ESP32 Static IP Setup

Perfect! Since you want the **ESP32 to use a static IP** and send data to Unity **without needing to know the phone's IP**, the best solution is:

✅ ESP32 = UDP Server

✅ Unity (Mobile App) = UDP Client (connects to known ESP32 IP)

🔧 ESP32 Setup (with Static IP, Receives Messages from Unity)

Modify your code to:

1. Set static IP for ESP32
2. Respond to any UDP packets Unity sends (so Unity knows the ESP32 IP)

✏️ Updated ESP32 Code:

```cpp
#include <WiFi.h>
#include <WiFiUdp.h>
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

Adafruit_MPU6050 mpu;
WiFiUDP udp;

// Wi-Fi credentials
const char* ssid = "Rashmika";
const char* password = "rashmika";
```

```
// Static IP configuration
IPAddress local_IP(192, 168, 108, 222);  // <-- Set a fixed IP
IPAddress gateway(192, 168, 108, 1);
IPAddress subnet(255, 255, 255, 0);

int udpPort = 4210;
IPAddress remoteIP;
int remotePort;

int buttonPins[6] = {14, 27, 26, 25, 33, 32}; // Last one is B6

void setup() {
  Serial.begin(115200);
  delay(1000);

  // Apply static IP BEFORE WiFi.begin()
  if (!WiFi.config(local_IP, gateway, subnet)) {
    Serial.println("Failed to configure static IP");
  }

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected. ESP32 IP: ");
  Serial.println(WiFi.localIP());

  Wire.begin();
  if (!mpu.begin()) {
    Serial.println("MPU6050 not found");
    while (1) delay(10);
  }

  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);

  for (int i = 0; i < 6; i++) {
    pinMode(buttonPins[i], INPUT_PULLUP);
  }
  udp.begin(udpPort);
}

void loop() {
  // Wait for Unity to send first packet
  int packetSize = udp.parsePacket();
  if (packetSize) {
    remoteIP = udp.remoteIP();
    remotePort = udp.remotePort();
  }

  if (remotePort == 0) return;  // Wait until Unity connects

  bool b6Pressed = digitalRead(buttonPins[5]) == LOW;

  String data;

  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  if (b6Pressed) {
    data = String(a.acceleration.x, 2) + "," + String(a.acceleration.y, 2) + "," + String(a.acceleration.z, 2) +
",";
    data += String(g.gyro.x, 2) + "," + String(g.gyro.y, 2) + "," + String(g.gyro.z, 2);
  } else {
    data = "0,0,0,0,0,0";
  }

  for (int i = 0; i < 6; i++) {
    int state = digitalRead(buttonPins[i]) == LOW ? 1 : 0;
    data += "," + String(state);
  }

  udp.beginPacket(remoteIP, remotePort);
  udp.print(data);
  udp.endPacket();

  Serial.println(data);
  delay(30);  // ~33Hz
}
```

🎮 Unity Mobile Code (C# UDP Client, sends handshake first)

✅ Unity C# Example Script (UDP receive + handshake):

```csharp
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Threading;
using UnityEngine;

public class ESP32Receiver : MonoBehaviour
{
    UdpClient udpClient;
    Thread receiveThread;

    public string esp32IP = "192.168.108.222";  // Must match ESP32 static IP
    public int esp32Port = 4210;

    void Start()
    {
        udpClient = new UdpClient();

        // Send handshake message to ESP32 so it knows our IP and port
        IPEndPoint remoteEndPoint = new IPEndPoint(IPAddress.Parse(esp32IP), esp32Port);
        byte[] handshake = Encoding.UTF8.GetBytes("HelloESP32");
        udpClient.Send(handshake, handshake.Length, remoteEndPoint);

        // Start receive thread
        receiveThread = new Thread(ReceiveData);
        receiveThread.IsBackground = true;
        receiveThread.Start();
    }

    void ReceiveData()
    {
        IPEndPoint anyIP = new IPEndPoint(IPAddress.Any, 0);

        while (true)
        {
            try
            {
                byte[] data = udpClient.Receive(ref anyIP);
                string text = Encoding.UTF8.GetString(data);
                Debug.Log("Received from ESP32: " + text);
            }
            catch (SocketException err)
            {
                Debug.Log("Socket error: " + err);
                break;
            }
        }
    }

    private void OnApplicationQuit()
    {
        receiveThread?.Abort();
        udpClient?.Close();
    }
}
```

✅ What This Setup Solves

- You **don't need to know your mobile IP** — Unity sends a message, and ESP32 learns it.
- ESP32 always uses a **fixed IP**, easy to set in Unity.
- This works well on **mobile phones**, **Wi-Fi**, and **Unity builds**.

✅ Final Checklist

Task	Done?
ESP32 has a fixed IP (e.g. `192.168.108.222`)	✅
Unity sends a "hello" to ESP32 once	✅
ESP32 responds using `udp.remoteIP()` and `udp.remotePort()`	✅

Task	Done?
Mobile & ESP32 on **same Wi-Fi network**	✅
UDP port (`4210`) matches on both sides	✅

Would you also like to support **SoftAP** mode (no router needed)?