

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CADENCE DESIGN SYSTEMS, INC.,
Petitioner

v.

SEMICONDUCTOR DESIGN TECHNOLOGIES, LLC,
Patent Owner

Case IPR2024-00356
U.S. Patent No. 7,603,636

**PATENT OWNER PRELIMINARY RESPONSE
UNDER 37 C.F.R. § 42.107(a)**

Mail Stop “PATENT BOARD”
Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	OVERVIEW OF THE '636 PATENT	5
III.	SUMMARY OF THE ASSERTED REFERENCES.....	7
A.	Property-Generating References	8
1.	Kobayashi.....	8
2.	Feyerabend	9
B.	Assertion-Generating References.....	10
1.	Abarbanel	10
2.	Narain.....	11
IV.	CLAIM CONSTRUCTION	15
V.	ARGUMENT.....	15
A.	Grounds 1 and 4: Petitioner Fails to Show that Abarbanel and Kobayashi Render the Challenged Claims Obvious.	16
1.	Petitioner Fails to Show that Kobayashi Discloses the Claims' "Property" Limitation.	16
2.	Petitioner Fails to Establish that a POSA Would Have Combined Kobayashi with Abarbanel.....	18
a.	Petitioner's Obviousness Combination Is Unclear and Confusing.	19
b.	Petitioner's Proposed Combinations Destroy Abarbanel's Principal Purpose and Impermissibly Blur Statelist Data with RCTL Data.	22
c.	Petitioner's Remaining Arguments Amount to Conclusory and Generic Assertions About What Was "Known" and "Common" in the Art.	28
B.	Grounds 2 and 5: Petitioner Fails to Show that Narain and Kobayashi Render the Challenged Claims Obvious.....	31
1.	Petitioner Fails to Show that Kobayashi Discloses the Claims' "Property" Limitation.	31

2.	Petitioner Also Fails to Show that Narain Discloses the Claims’ “Property” Limitation.	32
a.	“VX” Directives for Built-In Assertions	32
b.	Predetermined Properties for Sentry Verification Entities	35
3.	Petitioner Fails to Establish that a POSA Would Have Combined Kobayashi with Narain.	39
a.	Petitioner’s Obviousness Combination Is Unclear and Confusing.	39
b.	Petitioner’s Arguments for Combining Narain and Kobayashi Are Wholly Conclusory and Generic.....	40
C.	Grounds 3 and 6: Petitioner Fails to Show that Abarbanel and Feyerabend Render the Challenged Claims Obvious.....	41
1.	Petitioner Fails to Show that Feyerabend Discloses the Claims’ “Graphically Editing” Limitation.	41
2.	Petitioner Fails to Establish that a POSA Would Have Combined Feyerabend with Abarbanel.	43
a.	Petitioner’s Obviousness Combination Is Again Unclear and Confusing.	44
b.	Petitioner’s Arguments for Combining Feyerabend and Abarbanel Are (Once Again) Conclusory and Generic.....	45
c.	Petitioner Fails to Show that a POSA Would Have Had a Reasonable Expectation of Success in Using Feyerabend’s Real-Time Properties in Abarbanel’s System.	45
VI.	CONCLUSION	48

PATENT OWNER'S EXHIBIT LIST

Exhibit No.	Description
2001	Declaration of John E. Berg in Support of Patent Owner Preliminary Response

I. INTRODUCTION

On account of trying to pack as many arguments as possible into the Petition, Petitioner's obviousness challenge to the '636 patent is confusing, ambiguous, and underdeveloped. Petitioner barely engages with the technology at all, for example, and does not even bother quoting the patent claims in full. The Petition does not even use complete sentences in certain places. All of this leads to a difficult read, particularly given the sophisticated nature of the integrated-circuit ("IC") technology at issue.

But Petitioner's corner-cutting goes far beyond mere readability—it affects the *substance* of Petitioner's obviousness arguments. This is perhaps best illustrated by Petitioner's motivation-to-combine analysis. For each of its obviousness combinations, Petitioner asserts that the various references *could* be combined merely because their teachings were "known" or "common" in the art, without explaining *why* the references *would* be combined. *See, e.g.*, Pet., 14-15, 16-18, 26-29. Further, Petitioner invokes all of *KSR*'s generic rationales for combining references without explaining *why* each rationale applies or providing any underlying factual support. *See, e.g., id.*, 15-18.

Still further, in an effort to make every argument that could possibly support a motivation to combine the references, Petitioner asserts that a first reference in each combination would be used to improve upon and modify a second reference,

while simultaneously asserting that the second reference would be used to improve upon and modify the first. *See, e.g., id.*, 61 (“Adding Feyerabend’s timing diagram editor to Abarbanel simplifies the design/verification process by enabling a more user-friendly editor, while improving verification by generating checkers automatically. Adding Abarbanel’s FoCs to Feyerabend reduces cost and labor.”).

Such contradictory, confusing, and conclusory arguments are legally insufficient to carry Petitioner’s burden. *See Virtek Vision Int’l ULC v. Assembly Guidance Sys., Inc.*, 97 F.4th 882, 886-889 (Fed. Cir. 2024) (rejecting petitioner’s argument that a POSA would have combined prior-art references merely because the references were “known,” and further rejecting petitioner’s bare invocation of *KSR* as a substitute for reasoned analysis); *Sisvel Int’l S.A. v. Sierra Wireless, Inc.*, 82 F.4th 1355, 1364 (Fed. Cir. 2023) (rejecting petitioners’ “kitchen-sink of unclear and confusing motivation-to-combine arguments”). This fundamental failure pervades the Petition and dooms each of Petitioner’s unpatentability grounds.

As to Grounds 1 and 4 (Abarbanel and Kobayashi), Petitioner has not shown that Kobayashi discloses the “property” limitation (“a property which verifies the specification of the semiconductor integrated circuit by reading the design data”) recited in every claim of the ’636 patent. Kobayashi is directed to IC fabrication, not IC verification. Accordingly, Kobayashi does not describe verifying a

specification of the IC, let alone using the claimed “property” to do so. Further, because the claimed IC “assertion” must be generated from the claimed property, Petitioner has likewise failed to show that the prior art discloses the “assertion” limitation.

Still further, Petitioner has not shown that a POSA would have combined Abarbanel and Kobayashi in the first place. Although unclear, Petitioner seems to suggest that Kobayashi’s “statelist” (the alleged “property”) could simply be plugged into Abarbanel’s system to generate the claimed “assertion.” But Petitioner ignores that Abarbanel’s system is limited to using a particular type of specification programming language called RCTL, which Abarbanel expressly touts as providing benefits that are critical to its solution and that statelists cannot provide. Petitioner’s modification to Abarbanel impermissibly negates the very benefits that Abarbanel is designed to achieve.

As to Grounds 2 and 5 (Narain and Kobayashi), Petitioner again fails to show that Kobayashi discloses the “property” limitation for the same reasons as stated above. Petitioner likewise fails to show that Narain discloses the limitation. Although unclear, Petitioner appears to point to Narain’s “built-in assertions” and Sentry verification entities as satisfying the limitation. But the built-in assertions are just that—IC *assertions*, not IC properties. And the built-in assertions are manually input into code, not read from IC specification design data, as required

by the claims. As for the Sentry verification entries, each entry is associated with a set of predetermined properties that are preprogrammed, not read from IC specification design data. And, again, because the prior art does not satisfy the “property” limitation, the art also does not satisfy the “assertion” limitation. Still further, similar to the deficiencies described above, Petitioner’s motivation-to-combine arguments rely solely on confusing, conclusory, and generic assertions rather than reasoned analysis.

Finally, as to Grounds 3 and 6 (Abarbanel and Feyerabend), Petitioner fails to show that Feyerabend discloses the “graphically editing” limitation (“graphically editing the specification of the semiconductor integrated circuit”), which is likewise recited in every patent claim. Petitioner speculates that Feyerabend uses software to permit a user to graphically edit an IC specification, but Feyerabend contains no teaching or suggestion that its specification can be graphically edited, let alone describe software for doing so. Lastly, Petitioner fails to show that a POSA would have combined Feyerabend with Abarbanel, once again relying on conclusory allegations instead of reasoned, evidence-backed analysis.

At bottom, Petitioner has failed to carry its burden to show that any claim of the ’636 patent is unpatentable. The Board should thus reject each of Petitioner’s unpatentability grounds and deny institution.

II. OVERVIEW OF THE '636 PATENT

The '636 patent relates to semiconductor technology and, in particular, generating and verifying an IC's design data in a process known as "assertion verification." EX1001, 1:8-19, 1:46-54, 3:35-38.

The process begins when a computer system allows a user to graphically edit an IC "specification" (i.e., a document describing the functional requirements of the IC) and generates a "property" that verifies the specification based on design data. *See, e.g., id.*, 4:45-67, 8:4-20. The property, in turn, is converted into an "assertion," which is then used to test the IC to ensure that it functions as required by its specification. *Id.* An "assertion" is a statement written in a hardware-description programming language that reflects the IC design and is used to model the circuit at the register-transfer level. *Id.*, 1:55-57, 2:37-52. The assertion is ultimately used by the system to verify the IC—i.e., to test whether the IC functions as required by the specification. *Id.*, 1:8-19, 1:46-54, 1:55-2:19.

