Implementation of USB 3.0 SuperSpeed Physical Layer using Verilog HDL

Hardik Trivedi
School of Information Sciences
Manipal University
Manipal, India

Rohit kumar
School of Information Sciences
Manipal University
Manipal, India

Ronak Tank
School of Information Sciences
Manipal University
Manipal, India

Sundaresan C.
School of Information Sciences
Manipal University
Manipal India

Madhushankara M.
School of Information Sciences
Manipal University
Manipal India

ABSTRACT
In this proposed design it mainly includes USB 3.0, Physical Layer along with USB 2.0 functionality with Super speed functionality. Physical Layer mainly contains PCI Express and PIPE interface. This proposed design transferred data from transmitter to receiver serially. This design manages to transfer data either on 2.5GT/s or on 5.0GT/s depends upon the mode and rate. The design generates clock that runs on two different frequencies i.e. 125MHz and 250MHz that used to transfer data on parallel interface. This Design manages to capture the data that are coming asynchronously and lock the receiver clock with incoming asynchronous serial data. The architecture for USB 3.0 Physical Layer has been proposed in this paper. The proposed model is implemented and verified using Verilog HDL.

1. INTRODUCTION
The physical layer defines the PHY portion of a port and physical connection between a downstream facing port (Host) and upstream facing port of the device [1], [4]. Super speed physical connection is the comprised of two differential data pairs transmit and receive path. The nominal data rate is 5GB per second. As per Electrical aspects of each path are characterized as a transmitter, channel and receiver. These are unidirectional differential link. Each differential link is AC coupled with capacitors located on the transmitter side. The channels include electrical characteristics of the cables and connectors. The each differential link is initialized with receiver terminations. The Transmitter is responsible for detecting Receiver at other end of the link. When receiver termination is present but no signaling is occurring then it is consider to be an electrical idle state and in this state, low frequency periodic signal (LFPS) is generated which is easy to generate and consume very less power [5]. Each PHY has its own clock domain. The USB 3.0 cable does not include a reference clock so bit level timing synchronization relies on the local receiver to align bit recovery clock to the remote transmitters clock by phase-locking to the signal transitions in the received bit stream [7]. To ensure the proper transitions occur in the bit stream independent of the data content being transmitted, the transmitter encode the data and special characters using 8b/10b code. Control symbols are used to achieve byte alignment and are used for framing data and managing the link.

2. LITERATURE SURVEY
It is a specification to establish communication between devices and a host controller (usually a personal computer), developed and invented by Ajay Bhatt, while working for Intel. Universal Serial Bus (USB) has effectively replaced a variety of interfaces such as serial and parallel ports. USB can connect computer peripherals such as mice, keyboards, digital cameras, printers, personal media players, flash drives, Network Adapters, and external hard drives. For many of those devices, USB has become the standard connection method [1]. Initially, USB 1.0: Released in January 1996 which specified data rates of 1.5Mb/s (Low-Bandwidth) and 12Mb/s (Full Bandwidth) it does not allow for extension cables or pass through monitors due to timing and power limitations. USB 1.1: Released in September 1998. This version of USB overcame the problems identified in 1.0, mostly relating to hubs. Earliest revision to be widely adopted.

In USB 2.0: Released in April 2000. Added with maximum bandwidth of 480 Mb/s (now called"Hi-Speed"). Micro-USB Cables and Connectors.

In USB 3.0 implementation, focusing on USB 2.0backward compatibility and on the major features associated with the Super-Speed (SS) functionality which is 10 times faster than USB 2.0 version.

The goal is to transfer data either on 2.5GT/s or on 5.0GT/s depends upon the mode and rate.

3. USB 3.0 PHYSICAL LAYER
Superspeed PHY Layer handles the low level USB Superspeed protocol and signaling.

It includes following features:

- Standard PHY interface enables multiple IP Sources. Supports 5.0Gb/s serial data transmission rate.
- Utilize 8bits, 16bits or 32bits parallel interface to transmit and receive.
- Allow integration of high speed components into single functional block as seen by the end point device manager.
- Data and clock recovery from serial stream.
- Holding registers to transmit and receive data.

- Support direct disparity control for transmitting compliance patterns.
- 8b/10b encode/decode and error indication.

The physical layer defines the signaling technology for the Super Speed bus. This defines the electrical requirements of the Super Speed physical layer. This defines the electrical-layer parameters required for interoperability between Super Speed components [9]. The PHY Interface for the PCI Express and USB Super Speed Architectures (PIPE) is intended to enable the development of functionally equivalent PCI Express and USB Super Speed PHY's.

