

Case No: IPR2023-01373
Patent No. 7,502,958

BMW of North America, LLC and Robert Bosch LLC,
Petitioners,

v.

Foras Technologies Ltd.,
Patent Owner

Petitioners' Oral Hearing Demonstratives
January 7, 2025

BMW and BOSCH EXHIBIT 1037
BMW-BOSCH v. FORAS
IPR2023-01373

Overview of the '958 Patent

Overview of the '958 Patent

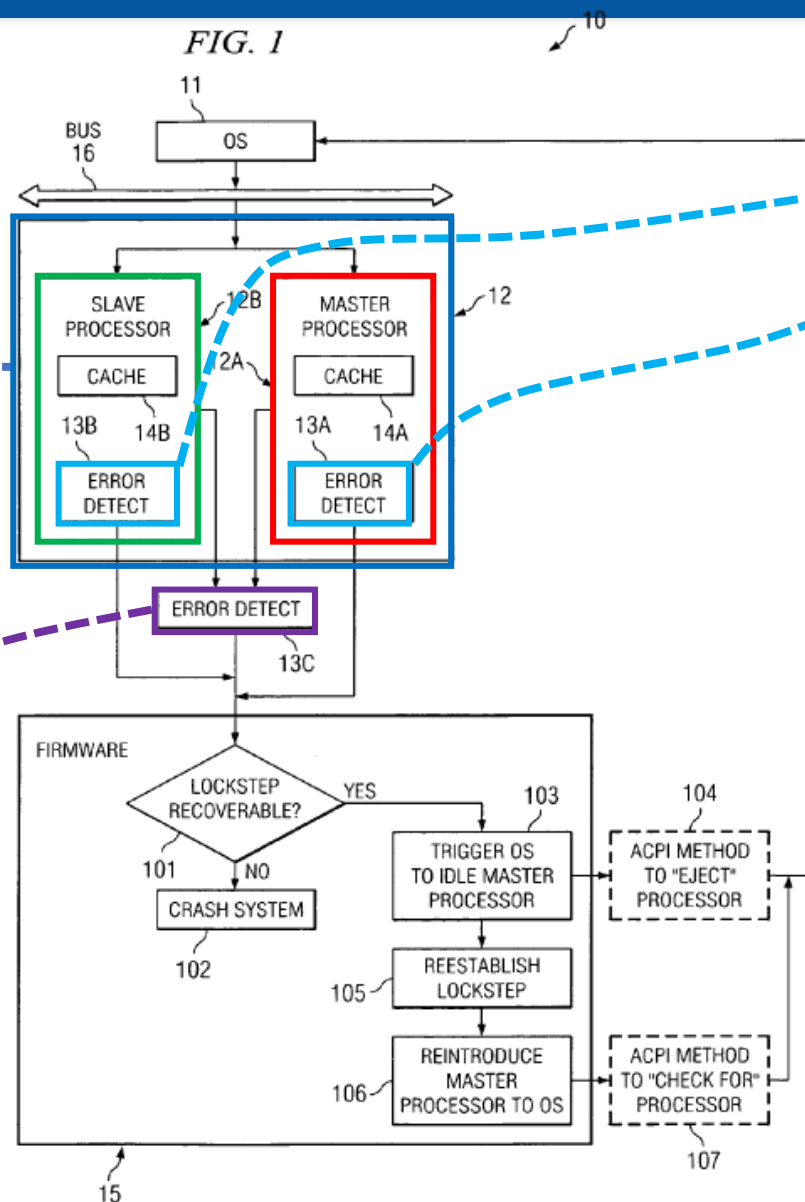
Lockstep processor pair includes a **slave processor** and a **master processor** receiving same instructions and performing same operations

Error detect logic compares the results of the lockstep processors and detects a discrepancy between the results ("lockstep mismatch")

Error detect logics in the processors detect errors in respective processors that may cause lockstep mismatch ("precursor to lockstep mismatch")

Using lockstep processors was a typical approach to detect processor errors and known in the art

Pet. 2-5; Ex. 1002, ¶¶ 42-46;
Ex. 1006 (Nguyen), [0005]-[0007], [0020]-[0024], [0030], [0046], FIG. 1;
Ex. 1008 (Safford), 2:55-3:9, FIG. 2.



Overview of the '958 Patent

Lockstep processor pair includes a **slave processor** and a **master processor** receiving same instructions and performing same operations

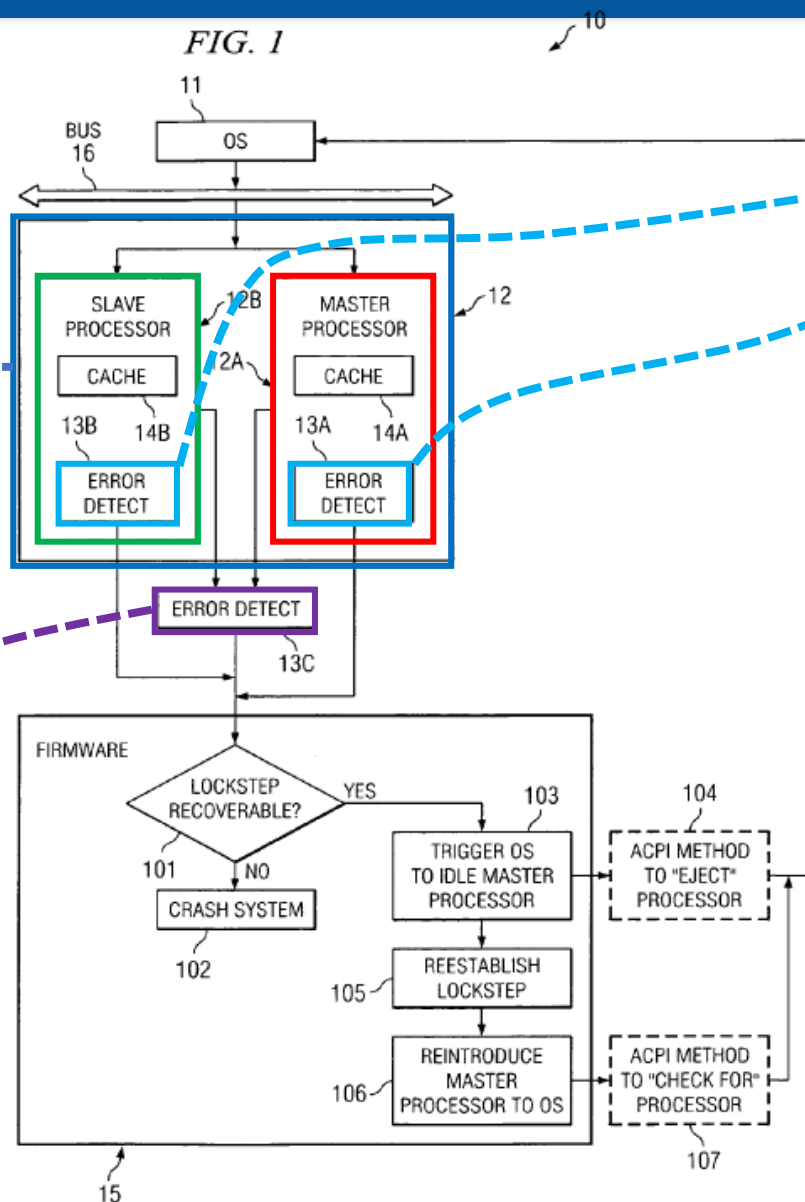
Error detect logic compares the results of the lockstep processors and detects a discrepancy between the results ("lockstep mismatch")

Error detect logics in the processors detect errors in respective processors that may cause lockstep mismatch ("precursor to lockstep mismatch")

Detection of "loss of lockstep" includes detecting two types of loss of lockstep:

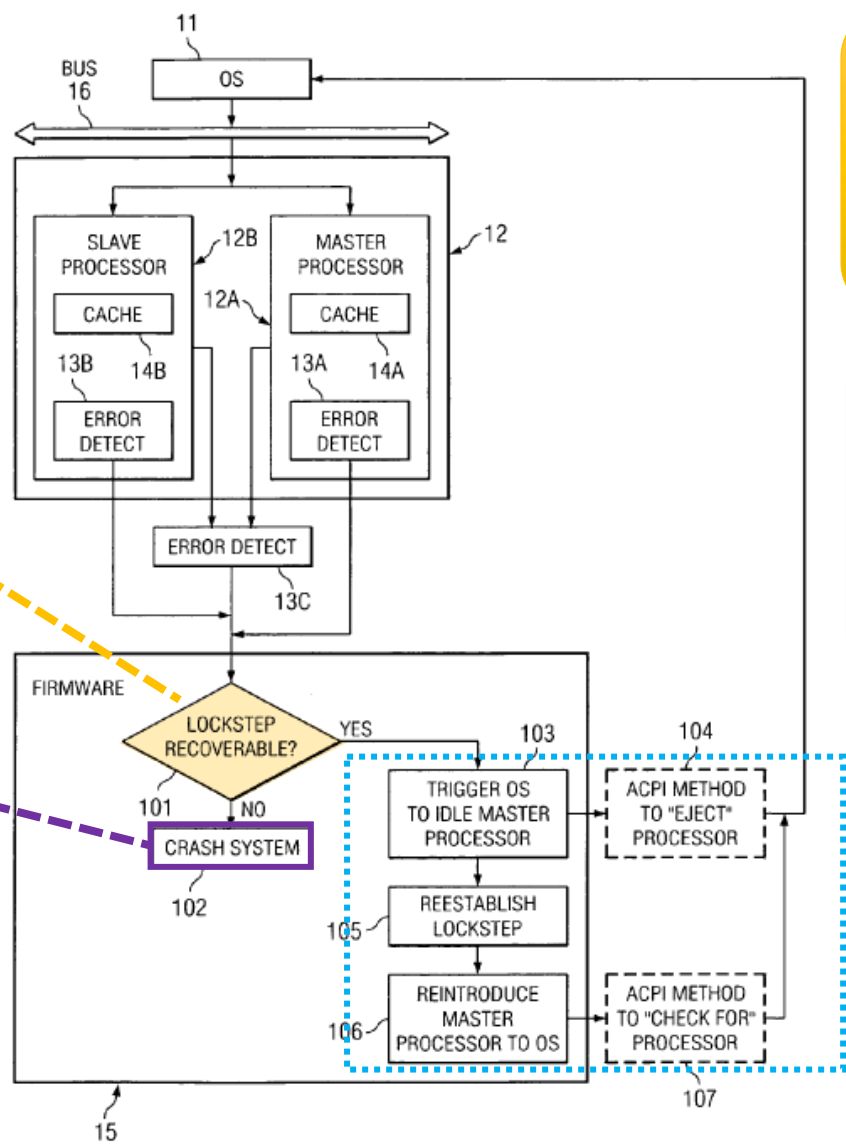
- (1) lockstep mismatch error;**
- (2) an error in one of the processors that may eventually cause a lockstep mismatch error**

Ex. 1001, 2:26-59, 4:56-67.