Importantly, the patent makes clear that the IC assertion is generated *automatically* from the IC property. *See, e.g., id.*, 4:39-44, 8:4-20, FIG. 2. In fact, the patent distinguishes and disparages prior-art systems in which assertions were *manually* generated. Such "conventional technologies," the patent explains, led to "mistake[s]" in the assertions, which caused the assertions to diverge from the specifications themselves. *Id.*, 3:43-47, 3:54-63, 4:29-31. This meant that

“matching the circuit specifications with the assertion description cannot be assured.” *Id.*, 4:29-31; *see also id.*, 4:4-11 (disparaging prior-art reference on the basis that its “model of the state transition machine ... is formed manually” such that “mixing a mistake in an object to be matched[] ... cannot be avoided”).

The invention of the '636 patent “solve[s] the problems” in the prior art by *automatically* generating an assertion, thus “enabl[ing] high reliability and high efficiency in assertion verification.” *Id.*, 4:35-38. The patent claims recite methods, systems, and computer-readable media that leverage this novel feature.

Claims 1, 7, 8, and 14 are independent. Claim 1 recites:

1. An assertion generating system that generates an assertion description which is used for assertion verification of a semiconductor integrated circuit, comprising:
 - a specification inputting unit that generates design data or specifications and a document for confirming a specification of the semiconductor integrated circuit by *graphically editing the specification* of the semiconductor integrated circuit based on user operations;
 - a computer readable storage device having a first storing unit that stores the design data generated by the specification inputting unit;
 - a property generating unit that generates *a property which verifies the specification* of the semiconductor integrated circuit by reading the design data generated by the specification inputting unit from the first storing unit and using the read design data;

a second storing unit that stores the property generated by the property generating unit, wherein the second storing unit may or may not be a part of the computer readable storage device; and an assertion generating unit that *automatically converts the property into an assertion* description if the property is to be verified during assertion verification by reading the property generated by the property generating unit from the second storing unit, wherein the property generated by the property generating unit is a selection condition with respect to a state transition, a logic value of at least one or more signals, or at least one or more signals in the design data.

Id., 13:29-57 (emphases added).

Relevant here are the claim’s “graphically editing,” “property,” and “assertion” limitations. The “graphically editing” limitation requires that an IC specification be “graphically edit[ed].” The “property” limitation requires that an IC property be generated by reading specification design data and used to “verify the specification.” And, finally, the “assertion” limitation requires that an IC assertion be “automatically” generated from the property. As explained below, Petitioner has not shown that the prior art discloses these limitations.

III. SUMMARY OF THE ASSERTED REFERENCES

Petitioner relies on two types of references: (A) those describing the alleged generation of IC properties from IC specifications (Kobayashi and Feyerabend),

and (B) those describing the alleged generation of IC assertions from properties (Abarbanel and Narain). None of the references renders the '636 patent claims unpatentable.

A. Property-Generating References

1. Kobayashi

Kobayashi discloses a “computer-aided design system and method for designing an application specific integrated circuit.” EX1005, Abstract. The purpose of Kobayashi is to design a circuit, not to generate properties or assertions, much less perform IC verification using properties or assertions. *Id.*, 1:9-12, 1:63-2:63, 3:49-4:46.

The design process begins with the display and manipulation of an IC specification in the form of a flowchart. Kobayashi describes using the flowchart to “translate[] the functional architecture independent specifications into structural [and] architecture specific level definition of an integrated circuit, which can be used directly to produce” the chip. *Id.*, 2:28-34.

Next, using the flowchart, Kobayashi’s system generates a “statelist,” which is a “a structure that can be used by the Cell Selector” to “map[] [functional] blocks to cells selected from the cell library 34.” *Id.*, 9:8-13, 9:21-22. Although the statelist reflects the states and transitions depicted in the flowchart, Kobayashi does not describe that its statelist verifies the flowchart. Rather, the statelist is used to

help map functional blocks to cells and, ultimately, to fabricate the IC. *Id.*

2. Feyerabend

Feyerabend presents a “semantical basis of a graphical specification language, called real-time symbolic timing diagrams (RTSTD), to express real-time requirements of embedded systems.” EX1007, 1 (Abstract). In particular, Feyerabend extends existing symbolic timing diagrams used in “safety-critical systems” like airbags “to real time.” *Id.*, 1-2.

RTSTD includes an interface definition, waveforms, constraints between events of different waveforms, and an activation mode. *Id.*, 3. The RTSTD is translated into “a timed propositional logic (TPTL[]) dialect that uses clocks and reset quantification.” *Id.*, 6 (citation omitted). The process involves “translat[ing] the set of waveforms with the constraints imposed to an acyclic fragment of a time Büchi automaton called unwinding structure.” *Id.*, 7. “The unwinding structure is then translated to TPTL yielding the raw formula of the RTSTD body.” *Id.*

Feyerabend explains its real-time constraints make verification extremely difficult. Feyerabend acknowledges that there were preexisting “temporal logics” but notes that “adding real time to temporal logics makes the specifications even more difficult to read and makes the need for an intuitive specification style even more pressing.” *Id.*, 2. Feyerabend likewise comments on related work involving “[c]ombining real time temporal logic with a graphical interface,” but concludes

that it is “undecidable and thus not suitable for automated proof.” *Id.* In the end, Feyerabend notes that there are “major obstacle[s]” involved in “TPTL model checking” and states that “future research” is required to “focus on extending model checking techniques based on RTSTD.” *Id.*, 12.

Notably, Feyerabend does not describe any way to graphically edit its waveforms using software. Rather, Feyerabend simply provides a screenshot of a waveform presented in a user interface. *Id.*, 3.

B. Assertion-Generating References

1. Abarbanel

Abarbanel describes “a tool called FoCs (‘Formal Checkers’) ... [that] take[] RCTL properties as input and translates them into VHDL programs (‘checkers’) which are integrated into the simulation environment and monitor on a cycle-by-cycle basis for violations of the property.” EX1004, 1.

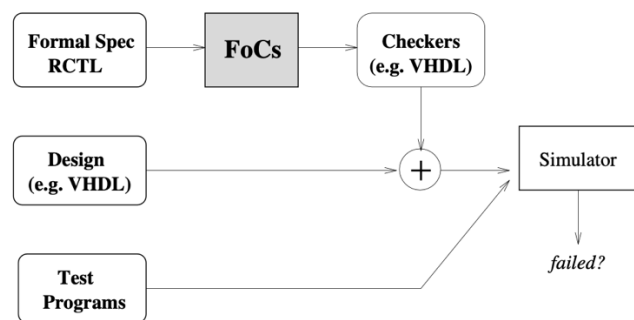


Fig. 1. FoCs Environment

EX1004, FIG. 1.

RCTL is a specification language that “includes a rich and useful set of CTL

safety formulas and regular expressions.” *Id.*, 1 n.1. Abarbanel emphasizes the “conciseness and expressiveness of RCTL formulas” compared to other languages and formats. *Id.*, 2. Abarbanel likewise emphasizes that “the simple, concise syntax and semantics of RCTL, coupled with online simulation checking, is highly useful.” *Id.*, 4. And, while Abarbanel notes that a particular prior-art reference which uses statelists (Schlipf, EX1029) “inspired” Abarbanel’s work, it distinguishes the reference on the basis that RCTL engenders “conciseness and readability” that cannot be achieved using statelists. *Id.*, 5.

2. Narain

Narain relates to “functional verification of digital designs.” EX1006, 1:17-18. A designer may specify design intent by annotating RTL code or providing a separate control file. *Id.*, 7:26-30. There are two ways in which designer can specify intent. In a first approach, the designer may invoke “built-in assertions” using the “vx” notation shown in Table 1 below:

TABLE 1

<u>Built-in Assertions</u>	
Annotation	Meaning
// vx onehot (i, j, k);	Exactly one of i, j, or k must be high
// vx onecold (i, j, k);	Exactly one of i, j, or k must be low
// vx onetrue (i, j, k);	At most one of i, j, or k can be high
// vx onefalse(i, j, k);	At most one of i, j, or k can be low

EX1006, TBL. 1.

