VHDL Design and Synthesis of PCI Express Bus Controller

Abhinav Bhat[1]
Faculty of Technology
UTU,Dehradun,Uttarakhand
manishbansal635@gmail.com

Amit Kumar[2]
Faculty of Technology
UTU,Dehradun,Uttarakhand
amit989762@gmail.com

Aakanksha Devrari[3]
Women Institute of Technology
UTU,Dehradun,Uttarakhand
aks.uit08@gmail.com

Manish Bansal[4]
Faculty of Technology
UTU,Dehradun,Uttarakhand
abhinavbhatt91@gmail.com

Abstract—This paper presents the design and implementation of 64-bit Peripheral Component Interconnect Express (PCIe) using VHDL. In this thesis, MAC and Physical layer are interconnected using high speed serial communication PCIe bus using a dedicated controller. Peripheral devices like Keyboard, Keypad and RS232 are designed to receive data form external world. Timer is designed for up-count and down-count purpose. DMA is designed to increase the speed of system. CCU is designed to control the flow of data between receiver and transmitter. Finally PCIe bus is designed to interconnect all these modules as a high speed serial communicator. This design is synthesized on Virtex 4 xqr4vsx55-10cf1140 FPGA. VHDL programming language is used to develop the design and simulation results are obtained using Xilinx 14.3 ISE design suit.

Index terms— PCB, PCIe, DMA, DTMF, RTC, DMA, CCU, CPU

II. INTRODUCTION

The expansion card also known as adapter card, expansion board or accessory card is a printed circuit board (PCB) that can be inserted into an expansion slot on a compute rmotherboard. It boosts up the performance of the computer and increases functionality using expansion bus. It is a bus which transfers data or information between the internal hardware of a computer including the CPU and RAM and external peripheral devices.

The most advance and common type of expansion slot being used these days is PCI Express (PCIe). PCI Express (PCIe) is a point-to-point (PnP) connection, i.e., it connects only two devices at a time; and no other device can share this connection during this time. On a motherboard which has standard PCI slots, all PCI devices and peripherals are connected to the PCI bus and they share the same data path, so performance may decrease because it may happen that more than one device wants to transmit or to receive data at the same time. A motherboard which has PCI Express slots, each PCIe slot is connected to the motherboard processor using a dedicated lane, not sharing its lane or data path with other PCIe slots. Also, devices or peripherals integrated on the motherboard, such as SATA, network and USB controllers, are usually connected to the motherboard processor using dedicated PCIe connections.

The PCI Express or simply PCIe connection is based on the concept of a "Lane," which is a single-bit (1 bit), full-duplex, high speed serial communication. Lanes can be grouped together to increase bandwidth in order to increase the system speed. Let's take an example, when two different devices are using four lanes for their connection, it will be considered as an "x4" connection and it will be able to achieve four times (x4) more bandwidth than a single connection (x1), i.e., a single lane. In Figure 1, we have illustrated two devices that are connected using two lanes, i.e., an 'x2' connection. Although any number from one to 32 lanes can be grouped, the most common numbers are x4, x8, and x16.

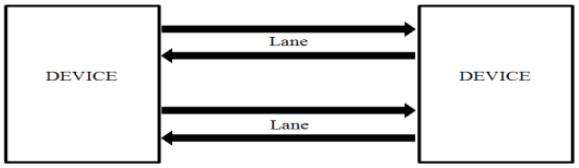

Fig.1 PCI Express x2 connection

III. INTRODUCED ARCHITECTURE

The proposed architecture for PCIe of 64 bit is shown in the Figure 2. The major parts of the architecture are the CPU, CCU, peripherals devices such as keyboard, keypad, RS232, timer and DDR2 and transmitter which generates the data and receiver. MAC data process controller is used to control the flow of data in various peripheral devices.

PCIe bus is used for interfacing between the devices. Direct Media Access (DMA) is used between DDR2 RAM and the bus for fast data transfer. Physical layer and PCIe bus communicate with each other i.e. transferring of data through the MAC-PHY interface present between them. Arithmetic and logic unit (ALU) is present in the Central Processing Unit (CPU).

