Design and Implementation of High Speed Data Transmission over Dual Independent Aurora Channels on One GTX dual tile usingVirtex-5 FPGA

S.Venkata Kishore
Embedded Systems (M.Tech),
ECE Department,
Gudlavalleru Engineering College,
Gudlavalleru, Krishna District,
Andhra Pradesh -521356, India.
kishorebapatla@gmail.com

M.Srilatha
Assistant professor,
ECE Department,
University College of Engineering (A)
Osmania University, Hyderabad,
Andhra Pradesh -500 007, India.
srilatha.mamidigalla@gmail.com

Mr.Y.S.Chakrapani
Associate professor,
ECE Department,
Gudlavalleru Engineering College,
Gudlavalleru, Krishna District,
Andhra Pradesh -521356, India.
srichakrapani@gmail.com

Dr. M. Kamaraju,
Professor,
Head of the ECE Department,
Gudlavalleru Engineering College,
Gudlavalleru, Krishna District,
Andhra Pradesh -521356, India.
madduraju@yahoo.com

Mr. S.R.Pankaj Kumar Scientist 'F',
Defence Electronics Research Laboratory (DLRL),
Ministry of Defence, Government of India
Chandrayangutta lines,
Hyderabad- 500 005, India.
pankaj_salla@rediffmail.com

Abstract - This paper proposes a design and implementation of high speed data transmission over dual independent aurora channels on One GTX (Gigabit Transceivers) DUAL TILE by configuring multi-gigabit transceivers (MGT's), which are present in the virtex-5 FPGA using aurora protocol. (Here GT means Gigabit transceivers (GT) and X indicates that these transceivers belong to Virtex-5 FXT platform. FXT platform supports High-performance embedded systems with advanced serial connectivity). Firstly, a 128 bit parallel data is to be generated using simulators, which are implemented using VHDL language. The asynchronous first-in first-out (AFIFO) takes this 128 bit data as input and produces an output of 16 bit parallel data. This data goes to the aurora module in parallel form as successive frames (i.e. 8 frames, each frame consists of 2 bytes). Finally, the 128 bit parallel data is transmitted to the receiver module serially over fiber optic cable at the rate of 3.125Gbps using architectural features of virtex5 FPGA. To achieve high speed, Multi-Gigabit Transceivers (MGT) are used. In virtex-5 FPGA, these Multi-Gigabit Transceivers are available as hard IPs which operates at the clock rate of 156.25 MHz (MGT clock). For configuring these MGT's, Aurora protocol is used, which converts the parallel data into serial data and vice versa. Finally the data is transmitted on two independent aurora channels and further testing is carried out using chip scope pro analyzer to verify the integrity of data.

Keywords - Aurora protocol, Virtex-5 FPGA, MGT's, AFIFO, Dual Independent Aurora Channels, GTX DUAL TILE, Serial Data Transmission.

1. INTRODUCTION

In earlier days to achieve the high speed we were using parallel transmission. In parallel transmission, Binary data consisting of 1s and 0s may be organized into groups of n bits each. By grouping, we can send data n bits at a time instead of one. We use n wires to send n bits at one time. That way each bit has its own wire, and all n bits of one group can be transmitted with each clock pulse from one device to another. For $n = 8$. Typically, the eight wires are bundled in a cable with a connector at each end. The advantage of parallel transmission is speed. All else being equal, parallel transmission can increase the transfer speed by a factor of n over serial transmission. A significant disadvantage of parallel transmission is cost. Parallel transmission requires n communication lines (wires in the example) just to transmit the data stream. Because this is expensive, parallel transmission is usually limited to short distances.

In serial transmission one bit follows another, so we need only one communication channel rather than n to transmit data between two communicating devices. The advantage of serial over parallel transmission is that with only one communication channel, serial transmission reduces the cost of transmission over parallel by roughly a factor of n Since communication within devices is parallel, conversion devices are required at the interface between the sender and the line (parallel-to-serial) and between the line and the receiver (serial-to-parallel).Serial transmission occurs in one of two ways; asynchronous or synchronous.

