SYNOPSYS®

WHITE PAPER

Better, Faster, and More Efficient Verification with the Power of AI

Author

Taruna Reddy
Staff Product Marketing Manager

Overview

Every aspect of semiconductor development is hard and getting harder every day, but verification may be the most challenging stage of the process. For years, studies have shown that the percentage of time and resources devoted to verification has increased for each new chip generation. Thus, overall, verification is growing faster than other stages of chip development and chip projects. Teams are demanding that they be able to achieve better results in less time and with fewer resources. The electronic design automation (EDA) industry has responded to this crisis by applying the power of artificial intelligence (AI) to the various steps throughout the verification process. This white paper outlines some of the major challenges for verification, describes how AI provides assistance, and mentions specific capabilities available in the EDA solutions from Synopsys.

Key Verification Challenges

It is sometimes said that the goal of any chip verification tool, process, or methodology is to find more bugs more quickly. That's true enough on the surface, but the reality is broader and more nuanced. In fact, verification engineers care about three dimensions: quality of results (QOR), time to results (TTR), and cost of results (COR). QOR is the easiest to grasp; verification teams want to find all bugs, and all kinds of bugs. Sheer quantity is important, but the type and complexity of the bugs found also matter. Even finding 99.9% of all bugs is inadequate if those missed could cause a system hang or the failure of a fabricated chip.

There is no way to know for sure when all bugs have been found, but there are metrics to help a verification team decide when to declare success and tape out the chip. For simulation, functional and structural coverage metrics are the most common method. Achieving high coverage builds confidence in the correctness of the design, and missing coverage provides excellent guidance on what additional tests are needed. For static and formal analysis, the percentage of checks and properties proven is the main data point. Properties that cannot be 100% proven still provide bounded proof information to assess the depth of the analysis.

TTR is important for several reasons. Every chip project has a target release date determined by market opportunities and competitive pressures. The schedule is planned carefully to meet this date but, if convergence to coverage closure happens more slowly than expected, tape-out and therefore product shipment can be delayed. This can reduce profits and, in the worst case, render a chip uncompetitive by the time that it is released. Meeting an aggressive schedule requires fast testing and analysis in all verification tools, smooth coverage closure, and rapid debug. If the project falls behind schedule, the common response is to grow the verification team. As documented many years ago, adding more people to an engineering project does not necessarily reduce the time required for completion.

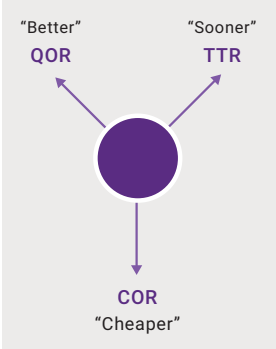

Figure 1: Three dimensions of chip verification

Resources also figure into the third dimension of COR. Adding more engineers increases project cost and reduces the potential profitability of the end product. Acquiring more compute servers—or renting them on the cloud—also adds to the cost. In addition, every additional simulation or formal run produces more results that may need to be analyzed. If additional verification tests are largely redundant with existing tests, the unnecessary increase in debug time may reduce verification efficiency rather than improve it. Finally, every chip turn adds a huge COR to the project budget and delays TTR, so high QOR is critical.

AI techniques, many based on machine learning (ML), can dramatically improve all three dimensions of chip verification. Figure 2 shows a high-level view of a typical verification flow. Before the register transfer level (RTL) code is written by the designers, the architecture team builds a virtual model of the chip and analyzes system performance. When key decisions have been made, the design team develops the RTL model. This is best accomplished by using an integrated development environment (IDE) with powerful language linting to catch coding errors that would otherwise consume time and resources to fix later in the project. Once lint errors are fixed, the first step in verification is usually performed by the designers themselves: running static verification tools to detect structural errors in the design.

Deeper analysis is provided by formal verification tools that try to prove key properties about the RTL design. These tools may also be run by the designers, although formal specialists may be involved as well. In parallel, the verification team develops the testbench and models that will run a series of tests to meet the goals of the verification plan, often taking advantage of existing verification IP (VIP). These tests are run primarily in simulation, but use of hardware emulation platforms has become more common in recent years. As mentioned earlier, coverage metrics are the primary method for gauging verification progress. Coverage closure is a highly iterative process, and this is often the longest portion of the verification schedule.

