Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/37/20230520

Design, refinement, and enhancement of FPGA-implemented UART circuitry

Zhangchi Li

School of Microelectronics, Southern University of Science and Technology, Shenzhen, 518055, China

12112108@mail.sustech.edu.cn

Abstract. This article provides an overview of the universal asynchronous receiver/transmitter, commonly referred to as Universal Asynchronous Receiver/Transmitter (UART). The UART stands as a noteworthy exemplar of a serial communication protocol, facilitating the exchange of data within a serial connection while accommodating full-duplex communication. The architecture of the UART hinges upon three principal constituents: the transmitter, the receiver, and the baud rate generator, the latter essentially a frequency divider. Each of these elements is meticulously crafted using the Verilog hardware description language, thereby ensuring distinct and efficient design. Furthermore, this discourse delves into refined iterations of these implementations. For instance, it introduces the concept of a baud rate self-adaptive function and expounds upon multibyte transmission techniques. In a concerted effort to streamline circuit design and curtail the electro circuit's footprint, a deliberate decision is made to forgo the integration of a parity check module. Consequently, the chosen data format is the widely adopted 8N1 (8 data bits, 1 stop bit and no parity bit) configuration.

Keywords: UART, Verilog, Baud Rate Generator, Multibyte Transmission.

1. Introduction

The UART, short for universal asynchronous receiver/transmitter, functions as an asynchronous serial communication protocol. Its applications are diverse, ranging from PC COM ports that maintain backward compatibility with UART, to Bluetooth-UART devices, as well as devices necessitating configuration processes or featuring external command interfaces. Lately, UART's reach has extended to numerous domains, including different environmental sensors and wireless communication modules. Notably, it finds use in fields like Field Programmable Gate Arrays (FPGAs), which serve as programmable logic devices [1]. Compared with traditional ASICs (Application-Specific Integrated Circuit), FPGA has higher flexibility in implementing design and has a wide range of application scenarios. It can be reprogrammed to implement various logic functions that ASICs do not have [2]. Additionally, FPGA can be tested using the development board, dwindling the time and cost of design, so the project is based on FPGA [3].

This paper delves into a comprehensive exploration of the design and optimization of a UART system, with a focus on its FPGA implementation. The study commences with a meticulous analysis of the fundamental theories underlying UART operation. This includes an in-depth examination of the UART protocol's working principles, followed by an insightful investigation into potential optimization directions. Subsections within this analysis encompass the development of a baud rate adaptive algorithm and the elucidation of principles governing efficient multibyte transmission. The research progresses to the construction of an optimized UART protocol using Verilog hardware description language. This entails the detailed design and implementation of crucial components, including the transmitter, receiver, and mechanisms for multibyte transmission. The ensuing phase involves subjecting the developed system to rigorous simulation, generating a wealth of data for subsequent analysis. Through thorough investigation and interpretation of simulation results, the efficacy and performance of the proposed optimized UART design are critically evaluated.

2. Basic theory analysis

2.1. Working principle of UART protocol

The UART is comprised of two main components: a transmitter and a receiver. The role of the transmitter involves converting parallel data from the computer into serial data for transmission. On the other hand, the receiver receives serial data from the device and converts it into parallel data to be sent to the computer. The term "asynchronous" is employed in the context of UART because the transmitter and receiver operate on distinct clocks. To synchronize the received data, a clock isn't employed; instead, the concept of the "baud rate" is introduced. This ensures that the UART's transmitter and receiver operate at the same transmission speed. The baud rate refers to the quantity of bits transferred through the communication protocol channel in a designated unit of time. Its measurement unit is bits per second (bps). Notably, the UART functions at various baud rates including 1200, 2400, 4800, 9600, and 115200 bps [4]. To achieve successful data transmission using UART, it's essential for both the transmitter and receiver to operate at an identical baud rate.

It should be noted that for baud rate generator, the clock is originated from the FPGA system clock. Therefore, the concept of divisor is introduced, which is defined as the quotient between the FPGA clock frequency and the baud rate set by the system, which represents the number of cycles required by the system to transmit each single bit of data. In this article, we use a baud rate of 9600 in the basic transceiver mode, and the corresponding divisor is 5208 (rounded), that is, one single bit of data use 5208 clock periods to transmit from transmitter to receiver [5]. UART transmitted data is organized into packets. Each packet contains 1 start bit, 5 to 9 data bits (depending on the UART), an optional parity bit, and 1 or 2 stop bits. For simplicity of analysis and reducing electro circuit proportion, 8N1, a data format which contains 8 data bits, 1 stop bit and no parity bit is adopted [6].

