Functional Verification of USB Mass Storage

Xiaobin Chu, Tiejun Lun, Yu Zong

Abstract—**This paper presents a functional verification of USB2.0 Card Reader, which includes verification environment, functional coverage model design and course of debug. This system not only finds bugs in the DUT, but also verifies the compliance between hosts and device. The methods of the verification and coverage model design facilitate the verification of USB Mass Storage project which need to be accelerated into market. The system of verification has advantage of being portable to other USB Mass Storage devices.**

Index Terms—**functional verification, USB mass storage**

I. Introduction

With the development of IC manufacture process and enhancement of its design complexity, the verification requirement to IC is much higher. The methods of pure directed testcase verification is not adapted to the verification of complex SOC. Modern verification includes assertion-based verification, functional coverage, constrained-random testing, coverage-driven verification, dynamic-formal verification and more. Each method has particular feature in verification flow. In the practical project, maybe more than one verification methods with different features are employed in the project to reduce the time of verification.

This paper describes the verification environment for the USB2.0 Card Reader, discusses the strongpoint of the architecture and the details of design coverage model which is central engine in the verification flow, and offers a method to debug USB Mass Storage. The conclusion at the end of the paper, discusses the general verification methods for USB Mass Storage.

Manuscript received October 16, 2007

Xiaobin Chu is with Beijing Microelectronics Technology Institute, CO 100076 China (e-mail:chxibin@163.com).

Tiejun Lun was with Xi'an Microelectronics Technology Institute, he is now with Beijing Microelectronics Technology Institute, CO 100076 China (email:lu_tie_jun@163.com).

Yu Zong was with LanZhou University, he is now with Beijing Microelectronics Technology Institute, CO 100076 China (e-mail:zongyv@sina.com).

II. About the DUT (Design under test)

A mass-storage device can provide access to data for just any purpose. Every time you load an application or save a file on a PC, you are using a mass-storage device. In a USB mass storage device, the hardware or firmware must perform the following functions:

(1).Detect and respond to generic USB requests and other events on the bus.

(2).Detect and respond to USB mass storage requests for information or actions from the device.

(3).Detect and respond to SCSI command received in USB transfers. These industry-standard commands read and write blocks of data in the storage media, request status information, and control device operation.

Our DUT is a USB mass storage Device (USB2.0 6-in-1 Card Reader IC), it integrates the component of SIE, 8051 CPU core, DMA, MSCI, ECC, etc. The SIE and 8051 CPU core are third part IP cores.

III. Verification Environment

Figure 1 shows architecture of the verification environment. The verification environment is composed of three levels. The top level is the stimulus generator which generates the random testcases and the directed testcases. The second level is the functional level containing the driver which translates the generator test to USB format, the USB protocol monitor which detects the USB bus, MMC/SD card or other storage model and functional coverage model. The lowest level is signal level which contains the host PHY model, device PHY model and DUT. The host PHY and device PHY are simulate circuits in real ship. The two PHY models of the environment imitating the function of simulate circuits can reduce the gap between the real working environment and verification environment. Functional coverage, derived from the explicit functional specification of the design, is widely acknowledged as a central technique for measuring the thoroughness of verification. According to the condition of our project which contains IP cores in the DUT, we don’t know the details of internal signal in the IP cores, also the DUT is so complex that we don’t have enough time to learn the specification of DUT to design the functional coverage model, it is impossible that we collect the functional coverage from the DUT. Another method of defining functional coverage model is derived from the stimulus, because the stimulus set is equivalence to the functional set of DUT. If we test enough possible cases from the host to the device, and the coverage collected from the stimulus arrives at the objective we expect, we can be confident to our DUT. The verification environment would be to verify DUT to make it compliant with different operation system like WIN-XP, WIN-2000, WIN-98. We will describe the compliance in the section of Functional Coverage Model Design.

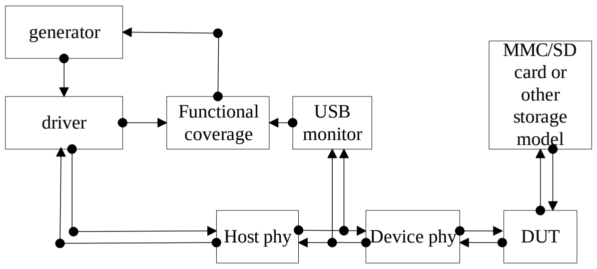

Figure 1.Verification Environment

IV. Functional Coverage Model Design

Simulation-based verification is the principle means of verification well accepted in the industry due to the tractability and usability of simulation progress. Simulation-based verification involves several steps, such as test generation and response evaluation, but coverage models are central to all steps in the simulation process. Functional coverage serves not only to reflect the quality of testing, but also to steer the verification resources towards areas of insufficient testing, and to provide a measurable indicator of the progress of the verification. The definition of coverage tasks is derived from manual interpretation of the specification. To design functional coverage model for USB mass storage device, we interpret the following specifications:
(1).USB2.0 specification
(2).USB Mass Storage Class Bulk-Only Transport
(3).Information technology-SCSI Block Commands-2 (SBC-2)
(4).Information technology-SCSI Primary Commands-3 (SPC-3)
(5).Information technology-Multimedia Commands-4 (MMC-4)

