Statistical Analysis of Clock Domain Crossing

Rajat Singla (*rsingla@nvidia.com*), Tanneru Sai Pavan (*tsaipavan@nvidia.com*), Naveen Dugar (*naveen@realintent.com*), Varun Sharma (*vsharma@realintent.com)*

***Abstract*—Nvidia is an undisputed leader in the semiconductor industry with its advanced designs driving various market needs in datacenter, gaming, robotics, and artificial intelligence applications. These complexities in the applications translate into complex designs that we work on. Handling clock-domain-crossings ahead in the design cycle is a common challenge across the industry, and Nvidia designs are no exception here but follow a strict methodology to ensure the highest quality of RTL. As the complexity of design is increasing with new feature addition or complex algorithm development, the quantum of violations is going out of control for analysis. The fix of these large quanta often belongs to some deterministic fixes in constraints or RTL. Analyzing the same kind of structures hidden in large quantum is costly and affects the time to market. To address high-bandwidth high throughput designs in reporting unsafe clock-domain crossings effectively, we would like to introduce a statistical-based novel methodology called "Root-Cause Analysis" aka RCA. Given the intricate nature of clock-domain structures at Nvidia, this methodology has the potential to simplify the analysis of potential vulnerabilities at the crossings. It accomplishes this by offering upfront suggestions to identify possible root causes. In this paper, we aim to showcase several active cases where this tool's features could potentially expedite the signoff timelines. By reducing noise and adopting a non-iterative approach, this methodology significantly accelerates the CDC verification process. We attempt to present an evaluation study of this methodology using the Meridian CDC tool and present the results obtained from a real SOC design.**

Keywords— Clock domain crossing, Meridian CDC, CDC constraints, Multi-clock Design, SOC, CDC analysis tools, Statistical Analysis, Root cause identification, Verification sign-off

I. INTRODUCTION

For the contemporary multi-million instance SoCs, having thousands of clock domains has become the norm, and using various synchronization mechanisms for the crossings is an integral part of the design cycle. While statically verifying synchronization schemes in the design has been made easy by various EDA tools, the SNR can be too small, making the cleanup process quite cumbersome. The reason behind this is that all such tools work structurally and rely on user constraints to limit the state space for practical analysis. The completeness of constraints cannot be assumed and is difficult to finetune to produce optimal results. In the presence of missing or incomplete constraints, it is highly difficult to filter out the actual violations from the noise.

Having said that, the EDA tools also have evolved in recent times to not just report the deficiencies in clock-domain crossings but also to provide guidelines for resolving the issues, typically using AI/ML algorithms based on statistical and structural design data to provide more actionable suggestions.

MeridianCDC has been a leading industry-standard clock-domain verification tool for the better part of the last decade and has recently introduced a methodology named “Root-Cause Analysis” groups. Given the design complexities at Nvidia, this new methodology has the potential to make the analysis easier for our complex clock-domain structures for potential vulnerabilities at the crossings, which it achieves by providing additional upfront suggestions to find the possible root causes. This methodology doesn't just identify the root cause of many of the reported violations but can also provide insights to fix them by either adding a suitable constraint or correcting the design where necessary. Since this methodology groups violations based on their root cause, the identified fixes would be more concise, and provide better coverage overall, in comparison to the incumbent manual process.

The paper will attempt to highlight a few real-time cases where the tool's feature could be used to pull in the signoff timelines potentially.

To evaluate this new feature supported by the MeridianCDC tool, a set of representative designs was identified to cover the maximum spectrum of violations seen. In these designs, the analysis groups were reported using the tool, and the results were analyzed thoroughly. The evaluation was done qualitatively by analyzing the actionability of each of the groups reported by the tool, and quantitatively by mapping the number of root-cause groups reported and the corresponding number of violations that get covered per RCG.

II. TRADITIONAL CDC VERIFICATION METHODOLOGY

As in any competitive system on chip designs in the industry, our designs consist of multiple clock domains operating asynchronously. A challenge we face is that the signals crossing between these domains can breach the setup/hold window of the receiving clock, leading to metastability issues. These failures related to metastability can be sporadic and difficult to identify, potentially resulting in costly chip re-spins if discovered late in the design cycle. To address this, we rely on a tool called Meridian, which helps us detect metastability-related failures at the RTL (Register Transfer Level) stage.

