IPHV7I3025X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

DESIGN AND SIMULATION OF UART SERIAL COMMUNICATION PROTOCOL BASED ON VHDL

[#1] SUMEDHA LAXMI.B.S - M.Tech Student,
[#2]K.HARI BABU - Assistant Professor,
Dept of ECE,
MLR INSTITUTE OF TECHNOLOGY, DUNDIGAL, HYD, T.S., INDIA.

Abstract: UART (Universal Asynchronous Receiver Transmitter) is a type of serial communication protocol; mostly used for low speed, short distance and low cost data exchange between computer and peripherals. UART includes three important modules which are the baud rate generator, receiver and transmitter. The UART design in this paper is used for communication between FPGA (Field Programmable Gate Array) and TDC (Treatment Delivery Controller). The UART design consists of one start bit, 8 data bit and one stop bit. The VHDL code is written and simulated in XILINX 13.9 and tested using SPARTAN 6LX9 TQG144. A UART is usually an individual (or part of an) used for over a computer or peripheral device. UARTs are now commonly included in microcontrollers. A UART (Universal Asynchronous Receiver/Transmitter) is the microchip with programming that controls a computer's interface to its attached serial devices. 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. Serial transmission is commonly used with modems and for non-networked communication between computers, terminals and other devices. The Universal Asynchronous Receiver/Transmitter (UART) takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. Each UART contains which is the fundamental method of conversion between serial and parallel forms. Serial transmission of digital information (bits) through a single wire or other medium is much more cost effective than parallel transmission through multiple wires. This project proposes the hardware implementation of vlsi architecture for UART that have been modified to improve performance. This project was implemented in verilog the coding is done in Verilog HDL and the synthesis is done using Xilinx Spartan library.

Index Terms—Baud Rate, Configurable Logic Blocks (CLBs), Field Programmable Gate Array (FPGA), Input Output Blocks (IOBs),Serial communication,Treatment Delivery Contoller (TDC), UniversalAsynchronous Reciever Transmitter (UART).

I.INTRODUCTION

Serial communication is a popular means of transmitting data between a computer and peripheral devices such as a programmable instruments or even another computer. Serial communication uses a transmitter to sends data, one bit at a time, over a single communication line to a receiver. This is contrast to parallel communication, where all the bits of each symbol are sends together. Serial communication buses are becoming more common as improved technology enables them to transfer data rate at high speeds.

Serial communication is popular because most computers have one or more serial ports, so no extra hardware is needed then a cable to connect the instruments to the computer or two computers together. Serial communication requires four important parameter: the baud rate, start bit, optional parity bit and stop bit. Asynchronous serial communication has ad-vantages of less transmission line, high reliability and long transmission, therefore is widely used in data exchange between computer and peripherals. Asynchronous serial communication is usually implemented by Universal Asynchro-nous Receiver and Transmitter (UART) [1]-[2].

A UART is a Universal Asynchronous Receiver Transmitter, which is used to communicate between two devices. Most com-puters and microcontroller includes one or more serial I/O de-vices, such as keyboard or serial printer. UART allows full- duplex communication in serial link, thus has been widely used in the data communication and control system. In actual application, usually only a few key features of UART are needed [3].

Asynchronous serial communication has advantages of less transmission line, high reliability, and long transmission distance, therefore is widely used in data exchange between computer and peripherals. Asynchronous serial communication is usually implemented by Universal Asynchronous Receiver Transmitter (UART) [1]. UART allows full-duplex communication in serial ink, thus has been widely used in the data communications and control system. In actual

applications, usually only a few key features of UART are needed. Specific interface chip will cause waste of resources and increased cost. Particularly in the field of electronic design,SOC technology is recently becoming increasingly mature. This situation results in the requirement of realizing the whole system function in a single or a very few chips. Designers must integrate the similar function module into FPGA. 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, which effectively solves the above problem [2] [3].

