AI-Generated SVA from Requirements: Synopsys, Cadence and Siemens Compared with RAG-SVA

Oct 19, 2026 ·

Ben Cohen · Kais Chibani · systemverilog.us

Abstract

Of the three major EDA vendors, Cadence (ChipStack) and Synopsys (GenFV) generate SVA at module scope, while Siemens (Property Assist) still works mainly one property at a time. All three accept English requirements plus RTL. None publicly demonstrates SVA generated from a timing diagram, PSL conversion, or linked sources, and none halts on ambiguous or contradictory requirements. For standard protocols such as AMBA and RISC-V, all three rely on pre-built assertion IP rather than generation. RAG-SVA differs on three axes: it accepts multi-format input (documents, slides, text, images, links, RTL, PSL, SVA), it is governed by an explicit, citable rule corpus, and it stops to resolve ambiguity before writing code. Its main gap is the absence of a built-in prover loop.

> ***Disclosure: AI-generated content.*** *This paper was generated with AI (Claude, Anthropic) under the direction of Ben Cohen. Vendor information was obtained from publicly available sources as of October 19, 2026: vendor press releases and product briefs, vendor blogs, trade-press articles (EE Times, Embedded.com, SiliconANGLE), and DVCon proceedings, including the 2026 Synopsys/NVIDIA/Google tutorial. RAG-SVA information comes from the authors' framework and rule corpus. Gated vendor user guides were not consulted, so vendor capabilities may exceed what is public. All references are listed at the end.*

1. Introduction

AI-assisted SVA generation has moved from research prototypes into commercial EDA flows. Each major vendor now advertises converting requirements into assertions. Two practical questions decide whether these tools help with real designs:

1. **Scope:** can the tool take the requirements for a whole module (a bus interface, a CPU pipeline, a controller) and produce a complete, consistent assertion set, or does it work one property at a time?
2. **Inputs:** can it consume the formats engineers actually use (English specs, timing diagrams, architecture diagrams, legacy RTL, PSL or SVA, slides, links), or only typed English plus RTL?

This paper answers both questions for Synopsys, Cadence and Siemens from public material, and compares each with RAG-SVA, the rule-governed framework described in *Knowledge-Driven AI for SystemVerilog Assertions* (Cohen and Chibani, 2026).

2. Method and sources

The vendor findings are an AI-assisted desk review of public sources, not a hands-on evaluation. Claude searched the web and read the following material, then the findings were organized by the authors' two questions:

- **Synopsys:** DVCon slides on VC Formal GenFV (Formal Advisor), and the DVCon 2026 tutorial *AI Meets Formal* by Synopsys, NVIDIA and Google, which is the most detailed public source of any vendor.
- **Cadence:** the ChipStack AI Super Agent product brief, press release and blog, plus EE Times and SiliconANGLE coverage.
- **Siemens:** the Questa One launch announcement, the Property Assist webinar abstract on Verification Academy, and coverage of the 2026 Questa One Agentic Toolkit.

A capability is reported as present only when a source states it. "Not stated" means no public source was found, not that the capability is absent. Customer results are vendor or customer claims and are labeled as such.

3. Vendor approaches

The three vendors differ mainly in where their expertise lives: an example database (Synopsys), a design knowledge graph (Cadence), or a hidden prompt optimizer (Siemens).

3.1 Synopsys VC Formal GenFV (Formal Advisor)

- **Interfaces:** Verdi GUI, a VS Code extension, and batch Tcl, with an LLM gateway to Azure OpenAI, Gemini, AWS Bedrock, Llama and in-house models.
- **Knowledge:** a RAG database of English-query to accepted-SVA pairs in JSONL, plus a document database of specs, coding guidelines and best known methods. Proven assertions are added back over time.
- **Flow:** the user selects testplan lines; the LLM generates several candidate properties; an agentic mode checks syntax, Tcl, vacuity and tautology and runs VC Formal; the user accepts.
- **Extras:** a helper-assertion agent that generates invariants to turn bounded proofs into full proofs.

3.2 Cadence ChipStack AI Super Agent

- **Knowledge:** a "mental model," a structured representation of design intent, hierarchy and relationships built from multi-modal specifications, design sources and historical context. Cadence reports that disabling it cut verification accuracy by about 30 to 40% in an internal experiment.
- **Flow:** generates or upgrades a formal verification plan and SVA, and proves them with Jasper; also builds UVM sequences, checkers and coverage for simulation. It can run end to end or pause for engineer review at each stage.
- **Status:** early access; Cadence cites NVIDIA, Altera and Tenstorrent as customers.

