2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Accelerating Silicon Bug Detection and Optimizing Execution Flow through Intelligent Adaptive Glitch Detectors in AMS Verification

Aadhar Sharma, Avinash Chaudhary, Bhavya Shah, Saurabh Kandpal, Sooraj Sekhar
Embedded Processing, Texas Instruments (India) Pvt. Ltd., Bangalore, India
(*a-sharma4, avchau, b-shah, s-kandpal, sooraj.sekhar@ti.com*)

***Abstract*- The paper presents a methodology aimed at streamlining verification processes by leveraging compact, generic Verilog-AMS based assertion and checker modules. These modules can be flexibly interconnected to construct sophisticated checkers tailored for diverse circuit designs. Proposed approach facilitates comprehensive validation of IP/design integration at higher abstraction levels and automates module verification within the SoC context. This reduces the need for manual SoC-level checks, focusing them on specific system scenarios that may have been overlooked during module-level verification. Notably, our adaptive checkers efficiently support mixed-signal verification environments encompassing both SPICE circuits and analog behavioral models (ABMOD). This makes the design self-checking for verification purposes thereby improving the Analog and Mixed Signal (AMS) verification removing the need for manual waveform reviews and providing confidence to the SoC verification by identifying analog bugs which might reach to silicon.**

I. Introduction

The increase in control requirements, computational performance, and number of intricate applications to be implemented have led to a rise in the complexity of SoC design. The highly constrained specification presents several new challenges. Balancing power, performance, and form-factor to meet expectations has become a complex optimization problem. High-performance computing (HPC) applications like cloud infrastructure and data data-driven applications like machine learning training focus on performing tasks at very high speeds, so performance optimization is prioritized. On the contrary, in hard-to-access industrial embedded systems, battery-operated Internet-of-things, power optimization to increase the life of operation is given utmost priority. With recent advancements in Artificial Intelligence-of-Things (AIoT) edge devices, priorities are shifting toward power-efficient communication, inference, and retraining of AI-enabled devices. The strategies mentioned below are widely adopted in SoC design for low-power applications.

1) Dynamic Voltage and Frequency Scaling
2) Clock and power gating
3) Defining Power domains and Power Modes
4) Multi-Threshold design

By adding multiple clocks to the design, one gets the freedom to choose the operating frequency of different IPs in accordance of their sensitivity; for example, if a specific IP communicates externally at a lower frequency than the internal SoC, the IP can be clocked at a lower frequency, which can communicate internally through a synchronizer interface for clock domain crossing. Similarly, various IPs can operate at different voltage levels and can be interfaced with each other via level shifters for power domain crossing. If a part of the SoC is not in use, clock can cut off to that domain. If the recovery time for the system is inconsequential to applications, then the power supply to the entire power domain can be cut off. The gains in power efficiency are justified regardless of the implementation of additional circuitry, such as level shifters, synchronizers, and retention logic to enable the features mentioned above. The introduction of all or a combination of these features adds a layer of complexity to the verification of the SoC design [1]. For example, specific scenarios of the frequency scaling feature, which may require switching reference to the clocking circuitry, cannot be modeled and verified using purely digital simulations.

Analog and Mixed Signal simulations are to be carried out to model silicon behavior with acceptable accuracy. AMS simulations are compute-intensive and consume significant time per test case depending upon the complexity of the design; hence, they are not widely implemented. Time-to-market constraints can limit quality SoC verification, which can be improved by expediting the regression. A substantial portion of the issues caught by performing AMS simulations in the test cases are related to the kickback glitches or some functional failure that happens due to these glitches. In AMS regression, the main focus of verification is the SoC-level interaction of different IPs and whether there is some anomalous behavior breaching specification limits due to the loading effect. In low-power circuits wherein, one cuts off the power to IPs to save power causes a kickback on the voltage rail when the power supply is re-enabled. Consider the scenario represented in Figure 1, a SoC containing 2 IPs present in different power domains shown. The IP2 is power gated in certain power mode, now consuming power from a voltage stored in the capacitor. When we switch the power model and connect the IP2 to the supply, there is a kick back observed. If this source is used as a reference somewhere in IP1/2, it can cause sampling of wrong data or even cause metastability conditions. Similarly, the kickback glitches on the supply rails can unwantedly assert a brownout reset leading to failures. AMS runs have a capability to show these issues which need to be found as soon as possible.

