

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

VALEO SE and VALEO SCHALTER UND SENSOREN GMBH,
Petitioner,
v.
FORAS TECHNOLOGIES LTD.,
Patent Owner.

IPR2024-00823
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*.

DECISION
Granting Institution of *Inter Partes* Review
35 U.S.C. § 314

Granting Motion for Joinder
35 U.S.C. § 315(c); 37 C.F.R. § 42.122

I. INTRODUCTION

Valeo SE and Valeo Schalter und Sensoren GmbH (collectively, “Petitioner”) filed a Petition (Paper 1, “Pet.”) to institute an *inter partes* review of claims 1–8 and 13–22 of U.S. Patent 7,502,958 B2, issued on March 10, 2009 (Ex. 1001, “the ’958 patent”). Foras Technologies Ltd. (“Patent Owner”) filed a Preliminary Response. Paper 8 (“Prelim. Resp.”). Petitioner filed a Motion for Joinder with the Petition. Paper 4 (“Joinder Motion”). This motion seeks joinder with IPR2023-01373. The petitioners in that proceeding are BMW of North America, LLC and Robert Bosch LLC (collectively, “BMW”). We instituted trial in IPR2023-01373 on March 20, 2024. *See* IPR2023-01373, Paper 11.

Institution of an *inter partes* review is authorized when “the information presented in the petition . . . and any response . . . shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” 35 U.S.C. § 314(a). Based on the current record, and for the reasons explained below, we determine that Petitioner has established a reasonable likelihood that it would prevail with respect to at least one challenged claim, and we institute an *inter partes* review. Further, we grant Petitioner’s Joinder Motion and join Petitioner as a party to IPR2023-01373. *See* Section V, *infra*.

II. BACKGROUND

A. *Real Parties-in-Interest*

Petitioner identifies Valeo SE and Valeo Schalter und Sensoren GmbH as the real parties-in-interest. Pet. 72. Patent Owner identifies itself

as the real party-in-interest. Paper 6, 2 (Mandatory Notices of Patent Owner).

B. Related Matters

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

BMW of North America, LLC and Robert Bosch LLC v. Foras Technologies Ltd., IPR2023-01373 (PTAB).

Bayerische Motoren Werke AG et al. v. Foras Technologies Ltd., No. 1:24-cv-00363 (E.D.Va.).

HL Klemove Corp. v. Foras Technologies Ltd. (intradistrict transferred to 2:23-cv-00647), No. 1:23-cv-01689 (E.D.Va.).

HL Klemove Corp. v. Foras Technologies Ltd., No. 2:23-cv-00647 (E.D.Va.).

Foras Technologies Ltd. v. Toyota Motor North America, Inc. (now dismissed), 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. 73–74; Paper 6, 2–3.

C. U.S. Patent 7,502,958 (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).” *Id.* at 3:63–65.

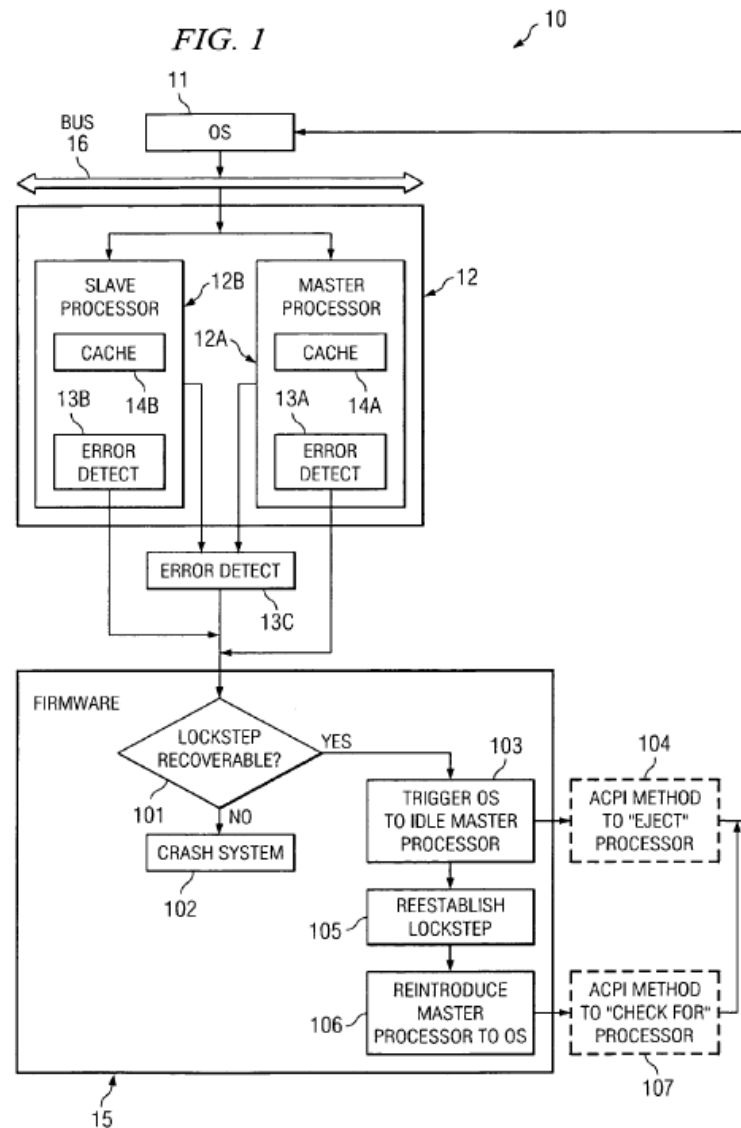


Figure 1, above, depicts a “System 10 [that] includes OS [operating system] 11, as well as master processor 12A and slave processor 12B (collectively referred to as a lockstep processor pair 12).” *Id.* at 4:16–18. The “[m]aster processor 12A includes cache 14A, and slave processor 12B includes cache 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. The 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.” *Id.* at 5:2–3. In the same 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 shown in the Figure 1 embodiment crashes the system in block 102, which is also shown in Figure 1. *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.

D. Illustrative Claim

Independent claim 19 is reproduced below, with Petitioner’s identifiers in brackets. *See* Pet. 33–40.

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:

[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.

E. Prior Art and Asserted Challenge to Patentability

Petitioner asserts that claims 1–8 and 13–22 are unpatentable based upon the following challenge (Pet. 1):

Claim(s) Challenged	35 U.S.C. §¹	Reference(s)/Basis
1–8, 13–22	103(a)	Bigbee, ² Nguyen ³

¹ 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. *See* Ex. 1001, code (22). Pre-AIA statutes apply for the purposes of this Decision.

² Bigbee et al., US 2003/0126498 A1, published July 3, 2003 (Ex. 1005) (“Bigbee”). Petitioner contends that Bigbee is prior art under at least 35 U.S.C. § 102(b). Pet. 1.

³ Nguyen et al., US 2004/0123201 A1, published June 24, 2004 (Ex. 1006)
Footnote continued on the next page.

In support of its challenge, Petitioner relies on the Declaration of Michael C. Brogioli, Ph.D. *See* Ex. 1002.

III. ANALYSIS

A. Legal Standards

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).⁴

“In an [*inter partes* review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3)); *see also Intelligent Biosystems, Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1369 (Fed. Cir. 2016). At this preliminary stage, we determine whether the information presented in the Petition shows there is a reasonable likelihood that Petitioner would

(“Nguyen”). Petitioner contends that Nguyen is prior art under at least 35 U.S.C. §§ 102(a), 102(e). Pet. 1.

⁴ Neither party has presented evidence on this fourth *Graham* factor.

prevail in establishing that at least one of the challenged claims would have been obvious.

B. 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 “[a]dditional education could substitute for professional experience and vice versa.” *Id.* (citing Ex. 1002 ¶ 29).

Patent Owner asserts:

A person of ordinary skill in the art of the subject matter of the ’958 patent 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. Additional education in the field could be substituted for industry experience, and vice versa.

See Prelim. Resp. 3.

The parties’ positions appear to be more or less the same. Based on the current record, we adopt Petitioner’s proposed level of ordinary skill in the art for purposes of this Decision, as it is consistent with the level of ordinary skill in the art reflected by the prior art of record. We note that our Decision would be the same under either party’s proposed construction.

C. Claim Construction

A claim “shall be construed using the same claim construction standard that would be used to construe the claim in a civil action under 35

U.S.C. § 282(b).” 37 C.F.R. § 42.100(b) (2024). Petitioner contends 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. 16 (citing Ex. 1001, 4:56–67). In other words, Petitioner contends 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 contends 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 contends 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) (first alteration in original). Petitioner contends the specification “is consistent with the dependent claims, which expressly define detecting loss of lockstep as either of the two options.” *Id.* at 17 (citing Ex. 1001, 15:7–19, 15:30–49 (claims 1, 4–8)).

Petitioner contends 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.’” *Id.* (citing Ex. 1001, 5:56–6:1).

“Patent Owner does not believe either term requires construction, because . . . Petitioner fails to demonstrate obviousness regardless of how these terms are construed.” *See* Prelim. Resp. 2 (referencing Pet. 17, 19).

At this stage of the proceeding, only terms that are in controversy need to be construed, and then only to the extent necessary to determine whether to institute *inter partes* review. *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))). After reviewing the parties’ evidence and contentions, we do not observe a dispute over the meaning of these terms that is necessary to resolve at this stage.

Accordingly, we determine that we do not need to expressly construe any term for purposes of this Decision.

