UART Half-Duplex Serial Port Module Design

Pranavi Mallojjala
Dept. of ECE
VMTW,
Hyderabad,
India,
pmallojjala@gmail.com

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

Keywords—UART, HDL, packet sniffing

I. INTRODUCTION

A universal asynchronous receiver-transmitter (UART) is a piece of computer hardware that enables asynchronous serial connections and allows the user to customize the data format and transfer speeds. It sends data bits one at a time, beginning with the least significant bit and progressing in significance, employing start and stop bits to ensure that the communication channel can handle precise timing. In UART communication, two UARTs directly communicate with each other. The transmitting UART turns parallel data from a controlling device, such as a CPU, into serial data, which it then sends in serial to the receiving UART, It transforms serial data into parallel data for the receiving device. To communicate data between two UARTs, just two wires are required. Data transfers from the sending UART's Tx pin to the receiving UART's Rx pin as shown by UART block diagram in Figure 1. Asynchronous serial connections are made possible by UART, and the user can control the data format and transmission speeds. It sends data bits one at a time, beginning with the least important bit and increasing its significance. To ensure that the communication channel can support exact timing, start and stop bits are employed. Direct UART communication between two UARTs is used. After the transmitting UART has transformed the serial 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 needed for data transmission between two UARTs.

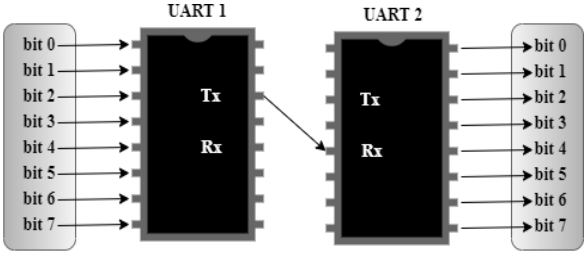

Figure 1: UART Block Diagram

II. TYPES OF UART

A. *Half-Duplex UART*

It allows for two-way communication between the parties, but only in one direction at a time. A walkie-talkie, or a two-way radio with a push-to-talk button, is an illustration of a half-duplex gadget. It is also known as a semi-duplex design.

B. *Full-Duplex UART*

It enables simultaneous communication between both sides. The common telephone service is an example of a full-duplex device. As the microphone transmits the speech of the local party, the earphone reproduces the speech of the remote party. In more precise terms, there are two communication channels between them, as there is a two-way communication channel between them.

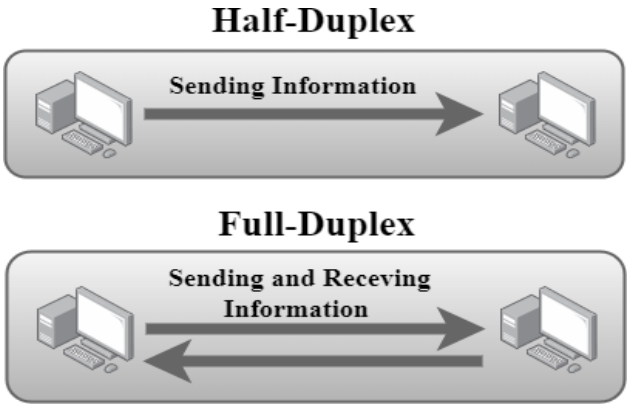

Figure 2: Full & Half Duplex UART

III. WORKING

A data bus transports data to the UART. The data bus is used to provide data to the UART from another device UART, such as a CPU or microcontroller. Data is transmitted from the data bus to the transmitting UART in parallel. After receiving parallel data from the data bus, the transmitting UART adds a start bit, a parity bit, and a stop bit to construct the data packet. The data packet is then serially and bit by bit sent to the Tx pin, is shown in Figure 3.

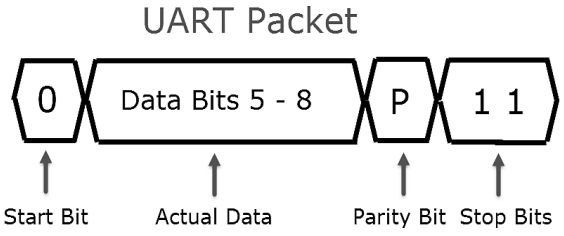

Figure 3: UART Packet Representation

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

A. Start Bit

It is the progress start bit of a UART transmission. It indicates that the data line is waking up from its idle state. Because the idle state is logic high, the start bit is logic low.

B. Data Frame

The data being transmitted is included in the data frame. The data frame is made up of the start bit, data bits, parity bit, and one or two stop bits.

C. Parity Bit

At Tx, an additional bit is created using a parity generator to determine whether the data received at Rx is accurate.

