UART Communication Module in Verilog

Course: Digital Systems & Design (DSD)

Title: UART Communication Module in Verilog

Team: Satyam Bhalotia (24BCI0055), Atharva Sheersh Pandey (24BCI0069), Aayush Jaiswal (24BCI0042)

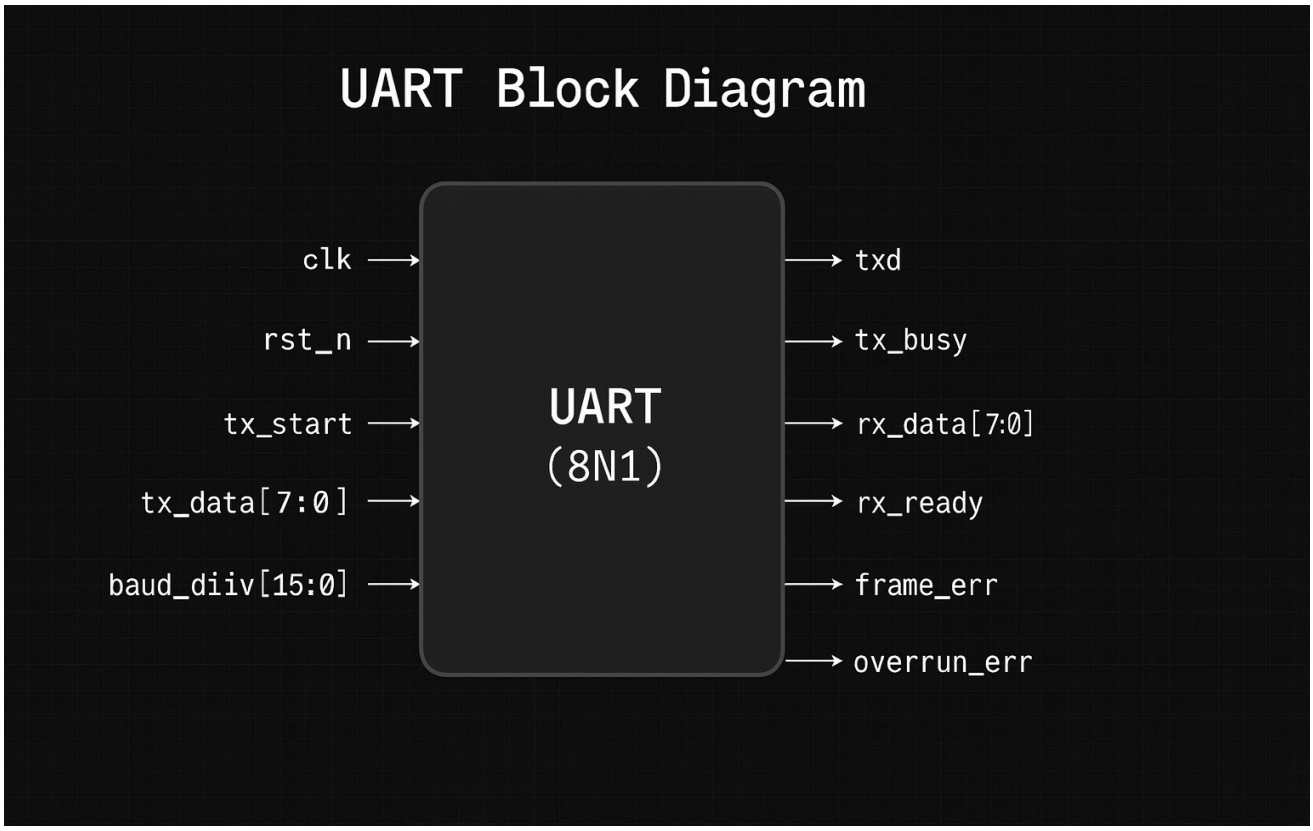
Objective

Design a compact, synthesizable UART (Universal Asynchronous Receiver/Transmitter) IP core in Verilog, verify it with a self-checking testbench, and benchmark its area/timing against common reference UART IPs. Primary goal: optimize for area without sacrificing basic UART functionality.

What is UART?

UART (Universal Asynchronous Receiver/Transmitter) is a simple serial interface that sends and receives data one bit at a time without a shared clock. Devices agree on a baud rate, then frame bytes with a start bit, 8 data bits, optional parity, and stop bit, converting between parallel and serial forms.

Diagram



Scope & Features

- **8 data bits – No parity – 1 stop bit (8-N-1)**. Parity can be enabled via parameter when needed.
- **Configurable baud via clock-enable tick** (baud-rate tick from system clock), e.g., **9,600–1,000,000 bits/s**.
- Separate **Transmitter (TX)** and **Receiver (RX) Finite State Machines (FSMs)**:
  TX serializes **start → data[7:0] (Least Significant Bit first, LSB-first) → stop**;
  RX employs **16× oversampling** with **majority voting** and **start/stop validation**.
- Clean interface through **Control and Status Registers (CSR)**: **TXDATA, RXDATA, STATUS, CTRL** (enable, baud select, optional parity).
- **Clock Domain Crossing (CDC)** & resets: **2-flip-flop synchronizer** on rx_in; **synchronous active-low reset**; **Register-Transfer Level (RTL)** written to be synthesis-friendly.

High-Level Architecture

- **Baud Tick Generator → TX Finite State Machine (TX FSM) → tx_out**
- **Baud Tick Generator + Oversampling → RX Finite State Machine (RX FSM) → rx_byte, status flags**
- Lean **Control/Status Register (CSR) block** for data transfer and configuration.

Verification Plan

- **Self-checking testbench**: random bytes, back-to-back frames, varying baud rates, **TX↔RX loopback**.
- **Assertions**: proper start/stop timing, no restart in middle of a frame, correct **busy/ready** protocol.
- Waveforms & logs: **Value Change Dump (VCD)** / **Fast SignalDataBase (FSDB)** for timing inspection; **pass/fail counters**.

Area Optimization Strategy

- Minimal baseline (no deep First-In, First-Out buffers (FIFOs)), shared counters/comparators.
- Tight Finite State Machine (FSM) encoding (consider one-hot vs Gray-code).
- Utilize constant-propagated parameters; eliminate redundant status logic.

Expected Deliverables

1. Verilog RTL (TX, RX, BaudGen, top + CSR).
2. Self-checking testbench with reports & waveforms.
3. Synthesis & P&R summaries (area/timing) plus comparison table.
4. Short write-up on trade-offs and area gains vs. reference IPs.