Experiment No._03

Title: Sensor and actuator interfacing with ESP32

Batch: B – 1 **Roll No.: 16010422234** **Experiment No.: 03**

Aim: and actuator interfacing with ESP 32

Resources needed: Internet, Raspberry Pi module, Sensors and Actuators

---

Theory:
Pre Lab/ Prior Concepts:

The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enable these objects to connect and exchange data. Each thing is uniquely identifiable through its embedded computing system but is able to inter-operate within the existing Internet infrastructure.

Role of Sensor in IoT:
Sensors are now found in a wide variety of applications, such as smart mobile devices, automotive systems, industrial control, healthcare, oil exploration and climate monitoring. Sensors are used almost everywhere, and now sensor technology is beginning to closely mimic the ultimate sensing machine, the human being. The technology that allows this to happen is sensor fusion, which leverages a microcontroller to fuse the individual data collected from multiple sensors to get a more accurate and reliable view of the data than one would get by using the data from each discrete sensor on its own. Sensor fusion creates a situation in which the whole is much greater than the sum of its parts.

Role of Actuator in IoT:
An actuator is a component of a machine that is responsible for moving and controlling a mechanism or system, for example by opening a valve. In simple terms, it is a "mover".
An actuator requires a control device (controlled by control signal) and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic, or hydraulic fluid pressure, or even human power. Its main energy source may be an electric current, hydraulic pressure, or pneumatic pressure. The control device is usually a valve. When it receives a control signal, an actuator responds by converting the source's energy into mechanical motion. In the electric, hydraulic, and pneumatic sense, it is a form of automation or automatic control. The displacement achieved is commonly linear or rotational, as exemplified by linear motors and rotary motors, respectively. Rotary motion is more natural for small machines

making large displacements. By means of a leadscrew, rotary motion can be adapted to function as a linear actuator (a linear motion, but not a linear motor).

Activity:

1. List out sensors which can be used with ESP32.

ESP32 supports a wide variety of sensors for different IoT applications. Below is a categorized list of sensors that can be used with ESP32:

1. Environmental Sensors
   - DHT11 / DHT22 – Temperature & Humidity Sensor
   - BME280 – Temperature, Humidity, and Pressure Sensor
   - BMP180 / BMP280 – Barometric Pressure Sensor
   - MQ Series (MQ-2, MQ-3, MQ-7, etc.) – Gas Sensors (Smoke, CO, Alcohol, etc.)
   - DS18B20 – Waterproof Temperature Sensor

2. Motion & Proximity Sensors
   - PIR Sensor (HC-SR501) – Motion Detection
   - Ultrasonic Sensor (HC-SR04) – Distance Measurement
   - IR Sensor Module – Object Detection
   - ToF Sensor (VL53L0X / VL53L1X) – Time of Flight Distance Measurement

3. Light & Optical Sensors
   - LDR (Light Dependent Resistor) – Light Intensity Measurement
   - TSL2561 / BH1750 – Ambient Light Sensor
   - TCS3200 – Color Sensor

4. Touch & Gesture Sensors
   - Capacitive Touch Sensors – Touch Detection (ESP32 has built-in touch sensing pins)
   - APDS-9960 – Gesture Recognition Sensor

5. Sound & Vibration Sensors
   - Microphone Module (MAX9814, KY-037) – Sound Detection
   - Piezo Vibration Sensor – Vibration Detection

6. Air Quality & Gas Sensors
   - MQ-135 – Air Quality Sensor
   - MH-Z19B – CO2 Sensor

7. Current & Voltage Sensors
   - ACS712 – Current Sensor
   - INA219 – Voltage & Current Sensor

8. Magnetic & Position Sensors
   - Hall Effect Sensor (A3144, KY-003) – Magnetic Field Detection (ESP32 also has a built-in Hall sensor)
   - MPU6050 / MPU9250 – Accelerometer & Gyroscope
   - GY-NEO6MV2 – GPS Module

These sensors can be interfaced with ESP32 using I2C, SPI, UART, or GPIO connections, depending on their communication protocol.