D. Alleged Obviousness over Bigbee and Nguyen

Petitioner contends claims 1–8 and 13–22 would have been obvious over Bigbee in view of Nguyen. Pet. 18. Patent Owner disputes Petitioner’s contentions. Prelim. Resp. 8–18. In particular, Patent Owner contends a person of ordinary skill in the art (POSITA) “would not have been motivated to combine Bigbee and Nguyen.” *Id.* at 8–17 (emphasis omitted). Patent Owner further contends the “Bigbee-Nguyen combination does not render obvious ‘if determined that a lockstep mismatch has occurred between the lockstep pair of processors, then determining that the loss of lockstep is not recoverable,’” as recited in dependent claim 8. *Id.* at 18 (emphasis omitted).

Petitioner’s table “[a]pplying the combination of Bigbee and Nguyen to the six components of the independent claims” is reproduced below:

Core Claim Requirement	Prior Art Reference
1) Detecting loss of lockstep between a pair of lockstep processors	Bigbee
2) With firmware, determining if lockstep is	Bigbee (with firmware) Nguyen (determining if

recoverable	lockstep is recoverable)
3) With firmware, determining if lockstep mismatch has occurred	Bigbee
4) Triggering an operating system to idle the processors	Bigbee
5) Attempting to recover lockstep	Bigbee
6) If recovered, trigger the operating system to recognize the processors to be available	Bigbee

Pet. 25–26 (citing Ex. 1002 ¶ 78).

Having considered the arguments presented in the Petition, the Preliminary Response, and the evidence of record, we determine that Petitioner has demonstrated a reasonable likelihood of success in proving that at least one claim of the ’958 patent is unpatentable as obvious over the combination of Bigbee and Nguyen.

Accordingly, we institute an *inter partes* review of all challenged claims (i.e., claims 1–8 and 13–22) on all grounds set forth in the Petition. Our determinations at this stage of the proceeding are based on the evidentiary record currently before us. This decision to institute trial is not a final decision as to the patentability of any claim for which we have instituted an *inter partes* review. *See TriVascular, Inc. v. Samuels*, 812 F.3d 1056, 1068 (Fed. Cir. 2016) (noting that “there is a significant difference between a Petitioner’s burden to establish a ‘reasonable likelihood of

success’ at institution, and actually proving invalidity by a preponderance of the evidence at trial”). We will base any final decision on the full record developed during trial.

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.” *Id.* at 18. Petitioner also contends, “Bigbee, therefore, discloses all elements of claims 1-8 and 13-22 except for those elements that require determining if the lockstep of a lockstep processor pair is recoverable.” *Id.* (citing Ex. 1002 ¶ 76). Petitioner further 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.” *Id.* (citing Ex. 1002 ¶ 77). Petitioner argues, “Bigbee’s firmware implementing Nguyen’s determining if the lockstep is recoverable renders claims 1-8 and 13-22 obvious.” *Id.*

1. Bigbee (Ex. 1005)

Bigbee is titled, “Method and Apparatus for Functional Redundancy Check Mode Recovery.” Ex. 1005, code (54). Bigbee’s Figure 1, reproduced below, “illustrates an abstraction layer block diagram of a data processing system.” *Id.* ¶ 6.

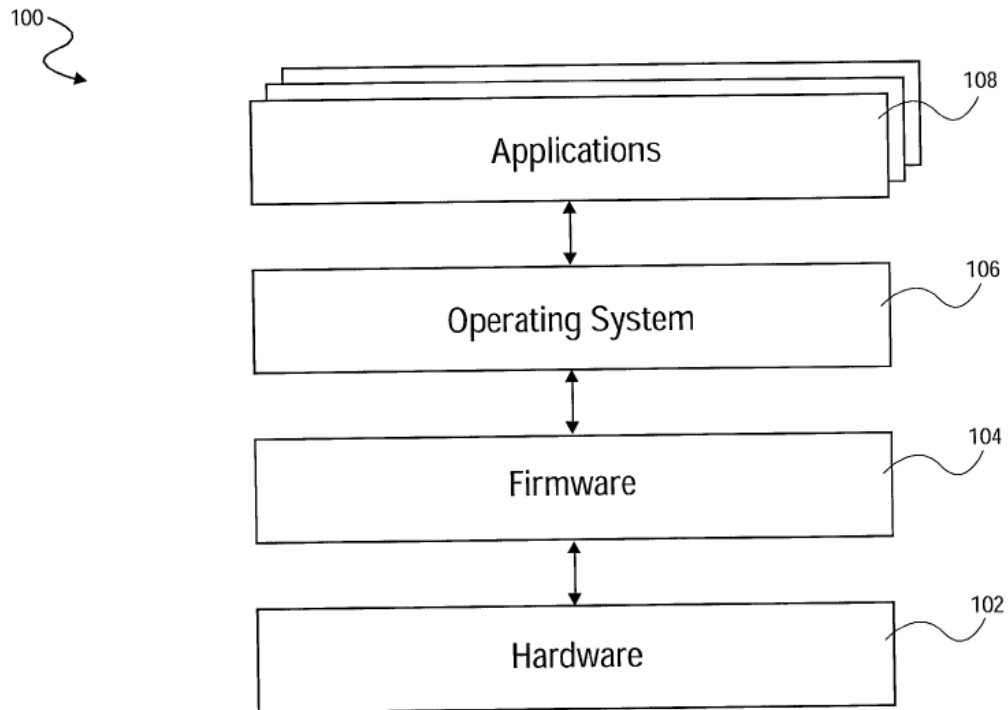


FIG. 1

Describing Figure 1, Bigbee states, “[t]he illustrated data processing system 100 includes hardware 102, system management programs according to the present invention including firmware 104 and operating system 106, and application programs 108.” *Id.* ¶ 14.

Bigbee Figure 2, reproduced below, “illustrates an exemplary data processing system.” *Id.* ¶ 7.

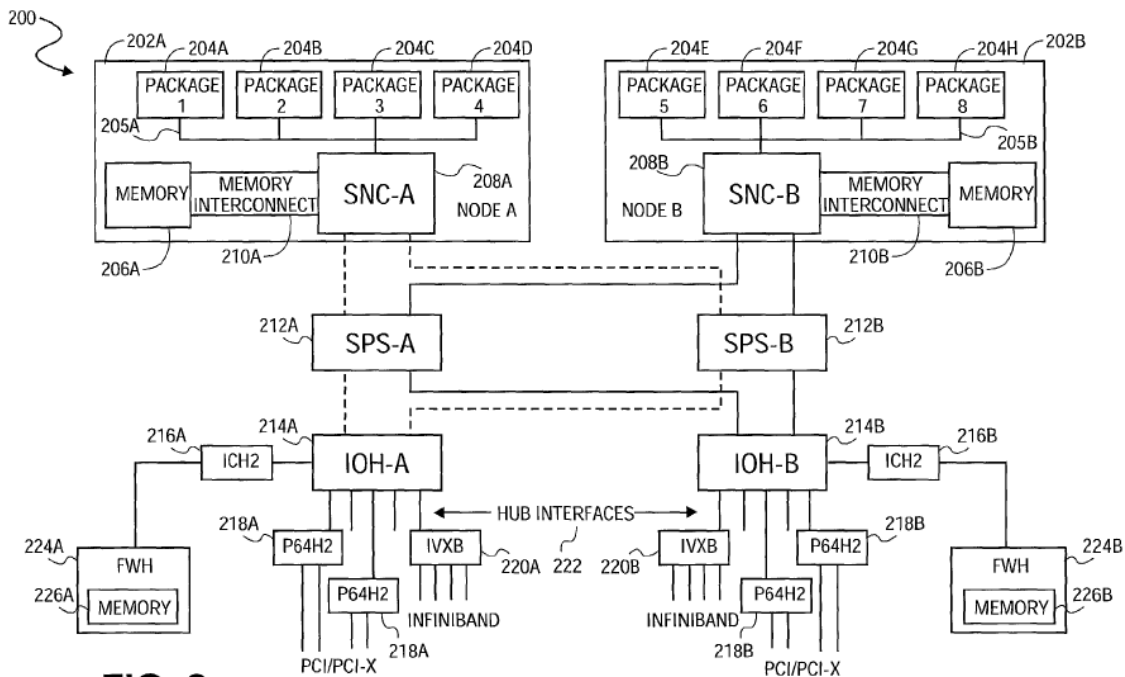
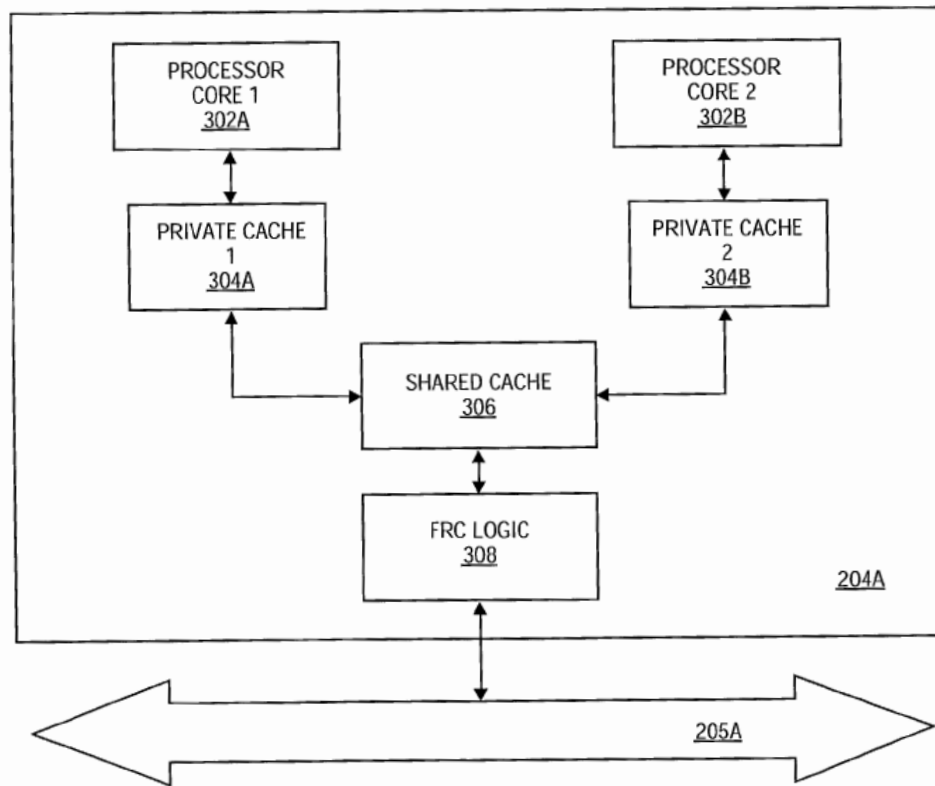


