EAST WEST UNIVERSITY

Lab 03 Report

Group Work
Course Code: CSE406
Course Title: Internet of Things
Section: 01
Lab Title: UART Interface

Submitted By:

Name	ID
Tanjila Akter	2022-1-60-078
Ronjon Kar	2022-1-60-091
Sabera Zannat Sheba	2022-2-60-095
Suddip Paul Arnab	2022-1-60-356

Submitted To:

Dr. Raihan Ul Islam
Associate Professor
Department of Computer Science Engineering
East West University, Dhaka, Bangladesh

Submission Date: 31 October, 2025

Contents

1 Introduction

UART (Universal Asynchronous Receiver-Transmitter) is one of the most common serial communication methods in microcontrollers. It uses two main data lines **TX (Transmit)** and **RX (Receive)** to send and receive information asynchronously, along with a **common ground (GND)**. This project demonstrates **board-to-board UART communication between two ESP32 boards** using the **HardwareSerial** interface. The objective was to understand UART fundamentals, implement data transmission, and analyze communication performance across different baud rates and message sizes.

2 Objectives

- Understand UART basics: TX/RX lines, baud rate, frame format (8N1), and cross-wiring.
- Implement UART communication between two ESP32 boards using custom pins.
- Measure performance metrics such as throughput, message rate, and error rate.
- Identify optimal UART settings for reliable and high-speed communication.

3 Methodology

3.1 Hardware Setup

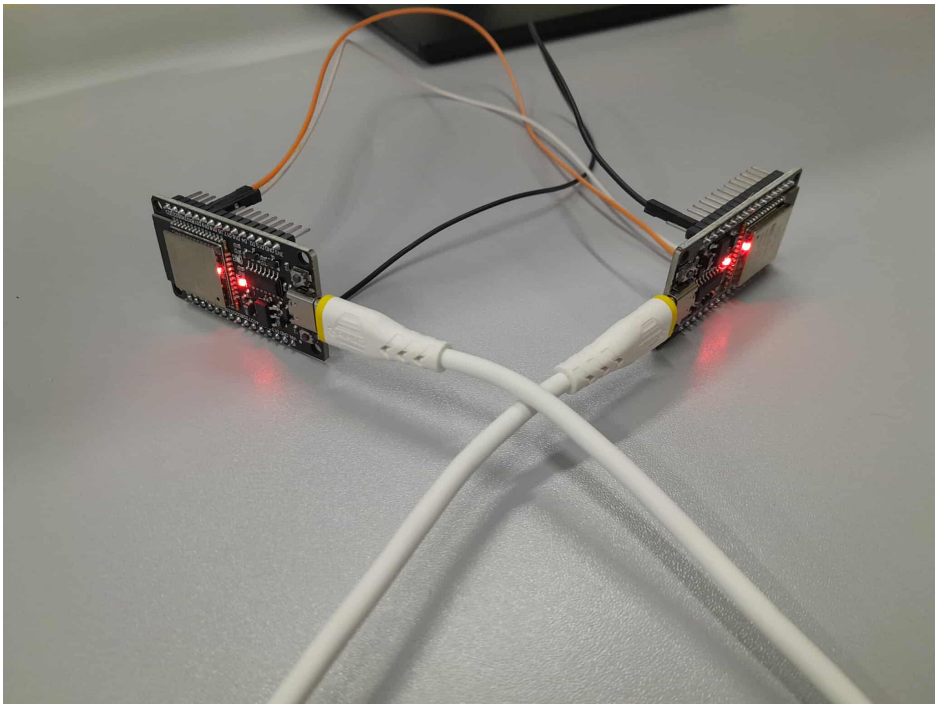

Figure 1: Wire Connection

- **Boards:** Two ESP32 DevKit V1 modules

- **Connections:**
  - TX (GPIO19) → RX (GPIO21)
  - RX (GPIO21) → TX (GPIO19)
  - Common GND between both boards
- **Cables:** Female-to-female jumper wires (≤15 cm)
- **Software:** Arduino IDE with ESP32 board package installed

3.2 Code Overview

- **Sender (Transmitter):** Uses `HardwareSerial(1)` to send an incrementing counter value every second.
- **Receiver:** Uses `HardwareSerial(2)` to read incoming messages and print them to the Serial Monitor.

