ECE 430/530 Digital Hardware Design
Project 02 - UART Design

Richard Groves

December 2, 2023

1 Overview

This project focuses on the design, verification, and implementation of a Universal Asynchronous Receiver Transmitter (UART) interface. The project's primary aim is to develop a UART module that can be tested and demonstrated on an FPGA prototyping board. The UART module designed in this project includes two key components: a transmitter and a receiver. The transmitter operates by reading a data word from a FIFO, and transmitting it in a serialzed manner. The receiver functions inversely, receiving serial data and recompiling it to be stored in a FIFO.

A key feature of the UART design is its ability to handle asynchronous data transmission. This method operates without a clock signal and ensures data integrity by including start and stop bits.

2 Design Procedure

2.1 UART Receiver (UART_receiver.v)

The UART receiver module is tasked with receiving and decoding the UART data. It extracts data bits, identifies start and stop bits, and calculates parity. This module plays a crucial role in detecting any errors in the received data stream.

2.2 UART Transmitter (UART_xmtr.v)

The UART transmitter module is responsible for handling data transmission, which includes the management of start/stop bits and an optional parity bit for data integrity. The module is designed to serialize the data bits and ensure correct framing for UART communication.

2.3 Top-Level UART Module (UART_top.v)

As the top-level module, UART_top integrates various submodules including the baud rate generator, transmitter, and receiver, along with FIFOs for efficient data handling. This module orchestrates the overall UART communication process, ensuring seamless data flow between the transmitter and receiver.

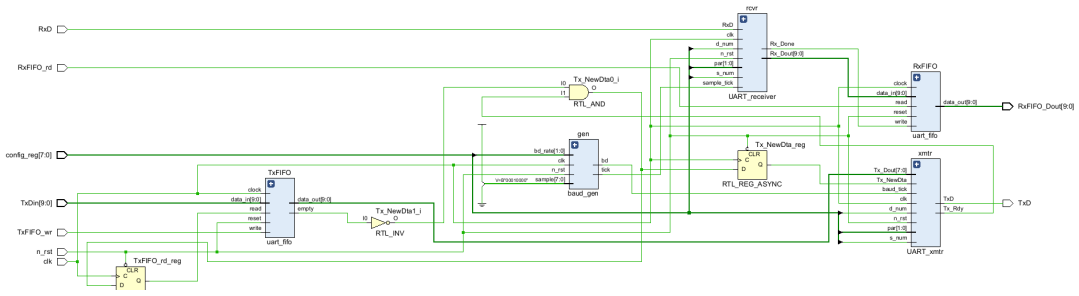

Figure 1: Complete UART System Schematic

2.4 Baud Rate Generator (baud_gen.v)

The baud rate generator module generates the required baud rate for UART communication. It divides the input clock frequency to produce a stable baud rate and includes a reset functionality.

2.5 Testbench for Baud Rate Generator (tb_baud_gen.v)

This testbench verifies the functionality of the baud rate generator by simulating the input clock and reset signals. It ensures that the baud_gen module accurately generates the desired baud rates.

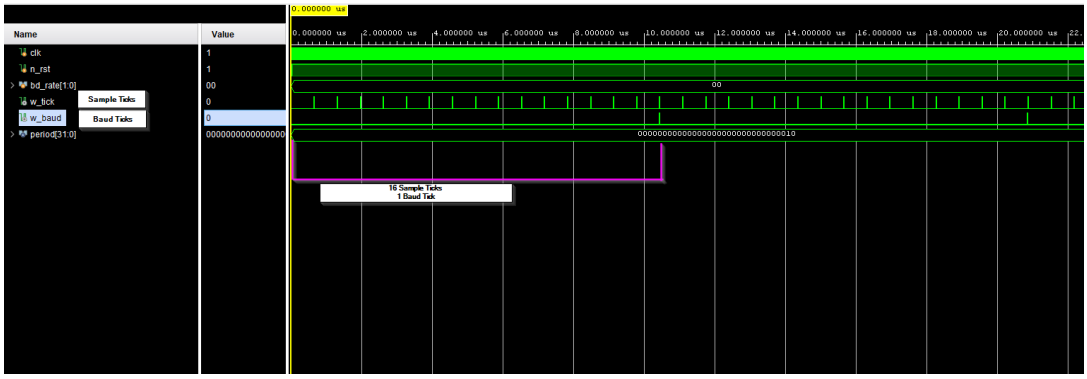

Figure 2: Simulation of Baud Rate Generator

2.6 Testbench for Top-Level UART (tb_UART_top.v)

This testbench module simulates the overall UART system's behavior, focusing on the functionality of the top-level UART module. In this testbench, tasks are employed to modularize and simplify the simulation process. Tasks in Verilog are used to encapsulate repetitive or complex operations into single, reusable blocks. These tasks can include stimulus generation for the UART module, response analysis, and specific functional checks. By using tasks, the testbench becomes more organized and maintainable, allowing for easy modification and extension of the test scenarios.

