"Microcontrollers Explained: Build a Legal WiFi Detector with ESP32" What is a Microcontroller?

1. Definition

A **microcontroller** is a small computer built into a single integrated circuit (chip). It contains:

- A processor (CPU)
- Memory (RAM and Flash)
- Input/Output (I/O) pins
- Communication modules (WiFi, Bluetooth, UART, etc.)

Unlike a desktop computer, a microcontroller is designed to control specific tasks inside electronic systems.

Examples used in this module:

- **ESP32** (by Espressif Systems)
- **M5Stack Core** (ESP32-based development kit by M5Stack)

2. What Does a Microcontroller Do?

A microcontroller:

1. Reads inputs (buttons, sensors, WiFi signals)
2. Processes data using programmed instructions
3. Controls outputs (LEDs, screens, motors) In this project:

- The ESP32 scans WiFi networks

- It counts detected networks It turns on:
    - Green LED (Normal)
    - Yellow LED (Caution)
    - Red LED (Alert)

It is acting as a **decision-making controller**

3. Microcontroller vs Microprocessor

Feature	**Microcontroller**	**Microprocessor**
Purpose	Embedded control	General computing
Memory	Built-in	External required
Example	ESP32	Laptop CPU

Used in IoT devices, robotics PCs, servers

Microcontrollers are used in:

- Washing machines
- Cars
- Smart TVs
- Medical devices
- IoT systems

4. Internal Components of ESP32 The ESP32 contains:

- Dual-core processor
- WiFi module
- Bluetooth module
- GPIO pins (General Purpose Input Output)

- ADC (Analog to Digital Converter)
- Flash memory

This makes it ideal for:

- Wireless detection
- Sensor monitoring
- Smart automation

5. Why ESP32 is Used in This Project We use the ESP32 because:

- It has built-in WiFi
- It supports scanning nearby networks
- It can control LEDs
- It is affordable and educational
- It supports programming in Arduino IDE Most importantly:

It performs PASSIVE detection only. It does NOT transmit interference.

6. Step-by-Step: How the Microcontroller Works in This Project

1. Power is supplied through USB.
2. The program starts running.
3. WiFi scan begins.
4. Number of networks is counted.
5. Conditional logic checks thresholds.
6. LED color output is activated.

7. Process repeats every 5 seconds.

This is called a control loop.

7. Teaching Explanation (Instructor Notes) When explaining to students:

- Compare a microcontroller to a "tiny brain" of a device.
- Explain that it follows instructions written in code.
- Emphasize embedded systems concept.
- Reinforce that detection is legal, jamming is illegal.

Student Module

Wireless Interference Detection Project

- **LEGAL NOTICE – DETECTION ONLY:** This module describes a wireless interference DETECTION
- system. It does NOT transmit signals and MUST NOT be modified into a jammer. Jamming communications is illegal in most countries and may result in severe penalties.

What You Will Learn

- How WiFi scanning detects nearby networks.
- Difference between detection and illegal jamming.
- How LEDs/LCD provide status feedback.

Color Meaning

- GREEN Normal wireless environment
- YELLOW Caution – increased signal activity
- RED Alert – strong interference detected

Lab Activity

- Measure signal levels in 3 different locations.
- Record number of networks detected. Explain why results differ.

Worksheet Questions

- Why is wireless jamming illegal?
- What does GREEN indicate?
- What environment produced RED? Why?
-

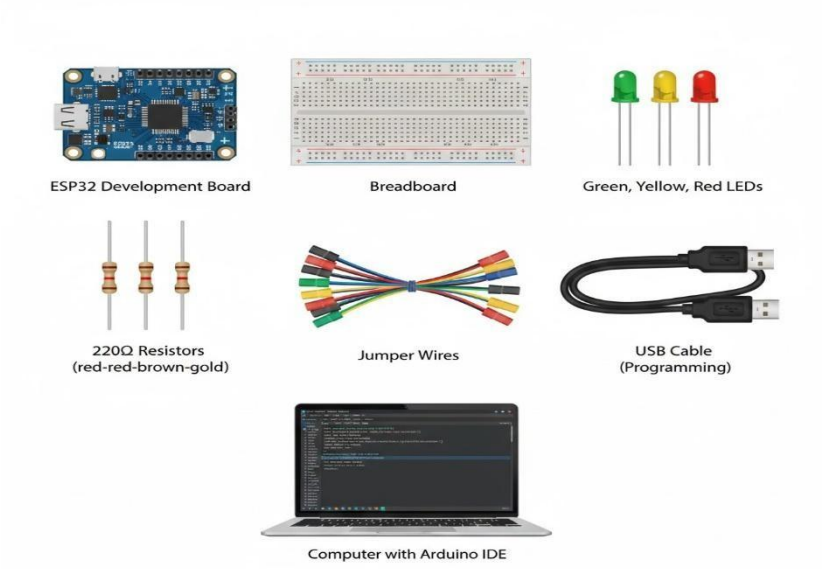
ExperimentActivity

🔧 Step-By-Step Build Procedure (Student Instructions)

Before You Start

Make sure you have the following materials:

- ESP32 development board
- Breadboard
- 3 LEDs (Green, Yellow, Red)
- 3 × 220Ω resistors
- Jumper wires
- USB cable
- Computer with Arduino IDE installed



Step 1 – Insert the ESP32 into the Breadboard

1. Place the ESP32 across the center gap of the breadboard.
2. Make sure both rows of pins are accessible.
3. Do NOT connect USB power yet.

