ESP32 → ESP8266 UART Communication

Concept Overview

What is happening?

- **ESP32** sends a message over **UART** to the **ESP8266**.
- The **ESP8266** reads this message via `SoftwareSerial` and prints it to its **USB Serial Monitor**.

ESP 32 Code on platform IO :

```
#include <Arduino.h>

HardwareSerial MySerial(1);  // Use UART1

void setup() {

    Serial.begin(115200); // Monitor on USB

    MySerial.begin(9600, SERIAL_8N1, 16, 17); // RX=16, TX=17

    delay(1000);

}

void loop() {

    MySerial.println("Hello from ESP32 via UART!");

    Serial.println("ESP32 Sent: Hello to ESP8266");

    delay(1000);

}
```

ESP32 Code Explanation (Sender) — Line by Line:

```
#include <Arduino.h>
```

Includes the basic Arduino core functions.

```
HardwareSerial MySerial(1);  // Use UART1
```

Declares a **HardwareSerial object** named `MySerial` using **UART1** on the ESP32.

- ESP32 has **3 UARTs**: UART0 (USB), UART1, UART2.
- You're using **UART1** (you can pick pins freely here).

```
void setup() {
```

Runs once at the beginning.

```
  Serial.begin(115200); // Monitor on USB
```

Start the **USB Serial Monitor (COM port)** at 115200 baud, so you can see debug prints from the ESP32 in PlatformIO.

```
  MySerial.begin(9600, SERIAL_8N1, 16, 17); // RX=16, TX=17
```

Starts **UART1 at 9600 baud**, 8-bit data, no parity, 1 stop bit (**8N1**).

- **The RX pin is GPIO16** (unused here because you are only sending).
- **The TX pin is GPIO17** — sending data to the ESP8266's RX.

```
  delay(1000);
```

Short 1-second delay to make sure everything initializes properly.

```
}

void loop() {
```

Code here runs **again and again forever**.

```
  MySerial.println("Hello from ESP32 via UART!");
```

Send the text message `"Hello from ESP32 via UART!"` over **UART1 to ESP8266**.

```
  Serial.println("ESP32 Sent: Hello to ESP8266");
```

Prints to the ESP32's own **USB Serial Monitor (PlatformIO)** — just for your debugging view.

```
  delay(1000);
```

Wait 1 second before sending the next message (otherwise it would flood the line).

```
}
```

End of loop.

ESP8266 code on ArduinoIDE

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(D2, D1); // RX, TX (ESP8266: D2 is GPIO4, D1 is GPIO5)

void setup() {
  Serial.begin(115200);     // USB Serial Monitor
  mySerial.begin(9600);     // UART from ESP32
}

void loop() {
  if (mySerial.available()) {
    String msg = mySerial.readStringUntil('\n');
    Serial.println("Received: " + msg); // Print received UART message to serial
monitor
  }
}
```

ESP8266 Code Explanation (Receiver) — Line by Line:

```
#include <SoftwareSerial.h>
```

Includes the **SoftwareSerial library** — required because ESP8266 has only **one hardware UART (used by USB)**.

```
SoftwareSerial mySerial(D2, D1); // RX, TX (ESP8266: D2 is GPIO4, D1 is GPIO5)
```

Creates **SoftwareSerial instance**:

- **RX: D2 (GPIO4)** — this gets data from ESP32's TX.
- **TX: D1 (GPIO5)** — not used because ESP8266 only receives it here.

```
void setup() {
```

Setup runs once at startup.

```
 Serial.begin(115200);     // USB Serial Monitor
```

Start the **USB Serial Monitor** (so you can see data in Arduino IDE Serial Monitor at 115200 baud).

```
 mySerial.begin(9600);     // UART from ESP32
```

Starts **SoftwareSerial** at 9600 baud — matches the ESP32's UART1 baud rate (9600).

```
}
```

End of setup.

```
void loop() {
```

The main program loop — runs forever.

```
 if (mySerial.available()) {
```

Checks if any data has been received from ESP32 via SoftwareSerial.

```
   String msg = mySerial.readStringUntil('\n');
```

Read incoming text from ESP32 **until it sees a newline `\n`**.
This matches the ESP32's `MySerial.println()` — which sends a newline at the end.

```
   Serial.println("Received: " + msg); // Print received UART message to serial monitor
```

Prints the received message on the **USB Serial Monitor (Arduino IDE)** for you to see.

```
 }
```

End of `if()` check.

```
}
```

End of loop.

Wiring Explanation :

ESP32 (Dev Board)	ESP8266 (NodeMCU)	Purpose
GPIO17 (TX1)	D2 (GPIO4, RX)	ESP32 sends UART data to ESP8266
GND	GND	Must share common ground

Important Additional Details You Applied (Whether You Realized or Not!)

✅	Detail
✔	**ESP32 TX connected to ESP8266 RX (not TX-TX!)**
✔	**Baud rate for UART = 9600 on both sides** (critical!)
✔	ESP32 prints debug to PlatformIO Serial Monitor @ 115200
✔	ESP8266 prints to Arduino IDE Serial Monitor @ 115200
✔	You disconnected UART wires during upload (or avoided flashing errors)
✔	Both boards share **the same GND**

What’s happening in real life?

1. ESP32 Hardware UART1 sends `"Hello from ESP32 via UART!\n"` at 9600 baud on GPIO17 (TX1).
2. ESP8266 SoftwareSerial listens on D2 (GPIO4 RX) at 9600 baud.
3. When ESP8266 receives the message, it reads until the newline character (`\n`).
4. The received string is printed to the USB Serial Monitor on the laptop via ESP8266’s default UART (Serial).