

INDEX**EXPERIMENTS ON TINKERING LAB:-**

S.No	Name of the Experiment	Page No.	Date	Marks	Sign
1.	Parallel and Series Circuits				
2.	Traffic Light Circuit Simulation in Tinkercad				
3.	Automatic Street Light Using LDR				
4.	Simulation of LED Blinking using Arduino on Tinkercad				
5.	LED Blinking using Arduino UNO				
6.	Interface an IR Sensor and Servo Motor				
7.	Measure the Distance of an Object using the HC-SR04 Ultrasonic Sensor				
8.	RGB LED push button colour change				
9.	Multiple tones with one Piezo Buzzer				
10.	Design a 3D model of a Spanner using Tinkercad				

Exp.No:1

Date:

Parallel and Series Circuits

Aim: To make your own parallel and series circuits using breadboard for any application of your choice activity in Tinkercad.

Apparatus Required:

1. Tinkercad (Online Software)
2. LED
3. 9 V batteries/ TRPS
4. Breadboard
5. Jumper wires
6. Resistors
7. Push button

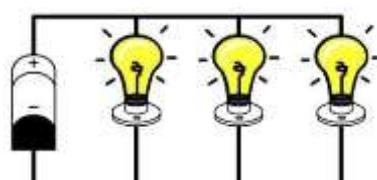
Theory:

About Tinkercad

Tinkercad Circuits is online software which is the easiest way to get students started with learning electronics. Using an interactive circuit editor, students can explore, connect, and code virtual projects with simulated components. Tinkercad helps students facilitate their learning virtually to understand and breakdown complex concepts.

Parallel Circuits:

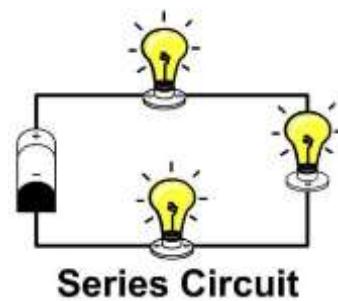
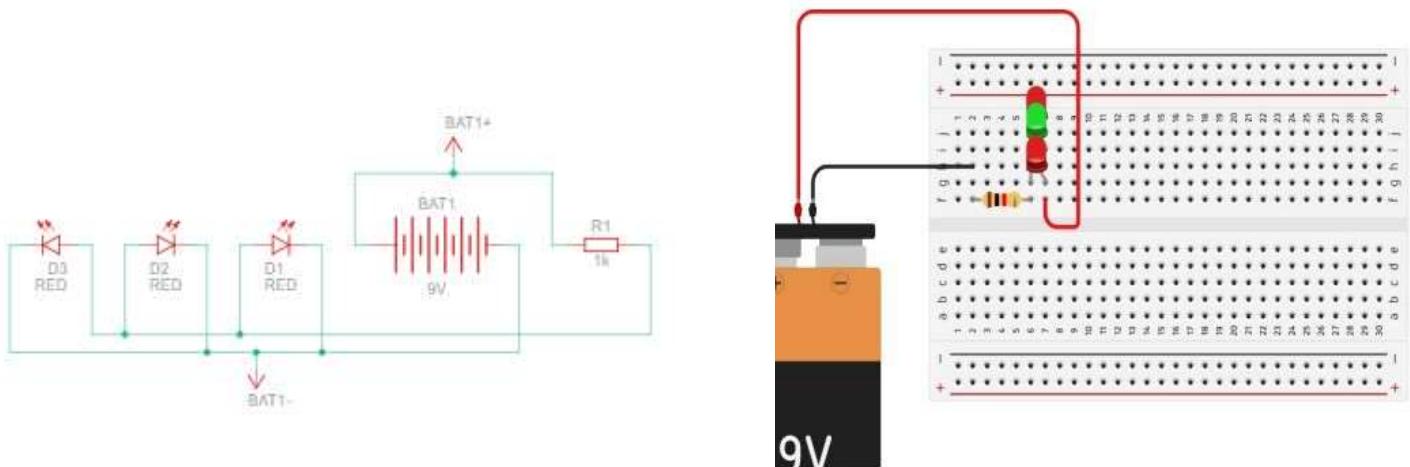
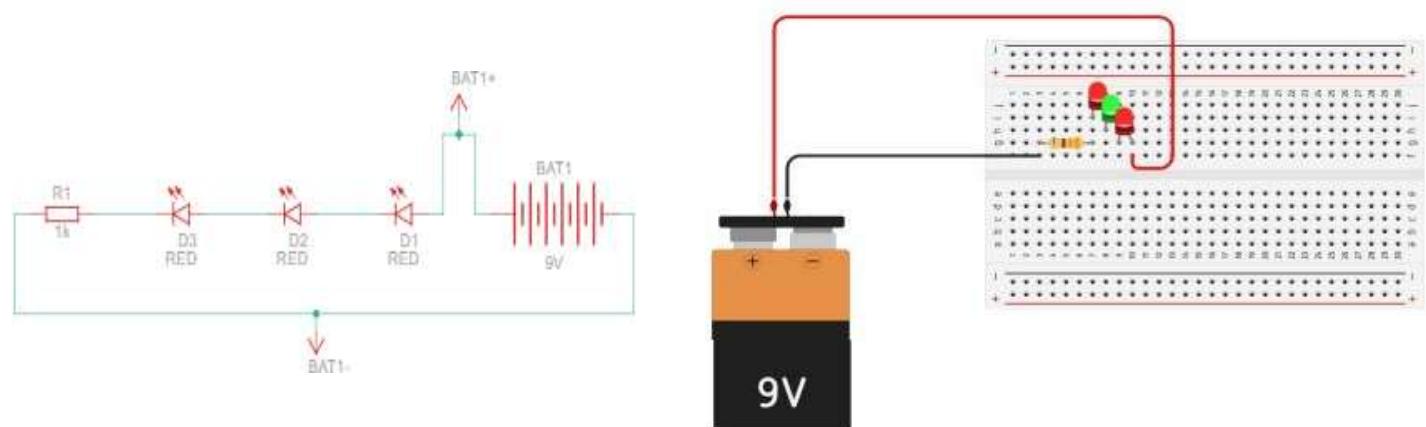
A circuit is said to be parallel when the electric current has multiple paths to flow through. The components that are a part of the parallel circuits will have a constant voltage across all ends.



Parallel Circuit

Series Circuits:

A circuit is said to be connected in series when the same current flows through all the components in the circuit. In such circuits, the current has only one path. This is nothing but a series of multiple tiny bulbs connected in series. If one bulb fuses, all the bulbs in the series do not light up.

**Circuit Diagram:****Fig (i): Parallel circuit****Fig (ii): Series Circuit**

Procedure:

Parallel circuit:

1. Visit the Tinkercad website (<https://tinkercad.com>) and create an account or log into an existing one. Then select Circuits on the left side of the screen.
2. Click on “Create new Circuit”.
3. Insert LED bulbs into the breadboard as shown in the fig (i).
4. Insert the Resistor into the breadboard and connect it to the negative terminal of LED.
5. Connect the battery red wire (+) to the positive side of the LED and connect black wire (-) to the other end of the resistor.

Series Circuit:

1. Visit the Tinkercad website (<https://tinkercad.com>) and create an account or log into an existing one. Then select Circuits on the left side of the screen.
2. Click on “Create new Circuit”.
3. Insert LED bulbs into the breadboard as shown in the fig (ii).
4. Insert the Resistor into the breadboard and connect it to the negative terminal of LED.
5. Connect the battery red wire (+) to the positive side of the LED and connect black wire (-) to the other end of the resistor.

Result:

Traffic Light Circuit Simulation in Tinkercad

Aim: To demonstrate a traffic light circuit using breadboard activity in Tinkercad.

Apparatus required:

1. Tinkercad (Online Software)
2. LED's - 3 (Red, Yellow, Green)
3. 9 V batteries/TRPS
4. Breadboard
5. Jumper wires
6. Multimeter
7. Resistors
8. Push button-03

Theory:

A **switch** is a component that controls whether an electrical circuit is **open** or **closed**. It allows control over the flow of current in a circuit **without** the need to manually cut or splice the wires. Switches are **essential components** in any circuit that requires user interaction or control.

