ERC: An exhaustive classification of false errors

Meryam Bouaziz, Application Engineer – Aniah
April 9, 2024

28 rue gustave Eiffel
38000 Grenoble
France
Phone +33 9 74 98 10 32 • E-mail contact@aniah.fr

Table of Contents

Abstract

This whitepaper presents a classification and root-cause analysis mechanism for efficient and reliable analysis of Electrical Rule Checking (ERC) results. The systematic classification of root-causes enables efficient analysis for each of the 4 identified classes.

All formal verification tools, including ERC, must reach a trade-off between "false negatives" (i.e., real design errors that are not detected) and "false positives" (or false errors, locations where errors are erroneously reported).

Aniah OneCheck ERC verification tool takes a no-compromise approach to verification coverage. It aims to avoid false negatives, which can lead to silicon bugs, at all costs.

An effective ERC verification tool must reduce the burden of false positive analysis to a level that allows for wide deployment within chip design teams. The key to achieving this result lies in an exhaustive classification of false positives, along with clustering by a few, massively replicated root causes. This work details the classes of false errors and corresponding root-cause analysis mechanisms.

ERC verification is Formal Analysis for transistors and circuits

Nowadays, the complexity of circuits reaches unprecedented levels, largely due to technological advancements that enable the integration of more and more functions (such as IP, standard cells, and chiplets) into a single chip.

Electrical Rule Checking (ERC) encompasses both standard (industry-wide applicable) and specific rules. To achieve high-quality integrated circuit (IC) design, verifying a comprehensive set of ERC rules is essential. This set includes rules for both analog and digital circuitry verification.

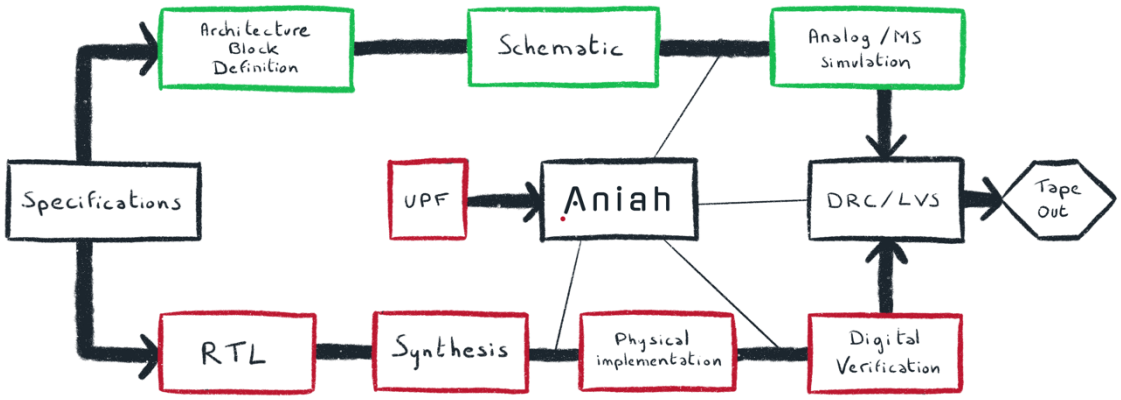


Figure 1: Flow diagram of ERC detection in IC design flows

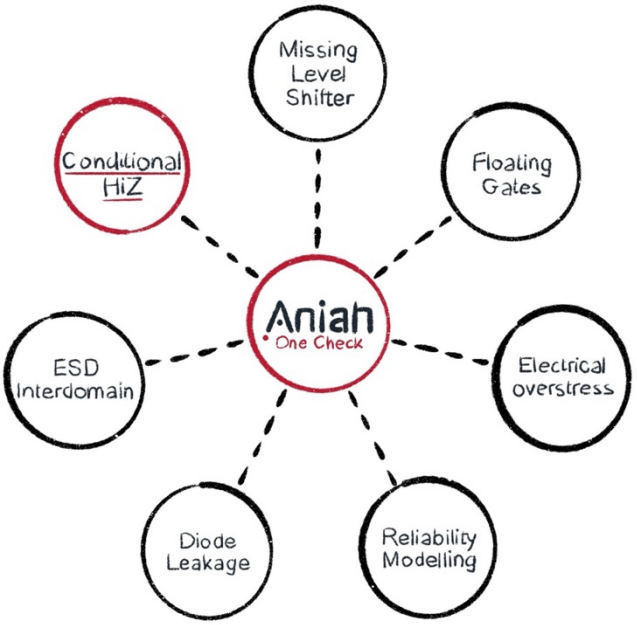
ERC ensures the robustness of designs at the analog and mixed-circuit levels, safeguarding against circuit design rules violations. Within formal circuit checkers, ERC can detect issues such as missing level shifters, floating gates or bulks, and diode leakage, among others.