To enable communication, a mass storage device should implement the minimum USB command and SCSI command, the minimum USB command:
Get Descriptor
Set Address
Set Configuration
Clear Feature
The minimum SCSI command:
Inquiry
Read Capacity (10)
Read (10)
Request Sense
Test Unit Ready
Write (10)
To guarantee the device working normally, we should add other commands that host sends these possibly in some cases.

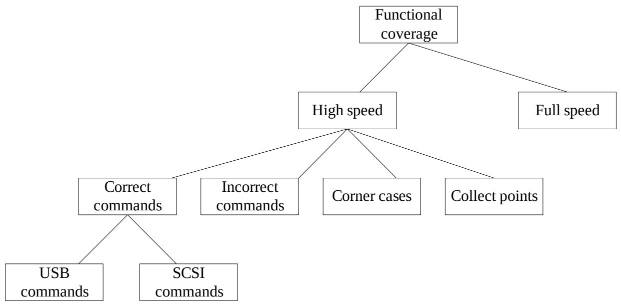

Figure 2.Functional Coverage Model

Figure 2 shows the architecture of hierarchical functional coverage model. Device should support two speed modes that high speed and low speed. The node of high speed branching four lines which are “correct commands”, “incorrect commands”, “corner cases” and “collect points” is the same with the node of full speed. The “correct commands” contain the USB standard commands and SCSI commands. The “incorrect commands” contain wrong commands that don’t comply with the specification like commands that don’t have valid CBW or meaningful CBW which defined in USB Mass Storage Class Bulk-Only Transport specification. The “incorrect commands” also contain the correct commands that the device don’t support. The “corner cases” are the thirteen cases that defined in the specification of USB Mass Storage Class Bulk-Only Transport spelling out how the host and device should behave after the host sends a command in each of thirteen cases. Above three functional coverage models are derived from specification manually. The “collect points” which are not derived from specification are used to verify the compliance between host and device. In practice, the compliance is much important as the device would be working in different PC hosts with different system operation. The fast way designing the compliant functional coverage point is to capture the signal from the different real PC and device via USB2.0 protocol analyzer. Though investment of USB2.0 protocol analyzer is hug, it can reduce the time of verification and guarantee the function of DUT.

Attribute	descriptor	Value
behave	operation	Clear Feature, Get Configuration Get Descriptor Get Interface Get Status Set Address Set Configuration
initiator	Who is initiator	USB host
response	result	ACK NAK STALL NYET DATA (according to the command)

Table 1: Attributes of USB command model

status	command
Default status	Get Descriptor Set Address
Address status	Get Descriptor Set Address Clear Feature Get Configuration Get Status Set Configuration
Configured status	Clear Feature Get Configuration Get Descriptor Get Interface Get Status Set Configuration

Table 2: Restriction for USB command

Table 1 shows an example of the attributes and their values for USB command functional coverage model of correct model. Table 2 shows an example of restriction for USB command. The crucial issue for the functional coverage metrics is to find the right granularity. If the metrics are too details, they will almost define testcases and so it is equivalent to manual writing of testcases.

How do we collect the coverage from the verification environment? From the attributes of USB command model, we can define the verification task that a pair of <command, response >, where the command is any the possible USB commands and the SCSI commands, where the response is one of the possible responses (ACK, NAK, STALL, NYET) for the USB commands and (00h, 01h, 02h in bCSWStatus of Command Status Wrapper) for the SCSI command. Functional coverage model detects the “driver” which sends USB commands or SCSI commands, and “USB monitor” detects the response from the USB bus. Functional coverage model collects the information about both of command and response. We can aggregate the holes from commands and responses to check which attributes are not covered, then analyze the holes and modify the “generator” to cover the holes.

Based on the feature of the functional coverage, we design the random test to verify the “correct commands” and “incorrect commands” and the directed test to verify the “corner cases” and “collect point”.