A switch can exist in only one of two states: **open** or **closed**.

- In the **off** state, a switch creates an open gap in the circuit, effectively forming an **open circuit**, which prevents current from flowing.
- In the **on** state, the switch behaves like a perfectly conducting wire, effectively **closing the circuit**, turning the system "on," and allowing current to flow freely through the rest of the system.

Circuit diagram:

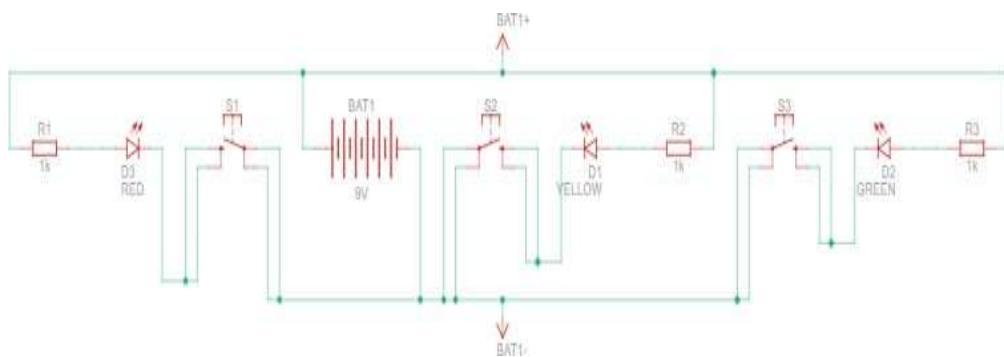


Fig.1 Circuit Diagram for Traffic Light Control

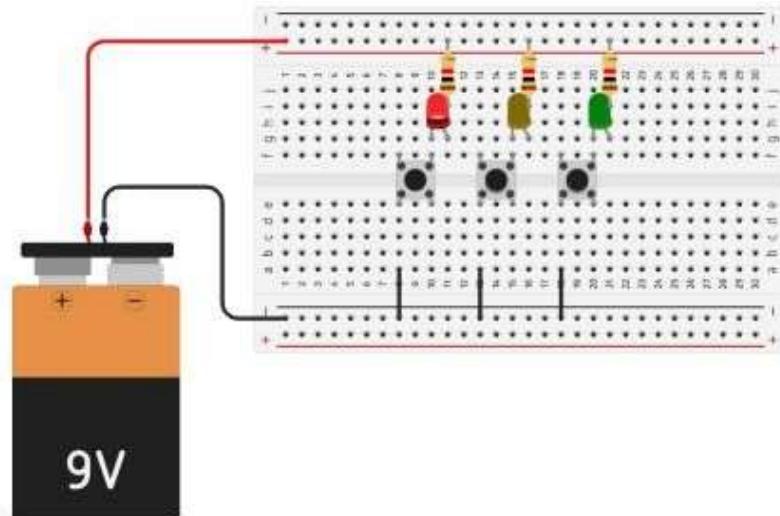


Fig.2 Tinkercad Connection Diagram

Procedure:

1. Insert LEDs into the breadboard.
2. Insert the Resistor into the breadboard and connect it to the positive terminal of LED.
3. Add the push button and connect it to the negative side of LED like as shown in the picture.
4. Connect the 9v battery to the circuit.
5. Connect the pushbutton and neutral of the battery by using the connecting wires.
6. To turn on red light, press the press the push button that is connected to the Red LED.
7. To turn on yellow light, press the push button that is connected to the Yellow LED
8. To turn on green light, press the push button that connected to the Green LED

Result:

Exp.No:3**Date:**

Automatic Street Light Using LDR

Aim: To Build and demonstrate automatic Street Light using LDR activity in Tinkercad.

Apparatus required:

1. Tinkercad (Online Software)
2. Resistor-47KΩ, 330Ω
3. Transistor-bc547
4. Battery-9V/TRPS
5. LED
6. LDR-Light Dependent Resistor

Theory:

When sufficient light falling on LDR the resistance of LDR is very low, as a result, all current is flowing from resistor R2 and LDR so LED D1 is not glowing. When there is darkness, and no light is falling on LDR, so the LDR resistance became very high so, current will flow to the base of transistor Q1 BC547, so transistor became turn on and LED D1 to glow.

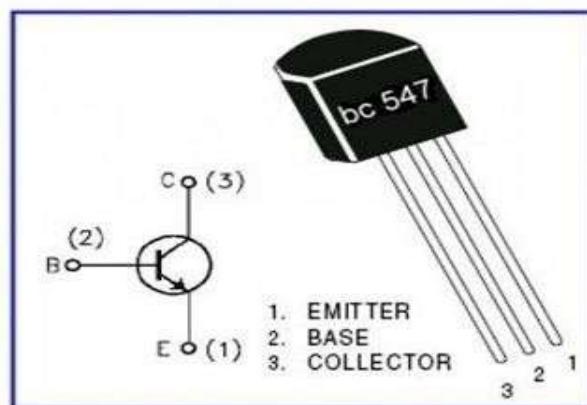


Fig a. Transistor BC547

Circuit Diagram

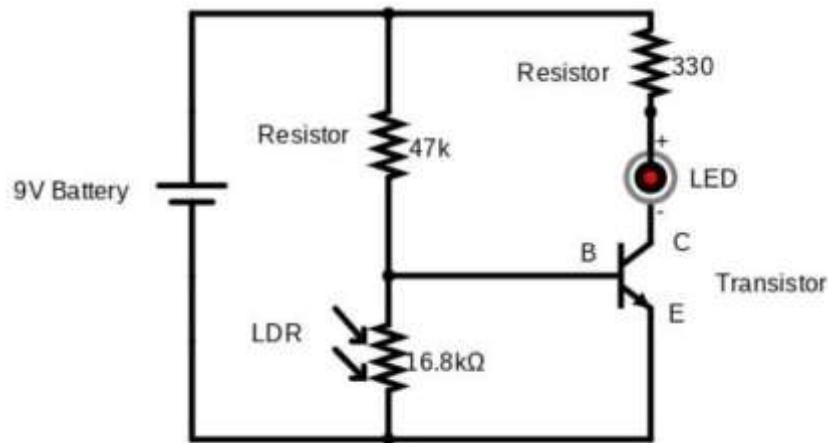


Fig.1 Circuit Diagram for Automatic Street Light using LDR

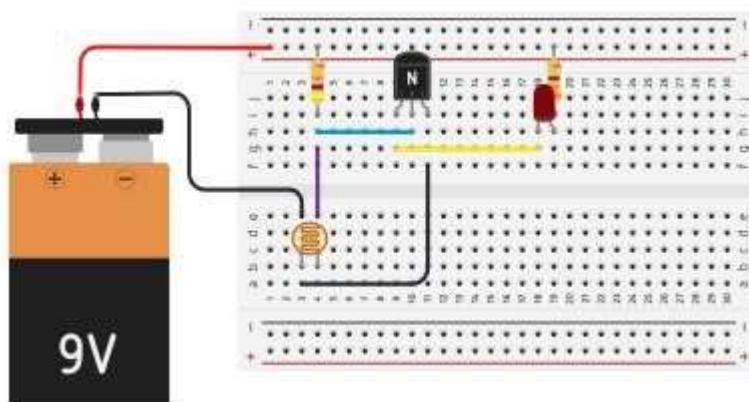


Fig.2 Tinkercad Connection Diagram

Procedure:

1. Identify the terminals of transistor as shown in the fig.2.
2. Connect positive and negative terminal of the battery as shown in the fig.1.
3. Place the LED, Resistors, Transistor and LDR in breadboard.
4. Connect each components carefully as per the circuit fig.1.
5. Check the working of circuit by placing the circuit in a dark area.

Result:

Exp.No:4 **Date:**

Simulation of LED Blinking using Arduino on Tinkercad

Aim : To Simulate the Arduino LED blinking activity in Tinkercad.

