International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 5 Issue 4 ǁ Apr. 2017 ǁ PP.08-15

Research on UVM Verification Platform Based on AXI4 Protocol Intellectual Property

Yiqingming[1], Lizhaobin[2], Shimin[3]

Collage Of Information Science And Technology, Jinan Univeristy, China, 510632

ABSTRACT: As the improvement of the status of verification and the verification technology in IC and SoC, this paper designs a verification platform based on Universal Verification Methodology (UVM) and finish the verification of a IP core based on AXI4 protocol ,which Combines the technology of Assert.This platform adopts a VIP RAM IP core based on AXI4 protocol as the Design Under Test(DUT),which takes advantage of the inheritance and reuse of UVM class to build the Universal Verification Component(UVC) and the tree of UVM.It designs transaction of UVM platform as the constrained random excitation,which completes the system verification toward the DUT.At last,the Simulation shows that the UVM platform finishes the verification of the DUT,which reaches the functional coverage on 100%.In addition, instead of the traditional simple testbench, the UVM verification platform in this paper shows the better reusability and portability by the use of the reusability of the UVC,which can be used to verify the other DUT on AXI4 protocol.

Keywords: UVM;Verification methodology;Verification platform;AXI4 protocol;VIP

I. VERIFICATION TECHNOLOGY OVERVIEX

With the application of Intellectual Property(IP) and Intellectual Property Verification(VIP),the efficiency of design and verification on IC and Soc industry has been greatly improved.In addition, verfication occupys 70% or more during the entire design cycle,which actually plays an important role in the project of Soc.[1]

The verification technology can be divided into the verification language and verification platform.The former one is the implementation of the verification technology while the latter one is the framework and theory of verification technology.As to the verification language,in addition to Verilog,E,Vera,SystemC and SystemVerilog(SV) are the common language on verificaion.SV is the most important and common language among the modern verification technology,which has the feature of Object Oriented Programming(OOP) including encapsulation,inheritance, polymorphism and some special feature on constraint and functional coverage.[2、4]As to the verification platform,it mainly reflected in the development of the verification methodology on each EDA vendor,as shown in Figure1.[5、6]During the third period,the verification methodology based on SV has been developed.In 2011,the company Accellera put forward the Universal Verification Methodology(UVM),which quickly get the approvement of major EDA vendor(the company Sysnopsys,Mentor,Cadence)UVM has become the mainstream of the methodology in the industry, and represents the latest development direction of verification methodology since it inherits the advantages of VMM and OVM and overcomes the disadvantages of them.

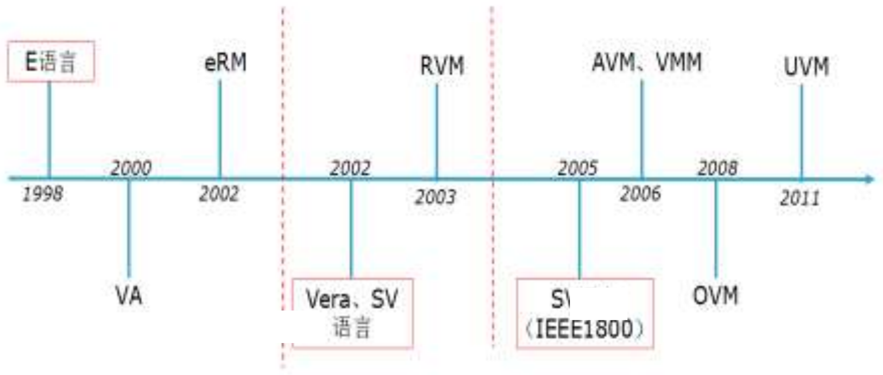


Figure1 Verification Language and Verification Methodology

The UVM based on Systemverilog and AXI4 protocol are studied.A UVM platform based on AXI4 protocol IP core is proposed, which uses XILINX IP core according to AXI4 protocol as the DUT. Finally, the system verification based on UVM is realized that the UVM platform achieves the meaning of standardization, reusability, portability.

II. THE VERIFICATION METHODOLOGY OF UVM

UVM is the next generation verification methodology proposed by Accellera, which inherits the verification methodology of OVM.In 2010, UVM1.0EA(EA:early adoption) was proposed while the official version UVM1.0 was proposed in 2011. In March 2013，UVM1.1d was proposed, which is the version used in this paper.In short, UVM is the new generation of verification methodology because it inherits the advantages of the OVM like the mechanism of factory, phase and so on while it adopts the advantages of the VMM like the solution of RAL, callback and so on.