D. Stop Bit

As an indication that the data packet is complete, the Tx UART adjusts the voltage on the data transmission line from low to high for at least two-bit lengths.

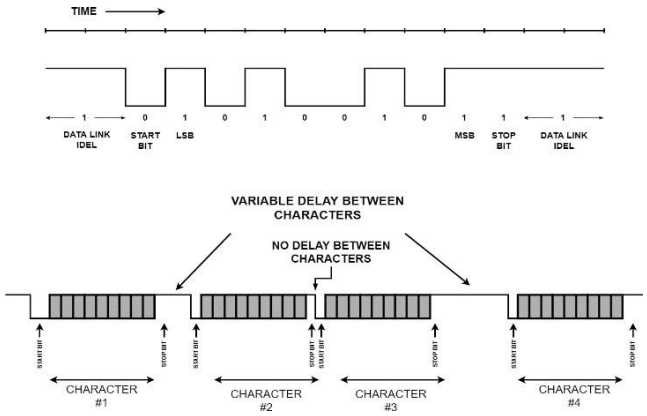

Figure 4: UART Working

Here, Figure 4 depicts a working of UART. Above bitstream shows that when start bit goes from high to low 1 byte data transfer starts and then the data will be transfer serially bit-by-bit from LSB to MSB. After reaching MSB, stop bit will get high an make data link idle high which indicate completion of transmission process. In the same way, at receiver module same process followed for checking whether the data received is correct or not.

In this paper, designed UART is operated at the baud rate of 115200 Hz to synchronize transmitting and receiving UART, which plays important role in Clocks per bit calculation. It can be find out by given 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

It is designed to transmit 8 bits of serial data, one start bit, one stop bit, and no parity bit. When transmit is complete transmitter idle line will be driven high for one clock cycle.

B. Receiver Module

This receiver can receive 8 bits of serial data, one start bit, one stop bit, and no parity bit. When receiver is complete, the receiver idle line will be driven high for one clock cycle.

IV. FSM Model for UART

UART Half-duplex serial port module is designed in this paper using Verilog HDL. This FSM (Finite State Machines) model is based on a fictitious machine that has one or more states. This machine can only have a single state active at a time. It implies that in order for the machine to carry out various tasks, it must change states. Here, two FSM models are used for better understanding and optimization of Module from the perspective of speed and power dissipation.

A. FSM for Transmitter (Tx_FSM)

A total of five states is used in transmitter module. Each state is designed to be followed by bitstream defined in Section III.

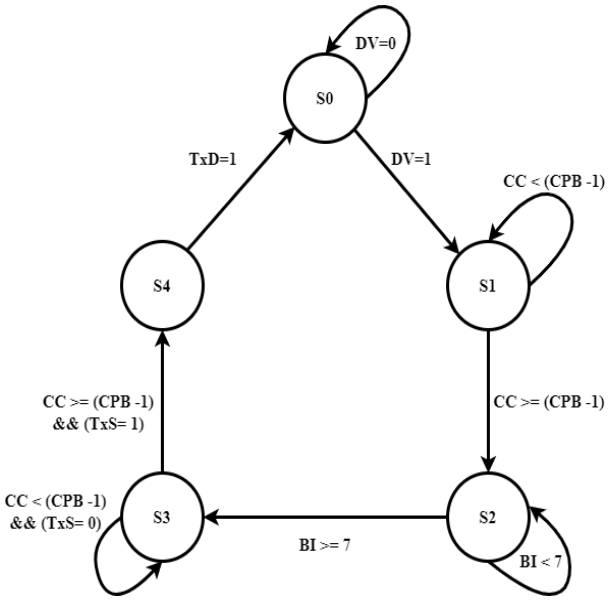

Figure 5: UART Tx_FSM

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.

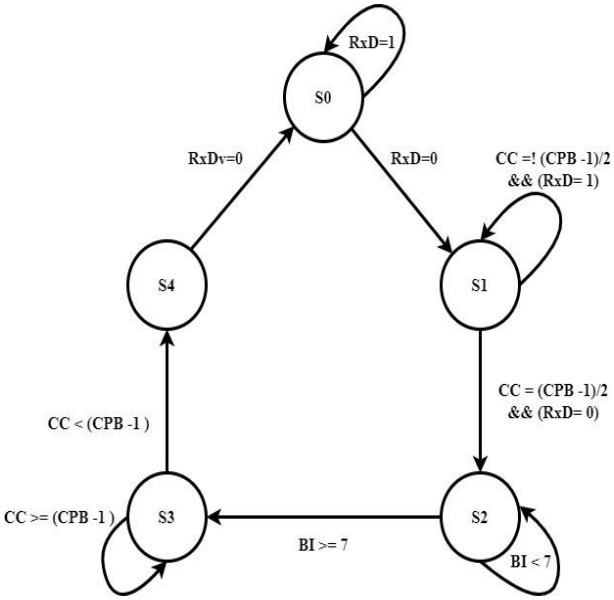

Figure 6: UART Rx_FSM

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:

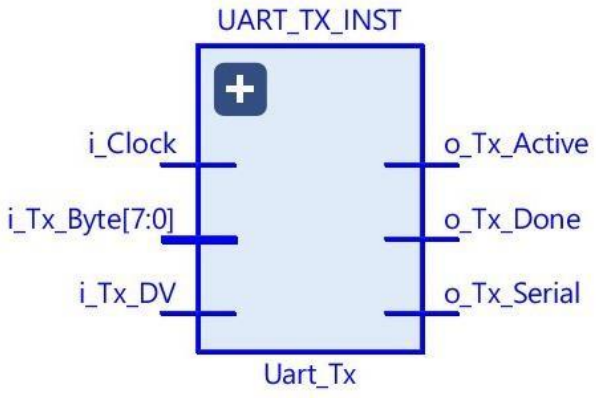

Figure 7: UART Transmitter outer block diagram

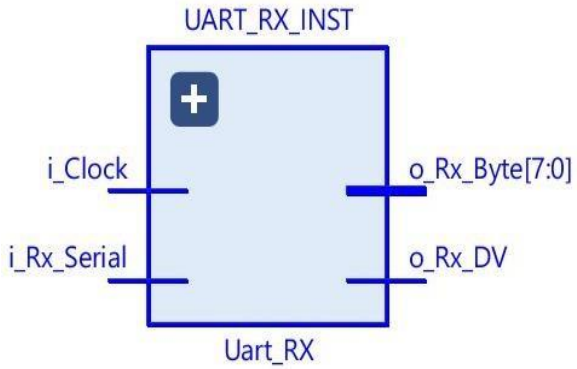

Figure 8: UART Receiver outer block diagram

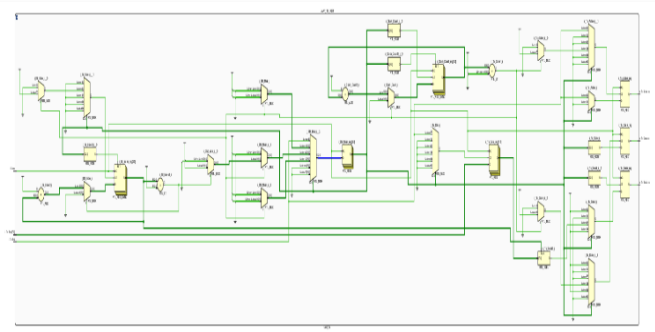

Figure 9: RTL Schematic of Transmitter

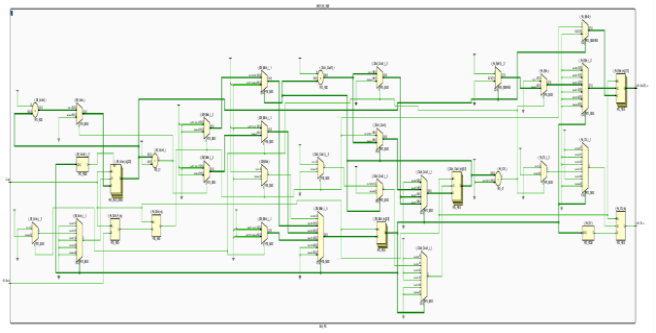

Figure 10: RTL Schematic of Receiver

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.

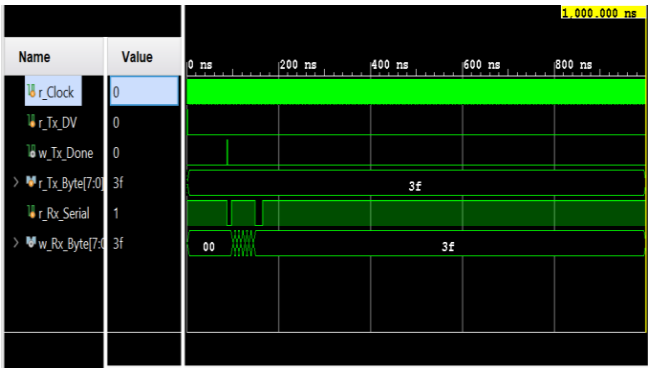

Figure 11: Simulation Result

```
# 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 = 00:00:01 ; elapsed = 00:00:06 . Memory (MB): peak = 2886.238 ; gain = 0.000
```

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 have worked on fundamental UART design using the Xilinx Vivado 2018.3 tool and Verilog HDL. This UART architecture has been shown to transfer data in as little as 87ms. Additionally, this UART design's power consumption is 0.122 W, with 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.