Design and FPGA Implementation of a High Speed UART

Sonali Dhage, Manali Patil,Navnath Temgire,Pushkar Vaity, Sangeeta Parshionikar

Abstract- The Universal Asynchronous Receiver Transmitter (UART) is a device used for serial communication between computers and other peripheral devices. This paper deals with designing of a high speed UART using Verilog Hardware Description Language. The designed UART is a full duplex UART and it has a 10-bit frame format with a start bit, 8-data bits and one stop bit. The UART also has configurable baud rates. Buffers are used to hold the data temporarily during communication and the FIFO shift registers are used to shift the data in and out of the communicating devices. The different baud rates and the high oversampling rate at the receiver are the factors which make this design compatible for high speeds. The transmitter and receiver are designed in Verilog HDL, simulated and synthesized using Xilinx ISE 14.5. The design is successfully downloaded and verified on Spartan – 3E Digilent Basys2 FPGA board.

Index Terms – Baud rate, Oversampling rate, Receiver, Shift registers, Transmitter, Verilog

1 INTRODUCTION

The UART is an abbreviation for Universal Asynchronous Receiver Transmitter. The UART is a combination of hardware which is used for serial communications over a computer or over any peripheral device serial port. The UART is generally used continuatively with the communication standards such as RS-232 or RS-485. As we know, the processing of data in our personal computers or any other peripheral devices happens in the parallel form as it ensures speed. But, when the data in these individual systems is to be communicated to the outside devices, it is to be converted into a serial form as communication of parallel data proves to be cost inefficient. Hence for this purpose, we need a device which is called as the UART. This device (UART) is completely responsible for breaking the parallel native data in any sending system into a serial form and then it is again used to convert this very same data into parallel form at the receiving device.

This device is designated as 'universal' because of the fact that, the transmission speeds and the data formats can be configured as per the requirements. Also the designation 'asynchronous' is given because, the transmitter and the receiver are not synchronized by a clock signal. , the start and the end of the data in UART's is indicated by sending 2 extra bits namely 'Start bit' and the 'Stop bit'[1].

- *Sonali Dhage, Manali Patil,Navnath Temgire,Pushkar Vaity are currently pursuing bachelors degree program in electronics engineering in Mumbai University, India.*
  E-mail: sonalidhage91 @gmail.com
- *Sangeeta Parshionikar is currently an assistant professor with Department of Electronics Enginneering, FRCRCE, Mumbai University, India.*
  E -mail: sangeeta@frcrce.ac.in

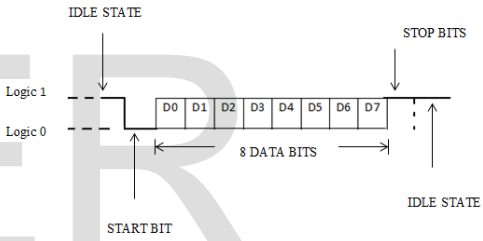

Fig.1.1 Basic frame format for UART communication

The figure above shows the basic frame format used for the UART communications. The line on which the data bytes are to be sent is held at logic 1 when there is no data on the line, or in other words, when the line is idle. As discussed earlier, there is no clock signal sent between the transmitter and the receiver for synchronization. The synchronization between the transmitter and the receiver is done by the start and the stop bits. Initially the line is held high when it is idle. When the data byte is to be sent, the start bit is added before the original data bytes by the transmitter. This start bit is logic 0 bit. When the logic 0 signal on the line is read by the receiver, it is synchronized to the transmitter. The receiver comes to know that a data is about to come. Then the original data is sent after the start bit and then the end of data is indicated by a stop bit which is logic 1 bit. This is the basic format in which the data is transmitted and received between 2 UART's. The stop bits and the number of data bits in a UART frame are configurable. There can be up to 1 or 2 stop bits and from 5-8 data bits in any UART frame. Also to ensure that the data which is received is same as the data which is transmitted, an optional parity bit can also be transmitted along with the UART frame. This parity bit is sent after the

data bits and before the stop bit. The parity bit can be 1 or 0. The parity of the data can be either odd or even. The parity of the transmitted data and the received data should be same. If the parity of the data's at both the ends are different, then it indicates that an error occurred during transmission and hence the whole data byte is to be transmitted again.