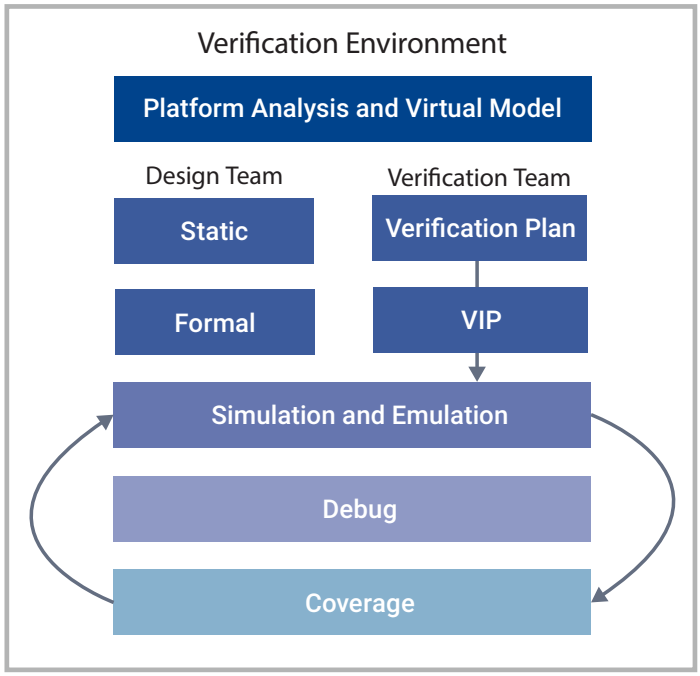

Figure 2: High-level view of the chip verification flow

AI Technologies Applied to Static Verification

As static verification has added more types of checks, it has become an increasingly important part of chip verification. A modern static solution must include sophisticated checks for clock domain crossings (CDCs), reset domain crossings (RDCs), and low-power design structures. With such a rich array of analysis, around 10% of all bugs found for a typical project are detected and fixed in this stage. The main issue with static tools is that they tend to be noisy, often reporting thousands of potential violations. Some of these can be resolved by fine-tuning input settings, such as selecting which violations are reported as errors and which are only warnings. However, the main cause of the noisiness is that a single underlying design flaw can lead to many related violations. For example, a bug in the clocking logic might lead to reported violations for every flip-flop using that clock. For debug efficiency, the designer needs to be able to focus on unique issues that, once fixed, will resolve multiple violations.

Root-cause analysis (RCA) is the AI-enabled technique to meet this requirement. As shown in Figure 3, the first step in this divide-and-conquer process is to cluster the violations based upon common signatures using ML methods. In the example just mentioned, all violations related to the clocking logic bug would be consolidated into a single cluster. The goal is for the designer to look at just one violation in each cluster and make a fix that will resolve all the violations in the cluster. Thousands of violations are typically reduced to dozens of clusters, saving designers a great deal of time and effort. Since clustering uses unsupervised ML algorithms, it runs automatically with no user guidance required. It identifies cause-effect relationships for the violations, learning as the project progresses, and performs RCA, further reducing the burden on the project team. A combination of graphical displays and ready-to-use Tcl scripts guides the designers to the root cause of the violations in each cluster.