Actually, UVM is the libary of Systemverilog, chich includes basis class, public class and macro class.Universal Verification Component(UVC) can be created by using the class of UVM and the mechanism of factory to register and administrate the component.

The UVM verification platform uses a layered architecture to improve reusability and portability. A typical UVM verification platform consists of the following UVC:DUT, driver, sequencer, monitor, reference model, scoreboard.The driver and the monitor package in the form of agent while all of the UVC package into the verification environment env.[7]

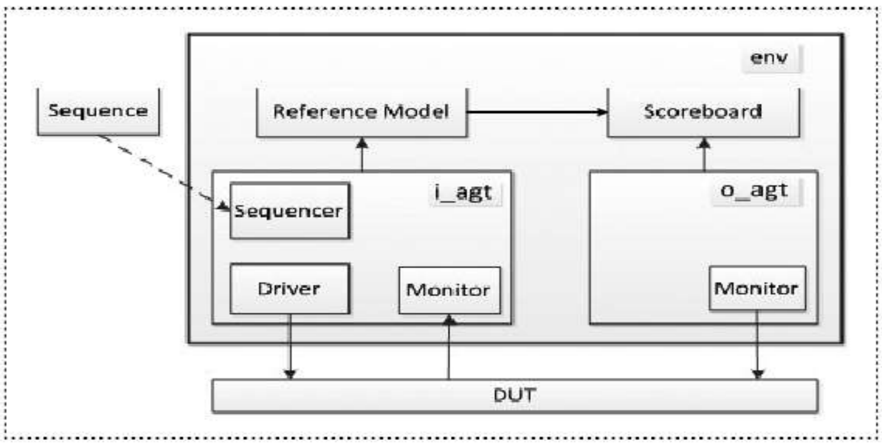


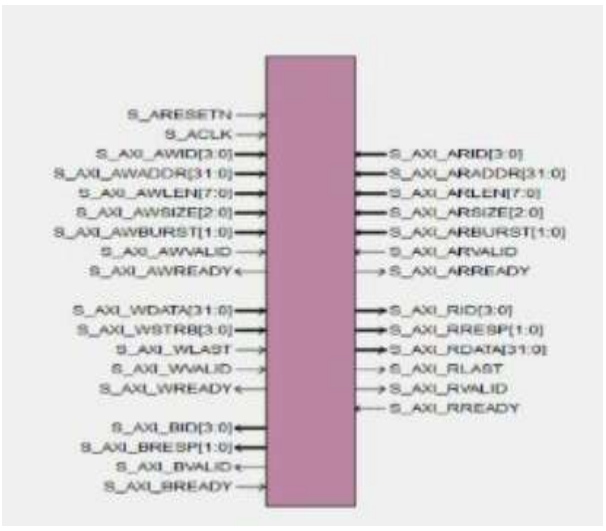
Figure2 UVM Platform

When the verification platform runs, it run automatically according to the mechanism of phase. In particular, a random excitation with constraints is generated from the sequence while it is transmitted to the sequencer and sent to the driver. The driver encourages the excitation to DUT. Then, monitoring, and acquisition is done through the monitor, then it convert the output signal back into the data stream (transaction) forms, sending the transaction to the reference model and the scoreboard (respectively by using the i_agt and o_agt monitor but the allocation mode of configuration is different: ACTIVE/PASSIVE), two output data from reference model and DUT is compared in the scoreboard, which gives the final verification result after comparing.[8]

III. DESIGN OF VERIFICATION PLATFORM

3.1 DUT

The DUT module of AXI4 protocol IP core is generated by software of the XILINX ISE,which is shown in Figure 3.It a AXI4 RAM module verified with read and write function according to the AXI4 protocol, which is generated by the technology of IP core generation from XILINX ISE.AXI4 protocol is the new generation Bus protocol proposed by the company of ARM and XILINX. Because of its superior performance, it has been widely used in Engineering



By using the standardized IP core(AXI4 RAM) as DUT, this paper focuses on the design of the UVC and the overall optimization of the UVM verification platform. After the optimization of the verification platform, the system can be extended to the other DUT on AXI4 protocol.

