An FPGA-Based Adaptive Solution for Synchronous Configuration in UART Communication

Ziyi Shao *

Department of Communication engineering, Xidian University,710000, Xi'an, China

* Corresponding Author Email: 21012100048@stu.xidian.edu.cn

Abstract. To address the challenges associated with data reception failures, which arise due to abrupt alterations in the baud rate and the data format of the transmission signal in traditional Universal Asynchronous Receiver-Transmitter (UART) scenarios, this study presents an adaptive UART parameter solution underpinned by Field-Programmable Gate Array (FPGA) technology. Initially, this manuscript elucidates the underlying principles of UART communication and delineates the pivotal constituents of the UART protocol. Subsequently, it offers an in-depth analysis, from the transmission perspective, of how specific parameters influence the effectiveness of the UART protocol. Based on the software and hardware embodiments associated with the dual primary functions – baud rate adaptation and data format adaptation – this paper extensively details the flow chart for baud rate adaptation, the circuitry for edge detection, methodologies for baud rate computation, and the parameter eigenvalue approach. Following this, Verilog, a hardware description language, is employed for coding and subsequent simulation within the Quartus II environment. Empirical evidence corroborates the efficacy of the proposed design in achieving synchronous configuration between the baud rate and the data format. The findings of this research bear substantial implications for augmenting both the adaptability and dependability of UART communication systems.

Keywords: UART, FPGA, parameter adaptation, baud rate, Verilog.

1. Introduction

Universal Asynchronous Transceiver Transmitter (UART) is widely used in serial communication subsystems of microprocessors, microcontrollers, and computers [1]. It has the function of transferring data serially and can be implemented either as a UART chip or embedded inside the chip as a standalone data conversion module. With the development of FPGA technology, designers often use FPGAs to implement UART interfaces. This not only simplifies the circuit, but also improves the stability and reliability of the system. Before using the UART protocol, both sides of the communication need to set the same baud rate, data bits, parity bits, stop bits and other parameters [2]. In the online traditional UART usage scenario, when the communication baud rate and data format (data bit, parity bit, stop bit) of the transmitted signal suddenly change, it will lead to reception failure because the receiver does not make timely judgment and adjustment. Adding the parameter adaptive algorithm to the design of the UART module can solve this traditional drawback, realize the baud rate conversion for multi-device transmission or long-distance and short-distance transmission, and expand the use of the traditional UART protocol scenarios.

This paper proposes a design scheme to realize parameter detection and automated configuration of UART module on FPGA board. Starting from the foundation of UART communication principles, it will explain the concepts of several key elements of the UART protocol. Then from the transmission level to analyze the role and influence of transmission parameters on the transmission effect when using the UART protocol. The concept and design scheme of parameter adaptive algorithm is emphasized, in which the design scheme is divided into baud rate adaptive and data format adaptive two major parts with related framework logic and hardware design scheme. According to parameter adaptive module design finally, the Quartus II development environment and Verilog hardware language are used to complete the software implementation and simulation test, and the ideal parameter adaptive function is obtained by analyzing the results [3].

2. Theoretical basis for UART communication protocol and parameter adaptation

2.1. Fundamentals of UART Communication Protocols

Investigating Asynchronous Serial Data Transmission and Formatting: In asynchronous transmission methodologies, there is no requisite for the transmission of a clock signal from sender to receiver. Both transmitting and receiving entities must establish and concur upon specific timing parameters a priori. Additionally, to ensure synchronization between these units, unique bits, namely start and stop bits, are appended to each data word [1].

Typical UART data frame is shown in Fig. 1, in the UART serial communication protocol, the UART data transmission link takes the data frame as the basic unit. Each string has a start bit of 1bit as a flag bit, followed by data bits of 5-8bit, a parity bit of 1bit (when the parity is not used as a data bit), and a stop bit of 1~2bit. During the communication process, when the signal channel is idle, the level within the signal will be pulled up.