Overview of the '958 Patent

FIG. 1



The '958 patent describes using firmware to respond to the detected loss of lockstep to maintain high availability

system. That is, LOL in one pair of lockstep processors in a system halts processing of the system even if other processors that have not encountered an error are present in the system. However, with the increased desire for many systems to maintain high availability, crashing the system each time LOL is detected is not an attractive proposition. This is particularly

Ex. 1001, 3:5-16, 4:14-16, 5:49-62, FIG. 1.

Firmware determines whether lockstep is recoverable from the detected loss of lockstep

If firmware determines lockstep is not recoverable, it crashes the system

If firmware determines lockstep is recoverable, it cooperates with the OS to recover lockstep

Overview of the '958 Patent

The parties' disputes focus on common limitations of independent claims 1 and 19 that require using firmware for determining if lockstep is recoverable

Representative Claim

19. 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:

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, 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 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 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; *see also id.* at 15:7-19 (Claim 1).

The '958 Patent's independent claims recite six primary components, all of which were disclosed in the prior art and all of which would have been obvious to a POSITA at the time (Ex. 1002, ¶¶61):

- 1) Detecting loss of lockstep between a pair of lockstep processors;
- 2) With firmware, determining if lockstep is recoverable;
- 3) With firmware, determining if lockstep mismatch has occurred;
- 4) Triggering an operating system to idle the processors;
- 5) Attempting to recover lockstep; and
- 6) If recovered, triggering the operating system to recognize the processors to be available.

Pet. 2-16; Ex. 1002, ¶¶ 59-61.

Overview of the '958 Patent

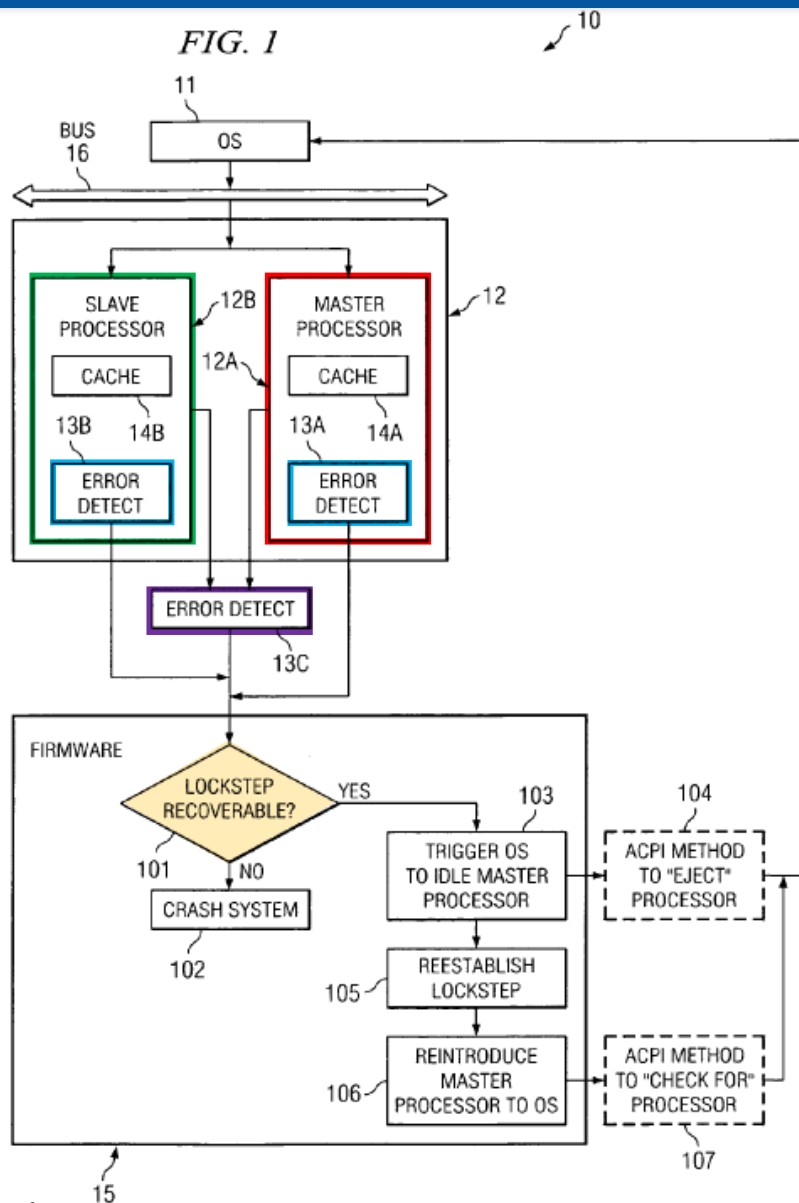
The parties' disputes do not apply to claim 13 and its dependent claims 14-18, which do not require using firmware for determining if lockstep is recoverable

13. A system comprising:
an Advanced Configuration and Power Interface (ACPI)-compatible operating system;
master processor operable to process instructions scheduled by said ACPI-compatible operating system;
slave processor lockstepped with the master processor operable to perform redundant processing of said instructions scheduled for the master processor; and
firmware operable, responsive to detection of loss of lockstep between the master and slave processors, to use an ACPI method to trigger the operating system to idle the master processor, attempt to recover lockstep between the master and slave processors, and if recovery of lockstep is successful then use an ACPI method to reintroduce the master processor to the operating system.

14. The system of claim **13** further comprising:
error detection logic for detecting loss of lockstep between the master and slave processors.
15. The system of claim **13** further comprising:
error detection logic for detecting a precursor to loss of lockstep.
16. The system of claim **15** wherein said precursor to loss of lockstep is treated as said detection of loss of lockstep.
17. The system of claim **15** wherein said error detection logic comprises a parity-based mechanism.
18. The system of claim **13** further comprising error detection logic for detecting lockstep mismatch between the master and slave processors.

Ex. 1001, 16:1-28 (Claims 13-18); Pet. 50-60, 67-69.

Overview of the '958 Patent



No dispute each claimed element was known in the art

The '958 Patent's independent claims recite six primary components, all of which were disclosed in the prior art and all of which would have been obvious to a POSITA at the time (Ex. 1002, ¶61):

- 1) Detecting loss of lockstep between a pair of lockstep processors;
- 2) **With firmware, determining if lockstep is recoverable;**
- 3) With firmware, determining if lockstep mismatch has occurred;
- 4) Triggering an operating system to idle the processors;
- 5) Attempting to recover lockstep; and
- 6) If recovered, triggering the operating system to recognize the processors to be available.

Known

Known

Known

Known

Known

Known

Ex. 1001, FIG. 1.

Pet. 2-14, 15-16; Ex. 1002, ¶¶ 42-61.

Claim Construction

Overview of the '958 Patent

Patent Owner does not dispute Petitioners' proposed interpretation of these claim terms

Claim Term	Petitioners	Patent Owner's Response	Institution Decision
"detection of loss of lockstep" and "detecting loss of lockstep"	"including at least one of (1) detecting a lockstep mismatch error; and (2) detecting an error that may eventually cause a lockstep mismatch error"	No dispute; no construction required	Not disputed; no construction necessary
"lockstep is recoverable" and "loss of lockstep is recoverable"	"lockstep is recoverable from the detected loss of lockstep"	No dispute; no construction required	Not disputed; no construction necessary

Pet. 16-19; POR 1-2; Institution Decision 9-10.

Ground of Unpatentability

Overview of the Ground

Exhibit	Reference	Pre-AIA Prior Art
Ex. 1005	US 2003/0126498A1 (“Bigbee”), filed January 2, 2002, published July 3, 2003	§ 102(b)
Ex. 1006	US 2004/0123201 A1 (“Nguyen”), filed December 19, 2002, published June 24, 2004	§§ 102(a), (e)

Claims 1-8 and 13-22 are unpatentable under the following grounds:

Ground	Claims	Reference(s)	Basis
1	1-8, 13-22	Bigbee in view of Nguyen	§ 103

Pet. 1; Institution Decision 6-7.

