Embedded Programming Using Arduino

Mr Ganesh Attarde

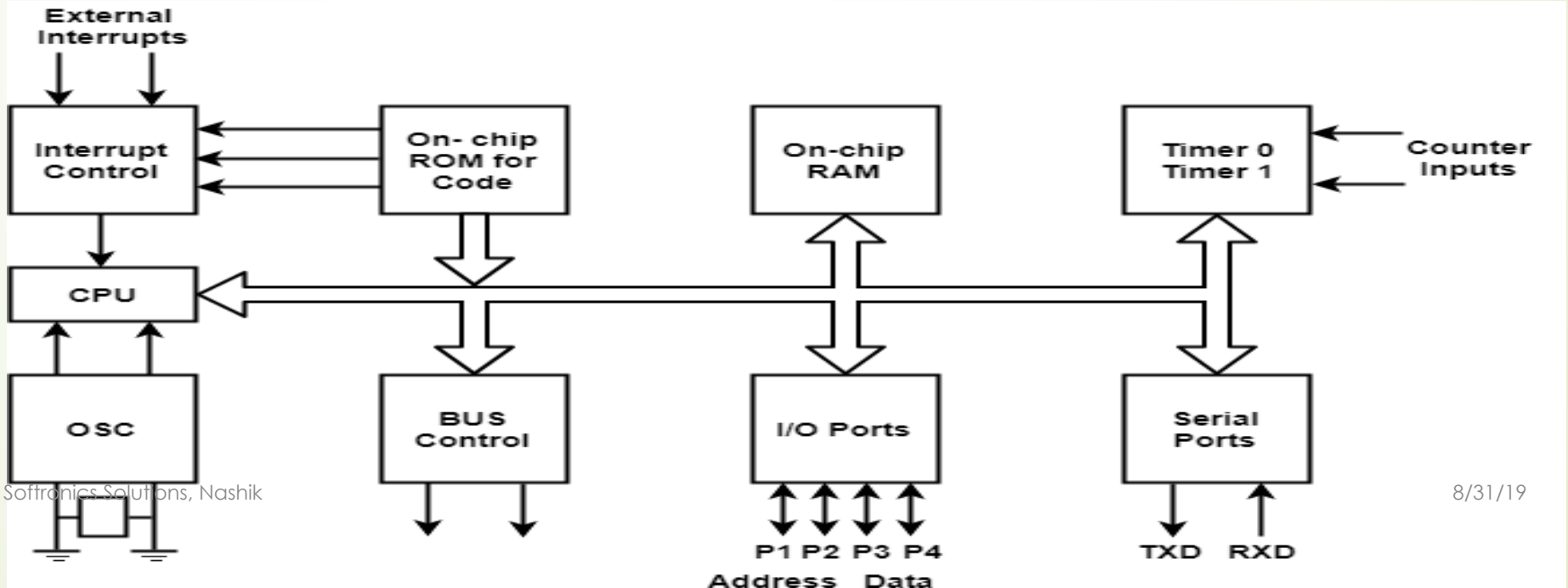
GB Softronics Solutions, Nashik

Outline:

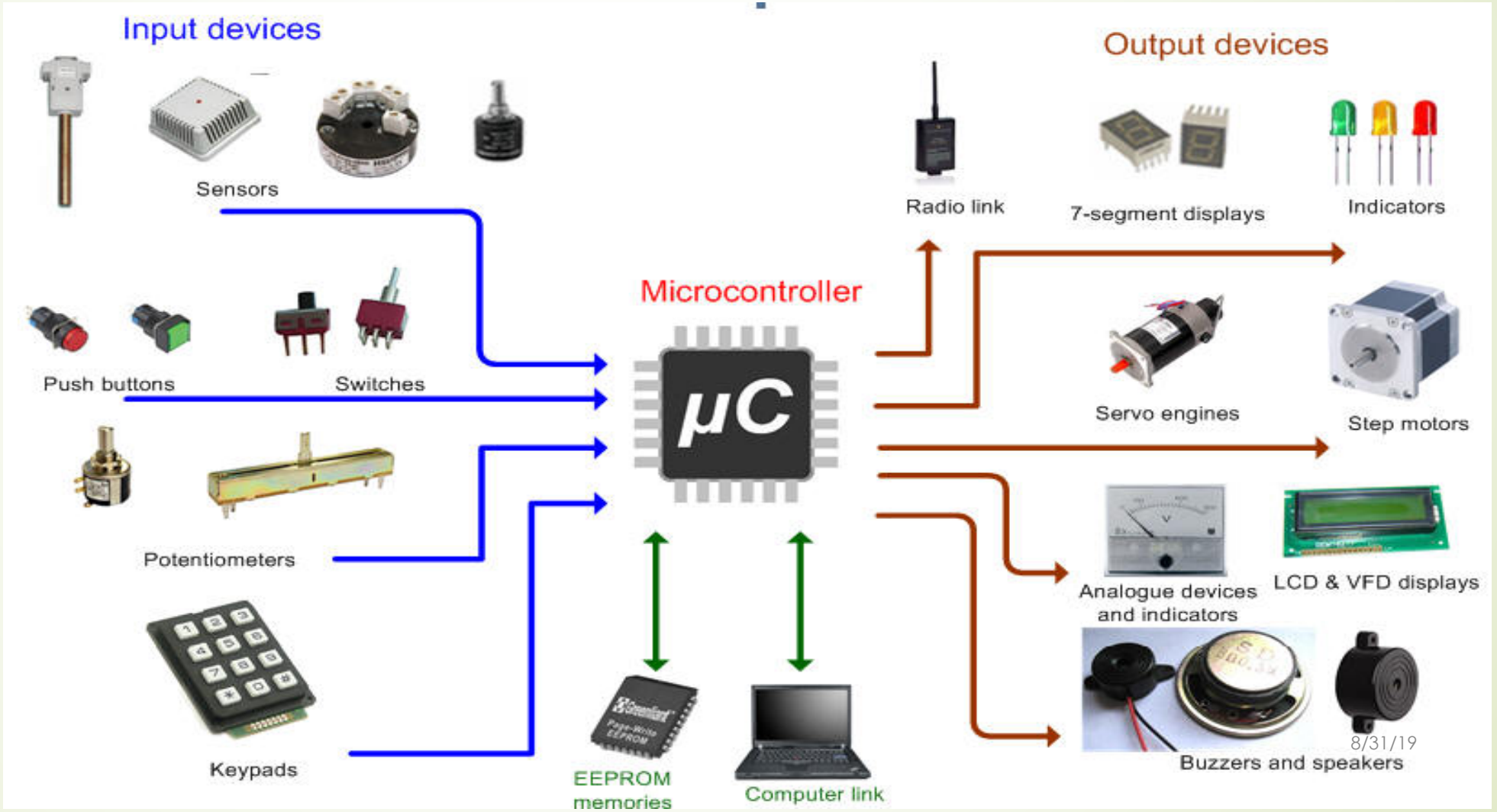
- Day 1:
  - Introduction to Microcontrollers
  - Role of Embedded Systems
  - Introduction to Breadboard
  - Arduino Introduction
  - LED Interfacing
  - Hands on LED Interfacing
  - Serial Communication
  - Hands on Serial Communication
  - Bicolor LED Interfacing
  - RGB LED Interfacing

Introduction to Microcontrollers

- A microcontroller is a computer present in a single integrated circuit which is dedicated to perform one task and execute one specific application.
- It contains memory, programmable input/output peripherals as well a processor. Microcontrollers are mostly designed for embedded applications and are heavily used in automatically controlled electronic devices such as cellphones, cameras, microwave ovens, washing machines, etc. *[Ref:www.Techopedia.com]*



Applications of Microcontroller



What is an Embedded System?

- The integration of:
  - Sensors
  - Actuators
  - Intelligence
- with a system to produce:
  - More capable, versatile, and robust performance

Embedded Systems Applications

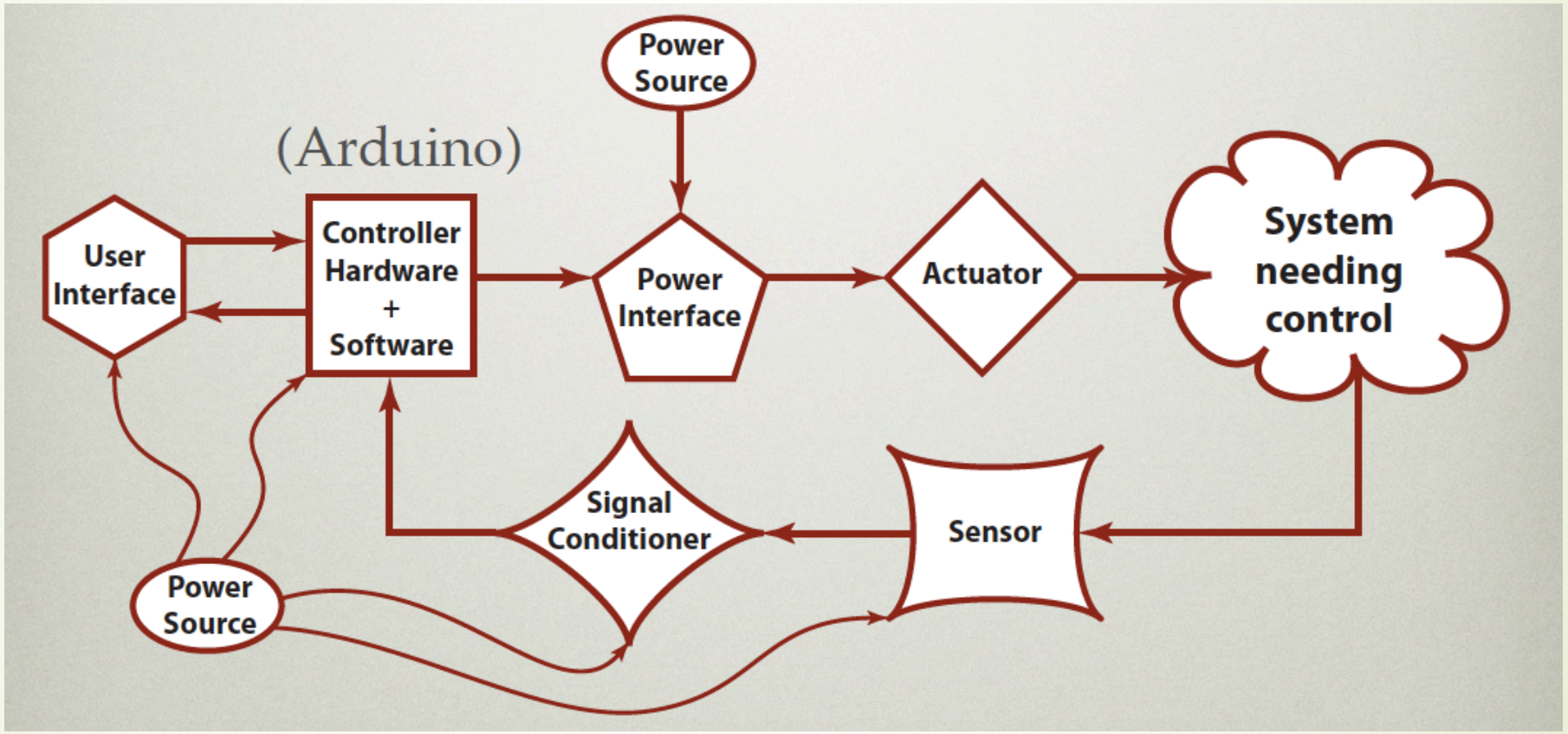
A microcontroller is a correct tool to use when:

- Intelligence is required in the system.
- The complexity of a system is reduced when using one.
- The cost of the microcontroller is “less” than using discrete components to do the same job.
- A variety of sensors and actuators must be integrated in the system.
- Communication with other devices is necessary.

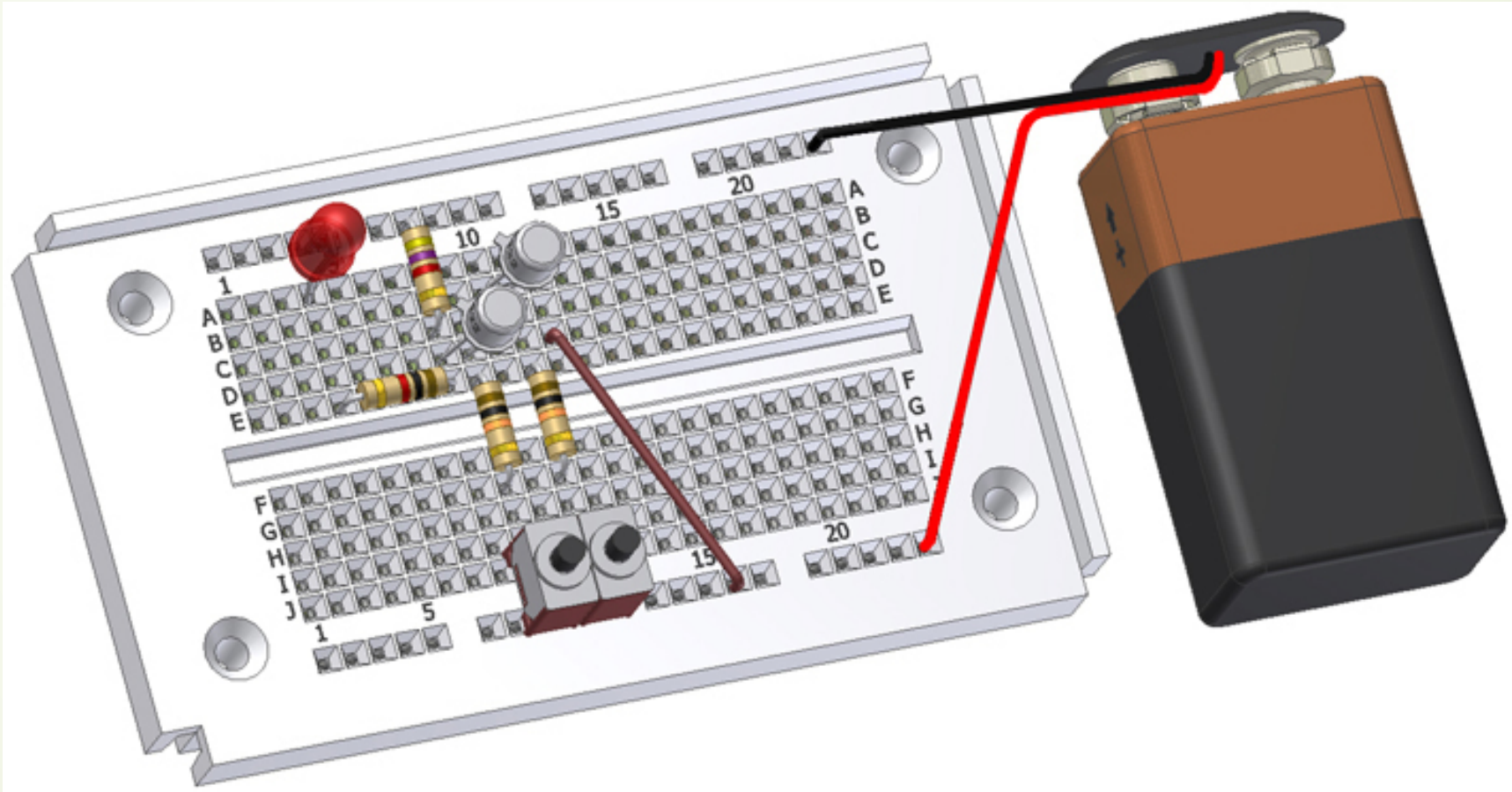
A microcontroller is NOT the correct tool to use when:

- The system requires little or no intelligence.
- The system can be made easier and/or cheaper using discrete-components

The Embedded System Concept Map

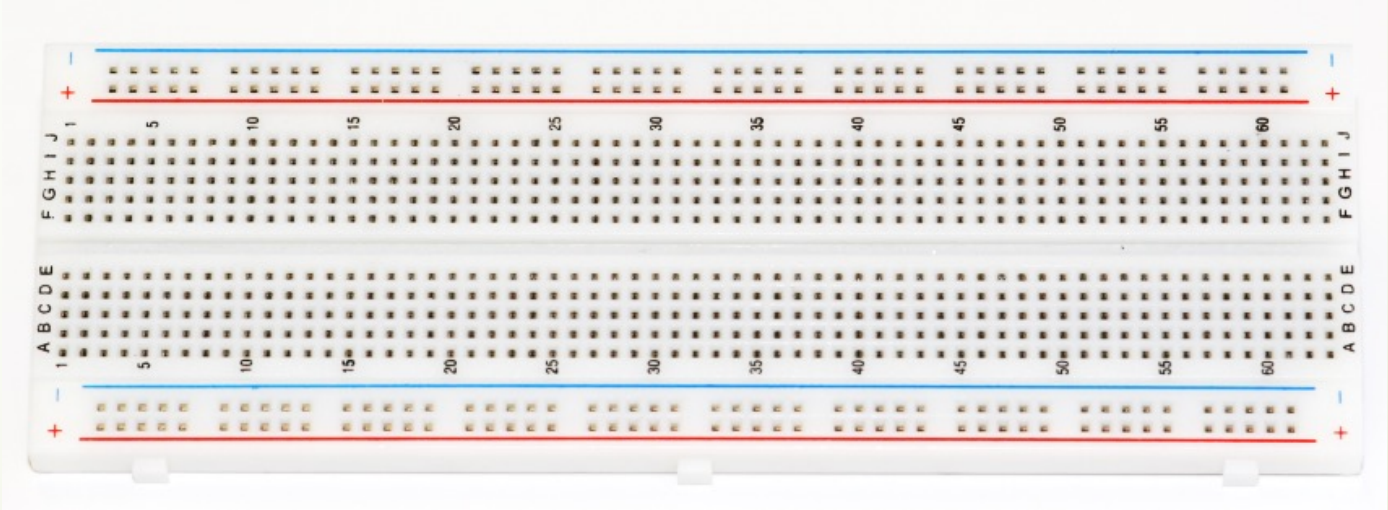


Breadboard Basics



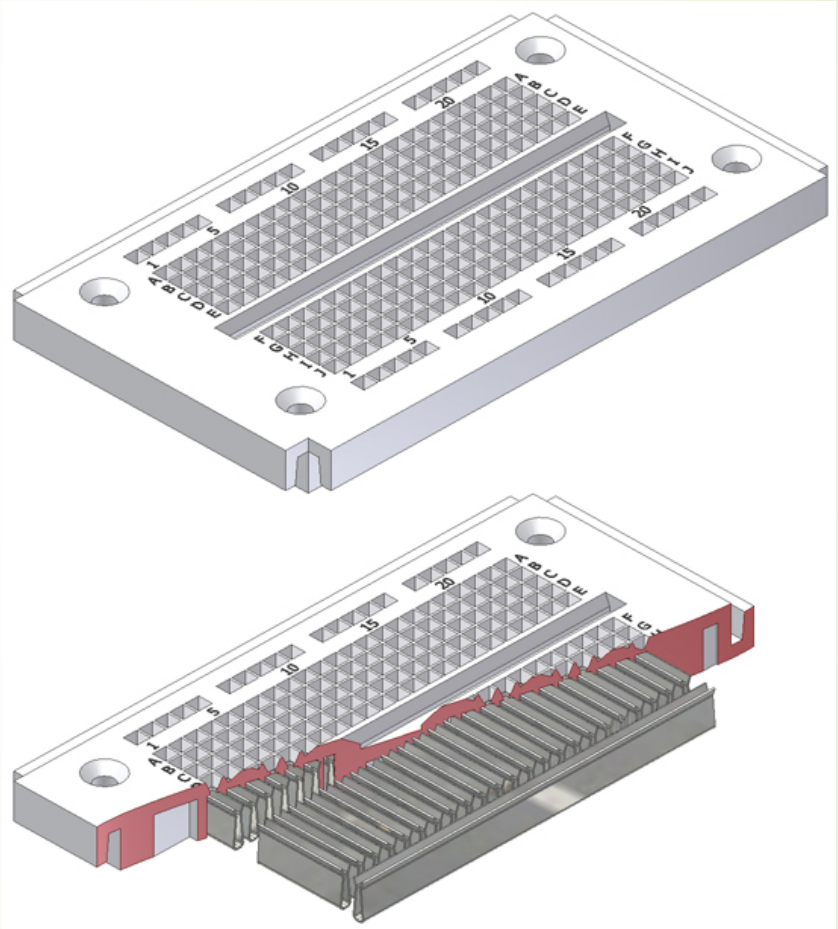
What is Bradboard?

- A breadboard, sometimes called a proto- board,
- It is a reusable platform for temporarily built electronic circuits.



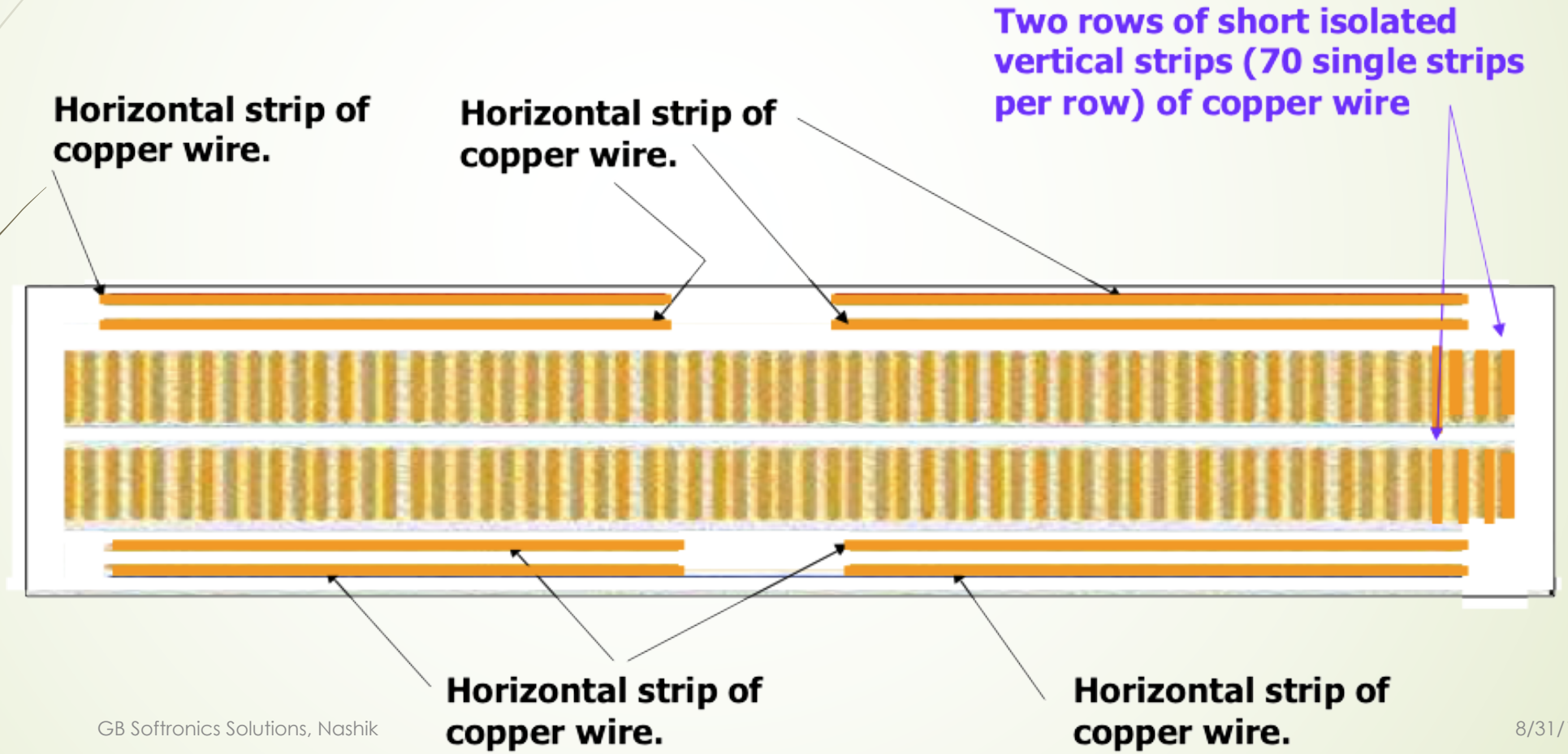
How a Breadboard Works?

- Electric component leads and the wire used to connect them are inserted into holes that are arranged in a grid pattern on the surface of the breadboard.
- A series of internal metal strips serve as jumper wires. They connect specific rows of holes.

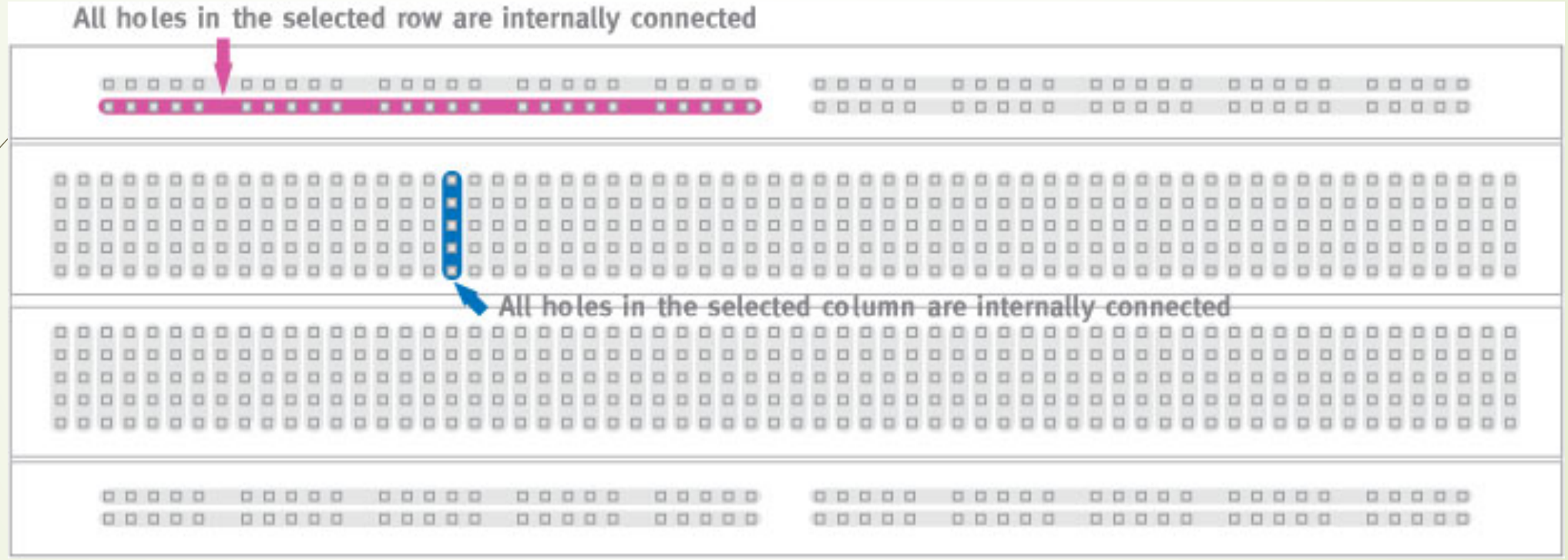


The Breadboard

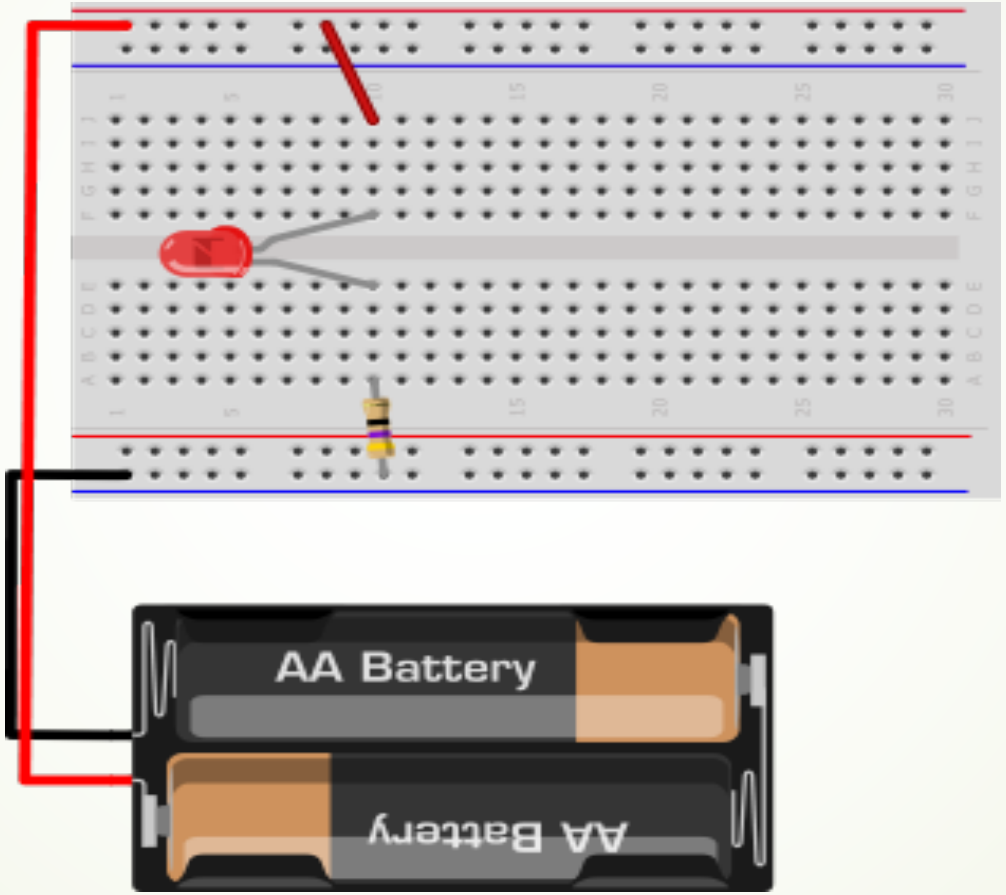
What You Don't See:



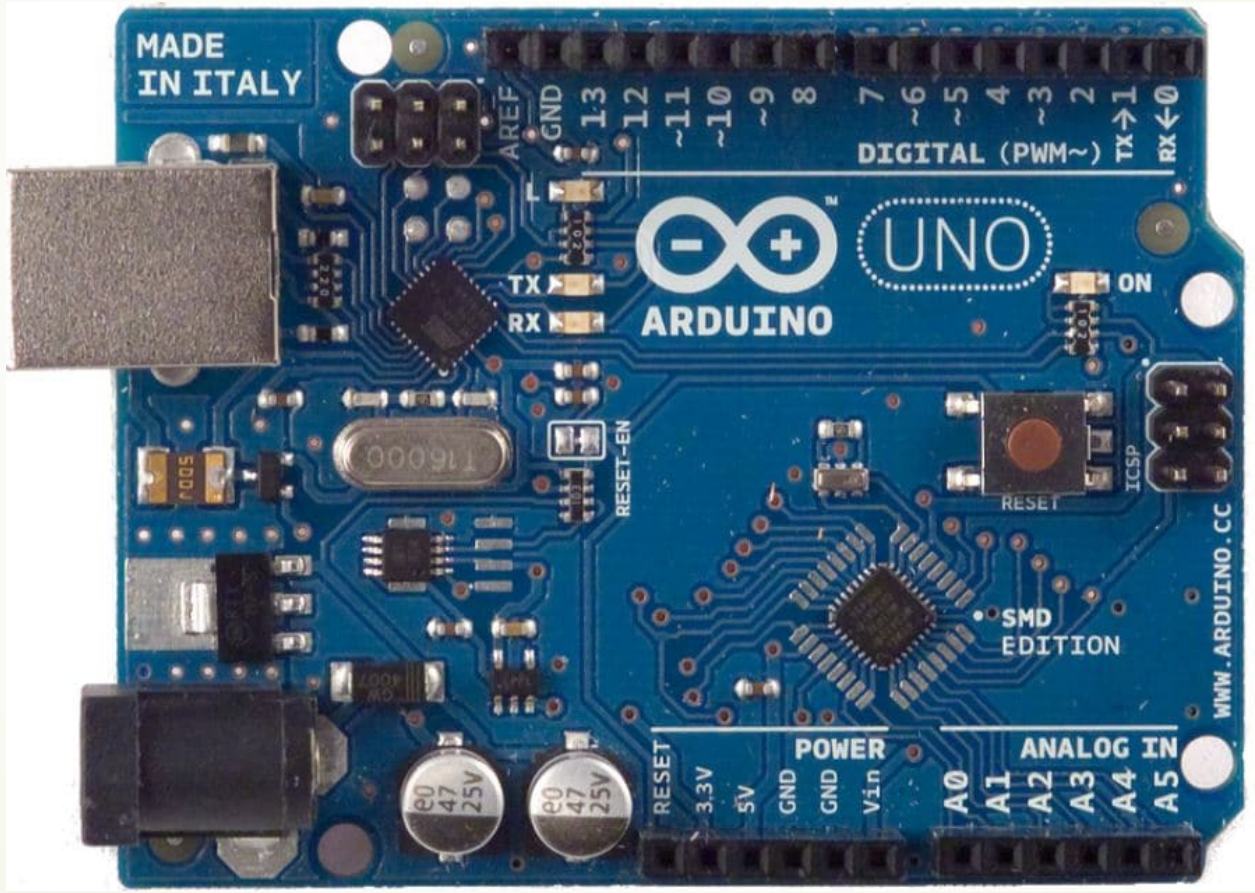
Another View



LED Connection on Breadboard



Introduction to Arduino Uno



Arduino

- Definition Taken from the official web site (arduino.cc):
  - **Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.**
- In the year 2005 that the first ever Arduino board was born in the classrooms of the Interactive Design Institute in **Ivrea, Italy**.
- The Arduino, created by **Massimo Banzi** and other founders

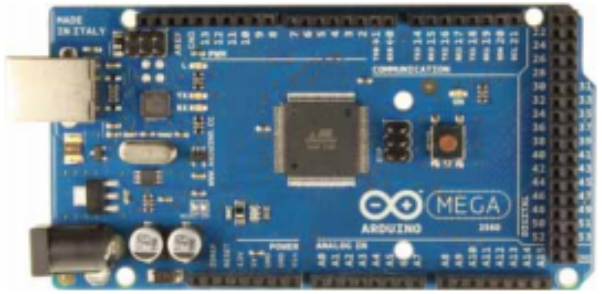
Arduino Developers

Arduino developer team - David Cuartielles, Gianluca Martino, Tom Igoe, David Mellis, and Massimo Banzi.

