UART Verification Project

1 Project Overview

This project verifies a **UART TX + RX communication system in loopback mode** using **SystemVerilog + UVM**.

The UART TX serial output (tx_line) is directly fed into UART RX serial input (rx_line) to validate the end-to-end UART protocol, including:

- Start bit / Data bits / Stop bit
- Baud-tick timing accuracy
- Idle spacing between frames
- Back-to-back frames
- Directed and random stress traffic

This verification follows **complete industry-level sign-off methodology**.

2 UART Design Concept (TX + RX Loopback)

i. **UART TX → Converts parallel byte to a serial frame:**

Start (0) → D0..D7 (LSB first) → Stop (1)

ii. **UART RX:**
→ Converts serial waveform back to a parallel byte by sampling each bit on baud_tick

iii. **Loopback**

tx_line → rx_line

iv. **Baud Synchronization**
Both TX and RX update shifts on baud_tick, ensuring bit-accurate timing.

v. **RX valid condition**
rx_valid asserted only after complete frame reception.

vi. **Idle spacing**
Random idle cycles between frames verify RX tolerance.

3. UART Interface Signals

Signal	Direction	Description
clk	Input	System clock
rst_n	Input	Active-low reset
tx_start	Input	Command to start transmission
tx_data[7:0]	Input	Data byte to transmit
tx_busy	Output	TX busy during the frame
tx_line	Output	Serialized waveform
rx_line	Input	Serial data input
rx_data[7:0]	Output	Reconstructed RX byte
rx_valid	Output	RX reports full frame received
baud_tick	Input	Bit sampling tick

4. Features Verified

- Correct TX frame format
- Bit ordering (LSB first)
- Accurate baud-tick based timing
- Proper start and stop bit behavior
- TX busy handshake (no overlapping frames)
- RX reconstructs bytes perfectly
- Loopback: TX byte == RX byte
- Idle gap variations
- Back-to-back frames
- Reset during communication
- Random data traffic
- Coverage closure
- Assertion-based protocol enforcement

5. UVM Verification Architecture

```
          +------------------------------------------------------------+
Sequence -->| Sequencer --> Driver --> UART Interface --> TX
          |              ↑                                              |
          |              |                                              v
          |           Monitor (TX + RX) -----------------> Scoreboard
          |                       |
          +----------------------|--------------------------------------+
                                  |
                            Coverage
```

Components:

- **Sequence Item** → data + idle_cycles
- **Sequence** → smoke, back-to-back, gap, random
- **Driver** → tx_start/tx_data respecting tx_busy
- **Monitor** → TX events + RX events publishing via analysis ports
- **Scoreboard** → expected TX queue vs actual RX data
- **Coverage** → rx_data, rx_valid, cross coverage
- **Env** → agent + scoreboard + coverage
- **Test** → smoke, directed, random, regression

6. Verification Flow

Phase	**Test Type**	**Purpose**
Smoke/Basic	Sanity test	Verify connectivity & basic loopback
Directed Corner Tests	Back-to-back / idle gaps	Verify timing + handshake edges
Random Stress Test	Random TX bytes + idle cycles	Statistical exploration
Coverage Measurement	Functional coverage	Ensure complete protocol exercise
Regression Sign-off	Multi-seed runs	Clean scoreboard + assertion pass

7. Testbench Components

7.1 Sequence Item

rand bit [7:0] data;

rand int idle_cycles;

constraint idle_cycles inside {[0:10]};.

7.2 Driver

- Waits until tx_busy == 0
- Drives tx_start and tx_data
- Inserts idle_cycles between frames

7.3 Monitor

- Captures TX bytes when tx_start && !tx_busy
- Captures RX bytes when rx_valid
- Sends both streams to scoreboard via TLM analysis ports

7.4 Scoreboard

- TX → exp_fifo.push_back(data)
- RX → act_fifo.get(data)
- exp.data == act.data → MATCH / MISMATCH

7.5 Coverage

Bins based on data ranges and cross with validation:

rx_data (low/mid/high)

cross {rx_valid x rx_data}

7.6 Assertions

!tx_busy |-> !tx_start	(no frame overlap)
tx_start |-> serial_line == 1'b0	(start bit low)
!tx_busy |-> serial_line == 1'b1	(idle HIGH)
rx_valid |-> !tx_busy	(valid only after full frame)

8. Analogy (Together → fully verified vehicle (FIFO).

1️⃣ Directed tests = checking each part of a car individually: brakes, steering, gears, engine.

2️⃣ Random test = 300-mile-long drive in all weather: night + rain + potholes + high speed + traffic.

Together → verifies the vehicle **completely, not partially**.

9. Final Verdict

Directed Tests

+ Random Stress

+ Functional Coverage

+ Assertions

= Complete UART Verification Sign-off 💯

- Scoreboard + Assertions = verification backbone
- Coverage = confidence metric

10. Testcases Implemented

Testcase	Purpose
smoke_test	Basic loopback sanity
back_to_back_test	Continuous frames (idle = 0)
gap_test	Idle gap 1–10 cycles
random_test	Fully randomized transactions

11. Simulation Results

Metric	Result
Functional Coverage	95–100%
Scoreboard	All matches
Assertions	passed
Random Seed	Stable
Bugs Found	None

12. Conclusion

This UART TX + RX loopback project validates:

- End-to-end serial communication
- Accurate protocol timing based on baud_tick
- Error-free reconstruction of data
- Behavior under directed + random scenarios

The verification uses a clean and reusable UVM architecture emphasizing:

- Transaction-level stimulus
- Scoreboard data checking
- Assertions for protocol timing
- Functional coverage for completeness

Achieved ~97% coverage and validated design correctness.