Basic UART communication needs only two signal Basic UART communication needs only two signal lines(RXD, TXD) to complete full-duplex data communication.TXD is the transmit side, the output of UART; RXD is the receiver, the input of UART. UART's basic features are: There are two states in the signal line, using logic 1 (high) and logic 0(low) to distinguish respectively. For example, when the transmitter is idle, the data line is in the high logic state. Otherwise when a word is given to the UART for asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted.

The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter. These two

clocks must be accurate enough to not have the frequency drift by more than 10% during the transmission of the remaining bits in the word [4].

After the Start Bit, the individual data bits of the word are sent, with the Least Significant Bit (LSB) being sent first. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits, and the receiver "looks" at the wire at approximately halfway through the period assigned to each bit to

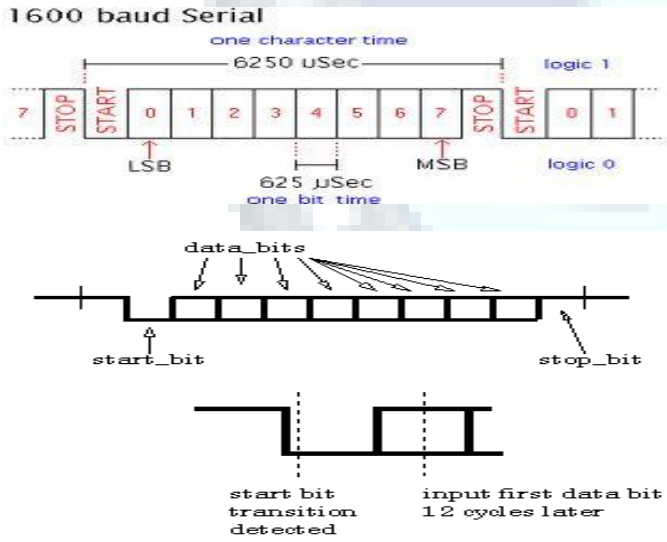
IPHV7I3025X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

determine if the bit is a 1 or a 0. For example, if it takes two seconds to send each bit, the receiver will examine the signal to determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on.

When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates. The Parity Bit may be used by the receiver to perform simple error checking. Then at least one Stop Bit is sent by the transmitter. When the receiver has received all of the bits in the data word, it may check for the Parity Bits (both sender and receiver must agree on whether a Parity Bit is to be used), and then the receiver looks for a Stop Bit. If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be garbled and will report a Framing Error to the host processor when the data word is read. The usual cause of a Framing Error is that the sender and receiver clocks were not running at the same speed, or that the signal was interrupted.

Regardless of whether the data was received correctly or not, the UART automatically discards the Start, Parity and Stop bits. If the sender and receiver are configured identically, these

bits are not passed to the host. If another word is ready for transmission, the Start Bit for the new word can be sent as soon as the Stop Bit for the previous word has been sent. Because asynchronous data are "self-synchronizing", if there are no data to transmit, the transmission line can be idle. The UART frame format is shown in Fig. 1.



In this paper, the top to bottom (Top to Down) designmethod is used. The UART serial communication module I divided into three sub-modules: the baud rate generator,receiver module and transmitter module, shown in Fig. 2.Therefore, the implementation of the UART communicationmodule is actually the realization of the three sub-modules. Thebaud rate generator is used to produce a local clock signalwhich is much higher than the baud rate to control the UARTreceive and transmit; The UART receiver module is used toreceive the serial signals at RXD, and convert them intoparallel data; The UART transmit module converts the bytesinto serial bits according to the basic frame format and transmits those bits through TXD.

