Memory Controller using Functional Coverage Driven Functional Verification using SV and UVM

Sunkara Dedeepya
(M.Tech 2nd year):
Digital Electronics Communication System,
Sri Padmavathi Mahila Vishva Vidyalam,Tirupathi

R. Divya
(Asst.Prof):
Digital Electronics Communication System, SPMVV,
Tirupathi India,

***Abstract*—Memory controller is a universal core, which is a part of SOC (System On Chip).It supports variety of memory devices such as SRAM, SDRAM, FLASH etc.The chip size decreases, physical testing becomes much complex to overcome. This paper focuses on HDL verilog is coded by taking certain desired characteristics and modeled into a memory controller design. The testing of this design, functional coverage using ASIC verification languages are SV and UVM. The memory controller design includes two interfaces wishbone and memory interface. The wishbone interface provides synchronization for connecting processor to memory. The memory interface provides synchronization for connecting memory to memory devices. The wishbone interface provides 100% functional coverage using UVM by covering more than 30 test cases [1]. The total coverage using system verilog provides 50%.The UVM testbench provides minimum effort by the test writers and faster execution[1]. The design supports variety of features which are analyzed using multimaster memory. The simulation results with respect to number of codelines in SV and UVM are analyzed in graphical form. The simulation on UVM count analysis with respect to time is analyzed. The power calculation with respect to time is calculated for SRAM, SCSD. The result of implementation of two testbenches in SV and UVM has tracked all set of values corresponding to design. The UVM testbench provide better placement in routing architecture than SV testbench. This UVM design becomes easy for ASIC implementation**

Keywords—UVM; SV; Verilog; HDL; ASIC; SOC; Functional coverage;

I. INTRODUCTION

The memory controller manages the data going to the processor to memory devices and memory devices to processor. It provides the support to processor to interface with different memories such as DDR4, SSDRAM, SCSD, SRAM, SDRAM, and FLASH etc each are uniquely programmable and has a synchronous interface [14].The processor ask for the memory access, that is read and write the memory resides on single binary data The memories such as SRAM is fast because the instruction fetch and data access happens in both one clock cycle[18].Other hand the memory SDRAM memory controller is an IC (Integrated Circuit) thatis called as a northbridge.This north bridge that is connected to the processor directly through the front side bus (FSB) so that it is responsible for every task .The processor speed increases on the motherboard. This memory is then segregated into equally sized independent section called banks and speed up access. The four banks need two bits of information[13]. The SDRAM [14] can be operated in two modes, Active mode [14] and Precharge mode [4]. The first mode keeps the column open and applies the bank address. The current column is accessed by write back the data into exact.For synchronous chip select devices, the TMS (Timing Mode Set Registers)[3]register is responsible to hold the internal counters, memory bus clock responsible to clock all the internal counters. It also maintains the time required for accessing various parameters.

II. METHODOLOGY

A. *Design of RTL*

In VLSI design technology different chips are being design and fabricate an IC (Integrated Circuit). Application Specific Integrated Circuits[7] is the process in which the for the “RTL(Register Transfer Level)” design languages used are Verilog[16] those are hardware description languages which are being used to design. The “RTL” design is a design process of exacting desired characteristics, models into a digital circuit in terms of flow of digital data [18]. According to Moore’s law [7]is mostly related to scaling of the MOSFET (Metal Oxide Semi Conductor Field effect transistor).This made possible to design high density IC chips. At each level of the verification is necessary generally verification covers two important things which are necessary

➢ Whatever the features being specified for the design that will be designed in future.

➢ What we design for the customer’s need that features that is designed being specified.

B. *Testing of Hardware design*

Errors are possible during the design which is an unforeseen behavior. The errors are corrected at the earlier stages of verification of the design. The verification methodology system verilog is indeed more advantage than verilog. System verilog is suitable mostly for the verification of OVM (Open Verification Methodology) [7].System verilog has its own assertions which are used to check the behavior of the design. The language is completely comprises OOPS [14] concept which make the designers to code easily. The verification of the RTL design in verilog [8] the testbench model is designed. The testbench verification is user designed friendly model that is done in both system verilog and universal verification methodology. The testbench model shown in figure1