This capability underscores the importance of ERC in identifying potential design flaws that could compromise the functionality or reliability of the final product.

Introduction to some of the most common design flaws

Conditional HIZ
This term represents a high-impedance state often encountered in digital circuits, where the node is not actively driven to either a logic high or logic low state.

Floating Gates
These are physically or electrically unconnected nets, which can lead to unpredictable circuit behavior.



Floating Bulk
This condition occurs when the bulk terminal of a MOSFET is not connected to a fixed voltage reference, potentially affecting the transistor's operation.

Diode Leakage
Refers to an undesired current flow through a diode, particularly under reverse bias conditions, even when it should ideally block current.

Electrical Overstress
Occurs when the voltage across the terminals of a device exceeds its absolute maximum ratings, risking damage to the device.

Missing Level Shifter
This issue refers to the improper usage or absence of a level shifter between power domains, specifically when a signal crosses voltage domains without an appropriate isolation cell.

Figure 2: Aniah OneCheck ERC Capabilities

What does a “false positive”, or false error mean in the context of ERC?

Electrical rules apply to topological structures rather than single device/pin checks, introducing a high degree of variability. These rules may encompass both geometrical and electrical characteristics.

With common tools, it is widely common to detect “errors by an ERC review” such as short circuits, open circuits, floating nodes. ERC thus represents a form of formal verification that navigates the subtle balance between identifying true and false errors, a central theme of this whitepaper.

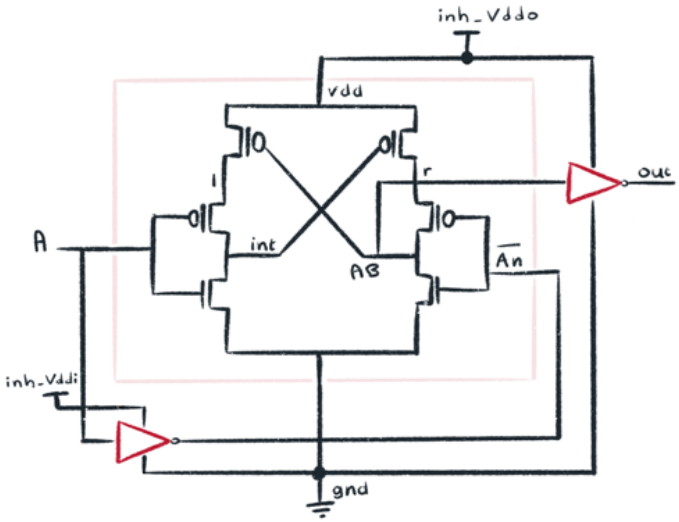


Figure 3: A false positive example

The terms “Positive error” and “negative error” are commonly encountered in the realm of ERC verification. Their interpretation can vary depending on the tools used and the specific conditions applied. It is essential to approach these errors with a keen awareness of their potential impact.

But what does this mean? The terms positive and negative relate to the result of hypothesis or assertion, which leads to the “positive error” referring to the hypothesis being true and “negative error” means it hypothesis were false. This could be a benefic or not as information, only the designer, product architect, … could decide whether it is true or false.

To verify the correctness of analog circuits including Integrated circuits, we always refer to formal verification for its consistency and for the mathematical approach. It also operates by checking all the possible input scenarios to ensure that the design is compatible with the specifications. Formal verification, unlike functional simulations, which are based on a reduced number of scenarios, aims to provide a full coverage for all possible input combinations based on mathematical proof, thereafter, errors results are a mix between false and true errors.

Cross-Coupled LS

The voltage interval (range between low and high voltages) is an approximation and introduces false errors, and the P-Mos that ensures translation will leak (see how cross-coupled works in LS).

A special focus on “false error” or as best known “positive error” that are classified into four sections: Topology specific, Analog path, Impossible path “logically” & Inverter in XXX, that could be good or bad positive and it depends on several aspect, as: conditions applied in the design, conditions waived, design violation and many others …

Classification of false errors commonly encountered in ERC

As already mentioned, “false error” represents a significant rate in "error detection" analysis, so in this case, to better understand them, let’s classify them into 4 classes with detailed examples.