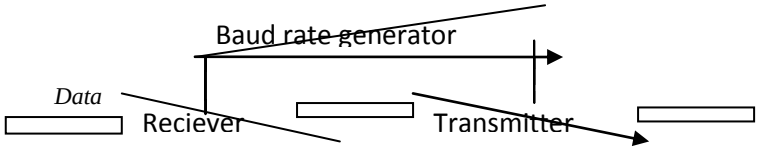


Figure 2. UART Module

A. Baud Rate Generator

Baud Rate Generator is actually a frequency divider. Thebaud rate frequency factor can be calculated according to agiven system clock frequency (oscillator clock) and therequested baud rate. The calculated baud rate frequency factoris used as the divider factor. In this design, the frequency clockproduced by the baud rate generator is not the baud rate clock,but 16 times the baud rate clock. The purpose is to preciselysample the asynchronous serial data at the receiver. Assumethat the system clock is 32MHz, baud rate is 9600bps, and thenthe output clock frequency of baud rate generator should be 16* 9600Hz. Therefore the frequency coefficient (M) of the baudrate generator is:

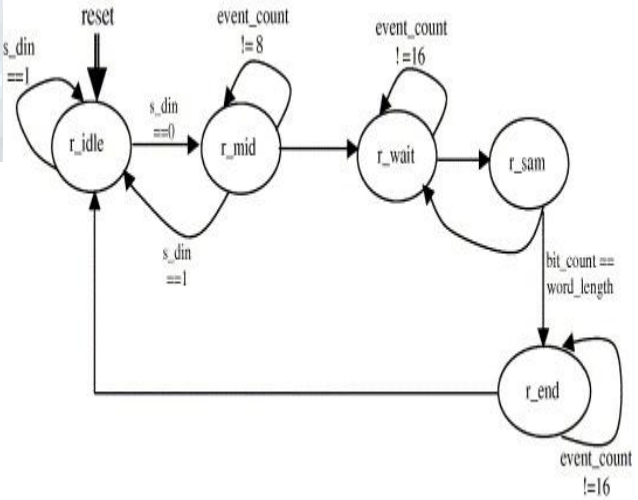
M =32MHz/16*9600Hz=208

When the UART receives serial data, it is very critical todetermine where to sample the data information. The ideal timefor sampling is at the middle point of each serial data bit. Inthis design, the receive clock frequency is designed to be 16times the baud rate, therefore, each data width received byUART is 16 times the receive clock cycle.

The baud generator is capable of creating a clock by dividing the system clock by any divisor from 2 to 216-1. The divisor is the unsigned value stored in the Divisor Register. The output frequency of the baud generator is 16 times the baud rate. This means, the Divisor Register must hold a value equal to the system clock divided by baud rate and further divided by 16. This output clock can be used as the receive reference clock by the receiving UART.

B. Receiver Module

During the UART reception, the serial data and thereceiving clock are asynchronous, so it is very important tocorrectly determine the start bit of a frame data. The receivermodule receives data from RXD pin. RXD jumps into logic 0from logic 1 can be regarded as the beginning of a data frame.When the UART receiver module is reset, it has been waitingthe RXD level to jump. The start bit is identified by detectingRXD level changes from high to low. In order to avoid the



misjudgment of the start bit caused by noise, a start bit error detect function is added in this design, which requires the received low level in RXD at least over 50% of the baud rate to be able to determine the start bit arrives. Since the receive clock frequency is 16 times the baud rate in the design, the RXD low level lasts at least 8 receiving clock cycles is considered start bit arrives. Once the start bit been identified, from the next bit, begin to count the rising edge of the baud clock, and sample RXD when counting. Each sampled value of the logic level is deposited in the register rbuf [7, 0] by order.

IPHV7I3025X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

When the count equals 8, all the data bits are surely received, also the 8 serial bits are converted into a byte parallel data..The serial receiver module includes receiving, serial and
parallel transform, and receive caching, etc. In this paper weuse finite state machine to design, shown in Fig. 3.
The state machine includes five states: R_START (waitingfor the start bit), R_CENTER (find midpoint), R_WAIT(waiting for the sampling), R_SAMPLE (sampling), and
R_STOP (receiving stop bit).

