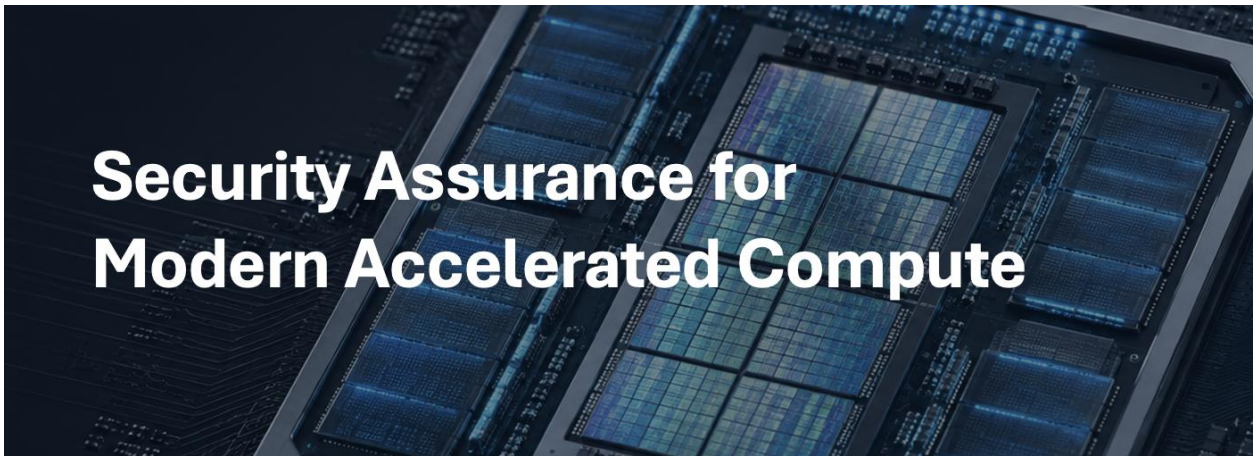




Security Assurance for Modern Accelerated Compute

Executive Summary

As accelerated computing becomes central to AI applications, hardware security becomes a critical system requirement to protect consumer data, trust, and safety. Increasing architectural complexity, including shared compute resources, chiplets, and high-bandwidth memory, creates new risks around isolation, privilege control, and secret handling that are ultimately enforced by the hardware itself.

In this white paper, we will discuss the importance of understanding the specific security requirements for modern accelerated compute infrastructure and how chip-level security drives the foundation for all of it. We'll look at why you can't assume features are trusted and why traceable assurance evidence is needed to confirm proper implementation. We'll then show how CODAx, Caspia's RTL security verification solution, addresses this need by applying security-focused analysis directly to RTL, producing actionable findings that help design teams identify and localize vulnerabilities and incorporate security assurance into the normal engineering workflow.

The Security Landscape for Accelerated Compute

Accelerated compute consists of specialized hardware components that dramatically speed up work that would normally bog down CPUs. The most common types are GPUs, TPUs, FPGAs, and now dedicated AI accelerators. They are becoming more prominent and ending up in many system architectures as AI applications broaden. Figure 1 summarizes



how we got here. This is evolving the fundamental compute stack that was historically CPU-driven.

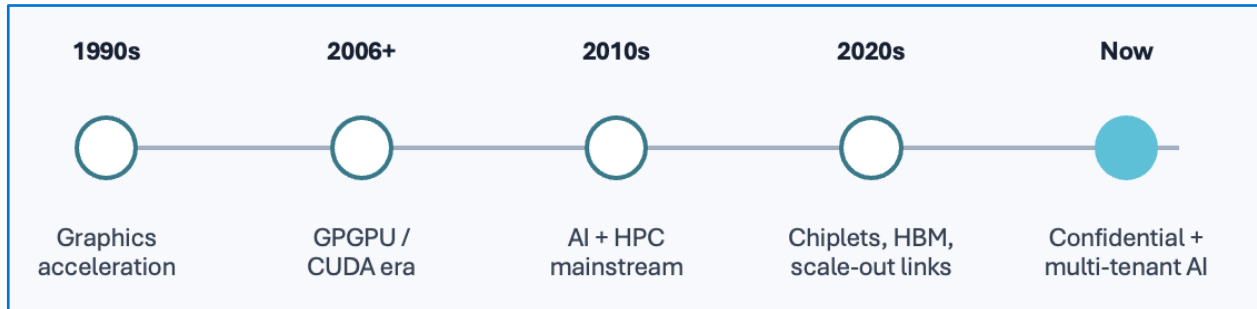


Figure 1. Historical changes in accelerated compute architecture.

This architectural shift also changes the security model. In cloud and hyperscale environments, a single accelerator may serve multiple tenants and process sensitive data from workloads that must remain isolated. The hardware must consistently enforce that separation and restrict privileged operations. This is especially critical for confidential compute. A weakness in the underlying hardware logic can cross workload or system boundaries, while correcting it after fabrication can be difficult and costly.