The testbench makes use of file operations for loading configurations and managing data. It starts by reading from a file to load configuration register values for the UART. Additionally, the testbench reads a set of four data words from the file. These data words are used to simulate the RxD signal. The same data words are placed into the TxD FIFO queue. At the end of the simulation, the testbench includes a process where everything stored in the RxD FIFO is read and written to an output file.

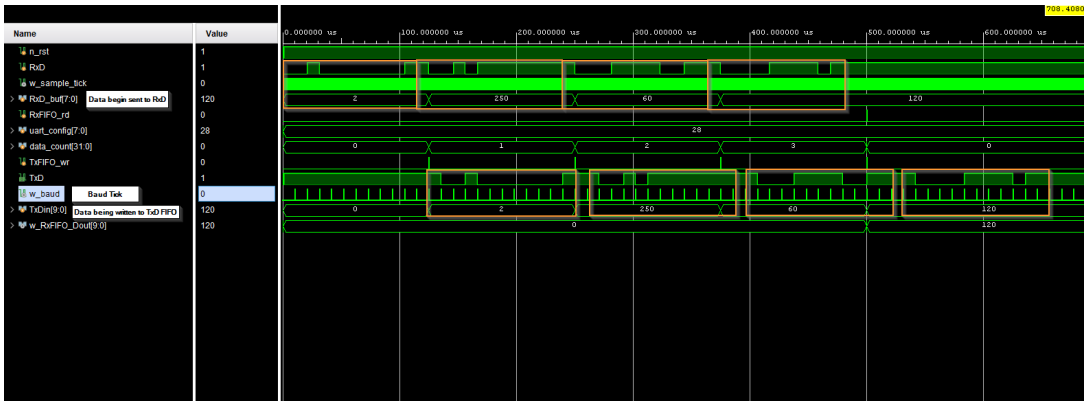

Figure 3: Simulation of Top-Level UART Module

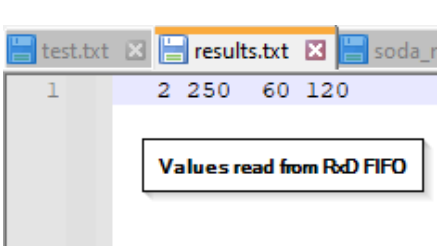

Figure 4: Values read from RxD FIFO

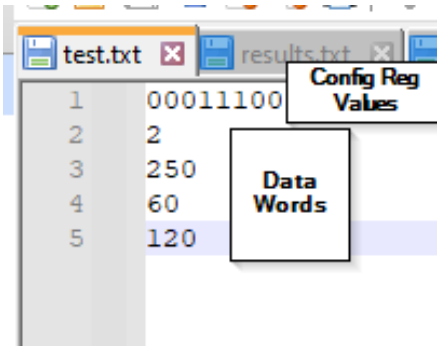

Figure 5: Config Register Values and Data Words

3 Implementation

The implementation of this project involved a unique approach, leveraging and repurposing code from a previous Soda Machine project. The project's architecture was designed to facilitate communication between two FPGAs using UART protocols.

3.1 FPGA Configurations

The first FPGA was programmed with the existing Soda Machine code, augmented with the UART transmitter module. This FPGA served as the control unit for the Soda Machine, handling user inputs and the corresponding logic for soda price selection, coin addition and soda dispensing. The second FPGA was programmed to drive an OLED display, incorporating the UART receiver module. This FPGA's primary role was to receive data transmitted from the first FPGA and use it to control the OLED display, showing information relevant to the Soda Machine's operations.

4 Conclusion

This project covered the design, implementation, and verification of a UART communication system. The testbenches were instrumental in ensuring the functionality and reliability of each component under various scenarios. I went beyond the project's scope by integrating a previous Soda Machine project with the UART modules, effectively dividing the code between two FPGAs. This endeavor served to demonstrate UART's versatility in real-world scenarios.