- R_START Status: When the UART receiver is reset, thereceiver state machine will be in this state. In this state, thestate machine has been waiting for the RXD level to jump over from logic 1 to logic 0, i.e. the start bit. This alerts the beginning of a new data frame. Once the start bit is identified, the state machine will be transferred to R_CENTER state. In Fig. 3, RXD_SYNC is a synchronization signal of RXD. Because when sampling logic 1 or logic 0, we do not want the detected signal to be unstable. So we do not directly detect RXD signal, but detect the synchronization signal RXD_SYNC.
- R_CENTER Status: For asynchronous serial signal, in order to detect the correct signal each time, and minimize the total error in the later data bits detection. Obviously, it is the most ideal to detect at the middle of each bit. In this state, the task is to find the midpoint of each bit through the start bit. The method is by counting the number of bclkr (the receiving clock frequency generated by the baud rate generator) In addition, the start bit detected in the R_START may not be a really start bit, it may be an occasional interference sharp pulse (negative pulse). This interference pulse cycle is very short. Therefore, the signal that maintains logic 0 over 1 / 4 bit time must be a start bit.
- R_WAIT Status: When the state machine is in this state, waiting for counting bclkr to 15, then entering into R_SAMPLE to sample the data bits at the 16th bclkr. At the same time determining whether the collected data bit length has reached the data frame length (FRAMELEN). If reaches, it means the stop bits arrives. The FRAMELEN is modifiable in the design (using the Generic). In this design it is 8, which corresponds to the 8-bit data format of UART.
- R_SAMPLE Status: Data bit sampling. After sampling the state machine transfers to R_WAIT state unconditionally, waits for the arrival of the next start bit.
- R_STOP Status: Stop bit is either 1 or 1.5, or 2. Statemachine doesn't detect RXD in R_STOP, but output frame receiving done signal (REC_DONE <= '1 '). After the stop bit, state machine turns back to R_START state, waiting for the next frame start bit.

C. Transmit Module

The function of transmit module is to convert the sending8-bit parallel data into serial data, adds start bit at the head of the data as well as the parity and stop bits at the end of the data. When the UART transmit module is reset by the reset signal, the transmit module immediately enters the ready state to send. In this state, the 8-bit parallel data is read into the register txdbuf [7: 0]. The transmitter only needs to output 1 bite very 16 bclkt (the transmitting clock frequency generated by the baud rate generator) cycles. The order follows 1 start bit, 8data bits, 1 parity bit and 1 stop bit. The parity bit is determined according to the number of logic 1 in 8 data bits. Then the parity bit is output. Finally, logic 1 is output as the stop bit.

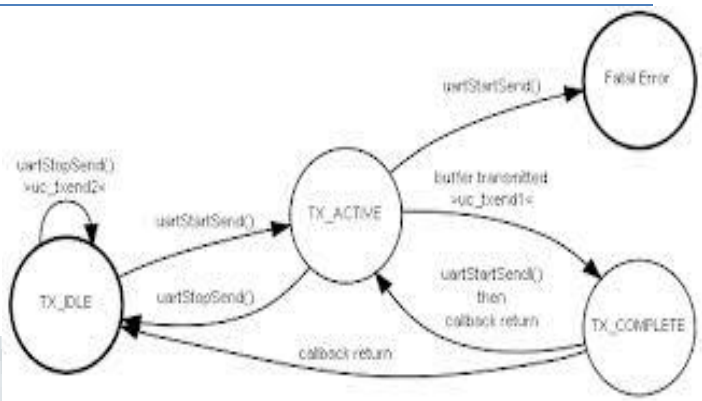
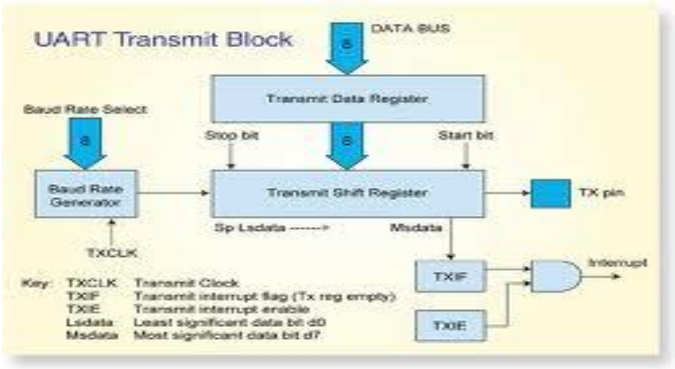


