FPGA Execution of USB Transceiver Macrocell Interface with USB 2.0 Particulars

C Vinod Kumar* and Naveen Kumar M**
*M.Tech. DCE, Siddaganga institute of Technology, Tumakuru, India
cvinod.rymec@gmail.com
**Assistant professor. Department of TCE, Siddaganga institute of Technology, Tumakuru
navinvcm@gmail.com

Abstract: The universal serial bus (USB) gadgets are designed using application specific integrated circuits (ASIC) innovation with implanted USB 2.0 support. Universal serial bus transceiver macro cell interface (UTMI) is a bi-directional serial transport interface between USB gadgets through two wire information lines (D+ and D-). The USB 2.0 particulars characterize three sorts of UTMI execution relying on their information exchange rate. Those are low speed (LS) 1.5 MHz only, full speed (FS) 12 MHz only, and high speed (HS) 480 MHz/full speed (FS) 12 MHz. To handle data recovery in vendors Verilog code, the operating frequency should be low for full speed devices. The transmitting section of the UTMI sends information to various USB gadgets through information lines while receiving section of the UTMI gets information on similar information lines. The field programmable gate arrays (FPGA) execution of UTMI with HS/FS information transmission rate giving USB 2.0 details. The UTMI block is outlined utilizing Verilog code and it is incorporated, simulated, programmed to the Spartan 6 group of FPGA.

Keywords: USB, UTMI, Transmitter, Receiver, and FPGA.

Introduction

The gate arrays work easily in the vicinity of 30MHz to 60MHz, while USB 2.0 flagging running at several MHz, the current system of designing UTMI should change. It is hard to accumulate Verilog code without alteration when operating frequencies are high. The main agenda of designing UTMI is to quicken USB 2.0 fringe improvement. UTMI can transmit and get information to or from USB gadgets. The functional blocks present in the USB controller are serial interface engine (SIE), UTMI and device specific logic (DSL). The ASIC functional block of USB controller is demonstrated in fig. 1. The UTMI handles low level USB signaling and protocols.
Data serialization and deserialization, clock recovery and synchronization, bit stuffing, non-return to zero invert (NRZI) encoding are the some features involved in this block. This UTMI is designed in such a way that it has to support HS/FS, FS only and LS only UTM implementation. This has an advantage that it permits a single SIE implementation of any type of speed USB transceiver. The SIE control logic consists of USB product identification (PID), address acknowledgment and state machine logic to operate USB information and exchanges. UTMI allows SIE Verilog code to drop into an ASIC which is compatible. Macrocells or macro blocks are embedded in most of the FPGA. These are created at the transistor level. Furthermore their functionalities are like the logic cells. Logic cells are configurable logic block with D type flip-flop. These are programmed to perform arithmetic functions. Memory blocks, combinational multipliers, clock management circuits and input/output interface circuits are the most commonly used macro cells.

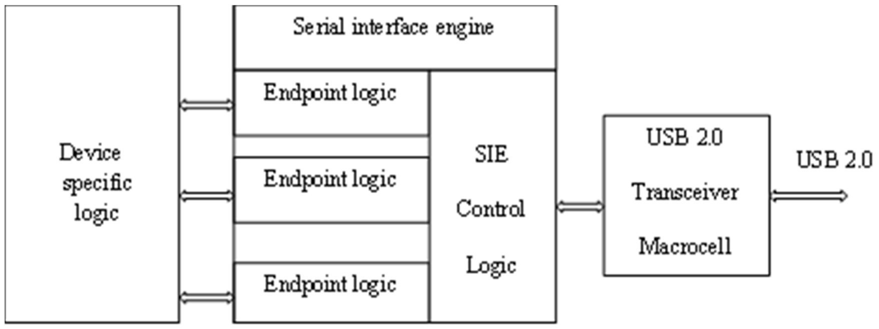


Figure1. ASIC functional block diagram of USB controller

Key features of the UTMI

1. It will eliminate the USB 2.0 logic design for peripheral designers.
2. For transmission and reception of USB 2.0 cable data, the UTMI utilizes 8 bit parallel interface.
3. Synchronous pattern, end of packet generation and checking is one of the key features of the UTMI.
4. This will integrate the USB components into a single functional unit.
5. It will support high speed as well as full speed operation to support dual mode gadgets.
6. This will facilitates information and clock recovery from the serial stream of data frame.
7. It has a capacity to switch between FS/HS operation termination and flagging.