FIG. 2

Describing Figure 2, Bigbee states, “[t]he data processing system 200 of the illustrated embodiment includes a first node 202a ‘Node A’ and a second node 202b ‘Node B’ coupled together and to various shared resources (e.g. memory, mass storage devices, I/O devices, etc.) via a shared interconnect including one or more scalability port switches.” *Id.* ¶ 26. Bigbee also states that the plurality of processor packages 204a-d may contain one or more physical processor cores that include a pair of processor cores operable in an FRC mode as a single logical processor. *Id.* ¶ 27. Bigbee further states, “[a]dditional nodes (e.g. Node B 202b) may include substantially identical components operating in a substantially identical manner as those of Node A 202a.” *Id.*

Bigbee Figure 3, reproduced below, “illustrates an exemplary block diagram of processor package 204a of [Figure 2].” *Id.* ¶ 8.



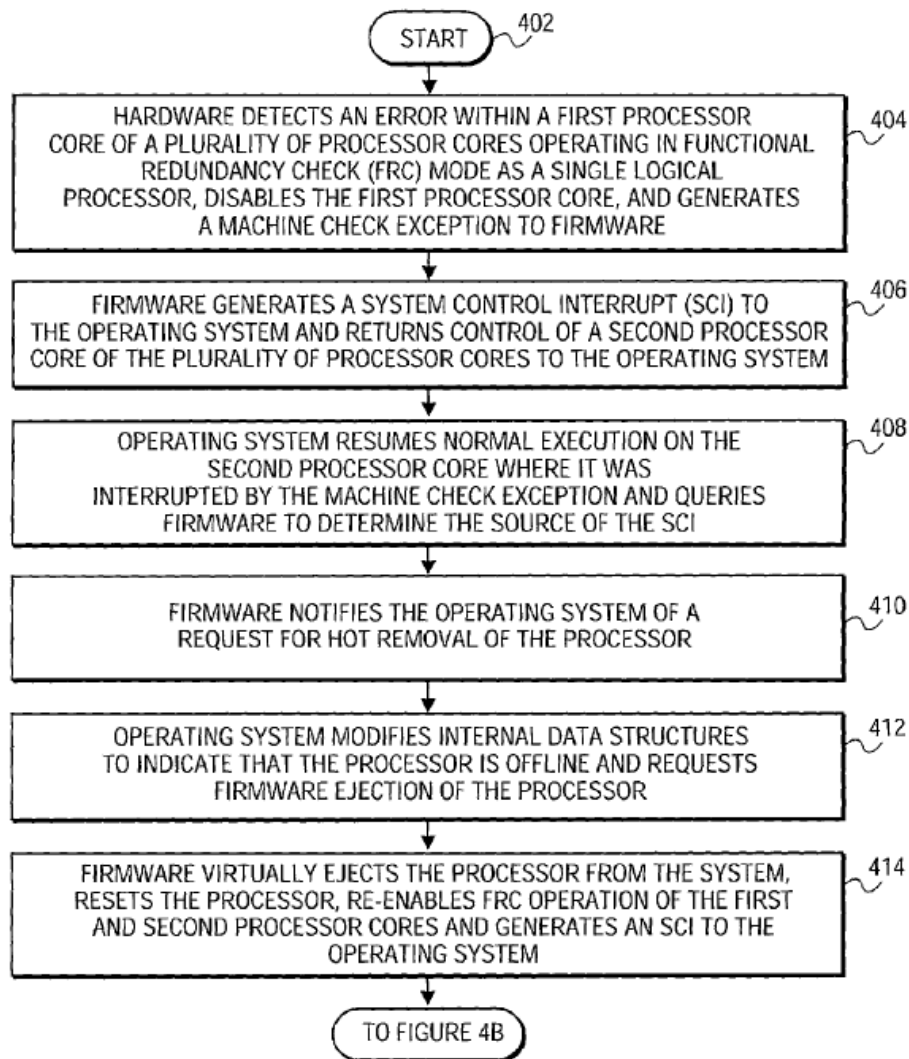
Describing Figure 3, Bigbee states:

[A] processor package 204a may be coupled to a shared “front side” bus 205a, and may include functional redundancy check (FRC) logic 308, a shared cache memory 306, and a first processor core 302a and a second processor core 302b, each having an associated private cache memory 304a and 304b, respectively.

Id. ¶ 32.

Bigbee also states in regard to Figure 3, “[b]oth cores 302a and 302b operate in an FRC mode in which identical instructions may be provided to each core and concurrently executed on identical data utilizing FRC logic 308.” *Id.* ¶ 33.

Bigbee Figure 4a, reproduced below, “illustrates a process flow diagram.” *Id.* ¶ 9.



Describing Figure 4, Bigbee states:

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).

Id. ¶ 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 Figure 1, reproduced below, “is a block diagram of a processor including dual execution cores and FRC-detection and handling logic.” *Id.* ¶ 11.

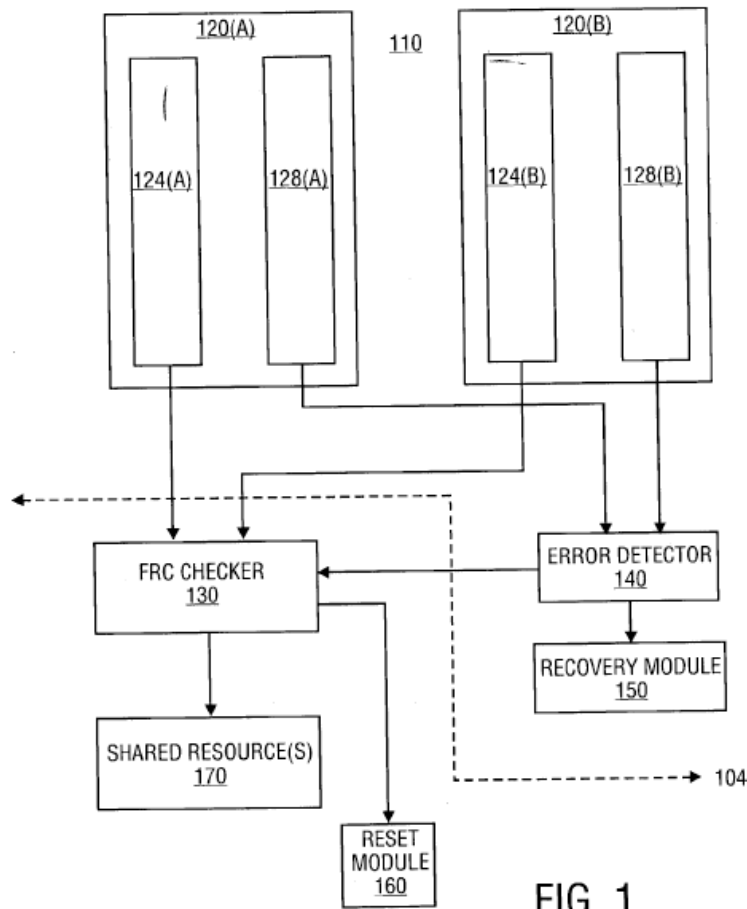


FIG. 1

Describing Figure 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.” *Id.* ¶ 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 Figure 7, reproduced below, is a block diagram of “an FRC checker that mitigates race conditions between recoverable and non-recoverable error mechanisms.” *Id.* ¶ 18.

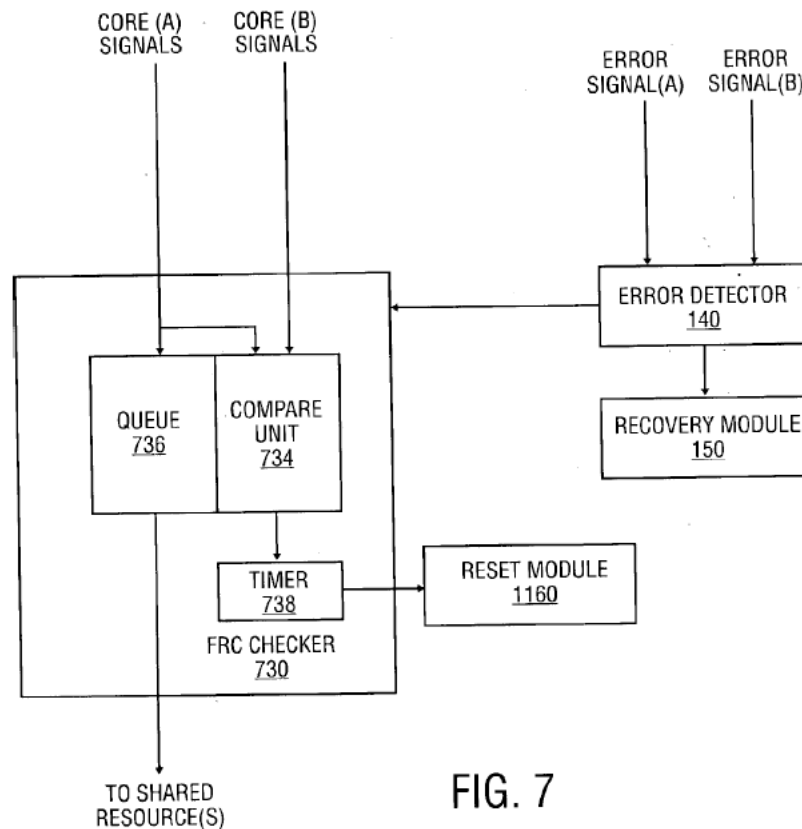


FIG. 7

Describing Figure 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.