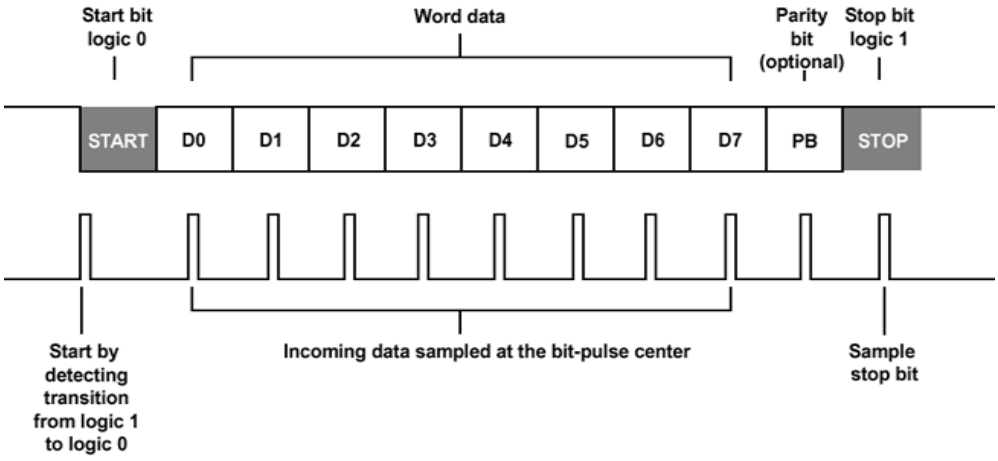


Fig. 1 Typical UART data frame [1]

Baud Rate Analysis: The term 'baud' is conventionally equated with symbols per second or pulses per second, representing the unit of symbol rate, frequently denoted as baud or modulation rate. Notably, a deviation from this standard definition is observed among modem manufacturers, where 'baud' is often, albeit inaccurately, used to denote bits per second [1].

2.2. Definition and Impact of UART Parameters

The configuration of parameters within the Universal Asynchronous Receiver-Transmitter (UART) protocol plays a pivotal role in dictating aspects such as communication rate, data bit quantity, and checksum methodologies. The ensuing discussion delineates some prevalent parameters and their associated ramifications:

Baud Rate: Ensuring synchronous serial communication between two microcontrollers mandates the stipulation of identical baud rates. When a particular baud rate is selected by the designer, it signifies a directive for the microcontroller to operate – both in terms of transmission and reception – at that specific rate. A disparity in set baud rates can result in the receiver overlooking transmitted bits due to a misalignment in data processing speeds. Standard baud rates encompass values such as 600, 2400, 4800, 9600, 14400, 38400, 57600, and 115200.

Data Bits: The 'Data Bits' parameter elucidates the quantity of legitimate data bits encompassed within each data byte. Typical values manifest as 5, 6, 7, or 8 bits. While an elevated data bit count can encapsulate a broader spectrum of data, it might concurrently augment transmission intricacy.

Stop Bits: This refers to the number of stop bits dispatched subsequent to every data byte, with prevalent values being 1 or 2 bits. An increment in stop bits can enhance system stability and fault resilience, albeit at the cost of augmented bandwidth consumption and transmission duration.

Parity: Employed as a mechanism to discern transmission discrepancies, parity checks are instrumental. Conventional parity classifications include odd parity, even parity, and the absence of parity. By instituting an odd or even parity verification on the bits within a data byte and appending a corresponding parity bit, certain transmission anomalies can be discerned and rectified.

2.3. Concepts and Methods of Parameter Adaptation

2.3.1 The concept of parameter adaptation

In order to enable the receiving device to adapt to the communication link from a variety of different baud rate transmitting devices without the need to establish the baud rate beforehand, many MCUs or embedded processors integrate an automatic baud rate detection function at the hardware or software level, which is able to automatically detect the baud rate of the host device and configure it to achieve the purpose of self-adaptation [3].