4. USB 3.0 PHYSICAL LAYER DESIGN

The pin diagram for the proposed USB3.0 Physical Layer architecture is shown in figure 1. The description for the PHY pins is given below.

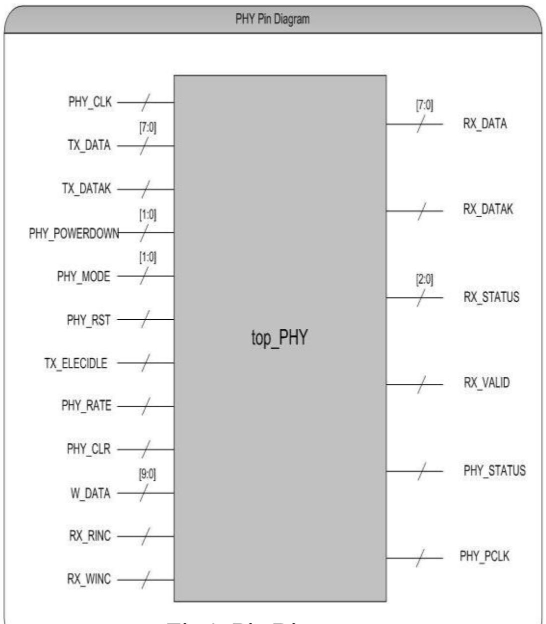

Fig 1: Pin Diagram

1) TX_DATA: It's a pin which shows Parallel PCI Express or Super speed data input bus.

2) TX_DATAK: This pin used for Data control for symbol or transmit data.

3) PHY_POWERDOWN: This pin shows USB Super speed in which power saving state.

4) PHY_MODE: This pin indicates that in which mode it is working (PCI Express mode or USB Super speed mode).

5) PHY_RST: This pin resets the transmitter and receiver

6) PHY_ELECIDLE: When PHY is in P0 or P1 power state this TX ELECIDLE signal is asserted.

7) PHY_RATE: This pin controls the link signaling rate. 0 use for 2.5 GT/s signaling rate. 1 use for 5.0 GT/s signaling rate.

8) PHY_CLR: This pin is used to reset the write part of the asynchronous fifo1 module.

9) W_DATA: This pin is used to write data in the fifo1 module.

10) RX_RINC: This pin shows a read enable of the asynchronous fifo.

11) RX_WINC: This pin shows a write enable of the asynchronous fifo.

12) RX_DATA: This pin shows the parallel PCI output bus that is received data.

13) RX_DATAK: This pin is used for data control bit for received symbol.

14) RX_STATUS: This pin used to indicate that received data is proper or it has some errors.

15) PHY_STATUS: This pin is used to communicate completion of several PHY functions.

16) PHY_VALID: This pin indicates symbol lock and valid data on RX DATA and RX DATAK.

17) PHY_CLK: This clock pin is used to generate BIT CLK and PCLK

5. USB 3.0 PHYSICAL LAYER MICROARCHITECTURE

5.1 PHY Transmitter Architecture

The architecture for PHY Transmitter is shown in figure 2.

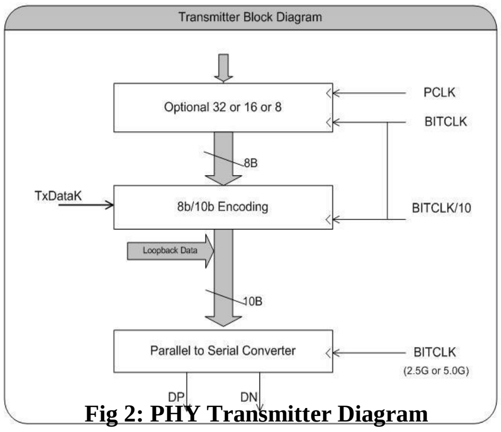

Fig 2: PHY Transmitter Diagram

The proposed PHY Transmitter has 3 components. They are:

5.1.1 Scrambler (Optional)

The Scrambler uses an algorithm to pseudo-randomly scramble each byte of data. Scrambling eliminates repetitive patterns in the bit stream. Repetitive patterns results in large amounts of energy concentrated in discrete frequencies which lead to significant EMI noise generation. Scrambling spreads energy over a frequency range, hence minimizing average EMI noise generated.