Concurrent Proceedings at the Patent Office

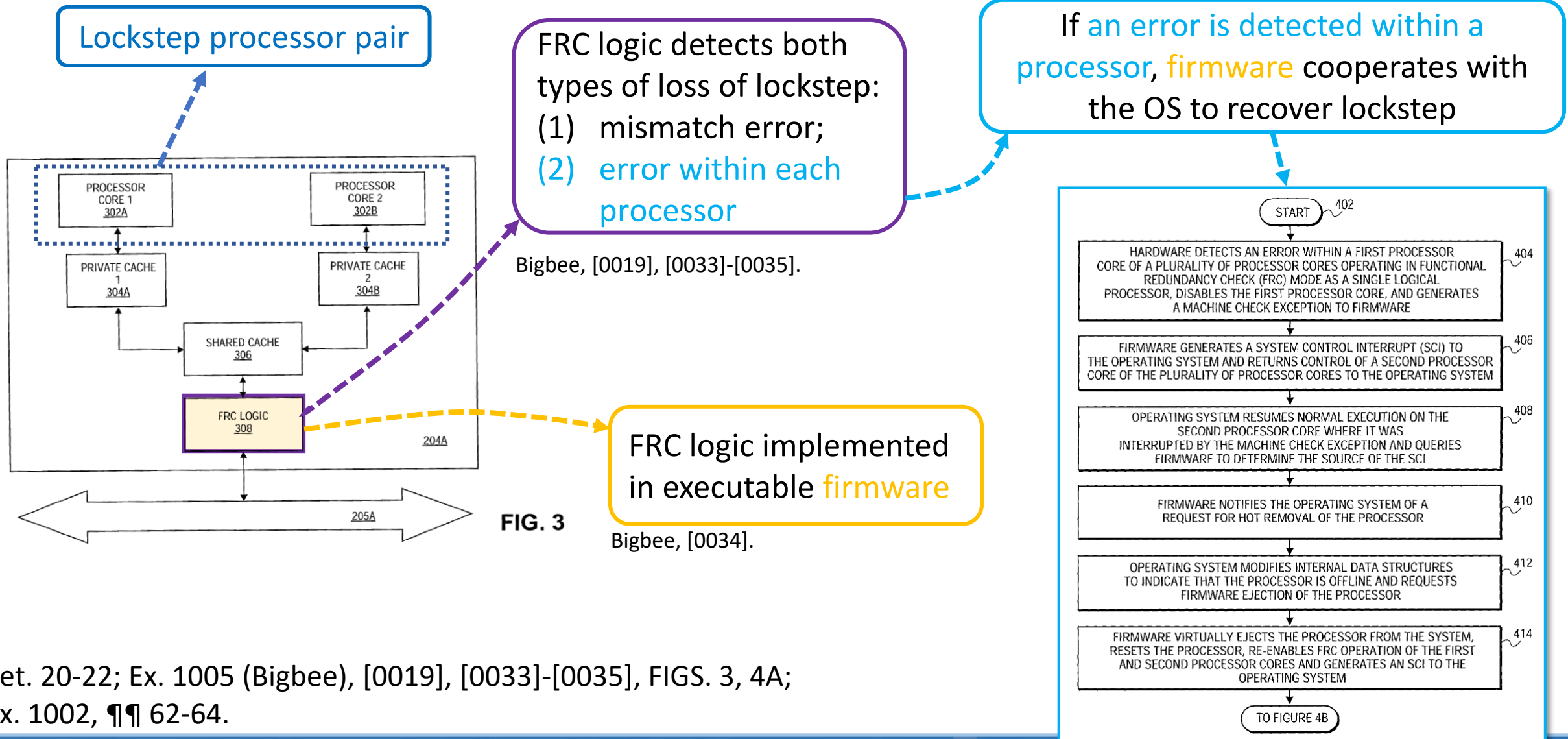
Case No.	Claims	Concurrent Proceeding	Status
Control No. 90/019,245	1-25	<i>Ex Parte</i> Reexamination granted on 11/29/2023	Board stayed the reexamination pending termination or completion of this IPR proceeding on 06/24/2024
IPR2024-00823	1-8, 13-22	IPR petition and joinder motion filed on 04/18/2024	Board instituted IPR and granted joinder with this IPR proceeding on 10/23/2024
IPR2024-00813	1-8, 13-22	IPR petition and joinder motion filed on 04/19/2024	Board granted pre-institution joint motion to terminate on 07/09/2024
IPR2024-00727	1-8, 13-22	IPR petition and joinder motion filed on 04/19/2024	Board granted pre-institution joint motion to terminate on 11/06/2024
IPR2024-00969	9-12, 23-25	IPR petition filed on 06/24/2024	Board granted pre-institution joint motion to terminate on 11/06/2024

Grounds of Rejection in Stayed *Ex Parte* Reexamination

Ground	Claims	Reference (s)
1	1, 4–8, 19–22	Bigbee
2	2–5, 13–18	Bigbee in view of Safford
3	1–8, 19–22	Safford in view of Bigbee
4	9–12	Safford as evidenced by Cepulis in view of Bigbee
5	13–18	Safford in view of Marisetty
6	23–25	Safford as evidenced by Cepulis in view of Marisetty

Control No. 90/019,245, Non-Final Office Action dated March 19, 2024.

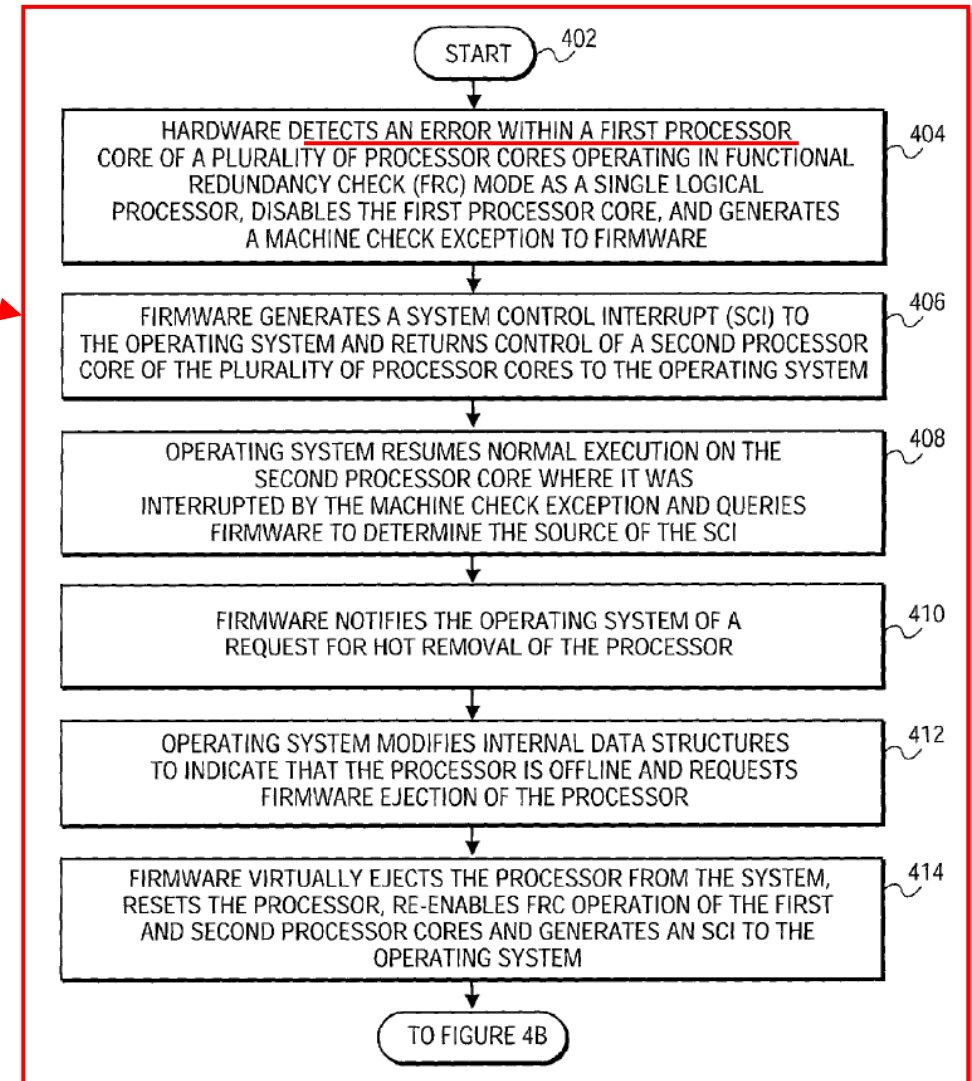
Bigbee discloses every claim element except determining if lockstep is recoverable



Pet. 20-22; Ex. 1005 (Bigbee), [0019], [0033]-[0035], FIGS. 3, 4A;
Ex. 1002, ¶¶ 62-64.

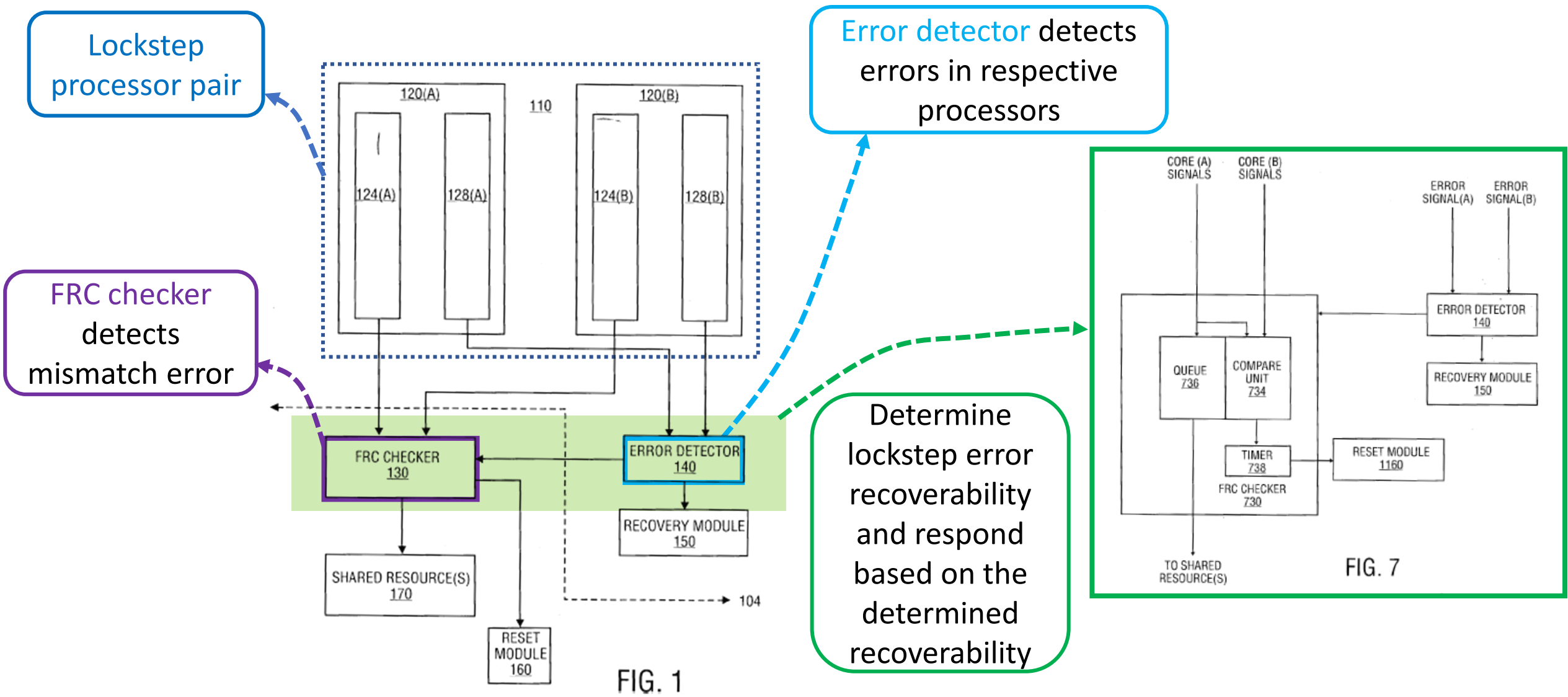
Bigbee only focuses on handling recoverable errors detected in the processors

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?



