Security vulnerabilities and hardware Trojans in RISC-V processors

By Sergio Marchese, Technical Marketing Manager, OneSpin Solutions

Semiconductor chips and system-on-chip (SoC) devices implemented in application-specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs) are the backbone of modern life. Today's Internet-of-Things (IoT), smartphones, communications equipment and advanced driver assistance systems (ADAS), and tomorrow's connected autonomous vehicles (CAVs), smart power-generation plants and various infrastructures all require complex electronic systems. Processor cores are often integrated into these chips to perform essential control and data-processing functions. Processors provide flexibility and programmability through their instruction set architecture (ISA), which defines the interface between hardware and software.

RISC-V ISA

RISC-V is an open-source ISA invented at the University of California. It is managed by the non-profit RISC-V Foundation, which currently counts over 300 members. RISC-V is the first open-source ISA to become a genuinely viable industrial choice for a broad range of applications. Its ecosystem of tools, software and expertise is robust and growing steadily. Many individuals and organisations have already donated open-source hardware intellectual properties (IPs) implementing the RISC-V ISA. The OpenHW Group, for example, is aiming to make a reality the long-awaited prospect of open-source hardware – particularly processor cores – for high-volume chips.

The reasons for RISC-V popularity are manifold. Built from the ground up with custom extensibility in mind, RISC-V allows a new level of hardware optimisation for specific workloads. Moore's Law is slowing down, making customisation crucial to performance improvements that semiconductor manufacturing processes are struggling to provide. Moreover, the RISC-V architecture is free from

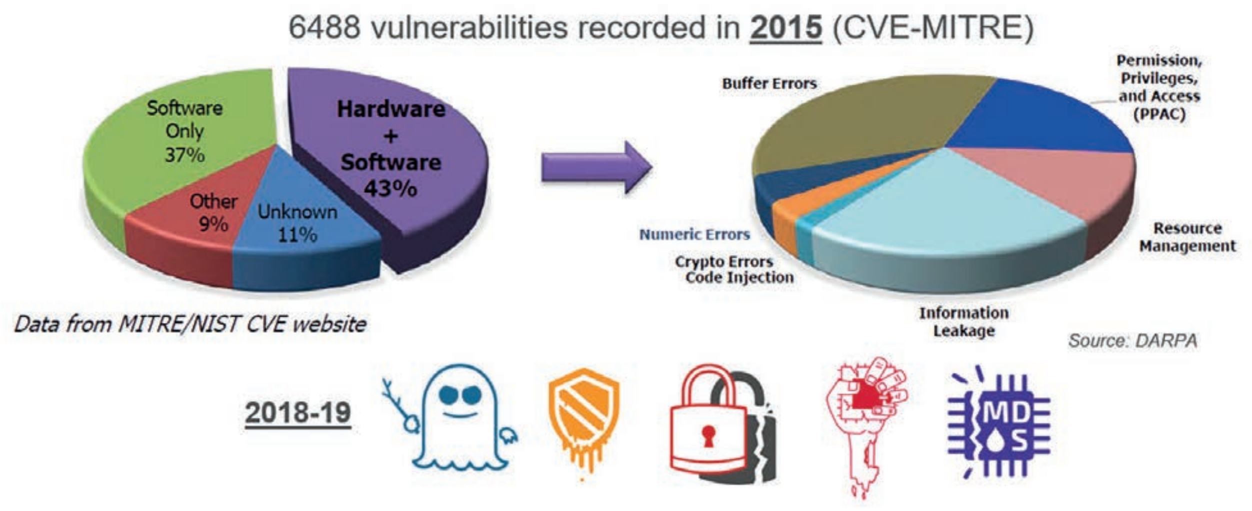


Figure 1: The CVE-MITRE database recorded 6488 vulnerabilities in 2015, 43% of which can be classified as software-assisted hardware vulnerabilities. In 2018 and 2019, researchers have discovered and reported numerous vulnerabilities in processors, including Meltdown and Spectre, Foreshadow, ZombieLoad, and RIDL and Fallout [Sources: DARPA and OneSpin]

> **Security vulnerabilities in electronic systems have stemmed from the system or the software, but, more recently, hardware IP, prominently processors, have become a concern**

licensing costs and royalties, enabling many to develop innovative, affordable products; much is happening in the field of IoT and wearable devices with artificial intelligence capabilities, for example.

SoC integrators often use open-source or third-party RISC-V processor IPs. These designs and their associated toolchains can be augmented with custom instructions. A high-quality verification environment delivered with the IP and additional system-level testing can provide confidence that the IP has no critical bugs. Unfortunately, for many applications this is not enough, and there are other serious risks to consider.