2. Integrate a variety of available sensors in the lab with ESP32, list out step performed, circuit, code, output and components used.

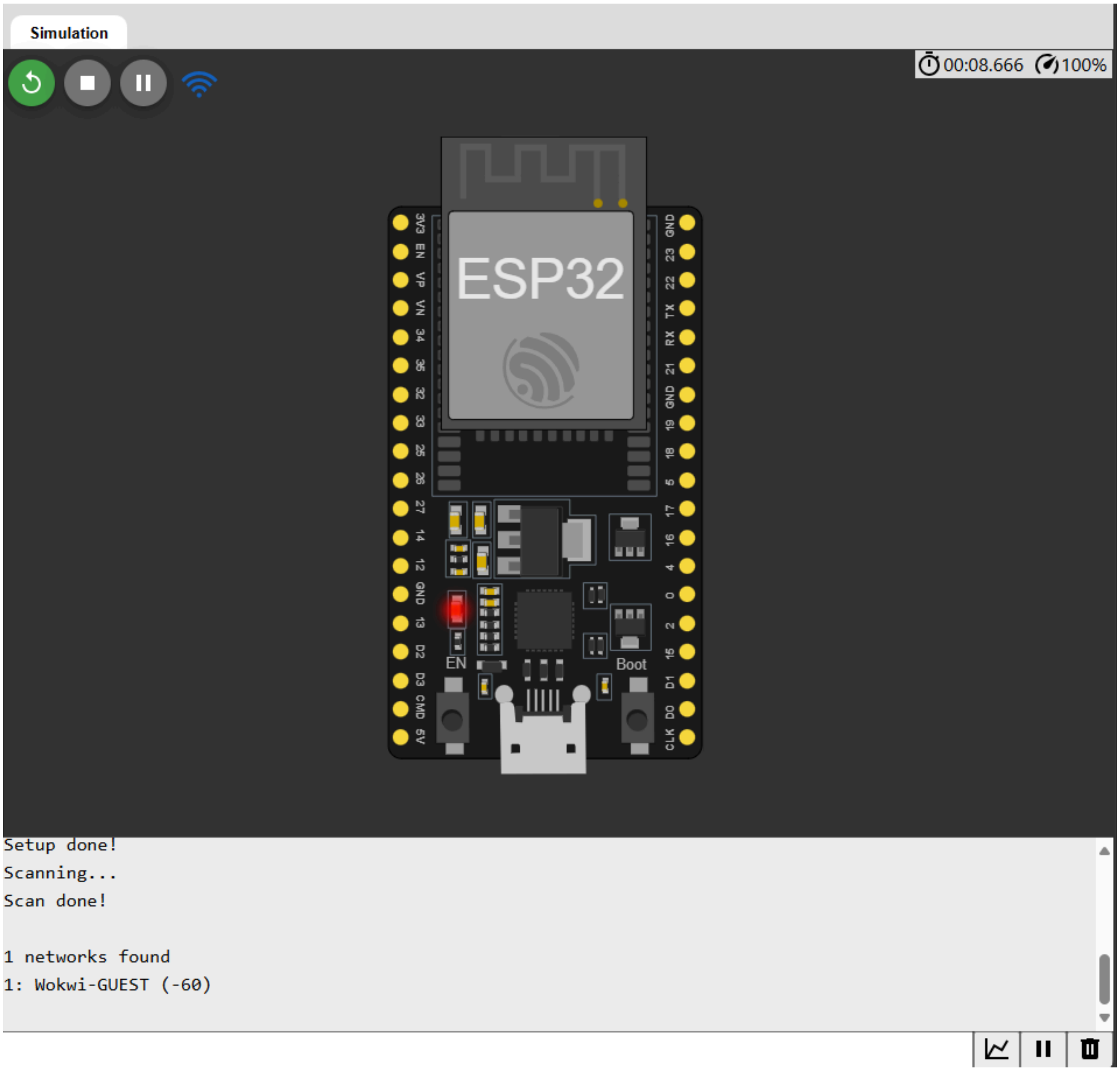
Results: (Program printout with output / Document printout as per the format)