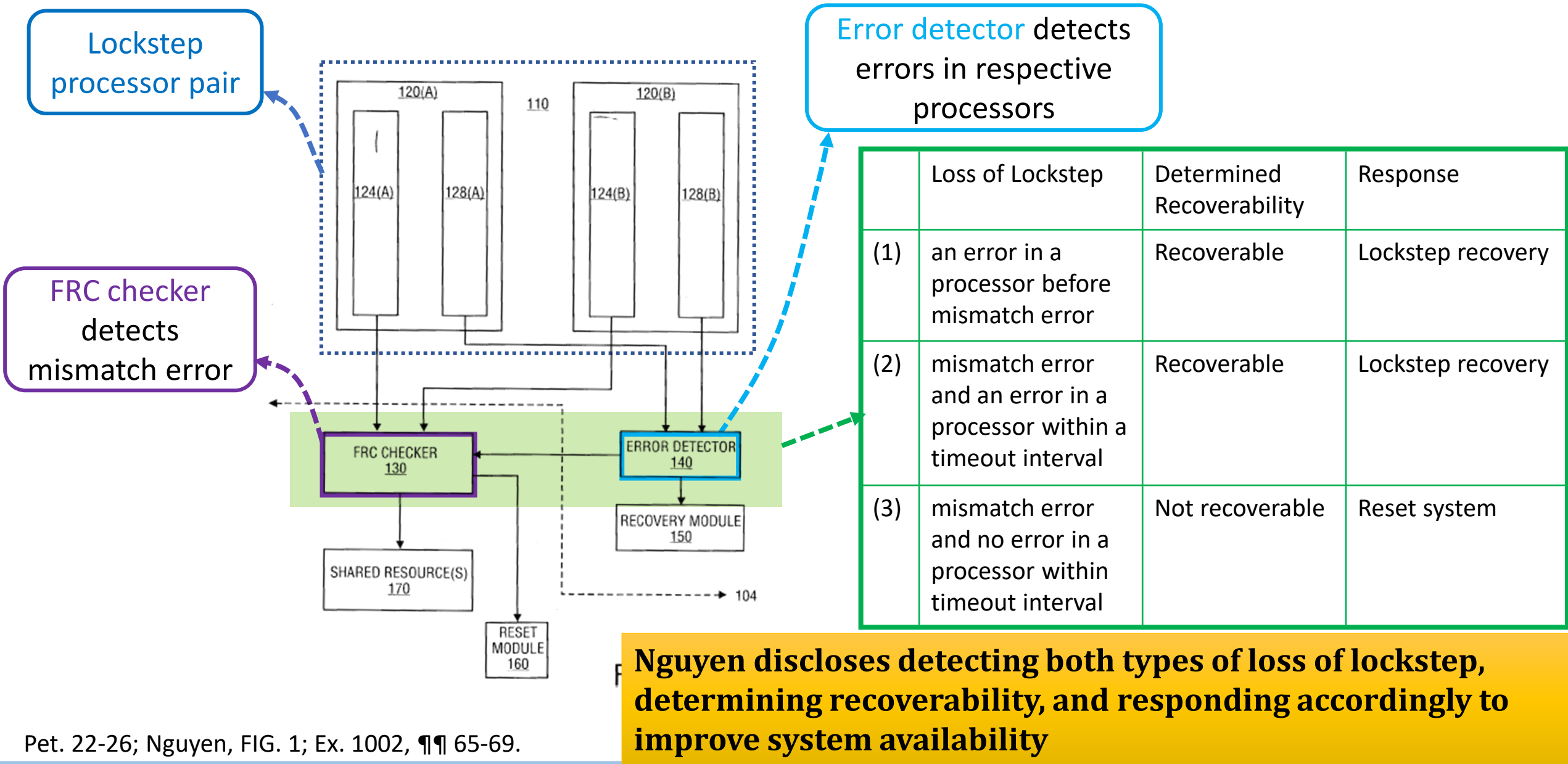
Pet. 20-22, 32-35; Bigbee, [0019], [0033]-[0037], FIG. 4A;
Ex. 1002, ¶¶ 62-64, 112-119.

Nguyen Discloses Determining Recoverability of Loss of Lockstep



Pet. 22-26; Ex. 1006 (Nguyen), FIGS. 1, 7, [0005], [0007]-[0008], [0022]-[0025], [0040]-[0041], [0046], [0056]-[0058]; Ex. 1002, ¶¶ 65-69.

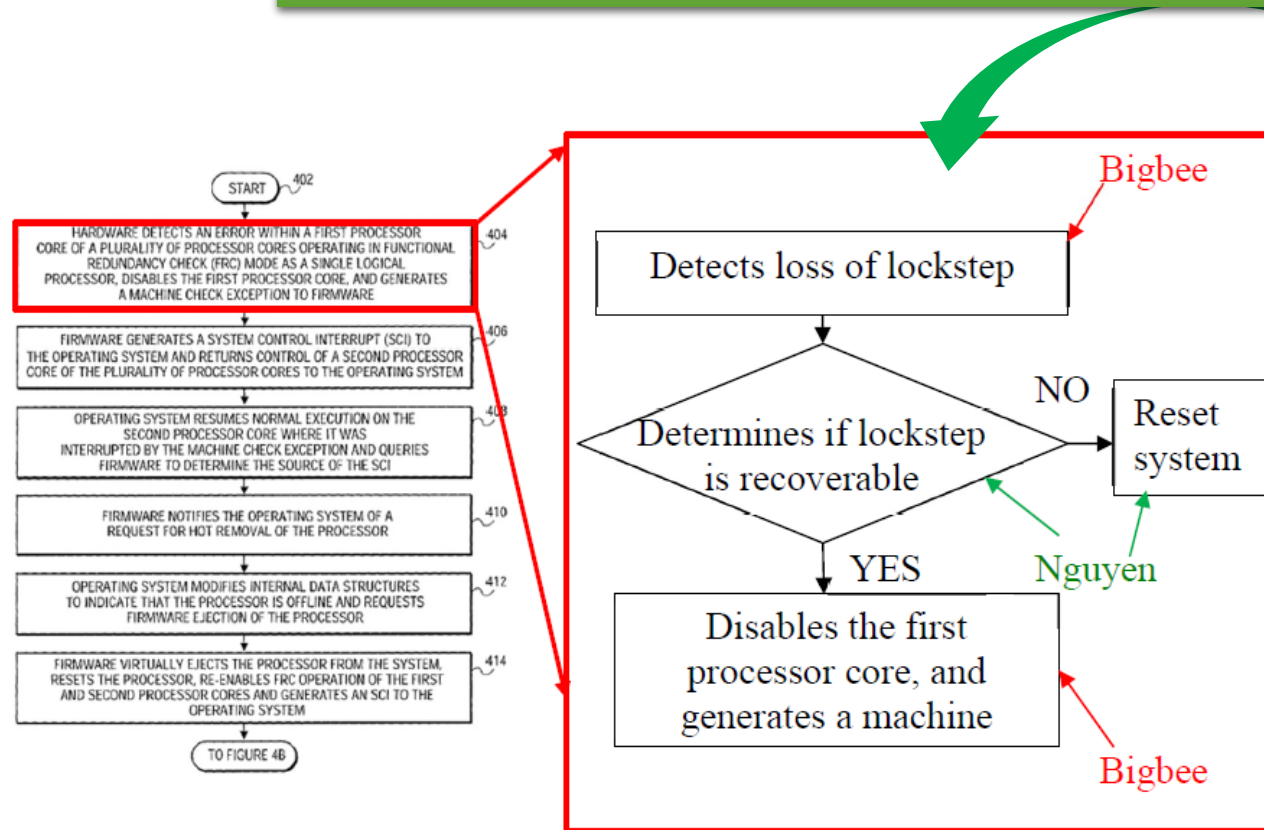
Nguyen Discloses Determining Recoverability of Loss of Lockstep



Pet. 22-26; Nguyen, FIG. 1; Ex. 1002, ¶¶ 65-69.

Bigbee in View of Nguyen

A POSITA would have incorporated Nguyen's threshold determining if lockstep is recoverable from the detected loss of lockstep to fill a gap in Bigbee



	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Pet. 27-29, 40-41; Ex. 1002, ¶¶ 66-69, 77-79; Bigbee, FIG. 4A; Nguyen, [0007], [0022]-[0024], [0056]-[0057].

Bigbee in View of Nguyen

Bigbee discloses detecting both recoverable and non-recoverable errors; it only discloses handling recoverable errors and omits the process after a non-recoverable error is detected

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?

recoverable or non-recoverable, and addressing the error accordingly. Bigbee’s recoverable error specific system would have been understood to be ready for the improvement of Nguyen’s determination and non-recoverable error responsive system to provide the known benefits of reducing impact of the detected errors, and increasing the system’s fault tolerance and availability by reducing system downtime.

Nguyen discloses one known solution to address both recoverable and non-recoverable errors to provide the known benefits of reducing impact of the detected errors and increase system availability

	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Pet. 30-35; Ex. 1002, ¶¶ 80-98; Nguyen, [0007]-[0008], [0022]-[0023], [0040].

The Parties' Limited Dispute

No Dispute

1. Petitioners' Statement of Material Facts Under 37 C.F.R. § 42.22(c)
2. Petitioners' expert Dr. Brogioli's summary of the state of the art
3. Petitioners' proposed claim construction
4. Petitioners' definition of level of ordinary skill in the art
5. Bigbee discloses every claim element except determining if lockstep is recoverable
6. Nguyen discloses determining if lockstep is recoverable
7. Petitioners' obviousness showing for claims 13-18 based on Bigbee and Nguyen
8. There are no pertinent secondary considerations

Two Disputed Issues

Patent Owner’s disputed issues involve only the second component of the claims

- 1. Whether Bigbee combined with Nguyen handles recoverable lockstep errors differently from non-recoverable lockstep errors
- 2. Whether a POSITA would have used Bigbee’s firmware for determining lockstep error recoverability

Pet. 27-29; Ex. 1002, ¶¶ 77-79; POR 12-17.

