

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

BMW OF NORTH AMERICA, LLC, ROBERT BOSCH LLC, and
VALEO SCHALTER UND SENSOREN GMBH,
Petitioner,
v.
FORAS TECHNOLOGIES LTD.,
Patent Owner.

IPR2023-01373¹
Patent 7,502,958 B2

Before CARL W. WHITEHEAD JR., THOMAS L. GIANNETTI and
KARA L. SZPONDOWSKI, *Administrative Patent Judges*.

WHITEHEAD JR., *Administrative Patent Judge*.

JUDGMENT
Final Written Decision
Determining All Challenged Claims Unpatentable
35 U.S.C. § 318(a)

¹ Valeo SE and Valeo Schalter und Sensoren GMBH filed a motion for joinder and a petition in IPR2024-00823 and have been joined as petitioners in this proceeding. See Paper 33, 45–46; IPR2024-00823, Paper 9.

I. INTRODUCTION

A. Background

BMW of North America, LLC and Robert Bosch LLC (“Petitioner”) filed a Petition to institute an *inter partes* review of claims 1–8 and 13–22 (the “challenged claims”) of U.S. Patent 7,502,958 B2, issued on March 10, 2009 (Ex. 1001, “’958 patent”). *See* Paper 1 (“Pet.”). Foras Technologies Ltd. (“Patent Owner”) filed a Preliminary Response. *See* Paper 7 (“Prelim. Resp.”).

We determined that Petitioner established a reasonable likelihood that it would prevail with respect to at least one claim challenged in the Petition. Paper 11 (“Institution Dec.”). We therefore instituted *inter partes* review as to all of the challenged claims of the ’958 patent and all of the asserted grounds of unpatentability. *See SAS Inst. Inc. v Iancu*, 138 S. Ct. 1348, 1356 (2018); 37 C.F.R. § 42.108(a) (“When instituting inter partes review, the Board will authorize the review to proceed on all of the challenged claims and on all grounds of unpatentability asserted for each claim.”).

Following institution of the trial, Patent Owner filed a Response to the Petition (Paper 18, “PO Resp.”), Petitioner filed a Reply to Patent Owner’s Response (Paper 22, “Pet. Reply”), and Patent Owner filed a Sur-reply (Paper 25, “PO Sur-reply”).

An oral hearing was held on January 7, 2025. A transcript of the hearing is part of the record. Paper 32 (“Hearing Tr.”).

We have jurisdiction under 35 U.S.C. § 6. This is a Final Written Decision issued pursuant to 35 U.S.C. § 318(a). For the reasons we discuss below, we determine that Petitioner has proven by a preponderance

of the evidence that all challenged claims of the '958 patent are unpatentable.

B. Related Proceedings

Petitioner and Patent Owner identify the following cases as related to the '958 patent:

Foras Technologies Ltd. v. Toyota Motor North America, Inc., No. 2-23-cv-00321 (E.D. Tex.).

Foras Technologies Limited v. Volkswagen AG et al., No. 2-23-cv-00314 (E.D. Tex.).

Foras Technologies Limited v. Nissan Motor Company, Ltd. et al., No. 1-23-cv-00640 (W.D. Tex.).

Foras Technologies Ltd. v. Bayerische Motoren Werke AG et al., No. 6-23-cv-00386 (W.D. Tex.).

Foras Technologies Ltd. v. Kia Corporation and Kia America, Inc., No. 2:23-cv-00219-JRG (E.D. Tex.).

Foras Technologies Limited v. Toyota Motor North America, Inc. et al., No. 2-23-cv-00150 (E.D. Tex.).

Ex Parte Reexamination, Control No. 90/019,245 (USPTO filed Sept. 1, 2023).

Pet. 1, 78; Paper 5, 3 (Mandatory Notices of Patent Owner).

C. Real Parties-in-Interest

Petitioner identifies BMW of North America LLC, Robert Bosch LLC, Bayerische Motoren Werke AG, and Robert Bosch GMBH as the real parties-in-interest. Pet. 1, 77. Patent Owner identifies itself, Foras Technologies Ltd., as the real party-in-interest. Paper 5, 2 (Mandatory Notices of Patent Owner). Valeo SE and Valeo Schalter und Sensoren

GmbH identify themselves as the real party-in-interest. IPR2024-00823, Paper 1 at 1, 72.

D. The '958 Patent (Ex. 1001)

The '958 patent is titled “System and Method for Providing Firmware Recoverable Lockstep Protection,” and is generally directed to “a method compris[ing] detecting loss of lockstep for a pair of processors.” Ex. 1001, code (54), 3:51–52. The '958 patent’s method “further comprises triggering, by firmware, an operating system to idle the processors, and recovering, by the firmware, lockstep between the pair of processors.” *Id.* at 3:53–55. Subsequently, when the “lockstep is recovered between the pair of processors, the method further comprises triggering, by the firmware, the operating system to recognize the processors as being available for receiving instructions.” *Id.* at 3:55–59.

Figure 1 of the '958 patent, reproduced below, “shows an example embodiment of a system that uses firmware for controlling recovery from detected loss of lockstep (LOL).” Ex. 1001, 3:63–65.

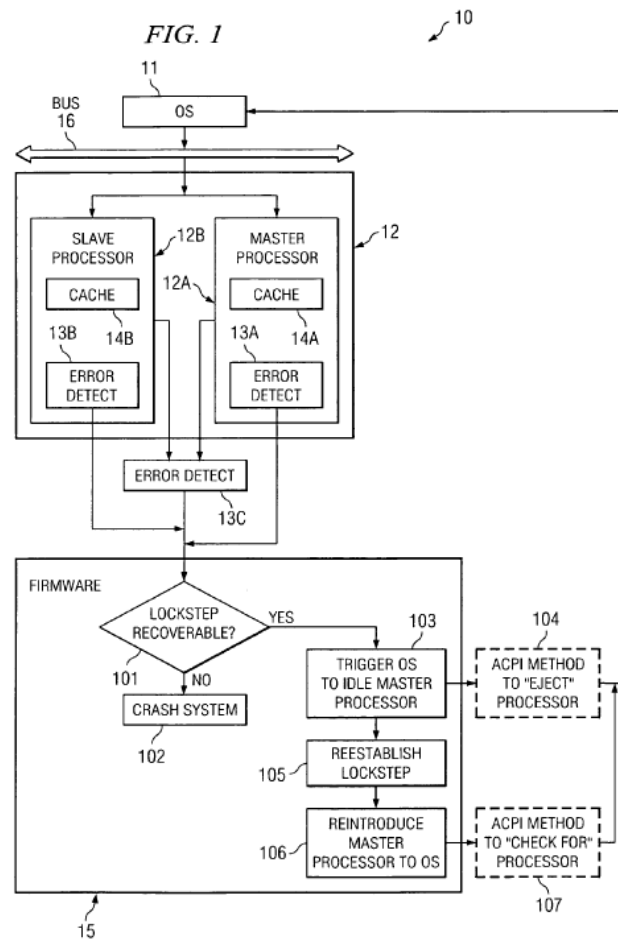


Figure 1, above, depicts a “System 10 [that] includes OS [operating system] 11, as well as master processor 12A and slave processor 12B (collectively referred to as a lockstep processor pair 12).” Ex. 1001, 4:16–18. The “[m]aster processor 12A includes cache 14A, and slave processor 12B includes cache 14B. OS 11 and lockstep processor pair 12 are communicatively coupled to bus 16.” *Id.* at 4:22–25. The system 10 includes master processor 12A with error detect logic 13A, and slave processor 12B with error detect logic 13B. *Id.* at 4:37–39. Additionally, error detect logic 13C is included as shown in Figure 1. Error detect logic 13C “include[s] an XOR (exclusive OR) gate, for detecting a lockstep

mismatch between master processor 12A and slave processor 12B.” *Id.* at 4:47–49.