MGTs are used increasingly for data communications because they can run over longer distances. The main function of these MGT's are to convert the parallel data into serial data and vice versa and this action is performed by configuring the MGT's using aurora core. These MGT's are present in the FPGA as hard IPs and we have to interface to our application by configuring hard IPs using soft IPs such as aurora core in order to achieve the high speed serial data transmission. The MGT's are configured as either transmitter or receiver to

perform the data transmission.

The Rocket IO GTX transceiver is a power-efficient transceiver for Virtex-5 FPGAs. The GTX transceiver is highly configurable and tightly integrated with the programmable logic resources of the FPGA. GTX transceivers are placed as dual transceiver GTX_DUAL tiles in Virtex-5 FXT devices. This configuration allows two transceivers to share a single PLL with the TX and RX functions of both, reducing size and power consumption. Multi Gigabit Transceiver (MGT) is a Serialiser /Deserialiser (SerDes) capable of operating at serial bit rates above 1 Gigabit/second.

2. AURORA PROTOCOL

A: Introduction:

The Logic CORE™ IP Aurora 8B/10B core implements The Aurora 8B/10B protocol using the high-speed serial Transceivers on the Virtex-5 LXT, SXT, FXT, and TXT Family. The Aurora 8B/10B core is a scalable, lightweight, link layer protocol for high-speed serial communication. The protocol is open and can be implemented using Xilinx® FPGA technology. The protocol is typically used in applications requiring simple, low-cost, high rate, data channels. In our application virtex5 FXT is used because it is the CORE Generator software produces source code for Aurora 8B/10B cores with variable data path width. The cores can be simplex or full-duplex.

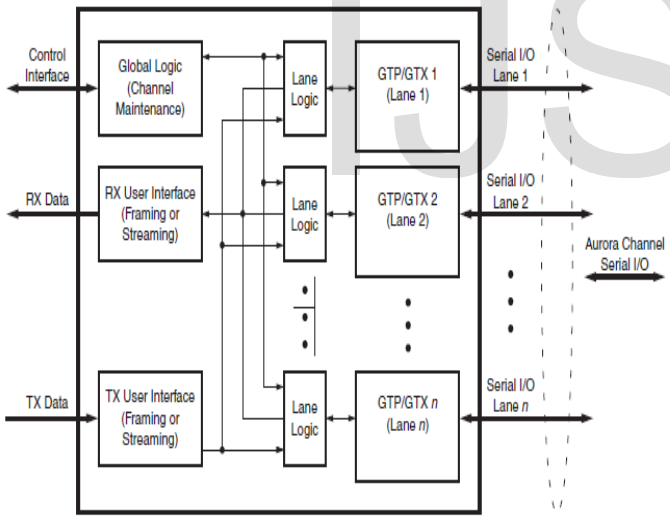

Fig.2: Aurora 8B/10B Core Block Diagram

B: Functional Blocks:

Fig2 shows a block diagram of the Aurora 8B/10B core. The major functional modules of the Aurora 8B/10B core are:

- **Lane logic**: Each GTP/GTX transceiver is driven by an instance of the lane logic module, which initializes each individual GTP/GTX transceiver and handles the encoding and decoding of control characters and error detection.
- **Global logic**: The global logic module in each Aurora 8B/10B core performs the bonding and verification phases of channel initialization. While the channel is operating, the module generates the random idle characters required by the Aurora protocol and monitors all the lane logic modules for errors.
- **RX user interface**: The RX user interface moves data from the channel to the application. Frames are presented using a standard Local Link interface.
- **TX user interface:** The TX user interface moves data from the application to the channel. A standard Local Link interface is used for data frames. The module has an interface for controlling clock Compensation (the periodic transmission of special characters to prevent errors due to small clock frequency Differences between connected Aurora 8B/10B cores).