Ordinarily, the UART data transmission line maintains a high voltage level during periods of data inactivity. Initiating the data transfer entails the transmitting UART shifting the transmission line from high to low for a single clock cycle. Upon recognizing this transition from high to low voltage, the receiving UART commences the reading of bits within the data frame, synchronized with the baud rate frequency. The conclusion of data packets is indicated through stop bits. To signify this, the transmitting UART elevates the data transmission line's voltage from low to high for a minimum of one bit duration. The arrangement of data packet configuration within the UART protocol can be observed in Figure 1.

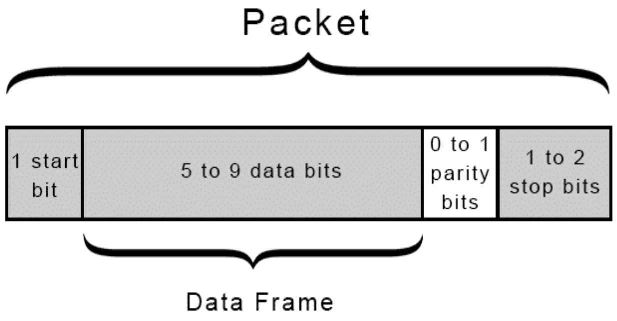

Figure 1. UART data packet configuration (Photo/Picture credit: Original).

2.2. Analysis of the optimization direction

2.2.1. Baud rate adaptive algorithm

In serial communication, only data bits are transmitted on the signal line, and no clock information is transmitted. Therefore, the UART receiver unit must know the baud rate of the input data in advance, and it must confirm how many bits per second the input data on the serial data line are transmitted in order to sample the data correctly [7]. Due to the influence of factors such as the distance of communication, the degree of environmental interference, and the distinct working rates of front-end machines in the distributed system, in order to ensure the quality of communication, the system needs to be able to realize the function of changing the system communication rate in real time in point-to-point communication. For instance, when the communication distance is long, the environmental interference is large, or the working frequency of the front-end machine is low, the system needs to reduce the communication rate to ensure the communication quality; When the communication distance is close, the environmental interference is small, or the working frequency of the front-end machine is high, the communication rate can be increased to boost the data transmission rate [8]. Therefore, the design of baud rate adaptive generator has become a direction in urgent need.

The detection methods of baud rate currently in effect include hardware method and software method. Hardware method includes counter-based detection method and phase-locked loop-based detection method. Although the hardware method has a faster detection speed, it needs to occupy additional hardware resources. Software method has a table lookup method, a calculation method and a detection method based on a finite state machine, the table lookup method and the calculation method need to spend a certain amount of time on search and calculation, the detection speed is dissatisfactory as well. The method based on the finite state machine is very accurate, but the design is more tedious.

The new baud rate self-detection improvement scheme proposed adds a communication rate measurement module composed of FPGA in the communication line at the receiving end. The receiver can use the module to measure the communication rate set by the transmitter and modify the serial communication rate with the measured value, in order to achieve the purpose of unified communication rate in the system. The whole process is as follows: take the assumption that communication rate below 300bps is not used by the system. Therefore, under normal circumstances, the widest low voltage width that can be theoretically obtained is 30 ms(when sending 0x00 at 300bps), and low-level signal with a width of more than 30 ms will not appear in the communication line during normal communication. Utilizing this feature, we can make the transmitter send 0x00 at a rate of 150bps when rate change is needed, then a low voltage pulse with a width of 60 ms will be generated in the communication line, and when the receiver detects that the low-level pulse width exceeds 50 ms, a communication rate measurement process is triggered to measure the pulse width of the next data byte [9].

After triggering the receiver to enter the communication rate measurement process, the transmitter immediately sets the baud rate to the new baud rate, and sends 0x00 data again at this rate, generating the widest low-level waveform in the communication line under the new baud rate conditions for the receiver to measure. After the receiver measures the low voltage width, dividing the measured value by 9 yields the time it takes to transmit one bit of data at the new baud rate. After the receiver completes the new rate setting, it transmits 0x55 and 0xaa (under this condition, each data in the communication line changes alternately, so the pulse width is the narrowest, the signal frequency is the highest, therefore representative), and the new rate setting of the host computer is completed and normal communication can be carried out.