78. Applying the combination of Bigbee and Nguyen to the six core components of the independent claims, each limitation is expressly disclosed in the combination:

Core Claim Requirement	Prior Art Reference	
1) Detecting loss of lockstep between a pair of lockstep processors	Bigbee	No Dispute
2) With firmware, determining if lockstep is recoverable	Bigbee (with firmware) Nguyen (determining if lockstep is recoverable)	Disputed
3) With firmware, determining if lockstep mismatch has occurred	Bigbee	No Dispute
4) Triggering an operating system to idle the processors	Bigbee	No Dispute
5) Attempting to recover lockstep	Bigbee	No Dispute
6) If recovered, trigger the operating system to recognize the processors to be available	Bigbee	No Dispute

The Board Has Resolved the Two Disputed Issues

Dispute	Patent Owner's Response	Board's Correct Resolution
Whether Bigbee combined with Nguyen handles recoverable lockstep errors differently from non-recoverable lockstep errors	Bigbee combined with Nguyen would only handle recoverable errors detected in the respective processors in step 404 of Figure 4 of Bigbee	"Petitioner has sufficiently shown that the combination of Bigbee and Nguyen teaches the recited elements for purposes of institution, and has provided reasons sufficiently supported by factual underpinnings to combine the references with a reasonable expectation of success."
Whether a POSITA would have used Bigbee's firmware for determining lockstep error recoverability	Implementing Nguyen's FRC checker in firmware would add complexity, expense, be more difficult to assure reliable performance, and introduce more errors	

POR 12-17; Sur-reply 3-8; Institution Decision 32-33.

1. Whether Bigbee combined with Nguyen handles recoverable lockstep errors differently from non-recoverable lockstep errors?

Yes.

Patent Owner's Concessions

1. Patent Owner concedes that Nguyen determines lockstep recoverability from lockstep errors
2. Patent Owner concedes that Bigbee's error in an individual processor is akin to Nguyen's recoverable error
3. Patent Owner concedes to known reasons/benefits for distinguishing recoverable and non-recoverable lockstep errors as explained in the Petition and by Dr. Brogioli

POR 7-8, 13-14; Sur-reply 4, 6; Ex. 2007, ¶¶ 29-30.

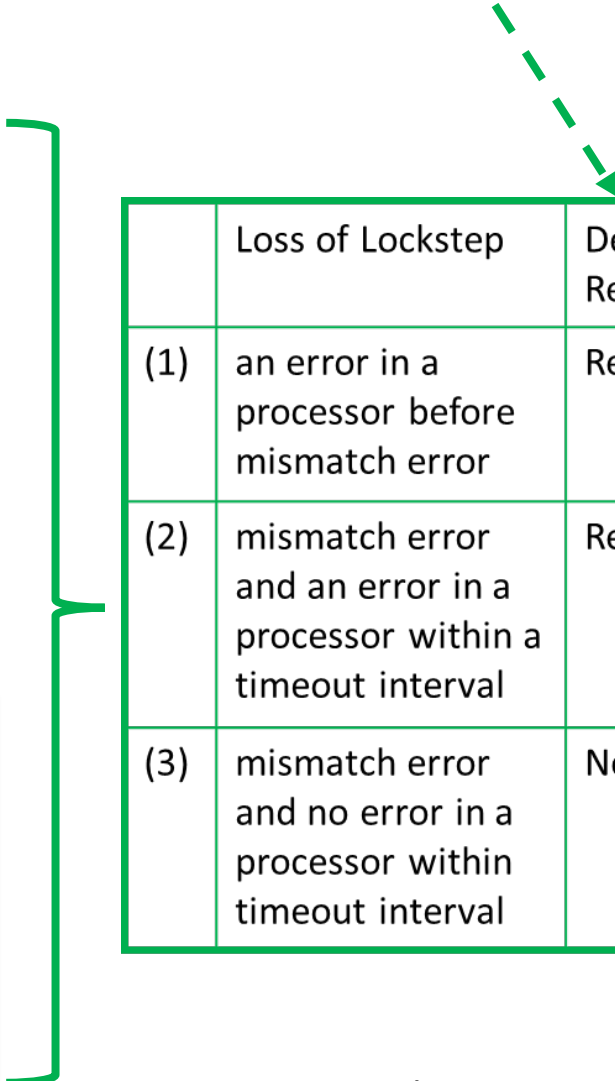
Nguyen Discloses Determining Recoverability of Loss of Lockstep

1. Patent Owner concedes that **Nguyen determines recoverability of lockstep errors**

Nguyen determines an 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 either before, or within a timeout interval after, a mismatch error is detected. EX1006 [0005], [0007], [0023]-[0024], [0030], [0046] ;

Pet. 40-41; EX1002 ¶ 106. Outside of these conditions (e.g., whenever an individual error is detected prior to detection of a mismatch error), **Nguyen will not determine that the error is non-recoverable.** *Id.* When

within an individual processor. **Nguyen is explicit that when an error is detected within an individual processor, that error is treated as recoverable.** There is simply no disclosure in Nguyen of any other outcome. **Nguyen only determines mismatch errors are non-recoverable when they are detected prior to detection of an individual processor error.**



	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

POR 7-8; Sur-reply 4.

Bigbee's Recoverable Error Is Akin to Nguyen's Recoverable Error

2. Patent Owner and Mr. Jones concede that Bigbee's error in an individual processor is akin to Nguyen's recoverable error

In Nguyen, Error Detector 140 receives signals from error pipeline 128, which represents the logic that detects errors in the data. EX1006 at [0021]; EX2007 ¶ 32. As Nguyen describes, “soft errors in these arrays may appear as parity or ECC error flags when the corrupted data is accessed.” *Id.* Bigbee similarly describes that errors in an individual processor can be “error correcting code (ECC) errors...or parity errors.” EX1005 [0018]. And Dr. Brogioli similarly explained that “a digital bit

POR 13.

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?

Bigbee's recoverable error is akin to Nguyen's recoverable error

14:18-15:10, 75:16-77:11. As Dr. Brogioli correctly notes, Bigbee's determination of loss of lockstep at step 404 in Fig. 4A is a determination of loss of lockstep from a recoverable error. Ex. 1002, ¶ 81 (citing Ex. 1005, [0019], [0035], [0037], “error recovery routine re-enables FRC operation of the processor cores”).

recoverable”) would be provided by Nguyen. This step, however, is completely unnecessary because Bigbee has already “determine[d] if lockstep is recoverable.”

Ex. 2007, ¶¶ 29-30.

	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Bigbee in View of Nguyen Provides Known Benefits

3. Patent Owner concedes to known reasons/benefits for distinguishing recoverable and non-recoverable lockstep errors as explained in the Petition and by Dr. Brogioli

B. Determining Whether Lockstep Is Recoverable Was Known

It was known that lockstep is recoverable from some errors in a lockstep processor, such as soft errors, but not recoverable from some other errors, such as some lockstep mismatch errors.

C. Handling Recoverable or Non-Recoverable Errors Differently Was Known

It was known that once an error was detected in FRC processors, the appropriate action responsive to the recoverability of the detected error should be taken to handle non-recoverable and recoverable errors to reduce impact of the errors on the system.

11. On or before October 25, 2004, it was known that treating a non-recoverable error as a recoverable error would reduce system reliability.

12. On or before October 25, 2004, it was known that treating a recoverable error as a non-recoverable error would reduce system availability.

Pet. 2-12, 16; *see also* Ex. 1002, ¶¶ 42-58.

Non-disputed State of the Art

detected error ... is recoverable by Bigbee's firmware FRC logic to appropriately handle the detected error." This is irrelevant. Patent Owner has not argued that a POSITA would have failed to recognize potential benefits of handling recoverable and non-recoverable errors differently. Patent Owner argues that the Bigbee-Nguyen "combined system" does not achieve those benefits because it cannot make any distinction between a recoverable and non-recoverable error.

Sur-reply 6.

Non-disputed benefits known to a POSITA

Non-disputed Statement of Material Facts

Bigbee in View of Nguyen Provides Known Benefits

Bigbee discloses detecting both recoverable and non-recoverable errors; it only discloses handling recoverable errors and omits the process after a non-recoverable error is detected

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?

recoverable or non-recoverable, and addressing the error accordingly. Bigbee's recoverable error specific system would have been understood to be ready for the improvement of Nguyen's determination and non-recoverable error responsive system to provide the known benefits of reducing impact of the detected errors, and increasing the system's fault tolerance and availability by reducing system downtime.

Nguyen discloses one known solution to address both recoverable and non-recoverable errors to provide the known benefits of reducing impact of the detected errors and increase system availability

	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Pet. 30-35; Ex. 1002, ¶¶ 80-98; Nguyen, [0007]-[0008], [0022]-[0023], [0040].

Patent Owner Mischaracterizes the Combination

1. Nguyen would only handle recoverable errors detected in the respective processors in step 404 of Figure 4A of Bigbee
2. “Bigbee has already ‘determine[d] if lockstep is recoverable.’”
3. Petitioners’ expert Dr. Brogioli’s testimony shows that the combination “begins with step 404 in Bigbee’s Figure 4A”

WRONG

Patent Owner Mischaracterizes the Combination

1. Patent Owner argues that Nguyen would only handle recoverable errors detected in the respective processors in step 404 of Figure 4A of Bigbee

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.” EX1005 Fig. 4A (block 404); EX2009 (Brogioli Dep.) 74:14-76:8; EX2007 ¶ 27. As explained below, this means that Nguyen will never determine that lockstep is non-recoverable. Every 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 ~~any~~ and individual processor errors. EX2007 ¶ 27-30.

WRONG

POR 12.

Patent Owner’s focus solely on Nguyen’s handling recoverable errors, despite Nguyen’s disclosure of handling both recoverable and non-recoverable errors, is rigid and impermissible

Reply 8-10.

Bigbee

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?