Apparatus Required:

1. Tinkercad (Online Software)
2. Arduino Uno Board
3. LED
4. 220 OHM Resistor
5. Breadboard
6. Jumper Wires(Male-Male) - 12 Nos

Theory:

About Tinkercad

Tinkercad Circuits is online software which is the easiest way to get students started with learning electronics. Using an interactive circuit editor, students can explore, connect, and code virtual projects with simulated components. Tinkercad helps students facilitate their learning virtually to understand and breakdown complex concepts.

About Arduino:

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects, You can connect various sensors to their input pin and get the output accordingly.

The Arduino hardware and software were designed for artists, designers, hobbyists, hackers, newbies, and anyone interested in creating interactive objects or environments. Arduino can interact with buttons, LEDs, motors, speakers, GPS units, cameras, the internet, and even your smartphone or your TV!

Arduino's boards are available in different sizes, form factors, and different no. of I/O pins etc. Some commonly known and frequently used Arduino boards are Arduino UNO, Arduino Mega, Arduino Nano, Arduino Micro, and Arduino Lilypad.

The Uno is one of the more popular boards in the Arduino family and a great choice for beginners. Let's understand how to use Arduino using a small activity.

Arduino Uno

Arduino UNO is a basic and inexpensive Arduino board and is the most popular of all the Arduino boards. Arduino UNO is considered to be the best prototyping board for beginners in electronics and coding. Arduino Uno is a microcontroller board based on

an 8-bit ATmega328P microcontroller. Along with ATmega328P, it consists of other components such as a crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller. Arduino Uno has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6 analog input pins, a USB connection, A Power barrel jack, an ICSP header, and a reset button.

Circuit Diagram:

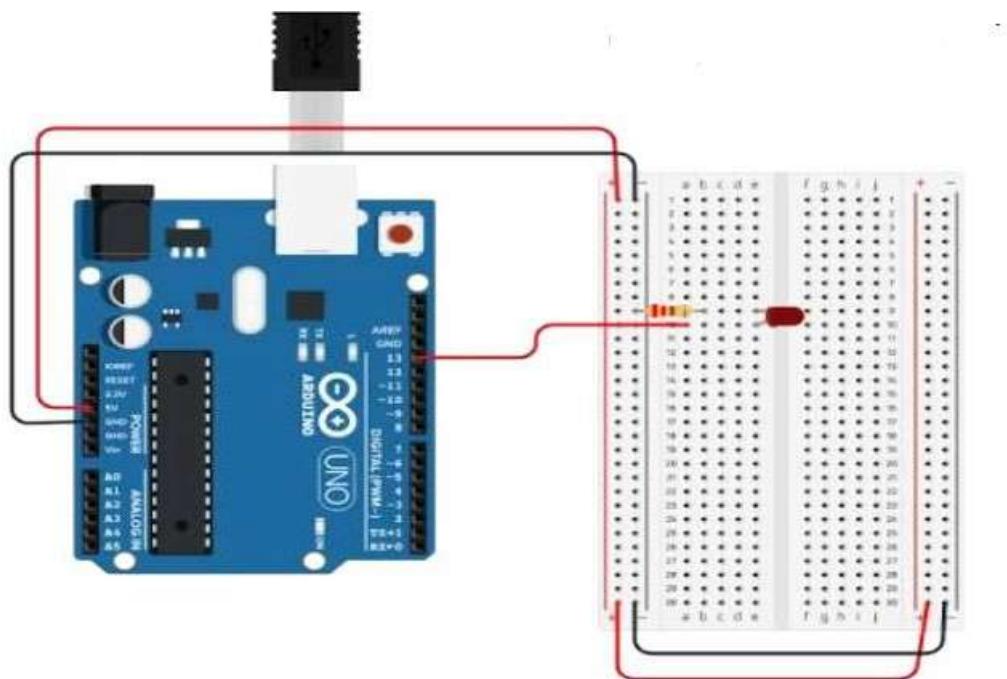


Fig: LED Blinking Circuit using Tinkercad

Procedure:

1. Visit the Tinkercad website (<https://tinkercad.com>) and create an account or log into an existing one. Then select Circuits on the left side of the screen.
2. Click on “Create new Circuit”.
3. Add the Breadboard and Arduino Uno and Connect +ve and -ve wires to the breadboard.
4. Add an LED and resistor from component list.
5. Add a wire from Arduino Pin 13 to the Anode of LED.
6. Open the Coding Window and write the Code.
7. Start the Simulation and Check for error, if error shows overcome those errors and re-run the simulation button.

Arduino Program Code:

```
void setup()
{
pinMode(13, OUTPUT);
}

void loop()
{
digitalWrite(13, HIGH);
delay(1000); // Wait for 1000 millisecond(s)
digitalWrite(13, LOW);
delay(1000); // Wait for 1000 millisecond(s)
}
```

Result:

Exp.No:5**Date:**

LED Blinking using Arduino UNO

Aim: To Build and demonstrate an Arduino LED blinking activity using Arduino IDE.

Apparatus required:

1. Arduino IDE (Online Software)
2. Arduino Uno Board
3. LEDs
4. 220 OHM Resistor
5. Breadboard
6. Jumper Wires (Male-Male) - 12 Nos

Theory:

Arduino IDE is an open source software for programming Arduino boards, We already tried arduino simulation in Tinkercad now we are going to do the same activities using physical arduino board, we need to download and install Arduino IDE 2,0 software for programming arduino .

The Arduino IDE 2.0

The Arduino IDE 2.0 is an open-source project. It is a big step from its sturdy predecessor, Arduino IDE 1.x, and comes with revamped UI, improved board & library manager, debugger, autocompletes feature, and much more.

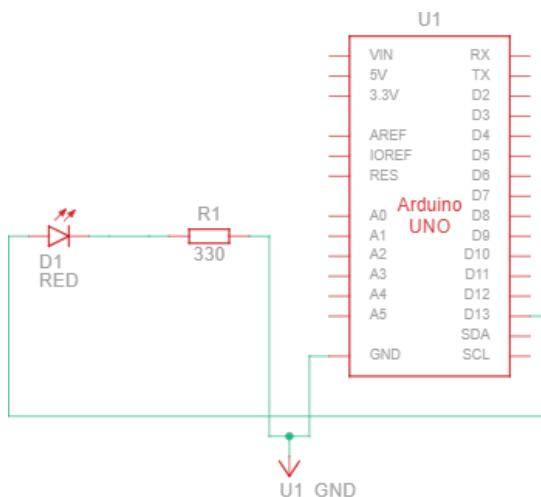
Circuit Diagram:

Fig.1 Circuit Diagram for Arduino LED blinking.

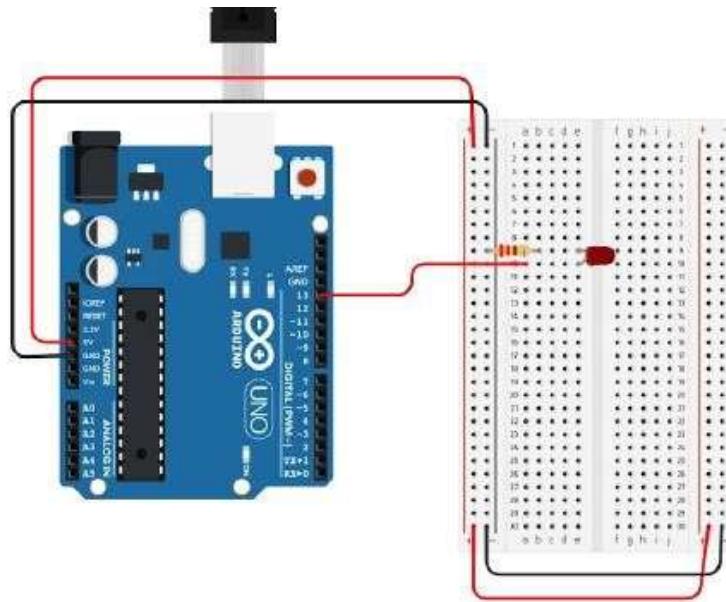


Fig 2. LED Blinking Circuit using Arduino Uno

Procedure:

1. Download & Install Arduino IDE 2.0 on your computer.
2. Connect circuit for a single LED blinking as per the circuit diagram.
 - Positive (Long Leg) of LED Connected to Pin no 13 of Arduino.
 - Negative (Short Leg) of LED to GND via resistor.
 Open Arduino software and write the code for LED Blinking.
3. Connect Arduino board to the computer USB via Arduino USB cable.
4. Select Board, Go to Tools>Boards>Arduino AVR Boards>Arduino Uno.
5. Select Port, Go to Tools and select the COM Port (COM Port number will vary in my case it is COM3).
6. Upload the code by clicking the Upload button in Arduino and wait for a minute, you will get a notification when the upload is completed.
7. Check the Working, LED should blink in a 1 second interval.