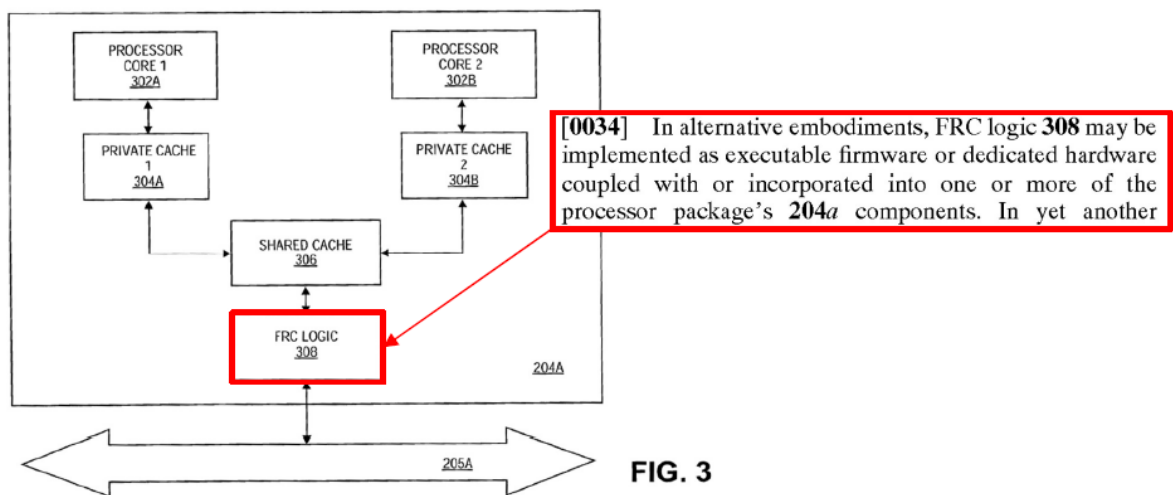
Id. ¶ 56.

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.” *Id.* ¶ 57.

3. Analysis of independent claim 19 and dependent claim 8

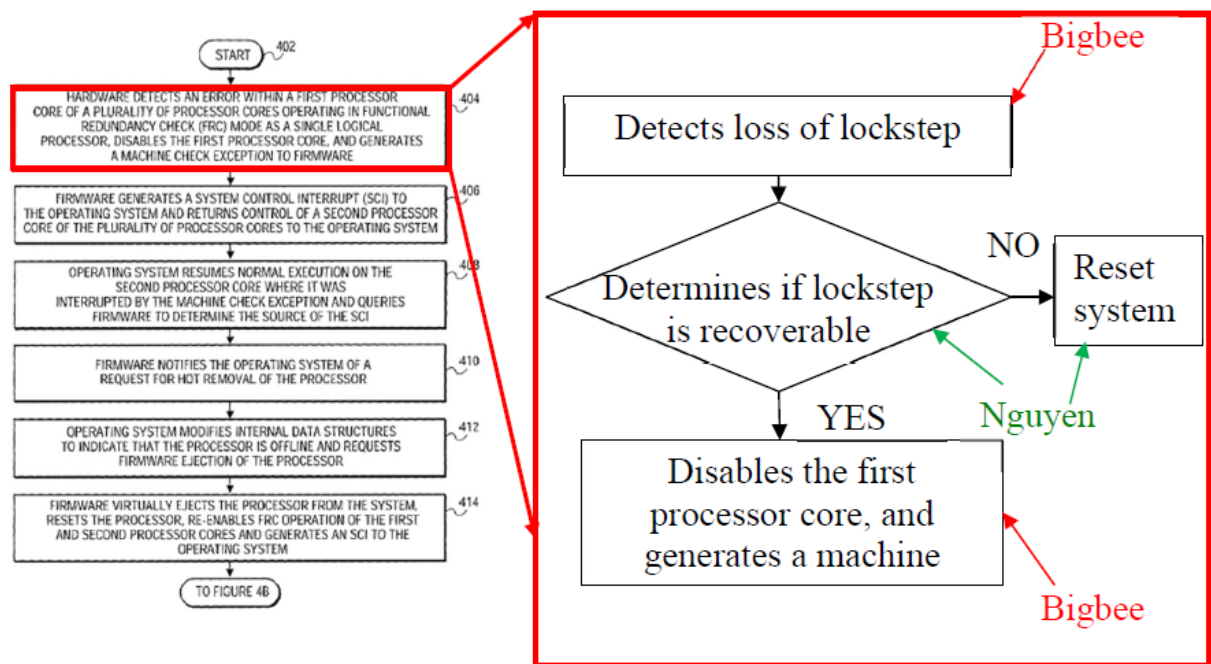
Petitioner contends, “Bigbee discloses all challenged claim limitations except ‘determining if lockstep is recoverable’”; however, Nguyen addresses Bigbee’s noted deficiency by disclosing “a method of determining if the lockstep of the processor pair is recoverable or non-recoverable.” Pet. 25. Petitioner 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). In support, Petitioner identifies certain passages and figures in the references and explains their significance with respect to the corresponding claim limitations. *Id.* at 33–43. Petitioner also provides reasons that a person of ordinary skill in the art would have been motivated to combine the teachings of Bigbee and Nguyen. *Id.* at 25–33.

Petitioner sets forth the combination with annotations to Bigbee’s Figures 3 and 4A. *Id.* at 26–27. Bigbee’s Figure 3, as discussed above and annotated by Petitioner, is reproduced below:



Bigbee’s Figure 3, “illustrates an exemplary block diagram of processor package 204a of [Figure 2].” Ex. 1005 ¶¶ 8, 32–34. Petitioner annotates Bigbee Figure 3 by adding a red border around FRC logic block 308 and by adding a sentence from Bigbee’s paragraph 34, surrounded by a rectangular box with red borders. Pet. 26.

Bigbee’s Figure 4A, as discussed above and annotated by Petitioner, is reproduced below:



Bigbee’s Figure 4A, “illustrates a process flow diagram.” Ex. 1005 ¶¶ 9, 35–36. 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. 27.

Petitioner contends a “POSITA 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.” Pet. 27 (citing Ex. 1002 ¶¶ 80–86). Petitioner further contends that a “POSITA would have been motivated to implement Nguyen’s determination of whether the detected error causing the loss of lockstep is recoverable by Bigbee’s firmware FRC logic to appropriately handle the detected error.” *Id.* Petitioner argues, “[b]ecause different responsive actions should be taken to handle each of non-recoverable and recoverable errors, determining whether the error is recoverable or non-recoverable is critical to reduce the impact of the errors on the system.” *Id.* at 27–28 (citing Ex. 1002 ¶¶ 87, 47–56; Ex. 1010 ¶ 100; Ex. 1015, 4:43–5:63, 7:24–7:27; Ex. 1016 ¶¶ 21, 27, 154).

Petitioner asserts “non-recoverable errors cause a risk of data corruption or propagating an incorrect result if the system is not immediately rebooted or reset to return to a safe state” wherein “recoverable errors can be recovered using known recovery routines that reduce the impact on the system.” *Id.* at 28 (citing Ex. 1002 ¶¶ 51–56, 88, 89). Petitioner asserts a “POSITA would have understood that it is critical to handle recoverable errors differently from non-recoverable errors” because “handling recoverable errors differently from nonrecoverable errors prevents loss of system availability.” *Id.* at 28–29 (citing Ex. 1002 ¶ 89; Ex. 1006 ¶¶ 7, 8, 22, 23, 40). Petitioner further asserts, “Nguyen teaches that treating a recoverable error as a non-recoverable error would reduce system availability, i.e., increase system downtime” whereas “[r]ecovery of lockstep from a recoverable error can be implemented with lower latency compared to the reset or repair/replacement processes for handling a non-recoverable

error.” *Id.* at 29 (citing Ex. 1002 ¶ 89; Ex. 1006 ¶¶ 7, 8, 22, 23, 40).

Petitioner explains, “handling non-recoverable errors differently from recoverable errors prevents reduction in system reliability” while “handling non-recoverable errors as recoverable creates risks of having corrupt data or incorrect results propagating through the system.” *Id.* at 29–30 (citing Ex. 1002 ¶ 90; Ex. 1010 ¶¶ 101, 103; Ex. 1015, 2:38–2:52; Ex. 1008, 1:46–1:50).