Nguyen

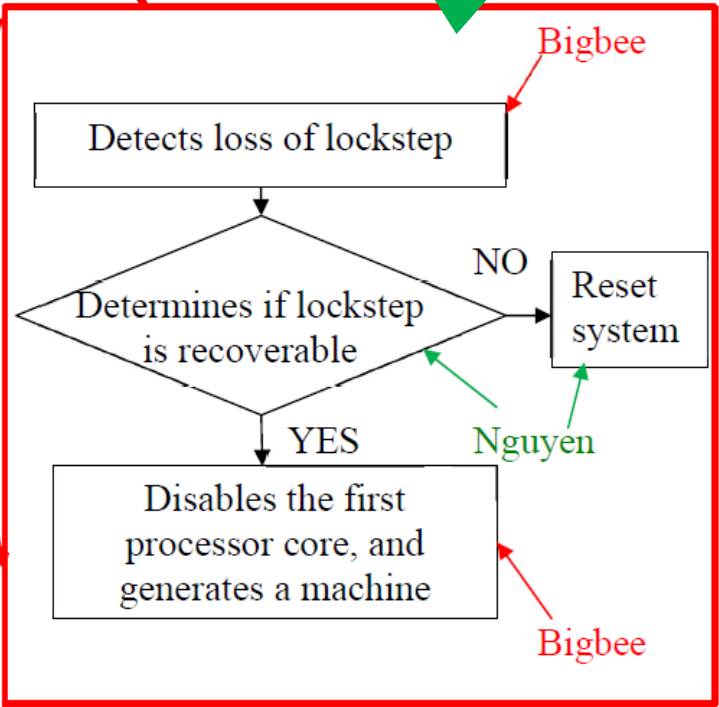
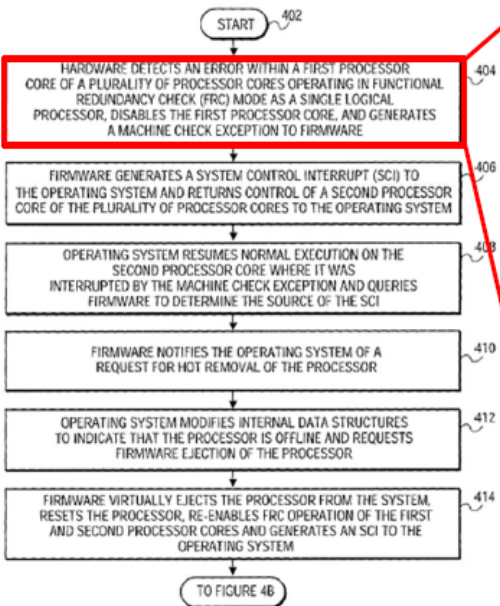
	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Ignored by Patent Owner

Bigbee in View of Nguyen

A POSITA would have incorporated Nguyen’s threshold determining if lockstep is recoverable from the detected loss of lockstep to fill a gap in Bigbee

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?



	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Pet. 27-29, 40-41; Ex. 1002, ¶¶ 66-69, 77-79; Bigbee, FIG. 4A; Nguyen, [0007], [0022]-[0024], [0056]-[0057].

Patent Owner Mischaracterizes the Combination

2. Mr. Jones: “Bigbee has already ‘determine[d] if lockstep is recoverable.’”

14:18-15:10, 75:16-77:11. As Dr. Brogioli correctly notes, Bigbee’s determination of loss of lockstep at step 404 in Fig. 4A is a determination of loss of lockstep from a recoverable error. Ex. 1002, ¶ 81 (citing Ex. 1005, [0019], [0035], [0037], “error recovery routine re-enables FRC operation of the processor cores”).

Ex. 2007, ¶ 29.

30. After this determination of loss of lock step from a recoverable error, Dr. Brogioli suggests that the next step to be taken (“Determines if lockstep is recoverable”) would be provided by Nguyen. This step, however, is completely unnecessary because Bigbee has already “determine[d] if lockstep is recoverable.”

Ex. 2007, ¶ 30.



Bigbee only focuses on handling recoverable errors detected in the processors

Reply 8-9; Pet. 27-29; Ex. 1002, ¶¶ 77-79.

Bigbee

	Loss of Lockstep	Response
(1)	Error within each processor	Lockstep recovery
(2)	Lockstep mismatch error	?

Ignored by Patent Owner

Patent Owner Mischaracterizes Dr. Brogioli's Opinion

3. Patent Owner Mischaracterizes Petitioners' Expert Dr. Brogioli's Opinion

These descriptions in the Petition, Dr. Brogioli's declaration, and Dr. Brogioli's testimony show that Petitioner's "combined system" allegedly follows the following process. It begins with step 404 in Bigbee's Figure 4A, which says "HARDWARE DETECTS AN ERROR WITHIN A FIRST PROCESSOR CORE OF A PLURALITY OF PROCESSOR CORES." Following this detection, which Petitioner annotates as "Detects loss of lockstep," the combined system performs a step from

POR 9, 11-12 (citing Ex. 2009 (Brogioli Dep.), 76:10-77:11)

Patent Owner argues that the combination only follows detecting an error within a processor

WRONG

cannot be determined) (Bigbee, [0033]). But it only discloses a responsive routine for recovering from recoverable errors. See above Section VII.A. Bigbee does not disclose handling non-recoverable errors. Because Bigbee only discloses recovering from recoverable lockstep errors, it does not disclose performing a threshold determination of whether detected errors are recoverable or non-recoverable. Bigbee, [0019], [0035], [0037] (error recovery routine re-enables FRC operation of the processor cores).

82. A person of skill in the art at the time would have recognized that Bigbee's disclosure omits the portion of the process after a non-recoverable error is detected (e.g., a mismatch error when the processor causing the error cannot be determined) and would have been motivated to add a process for handling non-recoverable errors. Nguyen is one such reference.

Ex. 1002, ¶¶ 81-82.

Dr. Brogioli explains that Bigbee detects non-recoverable errors but does not disclose addressing such errors

Patent Owner Mischaracterizes Dr. Brogioli's Opinion

Dr. Brogioli explained that Bigbee does not disclose performing the determination, which Nguyen discloses

cannot be determined) (Bigbee, [0033]). But it only discloses a responsive routine for recovering from recoverable errors. See above Section VII.A. Bigbee does not disclose handling non-recoverable errors. Because Bigbee only discloses recovering from recoverable lockstep errors, it does not disclose performing a threshold determination of whether detected errors are recoverable or non-recoverable. Bigbee, [0019], [0035], [0037] (error recovery routine re-enables FRC operation of the processor cores).

82. A person of skill in the art at the time would have recognized that Bigbee's disclosure omits the portion of the process after a non-recoverable error is detected (e.g., a mismatch error when the processor causing the error cannot be determined) and would have been motivated to add a process for handling non-recoverable errors. Nguyen is one such reference.

Ex. 1002, ¶¶ 81-82.

Dr. Brogioli explained in deposition that the combination is incorporating Nguyen's determination by Bigbee

1 A. So here in block 404 of Figure 4a of
2 Bigbee is where Bigbee's hardware detects an error
3 within the first processor core of the -- let's say
4 the two cores in that logical processor that were in
5 functional redundancy check mode. And this is the
6 step in which I incorporate the -- determining if
7 that lockstep is recoverable using the portion from
8 the Nguyen reference.

16 A. That I would say it's -- I'm
17 incorporating Nguyen's determination of whether
18 these Bigbee error -- errors detected by Bigbee are
19 recoverable or non-recoverable and to properly
20 handle the detected errors and the benefits that
21 come with them.

Ex. 2009 (Brogioli Dep.), 76:1-8, 78:16-21.

Patent Owner's Arguments for Claim 8 Similarly Fail

Claim 8

8. The method of claim 1 further comprising:
if determined that a lockstep mismatch has occurred
between the lockstep pair of processors, then determin-
ing that the loss of lockstep is not recoverable.

	Loss of Lockstep	Determined Recoverability	Response
(1)	an error in a processor before mismatch error	Recoverable	Lockstep recovery
(2)	mismatch error and an error in a processor within a timeout interval	Recoverable	Lockstep recovery
(3)	mismatch error and no error in a processor within timeout interval	Not recoverable	Reset system

Nguyen discloses determining lockstep is not recoverable if lockstep mismatch has occurred

Patent Owner concedes that Nguyen determines lockstep is not recoverable if lockstep mismatch has occurred

F. Nguyen (EX1006)
Petitioner relies on a second reference presented as Exhibit 1006 (“Nguyen”), which describes a scheme for processing lockstep errors. Nguyen determines an 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 either before, or within a timeout interval after, a mismatch error is detected. EX1006 [0005], [0007], [0023]-[0024], [0030], [0046] ; Pet. 40-41; EX1002 ¶ 106. Outside of these conditions (e.g., whenever an

POR 7-8.

outcome. Nguyen only determines mismatch errors are non-recoverable when they are detected prior to detection of an individual processor error.

Sur-reply 4.

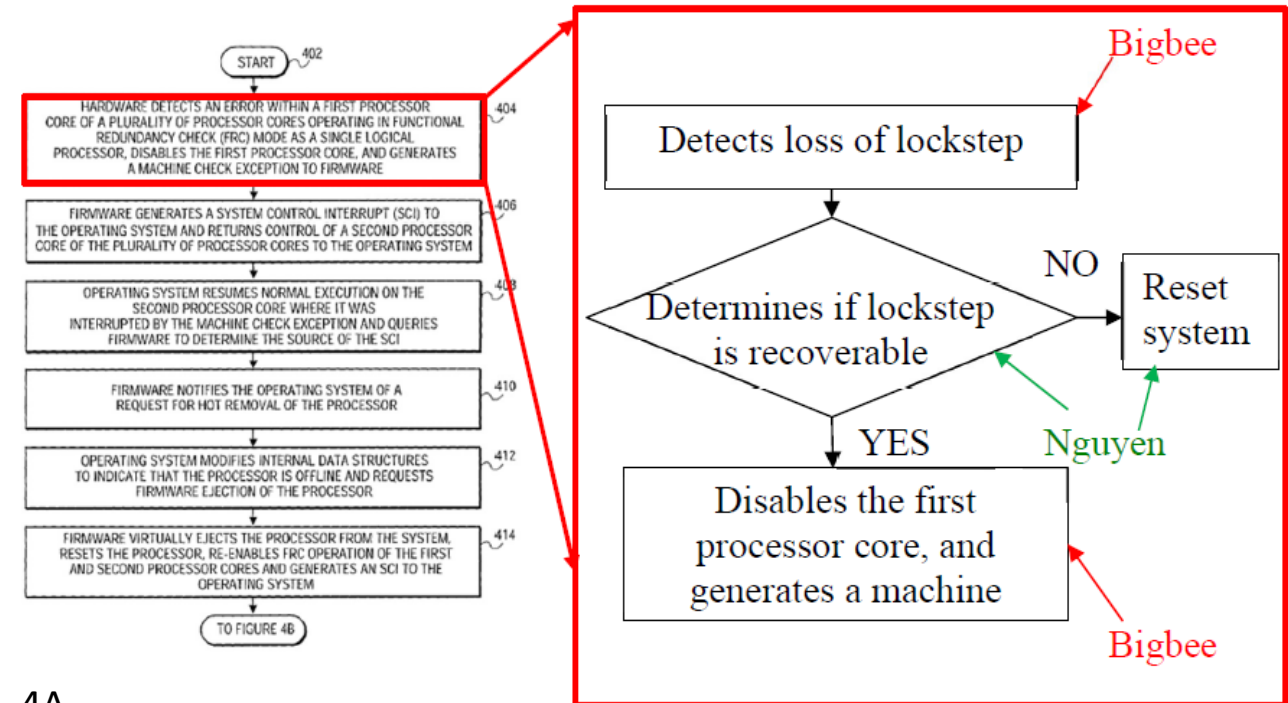
Patent Owner's Arguments for Claim 8 Similarly Fail

Patent Owner's unsupported attorney argument is not evidence and cannot rebut the reasoned and supported testimony of Petitioners' expert, Dr. Brogioli

In addition to the reasons provided above in §III.A, Petitioner has also failed to show that claim 8 is obvious in view of Bigbee and Nguyen. In particular, Petitioner has not shown that the Nguyen-Bigbee “combined system” ever makes a determination that “loss of lockstep is not recoverable.” As previously described, 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. EX1006 [0005], [0007], [0023]-[0024], [0030], [0046]; Pet. 40-41; EX1002 ¶ 106.