Arduino Program Code:

```

int ledPin = 13;// LED connected to digital pin 13
void setup() {
  pinMode(ledPin,OUTPUT);
}
void loop()
{
  
```

```
digitalWrite(ledPin,HIGH); // set the LED on  
delay(1000); // wait for a second  
digitalWrite(ledPin,LOW); // set the LED off  
delay(1000); // wait for a second  
}
```

Result:

Exp.No:6**Date:**

Interface an IR Sensor and Servo Motor

Aim: To Interface an IR Sensor and Servo Motor with Arduino.

Apparatus required:

1. Arduino Uno board
2. IR sensor
3. Servo motor
4. Jumper wires
5. Bread board

Theory:

An **IR (Infrared) Sensor** is used to detect the presence of an object or obstacle without physical contact. It works by emitting infrared light from a transmitter and detecting the reflected light using a photodiode. If an object is detected (i.e., the IR light is reflected back), the sensor outputs a digital signal (typically LOW or HIGH depending on the sensor logic).

A **Servo Motor** is a type of actuator that can rotate to a specific angle, typically between 0° and 180°. It is controlled by sending a **PWM (Pulse Width Modulation)** signal to its control pin. The width of the pulse determines the angular position of the shaft.

By combining an IR sensor with a servo motor using an **Arduino UNO**, you can create an automated motion or response system. For example, the servo can rotate when the sensor detects an object—this principle is often used in obstacle-avoiding robots, automatic gates, and smart dispensers.

Procedure:

1. Connect the circuit as per shown in fig.

First, connect the Servo to Uno.

- Connect the servo's brown wire to Uno's ground pin using a black male-to-male jumper cable.
- Next, connect the red wire to Uno's VIN pin using a red male-to-male jumper cable.
- Then, connect the orange wire to Uno's PWM pin 5 using a yellow male-to-male jumper cable.

2. Now, let's connect the IR sensor to the Arduino Uno.

- Connect the sensor's VCC pin to Uno's 5V pin using a red male-to-female jumper cable.

- Next, connect the sensor's ground pin to Uno's ground pin using a black male-to-female jumper cable.
 - Then, connect the sensor's out pin to Uno's digital pin 2 using a green male-to-female jumper cable. IR + Uno + Servo.
3. Connect Uno to the computer and Upload the code using Arduino IDE. Check the working of the program and try to move servo to 45 degrees.

Circuit Diagram:

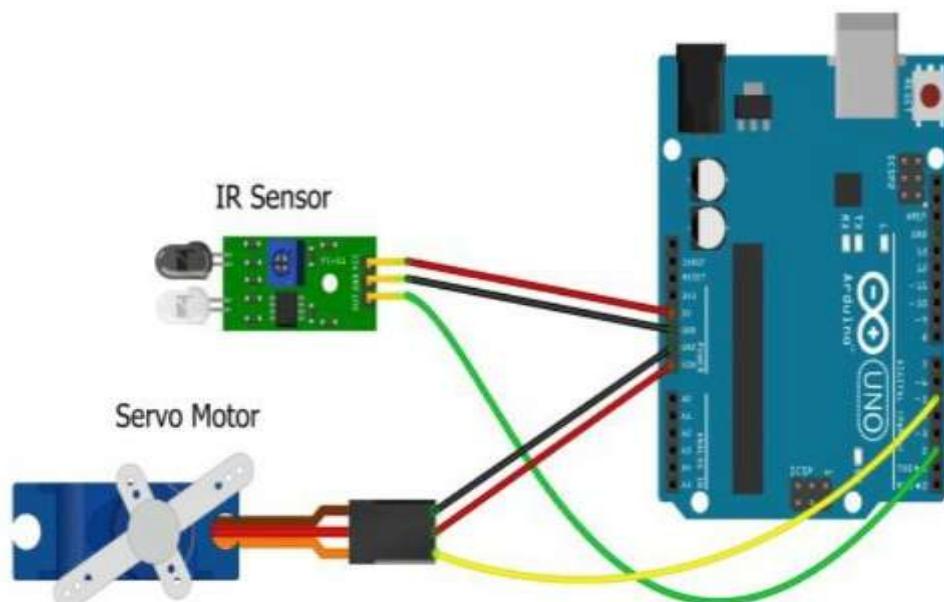


Fig : Interface an IR Sensor and Servo Motor with Arduino Uno

Arduino Program Code:

```
/*
If Sensor detects any object Servo Motor will rotate 90 Degrees
*/
#include <Servo.h>
Servo Servopin;
int sensorpin=2;
int val=0;
void setup(){
  Servopin.attach(3); //Sevo signal pin connected to pin 3
```

```
}

void loop(){
    val = digitalRead(sensorpin); //Reading value from sensor
    if (val ==0){
        Servopin.write(90);//command for rotating 90 Degree
        delay(10);
    }
    else
    {
        Servopin.write(0);
        delay(10);
    }
}
```

Result:

Exp.No:7**Date:**

Measure the Distance of an Object using the HC-SR04 Ultrasonic Sensor

Aim: To measure the distance of an object using the HC-SR04 ultrasonic sensor and Arduino and display the distance on the Serial Monitor.

Components Required:

- 1 x Arduino UNO
- 1 x USB Cable
- 1 x Ultrasonic Sensor (HC-SR04)
- 1 x Breadboard
- Jumper Wires (Male-to-Male)

Theory:

The HC-SR04 ultrasonic sensor uses sonic waves to measure the distance to an object. It emits a sound pulse from the Trig pin and waits for it to reflect back to the Echo pin. The time taken for the echo to return is used to calculate the distance.
 $\text{Distance (in cm)} = (\text{Time (in } \mu\text{s}) \times 0.0343) / 2$

Circuit Diagram:

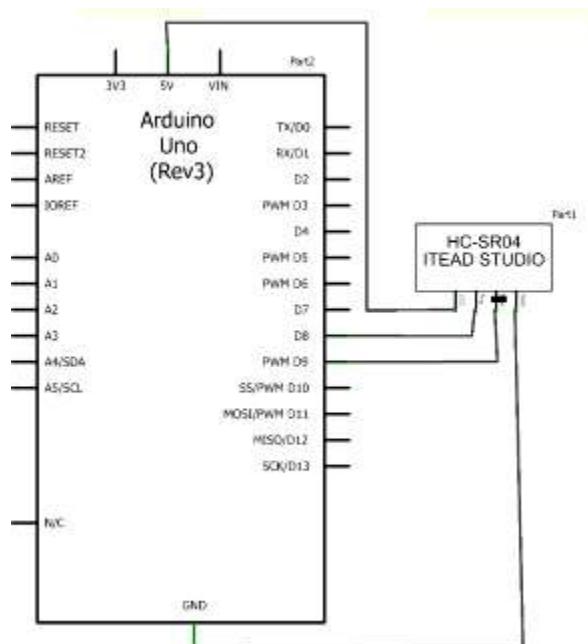


Fig.1 Circuit Diagram for Distance Measurement using Arduino and HC-SR04

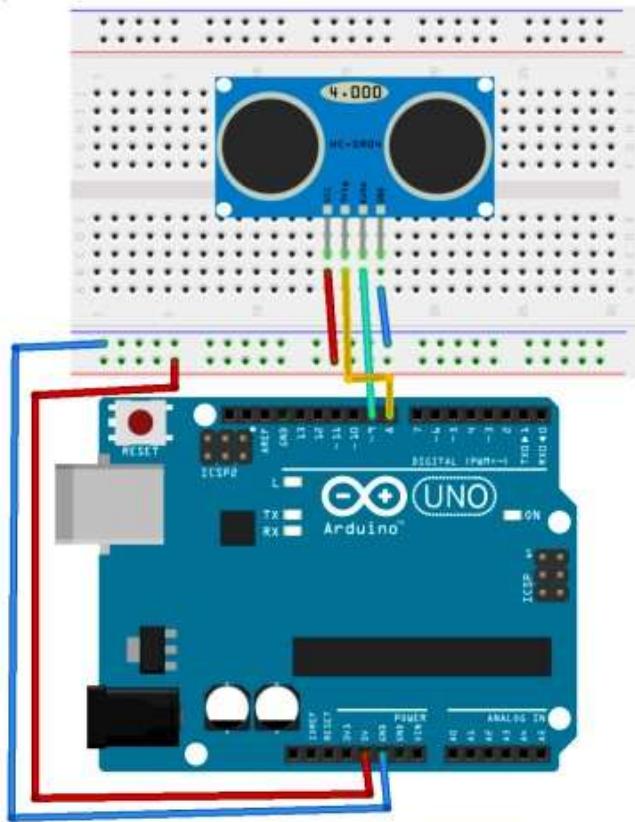


Fig 2. HC-SR04 Distance Measurement Circuit with Arduino UNO

Connections:

HC-SR04 Pin	Arduino Pin
VCC	5V
GND	GND
Trig	Pin 8
Echo	Pin 9

Procedure:

1. Connect the HC-SR04 sensor to the Arduino board as per the circuit table.
2. Upload the code to the Arduino using Arduino IDE.
3. Open the Serial Monitor (Ctrl + Shift + M) and set baud rate to 9600.
4. Place an object in front of the sensor.
5. Observe the distance displayed on the Serial Monitor in mm.

Observations:

S.No	Object	Approx. Distance (mm)
1	Hand	
2	Book	
3	Wall	

Arduino Code:

```
#include <HCSR04.h>

const int TriggerPin = 8; //Trig pin
const int EchoPin = 9; //Echo pin
long Duration = 0;

void setup() {
    pinMode(TriggerPin,OUTPUT); // Trigger is an output pin
    pinMode(EchoPin,INPUT); // Echo is an input pin
    Serial.begin(9600); // Serial Output
}

void loop() {
    digitalWrite(TriggerPin, LOW);
    delayMicroseconds(2);
    digitalWrite(TriggerPin, HIGH); // Trigger pin to HIGH
    delayMicroseconds(10); // 10us high
    digitalWrite(TriggerPin, LOW); // Trigger pin to HIGH
    Duration = pulseIn(EchoPin,HIGH); // Waits for the echo pin to get high
    long Distance_mm = Distance(Duration); // Use function to calculate the distance
    Serial.print("Distance = "); // Output to serial
    Serial.print(Distance_mm);
    Serial.println(" mm");
    delay(1000); // Wait to do next measurement
}
```

```
long Distance(long time)
{
    long DistanceCalc; // Calculation variable
    DistanceCalc = ((time /2.9) / 2); // Actual calculation in mm
    //DistanceCalc = time / 74 / 2; // Actual calculation in inches
    return DistanceCalc; // return calculated value
}
```

Result:

Exp.No:8

Date:

RGB LED push button colour change

Aim:

To interface an RGB LED with Arduino UNO and program it to change colour using a push button.

Components Required:

- 1 x Arduino UNO
- 1 x USB Cable
- 1 x RGB LED (Common cathode or common anode)
- 3 x 220Ω Resistors
- 1 x Push Button
- 1 x Breadboard
- Several Jumper Wires

Theory:

RGB LEDs consist of three LEDs: red, green and blue. These three coloured LEDs are capable of producing any colour. Tri-colour LEDs with red, green, and blue emitters, in general using a four wire connection with one common lead (anode or cathode).

What we use in this experiment is a common anode RGB LED. The longest pin is the common anode of the three LEDs. The pin is connected to the +5V pin of the Arduino, and the rest pins are connected to pin D9, D10, and D11 of the Arduino with a current limiting resistor between them and the push button.

Procedure:

1. Connect the R, G, and B pins of the RGB LED to Arduino digital pins (e.g., 9, 10, 11) via 220Ω resistors.
2. Connect the cathode (or anode) to GND (or VCC depending on LED type).
3. Connect the push button between a digital pin (e.g., pin 2) and GND, with an internal pull-up enabled.
4. Write the Arduino code to change the LED colour each time the button is pressed.
5. Upload the code and observe the RGB LED changing colour on button presses.

Circuit Diagram:

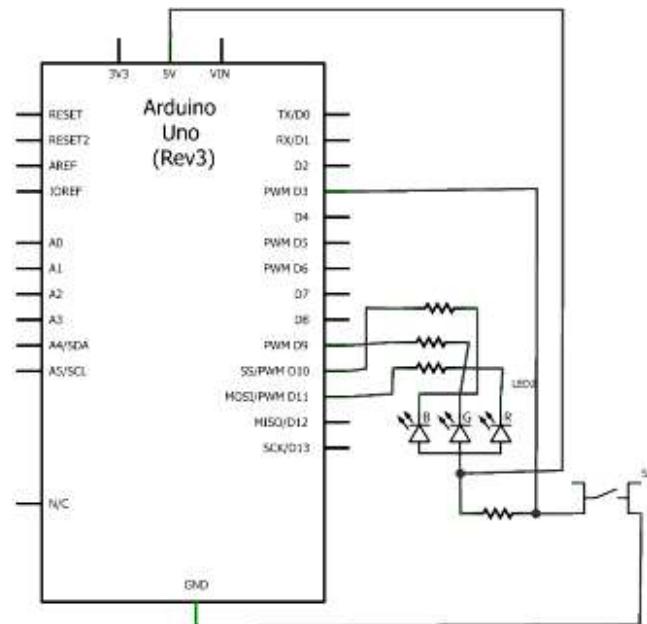


Fig.1 Circuit Diagram for Interfacing an RGB LED with Arduino UNO

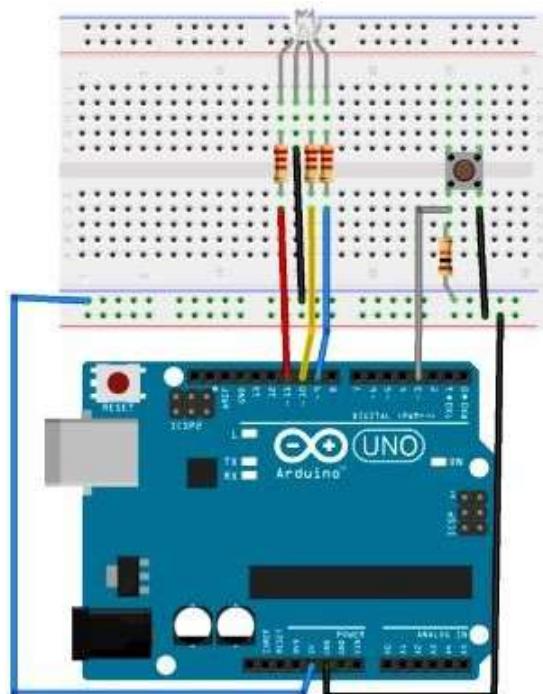


Fig 2. Connection Diagram for Interfacing an RGB LED with Arduino UNO

Arduino Code:

```
const int buttonPin = 3;
const int redPin = 11;
const int greenPin = 10;
const int bluePin = 9;
int counter = 0;
void setup()
{
    pinMode(buttonPin, INPUT);
    pinMode(redPin, OUTPUT);
    pinMode(greenPin, OUTPUT);
    pinMode(bluePin, OUTPUT);
}
void loop()
{
    int buttonState;
    buttonState = digitalRead(buttonPin);
    if (buttonState == LOW) // light the LED
    {
        counter++;
        delay(150);
    }
    if (counter == 0)
    {
        digitalWrite(redPin, LOW);
        digitalWrite(greenPin, LOW);
        digitalWrite(bluePin, LOW);
    }
    else if (counter == 1)
    {
        digitalWrite(redPin, HIGH);
        digitalWrite(greenPin, LOW);
        digitalWrite(bluePin, LOW);
    }
    else if (counter == 2)
    {
        digitalWrite(redPin, LOW);
        digitalWrite(greenPin, HIGH);
        digitalWrite(bluePin, LOW);
    }
    else if (counter == 3)
    {
```

```
digitalWrite(redPin, LOW);
digitalWrite(greenPin, LOW);
digitalWrite(bluePin, HIGH);
}
{
counter = 0;
}
}
```

Result:

Exp.No:9**Date:**

Multiple tones with one Piezo Buzzer

Aim: To generate multiple tones using a Piezo buzzer connected to an Arduino UNO by writing and uploading custom tone patterns.

