Proceedings of the 5th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/14/20230768

Design and implementation of UART receiving module based on FPGA

Xiaowen Zhang

Taiyuan University of science and technology, Taiyuan, China

202115020134@stu.tyust.edu.cn

Abstract. In modern electronic systems, data transmission is an essential requirement within and between boards, or between lower and upper computers. To ensure data transmission accuracy, communication protocols are established that must be followed by all parties involved. These protocols include UART (universal asynchronous transmitter and receiver), IIC (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), USB2.0/3.0(Universal Serial Bus), and Ethernet. Among these protocols, UART is the most basic one and is widely used in embedded devices due to its simple circuit structure and low cost. With the exponential growth of information technology, UART-based embedded devices can easily achieve wired and wireless communication through various communication interfaces and wireless modules. In this paper, the author presents an example of a receiving module for UART communication that converts parallel data into string data. The entire module is developed using the hardware description language Verilog HDL. Simulations are performed using ModelSim, and the results demonstrate that the simulation waveform is consistent with the expected receiving data. This approach facilitates the transformation of serial data to parallel data, improving the efficiency and accuracy of data transmission.

Keywords: UART, communication protocol, embedded devices, Verilog HDL, Modelsim.

1. Introduction

In recent years, FPGA technology has gained widespread popularity among designers due to its simple design language, high flexibility, and strong parallel computing ability made possible by advanced EDA technology. However, the lack of an integrated asynchronous serial communication function module within the FPGA chip poses some practical problems. Fortunately, these issues can be cleverly solved by incorporating a UART chip [1].

The Universal Asynchronous Receiver Transmitter (UART) is a popular interface used to control communication between computers and serial devices in the field of digital communication [2]. However, traditional serial interface chips are characterized by their complex structure, large volume, and high manufacturing cost, which not only increases circuit board complexity but also reduces its functionality [3]. Conversely, by integrating UART functions into several FPGA chips, greater flexibility and compactness can be achieved while also ensuring optimal performance. The high reliability, long transmission distance, and low relative cost make UART a widely used option [4], particularly in embedded microprocessor design where an asynchronous serial interface is essential [5]. This paper provides a detailed explanation and functional analysis of the code design for the data receiving module leveraging UART, and uses Modelsim to simulate the module. The results demonstrate that the simulated waveform aligns with reality, indicating that the design has achieved its intended goals [6].

2. UART'S therr-wire communication

UART is an asynchronous transceiver, and it is part of computer hardware. It converts the transmitted data between serial communication and parallel communication. At the sender, the digital signal used for communication must be converted into a one-by-digit serial signal before it can be transmitted to the receiver through a wired or wireless channel. At the receiver, the serial signal must be reduced to a parallel signal before it can be processed accordingly [7]. And when the UART equipment can realize full-duplex communication with only one data receiving line and one data sending line during communication. Typical serial communication is completed with three wires: transmission line (TX), receiving line (RX)and ground line (GND). when communication, in order to achieve normal communication between the two devices, the TX and RX of both sides must be cross-connected and GND also must be connected. As shown in Figure 1:

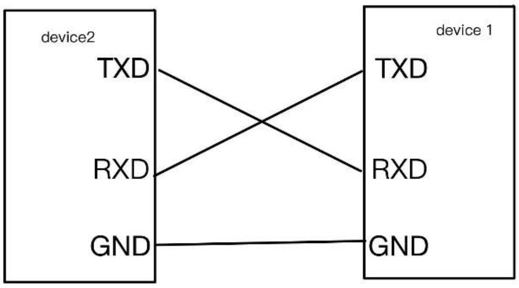


Figure 1. UART three wire communication.

3. The frame foemat of UART

Because it is asynchronous communication, when the data is sent, it will be sent in the form of the data frame. The UART frame format consists of stop bit, start bit, idle bit, check bit and data bit. The data bit follows the principle of sending low bits first and the high bits, which is often composed of 8-bit data. When the starting signal is detected, the data of the sender changes from the idle logic 1 high level to the logic 0 level, marking the beginning of data transmission, followed by the transmission of 5-8 bits of data from low to high [8]. The data position id followed by a check bit, which is generally a parity bit, but because it is inaccurate for long data verification, it is often omitted in practical application. The comes a stop bit of a logic 1 high level. After the stop bit, there is usually a one free bit. At this time, the bit logic 1 high level means that there is no data transmission, in order to prevent cumulative errors. As shown in Figure 2:

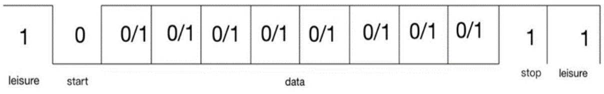


Figure 2. UART data frame format.