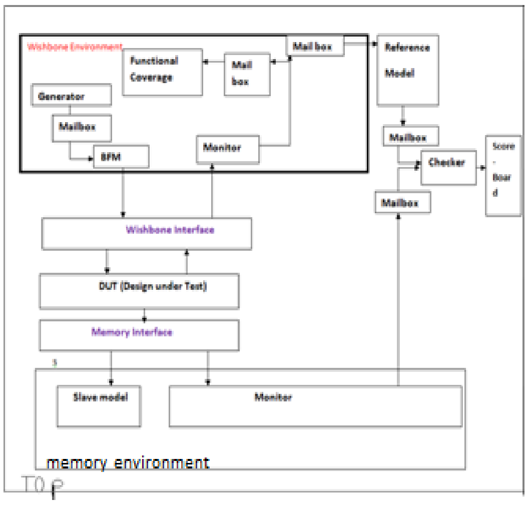


Figure.1.Testbench design of the memory controller in SV

The system verilog is a superset of constraints assertions, “OOPs” language [12]. The testbench of memory controller shown in figure1 narrates the way of coding. The coding is done in leaf level layout manner. The figure is then separated into two parts that is master and slave. The master and slave can be modified depending upon number of master and slaves. In this figure the slave are memory devices [13] .
The blocks of testbench are Generator, Monitor, functional coverage, BFM(Bus Functional Model),Wishbone interface, Memory interface, memory environment,monitor,functional coverage, wishbone environment, Reference model, Checker and Scoreboard. The discussion of each block is described.
Generator: It generates the transaction read transmission write _read signal, clock reset, and acknowledgment, refresh, and address signals these entire signals are transmitted to a configuration class. These transactions are then transferred to BFM.
BFM (Bus Functional Model): The BFM receives all the transaction from the generator and converts the transaction level into pin level design. These signals are then routed to the monitor.
Monitor: It advices the valid signal on the interface.Monitors are of two types: Passive and Active. The Passive monitor [5] dent to drive any signal of the transaction. Active monitor [5] drivesthe signal of the design.
Functional Coverage:.The functional coverage is definedterm as the percentage of verification objectives that have meet. The system verilog provides a total coverage is of 50%.

Checker and Scoreboard: The checker performs the comparison of the transaction. The checker converts the wishbone transaction [3] into memory transmission where memory transaction is approximately to half of the wishbone transaction. This is oftencalled as unpacking which is the reverse operation of packing. The compared result of analysis is a consigned to scoreboard [1].The scoreboard splurged how much the percentage of codecoverage.

III. UVM ARCHITECTURE

The uvm architecture is build on the basis of system verilog. The uvm architecture testbench contain class library that is used for building up reusable verification components. The UVM (universal Verification Methodology)[2] class library which internally the following contains automation, factory restoration, stimulus creation, standard etc which leads to the success of the its standard. The flow chart describes the goal of the uvm testbench figure2

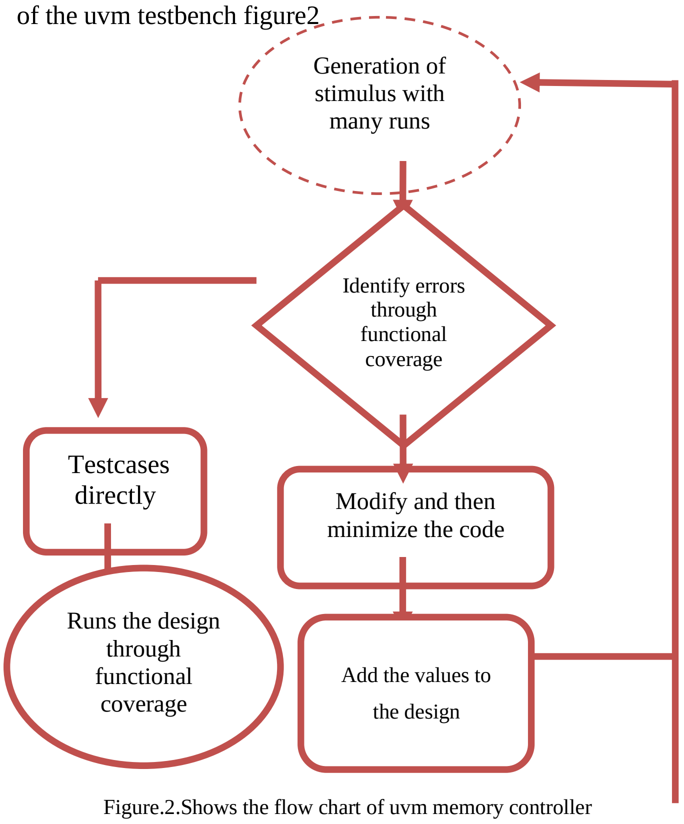

Generation of stimulus with many runs
Identify errors through functional coverage
Testcases directly
Modify and then minimize the code
Runs the design through functional coverage
Add the values to the design

Figure.2.Shows the flow chart of uvm memory controller

The verification has emerged into a complex set of design The UVM provides the path for the verification and can access the product that is designed without any restrictions [2]The verification methodology came into standardization laid on top of the OVM(Open Verification Methodology)[17] and later AVM(Advanced Verification Methodology), URM (Universal Reuse Methodology),*e*(eRM)are combined. Finally 300 members from the Accellera Company have worked to standardize the verification.The standard UVM has a set of API (Application Programming Interface) and base class library (BCL) which is used to enhance flexible design in different models, components can be reused. The API and BCL are based on IEEE standard1800™for the system verilog [2].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 11, November-2019

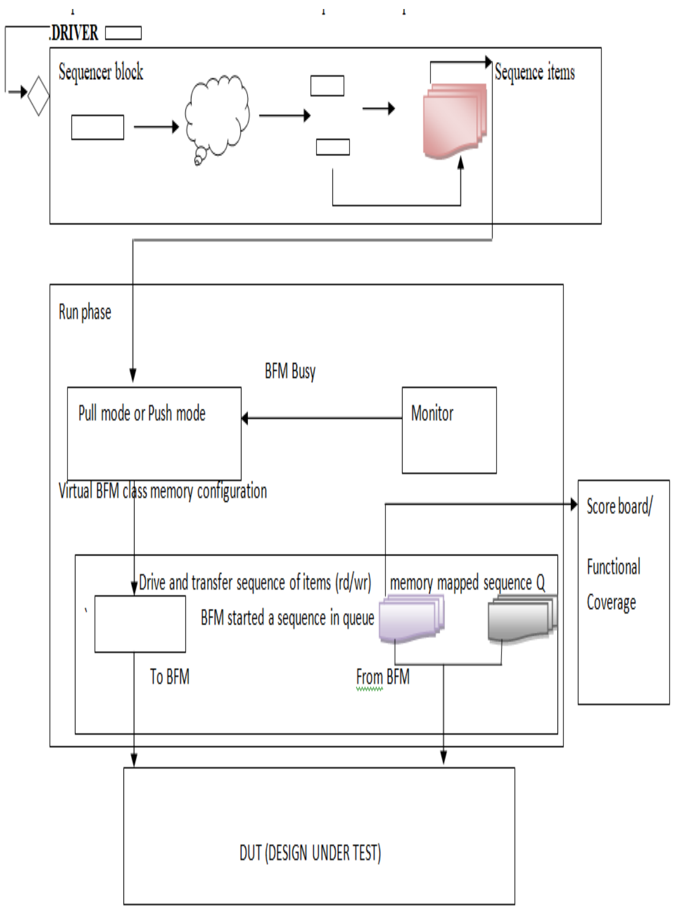


Figure.3.shows the testbench architecture of the UVM

UVM AGENT: The uvm agent generates verification logic for the design. The initiation and connection of the design is done by the test case developer. The standard agent in uvm describes it encapsulates sequencer, driver, monitor etc.

UVM DRIVER: TheUVM driver generates the sequence of the items. The uvm driver takes items till the items are done. The uvm stretches deeply each transaction and analyse these and convert them into pin level stimulus.

UVM SEQUENCER: The uvm sequencer generates compare the transaction and transmit those transaction to the monitor.More indicatively the UVM sequencer obstructs the flow of the UVM sequence [17] of transaction sequence of transmission regimented by one more UVM sequences.

UVM MONITOR: The UVM monitor of the design and take over the information and there of transaction of a sequence of items that are get hold of from the rest of the UVM testbench and disseminated to the UVM monitor. The UVM monitor converting a pin level activity to transaction

A. Abbreviations and Acronyms

- SRAM (Static Random Access Memory):The SRAM is one of the model of semiconductor random access memory.SRAM acronym abbreviates Static Random Access Memory where hanks always contingent on the positive edge of the clock the data are transmitted.
- SDRAM (Synchronous Dynamic Random Access Memory) needs different invigorating that cycles are arranged the shortest refreshing clock cycle.
- FLASH is potency takes more time than actually write cycles.
- UVM term delineates Universal Verification Methodology. SV term interprets System Verilog
- OOPS term defines Object Oriented Programming language is concept based on object programming and methods.
- ASIC defines Application Specific Integrated Circuit
- RTL Register Transfer Level.
- HDL defines Hardware Description Language.

B. Equations

The memory controller writes the data the memory devices. The power is calculated on considering particular case of transaction. Consider the SRAM memory the processor is transmitting the data through wishbone interface to checker. The power calculation is done below equation1

Consider the data is transmitted from t1 to t2.

$$p(t) = \frac{1}{T}\int_{t1}^{t2} x(t)dt \qquad (1)$$

P(t)= total power transmitted by SRAM during write transaction

T=Total time taken during the data transmission

t1-t2=time duration by considering each division as 0.5units

$p(t)=6.0215W$

Using same equation 1 the power calculated for SCSD(synchronous chip select devices)

$p(t)=3.193W$

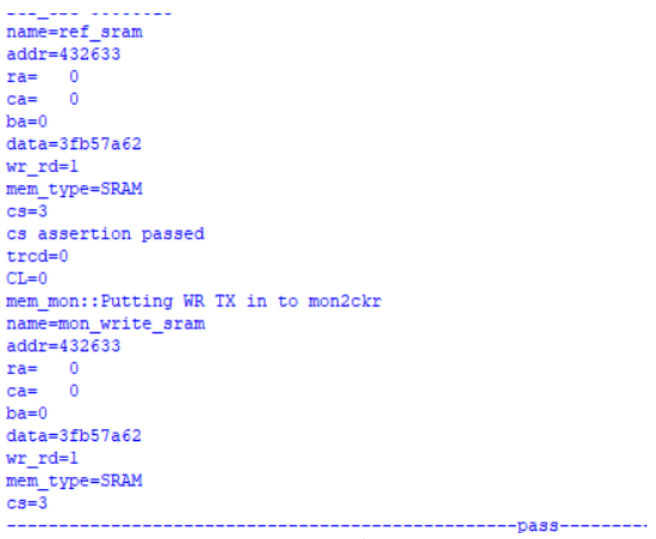
The same equation 1 the power calculated for SRAM for read transaction

$p(t)=38.848\mu W$

The same equation1 helps to calculate the power for SCSD during read transaction.

$p(t)=67.636\mu W$

In all the cases the read transaction takes less than write transaction



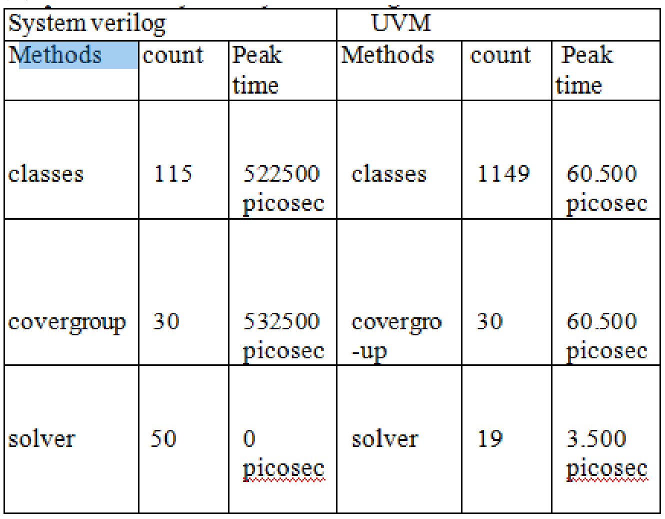
```
name=ref_sram
addr=432633
ra=    0
ca=    0
ba=0
data=3fb57a62
wr_rd=1
mem_type=SRAM
cs=3
cs assertion passed
trcd=0
CL=0
mem_mon::Putting WR TX in to mon2ckr
name=mon_write_sram
addr=432633
ra=    0
ca=    0
ba=0
data=3fb57a62
wr_rd=1
mem_type=SRAM
cs=3
-----------------------------------------------pass--------
```

Figure.10 shows the output simulation of SRAM during write transaction.

The figure.10 show the address and data of the wishbone. The chip is selected CS whatever data is there on that particular address data will be write on to memory. The ra is the row address and ca is the column address. SRAM is divided into word line and bit. During the write transmission bit is activated word line is activated and during read transmission the corresponding the word and bit line is activated. Then the difference is then sensed by a sense amplifier, which is basic differential amplifer.The allows the output quickly without charging.

IV. COMPARISON OF SV AND UVM

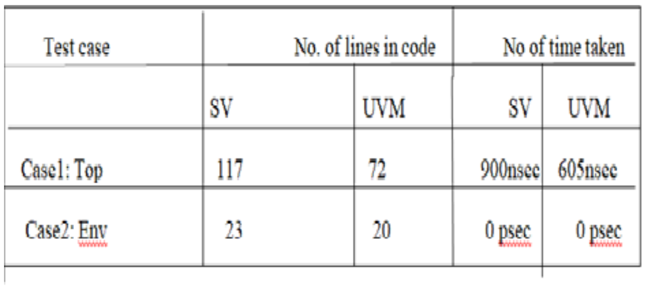
The verilog is the first starting language in mid late 1990.Unfortunaltely to end of the 1990 verilog has stuck to a problem to bring to an industry level standards .After the 1990a Co-design has broken this problem and start designing implementation of system verilog[7].The system verilog is having an ability to extend all the small group of codes which then being combined.System verilog which is an advanced version of IEEE standard Verilog 2005[7] that is useful for creating a real environment like structure called testbench environment for the Design Under Test(DUT). The verification analysis is done by using Mentor Graphics Questa sim tool 10.4e.The System Verilog module is used in writing the code. The verification environment that is coded completely which wraps top module and functional coverage. The module analogy is used static in behaviour so that memory is allocated during the compilation period. The allocated memory will be present throughout the simulation even if designer does not use the memory. The uvm methodology has originate from the system verilog which then combined with the open verification methodology (OVM)[5]. The first version of UVM language is released based on OVM.This methodology has class library for building advanced reusable verification component. The figure4 shows the comparative analysis of system verilog and UVM



System verilog			UVM		
Methods	count	Peak time	Methods	count	Peak time
classes	115	522500 picosec	classes	1149	60.500 picosec
covergroup	30	532500 picosec	covergro -up	30	60.500 picosec
solver	50	0 picosec	solver	19	3.500 picosec

Figure.4. shows the tabular comparison of system verilog versus UVM

The figure analyses that in uvm as the count increases in less time it has covered all the classes compared to sv. The covergroup in uvm has covered all the values in the bins in less time and provided 100% coverage [9]. The solver solves the bugs in test within less time. The uvm provides minimum effort to the human user.



Test case	No. of lines in code		No of time taken	
	SV	UVM	SV	UVM
Case1: Top	117	72	900nsec	605nsec
Case2: Env	23	20	0 psec	0 psec

Figure.5.shows the tabular comparison of system verilog and UVM

The figure analyses that to design the top in testbench the uvm takes less number of codelines and executes much faster. The environment is created on the top of the testbench of the design. The environment takes less time compared to the uvm.

The graphical analysis of this tabular form is shown below fig6.