3.2 UVM Tree

UVM provides the engineers a large number of standard library functions and base classes.Through the inheritance of the class, the UVC can be built in standard. Specially, sequence derived from uvm_object, sequencer derived from uvm_sequencer, driver derived from uvm_driver, monitor derived from uvm_monitor, scoreboard derived from uvm_scoreboard. reference model derived from uvm_component, i_agt and o_agt derived from uvm_agent, env derived from uvm_env, testcase derived from uvm_test. In addition, tb_top it the top module of the test. Thus, the UVM tree has been built, as shown in Figure 4.

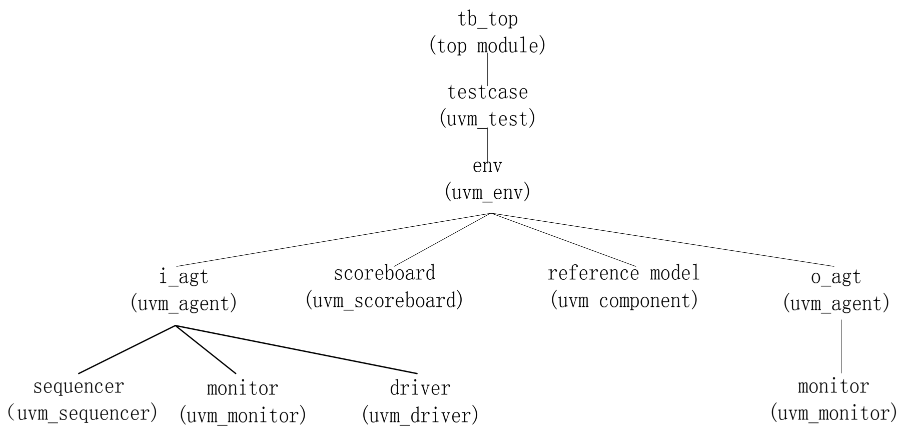


Figure4 UVM tree

Building a UVM tree is the best way to instantiate and manage all of the UVC, while each of the UVC is the node of the tree and tb_top is the root of the UVM tree.

3.3 Design of UVC

UVC is built and designed by using the library of UVM, but the main body is defined according to the function of each UVC. Among them, driver and sequence is the key of the design because the relevance to the AXI4 protocol contrary to the monitor, sequencer, scoreboard and so on, which just need to design according to the specification of UVM.

The function of the driver is to drive the transaction packet to the DUT according to the AXI4 protocol.In this paper, the main body of the driver consists of two tasks,which is the task reset_signals and tast get_and_drive.The former one is mainly responsible for reseting the signal of AXI4 while the later one is mainly responsible for the acquisition of excitation transformation, and drive to the DUT. Taking advantage of the verification language Systemverilog like the sentence of wait, @posedge,repeat and so on, the task get_and_drive can be realized

Actually, the creating and driving of excitation can be simply done by the driver instead of the sequence and sequencer in a simple UVM verification platform. However, the meaning of separate the function of driver into driver, sequence,sequencer is shown in the situation of multi-excitation needed, in which main_phase do not need to be rewrited in each of the UVC.

As to the sequence, three classes of sequence are created in this design and are named of axi4_short_rw、axi4_rw_base、axi4_sequence, which respectively correspond to three classed of testcase.The sequence of axi4_short_rw is responsible for creating the excitation of 16 address and data write-read information, and the sequence of axi4_rw_base is responsible for creating the excitation of 1024 address and data write-read information.Finally, the sequence of axi4_sequence is responsible for creating the excitation of 500 random write and 500 random read address and data information.

As to the other UVC, the sequencer is actually the passageway of the sequence,which is designed to start the sequence and transfer the excitation to the driver. It is a standard design to different UVM platform. The monitor is mainly consist of two task for read and write to finish the acquisition and transformation of the excitation. Specially, the monitor set two FIFO to save the information to wait for the DUT and reference model and eventually sent them to the scoreboard.The i_agt and the o_agt are just the package of driver and monitor. The reference model is a software simulation of the DUT. Thus, it is designed as a ram module to realize the transmission of read and wtite according to the AXI4 protocol. 3 ports are instantiated in the scoreboard:

```
uvm_blocking_get_port #(axi4_trans)   exp_port;
uvm_blocking_get_port #(axi4_trans)   act_read_port;
uvm_blocking_get_port #(axi4_trans)   act_write_port;
```

They are responsible for the receiving of the information from the FIFO of monitor. Then, the scoreboard compared the information automatically by keeping watch on the port on awlen, arlen, and the data of AXI4 protocol to realize the verification of DUT. In addition, SystemVerilog Asserttion(SVA) has been designed in virtual interface(vif) to verify the timing problems aimed at the timing of AXI4.

