

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROCHIP TECHNOLOGY INC.,
Petitioner,

v.

HD SILICON SOLUTIONS LLC,
Patent Owner.

IPR2021-01042
Patent 7,302,619 B1

Before FRANCES L. IPPOLITO, ROBERT L. KINDER, and
ARTHUR M. PESLAK, *Administrative Patent Judges*.

KINDER, *Administrative Patent Judge*.

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

I. INTRODUCTION

Microchip Technology Inc. (“Petitioner”) filed a Petition (Paper 3, “Pet.”) requesting an *inter partes* review of claims 1–32 of U.S. Patent No. 7,302,619 B1 (Ex. 1001, “the ’619 patent”). HD Silicon Solutions LLC (“Patent Owner”) filed a Preliminary Response to the Petition (Paper 7, “Prelim. Resp.”).

After the Institution Decision (Paper 9, “Dec. on Inst.”), Patent Owner filed a Patent Owner Response (Paper 17, “PO Resp.”), Petitioner filed a Reply (Paper 19, “Reply”), and Patent Owner filed a Sur-reply (Paper 21, “Sur-reply”). Thereafter, the parties presented oral arguments on September 14, 2022, and the Board entered a transcript into the record. Paper 29 (“Tr.”).

For the reasons set forth in this Final Written Decision pursuant to 35 U.S.C. § 318(a), we determine that Petitioner demonstrated by a preponderance of evidence that claims 1–11, 13–27, and 29–32 of the ’619 patent are unpatentable. Petitioner has not demonstrated by a preponderance of evidence that claims 12 and 28 of the ’619 patent are unpatentable.

A. *Related Proceedings*

The parties indicate that the ’619 patent has been asserted in *HD Silicon Solutions LLC v. Microchip Technology Inc.*, No. 6:20-cv-01092 (W.D. Tex), filed November 30, 2020 (“District Court Litigation”). Pet. 1; Paper 5, 2. On October 25, 2021, the District Court Litigation was transferred to the Northern District of California from the Western District of Texas. Ex. 3001 (Order Granting Microchip Technology Inc.’s Motion to Transfer Venue).

On May 28, 2021, Petitioner filed a separate petition in IPR2021-00872 also challenging all claims of the '619 patent. *See* Paper 2, 1. On December 15, 2021, we denied institution of trial in IPR2021-00872.

B. The '619 Patent

The '619 patent, titled “Error Correction in a Cache Memory,” is directed to “systems and methods for error correction of instructions in an instruction cache coupled to a processor.” Ex. 1001, codes (54), (57). The application leading to the '619 patent was filed on July 6, 2004. *Id.* at code (22). The invention seeks to address the issue of “bit flipping due to alpha radiation,” which is a condition resulting from radiation striking a storage component such that “a logical ‘1’” may “flip into a logical ‘0’ or vice versa.” *Id.* at 1:12–20. “Such bit flipping may result in data inaccuracies and fatal errors in the operation of network traffic processors.” *Id.*

According to the '619 patent, “a plurality of instructions stored in the instruction cache are fetched for execution by the processor, each of the instructions being fetched during a respective one of a plurality of instruction cycles of the processor.” *Id.* at code (57). The Specification describes an exemplary processor circuit as shown in Figure 1, below.

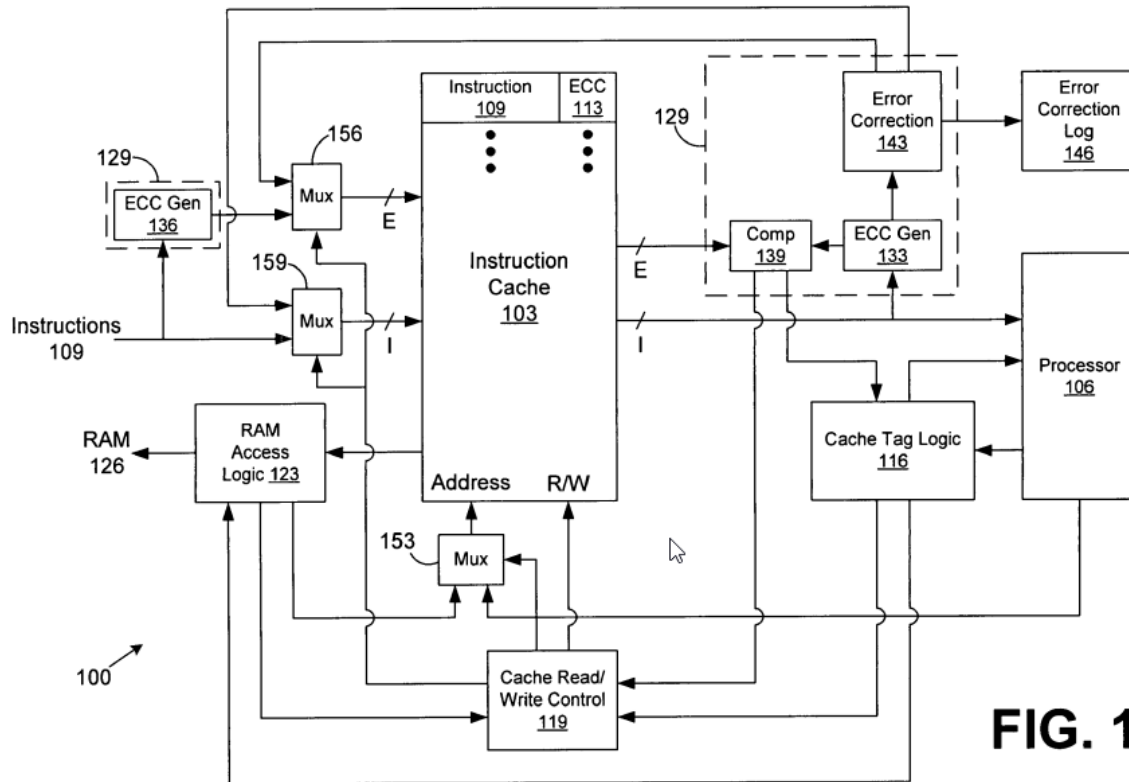


Figure 1 is a schematic that illustrates an instruction cache with error correction circuitry employed in a traffic signal processor. *Id.* at 1:30–32. Processor circuit 100 . . . includes instruction cache 103 and processor 106. *Id.* at 1:60–62. “The instruction cache 103 is a memory for the storage of instructions 109 that are accessed and executed by the processor 106.” *Id.* at 1:62–64. In relation to Figure 1 above, the ’619 patent describes “cache tag logic 116, cache read/write control logic 119, and the RAM access logic 123 facilitate the general operation of fetching instructions 109 from either the instruction cache 103 or from a random access memory 126 when such instructions 109 are requested by the processor 106.” *Id.* at 2:37–42.

The Specification describes how “processor circuit 100 also includes error correction circuitry 129 that is executed to correct any errors in instructions 109 and in the associated error correction codes 113 that can be corrected.” *Id.* at 2:43–46. Error correction circuitry 129 includes error

correction code generators 133 and 136, a comparator 139, and error correction logic 143. *Id.* at 2:49–51. According to the '619 patent, “the error correction circuitry 129 corrects single bit errors that occur due to bit flipping caused by alpha particles.” *Id.* at 2:46–48.

C. Illustrative Claim

Petitioner challenges claims 1–32 (“challenged claims”). Pet. 1. Claims 1, 21, and 29 are independent. Claim 1 is reproduced below.

1[pre]. A method for error correction in an instruction cache that is operatively coupled to a processor, comprising the steps of:

[1a] fetching a plurality of instructions stored in the instruction cache for execution by the processor,

[1b] each of the instructions being fetched during a respective one of a plurality of instruction cycles; and

[1c] performing error detection for each of the instructions concurrently with the fetching of a respective one of the instructions.

Ex. 1001, 10:39–48 (lettering identifiers added by Petitioner (Pet. 21–25)).

Claims 21 and 29 are both system claims and similar in claim scope. *Id.* at 12:8–16, 12:44–53. Claim 29 has a means plus function limitation.

D. The Evidence Relied on by Petitioner

Petitioner relies on the following evidence:

Reference	Exhibit
U.S. Patent No. 7,278,083 B2, filed on June 27, 2003 and issued on Oct. 2, 2007 (“Haswell”)	1006
U.S. Patent No. 7,395,489 B2, filed on Jan. 13, 2003, and issued July 1, 2008 (“Itou”)	1007
U.S. Patent No. 7,290,179 B2, filed on Dec. 1, 2003, and issued Oct. 30, 2007 (“Lempel”)	1008

E. The Asserted Grounds of Unpatentability

Petitioner asserts that claims 1–32 would have been unpatentable on the following grounds:

Basis	35 U.S.C. §¹	Claims Challenged
Haswell	102/103	1, 10, 11, 13, 21, 22, 24, 26, 29–31
Haswell, Itou	103	2–9, 14–20, 23–25, 27, 32
Haswell, Lempel	103	12, 28

See Pet. 4.

In support of its unpatentability arguments, Petitioner relies on the declaration testimony of Carl Sechen, Ph.D. Exs. 1002, 1020 (Reply Declaration). Patent Owner relies on the declaration testimony of Steven Przybylski, Ph.D. Ex. 2003 (the “Przybylski Declaration”).

II. LEGAL STANDARDS

A. Anticipation

To establish anticipation under 35 U.S.C. § 102, each and every element in a claim, arranged as recited in the claim, must be found in a single prior art reference. *Net MoneyIN, Inc. v. VeriSign, Inc.*, 545 F.3d 1359, 1369 (Fed. Cir. 2008); *Karsten Mfg. Corp. v. Cleveland Golf Co.*, 242 F.3d 1376, 1383 (Fed. Cir. 2001). Each element of the challenged claim must be found, either expressly or inherently, in the single prior art reference. *Verdegaal Bros., Inc. v. Union Oil Co. of Cal.*, 814 F.2d 628, 631

¹ The Leahy-Smith America Invents Act (“AIA”), Pub. L. No. 112-29, 125 Stat. 284, 285–88 (2011), revised 35 U.S.C. § 103 effective March 16, 2013. Because the challenged patent was filed before March 16, 2013, we refer to the pre-AIA version of § 103.

(Fed. Cir. 1987). Although the elements must be arranged or combined in the same way as in the claim, “the reference need not satisfy an *ipsissimis verbis* test,” i.e., identity of terminology is not required. *In re Gleave*, 560 F.3d 1331, 1334 (Fed. Cir. 2009); *In re Bond*, 910 F.2d 831, 832 (Fed. Cir. 1990). Thus, the dispositive question is whether one skilled in the art would have reasonably understood or inferred from a prior art reference that every claim element is disclosed in that reference. *Eli Lilly v. Los Angeles Biomedical Research Inst. at Harbor–UCLA Med. Ctr.*, 849 F.3d 1073, 1074–75 (Fed. Cir. 2017).

B. Obviousness

In *Graham v. John Deere Co.*, 383 U.S. 1 (1966), the Supreme Court set out a framework for assessing obviousness under § 103 that requires consideration of four factors: (1) the “level of ordinary skill in the pertinent art,” (2) the “scope and content of the prior art,” (3) the “differences between the prior art and the claims at issue,” and (4) “secondary considerations” of non-obviousness such as “commercial success, long-felt but unsolved needs, failure of others, etc.” *Id.* at 17–18. “While the sequence of these questions might be reordered in any particular case,” *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 407 (2007), the Federal Circuit has “repeatedly emphasized that an obviousness inquiry requires examination of all four *Graham* factors and that an obviousness determination can be made only after consideration of each factor.” *WPIP v. Kohler*, 829 F.3d 1317, 1328 (Fed. Cir. 2016) (“A determination of whether a patent claim is invalid as obvious under § 103 requires consideration of all four *Graham* factors, and it is error to reach a conclusion of obviousness until all those factors are considered.”).

We note that, with respect to the fourth *Graham* factor, the record in this proceeding does not include any argument or evidence directed to secondary considerations of nonobviousness.

III. LEVEL OF ORDINARY SKILL IN THE ART

In determining the level of skill in the art, we consider the “type of problems encountered in the art, the prior art solutions to those problems, the rapidity with which innovations are made, the sophistication of the technology, and the educational level of active workers in the field.”

Custom Accessories, Inc. v. Jeffrey-Allan Indus. Inc., 807 F.2d 955, 962 (Fed. Cir. 1986); *Orthopedic Equip. Co. v. United States*, 702 F.2d 1005, 1011 (Fed. Cir. 1983).

Petitioner contends that a person of ordinary skill in the art at the time of the invention of the '619 patent would have had the following education and experience:

at least (1) a master's degree in electrical engineering, computer science, or a related degree, and at least 3-5 years of experience in computer architecture and cache memory design; or (2) a bachelor's degree in electrical engineering, computer science, or a related degree, and at least 5-6 years of experience in computer architecture and cache memory design.

Pet. 5 (citing Ex. 1002 ¶ 27). Patent Owner does not dispute this level of skill. *See generally* PO Resp.

Based on the final record before us, including the prior art of record, we adopt Petitioner's proposal, which is also supported by Dr. Sechen's testimony. *See* Ex. 1002 ¶ 28.

IV. CLAIM CONSTRUCTION

Claims in an *inter partes* review are interpreted “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). Under that standard, the “words of a claim ‘are generally given their ordinary and customary meaning.’” *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (en banc) (internal citation omitted). “[T]he person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which [it] appears, but in the context of the entire patent, including the specification.” *Id.* at 1313. A “claim construction that excludes [a] preferred embodiment [described in the specification] is rarely, if ever, correct and would require highly persuasive evidentiary support.” *Adams Respiratory Therapeutics, Inc. v. Perrigo Co.*, 616 F.3d 1283, 1290 (Fed. Cir. 2010) (citation omitted). But “a claim construction must not import limitations from the specification into the claims.” *Douglas Dynamics, LLC v. Buyers Prods. Co.*, 717 F.3d 1336, 1342 (Fed. Cir. 2013).

Petitioner states that “[f]or this Petition, claim terms should be given their plain and ordinary meaning,” but Petitioner also recognizes that “[c]ertain claim terms are *means plus function* terms that should be interpreted under pre-AIA 35 U.S.C. § 112, ¶6 as indicated below.” Pet. 13. Petitioner provides interpretations of these proposed means plus function terms in its analysis. *See* Pet. 37–45. However, an important claim construction issue before us is the meaning of “instruction cache,” and we address this term first.

A. “instruction cache”

Each of claims 1, 21, and 29, specifically require a method or system for instruction error correction that uses “an instruction cache.” In its

Petition, Petitioner does not specifically advocate for an interpretation of “instruction cache.” In our Institution Decision, we noted the importance of the meaning of the term because “Patent Owner challenges two of the prior art references (Haswell and Itou) as not disclosing an instruction cache, as claimed, arguing instead that each reference uses a main storage memory or similar.” Dec. on Inst. 21–22. In the Institution Decision, we preliminarily found that the instruction cache is a memory for the storage of instructions that are accessed and executed by the processor. *Id.* (citing Ex. 1001, 1:62–64, code (57)). We also noted that lack of argument and evidence at that phase of the proceeding. *Id.* The parties responded by including additional argument, testimony, and evidence in the final record, which we now consider.

Patent Owner argues that an “instruction cache” is distinct from main memory. PO Resp. 31–32. Patent Owner first relies on the cross-examination testimony of Dr. Sechen (Exhibit 2005), Petitioner’s expert. *Id.* Dr. Sechen testified during questioning that the main memory is not an instruction cache in that “it wouldn’t satisfy the normal definition of an instruction cache, which has certain features.” *Id.* (quoting Ex. 2005, 61:20–62:7, 98:3–101:10). Patent Owner further argues that Dr. Sechen “also explained that the retrieval of data from an instruction cache is faster than from main memory due in part to its proximity to the processor.” *Id.* at 32 (quoting Ex. 2005, 166:8–18).

Patent Owner further relies on the description of an “instruction cache” found in the Specification of the ’619 patent, that “[t]he instruction cache 103 is a memory for the storage of instructions 109 that are accessed and executed by the processor 106.” *Id.* (quoting Ex. 1001, 1:62–64). Patent Owner also relies on the testimony of its own expert, Dr. Przybylski,

who testifies that “the retrieval of data from a cache is faster (and thus takes fewer instruction cycles) than retrieval from main memory,” and “[b]ecause an instruction cache only holds a subset of the contents of main memory at any given time, a cache requires tags, a tag storage, and hit/miss logic in order to autonomously determine whether the fetch of an instruction can be satisfied from the cache.” Ex. 2003 ¶¶ 65, 66; Ex. 1001, 3:19–21.

Based on this evidence and testimony, Patent Owner contends that “[t]he storage of both tags and instructions in an instruction cache is a defining characteristic of an instruction cache,” and “an instruction cache must store both tags and instructions in order to determine cache hits and misses.” PO Resp. 33 (quoting Ex. 2005, 45:9–12). “Accordingly, a POSITA [person of ordinary skill in the art] would have understood the term ‘instruction cache’ to mean a memory that includes (i) an instruction store for storing instructions and (ii) a tag store for storing tags that identify which instructions currently reside in the instruction store.” PO Resp. 34–35 (citing Ex. 2003 ¶ 69); *see also* Sur-reply 2 (“this meaning accurately reflects the basic fact that an ‘instruction cache’ must store both instructions and tags”).

Petitioner contends that a POSITA would have interpreted “instruction cache” according to its ordinary and customary meaning. Reply 1. Petitioner first notes that “[t]he ’619 patent and its file history do not define the term ‘instruction cache,’” and “by 2004 the use of cache memories was well-known and common.” *Id.* at 2. Dr. Sechen testifies that “a POSITA would have interpreted ‘instruction cache’ according to its ordinary and customary meaning,” because “the ’619 patent and its file history do not disclose or suggest any special or particular meaning of this term.” Ex. 1020 ¶ 8. Petitioner contends that the well-known meaning of an

“instruction cache” is a fast, small memory that is accessed before the main memory and that holds instructions. Reply 2.

Petitioner and Dr. Sechen rely on several textbooks to support Petitioner’s proposed ordinary meaning. *Id.* (citing Ex. 1020 ¶¶ 9–12). This evidence discloses that the term cache is (i) the name given to the first level of the memory hierarchy, (ii) that a cache memory is small but fast for maintaining the contents of the most commonly accessed locations, (iii) a cache is placed between the main memory and the CPU, and (iv) a cache may be the memory closest to the processor that is once again fast and relatively small. *Id.* at 2–3 (citing Exs. 1021, 1022, 1023).

Contrastingly, “Dr. Sechen explains that the textbooks do not mention tags or a tag store in defining the cache memory.” *Id.* at 4 (citing Ex. 1020 ¶ 13; Ex. 1021, 377; Ex. 1022, 266; Ex. 1023, 132; Ex. 1024, 294, 313–14). Thus, according to Petitioner, “POSITA would have interpreted the ‘instruction cache’ of the ’619 patent according to its plain and ordinary meaning, i.e., a fast and small memory (compared to the main memory) that is accessed before the main memory and that holds instructions.” *Id.* (citing Ex. 1020 ¶ 13).

Petitioner alleges that “PO’s construction is designed just to avoid the present prior art references because it is directed to a particular implementation of the instruction cache not actually disclosed in the ’619 patent.” Reply 4. Petitioner alleges that Patent Owner had to redraw Figure 1 of the ’619 patent, “[b]ut, the ’619 patent does not store tags in the ‘instruction cache 103.’” *Id.* at 4–5 (citing Ex. 1020 ¶ 15; Ex. 1025, 13:22–24, 15:8–11). Petitioner points out that “the ’619 patent does not explicitly disclose storing of tags anywhere.” *Id.* at 4 (citing Ex. 1020 ¶ 15). Instead, Petitioner relies on the ’619 patent disclosure that the “cache tag logic 116

maintains a list of all instructions 109 that are stored in the instruction cache 103,” which according to Petitioner “maintaining a list of instructions means storing tags in the cache tag logic 116.” *Id.* at 4–5 (quoting Ex. 1001, 3:21–23; citing Ex. 1025, 18:11–19:7). “Thus,” according to Petitioner, “if the tags are stored at all in the ’619 patent, they are stored in a tag store within the ‘cache tag logic 116,’ and not within the ‘instruction cache 103,’ as PO’s construction requires.” *Id.* at 5 (citing Ex. 1020 ¶ 16; Ex. 1025, 13:22–24, 15:8–11 (stating that “the tags are stored in cache tag logic 116 and not in instruction store 10”)). Petitioner further contends that “[p]erhaps recognizing this problem, Dr. Przybylski calls the ‘instruction cache 103’ an ‘instruction store,’” but “that term appears nowhere in the ’619 patent.” *Id.* (citing Ex. 1020 ¶ 16).