5.1.2 8b/10b Encoder

The scrambled 8b characters are encoded into 10b symbols by the 8b/10b Encoder logic. And there is a 25% loss in transmission performance due to the expansion of each byte into a 10-bit character [3], [6]. A character is defined as the 8b un-encoded byte of a packet. A Symbol is defined as the 10b encoded equivalent of the 8-bit character. The purpose of 8b/10b Encoding the packet characters is primarily to create sufficient 1-to-0 and 0-to-1 transition density in the bit stream so that the receiver can re-create a receive clock with the aid of a receiver Phase Lock Loop (PLL) [11].

5.1.3 Parallel-to-Serial converter

The 10b symbols are converted to a serial bit stream by the Parallel-to-Serial converter. This logic uses a 2.5 GHz clock to serially clock the packets out on each Lane.

5.2 PHY Receiver Architecture

The architecture for PHY Receiver is shown in figure 3. As the data bit stream is received, the receiver PLL is synchronized to the clock frequency with which the data was clocked out of the remote transmitter device. The transitions in the incoming serial bit stream are used to re-synchronize

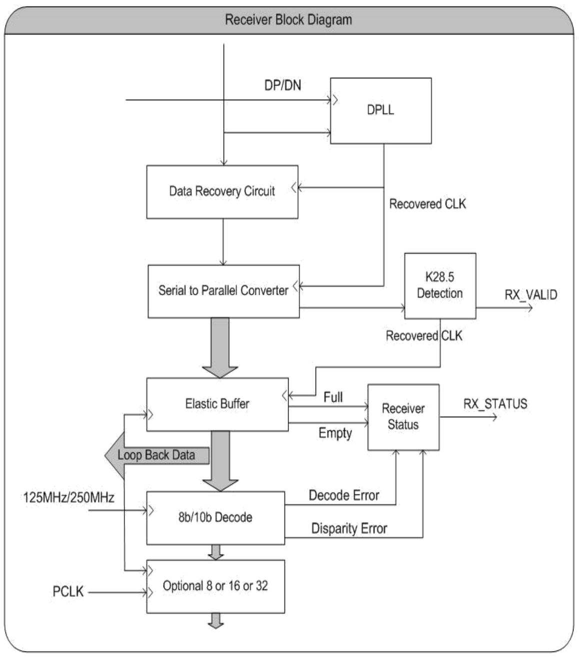

Fig 3: PHY Receiver Diagram

the PLL circuitry and maintain bit and symbol lock while generating a clock recovered from the data bit stream [7].

The serial-to-parallel converter is clocked by the recovered clock and outputs 10b symbols.

The 10b symbols are clocked into the Elastic Buffer using the recovered clock associated with the receiver PLL. The Elastic Buffer is used for clock tolerance compensation; i.e. the Elastic Buffer is used to adjust for minor clock frequency variation between the recovered clock used to clock the incoming bit stream into the Elastic Buffer and the locally-generated clock associated that is used to clock data out of the Elastic Buffer [8].

The 10b symbols are converted back to 8b characters by the 8b/10b Decoder. The 8b/10b Decoder also looks for errors in the incoming 10b symbols. For example, error detection logic can check for invalid 10b symbols or detect a missing Start or End character.

The De-Scrambler (Optional) reproduces the de-scrambled packet stream from the incoming scrambled data stream. The De-Scrambler implements the inverse of the algorithm implemented in the transmitter Scrambler.

6. SIMULATION RESULTS AND ANALYSIS

This design generates clock with the different clock frequency i.e. 125MHz, 250MHz and 2.5GHz. For data rate purpose design generates PCLK depends upon Power state, Rate and PHY mode.

The Encoder used to convert the 8b input data into 10b of dc balanced code. This design will generate +ve and -ve disparity code on the consecutive clock cycle. The proper encoded output is available after one clock cycle.

As we seen in simulation design, clock divider which is used to divide the BIT RATE clock by 10. This clock is used to hold the 10bit encoded data until each bit is serially transmitted using parallel to Serial conversion. In the transmitter side the parallel to serial converter used to convert the 10bit encoded parallel data into 1bit serial data.

Again this captured 10bit data. As the receiver accepting single bit data continuously and serial to parallel converter will continuously generate the 10bit data there are chances that Receiver might receive different data other than transmitted data but in order to save memory and to capture the proper data especially K28.5. Output will get input on every 10 clock cycle of BIT CLK and receiver get proper data.

This design generates proper Rx STATUS based on the incoming 10bit data or elastic buffer status.