Type 1 - Topology specific

Also described as “not an error” because the condition is violated as predicted by the propagation.

Example 1: Missing level shifter

The transistors of the LS that do the translation are reported ➔ it cannot leak.

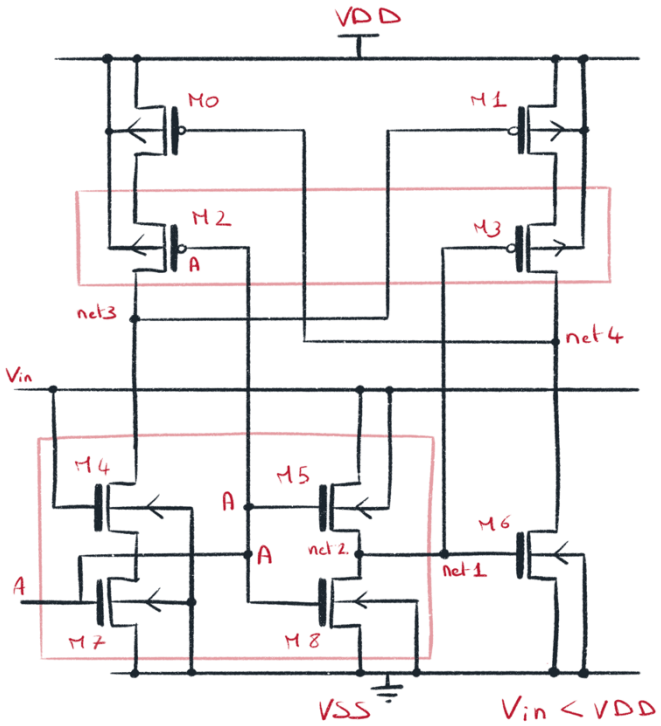


Figure 4: Missing level shifter

- M2 and M3 are flagged because they receive a voltage from the M4-M7 and M5-M8 inverters, Vin, that is below their own, VDD.
- The transistors cannot be properly shut, but there is no leakage because of the feedback on M0, M1

Example 2: bulk leakage “common bulk in p-mos”

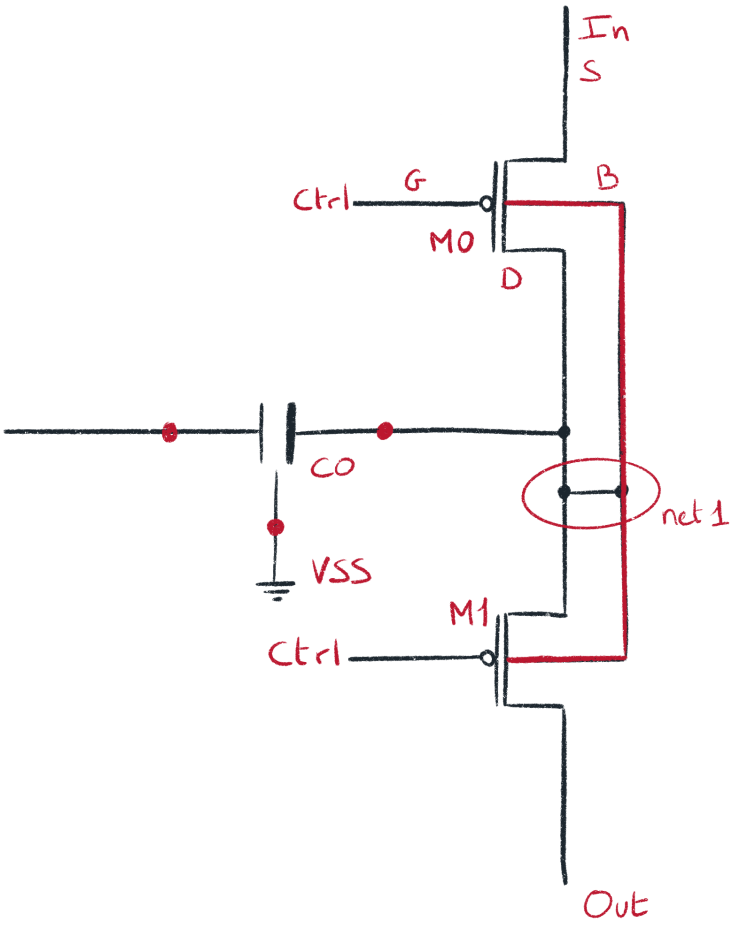


Figure 5: Bulk leakage

- Two p-mos share and isolate the bulk.
- There’s a possible leakage from M0 source and M1 drain (D/S). This leakage, however, raises the bulk voltage until the leakage stops.
- It must be checked that there is no electrical path out of the bulk net “B and D can be connected if the net is not connected to anything else.”