4. Transmission of data in UART

UART serial communication is an asynchronous serial full-duplex communication mode. Serial communication refers to the transmission od each data of the information from low to high one by one, and each data status cycle is set in advance according to the size of the baud rate. The biggest advantage of serial communication is that data transmission can be completed with only one data cable, with strong anti-interference ability and low cost [9]. For these reasons, serial communication can better realize the long-distance transmission of data. At the same time, asynchronous communication means that the two sides do not need common clock, that is, the receiver does not know when the sender will send it, so in the message, there must be a message prompting the receiver to start receiving, such as the start bit, and there is a stop bit at the end. The full-duplex communication means that we can receive data while sending data, and the two are synchronized. It’s like we usually make phone calls, and we can hear each other’s voices while talking [10]. The full-duplex mode adopts a point-to-point connection between the sender of the sending device and the receiver of the receiving device, which means that higher data transmission speed can be obtained under the full-duplex transmission mode. The TX terminal is used for data transmission and the RX terminal is used for data reception. The signal line is high level when it is idle. When the RX end of FPGA detects that there is a falling edge on the signal line, it means that there is data transmitted, and the data is received according to the pre-set baud rate. Since the data is serial from low to high, the received data will be temporarily stored in the register. After receiving 1byte of data, the received data will be saved through serial-parallel conversion. When sending, the data is sent out according to the set baud rate through the TX signal line, and the data is sent according to the format od the data frame, that is, the low level of 1bit is sent first, and then the data is sent from low to high.

5. System block diagram of UART receiving module

The figure 3 shows the system block diagram od the UART receiving data module, which is implemented by 5 signals. Respectively sys_clk: 50M system clock, sys_rst_n: system reset, usrt_rxd: receiving data cable, uart_done: data reception completion mark, uart_rx_data: received data. The function of the receiving module is to convert the serial signal to be received into a parallel signal suitable for channel transmission.

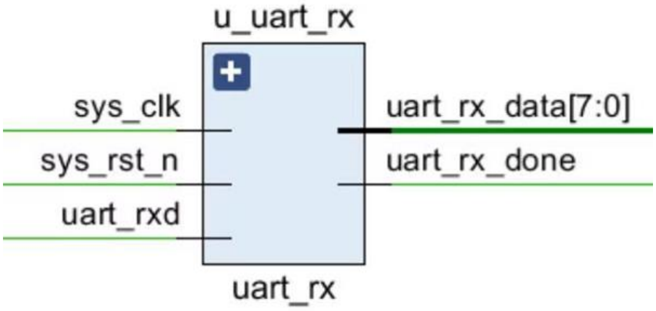


Figure 3. Receiving module block diagram.

6. Receiving module

In addition to 5 signals, it is also necessary to quantify the baud rate. In this experiment, the baud rate id adjustable, which is defined here as 9600bps, with no check bit. The most important thing is to capture the falling edge of the data line, which is used to mark the beginning of data transmission and hit the data line twice at the same time.

There are two functions: one is to synchronize the signals of different clock domains to prevent the metastable state, and the other is to capture the falling edge. After capturing the falling edge of the data (starting bit 0), the transmission start flag bit is raised, and the transmission start flag bit is pull down in the middle of the transmission process of the transmission process of the ninth data (termination bit), and the flag transmission ends. The reason why the ninth data transmission process is lowered is that the data is relatively stable at this time. Each count of a BPS_CNT, that is, the number of clocks required to transmit one bit of data, will add one data counter and clear the clock counter.Then in the process of each data transmission, the data on the data line is assigned to the data register. .

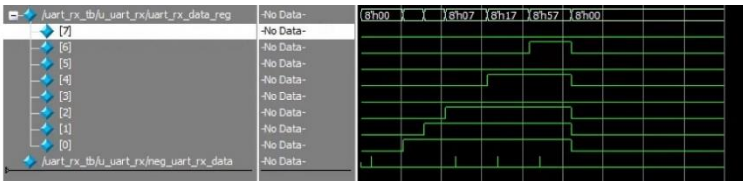
Finally, when the data is transmitted to the transmitted to the termination bit, the transmission is raised to complete the flag bit and the data is output. This is the implementation of the entire receiving module code.