3.3 Siemens Questa One Property Assist and Agentic Toolkit

- **Property Assist:** rewrites the user's English prompt into an optimized LLM prompt, gathers candidate SVA, and presents the best one. MediaTek reports it saved weeks of engineering time.
- **Agentic Toolkit (2026, early access):** agents reach Questa engines through the Model Context Protocol (MCP), backed by expert-curated prompt libraries. A Verification Planning Agent generates a verification plan from the specification.

4. Module scope versus one property at a time

Cadence and Synopsys work at module scope; Siemens Property Assist works one property at a time.

Tool	Scope	Evidence
Synopsys GenFV	Module: full FV testbench (asserts, assumes, bind files, Tcl)	Linked-list FIFO, arbiter/mux and PCIe scheduler. Quality score reached 75% on the FIFO and 61% on the arbiter, but stayed near 3% on the scheduler, where Synopsys says it struggles with verification intent. Driven by user-selected testplan lines.
Cadence ChipStack	Module to SoC: formal plan + SVA, proven in Jasper	Tenstorrent reports up to 4x less formal verification time on three critical blocks (customer claim). No public quality metric or SVA examples.
Siemens Property Assist	One property per prompt	Module-level planning is moving to the Agentic Toolkit's Verification Planning Agent; full module SVA sets not yet shown.
RAG-SVA	Module: a requirement set for an entity	Requirements are checked as a set for consistency before code (Section 8); decomposition is controlled by RULE DECOMP1 (ATOMIC, PIPELINE, STAGE=N).

Synopsys' own data is the clearest public signal: module-scope generation works on mid-size blocks and breaks down on complex control logic, because the tool cannot recover intent that the requirements leave unstated.

5. Standard protocols (AMBA, CPU): assertion IP rather than generation

For AMBA and processor ISAs, vendors ship pre-written, pre-proven assertion IP rather than generating SVA from requirements.

- **Synopsys:** VC Formal assertion IP, including a RISC-V ISA AIP that tracks control flow, hazards and register values on cores such as CVA6 and Ibex, plus AIP for pipelines, FPUs and security.
- **Siemens:** Questa Formal Verification IP for AMBA protocol compliance.
- **Cadence:** Jasper offers its own protocol assertion IP; it was not covered by the sources reviewed here.

AI's role around assertion IP is mainly instantiation and binding. NVIDIA's DVCon work used RAG to automate VIP instantiation; it did not write AMBA assertions from the specification.

Requirement-driven generation matters for the custom logic around the protocol: a CPU's control FSM, a bus bridge, an arbiter. That is where Synopsys' quality data drops, and where RAG-SVA's ambiguity handling (Section 8) applies.

6. Input formats

All three vendors accept English plus RTL; only RAG-SVA documents timing-diagram images, PSL conversion and links as inputs.

Input	Synopsys GenFV	Cadence ChipStack	Siemens Property Assist / agents	RAG-SVA
English requirements	Yes: selected testplan text + prompts	Yes: specifications	Yes: plain-English prompts	Yes
Spec documents (PDF, DOCX)	Yes, via a document RAG (Google's Gemini setup)	Yes ("multi-modal specifications")	Planning Agent reads specs; formats not stated	Yes: DOCX, PDF, TXT
Slides (PPTX)	Not stated	Not stated	Not stated	Yes
RTL	Yes, required	Yes; can start from existing RTL and testbenches	Yes, via MCP design context	Yes: SystemVerilog or VHDL
Existing SVA / legacy code	Partly: annotates existing SVA, builds on an existing setup	Partly: upgrades existing plans, uses historical context	Not stated	Yes: SVA, plus C++ models
PSL to SVA	Not stated	Not stated	Not stated	Yes: converted to formal-safe SVA
Timing diagrams (PNG, JPG)	Not stated	Unclear: "diagrams" feed the mental model; SVA from waveforms not shown	Not stated	Yes
Architecture diagrams	Not stated	Claimed in general terms	Not stated	Yes
Links	Not stated	Not stated	Not stated	Yes

RAG-SVA also back-translates generated SVA into structured "formal English" so engineers can audit what the AI understood (Section 7).

7. Comparison with RAG-SVA

The vendors are strongest at closing the loop with the prover; RAG-SVA is strongest at making the assertion right by construction.