Type 2 - Analog path

The analog path error arises from the limitations of interval-based voltage modeling. Taking the example of OX1 (thin oxide device) and in a case of electrical overstress error detection: The voltage value is an equilibrium, which means that there is a quiescent current, in other words the path is made up of resistors and transistors with a limited Vds.

Example 1: Differential amplifier

➔ This structure comes in many variants.

- M1 and M2 must be biased properly to be used as amplifiers. That requires a current to flow through M3 to M1 and M2.
- There's a voltage drop along the current path which may be such that thin oxide device can be used at some point. Taking an example: Spice return the exact value of VDD at 1.1V and the voltage range returns 3.3v while the MOS only supports 1.1v; in this case if we take voltage range, we can notice that it's not accurate even if simulation proves otherwise.

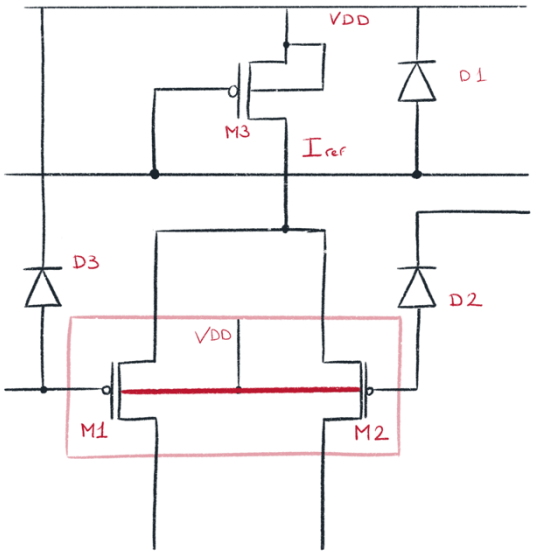


Figure 6: Differential amplifier

Example 2: Simple current mirror

We consider in this case that the voltage drop on series devices is not considered.
To explain this case in a simple manner:

- It's a differential amplifier with a simple current mirror.

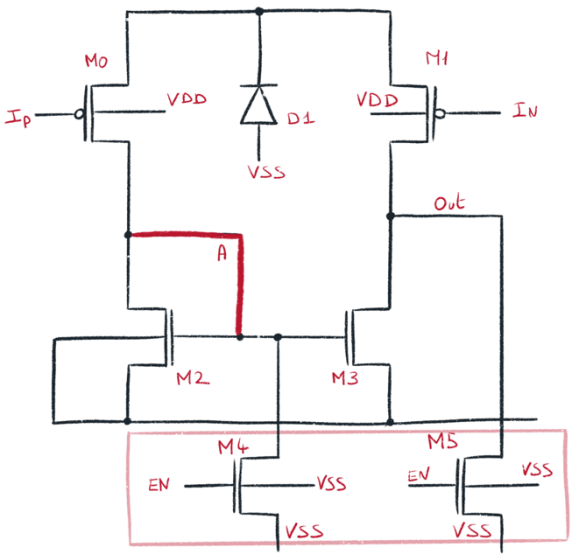


Figure 7: Simple current mirror

Type 3 - Impossible path “logically”

It is interesting to know that for this type of error, the propagation is correct, but the error cannot occur due to a combination of signal values on transistor gates along the path. Ex: test multiplexer

Example 1: Test multiplexer

- Every “Macro” core contains a switch.
- The switches on each level of the tree “Analog testmux” are mutually exclusive. The propagation ignores it and propagates the highest voltage in the tree everywhere.
- This is a global view of “Analog testmux structure” or as we can mention it as a “multiplexer test”, with 4 main input multiplexers on the left and a 1 output, internal and external, also multiplexed, on the right.
- Here’s more in detail, to see how this structure propagates.
- Core X = refers to Switches / MLS is “missing level shifter error”

