

*BMW of North America, LLC and
Robert Bosch LLC,
v.
Foras Technologies Ltd.*

IPR2023-01373, U.S. Patent No. 7,502,958

Patent Owner's Demonstrative Exhibits

January 7, 2025

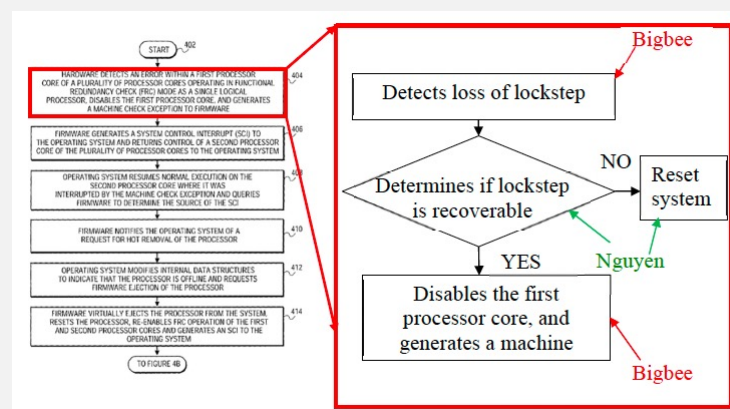
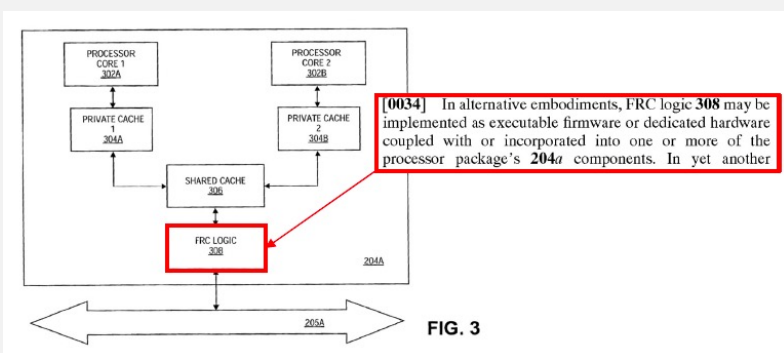
Ex. 2010

Bigbee does not disclose determining if lockstep is recoverable.

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.

Paper 18, 8; Paper 1, 19; Ex. 1002, Brogioli Dec., para. 76.

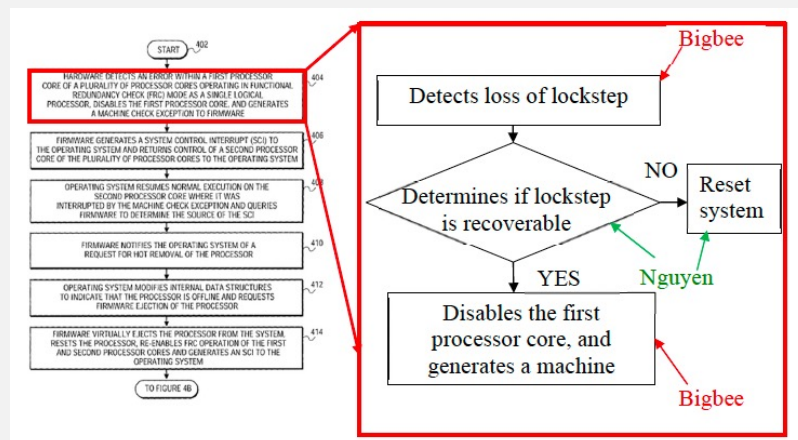
Bigbee + Nguyen does not disclose determining if lockstep is non-recoverable.



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").

Paper 18, 10-12; Paper 1, 29; Ex. 2007, Jones Dec., para. 29.

Bigbee + Nguyen does not disclose determining if lockstep is non-recoverable.