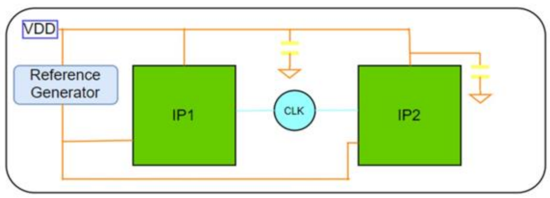

Figure 1: Potential Issue found in AMS runs.

Through the time-consuming AMS simulation, we can model the silicon behavior such that we can see the kickback glitches that would have been caught in post-silicon simulation. Finding these critical issues at such later stage comes with its own challenges to fix them. Debugging such design issues through AMS simulation can be advantageous and can save time in long run. In an SoC, thousands of nets are being probed, and manually checking for glitches on each one of these nets is very difficult and time-consuming. Moreover, the glitch profile is such that it may not be visible without zooming in on the waveform in its neighboring region. Manually checking for glitches for all the signals by zooming at every instant is a very laborious task and can lead to missing of certain bugs.

Figure 2: Illustration of possible issue aliasing due to manual review

As seen in Figure 2, the zoomed-out version, the anomalies are not evident, and only by zooming in can one know if it violates the specifications. By performing Manual waveform reviews, there is a possibility of missing issues. Existing assertion-based checkers for mixed signal verification [2]-[4] require various references which have deviations depending upon the conditions. Hence, there was a need to build a methodology for glitch detection that ensures scalability and deployability to improve the AMS quality. Motivated by this, we created checkers to aid the manual waveform review process in order to improve the scope and quality of the AMS verification.

II. Proposed Solution

A. *Adaptive-Magnitude analog glitch detector*

Checker in the proposed format can be used for detecting any kickback glitches on the voltage (power supply and reference current) rails. The low power design practices adopted to meet highly constrained specification can cause design failures that arise due to the non-idealities present in the silicon, and hence must be found as early as possible through AMS simulations which can be facilitated through this checker.

B. Adaptive-Frequency oscillator glitch detector

Adaptive-Frequency oscillator glitch detector calculates and monitors the frequency variation in the oscillators throughout the simulation irrespective of frequency range, temperature and power modes and flags any anomalous change which may cause the system to miss the timing requirements. Using the oscillator glitch detector, one can easily find co-existence issues with any signal which affects the clock frequency that may arise due to non-idealities like coupling paths or any design related issues.

C. Automated integration of checkers in testbench for accelerated bring-up

Python based automation to integrate the proposed analog checkers with parameter input from designer specification is used to fortify the waveform review process. User need to prepare an assertion-spec file with minimal info such as signal name and glitch threshold and with that a python script will generate Verilog-AMS (VAMS) assertion file that can be directly incorporated into the AMS/DMS testbench.

Figure 3: Proposed methodology flow

III. Methodology of Glitch Detection

A. Principle

Model creates a nominal level by computing the finite impulse response (FIR) of the signal. This allows the user to deploy checkers on the signal without holding any prior knowledge about its function and operating level. Due to this strategy, the model becomes invariant for different testcases, process, voltage and temperature (PVT) conditions, ensuring exhaustive deployment, unlike reference-based glitch checkers, which require complete design information and 100s of checkers per signal for different PVT scenarios [5].

B. Sampling

Sampling for the signal is performed at the frequency of analog solver which can satisfy the Nyquist sampling criteria for any glitch width. Instead of sampling the signal at a constant high-frequency clock, which will add a limit on the glitch width that could be found, the signal is sampled on a dynamic frequency clock as shown in Figure 4. As the activity in the signal increases, the sampling frequency also increases, ensuring we find glitches of any width.

