Verification of UART

Subhadip Sen

20th February, 2025

Contents

1. Introduction
2. Objective
3. Block Diagrams
4. FSM of different blocks
5. Testbench
6. Simulation Results

1. Introduction:

UART (Universal Asynchronous Receiver Transmitter) is a simple, widely used serial communication protocol that enables full-duplex data exchange between two devices using just two wires: **TX (transmit)** and **RX (receive)**.

Unlike synchronous protocols, UART does **not use a clock line**. **TX** and **RX** uses **local clocks** with nominal value, **not synchronized**. It relies on **start bits, stop bits, and baud rate** to frame and synchronize data. It is commonly used in microcontrollers, GPS modules, Bluetooth modules, and other embedded systems.

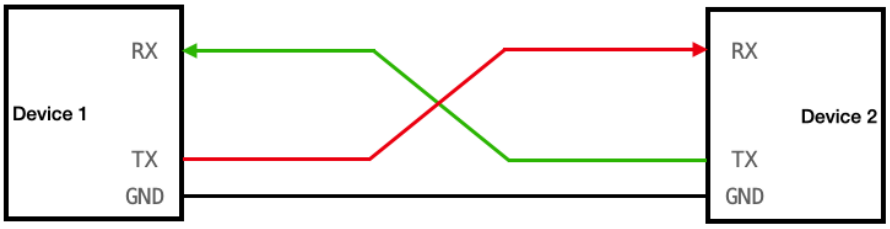


Fig 1: TX & RX

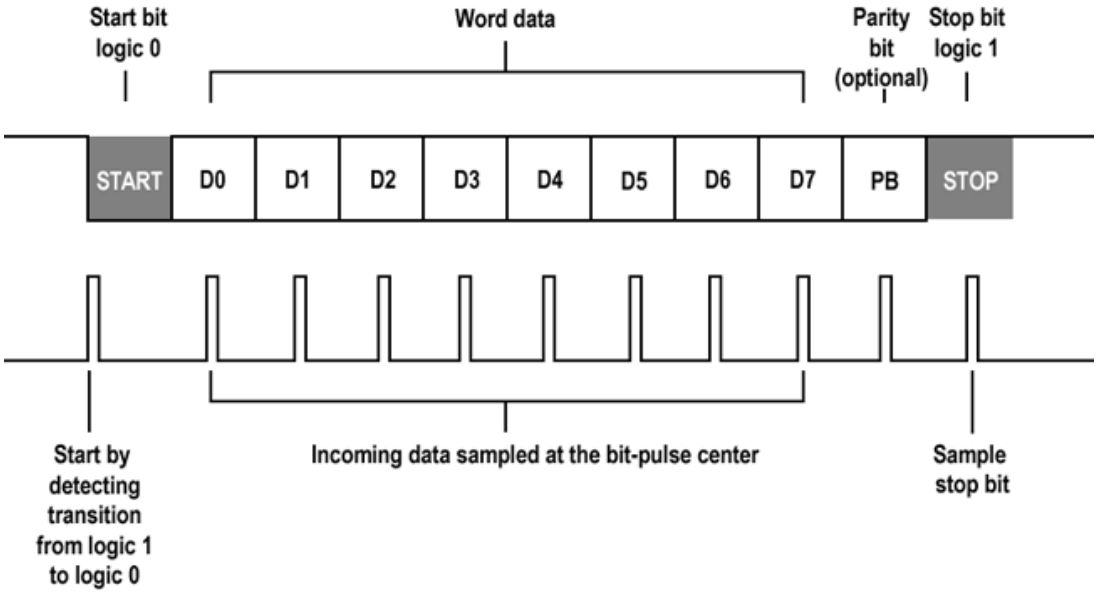


Fig 2: Frame Structure

2. Objective:

The objective of this project is to **verify a UART (Universal Asynchronous Receiver Transmitter) module** using System Verilog and UVM (Universal Verification Methodology). The testbench is designed to check functional correctness of UART transmission and reception logic under various scenarios, including:

- Valid data transmission and reception
- Handling of start and stop bits
- Detection of framing/parity errors
- Multiple data patterns and edge cases
- Configurable baud rate simulation

This project demonstrates a **complete verification flow**, including testbench architecture, and protocol-level checks.