Petitioner next argues that Patent Owner’s expert, Dr. Przybylski, improperly redraws Figure 1 of the ’619 patent to conform to Patent Owner’s skewed claim construction. Reply 5. Specifically, Petitioner argues “to include the tag store in the instruction cache, Dr. Przybylski redraws FIG. 1,” such “that ‘[i]tem 103 is not an instruction cache,’ instead, ‘[i]t is an instruction store,’ and that the instruction cache includes ‘pretty much everything [in FIG. 1] with the exception of processor 106.’” *Id.* (citing Ex. 1020 ¶ 17; quoting Ex. 1025, 15:12–20, 15:21–16:22 (deposition of Dr. Przybylski)). According to Petitioner, “[t]his contradicts the ’619 patent, which consistently refers to the module 103 as the ‘instruction cache 103,’” and is thus “an attempt to avoid the present prior art.” *Id.* (citing Ex. 1020 ¶ 17; Ex. 1001, 1:60–64, 2:24–30, 2:33–42, 2:60–63, 3:6–9, 3:19–34, and 4:4–19, Fig. 1).

Petitioner further disagrees with Patent Owner’s proposed construction “because it selectively incorporates only one implementation

feature of a cache while ignoring others,” and “[s]pecifically, PO argues that ‘a defining feature of an instruction cache is the use of tags,’” yet, “tags along with additional features and parameters are used to describe cache implementations.” Reply 6 (citing Ex. 1020 ¶ 18)) (quoting PO Resp. 34; Ex. 2003 ¶ 67). Petitioner, relying on Dr. Sechen, notes that “the number of words in each cache block, type of mapping between the main memory and the case (which can be direct, associative, or set-associative), [and] the number of sets when set-associative mapping is used,” are further examples of features and parameters used to describe cache implementations. *Id.* (citing Ex. 1020 ¶ 18). Dr. Sechen testifies that these aspects of a cache require the specification of the block-offset and set-index addresses, in addition to the tags. Ex. 1020 ¶¶ 19–20 (citing Ex. 1021, 383; Ex. 1024, 316–21, Ex. 1022, 268–75). Petitioner concludes that “[t]hese features, including tags, do not define what a cache memory is – but rather how it is implemented or accessed.” Reply 7 (citing Ex. 1020 ¶ 21).

Patent Owner responds to several of Petitioner’s arguments in its Sur-reply. Patent Owner, relying on the testimony of Dr. Przybylski, contends “that the term ‘instruction cache’ means ‘a memory that includes an instruction store for storing instructions and a tag store for storing tags that identify which instructions are currently resident in the instruction store,”” and the “‘instruction cache’ must store both instructions and tags.” Sur-reply 3 (citing Ex. 2003 ¶ 69). Patent Owner alleges that Dr. Sechen’s previous testimony “agreed that an instruction cache necessarily stored tags.” *Id.* at 3–4 (quoting Ex. 2005, 22:21–23:8). Further, the four references (texts) referenced by Petitioner “describe caches that store tags in addition to data or instructions,” according to Patent Owner. *Id.* at 4–5 (quoting portions of Exs. 1021, 1022, 1023, 1024). Patent Owner notes that

one reference “expressly states that ‘*the tags are stored in a special tag memory*’ where they can be searched in parallel,” and another reference describes tags as being stored in tag memories. *Id.* at 6 (quoting Ex. 1022, 35). Thus, according to Patent Owner, “Dr. Sechen’s own references thus support the concept of distinct stores for tags and instructions, as encapsulated by Dr. Przybylski’s proposed meaning of ‘instruction cache.’” *Id.*

Next, addressing Dr. Sechen’s proposed definition of the term “instruction cache,” (a fast and small memory) Patent Owner argues it lacks evidentiary support because “[t]his meaning fails to encompass all memories that store tags,” and “it fails to clearly delineate whether any memory storing instructions would (or would not) be an instruction cache.” Sur-reply 7.

Based on the final record before us, we are more persuaded by Petitioner’s proposed claim interpretation of “instruction cache,” but we determine that neither definition is particularly useful in defining the scope of the term, primarily because neither definition is rooted in the intrinsic evidence of record. Both parties rely primarily upon extrinsic evidence to support their proposed interpretation. “[W]hile extrinsic evidence can shed useful light on the relevant art, [the Federal Circuit has] explained that it is less significant than the intrinsic record in determining the legally operative meaning of claim language.” *Phillips*, 415 F.3d at 1317 (internal quotations omitted). Although we do not necessarily disagree with certain aspects of either party’s extrinsic evidence and testimony – indeed an instruction cache would typically be smaller and faster than main memory (Petitioner) and may also have a tag and instruction store (Patent Owner) – each party’s proposed definition simply focuses on certain features of a cache that support their ultimate comparison to the prior art. For example,

we agree with Petitioner that Patent Owner's proposed construction draws on features and parameters used to describe cache implementations.

Based on the teachings of the '619 patent, we determine that the features highlighted by each party, however, should not be "the" singular definition of the term "instruction cache." Further, for the reasons set forth below, each party's proposed construction based on extrinsic evidence is not particularly helpful in our understanding of the term in light of the teachings of the '619 patent. *See id.* at 1318 ("each party will naturally choose the pieces of extrinsic evidence most favorable to its cause, leaving the court with the considerable task of filtering the useful extrinsic evidence from the fluff").

First, we consider Petitioner's proposed definition of "instruction cache" as "a fast and small memory (compared to the main memory) that is accessed before the main memory and that holds instructions." Ex. 1020 ¶ 13. Petitioner's proposed definition rests largely on text books and the testimony of Dr. Sechen. *See* Reply 2–4; Ex. 1020 ¶ 13. Although we agree with Petitioner that a POSITA would recognize these features as being present in an "instruction cache," (Reply 2–4), interpreting the claim term in a manner that requires measurements and comparisons to determine whether the memory was "fast" and "small" enough to qualify as an "instruction cache" does not provide a useful tool for our analysis and notice to the public of the metes and bounds of the invention. *See* Sur-reply 7 (highlighting issues with Petitioner's proposed comparison with main memory). Indeed, the underlying purpose of claim construction is to create a definition that is more useful for comparison to the prior art (or accused product) than just the claim language itself. In this instance, adopting Petitioner's proposed construction verbatim, which highlights certain

attributes of an “instruction cache,” would not be more useful than simply comparing the claim language itself.

Petitioner has established, however, that its proposed attributes of an “instruction cache” would be features that a person of ordinary skill in the art may consider when determining whether a memory may function and perform as an instruction cache. Indeed, Patent Owner agrees in part that “[c]aches are different from main memory because the retrieval of data from a cache is faster (and thus takes fewer instruction cycles) than retrieval from main memory.” PO Resp. 32. In that sense, we find Petitioner’s evidence useful in our analysis of Haswell’s memory below.

For different reasons, Patent Owner’s proposed construction is not helpful because it relies primarily on extrinsic evidence and focuses on features and parameters used to describe cache implementations, while ignoring other features. In light of the ’619 patent Specification, we disagree with limiting an “instruction cache” to just a memory that includes (i) an instruction store for storing instructions and (ii) a tag store for storing tags that identify which instructions currently reside in the instruction store. There is no dispute that an instruction cache may have these attributes. *See* Tr. 19:24–20:7 (PTAB: “So you’re saying it could have those things, but those are just features, and we shouldn’t necessarily limit the construction to those particular features? MR. DUTTA: That’s right, Your Honor.”). There is no indication based on the teachings of the ’619 patent, however, that every instance of the claimed instruction cache must have an instruction store for storing instructions and a tag store for storing tags that are both features within an instruction cache.