Components Required:

- 1 x Arduino UNO
- 1 x USB Cable
- 1 x Piezo Buzzer
- 1 x Breadboard
- 2 x Jumper Wires

Theory:

The working principle of buzzer it to use PWM generating audio to make the air to vibrate. Appropriately changed as long as the vibration frequency, it can generate different sounds. For example, sending a pulse of 523Hz, it can generate Alto Do, pulse of 587Hz, it can generate midrange Re, pulse of 659Hz, it can produce midrange Mi. By the buzzer, you can play a song.

We should be careful not to use the UNO R3 board analog Write () function to generate a pulse to the buzzer, because the pulse output of analog Write () is fixed (500Hz).

Procedure:

1. Setup the Components:
 - Place the Piezo buzzer on the breadboard.
 - Connect the positive terminal of the buzzer to digital pin 8 of the Arduino UNO.
 - Connect the negative terminal of the buzzer to GND on the Arduino UNO using a jumper wire.
2. Write the Arduino Code:
 - Open the Arduino IDE on your computer.
 - Type following code
3. Upload the Code:
 - Connect the Arduino UNO to your computer via USB cable.
 - Select the correct board and port in the Arduino IDE.
 - Click the Upload button.
4. Observe the Output:
 - The buzzer will beep with a 1-second tone, followed by 1 second of silence, repeatedly.
 - You can modify the tone () frequency and delay () duration to produce different sound patterns.

Circuit Diagram:

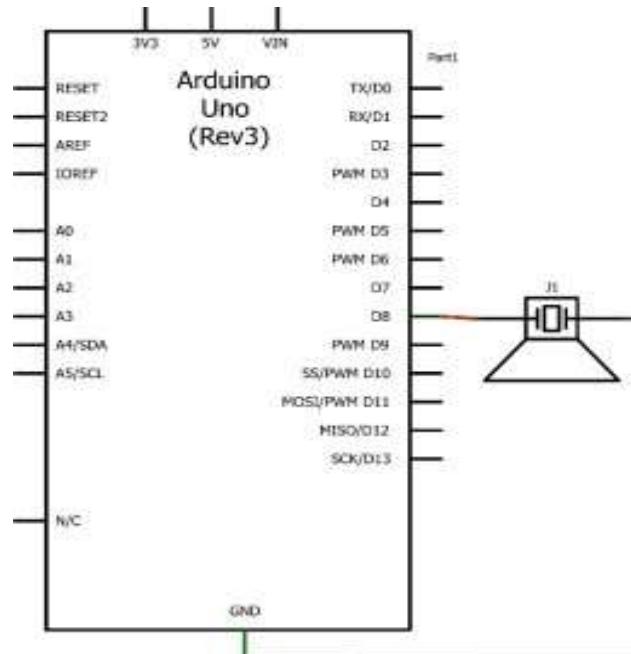


Fig.1 Circuit Diagram for Piezo Buzzer Connection with Arduino UNO

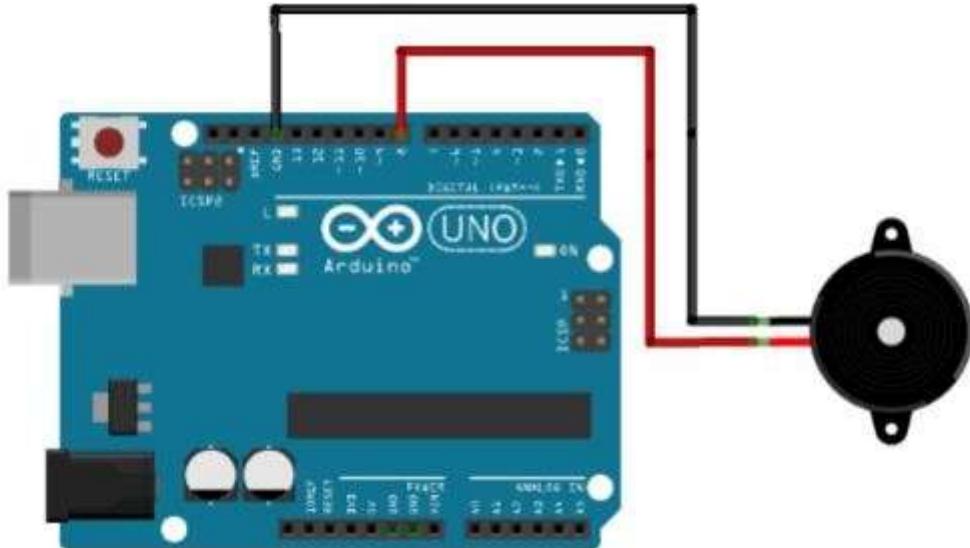


Fig.2 Connection Diagram for Piezo Buzzer Connection with Arduino UNO

Arduino Code:

```
const int buzzer= 8;  
void setup() {  
pinMode (buzzer,OUTPUT);  
}  
void loop() {  
tone(buzzer,1000);  
delay(1000);  
noTone(buzzer);  
delay(1000);  
}
```

Result:

Exp.No:10

Date:

Design a 3D model of a Spanner using Tinkercad

Aim: To design a 3D model of a spanner using Tinkercad software.

Components Required:

- 1) Laptop/desktop with internet connection.
- 2) Tinkercad account.

Theory:

A spanner is a mechanical tool used to grip and turn objects such as nuts and bolts. In mechanical systems, the spanner operates as a first-class lever, where effort is applied at one end, the load at the other, and the fulcrum lies in between.

Using 3D modelling software like Tinkercad, we can digitally create a spanner by breaking it down into basic shapes—ring, handle, and jaw—and assembling them. This design is then exported to a 3D printer to produce a functional prototype.

Tinkercad supports the principles of 'divide and conquer', meaning complex objects are designed as smaller geometric units and then grouped together. In this activity, we'll create the ring, handle, and jaw separately and merge them.

Procedure:

Step 1: Design the Ring

- Drag a cylinder shape into the workspace.
- Set its diameter to 25 mm and height to 4 mm.
- Drag another cylinder (to create a hole), set its diameter to 12.5 mm, and mark it as a hole.
- **Align** both cylinders and **Group** them to create the ring.