Figure 4: Sampling clock with dynamically changing frequency

C. Data Type Conversion

As shown in Figure 5, each signal for which checker is added will be passed through a datatype converter, which automatically recognizes and converts it appropriately so that the checkers can perform algorithmic computations. In AMS simulations, different combinations of models (SPICE/ABMOD) are used in a testcase [6]. Tool ensures easy adoption by passing each signal through datatype converter, which identifies the datatype and performs conversion. Due to this, one has to setup the checkers only once at the start of the regression as no updates are needed for varying design models (SPICE/ABMOD) for IPs in different testcases.

D. Anomaly Characterization

The difference of input signal and the FIR response of the same signal has a characteristic ripple like pattern as shown in Figure 5. Anomalies inferred as glitch by analyzing magnitude and timing information of the pattern.

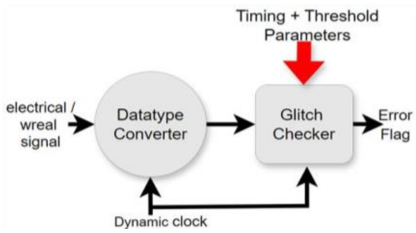

Figure 5: Abstract view for flow of information in proposed setup

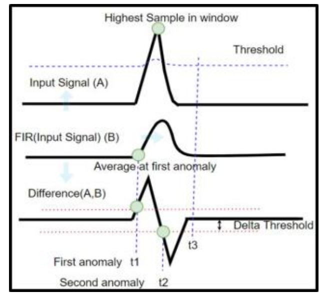

Figure 6: Illustration of working of the glitch checkers

E. *Working of the glitch checker*

The datatype converter identifies the type of the signal and converts all the electrical signals into *wreal* using the system task, which then can be passed to the System Verilog (SV) model of the glitch checker represented as Input Signal (A) in Figure 6. A moving average which is an inertial version of the input signal is computed by the checker by taking a FIR of signal (A) on a dynamic clock. The checker ensures the detection of two consecutive anomalies by checking that the difference is greater than the delta threshold, which is set at 0.005 percent of the input signal at that instant. The instant when the difference crosses the delta threshold for the first time is marked, and the average value at that instant is stored in the *t1* variable, which is used in final thresholding. The timeout window for finding the second anomaly and marking it with *t2* starts after finding the first anomaly for further consideration as glitch is set as 500 ns and can be configured depending on the maximum glitch width one considers as harmful. Each new sample is additionally compared to find the highest and lowest sample magnitude in the time window. Finally, at the end of the timeout window, if the deviation of the highest/lowest sample in timeout window from the average at the first anomaly is greater than the threshold, then we consider this anomaly as a glitch provided a second anomaly is also detected within this timeout window.

IV. Strategic Features of Optimizations in Checkers

A. *Filtering of glitches*

During the level shifting in the signal there are certain ripples before final settling. If the user does not want to consider the 'Kink' effect observed while changing the level of the signal as a glitch, then there is an internal tolerance limit upper/lower parameter (given as a percentage of input level) to ignore these anomalies even if they satisfy all the other conditions. Using this strategy, tool infers the signal behavior as not glitchy if at the end of the timeout, average settles within the tolerance limit from the highest/lowest sample of the timeout period.

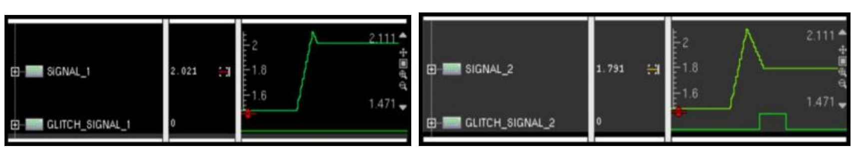

Figure 7: Working of the checkers to ignore the 'kink' effect

B. *Runtime Optimization*

During the period where any electrical signal changes the analog solver has very high activity so the signal is sampled at a very fast rate due to which delta threshold breaches are not properly observed. The lower limit on time between two samples is set to get around the oversampling issue and to save the computations. If the input is steady but if there is very high activity due to some other signal in the design the steady signal is sampled at a very fast rate. In order to gate the unwanted computations for the steady signal a lower limit on magnitude change required between two sample for computations to continue is also set.

