International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Development of a Reconfigurable Serial Communication Engine Based on UART Protocol

Mohammad Yusub[1], P. Aishwarya[2], M. Pallavi[3], Ch. Ranjith Kumar[4]

Department of Electronics and Communication Engineering, CMR Technical Campus, Hyderabad, India

Project Guide: **V. Mounika,** Associate Professor

Corresponding author email: [Add email before submission]

Abstract

Universal Asynchronous Receiver/Transmitter (UART) is widely used for serial data exchange between processors, controllers, and peripheral devices because it uses a simple interface and does not require a shared clock line. Conventional UART implementations often use predefined communication settings, which can limit adaptability when system requirements change. This paper presents the project design of a reconfigurable UART serial communication engine using Verilog Hardware Description Language (HDL). The architecture includes a transmitter, receiver, baud-rate generator, shift registers, and finite-state-machine (FSM) control logic. The intended configuration options include baud rate, data length, parity mode, and stop-bit selection. The supplied project material includes simulation waveform screenshots; exact measured performance values are not readable from the composite screenshot, so no numerical performance claims are made.

Keywords— UART, Verilog HDL, FPGA, serial communication, finite-state machine, configurable baud rate.

1. INTRODUCTION

In modern digital systems, efficient and reliable data exchange between microprocessors and peripheral devices is essential. Parallel communication requires more physical connections, while serial communication reduces the number of signal lines by transmitting data sequentially. UART is widely used because it offers a simple serial interface without a shared clock. This project focuses on a reconfigurable UART engine described using Verilog HDL, with configurable baud rate, data length, parity mode, and stop bits.

2. LITERATURE SURVEY

The project presentation reviews FPGA-based UART modules, configurable baud-rate generation, receiver design, microprogrammed controllers, built-in self-test, and error-control or cryptographic support. These approaches commonly separate transmitter, receiver, timing logic, and control. References and publication metadata should be verified against original sources before submission.

3. EXISTING METHOD

Conventional UART systems use selected baud-rate settings and a predefined frame format, typically relying on shift registers, counters, and FSM control. Fixed parameters may require design changes when communication settings change. Buffering and synchronization can also be concerns in systems with changing data rates or multiple clock domains.

4. PROBLEM STATEMENT

Fixed UART configurations can limit flexibility when devices use different data rates or frame formats. Limited buffering and error handling may contribute to data loss or transmission errors under some conditions. This project targets a configurable engine that maintains correct transmit and receive timing.

5. PROPOSED MODEL

The model uses Verilog HDL and contains transmitter and receiver paths, a baud-rate generator, shift registers, and FSM control. The intended configurable parameters are baud rate, data length, parity mode, and stop-bit count.

5.1 Transmitter section

The Transmit Hold Register (THR) temporarily stores parallel data. The Transmit Shift Register (TSR) converts it to a serial stream. Control logic sequences the start bit, data bits, optional parity bit, and stop bit or bits.

5.2 Baud-rate generator

The baud-rate generator provides timing enables. Its divider depends on the system-clock frequency and target baud rate. The supported range and timing error must be verified from the implementation.

5.3 Receiver section

The Receiver Shift Register (RSR) collects incoming serial bits after start-bit detection. Control logic checks parity and stop-bit timing where implemented. The Receiver Hold Register (RHR) stores the completed received word for the CPU or microcontroller.

6. BLOCK DIAGRAM DESCRIPTION

CPU → THR → TSR → TX line; RX line → RSR → RHR → CPU. Transmitter and receiver control logic manage frame operations, and the baud-rate generator supplies timing. A clean schematic export may be added if available.

7. RESULTS AND DISCUSSION

The supplied project image contains multiple simulation waveform views and a design-schematic view. The screenshots are included below as qualitative evidence of the design and simulation workflow. Signal labels and exact timing values are not sufficiently legible in the composite image to support numerical performance claims.

For a final journal submission, use separate high-resolution waveform exports and explain the input data, configuration, expected output, observed output, and pass/fail result for each test case.

