Design and Implementation of High-Speed Universal Asynchronous Receiver and Transmitter (UART)

MADDI BINDU
Department of ECE
Institute of Aeronautical Engineering
Telangana, India
21951a0436@iare.ac.in
0009-0009-5707-1014

YANAMALA ANAND
Department of ECE
Institute of Aeronautical Engineering
Telangana, India
21951a0417@iare.ac.in
0009-0004-8105-4840

KALLA SIDDHARTHA
Department of ECE
Institute of Aeronautical Engineering
Telangana, India
21951a04k0@iare.ac.in
0009-0001-9341-5365

P ANNAPURNA
Assistant Professor
Department of ECE
Institute of Aeronautical Engineering
Telangana, India
p.annapurna@iare.ac.in

Abstract:

The Universal Asynchronous Receiver and Transmitter (UART) are described, which is essentially a serial data transfer protocol used in digital circuit applications. The UART transmitter architecture has a baud rate generator, a parity generator, a transmitter finite state machine (FSM), and a parallel in serial out (PISO) register. The UART receiver is composed of a baud rate generator, a negative edge detector, a parity checker, a receiver Finite State Machine (FSM), and a serial in parallel out (SIPO) register. The transmitter and the receiver have the same baud rate generator; therefore, the transmitter/receiver baud rate is the same. The baud rate generator is the same as the frequency divider circuit. A UART transmitter data frame has 1 start bit, 8 transmit data bits, 1 parity bit, and 1 stop bit. The transmission rate of the transmitter and receiver is 4 Mbps using a 64 MHz system clock. Implementation, simulation and synthesis are done using Xilinx Vivado version 2018.2. The design is verified using a simulated waveform and synthesized on an FPGA Zed board.

Keywords: Finite State Machine (FSM), Parallel in Serial out (PISO), Serial in Parallel out (SIPO), FPGA

I. INTRODUCTION

A Universal Asynchronous Receiver and Transmitter is that kind of serial communication in which it allows devices to exchange data by converting parallel data from the microcontroller into serial format and vice-versa [13]. In the design and implementation of high-speed UART, optimal baud rate for maximum trans-mission of data is maintained with error-free and reliable data exchange through proper selection of baud rates, clock management, and inclusion of mechanisms to detect errors [11]. A good UART design will often enhance system performance, primarily for applications where quick and accurate data transfer is required in such areas as embedded systems and communication networks [7].

I.1 Background and Motivation

The increasing need for the efficient, reliable, and high-speed serial communication in all modern electronic applications is a call for the development of high-speed UARTs. System complexity, performance demands by an embedded system, an IoT device, and

other digital technologies are at such a level that traditional UARTs are not able to meet the data transfer rates necessary for the required level of efficiency [3]. High-speed UARTs address the challenges by enabling fast and reliable data transmission, providing reduction in latency values, and thus improving the overall system performance. The advancement ensures the devices can handle greater volumes of data and maintain robust communication, critical in real-time applications and complicated data exchange [12]. High-speed UARTs are versatile and support any device or protocol, which makes them a fundamental of the constantly changing world of internet of things and data-based technologies [15].

I.1.1 Overview of High-Speed UART

High-speed UARTs are advanced serial communication protocols that enable fast and effective data transfer by supporting higher baud rates compared to regular UARTs. Crucially important to the modern applications of embedded systems, IoT devices, and real-time processing, these UARTs boost system performance with lower latency, greater data throughput rates, and integrity of data through advanced features for error detection and correction [6]. The ability to be versatile and accommodate a wide number of devices and standards is crucial for flawless transmission in demanding environments such as automotive electronics, industrial automation, and many more [10].

I.1.2 Importance of High-Speed UART

High-speed UART (Universal Asynchronous Receiver-Transmitter) is crucial in modern electronic systems for several reasons: Increased Data Transfer Rates, Efficiency in Communication, Enhanced Performance in Complex Systems, Improved Reliability, Support for Modern Protocols, Future-Proofing, Reduced Latency, Flexibility in Design, Power Efficiency etc. They continue to play a significant role in the communication of fast and reliable data in modern electronic systems, linking different peripherals and parts, such as microcontrollers, sensors, and so on [14]. Their high baud rates allow fast transfer of data, which in many cases is vital in applications designed to interchange data in real-time-such as embedded systems, IoT devices, automotive electronics, and industrial automation. By reducing latency and increasing data throughput, high-speed UARTs improve the performance of overall systems and ensure efficiency. More importantly, they usually integrate advanced error-detection and correction mechanisms, thus ensuring data integrity at their higher speeds [2].

