We take you to the leaders.

HDL VERIFICATION SPECIAL SECTION

by Nupur Shah,
Design Engineer,
nupur@xilinx.com

Verifying PCI Designs

Your PCI design must be PCI compliant to work along with other vendors' PCs and PCI add-in boards. Therefore the final stage in the design flow of a PCI design is design verification, which consists of two steps: **functional verification** and **timing verification.**

By using a pre-designed PCI interface core, with predictable timing (such as Xilinx LogiCORE PCI products) the PCI protocol and timing is already verified. Therefore, you can focus on your unique back-end design and how it interfaces to the PCI core. However, even though the direct interface to the PCI bus is pre-verified, it is still your responsibility to verify the compliance of your final design.

The PCI protocol is very complex, and as a result, it is difficult to test every possible combination of transactions. Although simulation covers most of the functional test, the only way to verify full PCI compliance is to test the actual hardware. This can be accomplished at the quarterly PCI-SIG "plug-fests." At these events, PC manufacturers and PCI board manufacturers gather to test their

products. By implementing your PCI design in a Xilinx FPGA, the hardware can easily be changed and fixed if a problem is found.

PCI Functional Verification

To test PCI compliance, you will need to implement the PCI Special Interest Group (SIG) Test Scenarios for Compliance Testing. These scenarios test the basic transactions between two agents on a PCI bus; one agent being the device under test (DUT) and other being the behavioral model of a PCI Initiator. These tests verify if the DUT is PCI complaint or not. You have the option to purchase a testbench from a third party vendor. However, learning these testbenches can prove to be just as difficult as developing one yourself, and there is no guarantee that they provide full fault coverage.

If you choose to develop a testbench, there are two types of testing that must be performed in order to verify that your design is functional: system-level testing and PCI bus protocol testing. System-level testing sets up modules in a system to transfer data back and forth. It checks to see if the data was actually sent and whether it arrived at the destination or not. It also checks for the validity of the data. However, the correct operating procedure is not checked in system-level testing.

PCI bus protocol testing is used to determine if the modules in a system operate within the rules of the protocol. Besides verifying the data, protocol testing verifies that the agents are PCI compliant. PCI protocol tests should include the basic functional testing that is outlined in the PCI test scenarios. However, these test scenarios do not test every bus situation. There are several other conditions that have high probability of occurring and are recommended for implementation.

Some of the recommended exercises that should be implemented are:

Target termination sequences: Sequences where the target issues a termination when the master inserts wait states before, during, and after the termination.

Parity checking: Sequences where the incorrect address and data parity are generated to check the ability of the DUT to report errors.

Protocol checker: Checks to see if all the operating rules are obeyed.

> *"The PCI protocol is very complex, and as a result, it is difficult to test every possible combination of transactions."*

Developing an extensive testbench will help you determine the correctness of the design before continuing on to the implementation stage of the design cycle. After implementation, the testbench can be used for backannotated timing simulation.

PCI Timing Verification

Timing verification occurs after the design has been implemented. There are two stages to timing verification: static timing analysis and back annotated timing simulation.

Static timing analysis is used to determine if the design meets all the PCI timing specifications such as setup/hold time and clock-to-out timing. Static timing analyzers are provided by FPGA vendors and will determine if 100% of the design met the timing specifications. You have the responsibility of specifying these timing parameters to the tool and determining what parts of the design should be attached to the each parameter. If the design does not meet timing, the static timing analyzer can be used to probe the design and determine where the design is failing. Based on this investigation, you have the option to re-implement the design using a tighter set of timing specifications or redesign parts of the design to contain less levels of logic.

After the design passes the static timing analysis, it is beneficial to run the physical netlist, with timing information, through the functional testbench. This will allow you to determine if, in fact, the design meets timing using actual physical delays (not simply the unit delays that are applied during functional simulation). Once you have verified the design, it is ready to be downloaded to a device and used in a physical system.

Conclusion

Using a pre-verified Xilinx PCI LogiCORE can save you a lot of time and effort. These cores have been proven in hundreds of designs. However, you must still test your completed design to guarantee full PCI compliance. ◆