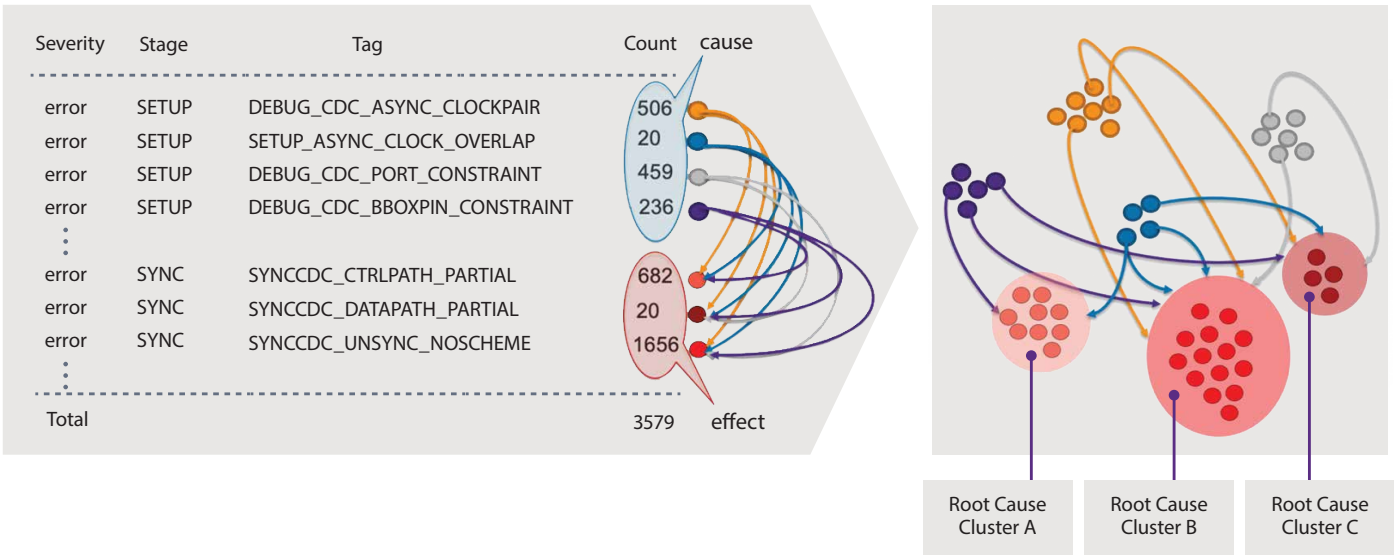

Figure 3: Violation clustering in static verification

The Synopsys VC SpyGlass® platform, the industry's leading static verification solution, and Synopsys VC LP, a comprehensive static low-power verification solution, provide all these capabilities and more. SmartGroup technology performs advanced clustering to dramatically reduce the number of violations to be examined and RCA accelerates the debug of each cluster. The suggested corrective action for a violation might be a change in the RTL design, but often it will be a refinement or addition to a constraints file. The combination of ML-based clustering and RCA yields up to 10X debug efficiency improvement for static verification on a typical chip project.

AI Technologies Applied to Formal Verification

Although testbench-based simulation is the best known and most widely used verification technique, it has its limitations. A bug in the design can't be detected and fixed unless the simulation stimulus triggers it and changes the expected results in an observable way. Thus, there is always a chance that some bugs escape because they were never activated, or their effects were never checked. Formal verification offers comprehensive analysis of the design against a set of properties. No stimulus is needed; the analysis considers every possible legal input sequence. This can detect deep bugs that are difficult to stimulate in simulation, typically finding about 20% of total bugs. Formal verification can also prove that no further bugs exist for a given property, providing a level of confidence with no equivalent in simulation or emulation.

Formal verification tools often have dozens of solvers or engines under the hood to tackle the hundreds or thousands of properties that need to be verified in the design. The performance of a formal verification tool and the size of the design it can handle depend on the performance of the engines and the orchestration of the engines. Over the last 20 years, there have been many advances in formal technology. Recently, AI and ML techniques have been adopted with great success in improving convergence and performance. Synopsys VC Formal™ is the first formal verification tool in the industry to utilize ML in engine orchestration, regression, and debugging.

The goal of engine orchestration is to assign the right engine to the right property to achieve the shortest runtime and best convergence, within the physical compute resource and time constraints. VC Formal uses reinforcement learning on the fly as it processes each property, learning from what worked and what didn't to guide orchestration on the next set of actions. This is called Smart Strategy Selection. In addition, the decisions made for each property are saved in a database at the end of the run so that subsequent runs can leverage the learning from the previous run for even better results. This is called Regression Mode Accelerator (RMA).

When the RTL design, or the formal testbench, is revised during the design and verification process, formal verification is often run as part of the nightly or weekly regression to make sure the changes did not introduce new bugs in the design. As shown in Figure 4, each subsequent VC Formal run reads in the design and the learning database from the previous run to decide which property status can be safely maintained, which properties need to be rerun due to the changes, and which inconclusive properties should receive more time and resources from the orchestration. Even for properties that need to be reverified, the learning from the previous run guides the orchestration process, making the second run faster. Benchmark data shows that these ML techniques can result in over 10X speedup and additional convergence, reducing TTR and COR while improving QOR. Figure 5 shows an example of these benefits on a real-world design.

