International Journal of Computer Applications Technology and Research
Volume 3– Issue 5, 292 - 295, 2014, ISSN: 2319–8656

Functional Coverage for Low Power DDR2 Memory Controller in UVM

Nishanthi G
Dept. of Electronics and Communication
BNM Institute of Technology
Bangalore, India

Yasha Jyothi M Shirur
Dept. of Electronics and Communication
BNM Institute of Technology
Bangalore, India

Ramudu B
Graphene Semiconductors India Pvt Ltd
Bangalore, India

Abstract: It is a well-known fact today that verification consumes approximately 70% of the product cycle and it is one of the main hindrance in developing a complex design. The advanced CAD tools reduces the design time but still the verification time is increasing with the design complexity. Writing directed tests for every feature of a complex design is a mind-numbing task. To alleviate this, constrained random verification is done. To check if all the design specifications are covered by these several random test cases a metric called Functional Coverage is needed. This metric gauges the progress of the verification and indicates if the destination is reached. This paper presents the development of functional coverage model for Low Power Double Data Rate 2 Memory Controller [LPDDR2 MC] in Universal Verification Methodology [UVM]. In this design there are 33 AXI v1.0 compliant masters which can write/read to/from a single memory. The LPDDR2 MC and the LPDDR2 memory model is JESD209-2F [JEDEC-standard] compliant. Hence the challenge is in identifying all the functional coverage points as per the specifications of this Design under Verification [DUV]. In this paper coverage models for AXI master interface and memory interface are implemented. 100% functional coverage was achieved when all the test cases were fired. Even after adding extra test cases functional coverage remained constant. The coverage models are reusable and thereby reduces verification time.

Keywords: Functional coverage; AXI; UVM; LPDDR2 MC; DDR2 memory model; DUV

1. INTRODUCTION

Functional verification plays a key role in a product cycle. It checks if the design intent is retained in it its implementation. For complex designs there will be huge number of specifications. So writing individual test cases for all the design features is a complicated task. Only way to ease this is through constrained random stimulus generation so that thousands of tests can be generated automatically in the simulation environment. To know which functionality is exercised by these random test cases numerous waveforms should be analyzed after each simulation. This is again a wearisome task. Hence, functional coverage is needed to determine what functionality was implemented in the test case without the need of visual examination of the waveforms.

Figure 1 shows the flow of functional coverage based verification.

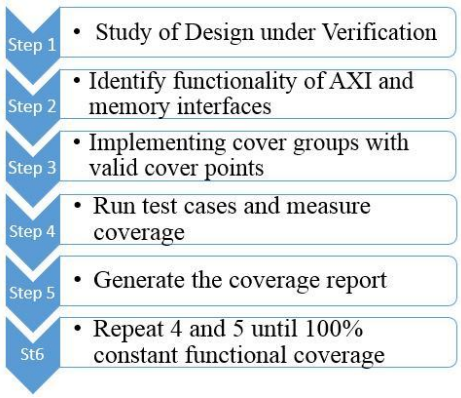

Figure 1. Flow of functional coverage of LPDDR2 MC

The cover group mechanism is coded in System Verilog and the verification methodology followed is UVM which is standardized by Accelera.

The UVM environment consists of different verification components. A verification component is a ready-to-use, configurable verification environment for a full system, design sub module or an interface protocol [3]. It can be customized as per the design behavior which makes it reusable. Coverage model is one of such verification component.

The paper is organized as follows. Section 2 describes the DUV. Section 3 describes the coverage model in UVM and the identified cover points.

2. DESIGN UNDER VERIFICATION

Figure 2 shows the simplified block diagram of LPDDR2 memory controller, it is used to drive LPDDR2 SDRAM. LPDDR2 devices use a double data rate architecture on the command/address bus to reduce the number of input pins in the system [2]. This supports 33 AXI compliant. The chosen memory model follows JEDEC standard. Hence, the memory controller can support LPDDR2 memory devices operating in a frequency range of 100MHz to 533MHz and of capacity 64 Megabits 8 Gigabits comprising 8 memory banks [2].

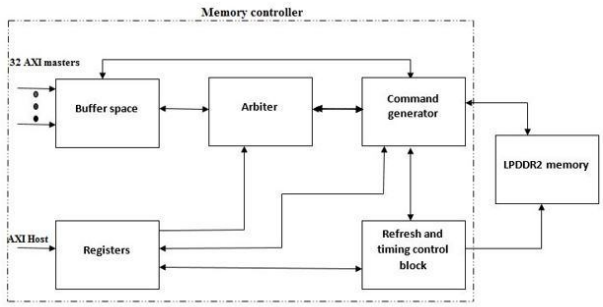

Figure.2 Simplified block diagram of LPDDR2 MC

The design flow is as follows. First the AXI host master will configure the register block. After the host configuration is complete, initialization of the memory takes place through refresh and timing control block.

Second the masters writing/reading to/from the memory will take place.

2.1 Functional coverage at interfaces

Figure 3 shows the interfaces with reference to the top module of the DUV with AXI interface and memory interface signals where the functional coverage is measured. The functionalities of the AXI protocol and the memory interface signals were identified to create a coverage model.