Design aspects of UTMI

The UTMI has been subdivided into transmitter and recipient module. The plan contemplations of these considerations of these modules are clarified in detail.
The outline of UTMI transceiver is shown in fig. 2 the transmit hold register receives parallel data from SIE. This data is converted serially when it is passed through transmit shift register. Bit stuffing algorithm is performed on the serial data to provide information transition that is data move, for clock recovery and NRZI encoding. In synchronous communication, the sender and the receiver should be synchronized with each other before the information is sent. With specific end goal to keep up clock synchronization over longer duration, a unique kind of flag byte is implanted into the stream of data. This will helps in maintaining the timing between transmission and reception. There is one such strategy for inserting timing data is called NRZI encoding. At that point the encoded information is sent onto the serial transport. This data is decoded when it is gotten on the serial transport.
Bit unstuffing algorithm is used to remove embedded zero after each 6 continuous ones are detected in the information stream. Bit unstuffed information is sent over the receive shift register to convert the serial data stream to parallel data. The original data is recovered from the receive hold register. This information will be set on the parallel interface where sampling is done by the SIE.

Transmitter module

Transmitter of the UTMI performs several tasks. In order to maintain a proper synchronization between the sender and the receiver, there should be a transition at least every six bits in the data stream. This can be done by appending a zero bit after each six successive ones in the data stream. Bit stuffed binary information is NRZI encoded before it is sent to the USB host.

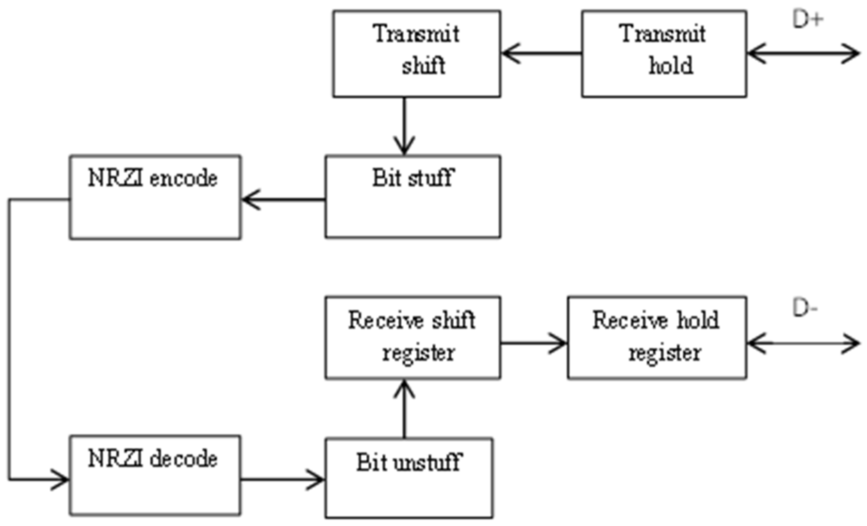


Figure2. UTMI transceiver outline

The transmitter module is outlined by considering the accompanying particulars.

a. When the transmitter is initiated by SIE, the flag byte “01111110” which is used as a synchronous pattern is transmitted.
b. After the six continuous ones are detected in the information stream a zero bit is embedded.
c. Non return to zero invert (NRZI) encoding is used because it requires half the base band bandwidth compared to other encoding techniques.
d. Two single ended zeroes in the synchronous pattern indicating end of packet transmission.

Receiver module

The complexity in designing the receiver is more compared to that of the transmitter and requires additional functionalities. When the NRZI encoded information is accessible on the serial transport it is decoded and bit unstuffed to recover the original data. The recovered information is sent to the receive shift register from where the serial information is changed to parallel. If there is any mismatch in the received data, retransmission of the data packets is requested from the host.

