The Use of SystmeC for Design Verification of PCI Express Endpoint RTL Core

Sun-Wook Kim*, Young-Woo Kim** and Kyoung Park***
*Dept of Internet Server, Eletronics and Telecommunications Research Institute,DaeJeon,305-350,Korea
Tel : +81-42-860-5842 Fax : +81-42-860-6699 E-mail: swkim99@etri.re.kr
**Dept of Internet Server, Eletronics and Telecommunications Research Institute, DaeJeon, 305-350, Korea
Tel : +81-42-860-1619 Fax : +81-42-860-6699 E-mail: bartmann@etri.re.kr
***Dept of Internet Server, Eletronics and Telecommunications Research Institute, DaeJeon,305-350, Korea
Tel : +81-42-860-3809 Fax : +81-42-860-6699 E-mail: kyoung@etri.re.kr

Abstract: In this paper, we present a design and experiment of PCI Express core verification model. The model targeting Endpoint core based on Verilog HDL is designed by newly-emerging SystemC, which is a new C++ class library based system design approach. In the verification model, we designed and implemented a SystemC host system model which acted as Root Complex and device driver dedicated to the PCI Express Endpoint RTL core. The verification process is scheduled by scenarios which are implemented in host model. We show that the model is useful especially for verifying the RTL model which has dependencies on system software.

Keywords: PCI Express Endpoint, SystemC, Verification, Host system model

1. INTRODUCTION

Verification is a process used to demonstrate the functional correctness. Due to the increasing complexity and size in designing protocol processing hardware, verification has become a large and intractable problem, and the testing for system level verification takes more and more time. In millions gate ASIC, for example, verification consumes 70% of the design effort, and the code that implements the testbenches makes up to 80% of the total code volume [1].

In recent years, new methods and tools for verification are being needed in order to make verification process more easy, more precise and faster. SystemC is considered one of the languages that will allow and increase abstraction level for hardware modeling and will ease the effort of integrating the software modules in the context of a formalized system level design and verification methodology [2]. And SystemC is a C++ class library and provides support for hardware-oriented data types like modules, ports and signals, which can be used to effectively create a cycle-accurate model of software algorithms, hardware architecture, interfaces of SoC (System On Chip) and system-level design. Using C/C++ for all system level modeling enables flexibility and faster simulation in comparison with other methodologies (10 to 100 times faster than VHDL or Verilog). It is a considerable merit that SystemC can do co-simulation with other HDL (Hardware Description Language): VHDL and Verilog. Also, designs of hardware based VHDL/ Verilog can be tested by the same SystemC stimuli, and verification of VHDL/Verilog code can be done faster [1].

In this paper, we designed and implemented a verification model about PCI Express Endpoint RTL core using SystemC, and described the verification process which is to verify the functional correctness of core using the developed SystemC model.

We have organized the paper as follows. In section 2 of this paper, we introduce the PCI Express technology which is one of new I/O interconnection which is introduced by PCI-SIG in 2002. In section 3, we show the architecture of verification model and implementation detail in SystemC environment. Experimental results about verification process are described in section 4. Finally, conclusions and plans for future work are described in section 5.

2. THE PCI EXPRESS SYSTEM ARCHITECTURE

PCI-Express architecture is a new serial interconnect technology that retains the PCI usage model and software interfaces while supporting chip-to-chip, board-to-board and adapter solutions at an equivalent or lower cost structure than existing PCI designs [3]. The PCI Express technology is defined as serial I/O point-to-point interconnect, packetized protocol and a layered architecture that enables attachment to copper, optical, or emerging physical signaling media. The PCI Express is specified in layers as shown in Fig. 1.

Compatibility with the PCI addressing model (a load-store architecture with a flat address space) is maintained to ensure that all existing applications and drivers operate unchanged. The software layers will generate read and write requests that are transported by the transaction layer to the I/O devices using a packet-based split-transaction protocol. The link layer adds sequence numbers and CRC to these packets to create a highly reliable data transfer mechanism. The basic physical layer consists of a dual-simplex channel that is implemented as a transmit pair and a receive pair. The initial speed of 2.5Gb/s/direction provides a 200MB/s communications channel that is close to twice that classic PCI data rate [4]. Also, It contains a Root Complex (the Host Bridge) and

several Endpoints (the I/O devices) [4]. Multiple point-to-point connections are established by Switch which replaces the multi-drop bus and is used to provide fan-out for the I/O bus.