Similar to any other tool, Meridian accepts different inputs, performs parsing, and conducts verification on these inputs. This process is illustrated in the first step shown in the diagram, where the tool reads libraries, Verilog files, and associated constraints, among others. In the second step marked "Setup analysis," the tool examines user specifications (definitions of clocks, resets, constants, inputs, outputs) for correctness, consistency, and completeness. The third and final step, referred to as "CDC Structural Analysis," specifically focuses on CDC verification. The tool scans the design, identifying structures that do not conform to safe CDC requirements.

Meridian CDC offers several debugging capabilities, such as an "idebug" GUI, which assists in pinpointing errors and reducing the debugging effort. The CDC verification engineer carefully analyzes the results, eliminates false violations by updating design constraints or waivers, and then re-runs the tool. This iterative process of running the tool and fine-tuning design constraints or waivers continues until satisfactory results are achieved.

Once the design constraints are stable and the CDC verification results are satisfactory, the engineer moves forward to address any identified issues. However, the traditional approach to this process has its drawbacks. It involves an iterative and potentially error-prone constraint-tuning process, as well as the complexity of gathering relevant design information from geographically dispersed block designers. The inherent delay involved in this process may essentially diminish the sheen of using RTL-CDC overall.

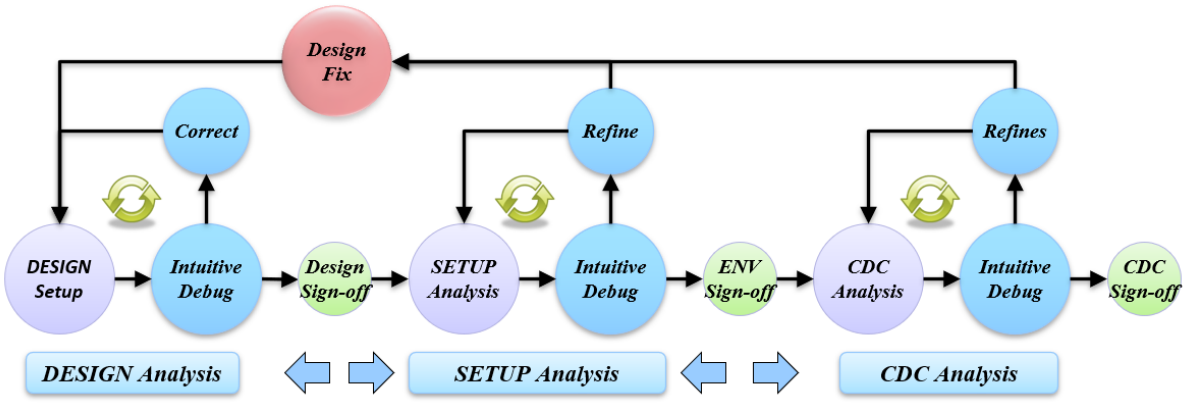


Fig-1: Conventional CDC verification method

III. CDC VERIFICATION WITH ROOT CAUSE GROUPS

The CDC methodology is undergoing an update to include RCA (Root Cause Analysis) checks enabled by default. This enhancement aims to provide better design constraints and minimize undesired CDC violations. The results of the RCA checks will be incorporated into the CDC Structural violations, enabling designers to understand the impact of unconstrained or misconstrained signals identified by the RCA checks. After analyzing these RCA checks, if the designer determines that the reported signals align with the design intent, they can apply appropriate constraints. However, if there are real CDC Structural violations associated with the reported signals, a thorough review and analysis of the violations are necessary.

The inclusion of RCA checks offers several benefits,

1. Reduces the time, cost, and effort required in the CDC verification process significantly.
2. When a new design undergoes CDC analysis, the RCA checks help expedite the CDC setup time.
3. The presence of design changes, such as the addition of new clocks or resets, may go unnoticed by the CDC verification engineer. RCA checks play a vital role in identifying the relationship between these new changes and the pre-existing design elements.
4. By validating and providing accurate constraints before running the CDC analysis, a significant amount of runtime can be saved. This enables the CDC tool to focus more on identifying genuine bugs rather than wasting time analyzing false issues.