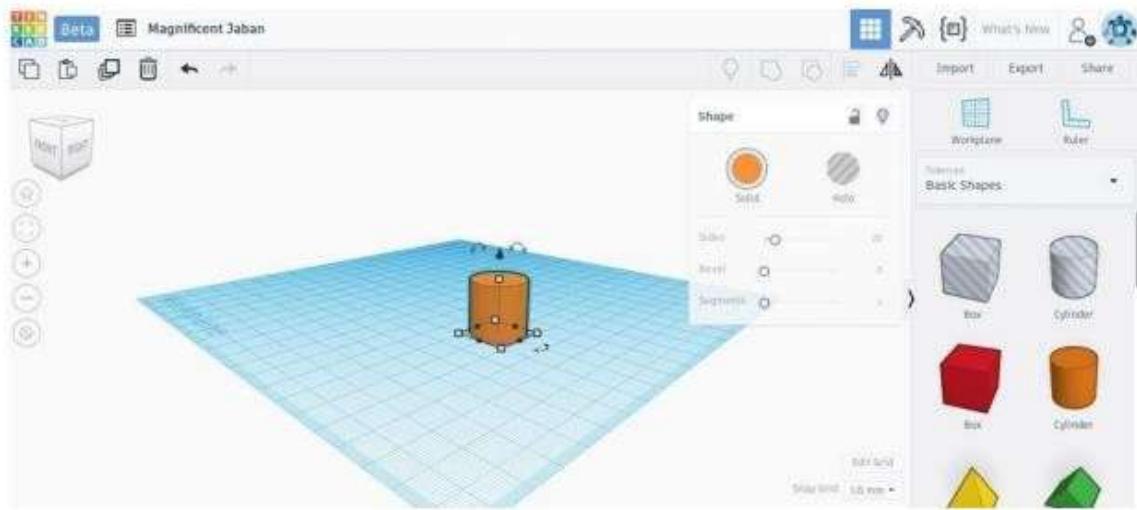
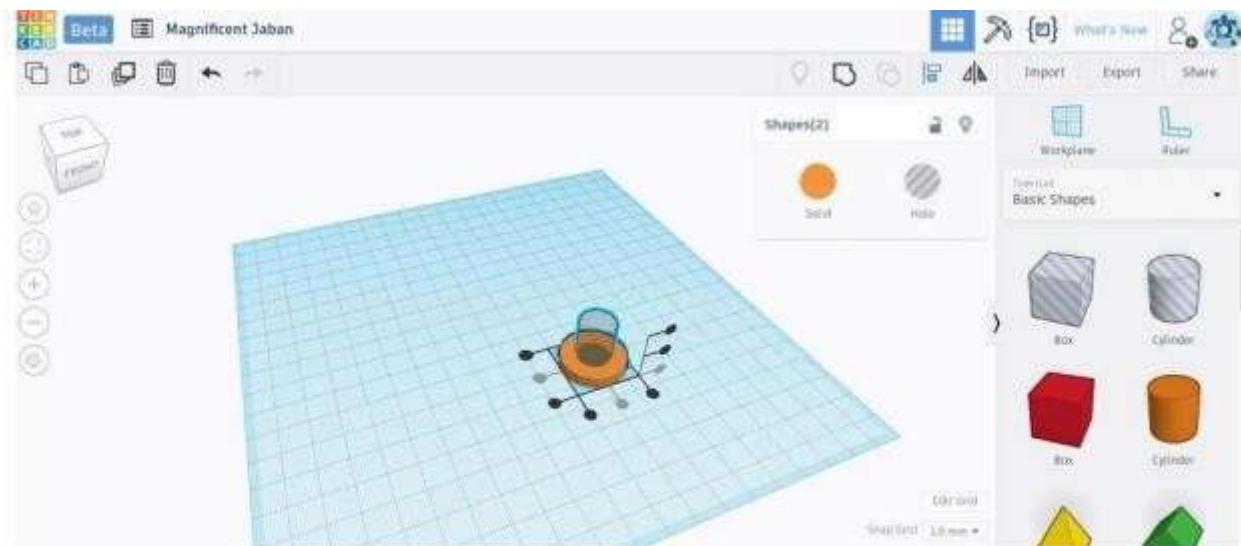


Fig.1 Drag a cylinder shape into the workspace



Note - The black alignment handles help align two shapes in Tinkercad.

Fig.2 Align both cylinders and Group them to create the ring.

Step 2: Design the Handle

- Drag a box into the workspace.
- Set length = 100 mm, width = 10 mm, height = 4 mm.
- Align and group it with the ring so it connects at the center.

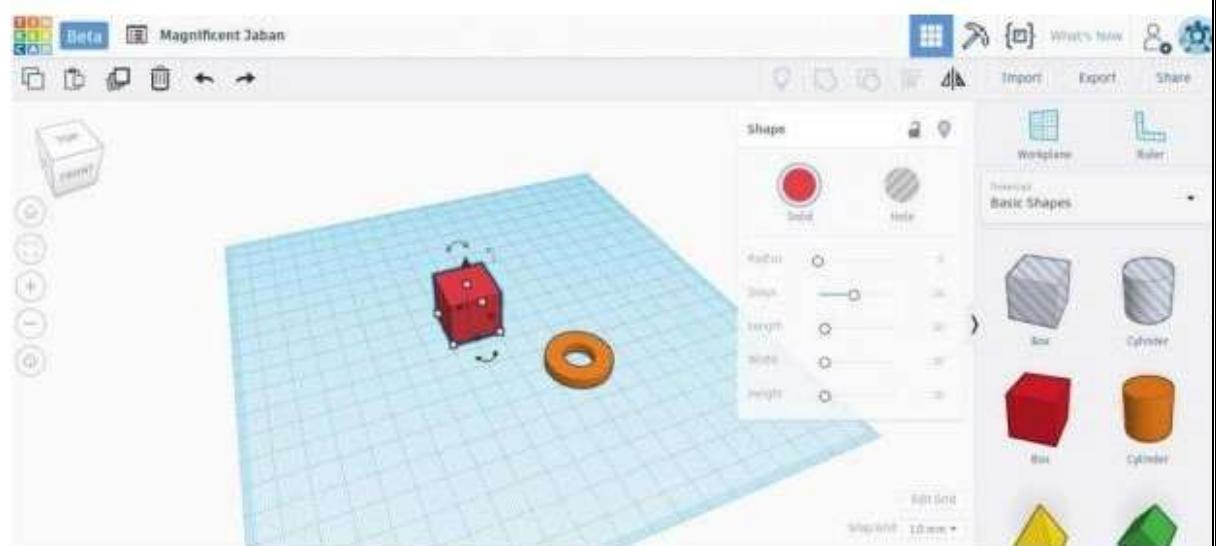


Fig.3 Drag a box into the workspace.

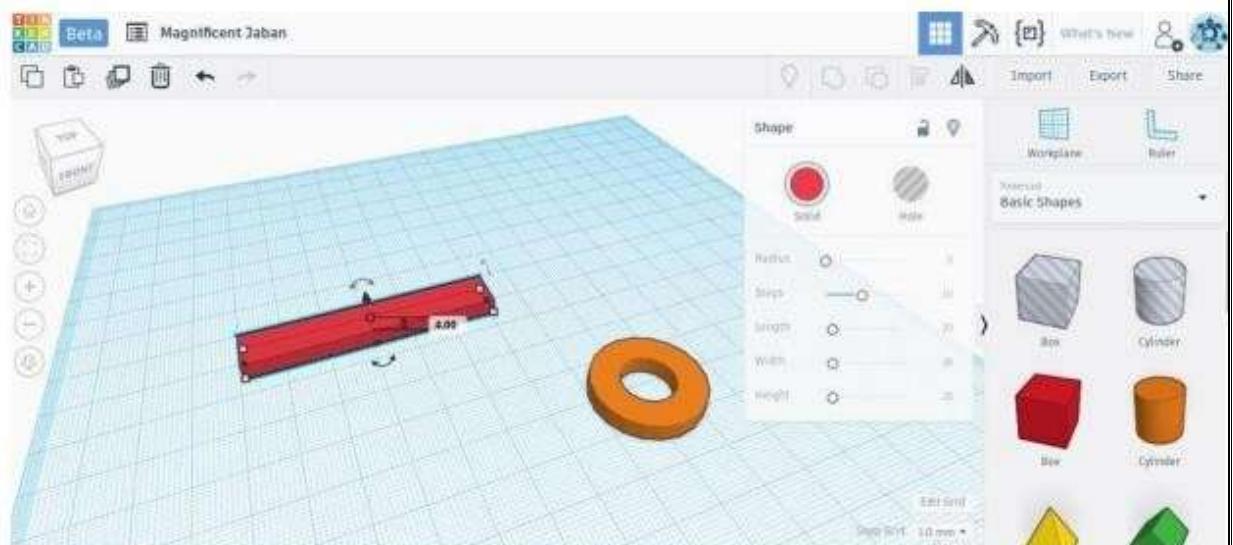


Fig.4 Handle is ready.

Step 3: Design the Jaw

- Create another cylinder for the jaw base: diameter = 30 mm, height = 4 mm.
- Use a polygon shape, set it as a hole, and adjust its width to 15 mm.
- Align and group with the jaw base.

