DAY 93 - 100 DAYS VERIFICATION CHALLENGE

Topic: UVM Transaction, sequence_item, sequencer-driver methods

DAY 93 CHALLENGE:

1. Describe below methods in UVM Transaction with syntax & an example:
   i. copy
   ii. do_copy
   iii. compare
   iv. convert2string
   v. print
   vi. sprint
   vii. record
   viii. pack
   ix. unpack
   x. to_struct
   xi. from_struct
2. Describe below methods in sequence_item with syntax & an example:
   i. start_item()
   ii. finish_item()
   iii. get_response()
3. Describe below methods in sequencer-driver API with syntax & an example:
   i. get_next_item
   ii. try_next_item
   iii. item_done
   iv. peek
   v. get
   vi. put

DAY : (93)

Topic : uvm Trasaction , sequence item , Sequencer - driver methods:

Sol^n(1) Describe Below methods in uvm Transaction with Syntax & an example :

i.> Copy :

- The copy method is used to create a shallow copy of object This means that a new object is created that has the same attributes as the orignal object.
- but the new object does not Contain any dynamic data Structures , such as arrays or linked Lists.

ii.> do_copy: do_copy method is called by the copy() method.
- derived classes should implement this method to perform a deep copy of the transaction data.

iii.> Compare :

- The compare method returns 1 if Comparison matches for the current object when its Compared with the RHS argument object. it does a deep Comparision

iv.> convert2string:

- The Convert2 string() method is called by the user to provide object information in the form of a string .
- The Convert2string() is used to Convert each property of the transaction object into a string .
- Convert2string() to Convert the properties of the super class into the string .

```
virtual function string convert2string();
    string s = super.convert2string();
        s= $ s. $psprintf ("name : %s", get_name());
    return s;
endfunction
```

v.> Print : The print method is used to deep print uvm object class properties in well-formatted manner.

VI) Sprint : Similar to Convert2string Converts the transaction data to a string representation & returns it,

VII) Record : Records the transaction data using the specified recorder.

VIII) Pack : Pack() method pack_bytes(), & pack_ints() methods pack() method is used to pack each property of transaction object using a uvm_packer object.

IX) unpack : The unpack method is used to extract value from an array of type bit/byte/int & store it into a class format.

X) to_struct : Converts the transaction data to user defined struct format.

XI) from_struct : populates the transaction data from the user defined structure my_struct.

Sol^n ② Describe below methods in Sequence_item with syntax & an example:

I) Start_item(); This method is used to initialize the sequence item & mark the begging of a task & action

mySequenceitem . start_item();

II) finish_item(): This task initiales the generation of the sequence_item. The is_rand paramet when set to 1, indicate whether randomization should be applied during the generation of the item The finish_item task marks the compilation of the sequence_item generation.

• This method is called to finish the sequence item after all necessary data has been set or the task has been compleled.

mysequence . finish_item();

III) get_response() : this method is used to retrive any response associated with the sequence item. such as a response from the target after sending a request.

Sol^n③ Describe below methods in sequencer - driver API with Syntax & an example.

i.) get-next_item : This method is like asking the driver to give you the next task to do. it gets the next transaction item from the driver's queue and hands it over to you.

ii. try-next-item :

- similar to get_next_item, but it doesn't wait if there's no task available immediately. it tries to get the next_item and if there isn't one, it moves on without waiting.

iii item-done :

- when you have finished processing a task that you received from the driver, you tell the driver that you are done with it using this method.

v.) Peek :

- This allows you to inspect the next item in the sequencers request without removing it from the queue.

vi Get :

- similar to get_next_item, this task retrives the next_item from the sequencer's request queue, & it raises an error if the queue is empty.

vi) Put :

- This task is inserts the sequence item into sequencers request to queue.

DAY 95 - 100 DAYS VERIFICATION CHALLENGE

Topic: UVM Miscellaneous

DAY 95 CHALLENGE:

1. What is TLM FIFO?
2. What is the difference between `uvm_do and `uvm_ran_send?
3. What is the difference between uvm_virtual_sequencer and uvm_sequencer?
4. What are the benefits of using UVM?
5. What is the super keyword? What is the need of calling super.build() and super.connect()?
6. How to declare multiple imports?
7. What is the advantage of `uvm_pre_body and `uvm_post_body?

Day : (95)

Topic : UVM miscellaneous

Sol^n ① what is TLM FIFO ?

- Tlm fifo are used in UVB-Based Testbenches to tranfer transaction objects Between components, such as Between the sequencer and the driver or btw the driver & the monitor.
- Tlm FIFO is usefull for decoupling the producer and Consumer components in the Testbench, as they provide a level of abstraction that allows components to operate independently of each other, while ensuring the transaction s are exchanged in synchronized & thread-safe manner.

Sol^n ② what is the difference Between 'uvm_do' and 'uvm_ran_send'

- uvm_do ⇒
- its used to directly execute a sequence item without randomization
- its usefull when you need precise control over what tx is being generated.

- uvm_ran_send : it directly sends a randomized sequence item without creating it. so make sure the sequence item is created first.

Sol^n ③ what are the Benefit of using uvm ?

① modularity and Reusablity.
ii.> Separating Test from Testbench
iii> Sequence methodology
iv.> Configuration mechanism.
v.> while also facilitating through varification through coverage - driven technique & debugging.

Date
Page

Solⁿ ④ what is the difference Btw uvm_virtual_sequencer and uvm_sequencer ?

- A virtual sequence is derived from uvm_sequence.
- A virtual sequencer is derived from uvm_sequencer as a base class.
- uvm_sequencer is the path where the data is transffered from sequence to driver.
- virtual_sequencer controls other sequencers, its not affected to any other driver and can not process any sequence-item too.

Solⁿ ⑤ what is super keyword ? what is the need of calling super.build() and super.connect()?

- super is a keyword used to call methods or functions defined in the base class or parent class.
- super.build() is top-down method, its important for instantiating the verif components.
- super.connect() is bottom-up method, its important for connecting the tlm ports, tlm sockets & setting explicit-phase timeouts.

Solⁿ ⑥ How to declare multiple imports ?

uvm_analysis_imp_port_a#(tx, component_b) ap_imp_a;
uvm_analysis_imp_port_b#(tx, component_b) ap_imp_b;

Solⁿ ⑦ what is the advantage of 'uvm_pre_body and 'uvm_post_body?

- uvm_sequence has two callback methods pre_body & post_body, which are executed before & after the sequence body() method execution.
- These callback are called only when start_sequence() of sequencer or start() method of the sequence is called.
- user should not call these methods.