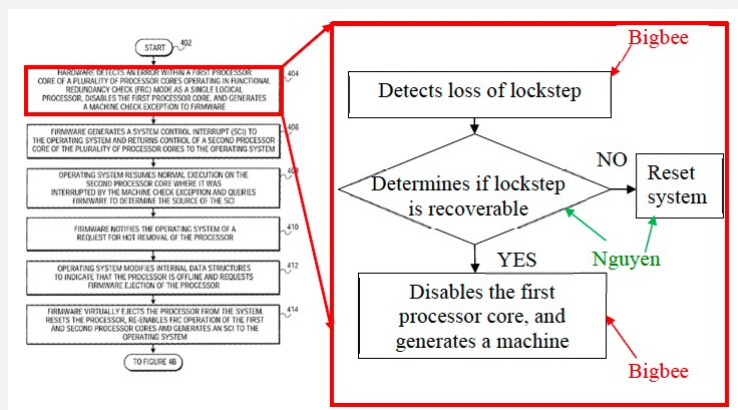


11 Q. So is it fair to say that you're
12 inserting Nguyen's determination of recoverability
13 in -- as part of this process in block 404 of
14 Bigbee?

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.

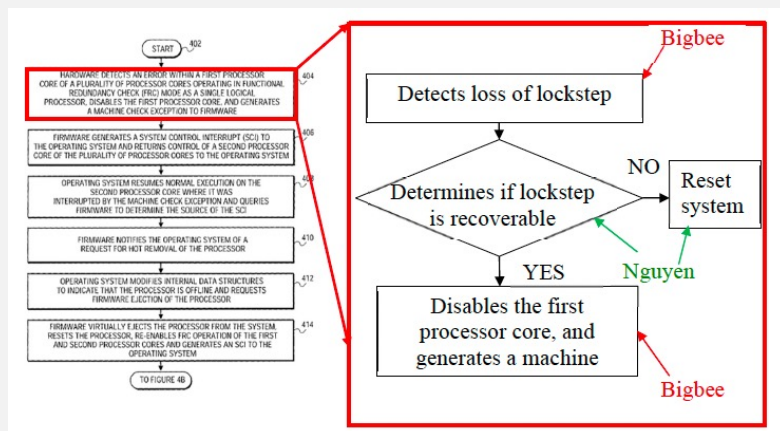
Paper 18, 12; Paper 1, 29; Ex. 2009, Brogioli Dep., 78.

Bigbee + Nguyen does not disclose determining if lockstep is non-recoverable.



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.” A POSITA would not be motivated to make this combination. Not only would a POSITA find the inclusion of Nguyen’s step of “determin[ing] if lockstep is recoverable” to be duplicative and unnecessary, but would also find that the inclusion would come with increased cost.

Bigbee + Nguyen does not disclose determining if 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 recoverable individual processor errors. EX2007 ¶ 29-30.

Nguyen’s alleged “determination” technique would not, as combined with Bigbee, distinguish recoverable from non-recoverable errors.

Paper 18, 12, 14; Ex. 2007, Jones Dec., para. 29-30.

Bigbee + Nguyen does not disclose determining if lockstep is not recoverable.

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.

The Bigbee-Nguyen combination does not render claim 8 obvious because it handles each error the same way and, since 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.

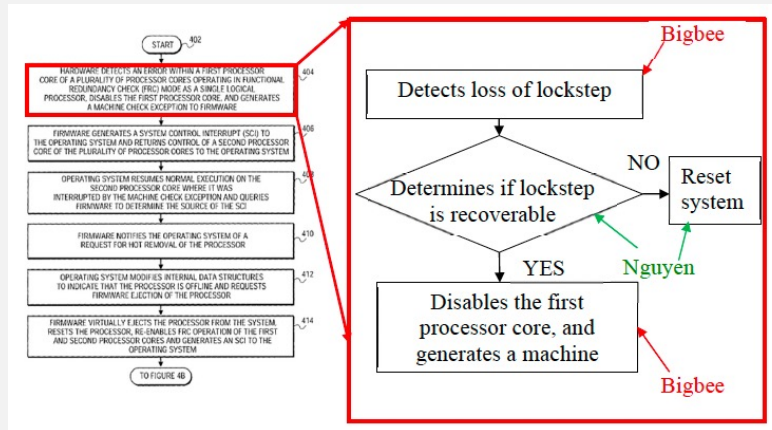
Paper 18, 18; Ex. 1001, '958 Patent, 15:46-49.

A POSITA would not implement Nguyen's recoverable lockstep determination in firmware

-the flexibility and ease of modification of firmware serve to introduce more opportunities for error, which 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.

Paper 18, 14-16; Ex. 2007, Jones Dec., para. 33.

A POSITA would not be motivated to combine Bigbee and Nguyen because the combination would not distinguish recoverable and non-recoverable errors.



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 recoverable individual processor errors. EX2007 ¶ 29-30.

