10
9
8
7
6
5
4
3
2
1
0
TX0
RX0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
fritzing

Code Part 1

Initialize serial communication

Establish variables

```
const int trigPin = 2;
const int echoPin = 4;

void setup() {
  Serial.begin(9600);
  pinMode(echoPin, INPUT);
  pinMode(trigPin, OUTPUT);
}

void loop()
{
  long duration, cm;
```

Code Part 2

'LOW' and 'High' pulse is produced to "clean" before every function call

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
```

Set the default to LOW

```
digitalWrite(trigPin, LOW);
```

Function Call: Read the signal
HIGH pulse whose duration is the time (in microseconds) from the sending of the ping to the reception of its echo off an object.

```
duration = pulseIn(echoPin, HIGH);
```

Code Part 3

sound waves travel at 340m/s

```
cm = duration * 0.034/2;
```

prints out

```
  Serial.print(cm);
  Serial.print(" cm");
  Serial.println();

  delay(100);
}
```

Speed Facts				
	Meter per Second	*Kilometers per Hour*	*Feet per Second*	*Miles per Hour*
Sound* (at sea level)	340	1,225	1,116	761
Light	299,792,458	1,079,252,849	983,571,056	670,616,629

Car reverse warning system

Ultrasonic Sensor

Buzzer

LED

- 3 states:
- Slow Blink - not near
- Normal Blinking - near
- Quick Blinking - at limit

Car reverse warning system

trigPin = 2;

echoPin = 4;

led = 8;

buzzer = 7;

```
const int trigPin = 2; //ultra trig int
const int echoPin = 4; //ultra echo int

void setup() {
  Serial.begin(9600);
  pinMode(8, OUTPUT); //led
  pinMode (7, OUTPUT);//buzzer

void loop()
{
  long duration, cm;

  pinMode(trigPin, OUTPUT);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  cm = duration * 0.034/2;
```

Car reverse warning system

if.. else loops

```
Serial.print(cm);
Serial.print(" cm");
Serial.println();
```

```
digitalWrite(8, LOW);
  delay(700);
        if(cm >= 25){
  tone(7, 500, 500);
  digitalWrite(8, HIGH);
  delay(500);
  digitalWrite(8, LOW);
  delay(500);
}       else if(cm >= 10){
  tone(7, 500, 150);
  digitalWrite(8, HIGH);
  delay(150);
  digitalWrite(8, LOW);
  delay(150);
}       else {
  tone(7, 700, 100);
  digitalWrite(8, HIGH);
  delay(10);
  digitalWrite(8, LOW);
  delay(10);
}
}
```

Summary of Code

<u>Initialize</u>
trigPin = 2
echoPin = 4
led = 8, buzzer = 7
variables:
duration, cm

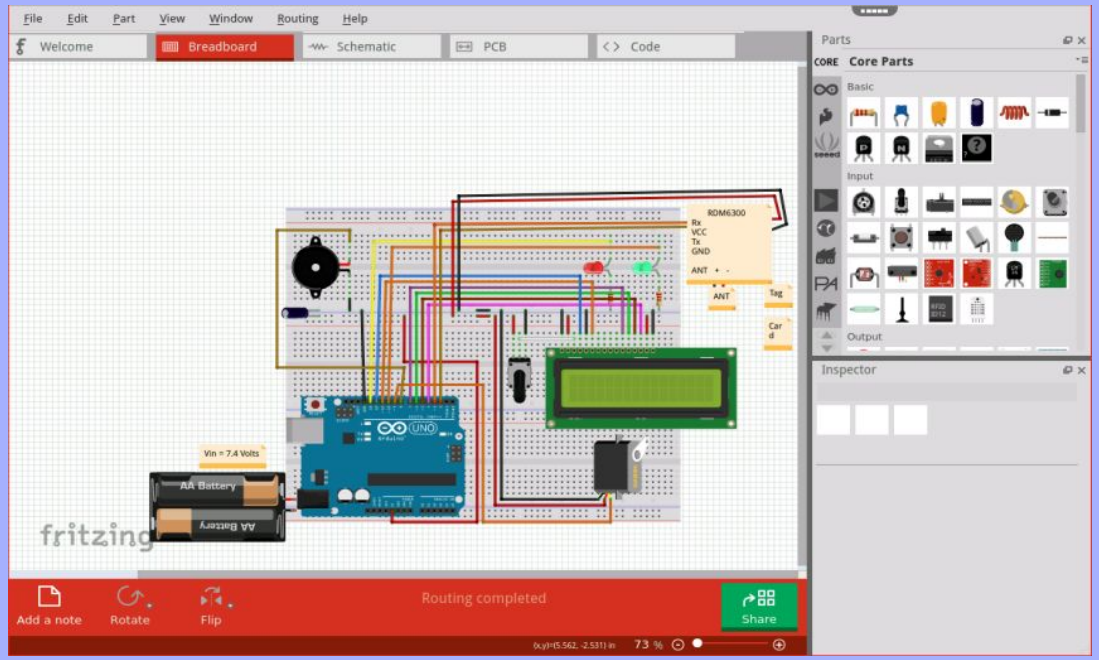
'Clearing' by
outputting
'HIGH' & 'LOW'
Calculating values

if >= 25cm
...

else if >= 10cm
...

else...

Bonus: Frizting



Discussing Future Workshops

JavaScript Programming 101 (for Web App dev)

Intro to Raspberry Pi **[$$$]**

Arduino 3a: Transmitter and Receiver wth nRF24L01

Potentiometer & Servomotor **[$]**

Arduino 3b: Bluetooth Transmissions with Android App

Machine Learning - TensorFlow on Google Cloud

ROS: Robotics Operating System

Image detection/ID with OpenCV on Visual Studios (Car Licence Plate)

Thank You

Want the slides? Pls give me feedback:
https://tinyurl.com/MAERCarduino2

https://www.facebook.com/ntu.mae.rc

https://www.instagram.com/mae.robotics/