Also, in the embodiment shown in Figure 1, firmware 15 is included in system 10. Firmware 15 “is invoked upon an error being detected by any of the error detect logics 13A, 13B, and 13C.” Ex. 1001, 5:2–3. In the embodiment shown in Figure 1, “upon firmware 15 being invoked responsive to detection of LOL (by any of error detect logics 13A, 13B, and 13C), firmware 15 determines, in operational block 101, whether the detected LOL is a recoverable LOL.” *Id.* at 5:49–53. Further, “firmware 15 determines in block 101 whether the detected LOL is of a type from which the firmware can recover lockstep for the lockstep processor pair 12 without crashing the system.” *Id.* at 5:53–56. If the lockstep is not recoverable from the detected LOL, firmware 15 crashes the system in block 102. *Id.* at 5:60–62. “[F]irmware 15 is implemented in a manner that allows for recovery from certain detected errors without requiring that OS 11 be implemented with specific knowledge for handling such recovery[, however], if the lockstep is determined to be recoverable, firmware 15 cooperates with OS 11 via standard OS methods to recover the lockstep.” *Id.* at 5:63–6:1.

E. Illustrative Claim

The Petitioner challenges claims 1–8 and 13–22, of which claims 1, 13 and 19 are independent. Pet. 19. Independent claim 1 is directed to method for detecting the loss of lockstep for a lockstep pair of processors by using firmware. Ex. 1001, 15:7–19. Independent claim 13 is directed to a system where upon the loss of lockstep between a pair of processors, the firmware uses an Advanced Configuration and Power Interface (ACPI) to idle a processor. *Id.* at 16:1–15.

Independent claim 19, reproduced below with Petitioner's identifiers in brackets, is illustrative:

[19 preamble] A system comprising a pair of lockstep processors and computer-executable firmware code stored to a computer-readable medium, the computer-executable firmware code comprising:

[19a] firmware code, responsive to detection of loss of lockstep between the pair of lockstep processors, for determining if the lockstep is recoverable for the pair of lockstep processors,

[19b] wherein said firmware code for determining if the lockstep is recoverable further comprises firmware code for determining if a lockstep mismatch has occurred between the pair of lockstep processors; and

[19c] firmware code, responsive to determining that the lockstep is recoverable, for triggering an operating system to idle the processors, attempting to recover lockstep between the pair of processors, and

[19d] if lockstep is successfully recovered between the pair of processors, triggering said operating system to recognize the processors as being available for receiving instructions.

Ex. 1001, 16:29–47; Pet. 36–49.

F. References and other Evidence

The Petition relies on the following references:

Name	Reference	Publication Date	Exhibits
Bigbee ²	US 2003/0126498 A1	July 3, 2003	Ex. 1005
Nguyen ³	US 2004/0123201 A1	June 24, 2004	Ex. 1006

In support of its challenge, Petitioner relies on the Declaration of Michael C. Brogioli, Ph.D. Ex. 1002. Patent Owner relies on the Declaration of Nigel A. Jones. Ex. 2007. Patent Owner has also submitted a deposition transcript for Dr. Brogioli. Ex. 2009.

G. Asserted Grounds of Unpatentability

Petitioner asserts the challenged claims are unpatentable on the following ground (Pet. 1):

Claim(s) Challenged	35 U.S.C. § ⁴	Reference(s)/Basis
1–8, 13–22	103	Bigbee, Nguyen

H. Overview of the Prior Art

1. Bigbee (Ex. 1005)

Bigbee is titled, “Method and Apparatus for Functional Redundancy Check [(FRC)] Mode Recovery.” Ex. 1005, code (54). Bigbee discloses that within some data processing systems:

² Petitioner contends Bigbee is prior art under at least 35 U.S.C. § 102(b). Pet. 1.

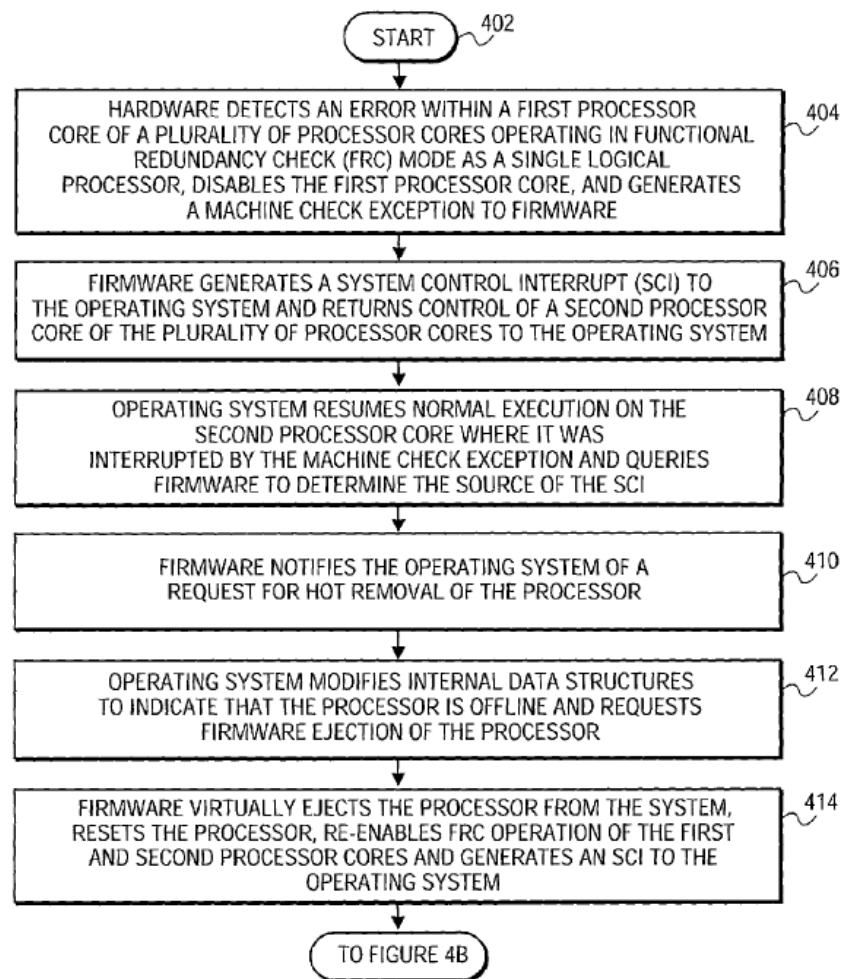
³ Petitioner contends Nguyen is prior art under at least 35 U.S.C. §§ 102(a) and 102(e). Pet. 1.

⁴ The relevant sections of the Leahy-Smith America Invents Act (“AIA”), Pub. L. No. 112–29, took effect on March 16, 2013. The application that issued as the ’958 patent was filed before this date. Ex. 1001, code (22). Pre-AIA statutes apply for the purposes of this Decision.

processor cores are operated in a functional redundancy check (FRC) mode in which identical instructions are provided to each core and concurrently executed on identical data so that any error occurring within a single core will produce an inconsistent result as compared to the remaining cores associated with a given processor.

Id. ¶ 2.

Bigbee's Figure 4a, reproduced below, "illustrates a process flow diagram." Ex. 1005 ¶ 9.