Patent Owner wrongly argues that the combination only begins with detecting an error within a processor

Dr. Brogioli: A POSITA would have incorporated Nguyen’s threshold determining if lockstep is recoverable from the detected loss of lockstep to fill a gap in Bigbee



Ex. 1001, 15:46-49 (Claim 8); POR 18; Ex. 1002, ¶¶ 81-86; Bigbee, FIG. 4A.

2. Whether a POSITA would have used Bigbee's firmware for determining lockstep error recoverability?

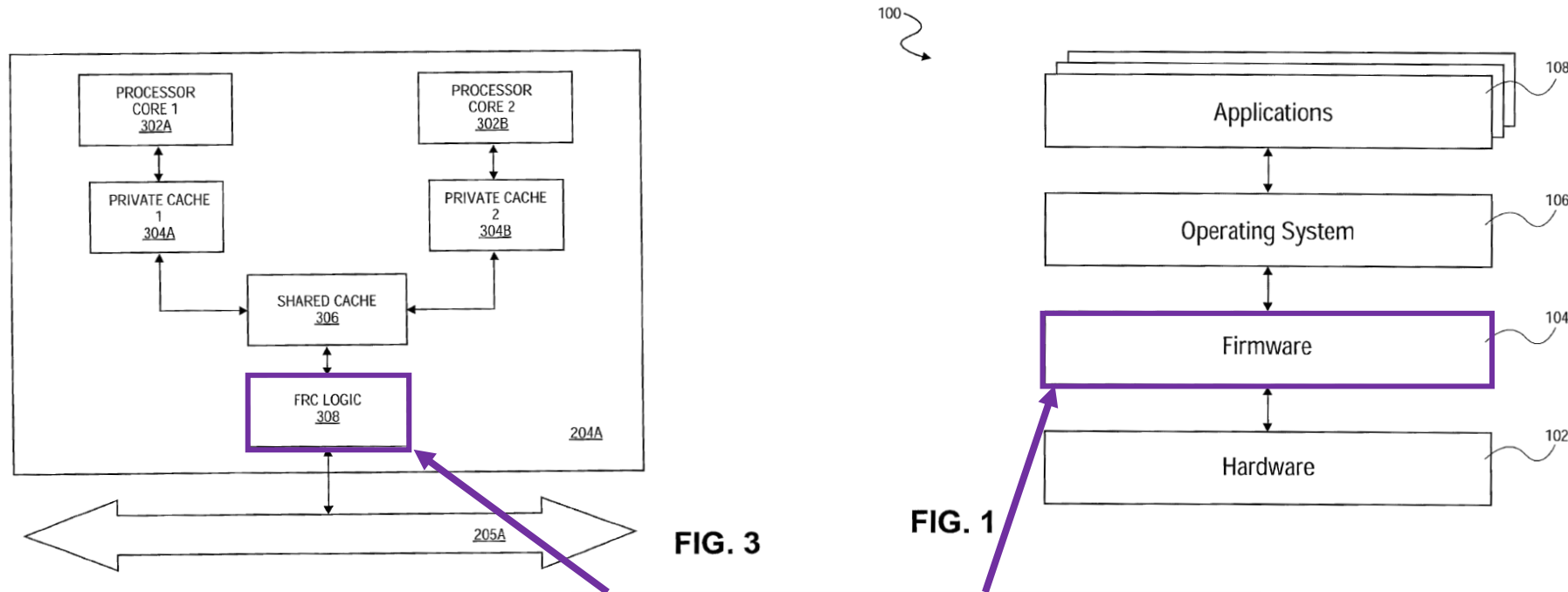
Yes.

No Dispute

1. Patent Owner does not dispute that Bigbee discloses using firmware to implement the FRC logic
2. Patent Owner does not dispute it was known to use firmware to detect processor error, determine detected error severity, and implement recovery routines
3. Patent Owner's expert, Mr. Jones, agrees with Petitioners' expert, Dr. Brogioli, on the benefits of using firmware

Bigbee Discloses Implementing the FRC Logic in Firmware

1. Patent Owner does not dispute that Bigbee discloses using firmware to implement the FRC logic



[0034] In alternative embodiments, FRC logic 308 may be implemented as executable firmware or dedicated hardware coupled with or incorporated into one or more of the processor package's 204a components. In yet another embodiment, a plurality of processor packages are provided, each including a single processor core with FRC logic being provided either within each processor package or external to each package to operate two or more of the package's associated processor cores in an FRC mode.

Bigbee, [0034], FIGS. 1, 3; Ex. 1002, ¶¶ 62-64, 94; *see also* Ex. 1009 (Safford-181), 15:60-64.

2. Patent Owner does not dispute it was known to use firmware to detect processor errors, determine detected error severity, and implement recovery routines

13. On or before October 25, 2004, firmware could be used for detecting lockstep errors.

14. On or before October 25, 2004, firmware could be used for handling processors errors.

Pet. 16 (Non-disputed Statements of Material Facts)

94. 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. Bigbee, [0034]. Implementing logical steps in firmware was known in the art. Bigbee, [0034], Safford-181, 15:60-64 (lockstep logic may be implemented in firmware)).

Ex. 1002, ¶¶ 57-58, 94-96; see Reply 11.

96. 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. Indeed, using such firmware, such as the PAL or SAL layer of ACPI BIOS, for implementing error diagnosis and recovery routines was well known and only involves ordinary skill and common knowledge. Shultz, [0027] (processor errors are classified into five different categories of increasing severity, including a firmware corrected category); [0154] (firmware SAL reports the appropriate error severity as being fatal or recoverable); Nguyen, [0045] (using PAL or SAL layers of BIOS for implementing error recovery mechanisms in Itanium[®] family processors); Marisetty, FIGS. 1-2, 5:48-8:64 (teaching using firmware including PAL and SAL to execute error handling routines); Schultz, FIGS. 1-3, [0017]-[0018] (teaching using firmware including PAL and SAL for correcting an error that is not correctable directly by hardware).

Benefits of Using Firmware Were Not Disputed

3. Patent Owner's expert, Mr. Jones, agrees with Petitioners' expert, Dr. Brogioli, that firmware is more flexible and inexpensive to update/modify

Petitioners' expert, Dr. Brogioli:

98. A person of ordinary skill in the art would have understood the benefits of implementing Nguyen's determination in firmware. As any person of ordinary skill in the art would have understood, firmware 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. Therefore, a firmware implementation of Nguyen's determining step would have been desirable.

Ex. 1002, ¶ 98.

Patent Owner's expert, Mr. Jones:

33. Dr. Brogioli also contends that a firmware implementation of Nguyen would be desirable because "firmware is flexible" and it "can be modified after production, updated, and supported relatively quickly and inexpensively." I agree with Dr. Brogioli that firmware is more flexible and more easily modified, but I do not agree that those qualities would be viewed by a person of skill in the art as desirable in the context of Nguyen's error detection hardware. To the contrary, a

Ex. 2007, ¶ 33.



Allied Erecting v. Genesis Attachments, 825 F.3d 1373, 1381 (Fed. Cir. 2016), quoting *Medichem, S.A. v. Rolabo, S.L.*, 437 F.3d 1157, 1165 (Fed Cir. 2006) ("[A] given course of action often has simultaneous advantages and disadvantages, and this does not necessarily obviate motivation to combine.").

Reply 13.

A Suitable Option Is Sufficient



Intel Corp. v. PACT XPP Schweiz AG, 61 F.4th 1373, 1381 (Fed. Cir. 2023) (a prior art teaching need not provide the best option if it provides a “suitable option”).

Reply 12.

The Board Has Resolved the Two Disputed Issues

Dispute	Patent Owner's Response	Board's Correct Resolution
Whether Bigbee combined with Nguyen handles recoverable lockstep errors differently from non-recoverable lockstep errors	Bigbee combined with Nguyen would only handle recoverable errors detected in the respective processors in step 404 of Figure 4 of Bigbee	"Petitioner has sufficiently shown that the combination of Bigbee and Nguyen teaches the recited elements for purposes of institution and has provided reasons sufficiently supported by factual underpinnings to combine the references with a reasonable expectation of success."
Whether a POSITA would have used Bigbee's firmware for determining lockstep error recoverability	Implementing Nguyen's FRC checker in firmware would add complexity, expense, be more difficult to assure reliable performance, and introduce more errors	

POR 14-16; Sur-reply 3-8; Institution Decision 32-33.

Expert Credential

Mr. Jones's Declaration Should Be Given No or Little Weight

Mr. Jones's Declaration Should Be Given No or Little Weight

1. Mr. Jones's testimony contradicts the prior art
2. Mr. Jones's testimony is unsupported
3. Mr. Jones's testimony is conclusory
4. Mr. Jones's testimony contradicts his own professional experience
5. Mr. Jones does not possess the undisputed minimum qualification of a POSITA

Ex. 2007, ¶¶ 32-33; Ex. 2008 (Mr. Jones's CV); Reply 2-5, 11-13; Sur-reply 10-13.

Mr. Jones's Testimony Should Be Given Little or No Weight

2. No supporting facts or data

impact of errors and increase system fault tolerance and availability. I disagree. A person of skill in the art would recognize that implementing Nguyen's "FRC checker's compare unit" in firmware code would be impractical for several reasons. First, detecting errors by comparing data from both cores using firmware would require a separate processor running approximately twenty times faster than the processor cores being monitored. This adds complexity and expense. Second, proving by formal methods that firmware configured to perform bit-by-bit error checking is operating correctly is much more difficult than proving the same for a hardware implementation. This means that it is more difficult to assure reliable performance in a firmware implementation compared to a hardware implementation. Third, firmware is more vulnerable than hardware to malicious attacks, which a person of skill in the art would consider as highly relevant in a system designed for reliability.

1. Contradicts explicit prior art disclosure

Bigbee, [0034];
Safford-181, 15:60-64.