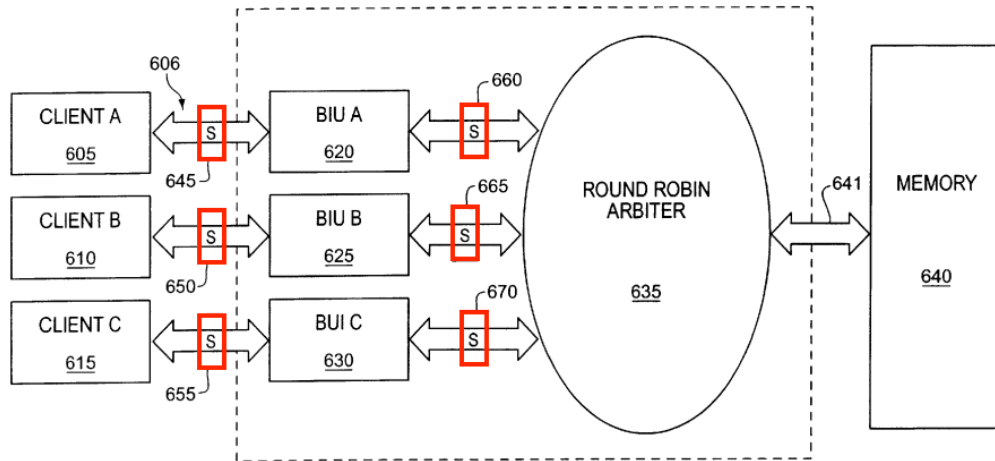
Built-in assertions “may be used in annotations to simplify assertion coding” and “cover common behaviors that many signal groups exhibit” to “allow the designer to explicitly specify design constraints for annotated signals.” *Id.*, 15:22-23, 15:40-41. The designer writes the assertions by manually inputting them into the code—i.e., the designer “plac[es] these annotations within the RTL code itself.” *Id.*, 15:16-18.

In a second approach, the designer may declare a Sentry verification entity, which creates a “sentinel variable,” by adding “simpl[e] annotations indicative of the expected state of variables at various points in the control flow structure of a finite state machine associated with the hardware design representation.” *Id.*, 13:30-43. Those annotations are shown below:

reg a;	// vx sentry a: clk;
wire b;	// vx sentry b: clk;
reg [0:15] c;	// vx sentry [1:2] c: clk;

Id., 16:21-25. Figure 6B shows the inclusion of Sentry verification entities 645, 650, 655, 660, 665, and 670 (in red) in the design of Figure 6A:

FIG. 6B



EX1006, FIG. 6B (annotated).

For each Sentry verification entity, Narain discloses testing a small, predetermined set of properties that it determined to be essential for verification: “Empirical analysis performed by the assignee has shown that verification of a relatively small set of properties at key points of a hardware design representation ... can guarantee the absence of errors.” *Id.*, 17:33-37. For example, the first predetermined property Narain discusses is “AID (Access of Invalid Data).” Narain explains that a “Sentry verification entity that is in the valid state should not receive data that flowed through a Sentry verification entity that was in the invalid state.” *Id.*, 17:58-61. Other “predetermined properties” include:

loss of valid data (LVD), assertion correctness (AC), conflicting assignments (CA), block enable (BE), assignment execution (AX), constant value memory element (CME), constant value variable (CV), Sentry activate always off (AAO).

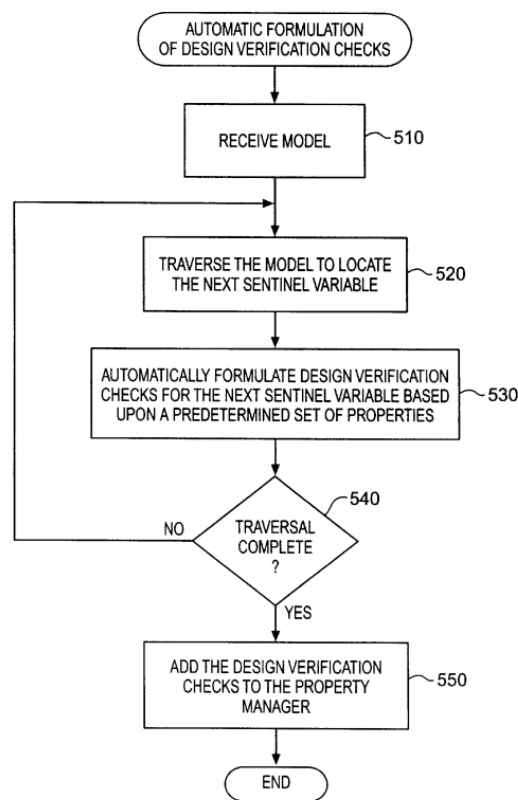
Id., 17:53-56; *see also id.*, 17:58-18:52 (describing the properties).

Figure 5 of Narain shows how the “predetermined properties” are used, where:

Upon locating the next sentinel variable, at processing block 530, design verification checks for the sentinel variable are automatically formulated based upon the predetermined properties. For example, each property may be expressed as one or more checks.

Id., 17:14-18; *see also id.*, 16:64-17:6.

FIG. 5



EX1006, FIG. 5.

IV. CLAIM CONSTRUCTION

While Patent Owner disagrees with some of Petitioner's claim-construction positions, the Board should decline to institute a proceeding because, as explained below, the relied-upon references fail to render the claims unpatentable even under Petitioner's proposed constructions. Patent Owner reserves the right to address claim construction in the event the Board institutes IPR.

V. ARGUMENT

In an IPR, “the burden of proving a proposition of unpatentability” is borne by (and remains at all times with) the petitioner. 35 U.S.C. §§ 314(a), 316(e); *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1375 (Fed. Cir. 2016). To satisfy that burden, a petitioner must “articulate specific reasoning, based on evidence of record, to support the legal conclusion of obviousness”—“mere conclusory statements” will not suffice. *Magnum Oil*, 829 F.3d at 1380. Indeed, “[i]t is of the utmost importance that petitioners in the IPR proceedings adhere to the requirement that the initial petition identify ‘with particularity’ the ‘evidence that supports the grounds for the challenge to each claim.’” *Intelligent Bio-Sys., Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1369 (Fed. Cir. 2016) (quoting 35 U.S.C. § 312(a)(3)).

As explained below, Petitioner has not satisfied its burden with respect to any of its obviousness grounds. To the contrary, Petitioner takes numerous

shortcuts in its analysis, resulting in an underdeveloped, unclear, and fatally deficient obviousness challenge.

A. Grounds 1 and 4: Petitioner Fails to Show that Abarbanel and Kobayashi Render the Challenged Claims Obvious.

Ground 1, covering independent claims 1, 7, and 8, and dependent claims 2 and 9, relies on the combination of Abarbanel and Kobayashi. But Petitioner has not shown either that (1) the prior-art combination discloses the claims’ “property” limitation, or (2) a POSA would have combined the references in the manner Petitioner suggests. Either of these deficiencies provides a basis for rejecting Petitioner’s Ground 1 arguments. And, because these same deficiencies also apply to Ground 4, the Board should reject that Ground as well.

1. Petitioner Fails to Show that Kobayashi Discloses the Claims’ “Property” Limitation.

Each independent claim requires that an IC “property” be created from an IC specification and used to “verif[y] the specification.” EX1001, 13:40-44 (claim 1), 14:39-43 (claim 7), 15:4-8 (claim 8), 16:26-30 (claim 14). Petitioner relies solely on Kobayashi to satisfy this limitation. In particular, Petitioner argues that Kobayashi’s “flowchart” (i.e., a graphical representation of transitions between IC states) corresponds to the claimed specification, and that Kobayashi’s “statelist” (i.e., a textual description of the various states depicted in the flowchart) comprises a list of claimed properties. Pet., 28-29. But this mapping is fundamentally flawed

for a simple reason: Kobayashi's statelist (the alleged list of properties) is not used to "verify" Kobayashi's flowchart (the alleged specification), as the claims require.

In the context of the '636 patent, "verify" means that the claimed property is used to troubleshoot an IC—i.e., to ensure that the IC functions in a manner described in its specification. *See, e.g.*, EX1001, 1:8-19, 1:28-29, 4:35-38, 5:30-56, 7:61-64, 8:4-20. But Kobayashi does not perform such verification. To the contrary, Kobayashi is directed to IC *design*, not IC verification. *See, e.g.*, EX1005, 1:9-13 ("This invention relates to the design of integrated circuits[.]"), 2:15-20 ("Thus, the present invention, for the first time, opens the possibility for the design and production of ASICs by designers, engineers and technicians who may not possess the specialized expert knowledge of a highly skilled VLSI design engineer.").

This distinction is important because the entire purpose of Kobayashi's statelist is to generate "topological characteristics of the integrated circuit chip" that are "needed for fabricating the chip." *Id.*, Abstract, 1:38-44. That purpose has nothing to do with verifying the IC, let alone verifying flowcharts (the alleged IC specifications). At best, the statelist *reflects* state transitions depicted *in* the flowcharts. But that is a far cry from "verifying" the flowcharts themselves. A transcript of an oral hearing, for example, might reflect the parties' and judges' statements made during the hearing, but the transcript does not "verify" the hearing

in any reasonable sense of the word.

Petitioner misleadingly cites (at 28) a passage of Kobayashi describing “simulation and verification of flowcharts,” EX1005, 4:56-60, to suggest that Kobayashi’s statelist performs the verification. But Petitioner conveniently omits the portion of the passage specifying that a “*flowchart simulator*” is the entity that verifies the flowcharts, *not* the statelist. *See id.* (noting that the system includes “a flowchart simulator 22 for simulation and verification of flowcharts”). Thus, to the extent Kobayashi “verifies” its flowcharts at all, it does not use its statelist to do so.

