Design and Simulation of UART for Serial Communication

[1]Manju Wadhvani

[1] Electronic and Telecommunication Engineering,
Chhatisgarh Swami Vivekanand Technical university,
Disha Institute of Management and Technology, Raipur, India
manju.wadhvani@gmail.com

[2]Prof. Zoonubiya Ali

[2]Electronic and Telecommunication Engineering,
Chhatisgarh Swami Vivekanand Technical university,
Disha Institute of Management and Technology, Raipur, India
zoonubiya khan@gmail.com

Abstract— Asynchronous serial communication is usually implemented by Universal Asynchronous Receiver Transmitter (UART), mostly used for short distance, low speed, low cost data exchange between processor and peripherals. UART allows full duplex serial communication link, and is used in data communication and control system. In parallel communication the cost as well as the complexity of the system increases due to simultaneous transmission of data bits on multiple wires. The UART simulated with VHDL language to achieve stable and reliable data transmission. This paper presents the simulation of UART with configurable baud rate.

Keywords— UART; Asynchronous serial communication; VHDL; simulation.

I. INTRODUCTION

The fact that serial communication uses a single data line instead of the 8 bit data line of parallel communication not only makes it much cheaper but also enables two computer located in to different cities to communicate over the telephone. Universal Asynchronous Receiver Transmitter (UART) is a kind of serial communication protocol; mostly used for short-distance, low speed, low-cost data exchange between computer and peripherals .Serial data communication uses two methods, asynchronous and synchronous. The synchronous method transfers a block of data (character) at a time, while the asynchronous method transfers a single byte at a time. It is possible to write software to use either of these methods, but the programs can be tedious and long. For this reason, there are special IC chips made by many manufacturers for serial data communications. These chips are commonly referred to as UART i.e; Universal Asynchronous Receiver Transmitter. Asynchronous serial data communication is widely used for character- oriented transmissions, while block oriented data transfer use the synchronous method. [1]

Universal Asynchronous receiver transmitter (UART) is mostly used for the exchange of data communication between computer and peripherals which is a serial communication protocol. Specifically, it provides the computer with the RS-232C Data Terminal Equipment (DTE) interface so that it can "talk" to and exchange data with modems and other serial devices.[2] It handles the conversion between serial and parallel data. Serial communication reduces the distortion of a signal, therefore makes data transfer between two systems separated in great distance possible. UART design mainly contains three blocks those are Transmitter, Receiver and Baud Rate Generator. Transmitter performs a parallel-to serial conversion for data transmission from the computer and the Receiver performs serial to parallel conversion for data coming in via the serial line. The baud rate generator is used to produce a local clock signal which is much higher than the baud rate to control the UART receive and transmit.

In addition to the basic job of converting data from parallel to serial for transmission and from serial to parallel on reception, a UART will usually provide additional circuits for signals that can be used to indicate the state of the transmission media and to regulate the flow of data in the event that the remote device is not prepared to accept more data. UART must have a larger internal buffer to store data coming from the modem until the CPU has time to process it.[3]

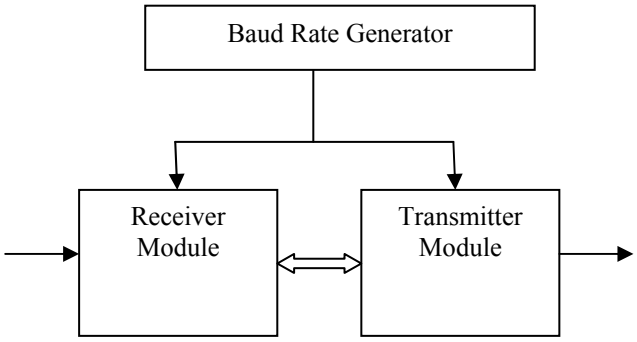


Figure 1. UART Module

II. UART DATA FRAME FORMAT

UART transmit and receive data format shown in Figure 2, usually include start bit, data bit, parity bit, stop bit and idle state. Start bit is the beginning of data transmission. When the transmitter sends a character data, a logic "0" signal is firstly send, which is the start bit, and the time width is a baud rate clock cycle. Start bit is followed by data bits, which are usually from 5 to 8 bits. The data bit from the least significant bit (LSB) begin to send. The data bits can be parity bit, which can be odd or even parity, as well as no parity bit. The parity bit or data bit (when no parity bit) is followed by stop bits, which are logic "1" signal containing 1,1.5 or 2 bits. Stop bits are the end of a data. Idle state is a logic "1 ".This data frame format is adopted by the start bit and stop bit to achieve character synchronization. UART has an internal configuration register, in which user can set the data bits, whether there is parity bit, as well as the type of parity and stop bits.[4]