Bigbee discloses:

The illustrated process begins (block 402) and thereafter hardware (e.g. FRC logic) detects an error within a first processor

core of a plurality of processor cores operating in an FRC mode as a single logical processor, disables the first processor core, and generates a system transparent machine check exception (block 404) which in turn causes firmware (e.g. SAL [system abstraction level] program code) to generate a system-visible system control interrupt (SCI) to operating system (OS) software and return control of at least a second, error-free processor core of the plurality of processor cores to the OS (block 406).

Ex. 1005 ¶ 35.

2. *Nguyen (Ex. 1006)*

Nguyen is titled, “On-Die Mechanism For High-Reliability Processor,” and generally is directed to “[a] processor [that] includes first and second execution cores that operate in a redundant (FRC) mode, an FRC check unit to compare results from the first and second execution cores, and an error check unit to detect recoverable errors in the first and second cores.” Ex. 1006, codes (54), (57).

Nguyen’s Figure 1, reproduced below, “is a block diagram of a processor including dual execution cores and FRC-detection and handling logic.” Ex. 1006 ¶ 11.

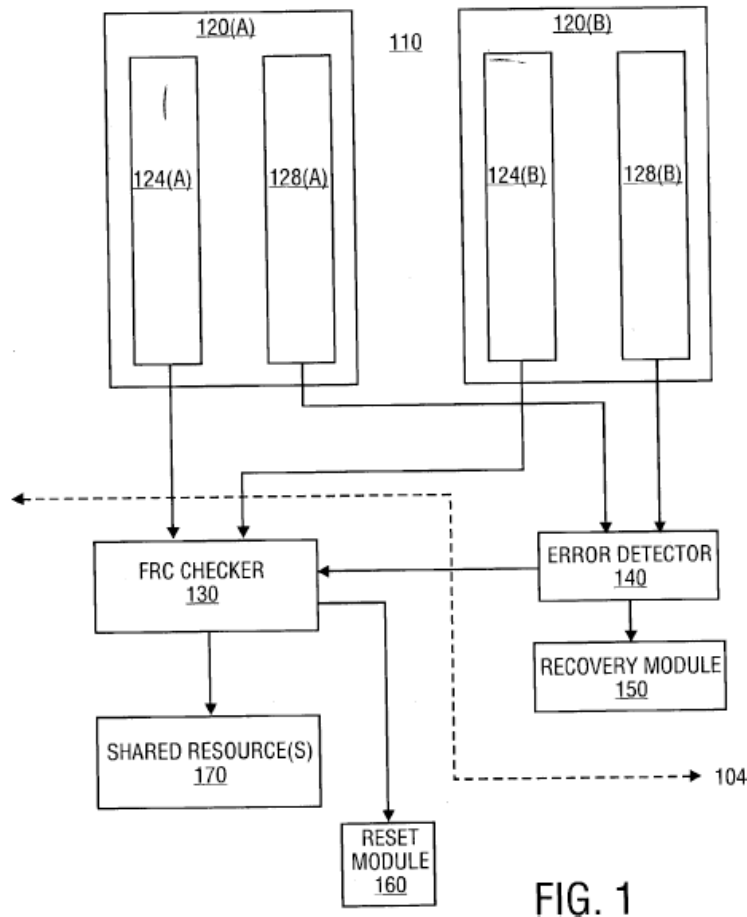


FIG. 1

Nguyen states, “[p]rocessor 110 includes first and second execution cores 120(a), 120(b) (generically, execution core 120), an FRC checker 130, an error detector 140, a recovery module 150, a reset module 160, and shared resources 170.” Ex. 1006 ¶ 20. Nguyen also states, “[i]f an error reaches error detector 140 for either core 120, recovery module 150 is activated to implement a recovery routine.” *Id.* ¶ 22.

Nguyen’s Figure 7, reproduced below, is a block diagram of “an FRC checker that mitigates race conditions between recoverable and non-recoverable error mechanisms.” Ex. 1006 ¶ 18.

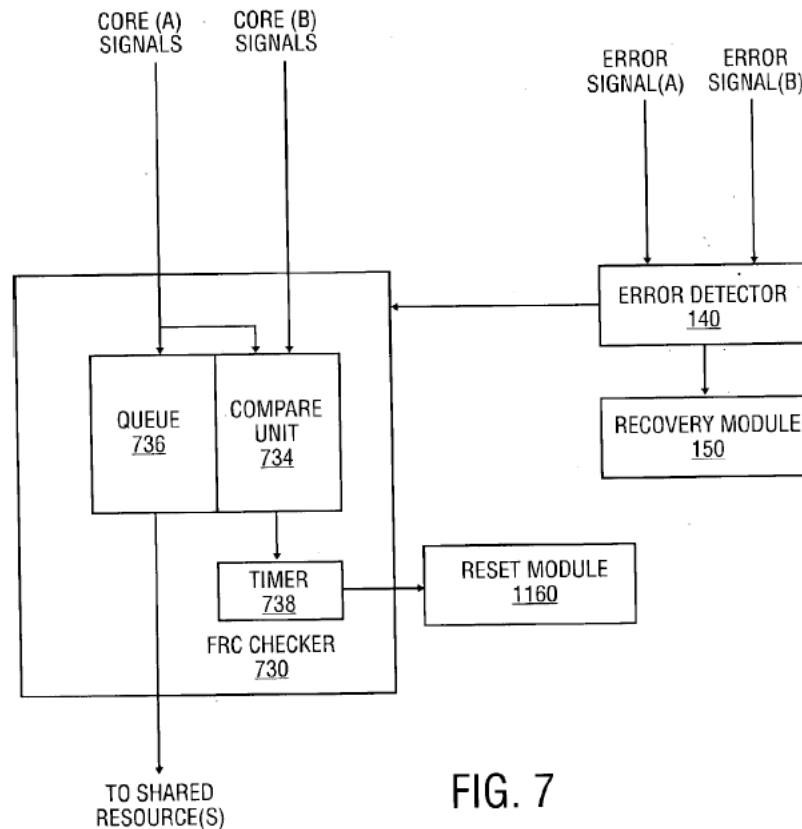


FIG. 7

Nguyen states:

FRC checker 730 includes a compare unit 734, queue 736, and timer unit 738. Queue 736 receives data from execution core (a), and compare unit 734 compares the data from cores A and B, and sets a status flag to indicate if the comparison yields a match. If the data matches, the status flag is set to indicate the match.

Ex. 1006 ¶ 56.

Nguyen further states, “[i]f the data does not match, the status flag is set to indicate the mismatch and timer unit 738 is triggered to begin a countdown interval” and “[i]f [the] error detector 140 receives an error flag

before the timeout interval expires, it disables FRC checker 730 and triggers recovery unit 150 to implement the recovery routine.” Ex. 1006 ¶ 57.

II. ANALYSIS OF THE CHALLENGED CLAIMS

A. Level of Ordinary Skill in the Art

Petitioner asserts a person of ordinary skill in the art “would have at least a bachelor’s degree in Computer Science, Computer Engineering, or equivalent, and at least two years of prior work experience with computer architecture design, including fault tolerant computer architecture design.” Pet. 1 (citing Ex. 1002 ¶ 29). Petitioner further asserts that “[i]f a person did not have a bachelor’s degree, additional relevant and practical experience with fault tolerant computer systems could provide an acceptable substitute.” Ex. 1002 ¶ 29. At the institution stage, Patent Owner did not challenge Petitioner’s definition. *See* Prelim. Resp. 13.

We regarded Petitioner’s description as consistent with the prior art before us. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001) (prior art itself may reflect an appropriate level of skill). Thus, for the purpose of our Institution Decision, we adopted Petitioner’s proposal. Institution Dec. 8–9.

In its Response, Patent Owner does not take exception to our definition. *See* PO Resp. 3. However, Patent Owner asserts that a person of ordinary skill in the art “would have the equivalent of a bachelor’s degree in Computer Science, Computer Engineering (or a related field) and approximately two years of work experience with computer architecture design.” *Id.* We find Patent Owner’s and Petitioner’s definitions of a person

of ordinary skill in the art are not meaningfully distinguishable from one another. Accordingly, we do not see the need to modify our adoption of Petitioner’s proposed level of ordinary skill in the art. However, our findings would not change even under Patent Owner’s proposed level of skill in the art.

Further, Petitioner, in its Reply, argues, “[t]he Board should give the declaration of Patent Owner’s witness Mr. Jones no weight because he does not possess the minimum qualification of the ordinary skill in the art.” Pet. Reply 2. Specifically, Petitioner argues that Mr. Jones does not have a degree in Computer Science or Computer Engineering or two years of prior work experience in computer architecture design. *Id.* at 2–3.

Patent Owner contends Petitioner’s assertion is unsupported because Mr. Jones has the qualifications to qualify as one of ordinary skill in the art according to our definition. *See* PO Sur-reply 10–13; *see also* Ex. 2007, 2–3. For example, Patent Owner argues that Mr. Jones has a degree in Electrical and Mechanical Engineering, decades of industry experience, has worked extensively in “system architecture, hardware design and layout, embedded firmware,” and “device drivers for complex integrated circuits,” has been widely published in the areas of embedded programming techniques, and has written guidelines for the Industry Software Reliability Association (MISRA) for safety critical computer systems. PO Sur-reply 10–11 (citing Ex. 2008, 1, 3).

We are satisfied that Mr. Jones qualifies as a person of ordinary skill in the art. *See, e.g.*, Ex. 2008, 1 (“Summary” including “embedded systems including both software and hardware”, “Representative Design Work” including, e.g., “Design and implement multiple device drivers for complex

Integrated circuits,” “Design of a highly fault tolerant, distributed data acquisition and control, system for a rebreather . . . system architecture, hardware design and layout, embedded firmware, documentation,” “hardware and firmware design,” “[h]ardware and firmware for controlling a unique, battery powered diesel burner...”), 2–3 (list of Publications). We are also satisfied that Mr. Jones’ degree in Electrical and Mechanical Engineering is sufficiently “equivalent” to a degree in Computer Engineering or Computer Science. *See* PO Sur-reply 11.

We decline to give Mr. Jones’ declaration *no weight* for several additional reasons. One, Petitioner has not moved or otherwise sought to exclude Mr. Jones’ declaration, nor has Petitioner filed an objection. *See* Pet. Reply 2–5; *see also* Hearing Tr. 32, 33:15–34:8.

Two, we are afforded broad discretion in assigning weight that is to be accorded expert testimony. *See Yorkey v. Diab*, 601 F.3d 1279, 1284 (Fed. Cir. 2010) (“[T]he Board was well within its discretion to give more credibility to [one expert over another] unless no reasonable trier of fact could have done so.”); *see also* CTPG⁵ 34–36. In sum, Petitioner has not persuaded us that Patent Owner’s expert’s testimony should be given *no weight* because Mr. Jones is not qualified as a person of ordinary skill in the art according to our definition. *See* Pet. Reply 2–5; *see also* PO Sur-reply 10–13.

⁵ Patent Trial and Appeal Board Consolidated [Trial Practice Guide \(CTPG\)](https://www.uspto.gov/TrialPracticeGuideConsolidated) (Nov. 2019) (available at <https://www.uspto.gov/TrialPracticeGuideConsolidated>).

While we do not see the need to exclude or give *no weight* to Mr. Jones' declaration, in the analysis that follows, when applicable, we weigh the credibility of Mr. Jones' testimony and provide our reasons therefore.

B. Claim Construction

Prior to institution, Petitioner contended that the terms “‘detection of loss of lockstep’ and ‘detecting loss of lockstep’ should be construed to include both mismatch errors and errors that may eventually result in mismatch errors.” Pet. 17 (citing Ex. 1001, 4:56–67). Petitioner also contended the terms include “a ‘precursor to lockstep mismatch,’” i.e., “detecting an error that may eventually cause a lockstep mismatch error.” *Id.* (citing Ex. 1001, 4:56–67). Petitioner further contended the “specification provides detecting these two types of errors as examples of detection of LOL.” *Id.* (citing Ex. 1001, 2:26–59, 4:37–67). Petitioner contended the specification states that “[l]ockstep mismatch is one way of detecting an LOL [between the master processor 12A and slave processor 12B].” *Id.* (citing Ex. 1001, 4:56–57) (alteration in original). Petitioner also contended the specification “is consistent with the dependent claims, which expressly define detecting loss of lockstep as either of the two options.” *Id.* at 18 (citing Ex. 1001, 15:7–19, 15:30–49 (claims 1, 4–8)).

Petitioner contended that the terms “‘lockstep is recoverable’ and ‘loss of lockstep is recoverable’ should be construed as ‘lockstep is recoverable from the detected loss of lockstep’ because the specification only describes lockstep recovery from the detected LOL—‘whether the detected LOL is a recoverable LOL.’” Pet. 19 (citing Ex. 1001, 5:56–6:1).

Patent Owner did not challenge Petitioner’s position on these terms. *See* Prelim. Resp. 1–13. We determined there were no claim terms that needed to be construed in our Institution Decision. *See* Institution Dec. 9–10. Post-institution, Patent Owner does not dispute Petitioner’s construction of these terms and “does not believe either term requires construction, because . . . Petitioner fails to demonstrate obviousness regardless of how these terms [(“lockstep is recoverable” and “loss of lockstep is recoverable”)] are construed. PO Resp. 2.

We agree with Patent Owner that no claim construction is necessary for purposes of this Decision. That is, we do not discern any claim terms that are in controversy on the full record. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) (“The Board is required to construe ‘only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy.’” (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

C. Obviousness

A claim is unpatentable under 35 U.S.C. § 103(a) if “the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.” *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations, including (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) objective evidence of nonobviousness,

i.e., secondary considerations. *See Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). Neither party has presented any evidence on the fourth *Graham* factor.

*D. Obviousness of Claims 1–8 and 13–22 Based Upon Bigbee and
Nguyen Combination*

Petitioner contends claims 1–8 and 13–22 would have been obvious over Bigbee in view of Nguyen. *See* Pet. 1.

Petitioner contends, “Bigbee discloses all challenged claim limitations except ‘determining if lockstep is recoverable.’” Pet. 27. However, Petitioner asserts that Nguyen addresses Bigbee’s noted deficiency by disclosing “a method of determining if the lockstep of the processor pair is recoverable or non-recoverable.” *Id.* Petitioner further contends, that “Bigbee’s firmware performing Nguyen’s determining if the lockstep is recoverable or non-recoverable renders claims 1-8 and 13-22 obvious.” *Id.* (citing Ex. 1002 ¶¶ 73–79).

Independent claim 1 recites, “responsive to said detecting loss of lockstep, using said firmware to determine if lockstep is recoverable for the lockstep pair of processors.” Ex. 1001, 15:14–16. Independent claim 13 recites, “firmware operable, responsive to detection of loss of lock step between the master and slave processors, to use an ACPI [advanced configuration and power interface] method to trigger the operating system to idle the master processor, attempt to recover lockstep between the master and slave processors.” *Id.* at 16:9–13. Independent claim 19 recites, “firmware code, responsive to detection of loss of lockstep between the pair

of lockstep processors, for determining if the lockstep is recoverable for the pair of lockstep processors.” *Id.* at 16:34–37.