2.2.2. Principle for multibyte transmission

Since a single UART protocol has a specified frame format, and the transmitter and the receiver clock signal are not synchronous (asynchronous communication), the specified data frame format can transmit up to 9 bits of data at a time (without parity bits). Nevertheless, in some specific cases, the system requires the transmission of multiple bytes at a time, where the processing of multibyte data can expand the application range of the UART protocol. The specific idea is to design nested submodules. Looking at the top module design, the input and output are multibyte data. While from the perspective of submodules, taking four-byte transmission as an example, six states are used to represent the different functions this protocol works under: transmission start state, four states of transmission of four distinct bytes and transmission end state. Successive transmission of bytes is accomplished through state conversion. In addition, two flag bits (usually two bits in the highest position) are designed in each byte, and the four bytes are respectively 00, 01, 10, 11. Resplicing the data transmitted to the receiving end in the order of binary values from smallest to largest, effective transmission of multiple bytes can be achieved [10]. Noteworthy is that the flag bits occupy the data frame, that is, there are only 6 valid data bits in each byte (excluding the start bit and two flag bits), so when it comes to four-byte transmission, only 24 bits of the 32-bit data transmitted in a single transmission are valid and can be successfully received by the receiver.

3. Frame building of optimized UART protocol (Verilog)

Since the principle of baud rate adaptive function has been explained in the second part, and it is essentially changing the size of divisor and operating the transmission procedure repeatedly, the adaptive baud rate generator will not be shown separately in this section, and the following is only the construction of the basic transceiver framework and the description of the multibyte transceiver module.

3.1. Transmitter

The figure 2 below shows the input, output signals and some relatively essential intermediate variables of the receiver. Moreover, the code structure is illustrated.

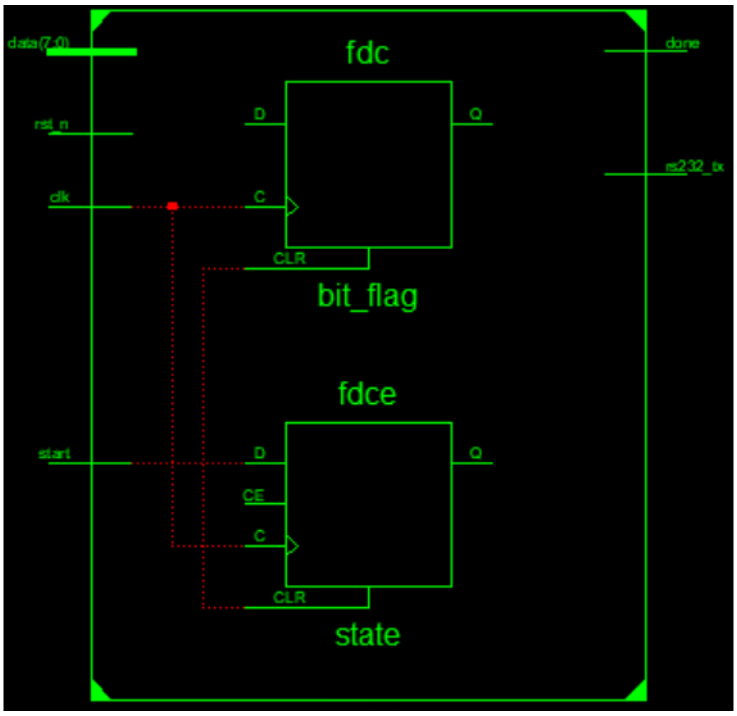

Figure 2. Signals and code structure of the transmitter (Photo/Picture credit: Original).

The procedure of transmission is analysed as follows: The input start signal pulls the state signal high and thus starts transmitting data. Each clock rising edge triggers baud _cnt autoincrement, which returns to zero when the divisor number, i.e., 5208 is reached. The zero clearing operation also indicates the accomplishment of a single-bit transmission. Meanwhile, a high voltage pulse bit _flag is made. The bit_flag rising edge leads up to the autoincrement of bit_cnt. Based on the number of the bit_cnt, which location of the input data the output end is received is determined. When the bit _cnt reaches 10, the done signal is pulled high, thereby pulling down the state signal and completing the transmission of one full byte of data.

3.2. Receiver

The figure 3 below shows the input, output signals and some relatively essential intermediate variables of the receiver. Moreover, the code structure is illustrated.