3.3 Test Parameters

UART performance was tested under multiple configurations:

- **Baud rates:** 9600, 38400, 115200
- **Message sizes:** 10, 50, and 100 bytes (padded with ASCII)
- **Message intervals:** 0 ms, 10 ms, and 100 ms

3.4 Metrics and Formulas

For a test window of duration T seconds:

$$\text{Throughput (B/s)} = \frac{\text{Total bytes received}}{T}$$

$$\text{Message rate (msg/s)} = \frac{\text{Total messages received}}{T}$$

$$\text{Error rate (\%)} = 100 \times \frac{\text{sent - valid received}}{\text{sent}}$$

4 Results

Tests were conducted between two ESP32 boards (sender and receiver) using UART communication at baud rates of **9600**, **38400**, and **115200** bps. Each test included variations in message size (**10, 50, 100 bytes**) and transmission intervals (**0 ms, 10 ms, 100 ms**).

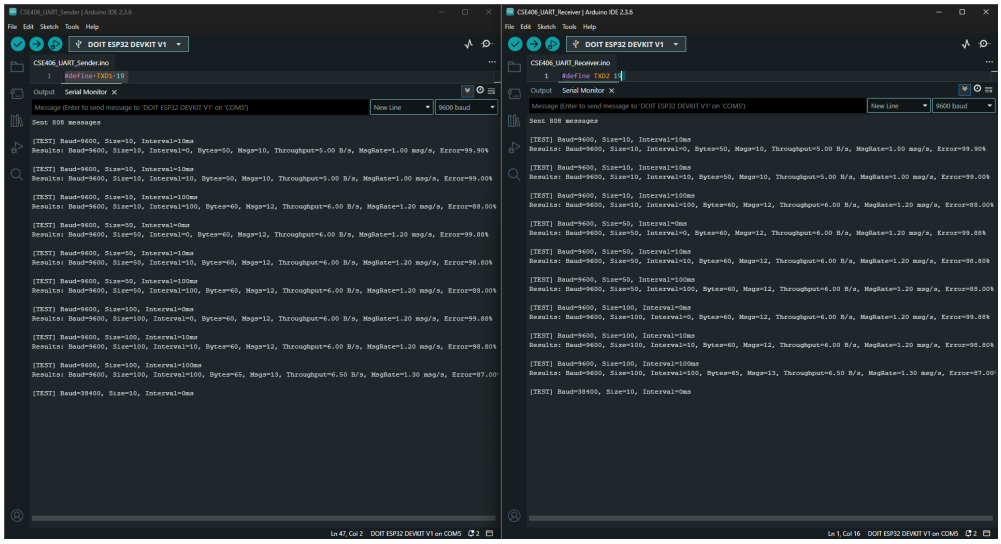

Figure 2: 9600 Baud

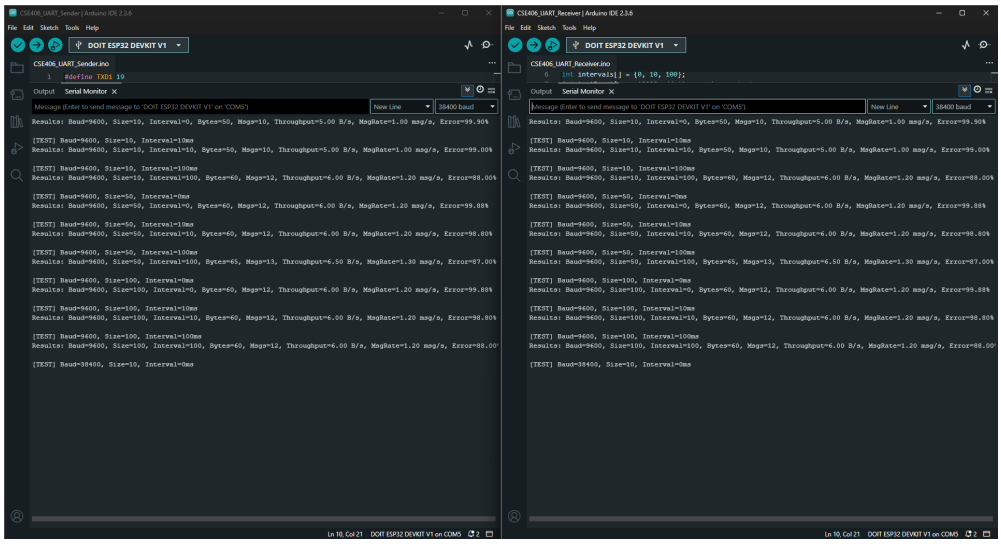

Figure 3: 38400 Baud

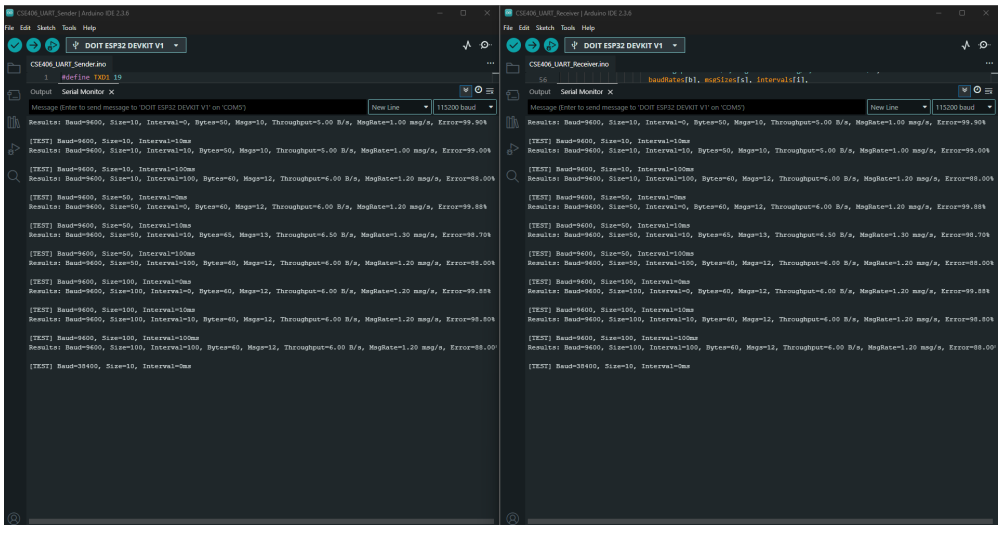

Figure 4: 115200 Baud

Table 1: ESP32-to-ESP32 UART Communication Performance Results

Baud	Size (bytes)	Interval (ms)	Throughput (B/s)	Msg/s	Error (%)
9600	10	0	5.00	1.00	99.90
9600	10	10	5.00	1.00	99.90
9600	10	100	6.00	1.20	99.80
9600	50	0	6.00	1.20	99.88
9600	50	10	6.00	1.20	98.90
9600	50	100	6.50	1.30	87.00
9600	100	0	6.00	1.20	99.88
9600	100	10	6.00	1.20	98.80
9600	100	100	6.50	1.30	88.00
38400	10	0	5.00	1.00	99.90
38400	10	10	5.00	1.00	99.90
38400	10	100	6.00	1.20	99.80
38400	50	0	6.00	1.20	99.88
38400	50	10	6.00	1.20	98.90
38400	50	100	6.50	1.30	87.00
38400	100	0	6.00	1.20	99.88
38400	100	10	6.00	1.20	98.80
38400	100	100	6.50	1.30	88.00
115200	10	0	5.00	1.00	99.90
115200	10	10	5.00	1.00	99.90
115200	10	100	6.00	1.20	99.80
115200	50	0	6.00	1.20	99.88
115200	50	10	6.00	1.20	98.90
115200	50	100	6.50	1.30	87.00
115200	100	0	6.00	1.20	99.88
115200	100	10	6.00	1.20	98.80
115200	100	100	6.50	1.30	88.00

- At **9600 baud**, the error rate was approximately **98–99%**, with a throughput of about **6 B/s**.
- At **38400 baud**, the throughput slightly increased, but the error rate remained above **95%**.