The efficiency in the actual transmission process of UART is affected by two factors: the number of communication devices and the communication distance. First, different types of new devices use different transmission rates, and it is impossible to change the communication rate and data format in real time, which also greatly reduces the communication efficiency when communicating with multiple peripheral devices. Second, the transmission rate and format used by the master device will change with the transmission conditions. For example, when the communication distance is far and the signal is easily damaged by flipping, the master device will reduce the communication rate in the communication channel to improve the quality of the communication; on the contrary, when the communication distance is close and the success rate of the signal transmission is high, the master device will increase the communication rate in the communication channel to improve the efficiency of the communication.

With the parameter adaptive method, the transmitter compiles and transmits the parameter configuration code after the transmission channel is established. The baud rate is detected by the receiver and the baud rate generator generates the corresponding sampling signal, and the valid data bits of the parameter configuration code are received serially. The baud rate and data format information are obtained by parsing the parameter configuration code.

2.3.2 Adaptive baud rate for UART serial port communication

According to the UART protocol, the automatic detection method of Modbus communication parameters can be used in this paper [4]. Fig. 2 illustrates the flowchart of baud rate adaptation. The baud rate adaptation is implemented by setting a fixed baud rate synchronization bit that is provided to the receiver as a baud rate measurement flag. The transmitter sends a configuration code beginning with the 1xx waveform to the receiver via the UART serial port. The receiver measures the duration T from the start to the first data bit and then calculates the baud rate of the transmitter using function (2).

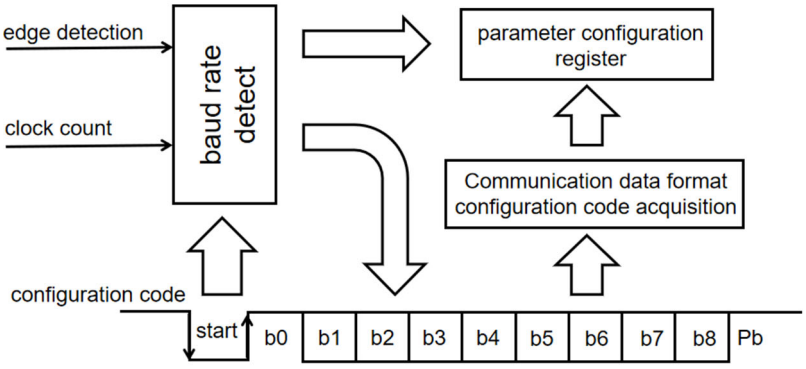


Fig. 2 Baud rate adaptive flow chart [4]

The receiver first confirms the start bit with logic "0" by the edge detection method (the first negative edge arrives when the start bit is received). Meanwhile, the system clock counter starts counting. When the next positive edge arrives, the start bit is received and the system clock counter stops counting. The duration T and the serial communication baud rate are obtained, and the valid data bits of the parameter configuration code are received at this baud rate and written to the parameter configuration register. This register can be accessed by multiple modules such as the baud rate generator to extract the parameter configuration information and realize parameter adaptation.

a) Edge Detection Module Design

Edge detection is the detection of level changes on a signal line, including positive and falling edges. When the asynchronous serial communication channel is idle, the level in the channel will be pulled up to a high level with a logic number of "1"; when transmission starts with a start bit "0" as a sign of the beginning of the transmission, and then send the data bit, stop bit, and idle bit are sent sequentially. In order to eliminate the possibility of substability caused by edge detection, two D flip-flops are designed in the FPGA with a common clock signal, as shown in Fig. 3. When the rising edge arrives, the output of the D flip-flop is D1[1], D2[0]; when the falling edge arrives, the output of the D flip-flop is D1[0], D2[1].

