A Case Study on Formal Sequential Equivalence Checking based Hierarchical Flow Setup towards Faster Convergence of Complex SOC Designs

Anantharaj Thalaimalai Vanaraj 1 , Reshi Razdan 2
1Samsung Austin Semiconductors, Advanced Computing Lab, San Jose, 95134, CA, USA
2Samsung Austin Semiconductors, Samsung Austin Research Center, Austin, 78746, TX, USA
* Corresponding author: Anantharaj Thalaimalai Vanaraj, Samsung Austin Semiconductors - Advanced Computing Lab, 3655 N First St., San Jose, 95134, CA, USA; email: tvarvl@gmail.com

ABSTRACT: Functional Verification Sign-Off is the crux of the design verification problem faced by latest Silicon Designs on the Simulation/Stimulus Driven and the Formal Verification Platforms. Formal Verification Convergence is a custom specific criterion depending on the success, failure, exhaustiveness and reachability of the verification goals generated and validated by the Formal Tool. One of the key techniques in Formal Verification is the ability to mathematically prove the equivalence between two different versions of the same RTL Designs. Those RTL Design versions may differ in terms of Feature addition/removal or Bug Fixes or Low Power Capability or specific requirements. Synopsys VC Formal TM tool provides this formal verification technique using a built-in Formal Application known as ’Sequential Equivalence (SEQ)’ App. This Case Study outlines various approaches in deploying Formal SEQ App and an approach towards Faster Convergence.

KEYWORDS: Register Transfer Logic (RTL), Functional Verification, Formal Verification, Sequential Equivalence (SEQ), Sequential Equivalence Check (SEC), Synopsys VC Formal TM tool, Formal Convergence, Universal Verification Methodology (UVM), Portable Stimulus Standard (PSS), Artificial Intelligence (AI), Machine Learning (ML). Object Oriented Programming (OOPs), Factory Pattern, Design Under Test (DUT), System On Chip (SOC), Synopsys SolvNet Plus TM, Specification (SPEC), Implementation (IMPL), Return Of Investment (ROI)

1. Introduction

Complex & Computationally Intensive SOC Designs drive the functional verification complexity much beyond the realm of conventional verification techniques. Further the functional verification complexity is compounded by the shorter time-to-market requirements & performance intensive applications of the latest Silicon Designs[1].

Semiconductor & EDA Industries in collaboration with Research Community pushing the functional verification capabilities to address the ever-increasing design complexity. Those functional verification capabilities are improved through advent of language Capabilities like OOPs, Software Inspired capabilities like factory pattern, verification standards like UVM & PSS. Noticeably these exciting interventions are mostly centered on the heavily leveraged & standardized simulation driven dynamic verification platform.

2. Formal Verification

Formal Verification has been around for few decades co-existing with simulation driven dynamic verification platform. Concurrent and Exponential growth of AI/ML driven EDA Tool mathematical proofing capabilities, Hardware Computing Resources & Silicon Design Complexities bringing forth the Formal Verification towards addressing the verification gap (Figure 1).

Advanced Silicon Solutions for Artificial Intelligence, Machine Learning, Real Time Data Processing & High-Performance Computing are driving the Silicon Design Complexity that can be effectively handled by Mathematical Proofing techniques than Simulation driven verification techniques[2][3][4][5]. Hence, it’s time for Formal Verification to be understood, leveraged & deployed for Silicon Design Verification.
3. Formal Convergence is a Challenge

Formal Verification Convergence is defined by the ability to mathematically prove the absence of bugs in the Design Under Test (DUT) based on the Formal Constraints, Checks & Verification Setup[6]. Formal Convergence as a criterion is characterized by the success, failure & inconclusiveness of the formal properties in each Formal Application Mode. Similar to Simulation’s Verification SignOff Criteria, the Formal Convergence depends on multitude of factors like DUT Complexity, Formal Constraints Complexity, Formal Checks & much more (Figure 2).

Formal Convergence is the single most complex & demanding activity[5] in the Formal Verification Flow (Figure 3). In this paper, we will discuss the Formal Convergence of Sequential Equivalence Checking Mode of the Formal Verification Platforms using Synopsys VC Formal™ Tool.

5. SEQ Conventional Flow

In the conventional Flow of SEQ Mode, the Formal Engine checks the functional behavior at all the output ports of the top-level block across two versions of the same DUT (e.g., RTL Model A & RTL Model B as shown in Figure 5). Formal Engine proves or disproves the functional equivalence of the output ports of the top-level block in the SPEC & IMPL DUTs adhering to the defined Formal Input Constraints.

