# IoT Sensor Interfacing Guide with NodeMCU and Raspberry Pi

---

## 1. LM35 Temperature Sensor with NodeMCU

**Wiring:**

- VCC -> 3.3V on NodeMCU
- GND -> GND on NodeMCU
- OUT -> A0 (Analog)

**Code:**

```
float temp;
int temppin = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorvalue = analogRead(temppin);
  float voltage = sensorvalue * (5.0 / 1023.0);
  temp = voltage / 0.01;
  Serial.print("TEMPERATURE = ");
  Serial.print(temp);
  Serial.println("oC");
  delay(1000);
}
```

---

## 2. Soil Moisture Sensor with NodeMCU

**Wiring:**

- VCC -> 3.3V
- GND -> GND
- A0 -> A0 (Analog)

**Code:**

```
const int sensor_pin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensor_analog = analogRead(sensor_pin);
  float moisture_percentage = (100 - ((sensor_analog / 1023.0) * 100));
  Serial.print("Moisture Percentage = ");
  Serial.print(moisture_percentage);
  Serial.println("%");
  delay(1000);
}
```

---

## 3. Raindrop Sensor with NodeMCU

**Wiring:**

- VCC -> 3.3V
- GND -> GND
- A0 -> A0 (Analog)

**Code:**

```
#define POWER_PIN D7
#define AO_PIN    A0

void setup() {
  Serial.begin(9600);
  pinMode(POWER_PIN, OUTPUT);
}

void loop() {
  digitalWrite(POWER_PIN, HIGH);
  delay(10);
  int rainValue = analogRead(AO_PIN);
  digitalWrite(POWER_PIN, LOW);
  Serial.println(rainValue);
  delay(1000);
}
```

---

## 4. Ultrasonic Sensor with NodeMCU

**Wiring:**

- VCC -> VIN
- GND -> GND
- TRIG -> D6 (GPIO 12)
- ECHO -> D5 (GPIO 14)

**Code:**

```
const int trigPin = 12;
const int echoPin = 14;
#define SOUND_VELOCITY 0.034
#define CM_TO_INCH 0.393701

void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH);
  float distanceCm = duration * SOUND_VELOCITY / 2;
  float distanceInch = distanceCm * CM_TO_INCH;

  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  Serial.print("Distance (inch): ");
  Serial.println(distanceInch);
  delay(1000);
}
```

---

## 5. PIR Motion Sensor with NodeMCU

**Wiring:**

- VCC -> 3.3V
- GND -> GND
- OUT -> D1 (GPIO 4)

**Code:**

```
int sensor = 4;

void setup(){
  pinMode(sensor, INPUT);
  Serial.begin(9600);
}

void loop(){
  int state = digitalRead(sensor);
  if (state == HIGH){
    Serial.println("Motion detected");
  } else {
    Serial.println("Motion absent");
  }
  delay(1000);
}
```

---

## 6. LED Blink with Raspberry Pi

**Wiring (Single LED):**

- GPIO 8 -> LED Anode
- GND -> LED Cathode

**Code:**

```
import RPi.GPIO as gp
from time import sleep

gp.setwarnings(False)
gp.setmode(gp.BOARD)
gp.setup(8, gp.OUT, initial=gp.LOW)
```

```
while True:
    gp.output(8, gp.HIGH)
    print("LED ON")
    sleep(1)
    gp.output(8, gp.LOW)
    print("LED OFF")
    sleep(1)
```

**Wiring (Multi LED):**

- GPIO 8 -> LED1
- GPIO 16 -> LED2
- GND -> Both LEDs Cathode

**Code:**

```
import RPi.GPIO as gp
from time import sleep

gp.setwarnings(False)
gp.setmode(gp.BOARD)
gp.setup(8, gp.OUT, initial=gp.LOW)
gp.setup(16, gp.OUT, initial=gp.LOW)

while True:
    gp.output(8, gp.HIGH)
    gp.output(16, gp.LOW)
    sleep(1)
    gp.output(8, gp.LOW)
    gp.output(16, gp.HIGH)
    sleep(1)
```

---

## 7. Raindrop Sensor with Raspberry Pi

**Wiring:**

- VCC -> 3.3V
- GND -> GND
- OUT -> GPIO 18

**Code:**

```
from time import sleep
from gpiozero import InputDevice

no_rain = InputDevice(18)

while True:
    if not no_rain.is_active:
        print("No rain")
    else:
        print("Raining")
    sleep(1)
```

---

## 8. Buzzer with Raspberry Pi

**Wiring:**

- VCC -> 5V
- GND -> GND
- Signal -> GPIO 23

**Code:**

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)

while True:
    GPIO.output(23, GPIO.HIGH)
    print("Beep")
    sleep(0.5)
    GPIO.output(23, GPIO.LOW)
    print("No Beep")
    sleep(0.5)
```

---

## 9. Soil Moisture Sensor with Raspberry Pi (Digital)

**Wiring:**

- D0 -> GPIO 14 (Pin 8)
- Red LED -> GPIO 12 (Pin 32)
- Green LED -> GPIO 16 (Pin 36)
- VCC -> 5V
- GND -> GND

**Code:**

```
import RPi.GPIO as gp

gp.setwarnings(False)
gp.setmode(gp.BOARD)
gp.setup(8, gp.IN)
gp.setup(36, gp.OUT)
gp.setup(32, gp.OUT)

while True:
    try:
        print(not gp.input(8))
        gp.output(36, gp.input(8))
        gp.output(32, not gp.input(8))
    except:
        gp.cleanup()
```

---

## 10. Ultrasonic Sensor with Raspberry Pi

**Wiring:**

- TRIG -> GPIO 23
- ECHO -> GPIO 24
- VCC -> 3.3V
- GND -> GND

**Code:**

```
import RPi.GPIO as GPIO
```

```
from time import sleep, time

GPIO.setmode(GPIO.BCM)
TRIG = 23
ECHO = 24
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

while True:
    GPIO.output(TRIG, False)
    sleep(2)

    GPIO.output(TRIG, True)
    sleep(0.00001)
    GPIO.output(TRIG, False)

    while GPIO.input(ECHO) == 0:
        pulse_start = time()
    while GPIO.input(ECHO) == 1:
        pulse_end = time()

    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration * 17150
    distance = round(distance, 2)

    print("Distance:", distance, "cm")
```

---

## ✅ End of Guide

You now have a complete reference for interfacing NodeMCU and Raspberry Pi with common IoT sensors.