In short, Petitioner has pointed to no evidence suggesting that Kobayashi’s statelist verifies its flowcharts. This deficiency alone dooms Petitioner’s obviousness challenge is Grounds 1 and 4.¹

2. Petitioner Fails to Establish that a POSA Would Have Combined Kobayashi with Abarbanel.

Grounds 1 and 4 fail for the additional reason that Petitioner does not set forth a cogent and legally supportable theory for combining Abarbanel and Kobayashi.

A proper motivation-to-combine analysis requires “specific reasoning, based

¹ And, because the claims require that the claimed IC “assertion” be automatically generated from the claimed property, Petitioner has likewise failed to show that Grounds 1 and 4 satisfy the “assertion” limitation.

on evidence of record,” *Magnum Oil*, 829 F.3d at 1380, including a reasoned “explanation as to *how* or *why* the references would be combined to produce the claimed invention,” *Trivascular, Inc. v. Samuels*, 812 F.3d 1056, 1066 (Fed. Cir. 2016) (quoting *Kinetic Concepts, Inc. v. Smith & Nephew, Inc.*, 688 F.3d 1342, 1368 (Fed. Cir. 2012)). Importantly, “[t]he obviousness inquiry does not merely ask whether a skilled artisan *could* combine the references, but instead asks whether they *would* have been motivated to do so.” *Adidas AG v. Nike, Inc.*, 963 F.3d 1355, 1359 (Fed. Cir. 2020) (cleaned up and emphasis added). In other words, specific evidence and rationale must be proffered to show why a POSA would have picked out the references and combined them in such a way as to arrive at the claimed invention. *Personal Web Techs., LLC v. Apple, Inc.*, 848 F.3d 987, 993-994 (Fed. Cir. 2017).

Petitioner has not satisfied that standard here. As explained below, Petitioner’s arguments for combining Kobayashi with Abarbanel are insolubly ambiguous and unclear. And, to the extent the arguments can be understood, they impermissibly rest on obviousness theories that defeat the very objective of Abarbanel’s IC-verification solution. Petitioner’s arguments should be soundly rejected.

a. Petitioner’s Obviousness Combination Is Unclear and Confusing.

To start, Petitioner is vague about what its proposed combination even is. In

certain places, the Petition asserts that a POSA would have found it obvious to use Kobayashi's statelist (i.e., the alleged list of "properties") as an input in Abarbanel's FoC (the alleged "assertion generating unit") in order to generate the claimed assertion. *See, e.g.,* Pet., 16 ("The combination involved the simple substitution of one known element (Kobayashi's statelist language) for another (RCTL, as taught by Abarbanel) to obtain a predictable result"), 17 ("It was natural to use Kobayashi's teachings ... to ... create a statelist of properties, as input to Abarbanel's methodology"), 29 (referring to Kobayashi's "statelist properties" being converted into an assertion by way of Abarbanel's FoC), 32 (arguing that Kobayashi's statelist entries are provided "as input to" Abarbanel's system for conversion into an assertion). This theory of obviousness is predicated on Petitioner's contention that a POSA would have improved Abarbanel's system based on Kobayashi's teachings. *See, e.g., id.*, 16 ("The combination applies Kobayashi's known techniques to Abarbanel's known methodology, which was ready for improvement" (citation omitted)).

But elsewhere, the Petition argues that a POSA would have first modified Kobayashi's statelist into RCTL code, and then used *that* code as an input in Abarbanel's FoC. *See, e.g., id.*, 16 ("Kobayashi's teachings would thus be adapted to use RCTL as a simple substitution."), 26 (arguing that a POSA would have "express[ed] Kobayashi's statelist/properties using RCTL[]"), 27 (arguing that a

POSA would have used “RCTL for Kobayashi’s statelist”), 31 (“Ground 1 uses RCTL for the statelist.”). This second theory of obviousness is predicated on Petitioner’s contention that a POSA would have improved *Kobayashi’s* system based on *Abarbanel’s* teachings—i.e., the exact opposite premise on which the first theory is based. *See, e.g., id.*, 16 (“[A] POSITA would have been motivated to make this straightforward combination because Kobayashi’s teachings were ready for improvement.”).

These are two different obviousness theories. And yet, Petitioner does not clearly adopt one or the other, nor present them as alternative theories. Petitioner’s inability to clearly (and consistently) articulate its proposed combination is fatal to its challenge.

The Federal Circuit’s decision in *Sisvel International S.A. v. Sierra Wireless, Inc.*, 82 F.4th 1355 (Fed. Cir. 2023), is instructive on this point. There, the court affirmed the Board’s ruling upholding the claims at issue where the petitioners’ arguments were “inconsistent as to how” the prior art references (Chen and GSM) “should be combined.” *Id.* at 1364. “In some instances,” the petitioners argued “that a skilled artisan would improve Chen with ... GSM.” *Id.* Yet, “[i]n other instances,” the petitioners argued the exact opposite—i.e., “that a skilled artisan would improve ... GSM ... with Chen.” *Id.* In light of this inconsistency, the court was “at a loss in trying to decipher” the petitioners’ “kitchen-sink of unclear and

confusing motivation-to-combine arguments.” *Id.*

Similarly here, Petitioner’s kitchen-sink approach has led to obviousness arguments that are “unclear and confusing” (to say the least). Indeed, it is difficult to see how Petitioner could possibly meet its burden to show that the ’636 patent claims are obvious when Petitioner itself cannot even articulate its obviousness theory with sufficient particularity. *Magnum Oil*, 829 F.3d at 1375. Petitioner’s arguments should be rejected.

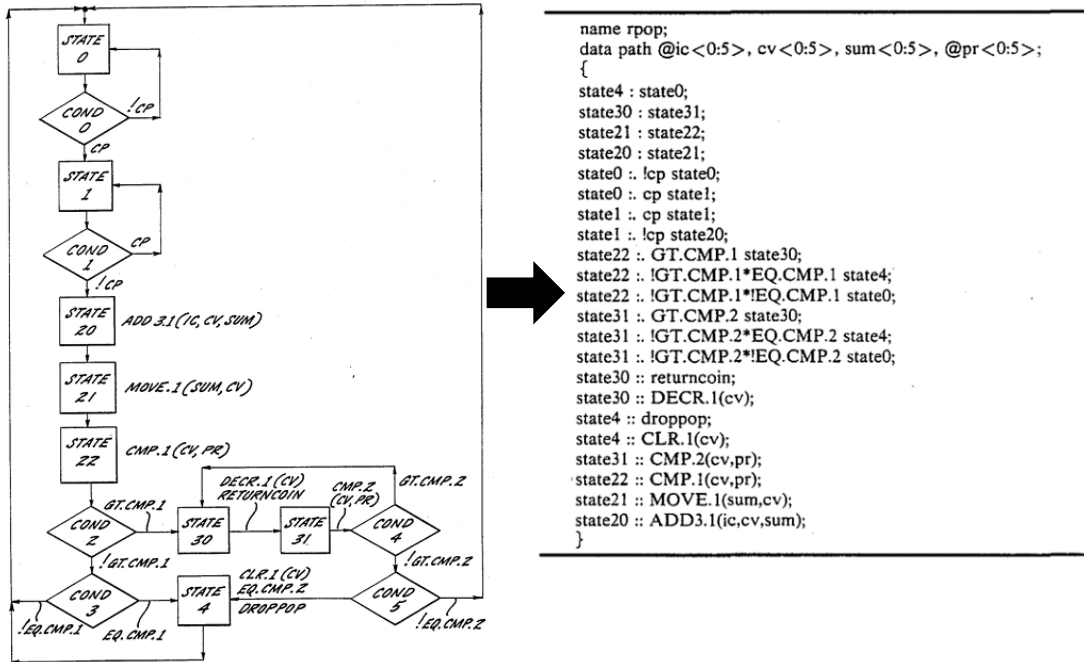
b. Petitioner’s Proposed Combinations Destroy Abarbanel’s Principal Purpose and Impermissibly Blur Statelist Data with RCTL Data.

In any event, both of Petitioner’s obviousness theories fail, as they improperly blur the distinction between Kobayashi’s statelists and Abarbanel’s RCTL—a distinction that is critical to Abarbanel’s IC-verification system.

To the extent Petitioner argues that a POSA would have simply taken Kobayashi’s statelists and plugged them into Abarbanel’s FoC, that argument ignores that Abarbanel is predicated on using RCTL, *not* statelists. The two are fundamentally different. *See Adidas*, 963 F.3d at 1359 (“Fundamental differences between the references are central to th[e] motivation to combine inquiry.”).

As shown below, statelists comprise a series of short, declarative statements that describe an IC’s operational states, as well as conditions that trigger changes to such states. EX2001, ¶25. In effect, statelists describe in text what flowcharts

depict in diagrammatic form:



EX1005, FIG. 10, 14:8-29.

EX2001, ¶¶25-26.