For disparity error, if it is DC balanced data it will result 1 else it will result 0.

Here 8b/10b decoder properly decodes the encoded data and received proper 8 bit data which is transmitted initially.

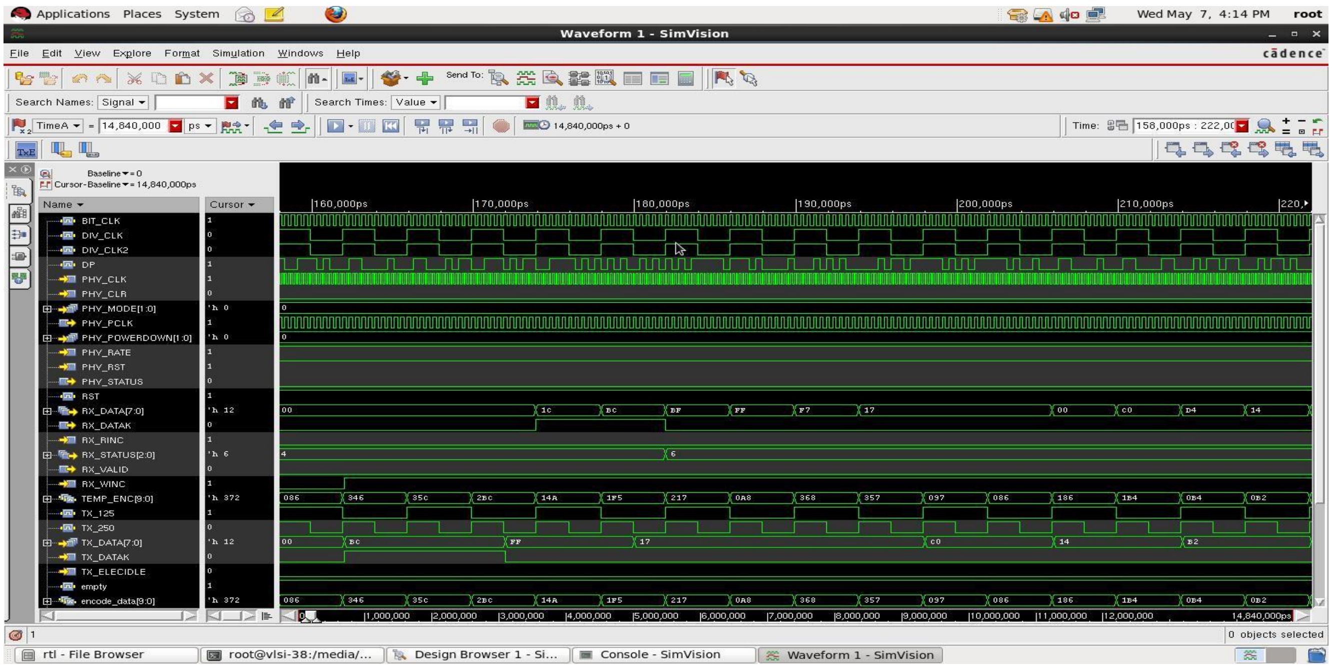

Fig 4: PHY Simulation Results1

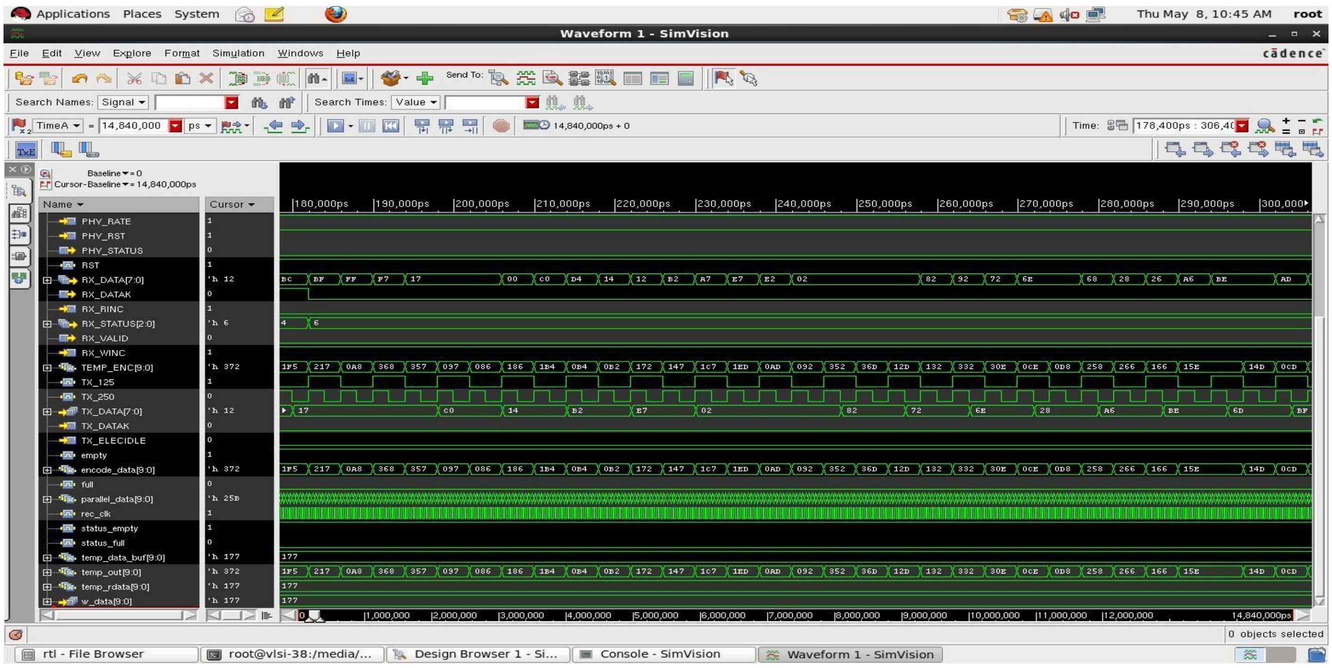

Fig 5: PHY Simulation Results2

7. CONCLUSION

This design explains the concept of USB 3.0 and Serial data communication. This design is able to transfer data on 2.5GHz clock as well as 5.0GHz. This design makes to understand the Super speed features of USB 3.0. Due to data bursting capability USB 3.0 provide far more speed than USB 2.0 as device does not have to wait for the hubs acknowledgment. It also helps to understand 8b/10b encoder and decoder and how they provide a dc balanced data. And this design is designed and verified using Verilog HDL in Cadence NCSim simulation tool.

8. ACKNOWLEDGMENTS

We would like to thank our professors Mr. Madhushankara M., Mr. Sunderasan C. and Asst. Prof Chaitanya CVS for being our advisor and guide. We are grateful to them for their continuous support and invaluable inputs they have been providing us through the development of the project. This work would not have been possible without their support and encouragement. We would also like to thank our friends Ashwin K Rao and Chandru Nagaraj for showing us some examples that related to the topic of our project.