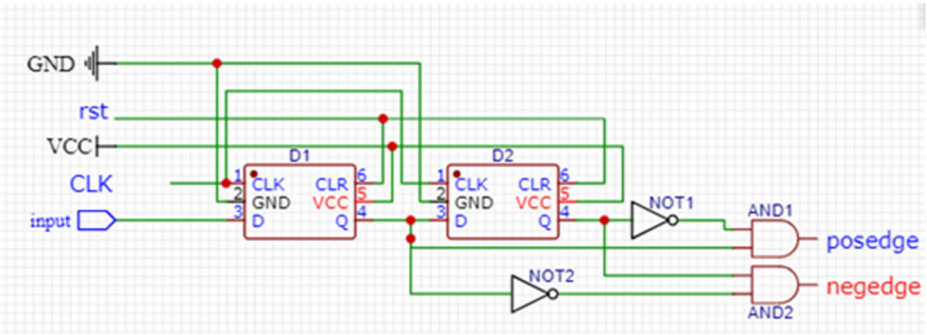


Fig. 3 Edge detection circuit diagram (Photo/Picture credit: Original)

b) Baud Rate Calculator

The 2 edge detection signals negedge_pulse and posedge_ pulse are used to generate the enable signal for the counter. The timer counts for the valid time and gets a low pulse width count value at the end. Considering the accuracy of the detection, the detection frequency of the clock should be much larger than the possible baud rate, so that the delay time of the signal flip-flop is as short as possible. The system sampling clock is set to 50MHz and the clock period T is 20ns. The calculation shows that, assuming that the baud rate of 115200 bps is used to transmit the data in the communication channel, the time measurement error ratio of the clock counts is less than 1.5%, which ensures a high time measurement accuracy. The bit width Bit_width is the time required for transmission of each data bit in the data, and the formula for calculating the bit width Bit_width is:

$$\textit{(unit:ns)} \tag{1}$$

The relationship between bit width and baud rate in numerical terms is:

$$\textit{(unit: bps)} \tag{2}$$

The commonly used transmission baud rate has a fixed standard value and has a certain deviation from the measured value. Compared with the traditional baud rate detection algorithm, which requires repeated baud rate detection to reduce the algorithm error, the baud rate determination using the table lookup method reduces the use of computing power in the FPGA and improves the efficiency of the FPGA. Considering the possible error of jitter in the data sent and received by the serial port, the error tolerance range of 3% baud rate is set in the table statistics to reduce the error in the design of the edge detection module [5]. The commonly used baud rates and their control bit ranges are shown in Table 1.

Table 1. Comparison of commonly used baud rates

Bit rate	Bit width (ns)	Bit width tolerance range/(ns）	
		Lowest bit width (-1.5%)	Maximum bit width (+1.5%)
600	1666666.667	1641666.667	1691666.667
2400	416666.6667	410416.6667	422916.6667
4800	208333.3333	205208.3333	211458.3333
9600	104166.6667	102604.1667	105729.1667
14400	69444.44444	68402.77778	70486.11111
38400	26041.66667	25651.04167	26432.29167
57600	17361.11111	17100.69444	17621.52778
115200	8680.555556	8550.347222	8810.763889

2.3.3 Adaptive data format for UART serial port communication

Proper UART serial communication requires that the two communicating parties are sufficiently well matched, otherwise communication errors may occur [6]. Therefore, the design has to complete the baud rate parameter configuration and data format multi-parameter automation configuration at the same time, in order to ensure the success rate of data transmission in the communication process. Since the baud rate and data format need to be configured with different binary feature values, a total of 3 baud rate feature values, 2 data length feature values, 1 parity bit feature value and 1 stop bit length feature value are needed, so an 8-bit parameter register combining the parity bit and the data bit is designed for storing the parameter feature values. The transmitter sends the feature value as data bits to the receiver, and the receiver confirms the baud rate and data format parameters of the next transmission through the recognition of the feature value, realizing the adaption of UART communication parameters. The following Table 2 and Table 3 shows the parameter register corresponding to the allocation of the characteristic value, especially note that when the data bit is 8 bits, the default parity mode is no parity.