RCTL by contrast, is a specification language comprised of Boolean formulas that express certain IC conditions. EX1023, 1 (“[W]e define the specification language RCTL which is an extension of CTL ... to express hardware properties that are difficult to express in CTL.”). Abarbanel provides two examples of such conditions:

$\text{forall } k: {}^2$
 $\{*, \text{start} \ \& \ \text{start_tag}=t \ \& \ \text{to_send}=k, \text{!end*}, \text{end} \ \& \ \text{end_tag}=t\} \rightarrow \{\text{sent}=k\}$

$\{*, \text{read}, \{\text{!snoop*}, \text{snoop}, \text{!snoop*}, \text{snoop}\} \ \& \ \{\text{!write*}\} \} \rightarrow \{\text{cover}\}$

EX1004, 3-4. The first example describes a condition requiring that “k” bytes be

sent at the end of time “t.” *Id.*, 3. The second example describes a condition requiring that a “snoop event” be performed twice between “read” and “write” commands. *Id.*, 4. Neither of these examples looks anything like Kobayashi’s statelist. And, as Mr. Berg explains, RCTL and statelists are *not* interchangeable, nor does one imply the other. EX2001, ¶¶22, 28.

This distinction is significant because Abarbanel focuses exclusively on IC properties expressed *as RCTL*. This is not by accident. Abarbanel emphasizes throughout its disclosure that RCTL provides a “conciseness,” “expressiveness,” and “ease of use” that is essential to Abarbanel’s IC-verification solution:

- “*RCTL includes a rich and useful set of CTL safety formulas and regular expressions.*” EX1004, 1 n.1 (emphasis added).
- “An equally important benefit of FoCs is *the conciseness and expressiveness of RCTL formulas.*” *Id.*, 2 (emphasis added).
- “The following example will demonstrate *the conciseness and ease of use of RCTL* for checker writing and the advantage of automatic checker generation.” *Id.*, 3 (emphasis added).
- “Thousands of FoCs checkers were written so far by virtue of the *great ease of writing RCTL specifications* and translating them into FoCs checkers.” *Id.*, 4 (emphasis added).

- “Our own experience has demonstrated that the *simple, concise syntax and semantics of RCTL*, coupled with online simulation checking, *is highly useful* to the verification teams with whom we work.” *Id.* (emphasis added).

EX2001, ¶¶18-20. In other words, Abarbanel’s FoCs are designed specifically to process IC properties expressed in RCTL (and not in any other language or format) in order to harness the benefits that RCTL provides. Stated differently, in Abarbanel’s system, RCTL is not merely a convenient programming language that the authors just happened to select to implement Abarbanel’s solution—*RCTL is an integral part of Abarbanel’s solution itself. Id.* ¶21.

Accordingly, modifying Abarbanel’s FoCs to process statelists instead of RCTL would eliminate *the very benefits* that Abarbanel expressly touts and is designed to achieve. *Id.* ¶30. Contrary to Petitioner’s conclusory assertion otherwise (at 17), such a modification would destroy the “entire premise” of Abarbanel’s system, which “counsel[s] strongly against obviousness.” *Medtronic, Inc. v. Teleflex Innovations S.a.r.l.*, 69 F.4th 1341, 1349-1350 (Fed. Cir. 2023); *Chemours Co. v. Daikin Indus., Ltd.*, 4 F.4th 1370, 1376 (Fed. Cir. 2021) (rejecting obviousness combination that “would destroy the basic objective” of the primary reference); *Adidas*, 963 F.3d at 1359 (finding no motivation to combine where the obviousness combination would “change the principles under which” the primary

reference operates); *Intel Corp. v. Koninklijke Philips N.V.*, 2024 WL 725243, at *1 (Fed. Cir. 2024) (similar).

Not only does Abarbanel use RCTL, it expressly *distinguishes* IC-verification systems that use *statelists* (i.e., the system described in Schlipf (EX1029)):

Schlipf et al describe a methodology and tool that inspired our work. They unify formal verification and simulation efforts by automatically translating *state machines* which represent the environment specification either to simulation behavioral models or to input for a model checker. ... *In contrast, our solution represents the specification as temporal logic formulas rather than state machines, for the sake of conciseness and readability.*

EX1004, 5 (emphasis added); *see also* Pet., 16 (conceding that a “statelist describes a state machine”). In other words, Abarbanel emphasizes the undesirability of the very combination that Petitioner proposes.² Abarbanel’s

² Petitioner misleadingly asserts that the cited passage “suggests using RCTL to describe state machines.” Pet., 26. Quite the opposite—the passage suggests that RCTL should be used *instead* of state machines. In a similar vein, Petitioner asserts that Schlipf (the reference cited in the above-quoted passage) teaches “automatically generating CTL/RCTL to describe state machines for verification.” *Id.*, 27. This is also misleading. Schlipf does not mention RCTL at all.

strong preference for RCTL over statelists would have dissuaded a POSA from using Kobayashi's statelists in Abarbanel's system. EX2001, ¶¶24, 27-28; *see Polaris Indus., Inc. v. Arctic Cat, Inc.*, 882 F.3d 1056, 1069 (Fed. Cir. 2018) (“[E]ven if a reference is not found to teach away, its statements regarding preferences are relevant to a finding regarding whether a skilled artisan would be motivated to combine that reference with another reference.”).

Petitioner's apparent second obviousness theory—that Kobayashi's statelists can be modified and converted into RCTL before being input into Abarbanel's FoCs—also fails. As explained above, statelists and RCTL are fundamentally different. Whereas the former describes an IC's state transitions using short textual statements, the latter describes specific IC conditions using formulas.

Petitioner does not even try to explain how one can be converted to the other, let alone establish that such conversion would be successful. Instead, Petitioner simply declares—without any evidentiary support—that conversion to RCTL is possible simply because RCTL was “known in the art.” Pet., 27. But such “conclusory statements” are insufficient to meet Petitioner's burden of proof. *Magnum Oil*, 829 F.3d at 1380; *see also infra* Section V.A.2.c.

In fact, Abarbanel itself suggests that converting statelists into RCTL “is *not* trivial,” at least in some cases. *See* EX1004, 3 n.2 (noting that the “implementation” of RCTL for Abarbanel's first example described above “is not

trivial”). Mr. Berg agrees, explaining that RCTL was difficult to learn and master.

EX2001, ¶¶23, 29. Thus, even if there were reasons to convert Kobayashi’s statelists into RCTL (Petitioner provides none), such conversion would be prohibitively complicated and would dissuade a POSA from doing so. *Provisur Techs., Inc. v. Weber, Inc.*, 2022 WL 17688071, at *5 (Fed. Cir. 2022) (affirming Board’s obviousness ruling where evidence showed that proposed modification would “add[] needless complexity” to the system).

c. Petitioner’s Remaining Arguments Amount to Conclusory and Generic Assertions About What Was “Known” and “Common” in the Art.

With its principal obviousness theories stripped away, Petitioner is left with only generic and conclusory statements about why a POSA allegedly would have combined Kobayashi and Abarbanel. Petitioner reasons, for example, that because Kobayashi’s and Abarbanel’s teachings were “known” and “common” in the art, a POSA would have combined them. Pet., 14-15 (“*[I]t was common* for engineers to use graphical representations of specifications, for example flowcharts that described IC requirements. *Therefore*, the prior art provides motivation to combine Abarbanel with teachings for graphically editing IC specifications, including flowcharts.” (citations omitted and emphases added)); *see generally id.*, 16-18, 26-29.

But this “short-cut logic” cannot satisfy Petitioner’s burden, as it would

impermissibly “lead to the conclusion that any and all combinations of elements known in this broad field would automatically be obvious, without the need for any further analysis.” *Securus Techs., Inc. v. Glob. Tel*Link Corp.*, 701 F. App’x 971, 977 (Fed. Cir. 2017). The recognition that the prior art’s teachings were “known” at best suggests that the teachings *could* be combined. But, as explained above, “[t]he obviousness inquiry does not merely ask whether a skilled artisan *could* combine the references”—rather, it “asks whether they *would* have been motivated to do so.” *Adidas*, 963 F.3d at 1359 (cleaned up and emphasis added); *Virtek Vision Int’l ULC v. Assembly Guidance Sys., Inc.*, 97 F.4th 882, 886 (Fed. Cir. 2024) (“It does not suffice to meet the motivation to combine requirement to recognize that two alternative arrangements ... were both known in the art.”). Petitioner has not performed that analysis here.

In a similar vein, again employing a kitchen-sink approach, Petitioner invokes all of *KSR*’s generic rationales for combining prior-art references to conclude that a POSA would have combined Abarbanel and Kobayashi, specifically. For example, the Petition states that “[g]raphically editing specifications” as taught by Kobayashi “was one of a finite number of identified predictable solutions” and thus would have been used to generate a statelist for use in Abarbanel’s FoCs. Pet., 15. But, aside from referring in passing to “editing text,” the Petition does not describe what those finite number of purported

solutions are. *Id.* Neither does Dr. Ghiasi—his declaration, like the Petition, is conclusory. EX1003, ¶84; *Xerox Corp. v. Bytemark, Inc.*, IPR2022-00624, Paper 12, at 5 (P.T.O. Dir. 2023) (rejecting expert testimony that merely “sets forth Petitioner’s conclusory assertions as though they are facts, rather than setting forth facts and evidence in support of Petitioner’s assertions”); *TQ Delta, LLC v. Cisco Sys., Inc.*, 942 F.3d 1352, 1358 (Fed. Cir. 2019) (“Conclusory expert testimony does not qualify as substantial evidence.”).

As another example, the Petition summarily asserts that “Kobayashi’s teachings were ready for improvement.” Pet., 16. But the Petition does not explain which of Kobayashi’s teachings ought to be improved, nor describe how such teachings would be improved by any teaching from Abarbanel. (And, as explained above, that statement directly conflicts with other statements suggesting Kobayashi’s teachings would be used to improve Abarbanel. *See supra* Section V.A.2.a.)

These are exactly the type of conclusory assertions that the Federal Circuit has rejected as insufficient to satisfy a petitioner’s burden. In *Virtek Vision*, for example, the court reversed the Board’s obviousness determination where the petitioner relied on *KSR*’s generic rationales without providing any supporting evidence, including “evidence that there are a finite number of identified, predictable solutions.” 97 F.4th at 888. Those same arguments are insufficient

here. The obviousness analysis is flexible, to be sure, but a bare invocation of generic motivation-to-combine rationales is no substitute for the reasoned, evidence-based analysis required to sustain Petitioner's burden. *Id.*; *see also Trivascular*, 812 F.3d at 1066 (“Although the *KSR* test is flexible, the Board must still be careful not to allow hindsight reconstruction of references ... without any explanation as to how or why the references would be combined to produce the claimed invention.” (cleaned up)).

B. Grounds 2 and 5: Petitioner Fails to Show that Narain and Kobayashi Render the Challenged Claims Obvious.

Ground 2, covering independent claims 1, 7, and 8, and dependent claims 2 and 9, is nearly identical to Ground 1 except that it replaces Abarbanel with Narain. Much like its failure with regard to Grounds 1 and 4, Petitioner fails to show both that (1) Kobayashi and (2) Narain disclose the claims' “property” limitation, and (3) a POSA would have combined the references in the manner that Petitioner suggests. And, because these same deficiencies also apply to Ground 5, the Board should reject both grounds.

1. Petitioner Fails to Show that Kobayashi Discloses the Claims' “Property” Limitation.

In Grounds 2 and 5, Petitioner once again relies on Kobayashi's statelist as satisfying the “property” limitation. Pet., 48. But, as explained above in connection with Ground 1, Kobayashi is not concerned with IC verification, and Kobayashi's

statelists are not used to “verify” Kobayashi’s flowcharts (the alleged IC specifications), as required by the claims. *See supra* Section V.A.1. Accordingly, Petitioner’s arguments for Ground 2 and 5 fail for the same reasons as its arguments for Ground 1 and 4 fail.

2. Petitioner Also Fails to Show that Narain Discloses the Claims’ “Property” Limitation.

Petitioner also seems to rely on Narain for teaching the claimed property, but Petitioner’s arguments are unclear. In certain places, the Petition points to Narain’s “plain text ‘condition or statement’ that defines operations intended and expected in design data to be verified,” and, in particular, Narain’s “exemplary ‘vx’ directives for specifying ‘design constraints.’” Pet., 48-49. Elsewhere, however, the Petition points to Narain’s Sentry verification entities. *Id.*, 50. Again, this lack of clarity dooms Petitioner’s challenge. *See supra* Section V.A. In any event, as explained below, both sets of arguments fail on their own terms.

a. “VX” Directives for Built-In Assertions

Petitioner first seems to rely (at 48-49) on Narain’s “vx” directives, which Narain describes as being used in “built-in assertions”:

TABLE 1

<u>Built-in Assertions</u>	
Annotation	Meaning
// vx onehot (i, j, k);	Exactly one of i, j, or k must be high
// vx onecold (i, j, k);	Exactly one of i, j, or k must be low
// vx onetrue (i, j, k);	At most one of i, j, or k can be high
// vx onefalse(i, j, k);	At most one of i, j, or k can be low

EX1006, TBL. 1.

This mapping fails for several reasons. First, as their name suggests, Narain’s built-in assertions are IC *assertions*, not properties. The built-in-assertions are “used in annotations to simplify assertion coding” and “cover common behaviors that many signal groups exhibit” to “allow the designer to explicitly specify design constraints for annotated signals.” *Id.*, 15:22-23, 15:40-41; *see also id.*, 18:24-31 (describing “assertions” as “condition[s] that a designer explicitly indicated should occur at a particular point in the design”). Narain’s description on this score mirrors the ’636 patent’s own description. *See* EX1001, 1:55-63 (describing an assertion as reflecting the designer’s “intention,” which, in turn is “an assumption and a precondition of circuit input”). Accordingly, Petitioner errs in equating Narain’s assertions with the claimed properties. *See* EX2001, ¶¶12-17.

Second, Narain’s built-in assertions are not read from the specification design data, as required by the claims. *See* EX1001, 13:40-44 (claim 1), 14:39-43 (claim 7), 15:4-9 (claim 8), 16:26-30 (claim 14). In Narain, the built-in assertions

take the form of annotations that the designer manually “plac[es] ... within the RTL code itself,” and “[s]uch assertions allow the designer to explicitly specify design constraints for annotated signals.” EX1006, 15:16-18, 15:40-41. In other words, the designer adds the built-in assertions manually—they are not read from the IC specification or design data.

In fact, Narain’s manual generation of IC assertions is precisely what the ’636 patent disparages and distinguishes from the scope of its invention. In particular, the patent explains that:

[I]n a case **where the assertion description language is written by a manual input** when the specifications are defined, **there is a possibility that a mistake may be made in the assertion itself**. As a result, a report of the warning of the assertion violation and the function coverage obtained by the assertion verification cannot be fed back to prevent the system revision and the revised edition of the system. Further, this requires a verification period for debugging the assertion description language. **Consequently, the verification efficiency becomes worse**.

EX1001, 3:43-53 (emphases added). To improve verification efficiency and eliminate such mistakes, the ’636 patent employs an *automatic* approach to assertion generation, which as discussed above, is reflected in the claims.

Accordingly, the patent disclaims the very manual assertion generation that is disclosed in Narain. *See Openwave Sys., Inc. v. Apple Inc.*, 808 F.3d 509, 514 (Fed.

Cir. 2015) (“[T]he specification of the patents-in-suit is rife with remarks that disparage and, therefore, disclaim mobile devices that incorporate computer modules.”).

For these reasons, Narain’s “vx” directives, in the form of built-in assertions, do not satisfy the claims’ “property” limitation.

b. Predetermined Properties for Sentry Verification Entities

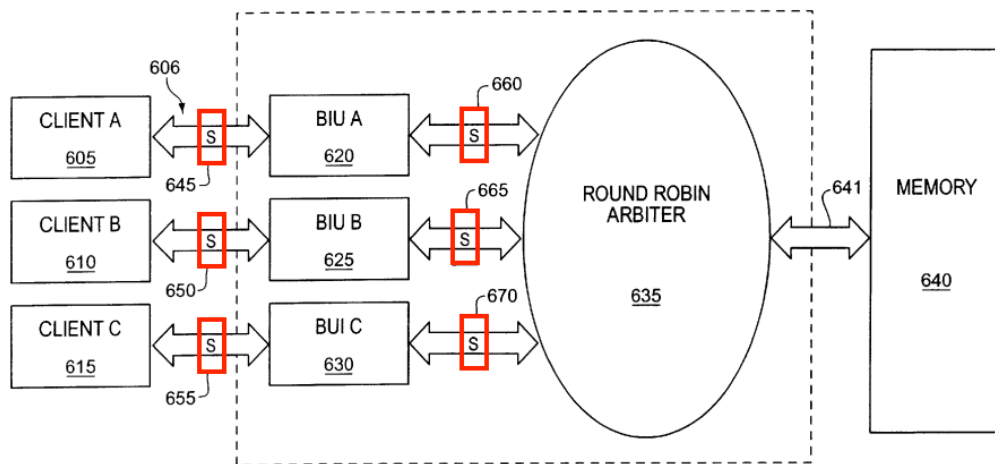
Elsewhere (at 50), the Petition also appears to point to Narain’s “Sentry verification entities” and “properties that are used to generate the design verification checks.” According to the Petition, Narain’s “[d]esign verification checks ... are automatically formulated based upon the predetermined properties.” Pet., 52 (citing EX1006, 17:7-21). But Narain’s “predetermined properties” likewise do not satisfy the “property” limitation, as those properties also are not read from the specification design data, as required by the claims.

Narain’s “predetermined properties” relate to declaring a Sentry verification entity. EX1006, 15:56-58. The designer may declare a Sentry verification entity, which creates a “sentinel variable,” by adding “simpl[e] annotations indicative of the expected state of variables at various points in the control flow structure of a finite state machine associated with the hardware design representation.” *Id.*, 13:30-43.

reg a;	// vx sentry a: clk;
wire b;	// vx sentry b: clk;
reg [0:15] c;	// vx sentry [1:2] c: clk;

Id., 16:21-25. Figure 6B, for example, shows the inclusion of Sentry verification entities 645, 650, 655, 660, 665, and 670 (in red) in the design of Figure 6A.

FIG. 6B



EX1006, FIG. 6B (annotated).