The receiver module is composed by considering the accompanying details.

a. Intimation to be sent to the SIE when the synchronous sequence is detected.

b. An error to be accounted for to the SIE, if zero is no recognized after six continuous ones.

c. End of packet pattern detection should also be intimated to the SIE.

UTMI Design flow

The design flow starts with the specifications those are to be considered for designing the transceiver. The specifications should define the functionality and complete architecture of the design. The behavioral descriptions of the design are created to understand the functionality and performance. Behavioral descriptions are written in hardware description language. Later this behavioral description is converted manually to a hardware description language (HDL). By using the electronic design automation (EDA) tools, the RTL description is converted to gate level net list. Verify whether the gate level net list meets the desired constraint time, chip area and power consumption. This net list acts like an input for the logic synthesis tool which creates the layout schematic of the circuit.

Verilog code is written separately for every blocks present in the transceiver module. Port mapping is used to connect the different Verilog modules of the UTMI circuit. There are two types of port mapping are present in HDL. Port mapping by name and port mapping by order, we can use any one of these for designing UTMI. A test bench code is written which include input sequence to be fed to the transceiver. Then the Verilog file is synthesized, if any error occurs while running the program, it is debugged to obtain the correct results or waveforms. Realizing the UTMI design on FPGA board consists of five steps, which are implemented using integrated synthesis environment (ISE) software tools produced by Xilinx for synthesis and analyzing the HDL designs. The area occupied by the UTMI design on the silicon chip is about 5.0mm × 5.0mm.

Step 1: Converting the Verilog description of the UTMI design in to a gate level net list by the process called synthesis.

Step 2: Mapping is done on the gate level net list to the technology specific components on the FPGA.

Step 3: Placing the mapped components in such a way that the delay and wiring should be minimized.

Step 4: To wire the components in the design routing is done to configure the programmable interconnects (wires).

Step 5: Xilinx software tool will convert the placed and routed UTMI design to a bit stream. The bit stream (as a .bit file) is loaded on to the FPGA board which is generated by the software tool.

The receiver output can be observed in the waveform window, when the Verilog code is successfully dumped to Spartan 6 family of FPGA board. The input to the design is provided by test bench Verilog code. This has a clock frequency of 4MHz, and the output (LED) light emitting diode blinks very fast. In order to observe the output waveforms clearly, we use the clock divider program in the design. Chip scope pro analyzer is a debugging tool used to compare the waveforms of the simulation with the FPGA board. Chip scope provides an interface between the personal computer (PC) and the FPGA board. The waveforms at each stage of UTMI design is observed by using this debugging tool. Once the waveforms of simulation are matched with the FPGA board then we conclude that the transmission and reception of data is carried successfully. For reliable transmission of data, the bit error rate should be kept minimum. File transfer time for USB 2.0 to transmit 10 GB data is 3 minutes and it will depend upon the cable length. Data transfer time will vary for different USB versions. It has a total bandwidth limitation of 60MBps. Bandwidth is shared among all attached high speed USB devices.

Simulation results

The register transfer level (RTL) schematic of USB transceiver and simulated waveforms as follows.