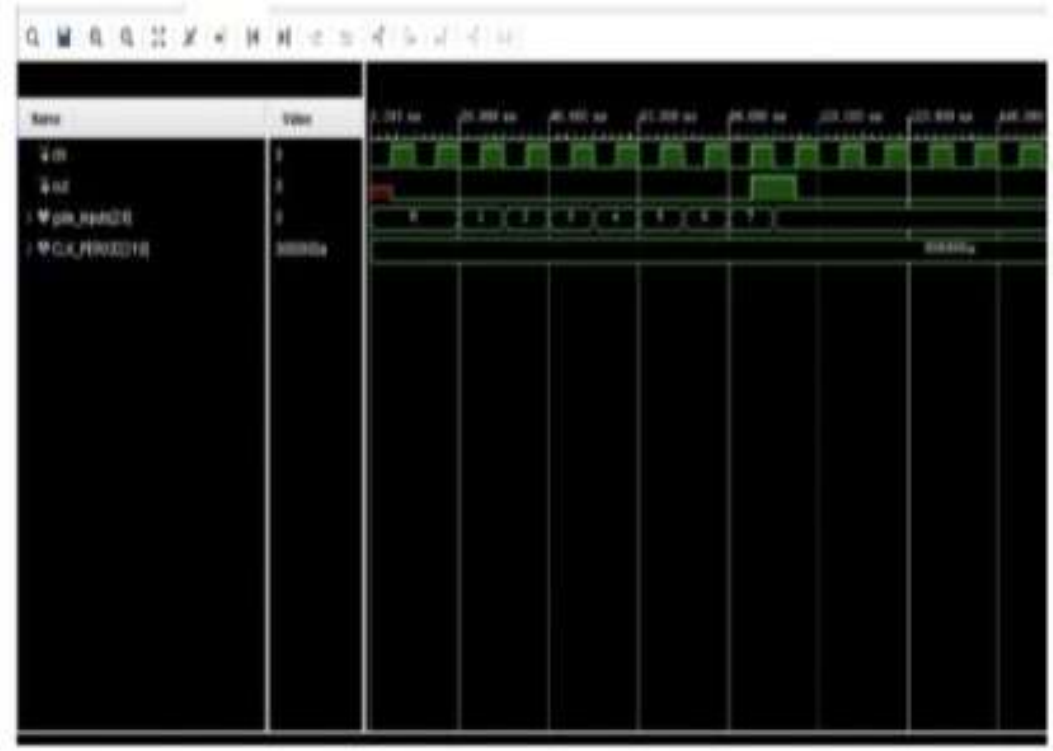


Fig. 1. UART simulation waveform — overview 1.