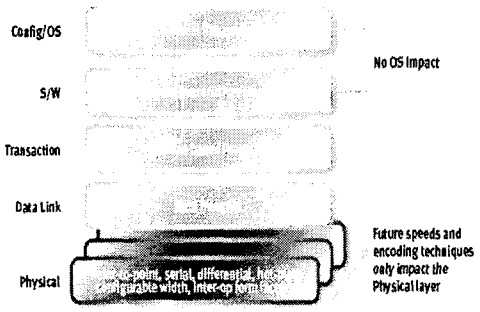

Fig. 1. The layered architecture of PCI Express.

The switch may provide peer-to-peer communication between different endpoints, and it is shown as a separate logical element but could be integrated into a host bridge component [3].
The PCI Express topology is represented on Fig.2.

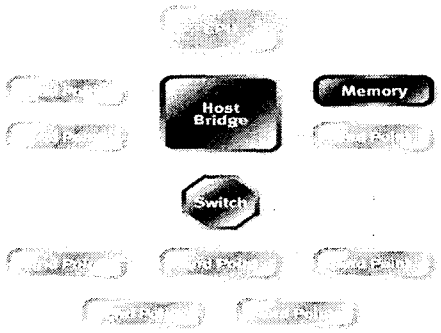

Fig. 2. The PCI Express Topology.

3. THE PCI EXPRESS ENDPOINT RTL CORE VERIFICATION MODEL

3.1. System modeling based on SystemC

SystemC is distributed and supported by the The Open SystemC Initiative(OSCI). OSCI, a consortium of major EDA and IP companies provides a set of C++ class libraries and a simulation kernel to extend the capabilities of C/C++ as a modeling tool. Adding C++ class libraries enables time control, concurrency and answer on stimulus to standard C/C++ language [5].

Structural design is implemented in SystemC using modules, ports, processes and signals [6]. A variety of data types are supported to include single bits, bit vectors and fixed-point integers. Four value logic signals are also supported to allow the implementation of tri-state buses.

Modules are the basic building blocks, and act as containers for collections of ports, signals, internal data structures and processes. The use of modules allows a complex system to be divided into a number of simpler elements early in the design process. Modules are declared using the SystemC reserved word SC_MODULE.

Process is internal function that establishes a module's behavior and functionality. It responds to changes in module inputs by updating the state of internal data structures, altering values present at the module's output ports, or both. SystemC provides three types of processes : Methods, Threads and Clocked threads. Each processes are declared using SystemC reserved words: SC_ METHOD, SC_THREAD and SC_CTHREAD.

Ports provide the data path between a module's environment and its internal processes. SystemC provides three types of ports: input port declared with the SystemC reserved word SC_IN, output port declared with the reserved word SC_OUT, and inout port declared with SC_INOUT.

Signals are closely related to ports. They provide the data paths necessary to connect devices and are declared using SC_SIGNAL [7].

3.2. PCI Express Verification Model

In this paper, we designed and implemented a verification model that is used to verify the functional correctness of a PCI Express Endpoint RTL core based on Verilog HDL (Hardware Description Language).

The verification model consists of four components: host model, PCI Express downstream model, DUT(Design Under Test), and application model. The full structure of PCI Express Endpoint core verification model is represented on Fig.3.

