**Week 8:** Connect a PIR sensor to the GPIO pins of the Raspberry Pi. Perform measurements to determine the range of the sensor, i.e., start with a small distance (e.g., a few inches) and see if the motion sensor responds. Repeat these for increasing distances until the sensor stops responding. Report the measured distance.

**Hardware Requirement**s**:**

1. jumping wires, breadboard

2. Raspberry Pi

3. PIR Sensor, Ultrasonic Sensor

**Procedure:**

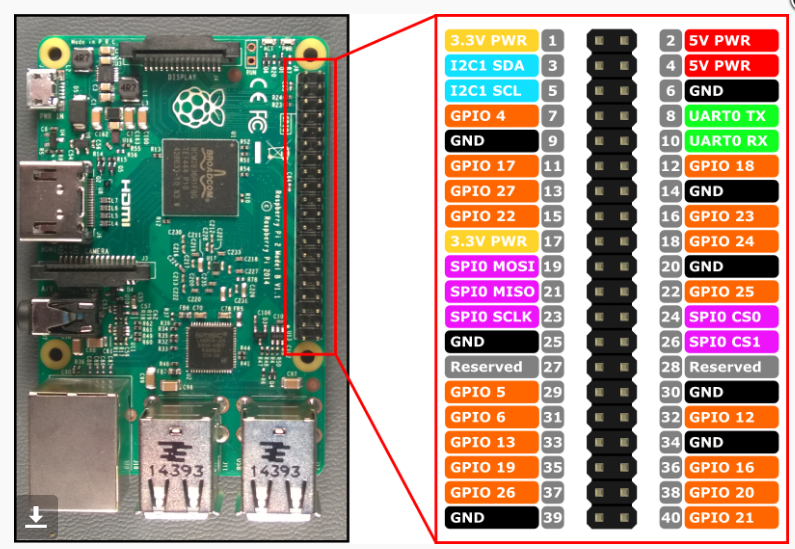
**PIR Sensor Connections:**

1. VCC of PIR Sensor to the 5v of Raspberry's GPIO.
2. GND of PIR Sensor to Ground (GND) on the Raspberry Pi
3. OUT of PIR Sensor to GPIO pin 17 on the Raspberry Pi.

**Ultrasonic Sensor Connections:**

1. VCC of Ultrasonic Sensor to 5V on the Raspberry Pi
2. GND of Ultrasonic Sensor to Ground (GND) on the Raspberry Pi
3. Trig of Ultrasonic Sensor to a GPIO pin 23 on the Raspberry Pi (e.g., GPIO23)
4. Echo of Ultrasonic Sensor to a GPIO pin 24 on the Raspberry Pi (e.g., GPIO24)

**Raspberry Pi Pin-out:**



**Program:**

import RPi.GPIO as GPIO

import time

**# Set up the GPIO mode**

GPIO.setmode(GPIO.BCM)

**# PIR Sensor Pin**

PIR\_PIN = 17

**# Ultrasonic Sensor Pins**

TRIG\_PIN = 23

ECHO\_PIN = 24

**# Set up the GPIO pins**

GPIO.setup(PIR\_PIN, GPIO.IN)

GPIO.setup(TRIG\_PIN, GPIO.OUT)

GPIO.setup(ECHO\_PIN, GPIO.IN)

def measure\_distance():

**# Send a pulse to the trigger pin**

GPIO.output(TRIG\_PIN, GPIO.LOW)

time.sleep(0.1)

GPIO.output(TRIG\_PIN, GPIO.HIGH)

time.sleep(0.00001)

GPIO.output(TRIG\_PIN, GPIO.LOW)

**# Wait for the echo pin to go high**

while GPIO.input(ECHO\_PIN) == GPIO.LOW:

pulse\_start = time.time()

**# Wait for the echo pin to go low**

while GPIO.input(ECHO\_PIN) == GPIO.HIGH:

pulse\_end = time.time()

**# Calculate the duration of the pulse**

pulse\_duration = pulse\_end - pulse\_start

**# Calculate distance (Speed of sound is 34300 cm/s)**

distance = pulse\_duration \* 34300 / 2

return distance

try:

print("Starting motion detection...")

while True:

**# Check for motion detected by PIR sensor**

if GPIO.input(PIR\_PIN):

print("Motion Detected!")