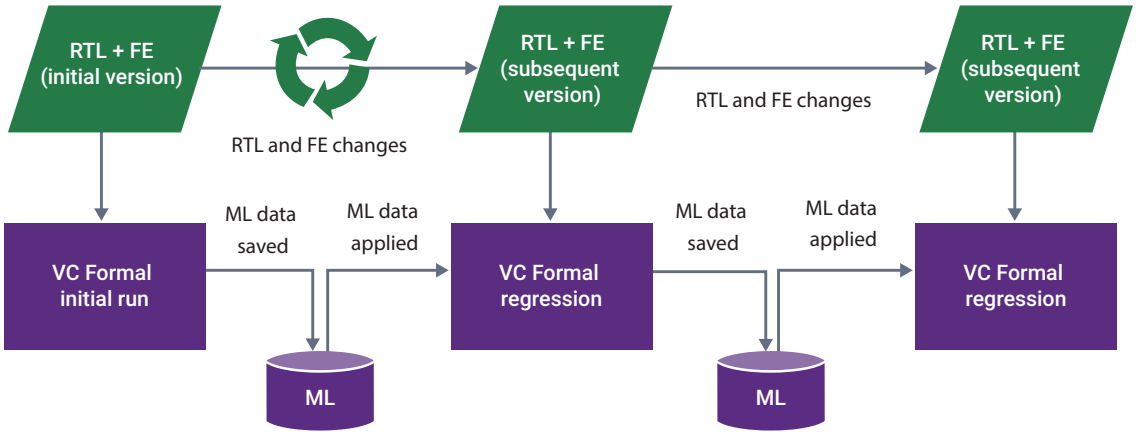

Figure 4: Regression Mode Acceleration (RMA) in VC Formal

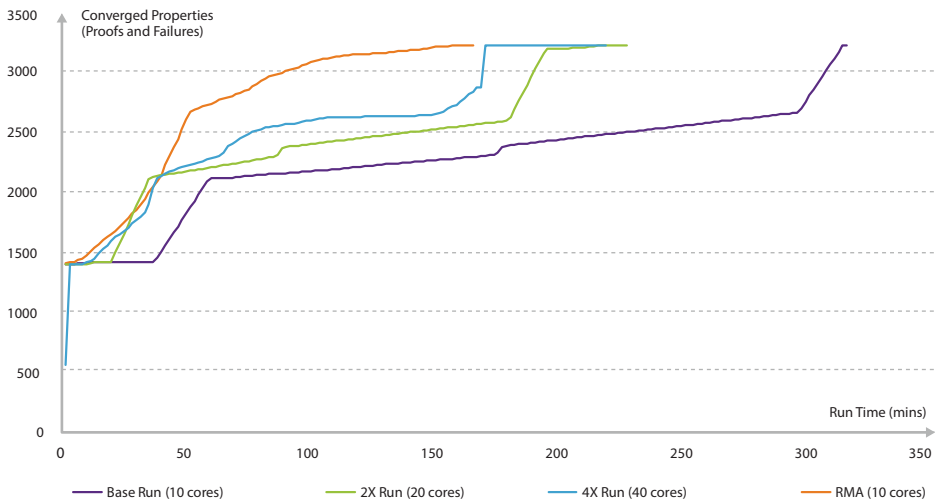

Figure 5: Faster property convergence with RMA

AI Technologies Applied to Simulation

Runtime performance is also important for simulation regressions, which may contain thousands of tests and run every night to verify all code changes to the RTL design and the testbench. Simulation remains at the core of chip verification, typically accounting for around 65% of the total bugs found. Sometimes the changes made to the design to fix bugs don't work properly or they introduce new problems, so frequent regressions are vital to detect issues quickly and keep the project on track. There are several factors affecting simulation and regression performance, with at least two of them amenable to AI-based improvements. The first is the setup for the simulation and regression runs. Modern simulators have numerous options and switches that can have a significant impact on performance. It takes time and expertise for verification engineers to optimize the simulator settings for a particular design and testbench. As the code evolves, some tweaking to the settings may be needed to maintain optimal performance. An automated process using ML to learn and maintain simulator options and switches can greatly improve regression performance and efficiency.