Simulation Code:

```
/* ESP32 WiFi Scanning example */

#include "WiFi.h"

void setup() {
  Serial.begin(115200);
  Serial.println("Initializing WiFi...");
  WiFi.mode(WIFI_STA);
  Serial.println("Setup done!");
}

void loop() {
  Serial.println("Scanning...");

  // WiFi.scanNetworks will return the number of networks found
  int n = WiFi.scanNetworks();
  Serial.println("Scan done!");
  if (n == 0) {
    Serial.println("No networks found.");
```

```
  } else {
    Serial.println();
    Serial.print(n);
    Serial.println(" networks found");
    for (int i = 0; i < n; ++i) {
      // Print SSID and RSSI for each network found
      Serial.print(i + 1);
      Serial.print(": ");
      Serial.print(WiFi.SSID(i));
      Serial.print(" (");
      Serial.print(WiFi.RSSI(i));
      Serial.print(")");
      Serial.println((WiFi.encryptionType(i) == WIFI_AUTH_OPEN) ? " " :
"*");
      delay(10);
    }
  }
  Serial.println("");

  // Wait a bit before scanning again
  delay(5000);
}
```

Simulation:

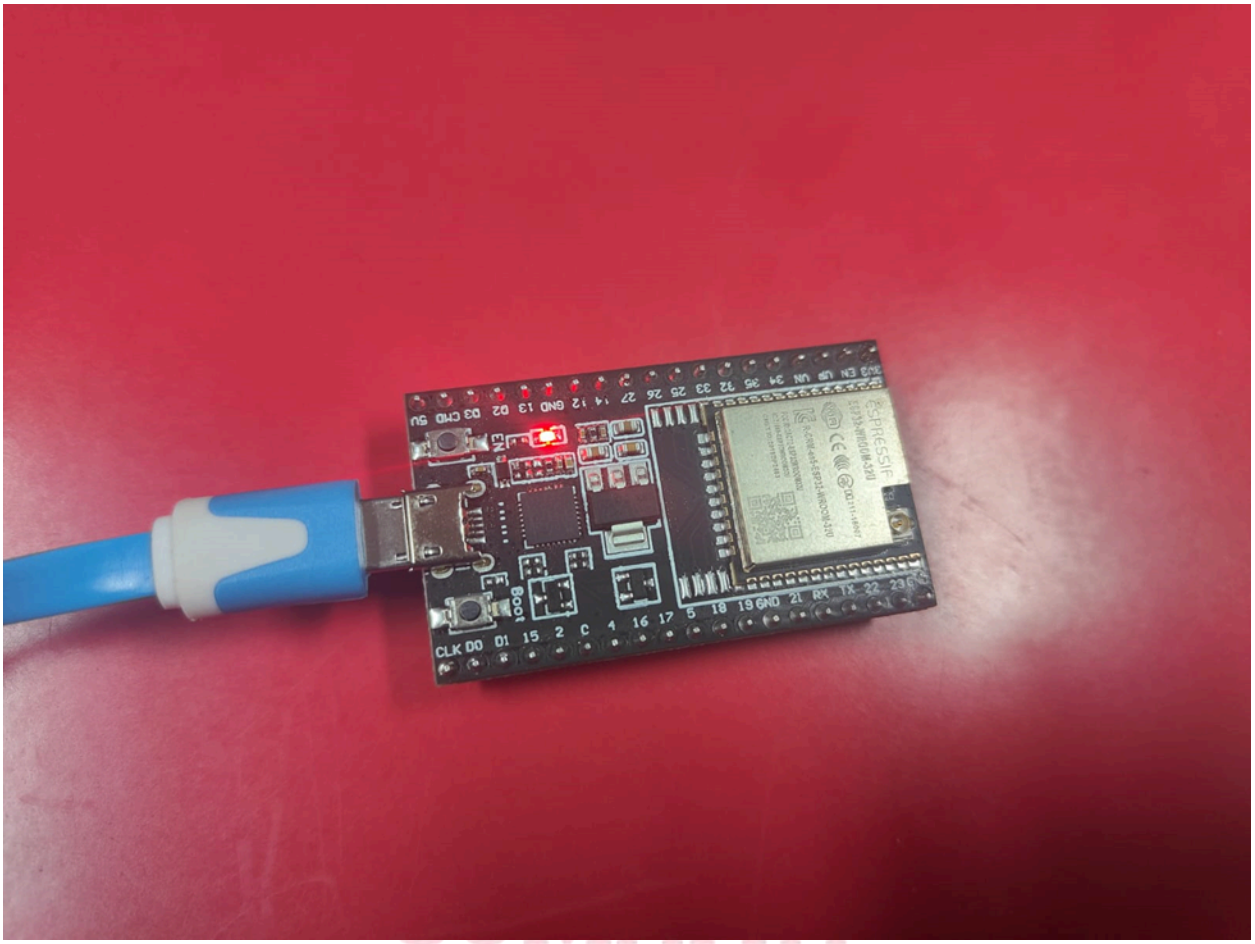

Simulation
00:08.666
100%
ESP32
EN
Boot
Setup done!
Scanning...
Scan done!
1 networks found
1: Wokwi-GUEST (-60)

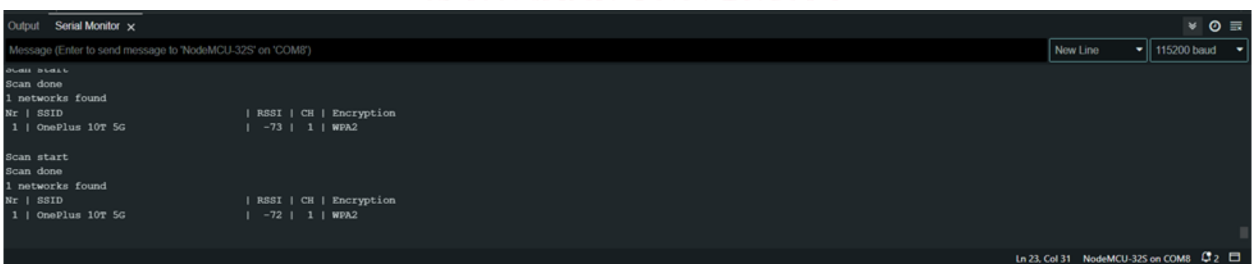
Arduino IDE Code:

```
/*
 *  This sketch demonstrates how to scan WiFi networks.
 *  The API is based on the Arduino WiFi Shield library, but has
significant changes as newer WiFi functions are supported.
 *  E.g. the return value of `encryptionType()` different because more
modern encryption is supported.
 */
#include "WiFi.h"

void setup() {
  Serial.begin(115200);

  // Set WiFi to station mode and disconnect from an AP if it was
previously connected.
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(100);

  Serial.println("Setup done");