Petitioner further explains:

Reducing system availability or reliability by handling errors incorrectly—either restarting the entire system when an error can be recovered or attempting to recover a non-recoverable error and causing error propagation—would have been undesirable for fault-tolerant multiprocessor systems, particularly for systems that need to have high reliability and availability, such as server systems [Ex. 1005 ¶ 30]), resources in a network [Ex. 1010 ¶ 191], electronic fund transfer systems or airline traffic control systems [Ex. 1012, 1:19–1:24].

Id. at 30 (citing Ex. 1002 ¶ 91).

Petitioner asserts, “a POSITA would have understood that fault tolerant systems would distinguish between recoverable and non-recoverable errors by first determining whether an error is recoverable and then responding accordingly based on the determination.” *Id.* (citing Ex. 1002 ¶ 92). Petitioner further explains that a POSITA having

knowledge of handling recoverable and non-recoverable errors detected in FRC processors, “would have been motivated to use Bigbee’s FRC logic to determine whether lockstep is recoverable and take the appropriate action responsive to the recoverability of the error, thereby reducing the impact of the error and improving system reliability and availability.”

Id. (quoting Ex. 1002 ¶ 93).

Petitioner further explains:

“[A] person of ordinary skill in the art would have understood how to implement Nguyen’s determining whether a detected error is recoverable or non-recoverable using Bigbee’s firmware FRC logic with a reasonable expectation of success” because “implementing logical steps in firmware was known in the art.” Ex. 1002 ¶94 (citing Ex. 1005 ¶ 34], Ex. 1009, 15:60–64 (lockstep logic may be implemented in firmware)).

Id. at 31.

Petitioner asserts:

A POSITA 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—its operating system is ACPI-compliant, and its firmware includes ACPI BIOS code and a processor abstraction level (PAL) that provides an abstraction of implementation-specific processor features. Ex. 1002 ¶95; [Ex. 1005 ¶¶ 16–18].

Id.

Petitioner further explains:

[a] person of ordinary skill in the art would have known how to use ACPI Source Language (ASL) to program Bigbee’s firmware code to implement Nguyen’s determining step using ordinary computer programming skill” because “using such firmware, such as the PAL or SAL layer of ACPI BIOS, for implementing error diagnosis and recovery routines was well known,” involving only ordinary skill and common knowledge.

Id. (citing Ex. 1002 ¶ 96; Ex. 1016, Figs. 1–3, ¶¶ 17, 18, 27, 154; Ex. 1015, Figs. 1, 2, 5:48–8:64) (alteration in original).

Petitioner explains, “Implementing Nguyen’s determining step in firmware was also known to be one of only three available alternatives—firmware, hardware, or combination—all of which would have been obvious

to try, and any of which would be a suitable option.” *Id.* at 32 (citing Ex. 1002 ¶ 97; *Intel Corp. v. PACT XPP Schweiz AG*, 61 F.4th 1373, 1380–81 (Fed. Cir. 2023)).

Petitioner further explains:

the benefits of implementing Nguyen’s determination in firmware, as would have been understood by a POSITA— “[F]irmware is flexible. It can be modified after production, updated, and supported relatively quickly and inexpensively. Hardware, on the other hand, is inflexible. Once manufactured, it cannot be easily modified, and any modification or revision is often expensive because it requires remanufacture of new components.”

Id. (citing Ex. 1002 ¶ 98) (alteration in original).

- a. [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. Pet. 33 (citing Ex. 1002 ¶¶ 100–101). Petitioner also 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 33–34 (citing Ex. 1005 ¶ 19, Fig. 3) (emphasis omitted). Petitioner’s annotated Bigbee Figure 3 is reproduced below:

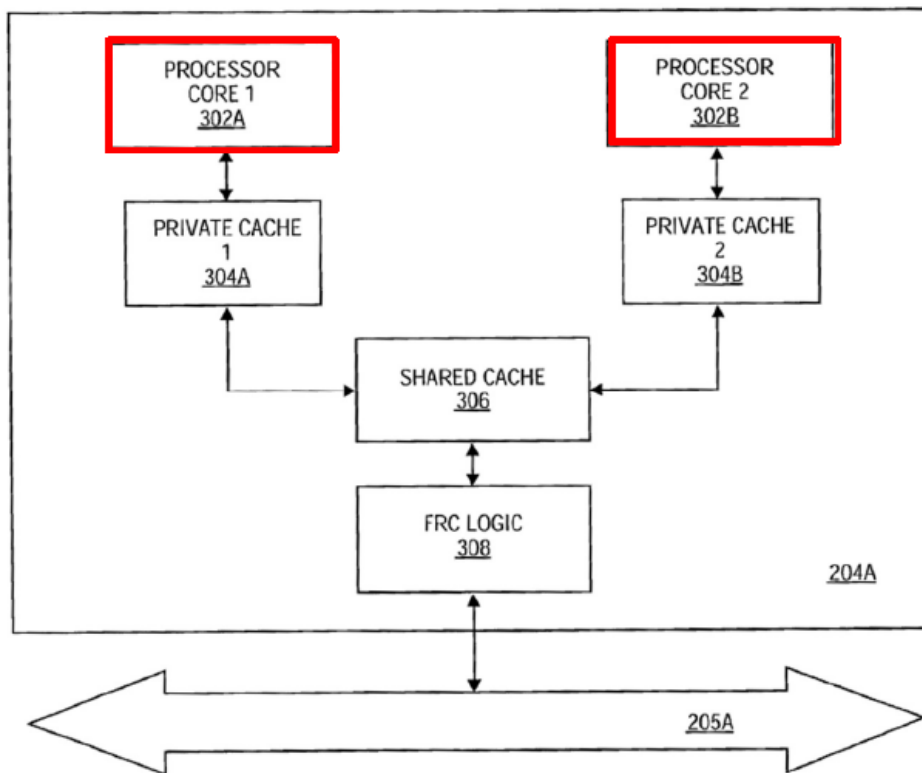


FIG. 3

Petitioner’s annotated Figure 3 above depicts Bigbee’s processor cores 1 and 2 (302A, 302B), annotated with a red border, within the processor package 204a. *Id.* at 34. Petitioner states “[f]irmware 104 includes ACPI [advanced configuration and power interface] BIOS including ACPI machine language program code (‘computer-executable firmware code’).” *Id.* (citing Ex. 1005 ¶¶ 16–17).

Petitioner’s annotated Bigbee Figure 2 is reproduced below:

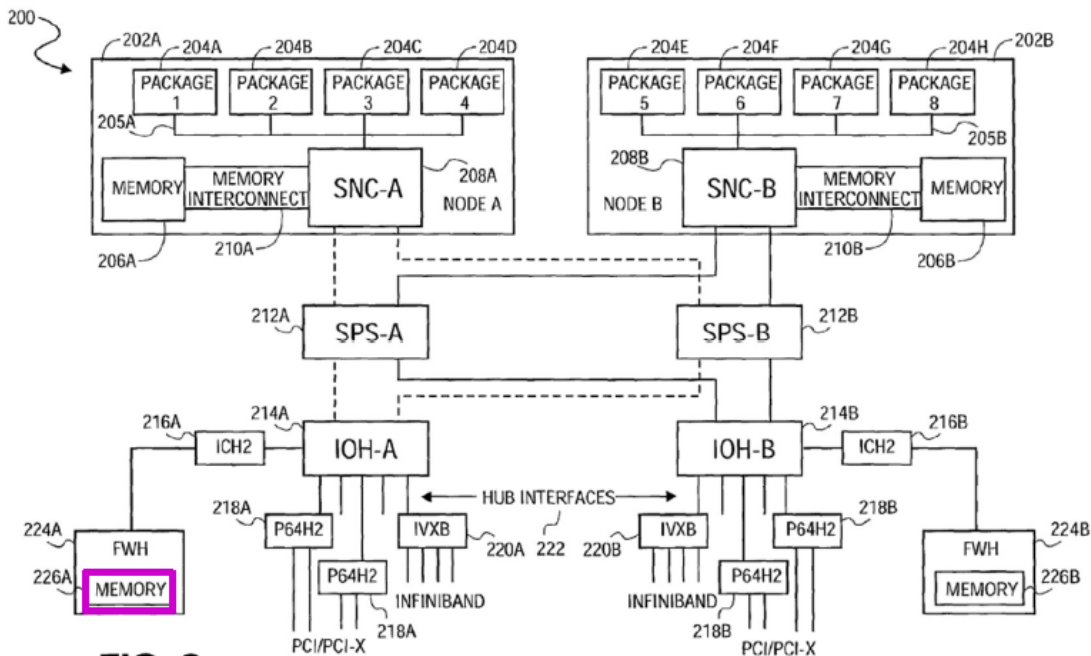


FIG. 2

Petitioner’s annotated Figure 2, above, depicts Bigbee’s data processing system including two or more nodes (202A, 202B), each of the nodes including a plurality of FRC processor packages. Pet. 34–35 (citing Ex. 1005 ¶¶ 26–27). Bigbee’s processing system includes non-volatile memory 226a (“computer-readable medium”), annotated with a purple border, that stores firmware code (e.g., SAL or ACPI BIOS program code). *Id.* at 34 (“executable instructions embodied within a machine accessible medium, such as a ROM, accessible by a machine (e.g., a data processing system or computer),” citing Ex. 1005 ¶¶ 29, 12–13, also citing Ex. 1002 ¶ 101).