Table 2. Configuration Code of baud rate and parameter configuration register address

Baud Rate/bps	Configuration Register [1:3]
600	000
2400	001
4800	010
9600	011
14400	100
38400	101
57600	110
115200	111

Table 3. Configuration Code of data format and parameter configuration register address

Format	Address	Configuration Code
Data length:5 bit	[5:4]	00
Data length:6 bit	[5:4]	01
Data length:7 bit	[5:4]	10
Data length:8 bit	[5:4]	11
Odd	6	1
Even	6	0
Stop length:1 bit	[8:7]	00
Stop length:1.5 bit	[8:7]	01
Stop length:2 bit	[8:7]	10

3. FPGA-based UART parameter adaptive design

3.1. Parameter Adaptive Module Design

As shown in Fig. 4, RXD and TXD are signal input buses and output buses for receiving data characters and sending characters. Wherein, clock counter 1 is a baud rate counter, and clock counter 2 is a data bit counter. The receive shift register and transmit shift register are mainly used to realize the serial receiving and transmitting function of data, and the communication parameter register stores the received parameter configuration code according to the storage address designed in Table 2 and Table 3. This register can be accessed by the receiver controller, baud rate generator, and transmitter controller, and the bit shift register has write access to it, and the data transceiver module has read access to it [7].

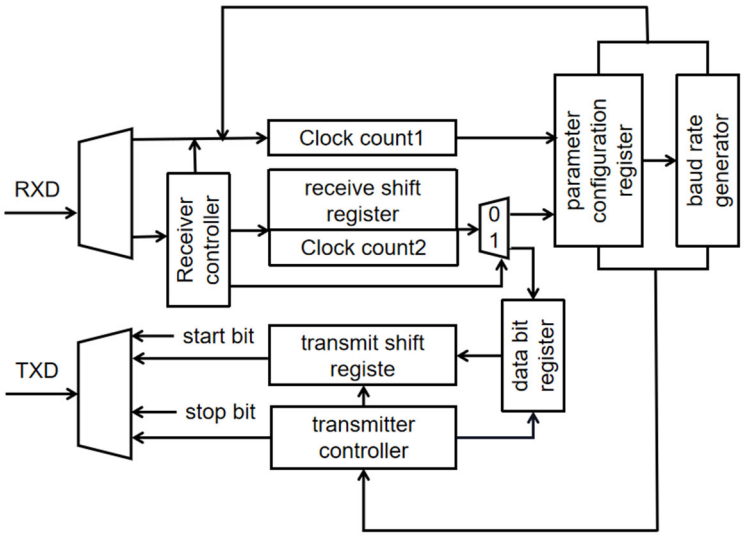


Fig. 4 FPGA architecture diagram [7]

3.2. System Software Design

The overall software design part adopts Quartus II 13.1 as the development environment, selects the CycloneIV series lightweight FPGA chip EP4CE6E22C8 as the carrier of the UART module, and adopts the hardware description language Verilog HDL. It is an IEEE-standard hardware description language with a very concise syntax compared with other hardware description languages [8]. The traditional digital logic circuit design method is mainly the schematic input method that in the design process special design tool and manual wiring design is required leading long period and Poor maintainability [9].

Quartus is an embedded design software developed by Intel, which is a comprehensive tool for the development of FPGAs and PLDs [10]. ModelSim, on the other hand, is a simulation platform developed by Mentor Graphics for verifying hardware designs. ModelSim supports an event-based simulation methodology that accurately simulates circuit behaviors and provides powerful debugging features. In addition, ModelSim can be tightly integrated with Quartus, which enables design verification and optimization to be carried out more efficiently. and optimization can be carried out more efficiently.

3.3. Performance Simulation and Evaluation of UART Parameter Adaptation

3.3.1 Simulation Design and Results of Parameter Adaptive Algorithm

a) The baud rate detection module accomplishes the baud rate detection task by determining whether the baud rate detection module is able to function properly when the analog communication parameters have not been subjected to configuration.

