The beginning of new norm: CDC/RDC constraints signoff through functional simulation

Suhas D S, Ponsankar Arumugam, Deepmala Sachan, Ritesh Jain

Intel Technology India Pvt. Ltd.

accellera SYSTEMS INITIATIVE

2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

INTRODUCTION

Clock Domain Crossing (CDC)and Reset Domain crossing(RDC) checks signoff poses several challenges in digital design, and addressing these challenges is crucial for ensuring the reliability and correctness of complex SoC's. While static analysis tools provide critical role in CDC/ RDC analysis, functional verification through simulations is **equally** necessary to validate the correctness of architectural assumptions to signoff the correctness of static analysis.

Current CDC/RDC constraints signoff challenges:

- **Accuracy of constraints** : CDC/RDC constraints used for static analysis gets written based on certain design assumptions but what if these assumptions are incorrect .
- **Thoroughness of constraints :** What if these assumptions are not complete ?
- **Validation of constraints :** Existing flow doesn't ensure the validity of these constrains.

Fundamental goal of this presentation is to provide holistic methodology for CDC/RDC constraints signoff which has been written based on design assumption using SystemVerilog Assertions(SVA) in functional simulations.

Following design assumptions are used for CDC/RDC analysis. In complex SoCs, Usage of SVA assertions during the design and validation phase accelerated the bug hunting multi-fold , resulted in quality functional sign-off.

1.Constant Signals: SVA for stability during crossings, preventing changes or glitches.

2.Quasi-Static Signals: Assertions ensuring stability during CDC/RDC analysis.

3.Gray Encoding: Assertions validating integrity during signal crossings.

4.Qualifiers: SVA capturing and verifying data transfer conditions.

5.Reset Ordering: Assertions validating reset ordering to prevent race conditions.

Clocks	**190 +**
Clock Domains	**60+**
Resets	**50+**
Reset Scenario	**39+**

SoC Complexity

SVA based functional simulation signoff for CDC/RDC Constraints

Static analysis checks are indispensable in verifying CDC and RDC in complex SoCs. These checks rely on various architectural assumptions to guide analysis. These assumptions play pivotal role in Full chip CDC analysis; hence it is imperative to validate them before final CDC/RDC closure. Following steps have been taken for SVA based functional simulation signoff for all assumptions which has been converted to constraints during static analysis

1. Generate the SystemVerilog Assertion (SVA) for each architecture assumptions.

2. Plug in the SV assertions in functional simulation and Validation of assertions.

3. Sign-off strategy for Uncovered Assertions: The goal is to get 100% assertion coverage however there would be certain exceptions due to SVA coding and testbench limitations. Such properties should be thoroughly reviewed with the architect and get sign-off.

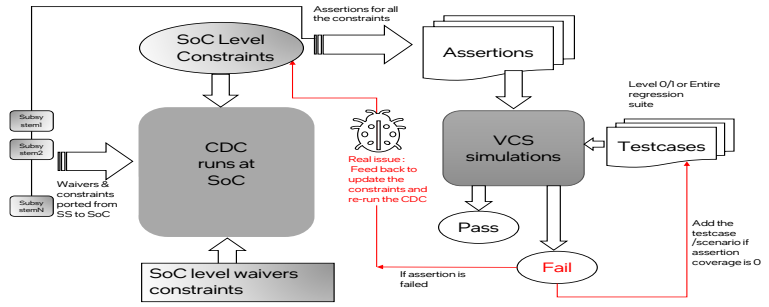

Figure-1: CDC constraints signoff flow based on SVA based validation

CASE Study: A Practical Exploration

1. False test failures due to testbench Architecture : Testcases are run on various TB models based on use cases where certain portion of design is stubbed out. When a design is stubbed out, a passive value is driven on input signals to avoid "x" propagation. In such cases, if value driven/forced is incorrect, the assertion will fail. In this case the testbench should be modeled to the architectural behavior. Many such instances were corrected in testbench.

In the Figure-3 the one pin of the AND gate is tied to zero in rtl. So SCA is given w.r.t the RTL tie off value, but during simulation there is force on this signal and the testcase used to toggling from 0 -> x -> 0 which resulted in CDC SVA failure.