V. STANDALONE TESTING RESULTS

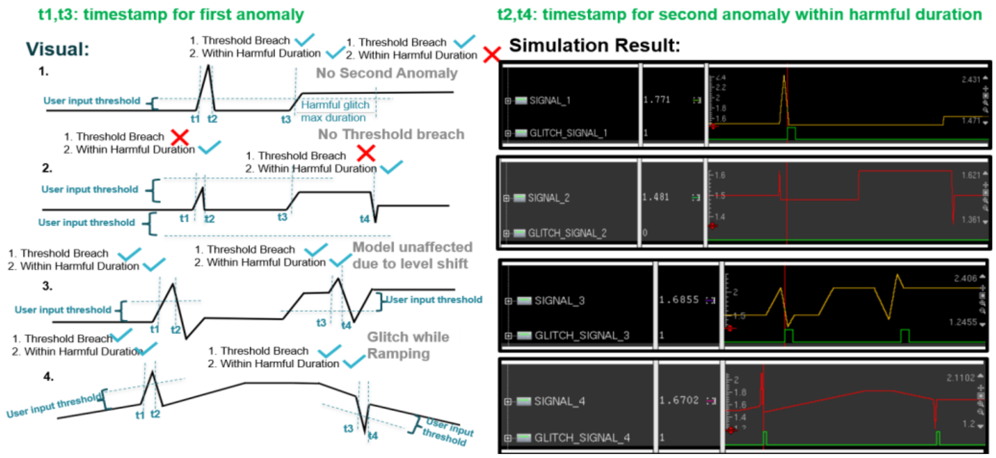

Figure 8: Standalone testing of the checkers on potential glitch scenarios

In the first visual shown in Figure 8, the signal first has a high magnitude glitch which is detected by the model as it is seen in the waveform snapshot. After a while the signal ramps to a value and stays there indefinitely which is not classified as a glitch which is expected. On the contrary, in the second visual the anomalies satisfy the time parameters but do not cross the threshold and are not flagged as glitch. For the third visual shown in Figure 8, the signal has rapid glitch like transition at first which is detected by the glitch detector model after which the moving average shifts to a higher value. The rapid high magnitude undershoot and overshoot are classified as glitch even after adding an offset to the moving average. This proves that the same model can detect the glitch at magnitude range by dynamically adapting the critical threshold to magnitude range. The proposed checkers is capable of finding glitches during ramping as indicated in visual 4.