The Aurora 8B/10B protocol uses a symbol-based method. The minimum unit of information that is transferred across an Aurora 8B/10B channel is two symbols, called a symbol-pair. The information on an Aurora 8B/10B channel (or lane) always comprises multiple symbol-pairs. Implementations of the Aurora 8B/10B protocol accept a stream of octets from user applications and transfer them across the Aurora 8B/10B channel as one or more streams of symbol-pairs. Transmission of user PDUs requires the following procedures:

- Padding
- Encapsulation with channel PDU delimiters
- 8B/10B encoding of channel PDU payload
- Serialization and clock encoding.

Reception of user PDUs involves the following procedures:

- Deserialization
- 8B/10B decoding of channel PDU payload
- Link layer stripping
- Pad stripping.

The Aurora 8B/10B core is a lightweight, serial communications protocol for multi-gigabit links. It is used to transfer data between devices using one or many GTP/GTX transceivers. Connections can be full-duplex (data in both directions) or simplex. Aurora 8B/10B cores automatically initialize a channel when they are connected to an Aurora channel partner. After initialization, applications can pass data freely across the channel as frames or streams of data. Whenever data is not being transmitted, idles are transmitted to keep the link alive.

Aurora frames can be any size, and can be interrupted at any time. Gaps between valid data bytes are automatically filled with idle to maintain lock and prevent excessive electromagnetic interference. The Aurora 8B/10B core detects single-bit, and most multi bit errors using 8B/10B coding rules. Excessive bit errors, disconnections, or equipment failures cause the core to reset and attempt to re-initialize a new channel.

C: Applications:

Aurora 8B/10B cores can be used in a wide variety of applications because of their low resource cost, scalable throughput, and flexible data interface. Examples of Aurora 8B/10B core applications include:

- **Chip-to-chip links**: Replacing parallel connections between chips with high-speed serial connections can significantly reduce the number of traces and layers required

on a PCB. The core provides the logic needed to use GTP/GTX transceivers, with minimal FPGA resource cost.

• **Board-to-board and backplane links**: The Aurora 8B/10B core uses standard 8B/10B encoding, making it compatible with many existing hardware standards for cables and backplanes. Aurora 8B/10B cores can be scaled, both in line rate and channel width, to allow inexpensive legacy hardware to be used in new, high performance systems.

• **Simplex connections (unidirectional)**: The Aurora protocol provides several ways to perform unidirectional channel initialization, making it possible to use the GTP/GTX transceivers when a back channel is not available, and to reduce costs due to unused full duplex resources.

• **ASIC applications**: The Aurora protocol is not limited to FPGAs, and can be used to create scalable, high performance links between programmable logic and high-performance ASICs. The simplicity of the Aurora protocol leads to low resource costs in ASICs as well as in FPGAs, and design resources like the Aurora bus functional model (ABFM 8B/10B) with compliance testing make it easy to get an Aurora channel up and running.

3. DESIGN OF APPLICATION

The following figure explains the design and its procedure to implement the high speed data transmission serially over dual independent aurora channels.