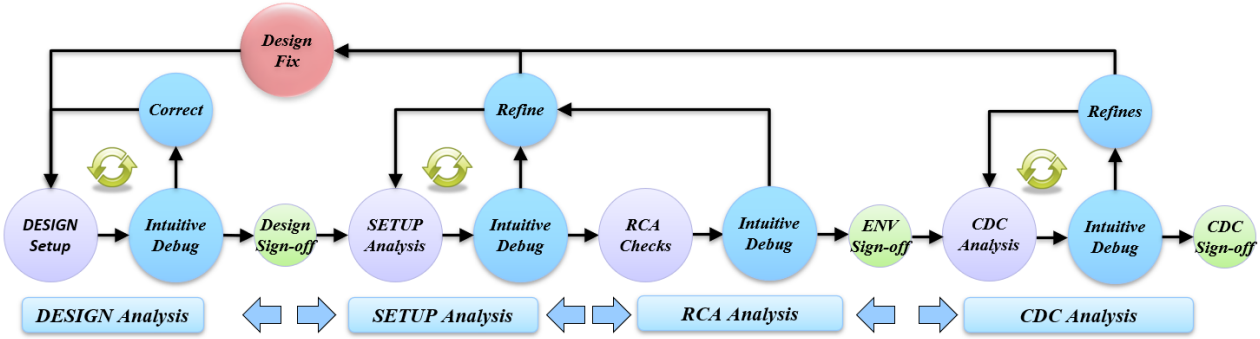


Fig-2: SDC setup-based CDC verification method

IV. UNDERSTANDING RCA

All industrial tools use static technology to generate violation quantum for users. This violation-quantum is sometimes noisy which is later found when users debug them one by one. Behind these violations, there are already some hidden root causes that can bring down these violations in an effective ballpark. Therefore, we need some way to analyze the given set of violations and produce probable root causes.

The tool proposes a new solution that works using statistical analysis on a given violation set to find possible groups having common root causes. This technology proposes multiple potential root causes which users can act upon in terms of constraints or RTL fix to bring down a large quantum in the effective count.

The proposed solution has multiple checks, we will try to understand one or two to understand the depth that probable root cause checks are offering.

Say we have m asynchronous clocks in design which gives in total clock domain crossings n(T). When the user starts its analysis, found that most of the crossings are within one clock pair, say, C1 to C2. The sync crossings path covered by (C1, C2) is in total n(C1,C2) such that they have a relationship as mentioned in equation (1), where configurable Th is 1%.

$$n(C1,C2)/n(T) <= \text{Th} \quad (1)$$

In statistical analysis, we figured out that more than Th are completely asynchronous which points to some serious problems in one of the following ways

1. Bug in identifying correct Sync/Async relationship between C1 and C2 by tool
2. The incorrect relationship defined by set_false_path or set_clock_groups for C1 and C2

Let's assume user-provided constraints are well understood by the tool i.e., first point 1 above is correct. The probable root cause is an incorrect relationship which user-defined because having such a large quantum of async crossings is not expected in any design.

Let's look at other very general RCA which impact the addition of stable/static constraints. Say out of total async crossing paths n(T) we have some violations which all have common drivers or driver K fanout to total n(K) flops provided bus is considered one. Then signal K is considered stable such that they have a relationship as mentioned in equation (2), where configurable Th is 1%.

$$n(K)/n(T) <= \text{Th} \quad (2)$$

This is how the feature points to probable root cause hidden in large violation-quantum. In upcoming sections of the paper, we will try to study in depth the few interesting scenarios caught by this proposed solution and how to save the user's effort by multiple times gain.

V. CASE STUDY

In order to demonstrate the effectiveness of the RCA checks methodology, we incorporated RCA checks into our CDC verification flow. We would like to emphasize the significant benefits we observed as a result of this evaluation exercise.

In one of our designs, we initially encountered a large number of violations, precisely 56,586 violations. To address this issue, we enabled root cause checks (RCA checks) and implemented the potential constraint fixes for three specific checks: RC1(ignoring RXs having large driver set), RC2(not actual async clocks), and RC3(stable Tx having large load set). As a result of these measures, the number of violations reduced by an impressive 86%.

The implementation of RCA checks proved invaluable in identifying missing constraints on global signals. For example, we discovered that a specific mode reset lacked a clock constraint, leading the tool to consider it asynchronous.

In another case, we uncovered that the outputs of shallow latches within a particular structure could be constrained as stable signals. Since this structure was highly repeatable and present in multiple IPs, applying project-level constraints allowed us to effectively reduce the violation count across different designs.