Aspect	Synopsys GenFV	Cadence ChipStack	Siemens Property Assist	RAG-SVA
Knowledge	Examples (query to accepted SVA) + guideline docs	Design knowledge graph ("mental model")	Hidden prompt optimizer, curated prompt libraries	Explicit rule corpus with IDs, rationale and precedence
Ambiguity or contradiction	Generates candidates; human reviews	Not documented	Picks the "best" answer	Halts and asks (RQM0, Categories 1 to 6)
Assumptions vs. obligations	Generates assumes; method not documented	Not documented	Not documented	Classified from port direction; assume signals never in an assert consequent
Decomposition	Not documented	Not documented	One property per prompt	RULE DECOMP1: ATOMIC, PIPELINE or STAGE=N
Style and traceability	Custom instructions (format, naming)	Not documented	Not documented	ap_/mp_/cp_ naming, headers, requirement trace
Prover loop	Strong: VC Formal in the agent loop	Strong: Jasper in the loop	Strong: Questa via MCP (agents)	Engineer runs any FV tool and feeds results back
Tool coupling	VC Formal	Jasper	Questa	Model- and tool-neutral

Where RAG-SVA leads. Its rules apply before generation: banned operators, bounded windows, determinism, node-independent latency from a package. Synopsys' instructions, by contrast, ban `strong` and `weak` outright, while RAG-SVA allows `strong()` for liveness only, with the reason recorded.

Where the vendors lead. Each closes the loop with its own prover and automates helper invariants (Synopsys). RAG-SVA could gain the same loop by driving an FV engine through MCP once vendors expose one, which Siemens' toolkit already does.

8. Worked example: contradictory requirements

RAG-SVA catches a contradiction on paper that an example-based generator would surface only as a formal failure.

Requirements for an entity with input `req` and outputs `ready` and `ack`:

- REQ-A: req to ack in 5 cycles.
- REQ-B: req to ready in 8 cycles.
- REQ-C: ready to ack in 1 cycle.

REQ-B plus REQ-C place `ack` at cycle 9; REQ-A requires cycle 5. No RTL satisfies all three. RAG-SVA halts under RQM0 Category 4 (logical flaw) and Category 3 (cycle counting), and asks:

1. Is "in N cycles" exact (`##N`) or within (`##[1:N]`)? Under "within" the set is consistent, but `ready` must arrive by cycle 4 and the 8-cycle bound never binds.
2. Which number is wrong: end-to-end latency 9, or req to ready 4?
3. Are the bounds fixed, or MIN/MAX values from a latency package?
4. Is `ready` a pulse or a level? Can requests overlap? Does `req` hold until `ack`?

After resolution, the chain is decomposed per RULE DECOMP1 into single-implication properties, with the end-to-end property kept as a consistency check. A tool that generates candidates from similar past examples would instead emit three plausible assertions, and the contradiction would appear only as a failing proof that someone must debug.

9. Conclusions and limitations

Commercial tools have reached module-scope generation, but they still trust the requirements they are given; RAG-SVA questions them first.

- **Scope:** Cadence and Synopsys generate at module scope; Synopsys' own data shows quality falling on complex control. Siemens remains per-property until its agents mature.
- **Protocols:** AMBA and ISA checking is served by assertion IP; requirement-driven generation matters for the custom logic around it.
- **Formats:** all vendors take English plus RTL; none publicly shows SVA from timing diagrams, PSL conversion or links. RAG-SVA handles all three.
- **Quality control:** vendors filter after generation with their provers; RAG-SVA constrains generation with rules and halts on ambiguity.
- **Complementary:** RAG-SVA output can feed any vendor's prover, and an MCP connection to a formal engine would close RAG-SVA's one gap.

Limitations. This is an AI-assisted review of public material as of October 19, 2026. Gated user guides and hands-on evaluation could change several "not stated" entries. Vendor results are vendor or customer claims. RAG-SVA has not yet been scored on Synopsys' quality metric (0.4 effectiveness + 0.3 correctness + 0.3 quality) on the same designs; doing so would make a strong next study.

References

1. DVCon 2026: AI Meets Formal — Practical Applications from Industry Leaders (Synopsys, NVIDIA, Google)
2. DVCon: Addressing Formal Verification Challenges with GenAI Technology and RISC-V Solutions (Synopsys)
3. Cadence ChipStack AI Super Agent product brief
4. Cadence press release: ChipStack AI Super Agent
5. Cadence blog: Reimagining Chip Design — From Spec to Signoff with AI Super Agents
6. EE Times: Cadence Unveils ChipStack AI Agent
7. SiliconANGLE: Cadence announces ChipStack Super Agent
8. Verification Academy: Generating SVA Properties with Property Assist
9. Siemens Questa One launch (PR Newswire)
10. Embedded.com: Siemens Agentic Toolkit Automates Chip Verification Workflows
11. B. Cohen and K. Chibani, *Knowledge-Driven AI for SystemVerilog Assertions*, VhdlCohen Publishing, 2026. systemverilog.us
12. B. Cohen, The RAG-SVA Paradigm Shift in Formal Verification (systemverilog.us)