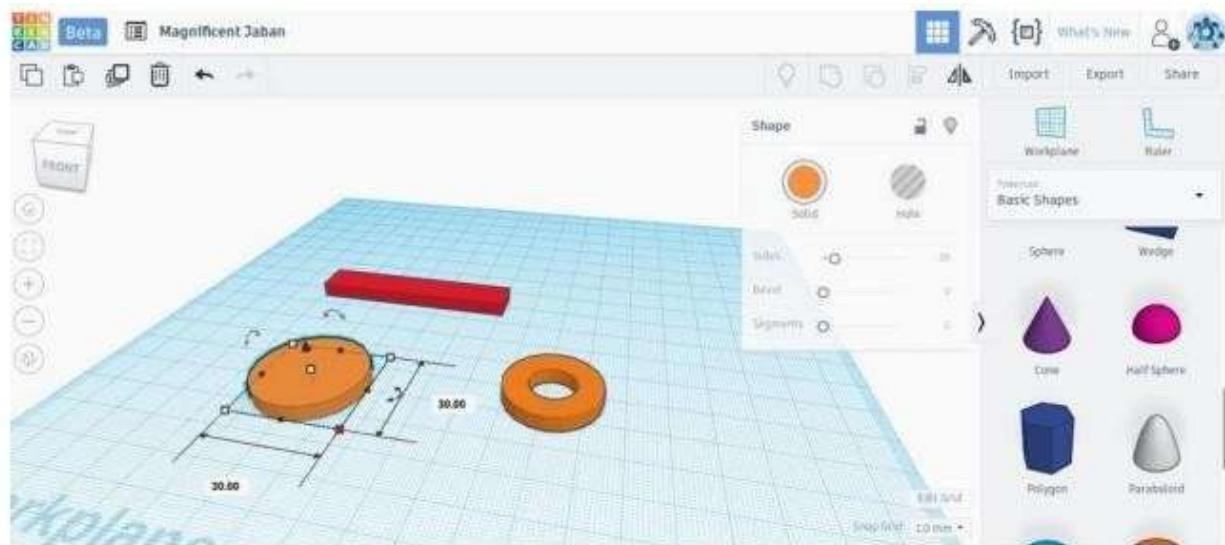


Fig.5 cylinder for the jaw base: diameter = 30 mm, height = 4 mm.

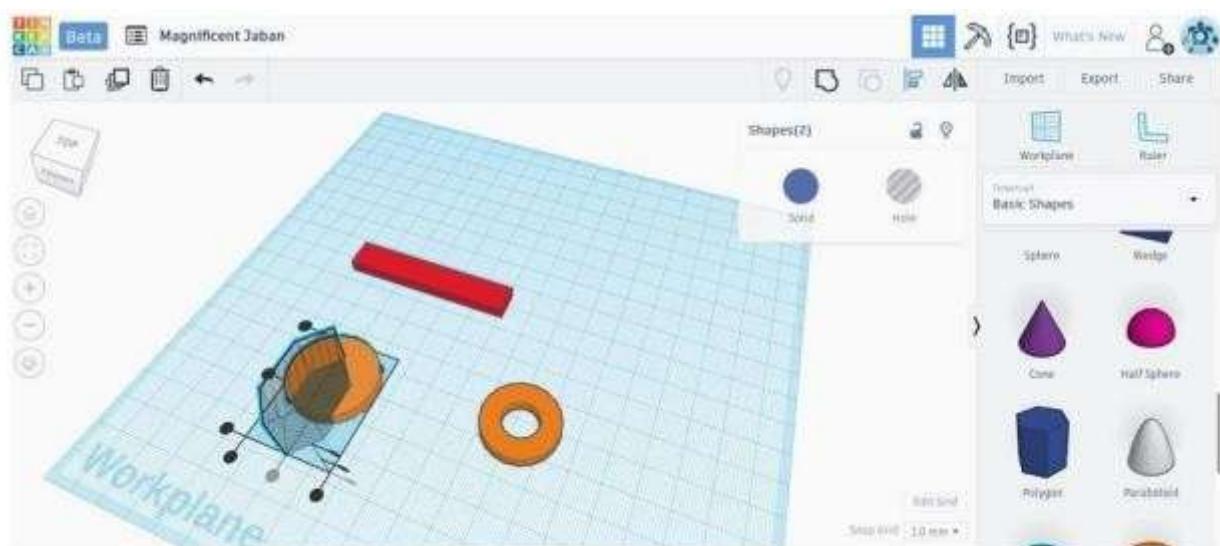


Fig.6 polygon shape, set it as a hole, and adjust its width to 15 mm.

Step 4: Assemble the Spanner

- Align the jaw to the end of the handle.
- Merge all three parts (ring, handle, jaw) using the Group tool.

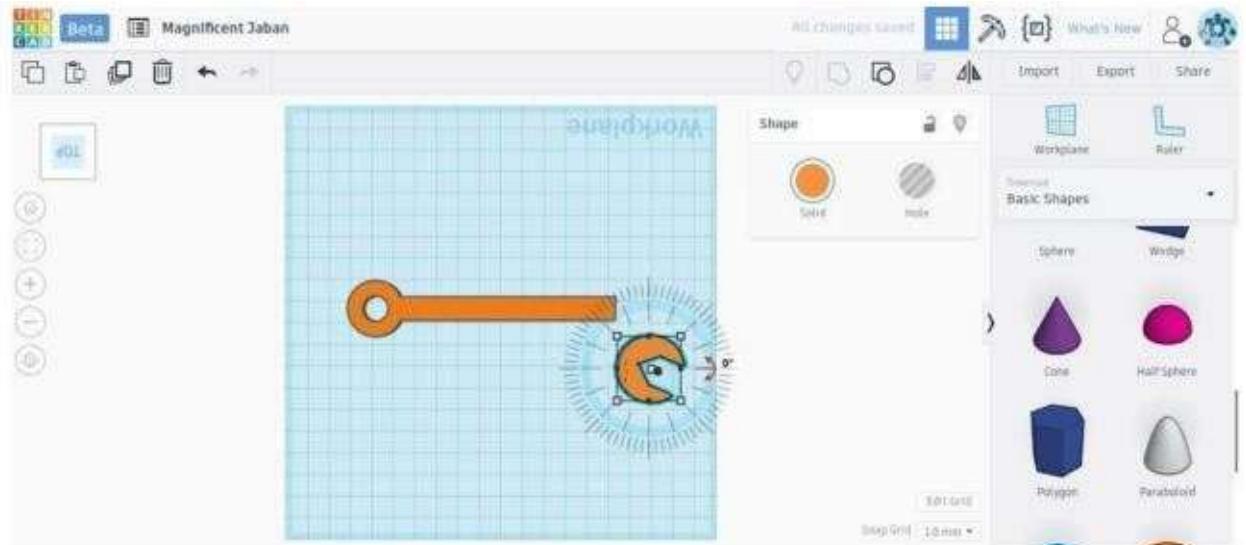


Fig.7 Align the jaw to the end of the handle

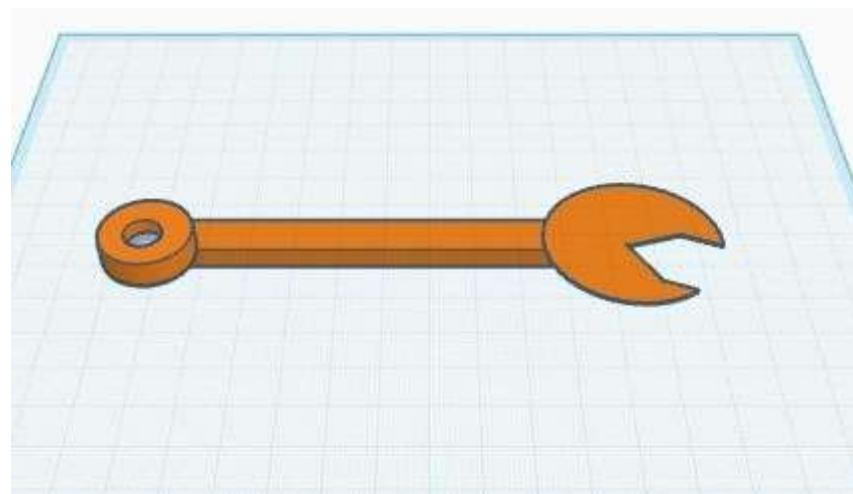


Fig.8 Merge all three parts (ring, handle, jaw) using the Group tool

Result: