## DAY 81 - 100 DAYS VERIFICATION CHALLENGE

**Topic: UVM Basics**

## DAY 81 CHALLENGE:

1. What is UVM? What is the need if UVM?
2. Why is UVM considered as a Methodology & not a Hardware Description Language?
3. What is the difference between SV & UVM?
4. Is UVM independent of System Verilog?
5. What are the disadvantages of UVM Methodology?
6. Draw the UVM testbech & explain each component with Syntax & the methods used:
   i. Sequences
   ii. Sequence
   iii. Driver
   iv. Monitor
   v. Agent
   vi. Env
   vii. Test
   viii. Scoreboard
   ix. Subscriber

Arti tyagi 2024

Date
Page

DAY : (81)

Topic : UVM BASICS

Sol^n ① what is uvm ? what is the need if uvm?

- uvm - Universal VERification Methodology. uvm is a methodology based on System Verilog language.
- its not a language on its own. it is standardized methodology that define several Base classes.
- uvm is a well defined class library which has reusable verification Components & object alredy available. using them we can build a verification enviorment with ease.
- uvm has 300+ Base classes & macros, whereas in uvm you can directly use them by extending from the already available base class.

Sol^n ② why is uvm considered as a methodology & not a Hardware Description language ?

- uvm is a methodology (a set of guidelines and a class library) that is built on top of System Verilog languge.

Sol^n ③ what is the difference Between SV & uvm ?

| System Verilog Test Bench | uvm Test Bench |
|---|---|
| ① No Base classes | ① Base classes are present |
| ⑪ we use class override Concept to override the Components | ⑪ we use factory override method to override the Components |
| ⑪ No Configuration dB. | ⑪ Configuration dB set/get methods are used to present. |
| ⑫ Components are Connected using mailbox and queue. | ⑫ Component are Connected using tlm and analysis ports. |
| ⑬ No phases | ⑬ phases are present to provide synchronization B/w various Components used in the testbench. |
| ⑭ No Reporting mechanisms. | ⑭ Reporting mechanisms helps to debug faster. |

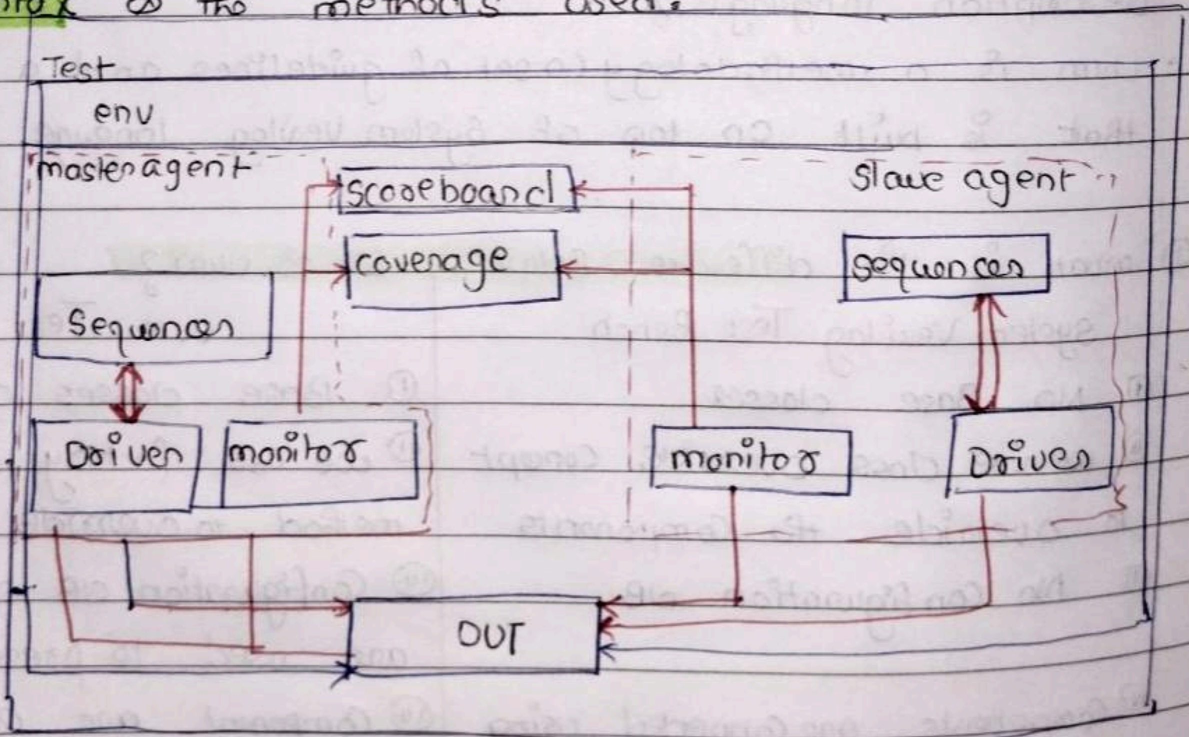
Solⁿ ④ is Uvm independent of System Verilog?

- Uvm is build on System Verilog, which provides Better way of verification, more random Constrained verif. it is re usable.
- Uvm is a methodology but System Verilog is a programming language, and Uvm also have pre-implemented Base classes.

Solⁿ ⑤ what are the disadvantage of Uvm methodology?

- learning curve is very high.
- it takes a lot of code to create basic testbench class But these bottleneck can be reduced via using a automated tool.
- still developing and not perfect / stable.

Solⁿ ⑥ Draw the uvm Testbench & explain each Component with syntax & the methods used:



Testbench Components / Objects:

① Sequencor :
- field required to generate the stimulus are declared in the sequence-item
- Sequence item is written by extending uvm seq-item;
- generating the randomized stimulus.

② Sequence : Sequence generates the stimulus & send to driven via sequences.
- Sequencer is written by extending uvm-sequences, there is no extra logic required to be added in the sequencer.

typedef uvm-sequencer ( tx-class) ~~m~~ Sqr ;

③ driver :
driver receives the stimulus from sequence via sequencer & drive on interface Signals:

driver extends uvm driver#(tx);

④ monitor :
monitor Samples the DUT signal through the virtual interface & convert signal level activity to the transaction level.

monitor extends uvm monitor;

⑤ Agent :
An agent is a container class contain a driver, sequencer & monitor. agent extends uvm_agent;

⑥ env : enviorment is the container class , it containts one or one more agents , as well as other components such as scoreboard , top-level monitor & checker . env extends uvm.env;

⑦ scoreboard ; it compares the actual value & expected output.

scoreboard extends uvm scoreboard;

⑧ Test : Test defines the test scenario for the tb.

Base_test extends uvm-test;

⑨ Subscriber : it subscribes the broadcaster ie: analysis port to receive broadcasted transaction.