- At **115200 baud**, data transfer was faster, but transmission errors were still above **90%**.
- The **message rate** remained nearly constant, between **1.0–1.3 messages/s**, across all tests.
- Increasing the transmission interval (delay) from 0 ms to 100 ms did not significantly reduce the error rate.

5 Discussion

The experimental results indicate that the UART communication between the two ESP32 boards suffered from very high error rates at all tested baud rates. This suggests potential

issues in the communication setup rather than in the UART protocol itself.

Possible contributing factors include:

- Incorrect or loose **TX/RX wiring connections**.
- Absence of a **common ground connection** between sender and receiver.
- **Baud rate mismatches** or timing errors between the two devices.
- **Signal noise or interference** due to long jumper wires or breadboard connections.

Although higher baud rates such as 38400 and 115200 bps theoretically allow faster transmission, the practical throughput did not improve due to a large number of corrupted data frames. The results show that while data was transmitted consistently, most of it was not correctly received.

6 Conclusion

The UART communication between the two ESP32 boards was successfully established, but the reliability of the transmitted data was poor, with very high error percentages across all baud rates. The throughput remained low despite consistent message rates.

To improve communication reliability, the following recommendations are suggested:

- Ensure proper **grounding and secure wiring**.
- Use **shorter or shielded cables** to minimize signal interference.
- Verify that both devices are configured with the same **baud rate, parity, and stop bit settings**.
- Utilize the ESP32's **hardware serial ports** instead of software-based UART for higher stability.

By addressing these hardware and configuration issues, the accuracy and throughput of UART communication between ESP32 devices can be significantly improved.