As a result, chip providers must now evaluate a broader set of threat scenarios and valuable assets that are used inside their chips. Table 1 summarizes what must be protected and how those assets could be targeted.

Table 1. Accelerate compute assets and relevant threat scenarios.

Asset	Why it matters	Representative threat scenarios
Workload data	Training data, prompts, activations, outputs, checkpoints	A malicious tenant or compromised host reads another workload's memory; stale data remains after a context switch; unauthorized DMA or link access exposes data in transit.
Model and software IP	Model weights, proprietary kernels, firmware, microcode	Model weights or kernel code are extracted through memory or debug interfaces; firmware is substituted or rolled back; a compromised update path introduces unauthorized code
Device trust material	Keys, device identity, measurements, lifecycle state	Keys are extracted or misused; measurements or attestation evidence are forged or replayed; provisioning or lifecycle controls are manipulated.
Control state	MMIO, schedulers, context tables, IOMMU and reset controls	Lower-privilege software writes protected registers; context or ownership state is altered; address checks are bypassed; fault injection forces control logic into a protected state.
Availability	Compute capacity, memory, interconnect and recovery paths	Malformed commands or resource exhaustion deny service; reset failures leave the device in an unsafe state; an attacker abuses recovery behavior to execute unauthorized code.



The security of accelerators within this context must be considered of high importance with its proliferation. The best time to challenge a hardware security assumption is before it is implemented in silicon. It will be a costly fix with major delays if found late. Figure 2 illustrates a real example of an accelerated compute vulnerability.

REAL EXAMPLE

AMD-SB-6027

Improper isolation in AMD GPU hardware register space allowed one VM/process to gain arbitrary read/write access to another tenant's data

Why it matters for design assurance

- A small isolation mistake can expose high-value AI data
- Mitigation can introduce performance or deployment tradeoffs

CVE-2024-36323 | AMD Security Bulletin AMD-SB-6027

Figure 2. Published accelerated compute hardware vulnerability.

Security Assurance Must Reach RTL

Dedicated security verification of chip-level features must be deployed early to enable cost-effective design changes and document early security checkpoints in the development lifecycle. A small control-path mistake can invalidate sound architecture because the hardware becomes the ultimate enforcement point for access and isolation.

Traditional functional verification is insufficient for this purpose because many security properties, such as information flow restrictions, single-write semantics for lifecycle registers, correctness of alert behavior, or prevention of unauthorized state transitions, may not be thoroughly verifiable with a functional verification approach as it does not typically explore the full range of adversarial scenarios (e.g., fault injection attacks, scan-chain exploitation, etc.).

This means that pre-silicon RTL security verification is becoming a high-leverage checkpoint. The objective is simple, challenge security assumptions while the design is still changeable. Using this approach, some questions worth answering before tape-out include:

- Can untrusted logic reach a protected state?
- Can privileged control be bypassed?
- Can secret-bearing states leak to outputs?
- Are reset and initialization states safe?
- Do debug or test paths create unintended access?



Figure 3 illustrates some of the places that security themes show up in RTL for AI accelerators.

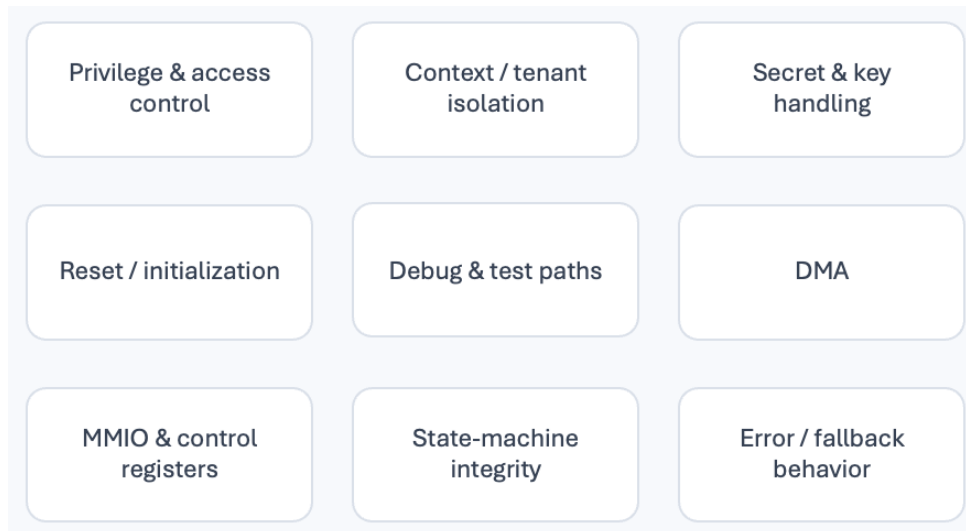


Figure 3. Security themes in RTL for AI accelerators.