Based on the baud rate and a fixed sampling clock frequency, the detection time can be calculated. Taking the sampling frequency of 50 MHz and baud rate of 115200 b/s as an example, with a period of 20 ns, the bit width corresponding to 115200 b/s is 8550-8810 ns. Compared with the sampling period of 20 ns, the delay caused by sampling is negligible, so the baud rate detection time is only related to the baud rate itself. As shown in Fig. 5, at a baud rate of 115200 b/s, accurate detection can be obtained after 433 counts at the earliest.

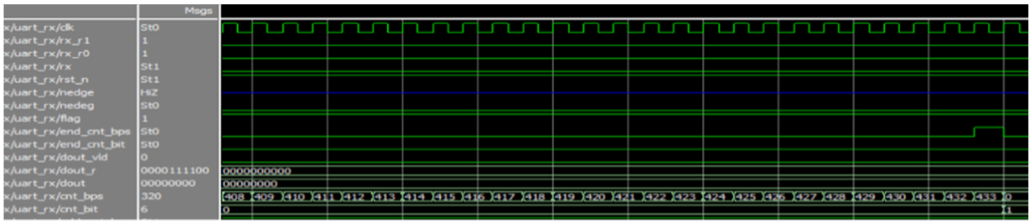


Fig. 5 Baud rate test results (Photo/Picture credit: Original)

b) This is shown in Figure 6. After obtaining a baud rate of 115200 from Table1, the baud rate generator generates the sampling frequency of the bit counter, and increments the count value in the bit counter with each sample until the receive stops.

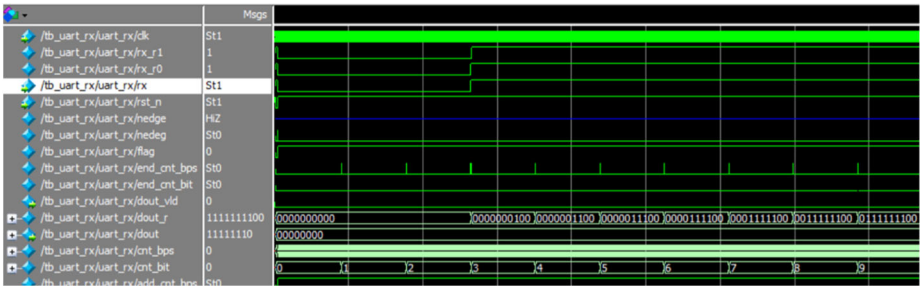


Fig. 6 Data bit counter simulation results (Photo/Picture credit: Original)

c) The data sending function of the uart when the communication parameters have been configured. The basic functions of a UART transmitter module are described below:

Data Input: Receive data from the upper level. Frame Synchronization: Adds start, stop and possibly parity bits to define the start and end of each data frame.The frame synchronization function of the UART transmitter module is designed to ensure the reliability and correctness of data transmission. By inserting start, data and stop bits, the receiver is able to parse the data frame correctly based on these flags.

As shown in Figure 7, the transmitter module receives data 10010101 from the data bit register, performs frame synchronization of the data, and adds the start bit of the logic signal "0" and the stop bit of the logic signal "1" to obtain 0100101011.

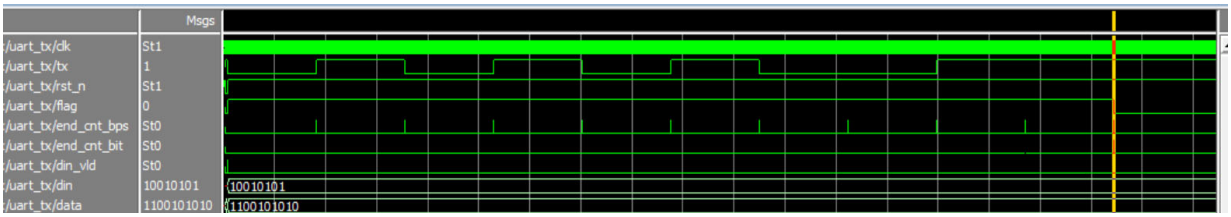


Fig. 7 Encapsulation of data bits on the transmitter side (Photo/Picture credit: Original)