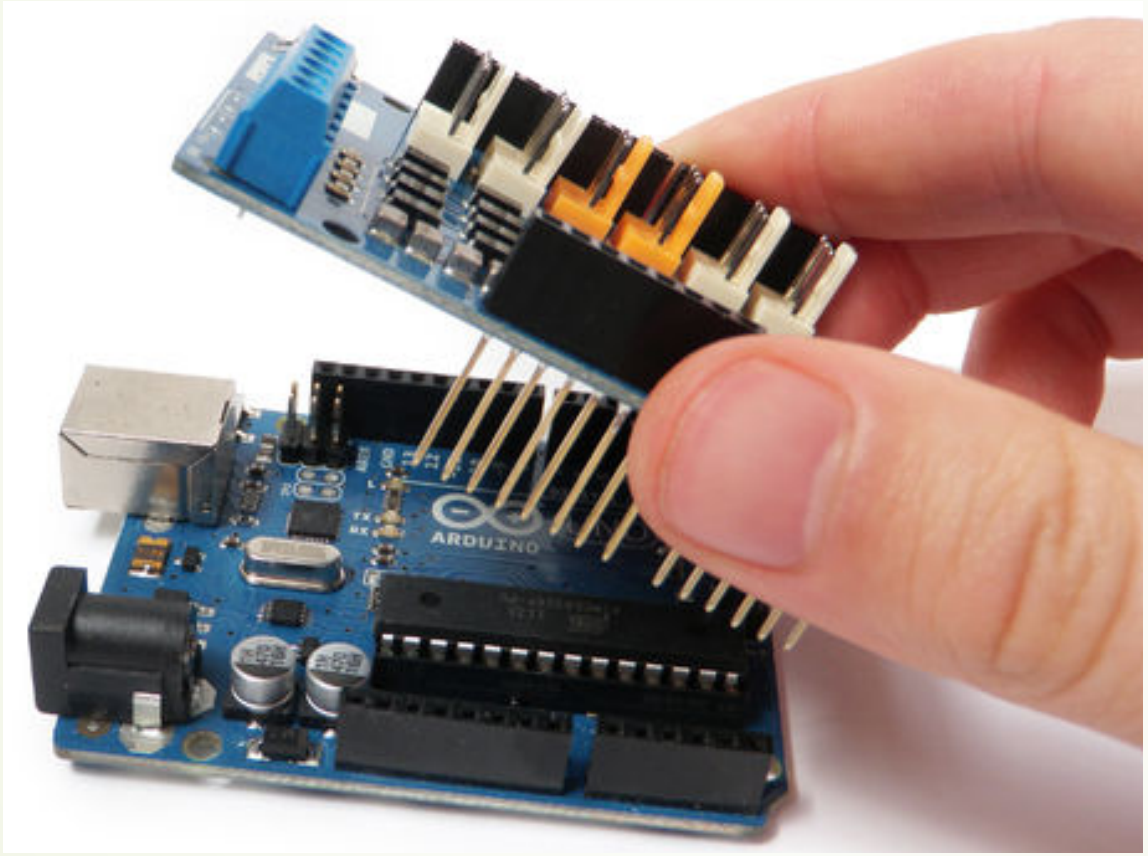
Photo Courtesy - Randi Klett/IEEE Spectrum

Types of Arduino Boards

- Arduino Uno
- Arduino Leonardo
- Arduino LilyPad
- Arduino Mega
- Arduino Nano
- Arduino Mini
- Arduino Mini Pro
- Arduino BT



Arduino Shield

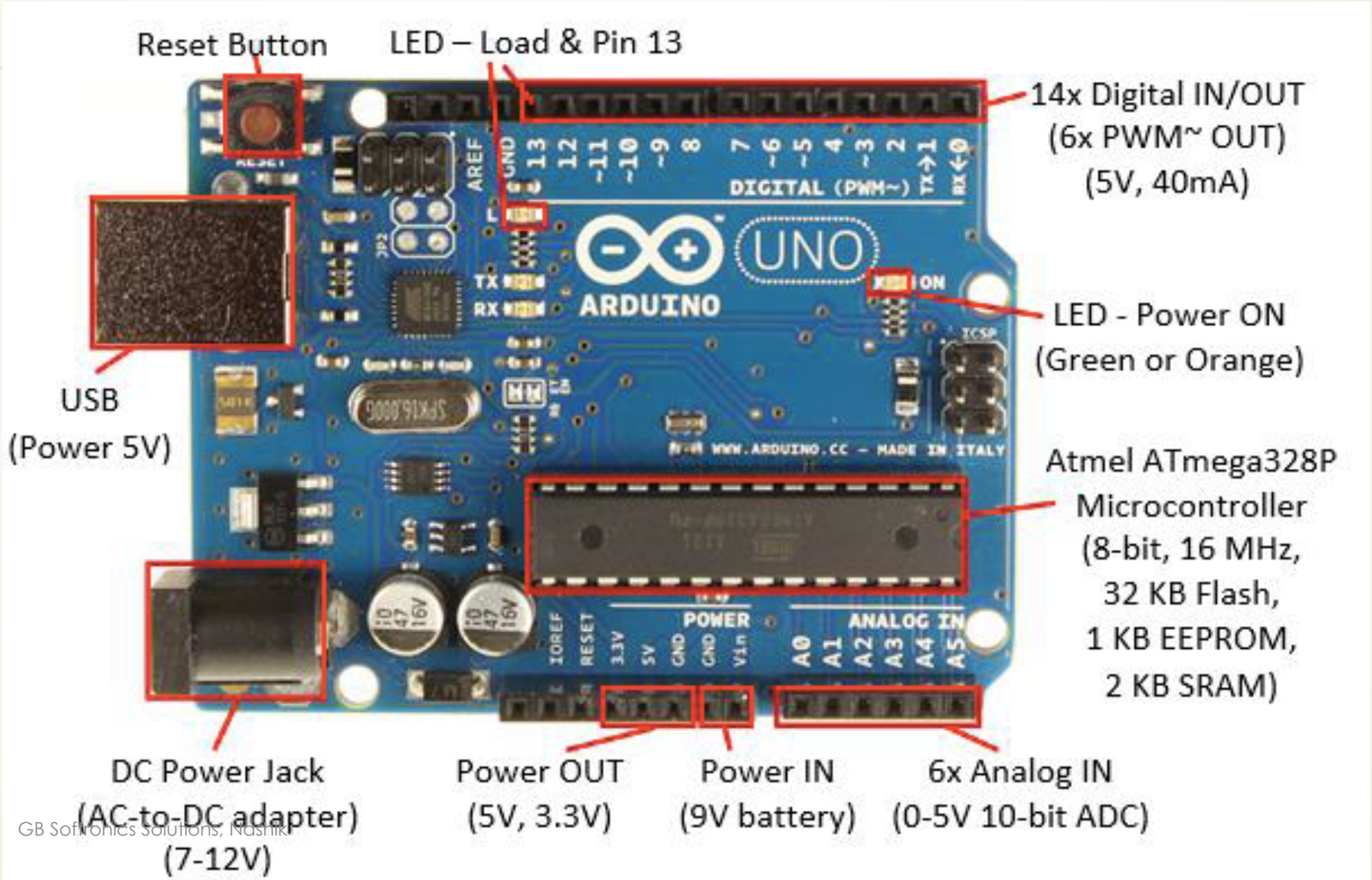


Shields are pieces of hardware that you can mount on the **Arduino** to give it a specific purpose or extra capabilities.

For example, you can use a motor **shield** to make it easier to control motors with **Arduino**, or you can use an Ethernet **shield** to connect your **Arduino** to the Internet.

More Shields

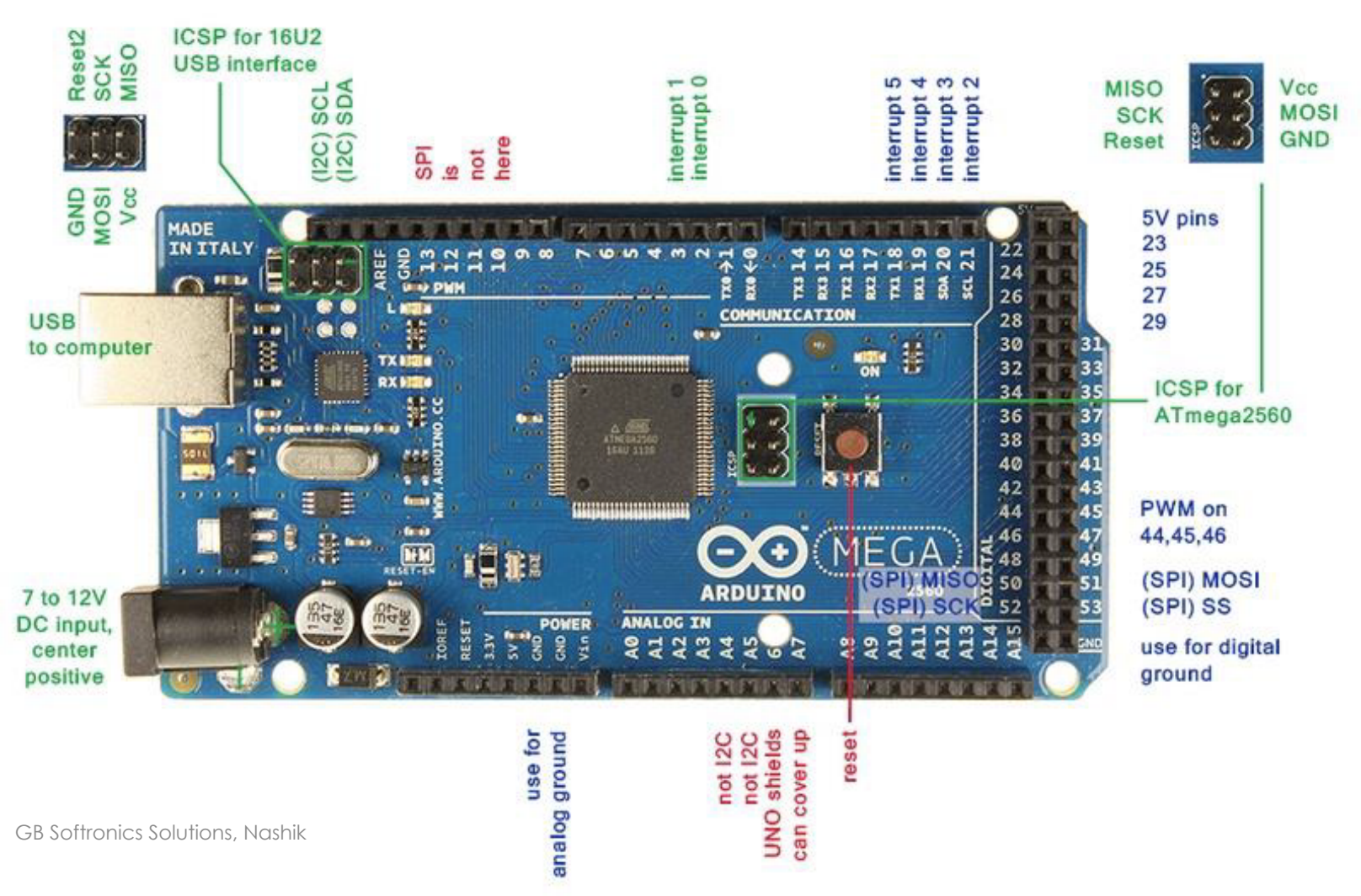
Arduino UNO Board



Specification of Arduino UNO

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40mA
DC Current for 3.3V Pin	50mA
Flash Memory	32KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2KB (ATmega328)
EEPROM	1KB (ATmega328)
Clock Speed	16MHz

Arduino Mega 2560



Specification of Arduino Mega 2560

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB (ATmega2560) of which 4KB used by bootloader
SRAM	8 KB (ATmega328)
EEPROM	4 KB (ATmega328)
Clock Speed	16 MHz

Arduino Program Development

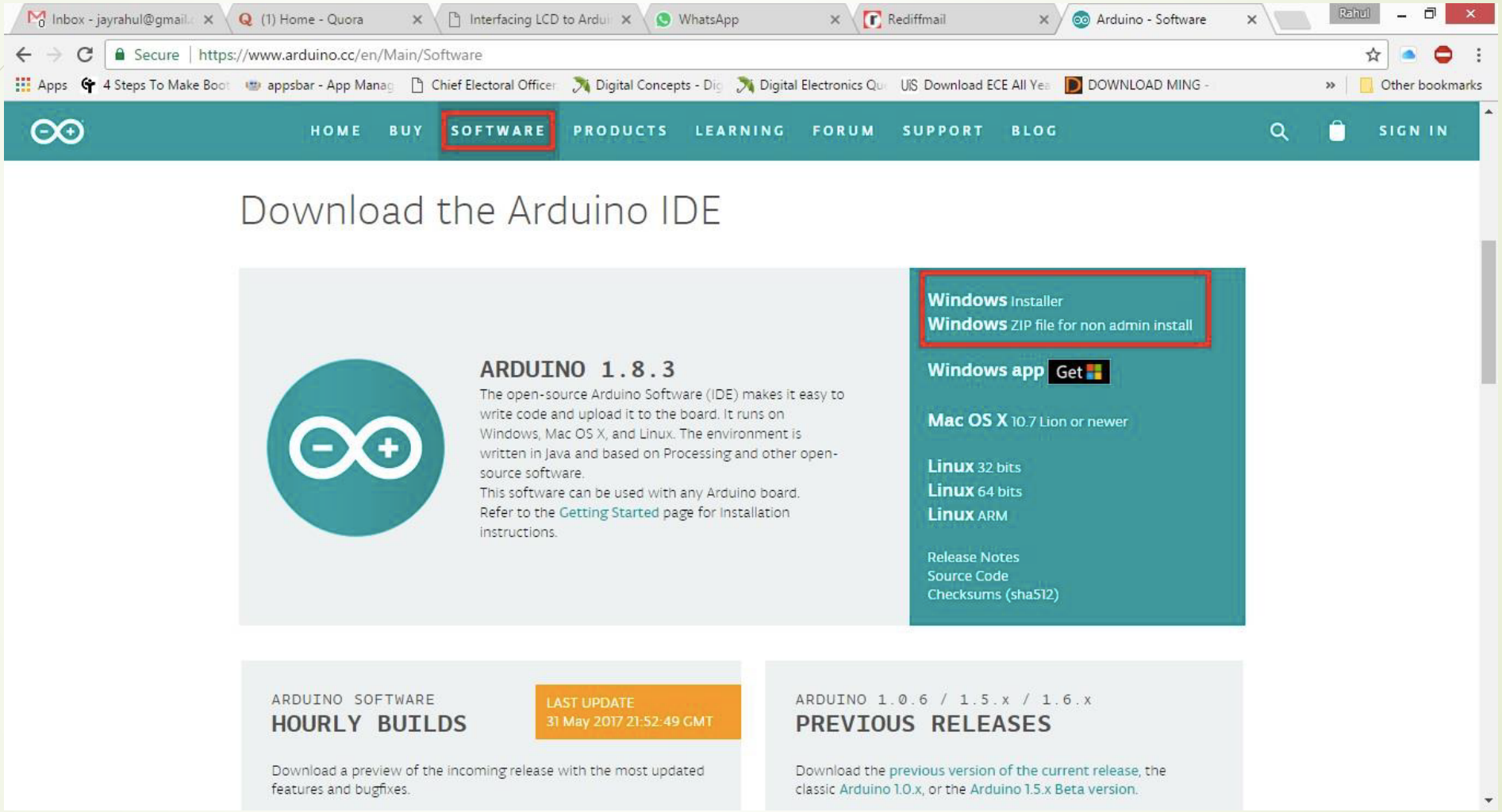
- Based on C++ without 80% of the instructions.
- Programs are called 'sketches'.
- Sketches need two functions:
  - void setup( )
  - void loop( )
- setup( ) runs first and once.
- loop( ) runs over and over, until power is lost or a new sketch is loaded.

Getting Started

Check out: http://arduino.cc/en/Guide/HomePage

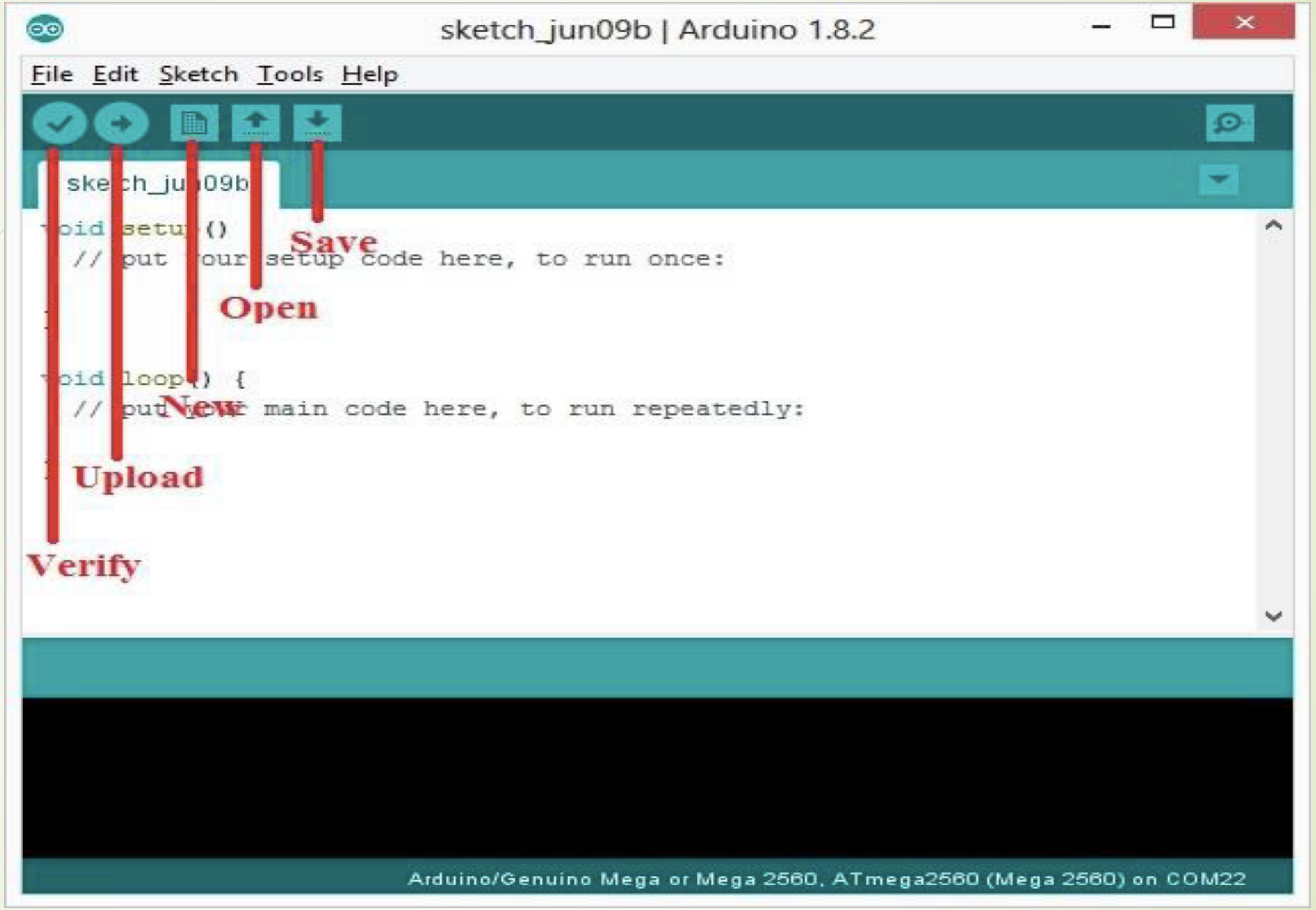
1. Download & install the Arduino environment (IDE)
2. Connect the board to your computer via the USB cable
3. If needed, install the drivers
4. Launch the Arduino IDE
5. Select your board
6. Select your serial port
7. Open the blink example
8. Upload the program

Download the Arduino IDE

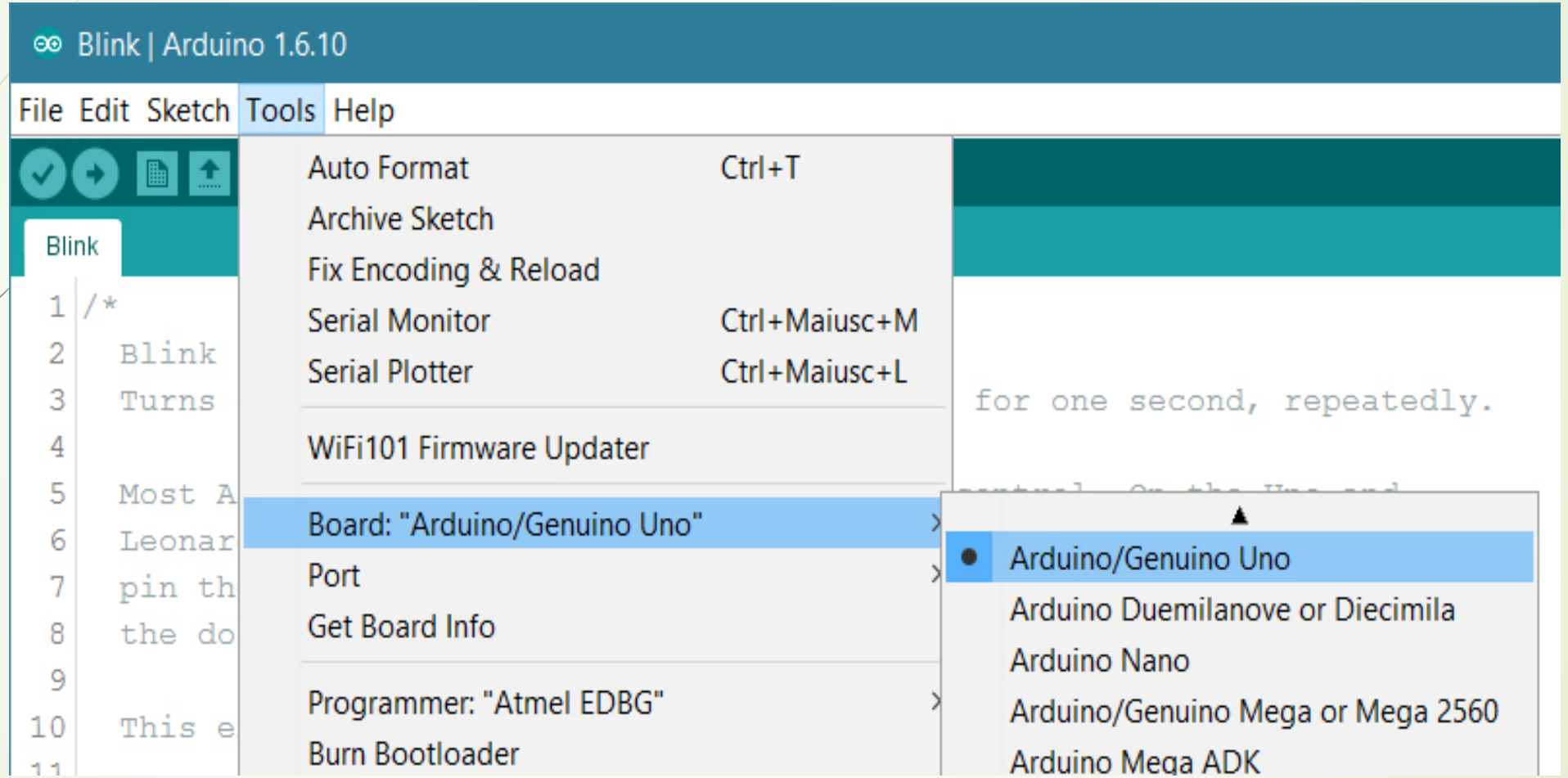


Arduino IDE

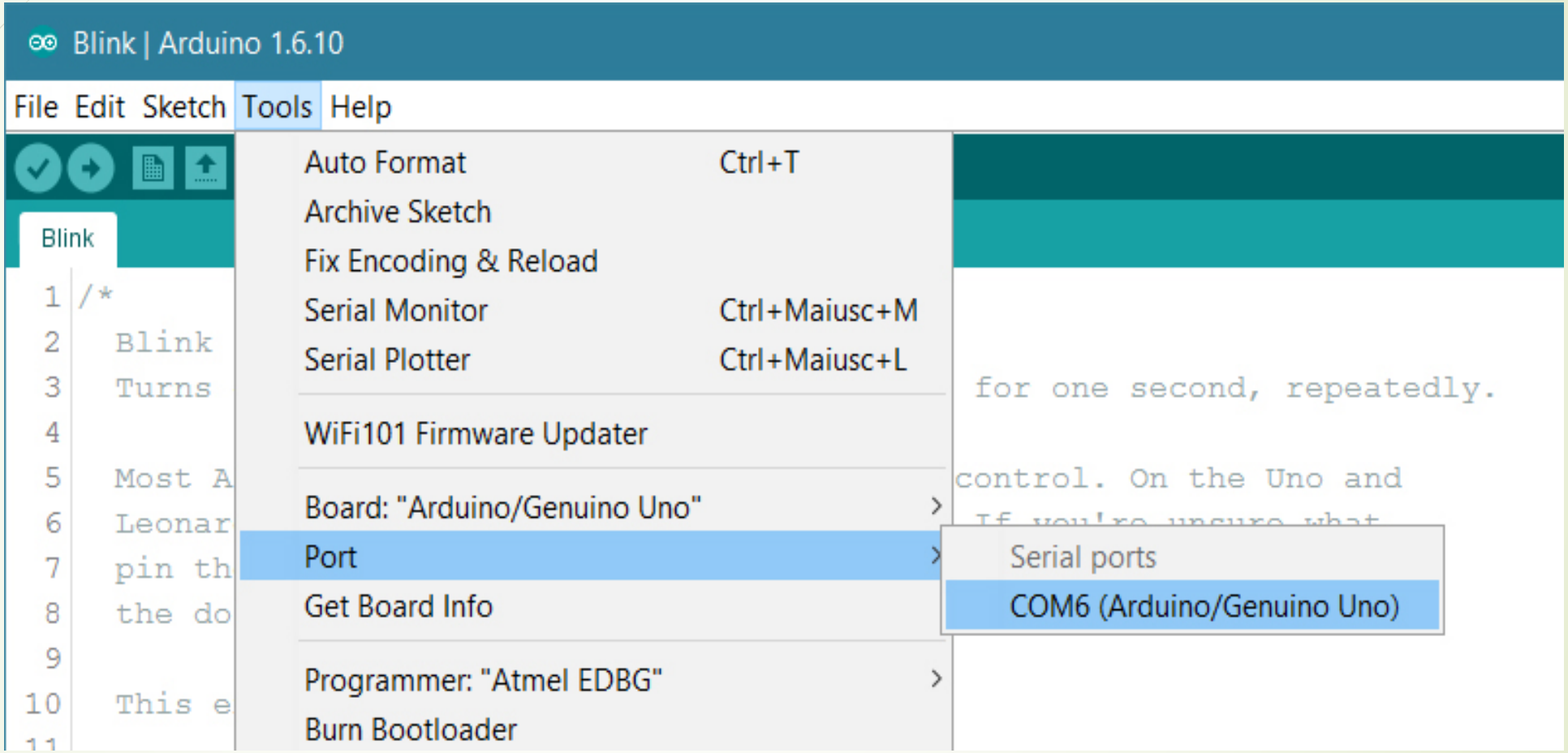

Sketch Name =>
sketch_jun09b | Arduino 1.8.2
File Edit Sketch Tools Help
<= Tool Bar
Serial Monitor =>
sketch_jun09b
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
Editor (Actual Code)
Console
Board Connected=>
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM22


sketch_jun09b | Arduino 1.8.2
File Edit Sketch Tools Help
sketch_jun09b
void setup()
// put your setup code here, to run once:
void loop() {
// put your main code here, to run repeatedly:
Save
Open
New
Upload
Verify
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM22