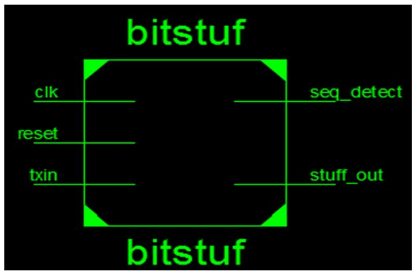


Figure3. RTL schematic symbol of sequence detection and bit stuffing

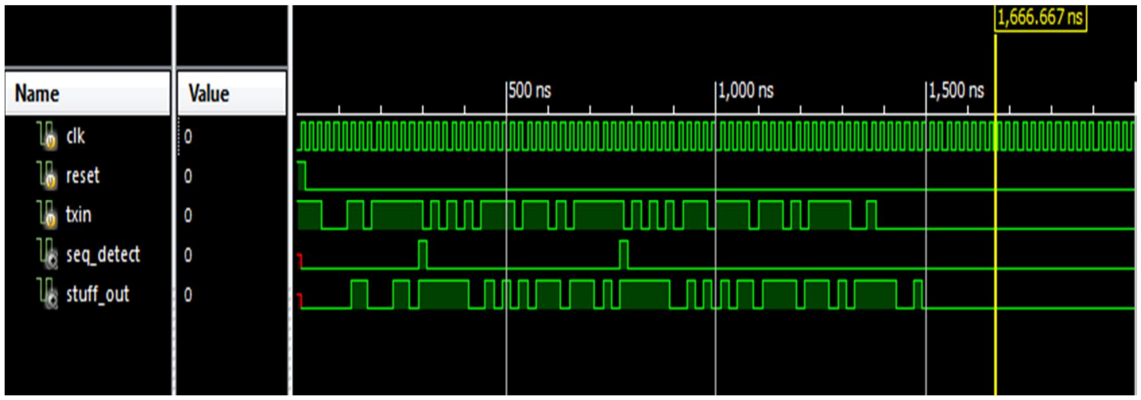


Figure4. Output waveforms of sequence detection and bit stuffing

Sequence detection and Bit stuffing is performed at the transmitting section of UTMI. Bit stuffing and destuffing is done at the data link layer. From above schematic it is clear that clock, reset, txin (transmitter input) is the input ports. Sequence detected (seq_detect) and stuffed output (stuff_out) are the output ports. The output waveforms of the bit stuff schematic are given above.
The stuffed output is fed as input to the NRZI encoder that is enc_di (encoder input data). NRZI encoded signal from port enc_do (encoder output data) has a transition if the bit being transmitted is logic 0 and there is no transition if the bit being transmitted is logic 1. Transition occurs at the leading edge of the clock signal. The encoded data is decoded, which is available at the port dec_do (decoded output data). The NRZI decoded data is same as that of the stuffed output. The simulated waveforms of the NRZI encoder and decoder are given in fig. 6.