**# Measure distance using ultrasonic sensor**

distance = measure\_distance()

print(f"Distance: {distance:.2f} cm")

else:

print("No motion detected")

**# Wait for a short time to avoid overwhelming the CPU**

time.sleep(1)

except KeyboardInterrupt:

print("Program terminated.")

finally:

GPIO.cleanup()

**Output:**

Starting motion direction

Motion detected

Distance: 10.34cm

No motion detected

No motion detected

No motion detected

Motion detected

Distance: 16.48cm

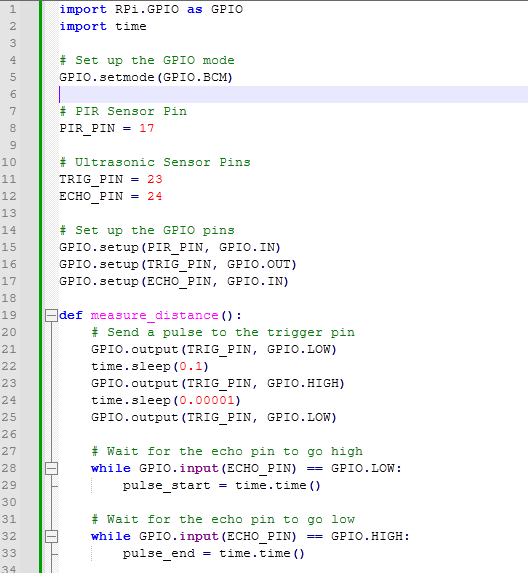
No motion detected

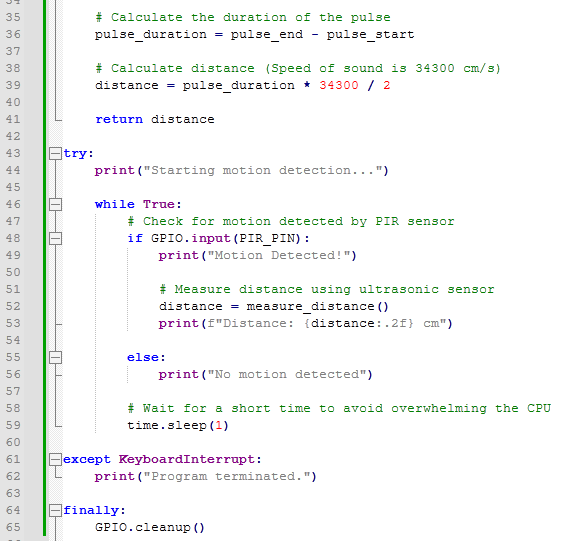
No motion detected

Motion detected

Distance: 42.51cm

No motion detected





**Explanation of Code:**

1. **GPIO Setup:**
   * The PIR sensor is set up on **GPIO17** for input.
   * The ultrasonic sensor uses **GPIO23** for the **Trigger** pin and **GPIO24** for the **Echo** pin.
2. **Distance Measurement Function:**
   * The function measure\_distance() sends a pulse to the **Trig** pin and then waits for the **Echo** pin to receive the pulse back.
   * The time it takes for the pulse to travel to the object and back is used to calculate the distance using the speed of sound formula:  
     Distance=Pulse Duration ×Speed of Sound/2
   * {Distance} = {Pulse Duration} x{Speed Sound}/{2} ​
3. **Main Loop:**
   * The loop continuously checks the PIR sensor. If motion is detected (PIR sensor reads HIGH), it measures the distance using the ultrasonic sensor and prints the result.
   * If no motion is detected, it simply prints "No motion detected."
   * The program runs until you stop it with **CTRL+C**, at which point it cleans up the GPIO configuration.