2 PROPOSED WORK

The main components of UART such as transmitter and receiver are described below:

2.1 UART TRANSMITTER MODULE

All the operations of the transmitter are with respect to a clock which runs at a multiple of the baud rate, typically 4xBaud rate. The transmitter has a transmission buffer and a shift register. The data to be transmitted is stored in the transmission buffer and it is given to the shift register when it is to be transmitted. One bit data is shifted out of the shift register every clock cycle. The transmitter design consists of a clock divider to generate the transmission clock from the clock source available, a shift register to shift the data out of the transmitter and a finite state machine design to control the operation of the transmission system. While transmission, the LSB bit is transmitted first after the start bit. Each bit is transmitted for the same amount of time (duration of each bit is same). The host system cannot deposit a new data byte into the shift register unless and until the previous data byte has been transmitted successfully. Hence a status bit is to be maintained at the transmitting side to indicate the readiness for transmission. This can be done by incorporating an interrupt for signaling. The most important thing which should be kept into mind before transmitting the data is that, the baud rates at both the transmitting and the receiving end should be the same to ensure proper transmission and reception of data [2]. Also the character lengths, parity bit and the stop bits should be configured properly at both the ends.

The different parts of this transmitter module are as follows:

- Clock divider for the transmitter (Prescaled counter).
- 9-bit shift register.
- FSM (Finite State Machine) showing the states of the transmitter.

The block diagram of transmitter is shown below:

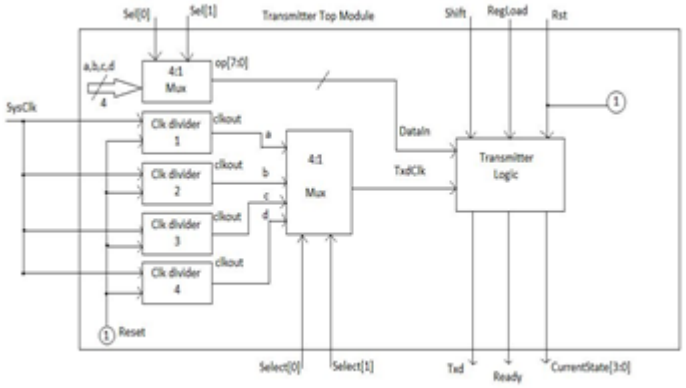

Fig.2.1 Block Diagram of Transmitter

The clock frequency available on the FPGA board is 50MHz. Now we need to select a baud rate of 9600bps and the frequency of the transmitter is 4xBaud rate. This comes out to be 4x9600=38400Hz. Hence the System clock is divided to achieve a clock frequency of 38400Hz for the operation of the transmitter. Thus the frequency division coefficient comes out to be 1302. The formula used for this is as follows:

Frequency division coefficient = 50MHz /4x9600

The shift register is used to shift the data out of the transmitter on to the Txd pin on the positive edge of every transmitter clock pulse. As the logic level on the UART transmission line is high when it is in idle state, all 1's are written to the shift register in case of reset condition.

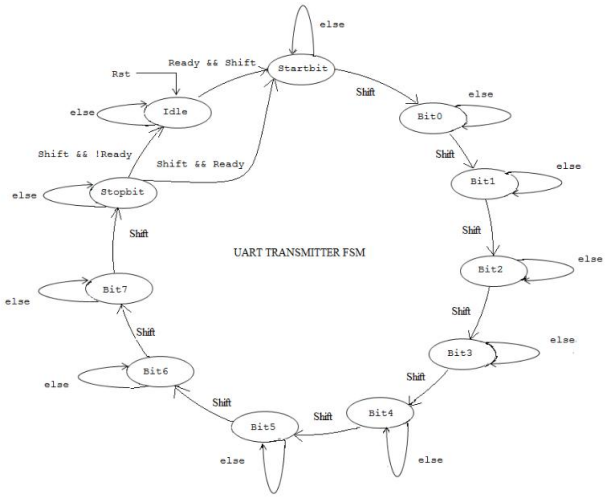

Fig.2.2 FSM for the Transmitter

The FSM is a design approach which shows that how many states a system is composed of and it shows in which state the system presently is and it also shows the next state for the system designed. The FSM of the transmitter is designed to consist of 11 states. These 11 states correspond to the 11 bits in the data frame of the UART

Initially, at reset, the system is in the idle state. Only when

the ready and the shift signals are high, the data is loaded into the shift register for transmission and the next state is the start bit state. During consecutive positive edges of the transmitter clock, if the shift signal is high, then the current state of the system becomes the next state or else, the system stays in the previous or the same state itself. In this way the system transitions from idle state to the stop bit state. Now if the start bit and the ready bits are both high, then the transmission of the next data frame begins and the system moves on to the start bit state. Conversely if the ready signal is not high, then the system moves on to the idle state and waits for the ready bit to become high.

