AUTHOR DETAILS
John Hallman
Product Manager for Trust & Security, OneSpin: a Siemens Business

1 The OneSpin 360 Trust Automation Platform App enables an objective, efficient, and repeatable security and trust assessment process

2 The OneSpin Processor Verification App ensures that an IP core implementation does everything it is supposed to do

CONTINUOUS VERIFICATION

JOHN HAMMAN looks at continuous SoC verification from pre-fabrication throughout the device lifecycle

The recent SolarWinds hacking incident that left many fortune-500 companies and US government networks exposed is a cautionary tale for unchecked software and hardware supply chain security vulnerabilities.

The highly sophisticated software supply chain attack occurred in the SolarWinds Orion IT monitoring system. Used by over 33,000 companies it monitors performance across multiple networks.

In March 2020, SolarWinds unintentionally sent a certified binary software update that included malicious code to 18,000 clients. The hacked code created a backdoor to companies' IT systems that allowed even more malware to be installed, giving the hackers the ability to spy on these companies. The backdoor communicates via HTTP to third-party servers and uses multiple blocklists to identify forensic and anti-virus tools running as processes, services, and drivers. This attack meant that highly confidential information was exposed and could have resulted in complete control of the systems being lost.

This attack confirms what is possible through software, and it is very easy to imagine how hardware and its supply chain are susceptible to comparable attack scenarios.

During pre-IC fabrication, a backdoor could be inserted at the time of design or within integrated IP. It could even occur during mask or silicon modification. After IC fabrication, malicious logic could find its way in through physical or packaging modifications, side-channel exploits and even maintenance or upgrade updates. The impact of these attacks on hardware is much more severe than software. With software, the impact can take hours or weeks to fix but is usually corrected with a software update. Resolving the hardware may require that the entire IC be redesigned and re-fabricated.

DESIGN AND VERIFICATION

Let us imagine we have a vending machine that dispenses hot coffee and another drink, which is supposed to taste like tea. The machine requires regular maintenance visits from a service technician, but senior management have run the numbers and have concluded there is a business case to upgrade the system to be "Internet connected".

Perhaps there's a need to increase service intervals through fault detection or predictive maintenance. Or to obtain better insights into customers' buying behaviours.

Many SoC designs today include smaller blocks, referred to as intellectual property (IP). These IP blocks, such as processors, on-chip communication modules, and data routing, pose an enormous attack surface and are often left unprotected or unchecked. Many IP blocks also come from external sources or 3rd party vendors to save on design costs or to leverage expertise outside the organisation. By not detecting vulnerabilities early in the design flow, they can remain dormant for an extended period of time in deployed systems, awaiting activation.

If left undetected, the costly and time-sensitive option of replacing silicon on deployed systems may be the only counter measure remaining, posing tremendous risk to the system or the mission.

Software updates may not be enough to close these gaps and therefore proactive detection methods are needed closer to potential attack points. There are many examples of well-hidden backdoors at the register transfer level (RTL), many of which come in the form of a triggering circuit. Using these circuits stealthily, an attacker may enable a change in function, such as insertion of new data or rerouting of data which now may become exposed, and an unknown function could make its way unwittingly into a design.

Functional verification processes aim to provide assurance that IPs behave as specified. Typically, detecting undocumented, hidden functions is not a primary objective. Simulation-based functional verification techniques are inherently incomplete as they rely on stimulus generation. Coding errors that are activated by long sequences of events or specific data values often remain undetected. Hardware Trojans, deliberately engineered to remain hidden,

©Andrii Symonenko - stock.adobe.com

> "IP BLOCKS, SUCH AS PROCESSORS, ON-CHIP COMMUNICATION MODULES, AND DATA ROUTING, POSE AN ENORMOUS ATTACK SURFACE AND ARE OFTEN LEFT UNPROTECTED OR UNCHECKED"

are particularly unlikely to be triggered. Formal verification tools enable exhaustive analysis of the design state space and do not rely on stimulus generation.

