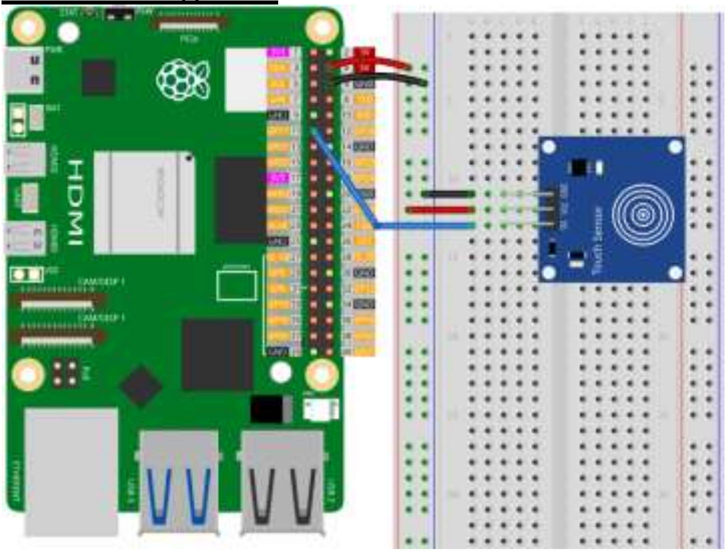
9. SOC Experiments using Raspberry PI or Ordroid Xu4: 02
   a. Touch sensor
   b. Tracking sensor

a. Touch sensor
   Components Required:
   Rasberry Pi
   Touch Sensor
   Breadboard
   Jumper Wires

Circuit Diagram:



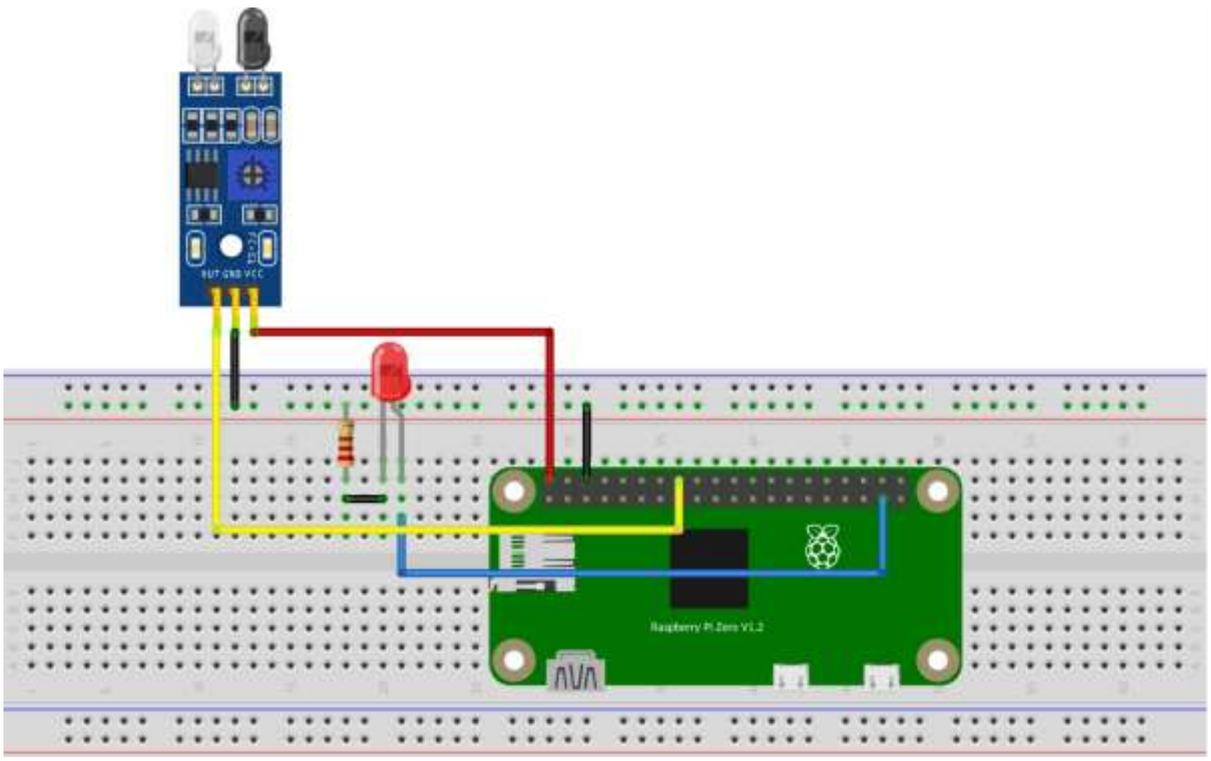
Program:

```
from gpiozero import Button
from signal import pause
# Function called when the sensor is touched
def touched():
# Print a message indicating the sensor is touched
print("Touched!")
# Function called when the sensor is not touched
def not_touched():
    # Print a message indicating the sensor is not touched
    print("Not touched!")
# Initialize a Button object for the touch sensor
# GPIO 17: pin connected to the sensor
# pull_up=None: disable internal pull-up/pull-down resistors
# active_state=True: high voltage is considered the active state
touch_sensor = Button(17, pull_up=None, active_state=True)
# Assign functions to sensor events
touch_sensor.when_pressed = touched
touch_sensor.when_released = not_touched
pause()  # Keep the program running to detect touch events
```