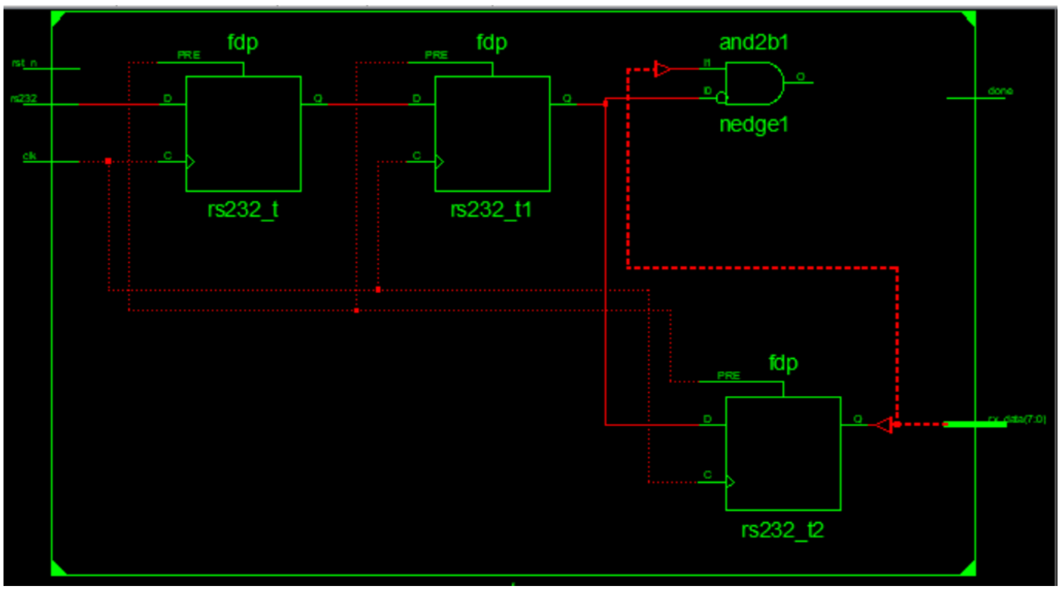

Figure 3. Signals and code structure of the receiver (Photo/Picture credit: Original).

The overall frame and intermediate variables of the receiver are similar to those of the transmitter. However, two minor differences should be mentioned. First, the rs232 signal undergoes three beats to eliminate metastability, which is equivalent to a two-stage D-trigger in digital circuit design. Also, the nedge signal is added as a flag signal for the state pull-up, similar to the start signal at the transmitter.

3.3. Multibyte transmission

In the multibyte data transfer module, the input 32-bit data stream is divided into 4 bytes. The sending and receiving steps of all submodules are the same as single-byte transmission, but the difference is that you need to choose which byte to transmit. To do this, set a 3-bit binary state signal to control 6 separate transmission states:transmission start state, four states of transmission of four distinct bytes and transmission end state. When tx_done (similar to the role of the done signal in transmitter module) equals to 1, we shift the state condition.

After a cycle of state shifting, we obtained 4 independent bytes at the receiver. Next, take the first two bits of the received one byte of data as flag bits. If the first two digits are 00/01/10/11, the received data will be the first byte/second byte/third byte/fourth byte received. Eventually, the data of received four bytes is combined, which makes up a 24-bit data received.

3.4. Simulation result analysis

The UART framework attempts to transmit three hexadecimal data: 55, 58, b8, and the corresponding simulation result is shown below in figure 4.

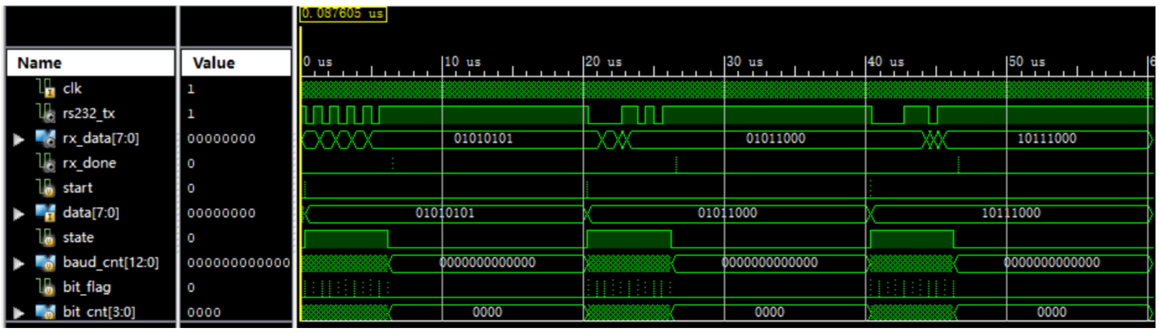

Figure 4. Simulation result (Photo/Picture credit: Original).

Run the testbench file of the receiver and display the changing waveform of some relatively important intermediate variable in the meantime. It can be seen from the simulation diagram that the transmitted data is converted to the high and low level specified in the RS232 interface and transmitted to the receiving end through the transmission mode from low bit to high bit, and the register storage function is realized at the same time, realizing the basic function of UART serial port transmission.