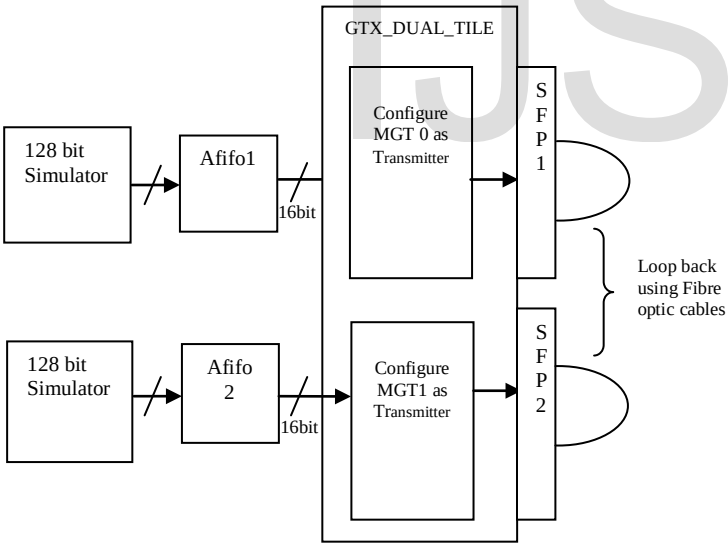

Fig. 3. Block Diagram of Simulator_Fifo_MGT's

The resources used for design and implementation of the high speed data transmission are as follows:

A. Software resources:

1. Xilinx 12.4 ISE tool.
2. Xilinx core generator tool.

B. Hardware Resources:

1. Virtex-5 Rocket IO development board.
2. SFP transceivers
3. Fibre optic cable
4. Switch Mode Power Supply(SMPS)
5. JTAG cable.

The Xilinx FPGA device (XCV5FX100T-1FF1136) used is XILINX CMOS VIRTEX-5 FX100T series with package ff1136 and speed grade -1. Here Virtex-5 FXT is a High-performance embedded system with advanced serial connectivity. It contains GTX transceivers capable of running up to 6.5 GB/s. Each GTX transceiver supports full duplex, clock and data recovery. And also contains Embedded IBM Power Pc 440 RISC CPUs.

4. STAGES IN IMPLEMENTATION

The implementation of above application involves the following stages. They are,

1) Implementation of Simulator and AFIFO.
2) Implementation of Dual Independent Aurora Links using Multi-Gigabit transceivers.

1) Implementation of Simulator and AFIFO :

Simulator logic is designed in such a way that it produces an output of 128 bit parallel data and developed using VHDL language [3]. The source code for FIFO was generated using core generator tool. It was customized during selection of FIFO. We have different types of FIFO's are available, among all the FIFO's block RAM FIFO is used in our application because it is having all the features such as FWFT and non-asymmetric ratio which are required in our project. These FIFO's are generated using Xilinx core generator tool. While generating the FIFO, user should select the write width as 128 bit, read width as 16 bit, native and the write depth is 1024.for writing and reading of data into FIFO and from FIFO, we use independent clocks.

Designing a FIFO memory subsystem seems simple and straight forward because it consists of a Random Access Memory (RAM) with two independently clocked ports. One port is used for writing. The other port is used for reading. In addition, the FIFO has two independent address counters to steer write and read data. Independently clocked means that there are no restrictions on the relationship between read and write clock frequency and phase. Independently clocked is also referred to as multi-rate or asynchronous FIFO operation.

To meet the challenge of fast, asynchronous FIFO design, the Virtex-5 family includes a dedicated, hard-coded FIFO controller inside each block RAM. The controller allows reliable FIFO operation at a clock rate of 500 MHz without using any extra logic. The designer supplies 4-bit, 9-bit, 18-bit, or 36-bit parallel input data, a continuously running write clock, a write clock enable signal, a continuously running read clock, and a read clock enable signal. Output data is always the width of input data.

The FIFO Empty signal goes high as a result of the read clock when the last data entry is read out of the FIFO. The designer must disable reads until the FIFO Empty signal goes Low again. The rising and the falling edges of the FIFO Empty signal are synchronous with the read clock, providing a totally synchronous interface. If the read clock enable signal is accidentally kept active after the FIFO is empty, a read error

flag is asserted, but the FIFO contents and addressing are not affected. FIFO Almost Empty and FIFO Almost Full are programmable status signals. These two signals can be used as a warning to slow the read or the write operation, or they can be used to indicate the data level in the FIFO. Non-asymmetric ratio of inputs and outputs is required in our application, the input to the FIFO is 128 bit parallel data and output is 16 bit parallel data.