V. Debug and Coverage Analysis

A mass storage device must have one IN endpoint and one OUT endpoint in addition to endpoint zero. Endpoint zero which is the default endpoint used for control transfer is bidirectional. We start to verify the DUT from endpoint zero of high speed which transfers the USB command and the response data if needed. We take the random test method to cover the functional coverage model. Figure 3 shows the coverage curve via random test. At the beginning of the random test, the curve ascends at a very fast rate. As the verification of processes, the rate decreases and the curve nearly keep horizontal level. As the coverage of tests has been measured, the next step in the verification process is to analyze the coverage data to find out the uncovered functional points. For example, it is much more information to report that the “NYET” response never occurred. The USB2.0 specification describes that if the endpoint instead responses to the OUT/DATA transaction with a NYET handshake, this means that the endpoint accepted the data not have room for another wMaxPacketSize data payload. So host should sends many wMaxPacketSize data to let the device respond with “NYET”.

Coverage isn’t improved using random test long time. Through analyzing the coverage, we find that the “corner cases” and “collect points” are not covered. The reason is that using the random test is difficult to trigger the device into the states which is defined in the thirteen cases and the “collect point” of functional coverage is captured from real PC and device, the fixed order of commands is not easily covered via random test. So we take the directed test for the “corner cases” and “collect points” of functional coverage model. Figure 4 shows the directed test improving the functional coverage to nearly 99% after the random test. The figures demonstrate that the combination of random test and directed test applied to the verification of USB mass storage is feasible.

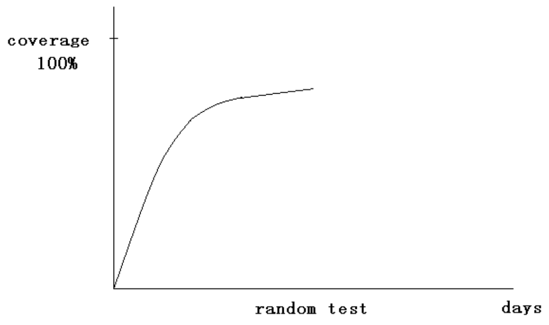

Figure 3.random test

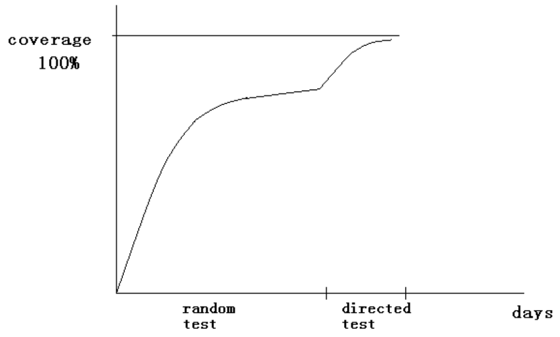

Figure 4.random test and directed test

VI. CONCLUSION

In this paper, we have presented an approach for the functional verification of USB2.0 card reader including the verification environment, the design of functional coverage model, debug and coverage analysis. Not only be the approach of verification applicable to USB2.0 card reader, but also fit for other USB mass storage.

REFERENCES

[1] Universal Serial Bus Specification Revision 2.0, USB 2.0 Transceiver Macrocell InterfaceSpecification, Universal Serial Bus Mass Storage Specification http://www.usb.org
[2] Information technology-SCSI Primary Commands -2 (SPC-2), Information technology-SCSI Block Commands-2 (SBC-2), Information technology Multimedia Commands -4 (MMC-4),http://www.t10.org
[3] A. Piziali. Functional Verification Coverage Measurement and Analysis. Kluwei Academic Publisher. 2004.
[4] Srikanth Vijayaraghavan, Meyyappan Ramanathan. A Practical Guide for SystemVerilog Assertions. Springer Ltd Publisher. 2005
[5] O Lachish, E.Marus, S.Ur, A.Ziv. Hole Analysis of Coverage Data, 39th DAC
[6] B.Beizer. Software Testing Techniques, Second Edition. Van Nostrand Reinhold. 1990
[7] J.Bergeron. Writing Testbenches: Functional Verification of HDL Models. Kluwer Academic Publishers. 2000. ISBN: 0-7923-7766-4
[8] Hans van der Schoot, Janick Bergeron, Transaction-level Functional Coverage in SystemVerilog. The proceeding of Design&Verification Conference DVC on 2006.
[9] Systemverilog Assertion Homepage (2003). http: //www.eda.org/sv-ac/
[10] Property Specification Language: Reference Manual. Version 1.1. Accellera June 2004.
[11] Oded Lachish, Eitan Marcus, Shmuel Ur, Avi Ziv. Hole Analysis for Functional Coverage Data. DAC 2002.