UART Half-Duplex Serial Port Module Design

Ekansh Bansal
Dept. of ECE
GLA University
Mathura, U.P., India

***Abstract*— This project explores the Universal Asynchronous Receiver/Transmitter (UART), a serial communication technique that eliminates the need of a clock to facilitate communication between two separate components on a device. UART is one of the most widely used communication interfaces that is crucial for microcontroller, computers, and packet sniffing. The main topic of this project is the design of a high-speed UART. The study begins by using Verilog HDL to describe the behavior of a UART. The focus will be on the transmitting and receiving of 1 byte data through the designed UART.**

Keywords—UART, HDL, packet sniffing

I. INTRODUCTION

The Universal Asynchronous Receiver-Transmitter (UART) is a computer hardware that allows for asynchronous serial connections and enables customization of data format and transfer speeds. It transmits data bits one at a time, starting with the least significant bit and increasing in significance, using start and stop bits to ensure accurate timing. In UART communication, two UARTs communicate directly with each other. The transmitting UART converts parallel data from a controlling device such as a CPU into serial data, which it then sends in serial form to the receiving UART. This receiving UART then converts the serial data back into parallel data for the receiving device. Only two wires are needed for data transfer between two UARTs, with data being transferred from the sending UART's Tx pin to the receiving UART's Rx pin as shown in a UART block diagram. UART makes asynchronous serial connections possible and allows for control over data format and transmission speeds. Data bits are transmitted one at a time, starting with the least significant bit and increasing in significance, start and stop bits are used to ensure accurate timing in the communication channel. Direct UART communication is used between two UARTs. After the transmitting UART converts parallel data from a controlling device like a CPU into serial form, the receiving UART changes the serial data back into parallel data for the receiving device. Only two wires are required for data transmission between two UARTs.

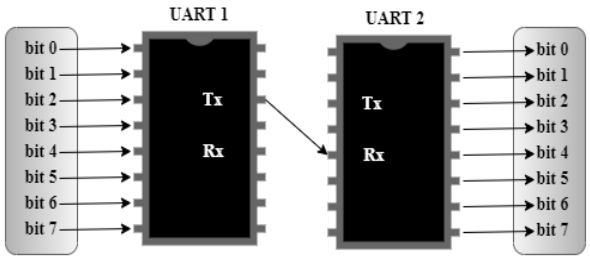

Figure 1: UART Block Diagram

II. TYPES OF UART

A. *Half-Duplex UART*

Half-duplex communication allows for two-way communication between parties, but only in one direction at a time. An example of a half-duplex device is a walkie-talkie, or a two-way radio with a push-to-talk button. It is also referred to as a semi-duplex design.

B. *Full-Duplex UART*

Full-duplex communication allows for simultaneous communication between both parties. An example of a full-duplex device is the common telephone service, where the microphone transmits the speech of the local party and the earphone reproduces the speech of the remote party. In more technical terms, there are two communication channels between the parties, allowing for two-way communication.

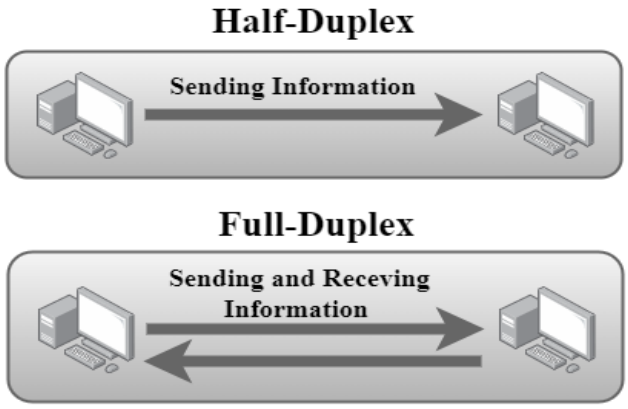

Figure 2: Full & Half Duplex UART

III. WORKING

The data bus is used to transport data to the UART from another device such as a CPU or microcontroller. The data is transmitted from the data bus to the transmitting UART in parallel. The transmitting UART adds a start bit, a parity bit, and a stop bit to the parallel data received from the data bus to construct the data packet. This data packet is then serially transmitted, one bit at a time, Tx pin as shown in Figure 3.