The use of formal provides confidence that the circuit under evaluation behaves as intended. A complete formal-based solution includes all possible scenarios during verification of RTL designs from the IP and block level to complete systems-on-chip (SoCs), as well as verification of implementation in FPGA devices.

This solution focuses on several key verification tasks:

- Agile design with early, automated design code verification
- Comprehensive, assertion-based verification with unique model-based mutation coverage measurement
- Scalable integration verification and functional analysis using automated apps

Many of these technologies are applicable for early detection of weaknesses and vulnerabilities, characteristic of hardware Trojans.

The formal methods, through use of properties and finely tuned formal assertion engines, exhaustively prove with confidence that the circuit under evaluation behaves as intended and confirms the absence of unwanted functions. Formal technology is great for security verification. Tools and methods detect common weaknesses and vulnerabilities, check structural and functional dependencies, and identify added/changed functionality, hidden instructions, and side channels.

The following are key verification technologies that can expose previously unknown hardware vulnerabilities.

AUTOMATED TRUST ASSESSMENT

The trustworthiness assessment process includes automated structural and formal analysis technology to examine an IC or IP's RTL code. The process uses the RTL model to establish confidence early in the design cycle. This early design phase offers one of the easiest entry points for an adversary to infiltrate a design with malicious code. Addressing issues at the RTL prevents their propagating to other phases of the design cycle. The confidence achieved at the RTL design phase can be maintained in subsequent design implementation steps. Technologies such as formal logic equivalence checking provide check points and verify that the design maintains its integrity through transformations such as synthesis and place and route.

Given the IP's RTL model and setup script, the automated structural and formal detection methods generate a report containing design information as well as a list of issues that could be caused by hardware weaknesses and vulnerabilities. The report includes suspect trigger structures, reliability issues, such as redundant code, and deadlock conditions. Insertion of such a trigger may happen in the update scenario as noted in the SolarWinds case and introduce a new unintended function that leaks out important data or creates a deadlock state condition to render the IC effectively inoperable. By detecting these trigger conditions, it is possible to identify the root cause early and mitigate the possibility of exploit from an attacker.

PROCESSOR INTEGRITY VERIFICATION

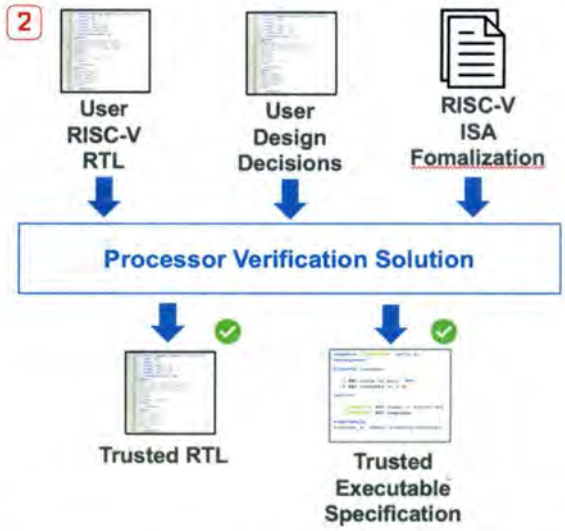
Processor integrity verification ensures that a processor core implementation does everything it's supposed to do and does not do anything it's not supposed to do. SoC designers can license a processor core having confidence that the core complies with the ISA specification, while IP vendors can support their own ecosystems and ensure that partners also comply.

Further, SoC designers can add custom features to the instruction set architecture (ISA) to support their specific applications. The technology ensures that nothing is broken as features are added and is flexible enough to verify new functionality.

In the processor verification application, formal-based functional verification techniques are extensively applied, typically using two primary modes of operation: automated formal applications, and assertion-based formal analysis targeting specific design features, functions, or behaviour. Automated formal applications detect common coding errors, protocol violations at standard interfaces, and other issues that do not directly highlight a deviation of the processor ISA.