Though Formal Engine highlights internal sequential mismatches among with sub-blocks in the given DUT, there is NO output port level checks on the internal sub-blocks. Therefore, the Design Complexity of the entire DUT along with Formal Input Constraints plays a key role in the Formal Convergence.

Performing internal sequential difference checks to ensure the DUT similarity within Design States.

Synopsys VC Formal™ tool auto generates these output port checks for the given DUT (referred as SEQ_Top). This SEQ mode reduces the verification turnaround time (TAT) and improves productivity by providing an approach to exhaustively verify the modified/implemented Design feature. This avoids the need to redo the verification of the entire design. Towards Formally verify the Clock Gating logic in our complex SOC design, we have developed Formal SEQ Verification Setup using Synopsys VC Formal™ tool. The two versions of the DUT are referred to as SPEC and IMPL in the SEQ mode nomenclature. Here specification-SPEC refers to the complex SOC design blocks without Clock Gating feature disabled and implementation-IMPL refers to complex SOC design blocks with Clock Gating feature enabled (Figure 4).
6. SEQ Hierarchical Flow

Towards verifying the sub-block in each Top-Level Block without the need to develop Formal verification Setup, Synopsys VC Formal™ tool provides an approach known as ‘SEQ Hierarchical Verification Flow’. In this approach, we will be verifying the sub-block of a given Top-Level Block with complete reuse of the Formal Verification Setup & Input Constraints in the SEQ Mode as shown in the Figure 6. For the merit of this paper, we are utilizing the configuration#1 of the SEQ Hierarchical verification Flow defined in the Synopsys SolvNet Plus™ documentation[8]. Please refer to the SolvNet Plus™ documentation[9] for more details on the same.

7. Value Addition by SEQ Hierarchical Flow

It is imperative to understand the value of sub-block or unit, or IP level verification compared to the Top/System/SOC level verification in the Simulation Stimuli driven verification platform for obvious reasons. Similarly, there exists great potential & ROI in verifying the Sub-blocks in Formal Verification provided the reusability of Top-Level Verification Setup & Constraints. Further the SEQ Hierarchical Verification Flow provides the much-needed boost to achieve Formal Verification Convergence at a much-desired rate. Figure 7 highlights the key value addition of Sub-block Verification in the SEQ Mode.

8. Case Study

In this paper, we will be discussing the Formal SEQ Verification of Two Subsystem Level Top Blocks ‘A’ & ‘B’. These Blocks are few among the many subsystems with different functional capabilities & design complexities in the complex SOC (Figure 8).

Refer below Table 1 for a high-level relative comparison of Blocks ‘A’ & ‘B’.

<table>
<thead>
<tr>
<th>Block</th>
<th>Design Complexity</th>
<th>Number of Sub blocks(first level)</th>
<th>Number of input ports</th>
<th>Number of output ports</th>
<th>Multipliers</th>
<th>Counters</th>
<th>Arithmetic Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>'A'</td>
<td>High</td>
<td>6</td>
<td>~250</td>
<td>~750</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>'B'</td>
<td>Medium</td>
<td>5</td>
<td>~150</td>
<td>~450</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

8.1. Block ‘A’ – SEQ Conventional Setup

As Shown in the Figure 9, the Formal Engine drives the input ports of both the SPEC & IMPL (Top level block ‘A’) based on the Formal Input Constraints and generates the auto checks to compare the output ports of the SPEC.
& IMPL (Top level block ‘A’). Here the Formal Convergence of the auto checks on the output ports is impacted by the Design Complexity of the entire DUT – Top Level Block ‘A’ and its Sub-blocks. Owing to the Complex nature of the SOC Design & large Cone of Influence (COI), we have achieved <80% Formal Convergence in our Case Study for Clock Gating Verification of Top-Level Block ‘A’.

Despite various efforts to improve Formal Convergence by Formal Techniques (like Black Boxing, Abstractions, Design reductions, Enhances Engine & Effort level and much more), our verification returns were stagnated with the unrelenting Formal Convergence at level of < 80% for this Top-Level Block ‘A’. Hence, we have decided to deploy SEQ Hierarchical Flow to achieve the Formal Convergence of this Top-Level Block ‘A’.

8.2. Block ‘A’ – SEQ Hierarchical Setups

In the SEQ Hierarchical Verification Flow (Figure 10, Figure 11 & Figure 12) of Block ‘A’, the Formal Engine drives the input ports of both the SPEC & IMPL (Top level block ‘A’) based on the Formal Input Constraints identical to the SEQ Conventional Verification Flow. But the Formal Engine generates auto checks to compare the output ports of the targeted Sub-block within the SPEC & IMPL blocks (Top level block ‘A’). Multiple types of SEQ Hierarchical Verification Flow Testbenches were developed specific to each Sub-block which were chosen on selection criteria (refer next section).