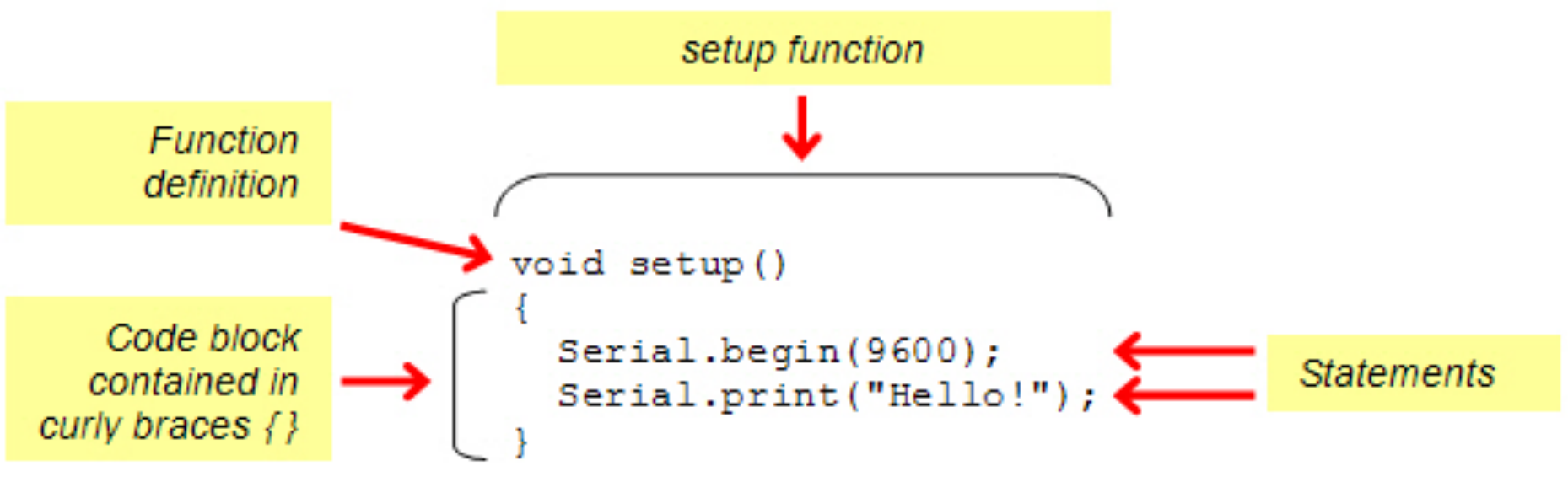
Select Board Type



Select Serial COM Port



Setup Function

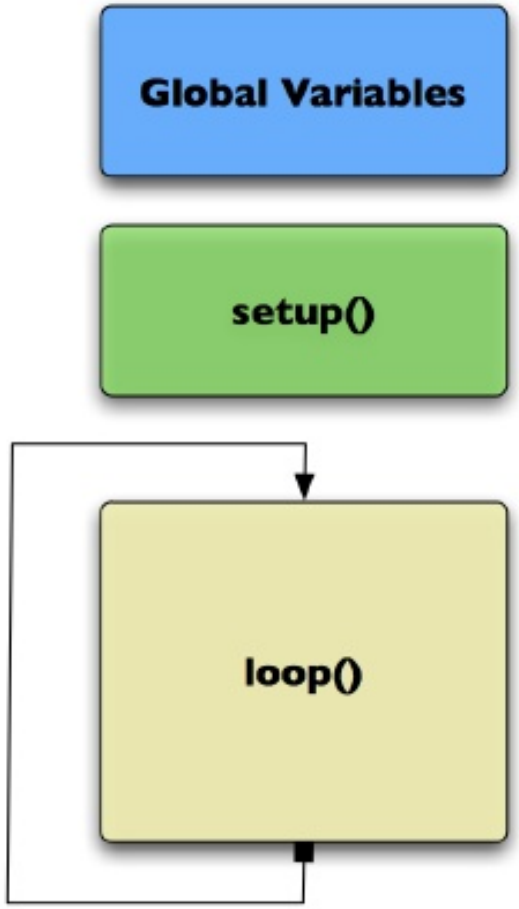


Setup Function will run only once at the start of program.

Loop Function

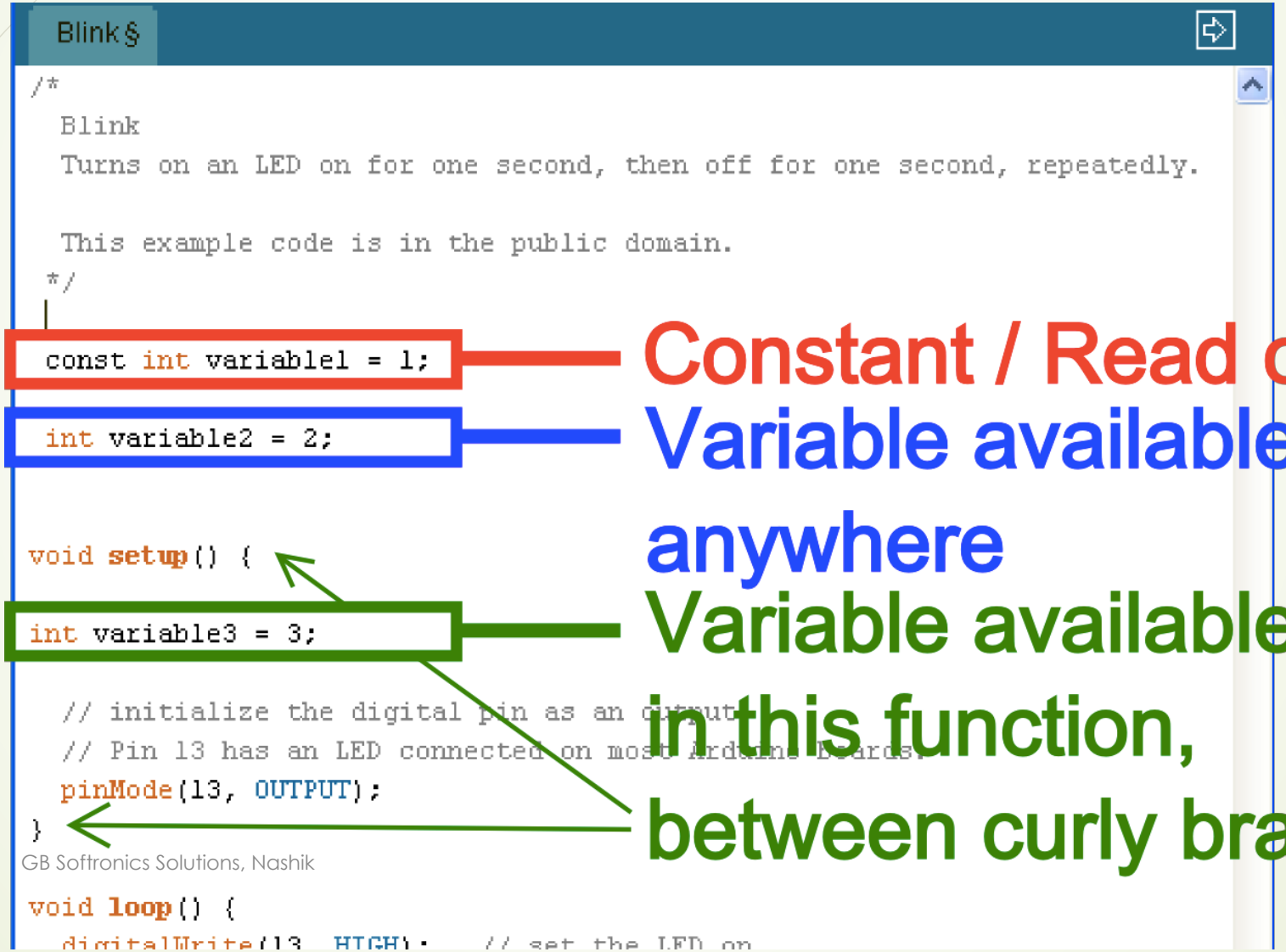
```
void loop()
{
        // Put your code here to run it continuously
}
```

An Arduino "Sketch"



- Declare variables at top
- Initialize
  - setup() – run once at beginning, set pins
- Running
  - loop() – run repeatedly, after setup()

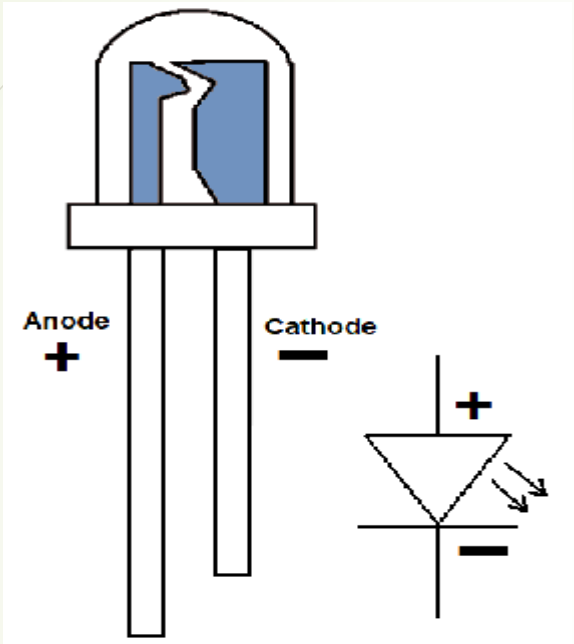
Variable Declaration



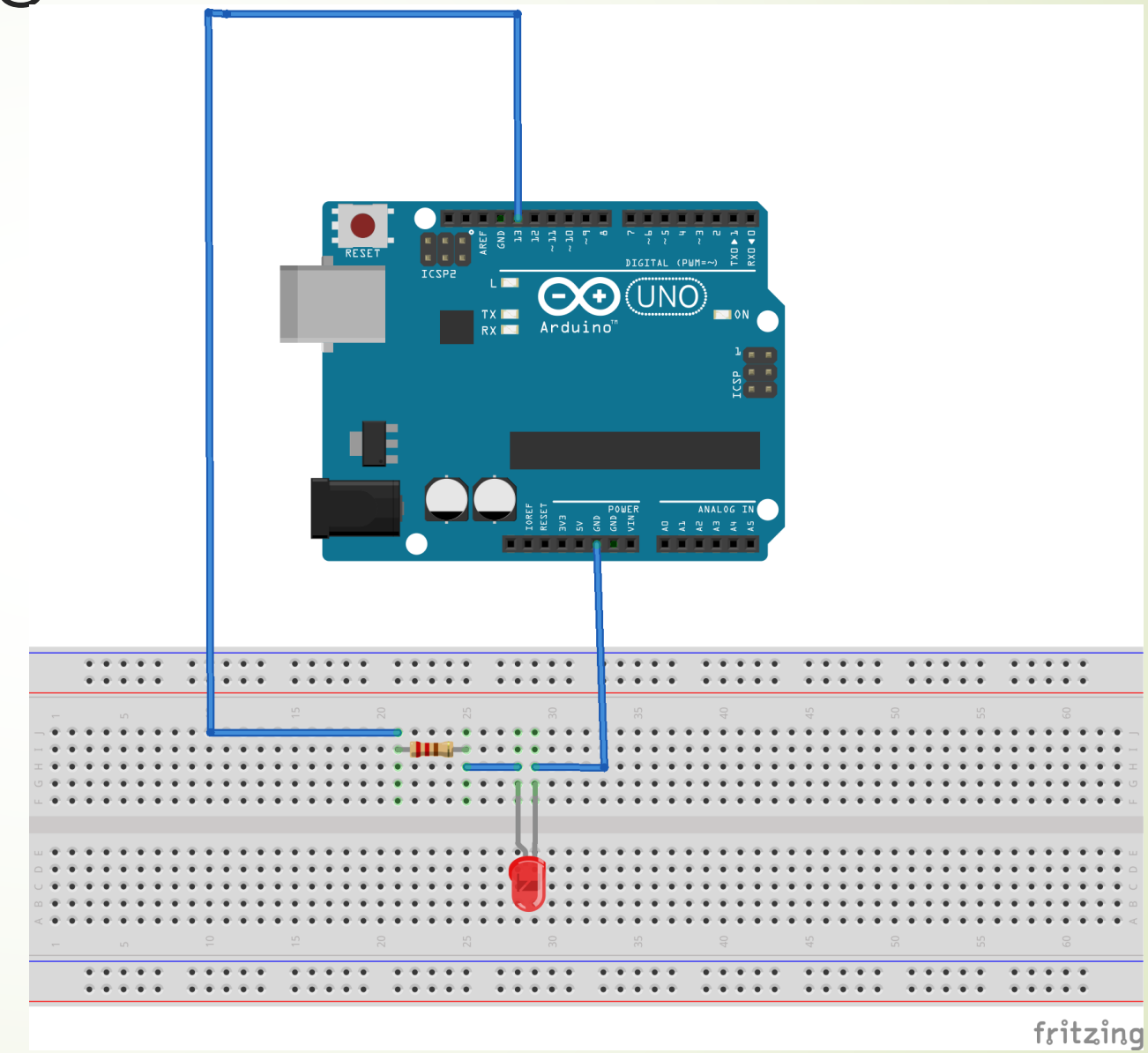
Some Important Functions

- Serial.print(value);
  - Prints the value to the Serial Monitor on your computer
- pinMode(pin, mode);
  - Configures a digital pin to read (input) or write (output) a digital value
- digitalRead(pin);
  - Reads a digital value (HIGH or LOW) on a pin set for input
- digitalWrite(pin, value);
  - Writes the digital value (HIGH or LOW) to a pin set for output

LED Interfacing



100 Ohm Resistor



Program

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

- Outputs/Inputs are declare in setup, this is done by using the pinMode function
- This particular example declares digital pin # 13 as an output, remember to use CAPS for OUTPUT

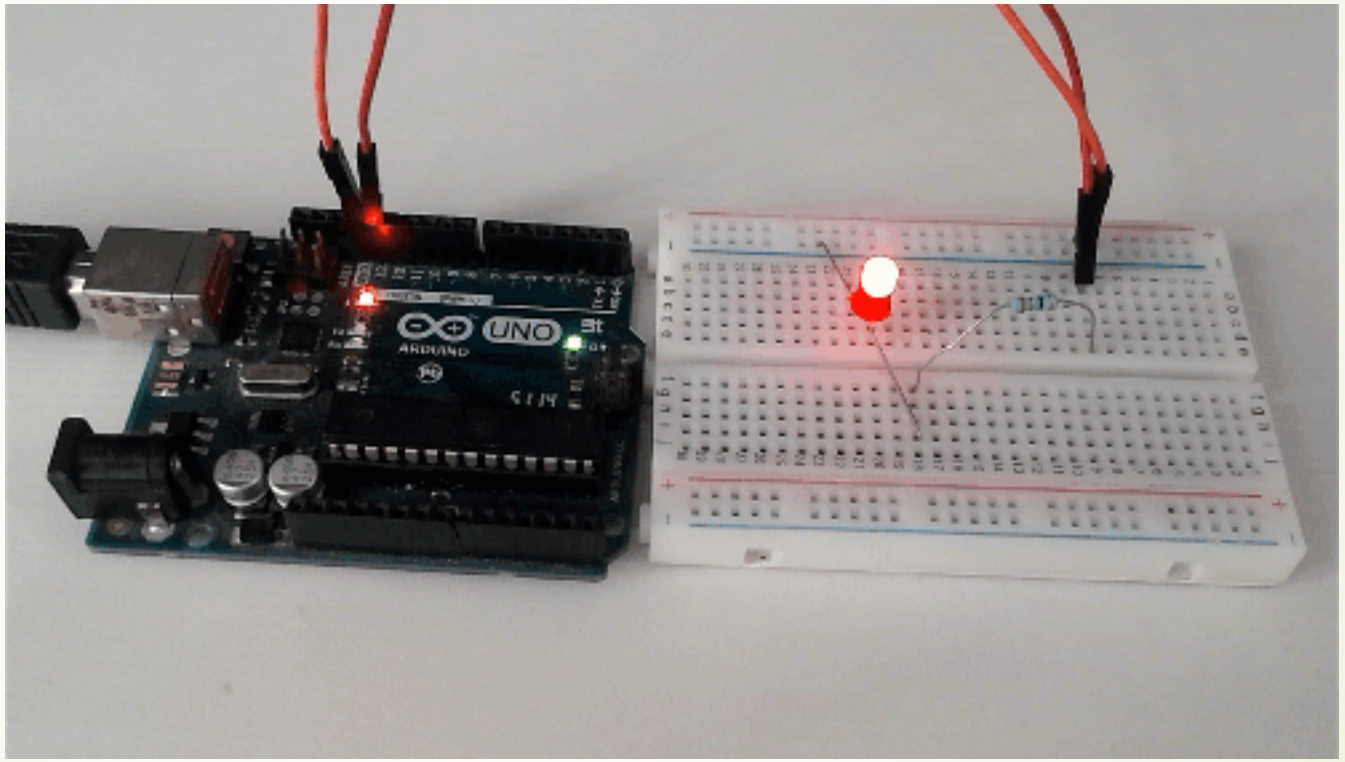
LED Blinking

```
void setup( ) {
        pinMode(13, OUTPUT);
}
void loop( ) {
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13, LOW);
        delay(1000);
}
```

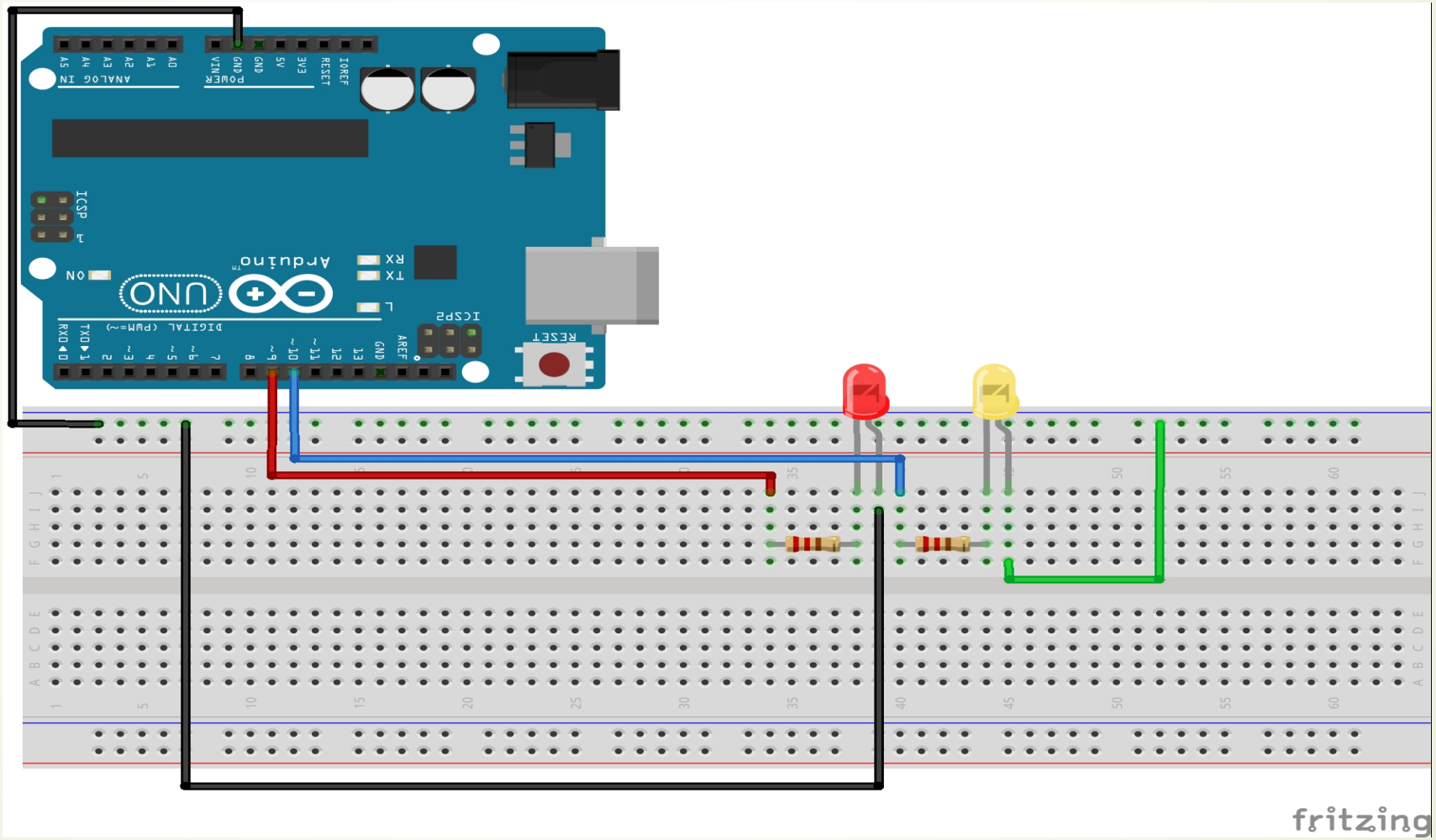
Compiling and Uploading the Code

1. Write the code
2. Compile the code
3. Check Arduino Port Connection
4. Upload the Code
5. The Arduino and Connected Circuits start to show behavior based on the uploaded code

Output



Two LED's Interfacing



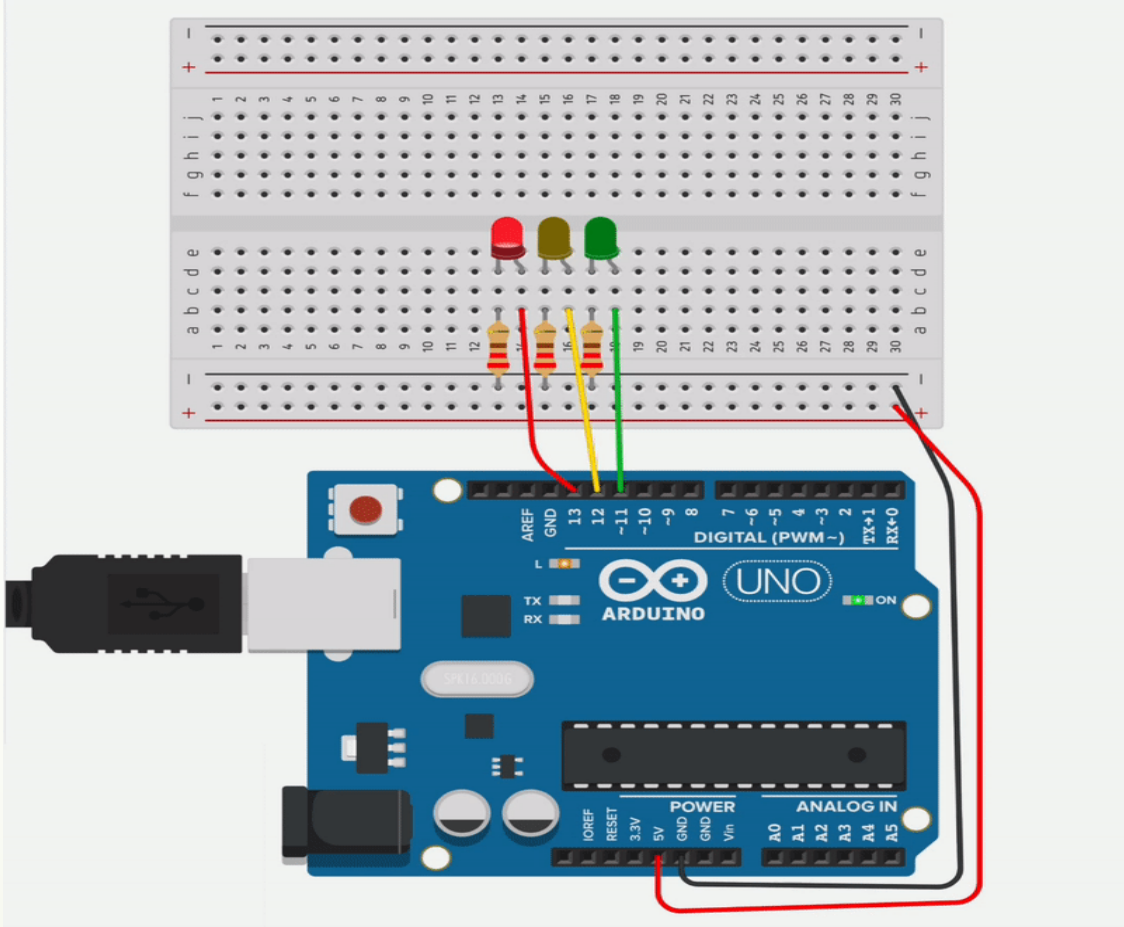
Program

```
void setup( ) {
        pinMode(12, OUTPUT);
        pinMode(13, OUTPUT);
}
void loop( ) {
        digitalWrite(12, HIGH);
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(12, LOW);
        digitalWrite(13, LOW);
        delay(1000);
}
```

Assignments

- Interface Two LED's with Arduino to pin 2 and 3 respectively. Write a program to toggle the LED's with delay of 0.5 Sec.
- Interface Four LED's with Arduino to pin 2,3,4 and 5 respectively. Write a program to blink all LED's with a delay of 0.2 Sec.
- Interface Three LED's with Arduino to pin 2,3 and 4 respectively. Write a program to toggle all LED's with a delay of 0.3 Sec.
- Interface Three LED's with Arduino to pin 2,3 and 4 respectively. Write a program to get running LED's effect with a delay of 0.5 Sec.
- Interface Four LED's with Arduino to pin 2,3,4 and 5 respectively. Write a program to get running LED's effect with a delay of 1 Sec.

Three LED's Running LED Output



Starting Serial Communication

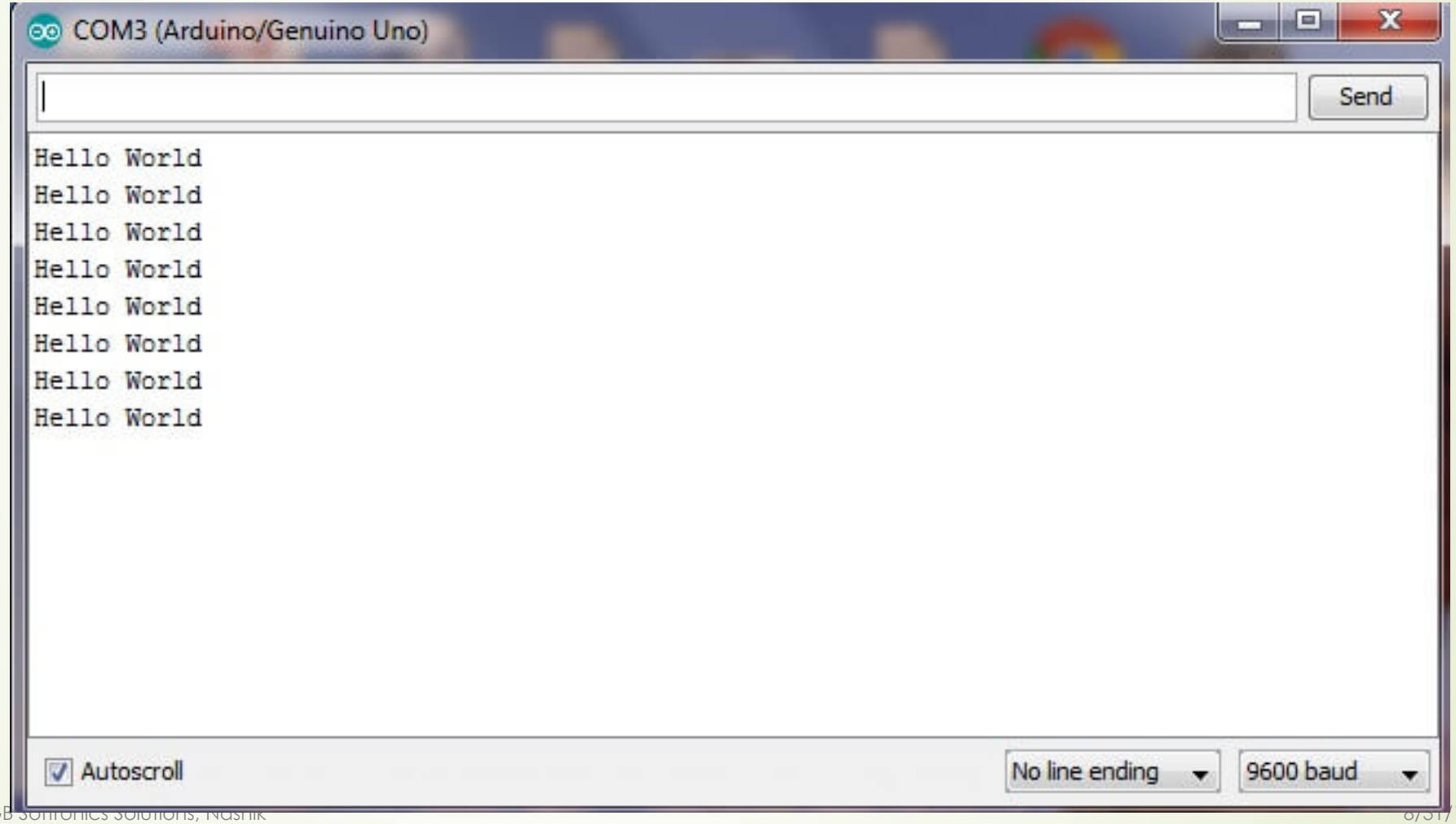
```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  Serial.begin(9600);
}
```

- Serial communication also begins in setup function
- This particular example declares Serial communication at a baudrate of 9600.
- More on Serial later...