}

void loop() {
  Serial.println("Scan start");

  // WiFi.scanNetworks will return the number of networks found.
  int n = WiFi.scanNetworks();
  Serial.println("Scan done");
  if (n == 0) {
    Serial.println("no networks found");
  } else {
    Serial.print(n);
    Serial.println(" networks found");
    Serial.println("Nr | SSID                             | RSSI | CH |
Encryption");
    for (int i = 0; i < n; ++i) {
      // Print SSID and RSSI for each network found
      Serial.printf("%2d", i + 1);
      Serial.print(" | ");
```

```
      Serial.printf("%-32.32s", WiFi.SSID(i).c_str());
      Serial.print(" | ");
      Serial.printf("%4ld", WiFi.RSSI(i));
      Serial.print(" | ");
      Serial.printf("%2ld", WiFi.channel(i));
      Serial.print(" | ");
      switch (WiFi.encryptionType(i)) {
        case WIFI_AUTH_OPEN:            Serial.print("open"); break;
        case WIFI_AUTH_WEP:             Serial.print("WEP"); break;
        case WIFI_AUTH_WPA_PSK:         Serial.print("WPA"); break;
        case WIFI_AUTH_WPA2_PSK:        Serial.print("WPA2"); break;
        case WIFI_AUTH_WPA_WPA2_PSK:    Serial.print("WPA+WPA2"); break;
        case WIFI_AUTH_WPA2_ENTERPRISE: Serial.print("WPA2-EAP"); break;
        case WIFI_AUTH_WPA3_PSK:        Serial.print("WPA3"); break;
        case WIFI_AUTH_WPA2_WPA3_PSK:   Serial.print("WPA2+WPA3"); break;
        case WIFI_AUTH_WAPI_PSK:        Serial.print("WAPI"); break;
        default:                        Serial.print("unknown");
      }
      Serial.println();
      delay(10);
    }
  }
  Serial.println("");

  // Delete the scan result to free memory for code below.
  WiFi.scanDelete();

  // Wait a bit before scanning again.
  delay(5000);
}
```