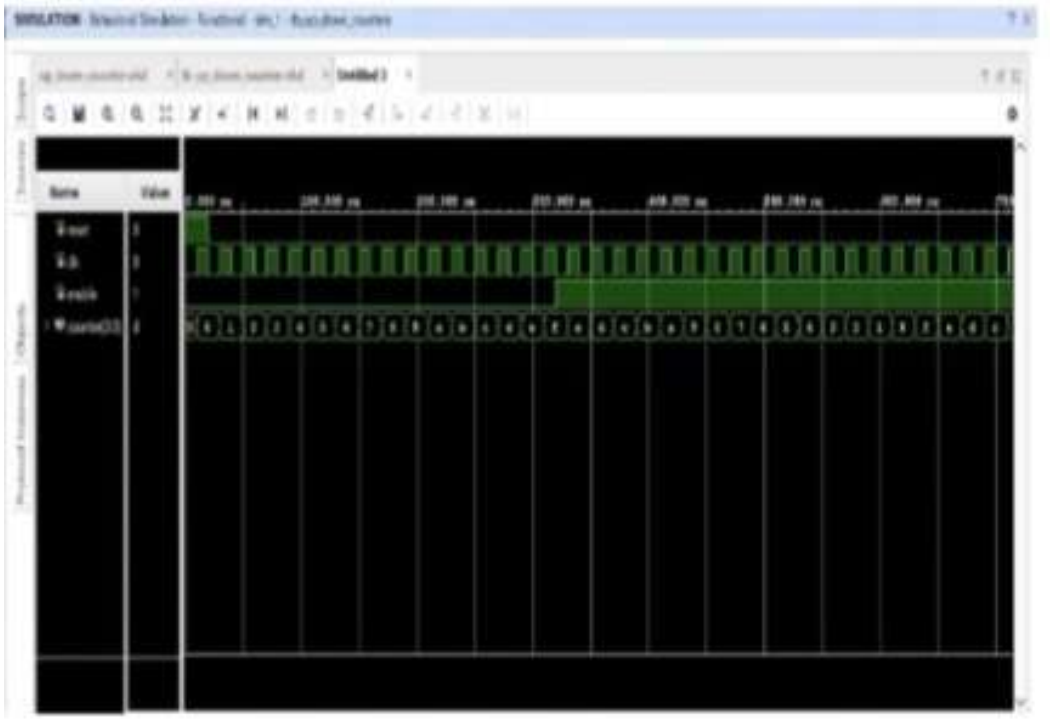


Fig. 2. UART simulation waveform — overview 2.



Fig. 3. Simulation signal view — screenshot 3.



Fig. 4. Simulation waveform — screenshot 4.



Fig. 5. Simulation waveform — screenshot 5.



Fig. 6. Detailed signal transition view — screenshot 6.

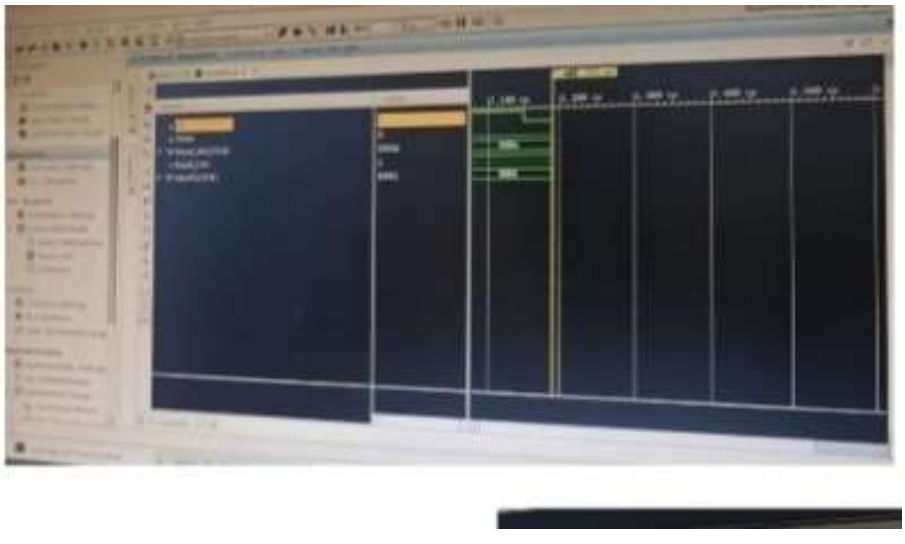


Fig. 7. Simulation environment and signal view.

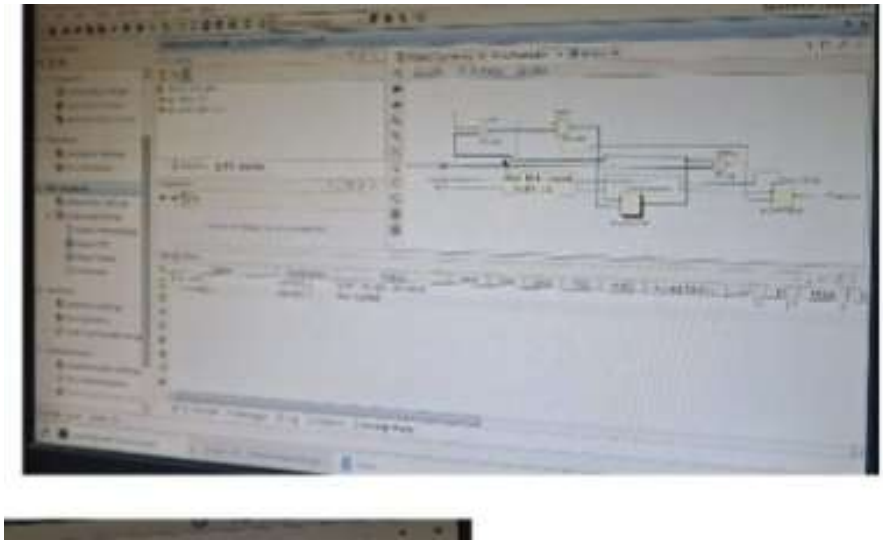


Fig. 8. UART design schematic and simulation environment.