Program to print Hello World on Serial Monitor contineously:

```
void setup() {
        Serial.begin(9600);
}
void loop() {
        Serial.println("Hello World!");
        delay(1000);
}
```

Output


COM3 (Arduino/Genuino Uno)
Send
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Autoscroll
No line ending
9600 baud

Control LED through Serial Monitor

```
void setup() {
        Serial.begin(9600);
        pinMode(2,OUTPUT);
}
void loop() {
        if(Serial.available()>0) {
            char rxchar = Serial.read();
            switch( rxchar ) {
                case '1' :   digitalWrite(2,HIGH);
                            break;
                case '0' :  digitalWrite(2,LOW);
                            break;
            }
        }
}
```

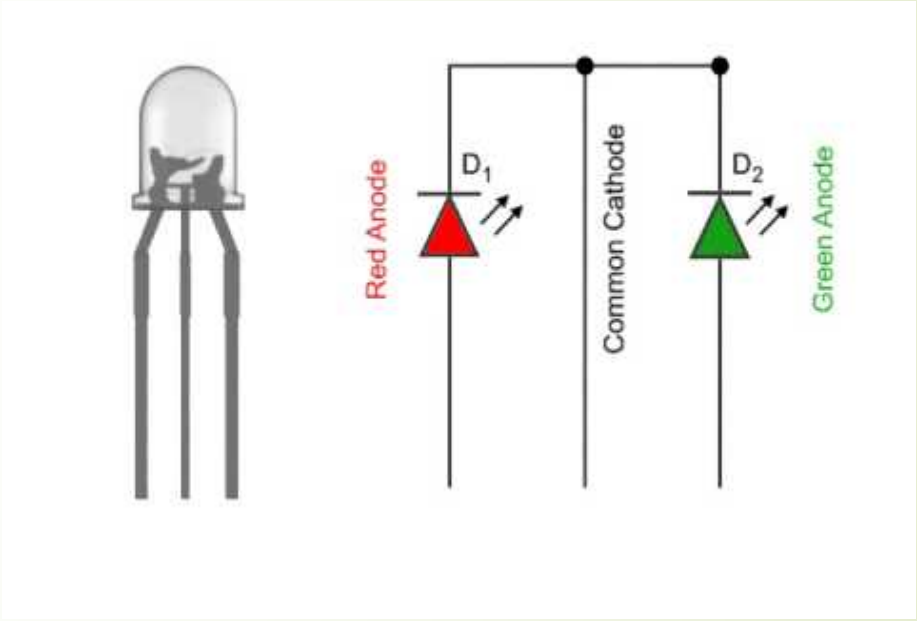
Assignments

- Interface Two LED's to arduino pin 2 and 3. Write a program to control these LED's through Serial.

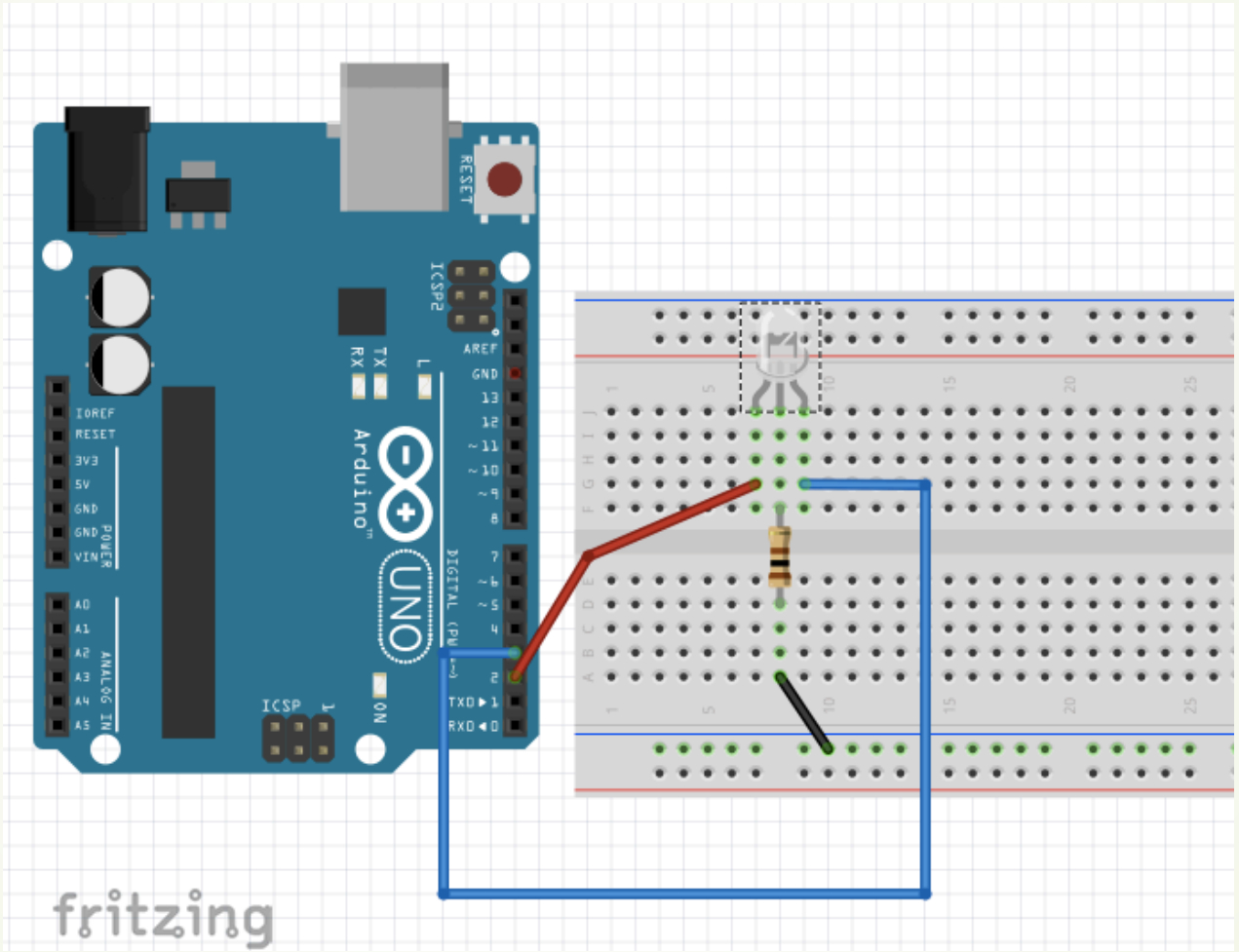
- Interface Four LED's to arduino pin 2,3,4 and 5. Write a program to control these LED's through Serial.

Bi-color LED


CHANZON
Bi-color
Red & Blue
Common Cathode


Red Anode
D1
Common Cathode
D2
Green Anode

Interfacing Bi-Color LED



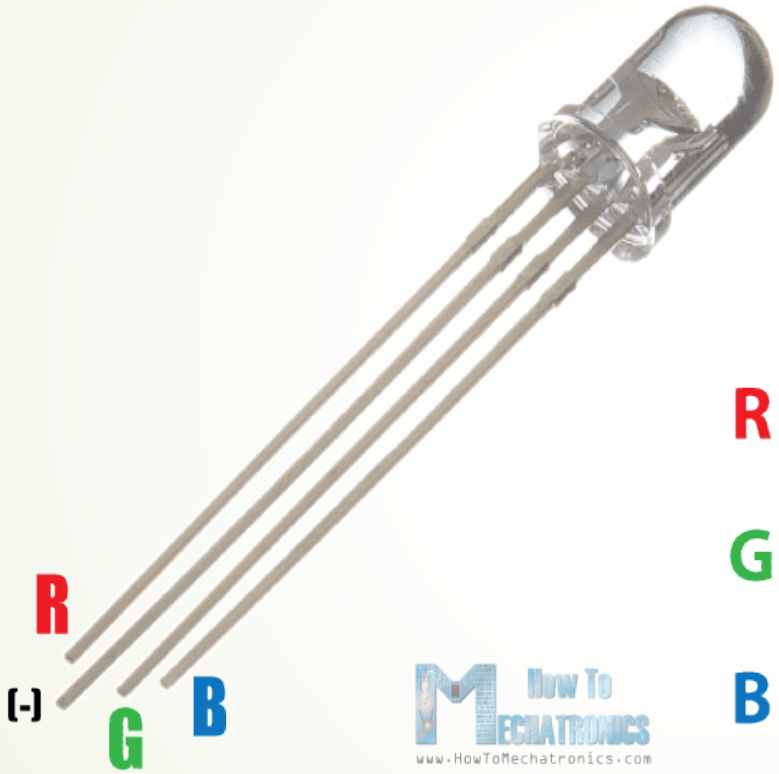
Program

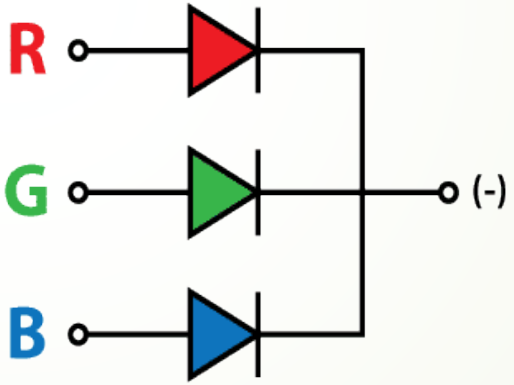
```
void setup( ) {
        pinMode(2, OUTPUT);
        pinMode(3, OUTPUT);
}
void loop( ) {
        digitalWrite(2, HIGH);
        digitalWrite(3, LOW);
        delay(1000);
        digitalWrite(2, LOW);
        digitalWrite(3, HIGH);
        delay(1000);
}
```

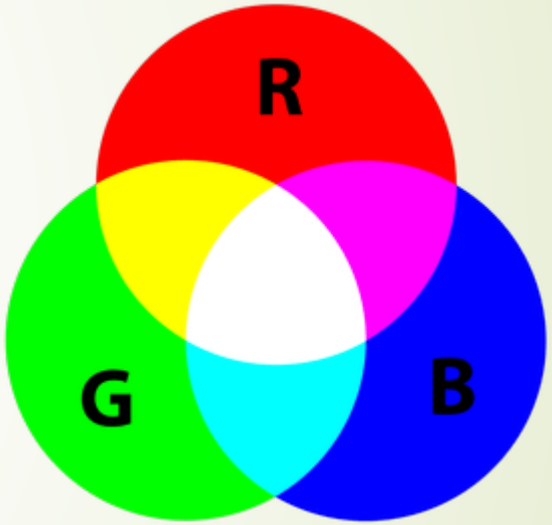
Assignment

- Interface Bi-Color LED with Arduino. Write a program to Control display LED Color through Serial Monitor.
- eg If user send letter 'r' through serial, RED LED should displayed and if user send letter 'b', then blue LED should displayed.

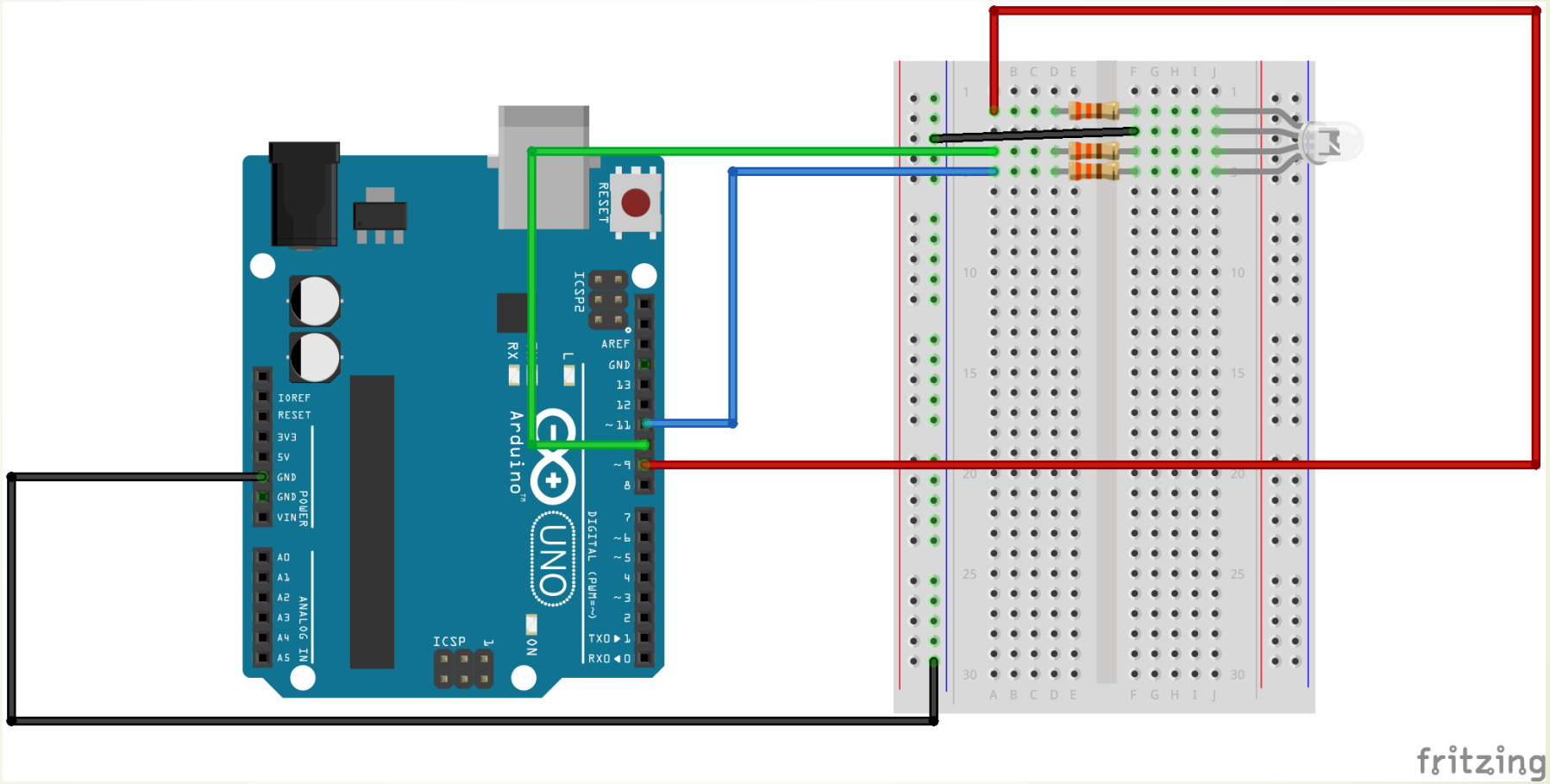
RGB LED







RGB LED Interfacing



Program

```
int redPin = 9;
int greenPin = 10;
int bluePin = 11;
void setup() {
        pinMode(redPin,OUTPUT);
        pinMode(bluePin,OUTPUT);
        pinMode(greenPin, OUTPUT);
}
void loop() {
        digitalWrite(redPin,HIGH);
        digitalWrite(greenPin,LOW);
        digitalWrite(bluePin,LOW);
        delay(1000);
        digitalWrite(redPin,LOW);
        digitalWrite(greenPin,HIGH);
        digitalWrite(bluePin,LOW);
        delay(1000);

        digitalWrite(redPin,LOW);
        digitalWrite(greenPin,LOW);
        digitalWrite(bluePin,HIGH);
        delay(1000);

}
```

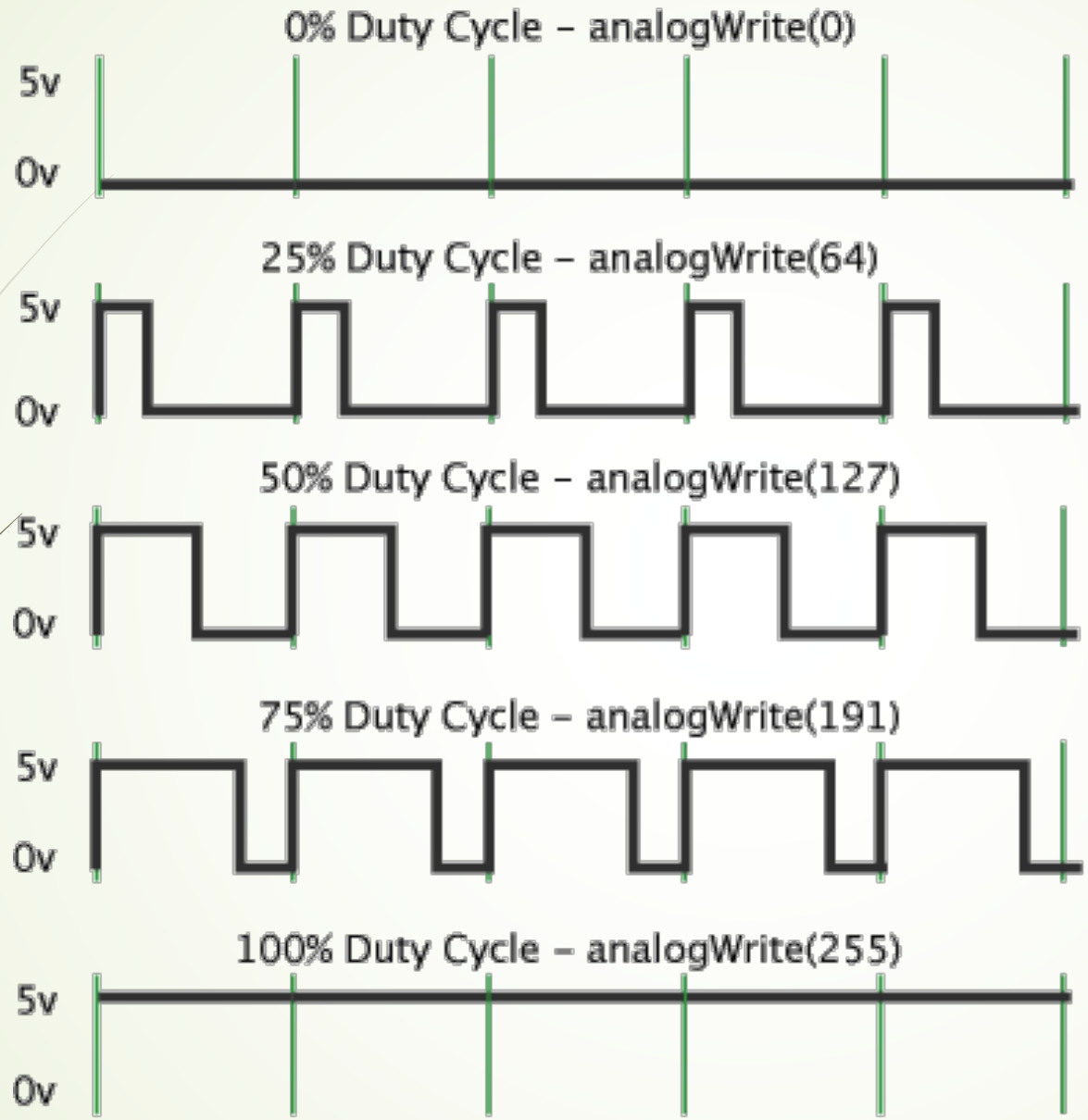
Assignments

- Interface RGB LED with arduino to pins 2,3 and 4. Write a program to display color Yellow, Pink, Cyan alternate with an delay of 1 Sec.
- Interface RGB LED with arduino to pins 2,3 and 4. Write a program to Control RGB LED Color through Serial by sending different commands to display various colors.

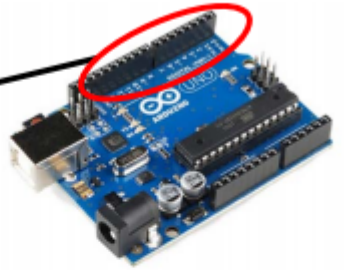
Brightness Control of LED

- LED Brightness can control through PWM Pins of Arduino UNO.
- Arduino UNO have 6 PWM Pins. **Pin 3,5,6,9,10 and 11**
- Writes an analog value (PWM wave) to a pin.
- Can be used to light a LED at varying brightnesses or drive a motor at various speeds.
- After a call to analogWrite(), the pin will generate a steady square wave of the specified duty cycle until the next call to analogWrite() (or a call to digitalRead() or digitalWrite()) on the same pin.

Pulse Width Modulation



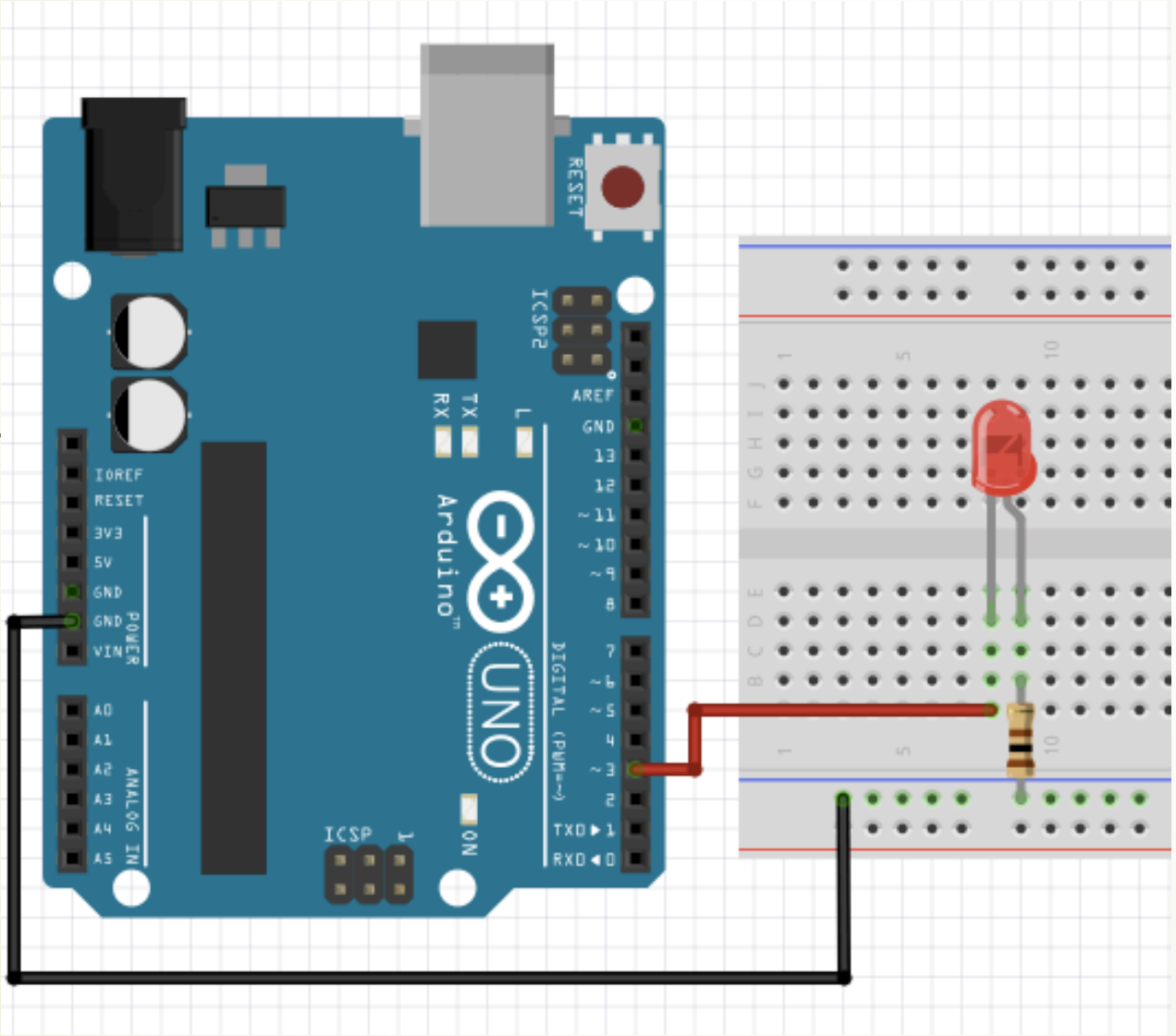
Analog Output (PWM)



Digital-to-Analog Conversion (DAC)

Pins:	PWM ~ (3, 5, 6, 9, 10 and 11)
Voltage range:	0-5
Value range:	0-255
Sketch function:	analogWrite

Circuit Diagram

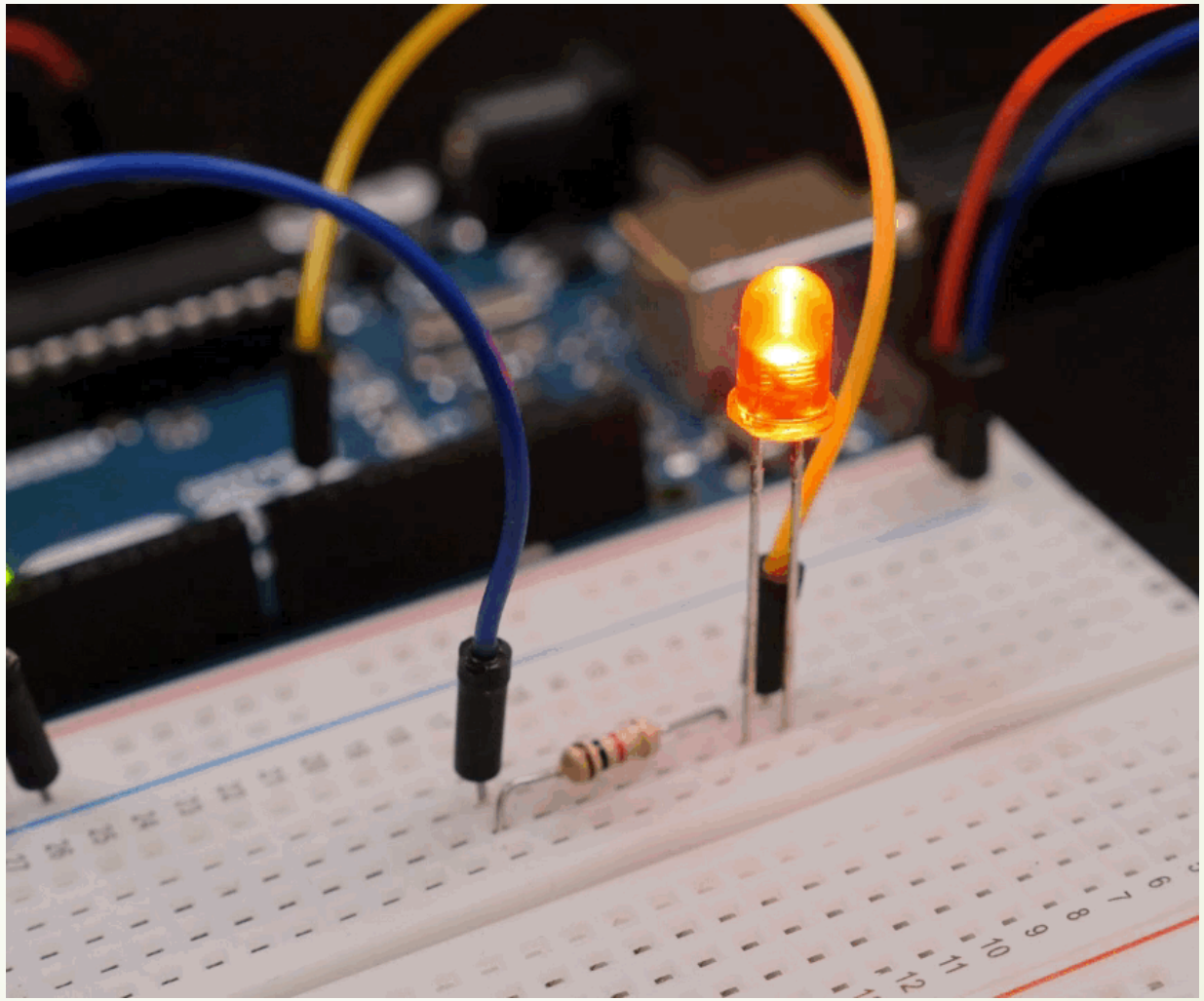


1) Connect Anode of LED to Pin 3(PWM Pin)
2) Cathode of LED is connected to Ground through 100ohm resistor

Program

```
int led = 3;
void setup() {
        pinMode( led, OUTPUT );
}
void loop() {
                for( int val=0 ; val <=255 ; val = val+5 ){
                        analogWrite(led, val );
                        delay(50);
                }
                for( int val=255 ; val >=0 ; val = val-5 ){
                        analogWrite(led, val );
                        delay(50);
                }
}
```

Output

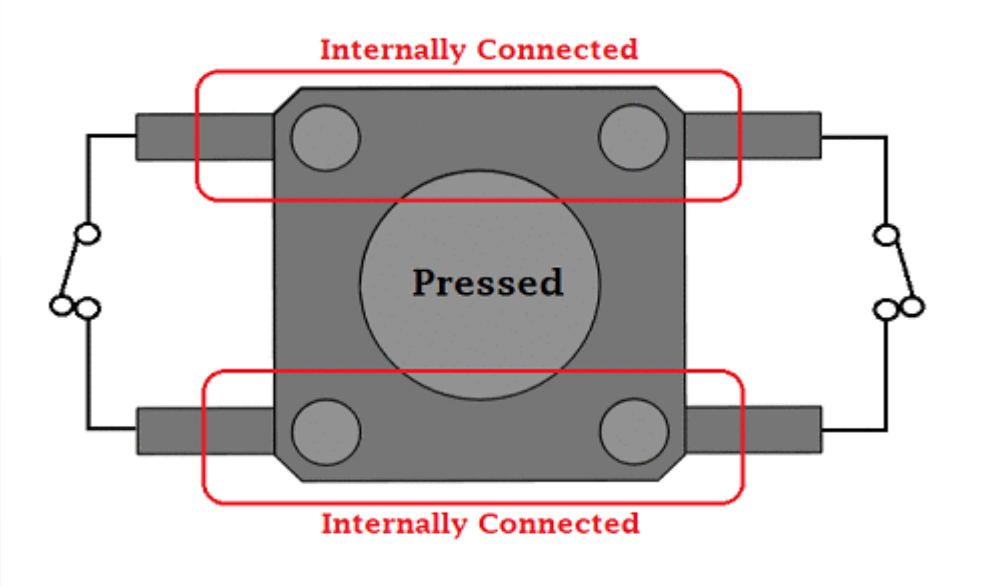


Brightness Control through Serial

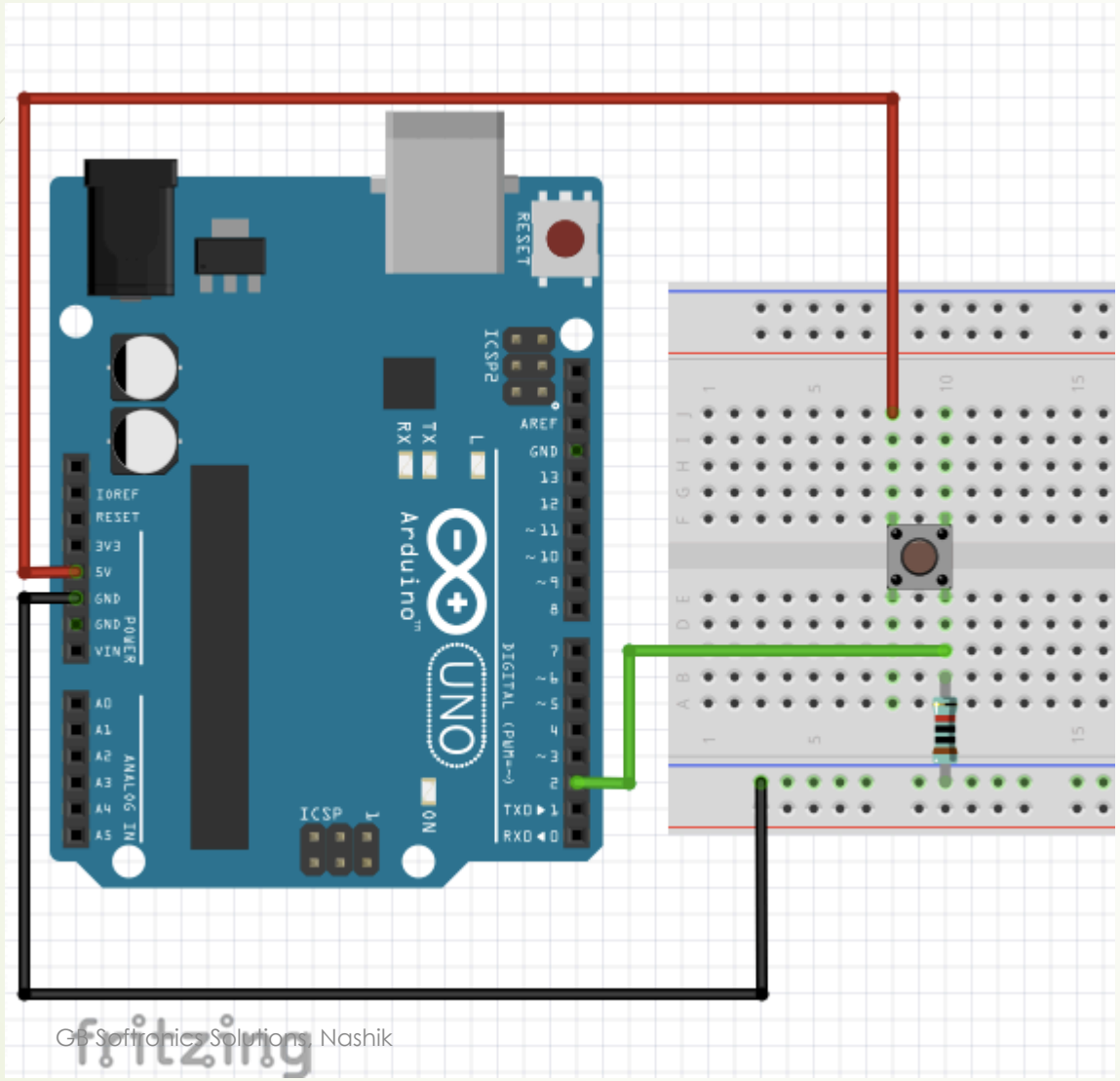
```
int led = 3;
void setup() {
        pinMode( led, OUTPUT );
        Serial.begin(9600);
}
void loop() {
            if(Serial.available() > 0){
                int val = Serial.parseInt();
                if(val<=255 && val>=0){
                    analogWrite(led,val);
                    delay(50);
                }
                else
                    Serial.println("Invalid Input!  Valid Input range is 0-255");
            }
}
```

Push Button Interfacing

- Push Button is used to get input from user.
- It is digital Sensor.
- It is push to on type switch.