II. RELATED WORK

In literature, many authors present various algorithms and methods. A high-speed UART design sometime necessitates the alteration of the basic UART architecture to admit higher data rates with signal integrity and system reliability. Early works by McLaughlin (1997) and other authors were first concerned with issues of fundamental operations for the UART such as the concept of introducing baud rate settings along with the basic principles of the asynchronous communication. Advancement in high-speed UART designs has been focusing on the method of achieving the data transmission speed at the same time as signal quality. Fong et al. (2003) and Grönlund et al. (2008) study identifies the strategy for high-speed UART designs through advanced error correction techniques and using high-frequency oscillators [2].

High speed UART design approaches within this area of study include the usage of FPGAs and ASICs. A number of studies and research on high-speed UART systems are given by Lewis et al. (2010) and Ramachandran et al. (2014), and they provide information concerning the hardware implementation of the system [3]. With a growing demand for efficient and reliable serial communication in modern electronic systems, the design and implementation of a high-speed Universal Asynchronous Receiver and Transmitter (UART) is motivated. Advanced data integrity and reliability techniques for high-speed UART communications are researched by Liao et al. (2012) and Zhang et al. (2017) [4]. The development of technology continues to evolve continuously, which gives a stimulus for devices that can handle large volumes of data without diminishing the higher speeds

generated, power efficiency, or their reliability as well. Examples of applications are in the forms of practical implementations of UART systems, which are generally in telecommunications, data acquisition, and embedded systems. A high-speed UART addresses those needs through a flexible communication solution that can easily adapt to various system architectures. The work of Kumar et al. (2016) and Wang et al. (2020) explains real-world applications and practical challenges encountered in the implementation of high-speed UARTs [5]. In the earlier design, the fixed frame length had to be transmitted but this method adds an additional clock line that indicates the start and end & saves the time. In addition, the clock line can also inform the receiver to repeat the last transmitted byte, so it increases the speed of communication [6].

This Approach includes switching activity, gate switching activity factor power consumption in UART. It determines actual number of power-consuming voltage transitions faced by the output capacitance in each clock cycle [7]. This Approach will check the receive data with error-free & baud rate generation at different frequencies. Verilog methodology to be developed testbench of UART & timing value of given verified operation of data correctly passed or not by mentor graphics tool [8]. This implementation includes Different Techniques like UART integration in OR1200, UART with Auto Tuning Baud Rate Generator and Asynchronous FIFO, Multi-UART Controller, Advanced Encryption Standard (AES) algorithm in UART, 9 bit UART Module [9]. Advanced research on design of high-speed UART system on embedded application: Such research would be on the architectural designs usable in high-speed communication sceneries, balancing performance, reliability, and cost [10].

III. PROBLEM STATEMENT & OBJECTIVE

With modern digital communication systems requiring high-speed data transmission for such applications like real-time video streaming and high-frequency trading, conventional UART interfaces fall behind these standards because they are limited to both baud rate, data integrity as well as latency. In designing a high-speed UART protocol that goes beyond the convectional standards, signal integrity must be addressed, and the transmission delays should be minimized and proper data synchronization achieved. Actually, while designing and implementing a high-speed UART, there are several key objectives which become quite crucial in meeting the demands of advanced communication systems.

The rate of data transfer should be as high as possible, thereby supporting fast data exchange & minimizing the latency in data transmission to ensure timely communication for some real time applications. Data integrity should be ensured by robust error detection and correction mechanisms, and above all, it should optimize the power consumption in view of energy efficient and battery-operated devices. The UART must ensure ease of integration with other systems by referring to standard interfaces and protocols, increase overall system throughput, and provide scalability and flexibility to suit possible upgrades and various applications.