Besides, we would like to thank our department School Of Information Science for providing us with a good environment and facilities to complete this project. It gave us an opportunity to participate and learn about Hardware designing. In addition, we would like to thank our professors that they provided us huge and valuable information as the guidance of our project.

9. REFERENCES

[1] "Universal Serial Bus 3.0 Specification", Revision 1.0, November 12, 2008.

[2] "Data Manual Texas Instruments" , Literature number: SLLSE16E, June 3, 2011

[3] A.X. Widmer and P.A. Franaszek, "A dc-balanced, partitioned block, 8B/10B transmission code, "IBM journal of Research and Development, vol.27, no.5, pp.440-451, Sep 1983.

[4] "Universal Serial Bus 2.0 Specification", Revision 1.0, March 13, 2006.

[5] "PHY interface for the PCI Express and USB 3.0 Architecture", March 11, 2009.

[6] "Lattice Semiconductor Corporation 8b/10b Encoder/Decoder", February 2012.

[7] Ching-Che Chung, Chen-Yi Lee, "An All-Digital Phase Locked Loop for high speed clock generation", Febr 6, 2003.

[8] Clifford E. Cummings and Peter Alfke, "Simulation and Synthesis Techniques for Asynchronous FIFO Design with Asynchronous Pointer Comparisons", SNUG 2002.

[9] Ravi Budruk, Don Anderson & Tom Sanely, 2004. "PCI Express System Architecture", Mindshare Inc., pp 419-434.

[10] Thatcher, Jonathan (1996-04-01). "Thoughts on Gigabit Ethernet Physical", IBM Retrieved on 2008-08-17.

[11] Jenming Wu & Yu-Ho Hsu, "8B/10B Codec for Efficient PAPR Re-duction in OFDM Communication Systems", International technology roadmap for Semiconductors (ITRS).