It performs many arithmetic and logical operation like addition, multiplication etc. It uses data from the memory and uses Accumulator to perform the involving calculations. Peripherals such as Keyboard, Keypad, RS232, Timer and DDR2 RAM are used in the system. Separate device addresses are specified for every peripheral. Only one device can have

the liberty to use PCIe bus at a time. This solves the problem of data congestion and increases the overall system speed.

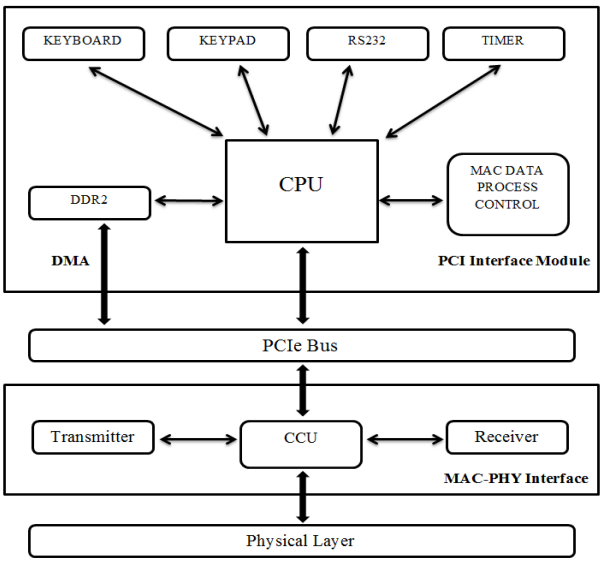

Fig.2 Proposed PCIe architecture

2.1 CPU

The first entity is the CPU which is responsible for all the arithmetic and logical calculations required for the data. A and B both are the 64 bits input lines, Device Address is of 3 bits that is used to enable the CPU. PCI_in is used to enable the PCI Interface Module which is the upper block of the proposed architecture. Y and Z both are 64 bits output lines. Select (3 bits) is used for the selection of the different operations of the CPU such as addition, subtraction, AND operation, OR operation, NOT operation, NAND operation, NOR operation, XOR operation, Increment operation, Decrement operation, Pass operation, Zero operation, Shift-left operation, Shift-right operation, multiplication and rotation operation.

2.2 DDR2 RAM

The DDR2 Ram consists of four register banks. Each bank has eight separate registers that can hold 64-bits at a time. It can store that much of data and can transmit it to the PCIe bus when it is enabled or the data is required. Direct Media Access (DMA) is used for the direct sharing of the data from the DDR2 Ram and PCIe bus without the involvement of the CPU. Due to this, the data that didn't have to be modified need not to be sent to CPU and can be directly shared with PCIe bus. That saves time and computational power of the system. CCU controls the flow of data between the transmitter and the receiving unit by temporarily storing the data.

2.3 KEYBOARD

A matrix keyboard is designed for the purpose of a peripheral device. It has 16 alpha-numeric keys, 0 to 9 for numeric and A to F for alphabets. Rows and columns are connected to power supply along with the resistors. A switch is used for every key. When a key is pressed, one row and column is shorted, hence completing the circuit; and that key is displayed.

2.4 KEYPAD

For designing the 16-key keypad, the concept of DTMF is used. Every key is composed of two specific frequencies. When a specific key is pressed, it generates dual tones (one frequency is higher than other) and then the output of low frequency and high frequency is sent to the PCIe bus output.

2.5 TIMER

Timer is used to count for every clock pulse that for incoming data up to 64-bits. It is synchronized with the clock pulse and provides the real time associated with real time clock (RTC).

2.6 DMA

Direct Media Access (DMA) is used for the direct sharing of the data from the DDR2 Ram and PCIe bus without the involvement of the CPU. Due to this, the data that didn't have to be modified need not to be sent to CPU. That saves time and computational power of the system.

2.7 CCU

MAC-PHY interface is controlled by the designed Central Control Unit (CCU) that controls the flow of the data between the transmitter and the receiver.

2.8 RS232

The RS232 utilizes many input signals such as 'cts', 'rts' etc. It receives data from the external input line and then re-transmits the data into the PCIe output data line when the peripheral device is enabled.

2.9 ARBITER LOGIC

There are many different modules and peripherals designed that share the PCIe bus in sequential manner. This means that every peripheral device or module have the luxury to utilize the functions of the designed bus but only one at a time. No two peripheral devices or modules can use PCIe bus simultaneously.