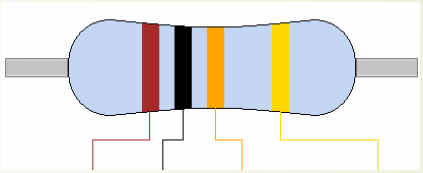


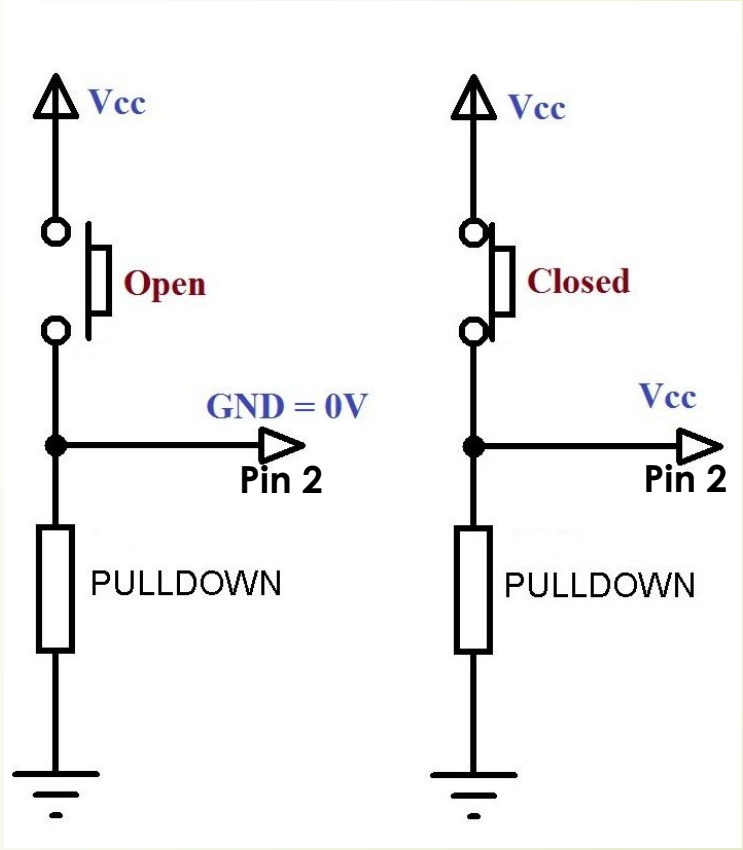
Circuit Diagram



GB Softronics Solutions, Nashik

1) Use Cross Terminals of Switch for Connections.
2) Use 10 Kohm Resistor





Program

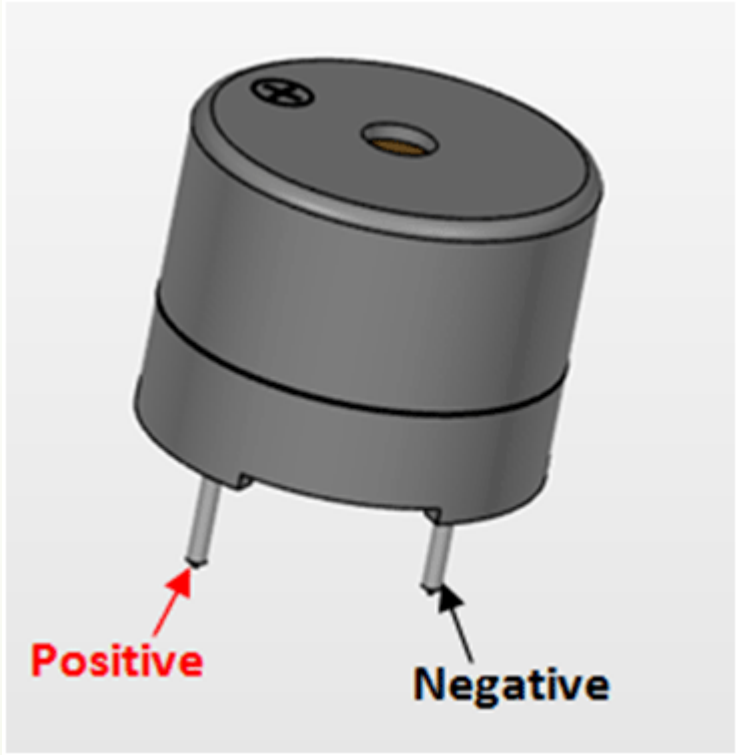
```
int sw = 2;
void setup() {
        pinMode(sw,INPUT);
        Serial.begin(9600);
}
void loop() {
        int swstatus = digitalRead( sw );
        if(swstatus == 1) {
            Serial.println("Switch Pressed");
        }
        delay(50);
}
```

Assignment

- Interface LED and Switch with arduino. If User press switch, then LED turn ON and If user Release switch, then LED turns off. Write the program for the same.
- Interface Two LED's and Two Switch's with arduino. Control the individual LED On-OFF by respective switch.

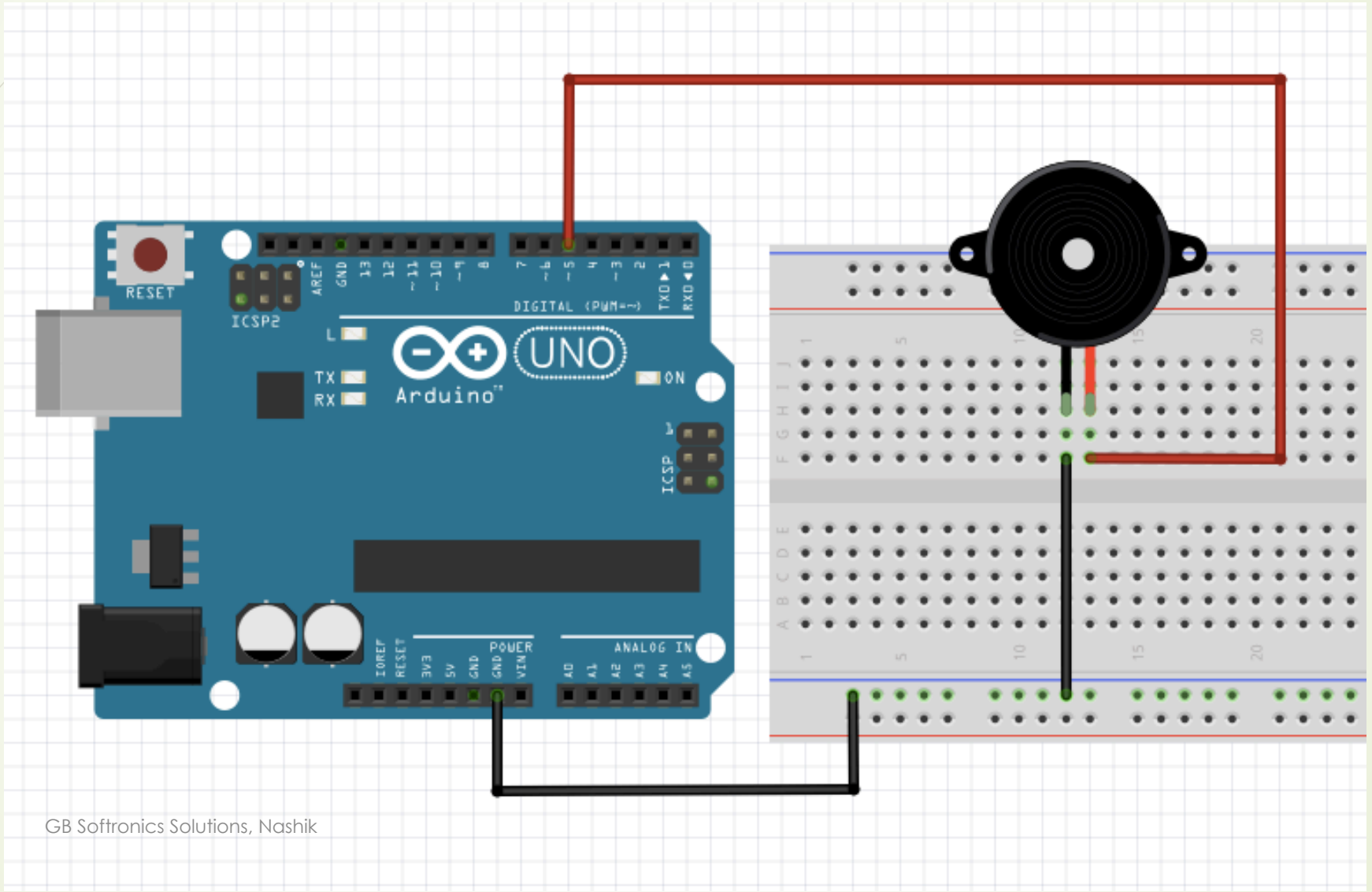
Buzzer Interfacing

- Buzzer is the output Device, use to alert the user. If power Supply provided to buzzer, it turns on and make sounds.



- **Long Terminal is Positive**
- **Short terminal is Negative**

Circuit Diagram



Program

```
int buzzer = 5;
void setup( ) {
        pinMode(buzzer, OUTPUT);
}
void loop( ) {
        digitalWrite(buzzer, HIGH);
        delay(1000);
        digitalWrite(buzzer, LOW);
        delay(1000);
}
```

Assignment

- Interface Switch and Buzzer with arduino. Write a program to turn on buzzer, if user press switch, otherwise buzzer remains off.
- Interface LED, Switch and buzzer with arduino. Write a program to turn on buzzer and LED, if user press switch, otherwise buzzer and led remains off.

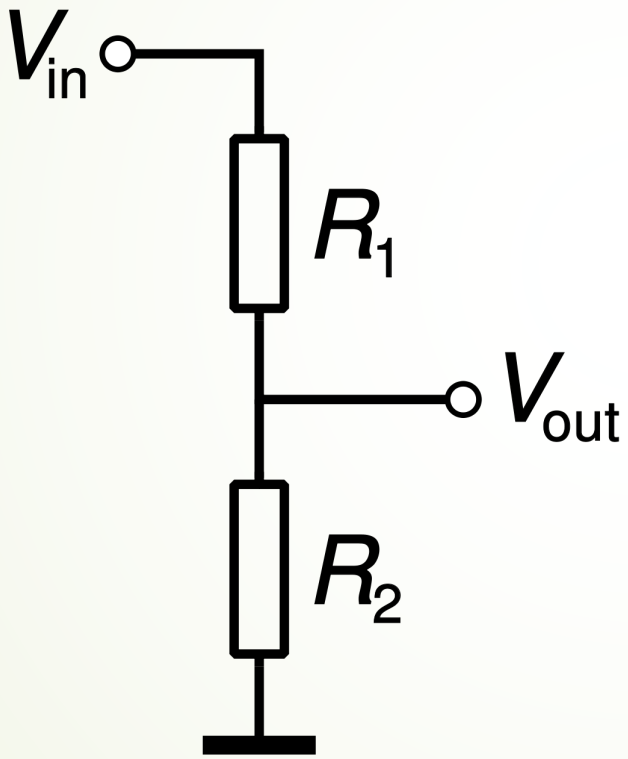
Analog Sensor Interfacing

- Analog Sensor will produce an analog output in the range of 0-5V.
- But Microcontroller only understand digital signals(Binary)
- So we need to convert analog signal to digital.
- Arduino UNO having 6 ADC(Analog to digital Converters) which is used to convert analog signal to digital signal.
- Arduino UNO ADC having the resolution of 10bits.
- 10 bits resolutions means, minimum digital output from ADC is 0000000000 ie decimal 0 and maximum digital output from ADC is 1111111111 ie decimal 1023. So we get digital output in the range from **0 to 1023.**
- Analog inputs are A0,A1,A2,A3,A4,A5
- Remember, Input voltage rage to ADC is **0V to 5V** only

Potentiometer Interfacing

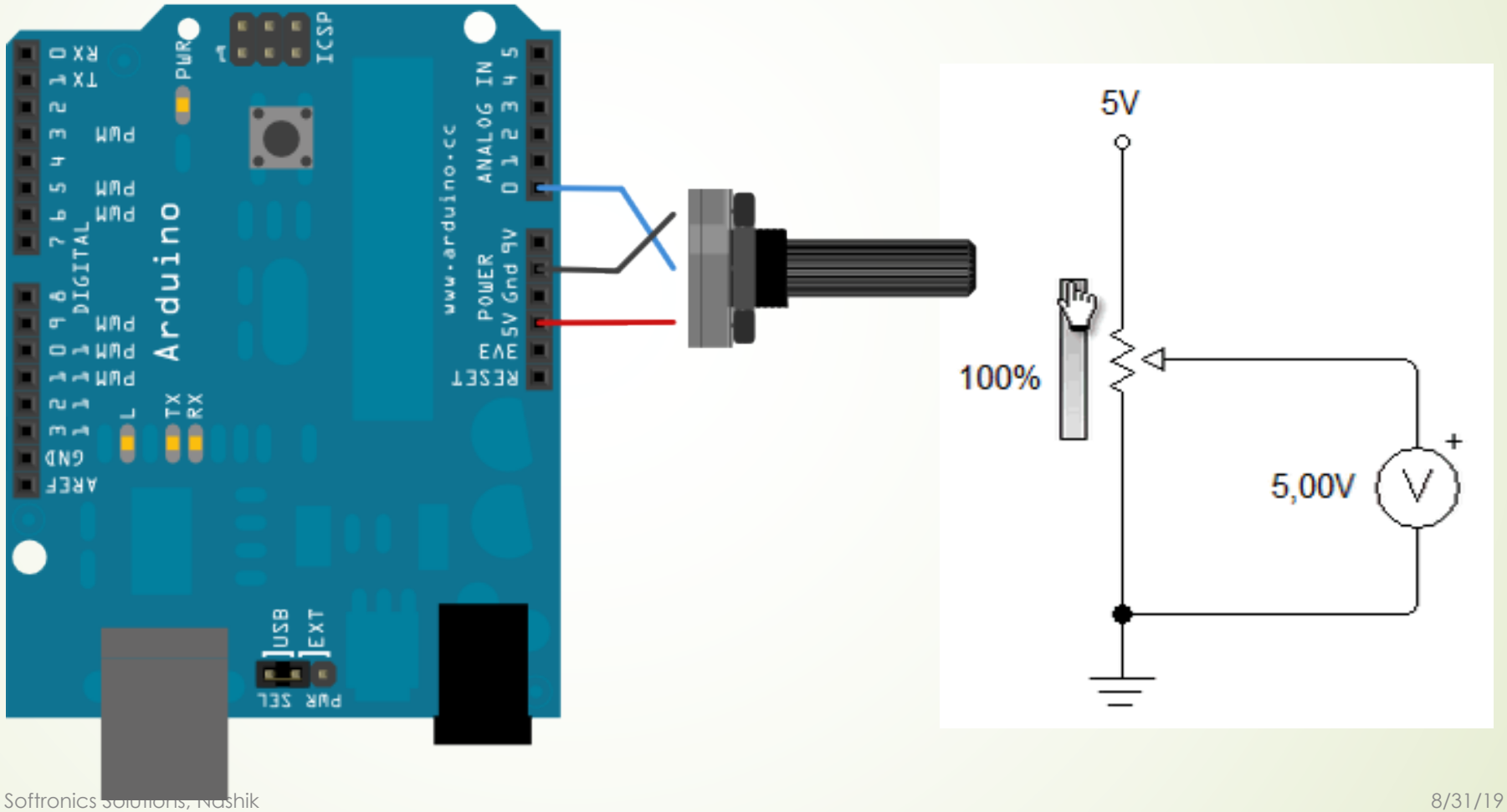
- A potentiometer is a simple knob that provides a variable resistance, which we can read into the Arduino board as an analog value. In this example, that value controls the rate at which an LED blinks.
- We connect three wires to the Arduino board. The first goes to ground from one of the outer pins of the potentiometer. The second goes from 5 volts to the other outer pin of the potentiometer. The third goes from analog input 2 to the middle pin of the potentiometer.
- By turning the shaft of the potentiometer, we change the amount of resistance on either side of the wiper which is connected to the center pin of the potentiometer. This changes the relative "closeness" of that pin to 5 volts and ground, giving us a different analog input. When the shaft is turned all the way in one direction, there are 0 volts going to the pin, and we read 0. When the shaft is turned all the way in the other direction, there are 5 volts going to the pin and we read 1023. In between, analogRead() returns a number between 0 and 1023 that is proportional to the amount of voltage being applied to the pin.

Voltage Divider Rule



$$V_{out} = \frac{V_{in} \times R_2}{(R_1 + R_2)}$$

Circuit Diagram



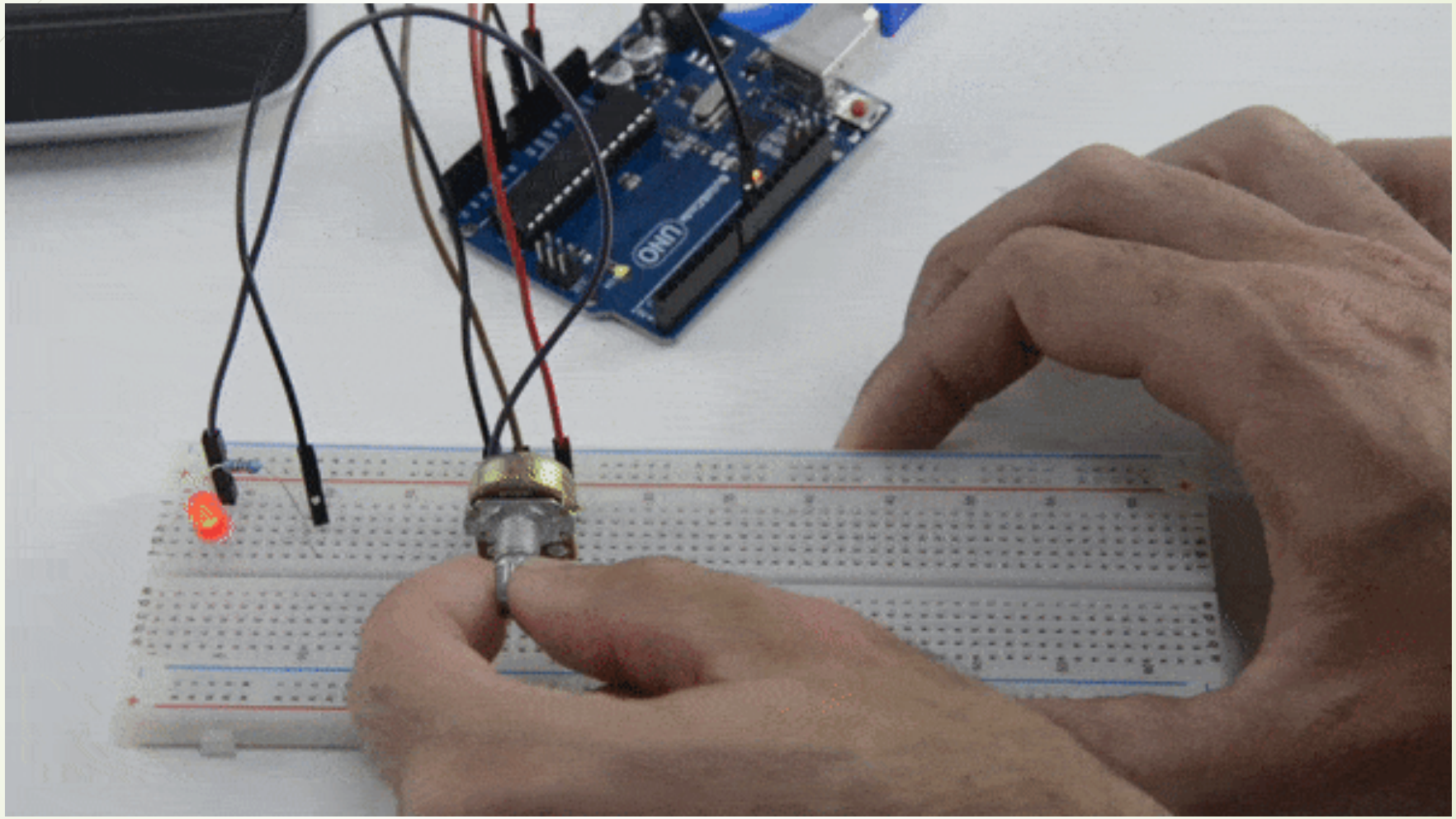
Program (Output on Serial Monitor)

```
void setup() {
    Serial.begin(9600);
}
void loop() {
        int potval = analogRead(A0);      // POT connected to A0
        Serial.print("POT Value = ");
        Serial.println(potval);
        delay(1000);
}
```

Changing LED brightness with POT

```
int led = 3;
void setup() {
      Serial.begin(9600);
      pinMode(led,OUTPUT);
}
void loop() {
      int potval = analogRead(A0);
      Serial.print("POT Value = ");
      Serial.println(potval);
      int val = map(potval,0,1023,0,255);
      analogWrite(led,val);
      delay(100);
}
```

Output



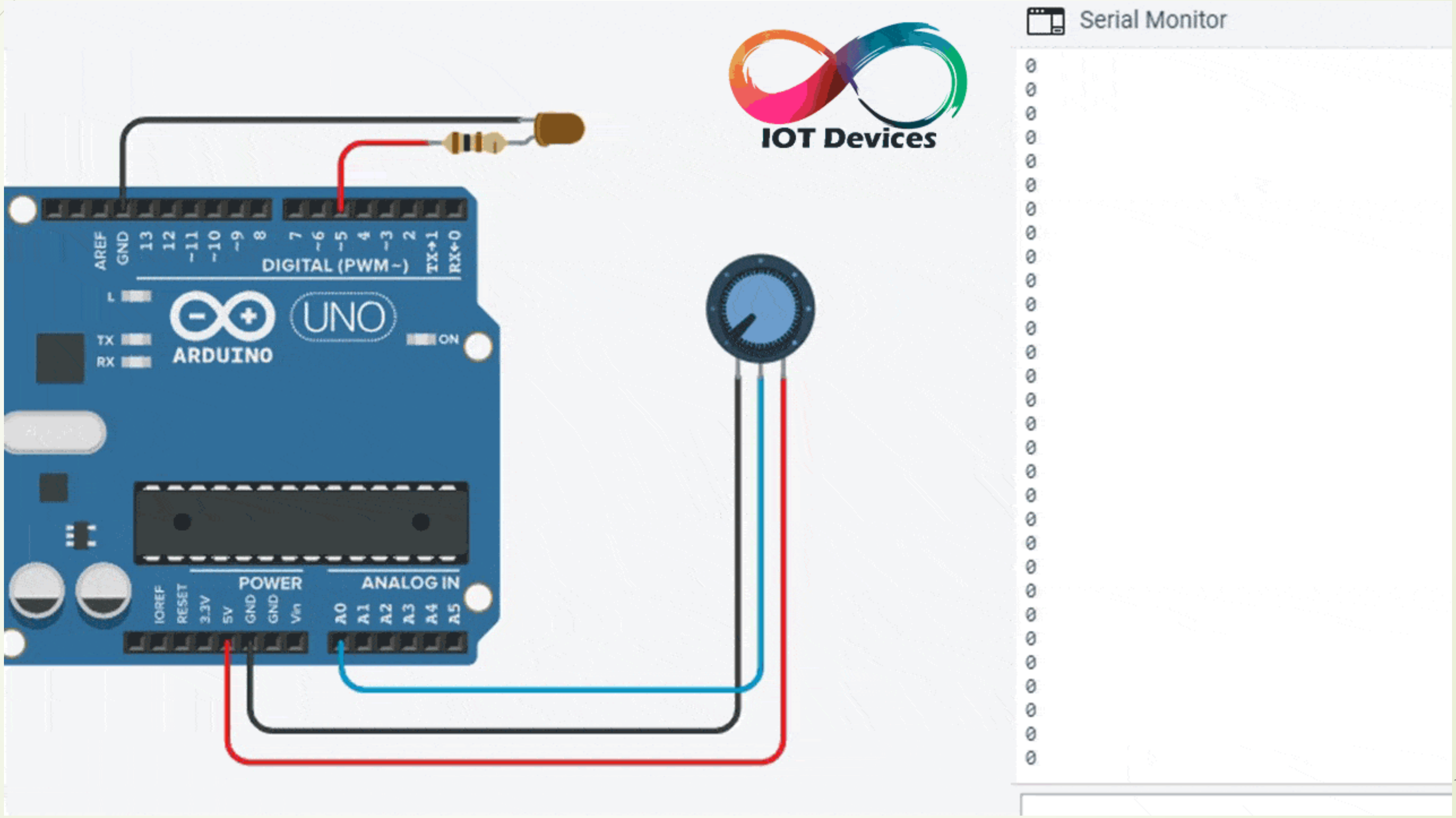
LED Control on Analog Sensor value

```
int led = 2;
void setup() {
      Serial.begin(9600);
      pinMode(led,OUTPUT);
}
void loop() {
      int potval = analogRead(A0);
      Serial.print("POT Value = ");
      Serial.println(potval);
      if( potval >=500 ) {
            digitalWrite(led, HIGH);
            Serial.println("Alert! Beyond Threshold value");
      }
      else {
            digitalWrite(led, LOW);
            Serial.println("SAFE! Below Threshold value");
      }
      delay(100);
}
```

LED Blinking rate control

```
int led = 2;
void setup() {
      Serial.begin(9600);
      pinMode(led,OUTPUT);
}
void loop() {
      int potval = analogRead(A0);
      Serial.print("POT Value = ");
      Serial.println(potval);
      digitalWrite( led , HIGH );
      delay(potval);
      digitalWrite( led , LOW );
      delay(potval);
}
```

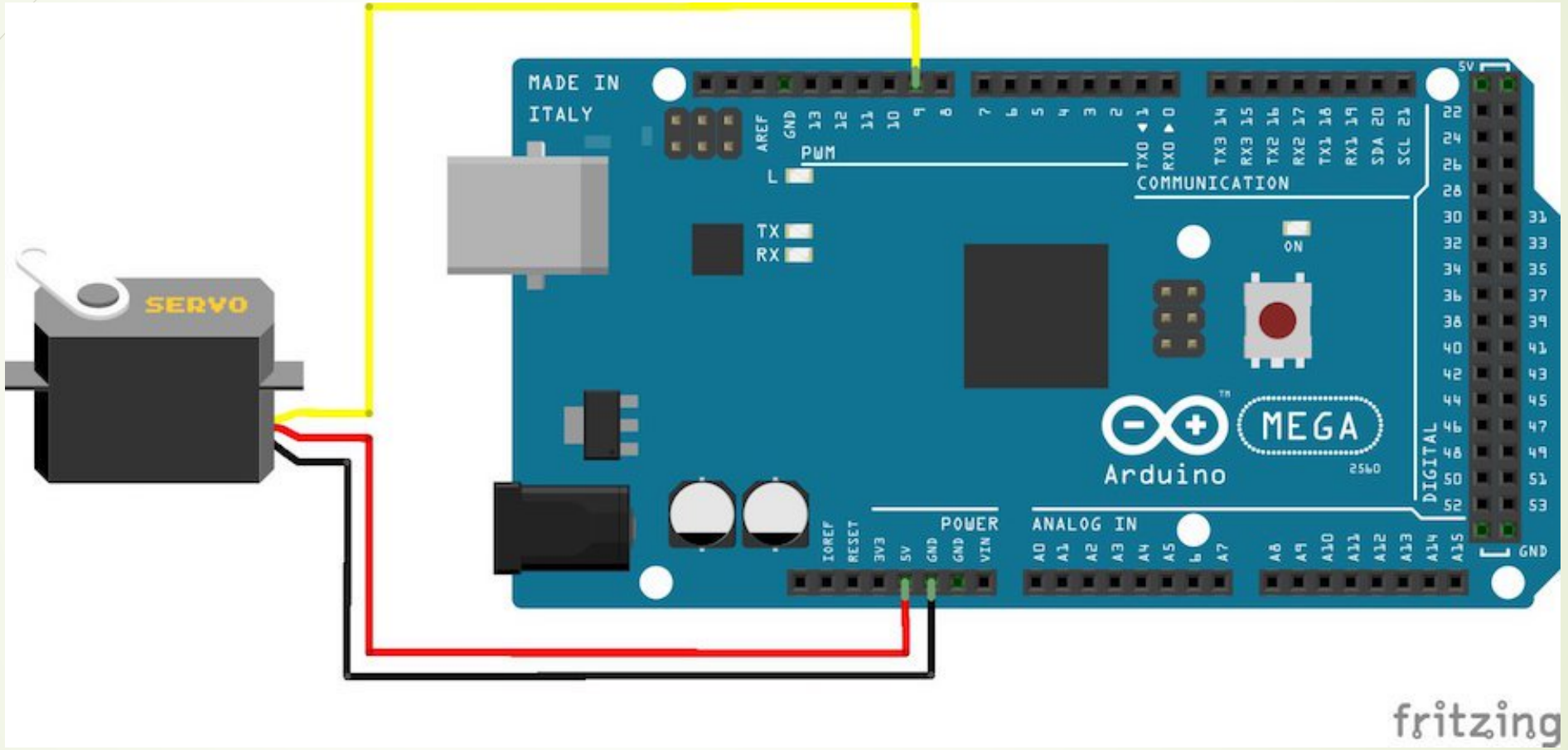
Output


Serial Monitor
IOT Devices
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0

Servo Motor Interfacing

- You can connect small servo motors directly to an Arduino to control the shaft position very precisely.
- Because servo motors use feedback to determine the position of the shaft, you can control that position very precisely. As a result, servo motors are used to control the position of objects, rotate objects, move legs, arms or hands of robots, move sensors etc. with high precision. Servo motors are small in size, and because they have built-in circuitry to control their movement, they can be connected directly to an Arduino.
- Most servo motors have the following three connections:
- Black/Brown ground wire.
  Red power wire (around 5V).
  Yellow or White PWM wire.
- In this experiment, we will connect the power and ground pins directly to the Arduino 5V and GND pins. The PWM input will be connected to one of the Arduino's digital output pins.