7. Receive module test code

In the step, it is need to precedent the receiving module being tested, and then calculate the time required to transmit each clock. At the same time, we should pay attention to the need to pull up when it is in the idle bit, and finally follow the principle of transmitting data from low to high. The whole receiving test module is completed.

8. Simulation waveform diagram

The waveform diagram of the receiving module is shown in the figure 4. In the figure, it is showed the 8-bit data stored in the register, which is 01010111 form right to left.



```
/uart_rx_tb/u_uart_rx/uart_rx_data_reg   -No Data-
    [7]                                  -No Data-
    [6]                                  -No Data-
    [5]                                  -No Data-
    [4]                                  -No Data-
    [3]                                  -No Data-
    [2]                                  -No Data-
    [1]                                  -No Data-
    [0]                                  -No Data-
/uart_rx_tb/u_uart_rx/neg_uart_rx_data   -No Data-
```

Figure 4. Simulation waveform.

9. Conclusion

In today's digital systems, efficient data transmission between components or devices is crucial, and communication protocols are required to ensure data correctness. Among these protocols, UART is the most basic and widely used due to its simple circuit structure and low cost. With the continuous advancement of technology, UART-based embedded devices can now also easily achieve both wired and wireless communication with the help of various communication interfaces or wireless modules.

This article presents an example of a receiving module in a UART experiment, which uses hardware description language Verilog HDL to convert parallel data into string data. The simulation using ModelSim demonstrates successful transformation from serial data to parallel data, confirming the effectiveness of the approach. Overall, this research highlights the importance of efficient communication protocols in modern electronic systems and the potential of UART-based embedded devices for achieving advanced communication capabilities.

References

[1] Gupta A K, Raman A, Kumar N, et al. Design and implementation of high-speed universal asynchronous receiver and transmitter (UART)[C]//2020 7th International Conference on Signal Processing and Integrated Networks (SPIN). IEEE, 2020: 295-300.

[2] Kamath A, Mendez T, Ramya S, et al. Design and implementation of power-efficient fsm based uart[C]//Journal of Physics: Conference Series. IOP Publishing, 2022, 2161(1): 012052.

[3] Kumar V, Rastogi A, Tomar V K. Implementation of UART Design for RF Modules Using Different FPGA Technologies[C]//IOP Conference Series: Materials Science and Engineering. IOP Publishing, 2021, 1116(1): 012131.

[4] Sidek A, Saon S, Kong K W F, et al. Booth Algorithm with Implementation of UART Module using FPGA[J]. International Journal of Integrated Engineering, 2020, 12(2): 151-158.

[5] Shrestha P, Subedi B, Girard M, et al. Wireless communication between FPGA and microcontroller[C]//2020 10th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2020: 0402-0405.

[6] Jemmali S, Manaï B, Hamouda M. Pure hardware design and implementation on FPGA of an EKF based accelerated SoC estimator for a lithium‐ion battery in electric vehicles[J]. IET Power Electronics, 2022, 15(11): 1004-1015.

[7] Talapala L P, Siddaiah N, Boppana M K, et al. Implementation of the advanced encryption standard algorithm on an FPGA for image processing through the universal asynchronous receiver-transmitter protocol[J]. International Journal of Electrical and Computer Engineering, 2022, 12(6): 6114.

[8] Prasanna T L, Siddaiah N, Krishna B M, et al. Implementation of the advanced encryption standard algorithm on an FPGA for image processing through the universal asynchronous receiver-transmitter protocol[J]. International Journal of Electrical & Computer Engineering (2088-8708), 2022, 12(6).

[9] Gao Z, Zhou B, Li Y, et al. Design and implementation of an on-chip low-power and high-flexibility system for data acquisition and processing of an inertial measurement unit[J]. Sensors, 2020, 20(2): 462.

[10] Badiganti P K, Peddirsi S, Rupesh A T J, et al. Design and Implementation of Smart Healthcare Monitoring System Using FPGA[C]//Proceedings of First International Conference on Computational Electronics for Wireless Communications: ICCWC 2021. Springer Singapore, 2022: 205-213.