Our analysis focuses on claim element [19a], which recites “firmware code, responsive to detection of loss of lockstep between the pair of lockstep processors, for determining if the lockstep is recoverable for the pair of lockstep processors.” Pet. 27–28 (citing Ex. 1002 ¶ 78), 38–41; Ex. 1001, 16:34–37. As discussed above, independent claims 1 and 13 contain recitations that are similar to determining if the “lockstep is recoverable” for a pair of lockstep processors as recited in independent claim 19. The dispositive issue discussed, therefore, is whether the Bigbee and Nguyen combination discloses the recoverable lockstep determination as recited in independent claims 1, 13 and 19. For context, we discuss the other limitations of claim 19 as well.

1. Analysis of Independent claim 19

[19 preamble]: “*A system comprising a pair of lockstep processors and computer-executable firmware code stored to a computer-readable medium, the computer-executable firmware code comprising*”

Petitioner contends that Bigbee discloses claim 19’s preamble. *See* Pet. 36 (citing Ex. 1002 ¶¶ 100–101). Petitioner contends, “Bigbee discloses a data processing system 100 including hardware 102, firmware 104, and operating system 106.” *Id.* (citing Ex. 1005 ¶¶ 14–25, Fig. 1). Petitioner states, “[h]ardware 102 includes two or more processor packages, each processor package including two or more processor cores operating lockstep in functional redundancy check (FRC) mode (‘a pair of lockstep processors’) as a single logical processor from the point of view of operating system

106.” *Id.* at 36–37 (citing Ex. 1005 ¶ 19, Fig. 3) (emphasis omitted). Patent Owner does not dispute Petitioner’s contentions. *See* PO Resp. 8. We find for the reasons given that Petitioner demonstrates by a preponderance of the evidence that the preamble of claim 19 is taught or suggested by Bigbee.⁶

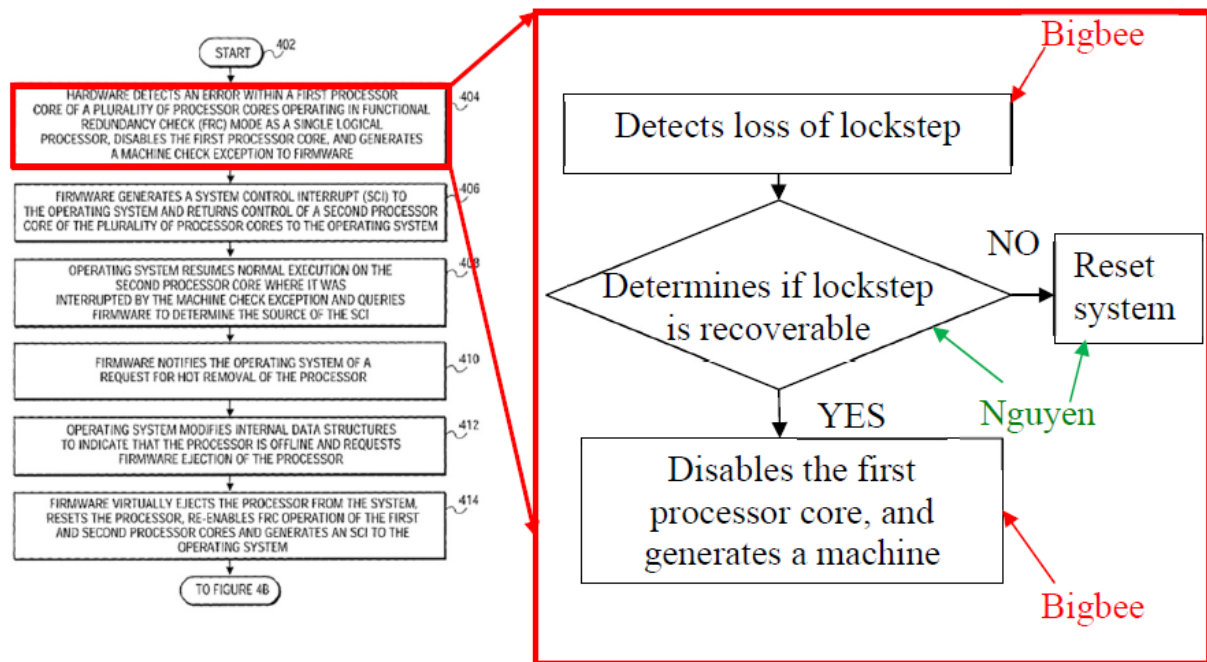
- a. [19a]: *“firmware code, responsive to detection of loss of lockstep between the pair of lockstep processors, for determining if the lockstep is recoverable for the pair of lockstep processors”*

As discussed above, Petitioner contends Bigbee discloses all challenged claim limitations except “determining if lockstep is recoverable.” *See* Pet. 19. However, Petitioner argues that Nguyen addresses Bigbee’s noted deficiency by disclosing a method of determining if the lockstep of the processor pair is recoverable or non-recoverable. *Id.* at 27. Petitioner contends, based upon Bigbee’s and Nguyen’s teachings, that a person of ordinary skill in the art “would have found it obvious to implement Nguyen’s determination of whether the detected error causing the loss of lockstep is recoverable by Bigbee’s firmware FRC logic to create a system that can handle both recoverable and non-recoverable lockstep errors.” *Id.* at 30 (citing Ex. 1002 ¶¶ 80–86). Petitioner further argues that a person of ordinary skill in the art “would have expected success in implementing Nguyen’s determining step in Bigbee’s firmware because Bigbee’s system is compliant with well-known industry standards.” *Id.* at 34 (citing Ex. 1002

⁶ We do not express an opinion on whether the preambles of the challenged claims are limiting.

¶ 95; Ex. 1005 ¶¶ 16–18). Petitioner contends, *inter alia*, that a person of ordinary skill in the art “would have been motivated to incorporate Nguyen’s determination of if lockstep is recoverable in Bigbee’s FRC logic to reduce impact of errors and increase system fault tolerance and availability and would have had a reasonable expectation of success in doing so.” *Id.* at 41 (citing Ex. 1002 ¶ 107); *see also id.* at 30–35.

Petitioner provides an annotated version of Bigbee’s Figure 4A to show Bigbee’s proposed modification in view of Nguyen’s teachings, which is reproduced below:



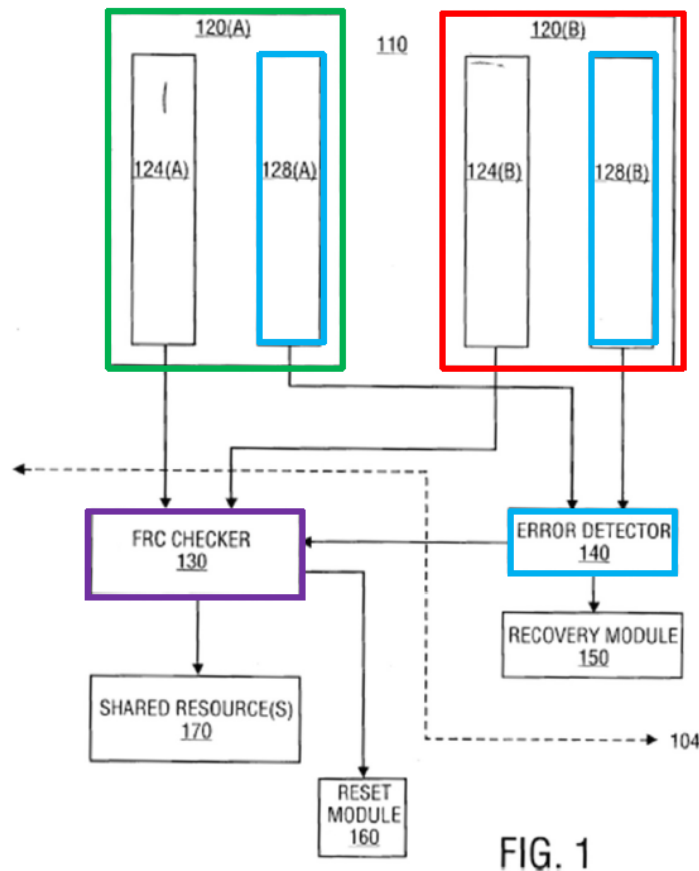
Petitioner annotates Bigbee’s Figure 4A by adding a red border around box 404 of the flow diagram and adding an additional flow diagram showing the proposed modification of Bigbee in view of Nguyen surrounded by a square with red borders. Pet. 29 (citing Ex. 1002 ¶ 79).