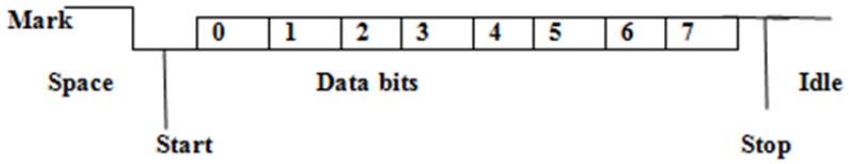


Figure 2. UART Frame Format

III. UART IMPLEMENTATION BY USING VHDL

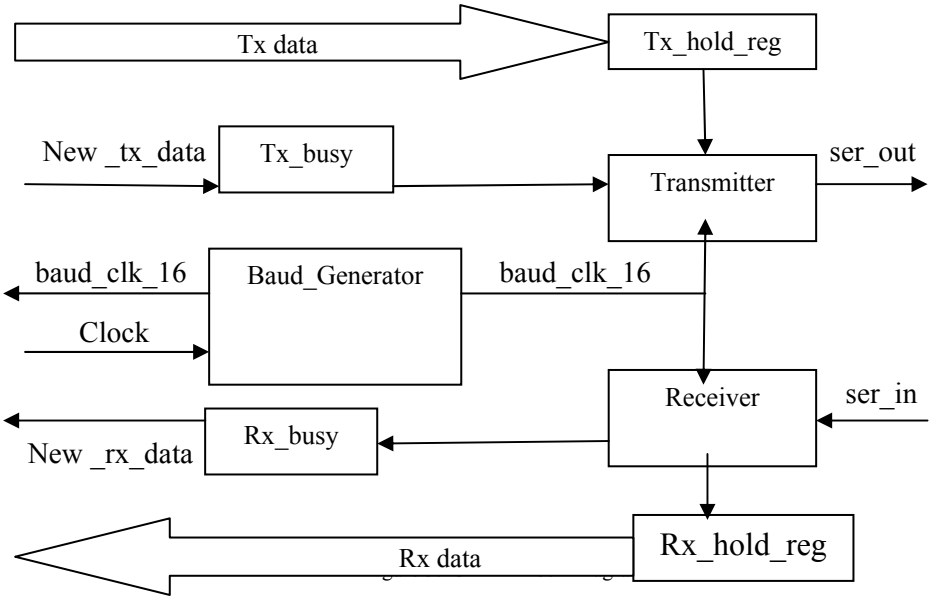
UART design mainly contains three blocks those are Transmitter, Receiver and Baud Rate Generator, plus two status registers tx_busy and rx_busy as shown in Fig. 3. The baud generator deals with the generation of baud frequency clock pulses from the input clock. The transmitter and receiver block, which perform the data transfer, are independent of each other and hence this design can achieve full duplex communication.

1)Baud Generator :

The baud generator is used to produce a local clock signal which is much higher than the baud rate to control theUART receive and transmit. It is a actually a frequency divider. The baud frequency factor can be calculated according to the given system clock frequency and the requested baud rate. In this design, the frequency clock produced by the baud rate generator is not the baud rate clock, but 16 times the baud rate clock. The faster clock allows the receiver to align the sampling pulse as desired and provides a faster response time from the transmitter. The baud clock frequency and the baud limit are calculated as per given Equations.

$$\text{Baud\_Freq} = \frac{16.\text{BR}}{\gcd(\text{Clock\_Freq}, 16.\text{BR})}$$

$$\text{Baud Limit} = \frac{\text{Clock\_Frequency}}{\gcd(\text{Clock\_Freq},16.\text{BR})} - \text{Baud\_Freq}$$



2) Transmitter block:

The function of transmit module is to convert the sending 8 bit parallel data into serial data, adds start bit at the head of the data as well as parity and stop bits at the end of the data. A new data transfer in the UART is initiated by the new_tx_data, which indicates the availability of the tx_data. This data is stored in the tx hold_reg in the next clock cycle. The parallel stream of data (tx_data) is converted into a serial stream using a shift register. The start of the shift operation sets the internal status register tx_busy to indicate a new data transfer. tx_busy can also be used as data validity signal for clock gating the transmitter block. The shift operation follows at the baud clock frequency to send the data out on the ser_out data line. A counter of 16 is used to generate the rising edge for the baud clock from the 16× the baud clock provided by the baud generator.