IV. METHODOLOGY

UART stands for Universal Asynchronous Receiver and Transmitter, that is one form of serial communication that transfers data over short distances at low speeds. The way UART works may be simplex, half duplex, as well as full duplex; in such a case, the communication may either be unidirectional or transmit and receive simultaneously. UART is the interface between slow speed and high-speed peripherals, wherein parallel data received by the computer are converted into serial data to be sent out, and at the receiving end, the serial data is changed back into parallel data. It is called asynchronous, because in this mode of device, there is no common clock shared between transmitter and receiver to govern data transmitted but uses start and stop bits. This is also called as 'universal' device because the speeds of transmission as well as the format of data can be configured as per requirement. UARTs can be configured to operate at various transmission speeds and data formats; however, one of the most common baud rates uses 1200, 2400, 4800, 9600, and 115200 bps. The protocol frames the data through start bits, stop bits, & optional parity bits so that data integrity

can be assured, and it is a versatile solution for many serial communication applications.

Working

i) Pin Description

In this work, the architecture of the transmitter and receiver block is implemented, simulated and synthesized using Verilog hardware descriptive language. The architecture and pin description of the UART transmitter and receiver are shown in fig 1. Fig. 1 shows the interface diagram of both the transmitter and receiver of the UART protocol.

Fig.1 (a) shows the interface diagram of transmitter which has a system clock, an active-low reset signal, transmits enable, transmits input data signal, even or odd parity select signal as the i/p signal & busy and serial out data as the o/p signal. Serial data i/p is 8 bits data and remaining is 1 bit.

Fig. 1(b) shows the interface diagram of the UART receiver which has system clock, active low reset, serial data in as input and parallel data out, data valid and busy signal as the output. Parallel data output is 8 bits data and remaining signal is 1-bit data.

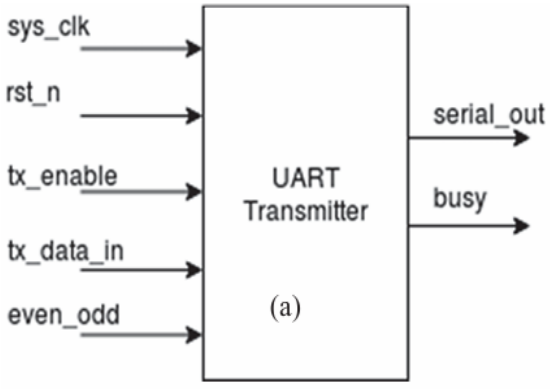

Figure-1(a): Transmitter of UART

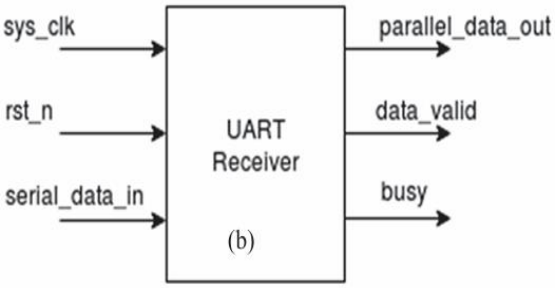

Figure-1(b): Receiver of UART

ii) UART Data Frame and Data Transmission

When data is transmitted through the UART transmitter, the first data frame is created. Transmitter sends data serially before it adds 1 start bit, 1 parity bit and 1 stop bit. So, the total data frame of 11 bits (including 8-bits transmitting i/p data bits). In reset conditions, serial data transmission is high logic (1). During transmission first start bit is sent i.e. logic low (0) is sent and after that data is transmitted, first LSB is sent and then parity bit and stop bit i.e. logic High (1) send and after that 1 continuously send.

Figure-2: Data Frame of UART Transmitter

iii) Architecture of UART Transmitter

In this section, the architecture of the UART transmitter is described. Fig. 4 describe the architecture of the UART transmitter. UART transmitter architecture is the integration of the following modules.

(a) Baud Rate Generator: Baud rate generator is nothing a frequency divider circuit. This module has a system clock (sys_clk), active low reset (rst_n) as an INPUT and output clock (baud_clk) as an OUTPUT. The main purpose of this module is to generate the baud rate of 4 Mbps with a system clock of 64MHz.