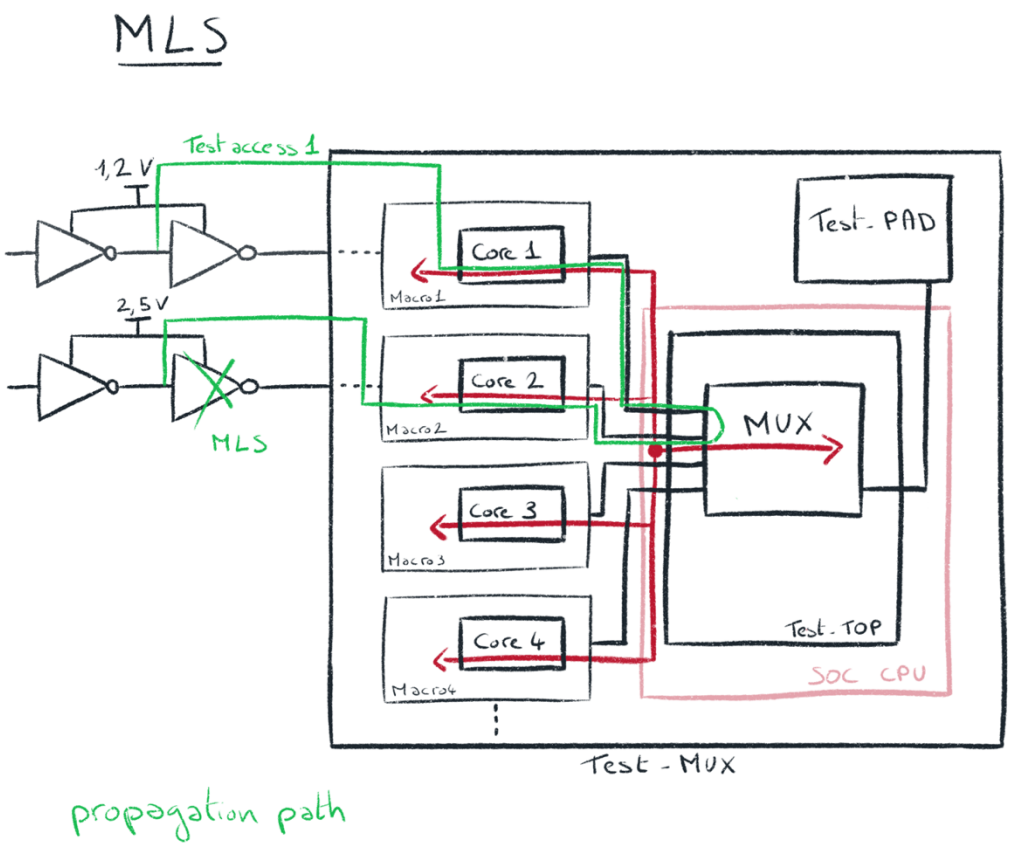


Figure 8: Test multiplexer

- The propagation from the input of inverters powered by “1.2V” voltage through “test access1”, core 1 and the MUX, then through Core 2, to inverters powered by “2.5v” will lead to a missing level shifter error (cross domains)

- The red arrow shows the analog testmux tree structure root-to-branch

Example 2: Inverter with locally switched power

To understand more the phenomena of "impossible logically path", let's take another example. Here is an inverter, driven by a lower range signal. When this signal is '1', a pdn pulls-down the inverter output and cuts its supply.

- Even though V(D) < V(V1), there won't be a leakage in the M1, M2 inverter because M3 cuts the path when the inverter receives a logical '1'.