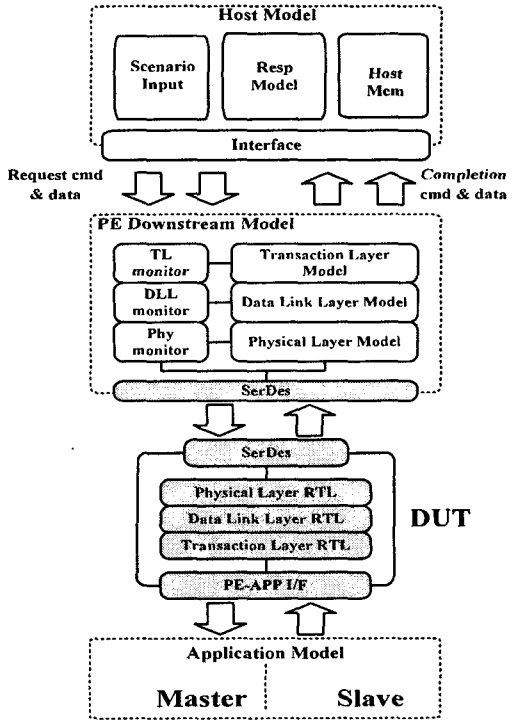

Fig. 3. The full Structure of PCI Express verification model.

The host model in verification model, which acts as Root Complex and device driver dedicated to the PCI Express Endpoint, is designed by using SystemC. The model generates a verification command, a response about request from DUT, and a completion of received packet from DUT.

The downstream model which generates and processes PCI Express packet has three layer models and monitor

models. The layer models process the PCI Express protocol like layers in PCI Express architecture, and the monitor models is used to inspect the data flow and protocol error in each layer models.

The DUT is a PCI Express Endpoint RTL core based on Verilog HDL.

The application model which is a functional model connected to a PCI Express Endpoint generates the response and request for PCI Express Endpoint device. It also makes it possible to verify the functional correctness of DUT.

3.3. SystemC Modeling of Host Model

In order to verify the PCI Express Endpoint RTL core, The host model generates transactions which are used to configure the PCI Express connection and to verify the function correctness. Also, the model validates the processing results about generated transactions.

The model makes out the scenario sets which are based on transaction functions and sends the composed scenario sets. Also, the model receives and processes the requests for memory and I/O space read/write from DUT. The processing results are stored in memory and/or sent to downstream model.

The host model consists of four modules: host memory module designed for host memory; scenario input module designed for operating the verification scenario sets based on implemented functions about PCI Express transactions; responder module designed for processing the transaction request from DUT and sending the processing results to downstream model; and host top module designed for integrating the modules in the host model.

The processes executed in each modules are clocked thread process type. A sensitivity list of clocked thread processes is limited to one edge of single clock input that matches the way a clocked hardware device typically functions. Designed processes are synchronized by positive edge of clock.

The host model block diagram is represented on Fig.4. Table 1 presents the behaviors and declarations of implemented processes in each modules.