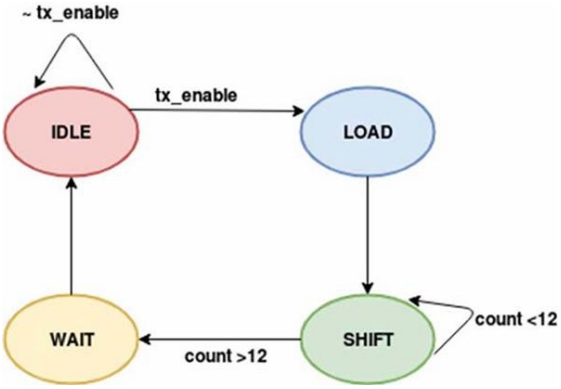

Figure-3: Finite state machine (FSM) of UART Transmitter

(b) Parity Generator: Parity generator is used to generate the parity bit. Which is useful for to check the output is correct or not at the receiver side. There are two types of parity, even parity, and odd parity. In this paper, odd parity is used to make the data frame during the transmission. To make odd parity, the input signal even_odd make logic high (1), take the xor operation on transmit input data to generate the parity bit like the output.

(c) Transmitter FSM: Transmitter FSM is used to change the state of the transmitter. It has baud clock (baud_clk), active low reset (rst_n) and transmits enable signal (tx_enable) signal as the input signal and busy, load and shift as the output of the transmitter FSM. The load and shift signal controls the PISO shift registers. Transmitter FSM has four states i.e. IDLE, LOAD, SHIFT and WAIT. WAIT is a dummy state which is used to clean the value. IDLE is the default state when the transmitter stays during a reset condition (rst_n). When transmission start (tx_enable is HIGH) the transmitter moves to LOAD state where data is loaded and the data frame is created. Further next clock (fsm_clk) i.e. baud clock (baud_clk) transmitter is the move to SHIFT state and data are transmitted serially until all the data is transmitted.

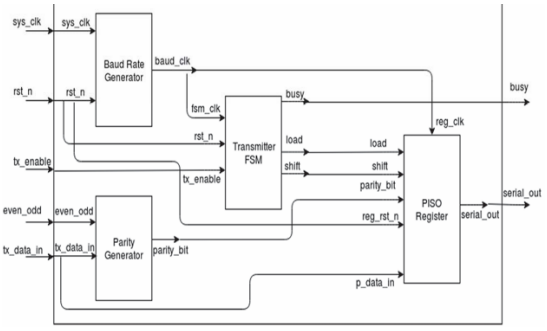

Figure-4: Architecture of UART Transmitter

(d) PISO Register: Parallel input serial output register used for transmits the data serially. Clock (reg_clk same as baud_clk), active low reset (rst_n), load, shift and input data that is transmitted (tx_data_in) is the input of PISO register and serial out (serial_out) is the output of the PISO register. When load signal is high then the data frame is created means start bit, parity bit and stop bit is added with transmit data bit (tx_data_in) and when load signal is high data is transmitted serially.

iii) Architecture of UART Receiver

In this section architecture of the UART, the receiver is described. Fig. 6 describes the architecture UART receiver which includes the baud rate generator, negative edge detector, receiver finite machine (FSM), serial in parallel out (SIPO) register and parity generator. These modules are described below.

(a) Baud Rate Generator: Baud rate generator is nothing a frequency divider circuit. This module has a system clock (sys_clk), active low reset (rst_n) as an INPUT and o/p clock (baud_clk) as an OUTPUT. The main purpose of this module is to generate the baud rate of 4 Mbps with a sys clock of 64MHz. The baud rate of the receiver is the same as transmitter.