8.3. Block ‘A’ – SEQ Conventional Setup with verified Sub-blocks Clock Gating Disabled

After successful verification closure of Sub-blocks S1, S2 & S3 from Top Level Block ‘A’ for SEQ Hierarchical Verification Flow setups as show in the Figure 10, Figure 11 & Figure 12. Here the Formal Convergence of these newly generated auto checks on the sub-block output ports can be achieved much faster due to the limited nature of the Design Complexity & Cone Of Influence (COI) on these output checks.
same Formal Verification Setup & Constraints except for Clock Gating disabled sub-blocks within the DUT. This clearly proves the effectiveness of the SEQ Hierarchical Verification Flow on the selective Sub-Blocks (S1, S2 & S3) of the Top-Level Block ‘A’ to achieve faster Formal Convergence.

### 8.4. Sub-block Selection Criteria

Even though the Sub-block SEQ verification setup can be developed quickly with reusable components from Top Level SEQ verification setup, it is ineffective to perform Sub-block verification on all the first level sub-blocks within a Top-Level Block. Henceforth we have devised a selection criterion to choose a sub-block for SEQ Hierarchical Verification Flow. Please note this selection criteria may be effective specifically for this case study.

Factors involved in the selection criterion of a sub-block are: (not limited to)

1. Number of Inconclusive Top Level Block Checks impacted by the COI of this sub-block
2. Placement of this sub-block within the Logic Levels of the Top-Level Block
3. Proximity of this Sub-block to the Input/Output Ports of the Top-Level Block
4. Fan-in & Fan-out nature of this sub-block
5. Design Complexity of this sub-block

### 8.5. Block ‘B’ – SEQ Verification Setups

Considering that the verification setups of Top-Level Block ‘B’ is like the Top-Level Block ‘A’ as discussed in the previous sections, we are omitting the details on Top Level Block ‘B’ SEQ Verification Setup in this paper.

### 8.6. SEQ Verification Metrics Table

Based on this case study execution of Formal SEQ Verification on Top Level Blocks ‘A’ & ‘B’, we have captured the key metrics from the Synopsys VC Formal™ Tool execution. These metrics were gathered from various Formal runs with unchanged Formal Verification setup (except SEQ Flow change), Input Constraints, Design Versions of IMPL/SPEC, Tool Settings like Engine Selection, Number of Workers and so.

#### Table 2: SEQ Metrics of Block ‘A’

<table>
<thead>
<tr>
<th>DUT</th>
<th>Formal SEQ</th>
<th>SEQ Goals Count</th>
<th>Goals Converged (%)</th>
<th>Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Block - A</td>
<td>Conventional</td>
<td>730</td>
<td>536 (72%)</td>
<td>150hrs</td>
</tr>
<tr>
<td>Sub-block - S1</td>
<td>Hierarchical</td>
<td>437</td>
<td>437 (100%)</td>
<td>91hrs</td>
</tr>
<tr>
<td>Sub-block - S2</td>
<td>Hierarchical</td>
<td>140</td>
<td>140 (100%)</td>
<td>34hrs</td>
</tr>
<tr>
<td>Sub-block - S3</td>
<td>Hierarchical</td>
<td>290</td>
<td>290 (100%)</td>
<td>51hrs</td>
</tr>
</tbody>
</table>

#### Table 3: SEQ Metrics of Block ‘B’

<table>
<thead>
<tr>
<th>DUT</th>
<th>Formal SEQ</th>
<th>SEQ Goals Count</th>
<th>Goals Converged (%)</th>
<th>Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Block - B</td>
<td>Conventional</td>
<td>452</td>
<td>97 (21.4%)</td>
<td>90hrs</td>
</tr>
<tr>
<td>Sub-block - S1</td>
<td>Hierarchical</td>
<td>410</td>
<td>409 (99.75%)</td>
<td>21hrs</td>
</tr>
<tr>
<td>Sub-block - S2</td>
<td>Hierarchical</td>
<td>107</td>
<td>107 (100%)</td>
<td>11hrs</td>
</tr>
</tbody>
</table>

### 9. SEQ Verification Formal Convergence Rate

We have analyzed the Rate of Convergence for all the SEQ Verification Flow Setups discussed in this case study on Top Level Blocks ‘A’ & ‘B’.