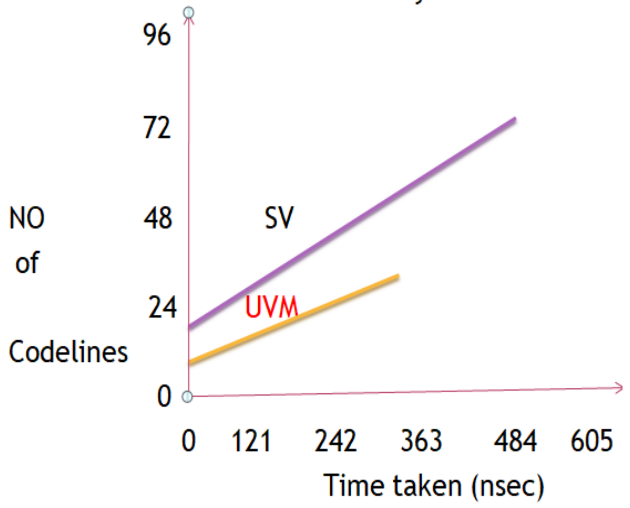
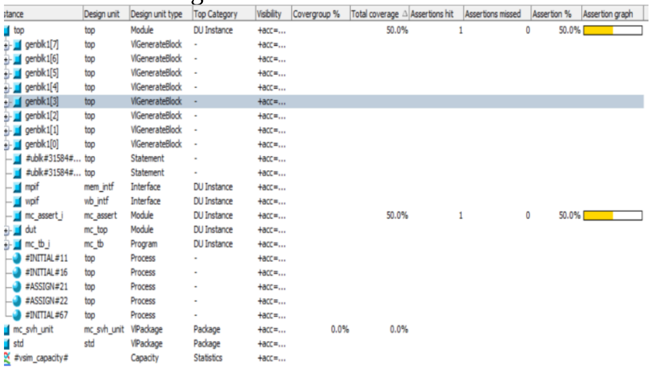


Figure.6.shows the graphical analysis of sv and uvm with respect to time (nsec)

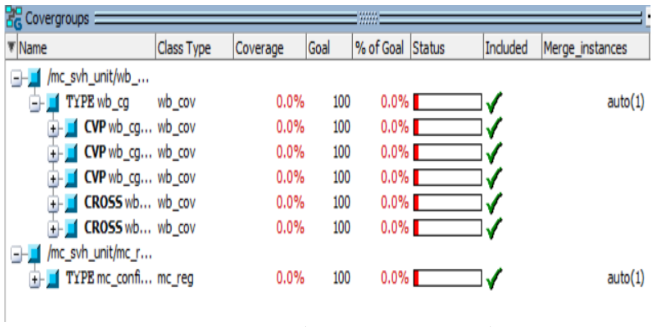
The figure.6.analyses that uvm has strong basic foundation.The figure.7.explains that the assertions are generated similarly to other design blocks and are active for the entire simulation. The simulator keeps an eye on what assertions have triggered, so you can gather functional coverage data on them. The system verilog provides a functional coverage of 50%.



stance	Design unit	Design unit type	Top Category	Visibility	Covergroup %	Total coverage	Assertions hit	Assertions missed	Assertion %	Assertion graph
top	top	Module	DU Instance	+acc=...		50.0%	1	0	50.0%	▨
genblk1[7]	top	VlGenerateBlock	-	+acc=...						
genblk1[6]	top	VlGenerateBlock	-	+acc=...						
genblk1[5]	top	VlGenerateBlock	-	+acc=...						
genblk1[4]	top	VlGenerateBlock	-	+acc=...						
genblk1[3]	top	VlGenerateBlock	-	+acc=...						
genblk1[2]	top	VlGenerateBlock	-	+acc=...						
genblk1[1]	top	VlGenerateBlock	-	+acc=...						
genblk1[0]	top	VlGenerateBlock	-	+acc=...						
#ublk#31584#...	top	Statement	-	+acc=...						
#ublk#31584#...	top	Statement	-	+acc=...						
mpif	mem_intf	Interface	DU Instance	+acc=...						
wpif	wb_intf	Interface	DU Instance	+acc=...						
mc_assert_i	mc_assert	Module	DU Instance	+acc=...		50.0%	1	0	50.0%	▨
dut	mc_top	Module	DU Instance	+acc=...						
mc_tb_i	mc_tb	Program	DU Instance	+acc=...						
#INITIAL#11	top	Process	-	+acc=...						
#INITIAL#16	top	Process	-	+acc=...						
#ASSIGN#21	top	Process	-	+acc=...						
#ASSIGN#22	top	Process	-	+acc=...						
#INITIAL#67	top	Process	-	+acc=...						
mc_svh_unit	mc_svh_unit	VlPackage	Package	+acc=...	0.0%	0.0%				
std	std	VlPackage	Package	+acc=...						
#vsim_capacity#		Capacity	Statistics	+acc=...						