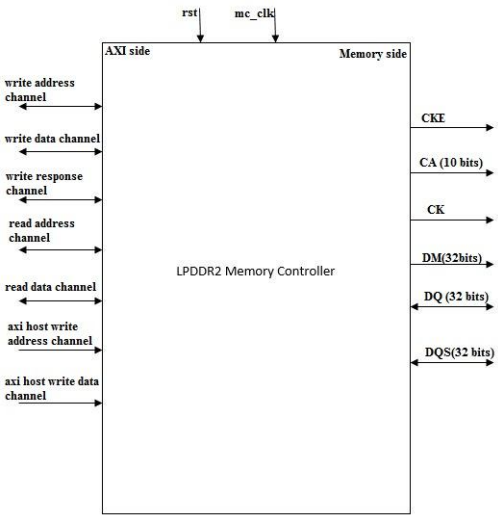

Figure.3 Interfaces of LPDDR2 MC

On the AXI side, one master has 5 channels through which the data transactions occur. The write/read address channels will carry address and control information to/from slave. The read data channel will carry both data and read response from the slave to the master. The write response channel will carry the write response for write transactions from slave to master [1]. The slave here is the memory controller.

On the memory side the CKE is the clock enable pin which will trigger the memory's clock. CA is the command/address bus which consists of bank, row, and column address information. CK is the clock signal to the memory which will be triggered from the memory controller block. DM (32 bits) is the data mask signal which indicates the valid data to and from the memory. DQS (32 bits) is the data strobe signal which indicates on which line the data is available to and from the memory.

3. COVERAGE MODEL IN UVM

3.1 UVM environment

Figure 4 shows the UVM environment diagram where the coverage collector is one of the verification component. This coverage model which is implemented in this paper gets inputs from both the AXI and memory monitor.

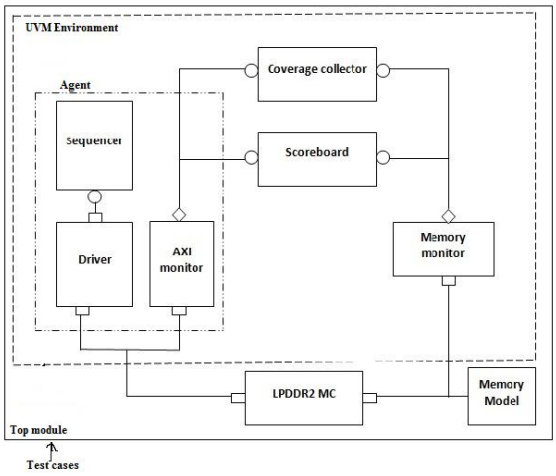

Figure.4 coverage collector in UVM environment

The environment consists of AXI agents, score board, coverage collector. The AXI universal verification component [UVC] also called as agent consists of sequencer, driver and monitor. Driver will drive the DUV with the transactions. The sequencer will control the transactions provided to the driver. Both memory and AXI monitors will monitor and collect the transactions whenever observed on the DUV interfaces. There are 33 AXI agents with monitors in each and one memory monitor and one Score board for data integrity check.

3.2 Coverage collector

Monitors will observe the transactions on all the interfaces and it will write into the coverage collector. The communication which happens in the UVM environment is transaction level modelling [TLM]. Coverage collector consists of different cover groups like AXI master cover group, AXI host cover group and Memory cover group. These cover groups has identified cover points on both the interfaces with respect to the design specifications. Each cover point will have coverage bins like valid, illegal and ignore bins. These bins are the check points which will be updated as per the received transactions. When enabled, each test case will generate a coverage database. After running all the test cases, different test cases databases are merged into a single data base. From this the overall coverage report is generated.

3.2.1 Communication between monitor and coverage collector

The TLM port connection between monitor and coverage collector is indicated by a diamond symbol at the monitor and a circle at the coverage collector. The diamond symbol is the analysis port and the circle is the port. One analysis port can be connected to any number of ports. The analysis ports will call a write function at the end of every transactions. This

write function is implemented in the coverage collector. The write function will sample the cover groups

3.2.2 AXI master cover group

In this cover group the cover points related to the AXI channel signals are identified for one master and instantiated for 32 times. For one master identified cover points are 20. The table 1 shown below shows the cover points of write address channel for this cover group.

Consider the cover point Burst type. It is a 2 bit signal which indicates the type of burst transaction. 00 for fixed, 01 for incremental, 10 for wrapping and 11 is reserved. According to the design specification the burst type followed is incremental. Therefore the valid bin is 01, invalid bin is 00 and 10, ignore bin is 11. If this cover point was not identified and if the test case did not exist for this functionality the design response will not be known. It means that there can be a bug in the design which was not verified even though the other test cases passed.

Table 1. AXI read/write cover group

Cover points in AXI read/write cover group	Description
Write address id	This checks if the number of transaction if from 1 to 16 is covered
Burst length	This checks if the data transfers within a burst covers all the length from 0 to 15
Burst size	This checks the number of data bytes within a beat covers all the size from 1 to 128 bytes
Burst type	This checks if the burst type wrapping, fixed and incremental is covered

3.2.2.1 AXI host interface cover group

In this cover group the cover points related to the AXI Host master signals are identified. The host master configures all the registers in the design to a default value. This will be the 33rd master. Identified cover points are 37. The table 2 shown below shows some of the cover points.