Fig.4 shows the transmit module state diagram. This state machine has 5 states: X_IDLE (free), X_START
(start bit), X_WAIT (shift to wait), X_SHIFT (shift), X_STOP (stop bit).

- X_IDLE Status: When the UART is reset, the state machine will be in this state. In this state, the UART transmitter has been waiting a data frame sending command XMIT_CMD. XMIT_CMD_P is a processed signal of XMIT_CMD, which is a short pulse signal. Since XMIT_CMD is an external signal, outside FPGA, its pulse width is unable to be limited. If XMIT_CMD is valid, it is still valid after sending one UART data frame. Then the UART transmitter will think by mistake that a new data transmit command has arrived, and once again start the frame transmit. Obviously the frame transmit is wrong. Here we limit the pulse width of XMIT_CMD. XMIT_CMD_P is its processed signal. When XMIT_CMD_P = '1 ', the state machine transferred to X_START, get ready to send a start bit.
- X_START Status: In this state, sends a logic 0 signal to the TXD for one bit time width, the start bit. Then the state machine transferred to X_WAIT state. XCNT16 is the counter of bclkt.
- X_WAIT Status: Similar with the R_WAIT of UART receive state machine.
- X_SHIFT Status: In this state, the state machine realizes the parallel to serial conversion of outgoing data. Then immediately return to X_WAIT state. X_STOP Status: Stop bit transmit state. When the data frame transmit is completed, the state machine transferred to this state, and sends 16 bclkt cycle logic 1 signal, that is, 1 stop bit. The state machine turns back to X_IDLE state after sending the stop bit, and waits for another data frame transmit command.



IPHV7I3025X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

II. COMMUNICATION BETWEEN FPGAAND TDC

The UART design has a application in the field of Medical Electronics. The UART design here is for communication between FPGA and TDC.

2.1 FPGA

A FPGA is a digital integrated ciruit that is used to program to do any type of digital function. FPGA has many advantages over the other controller. The advantages are processing of information faster, controller architecture can be optimized for space and speed and it also involve parallel processing.FPGA consist of three major configurable elements: Configurable Logic Blocks (CLBs), Input Output Blocks (IOBs)and Pro-grammable Interconnects. Many manufacturers deliver FPGAs such as Quick logic, Altera, Atmel, Xilinx, etc. In this paper to write and simulate the VHDL code Xilinx 13.9 used. FPGA used in the design is SPARTAN 6 LX9 TQG144 [4].

2.2 TDC

In Medical LINAC for cancer therphyTreatment Delivery Con-troller (TDC) is used to input all operator commands and con-sists of a high resolution colour monitor and a dedicated key-board. The monitor dispalys the treatment parameters that have been entered via dedicated keyboard. Some of the im-portant parameters shown are the selected photon energy, dose, dose rate, time, field size and patient information. The dedicated keyboard is also used to start and terminate the beam and control the mechanical motion. In addition the key-board incorporates emergency button which when pressed, will shut down all power system. There are many module in the the medical LINAC such as Pulse modulator cabinet, Drive stand, Gantry, Mirror assembly, Collimator etc and to control each module FPGA is used [5]. This FPGA's are inter-face with the hardware to communicate with the TDC.

III. DESIGN OF UART

UART frame is decided initially and then the communication is started. UART frame format consist of idle bit, start bit, data bits, parity bits and stop bits. Fig. 1 shows the UART frame format for communication.