Figure.7.shows the functional coverage and assertions of system verilog the uvm is having the concept of covergroup that will keep an eye on all the conditions that being observed during the simulation process.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 11, November-2019

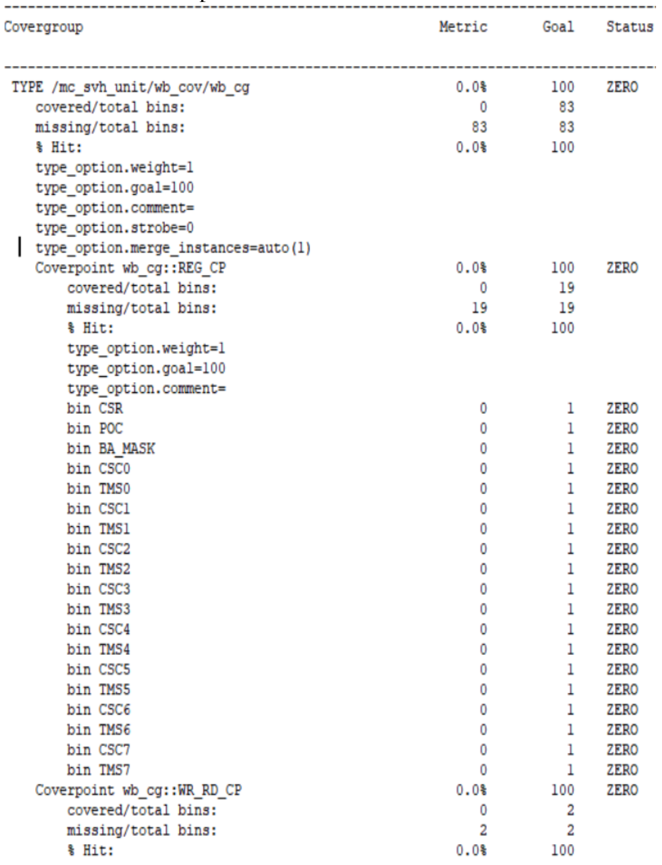


Covergroups

Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances
/mc_svh_unit/wb_...							
TYPE wb_cg	wb_cov	0.0%	100	0.0%	🟥	✓	auto(1)
CVP wb_cg...	wb_cov	0.0%	100	0.0%	🟥	✓	
CVP wb_cg...	wb_cov	0.0%	100	0.0%	🟥	✓	
CVP wb_cg...	wb_cov	0.0%	100	0.0%	🟥	✓	
CROSS wb...	wb_cov	0.0%	100	0.0%	🟥	✓	
CROSS wb...	wb_cov	0.0%	100	0.0%	🟥	✓	
/mc_svh_unit/mc_r...							
TYPE mc_confi...	mc_reg	0.0%	100	0.0%	🟥	✓	auto(1)

Figure.8.shows the functional coverage of uvm

The merge is used in this figure.8 to describe that taking all of the instances in the cover group and delivering the average across every coverpoint. The bin shown in figure 8 is declared to store a certain range of values. Fig.9 bins are defined by declaring a variable followed by {}.The crosscoverage shown in figure8 explains that it is defined in between the coverpoint.



```
-------------------------------------------------------------------------------
Covergroup                                         Metric      Goal    Status


-------------------------------------------------------------------------------
 TYPE /mc_svh_unit/wb_cov/wb_cg                     0.0%        100     ZERO
    covered/total bins:                                0         83
    missing/total bins:                               83         83
    % Hit:                                          0.0%        100
    type_option.weight=1
    type_option.goal=100
    type_option.comment=
    type_option.strobe=0
  | type_option.merge_instances=auto(1)
    Coverpoint wb_cg::REG_CP                        0.0%        100     ZERO
        covered/total bins:                            0         19
        missing/total bins:                           19         19
        % Hit:                                      0.0%        100
        type_option.weight=1
        type_option.goal=100
        type_option.comment=
        bin CSR                                        0          1     ZERO
        bin POC                                        0          1     ZERO
        bin BA_MASK                                    0          1     ZERO
        bin CSC0                                       0          1     ZERO
        bin TMS0                                       0          1     ZERO
        bin CSC1                                       0          1     ZERO
        bin TMS1                                       0          1     ZERO
        bin CSC2                                       0          1     ZERO
        bin TMS2                                       0          1     ZERO
        bin CSC3                                       0          1     ZERO
        bin TMS3                                       0          1     ZERO
        bin CSC4                                       0          1     ZERO
        bin TMS4                                       0          1     ZERO
        bin CSC5                                       0          1     ZERO
        bin TMS5                                       0          1     ZERO
        bin CSC6                                       0          1     ZERO
        bin TMS6                                       0          1     ZERO
        bin CSC7                                       0          1     ZERO
        bin TMS7                                       0          1     ZERO
    Coverpoint wb_cg::WR_RD_CP                      0.0%        100     ZERO
        covered/total bins:                            0          2
        missing/total bins:                            2          2
        % Hit:                                      0.0%        100
```

Figure.9. shows the metrics and goal of type wishbone coverage in uvm

The application of the design is not perfect then the covergroup will might receive extra information which leads to slow the process of the simulation and merging. The errors can be avoided by delivering the right choice while creating the cover group in the design. The right choice of covergroup designing initiate with cover group that is defined in a coverage class. The cover group includes features those are it includes a group of coverpoints.The clock pulse that runs in sync and sampling the coverage points. All the coverage points are linked with bin.. The bins are automatic or values can be user defined. Automatically a bin will be designed for every value of the variable of the coverpoint. In a coverpoint bins are declared with certain values that can be excluded from the coverage by declaring ignore bins.