The Dynamic Performance Optimization (DPO) technology in the Synopsys VCS® simulator is an application of AI to simulation performance. It uses ML and rule-based AI techniques to learn from prior regression runs and automatically tune VCS settings for optimal performance, as shown in Figure 6. This process is automatic, so no user input is required. Verification engineers have the ability to control some aspects of the flow if they choose to do so. For example, they might turn on learning mode only every few days, and in between run regressions with the same settings. DPO generally results in 1.3-2X faster simulation runs than manually experimenting with simulator settings.

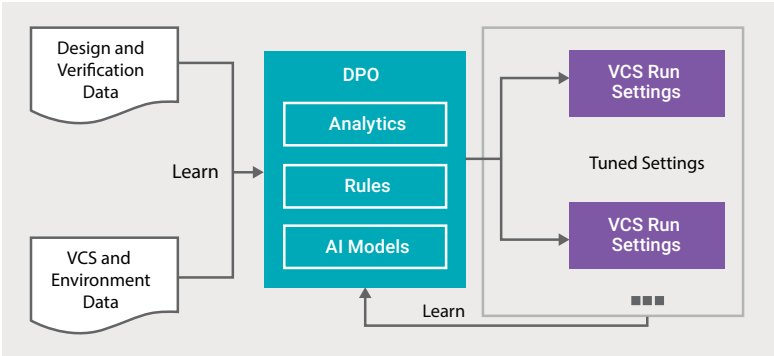

Figure 6: Dynamic Performance Optimization (DPO) in VCS

The biggest impact on overall regression performance is the time required for coverage closure. Historically, verification engineers reviewed simulation reports to identify unreached coverage and then modified tests or wrote new ones to try to cover the missing parts of the design. With constrained-random testbenches, they are more likely to modify testbench constraints to focus automatic stimulus generation toward missing coverage, but this is still a largely manual effort. A great deal of time can be wasted duplicating coverage already achieved. Improving this situation is another application for AI/ML in the verification process, and VCS provides a solution. Intelligent Coverage Optimization (ICO) optimizes the statistical quality of constrained-random stimulus and delivers insight into testing issues impacting coverage. On recent chip projects, ICO has been shown to accelerate coverage convergence by 2-3X. Verification teams can achieve higher total coverage in less time, shortening the schedule and saving resources.

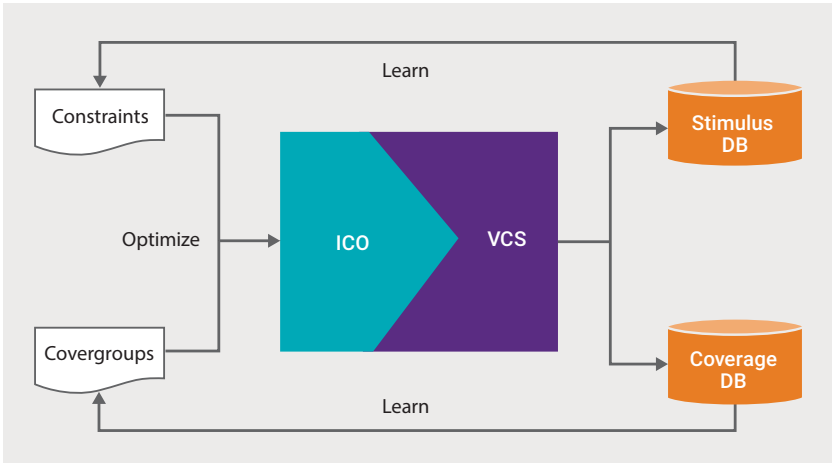

Figure 7: Intelligent Coverage Optimization (ICO) in VCS

AI Technologies Applied to Debug

As mentioned earlier, simulation regressions are run countless times during a project. Every time that a regression fails, the verification team must examine the reports and debug the causes of the failures. Because the RTL and testbench (TB) code is changing constantly as bugs are fixed and as new features are added and tested, regression failures are an everyday occurrence. Manually processing daily regression failures is a huge burden on verification engineering resources. Fortunately, AI/ML again provides relief. Even though the reasons for simulation test failures are usually much more complex than for static violations, the same principle of automated RCA still applies.