3.4 Design of Testcase and Verification Environment

Aimed at the characteristic of transmission on AXI4, 3 sequence have been designed, which respectively correspond to three classed of testcase,as shown in Table 1.

Table 1 The corresponding relation between testcase and sequence

testcase name	sequence name
axi_sht_rw	axi4_short_rw
axi_rw_test	axi4_rw_base
axi_test	axi4_sequence

The testcase in this design are derived from uvm_test:

```
class axi_sht_rw extends axi4_base_test ;……
class axi4_base_test extends uvm_test;……
class axi_rw_test extends axi4_base_test ;……
```

By using the mechanism of run_phase in each testcase, the instantiation of sequences are finished. When the platform need different excitations, juse need to change the body of runphase in the testcase instead of changing others UVC.The key core of the runphase is shown:

```
initial begin
run_test("axi_rw_test");
end
```

The verification environment env and testcase is the top of the UVM tree.The function of env is to put all UVC together and help the extension of the longitudinal reusability of the verification platform.[10] Finally, the top module tb_top instantiate the DUT module and the env(including all of the UVC).

3.5 Module communication

The communication of each of the UVC in the platform is based on the transaction, which is a stream of excitation according to the AXI4 protocol. The transaction in UVM verification platform is derived from uvm_sequence_item, in which it defines the specification of signal on AXI4 protocol and gets random excition with constraint.A transaction means a package about AXI4. When sequence generates the excitation, it sends as a transaction package to the sequencer, sequencer starts the sequence and sends the transaction to the follow-up UVC, the diagram of transmission is shown in Figure 5. Besides the driver, monitor and DUT, others UVC communicate by using the transaction.The driver\monitor and the DUT communicate with each other by using virtual interface(vif), which is actually a module programming with Verilog HDL to connect the driver, monitor and the DUT through ‘a set of calble’.

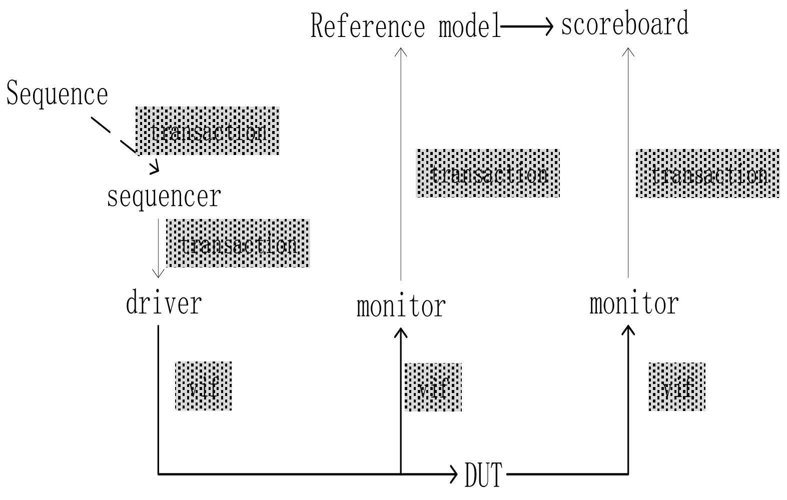
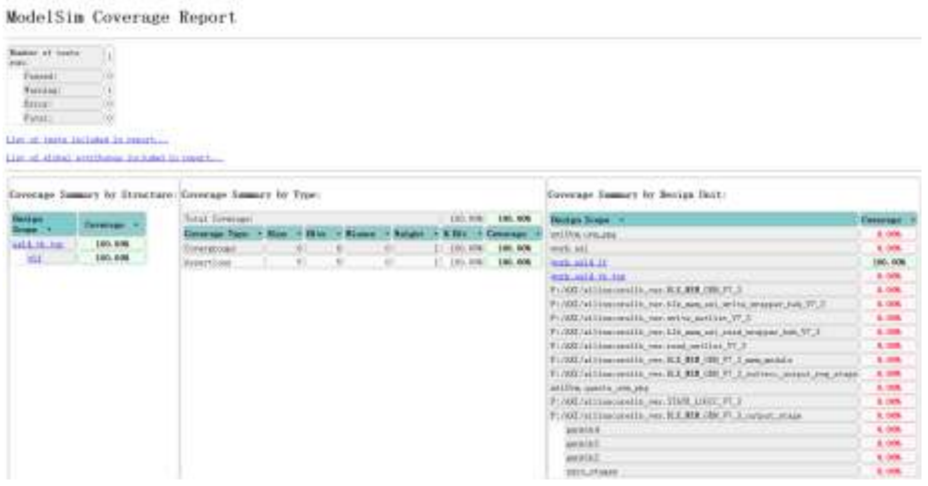