2.2 UART RECEIVER MODULE

The receiver design is a bit complex as compared to its transmitter counterpart. The receiver clock is typically 8xBaud rate. The main challenge at the receiver is to read the data successfully before the next data arrives. The receiver reads the data by sampling it in the middle of each bit. This ensures that the read data is really the original data bit and-not a spurious noise signal. The reception of the data bits at the receiver can be shown with an example. Consider the baud rate at the transmitter is 9600 bps. This means that the time period for one data bit is 1/9600 which comes out to be 104 usec. Hence now we know that the baud rate at the receiver needs to be same for proper reception. Thus now, the receiver will sample the data bits in the frame at their center i.e. at 52 usec.

The different parts of this receiver module are as follows:

- Clock divider for the receiver (Prescaled counter).
- 8-bit shift register.
- FSM (Finite State Machine) showing the states of the receiver.

The figure 2.3 shows block diagram of receiver.
The clock frequency available on the FPGA board is 50MHz.
The frequency of our transmitter is 38400Hz. We have selected the frequency of our receiver as 8 times that of the transmitting frequency which comes out to be 307200Hz. Hence we divided the System clock such that we achieved a clock frequency of 307200Hz for the operation of the receiver.
Thus the frequency division coefficient comes out to be 163.
The formula used for this is as follows:

Frequency division coefficient = 50MHz /307200Hz

The shift register is used to shift the data on to the Rxd pin into the receiver on every positive edge of the reception clock. All the 8 bits of the frame are first accumulated into the shift register and once the complete frame is received without any framing error, then this data is stored into the receiver memory.

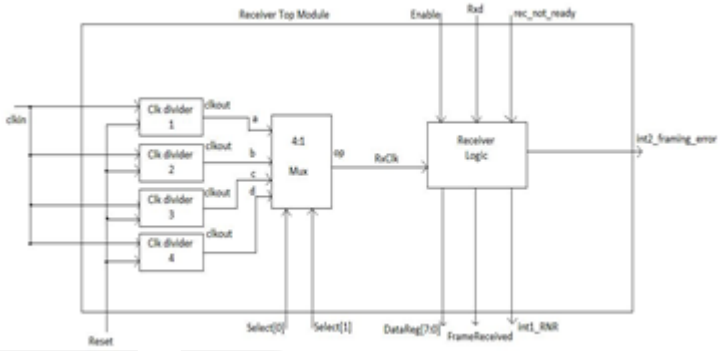

Fig.2.3 Block Diagram of Receiver

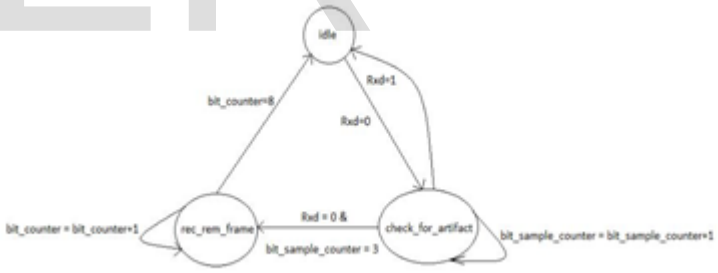

Fig.2.4 FSM for the receiver

The receiver finite state machine consists of 3 states. Initially the machine is in the idle state when there is no data on the Rxd pin. Now whenever the receiver comes across the start bit, then it has to make sure whether the start bit is really a start bit and not the result of any noise. Because the Rxd pin can go low momentarily due to any noise in the link from the transmitter to the receiver. Thus now the machine goes into the next state to check for noise. If there is any noise, then the system goes again into the idle state. This can be seen from the figure above. Else, the start bit is sampled and the system moves on to the next state to receive the rest of the frame which is nothing but the data byte and the stop bit. In this state, the data bits are received one by one until all the 8 bits are received. After the data bits are received, the stop bit is received and once the stop bit is received properly, the system goes into the idle state again until and unless the next start bit arrives.

The reception of the data is done using a method called oversampling. The transmitted signal can be successfully reconstructed at the receiving side if the sampling frequency is greater than the Nyquist rate which is nothing but twice the maximum frequency. In oversampling, the received signal is sampled at a rate which is significantly higher than the Nyquist rate. Now, for example if the sampling frequency is 8 times the maximum frequency, then the signal is said to be oversampled by a factor of 8.

3. RESULTS

The figures below show the simulation results of our designed transmitter and receiver. The results obtained in terms of speed for our UART design are as under. The transmitter and receiver design is successfully synthesized using Xilinx ISE 14.5 simulator. The design is downloaded and verified on Spartan – 3E Digilent Basys2 FPGA board [7].