For each Sentry verification entity, Narain discloses testing a small, predetermined set of properties that it determined to be essential for verification: “Empirical analysis performed by the assignee has shown that verification of a relatively small set of properties at key points of a hardware design representation ... can guarantee the absence of errors.” *Id.*, 17:33-38. For example, the first predetermined property Narain discusses is “AID (Access of Invalid Data).” Narain explains a “Sentry verification entity that is in the valid state should not receive data that flowed through a Sentry verification entity that was in the

invalid state.” *Id.*, 17:58-61. Other “predetermined properties” include:

loss of valid data (LVD), assertion correctness (AC), conflicting assignments (CA), block enable (BE), assignment execution (AX), constant value memory element (CME), constant value variable (CV), Sentry activate always off (AAO).

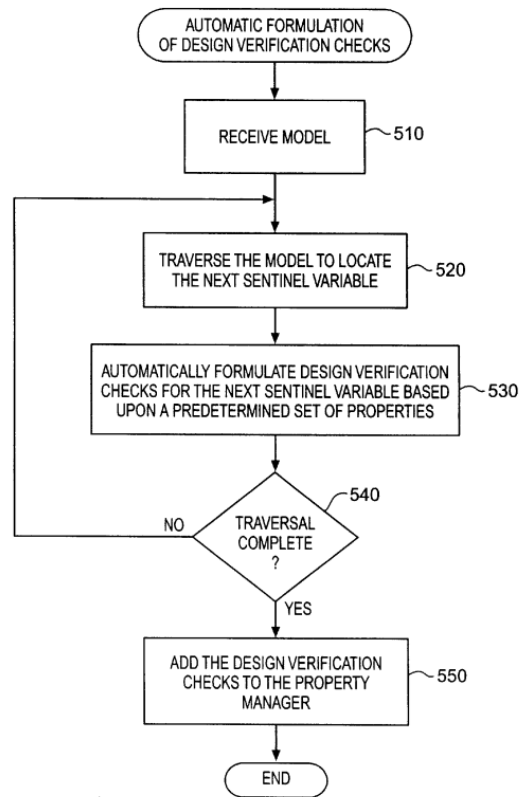
Id., 17:53-56; *see also id.*, 17:58-18:52 (describing these properties).

Figure 5 of Narain shows how the “predetermined properties” are used, where:

Upon locating the next sentinel variable, at processing block 530, design verification checks for the sentinel variable are automatically formulated based upon the predetermined properties. For example, each property may be expressed as one or more checks.

Id., 17:14-18; *see also id.*, 16:64-17:6.

FIG. 5



EX1006, FIG. 5.

Based on this disclosure, it is clear that Narain’s “predetermined properties” used for Sentry verification entities are not read from an IC specification’s design data, as required by the claims. Rather, they are a fixed set of conditions (e.g., AID, LVD) that must be checked for each Sentry verification entity.

Accordingly, Narain’s “predetermined properties” likewise do not satisfy the “property” limitation.³

³ And, because the claims require that the claimed IC “assertion” be

3. Petitioner Fails to Establish that a POSA Would Have Combined Kobayashi with Narain.

Even apart from the prior art’s failure to disclose the “property” limitation, Petitioner also fails to show why or how a POSA would have combined the references. Petitioner’s arguments on this score are (once again) unclear, confusing, and conclusory.

a. Petitioner’s Obviousness Combination Is Unclear and Confusing.

Similar to its argument in Grounds 1 and 4, Petitioner’s motivation-to-combine arguments for Grounds 2 and 5 are all over the map. On the one hand, Petitioner argues that a POSA would have found it obvious to simply plug Kobayashi’s statelist (the alleged list of properties) into Narain’s Intent-Driven Verification architecture (the alleged “assertion generating unit”) in order to generate an assertion. Pet., 42-44, 46, 49, 51, 52. On the other hand, Petitioner elsewhere argues that a POSA would have converted Kobayashi’s statelist into the “vx” notation, and plug *that* modified statelist into Narain’s Model Builder. *Id.*, 44, 49-50.

And, like its motivation-to-combine arguments with respect to Grounds 1

automatically generated from the claimed property, Petitioner has likewise failed to show that Grounds 2 and 5 satisfy the “assertion” limitation.

and 4, Petitioner’s arguments here are inconsistent as to which reference is being used to improve the other. In some places, the Petition asserts that Kobayashi’s teachings are being used to improve to Narain. *Id.*, 42 (“Ground 2 applies Kobayashi’s known techniques to Narain’s known verification methodology, which was ready for improvement ...”). In other places, the Petition asserts that Narain is being used to improve Kobayashi. *Id.*, 43 (“Kobayashi’s teachings were ready for improvement. ... Narain’s assertion-based verification improved on Kobayashi ...”).

These arguments are irreconcilable with one another. And, as discussed above, the Petition’s lack of clarity alone provides a basis to reject Petitioner’s motivation-to-combine arguments. *Supra* Section V.A.2.a; *Sisvel*, 82 F.4th at 1364.

b. Petitioner’s Arguments for Combining Narain and Kobayashi Are Wholly Conclusory and Generic.

Petitioner also recycles—nearly verbatim—the same conclusory and generic obviousness rationales proffered in Grounds 1 and 4 to support the argument that a POSA would have combined Narain and Kobayashi. For example, Petitioner asserts that Kobayashi’s “[g]raphically-edited flowcharts were common” and comprised “one of a finite number of known, predictable solutions for editing a specification.” Pet., 42; *see generally id.*, 41-46, 49-50. But, again, Petitioner does not explain why a POSA would have used Kobayashi’s statelist, in particular, in Narain’s system or what the allegedly finite number of solutions actually are.

As explained above, the absence of evidence-backed analysis provides an additional reason for rejecting Petitioner’s motivation-to-combine arguments. *See Virtek Vision*, 97 F.4th 882, 886-889; *supra* Section V.A.2.c.

C. Grounds 3 and 6: Petitioner Fails to Show that Abarbanel and Feyerabend Render the Challenged Claims Obvious.

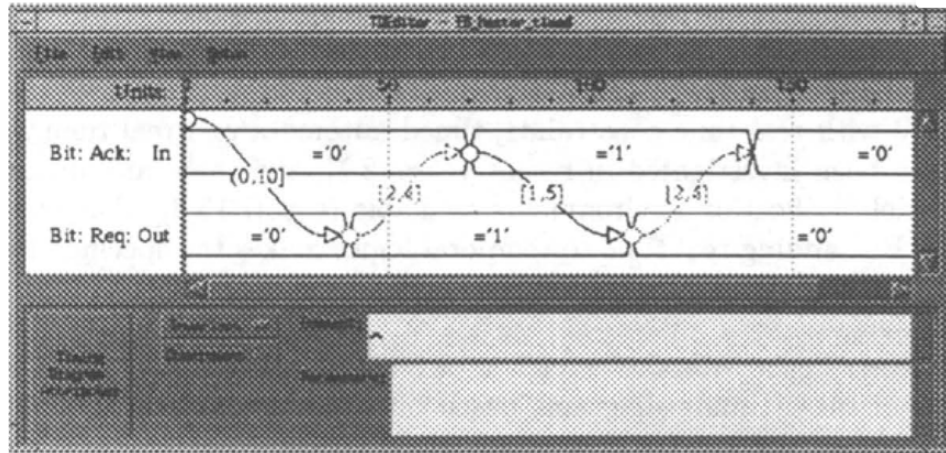
Ground 3, covering independent claims 1, 7, and 8, and dependent claims 2 and 9, is likewise nearly identical to Ground 1 except that it replaces Kobayashi with Feyerabend. Once again, Petitioner’s arguments fail. As explained below, Petitioner does not show that (1) the prior-art combination discloses the claims’ “graphically editing” limitation, and (2) a POSA would have combined the references in the first place. And, because these same deficiencies also apply to Ground 6, the Board should reject that Ground as well.

1. Petitioner Fails to Show that Feyerabend Discloses the Claims’ “Graphically Editing” Limitation.

Each independent claim of the ’636 patent requires that IC design data be generated “by graphically editing the specification.” EX1001, 13:34-35 (claim 1), 14:33-34 (claim 7), 14:67-15:1 (claim 8), 16:23-24 (claim 14). In Grounds 3 and 6, Petitioner relies solely on Feyerabend for this limitation.

In particular, Petitioner asserts that Feyerabend’s timing waveform comprises the claimed IC specification, and then points to Feyerabend’s screenshot of a waveform to conclude that Feyerabend discloses software which allows a user

to graphically edit the waveform:



Pet., 64-65 (citing EX1007, FIG. 1). Not so. Feyerabend provides a single screenshot of a waveform but does not describe the software that displays the waveform nor its capabilities. EX1007, 3. Feyerabend simply does not disclose or suggest that its waveforms (the alleged IC specifications) can be “graphically edited.”⁴

Perhaps recognizing this deficiency in Feyerabend, Petitioner pivots to *a different reference that is not even part of Petitioner’s obviousness combination*, Gupta (EX1027), for the proposition that “graphically editing timing diagrams was

⁴ This is true under either party’s proposed construction of “graphically editing.” Compare EX1031, Ex. A, 1 (Petitioner’s construction: “editing a visual expression, as opposed to editing text”), with EX1032, 6 (Patent Owner’s construction: “editing with a graphical user interface”).

known.” Pet., 65. But Gupta only makes a single general statement that “[a] graphics editor for handling timing diagrams” has “been developed [to] facilitate the task of specification and verification.” EX1027, 62. Gupta does not speak to the screenshot shown in Feyerabend, let alone suggest that Feyerabend’s waveforms can be graphically edited. And, notably, Feyerabend distinguishes “conventional timing diagrams [] widely used in hardware design” from “symbolic timing diagrams.” EX1007, 1.