2) Implementation of dual independent aurora links using multi-gigabit transceivers:

Multi-gigabit transceivers are present in FPGA and we use two different MGT's which are belongs to one GTX dual tile. These MGT's are configured using aurora protocol for developing the application of dual independent aurora channels for high speed serial data transmission at the rate of 3.125 GB/s.

A. Customizing Aurora core:

In virtex-5 series (Xc5vfx100t-1ff1136) we have mainly two different types of aurora versions are available, they are aurora 8B10B 5.2 and aurora8B10B 6.1.among these two types, select appropriate version depending upon application being used in your project. In order to use AXI interface is required for implementing the project I was selected aurora 8B10B 6.1.

To generate any aurora core, it involves mainly two steps. In first step user should select some core parameters for ex; number of aurora lanes, lane width, line rate and GT REFCLK etc. Here in my application, I was selected the number of lanes as ONE, lane width as 2. In each transceiver the user can select from 1 to 16 lanes. Lane width indicates that how many number of bytes that are taken as input to the aurora module. Available options are either TWO bytes or FOUR bytes. If we were selected the lane width as TWO then USER_CLK must equal to the REF_CLK in our design or if it is FOUR the USER_CLK becomes half of the REF_CLK. Since, Two bytes are selected so that the input to the aurora module has 16bits.

Here, I was selected the line rate as 3.125Gbps means that in each lane the data was transmitted at the rate of 3.125 Giga bits per second. Using virtex-5 boards we can achieve high speeds up to 3.125Gbps only. After selecting the core parameters, user select core features namely dataflow mode, interface and flow control. I was selecting DUPLEX, FRAMING and NONE respectively. I was selected in such a way that each has a full duplex communication and data has transmitted in each lane in the form of frames. Here in this we have two types of interfaces namely FRAMING and STREAMING. In framing there must be include or assign a SOF and EOF signals to each frame, whenever the destination is ready then only data is transmitted through the channel.

Virtex-5 FPGA (Xc5vfx100t-1ff1136) supports a maximum of 16 GTX transceivers. These 16 transceivers are placed on FPGA in the name of GTX_DUAL_X0Y0 to GTX_DUAL_X0Y07 as two columns. Each GTX_DUAL_TILE consists of two transceivers. In order I was successful with ISE12.4, IP core Aurora 8B10B v6.1 and followed user guides of aurora core. It's not too complicated, but it took a few steps including renaming/mapping/remapping signals.

First generate the aurora core from the core generator tool by selecting the appropriate transceiver in the GTX DUAL TILE. The selection of transceiver is based on the application being used in your project. To generate the DUAL independent aurora links in one GTX transceivers follow the few important steps. So they have total of 16 GTX transceivers are present.

a. In step1, Select the aurora lane=1, lane width=2 bytes, line rate 3.125Gbps, dataflow mode=duplex, interface=framing, flow control= none.

b. In step2, select any one of the Tile for ex, GTX_DUAL_X0Y6_0. Next click on generate, the aurora source code for X0Y6_0 was generated.

Then do the following modifications in each module of aurora, these are required for generation of dual independent aurora channels.

- <Design name>_tile.vhd: no changes required.
- <Design name>_tranceiver_wrapper.vhd: I changed each and every port of this module. Simply duplicate each port by renaming signal name as _0 and _1.

For example, LOOPBACK_IN becomes two ports: LOOPBACK_IN_0 and LOOPBACK_IN_1.

In the process of duplicating each module, you need to duplicate the signal declarations also, which are present in each module.

Finally debug your design, if any errors were found during generation bit file then try to solve it. You are facing simple errors only.

B. SFP Transceiver:

Small Form factor Pluggable transceiver (SFP) [5] is a compact, hot-pluggable transceiver used for converting the electrical signal into light signal and vice versa and these light signals are transmitted through fiber optic cable.