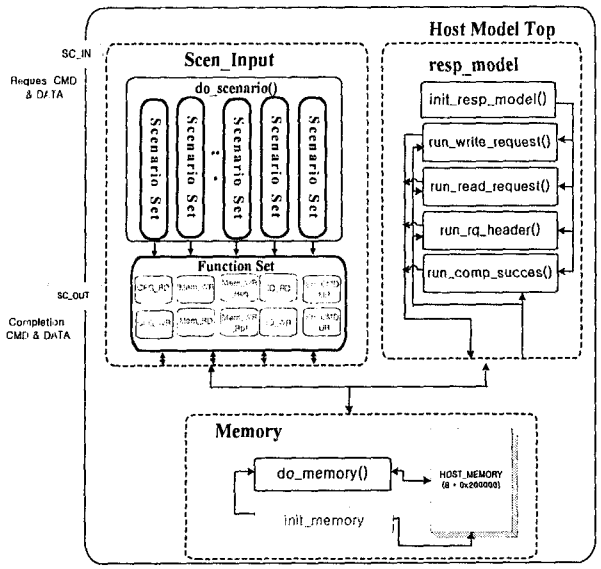

Fig. 4. The block diagram of host model

Table 1 The declared module and process in host model

Module	Process	
SC_MODULE (Scen_Input)	declaration	*SC_CTHREAD (do_scenario,clk_pos())*
	description	Generation and processing verification scenario sets.
SC_MODULE (resp_model)	declaration	*SC_CTHREAD (init_resp_model, clk_pos())*
	description	Initialize of responder module.
	declaration	*SC_CTHREAD (run_write_request, clk_pos())*
	description	Processing the memory write request from DUT.
	declaration	*SC_CTHREAD (run_read_request, clk_pos())*
	description	Processing the memory read request from DUT.
	declaration	*SC_CTHREAD (run_rq_header, clk_pos())*
	description	Store the request header for processing completion of request
	declaration	*SC_CTHREAD (run_comp_success, clk_pos())*
	description	Compare the completion header and request header for finish the request
SC_MODULE (Memory)	declaration	*SC_CTHREAD (do_mem, clk_pos())*
	description	Read and write data in memory for processing the memory transactions.

4. THE RESULT OF VERIFICATION EXPERIMENTS

The verification of PCI Express Endpoint RTL core based on verification model is progressed by executing the scenario sets in host model. The scenario sets play a role of testbench for verifying functional correctness of DUT.

Fig.5 presents the hierarchy of verification model structure. There are four components: RVS_Top designed for integrating several modules of verification; Host_Top designed for host model; RTL_TOP designed for merging the RTL core of DUT; and TL_TOP designed for PCI Express downstream model.

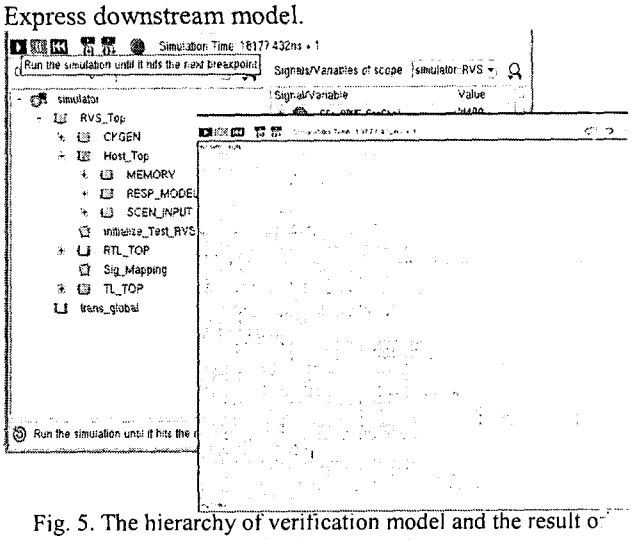

Fig. 5. The hierarchy of verification model and the result of initialization processing

Also, Fig.5 presents the results of initialization process results of the verification model.

In order to start the verification of DUT, one of the scenario sets must be selected by operator. A Scenario set like as the device initialization by device driver and several scenario sets for verifying all kinds of PCI Express transactions are included in the implemented scenario sets. The initial screen of verification process is represented on Fig.6. If the operator selects the eighth scenario set, the process for device initialization by device driver function is progressed.

Fig.6. The initial screen of verification process

The simulation through verification scenario sets generates a log file about test results during verification run. The log file includes requests information, received data, processing results, completion information and etc. Additionally, the simulation outputs a VCD trace file as shown in Fig.7 and Fig.8.

Fig 9 presents a VCD trace file about memory write transaction of responder model.

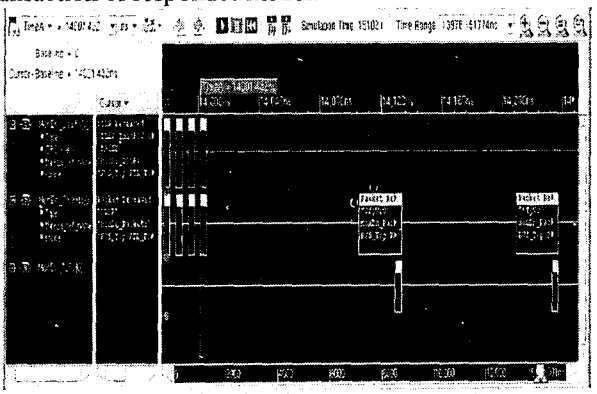
Fig.7. Trace file showing request transactions.