Table 2. AXI host cover group

Cover points in AXI host cover group	Description
Power down	This checks if the power down condition is covered
Read and write latency	This checks if all the read and write latency values are covered
Device density	This checks if all the device density from 64Mb to 8Gb is covered
Memory burst length	This checks if all the burst length 4,8,16 is covered

3.2.3 Memory interface cover group

In this cover group the cover points related to the memory interface are identified. Here the identified cover points are huge in number as there are 8 banks in the memory and there are different commands. This cover group checks for different memory commands like activate, pre-charge, write, read etc. are covered or not. This also covers the transition coverage like if a write or read operation occurred in the same memory bank back to back. This cover point is important because there might be a case where the write occurred say to bank 1 but a read did not occur to the same bank which will again indicate a presence of bug. Hence all the identified cover points are critical. Identified cover points are 248. Few of them are shown in the table 3 below.

Table 2. Memory cover group

Cover points in memory cover group	Description
Write after write in bank 0	This checks the transition coverage if there is back to back write in bank 0
Pre-charge all banks	This checks if all the banks are pre-charged (closed) condition occurred
Activate bank 1	This checks if the condition to activate bank 1 is occurred
Refresh all banks	This checks for the condition if all banks are refreshed

4. RESULTS AND ANALYSIS

For every test case coverage report was generated, 1st the illegal bins were analyzed to check if any test cases are out of the design specifications. Once the illegal bins are cleared all the valid bins are checked. Valid bins indicate the valid and critical functionality. If any valid bin was not hit, the test case was run with random seeds or run with missing test cases. The coverage report was merged once all the test cases were fired.

Figure 5 shows the screen shot of the functional coverage of the AXI write address channel. The cover point burst type is not covered as the valid bin is not hit indicating a missing test case. The overall AXI coverage is 54.54% because some of the other cover points were not covered.

Covergroup instance:\/test/#ublk#0#181/cg			100.00%	54.54%	54.54%
Coverpoints / Bins	At Least	Hits	Goal	Coverage	% of Goal
Coverpoint: write_burst_length (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
bin valid	1	3		Covered	--
Coverpoint: write_burst_size (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
illegal_bin illegal	1	1		Occurred	--
bin valid	1	2		Covered	--
Coverpoint: write_burst_type (covered 0 of 1 bins) (missing 1 of 1 bins)			100.00%	0.00%	0.00%
illegal_bin illegal	1	1		Occurred	--
ignore_bin ignore	1	2		Occurred	--
bin valid	1	0		ZERO	--
Coverpoint: write_addr_id (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
bin valid	1	3		Covered	--

Figure 5 Burst type uncovered

After running the missing test case the burst type valid bin was hit and also the other cover points were covered. Figure 6 shows the screen shot of the same.

Covergroup instance:\/test/#ublk#0#181/cg			100.00%	100.00%	100.00%
Coverpoints / Bins	**At Least**	**Hits**	**Goal**	**Coverage**	**% of Goal**
Coverpoint: write_burst_length (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
bin valid	1	30		Covered	--
Coverpoint: write_burst_size (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
illegal_bin illegal	1	11		Occurred	--
bin valid	1	19		Covered	--
Coverpoint: write_burst_type (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
illegal_bin illegal	1	21		Occurred	--
ignore_bin ignore	1	4		Occurred	--
bin valid	1	5		Covered	--
Coverpoint: write_addr_id (covered 1 of 1 bins) (missing 0 of 1 bins)			100.00%	100.00%	100.00%
bin valid	1	30		Covered	--

Figure 6 Burst type covered

5. ACKNOWLEDGMENTS

Our thanks to the BNMIT and Graphene management who have contributed towards this paper.

REFERENCES

[1] AMBA AXI protocol V1.0.

[2] JESD209-2F specifications.

[3] UVM user guide 1.1.

[4] System Verilog LRM IEEE 1800-2005.

[5] Yang Guo, Wanxia Qu, Tun Li, Sikun Li ,National University of Defense Technology, P. R. China, IEEE International conference, 2007. Coverage Driven Test Generation Framework for RTL Functional Verification.

[6] Yingpan Wu,Lixin Yu, Lidong Lan, Haiyang Zhou , Beijing Microelectronics Technology Institute, China,IEEE International symposium, 2008. A coverage driven constraint random based functional verification method of memory controller.

[7] Akhilesh Kumar and Chandan Kumar,Department of E&C Engineering, NIT Jamshedpur, Jharkhand, India. IJAET 2011. Functional Coverage Analysis of Ovm Based Verification of H.264 Cavld Slice Header Decoder.

[8] Young-Nam Yun, Jae-Beom Kim, Nam-Do Kim, Infrastruct Design Center, Samsung Electron. Co. Ltd., South Korea, soc design conference, 2011. Beyond UVM for practical SoC verification.

[9] www.verificationacademy.com.