Interfacing Diagram:



Program

```
#include <Servo.h>
Servo myservo;        // create servo object to control a servo
                      // twelve servo objects can be created on most boards
void setup() {
      myservo.attach(9);  // attaches the servo on GIO2 to the servo object
}
void loop() {
      int pos;
      for (pos = 0; pos <= 180; pos= pos +1) { // goes from 0 degrees to 180 degrees
            myservo.write(pos);           // tell servo to go to position in variable 'pos'
            delay(15);                    // waits 15ms for the servo to reach the position
      }
```

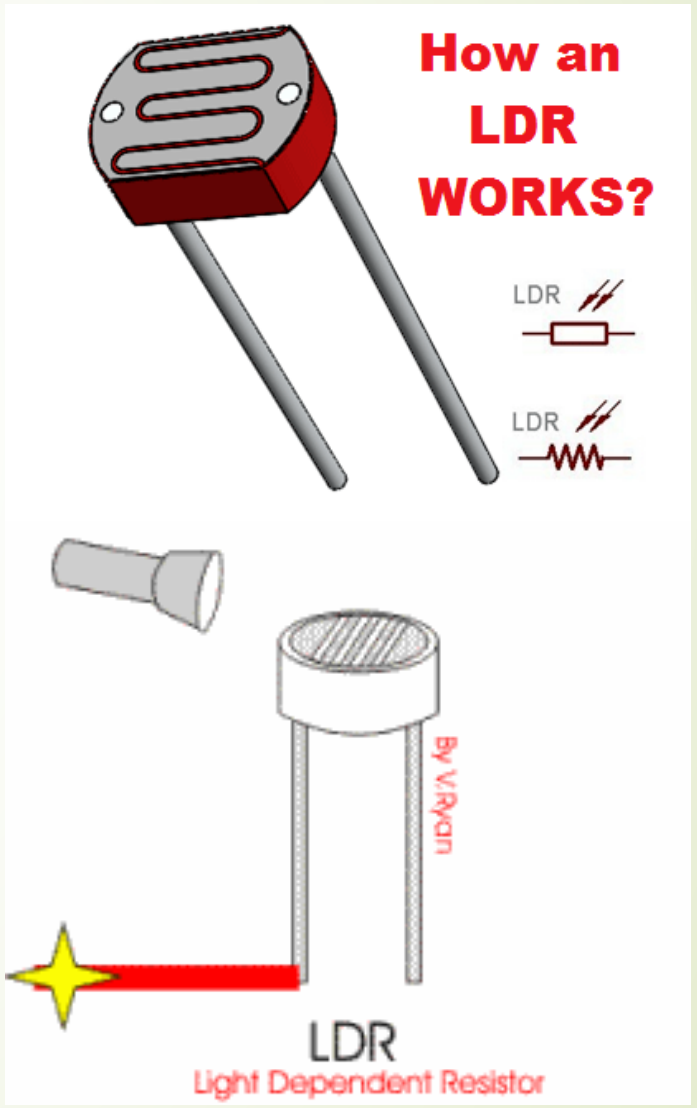
```
    for (pos = 180; pos >= 0; pos =pos- 1) { // goes from 180 degrees to 0 degrees
        myservo.write(pos);              // tell servo to go to position in variable 'pos'
        delay(15);                       // waits 15ms for the servo to reach the position
    }
}
```

Assignment

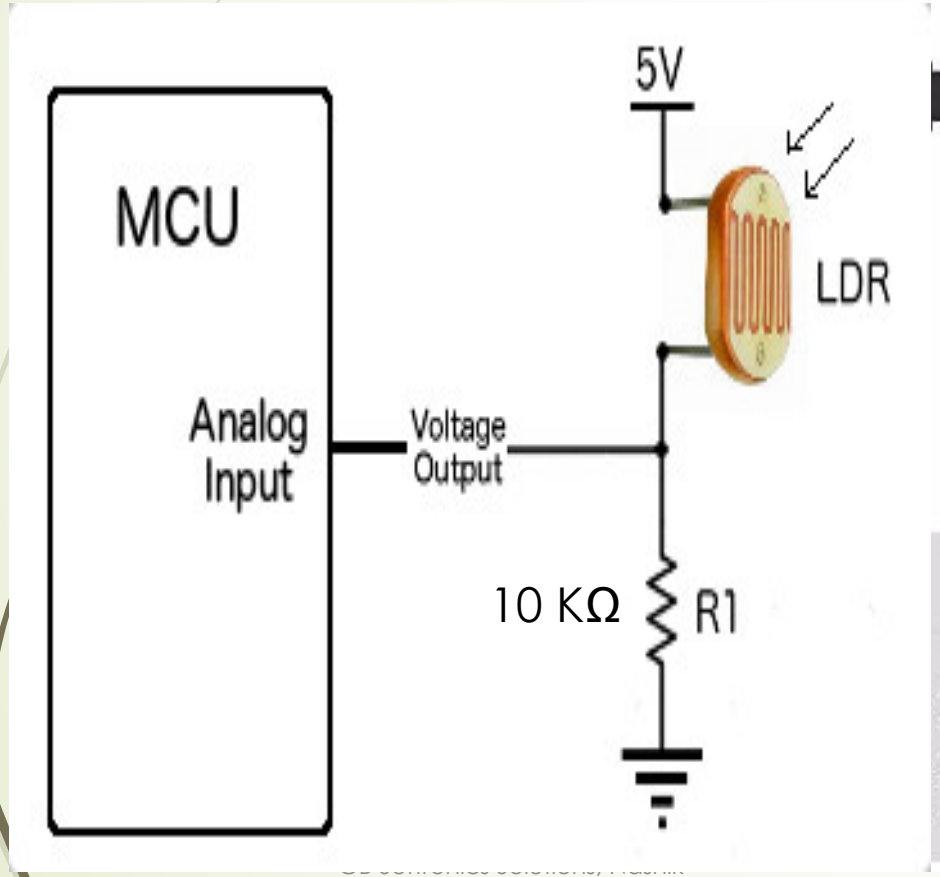
- **Interface Servo Motor and Pot to arduino. Write a program for positional control of servo motor depends on pot value.**

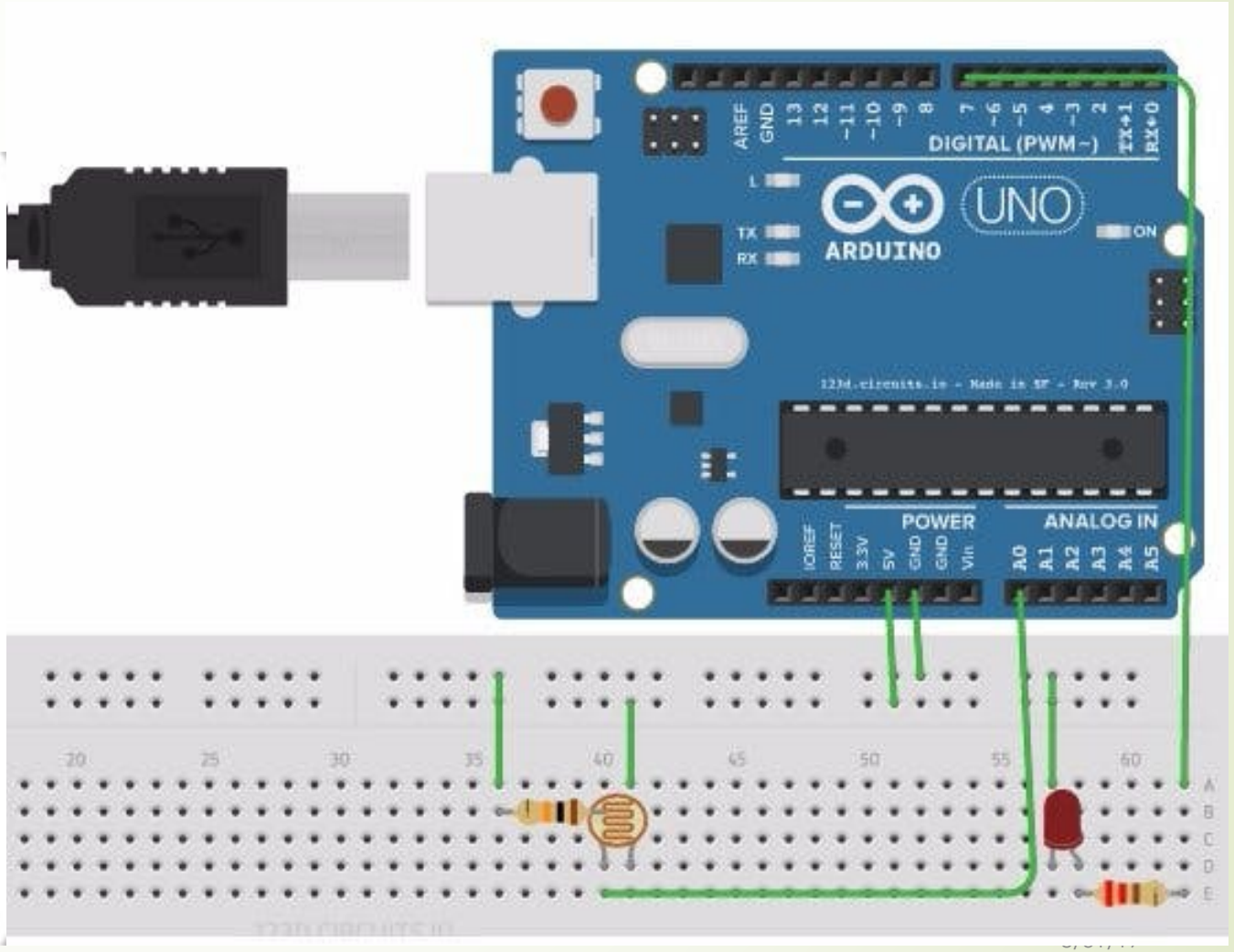
Light Dependent Resistor(LDR)

- *Light dependent resistors, LDRs, or photoresistors are often used to detect light and change the operation of a circuit dependent upon the light levels.*
- They can be described by a variety of names from light dependent resistor, LDR, photoresistor, or even photo cell, photocell or photoconductor.
- Applications: photographic light meters, fire or smoke alarms as well as burglar alarms, and they also find uses as lighting controls for street lamps.
-



Circuit Diagram





Program: (Output on Serial Monitor)

```
void setup() {
    Serial.begin(9600);
}
void loop() {
        int LDRval = analogRead(A0);      // LDR connected to A0
        Serial.print("LDR Value = ");
        Serial.println(LDRval);
        delay(500);
}
```

Changing LED brightness using LDR

```
int led = 2;
void setup() {
      Serial.begin(9600);
      pinMode(led,OUTPUT);
}
void loop() {
      int LDRval = analogRead(A0);
      Serial.print("LDR Value = ");
      Serial.println(LDRval);
      int val = map(LDRval,0,1023,0,255);
      analogWrite(led,val);
      delay(100);
}
```

LED Blinking rate control using LDR

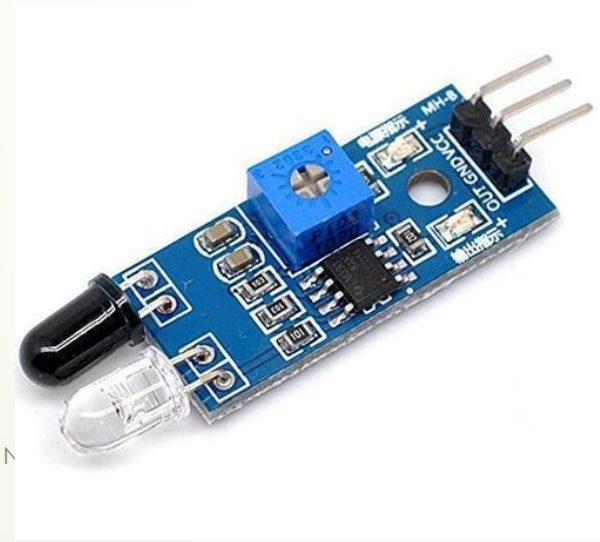
```
int led = 2;
void setup() {
    Serial.begin(9600);
    pinMode(led,OUTPUT);
}
void loop() {
    int LDRval = analogRead(A0);
    Serial.print("LDR Value = ");
    Serial.println(LDRval);
    digitalWrite( led , HIGH );
    delay(LDRval);
    digitalWrite( led , LOW );
    delay(LDRval);
}
```

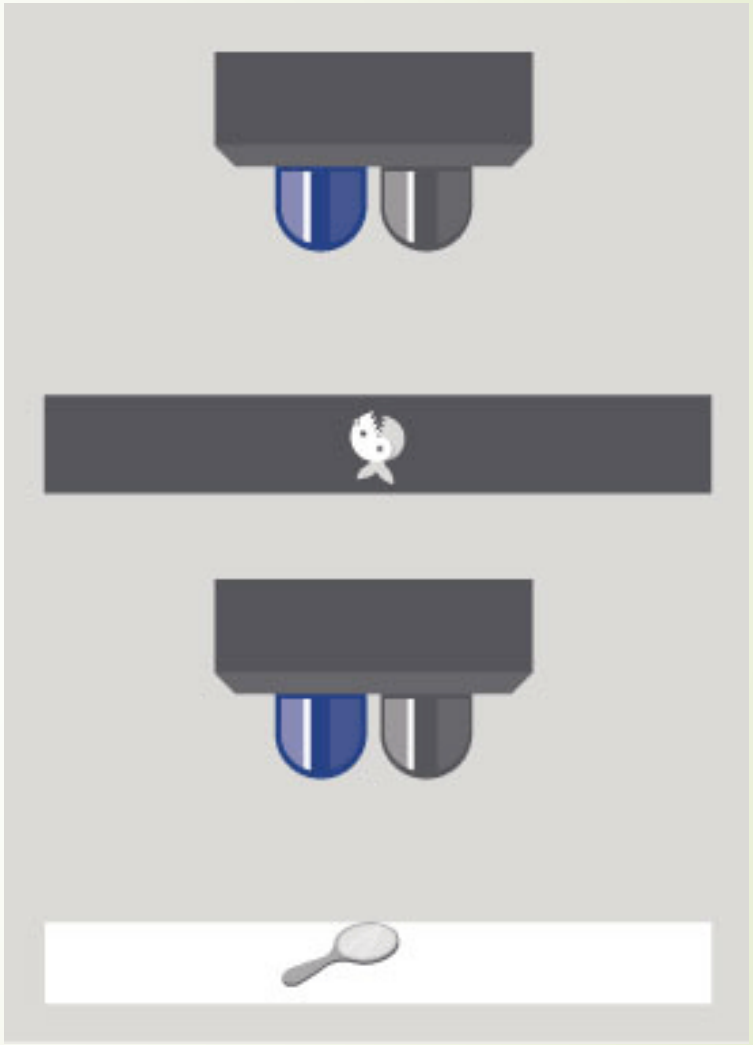
LED and Buzzer Control on Analog Sensor value

```
int led = 2;
int buzzer = 3;
void setup() {
      Serial.begin(9600);
      pinMode(led,OUTPUT);
      pinMode(buzzer,OUTPUT);
}
void loop() {
      int potval = analogRead(A0);
      Serial.print("POT Value = ");
      Serial.println(potval);
      if( potval >=700 ) {
            digitalWrite(led, HIGH);
            digitalWrite(buzzer,HIGH);
            Serial.println("Light Detected!");
      }
      else {
            digitalWrite(led, LOW);
            digitalWrite(buzzer,LOW);
            Serial.println("Darkness Detected");
      }
      delay(500);
}
```

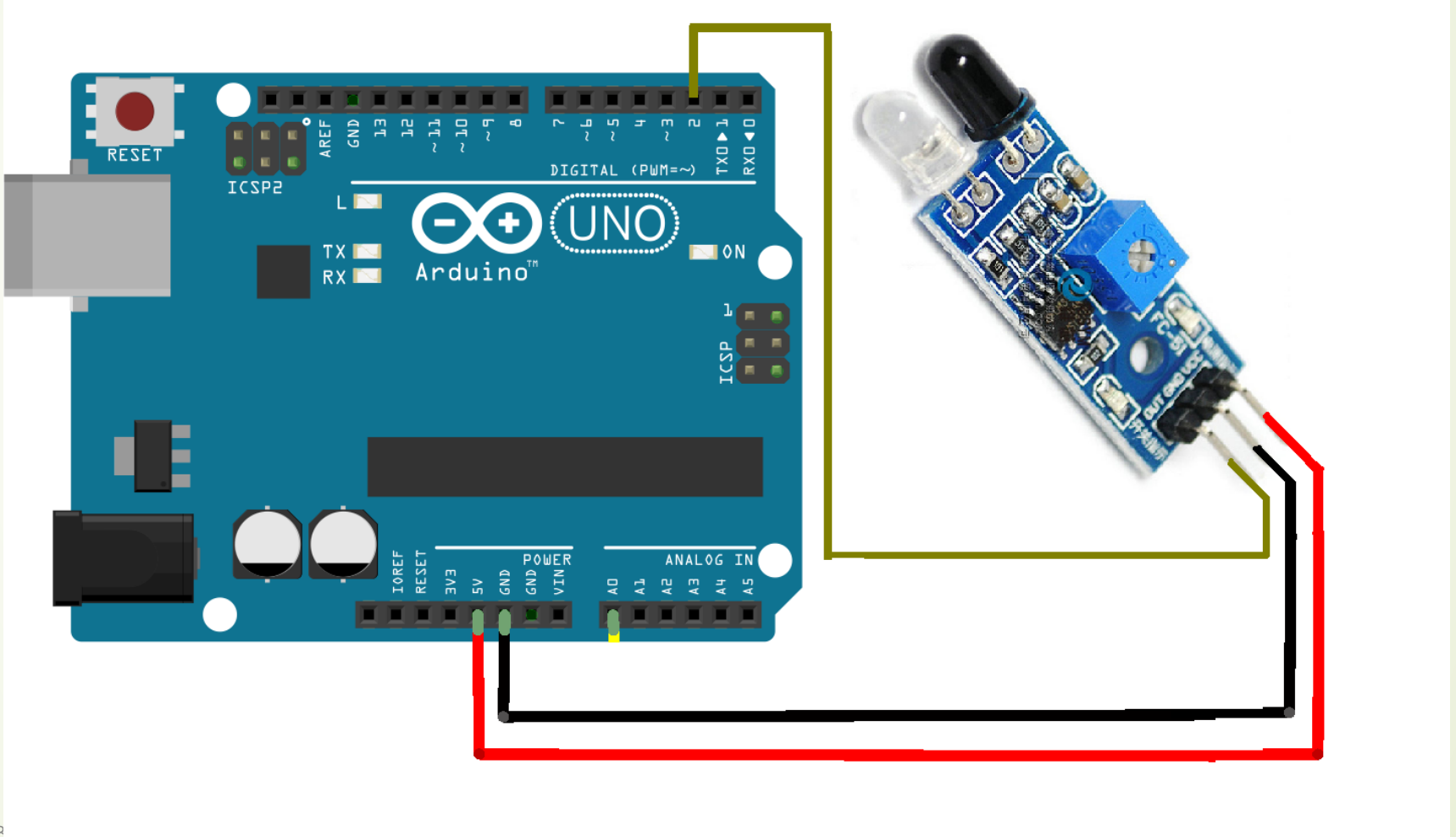
IR Sensor

- IR (Infrared) Sensor is used for obstacle detection.
- It having IR Transmitter(black LED) and IR Receiver(White LED).
- It is digital Sensor.
- When Obstacle detected, IR sensor will send digital 0 to Arduino and In case of absence of obstacle, IR sensor will send digital 1.





Circuit Diagram



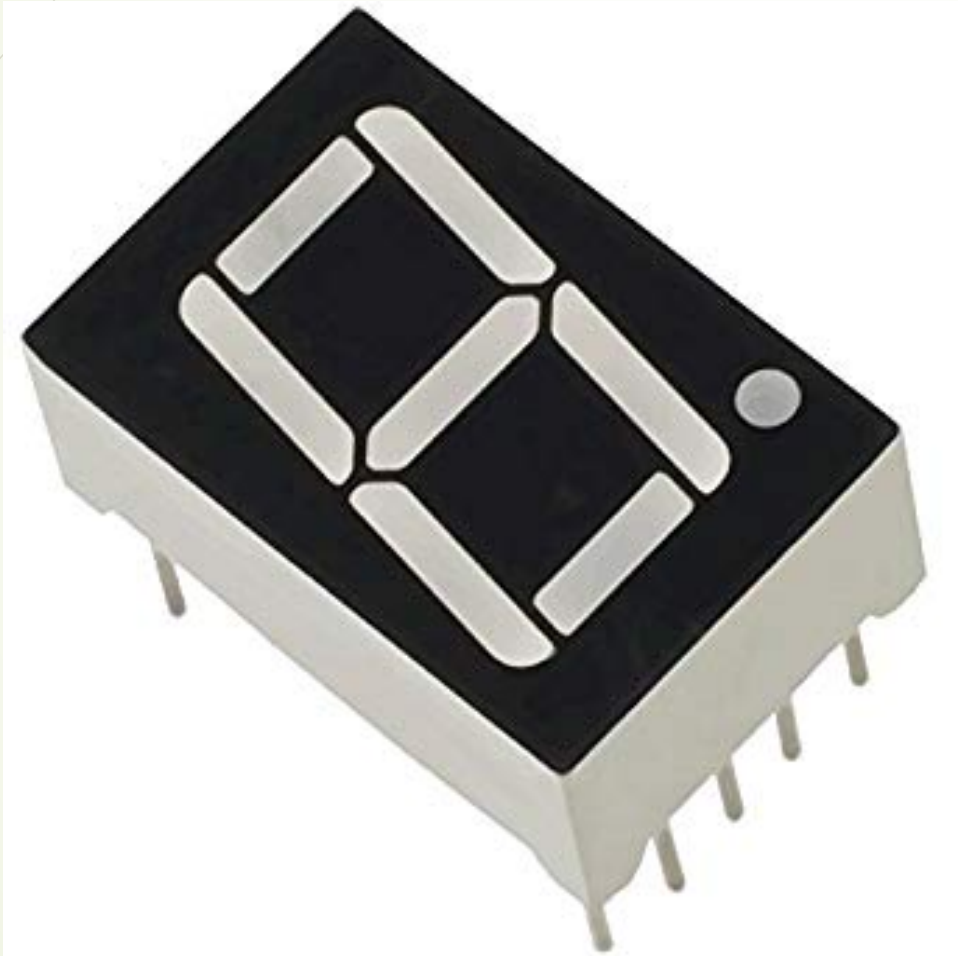
Program

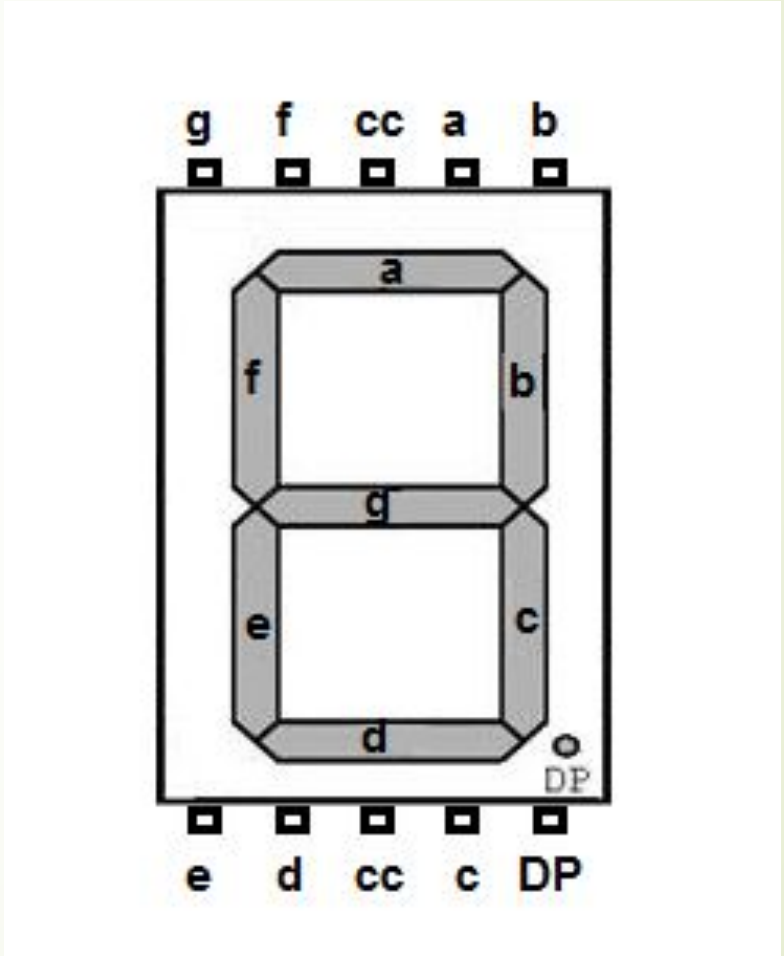
```
int ir = 2;
void setup() {
        pinMode(ir , INPUT);
        Serial.begin(9600);
}
void loop(){
        if( digitalRead( ir ) == 0) {
                Serial.println("Obstacle Detected !! ");
        }
        else {
                Serial.println("Obstacle Not Detected ");
        }
}
```

```
int ir = 2;
int led = 3;
void setup() {
        pinMode( ir , INPUT);
        pinMode( led, OUTPUT);
        Serial.begin(9600);
}
void loop(){
        if( digitalRead( ir ) == 0) {
                digitalWrite(led, HIGH);
                Serial.println("Obstacle Detected !! ");
        }
        else {
                Serial.println("Obstacle Not Detected ");
                digitalWrite(led, LOW);
        }
}
```

```
int ir = 2;
int buzzer = 4;
void setup() {
        pinMode( ir , INPUT);
        pinMode( buzzer, OUTPUT);
        Serial.begin(9600);
}
void loop(){
        if( digitalRead( ir ) == 0) {
                digitalWrite(buzzer, HIGH);
                Serial.println("Obstacle Detected !! ");
        }
        else {
                Serial.println("Obstacle Not Detected ");
                digitalWrite(buzzer, LOW);
        }
}
```