Vulnerabilities and Trojans

Traditionally, security vulnerabilities in electronic systems have stemmed from the system or the software. More recently, hardware IP – prominently processors – have also become a concern; see Figure 1. Processor implementations use pipeline-based microarchitectures and often include performance- and power-optimisation features. Complexity increases the risk of missing not only functional bugs but also security vulnerabilities.

The researchers who discovered the Meltdown and Spectre attacks in early 2018 have demonstrated that performance-optimisation features in processors can be used in unintended ways for nefarious purposes. Since then, many more vulnerabilities in both high-end and low-end processors have been discovered. Side channels and transient execution attacks may breach secure enclaves and allow malicious applications to leak confidential data or even take over the system. And unlike software, hardware issues cannot be easily repaired with over-the-air updates. Addressing a hardware problem through software often causes severe performance degradation.

A less likely risk but with far higher severity is the presence of malicious logic or hardware Trojans in the RISC-V core. A hardware Trojan is a logic function deliberately designed to be stealthy, activating in very rare circumstances. A specific sequence of data and control events that would normally not happen during system operation triggers the Trojan logic, which in turn delivers a damaging payload, leaking a secret or critically corrupting the system's behaviour. SoC integrators using open-source or third-party RISC-V cores should no longer ignore this risk.

Ensuring a processor doesn't do anything it's not supposed to is a challenging task that remains largely unaddressed. Safety-critical systems and those where protection of data privacy is paramount need efficient, high-quality solutions that address the risk of security vulnerabilities and Trojans.

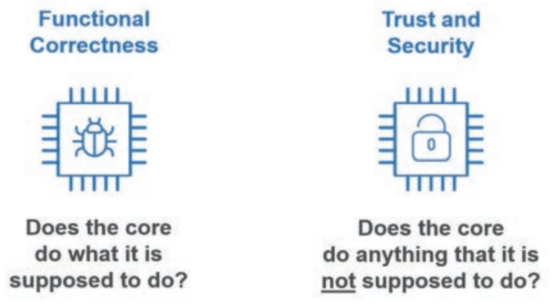
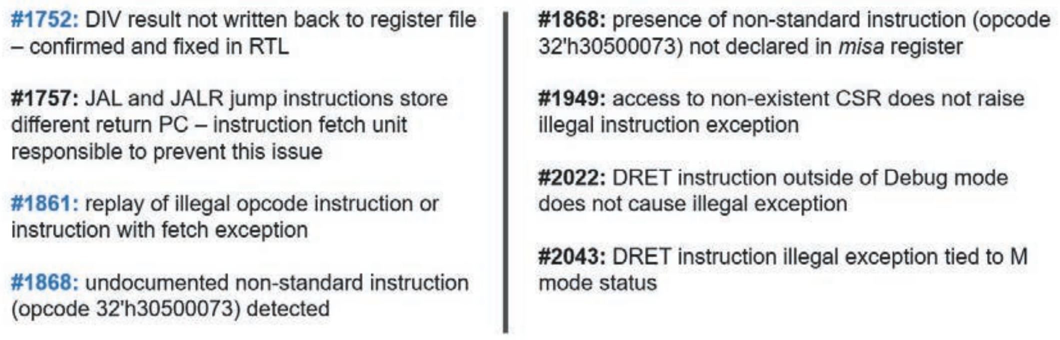

Functional Correctness
Trust and Security
Does the core do what it is supposed to do?
Does the core do anything that it is not supposed to do?

Figure 2: Functional correctness verification provides confidence that a processor implementation behaves as specified and satisfies the requirements of the end user. Trust and security verification, on the other hand, provides confidence that the processor has no undocumented functions, unforeseen side-channels, hardware Trojans or other vulnerabilities that could be exploited by malicious agents



#1752: DIV result not written back to register file – confirmed and fixed in RTL

#1757: JAL and JALR jump instructions store different return PC – instruction fetch unit responsible to prevent this issue

#1861: replay of illegal opcode instruction or instruction with fetch exception

#1868: undocumented non-standard instruction (opcode 32'h30500073) detected

#1868: presence of non-standard instruction (opcode 32'h30500073) not declared in *misa* register

#1949: access to non-existent CSR does not raise illegal instruction exception

#2022: DRET instruction outside of Debug mode does not cause illegal exception

#2043: DRET instruction illegal exception tied to M mode status

Figure 3: List of problems detected by OneSpin's RISC-V Integrity Verification Solutions and reported in the GitHub RocketCore project

Smart hardware assurance

Assuring the trust and security of RISC-V IP requires innovative and efficient technical solutions that are complementary to functional correctness approaches, mainly targeting the IP's intended use; see Figure 2. IP providers are responsible for applying state-of-the-art trust and security verification processes, whereas IP integrators should have access to independent assurance solutions that can be introduced quickly and without in-depth knowledge of the implementation.