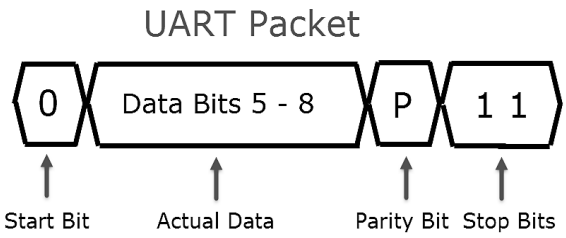

Figure 3: UART Packet Representation

This data packet or bitstream is divided into four parts as:

A. *Start Bit*

The start bit is the initial bit in a UART transmission. It indicates that the data line is transitioning from its idle state. The idle state is represented as logic high, therefore the start bit is logic low.

B. *Data Frame*

The data being transmitted is included in the data frame, which is composed of the start bit, data bits, parity bit, and one or more stop bits.

C. *Parity Bit*

At the transmitter, a parity bit is generated using a parity generator to check the accuracy of the data received at the receiver.

D. *Stop Bit*

The transmitter UART signals the completion of a data packet by transitioning the voltage on the transmission line from low to high for a minimum of two bit durations.

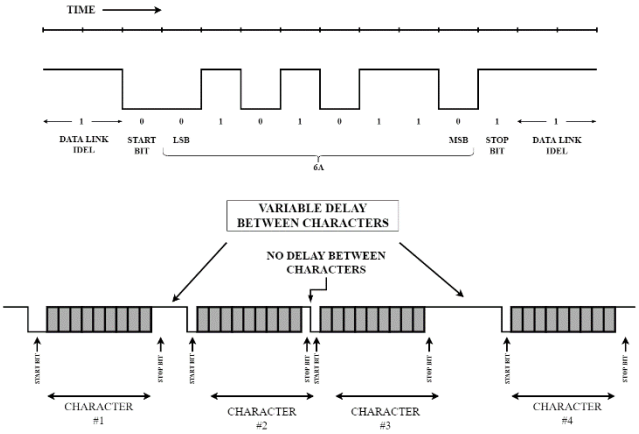

Figure 4: UART Working

The figure 4 illustrates how UART works. The bitstream indicates that when the start bit changes from high to low, data transfer begins and is transmitted bit-by-bit, starting with the least significant bit and ending with the most significant bit. Once the most significant bit is reached, the stop bit becomes high, indicating the completion of the transmission process. The same process is used at the receiver module to verify the accuracy of the received data. In this study, the UART is set to a baud rate of 115200 Hz to synchronize the transmission and reception, which is crucial for calculating the Clocks per bit. This can be calculated using the provided formula:

$$\text{CLKS\_PER\_BIT} = \frac{Frequency\ of\ i\_Clock}{Frequency\ of\ UART\ (Baud\ Rate)}$$

Frequency of clock: 10 MHz Clock

Baud Rate: 115200 Hz

$$\therefore \text{CLKS\_PER\_BIT} = \frac{10\times10^6}{115200} = 87$$

UART is consists of two modules as,

A. *Transmitter Module*

The design of the system is to transmit 8 bits of serial data, including one start bit, one stop bit, and no parity bit. Once the transmission is finished, the transmitter's idle line will be set to high for one clock cycle.

B. *Receiver Module*

This receiver is designed to receive 8 bits of serial data, including one start bit, one stop bit, and no parity bit. Once the reception is finished, the receiver's idle line will be set to high for one clock cycle.

IV. FSM MODEL FOR UART

This paper describes the design of a UART Half-duplex serial port module using Verilog HDL. The design is based on a Finite State Machine (FSM) model, which is a hypothetical machine with one or more states. Only one state can be active at a time, and the machine must change states to perform various tasks. The design utilizes two FSM models for better understanding and optimization of the module in terms of speed and power consumption.

A. *FSM for Transmitter (Tx_FSM)*

Total five states are used in transmitter module. Each state is designed to be followed by bitstream defined in Section III.