At this stage, Patent Owner does not dispute Petitioner’s contentions. *See* Prelim. Resp. Based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that Bigbee

teaches the recited elements of claim 19's preamble for purposes of institution.⁵

b. [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”*

Petitioner contends, “Bigbee in view of Nguyen renders obvious [19a].” Pet. 35 (citing Ex. 1002 ¶¶ 102–107). Petitioner also contends, “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.* (citing Ex. 1002 ¶ 103; Ex. 1005 ¶¶ 19, 34). Petitioner states that Bigbee discloses, in Figure 3, “a processor package including two processor cores, each with a private cache, which are coupled to a shared cache.” *Id.* (citing Ex. 1005 ¶ 32). Petitioner also states, “the two processor cores [shown in Figure 3] function as a single logical processor and operate in an FRC mode (‘the pair of lockstep processors’), executing identical instructions on identical data to generate results for storage in associated private cache and/or the shared cache.” *Id.* (citing Ex. 1005 ¶ 33). Petitioner contends,

Bigbee discloses ‘detection of loss of lockstep between the pair of lockstep processors’ because FRC logic detects both inconsistencies in the generated results that indicate an FRC mismatch error and errors within the processor cores of a

⁵ Petitioner does not argue that the preamble limits claim 19. Although we find that the evidence supports that the prior art teaches the preamble, we make no determination at this stage of the proceeding whether the preamble of claim 19 is limiting.

processor package that cause a break in the lockstep between the cores, disabling FRC mode operation.

Id. at 35–36 (citing Ex. 1005 ¶¶ 19, 33; citing also Ex. 1002 ¶ 104).

Petitioner states, “Nguyen discloses determining if lockstep is recoverable from lockstep errors, as was also known in the art.” *Id.* at 36 (citing Ex. 1002 ¶ 105). Petitioner contends, “Nguyen discloses responsive to the detection of an error in the processor cores (‘responsive to detection of loss of lockstep between the pair of lockstep processors’), the FRC checker and error detector operate together [to] determine whether lockstep is recoverable from the detected error.” *Id.* at 36–37 (citing Ex. 1006 ¶¶ 22–25, 30, 37, 39–41, 46, 56, 57; citing also Ex. 1002 ¶ 105 (citing the same)).

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.” *Id.* at 37 (citing Ex. 1002 ¶¶ 66–69, 106). Petitioner states, “**First**, Nguyen’s FRC checker and error detector determine a mismatch error that cannot be traced to the processor containing the underlying soft error is not recoverable.” *Id.* (citing Ex. 1006 ¶¶ 7, 23). Petitioner states, “**Second**, Nguyen’s FRC checker and error detector determine that lockstep is recoverable from a soft error detected by the error detector before detecting a mismatch error.” *Id.* (citing Ex. 1006 ¶¶ 5, 22–24, 46). Petitioner states, “**Third**, Nguyen’s FRC checker and error detector determine lockstep is recoverable from a mismatch error if an error in the processors is detected within a timeout interval from the detection of the mismatch error.” *Id.* (citing Ex. 1006 ¶¶ 23, 30; citing also Ex. 1002 ¶ 106; Pet. § VII.A.2).

Petitioner contends 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.* (citing Ex. 1002 ¶ 107); *see also id.* at 27–33.

Patent Owner argues, “[a] critical flaw in Petitioner’s combination is that—as Bigbee expressly discloses, and Petitioner’s expert acknowledged—the recovery process begins after Bigbee has detected an ‘error within a first processor.’” Prelim. Resp. 12 (citing Ex. 1005, Fig. 4A (block 404); Ex. 2003, 74:14–76:8; Ex. 2001 ¶ 27). Patent Owner argues, “this means that Nguyen will never determine that lockstep is non-recoverable.” *Id.* Patent Owner further argues, “[e]very error processed by Nguyen’s ‘determines if lockstep is recoverable’ routine will be handled the same way, i.e., it 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.” *Id.* (citing Ex. 2001 ¶¶ 29–30).

Patent Owner contends, “[t]his outcome is the result of how Nguyen’s ‘determination’ process works’ and as Petitioner acknowledges,” Nguyen determines a lockstep error is non-recoverable only when (1) a mismatch error is detected and (2) a soft error (e.g., individual processor error) is not detected before, or within a timeout interval after, a mismatch error is detected.” *Id.* at 12–13 (citing Ex. 1002 ¶ 106; Ex. 1006 ¶¶ 5, 7, 23, 24, 30, 46; Pet. 37). Patent Owner contends, “[o]utside of these conditions—e.g., whenever an individual error is detected prior to detection of a

mismatch error—Nguyen will not determine that lockstep is non-recoverable.” *Id.* at 13. Patent Owner asserts, “when Nguyen detects an error from an individual processor, it disables error detector 140 (used to detect mismatch errors) and directly activates recovery module 150 to implement a recovery routine.” *Id.* (citing Ex. 1006 ¶¶ 22, 23). Patent Owner argues, “[i]n Nguyen, Error Detector 140 receives signals from error pipeline 128, which represents the logic that detects errors in the data” and “[a]s Nguyen describes, ‘soft errors in these arrays may appear as parity or ECC error flags when the corrupted data is accessed.’” *Id.* (citing Ex. 1006 ¶ 21; Ex. 2001 ¶ 32). Patent Owner argues, “Bigbee similarly describes that errors in an individual processor can be ‘error correcting code (ECC) errors...or parity errors.’” *Id.* (citing Ex. 1005 ¶ 18) (alteration in original); *see* Ex. 2003, 14:18–15:10; Ex. 1002 ¶ 81.

According to Patent Owner, “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” and “Nguyen’s alleged ‘determination’ technique would not, as combined with Bigbee, have any capability to distinguish recoverable from non-recoverable errors.” Prelim. Resp. 14. Patent Owner argues, the Nguyen-Bigbee combined system “would provide none of the ‘critical’ benefits of handling recoverable and non-recoverable errors differently that Petitioner says a POSITA would recognize.” *Id.* Patent Owner surmises, “a POSITA would readily recognize that the Nguyen-Bigbee ‘combined system’ provides no benefit in terms of distinguishing recoverable from non-recoverable errors, does not allow non-recoverable errors to be treated differently from recoverable errors.” *Id.*

Based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that the combination of Bigbee and Nguyen teaches the recited elements of limitation 19a for purposes of institution and has provided reasons sufficiently supported by factual underpinnings to combine the references with a reasonable expectation of success.

Nguyen discloses, “[t]he present invention addresses mechanisms for combining recoverable and non-recoverable error handling mechanisms efficiently in FRC-enabled processors.” Ex. 1006 ¶ 9. Subsequently, Nguyen distinguishes recoverable from non-recoverable errors by treating non-recoverable errors differently from recoverable errors. Nguyen discloses, “If an error reaches error detector 140 from either core 120, recovery module 150 is activated to implement a recovery routine. Recovery can be implemented with relatively low latency by hardware, software, firmware or some combination of these.” *Id.* ¶ 22, Fig. 1. “However, an FRC error will be triggered if the corrupted data from one execution core and an uncorrupted version of the data from the other execution are allowed to reach FRC checker 130 before recovery module 150 is activated.” *Id.* Nguyen recognizes, “[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.” *Id.* Nguyen further recognizes that, “[n]ot all FRC-errors are traceable to underlying parity/ECC or other correctible soft errors. For those that are, it is faster for error detector 140 to address the underlying soft error than for FRC-checker to address the FRC error that results when the corrupted data reaches FRC boundary 104.” *Id.* ¶ 23 (emphasis

added). In this instance instead of the FRC-error being processed by the FRC checker 130 and the Reset module 160, the FRC error is directed to the Error detector 140 and eventually recovery module 150. *See id.* at Fig. 1.

Nguyen also discloses, “the recovery routine triggered by error detector 140 provides greater system availability than the reset routine triggered by FRC checker 130” and “[e]xpanding the FRC boundary thus increases both the amount of logic that is duplicated for execution cores 120 and the flight time, during which detectable errors may be identified.” *Id.* ¶ 30, Fig 1; *see* Pet. 37. Nguyen further discloses, “[t]he former increases FRC protection, albeit through a reset mechanism” as shown in Figure 1, elements FRC Checker 130 and Reset Module 160. Ex. 1006 ¶ 30. “The latter increases the likelihood that errors identifiable through parity, ECC [error correction code] or similar core-specific protections are handled through recovery rather than reset” as shown in Figure 1, elements Error Detector, 140 and Recovery Module 150. *Id.*

Additionally, Nguyen’s Figure 7, as discussed above, “is a block diagram illustrating an embodiment of an FRC-checker 730 that mitigates race conditions between recoverable and non-recoverable error handling.” Ex. 1006 ¶ 56. Nguyen discloses, “[e]mbodiments of the present invention detect virtually all errors that are undetected in non-FRC-enabled systems, and they support recovery of virtually all detectable errors, including those that are typically handled through reset in other FRC-enabled processors.” *Id.* ¶ 59. Although Nguyen is combined with Bigbee in Petitioner’s proposed combination, as discussed above, contrary to Patent Owner’s assertions, Nguyen addresses recoverable and non-recoverable errors differently.

Patent Owner argues:

In addition, Petitioner’s contention that it would have been obvious to “implement Nguyen’s determination of whether the detected error causing the loss of lockstep is recoverable by Bigbee’s firmware FRC logic” is flawed because Petitioner has not shown that implementing, in *firmware*, the portions of Nguyen on which it relies would have been obvious.

Prelim. Resp. 14 (citing Pet. 27; Ex. 1002 ¶ 86). Patent Owner contends, supported by testimony from Dr. Jones, that implementing Nguyen’s functionality in firmware code would be impractical, more difficult than a hardware implementation, and more vulnerable than hardware to malicious attacks. *Id.* at 15 (citing Ex. 2001 ¶¶ 32). Patent Owner further argues that Petitioner’s contentions are unsupported and that the combination would introduce more opportunities for error. *Id.* at 15–16 (citing Ex. 2001 ¶ 33).