Patent Owner’s proposed construction would require that “an instruction store” and “a tag store” be located coextensive with the

instruction cache in every instance, but nothing in the '619 patent requires such a finding. For example, the '619 patent does not explicitly disclose storing of tags anywhere. Ex. 1020 ¶ 15. Also, as Dr. Sechen persuasively testifies, the '619 patent does not store tags in the “instruction cache 103.” Ex. 1020 ¶ 15. Patent Owner’s expert, Dr. Przybylski, seemingly agreed on cross examination that the '619 patent does not store tags in the “instruction cache 103.” Ex. 1025, 13:22–24, 15:8–11. As explained by Dr. Sechen, “the '619 patent states that the ‘cache tag logic 116 maintains a list of all instructions 109 that are stored in the instruction cache 103,” (Ex. 1001, 3:21–23), which means that tags are stored in the cache tag logic 116. Ex. 1020 ¶ 15; Ex. 1025, 18:11–19:7 (Dr. Przybylski testifying that “the cache tag logic 116” “maintains a list not of the instructions per se, but the addresses of the instructions that are stored in the instruction store 103. In other words, it’s storing the tags in order to determine whether a hit or a miss has occurred.”). Thus, according to the '619 patent, if a tag store exists, the tag store may not even be within the “instruction cache 103,” as Dr. Przybylski’s construction requires. Ex. 1025, 13:22–24, 15:8–11 (stating that “the tags are stored in cache tag logic 116 and not in instruction store 103”); Tr. 22:2–23 (Counsel for Petitioner arguing “[t]here is no language stating that the tags are stored within the instruction cache as the ECC [error correcting code] bits are or the instructions themselves.”).

As Petitioner argues, Dr. Przybylski attempts to overcome this issue by calling the “instruction cache 103” an “instruction store,” though that term appears nowhere in the '619 patent. Reply 5 (citing Ex. 1020 ¶ 16). Dr. Przybylski includes the tag store, which may be within the cache tag logic 116, in the instruction cache by redrawing Figure 1, testifying that “[i]tem 103 is not an instruction cache,” instead, “[i]t is an instruction store,”

and that the instruction cache includes “pretty much everything [in FIG. 1] with the exception of processor 106.” Ex. 1025, 15:12–20, 15:21–16:22. To us, Dr. Przybylski’s arguments seemingly ignore the plain language of the patent and its disclosure in Figure 1 setting out a distinct “Instruction Cache 103,” as seen below.

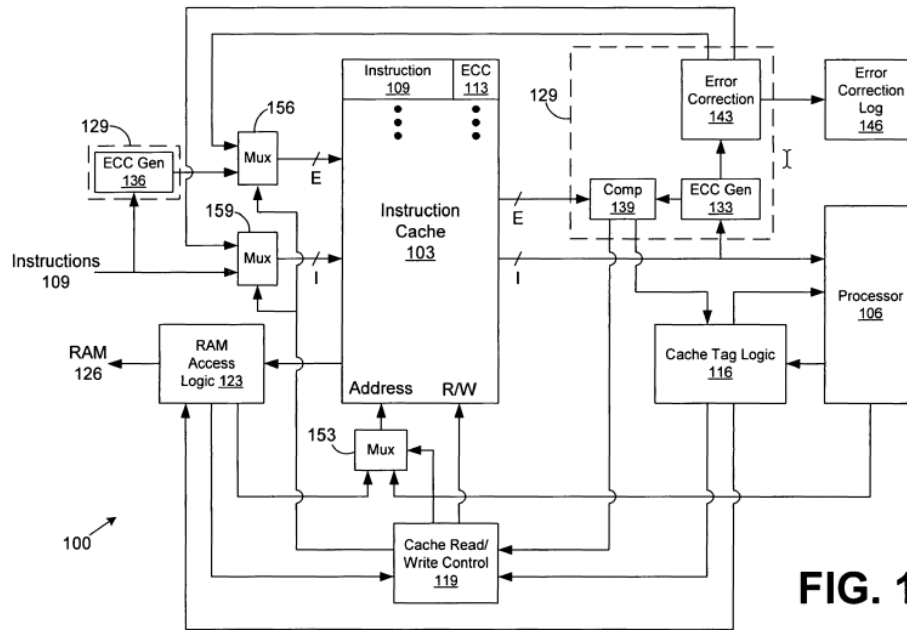


Figure 1 is a schematic that illustrates instruction cache 103 and processor 106, both separate from cache tag logic 116. Ex. 1001, 1:60–62. We agree with Petitioner that Patent Owner’s attempt to redraw the structure contained within the “instruction cache 103” “contradicts the ’619 patent, which consistently refers to the module 103 as the ‘instruction cache 103.’” Reply 5 (citing Ex. 1001, 1:60–64, 2:24–30, 2:33–42, 2:60–63, 3:6–9, 3:19–34, 4:4–19, Fig. 1); Tr. 44:13–24 (“The patent doesn’t specify that the cache tag logic 116 is inside instruction cache 103 in this specific -- there is no disclosure that the element illustrated in Figure 1 as 116 is inside the element

illustrated as 103.”). For the reasons set forth above, we determine that Dr. Przybylski’s construction is not supported by the ’619 patent.

In determining the scope of the term “instruction cache” we are more persuaded by the teachings of the ’619 patent. We maintain our initial interpretation set forth in the Institution Decision that the instruction cache is a memory for the storage of instructions that are accessed and executed by the processor. Dec. on Inst. 22; Ex. 1001, 1:62–64. Both parties cite, and rely upon, the portion of the Specification that states: “The instruction cache 103 is a memory for the storage of instructions 109 that are accessed and executed by the processor 106.” See Pet. 6; PO Resp. 32; Ex. 2003 ¶ 65. The Specification also explains that “the instruction cache 103 may be, for example, a Static Random Access Memory (SRAM) or other type of appropriate memory to the extent that such memory devices can provide the high speed access to the instructions 109 as required.” Ex. 1001, 2:26–30; see also *id.* at 2:37–42 (“cache tag logic 116, cache read/write control logic 119, and the RAM access logic 123 facilitate the general operation of fetching instructions 109 from either the instruction cache 103 or from a random access memory 126 when such instructions 109 are requested by the processor 106”). For us, these explanations from the Specification are sufficient to understand the meaning of “instruction cache.” Thus, based on the intrinsic evidence before us, an “instruction cache” means a memory for the storage of instructions that provides high speed access to the instructions to be executed by the processor.

B. “means for detecting an error”

Independent claim 29 recites “means for detecting an error in each of the instructions concurrently with the fetching of a respective one of the instructions.” Ex. 1001, 12:51–53.

Petitioner contends, and we agree, that “[t]his element is interpreted under pre-AIA 35 U.S.C. §112, ¶6 because it claims ‘means for’ and then recites the function of ‘detecting an error in each of the instructions concurrently with the fetching of a respective one of the instructions.’”