Note: the only ports that are not duplicated in this way are: ENCHANSYNC_IN,CHBONDDONE_OUT,CHBONDDONE_OUT_unused, REFCLK, and GTPRESET_IN and its related signals.

- <Design name>.vhd: In this module, again all the ports need to be duplicated for the 0 and 1 transceiver.

For example, DO_CC becomes DO_CC_0 and D_CC_1. The following ports do not need to be duplicated. GTPD2, RESET, GT_RESET. Similarly, you will need to instantiations of each of the following

- <design name>_AURORA_LANE_4BYTE
- <design name>_GLOBAL _LOGIC
- <design name>_AXI_TO_LL
- <design name>_TX_STREAM
- <design name>_LL_TO_AXI
- <design name>_RX_STREAM

Suppose if you are using framing interface then streaming modules are replaced by framing related modules.

Only one instantiation of

- <Design name>_GTP_WRAPPER.

Look at the signal declarations carefully to see what should be duplicated for the second lane.

- <design name>_example_design.vhd:the changes are

similar to those in <Design name>.vhd

C. JTAG Cable:

The full form of JTAG is Joint Test Action Group because it is developed by Joint Test Action Group and Sanctioned by IEEE as STD 1149.1 test access port and Boundary Scan Architecture in 1990. JTAG cable is used to downloading the BIT file or programming file from PC into the target hardware board.ie, virtex5 FPGA board.

5. Chip Scope Pro Analyzer

Simulation based method is widely used for debugging the FPGA design on computers. Time required for simulating complex design for all possible test cases becomes prohibitively large and simulation approach fails. For rapid testing, such designs can be loaded on to the target FPGAs and tested by applying test inputs and directly observing their outputs. As the complexity of the design under test increases, so does the impracticality of attaching test equipment probes to these devices under test. The Chip Scope Pro tools integrate key logic analyser and other test and measurement hardware components with the target design inside the FPGA. Computer based software tool communicate with these hardware components and provide a designer robust logic analyser solution.

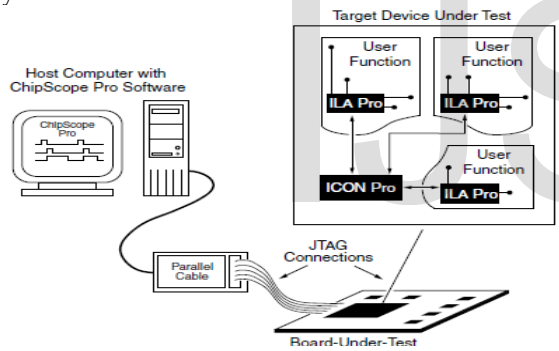

Fig:5.Chipscope pro system block diagram

This figure shows a block diagram of a system containing debug cores added using the Chip Scope Pro tools. You can place the ICON, ILA, VIO, and ATC2 cores (collectively called the Chip Scope Pro cores) into your design by generating the cores with the CORE Generator tool and instantiating them into the HDL source code. You can also insert the ICON, ILA, and ATC2 cores directly into the synthesized design net list using the Chip Scope Pro Core Inserter or Plan Ahead tools. You then place and route your design using the ISE implementation tools. Next, download the bit stream into the device under test and analyse the design with the Chip Scope Pro Analyser tool.

The type of cores required for using Chip Scope Pro analyzer are

1. ICON (Integrated Controller)
2. ILA (Integrated Logic Analyzer) and
3. VIO (Virtual Input/output).

Chip scope pro is a software logic analyser and can be integrated in ISE project as a component.To communicate with the host computer Chip Scope Pro Cores use the JTAG (either a parallel or a USB cable) Boundary scans port.

Chip Scope features

Chip Scope has following features:

– Can be connected to Inputs
– Can be connected to Outputs
– Can be connected to Intermediate signals
– Can have a variable number of inputs/triggers
– Records up to 16384 samples
– Can create virtual internal inputs (such as buttons)
Those are activated from the GUI application on a PC.