As described above, Petitioner has set forth reasons, supported by testimony from Dr. Brogioli explaining why, in Petitioner’s view, a person of ordinary skill in the art would have been motivated to implement Nguyen’s functionality using Bigbee’s firmware FRC logic. *See* Pet. 31–33. Patent Owner’s arguments raise issues that are best resolved on a full record. For purposes of institution, we find Petitioner’s contentions sufficient.

At this stage of the proceeding, we find Petitioner has sufficiently shown that the combination of Nguyen and Bigbee teaches limitation 19a for purposes of institution. *See KSR*, 550 U.S. at 418 (“As our precedents make clear, however, the analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim, for a court can take account of the inferences and creative steps that a person of ordinary skill in

the art would employ.”).

- c. [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. 38 (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.* (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).

In regard to Nguyen’s “determining if [a] lockstep is recoverable,” Petitioner provides two examples:

- (1) Nguyen’s FRC checker’s compare unit compares the data from both cores, and it sets a status flag to indicate whether the data matches or mismatches (“determining if mismatch has occurred”); and (2) determining if Nguyen’s error detector detects an error in the processors before (recoverable) or after (non-recoverable) the timeout interval of timer unit expires.

Id. at 38 (citing Ex. 1002 ¶ 110; Ex. 1006 ¶¶ 56–57; Pet. § VII.A.2).

Petitioner argues, “a POSITA would have been motivated to incorporate Nguyen’s determining 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.” *Id.* (citing Ex. 1002 ¶ 111).

Patent Owner argues that a “POSITA would not have been motivated to combine Bigbee and Nguyen.” *See* Prelim. Resp. 8–17 (emphasis omitted). As we discussed above in regard to limitation 19a, based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that the combination of Bigbee and Nguyen teaches the recited elements of limitation 19b for purposes of institution, and has provided reasons sufficiently supported by factual underpinnings to combine the references with a reasonable expectation of success.

d. [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. 39 (citing Ex. 1002 ¶¶ 112–116). Petitioner maintains that, “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 an obvious incorporation/modification to a person of ordinary skill. *Id.* (citing Pet. § VII.A.5). 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).

Petitioner argues, “**First**, Bigbee discloses ‘firmware code ... for triggering an operating system to idle the processors.’” *Id.* (citing Ex. 1002 ¶ 114). Petitioner states, “Bigbee discloses that firmware (e.g., SAL

program code) generates a system control interrupt (SCI) to the operating system.” *Id.* (citing Ex. 1005 ¶ 35, Fig. 4A (block 406)). Petitioner further states, “[i]n response to the SCI, the operating system queries firmware (ACPI BIOS) to determine the source of the SCI; firmware responds to the query by notifying the operating system of a request for processor hot removal (e.g., by a ACPI Notify command).” *Id.* at 39–40 (citing Ex. 1005 ¶ 36, Fig. 4A (blocks 408–410 (“triggering an operating system”))) (note omitted). Petitioner states, “[t]he operating system then modifies its internal data structures to indicate that the processor is offline and requests firmware ejection of the processor (e.g., by executing an _EJ_X control method (‘to idle the processors’)).” *Id.* at 40 (citing Ex. 1005 ¶¶ 36–37, Fig. 4A (block 412)).

Petitioner further argues, “**Second**, Bigbee discloses ‘firmware code ... for attempting to recover lockstep between the pair of processors.’” *Id.* at 41 (citing Ex. 1002 ¶ 115). Petitioner states, “[i]n response to the operating system’s ejection request, Bigbee’s firmware virtually ejects the processor from the system, resets the processor, re-enables FRC operation of the processor cores (‘attempting to recover lockstep between the pair of processors’) and generates an SCI to the operating system.” *Id.* (citing Ex. 1005 ¶ 37, Fig. 4A (block 414)).

Petitioner contends, “[t]he ’958 [p]atent similarly discloses idling the processors, the firmware uses an ACPI method to ‘eject’ the processors, thereby triggering the operating system to idle the master processor, rests the processor pair, and recovers lockstep.” *Id.* at 42 (citing Ex. 1001, 6:9–17, 6:40–42, 6:55–57; Ex. 1002 ¶ 116).

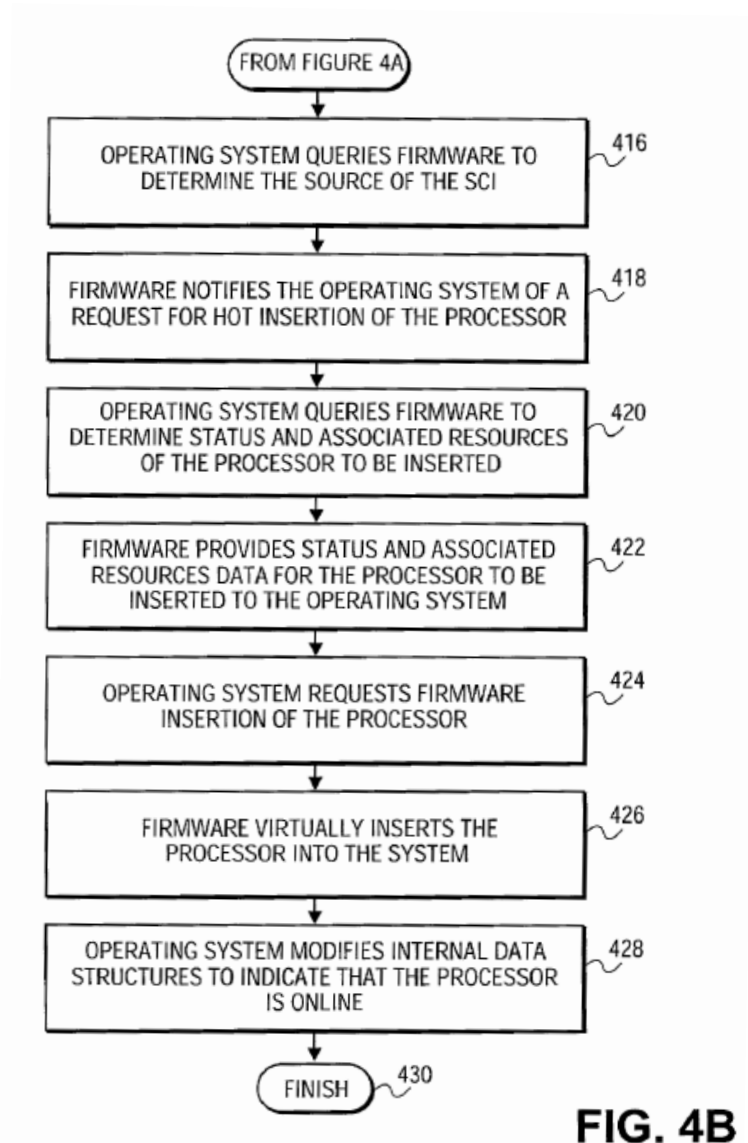
Patent Owner argues that a “POSITA would not have been motivated to combine Bigbee and Nguyen.” *See* Prelim. Resp. 8–17 (emphasis

omitted). As we discussed above in regard to limitation 19a, based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that the combination of Bigbee and Nguyen teaches the recited elements of limitation 19c for purposes of institution, and has provided reasons sufficiently supported by factual underpinnings to combine the references with a reasonable expectation of success.

- e. [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. 42 (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)).

Bigbee Figure 4B, reproduced below, “illustrates a process flow diagram for a second portion of a method” where the first portion of the method is shown in Figure 4A. Ex. 1005 ¶ 10.



Describing Figure 4B, Bigbee states, “in the process illustrated in FIG. 4b, operating System Software queries firmware (e.g. ACPI BIOS) to determine the source of the previously generated SCI (block 416). Firmware responds to the OS query by notifying the operating system of a request for processor hot insertion (block 418).” *Id.* ¶ 37.

Petitioner further states, “[t]he firmware also provides the operating system with status and resource data of the processor to be inserted via one

or more ACPI methods.” Pet. 42 (citing Ex. 1005 ¶ 37, Fig. 4B (block 430); Ex. 1002 ¶ 118). Petitioner states, “[t]he operating system then requests insertion of the processor by calling an ACPI method (e.g., _PS0), which the firmware virtually executes to insert the processor into the system.” Pet. 42 (citing Ex. 1005 ¶ 37, Fig. 4B (blocks 424, 426). Petitioner further states, “[o]nce the processor has been inserted, the operating system modifies its internal data structures to indicate that the processor is online and available (‘recognize the processors as being available for receiving instructions’).” *Id.* at 42–43 (citing Ex. 1005 ¶ 37, Fig. 4B (block 430); Ex. 1002 ¶ 118).

At this stage, Patent Owner does not dispute Petitioner’s contentions. *See* Prelim. Resp. Based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that Bigbee teaches the recited elements of limitation 19d for purposes of institution.

f. Claim 8, “The method of claim 1 further comprising: if determined that a lockstep mismatch has occurred between the lockstep pair of processors, then determining that the loss of lockstep is not recoverable.”

Dependent claim 8 depends from independent claim 1. Ex. 1001, 15:46. Noting that independent claim 1 is “substantially similar” to independent claim 19, Petitioner provides essentially the same analysis for independent claim 1 as for independent claim 19. Pet. 44–45. Patent Owner argues independent claim 1 and independent 19 together and provides the same arguments (Prelim. Resp. 8–18), which we have addressed above. For the same reasons as set forth above for independent claim 19, we determine that the information presented in the Petition demonstrates a reasonable

likelihood that claim 1 is unpatentable under 35 U.S.C. § 103(a) over Bigbee and Nguyen.