The status ofany data transfer is maintained by a status register to indicate a valid/invalid operation indicated by an interrupt signal. The resetting of the tx_busy status register is detected by another counter of 10 which counts the number of bits transmitted.

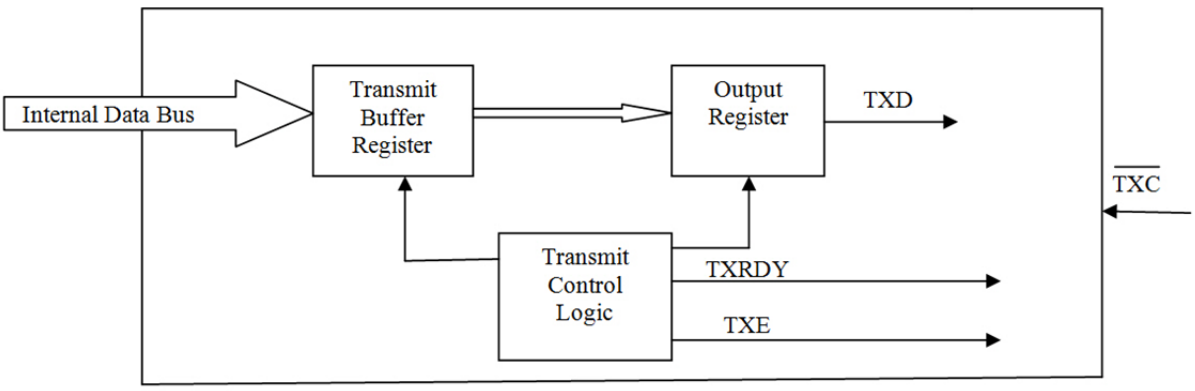


Figure 4. Transmitter Block

3) Receiver Block:

During the UART reception , the serial data and the receiving clock are asynchronous, so it is very important to correctly determine the start bit of a frame data. This block monitors the input line for the new data which is indicated by the start bit. The rx_busy register is set when the start bit is recognized. This block operates at two different edges of the baud clock pulse generated by a 16 bit shift counter from the 16× baud clock.

The sampling and shifting at the shift register occurs atthe middle of the incoming data pulse at the falling edge of the baud clock pulse which is generated at the count of 8 from the counter. After receiving the end bit of the ser_ in data, the serial data is moved to the rx_ hold_ reg register at the rising edge of the baud clock frequency generated at the count of 16 on the counter. Thus a parallel stream of 8-bit data (rx__ data) is generated which is indicated by the new_rx_data interrupt signal. There is also 3-bit counter to track the number of input bits received .[5,6,7]