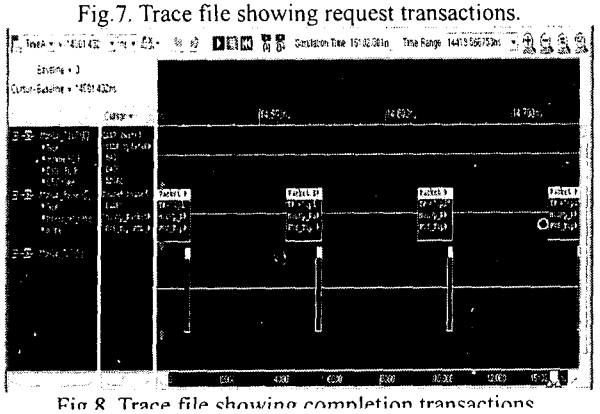
Fig.8. Trace file showing completion transactions

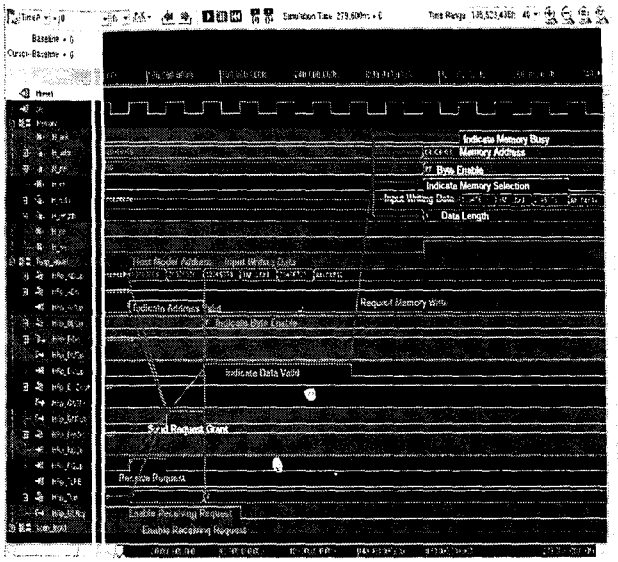

Fig.9. VCD Trace file showing the responder model activity for memory write transaction.

5. CONCLUSION

In this paper, we introduced the structure of PCI Express verification model and described a complete design and verification flow of host model which acts as Root Complex and device driver dedicated to the PCI Express Endpoint. The host model is designed by newly-emerging SystemC. Through the verification of PCI Express Endpoint RTL core using the host model, The developed verification model is effectively verifying the RTL code which has strong dependencies on device driver or system software. As a result, if a DUT is verified by a SystemC model which implement a device driver's behavior deeply related to H/W, we can verify not only functional correctness of H/W itself but also interaction with system S/W.

The RTL code which is verified by using the developed verification model have been implemented with FPGA and is now on testing on the Linux based system environment.

References

[1] A. Randjic, N. Ostapcuk, I. Soldo, P. Markovic and V. Mujkovic, "Complex ASICs Verification With SystemC," *Proc. of the Int. Conf. On Microelectronics*, Univ. of NIS, Yugoslavia, May 12-15, pp.671-674, 2002.

[2] M.Bombana and F.Bruschi, "SystemC-VHDL co-simulation and synthesis in the HW domain," *Proc. of the Design, Automation and Test in Europe Conference*, 2003.

[3] A.Bhatt, "Creating a PCI Express Interconnect," white paper, *Intel Corporation*, 2002.

[4] PCI Express Base Specification 1.0a, *PCI-SIG*, Apr. 15, 2003.

[5] P. R. Panda, "SystemC- A modeling platform supporting multiple design abstractions," *ISSS'01*, Montreal, Quebec, Canada, Oct. 1-3, pp.75-80, 2001.

[6] SystemC User's Guide Version 2.0, May 2001, *http://www.systemc.org*.

[7] J.Booth and J.Kulick, "SystemC Modeling of a Parallel Processor Broadcast Interconnection System," *Proc. of the IEEE Southeast Conference*, pp. 76-81, 2002.