Turning to claim 8, Petitioner contends, “Bigbee in view of Nguyen renders obvious claim 8.” Pet. 60 (citing Ex. 1002 ¶¶ 161, 162). “As explained for [19a], Nguyen discloses determining that lockstep is not recoverable from a non-recoverable FRC error indicating mismatch (‘if determined that a lockstep mismatch has occurred’).” *Id.* (citing §§ VII.A.2, VII.B.2, Ex. 1002 ¶ 162). Petitioner contends, “Nguyen discloses the FRC checker signals an FRC error after detecting a mismatch (‘if determined that a lockstep mismatch has occurred between the lockstep pair of processors’) before the error detector detects the underlying soft error and resets the system.” *Id.* (citing Ex. 1006 ¶ 22). Petitioner further contends, “the FRC checker signals an FRC error and triggers resetting the system if the error detector does not detect an error within a timeout interval after detecting a mismatch (‘if determined that a lockstep mismatch has occurred between the lockstep pair of processors’).” *Id.* (citing Ex. 1006 ¶¶ 56–58). Petitioner contends, “[i]n Bigbee combined with Nguyen, 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.* at 60–61 (citing Ex. 1005 ¶ 33; Ex. 1002 ¶ 162).

Patent Owner contends, “Petitioner has also failed to show that claim 8 is obvious in view of Bigbee and Nguyen.” Prelim. Resp. 18. Patent Owner contends, “Petitioner has not shown that the Nguyen-Bigbee ‘combined system’ ever makes a determination that ‘loss of lockstep is

not recoverable.” *Id.* Patent Owner argues, “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.* We disagree with Patent Owner’s assertions because, as discussed above in section III.D.3. a–c, although Nguyen is a combined system, Nguyen addresses recoverable and non-recoverable errors differently. Based on our review and consideration of the current record, we determine that Petitioner has sufficiently shown that the Bigbee and Nguyen combination teaches the recited elements of claim 8 for purposes of institution.

4. Conclusion for claims 1–7, 13–18 and 20–22

Based on our review and consideration of the current record, we determine that Petitioner has adequately shown for purposes of institution that the combination of Bigbee and Nguyen sufficiently teaches the limitations recited in claims 1, 19 and 8. We also determine that a person of ordinary skill in the art would have been motivated to combine the teachings of Bigbee and Nguyen as shown by Petitioner. Accordingly, we determine, on the current record and for purposes of this Decision, that the information presented in the Petition demonstrates a reasonable likelihood that claims 1, 19 and 8 are unpatentable under 35 U.S.C. § 103(a) over Bigbee and Nguyen. As set forth in Section II.E above, Petitioner additionally challenges claims 2–7, 13–18 and 20–22 as unpatentable over Bigbee and Nguyen (Pet. 49–72), which are unchallenged by Patent Owner on the record before us. *See* Prelim. Resp. 8–18.

We determine above that Petitioner demonstrates a reasonable likelihood of prevailing in its challenge to claims 1, 19 and 8 of the '958 patent; therefore, we institute *inter partes* review on all claims and grounds. See 37 C.F.R. § 42.108(a) (“When instituting . . . 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.”).

IV. JOINDER MOTION

Joinder in *inter partes* review proceedings is subject to the provisions of 35 U.S.C. § 315(c):

(c) JOINDER.—If the Director institutes an inter partes review, the Director, in his or her discretion, may join as a party to that inter partes review any person who properly files a petition under section 311 that the Director, after receiving a preliminary response under section 313 or the expiration of the time for filing such a response, determines warrants the institution of an inter parties review under section 314.

“To join a party to an instituted [*inter partes* review (IPR)], the plain language of § 315(c) requires two different decisions.” *Facebook, Inc. v. Windy City Innovations, LLC*, 973 F.3d 1321, 1332 (Fed. Cir. 2020). “First, the statute requires that the Director (or the Board acting through a delegation of authority[]) . . . determine whether the joinder applicant’s petition for IPR ‘warrants’ institution under § 314.” *Id.* “Second, to effect joinder, § 315(c) requires the Director to exercise h[er] discretion to decide whether to ‘join as a party’ the joinder applicant.” *Id.*

As the moving party, Petitioner bears the burden of proving that it is entitled to joinder. 37 C.F.R. § 42.20(c). A motion for joinder should: (1)

set forth the reasons joinder is appropriate; (2) identify any new grounds of unpatentability asserted in the petition; and (3) explain what impact (if any) joinder would have on the trial schedule for the existing review. *See Kyocera Corp. v. Softview LLC*, IPR2013-00004, Paper 15 at 4 (PTAB Apr. 24, 2013).

A motion for joinder must be filed “no later than one month after the institution date of any *inter partes* review for which joinder is requested.” 37 C.F.R. § 42.122(b).

A. Whether the Petition Warrants Institution

To determine whether joinder is appropriate, we must first determine whether the Petition warrants institution. *See supra*.

In the analysis above, we conclude, for reasons discussed, that Petitioner has demonstrated a reasonable likelihood of prevailing with respect to at least one claim of the ’958 patent challenged in the Petition. *See supra*. Accordingly, we determine that Petitioner has met its burden of showing that the Petition warrants institution of *inter partes* review on all claims and all challenges asserted in the Petition.

B. Whether to Join Petitioner as a Party to IPR2023-01373

As noted, Petitioner has filed a timely Joinder Motion in accordance with 35 U.S.C. § 315(c). *See* Paper 7 (according a filing date for the Petition of April 18, 2024); IPR2023-01373, Paper 11 (mailed March 20, 2024); *see also* Joinder Motion 4. Petitioner’s Joinder Motion is unopposed (although Patent Owner disputes the merits of the Petition and asserts that a trial should not be instituted).

Petitioner states that the Petition here “challenges the validity of the same claims of the ’958 Patent on identical grounds to those in the BMW Petition [in IPR2023-01373].” Joinder Motion 5; *see* IPR2023-01373, Paper 1. Petitioner continues, “[t]he only differences between the . . . Petition and the BMW Petition are the Mandatory Notices and [the Petition’s] analysis of the discretionary factors the Board employs for its institution decisions.” Joinder Motion 5 n.1. Petitioner further states, “[the Petition] also relies on the same supporting evidence . . . as is relied on in the BMW Petition.” *Id.* at 5. Therefore, the Petition does not present any new grounds or issues. *See* Joinder Motion 3–4–7. Petitioner has shown that the trial schedule in IPR2203-01373 will not be affected by joinder. *Id.* at 7–8. No changes in the schedule in IPR2023-01373 are anticipated or necessary, and the limited participation, if at all, of Petitioner will not impact the timeline of the ongoing trial. *Id.* Petitioner has agreed to an “understudy” role, which will simplify matters if Petitioner is joined to the ongoing IPR2023-01373, and joining Petitioner to IPR2023-01373 will be an efficient use of resources because it will allow the same issues to be decided in a single proceeding. *Id.* at 8.

Accordingly, we determine that Petitioner has met its burden of showing that we should exercise our discretion to join Petitioner to IPR2023-01373.

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that pursuant to 35 U.S.C. § 314(a) and 37 C.F.R. § 42.4, *inter partes* review is instituted as to the challenged claims of the '958 patent and all challenges of unpatentability presented in the Petition; and

FURTHER ORDERED that Petitioner's Motion for Joinder to IPR2023-01373 is *granted* and Petitioner is joined as a party to IPR2023-01373, pursuant to 37 C.F.R. § 42.122(b);

FURTHER ORDERED that all future filings are to be made only in IPR2023-01373;

FURTHER ORDERED that the grounds on which an IPR was instituted in IPR2023-01373 remain unchanged, and no new grounds are instituted in IPR2023-01373;

FURTHER ORDERED that the Scheduling Order currently in place in IPR2023-01373 (Paper 12) shall govern the schedule in that proceeding after joinder;

FURTHER ORDERED that, unless and until BMW is terminated from IPR2023-01373, Petitioner is bound by all filings by BMW without a separate opportunity to be heard, whether orally or in writing (except for filings regarding termination or settlement and filings where Petitioner receives permission from the Board to file an independent paper);

FURTHER ORDERED that Petitioner must obtain prior Board authorization to file any paper or take any action on its own in IPR2023-01373 (except a motion to modify these provisions on the ground that BMW is no longer an active party);

FURTHER ORDERED that the case caption in IPR2023-01373 for all further submissions shall be changed to reflect joinder of Petitioner, as indicated in the attached sample case caption;

FURTHER ORDERED that Petitioner's participation in the briefing, depositions, and oral argument shall be subject to BMW's acquiescence to Petitioner's participation and, absent our express authorization, Petitioner shall not file papers or exhibits separate and apart from BMW;

FURTHER ORDERED that the roles of BMW and Foras in IPR2023-01373 remain unchanged; and

FURTHER ORDERED that a copy of this Decision be entered into the file of IPR2023-01373.

IPR2024-00823
Patent 7,502,958 B2

For PETITIONER:

Russell E. Levine
Kourtney Baltzer
KIRKLAND & ELLIS LLP
rlevine@kirkland.com
kourtney.baltzer@kirkland.com
Valeo_Foras@kirkland.com

PATENT OWNER:

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

Robert A. Auchter
AUCHTER PLLC
robert@auchterlaw.com

IPR2024-00823
Patent 7,502,958 B2

[Sample Case Caption]

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

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

IPR2023-01373⁶
Patent 7,502,958 B2

⁶ 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.