4. Conclusion

In conclusion, this study delved into a comprehensive analysis, design, and optimization of a UART system based on FPGA technology. The investigation began with a thorough exploration of the fundamental working principles of the UART protocol, setting the foundation for subsequent advancements. The identification of optimization directions, particularly the development of a baud rate adaptive algorithm and principles for efficient multibyte transmission, underscored the study's focus on enhancing UART's performance. By implementing the optimized UART protocol using the Verilog hardware description language, the study successfully created a functional system composed of a transmitter, a receiver, and mechanisms for multibyte transmission. Extensive simulation and subsequent result analysis provided valuable insights into the system's behavior and performance characteristics. The findings demonstrated the efficacy of the proposed optimizations in achieving improved data transmission and reception within the UART framework.

Looking ahead, there are several avenues for further research and development in the realm of UART and FPGA technology. Firstly, the proposed baud rate adaptive algorithm and multibyte transmission principles could be refined and extended to accommodate a broader range of communication scenarios and requirements. Exploring advanced error correction techniques and error handling mechanisms could enhance the system's reliability and robustness. Additionally, the implementation could be extended to consider real-world hardware constraints and limitations, fostering a more practical understanding of the proposed optimizations. Collaborative efforts with hardware engineers could bridge the gap between theoretical advancements and tangible implementations, leading to more effective and efficient UART designs. Furthermore, the integration of emerging FPGA technologies and trends, such as heterogeneous computing and high-level synthesis, holds promise for enhancing UART system performance and energy efficiency. As FPGA capabilities continue to evolve, their potential to revolutionize serial communication protocols like UART remains an exciting area of research and innovation.

References

[1] Haripriya D.,Kumar Keshav,Shrivastava Anurag,Al Khafaji Hamza Mohammed Ridha,Moyal Vishal & Singh Sitesh Kumar.(2022).Energy-Efficient UART Design on FPGA Using Dynamic Voltage Scaling for Green Communication in Industrial Sector. Wireless Communications and Mobile Computing. doi:10.1155/2022/4336647.

[2] Kumar Vivek,Rastogi Aksh & Tomar V.K..(2021).Implementation of UART Design for RF Modules Using Different FPGA Technologies. IOP Conference Series: Materials Science and Engineering(1). doi:10.1088/1757-899X/1116/1/012131.

[3] Ngoc Pham Thai, Bao Ho Ngoc, Tan Do Duy, Phuc Quang Truong & Van Ca Phan.(2021).A novel multichannel UART design with FPGA-based implementation. International Journal of Computer Applications in Technology(4). doi:10.1504/IJCAT.2021.122350.
[4] Gorabal Amrutha & D K Nayana.(2018).FPGA Implementation of UART with Single Error Correction and Double Error Detection (UART-SEC-DED). International Journal of Engineering & Technology(3.12). doi:10.14419/ijet.v7i3.12.15856.
[5] Pavan Ambati & Mrudula Singamsetti.(2018).A New Approach for RFID Tag Data Reading in FPGA by using UART and FIFO. International Journal of Engineering and Manufacturing(IJEM)(2).
[6] Sharma Rashmi,Rohilla Lakshay,Oberai Arjun,Pandey Sujeet,Sharma Vaashu & Kalia Kartik.(2016).Voltage Scaling Based Wireless LAN Specific UART Design Based on 90nm FPGA. International Journal of Smart Home(3). doi:10.14257/ijsh.2016.10.3.13.
[7] G. Snehalatha.(2016).Implementation of High Speed Secure Communication Between Multiple FPGA Systems Using RTOS. Networking and Communication Engineering(4).
[8] Gupta Isha,Garima,Singh Swati,Kaur Harpreet,Bhatt Deepshikha & Vohra Aamir.(2015).28nm FPGA based Power Optimized UART Design using HSTL I/O Standards. Indian Journal of Science and Technology(17). doi:10.17485/ijst/2015/v8i17/76859.
[9] K. Chakrapani & P. Neelamegam.(2015).Reconfigurable wrapper architecture for UART CORE testing using FSM approach. Int. J. of Advanced Intelligence Paradigms(3/4). doi:10.1504/IJAIP.2015.073713.
[10] Brahm Prakash Dahiya,Mohammad Shamim & Sunil Kumar.(2015).Intelligent monitoring the crop field using wireless sensor network based on UART and FPGA techniques. Progressive Agriculture(1).