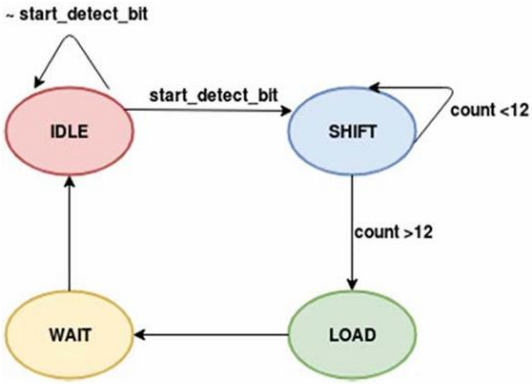

Figure-5: Finite state machine (FSM) of UART Receiver

(b) Negative Edge Detector: Negative edge detector is used for the detection of the start bit of the transmission data frame. Before the transmission transmits the data default signal is logic high. UART receiver receives the serial bits when the start bit comes then the signal goes from logic high to logic low. The negative edge detector is used to detect the start bit. After detecting the start bit state of receiver changes and store the transmission data. The negative edge detector is designed using a combination of D flip-flop and AND gate.

(c) Receiver FSM: Receiver FSM is used to change the state of the receiver. It has baud clock (baud_clk), active low reset (rst_n) and start to detect bit (start_detect_bit) signal as the input signal and busy, load and shift as the output of the receiver FSM. The load and shift signal controls the SIPO shift registers. Receiver FSM has four states i.e. IDLE, LOAD, SHIFT and WAIT. WAIT is a dummy state which is used to clean the value. IDLE is the default state when the receiver is staying during the reset condition (rst_n). The receiver starts when it detects the start bit (start_detect_bit) which is generated from the negative edge detector module. The transmitter moves to SHIFT state where shifting operation takes place until all the bits are not received. Further next clock (fsm_clk) i.e. baud clock (baud_clk) receiver movies from SHIFT state to LOAD state. Data is loaded after of the receiver (parallel_data_out). This all operation happens LOAD state. Further next clock (fsm_clk) i.e. baud clock (baud_clk) receiver moves to LOAD state to the WAIT state.

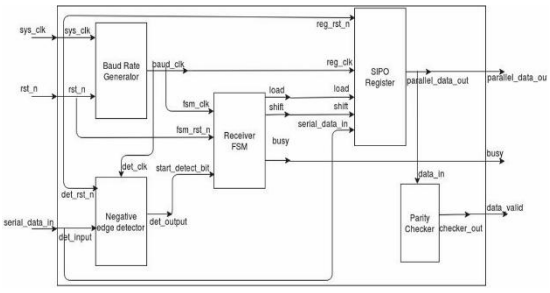

Figure-6: Architecture of UART Receiver

(d) SIPO Register: Serial input parallels the output shift register used for receiving the data serially. Clock (reg_clk same as baud_clk), active low reset (rst_n), load, shift and input data that is received (serial_data_in) is the input of SIPO register and parallel-out (parallel_data_out) is the output of the SIPO register. When shift signal is high then 1-bit shift at the positive edge of baud clock (baud_clk). When load signal is high 8 bits data is the load to the output of the receiver (parallel_data_out) after removing start bit, parity bit and stop bit.

(e) Parity Checker: Parity checker is used to checking received data is correct or not. It has one input (data_in) and one output (checker_out). A parity checker is used to check received data (data_valid) is valid or not. This verifies the parity bit from received data to ensure that no transmission errors occurred. In this paper odd parity is used to check received data is valid or not.

. RESULTS & DISCUSSIONS

In this project, simulation, synthesis, and simulation waveform of the transmitter and receiver of UART have been described. Designing, simulation, synthesis and implementation of high-speed Universal Asynchronous Receiver and Transmitter (UART) is successfully and verified using Verilog hardware descriptive language (HDL language).

The simulation and synthesis results are implemented Xilinx tool Vivado 2018.2. The high-speed serial data transfer at 4 Mbps rate with a clock frequency of 64 Mbps. UART is used for serial data transfer and the speed of UART is depending upon the transmission media of transmitter and receiver. The baud rate of the UART transmitter and receiver is the same. The verification of UART is verified using the simulation waveform of transmitter and receiver and the synthesis result is using FPGA.