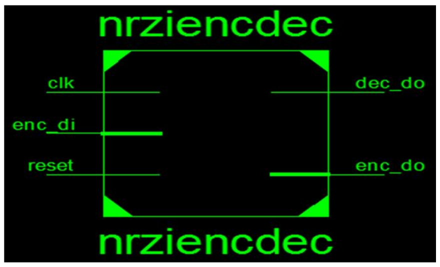


Figure5. RTL schematic symbol of NRZI encode and decode

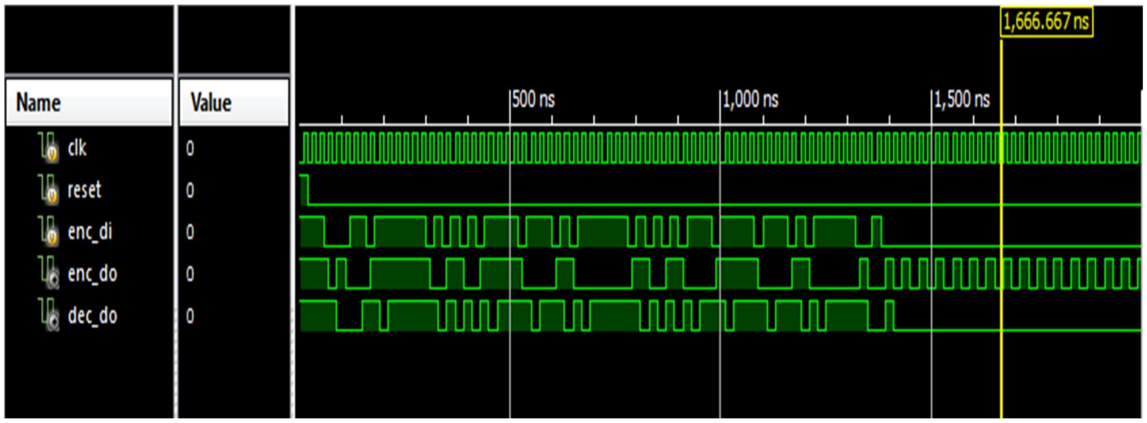


Figure6. Output waveforms of NRZI encode and decode

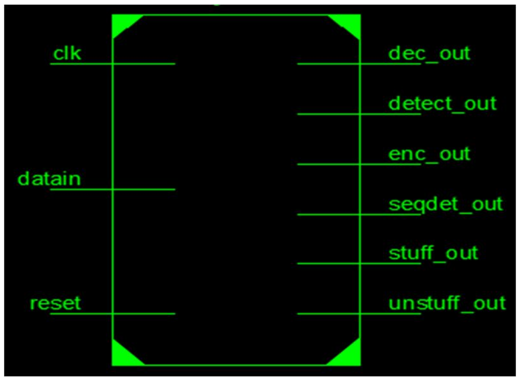


Figure7. RTL schematic symbol of UTMI

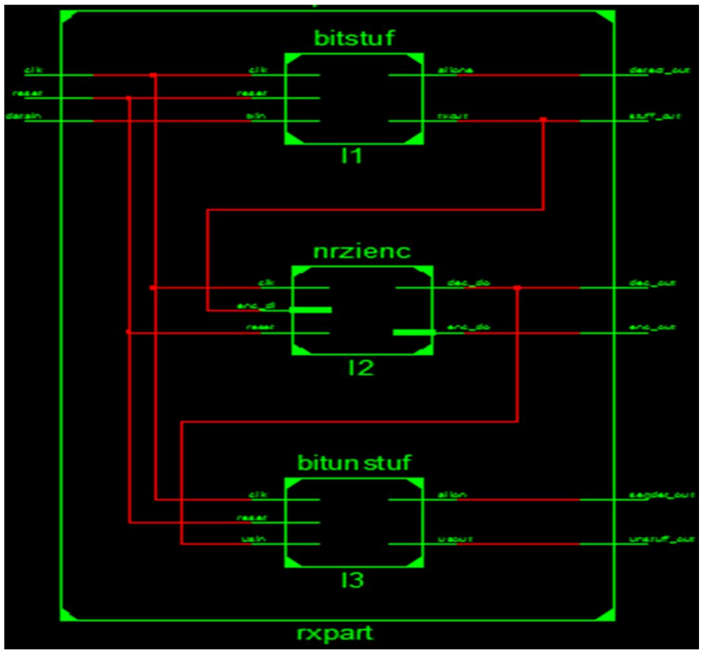
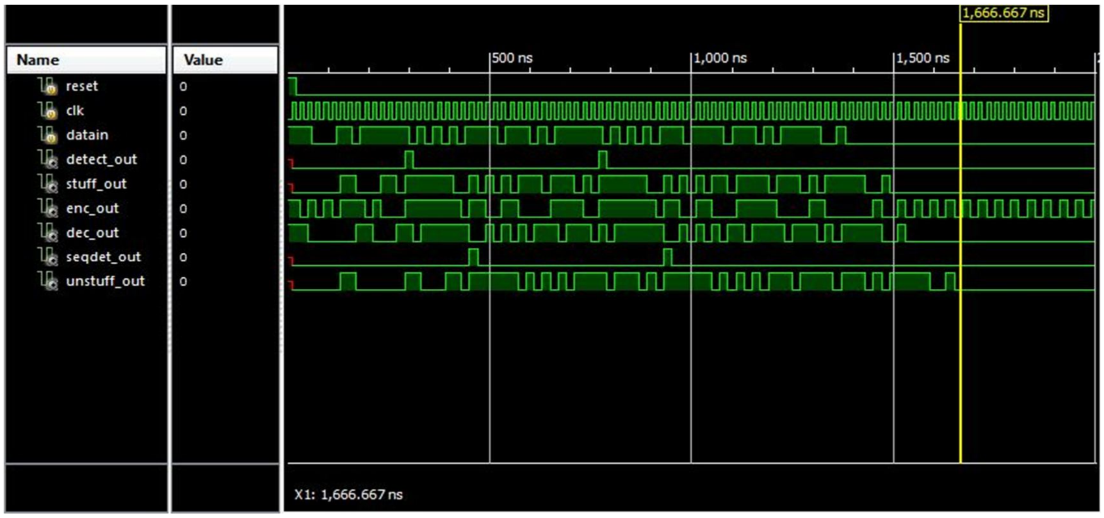


Figure8. RTL schematic of UTMI



Name	Value
reset	0
clk	0
datain	0
detect_out	0
stuff_out	0
enc_out	0
dec_out	0
seqdet_out	0
unstuff_out	0