Nguyen’s alleged “determination” technique would not, as combined with Bigbee, distinguish recoverable from non-recoverable errors.

Paper 18, 12, 14; Ex. 2007, Jones Dec., para. 29-30.

Nigel Jones Qualifies as a POSITA

9. I obtained a first-class honours degree in engineering from Brunel University, London U.K. I have forty years of experience designing complex electronic systems in a number of fields. I have extensive experience designing life support systems which are safety critical systems where errant behavior of a processor could lead to loss of life. I also have extensive experience in writing low-level, real-time firmware designed to interact with the hardware features of a microprocessor. I am a senior member of the IEEE and a member of the IEEE's Communications Society.

10. I have previously served as an engineering consultant and testifying witness on matters related to, and including telecommunication systems, cloud computing systems, and embedded computer system hardware and software. This experience is encompassed in my work as a system designer and architect, as well as my work as an engineering consultant and expert witness over the past 40 years.

Paper 25, 11; Ex. 2007, Jones Declaration, Para. 9-10.

Nigel Jones Qualifies as a POSITA

Mr. Jones has the equivalent of a bachelor's degree in Computer Science, Computer Engineering, or a related field.

Education
BSc., Electrical and Mechanical Engineering, <i>Brunel University, London</i> , 1983. Graduated with first class honours.

Paper 25, 11; Ex. 2008, Jones Resume, 2.

Nigel Jones Qualifies as a POSITA

Mr. Jones has extensive relevant industry experience.

Representative Design Work

- Matlab & Simulink modeling of a novel form of digital modulation.
- Design and implement multiple device drivers for complex Integrated circuits used in the Telecommunications industry.
- Design and implementation of firmware to control two different Microwave radios.
- Hardware design of a system for use on surface mount production lines.
- 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.
- Redesign of a large LCD panel. Project required reverse engineering of an existing product, together with hardware and firmware design.
- Hardware & firmware for controlling a unique, battery powered diesel burner. This involved the control and sequencing of various motors, power management, safety interlocks, and user interface.

Paper 25, 11; Ex. 2008, Jones Resume, 1-2.

Nigel Jones Qualifies as a POSITA

Mr. Jones published extensively on embedded programming techniques during the relevant time period.

Articles and Papers

Jones, Nigel. "Efficient C Code for 8-bit Microcontrollers," *Embedded Systems Programming*, Nov 1998.

Jones, Nigel. "Arrays of Pointers to Functions," *Embedded Systems Programming*, May 1999.

Jones, Nigel. "Controlling the Transmit Enable Line on RS-485 Transceivers," *Embedded Systems Programming*, Aug 1999.

Jones, Nigel. "In Praise of the #error Directive," *Embedded Systems Programming*, Sep 1999.

Stone, am Ende, Wefer, and Jones. "Automated 3D Mapping of Submarine Tunnels," *Robotics 2000: ASCE Conference on Robotics for Challenging Environments*, Feb 2000.

Jones, Nigel. "A 'C' Test: The 0x10 Best Questions for Would-be Embedded Programmers," *Embedded Systems Programming*, May 2000.

Jones, Nigel. "Support Multiple Languages," *Embedded Systems Programming*, Feb 2001.

Jones, Nigel. "Introduction to C's Volatile Keyword," *Embedded Systems Programming*, Jul 2001.

Jones, Nigel. "Introduction to Lint," *Embedded Systems Design*, May 2002.

Jones, Nigel. "MISRA C Guidelines," *Embedded Systems Design*, Jul 2002.

Jones, Nigel. "Introduction to MISRA-C," *Embedded Systems Design*, Jul 2002.

Jones, Nigel. "Optimal C Constructs for the 8051," *Embedded Systems Design*, Nov 2002.

Jones, Nigel. "Learn a New Trick with the offsetof() Macro," *Embedded Systems Design*, Mar 2004.

Jones, Nigel. "Minimizing ISR Overhead," *Embedded Systems Design*, Jan 2007.

Sieber, Jones, et. al. "Embedded Systems in the Poseidon MK6 Rebreather," *WISES 2009*, Jun 2009.

Jones, Nigel. "PO2 Sensor redundancy". Rebreather Forum 3 Proceedings pp. 193-202. ISBN #978-0-9800423-9-9

Paper 25, 11; Ex. 2008, Jones Resume, 2-3.