International Journal of Current Engineering and Technology
ISSN 2277 - 4106

Available at http://inpressco.com/category/ijcet

Review Article

FPGA Implementation of 9 bit Universal Asynchronous Receiver Transmitter

Ashwini D. Dhanadravye[Å*] and Samrat S. Thorat[Å]

[Å]Department of Electronics and Telecommunication, Government College of Engineering Amravati, Maharashtra, India

Accepted 25 May 2014, Available online 01 June 2014, **Vol.4, No.3 (June 2014)**

Abstract

Universal Asynchronous Receiver Transmitter (UART) is widely used serial data transmission protocol to support full duplex communication. This paper presents the design of 9 bit UART module based on VHDL. The design of 9 bit UART specialized with automatic address and data differentiator in the character itself thus allowing the incoming data to be transferred directly to the destination, in case address matches. UART design mainly consists of three important modules which are receiver module, transmitter module and baud rate generator. The UART design with VHDL as design language can be integrated into the Field Programmable Gate Array to achieve reliable, compact & stable data transmission. It's significant for the design of System on Chip. The whole design simulated using Active-HDL simulation tool and implemented onto FPGA board using Quartus II software.

Keywords: *Asynchronous serial communication, Quartus II, simulation, VHDL, UART*

1. Introduction

UART is Universal Asynchronous Receiver Transmitter used for serial communication over a computer or peripheral device serial port. UART performs parallel-to-serial conversion on data character received from the host processor into serial data stream, and serial-to-parallel conversion on serial data bits received from serial device to the host processor. UARTs are commonly used in conjunction with communication standards such as EIA, RS-232, RS-422 or RS-485. The Universal Asynchronous Receiver Transmitter (UART) is a popular and widely-used device for data communication in the field of telecommunication. It has many advantages such as simple resources, reliable performance, strong anti-jamming capability, easy to operate and realize and so on. This paper uses VHDL to implement the UART core functions and integrate them into a FPGA chip to achieve compact, stable and reliable data transmission

A UART has standard transmission protocol which consists of a start bit '0', 5-8 bits data, optional parity bit and stop bit '1'. Fig. 1 shows the data frame format of a UART. While in idle state, serial data line will be in logic '1' state. A start bit '0' at the beginning of the data frame will cause a falling edge on the serial data line. This marks the detection of a data character. The idea of start bit and stop bit in UART is to achieve data synchronization. An optional parity bit can be in odd parity or even parity. Odd parity means that sum of all bits gives an odd number, while even parity means sum of all bits gives an even number. The serial data frame is shifted out with the least significant bit (LSB) first**.**

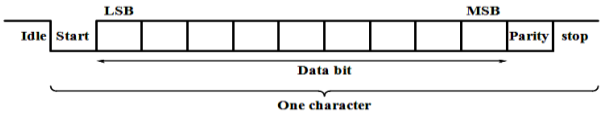

Fig.1 UART Transmission Protocol

2. The UART Module

A 9-bit UART is modification to the normal UART which facilitate with ninth bit to be used either as parity identifier or address/data differentiators. A 9 bit UART uses the ninth bit of character to differentiate between an address or a data. Thus the 9 bit UART module is able to distinguish between an address byte and data character resulting in saving a lot of processing time taken by slave devices. Since in transmission the slave devices will search for every character transmitted for address byte and try to match with its unique address.

The design of 9 bit UART proposed in this paper configures the ninth bit such that it is set to logic '1' to indicate an address character and set to logic '0' to indicate the data character. Thus in a transmission, processor will search for a parity bit and if it is found to be at logic '1' then it will try to match the address with its own unique address. If address matches, it receives the data bytes followed by an address. If address matching failed, processor will ignore the following data bytes. Thus the proposed design helps in reducing the processing time required in searching for address of a particular slave device.

3. Design of UART Submodules

The 9-bit UART module proposed in this paper consist of

*Corresponding author: **Ashwini D. Dhanadravye**

basic sub modules of UART which are receiver, transmitter, and baud rate generator. Therefore, the implementation of the UART communication module is actually the realization of the three sub-modules. Fig. 2 below shows the complete block diagram of the 9 bit UART. The baud rate generator is actually a frequency divider that can be calculated according to system clock frequency and the desired baud rate. The function of baud rate generator is to produce a local clock signal which is much higher than the baud rate to control the UART receive and transmit. The receiver performs serial-to-parallel conversion on the asynchronous data frame received from the serial data input. The transmitter module converts the bytes into serial bits according to the basic frame format received from the CPU.