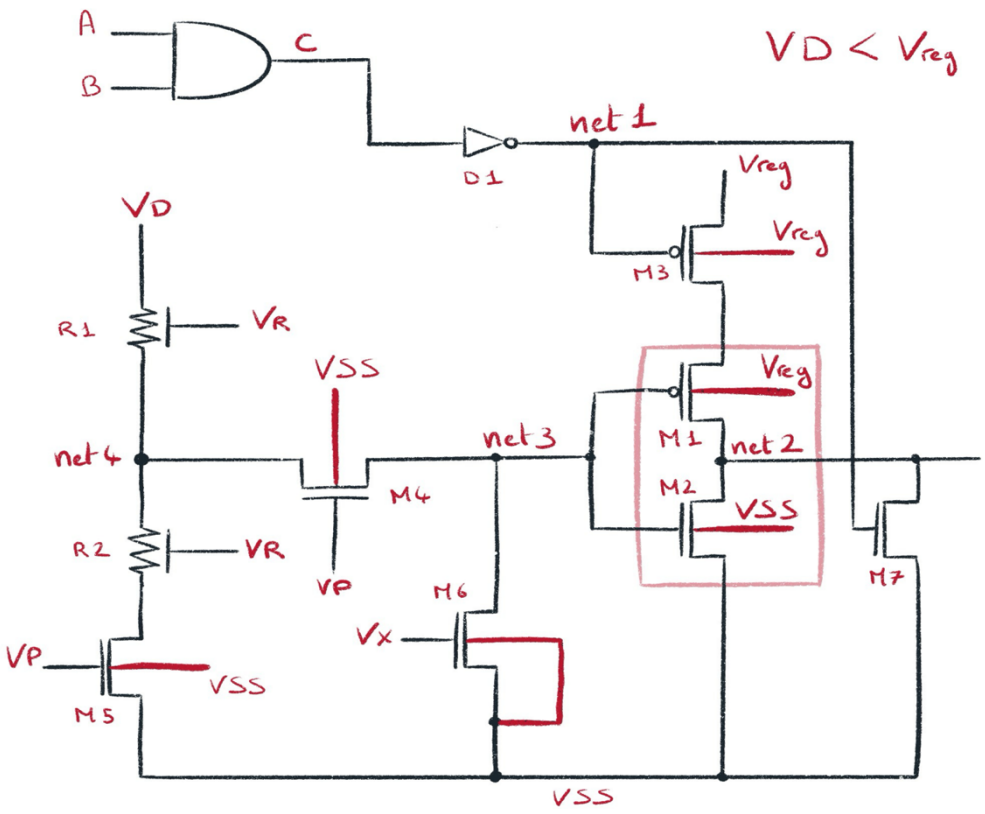


Figure 9: Inverter with locally switched power

Type 4 - Missing supply in setup

A partial set-up, with power supplies not defined, may lead to massive numbers of false errors. The propagation from other supplies will spread across all circuitries connected to the undefined power supply.

Example: Ring oscillator

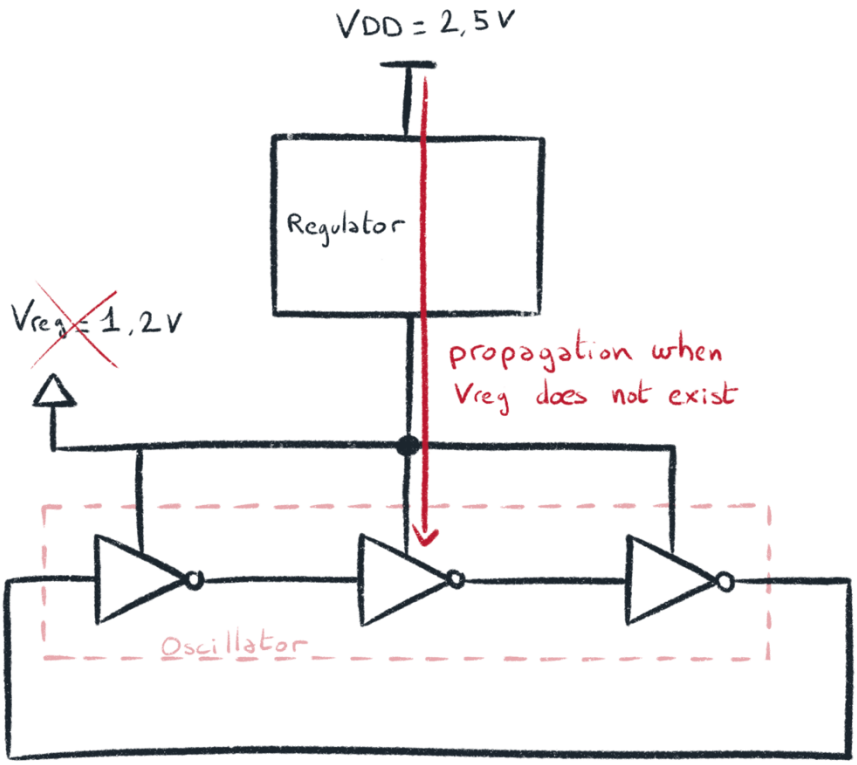


Figure 10: Ring oscillator

- The primary power supply here is Vreg = 2.5V, that will drive and start the regulator
- Devices used in the oscillator are thin oxide devices.
- If Vreg is not setup, then VDD will propagate directly to the thin oxide inverters of the oscillators, creating electrical overstress errors.