Furthermore, in certain designs, we encountered receiving flops or ports with high fan-in that were deemed irrelevant for CDC analysis due to their don't care nature.

We found the RCA checks to be quite powerful, that they revealed a correlation between multiple designs, since designs that are involved are on mobile applications, hence sharing some common design structures. These common design structures are pointed out by RCA recommendations to improve constraints around them.

These findings demonstrate the capabilities of RCA checks in detecting constraint issues, optimizing constraint application at different levels (such as project-level constraints), and identifying signals that can be disregarded for CDC analysis, ultimately enhancing the efficiency and accuracy of the CDC verification process.

Without the RCA checks, the initial CDC analysis would have required multiple iterations, involving the analysis of each violation category and subsequent design fixes or constraint additions. However, with the aid of RCA, the CDC verification engineer only needs to review the signals associated with grouped violations identified in the RCA checks. These checks also provide information regarding the impact of these signals. Consequently, the verification engineer gains insights into which signals are causing the majority of violations without even examining the structural violations. Once the verification engineer validates the signals reported in the RCA checks and applies the necessary constraints, only a manageable number of CDC violations remain for review, eliminating the need for multiple iterations. This allows the engineer to promptly address CDC errors. The results are summarized in Table-1.

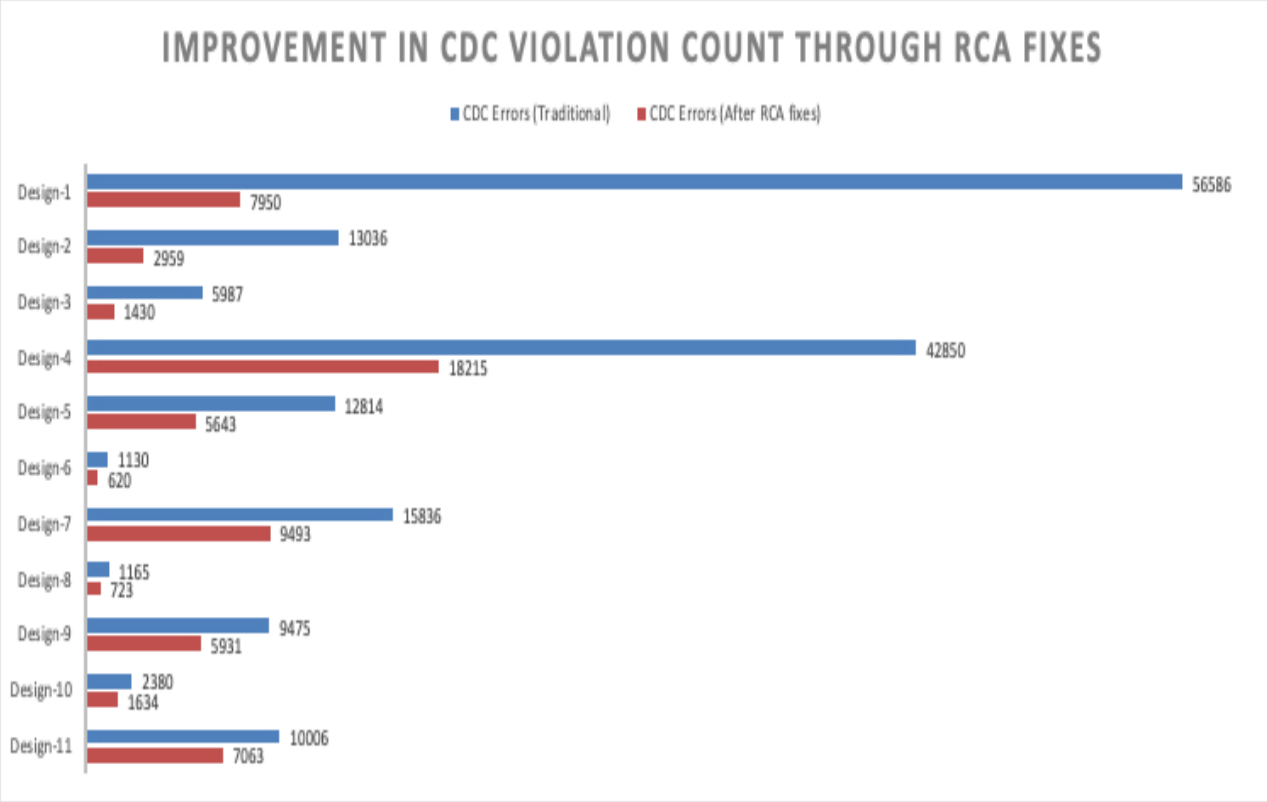
Design	CLK Groups	Design Size: eq NAND2 gates in Mil Inst	Conventional CDC Violation Count	RC1	RC2	RC3	Percentage Reduction in CDC Errors*
Design-1	81	3.51	56586	13	6	11	86%
Design-2	25	0.30	13036	0	1	74	77%
Design-3	20	3.40	5987	0	0	32	76%
Design-4	730	20	42850	0	81	162	57%
Design-5	57	12	12814	0	0	113	56%