Using the typical flow and tools available today creates some barriers to getting accurate answers to these questions. Some examples include:

Security intent is scattered: Specs, architecture reviews, tribal knowledge and expert intuition are hard to operationalize.

Traditional checks are not security-complete: Lint and formal apps solve different classes of design risk. Testbench-driven verification requires the upfront knowledge of what to check for.

And a key challenge:

Security knowledge does not scale: Large RTL bases change continuously, and the same questions must be re-asked after every revision, resulting in time consuming and error-prone manual reviews of code.

There is a technology available that addresses these obstacles, bringing early security analysis and optimization at RTL within reach.



How CODAx Fits Engineering Workflows

Caspia Technologies' industry-leading solution, CODAx, empowers design teams with an easy-to-understand and automated security verification tool, making secure designs more accessible than ever before. With CODAx, teams can gain security verification capabilities without needing to become security experts.

CODAx is a static security analyzer that checks for violations of security weaknesses in RTL code, with little to no human-in-the-loop. CODAx automatically identifies unsafe RTL coding practices that can raise concerns for issues such as confidentiality, integrity, and availability problems that are mapped to sources such as the Common Weakness Enumeration (CWE) database. CODAx can currently identify more than 50 security concerns and is continuously being expanded with new security directives. CODAx delivers a security-focused analysis layer for design teams that need evidence and actionable findings.

Figure 4 illustrates how CODAx fits into any standard design flow. It runs directly on RTL and is designed to fit into existing engineering flows rather than create a separate security silo.



Figure 4. CODAx in the design flow

Security Analysis of a Root-of-Trust

A root-of-trust is essential in the accelerated compute architecture as it provides the foundational layer of security for the entire SoC and surrounding system. It anchors system security through critical services such as:

- Identify
- Measured boot
- Attestation
- Lifecycle management



A popular hardware root-of-trust for datacenter-class SoCs is Caliptra from the CHIPS Alliance (<https://chipsalliance.github.io/caliptra-web/>). This is an open-source specification and hardware implementation that contains numerous security IPs. Security verification is crucial to ensure the security features are properly implemented and resilient.

CODAx is designed as a critical component of this security verification process. It provides fast, automated analysis to identify potential weaknesses and offers designers an early, clear view of their security posture during the development lifecycle, delivering significant value in ensuring long-term security for hardware platforms.

Caspia performed a security analysis of the open-source Caliptra hardware IP and found numerous security findings from CODAx security rule violations. Violations related to FSM security, insecure reset initialization, and entropy propagation were found. Figure 5 shows a summary.

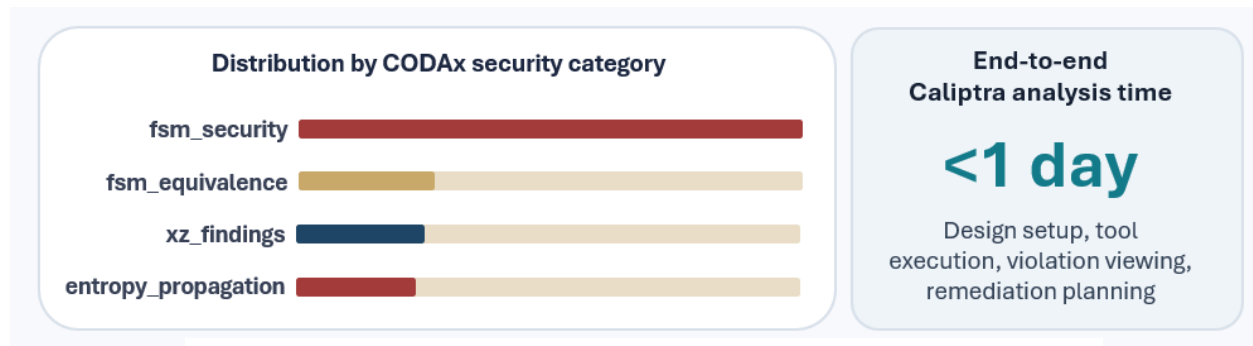


Figure 5. CODAx security analysis on Caliptra root-of-trust.

The security analysis with CODAx took less than a day to complete. This demonstrates a significant use-case for feasible security verification, without a security expert.

Please contact Caspia for more information on the Caliptra security findings.

Summary

The evolution of accelerated compute demands a critical requirement for early security verification. Caspia's CODAx can form the basis for an effective solution to the security challenges that lie ahead. CODAx is made for non-security engineers to enable the effective and scalable adoption for secure-by-design practices.

Visit the [Caspia Technologies website](#) to learn more about the early security verification solutions we offer. You can also download our [CODAx whitepaper here](#) to dig deeper.