Petitioner contends, “Bigbee discloses a data processing system including a plurality of lockstep processor pairs, firmware, and an operating system, along with a method of detecting and recovering loss of lockstep using firmware.” Pet. 19 (emphasis added). According to Petitioner, “Bigbee’s FRC logic detects loss of lockstep between the pair of lockstep processor cores and further discloses implementing the FRC logic by executable firmware.” *Id.* at 38 (citing Ex. 1002 ¶ 103; Ex. 1005 ¶¶ 19, 34); *see also id.* at 39–40 (further arguing that implementing FRC logic by executable firmware code was known in the art). Petitioner supports its contentions with testimony from Dr. Brogioli. *See id.* at 19–72; Ex. 1002 ¶¶ 73–180. Petitioner contends that Nguyen addresses Bigbee’s noted deficiency because “Nguyen discloses a high-reliability processor that includes a lockstep processor pair along with a method of detecting and recovering loss of lockstep of the processor pair that includes determining if the lockstep of the processor pair is recoverable.” Pet. 19–20 (citing Ex. 1002 ¶ 77) (emphasis added).

Petitioner contends, “Nguyen’s FRC checker and error detector determine lockstep is recoverable from the detected error in two instances and non-recoverable from the detected error in one instance.” Pet. 40–41 (citing Ex 1002 ¶¶ 66–69, 106).

Nguyen’s Figure 1 “is a block diagram of a processor including dual execution cores and FRC-detection and handling logic.” Ex. 1006 ¶ 11.

Nguyen’s Figure 1, annotated by Petitioner (Pet. 23) is reproduced below:



Petitioner annotates Nguyen’s Figure 1 by adding a green border around box 120(A) (execution core), adding a red border around box 120(B) (execution core), adding a purple border around box 130 (FRC checker) and adding a blue border around boxes 128(A), (B) (error pipeline) and 140 (error detector). *See* Ex. 1006 ¶¶ 21–22.

Petitioner contends Nguyen discloses that lockstep is not recoverable from an FRC error if the “FRC checker 130 signals an FRC error after detecting a mismatch error before the error detector 140 detects the

underlying soft error.” Pet. 24 (citing Ex. 1006 ¶¶ 7, 22, 40); *see* Ex. 1006 Fig. 1; *see also* Ex. 1002 ¶¶ 67, 106. Petitioner contends that, “this FRC error is non-recoverable mismatch error and a reset module 160 resets the system.” Pet. 24 (citing Ex. 1006 ¶ 22); *see* Ex. 1006, Fig. 1; *see also* Ex. 1002 ¶¶ 67, 106.

Petitioner contends Nguyen discloses that lockstep is recoverable from a soft error detected if the “error detector 140 detects a soft error in either processor core . . . before the FRC checker 130 detects an FRC error indicating mismatch.” Pet. 24 (citing Ex. 1006 ¶¶ 5, 22–24, 46); *see* Ex. 1006 Fig. 1; *see also* Ex. 1002 ¶¶ 68, 106. Petitioner contends, the “error detector 140 disables FRC checker 130 and recovery module 150 implements a recovery routine to restore the cores and the FRC mode (‘lockstep’).” Pet. 24.

Petitioner contends Nguyen discloses that lockstep is also recoverable from a mismatch error that is detected before an FRC error is triggered. Pet. 24 (citing Ex. 1006 ¶¶ 56–58); *see* Ex. 1002 ¶ 69. Nguyen’s Figure 7 shows the FRC checker 730 includes a compare unit 734 and a timer unit 738. Pet. 24; *see* Ex. 1006 ¶ 56; *see also* Ex. 1002 ¶ 69. Petitioner contends, the “[c]ompare unit 734 compares the data from both cores [(A), (B)] and sets a status flag to indicate if the comparison yields a match.” Pet. 24; *see* Ex. 1006, Fig. 7; *see* Ex. 1006 ¶ 56; *see also* Ex. 1002 ¶ 69. Petitioner contends, “[i]f the data does not match, [the] compare unit 734 sets the flags to indicate the mismatch and triggers timer unit 738 to begin a countdown interval.” Pet. 24–25; *see* Ex. 1006, Fig. 7, ¶ 57; *see also* Ex. 1002 ¶ 69. Petitioner contends, “[i]f error detector 140 receives an error flag before the timeout interval expires, it disables FRC checker 730 and triggers recovery

module 150.” Pet. 25; *see* Ex. 1006, Fig. 7, ¶ 57; *see also* Ex. 1002 ¶ 69. Petitioner contends, “But if the timeout interval does expire, the FRC checker [730] signals a non-recoverable FRC error and triggers reset module 160 to reset the system.” Pet. 25 (citing Ex. 1006 ¶¶ 22, 56–58); *see* Ex. 1006, Fig. 7; *see also* Ex. 1002 ¶ 69.

We agree with Petitioner’s and Dr. Brogioli’s analysis as it is consistent with our understanding of the references. Nguyen recognizes that, “[s]ince FRC errors indicate only that the execution cores disagree on a result, FRC errors are detectable but not recoverable” and therefore “the FRC handling routine typically resets the system to the last known point of reliable data.” Ex. 1006 ¶ 7; *see* Ex. 1002 ¶¶ 66–67. Nguyen realizes handling otherwise recoverable FRC errors through a “reset mechanism reduces system availability.” Ex. 1006 ¶ 8; *see* Ex. 1002 ¶ 66. Nguyen “addresses mechanisms for combining recoverable and non-recoverable error handling mechanisms efficiently in FRC-enabled processors.” Ex. 1006 ¶ 9.

Here, based upon the evidence before us, and for the reasons given, we agree with Petitioner’s determination that, “Bigbee discloses all challenged claim limitations except ‘determining if lockstep is recoverable.’” Pet. 27. We also agree with Petitioner’s determination, based upon the evidence before us, that “Nguyen discloses a method of determining if the lockstep of the processor pair is recoverable or non-recoverable.” *Id.* Patent Owner does not challenge Petitioner’s findings in regard to the individual teachings of Bigbee or Nguyen. *See* PO Resp. 6–8.

However, Patent Owner and its expert Mr. Jones disagree with Petitioner's analysis. *See* PO Resp. 8–18; Ex. 2007 ¶¶ 26–33. Patent Owner contends the “critical” flaw in Petitioner's Bigbee/Nguyen combination is “the recovery process begins after Bigbee has detected an ‘error within a first processor.’” PO Resp. 12 (citing Ex. 1005, Fig. 4A; Ex. 2009, 74:14–76:8; Ex. 2007 ¶ 27). Patent Owner argues, “this means that Nguyen will never determine that lockstep is non-recoverable” and therefore “[e]very error processed by Nguyen's ‘determines if lockstep is recoverable’ routine will be handled the same way.” *Id.* (citing Ex. 2007 ¶¶ 29–30); *see* PO Sur-reply 3–4.