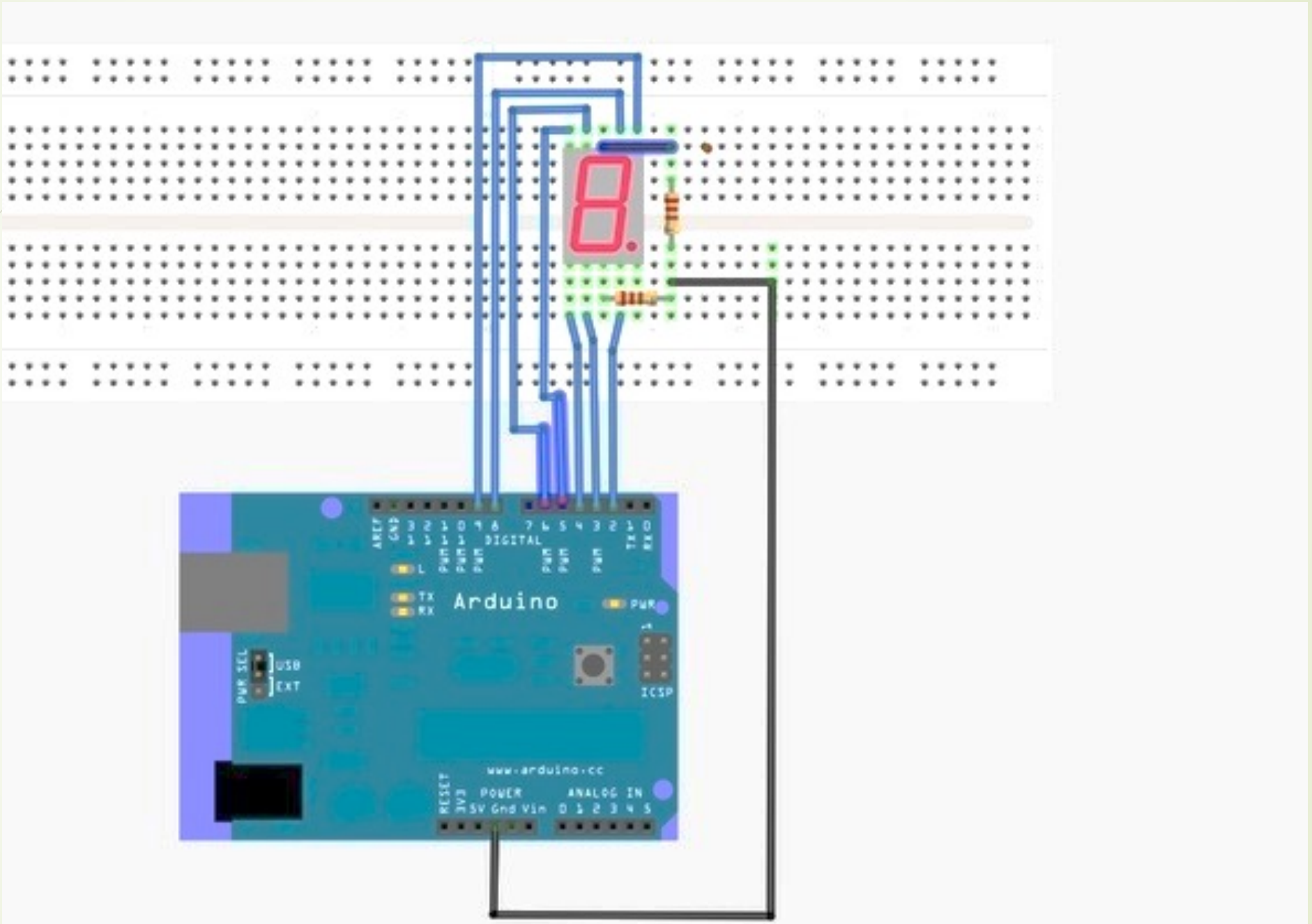
7 Segment Interfacing





7 Segment Codes

Numbers	Common Cathode		Common Anode	
	(DP)GFEDCBA	HEX Code	(DP)GFEDCBA	HEX Code
0	00111111	0x3F	11000000	0xC0
1	00000110	0x06	11111001	0xF9
2	01011011	0x5B	10100100	0xA4
3	01001111	0x4F	10110000	0xB0
4	01100110	0x66	10011001	0x99
5	01101101	0x6D	10010010	0x92
6	011111101	0x7D	10000010	0x82
7	00000111	0x07	11111000	0xF8
8	01111111	0x7F	10000000	0x80
9	01101111	0x6F	10010000	0x90


Arduino
DIGITAL
POWER
ANALOG IN
www.arduino.cc
ICSP
PWR SEL
USB
EXT

Program

```
int a = 2, b = 3, c = 4, d = 5, e = 6, f  = 7, g = 8;
void setup() {
    pinMode(a,OUTPUT);
    pinMode(b,OUTPUT);
    pinMode(c,OUTPUT);
    pinMode(d,OUTPUT);
    pinMode(e,OUTPUT);
    pinMode(f,OUTPUT);
    pinMode(g,OUTPUT);
}
```

Program

```
void loop() {
// Display zero
    digitalWrite( a, HIGH);
    digitalWrite( b, HIGH);
    digitalWrite( c, HIGH);
    digitalWrite( d, HIGH);
    digitalWrite( e, HIGH);
    digitalWrite( f, HIGH);
    digitalWrite( g, LOW);
    delay(1000);
}
```

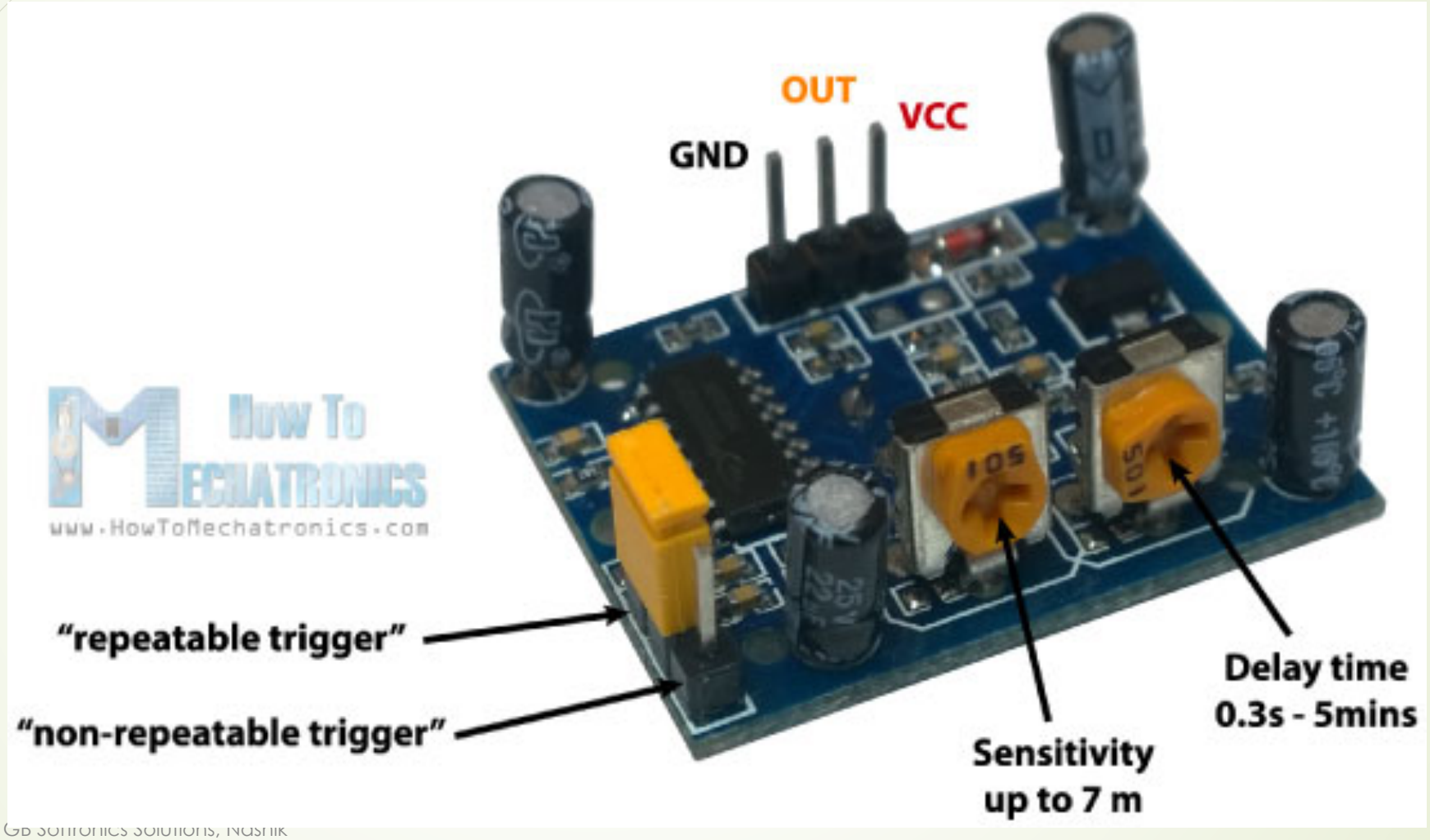
Assignment

- Write a program to display 0 to 9 continuously on seven segment display.
- Write a program to display 9 to 0 continuously on seven segment display.

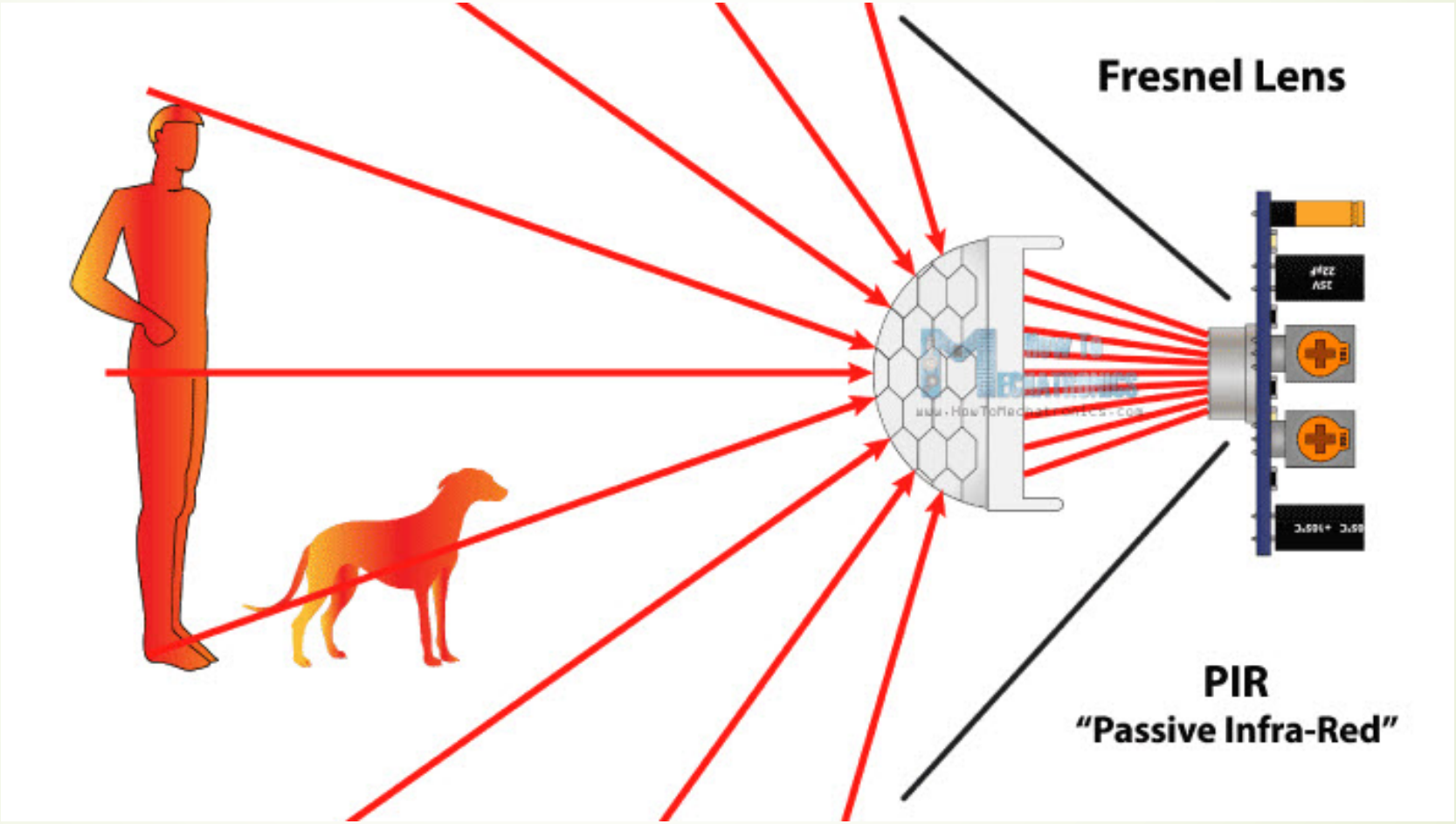
Motion Sensor

- A **passive infrared sensor** (**PIR sensor**) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view.
- They are most often used in PIR-based motion detectors. PIR sensors are commonly used in security alarms and automatic lighting applications.
- PIR sensors detect general movement, but do not give information on who or what moved. For that purpose, an active IR sensor is required.
- PIR sensors are commonly called simply "PIR", or sometimes "PID", for "passive infrared detector".
- The term *passive* refers to the fact that PIR devices do not radiate energy for detection purposes.
- They work entirely by detecting infrared radiation (radiant heat) emitted by or reflected from objects.

Pinout

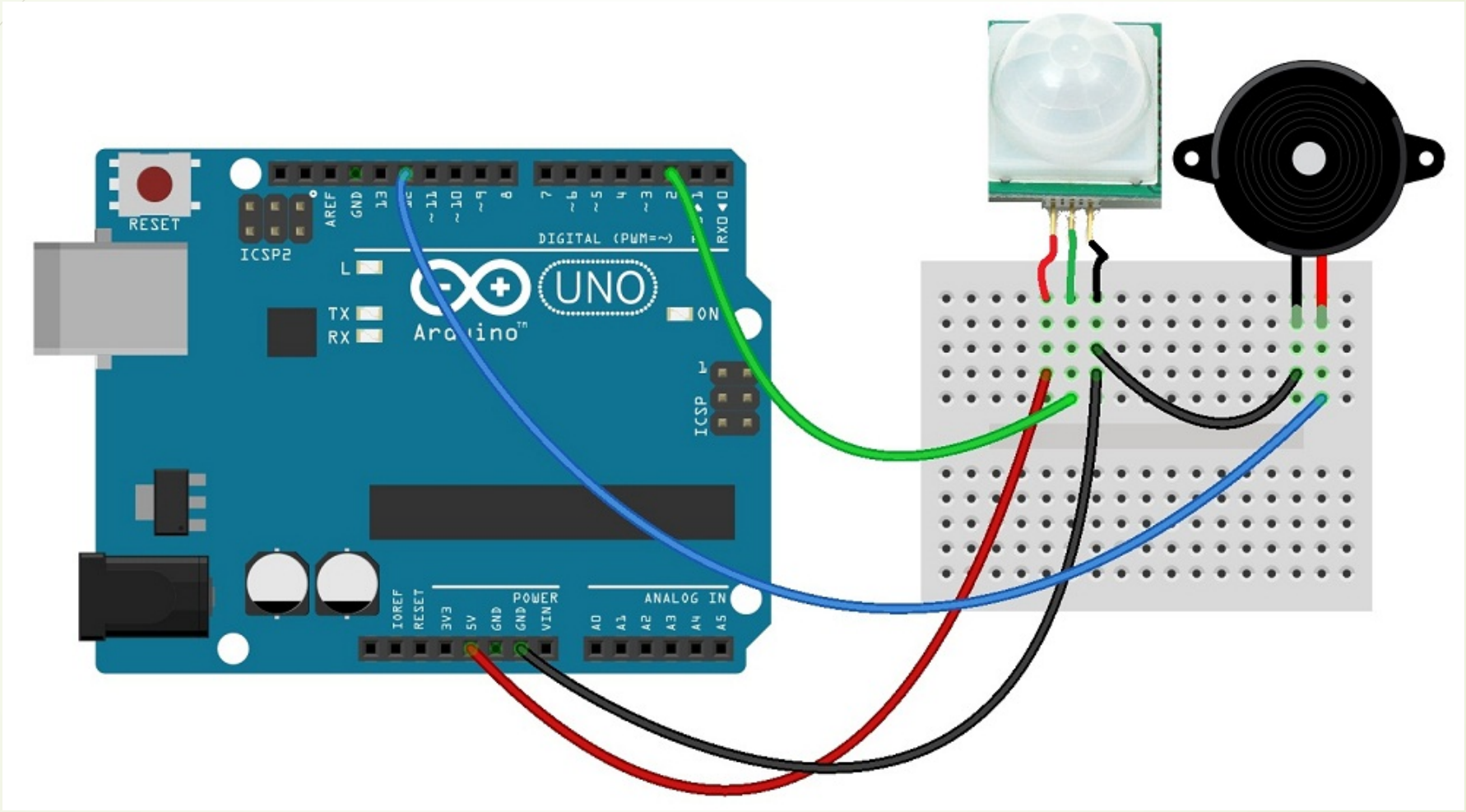

OUT
VCC
GND
How To
MECHATRONICS
www.HowToMechatronics.com
"repeatable trigger"
"non-repeatable trigger"
Delay time
0.3s - 5mins
Sensitivity
up to 7 m

Working



Working Video

Circuit Diagram



Program

```
int pir = 2;
void setup() {
      pinMode( pir , INPUT);
      Serial.begin( 9600 );
}
void loop() {
      int pirstatus = digitalRead( pir );
      if(pirstaus == 1 ) {
            Serial.println("Obstacle Detected ");
      }
      else {
            Serial.println( " Obstacle Not Detected ");
      }
}
```

Program for Motion Alert System

```
int pir = 2;
int buzzer = 12 ;
void setup() {
      pinMode( pir , INPUT);
      pinMode( buzzer , OUTPUT );
      Serial.begin( 9600 );
}
void loop() {
      int pirstatus = digitalRead( pir );
if(pirstaus == 1 )
{
            Serial.println("Obstacle Detected ");
            digitalWrite(buzzer , HIGH);
}
else
{
                  digitalWrite(buzzer, LOW);
}
}
```

Assignment

- Write a program to interface Motion Sensor, buzzer and LED to detect Human motion event. In case of Motion, Switch on buzzer and LED, otherwise it remains off.

Ultrasonic Sensor

- Ultrasonic Sensor HC-SR04 is used for distance measurement.
- It having 4 Pins, VCC, Trigger, Echo, GND
- Trigger pin has to configure as OUTPUT pin on Arduino.
- Echo pin has to configure as INPUT pin on Arduino.
- **HC-SR04 Sensor Features**
- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Measuring angle covered: <15°
- Operating Current: <15mA
- Operating Frequency: 40Hz

Pin Descriptions

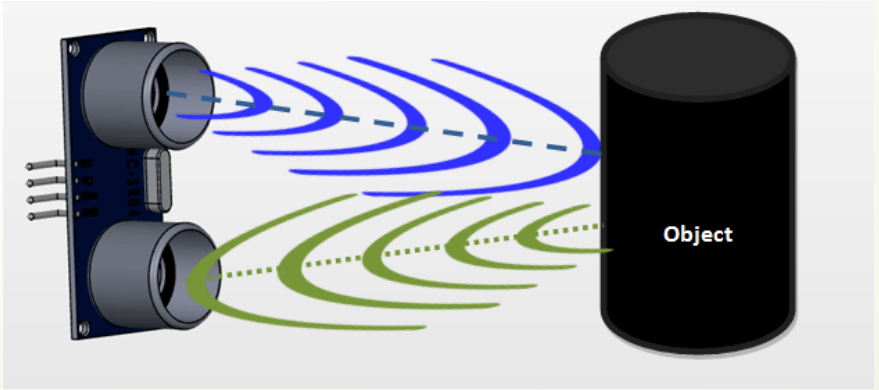
Pin Number	Pin Name	Description
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

Working

- As shown above the **HC-SR04 Ultrasonic (US) sensor** is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively.
- This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required.
- The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

Distance = Speed × Time

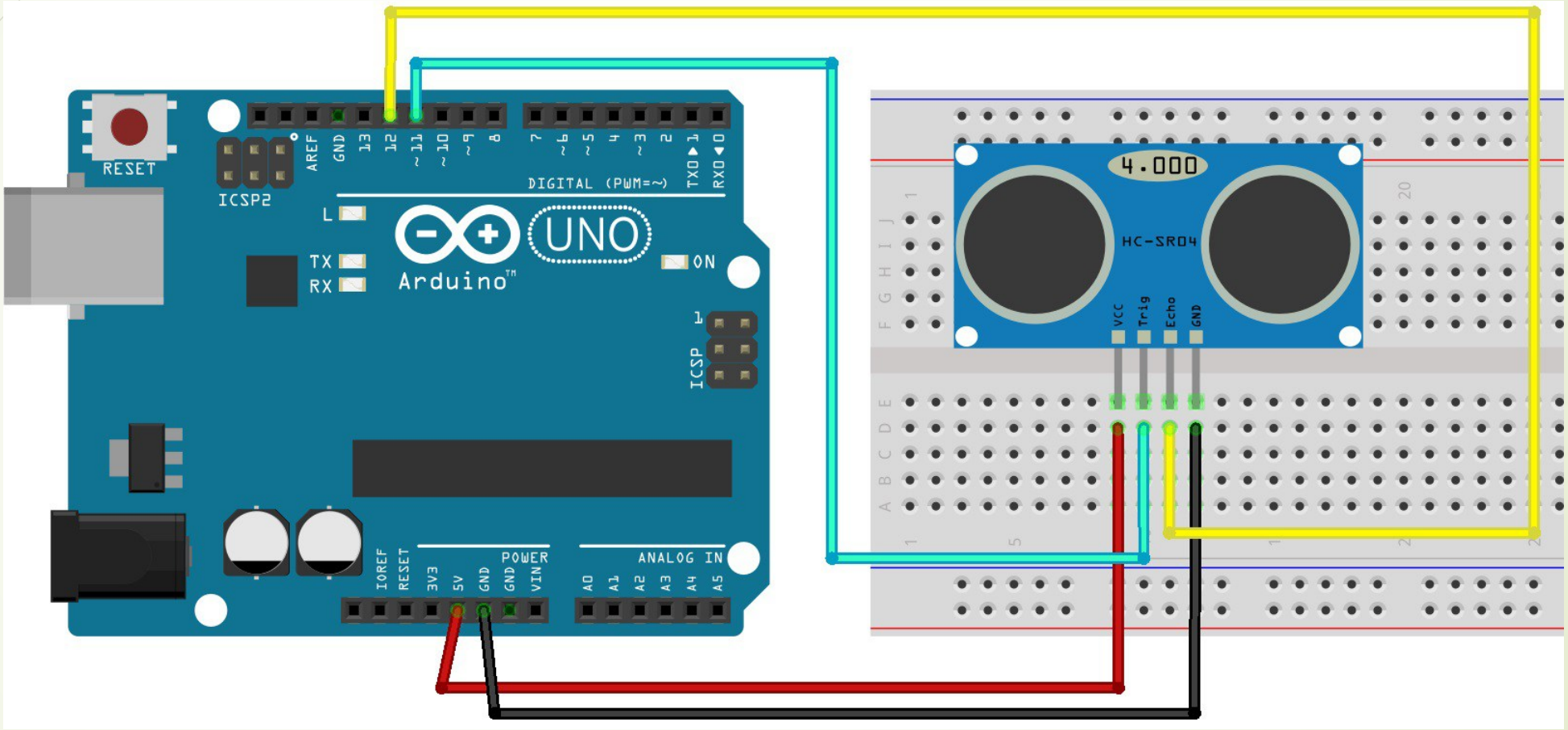
- The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module as shown in the picture below



Continue..

- Now, to calculate the distance using the above formulae, we should know the Speed and time.
- Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is **330m/s.**
- The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken.
- Now simply calculate the distance using a microcontroller or microprocessor.

Circuit Diagram

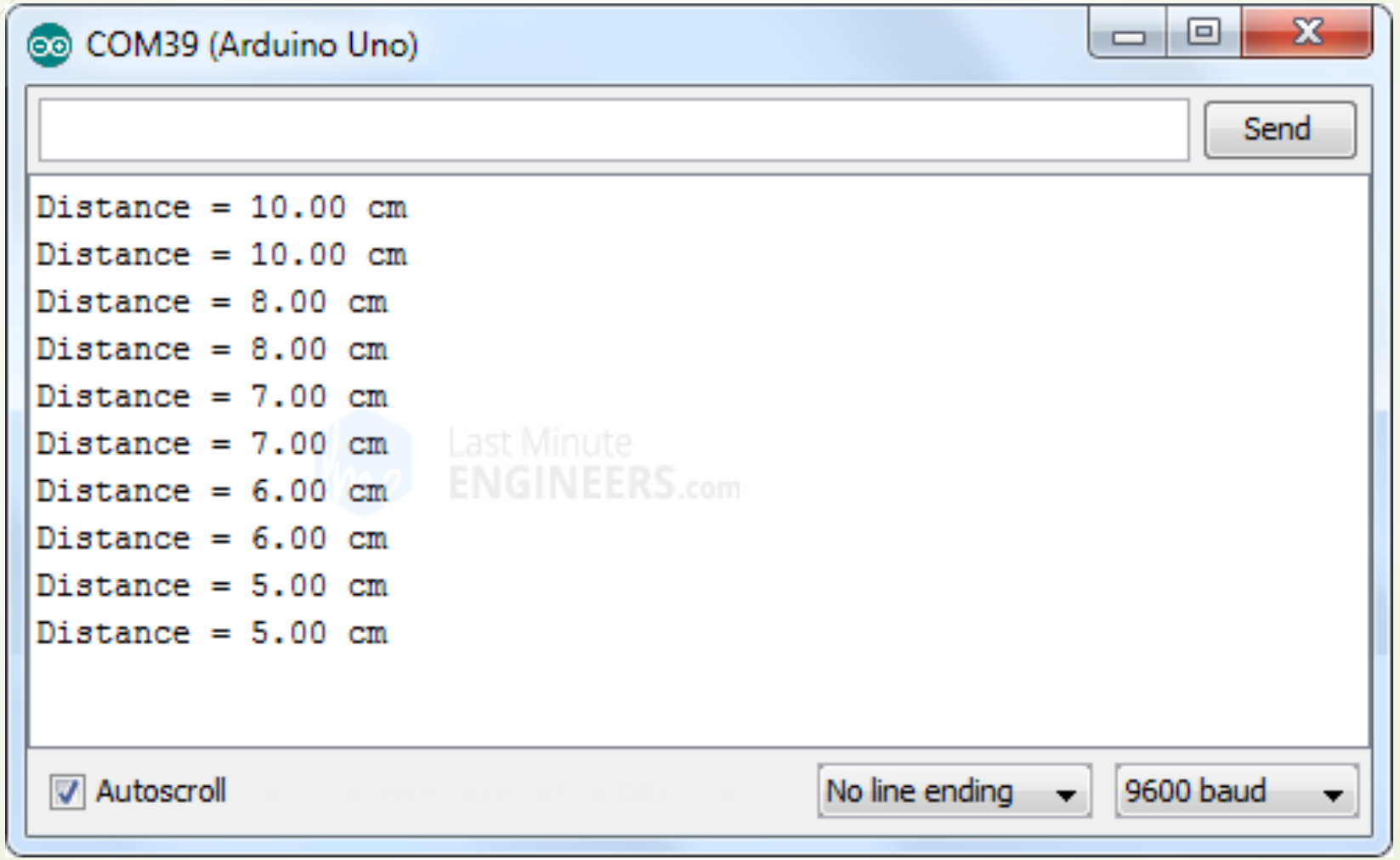


Program

```
int trigPin = 11;
int echoPin = 12;
long duration;
int distance;
void setup() {
      pinMode(trigPin, OUTPUT);
      pinMode(echoPin, INPUT);
      Serial.begin(9600);
}
```

```
void loop() {
      digitalWrite(trigPin, LOW);
      delayMicroseconds(2);
      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);
      duration = pulseIn(echoPin, HIGH);
      // distance = speed * time
      distance= (duration*0.0343)/2;
      Serial.print("Distance: ");
      Serial.print(distance);
      Serial.println(" cm");
      delay(2000);
}
```

Output


COM39 (Arduino Uno)
Send
Distance = 10.00 cm
Distance = 10.00 cm
Distance = 8.00 cm
Distance = 8.00 cm
Distance = 7.00 cm
Distance = 7.00 cm
Distance = 6.00 cm
Distance = 6.00 cm
Distance = 5.00 cm
Distance = 5.00 cm
Last Minute ENGINEERS.com
Autoscroll
No line ending
9600 baud

Program

```
int trigPin = 11;
int echoPin = 12;
long duration;
int distance;
void setup() {
      pinMode(trigPin, OUTPUT);
      pinMode(echoPin, INPUT);
      Serial.begin(9600);
}
void loop() {
      digitalWrite(trigPin, LOW);
      delayMicroseconds(2);
      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);
      duration = pulseIn(echoPin, HIGH);
     // distance = speed * time
      distance= (duration*0.0343)/2;
      Serial.print("Distance: ");
      Serial.print(distance);
      Serial.print(" cm");
      if( distance <= 5){
            Serial.println(Obstacle Present”);
      }
      delay(2000);
}
```