The Regression Debug Automation (RDA) capabilities in Synopsys Verdi® Automated Debug System automatically bin, probe, and discover the root causes of regression failures. RDA classifies and analyzes raw regression failures using AI and identifies root causes of failures in the design and testbench. RDA reduces the time for RCA and improves the overall debug effort by 2X.

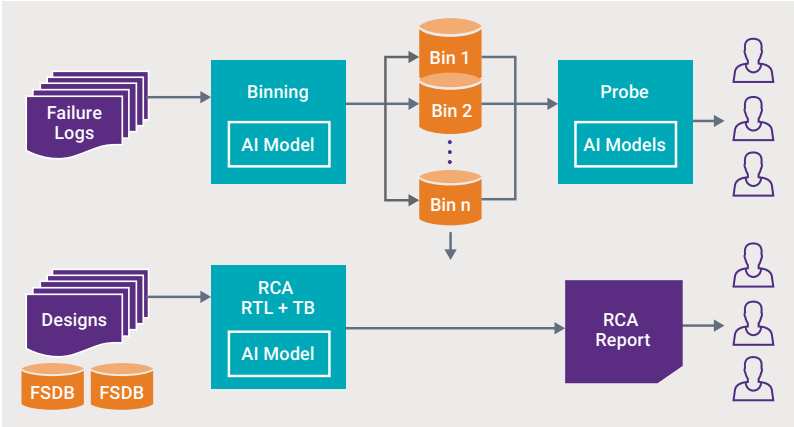

Figure 8: Regression Debug Automation (RDA) in Verdi

Conclusion

AI and ML technologies are finding more application in chip design and verification flows every day. Specifically for verification, AI/ML speeds up failure analysis for static verification, improves the performance of formal verification, makes simulation more efficient, accelerates coverage closure, and makes simulation debug faster and easier. Figure 9 summarizes the benefits of these process improvements for the verification team and the overall chip project.

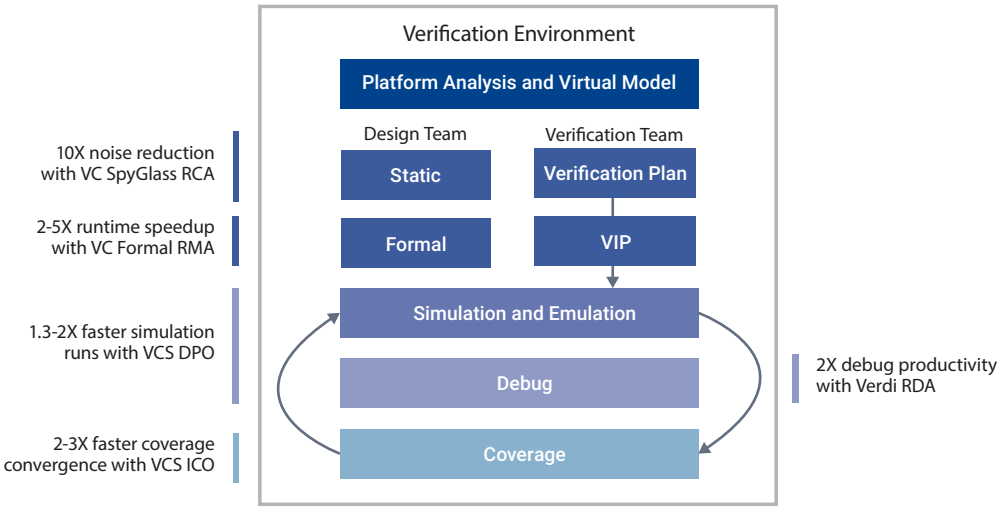

Figure 9: AI/ML benefits in the Synopsys verification flow

The Synopsys solution meets the verification team's goals and addresses their key challenges by providing better QOR, shorter TTR, and lower COR. Powerful and flexible AI/ML technologies to improve many aspects of verification at multiple points in the flow are available today.

05/20/22.CS89319133-SRG-Verification-WP.

Pub: May 2022