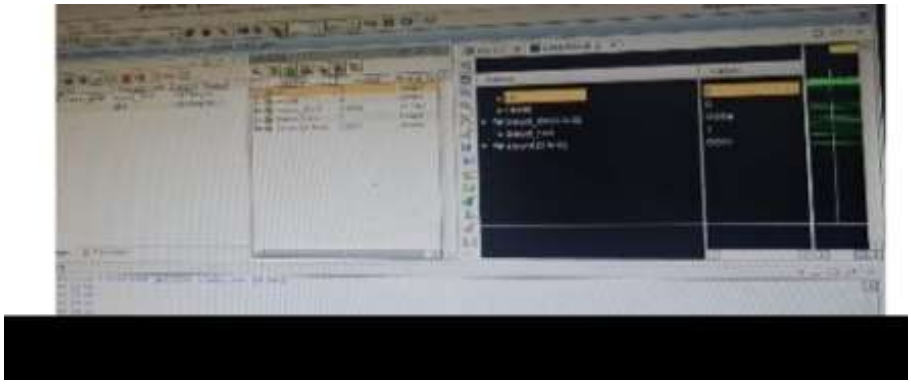


Fig. 9. Simulation workspace and signal view.

8. CONCLUSION

This paper describes a reconfigurable UART engine using Verilog HDL, with separate transmit, receive, timing, shift-register, and FSM-control blocks. Configurable communication parameters can make the interface adaptable to different embedded-system requirements. The supplied screenshots show waveform and schematic views, but detailed performance claims require readable waveforms and verified tests.

ACKNOWLEDGEMENT

The authors acknowledge the guidance of V. Mounika, Associate Professor, Department of Electronics and Communication Engineering, CMR Technical Campus. Confirm and edit before submission.

REFERENCES

[1] R. Kalyan Kumar et al., “Implementation of UART design using Verilog HDL to avoid data losses in transmission,” 2024 2nd International Conference on Networking and Communications (ICNWC), IEEE, 2024.

[2] P. Anuradha et al., “Enhancing high-speed data communications: Optimization of route controlling network on UART implementation,” IEEE Access, vol. 12, pp. 123514–123528, 2024.

[3] C. Naidu, “A 16-byte asynchronous receiver and transmitter using Verilog HDL for real time applications,” 2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT), IEEE, 2024.

[4] J. Cheng et al., "On the maximum UART size achieved in a class of constructions of optical priority transmissions," Journal of Communications and Networks, vol. 25, no. 4, pp. 429–439, 2023.

[5] S. Chaturvedi, M. N. Saranya, and R. Rao, "Design of Asynchronous Receiver for Asynchronous Transmission," 2022 International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), IEEE, 2022.

[6] D. Han and W. Chen, "Implementation of UART Module with Cryptographic Support: A cross-layer approach," IEEE Wireless Communications Letters, vol. 9, no. 9, pp. 1370–1373, 2020.

[7] S. Basu et al., "FPGA Implementation of UART with BIST Technique," 2019 4th International Conference on Electrical Information and Communication Technology (EICT), IEEE, 2019.

[8] K. Tsuru, "A Theoretic Universal approach to in-network management," IEEE/ACM Transactions on Networking, vol. 26, no. 6, pp. 2443–2456, 2018.

[9] Y. Sun et al., "Throughput modeling and analysis of random access in narrowband Internet of Things," IEEE Internet of Things Journal, vol. 5, no. 3, pp. 1485–1493, 2017.

[10] Masato and A. Lawi, "The impact of message transmission scheduling in DTN message delivery across multiple islands," 2017 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), 2017.