Figure5 transmission of each UVC

IV. SIMULATION RESULTS

Commonly, the VCS and Modelsim is used to be the mainstream simulation software for UVM.The Modelsim provides the support for windows since the vision of 10.1. The Modelsim 10.4se is adopted in this paper because it comes with the library of UVM1.1c and UVM1.1d. In addition, since AXI4 RAM module has been adopted as the DUT, the library of component on XILINX must be added, like xilinx_lib. At this time, the simulation environment of software has been built.

Start the modelsim and choose the testcase axi_rw_test in the tb_top, the UVM verification platform will be ready to run. When runing the simulation, the name of the testcase and other information will be viewed in the windows of modelsim. By using the order of showing the report of function coverage , the report of functional coverage is shown as Figure6, in which the functional coverage is 100% while all the assertion hit the timing.



ModelSim Coverage Report

Coverage Summary by Structure:

Design Scope	Coverage
[illegible]	100.00%
vif	100.00%

Coverage Summary by Type:

Coverage Type	Bins	Hits	Misses	Weight	% Hit	Coverage
Total Coverage:					100.00%	100.00%
Covergroups	[illegible]	[illegible]	[illegible]	1	100.00%	100.00%
Assertions	[illegible]	[illegible]	[illegible]	1	100.00%	100.00%

Coverage Summary by Design Unit:

Design Scope	Coverage
[illegible]	0.00%
[illegible]	0.00%
[illegible]	100.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%
[illegible]	0.00%

Figure6 the view of simulating sortware modelsim

By using the timing waveform analyzer to insert the waveform of vif, the timing waveform is shown as Figure 7. The sequence axi_rw_base is called by the testcase axi_rw_test, which generate 1024 addresses and datas to finish the transmission of write and read.Because it cause a large number of information,so the Figure 7 is just a zoom of the waveform while the loss of the data packet and the correct timing of the read and write timing are performed by automatic scoreboard and assertion.

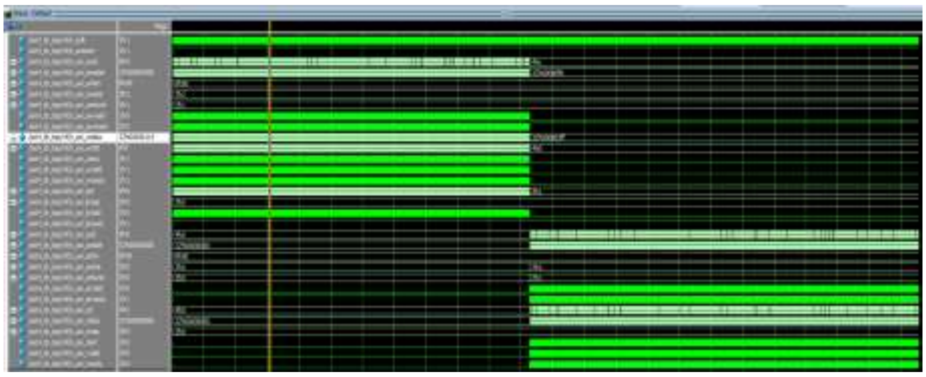
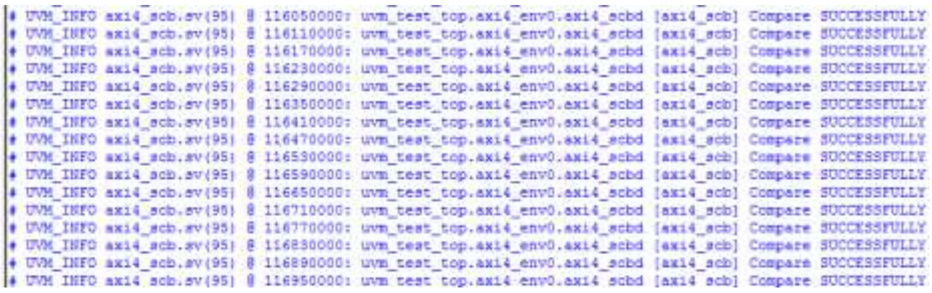