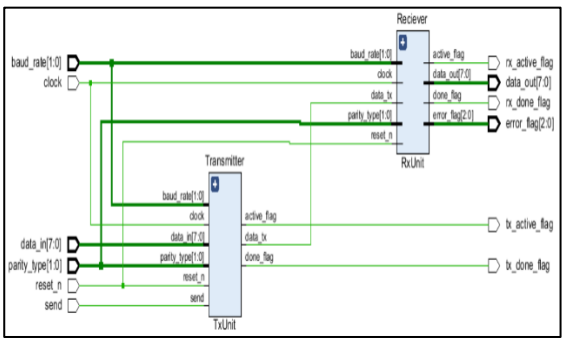

Figure-7: Schematic design of UART Transmitter and Receiver

We obtained the Schematic diagrams of Transmitter and Receiver along with the area and power Utilization. Baud rate has a system clock (sys_clk), active low reset (rst_n) as an INPUT and output clock (baud_clk) as an OUTPUT. The transmitter takes the parallel data from the computer and transmits the data serially. The receiver takes the data serially from the device and sends it to computer parallel fashion. Asynchronous word is used in the UART because both transmitter and receiver. To synchronize the received data there is no clock is required. The transmitter and receiver of the UART worked as the same baud rate. Baud rate is the unit data transfer through the channel in the communication protocol. Its unit is bits per second (bps).

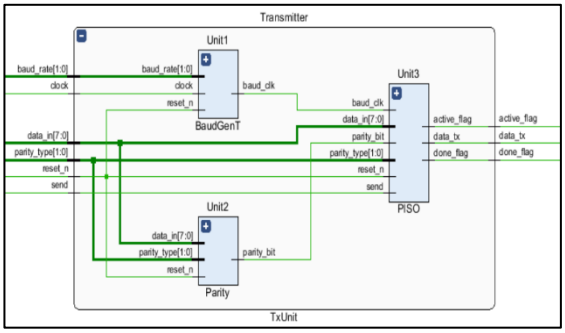

Figure-8: Internal Schematic design of Transmitter

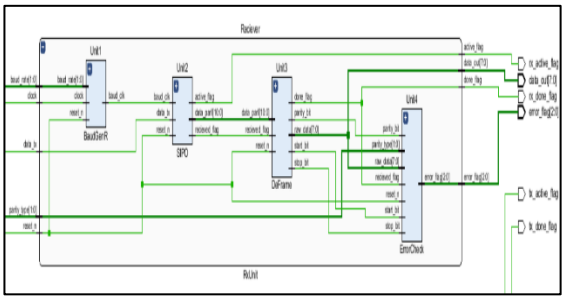

Figure-9: Internal Schematic design of Receiver

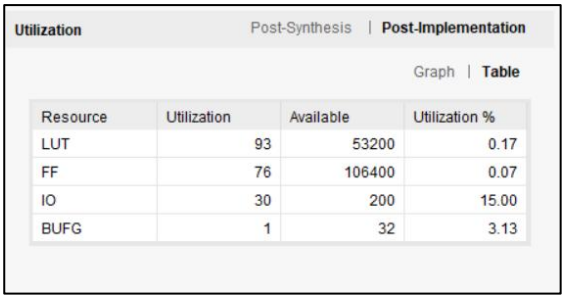

Figure-10: Area Utilization of UART

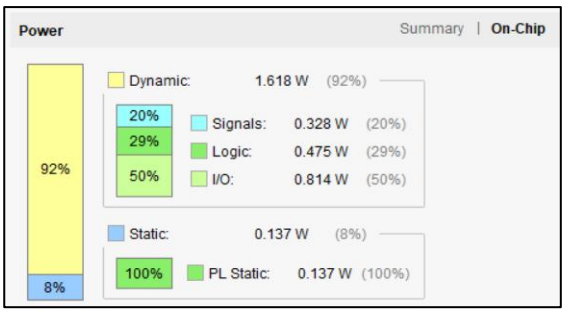

Figure-11: Power Utilization of UART

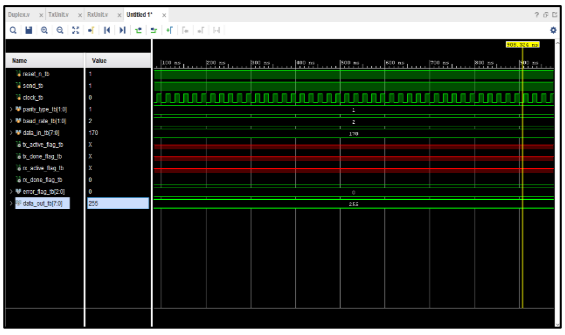

Figure-12: Results of Proposed UART

V. CONCLUSION & FUTURE SCOPE

UART with configurable baud rates and the high oversampling rate at the receiver is proposed. The transmitter and the receiver module of the UART using the structural approach is designed and successfully synthesized the same using Xilinx Vivado 2018.2. The UART with variable baud rates is successfully simulated and the design has been verified on FPGA. The design is compatible for high speed due to different baud rates and the high oversampling rate at the receiver. A maximum speed of 250Mbps is possible using this UART design.