3. Conclusory

Ex. 2007, ¶ 32; Reply 11-13.

Mr. Jones's Testimony Should Be Given Little or No Weight

is undesirable. A person of skill in the art would recognize that in a system designed to provide “high levels of reliability and availability,” (Ex. 1006 at [0004]) an implementation that can be easily modified is undesirable because it introduces more opportunities for intentional or unintentional errors. For the same reasons, I also disagree that a firmware implementation of Nguyen would have been obvious to try. Ex. 1002 at ¶ 97. A person of skill in the art would have instead understood the disadvantages of firmware in this context and would have been affirmatively dissuaded from using it. I therefore disagree that a POSITA

Ex. 2007, ¶ 33.

4. Mr. Jones's statements contradict his own work experience

- Design of a highly fault tolerant, distributed data acquisition and control, system for a rebreather. Responsibilities included system architecture, hardware design and layout, embedded firmware, documentation.

Ex. 2008 (Mr. Jones's CV), 1.

Mr. Jones's Testimony Should Be Given Little or No Weight

is undesirable. A person of skill in the art would recognize that in a system designed to provide “high levels of reliability and availability,” (Ex. 1006 at [0004]) an implementation that can be easily modified is undesirable because it introduces more opportunities for intentional or unintentional errors. For the same reasons, I also disagree that a firmware implementation of Nguyen would have been obvious to try. Ex. 1002 at ¶ 97. A person of skill in the art would have instead understood the disadvantages of firmware in this context and would have been affirmatively dissuaded from using it. I therefore disagree that a POSITA

Ex. 2007, ¶ 33.

Mr. Jones's statements defeat reality: no system designed for reliability would use firmware

1. Mr. Jones's statements contradict explicit disclosure in prior art

ON-DIE MECHANISM FOR HIGH-RELIABILITY PROCESSOR

BACKGROUND OF THE INVENTION

[0001] 1. Technical Field

[0002] The present invention relates to microprocessors and, in particular, to mechanisms for handling errors in FRC-enabled processors.

[0003] 2. Background Art

[0004] Servers and other high-end computing and communication systems are designed to provide high levels of reliability and availability. Soft errors pose a major chal-

Nguyen, [0001]-[0004].

[0014] The present invention provides a machine check architecture for error handling that is effective in maintaining system integrity by enabling coordination and cooperation between the processor 100, platform hardware 102, and the operating system software 110. The degree to which this error handling is effective in maintaining system integrity depends upon coordination and cooperation between the system CPUs, platform hardware fabric, and the system software. The machine check architecture provides error handling features for high reliability, availability, and serviceability. Error containment is the highest priority, fol-

Ex. 1016 (Schultz), [0014].

No Dispute on Definition of a Person of Ordinary Skill

Patent Owner does not dispute Petitioners' proposed definition of a POSITA, which the Board adopts at Institution

Petitioners' Definition:

II. Level of Ordinary Skill

A person of ordinary skill in the art ("POSITA") 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, as of the earliest priority date of the '958 Patent—October 25, 2004. Ex. 1002, ¶29. Additional education could substitute for professional experience and vice versa. *Id.*

Pet. 1; Ex. 1002, ¶ 29; Institution Decision 8-9.

Patent Owner's Definition:

C. Person of Ordinary Skill in the Art

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.

POR 3; Ex. 2007, ¶ 25.

Mr. Jones's Declaration Should Be Given No or Little Weight

5. Mr. Jones does not possess the required education and professional experience of a POSITA

**No required experience
of computer
architecture design**

**No required
education**

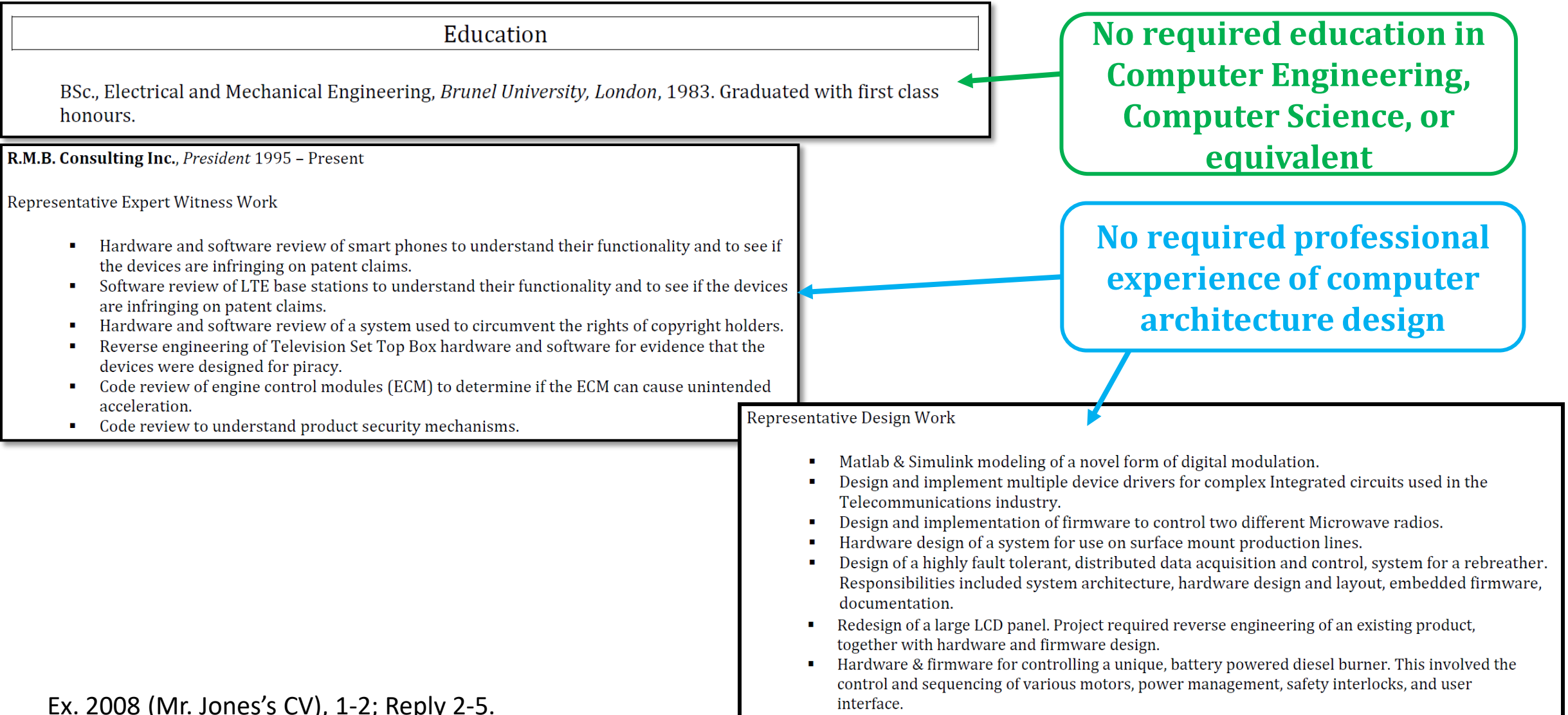
unsupported. As reflected in his declaration (EX2007) and his CV, Mr. Jones has a degree in Electrical and Mechanical Engineering, decades of relevant industry experience, and has been called on to testify as a technical expert in dozens of district court cases relating to computer architecture, computer hardware, computer software, and computer firmware.

**Incorrect –
no case
relates to
computer
architecture
design**

Sur-reply 10-13; see Reply 2-5; Ex. 2008 (Mr. Jones's CV).

Mr. Jones's Declaration Should Be Given No or Little Weight

5. Mr. Jones does not possess the required education and professional experience of a POSITA



Ex. 2008 (Mr. Jones's CV), 1-2; Reply 2-5.

Mr. Jones's Declaration Should Be Given No or Little Weight

5. Mr. Jones does not possess the required education and professional experience of a POSITA

- *Point 4 Data Corp. v. Tri-State Surgical Supply & Equip., Ltd.*, No. 11CV726CBARLM, 2019 WL 13277401, at *6 (E.D.N.Y. Mar. 27, 2019) (noting Mr. Jones' testimony as to **the design of a hardware dongle**);
- *Dish Network L.L.C. v. Whitcomb*, No. 11-CV-0333 WRBB, 2011 WL 1559825, at *3 (S.D. Cal. Apr. 25, 2011) (noting the Mr. Jones has "over 25 years of experience **designing and analyzing sophisticated electronic systems**");
- *Fraunhofer-Gesellschaft Zur Forderung Der Angewandten Forschung E.V. v. Sirius Xm Radio Inc.*, No. CV 17-184-JFB-SRF, 2017 WL 4564742, at *2 (D. Del. Oct. 13, 2017) (relying on Mr. Jones' testimony as to "source code" "VHDL" and their relationship to the "**design... [of] integrated circuits or chips**");
- *Burnett v. Ford Motor Co.*, No. 3:13-CV-14207, 2015 WL 3704599, at *2 (S.D.W. Va. June 12, 2015) (noting that Mr. Jones has "**designed both hardware and firmware**" and has performed "**reverse engineering of hardware and software, as well as code review**").

None of Patent Owner's highlighted testifying experience of Mr. Jones relates to computer architecture design

Sur-reply 12-13; see Reply 2-5; Ex. 2008 (Mr. Jones's CV).

Mr. Jones's Declaration Should Be Given No or Little Weight

5. Mr. Jones does not possess the required education and professional experience of a POSITA

No required experience
of computer
architecture design

No required
education

unsupported. As reflected in his declaration (EX2007) and his CV, Mr. Jones has a degree in Electrical and Mechanical Engineering, decades of relevant industry experience, and has been called on to testify as a technical expert in dozens of district court cases relating to computer architecture, computer hardware, computer software, and computer firmware.

Incorrect –
no case
relates to
computer
architecture
design



Kyocera Senco Indus. Tools Inc. v. Int'l Trade Comm'n, 22 F.4th 1369, 1376-77 (Fed. Cir. 2022) (“To offer expert testimony from the perspective of a skilled artisan in a patent case—like for claim construction, validity, or infringement—a witness **must** at least have ordinary skill in the art.”).

Nespresso USA, Inc. v. K-fee Sys. GmbH, IPR2023-00502, Paper 26, 8-14 (P.T.A.B. Sept. 10, 2024) (determining that Patent Owner's expert is not qualified to offer expert testimony from the perspective of a POSITA because the expert lacks the defined five years of professional experience).

Reply 2-5.

The Entire Record Shows the Challenged Claims Are Unpatentable