Hence every peripheral device is to be enabled or activated at a time which is using PCIe bus and others have to be in de-activated mode. Due to this ability of the design, there is no problem of data congestion or interference between other device's data and activated device can utilize whole bandwidth of the bus.

For this reason, Arbiter logic is developed. Every peripheral device or module has given an input pin 'Device_address'. It is a 3 bit input address pin. This pin is used to specify the

Arbiter address. Every peripheral device has provided its own different 3 bit Arbiter address. Table 1 shows the device in operation to the corresponding arbiter address.

Table 1 Arbiter Logic

Arbiter Address	Device Operating
000	RS232
001	DDR2
010	DMA
011	Timer
100	Keypad
101	Keyboard
110	CCU
111	CPU

2.10PCIE BUS CONTROLLER

Suppose the user doesn't have the knowledge of the Arbiter logic or the addresses to use the desired peripheral device or module. How the user can activate and then de-activate the particular peripheral device without knowing its address. This is the reason behind designing PCIe bus controller. Even the user doesn't know about the Arbiter address or 'Device_address', the desired peripheral device can be activated automatically.

For that, 'Control_logic' pin in the bus controller is to be set to '1' and the device address will be incremented by '001' for every clock pulse or event. And hence, every module will be checked after every clock pulse. Figure 3 shows the entity of the bus controller.

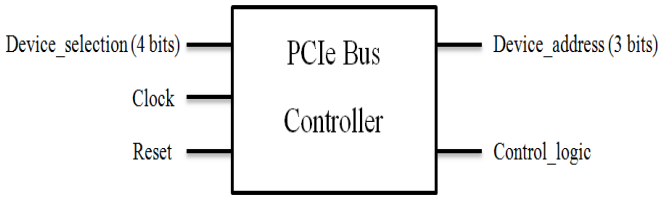

Fig. 3 Entity of PCIe Bus Controller

IV. SIMULATION RESULT

The top level RTL of the designed system is shown in the figure 4 and figure 5 shows the detailed RTL schematic diagram for the designed system.In figure4, inputs and outputs from every module are present along with the clock, ground and 'Device_address' pin. 3-bit input arbiter address can be seen in the RTL.