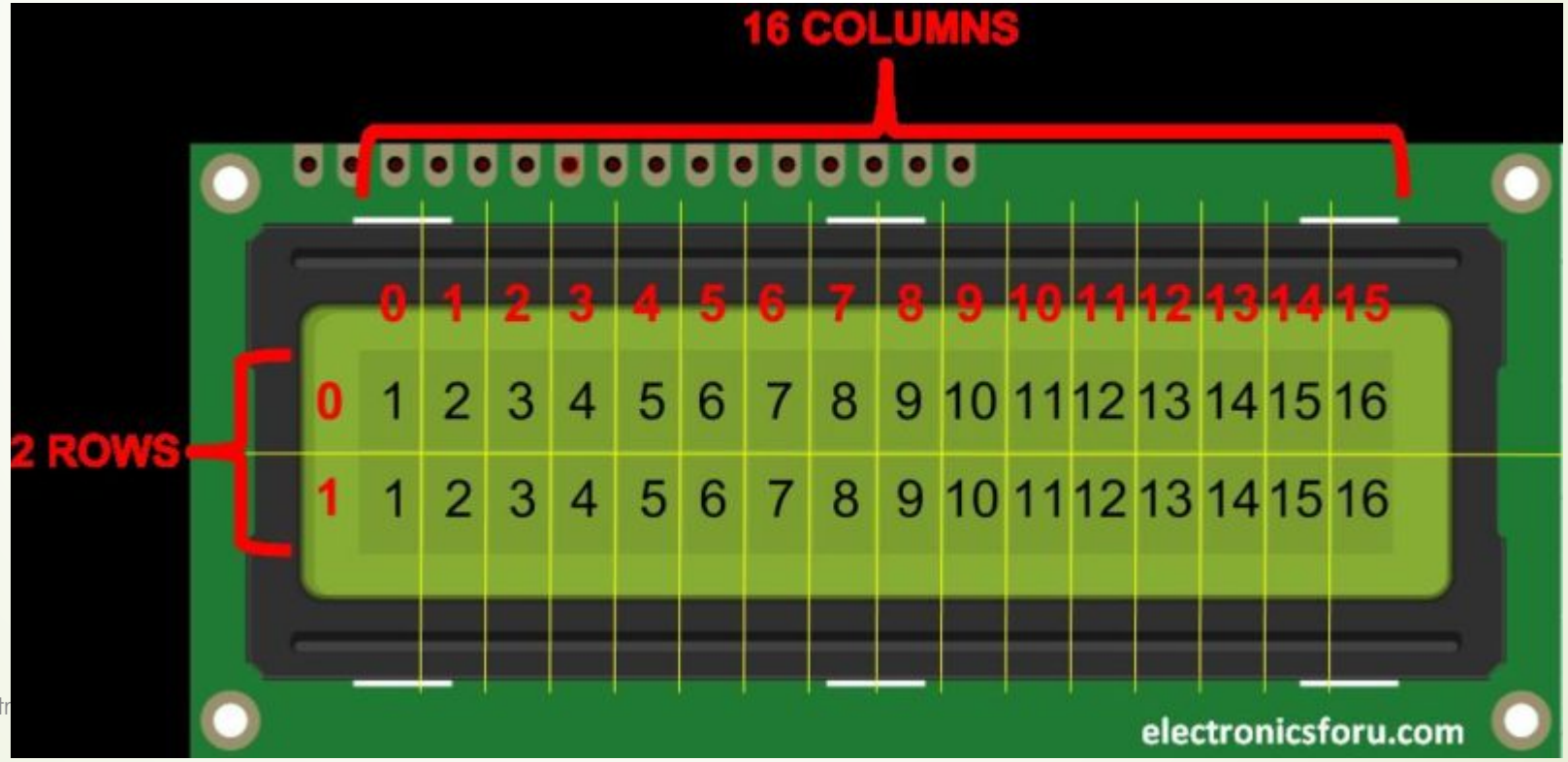
Ultrasonic Radar

Assignment

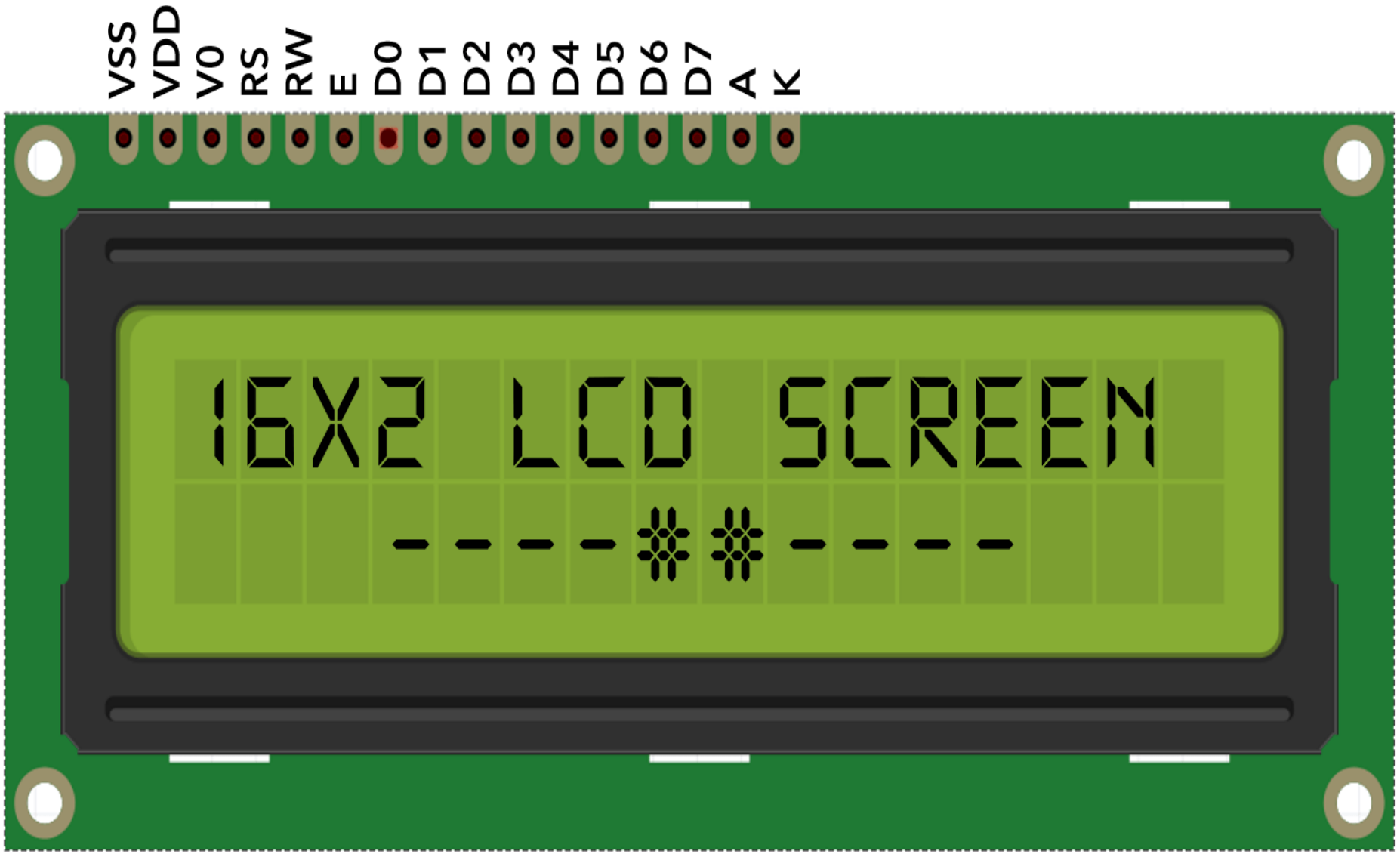
- Interface Ultrasonic Sensor, Buzzer with Arduino. Buzzer will start ringing when, obstacle is detected in 10 cm range, otherwise, buzzer remains off.

LCD Interfacing

- 16x2 character LCD Display
- 16 Columns and 2 rows available.
- To display Single character, 5x7 Matrix LED used means, 5 columns and 7 rows.
- SO, to display single character, total 35 pixels are used.

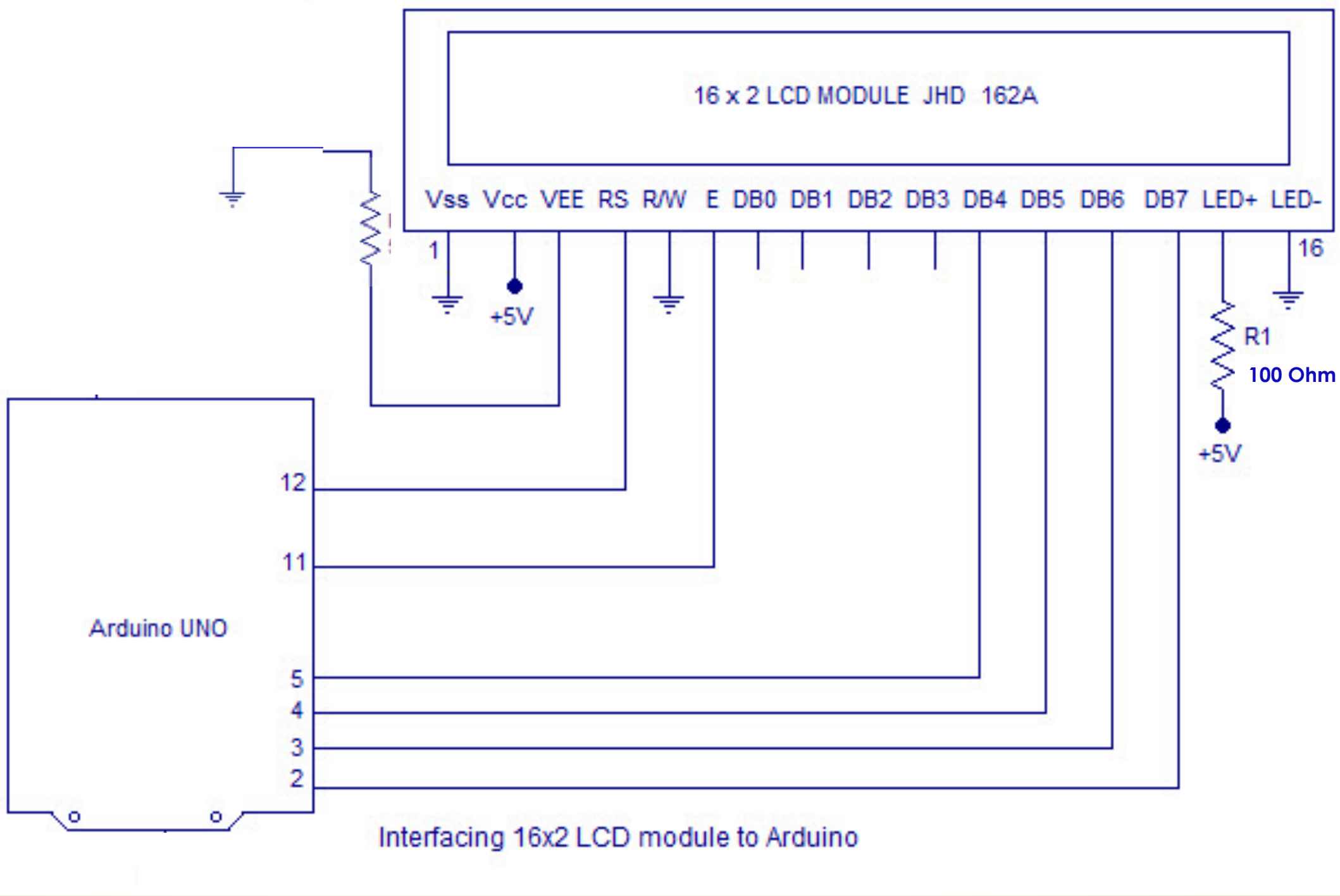


Pinout



No	Symbol	Function
1	VSS	Ground
2	VDD	5V +
3	V0	Contrast
4	RS	Register
5	RW	Read/Write
6	E	Enable
7	D0	Data bus
8	D1	Data bus
9	D2	Data bus
10	D3	Data bus
11	D4	Data bus
12	D5	Data bus
13	D6	Data bus
14	D7	Data bus
15	A	Anode (5V+)
16	K	Cathode (GND)

Circuit Diagram



Interfacing 16x2 LCD module to Arduino

Program

```
#include <LiquidCrystal.h>

//LiquidCrystal(RS,EN,D4,D5,D6,D7);

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
        lcd.begin(16, 2);

        lcd.setCursor(0,0);
        lcd.print("Hello World!");
}

void loop() {

}
```

Program

```
#include <LiquidCrystal.h>

//LiquidCrystal(RS,EN,D4,D5,D6,D7);

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
        lcd.begin(16, 2);

        lcd.setCursor(0,0);
        lcd.print("Hello World!");

        lcd.setCursor(0,1);
        lcd.print("Good Morning");

}

void loop() {
}
```

Program

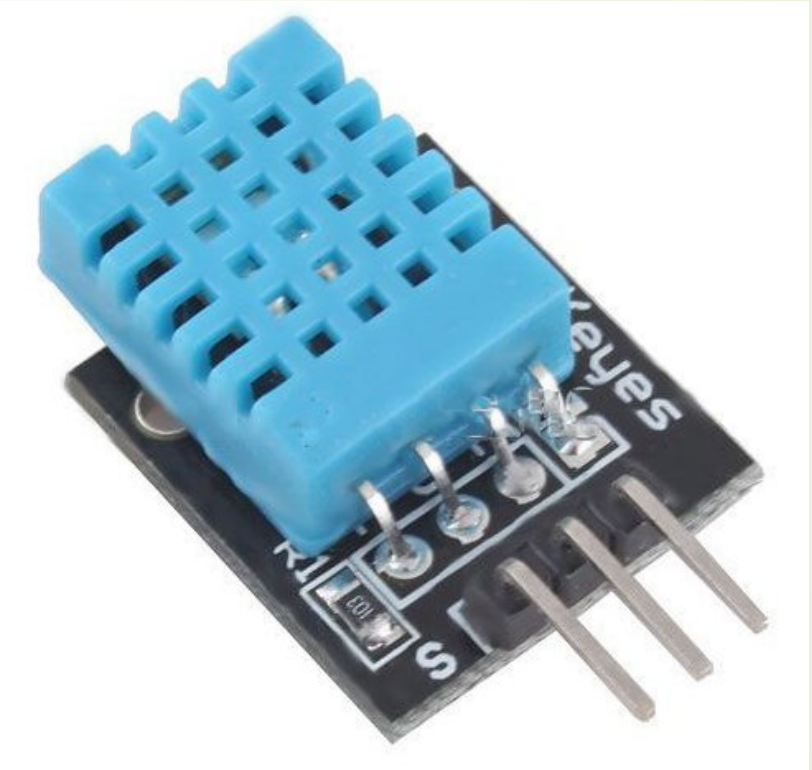
```
#include <LiquidCrystal.h>
//LiquidCrystal(RS,EN,D4,D5,D6,D7);
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
    lcd.begin(16, 2);
    lcd.print("hello, world!");
}
void loop() {
    lcd.setCursor(0, 1);  // print the number of seconds since reset
    lcd.print(millis() / 1000);
}
```

Assignment

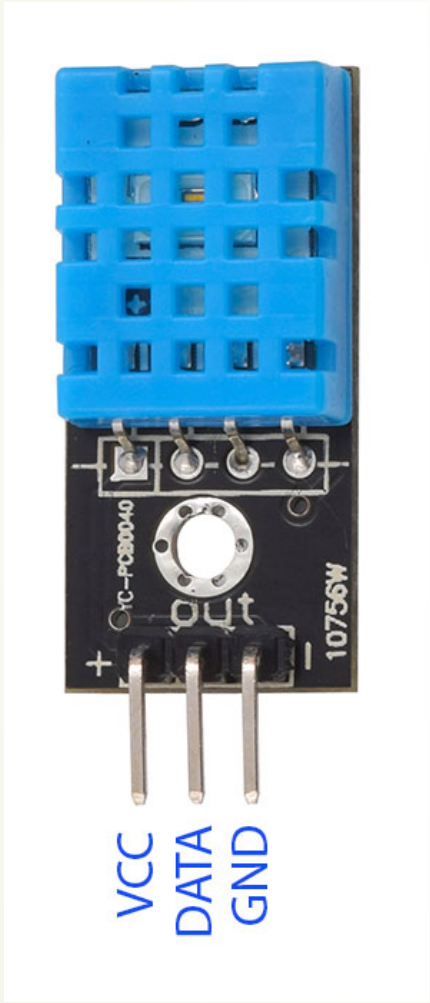
- Interface Potentiometer and LCD to arduino. Display POT reading on LCD.
- Interface LDR and LCD to arduino. Display LDR reading on LCD.
- Interface Motion Sensor and LCD to arduino and display “Motion Detected” and Motion Not Detected” on LCD on particular event.

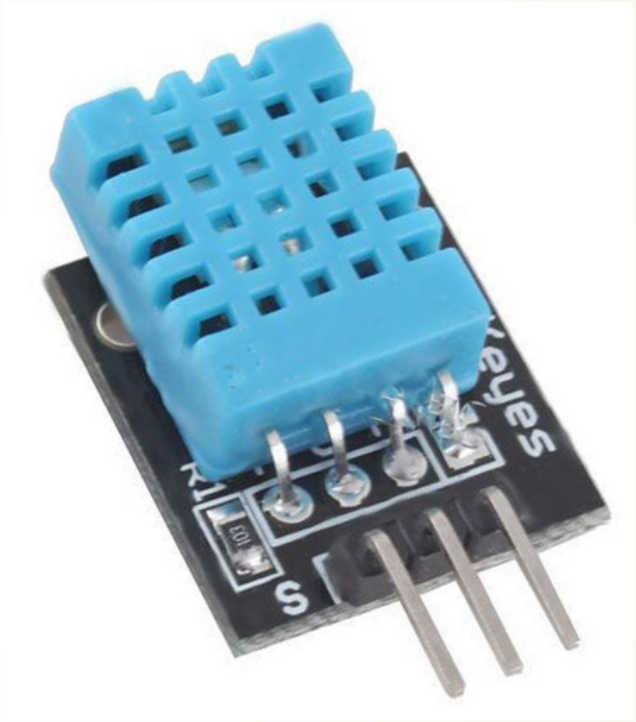
DHT11 interfacing with arduino

- DHT11 sensor is used to measure the **temperature** and **humidity**. It has a resistive humidity sensing component and a negative temperature coefficient (NTC). An 8 bit MCU is also connected in it which is responsible for its fast response. It is very inexpensive but it gives values of both temperature and humidity at a time.



DHT11 Module Pinout

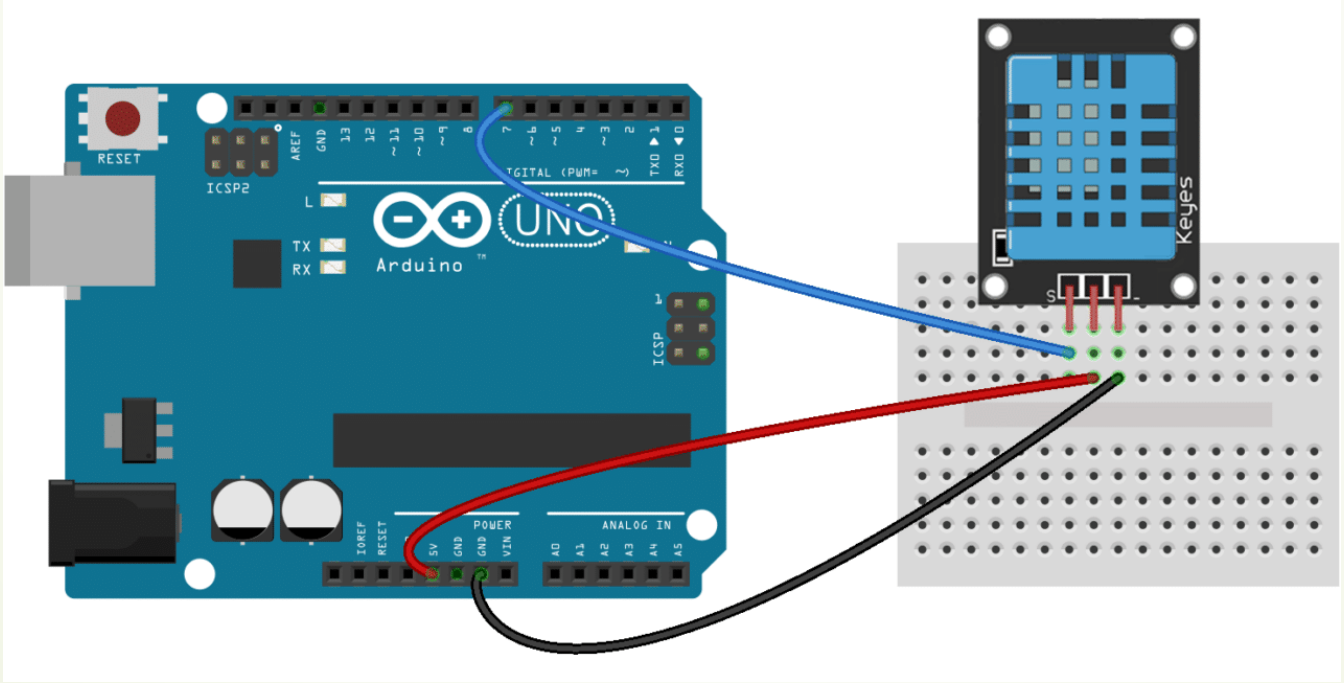




Data

DHT11 interfacing with arduino

- First of all connect the ground and the VCC of the DHT11 temperature and humidity sensor to the ground and 5v of the **Arduino**. Then connect the data pin of the DHT11 sensor to the pin 2 of the Arduino.

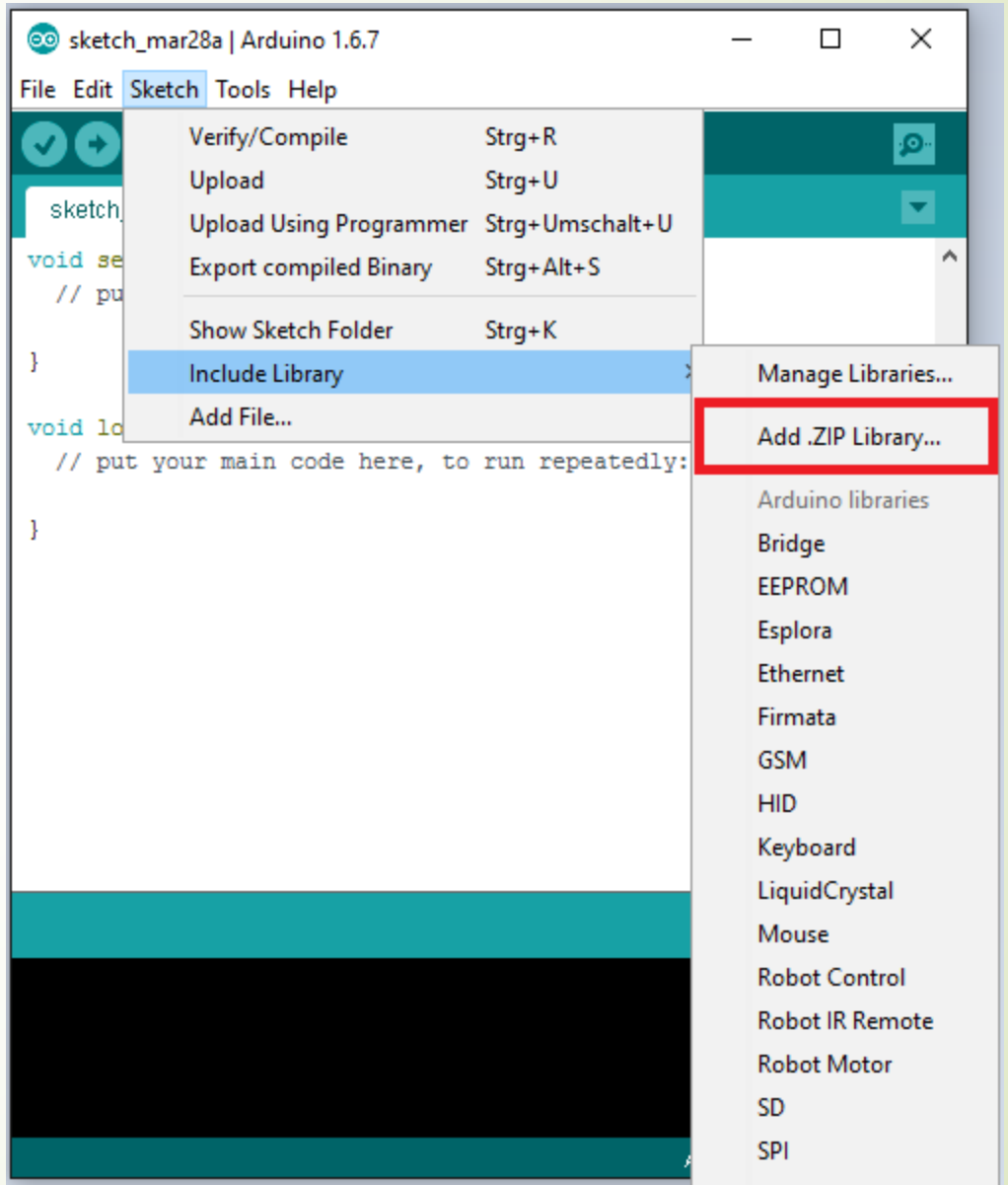


Check your Pin Layout of DHT11 before Connecting to Arduino Power Supply

Installing the DHT11 Library

- Adafruit_Sensor-master.zip
- DHT-sensor-library-master.zip

Note: Copy these zip files to D: drive, then provide path of these files.

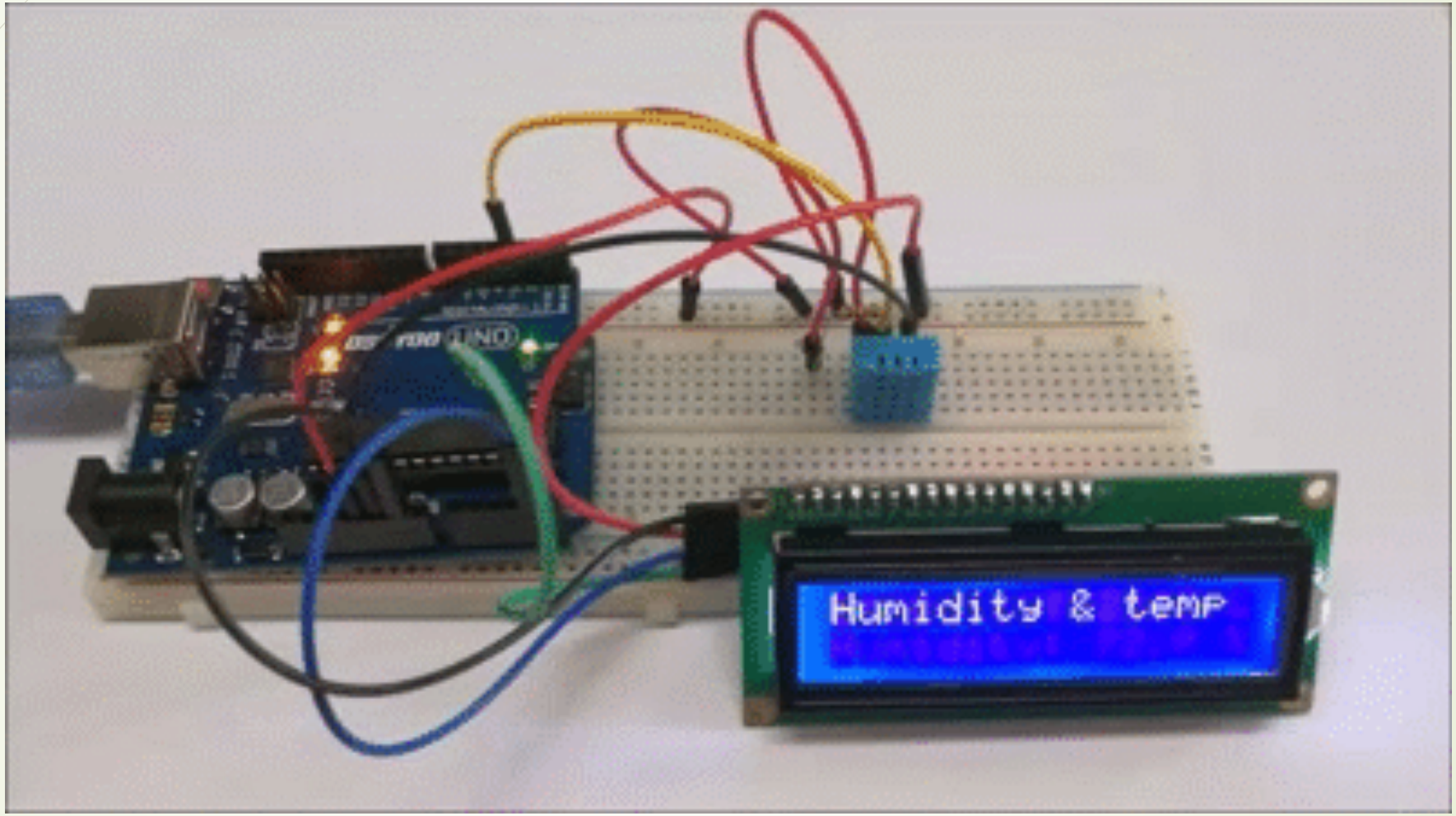


DHT11 interfacing with arduino

```
#include " DHT.h "          // Including the library of DHT11 temperature and
                            // humidity sensor
#define DHTPIN 2            // Selecting the pin at which we have connected
                            // DHT11
#define DHTTYPE DHT11       // Selecting the type of DHT sensors
DHT dht ( DHTPIN, DHTTYPE ) ;
void setup ( ) {
      Serial.begin ( 9600 ) ;
      dht.begin (  ) ;                  // The sensor will start working
}
```

```
void loop ( )
{
    float humidity = dht.readHumidity ( ) ;
    float temp = dht.readTemperature ( ) ;
    Serial.print ( " Temp is " ) ;
    Serial.print ( temp ) ;      // Printing the temperature on display.
    Serial.println ( " C " ) ;       // Printing " C "  on display.
    Serial.print ( " Humidity in % is : " ) ;
    Serial.print ( humidity ) ;   // Printing the humidity on display
    Serial.print ( " % \t " ) ;       // Printing "%" on display
}
```

Output



Project Topics

PROJ_1: Electronic Notice Board [Gr 1,2,19,20,27]

- Any Message send from Serial Monitor should get displayed on LCD. When new message sends, previous message gets automatically erased and replaced with new message.

PROJ_2: Electronic Smart Blind Stick [Gr 3,5,11,21,26]

- If someone is in front of blind person, LED and Buzzer should on and LCD will show the message" **Obstacle. Be Alert"** otherwise LED and Buzzer will remains off and LCD show the message **"Safe.. Keep Walking".**

PROJ_3: Home Automation through PC [Gr 4,6,12,22,28]

- Design and develop project to control 8 home devices through PC serial monitor, LCD connected on project will shows Status of Devices is on or off. Also show the status of all devices on serial monitor.

Project Topics

PROJ_4: Green house atomization [Gr 7,10,13,14,23]

- Design a system for green house. When temperature and humidity goes beyond defined threshold value, automatically LED and Buzzer will on and LCD and Serial Monitor show message **"Alert! Start Cooler & Sprinkler"** otherwise, it normally shows current readings of temperature and humidity on LCD.

PROJ_5: Automatic Street Light [Gr 8,15,16,24]

- Street Light should automatically ON at evening and automatically OFF at morning. LCD and Serial Monitor shows Light Intensity value on First Line and Status of Street Light on Second Line. USE RGB LED for street Light and use orange color.

PROJ_6: Motion enabled Room Light [Gr 9,17,18,25]

- Light present in Room should automatically ON when human motion is detected and automatically OFF in the absence of human motion. LCD and Serial monitor shows appropriate message as "Motion detected! Light ON" and "No Motion! Light OFF" when particular condition fulfilled.

BEST OF LUCK

Contact Details:

Mr. Ganesh L Attarde

Founder, GB Softronics Solutions, Nashik

Mob: 7588615326 Email: gbsoftronics@gmail.com

All Arduino Codes available at:

https://github.com/arduinoganesh/arduino