**How It Works:**

1. When the PIR sensor detects motion, the system triggers the ultrasonic sensor to measure the distance to the object.
2. If no motion is detected by the PIR sensor, the program will print "No motion detected."
3. If motion is detected, it will print both the message "Motion detected!" and the distance to the object as measured by the ultrasonic sensor.

To combine a PIR sensor and an ultrasonic sensor on a Raspberry Pi, you can create a Python script that will use both sensors to detect motion and measure distance. The PIR sensor will detect if there is motion, and the ultrasonic sensor will measure the distance to an object.

**Materials Needed:**

1. **Raspberry Pi** (any model with GPIO pins)
2. **PIR sensor** (e.g., HC-SR501)
3. **Ultrasonic sensor** (e.g., HC-SR04)
4. **Jumper wires**
5. **Breadboard** (optional but useful for prototyping)
6. **Power supply for Raspberry Pi**

**EXP 8.b:** PIR Sensor: Detecting Motion using Raspberry Pi. When the PIR sensor motion is detected then LED is turned ON and if the No motion is detected then LED is turned OFF.

**Procedure:**

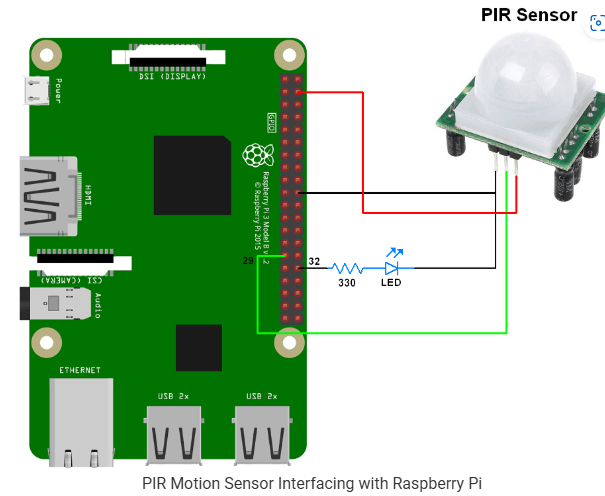
1. Vcc of PIR Sensor to the 5v of Raspberry's GPIO.

2. GND of PIR Sensor to the GND of Raspberry's GPIO.

3. OUT of PIR Sensor to 29 GPIO pin of Raspberry's GPIO. .

4. Connect the anode of the LED to GPIO 32 pin of Raspberry Pi.  
5. Connect the cathode of the LED to Ground GND pin of Raspberry Pi.

**Circuit Diagram:**



**Program:**

import RPi.GPIO as GPIO

PIR\_input = 29 #read PIR Output

LED = 32 #LED for signaling motion detected

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BOARD) #choose pin no. system

GPIO.setup(PIR\_input, GPIO.IN)

GPIO.setup(LED, GPIO.OUT)

GPIO.output(LED, GPIO.LOW)

whileTrue:

#when motion detected turn on LED

if(GPIO.input(PIR\_input)):

GPIO.output(LED, GPIO.HIGH)

print(“Motion detected: LED ON”)

else:

GPIO.output(LED, GPIO.LOW)

print(“No Motion: LED OFF”)

**Exp 8.c: PIR Sensor Motion Detection**

import RPi.GPIO as GPIO

import time

# Set the GPIO mode

GPIO.setmode(GPIO.BCM)

# Define the GPIO pin connected to the PIR sensor

PIR\_PIN = 17 # Change this if you used a different GPIO pin

# Set up the PIR pin as an input

GPIO.setup(PIR\_PIN, GPIO.IN)

# Function to detect motion

def detect\_motion():

try:

print("Starting motion detection. Press CTRL+C to exit.")

while True:

if GPIO.input(PIR\_PIN):

print("Motion detected!")

else:

print("No motion.")

time.sleep(1)

except KeyboardInterrupt:

print("Program exited.")

GPIO.cleanup()

# Run the detection function

detect\_motion()