Figure7 timing of axi_rw_test

The scoreboard is responsible for the comparing the output of the reference model and the DUT and automatically comparing whether the packet is lost, which will also be printed out. If the comparison is successful, then print out ‘Compare SUCCESSFULLY’, which means that the function of DUT is the same as the reference model and both of them accord with the AXI4 protocol. If comparison fail, then compare fail and ‘Compare FAIL’ will be printed out to means the design of DUT fail to accord with the AXI4 protocol.In this paper, the result of scoreboard is shown as Figure 10. As the Figure 8 shown, the DUT accord with the AXI4 protocol and so the verification of this DUT is successful.



```
# UVM_INFO axi4_scb.sv(95) @ 116050000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116110000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116170000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116230000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116290000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116350000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116410000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116470000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116530000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116590000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116650000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116710000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116770000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116830000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116890000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
# UVM_INFO axi4_scb.sv(95) @ 116950000: uvm_test_top.axi4_env0.axi4_scbd [axi4_scb] Compare SUCCESSFULLY
```

Figure8 scoreboard output

V. Summary

A UVM verification platform which can meet the requirements of UVM specification is built in this paper. It can realize the reuse of UVC, and has the advantages of portability and reusability. By using the function of automatically comparing and the report of functional coverage, the DUT of AXI4 RAM has been verified to be accorded with the transmission of AXI4 protocol, whith greatly saves the force of comparing the waveform artificially and improve the efficiency of verification. In this paper, a UVM verification platform based on AXI4 protocol is proposed to verify the DUT about AXI4, with the aim of focusing on the building and optimization of the UVM platform and extending the platform to verify the other DUT about AXI4 protocol.

REFERENCE

[1]. SaLahK.A UVM-based smart functional verification platform:Concept,pros,cons,and opportunities[C]Design&Test Symposium(IDT)，9th International,Dead Sea:IEEE,2014,pp.94-99

[2]. 张强.UVM 实战[M]北京：机械工业出版社,2014

[3]. 刘星江.一种基于 Verilog 的验证平台搭建及应用[J]. 信息安全与通信保密.2013

[4]. Mark Zwolinski（夏宇闻译）.SystemVerilog 数字系统设计[M]北京：电子工业出版社,2011

[5]. Rosenberg S,Meade K A.A practical guide to adopting the universal verification

methodology(UVM)[M/OL]San Jose:Design Systems Inc,2010
[6]. 谢峥,王腾,雍珊珊等.一种基于 UVM 面向 RISCCPU 的可重用功能验证平台[J]北京大学学报,2014
[7]. Accellera.Universal Verification Methodology（UVM） 1.1 User’s Guide[M].2011
[8]. 曹阳,胡月黎.基于 UVM 的存储控制器功能验证[J].计算机测量与控制.2015(3)
[9]. 易清明、曾杰麟、石敏.支持流水传输的 AXI4 master 转换接口设计[J].计算机工程,2016(4)
[10]. 王国军,景为平.基于 UVM 的验证平台设计研究[J]微电子学与计算机,2016
[11]. 张怡琳.基于 UVM 可重用验证平台的研究[D]西安电子科技大学,2015
[12]. 何冬明.RFID 数字控制器的 UVM 功能覆盖率验证[J]中国集成电路,2015
[13]. 吕毓达.基于 UVM 的可重用 SoC 功能验证环境[J]半导体技术,2015
[14]. 范宇杰.基于 UVM 验证方法学的 USIM 验证 IP 的设计与应用[D]西安电子科技大学,2015
[15]. 吴星星.基于 UVM 的 SPI 接口 IP 核的验证平台设计[D]安徽大学,2016
[16]. 熊涛.基于 UVM 验证方法学的纵向可重用研究[J]微电子学与计算机,2016