TrueConnect
Enriching Collaboration for Innovative Excellence
BENGALURU 2018

Implication & modus operandi applied for parallel register access WR/RD

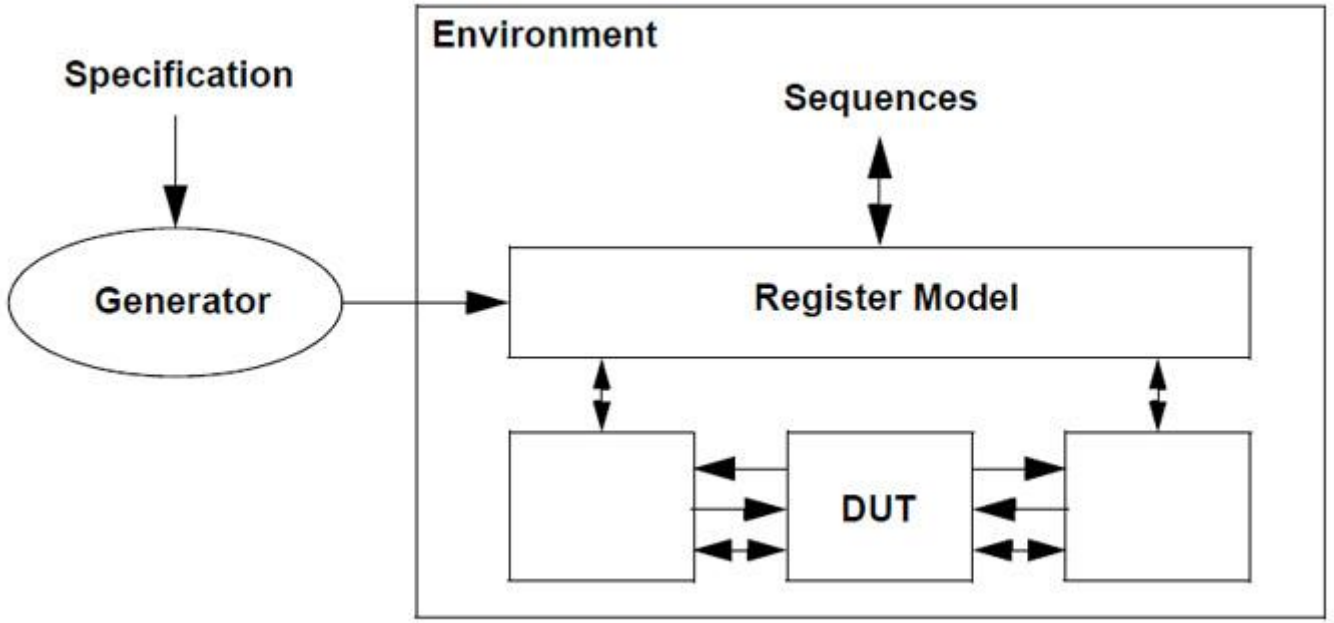
by

Jayakanthan Arthanari, Technologist
Vallivelraja Ponnudurai, Technologist

Introduction

- Register write/read using RAL model happens without any issues if we do procedurally.
- When doing with parallel threads few of the write/read get executed and few write/read get missed.
- Debugging this in random simulation tedious.
- Need to improve and optimize parallel write/read threads to improve failure debugging time in random constraint verification

Existing Approach



- regmodel.QUEUEID_ENABLE.set(queue_en);
  regmodel.QUEUEID_ENABLE.write(status,regmodel.QUEUEID_ENABLE.get(), UVM_FRONTDOOR);
-

Existing Approach -problem


```
class sequence1 extends base_sequence

    …….

   regmodel.QUEUEID_ENABLE.set(queue_en);

   regmodel.QUEUEID_ENABLE.write(status,regmodel.QUEUEID_ENABLE.get(),
UVM_FRONTDOOR);

…….

Endcalss


class sequence2 extends base_sequence

    …….

   regmodel.QUEUE_NUM.set(queue_en);

   regmodel.QUEUEID_ENABLE.write(status,regmodel.QUEUEID_ENABLE.get(),
UVM_FRONTDOOR);

…….

endcalss
```

```
class sequence3 extends base_sequence
    …….
   regmodel.QUEUEID_ENABLE.set(queue_en);

regmodel.QUEUEID_ENABLE.write(status,regmodel.QUE
UEID_ENABLE.get(), UVM_FRONTDOOR);
…….
Endcalss


fork
   sequence1
   sequnce2
   sequnce3

join
```

Proposed Approach -1

- Using semaphore for each and every register makes code cumbersome and also reduces simulation performance.
- semaphore.get(1);

  regmodel.QUEUEID_ENABLE.set(queue_en);

  regmodel.QUEUEID_ENABLE.write(status,regmodel.QUEUEID_ENABLE.get(), UVM_FRONTDOOR);

  semaphore.put(1)

Proposed Approach -2

- Create a generalized macro for write and read
- `define RAL_WRITE(regmodel,RgName, Value, DELAY=0) \
- ral_access_semaphore.get(1); \
- $display($time,`" RAL_WRITE : ``RgName`` 0x%0x`", Value); \
- regmodel.RgName.write(status, Value, UVM_FRONTDOOR);\
- ral_access_semaphore.put(1)

- Again need to include wherever needed !!!

Proposed Approach -3

- Create a generalized macro for write and read
- `define RAL_WRITE(regmodel,RgName, Value, DELAY=0) \
- ral_access_semaphore.get(1); \
- $display($time,`" RAL_WRITE : ``RgName`` 0x%0x`", Value); \
- regmodel.RgName.write(status, Value, UVM_FRONTDOOR);\
- ral_access_semaphore.put(1)

- Create a package for ral_model and add macro to that ral_model package

Results

- If We can create a macros and have single semaphore to handle all register, in this code is easier to maintain and very effective in simulation.
- It supports even if you have multiple register models.
- It will resolve issues faced for regressions for last 1 year
- It reduces, regression failures in random simulation due register configurations. So regression debugging 2WW time reduced for reach complete regression.
- It improves predictability.

REFERENCES

- [1] C. Spear, “System Verilog for Verification, A Guide to Learning the Testbench Language Features,” Springer, 2008G.
- [2] “SystemVerilog 3.1a Language Reference Manual”, Accellera, 2004
- [3] “UVM 1.2 User guide”, Accellera System Initiative, 2015.
- [4] “UVM 1.2 Class Reference”, Accellera System Initiative, 2014.
- [5]” Register Package Guide for UVM”, www.mentor.com

QUESTIONNAIRE???

Thank You !!!