3. Block Diagram:

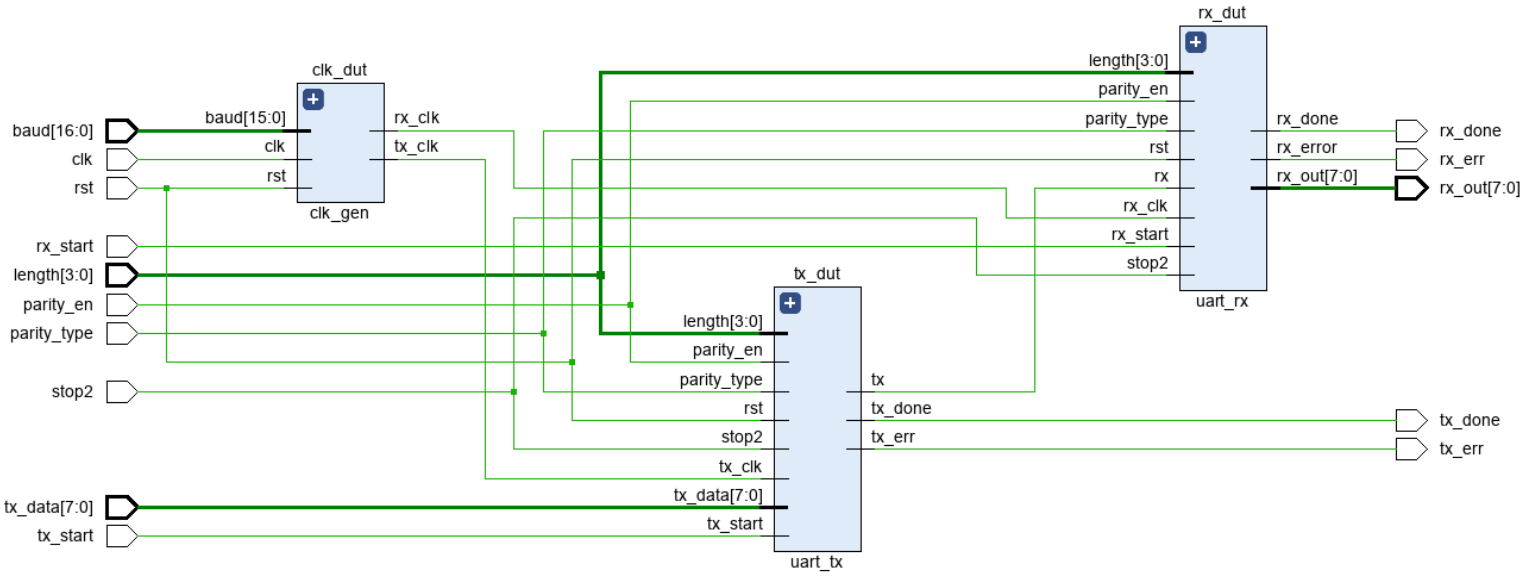


Fig 3: DUT block diagram

a. clk_dut:

The clk_dut block is responsible for generating timing signals for the UART design under test (DUT). It produces RX and TX clocks based on the configured baud rate to synchronize data transmission and reception.

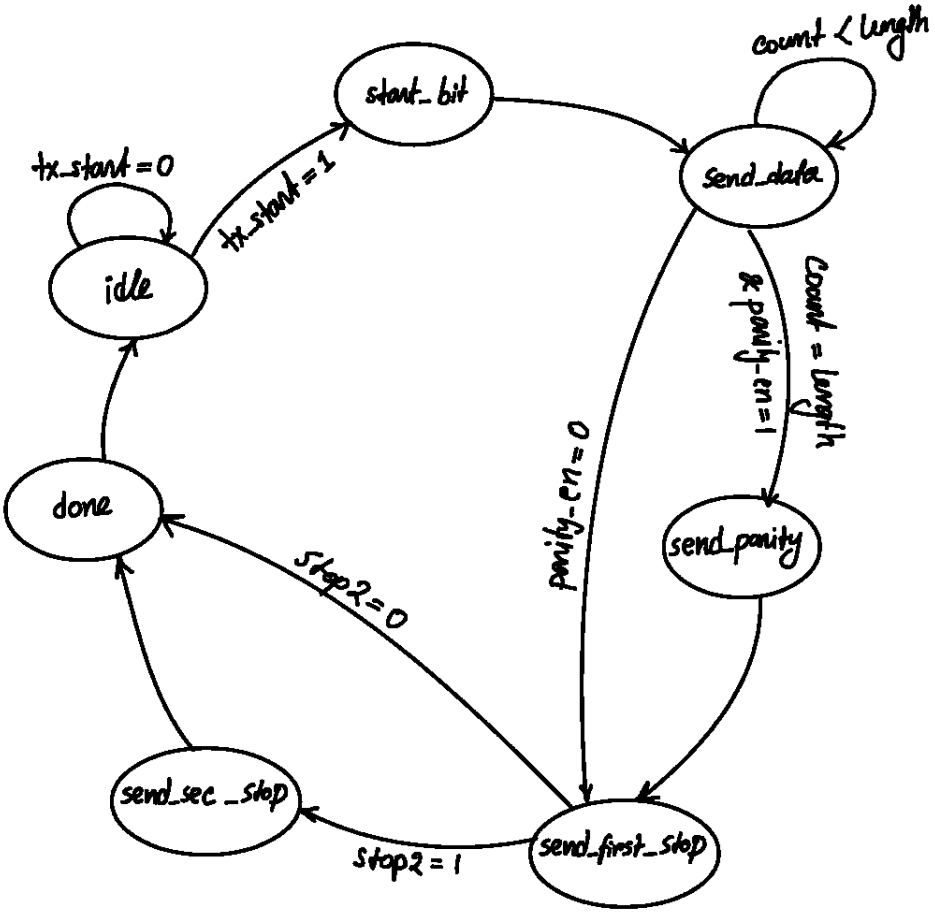
- It supports baud rates of 4800, 9600, 14400, 19200, 38400, 57600, 115200, 128000.
- Main clock frequency is assumed to be 50MHz.
- e.g. If baud is 4800, then divisor = floor [50M (clock frequency) / 4800] = 10416
  - ⇨ For 4800 baud rate TX clock will be high for 10416 + 1 (0 to 10416) main clock ticks and low for the same.