Pet. 38. Petitioner identifies the corresponding structure as the error correction circuitry 129. *Id.* (citing Ex. 1001, 7:53–55 (“error detection of the same instruction 109 is performed by the error correction circuitry 129 (FIG. 1)”). Patent Owner does not challenge this interpretation. We find Petitioner’s showing persuasive as to the interpretation of the “means for detecting an error” limitation.

C. *“means for generating a corrected instruction”*

Claim 30 recites “means for generating a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction.” Ex. 1001, 12:55–56.

Petitioner contends, and we agree, that this element “is interpreted under pre-AIA 35 U.S.C. §112, ¶6 because it claims ‘means for’ and then recites the function of “generating a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction.” Pet. 40–41. Petitioner identifies the corresponding structure as the error correction logic 143 or circuit 129. Pet. 41.

Petitioner relies on the ’619 patent’s disclosure of “when an error is detected ‘in either the instruction 109 or the error correction code 113, then the error correction logic 143 proceeds to generate a corrected version of the instruction 109 and its associated error correction code 113.’” *Id.* (citing Ex. 1001, 6:7–11; Ex. 1002 ¶ 134). Additionally, Petitioner notes that “the ’619 patent discloses that the error correction circuitry 129 (*e.g.*,

collectively) generates a corrected version of the instruction 109 and its corresponding error correction code 113. Pet. 41–42 (citing Ex. 1001, 10:1–3; Ex. 1002 ¶ 134) (footnote omitted). Thus, according to Petitioner, “a [person of ordinary skill in the art] POSITA would understand that either the error correction logic 129² or the error correction logic 143, which belongs to the error correction logic 129, generates the corrected instruction.” Pet. 42 (citing Ex. 1002 ¶ 134). Patent Owner does not challenge this interpretation.

We find Petitioner’s showing persuasive as to the interpretation of the “means for generating a corrected instruction” limitation.

D. “means for writing”

Independent claim 31 recites “means for writing the corrected instruction back to the instruction cache.” Ex. 1001, 12:59–60.

Petitioner contends, and we agree, that this element “is interpreted under pre-AIA 35 U.S.C. §112, ¶6 because it claims ‘means for’ and then recites the function of ‘writing the corrected instruction back to the instruction cache.’” Pet. 43. Petitioner identifies the corresponding structure as a multiplexer. *Id.*

Petitioner relies on the ’619 patent’s disclosure “that once the corrected version of the instruction is generated by the error correction logic 143, ‘the corrected version of the instruction 109 and its associated error correction code 113 are written back [from error correction logic 143] to the instruction cache 103 through the multiplexers 156 and 159 as shown.’” Pet. 43 (Ex. 1001, 6:7–19; Ex. 1002 ¶ 138). Petitioner additionally points

² The ’619 patent refers to element 129 as both “error correction logic” and “error correction circuitry.” Pet. 41, n. 2.

out where multiplexers 156 and 159, and error correction logic 143, are disclosed in Figure 1 of the '619 patent. Patent Owner does not challenge this interpretation.

We find Petitioner's showing persuasive as to the interpretation of the "means for writing" limitation.

E. "means for stalling the processor"

Independent claim 32 recites "means for stalling the processor during the generation of the corrected instruction and during the writing of the corrected instruction back to the instruction cache." Ex. 1001, 12:62–65.

Petitioner contends, and we agree, that this element "is interpreted under pre-AIA 35 U.S.C. §112, ¶6 because it claims 'means for' and then recites the function of 'stalling the processor during the generation of the corrected instruction and during the writing of the corrected instruction back to the instruction cache.'" Pet. 76. Petitioner identifies the corresponding structure as the cache tag logic. *Id.*

Petitioner relies on the '619 patent's disclosure "that 'the cache tag logic 116 causes the processor 106 to stall for a predefined number of instruction cycles.'" Pet. 77 (citing Ex. 1001, 5:49–53; Ex. 1002 ¶ 238). Patent Owner does not challenge this interpretation.

We find Petitioner's showing persuasive as to the interpretation of the "means for stalling the processor" limitation.

V. ANTICIPATION BASED ON HASWELL

Petitioner contends that the subject matter of claims 1, 10, 11, 13, 21, 22, 24, 26, and 29–31 of the '619 patent is anticipated by the disclosure of Haswell. Pet. 21–91. For the reasons that follow, and based on our review of the complete record before us, we are persuaded that Petitioner

establishes by a preponderance of the evidence that these claims are anticipated by Haswell.

A. Overview of Haswell

Haswell is titled “Method and System for Optimized Instruction Fetch to Protect Against Soft and Hard Errors,” and generally relates to “[a] method of detecting error during transfer of instructions from a data memory to a computer processor.” Ex. 1006, codes (54), (57). Haswell discloses a raw data signal being checked for an error detection code indicating data corruption and if the error detection code indicates data corruption, the raw data signal is substituted with a predetermined reserved signal or instruction and transmitted to the computer processor. *Id.* at code (57). Haswell explains that “[a]n attempt is made to correct the corrupted data in the raw data signal and, if it is corrected, the corrected data signal is subsequently retrieved and the corrected data signal is processed by the computer processor.” *Id.*

Haswell's instruction memory controller is depicted below in Figure 2.