We have presented FPGA realization of micro programmed implementation of UART controllers.

Our design is fully functional and synthesizable and can operate at a max clock frequency of 218.248 MHz. The design uses less number of FPGA resources compared to ROM based method. Our design is fully functional & synthesizable on both receiver and transmitting side. In this we use Verilog description language for obtaining the modules of Universal asynchronous receiver and transmitter. After studying the comparative analysis btw Verilog & VHDL.

In this work, designing, simulation, synthesis and implementation of high-speed Universal Asynchronous Receiver and Transmitter (UART) is successfully and verified using Verilog hardware descriptive language (HDL language). The simulation and synthesis results are implemented Xilinx tool Vivado 2018.2. The high-speed serial data transfer at 4 Mbps rate with a clock frequency of 64 Mbps. UART is used for serial data transfer and the speed of UART is depending upon transmission media of transmitter and receiver. The baud rate of UART

VI. REFERENCES

[1] McLaughlin, P. R. (1997). "A Study of the Universal Asynchronous Receiver Transmitter (UART)." IEEE Transactions on Communications.

[2] Fong, R. W., et al. (2003). "Designing High-Speed UARTs for Data Communication Applications." IEEE Transactions on Circuits and Systems.

[3] Lewis, G. D., et al. (2010). "FPGA-Based Implementation of High-Speed UARTs for Modern Communication Systems." IEEE Transactions on Computers.

[4] Liao, Y., et al. (2012). "Error Detection and Correction in High-Speed UART Systems." IEEE Transactions on Communications.

[5] Kumar, A., et al. (2016). "Practical Implementations of High-Speed UART for Data Communication Applications." IEEE Transactions on Instrumentation and Measurement.

[6] Vivek Kumar and Deepak Sharma, "Implementation Of Hybrid Serial Communication Protocol"2023.

[7] Pranay Anand Tiwari and Dr. Rajani Bisht, "Design of Low Power Universal Asynchronous Receiver and Transmitter" 2022.

[8] M. Vimala, Ajithkumar, Hariharan, Kannan, "Verification of Universal Asynchronous Receiver Transmitter by Mentor Graphics Tool" 2020.

[9] Ashwini D. Dhanadravye, Samrat S. Thorat, "A Review on Implementation of UART using Different Techniques " 2019.

[10] A. R. Duffy, "Design Considerations for High-Speed UART Communication in Embedded Systems," IEEE Transactions on Embedded Computing, vol. 14, no. 1, pp. 45-58, Jan. 2023.

[11] R. M. Ramirez and A. K. Gupta, "Design and Implementation of a High-Speed UART Transceiver Using FPGA Technology," Proceedings of the International Conference on FPGA Design, pp. 56-65, Mar. 2023.

[12] B. S. Gokhale, T. R. Gopal, and M. M. B. Krishna, "High-Speed UART Design for Embedded Systems," IEEE Transactions on Embedded Computing Systems, vol. 16, no. 2, pp. 1-13, June 2023.

[13] R. Chen and L. Zhang, "Optimizing High-Speed UART Designs for Low-Power Applications," IEEE Transactions on Low-Power Electronics and Design, vol. 18, no. 2, pp. 45-58, April 2023.

[14] L. Torres, D. Martinez, and J. Gonzalez, "High-Speed UART Protocols for Efficient Communication in Network Systems," IEEE Network, vol. 37, no. 6, pp. 62-73, December 2023.

[15] R. Brown, H. Thompson, and J. Moore, "Comparative Study of High-Speed UART Implementations for Embedded Applications," Embedded Systems Journal, vol. 23, no. 1, pp. 23-35, January 2024.