VI. INDUSTRIAL CASE STUDY & EVIDENCE

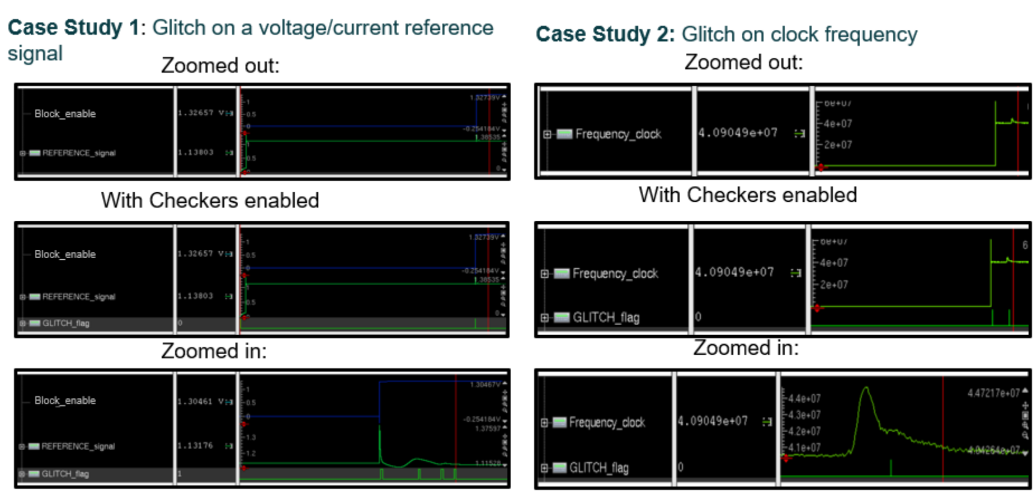

Figure 9: Category of issues found using the checkers

The first issue, represented in Figure 9 is related to the very large kickback observed on the references once the IP is enabled. For the proper functionality, few IPs require highly stable references. The checkers can efficiently find if there is a glitch, having a larger than expected kickback once the power mode of block is changed. The second issue, represented in Figure 9, shows that if there are some design issues in clock generation circuitry or if there are any anomalous variation in the frequency of the clock as a result of loading on references and other dependencies, are easily found using adaptive frequency oscillator glitch detector in AMS runs. Above mentioned are few of the categories of issues along with others like, reference coupling and co-existence issues that can be found using the proposed checkers.

VII. Overall Impact of Methodology

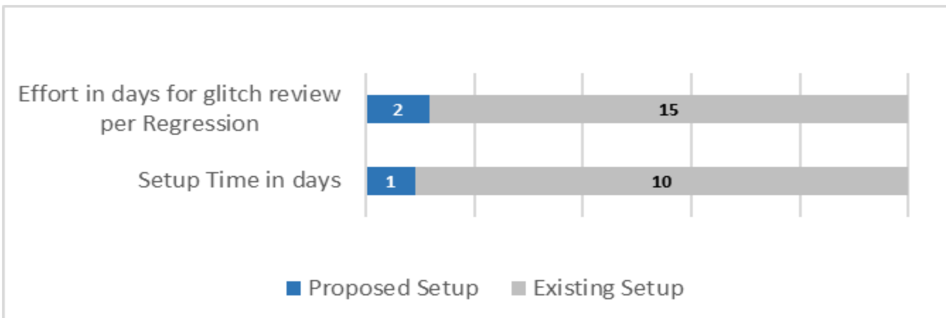

Figure 10: Efficiency Comparison of AMS verification timeline between proposed and existing setup

Due to the automation present in the insertion of checkers, the proposed methodology is highly scalable and has a very fast deployment as shown in Figure 10. The glitch flag guided debug has significantly aided the manual waveform review, has helped in faster cleaning of regression for the glitches. The strategies used in the methodology has enabled faster interception in live projects and ensures easy integration in future projects. The checker-based verification allows easy finding for one of the most prevalent issue of glitches without having any adverse effect on the runtime of the testcase.

VIII. Conclusions

This paper introduces novel glitch detector model capable of detecting kickback anomalies in signals including frequencies without prior knowledge of their properties. The checkers are meticulously crafted for seamless integration, promoting swift adoption with minimal effort by simply providing the signal name. The paper presents a methodology for comprehensive and automated deployment of the checkers for monitoring of design aspects based on user specifications. This approach enhances verification quality, facilitates manual waveform review, and expedites debugging processes and early finding of silicon bugs.

IX. References

[1] A. Nawal, and G. Jain, “Complex Low Power Verification Challenges in NextGen SoCs: Taming the Beast!” DVCon Europe 2014

[2] Lakshmanan Balasubramanian, et al, “Automated correct-by-construct methodology for RTL design and analog mixed-signal test bench generation enables early design closure of mixed-signal SoC,” DVCon India 2014

[3] Pooja Sundar, et al, “Assertion based self-checking of analog circuits for circuit verification and model validation in SPICE and co-simulation environments,” DVCon 2011

[4] Sundar, et al, “A user friendly and intuitive infrastructure that enables efficient analog mixed-signal verification using assertions based self-checking analog designs,” CDNLive India 2012

[5] Dhanasekar V, et al, “Analysis of Machine Learning Techniques for Time Domain Waveform Prediction in Analog and Mixed Signal Integrated Circuit Verification,”, ISQED 2023

[6] Aadhar Sharma, et al, “Enhancing Analog Mixed-Signal (AMS) Verification: Advanced Methods for Runtime and Scope Optimization,” DAC 2024