Design	CLK Groups	Design Size: eq NAND2 gates in Mil Inst	Conventional CDC Violation Count	RC1	RC2	RC3	Percentage Reduction in CDC Errors*
Design-6	5	0.02	1130	0	0	3	45%
Design-7	27	9.59	15836	22	1	42	40%
Design-8	17	5.71	1165	0	0	6	38%
Design-9	78	11.40	9475	0	0	15	37%
Design-10	22	0.35	2380	0	0	5	31%
Design-11	72	5.29	10006	1	0	5	29%

*Reduced if suggested recommendations by RCs applied on the design; Suggestions are reviewed initially before applying, but are not signed-off by the design team yet.

Table-1: CDC Violation Reduction with the Proposed RCA Flow

Figure-1: Conventional vs. Proposed CDC methodology Violation Count



VI. CONCLUSION

From the case study, the benefits of this methodology are obvious. Root-cause groups are a statistical tool to efficiently identify the common root cause of multiple CDC violations reported by the tool and suggest suitable and possible fixes for the identified deficiencies. This helps eliminate days spent creating, refining and qualifying the CDC setup. The methodology is generic and non-iterative. Noise in CDC results is eliminated mostly and the verification engineers are left with real CDC violations and can start fixing CDC results immediately. The proposed method accelerates CDC verification closure time and it results in valuable savings of time, effort, and costs.

VII. FUTURE SCOPE

Root Cause Analysis is a never-ending process. It gets mature and evolves with every learning from new design debug. For Nvidia, the obvious next step is to incorporate RCA in the flow and have a reasonable and actionable way of providing the set of suggestions from RCA to the designers. The aim is to have RCA as a check which will give designers upfront recommendations of probable constraint updates to improve the setup or the RTL.

For the tool vendor, the feedback from the evaluation adds to the statistical data and helps improve the accuracy of RCAs reporting. Based on the actual learnings from this evaluation, RCA will be getting new checks, few are as follows:

1. Crossing paths that don't converge to primary output even through sequential elements in fanout cone, will not have any impact even if metastability is present because it will be pruned off by synthesis engine or in the matured state of RTL.

2. Crossings belong to common debug actions because they all have a common through-node.

3. Extending to coherency checks which will have the following checks

    a. Identifying vector control which is shared by some N reconverging structure - Possibly global control signal which is independent

    b. Reconvergong structure participating in FIFO interface or controls from FIFO participating in reconvergence.

Along with new checks, the plan is to have a better correlation between root causes and crossings and utilize them for building a deep learning model on these recommendations with advanced statistical algorithms.

REFERENCES

[1] Clifford E. Cummings, “Clock Domain Crossing (CDC) Design & Verification Techniques Using System Verilog”, SNUG-2008

[2] Vishnu C Vimjam and Al Joseph, “Challenges in Verification of Clock Domain Crossings”, DAC knowledge center Article

[3] Ping Yeung, “*Five Steps to Quality CDC Verification”*, Mentor Graphics, Advanced Verification White Paper

[4] “Using the Synopsys Design Constraints”, Application Note v1.9, 2010

[5] Athaiya, S., Komondoor, R., Kumar, K.N.: Dataflow analysis of asynchronous systems using innite abstract domains (2021)

[6] Mark Litterick, “Pragmatic Simulation-Based Verification of Clock Domain Crossing and Jitter using System Verilog Assertions”, DVCon, 2006

[7] Chris Kwok, et al, “Using Assertion-Based Verification to Verify Clock Domain Crossing Signals”, DVCon, February 2003