Patent Owner also argues, for example, for every error processed, “[Nguyen] will follow the ‘YES’ path shown in Petitioner's annotations to Figure 4A and proceed through the remainder of Bigbee's routing for attempting to recover from recoverable individual processor errors.” PO Resp. 12. That is, Patent Owner argues, “Nguyen is explicit that when an error is detected within an individual processor, that error is treated as recoverable.” PO Sur-reply 4. According to Patent Owner, “Nguyen only determines mismatch errors are non-recoverable when they are detected prior to detection of an individual processor error. But the starting point for the ‘combined system’ is detection by Bigbee of an individual processor error.” *Id.*; *see* Pet. 29 (Ex. 1005, Fig. 4A (annotated)); *see also* Ex. 1006 ¶¶ 22–23. Patent Owner contends, “[a]s a result, the Nguyen-Bigbee ‘combined system’ that Petitioner says would be obvious, is a system without any mechanism for handling recoverable errors differently from non-recoverable errors.” PO Resp. 14.

Petitioner responds, “Patent Owner’s arguments contradict the express disclosure in Nguyen that determines error recoverability and handles recoverable errors and non-recoverable errors differently.” Pet. Reply 7–8 (citing Pet. 23–26, 40–41; Ex. 1006 ¶¶ 7, 8, 22–25, 37, 39–41, 46, 56–58; Ex. 1002 ¶¶ 66–69, 105, 106). We find Petitioner’s contentions persuasive because Nguyen discloses, “an FRC error will be triggered if the corrupted data from one execution core [(either core 120)] and an uncorrupted version of the data from the other execution [core] are allowed to reach FRC checker 130 before recovery module 150 is activated.” Ex. 1006 ¶ 22, *see id.* at Fig. 1; *see also* Ex. 1002 ¶¶ 65–68. Nguyen further discloses that, “[s]ince FRC errors are not-recoverable, reset module 160 resets the system, if FRC checker 130 signals an FRC error before the underlying parity/ECC error is detected.” Ex. 1006 ¶ 22. Accordingly, we do not agree with Patent Owner’s contentions that Nguyen only determines non-recoverable errors occur when they are detected prior to the detection of an individual processor error. *See* PO Resp. 12, 14; PO Sur-reply 4; *see also* Ex. 1002 ¶¶ 65–68 (citing Ex. 1006 ¶ 22). Moreover, Nguyen explicitly describes “a mechanism for handling recoverable and non-recoverable errors in multi-core processor[s]” wherein “[t]he multiple cores may be operated in FRC mode, in which case, one or more checker units compare signals from the cores to detect non-recoverable errors.” Ex. 1006 ¶ 58; *see* Pet. 23–25.

Therefore, we agree with Petitioner and find that for the reasons given, Nguyen discloses “determining if the lockstep is recoverable for the pair of lockstep processors,” as recited in claim 19. Accordingly, we disagree with Patent Owner that the combined Bigbee-Nguyen system “is a

system without any mechanism for handling recoverable errors differently from non-recoverable errors.” PO Resp. 14.

Patent Owner further argues modifying Bigbee’s firmware FRC logic is flawed because Petitioner has failed to show that implementing portions of Nguyen within firmware would have been obvious. PO Resp. 14–15 (citing Pet. 30; Ex. 1002 ¶ 86). However, we find that Bigbee explicitly teaches that “FRC logic . . . may be implemented as executable firmware,” which supports Petitioner’s reasoning. Ex. 1005 ¶ 34; *see* Pet. 33–34; *Brown & Williamson Tobacco Corp. v. Philip Morris Inc.*, 229 F.3d 1120, 1125 (Fed. Cir. 2000) (stating that evidence of a motivation to combine “may flow from the prior art references themselves”).

Patent Owner relies upon Mr. Jones’ testimony that modifying Bigbee’s firmware by implementing “Nguyen’s FRC checker compare unit” in firmware code would be impractical for “several reasons.” PO Resp. 15 (citing Ex. 2007 ¶ 32). Mr. Jones’ reasons are that “using firmware would require a separate processor running approximately twenty times faster than the processor cores”; proving “firmware configured to perform bit-by-bit error checking is operating correctly is much more difficult than proving the same for a hardware implementation”; and “firmware is more vulnerable than hardware to malicious attacks.” Ex. 2007 ¶ 32.

Petitioner argues, “[n]one of Mr. Jones’s statements is supported by objective evidence.” Pet. Reply 11. Petitioner further argues, “Mr. Jones’s statements, even if true, do not rebut Dr. Brogioli’s opinion that firmware, hardware, or a combination of firmware and hardware would have been a suitable option.” *Id.* at 12. We do not find Patent Owner’s arguments persuasive.

We find Mr. Jones’ testimony amounts to conjecture because it is unsupported by evidence, for the reasons explained by Petitioner. *See* Pet. Reply 10–11; Ex. 2007 ¶ 32; *see* 37 C.F.R. § 42.65(a) (“Expert testimony that does not disclose the underlying facts or data on which the opinion is based is entitled to little or no weight”). However, we find that Dr. Brogioli provides sufficient persuasive evidence as to how firmware FRC logic would be used and modified in the Bigbee/Nguyen combination. *See* Ex. 1002 ¶¶ 57–58, 96–98. As discussed above, Bigbee uses firmware FRC logic to check for an error within a processor core. *See* Ex. 1005 ¶¶ 35–37, Fig. 4A. Bigbee’s firmware FRC logic does not determine if lockstep is recoverable. *Id.* Similar to Bigbee, Nguyen also uses firmware FRC logic to check for an error within a processor core. *See* Ex. 1006 ¶¶ 9, 18, Fig. 7. However, the difference is that Nguyen’s firmware FRC logic determines if lockstep is recoverable. *Id.*

We find for the reasons given by Petitioner and Dr. Brogioli including those summarized *supra*, that it would have been obvious to one of ordinary skill in the art to modify Bigbee by implementing Nguyen’s determination in firmware. Pet. 33–35 (citing, e.g., Ex. 1002 ¶ 94 (stating that “[i]mplementing logical steps in firmware was known in the art”); ¶ 95 (stating that Bigbee’s system “is compliant with well-known industry standards”); ¶ 98 (stating that “firmware is flexible” and “can be modified after production, updated, and supported relatively quickly and inexpensively.”)). In contrast, Patent Owner’s arguments fail to convince us that a person of ordinary skill in the art, having knowledge of Nguyen’s teaching of determining lockstep recovery, would not have able to modify Bigbee’s firmware FRC logic to determine if lockstep is recoverable. *See*

KSR, 550 U.S. at 421 (“A person of ordinary skill is also a person of ordinary creativity, not an automaton.”); *see also id.* at 418.

Accordingly, Petitioner has presented reasoning, with rational underpinnings, that a person of ordinary skill in the art would have had reasons to modify Bigbee’s firmware FRC logic based upon Nguyen’s teachings and would have had a reasonable expectation of success in doing so. Accordingly, we find for the reasons given that Petitioner demonstrates by a preponderance of the evidence that this element is taught or suggested by the Bigbee/Nguyen combination.

- b. [19b]: “*wherein said firmware code for determining if the lockstep is recoverable further comprises firmware code for determining if a lockstep mismatch has occurred between the pair of lockstep processors*”

Petitioner contends, “Bigbee in view of Nguyen renders obvious [19b].” Pet. 41 (citing Ex. 1002 ¶¶ 108–111). Petitioner contends, “Bigbee in view of Nguyen discloses ‘the firmware code for determining if the lockstep is recoverable’ ([19a]) and further discloses ‘firmware code for determining if a lockstep mismatch has occurred between the pair of lockstep processors.’” *Id.* at 41–42 (citing Ex. 1002 ¶ 109). Petitioner contends, “Nguyen discloses ‘determining if a lockstep mismatch has occurred’ because Nguyen determines if the lockstep is recoverable after detecting a mismatch error.” *Id.* (citing Ex. 1002 ¶ 110) (emphasis omitted).