Figure9. Output waveform of the UTMI

The RTL schematic of complete USB transceiver is given in fig. 7 below. Bit destuffing is done after detecting six consecutive ones present in the bitstream. The unstuffed output present at port unstuff_out is the original signal that is transitted at the transmitter.
The I1, I2, I3 are the different instances present in the top module of the transceiver. These are interconnected using port mapping. The top module Verilog file is synthesized to obtain the results.
The fig. 9 shows the waveforms of the transceiver, the received output is the shifted version of the original signal transmitted.
The fig. 10 shows the experimental set up of implementing UTMI. The fig. 11 shows the chip scope analysis of the simulation and the FPGA board. This chip scope pro analyzer is used to debug the waveforms of the simulation.

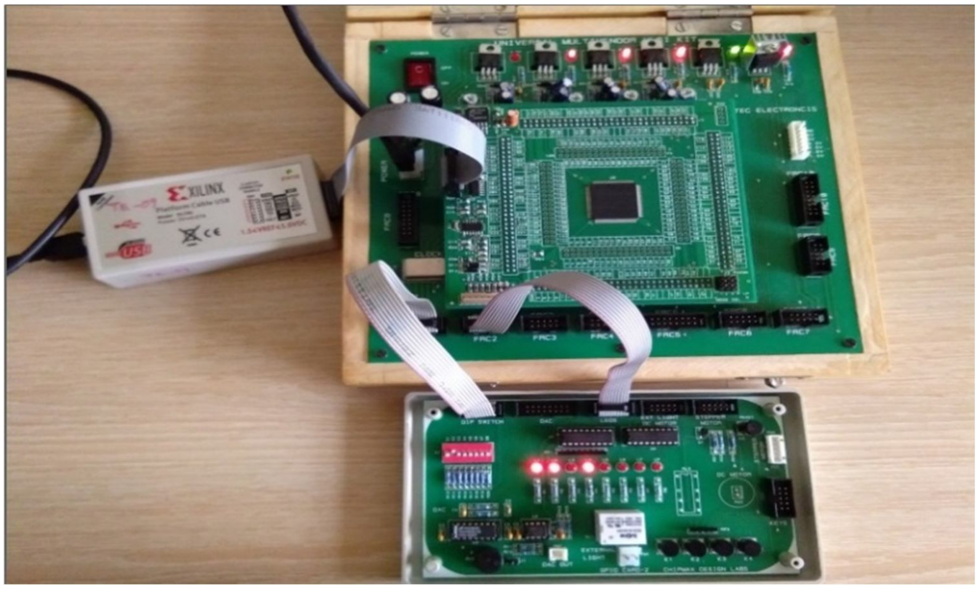
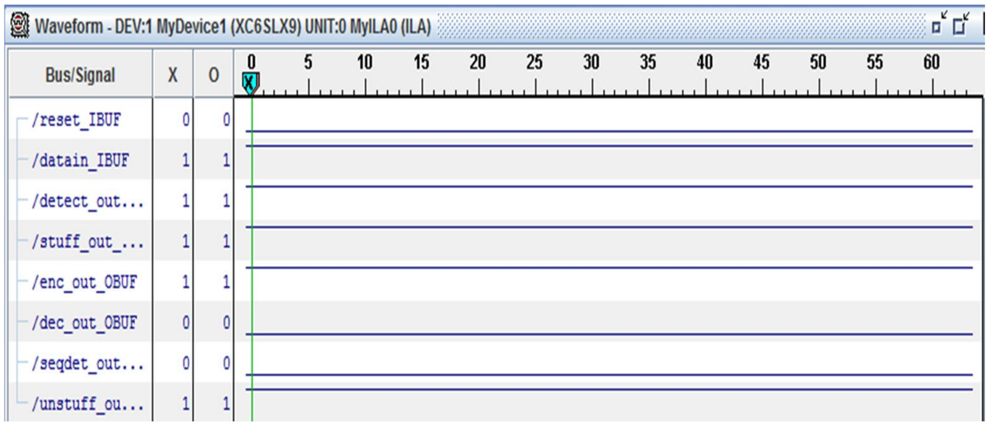