V. RESULT AND DISCUSSION

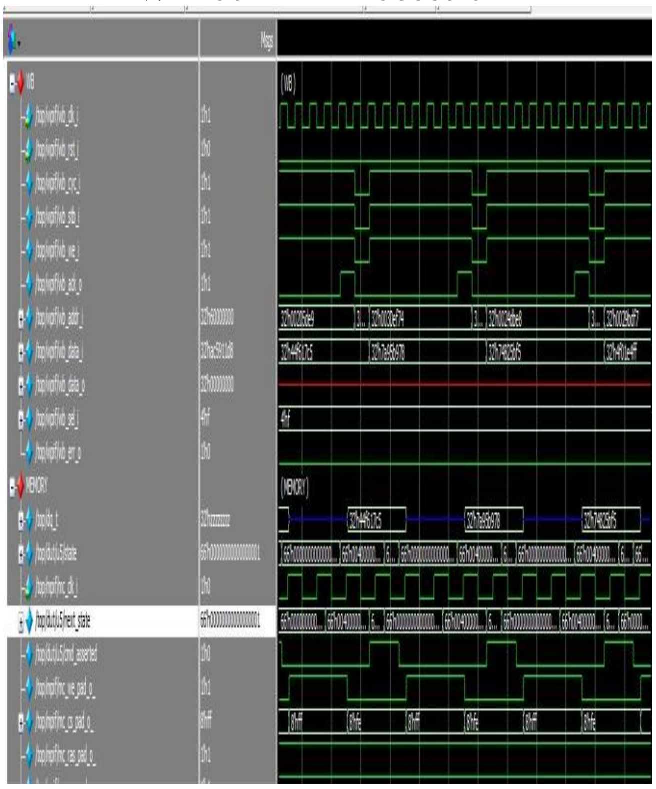


Figure.11. The SRAM write transaction

The SRAM acquires clock, reset, acknowledgment signal, data input, and chip select, address. The chip select is actively high then that particular chip is selected then clock cycle, strobe signal, acknowledgement signal, writes enable are actively high then the data is transmitted for that particular address.

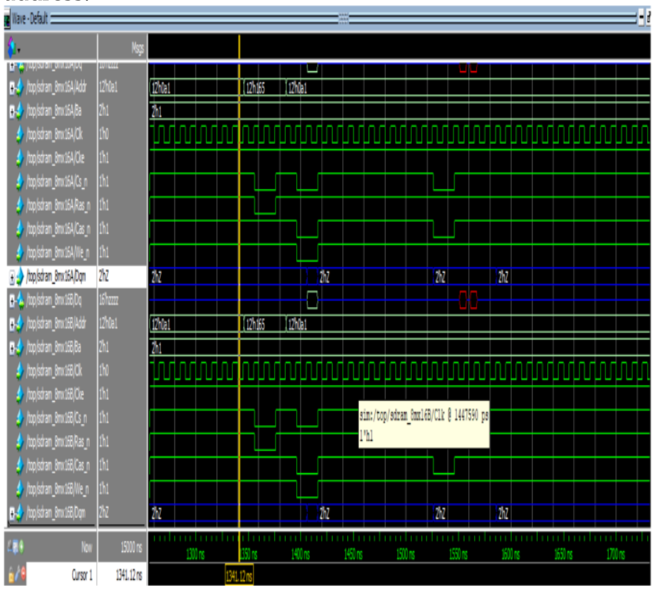


Figure.12. The SDRAM read transaction

The SDRAM particularly operates in two modes, Active mode and recharge mod. The Active mode reads the current column address and the Passive mode after a trdv duration of clock cycles for that particular column reads the particular data is transferred on to the memory device[14]. The row address is selected actively high, then the particular row is activated, then after columnis activated, then write enable is activated then the data writes on the memory[14].

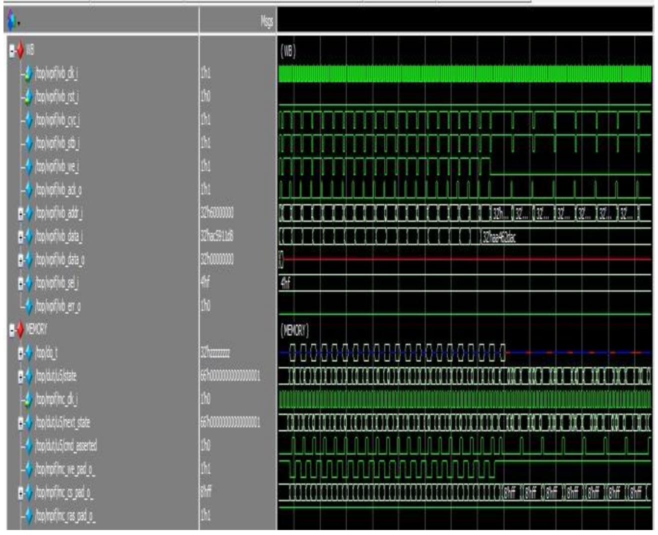

Msgs
WB
/top/wpif/wb_clk_i 1'h1
/top/wpif/wb_rst_i 1'h0
/top/wpif/wb_cyc_i 1'h1
/top/wpif/wb_stb_i 1'h1
/top/wpif/wb_we_i 1'h1
/top/wpif/wb_ack_o 1'h1
/top/wpif/wb_addr_i 32'h60000000
/top/wpif/wb_data_i 32'hac5911d8
/top/wpif/wb_data_o 32'h00000000
/top/wpif/wb_sel_i 4'hf
/top/wpif/wb_err_o 1'h0
MEMORY
/top/dq_t 32'hzzzzzzzz
/top/dut/u5/state 66'h00000000000000001
/top/mpif/mc_clk_i 1'h0
/top/dut/u5/next_state 66'h00000000000000001
/top/dut/u5/cmd_asserted 1'h0
/top/mpif/mc_we_pad_o_ 1'h1
/top/mpif/mc_cs_pad_o_ 8'hff
/top/mpif/mc_ras_pad_o_ 1'h1
(WB)
4'hf
(MEMORY)