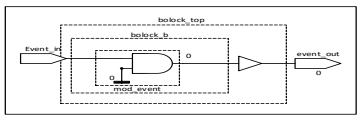

Figure-3: Schematic for false testcase failure

2. Removal of wrongly added set case analysis : These constraints were added on the output port of the unit level runs to match the abstract validation errors at SoC. This need not to be set, as internal logic will set according to the mode of operation.

We can see that in the Figure-4 the "enable" is ANDed with "scan_mode" inside the "sib_sti_inst" module, and in "scan_mode" that will be "0" in functional test. So, in this case the set case value for "bond_en" should be 0 for functional model analysis.

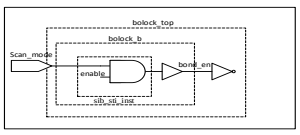

Figure-4: Schematic for wrong set case assumption

Summary : Methodology recommendations

As part of this methodology, we recommend having 100% Assertion coverage for constraints signoff however certain exceptions can be taken due to specific SVA coding and testbench limitation, but such properties must be thoroughly reviewed to get full confidence. In our project we have received 96.7% SVA coverage however remaining 3.3 % properties are categorized as invalid due to architectural assumptions.

Along with the modification of testbench to address uncovered assertions we have made following constraints update to strengthen the CDC analysis.

Removal of wrongly added set case analysis: These constraints were added on the output port of the unit level runs to match the abstract validation errors at SOC. this need not to be set, as internal logic will set according to the mode of operation.

```
set_case_analysis -name "soc.par1.scan_or_out" -value 0
```

Removal of wrongly added "reset-filter-path": It was wrongly added during CDC analysis to reduce huge waiver count: this was put to waive only where to_obj is present, but assertions are generated for from_rst to to_rst, which is huge verification overhead.

```
reset_filter_path -from_rst "rstA" -to_rst "rstB" -to_obj "flop/Q" -type rdc
```

Correction of SCA constraints: Few wrongly added constraint were corrected to its functional values.

Correction on data qualifier: All async FIFOs require proper qualifier constraints to avoid false assertions.

```
qualifier -dest_qual <FIFO-empty-signal> -from_obj <your-FIFO-register > -to_clk <destination-clock>

qualifier -src_stable -dest_qual <read-pointer> -from_obj <your-FIFO-register> -to_clk <destination-clock>
```

Data Hold assertions: All Memory or fifo assertions consider every data bit which are hard to hit. Hence first and last data bit of memory/fifo were considered to reduce overall assertion count which has drastic impact on simulation TAT.

With this, we strongly recommend SVA based constraint verification to go together along with regular CDC execution cycle without waiting for final constraints and Validation maturity.

CONCLUSIONS

During our journey of validating assumptions through SVA, we came across many challenges and learning. Some of key validation learnings are:

1. **Issues with reset initialization:** All RDC assertions with respect to reset domain crossing require analysis from time zero as it checks for destination reset to be asserted way before source reset.
2. **Strategy to preload or initialize memories with in-active values:** In certain scenarios this approach has been used to avoid false failures/Uncovered assertions.
3. **Failures due to power off conditions:** some false assertion failures have been noticed in power aware simulations due to "x" propagation during the power off. The Assertions need to be modeled accordingly to disable them during power off condition.
4. **Assertions having in active clocks in functional mode:** since our checks are done in functional mode, Test/DFX clocks and logic are not active hence, assertion containing them can't be hit as they are not active. The constraints were reviewed and SVA's were removed for such instances
5. **Absence of abstract for HIP/Memories:** All hard IP and memories should have an abstract model to avoid unnecessary assertions for simulation overhead.
6. **Testbench Modelling:** To avoid false failure, testbench models should be chosen appropriately so that undriven/falsely driven signal doesn't give unwanted coverage or failure.

Next steps: Exploration of formal for assertion validation: Multi die Graphics SOC's with huge complexity has lengthy regression cycles. Hence we will also explore implementing formal tools to achieve shorter runtime and quicker turn around. The challenge will be to have appropriate assumptions.

REFERENCES

[1]. Amit Kulkarni, Suhas D S, Deepmala Sachan, "Evolution of CDC recipe: Learning through real case studies and methodology improvements", in DVCON 2021

[2]. Rohit Kumar Sinha, "Enhancing Quality and Coverage of CDC Closure in Intel's SoC Design", in DVCON 2020

Intel Technology India Pvt. Ltd.