Formal methods can analyse hardware functions exhaustively and deliver proof that the IP or SoC precisely matches the expected behaviour, often captured in SystemVerilog assertions. Hardware formal verification using commercial model checkers has enjoyed widespread adoption over the past decade. Typically, IP providers and SoC integrators use formal verification experts to reduce the risk of missing functional bugs. Whilst certain well-defined formal verification tasks can be automated through apps, in general, significant engineering effort is necessary to capture the IP's expected behaviour in assertions. Furthermore, there's no guarantee that enough assertions have been written. Undocumented functions or unintended gaps in the set of assertions could lead to unverified IP functionality.

The open-source nature of RISC-V allows the development of prepackaged, independent, assurance solutions. The RISC-V ISA spec includes two volumes – *Vol1: Unprivileged ISA*, and *Vol2: Privileged Architecture*, with Vol2 covering privileged instructions, control and status registers, etc.

OneSpin's RISC-V Integrity Verification Solution, for example, can be applied to a wide range of microarchitectures. It covers both Vol1 and Vol2 ISA models, and can accommodate custom instructions.

Significant engineering effort is necessary to capture the IP's expected behaviour in assertions

A crucial aspect of this solution is that it is based on OneSpin's GapFreeVerification process, which delivers a rigorous proof that the set of assertions modelling the RISC-V ISA is complete and free from gaps, i.e. unverified portions of the design.

Formal verification of assertions or simulation tests cover portions of the design functionality to identify deviations of the RTL design from the specifications. There are methods, like RTL code structural and functional coverage metrics, that can help detecting gaps; however, they "hunt" for gaps, but cannot prove their absence.

GapFreeVerification includes an analysis of the set of assertions that systematically detects all gaps. Once gaps are resolved, by adding additional assertions, for example, GapFreeVerification proves that no gaps are left. This, in turn, ensures that no design functions have been left unverified – including potentially malicious ones. This aspect is of utmost importance when the detection of hardware Trojans or undocumented logic is a crucial goal. The solution allows SoC integrators with limited expertise on RISC-V and the RTL implementation under scrutiny to gain confidence in the quality and trustworthiness of the IP. IP developers can use it to detect security weaknesses and functional bugs before release.

Problem detection

The RISC-V integrity assurance process has been successfully applied to multiple RTL designs. Edaptive Computing, a company that integrates innovative solutions to rapidly optimise, assure and automate systems and processes for its customers, has applied the process to RocketCore – an open-source, silicon-proven 64-bit RISC-V core with a 39-bit virtual memory system. It has a five-stage, single-issue, in-order pipeline with out-of-order completion for long latency instructions such as division. It includes the advanced features of branch prediction and instruction replay.

The RISC-V Integrity Verification Solution was applied to the design with all instructions, privilege levels, and interrupts and exception mechanisms, and eight problems were detected; Figure 3.

- **Division corner-case:** A deep corner-case bug associated with the out-of-order completion of the division instruction. This issue could have caused a software program using division operation to compute incorrect results and lead to system misbehaviour. The problem appears only under certain, rare circumstances, which is why previous verification efforts missed it.
- **Replay of illegal instruction:** This is not a corner-case bug. Replaying an illegal instruction can waste processing cycles, but if it happens only in rare situations, the performance impact is negligible. However, there are other aspects to consider. Instruction replay may cause unnecessary memory requests. These requests may have side effects that could be used in side-channel attacks. As a result, this behaviour needs to be either eliminated or well understood and documented.
- **Undocumented instruction:** An undocumented, non-standard instruction called CEASE that stops the core was detected. In effect, the RISC-V RocketCore could do something it was not supposed to. Undocumented, hidden functions are not acceptable when trust and security are a concern, even when they relate to non-relevant uses in the end application.

Detecting on-chip Trojans

The RISC-V assurance process presented here detects scenarios that could affect security, and systematically unveils undocumented functions and hardware Trojans that impact the processor's behaviour, regardless of how rare and stealthy they might be. However, side channels are not systematically detected. Exhaustive detection of all potential side-channels requires a dedicated solution with appropriate technology. There are already prototypes that address this challenge.

Processor cores are crucial IP within embedded systems, but since a typical SoC integrates many other IPs this could also contain hardware Trojans. Unlike for RISC-V cores, independent trust assurance solutions might not be readily available. In this case, it would be valuable to have an automated, low-effort trust assessment process applicable to any IP. A process that doesn't include a trusted model of the IP cannot ensure a Trojan's absence. Nonetheless, it is possible to identify unusual and suspicious code patterns and known Trojan signatures and weaknesses that could be exploited for wrong-doings in later development stages.