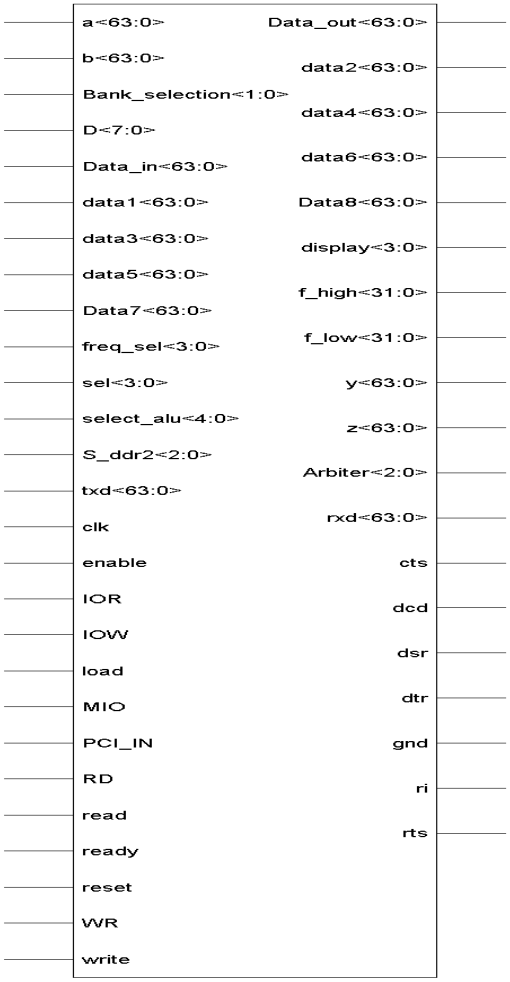

Fig. 4 Top Level RTL of designed system

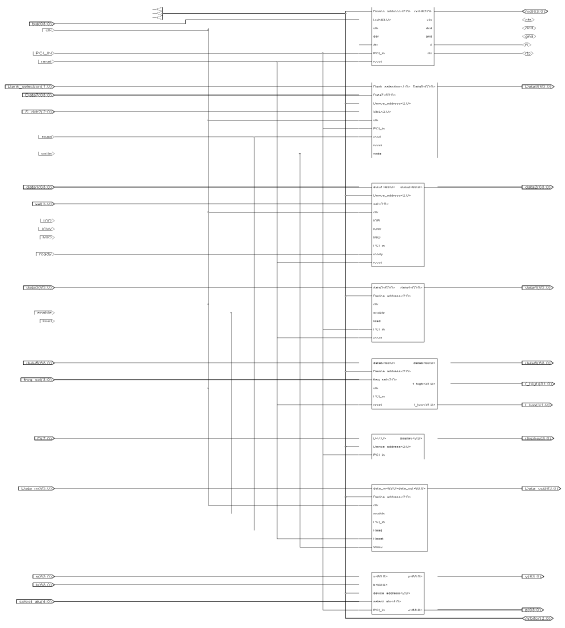

Fig. 5 Detailed RTL schematic of designed system

Figure 6 shows the simulation result for the designed Keypad.. 4567d is provided as the input data to 'data5' and '2' is supposed to be pressed on the keypad which can be seen on 'freq_sel' input pin. When the dive is enabled, 'data6' gets the data as output and corresponding frequencies can be viewed on 'f_low' and 'f_high' pins.

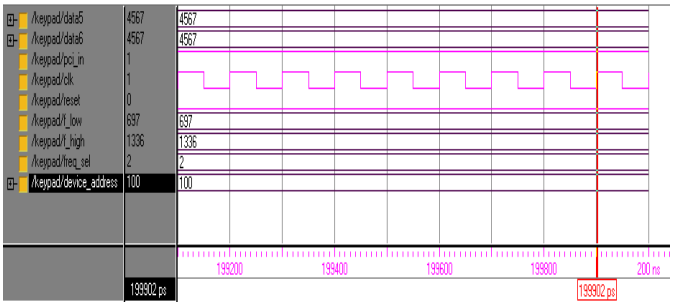

Fig. 7 Simulation waveform of the Keypad

Figure 7 shows the simulation result of the system. All the module inputs are given to their respective inputs. Arbiter address is set to '100' i.e. Keypad is activated. This means only Keypad has lever age to use the PCIe functionality. All output results are as expected and hence, correct working is verified of the designed system.

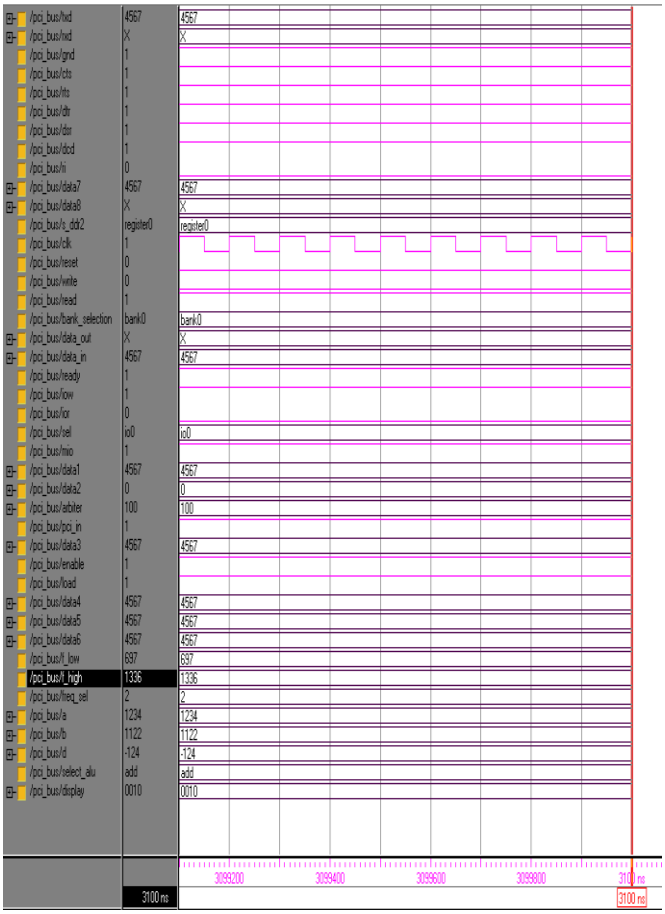

Fig. 7 Simulation waveform of the system

V. Synthesis Report

Device utilization summary on Virtex 4 xqr4vsx55-10cf1140 FPGA device for the designed system is shown in the table 2.