Patent Owner does not separately address [19b] limitations. *See* PO Resp. 8–17; PO Sur-reply 3–9. We find for the same reasons we discussed above, as well as Petitioner’s reasons, that Petitioner demonstrates by a

preponderance of the evidence that this element is taught or suggested by the Bigbee/Nguyen combination.

- c. [19c]: “*firmware code, responsive to determining that the lockstep is recoverable, for triggering an operating system to idle the processors, attempting to recover lockstep between the pair of processors, and*”

Petitioner contends, “Bigbee in view of Nguyen renders obvious [19c].” Pet. 42 (citing Ex. 1002 ¶¶ 112–116). Petitioner maintains that, incorporating “Nguyen’s determining if the lockstep is recoverable into Bigbee to take an appropriate action responsive to the severity of the error detected in the lockstep processor pair” would have been obvious to a person of ordinary skill. *Id.* at 43. Additionally, Petitioner states, “Bigbee in view of Nguyen further discloses that, responsive to determining if the lockstep is recoverable from the detected error, the firmware triggers the operating system to idle the processors and recover the lockstep between the pair of processors.” *Id.* (citing Ex. 1002 ¶ 113).

Patent Owner does not separately address [19c] limitations. *See* PO Resp. 8–17; PO Sur-reply 3–9. We find for the same reasons we discussed above, as well as Petitioner’s reasons, that Petitioner demonstrates by a preponderance of the evidence that this element is taught or suggested by the Bigbee/Nguyen combination.

- d. [19d]: “*if lockstep is successfully recovered between the pair of processors, [firmware] triggering said operating system to*

recognize the processors as being available for receiving instructions.”

Petitioner contends, “Bigbee discloses [19d].” Pet. 46 (citing Ex. 1002 ¶¶ 117–119). Petitioner states, “Bigbee discloses that its firmware generates a SCI to the operating system after recovering lockstep between the pair of processors (‘if lockstep is successfully recovered between the pair of processors’).” *Id.* (citing Ex. 1005 ¶ 37, Fig. 4A (block 414)). Petitioner also states, “[t]he operating system queries firmware (e.g., ACPI BIOS) to determine the source of the SCI, and the firmware responds to the query by notifying the operating system of a request for processor hot insertion (‘triggering said operating system’).” *Id.* (citing Ex. 1005 ¶ 37, Fig. 4B (blocks 416, 418)).

Patent Owner focuses on the “combined system” of the Bigbee and Nguyen combination and does not dispute Petitioner’s contentions in regard to Bigbee disclosing [19d]. PO Resp. 8–17; PO Sur-reply 3–9. We find for the same reasons we discussed above, as well as Petitioner’s reasons, that Petitioner demonstrates by a preponderance of the evidence that this element is taught or suggested by Bigbee.

2. Summary–Independent claim 19

We find for the reasons given that Petitioner has proven by a preponderance of the evidence that each element of claim 19 is taught or suggested by the combination of Bigbee and Nguyen, and we further determine that claim 19 would have been obvious in light of the Bigbee/Nguyen combination.

3. Dependent Claim 8

Claim 8 depends from claim 1 and recites “if determined that a lockstep mismatch has occurred between the lockstep pair of processors, then determining that the loss of lockstep is not recoverable.” Ex. 1001, 15:46–49. Petitioner argues that the combination of Bigbee and Nguyen teaches the limitations in claim 8. Pet. 66–67. Petitioner relies on its arguments for limitation 19[a], i.e., that “Nguyen discloses determining that lockstep is not recoverable from a non-recoverable FRC error indicating mismatch,” and argues that in the combination, “Bigbee’s FRC logic would have determined that the FRC mismatch error for which it was not known which processor core was responsible as not recoverable.” *Id.*

Patent Owner argues, “Petitioner has not shown that the Nguyen-Bigbee ‘combined system’ ever makes a determination that ‘loss of lockstep is not recoverable.’” PO Resp. 18. Patent Owner contends, “the Nguyen-Bigbee ‘combined system’ handles each error the same way and, because it begins with Bigbee’s detection of a recoverable error in an individual processor, it never determines that any loss of lockstep is not recoverable.” *Id.* at 18 (citing Ex. 1006 ¶¶ 5, 7, 23, 24, 30 and 46; Pet. 40–41; Ex. 1002 ¶ 106).

Patent Owner’s argument is unavailing for the reasons discussed above with respect to claim 19. We find for the reasons given that Petitioner demonstrates by a preponderance of the evidence that the combination of Bigbee and Nguyen teaches claim 8.

4. Independent Claims 1 and 13

Petitioner's analysis of claims 1 and 13 incorporates its analysis of claim 19. *See* Pet. 49–60. Patent Owner does not separately address these claims. *See generally* PO Resp. As discussed above, the claims recite determining “if lockstep is recoverable” or “if recovery of lockstep is successful” in the same manner as discussed in regard to claim 19. Ex. 1001, 15:15, 16:13–14. For the reasons given for claim 19, we determine that Petitioner has proven by a preponderance of the evidence that independent claims 1 and 13 would have been obvious in light of the Bigbee/Nguyen combination.

5. Dependent Claims 2–7, 14–18 and 20–22

Claims 2–7, 14–18 and 20–22 depend from independent claims 1, 13 or 19. Petitioner provides an analysis for each of these dependent claims in relation to the Bigbee/Nguyen combination. *See* Pet. 61–72. Patent Owner does not separately address these claims. *See generally* PO Resp. For the reasons set forth by Petitioner, we determine that Petitioner has proven by a preponderance of the evidence that dependent claims 2–7, 14–18 and 20–22 would have been obvious in light of the Bigbee/Nguyen combination.

III. CONCLUSION

For the foregoing reasons, we determine that Petitioner has proven by a preponderance of the evidence that claims 1–8 and 13–22 of the ‘958 patent are unpatentable.

In summary:

Claims	35 U.S.C. §	Reference(s)/Basis	Claim(s) Shown Unpatentable	Claim(s) Not Shown Unpatentable
1–8, 13–22	103	Bigbee, Nguyen	1–8, 13–22	

IV. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that Petitioner has demonstrated by a preponderance of the evidence that claims 1–8 and 13–22 of the '958 patent are unpatentable; and

FURTHER ORDERED that because this is a Final Written Decision, parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.1.⁷

⁷ Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this decision, we draw Patent Owner's attention to the April 2019 *Notice Regarding Options for Amendments by Patent Owner Through Reissue or Reexamination During a Pending AIA Trial Proceeding*. See 84 Fed. Reg. 16,654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. See 37 C.F.R. § 42.8(a)(3), (b)(2).

IPR2023-01373
Patent 7,502,958 B2

For PETITIONER:

Lionel Lavenue
Kara Specht
Cory Bell
Matthew Berntsen
Misook Kim
Xirui Zhang
FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, LLP
lionel.lavenue@finnegan.com
kara.specht@finnegan.com
cory.bell@finnegan.com
matthew.berntsen@finnegan.com
misook.kim@finnegan.com
xirui.zhang@finnegan.com

PATENT OWNER:

Brett Cooper
BC LAW GROUP, P.C.
bcooper@bc-lawgroup.com

Robert Auchter
AUCHTER PLLC
robert@auchterlaw.com