Aniah OneCheck root-cause analysis for all types of false errors

In addition to optimizing the trade-off between false positives and false negatives, Aniah OneCheck implements an efficient Root-Cause analysis methodology to Cluster false errors by (massively) common root-causes for all 4 types of false errors above-mentioned.

About Aniah

Aniah is a formal circuit checker, that verifies the IC design database against structural errors - violations of the circuit design good practices. It provides an unparalleled coverage:

- Vector less: all circuit states are verified in a single run with no need for stimulus to cover all possible circuit states. That extends to the verification of 100% of the power states of the circuit.

- A reliable flow: Aniah's flow has been built around the errors to provide critical feedback on coverage loss that may occur from the setup.

- Reliable, proven errors check: the errors check methodology has been validated on several circuits' complexity from major IC manufacturers.

In addition to targeted errors elimination, Aniah improves projects efficiency:

- Run early and debug fast: Aniah can be run from the earliest stages of the design before simulations and guide system-level integration.

- Continuous integration: Aniah checks design updates continuously and reports new errors without delay thanks to runtime in seconds on mixed-signals circuits with 10s million transistors.

- Predictable sign-off: the evolution of the errors reported by Aniah has proven to be a reliable metric of a project's maturity and the convergence towards sign-off.

Systematic Root Cause Analysis of False Positives

Now that we've identified the types of errors and their root-causes, it's time to find out how they can be resolved. Thanks to the intelligence of our Aniah OneCheck tool, we can distinguish whether the error is “positive” or “negative”.

Solution type 1: Topology specific

Multiple solutions have been proposed to solve topology-specific errors:

- Filtering based on errors clustered by cell name. This is highly efficient, with 10-30 cells to filter out on a typical mixed-signal IC. The reliability of this filtering is good, but requires the user to understand the underlying formal analysis methodology

- Identification of topologies with pattern-matching. This widely-used solution is an effective way to unburden users from identifying the topology and filtering the cells. Its reliability is challenged by the variability of topologies (it may filter real errors) and the burden it places on the engineers responsible for maintaining the database of topologies, which, because of the high count of variants, may contain thousands of elements and require constant updates

- Pseudo-electrical analysis to refine the interval-based error detection mechanism with a more accurate modelling of voltages and currents. This methodology is more complex, but has shown high reliability and has no maintenance cost.

Aniah OneCheck implements the latter methodology to detect topology-related false errors in missing levelshifters checks

Solution type 2: Analog path

Analog path rejection shares the same challenge as topology-specific rejection above-mentionned, with the added complexity of a possibly higher-scale, as analog path often span IC-wide, as central current references serve all IPs.

Aniah OneCheck enables users to efficiently identify, and filter false errors connected with any current (or voltage) reference net. Further developments in our tool will extend the capacity of our pseudo-electrical analysis to such cases.

Solution type 3: Impossible path “logically”

This type of false errors often forms the bulk of all false errors. Aniah OneCheck identifies the tree-like structure of analog testmuxes and provides elements for users to reject thousands of false errors in just a few clicks.

Solution type 4: Missing “supply” in setup

Aniah OneCheck clusters all errors related to a missing supply together, for an immediate analysis and update of the power supply set-up.

Conclusion

Electrical Rules Checking is essential for detecting circuits malfunctions and failure, but the debug of large numbers of false errors has limited deployment of such tools with chip design teams.

Chip design teams face huge challenges when grappling with false errors generated during ERC verification. These errors not only consume valuable time and resources but also undermine confidence in the accuracy of the verification process. As a result, the potential benefits of ERC, such as detecting circuit malfunctions and failures, are often overshadowed by the overwhelming task of false error debugging.

Considering these challenges, the importance to find a solution that tackles the issue of false errors in ERC verification becomes evident.

Aniah OneCheck emerges as a game-changer in this arena, offering a Smart Clustering approach that effectively minimizes the burden of false errors. By enabling reliable, large-scale deployment of ERC tools, Aniah empowers chip design teams to overcome the obstacles posed by false errors and unlock the full potential of error-free chip design.

Feedback from the readers is always welcome.

Have you ever experienced or have struggled with a similar experience in ERC detection errors? We'll be pleased to share your experience also your feedback, comments, detection tools and problems about the subject

Contact mail: contact@aniah.fr or visit us at www.aniah.fr or linkedin.com/aniah