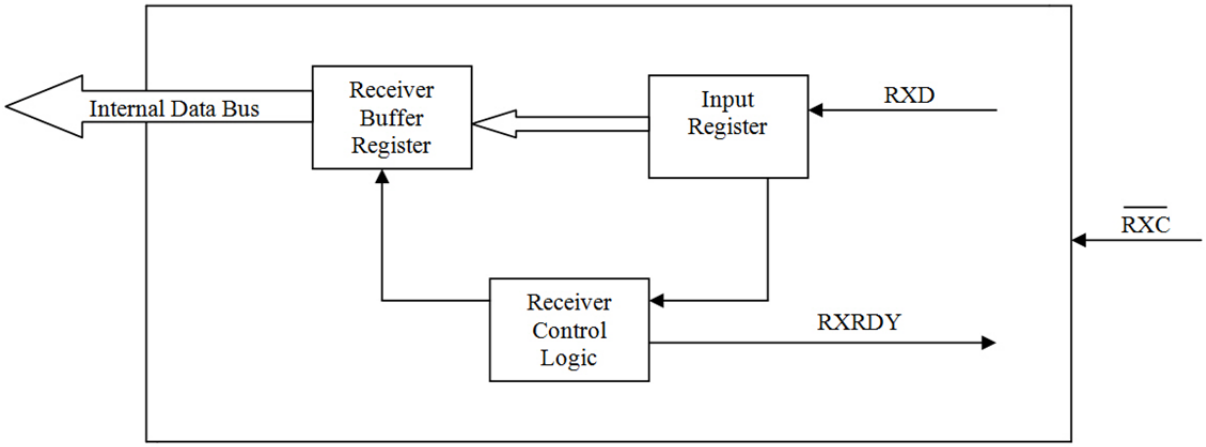


Figure 5. Receiver Block

IV. SIMULATION RESULT AND RTL VIEW OF TOP FILE

In this paper , the system clock frequency is 50 MHZ , baud rate is 19200bps and then the output clock frequency of baud rate generator should be 1*19200. Therefore the frequency coefficient

(M) =50 MHZ/1*19200=2604

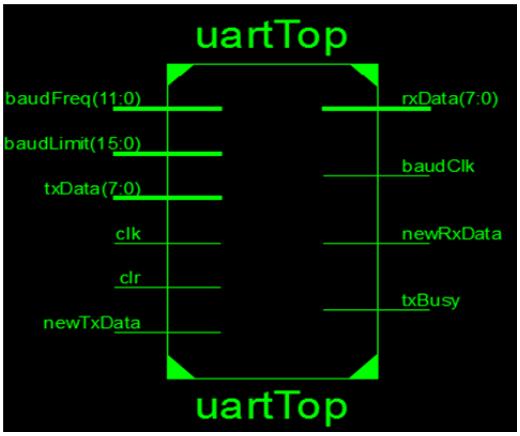


Figure 6. RTL View Of Top File

The simulation result shows the serial transmission and reception of 8 bit data .We have selected S0=1 and S1=0 to select the baud rate of 19200bps. It shows the output for the selected baud rate. In this paper we transmit 11001100 and at the receiver side we get the same output.

Figure 7. Simulation Waveform

V. RESULT

The proposed UART has been simulated and synthesized on the Xilinx ISE 14.2. and the desired output as shown in the figure which shows the transmission of 8 bit data. The synthesis result contains a table which shows the comparison between the old research and the proposed UART. While doing comparison we will find that our research shows great significance as our parameter is consuming less power to operate.

Device Utilization Summary			
Logic Utilization	Used	available	utilization
No. of Slice Registers	19	4800	0%
No. of Slice LUTS	23	2300	0%
No. of Fully used LUT-FF pairs	19	24	82%
No. of Bonded IOBS	14	102	13%

Table 1. Synthesis Result

ACKNOWLEDGMENT

We would like to acknowledge the faculties of Electronic & Telecommunication department, Disha college of Engineering and Technology, Raipur for their support. I Manju Wadhvani specially want to thank my guide professor and H.O.D. Zoonubiya Ali for their valuable guidance and constant encouragement towards the work.

REFERENCES

[1] Patel; Vatsalkumar Patel; Vikaskumar Patel, "VHDL Implementation of UART with Status Register", Sarvajanik College of Engg & Tech Surat, India.

[2] Chun-zhi Xia Yin-shui Wang Lun-yao "A Universal Asynchronous Receiver Transmitter Design" 978-1-4577-0321-8/11/26.00©2011 IEEE.

[3] FANG Yi-yuan, CHEN Xue-jun "Design and Simulation of UART Serial Communication Module based on VHDL"978-1-4244-9857-4/11/26.00© 2011 IEEE.

[4] Ms. Neha R. Laddha, Prof. A.P. Thakare "Implementation of Serial Communication using UART with configurable baud rate" International Journal on Recent and Innovation Trends in Computing and Communication volume1 Issue: 4 April 2013.

[5] Mangesh V. Benodkar, Prof. Umesh W. Hore "Design and simulation of Universal Asynchronous Receiver Transmitter on Field Programmable Gate Array Using VHDL"International Journal of Scientific Research Education volume:2 Issue: 7 July 2014.

[6] Dipanjan Bhadra, Vikas S.Vij, Kenneth S. Stevens "A Low Power UART Design Based On Asynchronous Techniques" 978-1-4799-00666-4/13/31.00© 2013 IEEE.

[7] B.Venkataramana, P.Shrikanth Reddy, P. kishore Kumar, Prof. K Sivasankaran "ASIC Implementation of Universal Asynchronous Receiver Transmitter using 45nm Technology " International Journal of Engineering and Technology volume:5 Issue: 3 July 2013.