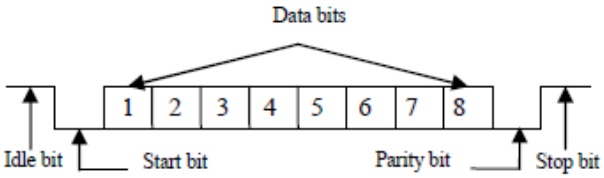


Fig.1 UART frame format

The UART serial communication module is divided into three sub-modules: Baud Clock, Transmitter module and Reciever module shown in Fig2

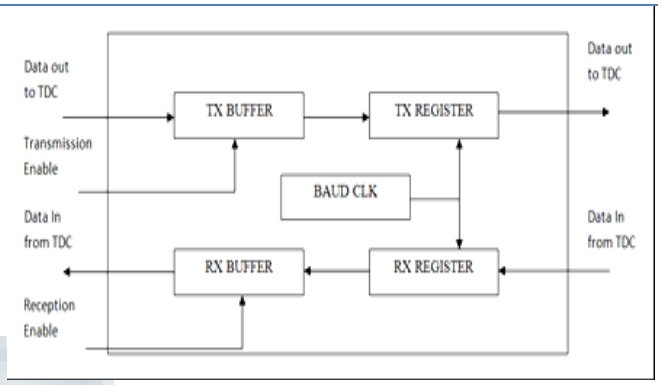


Fig.2 Block diagram for UART communication

IV. RESULTS AND DISCUSSION

UART consist of TX buffer and RX buffer. The TX buffer helps in transmission of data with the help of transmission enable signal. The transmission enable signal is enabled only when the enable bit toggle. The RX buffer helps in reception of data with the help of reception enable signal [6].

4.1 Transmission section

The Baud rate set for UART transmission and reception is 115200.

$$Time\ period = \frac{1}{Baud\ clock} \ldots\ldots\ldots (1)$$

Equation 1 shows the time period required according to the baud clock. Hence Time period for 1 bit= 8.62µs, since 8 data bit is re-quired for transmission, one start bit and one stop bit. Therefore total time period for 10 bit to transfer is 8.62*10= 86.2µs. The simulation is based on the first data is enter and then transmission is enable. After enabling the data transmission TI flag is set. For another data transmission makes enable transmission 0 and new data is entered. Again enable transmission is set to 1 to send another data hence it can send data again then make sent flag as 1.
Fig. 3 shows the RTL schematic of the transmission section. It gives a clear view of the data_in and baud clock is labeled as clk then ena-ble_trans shows the enable transmission.

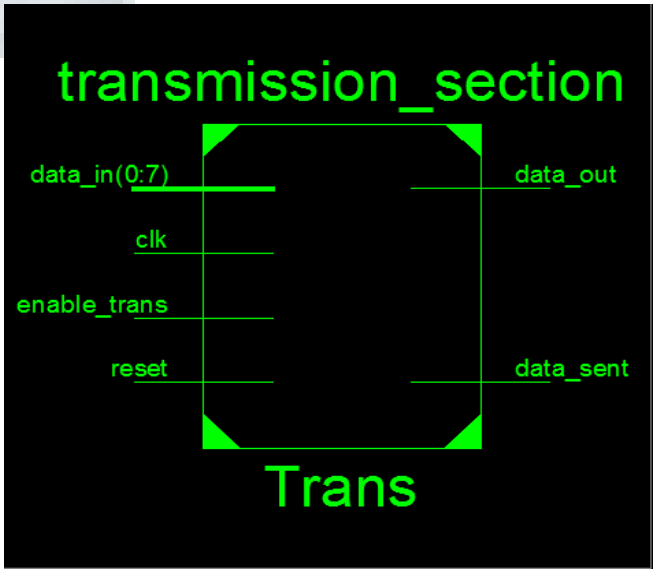


Fig. 3 RTL schematic of the UART transmission section

IPHV7I3025X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

4.2 Reception section

First input is set to 0 so as to start reception then it is latch till the last bit is received. The important part of UART reception is sampling time is set for baud clock and then data is re-ceived. Sampling time is made for better reception to get the required data. When the last bit is received as 1 the data is shifted in. Fig.4 shows the RTL schematic of the UART recep-tion. The RTL schematic shows the clear idea about the UART reception such as the clk and data_in is the input where asda-ta_out is given to TDC.