3.3.2 Analysis and Discussion of Simulation Results

Based on the FPGA to realize the parameter auto-configuration method of general-purpose asynchronous serial communication, combining the way of eigenvalue matching and the look-up table method, the adaptive detection and auto-configuration of baud rate and data format are successfully realized. Based on Cyclone IV series FPGA, the hardware circuit design of the UART interface of this scheme is realized using Verilog hardware description language. Through simulation and testing, it is confirmed that the method can quickly and stably establish a serial communication link. The

overall design has a high degree of integrity and stability. Compared with the traditional baud rate adaptation method, this method not only realizes baud rate adaptation, but also automatically configures the data format in the serial communication parameters. This complements and improves the application method of generalized serial communication, which is of high practical significance. Through this method, the parameter configuration process of communication devices can be simplified, and the compatibility and flexibility of the system can be improved. In summary, the FPGA-based generalized asynchronous serial communication parameter auto-configuration method shows good performance and reliability in practice. The application of this method provides an effective solution for the development and design of serial communication systems.

4. Conclusion

The UART protocol involves a number of parameters such as baud rate, data bits, parity bits, and stop bits, to name a few. The parameter adaptive algorithm aims to automatically adjust these parameters according to the actual needs, which is an innovative research result. At the same time, the adaptive design can be used to solve the problem of mismatch in communication rate between different devices and improve the efficiency of data transmission. FPGA is a kind of programmable logic device, which can realize a variety of digital circuit functions. Combining the UART protocol with FPGA can realize highly flexible and reconfigurable UART communication.

Implementing FPGA-based adaptive design of UART protocol parameters also requires consideration of circuit resource constraints. FPGAs have limited resources, and an efficient algorithm needs to be designed to optimize the use of resources to achieve the adaptive function. At the same time, the programming of FPGAs also needs to take into account issues such as timing constraints and signal integrity, which further increases the complexity and challenge of the design. With the continuous progress and development of FPGA technology, its resources and performance are also improving. This provides better conditions for realizing FPGA-based adaptive design of UART protocol parameters. In the future, the algorithm and design can be further optimized to achieve more efficient and reliable UART communication using more advanced FPGA chips.

References

[1] Dawoud D S, Dawoud P. Serial communication protocols and standards[M]. CRC Press, 2022.

[2] Wu Zhaohui, Qu Liguo. FPGA Design Algorithm and implementation of UART Baud rate detection circuit. Modern Electronic Technology, 2022, 45(20): 41-44.

[3] Sun Fuwen, Zheng Caijun, Liu Xinzhuo et al. Based on the FPGA serial port baud rate adaptive function. The design and implementation of electronic design engineering, 2019, 27(9): 69-73.

[4] Wei Zhenyi, Yang Jiang. An Automatic Detection Method for Modbus Communication Parameters. Industrial Control Computer, 2004(07): 32-33.

[5] Liu Pingfan, Zou Fang, He Zhaoyan, et al. Development of Edge Controller Digital IO Extension function Board. Automation Expo, 2023, 40(2):67-71.

[6] Zhang Bin. Research and implementation of intelligent Sensing system based on UART serial port. Chang 'a university, 2020.

[7] Yang Yang, He Lei, Wang Xu et al. UART serial communication parameters adaptive based on FPGA design and implementation of. Journal of electronic design engineering, 2021, 29(16): 21-25.

[8] Chen Hu. Baud Rate automatic detection technology of asynchronous serial Communication based on FPGA. Applied Science and Technology, 2019, 50(02): 60-65.

[9] Qi Xiaolei, Cai Xueliang, Sun Dewei. Principles and methods of FPGA design with Verilog HDL. Electronic Testing, 2008(03): 67-71.

[10] Wang Laihua. Hierarchical digital logic circuit design Method based on Quartus II. Electronic Technology, 2018, 47(11): 89-90+86.