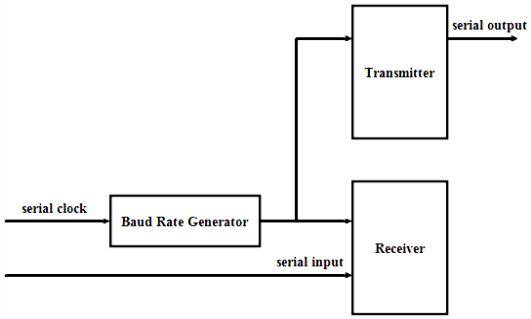

Fig.2 UART Module

3.1 Baud Rate Generator

The 9 bit UART module can be operate at any defined clock frequency and at the same time follows desired baud rate. The baud rate generator calculates the divide factor with the help of system clock frequency and desired baud rate. Thus the frequency clock produced by baud rate generator is not the baud rate but 16 times of baud rate clock. The purpose is to precisely sample the asynchronous serial data at the receiver because it is difficult to detect where to sample the input data. The calculation to get the divide factor is shown below.

$$\text{Divide Factor} = \frac{\text{Serial Clock Frequecy}}{16 \times \text{Desired Baud Rate}}$$

3.2 Receiver

Receiver module receives the serial data through RX input pin.

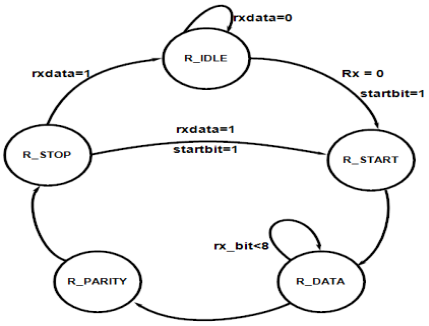

Fig.3 UART Receiver State Machine

It detects the start bit, when there is transition from logic 1 to 0 on the serial data line. UART receiver module will receive data with parity bit followed by start bit. Depending upon the status of ninth bit, receiver module can differentiate whether the incoming data is data character or address byte. This proposed design used the finite state machine to implement the receiver module.

This paper used finite state machine to implement receiver module. The operation of receiver completes using five states shown as below.

R_IDLE: When the UART receiver is reset, the receiver state machine will be in this state. In this state, state machine will wait for the start bit detection on the serial input line. Start bit will be detected when RX pin will transit from logic 1 to 0.

R_START: In this state, start bit will be detected as soon as RX pin goes to logic 0 and will wait for 16 times of baud rate clock before going to next state.

R_DATA: For asynchronous serial signal, most ideal time for sampling is at the middle point of the bit. Hence the task of this state is to read the middle point of each bit so as to minimize the total error in the incoming data detection. Each bit is then stored into an internal register reg[7:0] to form a complete 8 bit data.

R_PARITY: In this state, state machine will sample the ninth bit and determine whether the incoming data is address character or data byte or even or odd parity depending upon user requirement.

R_STOP: In this state, state machine will detect the stop bit with rxdata<='1', it means complete data has been received by the receiver and waiting for the next frame to come.

3.3 Transmitter

The function of the transmitter module is to convert the sending 8 bit parallel data into serial bits with the addition of start bit, stop bit and parity bit of '1' for address character, '0' for data byte or shows even or odd parity status depending upon user requirement. In case ninth bit configured as address/data differentiator, address can be broadcast address or unique address for specific slave device. Once address has been successfully transferred, UART transmit module will transmit the data bytes to the addressed device.

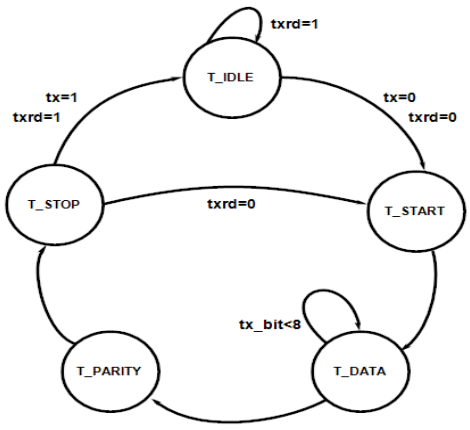

Fig.4 UART Transmitter State Machine

Transmitter module also implemented using internal finite state machine to transmit the parallel bytes into serial bits. TXRD and TX pins are the output signal of transmitter module shown in the figure below. Similar to receiver, transmitter completes its operation through five states which are T_IDLE, T_START, T_DATA, T_PARITY, T_STOP as shown in fig. 4.

T_IDLE: Transmitter module will be in this state, when UART is being reset. In this state, transmitter will wait for transition of txrd signal from '1' to '0' indicates that data is ready to transmit.

T_START: In this state, transmitter will add start bit on the serial line and will wait for 16 times of baud rate clock before going to the next state.

T_DATA: In this state, state machine will loads the internal register tx[7:0] with data to be transmitted starting from the least significant bit tx[0] to most significant bit tx[7].