b. Tracking Sensor

Components Required:
Rasberry Pi
Tracking Sensor
LED
10 Ohm Resistor
Jumper Wires

Circuit Diagram:



Program:

```
import RPi.GPIO as GPIO
import time

# declare the sensor and led pin
sensor_pin = 23
led_pin = 26

# GPIO setup
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(sensor_pin, GPIO.IN)
GPIO.setup(led_pin, GPIO.OUT)

try:
    while True:
        if GPIO.input(sensor_pin):
            # If no object is near
            GPIO.output(led_pin, False)
            while GPIO.input(sensor_pin):
                time.sleep(0.2)
            else:
                # If an object is detected
                GPIO.output(led_pin, True)
        except KeyboardInterrupt:
            GPIO.cleanup()
```

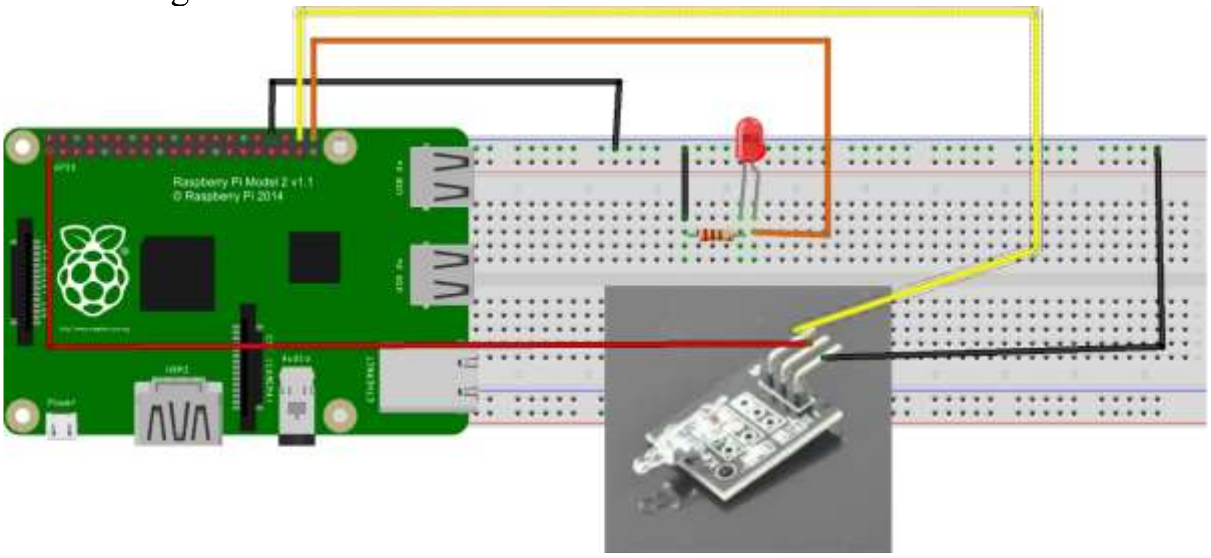
10. SOC Experiments using Raspberry PI or Ordroid Xu4: Control and communication Experiments

    a. Mercury tilt switch
    b. Laser emitter

a. Mercury tilt Switch

Components Required:
Rasberry Pi
Tilt Sensor
LED
10 Ohm resistor
Jumper Wires

Circuit Diagram:



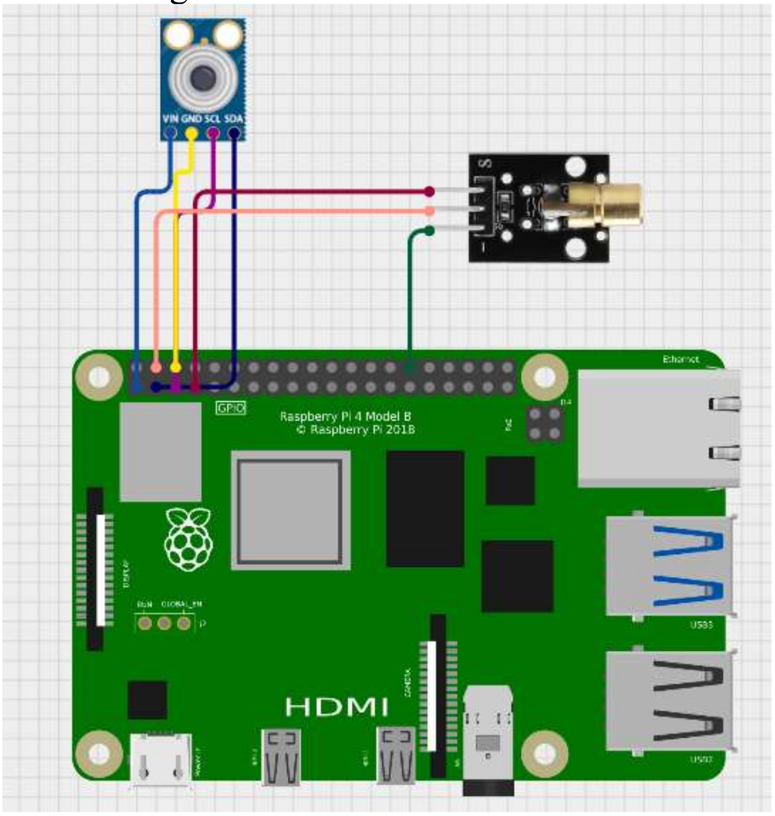
Program:

```
#Project tutorial URL http://osoyoo.com/?p=804
#Copyright Osoyoo.com

import RPi.GPIO as GPIO
import time

sensor_pin = 38
led_pin = 40

GPIO.setmode(GPIO.BOARD)

GPIO.setup(led_pin,GPIO.OUT)
GPIO.setup(sensor_pin, GPIO.IN)

current_state = 0

try:
    while True:
        time.sleep(0.1)
        current_state = GPIO.input(sensor_pin)
        if current_state == 1:
            print("tilt sensor value is %s" % (current_state))
            GPIO.output(led_pin,True)
        else:
            print("tilt sensor value is %s" % (current_state))
            GPIO.output(led_pin,False)

except KeyboardInterrupt:
    pass
finally:
    GPIO.cleanup()
```

b. Laser Emitter

Components Required:
Laser Emitter
Rasberry Pi
Jumper Wires
Breadboard

Circuit Diagram:

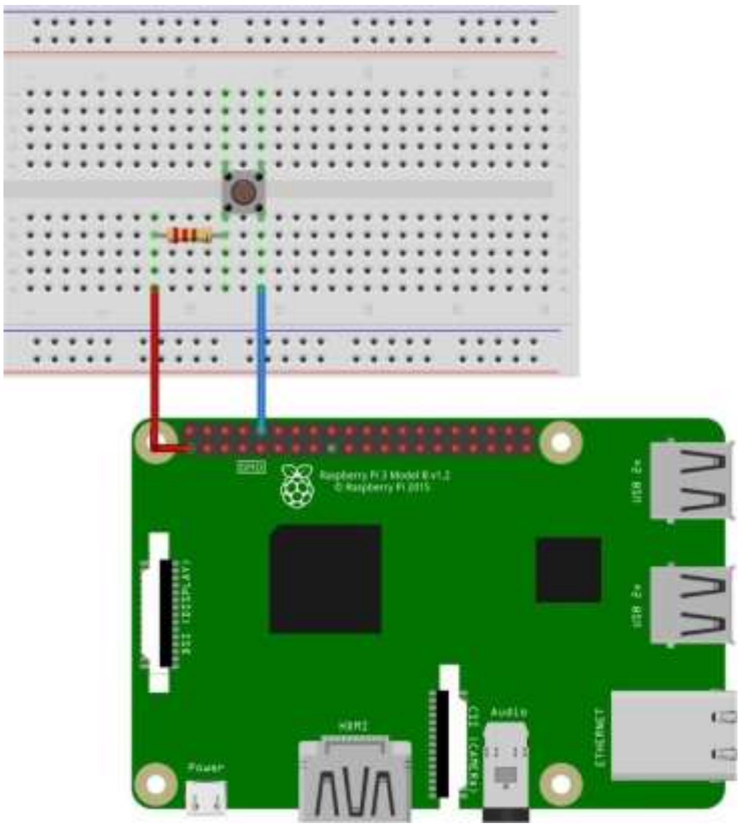


Program:

```
const int laserPin = 4; // GPIO4 on Raspberry Pi connected to SIG of KY-
008
const int delayTime = 1000; // Delay time in milliseconds

void setup() {
  pinMode(laserPin, OUTPUT); // Set laserPin as an OUTPUT
}

void loop() {
  digitalWrite(laserPin, HIGH); // Turn the laser on
  delay(delayTime); // Wait for a second
  digitalWrite(laserPin, LOW); // Turn the laser off
  delay(delayTime); // Wait for a second
}
```

11. SOC Experiments using Raspberry PI or Ordroid Xu4:
    a. Button
    b. IR emitter

a. Button

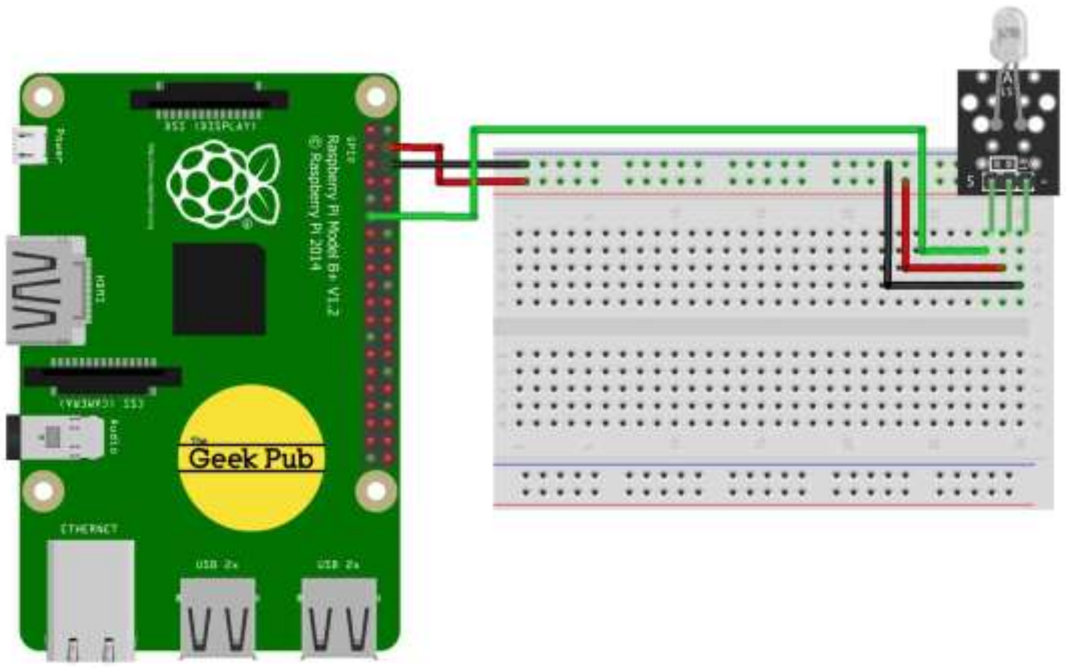


Program:

```
import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library

def button_callback(channel):
    print("Button was pushed!")

GPIO.setwarnings(False) # Ignore warning for now
GPIO.setmode(GPIO.BOARD) # Use physical pin numbering
GPIO.setup(10, GPIO.IN, pull_up_down=GPIO.PUD_DOWN) # Set pin
10 to be an input pin and set initial value to be pulled low (off)

GPIO.add_event_detect(10,GPIO.RISING,callback=button_callback) #
Setup event on pin 10 rising edge

message = input("Press enter to quit\n\n") # Run until someone presses
enter

GPIO.cleanup() # Clean up
```

c. IR Emitter

Circuit Diagram:



Program:

```
import os
import time
def send_ir_command(command):
   os.system(f"irsend SEND_ONCE myremote {command}")
   print(f"Sent command: {command}")

# Example usage
send_ir_command("KEY_POWER")
time.sleep(1)
send_ir_command("KEY_VOLUMEUP")
```

Assignment:

12. SOC Experiments using Raspberry PI or Ordroid Xu4:
    a. Ball Switch
    b. Tap Sensor