Figure10. Experimental set up of interfacing UTMI with the FPGA board



Bus/Signal	X	O
/reset_IBUF	0	0
/datain_IBUF	1	1
/detect_out...	1	1
/stuff_out_...	1	1
/enc_out_OBUF	1	1
/dec_out_OBUF	0	0
/seqdet_out...	0	0
/unstuff_ou...	1	1

Figure11. Chip scope analysis waveform for UTMI

Future scope

In this paper 8- bit USB transceiver is designed, it can also be extended to 16-bit. Cyclic redundancy check algorithms are used to correct errors present in the transmission. If SIE and device specific logic are designed using VHDL, UTMI, DSL and SIE together acts like a controller for any type of USB devices.

Conclusion

The Verilog code for each module in UTMI is designed and verified using Xilinx 14.7 software. The UTMI designed to support HS/FS operation providing the data transfer rate of 12MBps and 480MBps respectively. To satisfy higher data transfer rates full duplex signaling scheme is used. USB 3.0 utilizes this type of signaling method. Synchronization between transmitter and the receiver is maintained by using bit stuffing algorithms. This UTMI design is interfaced with the Spartan-6 FPGA board, and the waveforms are verified by using chip scope debugging tool.

Acknowledgment

We would like to offer our thanks to my project guide Mr. Naveen Kumar M assistant professor department of TCE, for his direction and steady support for the duration of time traverse of this project work. We want to thank the lab staffs for giving us Lab equipment's, kits and software tools which were basic for building up our project.

References

[1] K. Babulu and K. S. Rajan, "FPGA Implementation of USB Transceiver Macrocell Interface with USB2.0 Specifications," *2008 First International Conference on Emerging Trends in Engineering and Technology*, Nagpur, Maharashtra, 2008, pp. 966-970.

[2] F. A. Jolfaei, N. Mohammadizadeh, M. S. Sadri and F. FaniSani, "High Speed USB 2.0 Interface for FPGA Based Embedded Systems," *2009 Fourth International Conference on Embedded and Multimedia Computing*, Jeju, 2009, pp. 1-6.

[3] Guangling Guo, Zhiqiang Li and Fan Yang, "Design of high speed pulse data acquisition system based on FPGA and USB," *2011 International Conference on Multimedia Technology*, Hangzhou, 2011, pp. 5374-5376.

[4] C. C. Lun, A. bin Marzuki and S. H. Wei, "Analog front-end design implementation of USB2.0 OTG Attach Detection Protocol," *2012 4th International Conference on Intelligent and Advanced Systems (ICIAS2012)*, Kuala Lumpur, 2012, pp. 774-779.

[5] P.M.Szecówka and K. J. Pyrzynski, "USB receiver/transmitter for FPGA implementation," *2012 International Conference on Signals and Electronic Systems (ICSES)*, Wroclaw, 2012, pp. 1-6.

[6] M. K. Pandey, S. Shekhar, J. Singh, G. K. Agarwal and N. Saxena, "A novel approach for USB2.0 validation on System on Chip," *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, 2013, pp. 1-4.

[7] B. Hong, C. Shin and D. Ko, "Emulation based high-accuracy throughput estimation for high-speed connectivities: Case study of USB2.0," *2011 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, New York, NY, 2011, pp. 609-614.

[8] C. Lyu; J. Peng; W. Zhou; S. Yang; Y. Liu, "Design of a High Speed 360-degree Panoramic Video Acquisition System Based on FPGA and USB 3.0," in IEEE Sensors Journal , vol.PP, no.99, pp.1-1.

[9] P. Pranav, P. J. Brahmbhatt and U. S. H. Rao, "USB based high speed data acquisition system," *2015 5th Nirma University International Conference on Engineering (NUiCONE)*, Ahmedabad, 2015, pp. 1-5.

[10] X. Cao, J. Zhang, W. Yao and M. Ju, "High-Speed and Portable Data Acquisition System Based on FPGA," *2012 Fifth International Conference on Intelligent Networks and Intelligent Systems*, Tianjin, 2012, pp. 213-216.