**Slip 5: To write a program to connect with the available Wi-Fi using Arduino.**

To simulate the Wi-Fi connection in Proteus using the ESP8266 or ESP32 with  Arduino, it is important to note that Proteus does not have built-in support for  ESP8266 or ESP32 modules. However, you can still simulate the Arduino Uno and  connect it to a Wi-Fi module (like ESP8266) with some workaround steps. Proteus  is generally better for simulating circuits with basic components (LEDs, motors,  sensors, etc.), but it does not natively simulate Wi-Fi connections.

Here’s a guide to simulate the communication between Arduino Uno and  ESP8266 in Proteus:

**Components Needed for Proteus Simulation:**

1. Arduino Uno

2. ESP8266 Wi-Fi module (simulate serial communication; Proteus does not have  an exact ESP8266 module, but you can use a serial communication module  instead, or a generic module for testing)

3. LED (to represent output, which could be triggered when the Wi-Fi connection  is established)

4. Resistor (for the LED)

5. SoftwareSerial (for serial communication)

**Steps to Simulate the Arduino with ESP8266 in Proteus:**

**1. Create New Project:**

- Open Proteus and create a new project.

**2. Add Components:**

- Arduino Uno.

 - ESP8266 (use a generic serial module or simply connect to a virtual terminal  for testing communication).

 - LED and Resistor (220 ohms).

 - Jumper wires.

**3. Circuit Setup:**

- ESP8266 TX (Transmit) → Arduino RX (Pin 0).

 - ESP8266 RX (Receive) → Arduino TX (Pin 1).

 - ESP8266 VCC → 3.3V (ESP8266 operates on 3.3V, so you should use a 3.3V  supply).

 - ESP8266 GND → GND.

 - LED: Connect the anode (long leg) of the LED to digital pin 13 of the Arduino  and the cathode (short leg) to GND via a 220-ohm resistor.

**4. Configure Serial Communication in Proteus:**

- In the Proteus schematic, add a Virtual Terminal to simulate serial  communication and connect it to the ESP8266 TX/RX pins for debugging.

 - Connect the Virtual Terminal to the Arduino TX/RX pins (Pins 0 and 1), so that  you can view the serial output in the Proteus simulation.

**5. Arduino Code:**

Use the following Arduino code (similar to the one previously shared) that  attempts to connect to a Wi-Fi network.

#include <SoftwareSerial.h>

#include <ESP8266WiFi.h>

// Replace with your network credentials

const char\* ssid = "YOUR\_SSID"; // Wi-Fi network name (SSID) const char\* password = "YOUR\_PASSWORD"; // Wi-Fi password

void setup() {

 // Start serial communication with the computer

 Serial.begin(115200); // Adjust baud rate if needed

 // Start serial communication with ESP8266

 Serial1.begin(115200); // ESP8266 baud rate (depends on your module)  // Connecting to Wi-Fi

 WiFi.begin(ssid, password);

 Serial.println();

 Serial.println("Connecting to WiFi...");

 while (WiFi.status() != WL\_CONNECTED) {  delay(1000);

 Serial.print(".");

 }

 // If connected to Wi-Fi, print the IP address  Serial.println("Connected to WiFi");

 Serial.print("IP Address: ");

 Serial.println(WiFi.localIP());

}

void loop() {

 // Nothing to do in loop, just keep the connection alive }

**6. Upload the Code to Arduino:**

 - For simulating communication in Proteus, you need to use the HEX file generated when you upload the code from Arduino IDE.

 - Right-click on the Arduino Uno in the Proteus schematic and go to Properties.

 - In the Program File field, browse and select the .hex file generated by the  Arduino IDE after you upload the code.

**7. Run the Simulation:**

- After setting up the circuit and uploading the code to the Arduino, click the  play button in Proteus to start the simulation.

 - You should see the serial communication output in the Virtual Terminal (like  "Connecting to WiFi" and the IP address once connected).