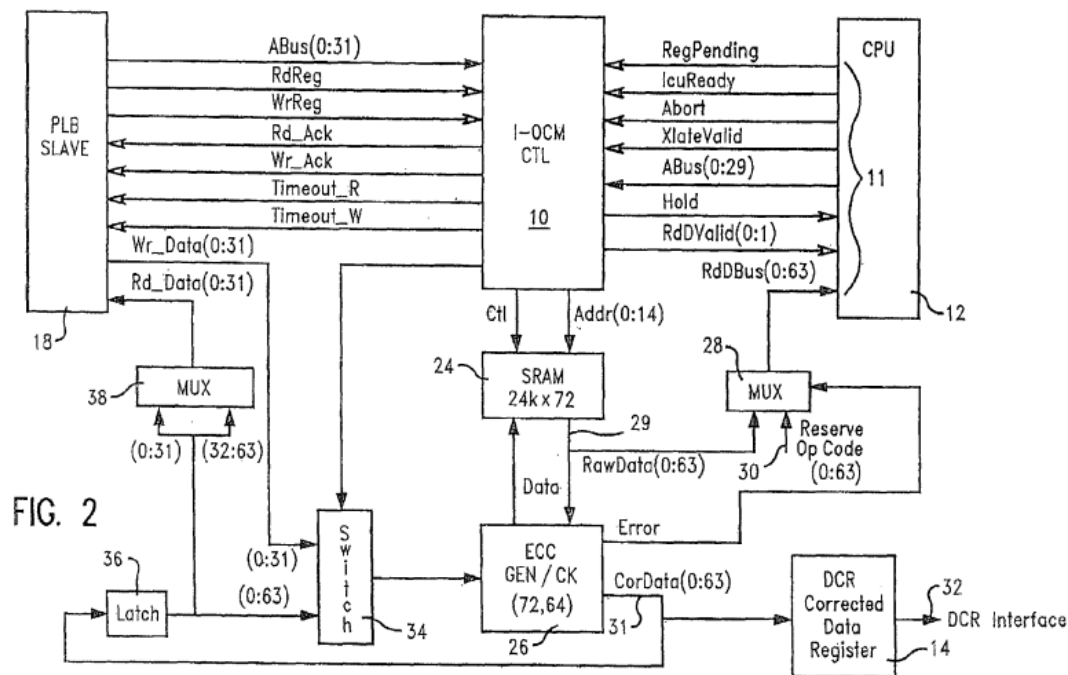


Figure 2 of Haswell is a schematic view of the instruction memory controller. Ex. 1006, 4:18–20. Haswell explains “an instruction side on-chip memory (I-OCM) unit 10 provides instructions to the computer processor unit (CPU) 12 via the processor’s I-OCM interface 11,” and that “[t]he purpose of the I-OCM unit is to minimize CPU instruction access time by providing a fast path to stored code.” Ex. 1006, 4:50–56. Haswell further discloses that “I-OCM unit 10 contains a large static random access memory (SRAM) 24 or other data memory device for instruction or other data code storage which is accessible for reads from CPU 12[.]” *Id.* at 7:10–13. According to Haswell, “[t]he I-OCM control unit 10 supplies instructions from the instruction SRAM 24 to CPU 12, via fetches over the CPU I-OCM interface 11.” *Id.* at 7:14–18.

Haswell describes how

The invention implements ECC [error correcting code] code to protect against soft and hard errors while providing instructions in a timely manner to avoid stalling the processor operation. When an error is detected, the unit replaces the corrupt instruction on the fly. This will then allow the processor to retrieve the corrected instruction in a subsequent operation.

Id. at 4:40–46. In the context of Haswell, “stalling” describes a situation where “instructions are not provided at the rate in which they are executed [by] the processor.” *Id.* at 1:17–19. Haswell further discloses that:

If an ECC error is detected in the data fetched from the SRAM, instead of passing on the corrupt instruction, I-OCM unit 10 swaps in a predetermined error-indicating signal or instruction 30 in place of the retrieved corrupted data, and presents it to CPU 12 via _ [sic] the multiplexer (MUX) 28, also within the same CPU clock cycle.

Id. at 7:41–46. Haswell explains that “[t]he predetermined error-indicating signal is a properly formed but reserved instruction or operation code” and “[i]f and when computer processor 12 attempts to execute the predetermined reserved error-indicating signal or reserved op-code 30, an interrupt is taken.” *Id.* at 7:47–48, 7:55–57.

B. Analysis of Independent Claim 1

1. “[pre] A method for error correction in an instruction cache that is operatively coupled to a processor, comprising the steps of:”³

Petitioner asserts that Haswell discloses the preamble because Haswell discloses “[a] **method of detecting error[s] during transfer of instructions from a data memory to a computer processor.**” Pet. 21

³ Patent Owner has stated that the preamble for claim 1 is not limiting, but regardless, the limitations are disclosed by Haswell. *See* Tr. 46:19–47:8.

(quoting Ex. 1006, code (57)) (citing Ex. 1002 ¶¶ 75, 76). Petitioner relies on Haswell's disclosure that "an instruction side on-chip memory (I-OCM) unit 10 provides instructions to the computer processor unit (CPU) 12 via the processor's I-OCM interface 11. The purpose of the I-OCM unit is to minimize CPU instruction access time by providing a fast path to stored code." Pet. 22 (quoting Ex. 1006, 4:50–56) (citing Ex. 1002 ¶ 77). Further, Petitioner notes that "Haswell discloses that 'I-OCM unit 10 contains a large static random access memory (SRAM) 24 or other data memory device for instruction or other data code storage which is accessible for reads from CPU 12.'" *Id.* (quoting Ex. 1006, 7:10–13). According to Petitioner, "[a] POSITA would understand that SRAM 24 is an instruction cache operatively coupled to CPU 12 via I-OCM unit 10." *Id.* (citing Ex. 1002 ¶ 78).

Patent Owner does not offer any argument related to the preamble.
See generally PO Resp.

Based on the final record, we determine Petitioner's contentions and supporting evidence persuasively show that Haswell discloses error correction as set forth in the preamble. Dr. Sechen testifies that Haswell discloses a method for error correction in an instruction cache and further explains how the instruction cache is operatively coupled to Haswell's processor. *See* Ex. 1002 ¶¶ 75, 78, 79. Haswell's focus is on error detection and the subsequent correction of errors. The Abstract discloses "[a]n attempt is made to correct the corrupted data in the raw data signal and, if it is corrected, the corrected data signal is subsequently retrieved and the corrected data signal is processed by the computer processor." Ex. 1006, code (57). Haswell states that "[t]his invention relates . . . to a method of and system for detecting error and implementing error correction during

transfer of instructions from memory to a computer processor.” *Id.* at 1:7–11; *see id.* at 2:13–15 (“a method and system for detecting error during transfer of data signals from a data memory that permits correction of the error”). Accordingly, Petitioner has shown by a preponderance of the evidence that Haswell discloses “error correction” as required by the preamble of claim 1.

2. “[1a] fetching a plurality of instructions stored in the instruction cache for execution by the processor”

Claim 1 further recites the step of “fetching a plurality of instructions stored in the instruction cache for execution by the processor.” Ex. 1001, 10:42–43.

Petitioner asserts that Haswell discloses an instruction side on-chip memory “I-OCM control unit 10 **supplies instructions** from the **instruction SRAM 24 to CPU 12**, via **fetches** over the CPU I-OCM interface 11.” Pet. 23 (quoting Ex. 1006, 7:14–16) (citing Ex. 1002 ¶ 81). “Thus,” according to Petitioner and Dr. Sechen, “a POSITA would understand that Haswell discloses claim element 1[a].” *Id.* (citing Ex. 1002 ¶ 81).

Dr. Sechen testifies that “Haswell discloses that ‘an instruction side on-chip memory (I-OCM) unit 10 provides instructions to the computer processor unit (CPU) 12 via the processor’s I-OCM interface 11. The purpose of the I-OCM unit is to minimize CPU instruction access time by providing a fast path to stored code.’” Ex. 1002 ¶ 77 (quoting Ex. 1006, 4:50–56). Dr. Sechen identifies the large static random access memory (SRAM) 24 within the I-OCM as containing instructions or other data code storage and also as accessible and reading from CPU 12. *Id.* Dr. Sechen testifies that “[b]ased on the above, a POSITA would understand that SRAM

24 is an instruction cache operatively coupled to CPU 12 via I-OCM unit 10.” *Id.* ¶ 78.

The heart of Patent Owner’s opposition is that Haswell’s SRAM 24 is not an instruction cache as required by claim 1. PO Resp. 31–41. The foundation of Patent Owner’s position is based on its proposed claim construction for “instruction cache.” *Id.* at 31–35. For the reasons set forth above, we do not agree with limiting the scope of an instruction cache as proposed by Patent Owner. We set forth Patent Owner’s arguments below, Petitioner’s response, and then our analysis.