NodeMCU-32S




Output
Serial Monitor
Message (Enter to send message to 'NodeMCU-32S' on 'COM8')
New Line
115200 baud
Scan done
1 networks found
Nr | SSID | RSSI | CH | Encryption
1 | OnePlus 10T 5G | -73 | 1 | WPA2
Scan start
Scan done
1 networks found
Nr | SSID | RSSI | CH | Encryption
1 | OnePlus 10T 5G | -72 | 1 | WPA2
Ln 23, Col 31
NodeMCU-32S on COM8

Questions:

1. How can ESP32 be used for data analytics?

Ans: ESP32 can be used for data analytics by collecting data from various sensors, processing it locally, and transmitting it to cloud platforms for further analysis. Here's how:

1. Data Collection: ESP32 can interface with sensors like temperature, humidity, motion, and gas sensors to gather real-time data.

2. Data Processing: It has onboard processing power to filter, preprocess, and store data temporarily.

3. Data Transmission: Using Wi-Fi or Bluetooth, ESP32 can send data to cloud platforms like Google Firebase, AWS IoT, or ThingSpeak for further analysis.

4. Edge Computing: It can perform basic analytics like threshold detection and anomaly detection before transmitting data.

5. Machine Learning (TinyML): With frameworks like TensorFlow Lite, ESP32 can run lightweight ML models to make predictions based on collected data.

6. Visualization & Insights: The data can be visualized on dashboards using platforms like Node-RED, Grafana, or Power BI, enabling informed decision-making.

Outcomes: CO3 – Realize design process of IoT applications and IoT challenges

---

Conclusion:

In this experiment, we successfully interfaced sensors and actuators with ESP32, understanding their role in IoT applications. Sensors help in data collection, while actuators enable control actions based on sensor inputs. ESP32 serves as an efficient microcontroller for IoT applications, supporting real-time data acquisition, processing, and communication. This experiment demonstrated the integration of hardware and software to create intelligent systems capable of automation and analytics.

Grade: AA / AB / BB / BC / CC / CD / DD

Signature of faculty in-charge with date

References:

Links:

https://en.wikipedia.org/wiki/Internet_of_things

Books:

1. Jan Holler, Vlasios Tsiatsis, Catherine Mulligan, Stefan Avesand, Stamatis Karnouskos, David Boyle, “From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence”, 1st Edition, Academic Press, 2014.
2. Vijay Madisetti and Arshdeep Bahga, “Internet of Things (A Hands-on-Approach)”, 1stEdition, VPT, 2014.
3. Dr. Ovidiu Vermesan, Dr. Peter Friess, “Internet of Things - From Research and Innovation to Market Deployment”, River Publisher, 2014