Step 2 – Connect the GREEN LED (GPIO 25)

1. Insert the long leg (anode) of the green LED into the breadboard.
2. Insert the short leg (cathode) into a different row.
3. Connect:

   • GPIO 25 → one end of a 220Ω resistor

   • Other end of resistor → long leg of green LED

   • Short leg of green LED → GND pin on ESP32

The green LED represents NORMAL condition.

Step 3 – Connect the YELLOW LED (GPIO 26) 1.

Insert the yellow LED into the breadboard.

2. Connect:

• GPIO 26 → 220Ω resistor → long leg of yellow LED

• Short leg → GND

The yellow LED represents CAUTION condition.

Step 4 – Connect the RED LED (GPIO 27) Insert

the red LED into the breadboard.

Connect:

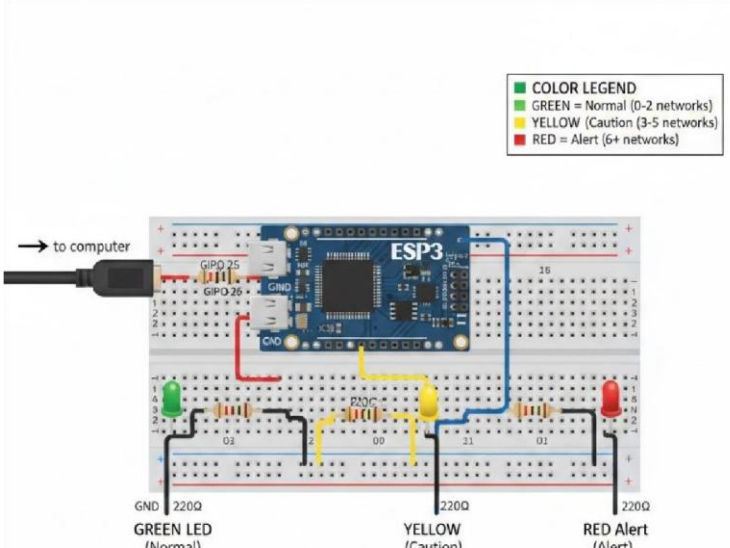
• GPIO 27 → 220Ω resistor → long leg of red LED
• Short leg → GND

The red LED represents ALERT condition.

Step 5 – Check All Connections Before

powering the board:

• Each LED must have its own 220Ω resistor



- All short LED legs must go to GND
- No wires should touch incorrectly
- Verify GPIO numbers are correct

This prevents short circuits and damage.

Step 6 – Connect and Power the ESP32

1. Connect the ESP32 to your computer using USB.
2. Wait for the computer to detect the device.

Step 7 – Configure Arduino IDE

1. Open Arduino IDE.
2. Go to Tools → Board → Select "ESP32 Dev Module".
3. Go to Tools → Port → Select the correct COM port.
4. Paste the provided code into the editor.

Step 8 – Upload the Program

1. Click Upload.
2. If upload fails:
    - Hold the BOOT button while clicking Upload.
    - Release when uploading begins.

Wait until "Done Uploading" appears.

Step 9 – Test the System

1. Open Serial Monitor.
2. Set baud rate to 115200.
3. Observe the number of WiFi networks detected.

4. Observe which LED turns on. Move to different locations to compare results. code

```
#include <WiFi.h> int
G = 25;

int Y = 26; int
R = 27;


void setup() {
Serial.begin(115200);  pinMode(G,
OUTPUT);  pinMode(Y, OUTPUT);
pinMode(R, OUTPUT);


  WiFi.mode(WIFI_STA);
WiFi.disconnect();  delay(100);
}
void loop() {
Serial.println("Scanning...");  int
n = WiFi.scanNetworks();


  Serial.print("Networks detected: ");
  Serial.println(n);
```

```
   if (n > 5)
{
digitalWrite
(R, HIGH);
digitalWrite
(Y, LOW);
digitalWrite
(G, LOW);
  }    else if (n > 2) {
digitalWrite(Y, HIGH);
digitalWrite(R, LOW);
digitalWrite(G, LOW);
  }
else {
    digitalWrite(G, HIGH);
digitalWrite(R, LOW);      digitalWrite(Y,
LOW);
  }


  delay(5000);
}
```