6. Results

The following are the results obtained during the transmission of 128 bit parallel data and reception as well for the final integrated project in chip scope analyzer.

In the figure 4 the signals are channel up, lane up, Tx_data and Rx_data are shown. For the effective data transmission channel up should be one. To keep the channel link continuously active, idle sequences are sent when actual data is not present.TX_TDATA0, TX_TDATA1 are the two signals in which the data is present and transmitted over two dual independent channels. Here data sent over the Tx_data is 00B9E2391640801E0024803203E807D0, the same data is received on the Rx_data signal line (rx_data_i_0, rx_data_i_1) after some delay that is 32 clock cycles of latency which is in the permissible limit (Maximum latency for a 2-byte designs from TX_SOF_N to RX_SOF_N is approximately 53 GTP/GTX transceivers clock cycles in simulation.)

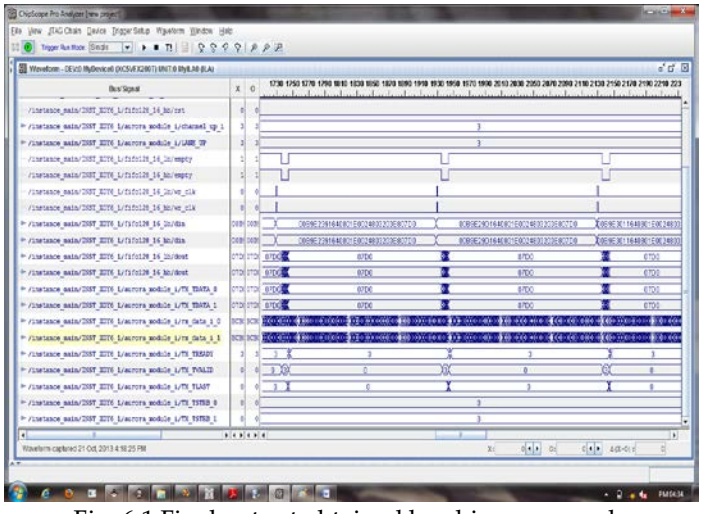

Fig. 6.1 Final output obtained by chip scope analyser

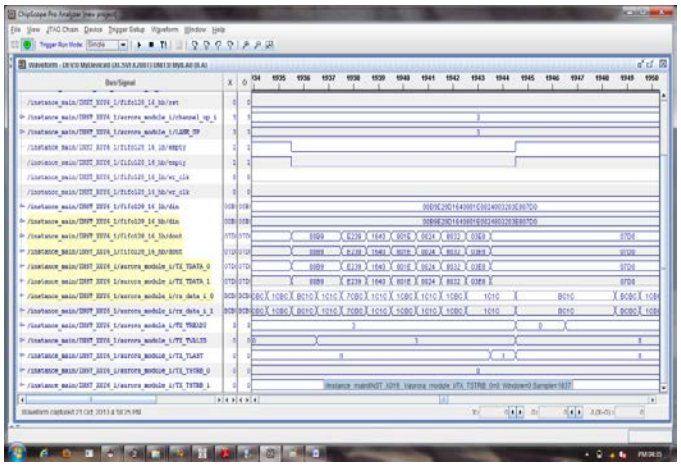

Fig. 6.2 The output shows the transmitted data over dual independent channels

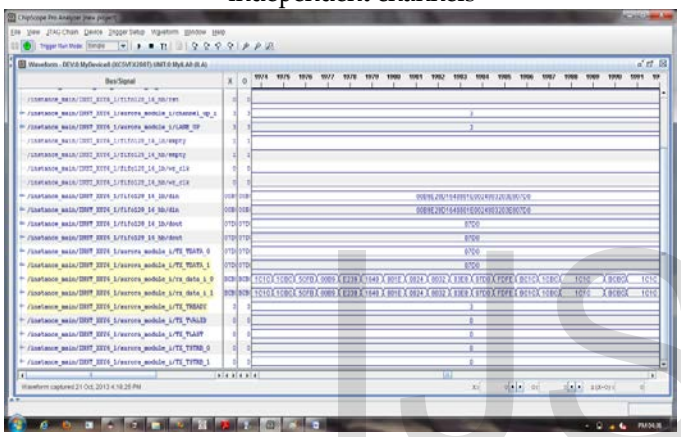

Fig.6.3 The output shows the received data over dual channels.