Table 2 Device Utilization Summary for PCIe system

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	862	24576	3%
Number of Slice Flip Flops	1038	49152	2%
Number of 4 input LUTs	1543	49152	3%
Number of bonded IOBs	1133	640	177%
Number of GCLKs	5	32	15%
Number of DSP48s	3	512	0%

Table 3 Timing details

Speed grade	-10
Minimum period:	5.320ns
Maximum Frequency	187.964MHz
Minimum input arrival time before clock	10.871ns
Maximum output required time after clock	4.807ns
Maximum combinational path delay	9.153ns
Total memory usage is	346380 kilobytes

VI. CONCLUSION

The simulation results of the modules, sub-modules and the designed system are obtained using Xilinx ISE 14.3 design suite successfully. In device utilization report, number of slice flip flops are 2%, number of slices are 3%, number of 4 input LUTs are 3%, number of bonded IOBs are 177% and number of GCLKs are 15% for the Virtex 4 xqr4vsx55-10cf1140 FPGA device. Minimum period is found to be 5.320ns, maximum frequency is 187.964MHz, and minimum input arrival time before clock is 10.871ns. Maximum output required time after clock is 4.807ns, maximum combinational path delay is 9.153ns and Total memory usage is found to be 346380 kilobytes. In this system, PCIe data bus is of 64-bit and optimized parameters are obtained.

REFERENCES

[1] A. Wilen, J.P. Schade and R. Thornburg, "Introduction to PCI Express: A Hardware and Software Developer's Guide", ISBN-13: 978-0970284693, Intel Press. 1 April, 2003.

[2] B.R. Williams and A. Chuvakin, "PCI Compliance: Understand and Implement Effective PCI Data Security Standard Compliance", Fourth Edition, Syngrees. January 2014.

[3] C.W. Liang, N.B.Z Ali and R.S.Nair, "Design of Low Cost FPGA Based PCI Bus Sniffer". *Electrical & Electronics Engineering Department, Universiti Teknologi Petronas, 31750 Tronoh, Perak, Malaysia*, pp (420-423), 2006.

[4] C.Y. Chang, Y.J. Chang, K.J. Lee, J.C. Yeh, S.Y. Lin and J.L. Ma, "Design of On-Chip Bus with OCP Interface". *Department of EE, National Cheng Kung University, Taiwan*. IEEE Xplorer, pp (211-214), 2010.

[5] G. Shiny and S.H. Rao, "Design of ON-Chip Bus OCP Protocol with Bus Functionalities". *International Journal of Power System Operation and Energy Management ISSN*, Volume-3, Issue-1, pp (14-18), 2013.

[6] G. Wang, G. Kang and H. Wang, "Design and FPGA Implementation of MAC-PHY Interface based on PCI Express for Next-Generation WLANs", *Key Laboratory of Universal Wireless Communications, Ministry of Education Beijing University of Posts & Telecommunications, Beijing 100876*, P.R.China. IEEE Xplorer, 2012.

[7] M. Shen and Z. Peng, "Design and implementation of power plants disturbance synchronous timing card of PCI-I04 bus based on STM32 processor", *China High-Tech Enterprises*, Volume-24, pp (108-110), 2012.

[8] M. Yang, L. Wu and H. Shi, "Design of PCI-104 Bus Interface of Low Power Embedded CPU based on FPGA", *2nd International Conference on Measurement, Information and Control*, IEEE Xplorer, pp (275-279), 2011.

[9] M.A Mazidi, J.G. Mazidi and R.D. McKinlay, "The 8051 Microcontroller and Embedded Systems", Second Edition.

[10] N. Kaur, A. Kumar and L. Gupta, "VHDL Design and Synthesis of 64 bit RISC Processor System on Chip (SoC)". *IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)*, Volume-3, Issue-5, pp (31-42), 2013.