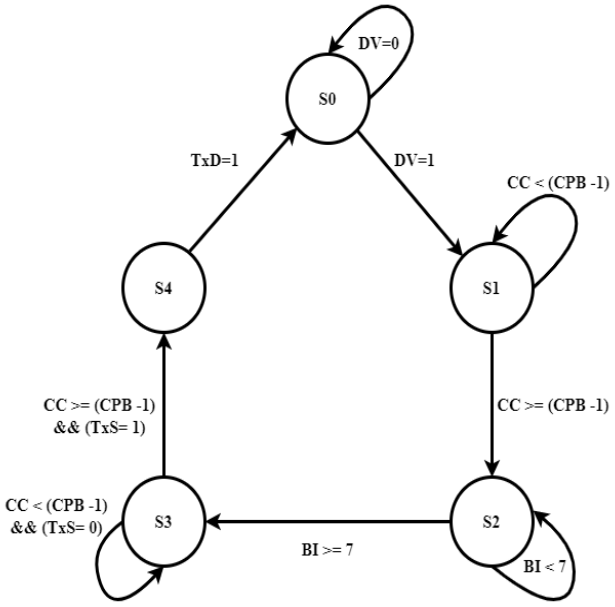

Figure 5: UART Tx_FSM

Where, notations are as:

S0: s_IDLE
S1: s_TX_START_BIT
S2: s_TX_DATA_BITS
S3: s_TX_STOP_BITS
S4: s_CLEANUP
DV: i_TX_DV
CC: r_CLCK_COUNT
CPB: CLK_PER_BIT
BI: r_BIT_INDEX
TxS: o_TX_SERIAL
TxD: r_TX_DONE

B. *FSM for Receiver (Rx_FSM)*

Total five states are used in receiver module. Each state

is designed to be followed by bitstream defined in Section III.

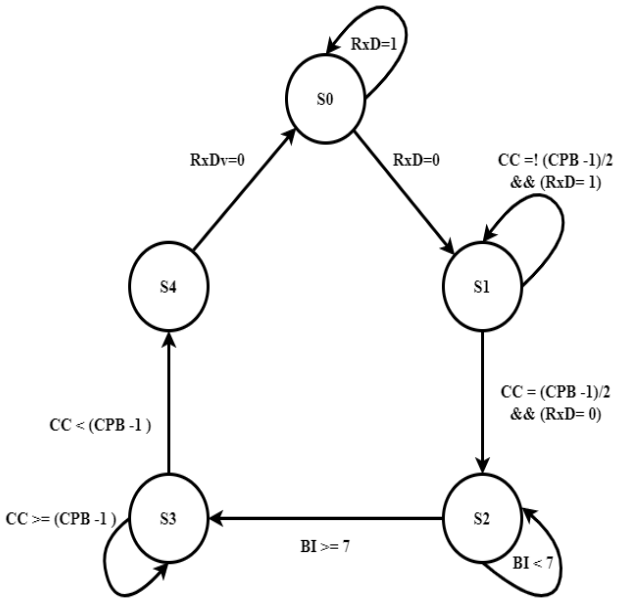

Figure 6: UART Rx_FSM

Where, notations are as:

S0: s_IDLE
S1: s_RX_START_BIT
S2: s_RX_DATA_BITS
S3: s_RX_STOP_BITS
S4: s_CLEANUP
RxD: r_RX_DATA
CC: r_CLCK_COUNT
CPB: CLK_PER_BIT
BI: r_BIT_INDEX
RxDv: r_RX_DV

In this paper, above shown FSM models are used to design UART Half-Duplex serial communication device. Designed for baud rate of 115200 Hz. This UART model is operated at 10 MHz internal clock.

V. SIMULATION RESULTS

In this paper, UART model is designed to transmit a 1 Byte data from one device to another at a high frequency. UART block diagram and RTL schematic which is generated is shown below:

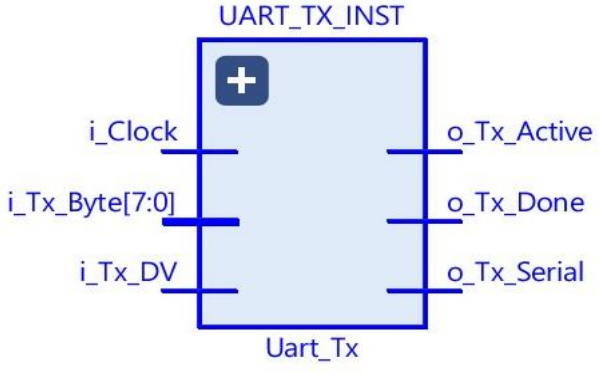

Figure 7: UART Transmitter outer block diagram

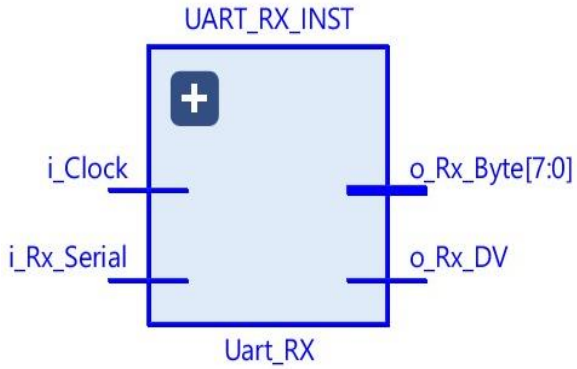

Figure 8: UART Receiver outer block diagram

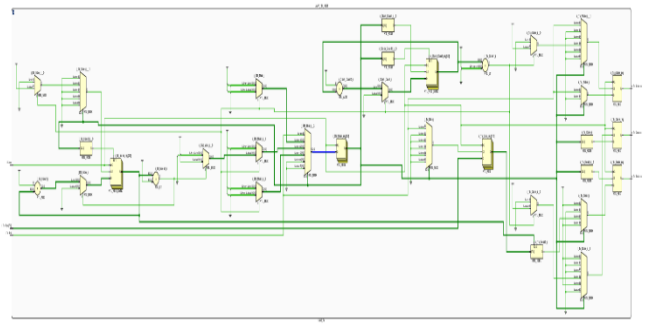

Figure 9: RTL Schematic of Transmitter

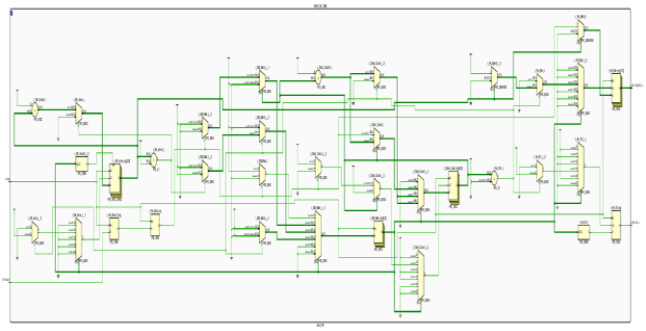

Figure 10: RTL Schematic of Receiver

Simulation result is displayed in Figure 11 tested for the "3f" (1 Byte) data transmission and it has successfully reached to the receiver side, then the result has been printed to the TCL console window as shown in Figure 12.

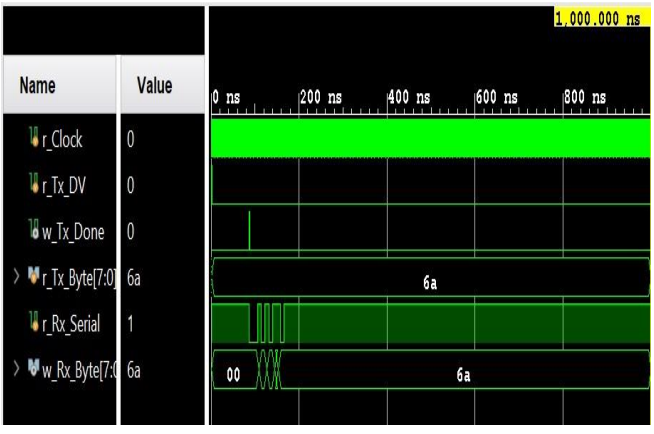

Figure 11: Simulation Result

```
# run 1000ns
Test Passed - Correct Byte Received
INFO: [USF-XSim-96] XSim completed. Design snapshot 'uart_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:04 ; elapsed = 00:00:11 . Memory (MB): peak = 754.238 ; gain = 6.785
```

Figure 12: TCL Console window message

Power analysis from Implemented netlist is shown in below in Figure 13.

Total On-Chip Power:	**0.122 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**25.2°C**
Thermal Margin:	59.8°C (31.6 W)
Effective ϑJA:	1.9°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	High

Figure 13: Power Analysis

VI. CONCLUSION

In this project, I focused on the basic design of UART using the Xilinx Vivado 2018.3 tool and Verilog HDL. The UART architecture was able to transfer data in a short time of 87ms. Furthermore, the power consumption of this UART design is 0.122 W and it has a thermal tolerance of 59.8 degrees Celsius.

VII. APPLICATIONS

- Establishing communication between 900m distance computers.
- Data transfer via serial port of a computer.
- The creation of baud rates for various purposes that aid in determining the rate of data transfer
- Wired data communication in a microcontroller.