Figure.13. The SRAM read transaction

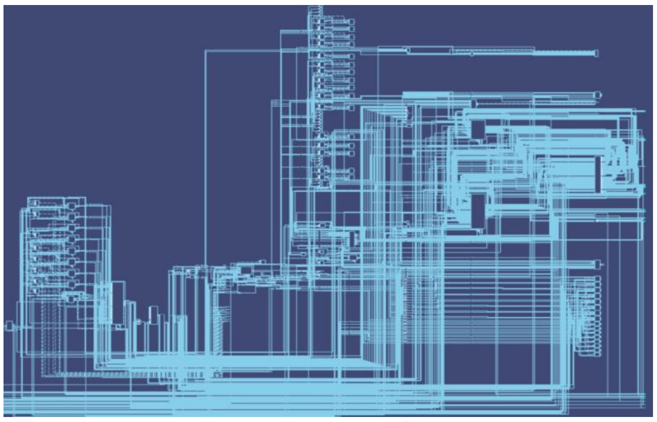


Figure.14.The schematic diagram of memory controller

ACKNOWLEDGMENT

I would like thank Sreenivas who trained me in coding.He is trainer .He did his master degree from IIT Madras in VLSI

REFERENCES

[1] .K.,Fawzy,H.,El-Ashry,S.,&Salah,K.May(2014).Anovel meory controlller architecture.In Electrical Enginnering/Electronics and Telecommunication and Information Technology(ECTI-CON),2014, 11th International Conference on (pp 1-4).IEEE
[2] IEEE standard for Universal Verification Methodology and language Reference manual IEEE std 1800.2™ - 2017
[3] Memory Controller Core using Rudolf Usselman,january21,2002
[4] Wishbone system on-chip(SOC) Interconnection of Architecture For portable Ip cores,Silicore,September7,2002
[5] Khaled Khalifa,Implementation and Verification of Generic Universal Memory controller based on UVM,978-1-4799-1999-4/15/
[6] Jain G. Bonnannoo ,H Gupta and A.Goyal,"Generic system verilog Universal Verification Methodology based reusable verification environment for efficient verification of image processing Ips/SOCs",International journal of VLSI design and communication systems (VLSICS),vol.3, no.6 (2012)
[7] C.spear,system verilog for verification,second edition:A Guide learning the testbench language,Features,2nd ed. Springer Publishing Company,Incorporated,2008
[8] Dinesh Katuri,S.Umadevi,Design and Verification of Memory Controller with Host Wishbone Interface,Springer Nature Singapore Pte,Ltd,2018,C.Labbe`et al.(eds.),Nano electronic Material and Devices,Lecture notesin Electrical enginnering466,//http:// doi.org/10.007/978-981-10-7191-1_19
[9] Ahmed EI- Yamany,Sameh,EI-Ashry Khaled salah,Coverage closure Efficient UVM based a Generic Verification Architecture For Flash memory Controller,2332-5674/16 © 2016 IEEE DOI 10.1109/MTV.2016.10,2016 17thInternational Workshop on Microprocessor and SOC
[10] Koushel agarwal,Vijay Kumar Magraiya,Dr.Anil kishore saxena ,verification verification and new design of Nand Flash memory Controller 978-0-7695-4958-3/13 © 2013 IEEE DOI 10.1109/CSNT.2013.163, International Conference on Communication systems and Network Technologies.
[11] Giupessa Scata ,Ashwin Padoor,Vladimir Milosevi,Accelerated SOC verification using UVM methodology for mixed low power signal design,Acellera systems,Initiative,texas Instrument design and verification conference Exhibition 2014.
[12] A Bhojak and T Prasad "A UVM based Methodology for processs verification"in DVCON,2015.
[13] Sharada Pattar,Mrs Rashmi S Bhaskar,verification of universal memory controller,(IJRET) e-ISSN:2395 -0056 volume 04 Issues :05|May -2017 p-ISSN:2395-0072
[14] AparajithaLenka,.shashibhushan,Testbenchdevelopmentand verification of memory controller,ISSN(print):2278-8948 volume -6 Issue-1_2,2017
[15] Questa ® SIM User's Manual Including Support for Questa SV/AFV software version 10.4e Mentor Graphics.
[16] IEEE standard for System Verilog and Unified Hardware Design ,Specifications and Verification Language Reference manualIEEE std 1800.2™ - 2017
[17] Nausha kotia,Mr.Prashant,D.Karandikar,Mr.Gardas Naresh kumar,UVM based Verfication based Environment for Performance Evaluation of DDR4 SDRAM Using Memory controller
[18] www.quora.com,mozhikunnathramdasexcellenceverification