Fig.3.1 Waveforms of Transmitter

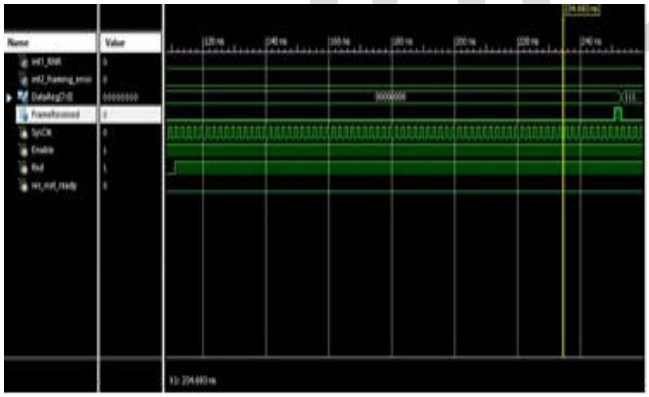

Fig.3.2 waveforms of receiver

'Baud rate' and 'bits per second' (bps) are the two terms which should not be interchanged. Baud rate specifies the number of times the signal change occurs from +v to –v on a particular link and 'bits per second' is the term used to give the count of the number of data bits travelling over a particular link. Hence the speed of the UART is in terms of bits per second (bps).
The different baud rates which we have used in our design are 9600, 57600, 115200 and 230400.

These results are summarized as under:

Table 1. UART speeds for different Baud rates

Baud Rate	9600	57600	115200	230400
Time period of 1 frame (10 bits)	52.12 μsec	34.76 μsec	17.4 μsec	8.72 μsec
Speed (bps)	191.865 Kbps	287.687 Kbps	574.712 Kbps	1.147Mbps

This UART can be demonstrated for the speed of 1.147Mbps at a baud rate of 230400. But this UART can be used for even higher speeds over other high speed links.
For baud rate of 25000000 for the 50 MHz clock on board, the 10 bit UART frame is equal to 40 ns. Thus one bit corresponds to 4 ns. Thus the speed is calculated by 1/4 ns which comes out to be 250Mbps. But oversampling won't be possible in this case as the receiver frequency will be just equal to that suggested by the Nyquist rate. Hence the reconstruction of data cannot be guaranteed in this case.
Hence for baud rate of 12500000, the 10 bit UART frame is still equal to 40 ns. Thus one bit still corresponds to 4 ns. Thus the speed remains 250Mbps but oversampling can be done in this case by a factor of 4 at the receiver. In this way, the reconstruction is possible at a speed as high as 250Mbps.
We also observed from the simulation that there is a trade-off between oversampling factor and the baud rate. If we increase the baud rate to increase speed, then the oversampling factor is to be reduced in order to achieve high baud rate.
Similarly, if we increase the oversampling factor, then we have to reduce the baud rate in order to increase the speed of data coming to the FPGA.

4. CONCLUSION

In this paper, UART with configurable baud rates and the high oversampling rate at the receiver is proposed.The transmitter and the receiver module of the UART using the structural approach is designed and successfully synthesized the same using Xilinx ISE 14.5. The UART with variable baud rates is successfully simulated and the design has been verified on Xilinx Spartan-3E FPGA. The design is compatible for high speed due to different baud rates and the high oversampling rate at the receiver. A maximum speed of 250Mbps is possible using this UART design.

5. REFERENCES

[1] FANG Yi-yuan, CHEN Xue-jun, "Design and Simulation of UART Serial communication Module using VHDL", *3rd Int'l Workshop on Intel. Sys. and App. (ISA 2011),* Wuhan, China, May 2011. 978-1-4244-9857-4/11.

[2] Badam Suresh, P. Teja Reddy, V.S.V. Srihari and S. Sivanantham, 'ASIC Implementation of Low Power Universal Asynchronous Receiver Transmitter', *World Applied Sciences Journal* 32 (3): 472-477, 2014, ISSN 1818-4952

[3] Ritesh Kumar Agrawal, Vivek Ranjan Mishra, 'The Design of High Speed UART', *Proceedings of 2013 IEEE Conference on Information and Communication Technologies* (ICT 2013), 978-1-4673-5758-6/13.

[4] Nennie Farina Mahat, 'Design of a 9 bit UART module based on Verilog HDL' *IEEE-ICSE , 2012 Proceedings*2012, Kuala Lumpur, Malaysia, 978-1-4673-2396-3/12.

[5] J. Norhuzaimin, and H.H Maimun, " The design of high speed UART," *Asia Pac. Conf. on Appl. Electromagnetics* (APACE 2005), Johor, Malaysia, Dec. 2005.

[6] Mohd Yamani Idna Idris and Mashkuri Yaacob, "A VHDL implementation of BIST technique in UART design", *Faculty of Computer Science And Information Technology,* University of Melaya, 2003.

[7] Xilinx Spartan-3E FPGA Series reference manual.

[8] Digilent Basys2 board reference manual.