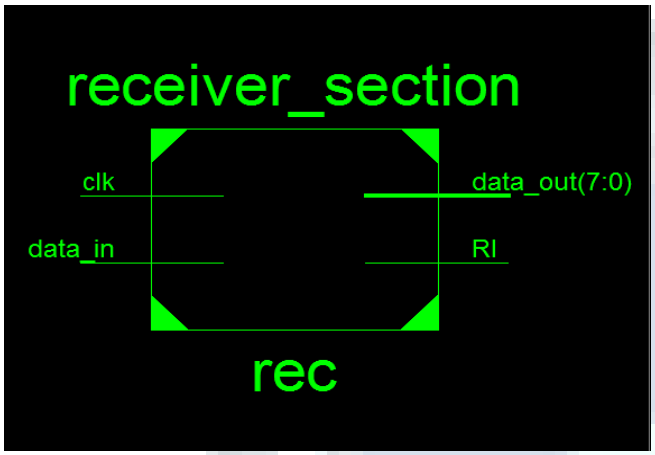


Fig. 4 RTL schemtic of UART reception

Fig.5 and Fig.6 are the simulation results of UART transmis-sion and reception. The data_in and data_out is shown with different colour so that the input and output is understood easily. Also the total time required for transmission is shown and it is 86.2µs and time to receive each bit is 8.62µs.

Fig. 5 Simulation Results for UART transmission

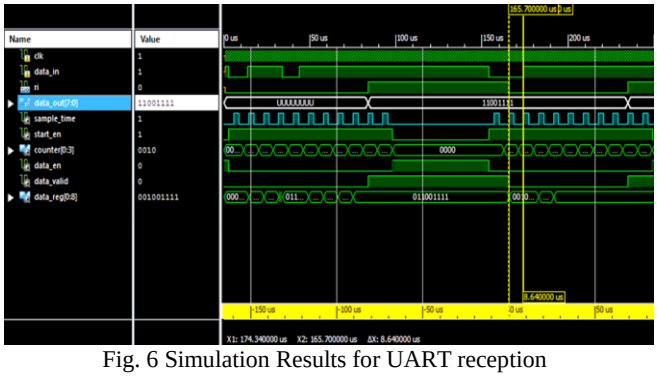


Fig. 6 Simulation Results for UART reception

V. CONCLUSION

UART is design and simulated for the communication be-tween FPGA and TDC. This design uses VHDL as the design language to achieve the modules of UART. Using XILINX 13.9 software for simulation and then it is tested on SPARTAN 6 LX9TQG 144. The results obtained are stable and reliable. The design has high integration, great flexibility, with some refern-ce value.

REFERENCES

[1] Fang Yi-Yaun, Chen Xue-jun, “Design and Simulation of UART Serial Com-minacation Module Based on VHDL”, Shanghai University of Engineering and Science, 2011.
[2] Liakot Ali, RoslinaSidek, IshakAris, AlauddinMohd. Ali, and Bam-bangSunaryo, “Design of a micro- UART for SOC application”, Computer and Electrical Engineereingvol 30, pp.257-268.
[3] L. K Hu and Q.CH. Wang, “UART-based Reliable Communication and performance Analysis”, Computer Engineering Vol 32, Dec 2006, pp. 247-249.
[4] Karen Parnell, Nick Mehta, “Programmable logic design Quck start Hand Book” , Third Edition, January 2002.
[5] Society of Applied Microwave Electronics Engineering and Research (S.A.M.E.E.R), Service and Maintenance manual, I.I.T Bombay, 2010.
[6] Dougleas Perry, “VHDL”, Third edition, 2003.