T_PARITY: Depending upon the 8 bit data, this state will add the parity bit of '1' for address character and '0' for data byte or may show the parity status based on design configuration

T_STOP: When the complete frame of data has been successfully transmitted, then transmitter module will undergo this state and will send stop bit of logic '1'. If txrd=0, it means transmitter is ready to transmit next data frame and will switch back to T_START state and repeat all steps until it goes to same state again. If there is no more data to be transferred it will go to T_IDLE state.

4. Simulation of Modules

The simulation software is Quartus II. And selected device is Altera's Cyclone II FPGA: EP2C35F672C6

4.1 Baud Rate Generator Simulation

During simulation, the system clock frequency is set to 27MHz, and baud rate is set to 9600bps. Therefore the receiving sampling clock frequency generated by the baud rate generator is 153600Hz, which is 16 times of the baud rate. Thus the frequency coefficient of baud rate generator can be calculated, which equals 176. The simulation report shows that this module uses 51 logic elements (<1%), total 34 register and meets timing requirement.

Fig.5 Baud Rate Generator Simulation Waveform

4.2 Receiver Simulation

During receiver simulation, the receiving sampling clock Frequency generated by the baud rate generator is set to 153600 Hz, UART receiving baud rate is set to 9600bps. The simulation report shows that this module uses 97 logic elements (<1%, 13 pins (3%)), total 77 register and meets timing requirement.

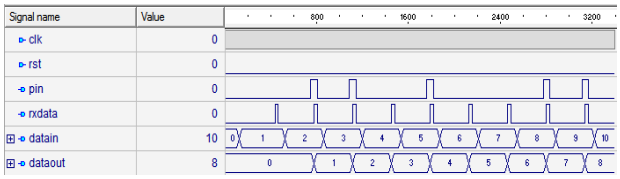

Fig.6 Receiver Simulation Waveform

4.3 Transmitter Simulation

During transmitter simulation, the sending clock frequency generated by the baud rate generator is set to 153600 Hz, and UART transmitting baud rate is set to 9600bps. Fig. 7 shows the transmitter module simulation diagram. The simulation report shows that this module uses 116 logic elements (<1%, 3 pins <1%), total 84 register and meets timing requirement.

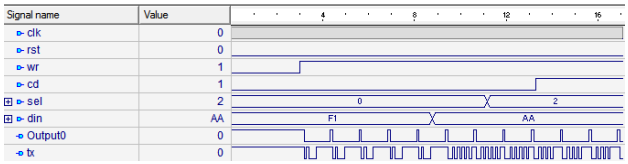

Fig.7 Transmitter Simulation Waveform

Conclusion

In this paper, UART has been modified with ninth bit facilitate to identify address character or data byte automatically thereby reducing lot of processing time required in searching the destined device. The 9 bit UART design is implemented using VHDL language and simulated to check the functionality of each sub modules. Using Quartus II software, Altera's cyclone II FPGA chip EP2C35F672C6 to complete simulation and test. Thus design shows greater flexibility, stability and reliability.

References

C. He, Y. Xia, and L. Wang (2011), A universal asynchronous receiver transmitter design, *International Conference on Electronics Comm. and Control (ICECC 2011)*, Ningbo, China, September. 2011.

Debjani Basu, Deepak kole, Hafizur Rahaman (2012), Implementation of AES algorithm in UART module for secured data transfer, *2012 International conference on Advances in Computing and Communication*, IEEE, DOI 10.1109/ICACC.2012.32.

J. Norhuzaimin, and H. H. Maimun (2005), The design of high speed UART, *Asia Pacific Conference on Applied Electromagnetic (APACE 2005)*, Johor, Malaysia, December. 2005.

Mahat N.F (2012), Design of a 9-bit UART module based on Verilog HDL, *in the proceedings of 10th IEEE International Conference on Semiconductor Electronics (ICSE)*, 19-21st September. 2012, pp. 570-573.

Mohd Yamani Idna Idris, Mashkuri Yaacob and Zaidi Razak (2006), A VHDL Implementation of UART Design with BIST Capability, *In the proceedings of Malaysian Journal of Computer Science*, June 2006, Vol. 19(1), pp. 73-86.

Shouqian Yu, Lili Yi, Weihai Chen, Zhaojin Wen (2007), Implementation of a Multi-channel UART Controller Based on FIFO Technique and FPGA" 1-4244-0737-0/07/$20.00 c 2007 IEEE.

Y. Fang and X. Chen (2011), Design and simulation of UART serial communication module based on VHDL, *In the proceedings of 3rd International Workshop on Intelligent Systems and Applications (ISA)*, IEEE, May 2011, DOI: 10.1109/ISA.2011.5873448, pp.1-4.

Z. Zhang, and W. Wu (2010), UART integration in OR1200 based SoC Design, *2nd International Conference on Computer Engg. and Tech. (ICCET 2010)*, Chengdu, China, April. 2010.