b. tx_dut:

The uart_tx block is responsible for **serializing parallel data** and transmitting it over the UART protocol with configurable frame format.

- **Data Bits**: Supports **5, 6, 7, or 8-bit** data transmission, based on configuration.
- **Parity**: Optional **parity bit**, configurable as **even** or **odd.**
- **Stop Bits**: Configurable **1 or 2 stop bits.**
- **Start Bit**: Always sends **1 start bit (logic 0)**
- **Idle Line**: Holds line at logic high (1) when not transmitting

FSM of TX:

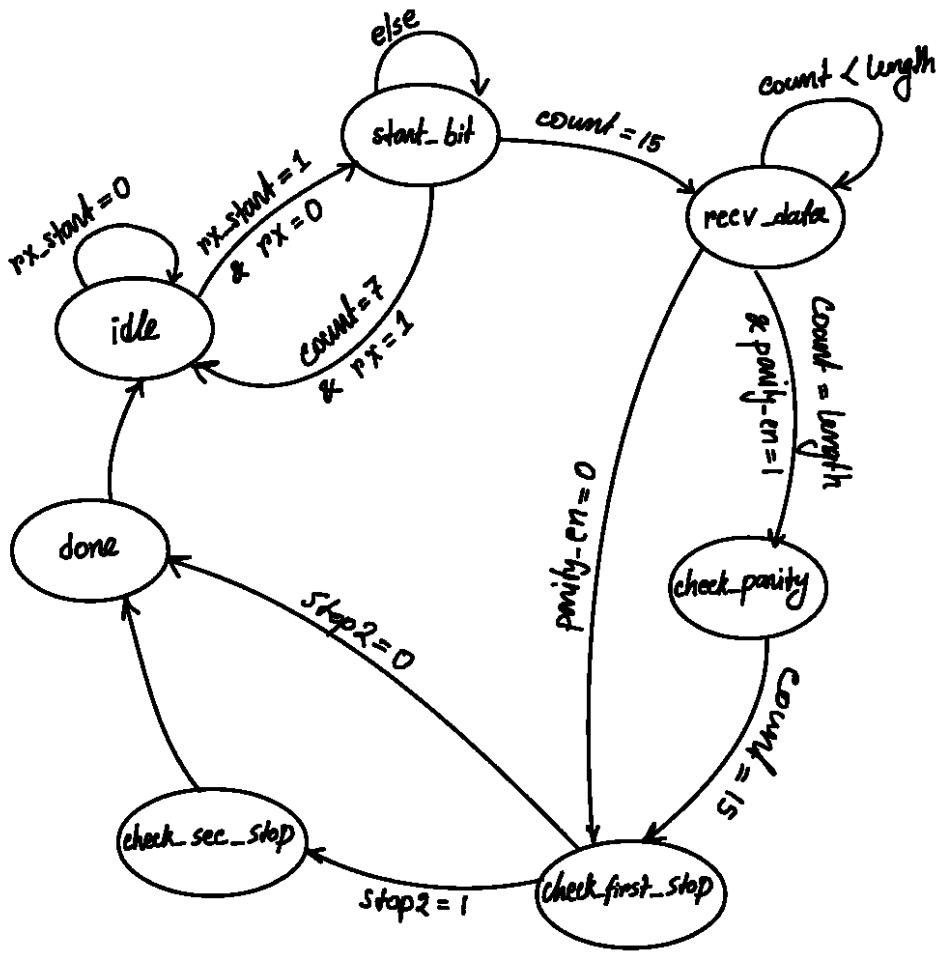


c. rx_dut:

The UART Receiver (RX) module is responsible for receiving serial data transmitted over the UART protocol. It detects the start bit, samples incoming data bits according to the configured baud rate, handles optional parity verification, and recognizes stop bits to complete a data frame. The received parallel data is then made available to the system.

- **Supports** 5, 6, 7, or 8 data bits
- Optional **parity check** (Even/Odd)
- **Configurable** number of **stop bits** (1 or 2)
- **RX** clock is **16x** of TX to increase timing accuracy
- **Frame error** detection (stop bit mismatch)
- **Parity error** detection (if parity is enabled)

FSM of RX:



4. Testbench:

This UART verification environment written using UVM methodology includes a comprehensive set of test cases designed to validate the transmitter (tx) and receiver (rx) functionality under various configurations.

The following scenarios were tested:

- baud: 9600, fixed length = 8, parity enabled, parity type: random, 1 stop bit
- random baud, fixed length = 8, parity enabled, parity type: random, 2 stop bit
- random baud, fixed length = 8, parity enabled, parity type: random, 1 stop bit
- random baud, fixed length = 5, parity enabled, parity type: random, 1 stop bit
- random baud, fixed length = 6, parity enabled, parity type: random, 1 stop bit
- random baud, fixed length = 7, parity enabled, parity type: random, 1 stop bit
- random baud, fixed length = 5, parity not enabled, parity type: random, 1 stop bit
- random baud, fixed length = 6, parity not enabled, parity type: random, 1 stop bit
- random baud, fixed length = 7, parity not enabled, parity type: random, 1 stop bit
- random baud, fixed length = 8, parity not enabled, parity type: random, 1 stop bit

Monitor captures the transmitter and receiver data, broadcast the packet through analysis port. **Scoreboard** receive the data from port implementation and compares the both data, reports the passing or failing status.

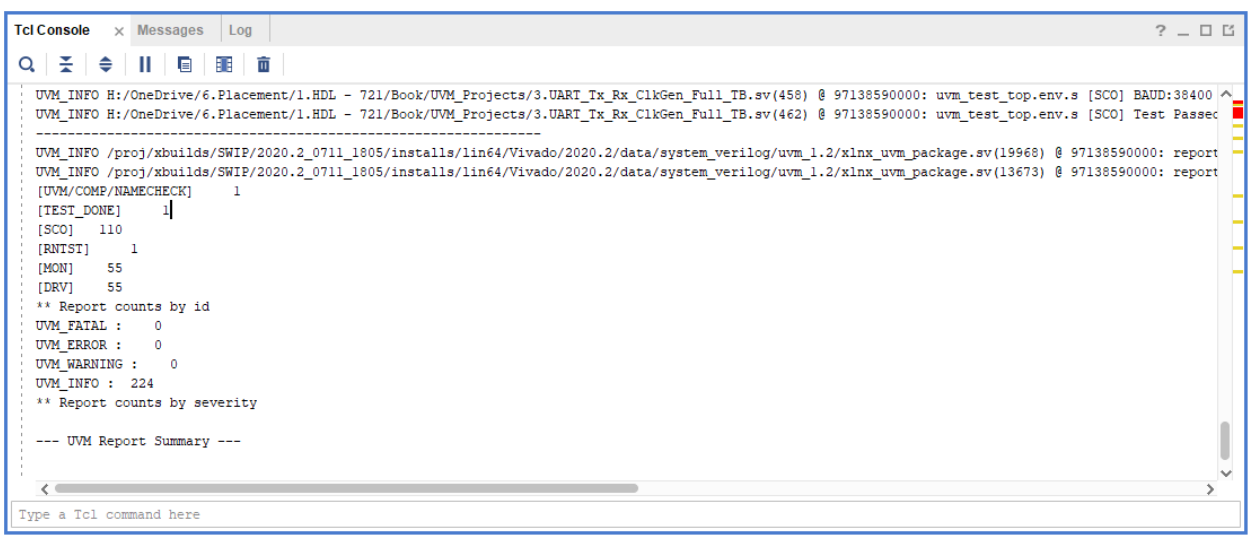


Fig 4: Simulator Output

Simulation results:

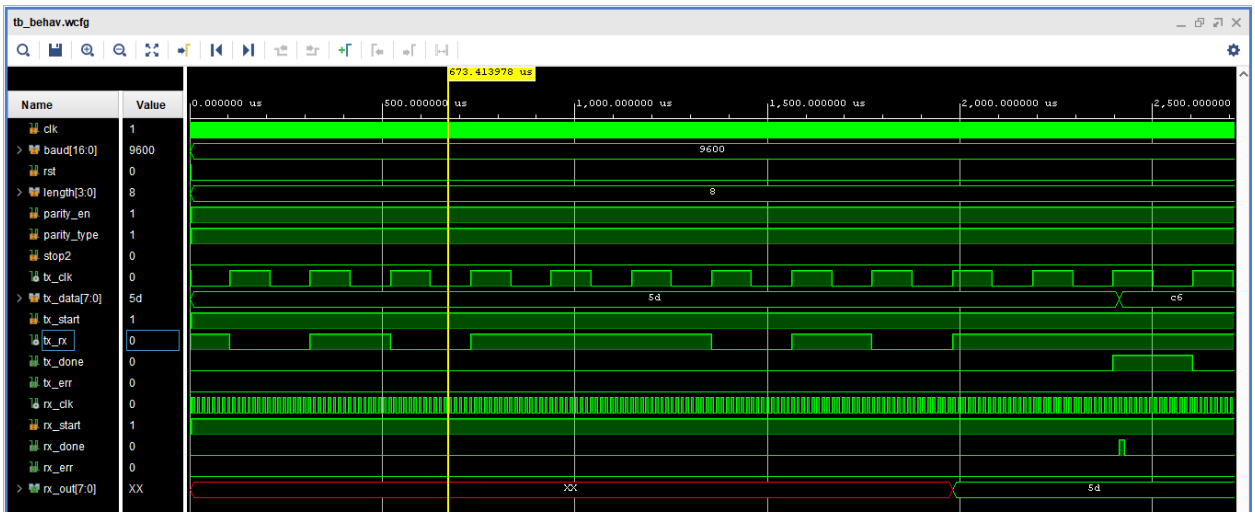


Fig 5: Single frame of transmission & reception

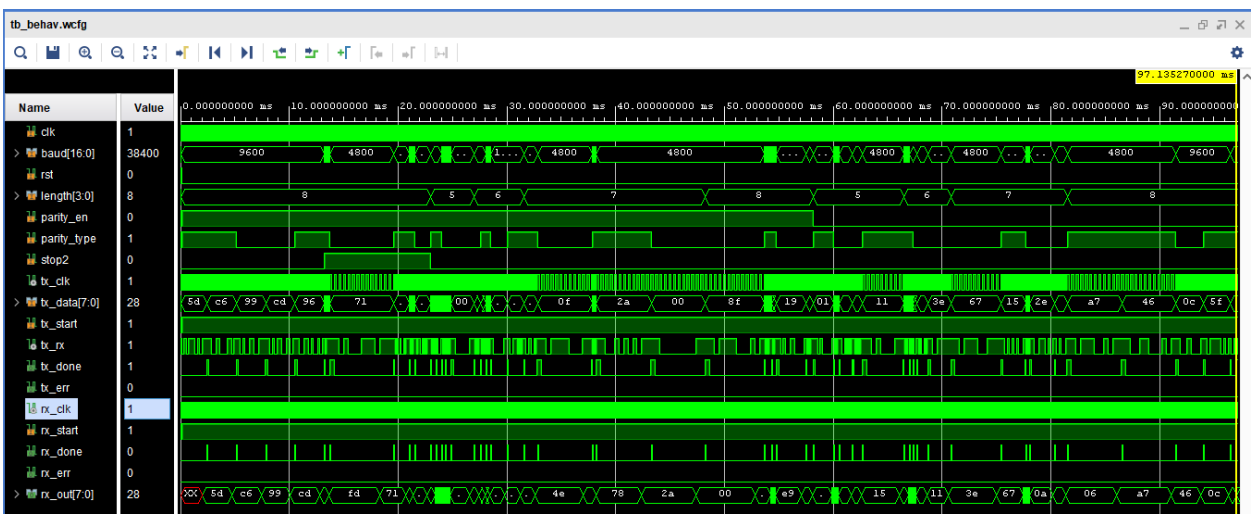


Fig 5: All sequence simulation result

It can be seen that all sequences are passing tx_err or rx_err does not indicate any error.

- System verilog Design & UVM verification environment written & simulated in Vivado.