7. Conclusion

The 128 bit parallel data's are transmitted serially at the rate of 3.125Gbps over dual independent aurora links present on one GTX DUAL TILE through fiber optic cable using multi gigabit transceivers and received by the multi-gigabit transceivers of the same TILE using optical fiber loopback. Finally both the transmitted data and received data's are verified by using chip scope analyzer software.

8. Future Scope

The present paper was implemented dual independent aurora links on one GTX DUAL TILE for serial data transmission at the rate of 3.125Gbps using aurora protocol. We can achieve the 6.25Gbps speed also with some other vertex series boards which supports 6.25Gbps line rate using aurora protocol. The other protocol is Serial Rapid IO (SRIO) which works on the principle of data packets switching is more efficient for error free transmission and speed can be increased to higher level

Acknowledgment

The authors would like to thank D.KRISHNAVENI, SC.'D', DLRL, HYDERABAD, for her support in implementation of the project and R.RAMMOHAN, SC.'D for giving his valuable suggestions.

References

[1] T.Vijay Basker Reddy, "*A Real Time Implementation of High Speed Data Transmission using Aurora Protocol on Multi-Gigabit Transceivers in Virtex-5 FPGA*", *IJRCTT: International Journal of Research in Computer and Communication Technology, vol-1, no.3, pp. 106-111, Aug 2012.*

[2] Clive MAX Maxfield, "*The design warriors' guide to FPGAs*", 2004.

[3] Volnei A. Pedroni, "*Circuit Design with VHDL*", 2004.

[4] Abhijit Athavale and Carl Christensen of Xilinx "*High speed serial I/O made simple- A designers' guide .with FPGA applications*", 2005.

[5] Xilinx, Aurora 8b/10b protocol specification, available at "*http://www.xilinx.com/support/documentation/ip_documentation/aurora_8b10b_protocol_spec_sp002.pdf*", SP002 (v2.2) April 19, 2010.

[6] Xilinx, LogiCORE IP Aurora 64B/66B v6.2, available at "*http://www.xilinx.com/support/documentation/ip_documentation/aurora_64b66b/v6_2/ds815_aurora_64b66b.pdf*", DS815 Jan, 2012.

[7] Xilinx, Chip Scope Pro Software and Cores User Guide, available at "*http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_1/chipscope_pro_sw_cores_ug029.pdf*", UG029 (v13.1), March 1, 2011.

[8] Xilinx, ISim User Guide user guide, available at "*http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_1/plugin_ism.pdf*", UG660 (v 13.1) March 18, 2011.

[9] Xilinx, LogiCORE ip aurora 8b/10 v6.2 user guide, available at "*http://www.xilinx.com/support/documentation/ip_documentation/aurora_8b10b_ug353.pdf*", UG353 (v6.2) July 23, 2010.

[10] Xilinx, LogiCORE IP FIFO Generator v8.1, DS 317, 2011, available at "*http://www.xilinx.com/support/documentation/ip_documentation/fifo_generator_ds317.pdf*", DS317 March 1, 2011.

[11] Joint Test Action Group, available at "*http://www.hardwarebook.info/JTAG*".

[12] *Data Sheet -850 nm, SFP (Small Form Pluggable), RoHS Compliant, Low Voltage (3.3 V) Digital Diagnostic Optical TransceiverbyAVAGOTechnologies.* "http://www.avagotech.com/docs/AV02-0881EN".