Further, Petitioner’s suggestion that Gupta’s graphical-editing technique could be used in Feyerabend merely because the technique “was known” in the art is insufficient to support a motivation to combine for all the reasons described above. *See supra* Section V.A.2.b; *Virtek Vision*, 97 F.4th 882, 886-889.

Dr. Ghiasi’s testimony on this score is of little help. *See* EX1003, ¶280. He merely “sets forth Petitioner’s conclusory assertions as though they are facts, rather than setting forth facts and evidence in support of Petitioner’s assertions.” *Xerox*, IPR2022-00624, Paper 12, at 5; *TQ Delta*, 942 F.3d at 1358.

In the absence of facts and evidence, Petitioner cannot satisfy its burden to show that the prior-art satisfies the “graphically editing” limitation.

2. Petitioner Fails to Establish that a POSA Would Have Combined Feyerabend with Abarbanel.

Even apart from the prior art’s failure to disclose the “graphically editing” limitation, Petitioner’s also fails to show why or how a POSA would have

combined the references.

a. Petitioner’s Obviousness Combination Is Again Unclear and Confusing.

Much like Petitioner’s arguments for Ground 1, Petitioner argues here that a POSA would have found it obvious to use Feyerabend’s TPTL formulas (i.e., the alleged “properties”) in Abarbanel’s FoCs (the alleged “assertion generating unit”) to generate an assertion. Pet., 60-61. But, once again, the Petition is unclear as to whether the proposed combination requires Abarbanel’s system to be modified to accommodate Feyerabend’s formulas, *see id.*, 61-62, or whether Feyerabend’s formulas would somehow be converted from TPTL to RCTL, *see id.*, 62-63, 69-70.

What is more, Petitioner simultaneously argues—in back-to-back sentences—that Abarbanel would be used to improve Feyerabend *and* that Feyerabend would be used to improve Abarbanel. *Id.*, 61 (“Adding Feyerabend’s timing diagram editor to Abarbanel simplifies the design/verification process by enabling a more user-friendly editor, while improving verification by generating checkers automatically. Adding Abarbanel’s FoCs to Feyerabend reduces cost and labor.”).

As discussed above, Petitioner’s inconsistent and confusing arguments doom its obviousness challenge. *Supra* Section V.A.2.a; *Sisvel*, 82 F.4th at 1364.

b. Petitioner’s Arguments for Combining Feyerabend and Abarbanel Are (Once Again) Conclusory and Generic.

Also like its motivation-to-combine arguments in the previous grounds, Petitioner’s arguments here are replete with conclusory and generic statements that lack any analysis or evidentiary support. In fact, Petitioner’s motivation-to-combine arguments for Grounds 3 and 6 are nearly identical to Petitioner’s arguments for Grounds 1 and 4 (except that they swap out the word “Kobayashi” with “Feyerabend”). *See generally* Pet., 59-63, 69-70. This copy-and-paste approach to the Petition only proves that the arguments are wholly generic and divorced from the prior art itself.

As explained above, the absence of evidence-backed analysis provides an additional reason for rejecting Petitioner’s motivation-to-combine arguments. *See Virtek Vision*, 97 F.4th 882, 886-889; *supra* Section V.A.2.c.

c. Petitioner Fails to Show that a POSA Would Have Had a Reasonable Expectation of Success in Using Feyerabend’s Real-Time Properties in Abarbanel’s System.

Finally, even if Petitioner had shown a motivation to combine Feyerabend and Abarbanel, Petitioner has failed to show that a POSA would have had a reasonable expectation in doing so.

Abarbanel and Feyerabend are directed to different goals, and their teachings are not necessarily compatible. Abarbanel uses RCTL for its IC specifications, *see*

EX1004, 1-2, whereas Feyerabend focuses on defining a model and semantics for expressing real-time symbolic timing diagrams, *see* EX1007, 1-2. Feyerabend describes using RTSTD to describe real-time requirements and TPTL to theoretically perform model checking. EX1007, 3-4, 6. Thus, combining these references—i.e., incorporating Feyerabend’s real-time waveform specification into Abarbanel—would require substituting Feyerabend’s RTSTD in place of Abarbanel’s RCTL.

This combination would not be expected to work because there is no indication that RCTL can handle real-time constraints. As explained above, Abarbanel provides two examples of RCTL properties:

$$\begin{aligned} & \textit{forall } k: ^2 \\ & \{*, \textit{start} \ \& \ \textit{start_tag}=t \ \& \ \textit{to_send}=k, \textit{!end*}, \textit{end} \ \& \ \textit{end_tag}=t\} \rightarrow \{\textit{sent}=k\} \\ & \{*, \textit{read}, \{\textit{!snoop*}, \textit{snoop}, \textit{!snoop*}, \textit{snoop}\} \ \& \ \{\textit{!write*}\} \} \rightarrow \{\textit{cover}\} \end{aligned}$$

EX1004, 3-4. Neither of these properties has real-time constraints. Rather, they are based on the occurrence of certain events (e.g., sending bytes or snoop events).

While they may be temporal in nature—i.e., the property may change over time—this is not “real time” as discussed in Feyerabend. EX1007, 1-2 (discussing past model checking techniques using “temporal logics” and noting that “adding real time to temporal logics” adds complexity). Petitioner does not address this fundamental difference between the references nor explain how Abarbanel’s RCTL

would accommodate Feyerabend's real-time constraints. *See Adidas*, 963 F.3d at 1359 ("Fundamental differences between the references are central to th[e] motivation to combine inquiry.").

And, indeed, Feyerabend itself emphasizes just how complicated its solution actually is. Feyerabend acknowledges there were preexisting "temporal logics" but notes that "adding real time to temporal logics makes the specifications even more difficult to read and makes the need for an intuitive specification style even more pressing." EX1007, 2. Feyerabend likewise comments on related work involving "[c]ombining real time temporal logic with a graphical interface," but concludes that it is "undecidable and thus not suitable for automated proof." *Id.* In the end, Feyerabend notes that there are "major obstacle[s]" involved in "TPTL model checking" and states that "future research" is required to "focus on extending model checking techniques based on RTSTD." *Id.*, 12.

Thus, Feyerabend itself suggests that it was not known how to perform verification based on the very semantic definition it describes. And yet, Petitioner nowhere grapples with these difficulties nor explains how such obstacles would be overcome in its prior-art combination. For example, Petitioner simply states that Abarbanel and Feyerabend have "complementary" teachings, without addressing how a POSA would address the challenges noted in Feyerabend. Pet., 62. This failure is yet another reason to reject Petitioner's obviousness arguments.

VI. CONCLUSION

The Petition fails to show a reasonable likelihood that any of the challenged claims are unpatentable in view of the cited art. Accordingly, the Board should reject all obviousness grounds and deny institution.

Respectfully submitted,

KRAMER ALBERTI LIM & TONKOVICH LLP

/ David L. Alberti /

David L. Alberti
Registration No. 43,465
Counsel for Patent Owner

Date: April 19, 2024

577 Airport Boulevard, Suite 250
Burlingame, California 94010
(650) 825-4300 x104

CERTIFICATE OF WORD COUNT (37 C.F.R. § 42.24(d))

1. This Patent Owner Preliminary Response complies with the type-volume limitation of 14,000 words, comprising 8,943 words, excluding the parts exempted by 37 C.F.R. § 42.24(a)(1).
2. This Patent Owner Preliminary Response complies with the general format requirements of 37 C.F.R. § 42.6(a) and has been prepared using Microsoft® Word 2016 in 14-point Times New Roman font.

CERTIFICATE OF SERVICE (37 C.F.R. § 42.6(e))

I certify that the above-captioned **PATENT OWNER PRELIMINARY RESPONSE UNDER 37 C.F.R. § 42.107(a)** and associated Exhibit 2001 were served in their entirety on April 19, 2024, upon the following parties via electronic mail:

Harper Batts (Lead Counsel)
Chris Ponder (Back-up Counsel)
Jeffrey Liang (Back-up Counsel)
James Young Hurt (Back-up Counsel)
SHEPPARD, MULLIN, RICHTER & HAMPTON LLP
HBatts@sheppardmullin.com
CPonder@sheppardmullin.com
JLiang@sheppardmullin.com
JHurt@sheppardmullin.com

Respectfully submitted,

KRAMER ALBERTI LIM & TONKOVICH LLP

/ David L. Alberti /

David L. Alberti
Registration No. 43,465
Counsel for Patent Owner

Date: April 19, 2024

577 Airport Boulevard, Suite 250
Burlingame, California 94010
(650) 825-4300 x104