#### 9.1. Block ‘A’ SEQ Verification Formal Convergence Rate

In this case study, we were able to achieve Formal Convergence at a faster rate for Top Level Block ‘A’ with clock gating disabled on verified sub-blocks (S1, S2 & S3) in comparison to the SEQ Conventional Verification Flow of Top-Level Block ‘A’ (refer Figure 14)

#### Figure 14: Formal Convergence Rate of Block ‘A’

### 9.1.2. Block ‘B’ SEQ Verification Formal Convergence Rate

In this case study, we were able to achieve Formal Convergence at a faster rate for Top Level Block ‘B’ with clock gating disabled on verified sub-blocks (S1 & S2) in comparison to the SEQ Conventional Verification Flow of Top-Level Block ‘B’ (refer Figure 15)
Conclusions

This case study explored the Formal SEQ Verification setups with conventional and hierarchical flows on the Top-Level Blocks ‘A’ & ‘B’. Further we have devised an added ability to disable clock gating for verified sub-blocks using SEQ Hierarchical flow in the Top-Level Block SEQ Conventional Flow.

Table 4: Relative Comparison of SEQ Flows

<table>
<thead>
<tr>
<th>Feature</th>
<th>Conventional</th>
<th>Hierarchical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testbench Top</td>
<td>Single</td>
<td>Multiple</td>
</tr>
<tr>
<td>Constraints Reuse</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Independent Sub-block Verification Support</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Efforts required for Convergence</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>Rate of Convergence</td>
<td>Average</td>
<td>Faster</td>
</tr>
</tbody>
</table>

In Table 4, we have highlighted the key comparative features of the SEQ Conventional and Hierarchical Flows.

Recommendations (1/3)

✓ Define customized sub-block selection criteria in each SOC Design for efficiency

✓ Deploy SEQ HIER flow at an early stage in RTL development cycle as well as for critical sub-blocks in SOC Designs

✓ Utilize Synopsys VC Formal™ ‘easy-to-setup’ tool options for SEQ Hierarchical Flow

Consequently, we were able to achieve faster convergence between the Top-Level blocks in the Formal SEQ Conventional Flow. Figure 16, Figure 17 & Figure 18 illustrates the recommendations on SEQ Hierarchical Flow from this Case Study.

Conflict of Interest
The authors declare no conflict of interest.

Acknowledgment
The authors acknowledge the support and guidance from the Synopsys VC Formal™ applications team and Samsung Austin Research Center (SARC) Design and verification teams.

References


ANANTHARAJ THALAIMALAI VANARAJ has more than two decades of semiconductor integrated chips development experience in the storage, wireless and computing product domains. He had completed bachelor's degree in electronics and communication engineering from Thanthai Periyar Institute of Technology, Vellore, Tamilnadu, India by 2002. He had received Master's degree in VLSI System from National Institute of Technology, Tiruchirappalli, Tamilnadu, India by 2004. In 2022, he was awarded PhD degree in post-CMOS technology by National Institute of Technology, Tiruchirappalli, Tamilnadu, India.

He is currently working as Formal Verification Lead with Samsung Austin Research Centre – Advanced Computing Lab (SARC-ACL), San Jose, California, USA. At SARC-ACL, he is driving the exploration and deployment of advanced formal techniques for verification of complex SOC designs. He has delivered multiple technical sessions in conferences, seminars and workshops. He has extensive research and development experience in NAND Flash Memory and ASIC/SOC design verification using IEEE 1800 ™ based System Verilog, System Verilog Assertion (SVA) and Universal Verification Methodology (UVM). He had received six US patents related to NAND Flash Memory and SSD products. He has published more than 10 research articles. He is also a reviewer and editorial board member in various reputed international journals. His research area involves NAND Flash Memory, Memory Architecture, CMOS VLSI Digital Design, IoT, Machine Learning, Digital Signal Processing, VLSI Logic/Functional Verification, and Quantum-dot Cellular Automata (QCA) designs.

RESHI RAZDAN has more than seven years of experience in the semiconductors and electronics product domain. He had worked with Huawei, Reliance Jio Infocom Limited and AMS Sensors. He completed a bachelor’s degree in electronics and communication engineering from Mumbai University in 2015. He completed Masters' degree from University of Maryland, College Park by 2020 with specialization in Computer Architecture, Digital Circuits designing and Verification of digital systems. Currently he is working as a senior formal verification engineer with Samsung Austin Research Center (SARC) at Austin, Texas. He is responsible for verifying one of the key subsystems of complex SOC design using various Formal Verification methodologies.