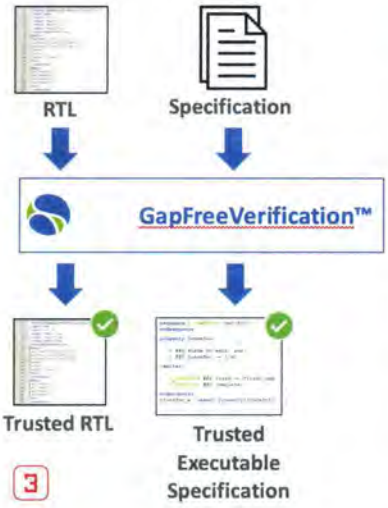
Additional assertions then target specific aspects of the microarchitecture and portions of the ISA. The method utilises Operational SystemVerilog Assertions (SVA) to formalize the processor ISA. The proof of Operational SVA is automated using commercially available hardware model checkers and does not require experience with theorem provers. The method delivers unbounded proofs of correct operation.

2



3 GapFreeVerification proves that the design functionality has been thoroughly verified and reveals any gaps or inconsistencies

4 Continuous verification



3

GAPFREEVERIFICATION

The processor verification methods, however, provide no guarantee that a set of assertions expresses the entire ISA specification. Moreover, the RTL model could also contain undocumented functions. This is where the GapFreeVerification technology is leveraged. The set of Operational SVA is analysed to detect gaps or inconsistencies, or otherwise provide a mathematical proof of its completeness. The method applies a rigorous definition of completeness that can be automatically checked.

The completeness criteria ensure that modifications to the I/O functionality of a compliant processor's RTL model, prevent the successful proof of at least one Operational SVA. In other words, the corrupted RTL model would not be compliant.

This approach was applied to the RISC-V Rocket Core design and uncovered seven issues and demonstrated how the regression verification tests uncovered a new undocumented function. We can quickly see how use of this technology prior to deployment of an update mitigates the risk of introducing new, changed, or unwanted functionality into the device, and ultimately the end product.

HARDWARE/FIRMWARE VERIFICATION

Vulnerabilities in hardware-firmware interaction also pose a high-severity security risk and are difficult to uncover and mitigate, given the requisite expertise in two knowledge domains. As an example, the boot ROM code is often synthesized directly into hardware primitives that are not as easily patchable in the field. Many of these vulnerabilities are detected during pre-silicon verification only through a combined analysis of the two components. This is challenging in many current EDA technologies, and new methods are greatly needed to address the performance, cost, and time-to-deployment demands.

This process detects low-level weaknesses in hardware-firmware interactions, including weaknesses that can only be exposed when using a cycle-accurate hardware model. It uses industry-proven formal verification technology to detect corner-case scenarios that could be leveraged to tamper with the intended firmware execution. Crucially, the process does not require in-depth knowledge of either the firmware or the hardware, or the creation of additional models. Moreover, it is highly automated, thus not requiring advanced expertise in formal verification.

The process has three key benefits: (1) the formal analysis detects corner-case issues that would be missed by methods based on simulation or emulation; (2) the process leverages commercial-grade, state-of-the-art technology with continuous support and proven on hundreds of semiconductor IP and IC development projects; and (3) the high level of automation means that users are not required to develop in-depth knowledge of the hardware, the firmware, or formal methods.

CONTINUOUS VERIFICATION

The pre-fabrication modelling of the design and verification is often an extensive and expensive process and produces an enormous amount of information. This information, though, is often left behind once the design enters fabrication, despite much being left in the lifecycle which requires verification. Here, we can employ continuous verification practices through the introduction of the 'digital twin'. The digital twin enables design models and verification originally performed prior to pre-fabrication to be combined with digital models virtualized from the physical device.

The ability to combine this original modelled data with physical dependency models with real-time data enables more system simulation and predictive analysis to improve end-to-end processes. This process also enables availability of the original automated verification methods as well as analysis solutions to be used with the most current models of the design in the system. Furthermore, if new security weaknesses or vulnerabilities become known, the model of the IC may be analysed for impact.

While not all exploits may be discovered prior to deployment, the continuous verification environment enables much faster risk assessment and mitigation action for issues that might otherwise go undetected or undiagnosed for months or even more. NE

4