Patent Owner contends that “Haswell’s SRAM is not an instruction cache,” because “a POSITA would have understood an instruction cache to be memory storing both instructions and tags for identifying whether a requested instruction is currently in the instruction cache.” PO Resp. 36 (citing Ex. 2005, 45:5–17, 65:11–14, 102:13–21). Patent Owner alleges that “Petitioner’s expert admitted that Haswell would need to disclose both a tag and instruction storage in order to disclose an ‘instruction cache’ as claimed.” *Id.* (emphasis omitted) (quoting Ex. 2005, 162:4–8) (emphasis omitted). According to Patent Owner, “Haswell does not disclose or even suggest storing tags,” and “Haswell does not even use the words ‘cache’ or ‘tag’ anywhere in the specification,” whereas “Haswell relates to the transfer of instructions from memory to a computer processor in an embedded computer system.” *Id.* at 37 (citing Ex. 1006, 1:9–19). Patent Owner explains that “[i]n Haswell’s system . . . the instructions are written to SRAM 24 from PLB slave port 18.” *Id.*

Patent Owner contends that “if Haswell’s SRAM or I-OCM control unit could make cache hit and cache miss determinations, it would need to access main memory whenever there was a cache miss,” yet, “Haswell does

not disclose or suggest that its SRAM 24 can access main memory.” PO Resp. 38–39. Patent Owner argues that “an instructions cache needs to be able to access main memory in the case of a cache miss,” but “Haswell’s SRAM 24 cannot.” *Id.* at 39 (citing Ex. 2003 ¶ 77).

Patent Owner dismisses Petitioner’s expert testimony that the different address widths shown in Figure 2 of Haswell are evidence of tag storage in the I-OCM controller because it fails to account for “how memory blocks of different sizes and types are accessed in embedded systems.” *Id.* at 40 (citing Ex. 2005 (Sechen Tr.) at 21:22–24:4; Ex. 2003 ¶¶ 78–80). Patent Owner relies on Dr. Przybylski’s testimony that “the different address widths shown in Figure 2 are consistent with the Haswell’s SRAM being on-chip memory but not an instruction cache because some of the bits are used to specify which memory is being accessed,” and “the difference between the sizes of the address buses to the I-OCM CTL unit 10 and the I-OCM CTL 10 to the SRAM 24 does not indicate that the SRAM is a part of an instruction cache or that a tag store exists.” *Id.* (citing Ex. 2003 ¶¶ 78–81). According to Patent Owner, “[a] POSITA would have understood that the difference merely indicates that the SRAM is a relatively small memory logically located within a larger address space.” *Id.* Patent Owner further argues “that I-OCM control unit 10 cannot request instructions from the PLB slave 18 or a memory on the far side the PLB slave 18,” and “[w]ithout a pathway for the I-OCM control unit to request instructions, it could not request an instruction in the event the processor requested an instruction not stored in SRAM 24.” *Id.* at 40–41 (citing Ex. 1006, Fig. 2, 4:56–58; Ex. 2003 ¶ 82).

Patent Owner concludes that “[b]ecause Haswell does not disclose or suggest an ‘instruction cache,’ it does not meet the limitation

‘[fetch/fetching] a plurality of instructions stored in the instruction cache,’ as recited in each of the independent claims.” PO Resp. 41. “Thus,” according to Patent Owner, “Haswell does not render any claims of the ’619 patent unpatentable.” *Id.* (citing Ex. 2003 ¶ 85).

In its Reply, Petitioner contends that “[a] POSITA would have understood that Haswell’s SRAM 24 is an instruction cache, because it is a fast and small memory (compared to the main memory) that is accessed before the main memory and that holds instructions.” Reply 8 (citing Ex. 1020 ¶ 23, Dr. Sechen Reply Decl.). According to Petitioner, “Haswell discloses that ‘an instruction side on-chip memory (IOCM) unit 10 provides instructions to the computer processor unit (CPU) 12,’” and “where the instructions are stored in the ‘static random access memory (SRAM) 24.’” *Id.* (citing Ex. 1006, 4:50–53, Fig. 1, 7:9–14, Fig. 2). Petitioner alleges that “[t]he SRAM 24 is an instruction cache . . . because access to it is fast, it is smaller than the main memory, and is accessed before the main memory.” *Id.* (citing Ex. 1020 ¶ 23).

Petitioner further contends that “Haswell discloses that ‘instructions are the raw, uncorrected output of the SRAM, which provides the best performance path,’ and further that the instructions ‘are transferred during a CPU access in order to minimize the number of cycles required for an instruction fetch’ and ‘to minimize the instruction access time.’” Reply 8 (citing Ex. 1006, 7:9–30, Fig. 2) (emphasis omitted). “Thus,” according to Petitioner, “the SRAM 24 is fast, and the CPU 12 can directly and quickly access instructions from the SRAM 24.” *Id.* (citing Ex. 1020 ¶ 24). Petitioner also establishes that Haswell’s SRAM 24 is small relative to the main memory. *Id.*

Dr. Sechen testifies that based on the size of Haswell's SRAM, and the knowledge of a POSITA in 2003, "a POSITA would have understood that SRAM 24 is not an 'onchip memory [that] performs much like a cache that never misses,' as Dr. Przybylski argues, *see* Ex. 2003 ¶¶ 83–84, but rather, an on-chip memory that is in fact an on-chip instruction cache." Ex. 1020 ¶¶ 28, 25–28. Dr. Sechen further opines that "the SRAM 24 is accessed before the main memory, because Haswell states that the SRAM 24 'is accessible for reads from CPU 12,' and that this 'provides the best performance path.'" *Id.* ¶ 29 (quoting Ex. 1006, 7:9–19, Fig. 2). Dr. Sechen relies on Haswell's disclosure "that when an instruction accessed from the SRAM 24 is erroneous, and cannot be corrected, the CPU 12 can 'retrieve the correct original data or instruction from a back-up memory, e.g., the main memory, via the PLB.'" *Id.* (quoting Ex. 1006, 8:9–11; citing *id.* at 7:55–67 (disclosing that the operations 1–4 are performed by the CPU 12, by executing the error handling routine).

Based on these disclosures in Haswell, Dr. Sechen reasons that "[b]ecause the CPU 12 can obtain instructions first from the SRAM 24, i.e., the instruction cache, and then from the main memory, as needed, Dr. Przybylski's allegation, that the I-OCM control unit 10 may not access the main memory via the PLB slave 18, *see* Ex. 2003 ¶¶ 47–48, 82, is irrelevant." Ex. 1020 ¶ 29. Dr. Sechen concludes that "a POSITA would have understood that Haswell's CPU 12 would access instructions from the SRAM 24 first, and in the event of uncorrectable errors, would access the instructions from a back-up memory, such as the main memory," and as such, "a POSITA would understand that the SRAM 24 is a cache memory according to the ordinary and customary meaning of that term." *Id.* ¶ 30.

Based upon the final record before us, we find Petitioner's, and Dr. Sechen's, contentions persuasive.

Petitioner has established persuasively that Haswell's SRAM 24 is an "instruction cache" or a memory for the storage of instructions that provides high speed access to the instructions to be executed by the processor. Haswell discloses that "an instruction side on-chip memory (I-OCM) unit 10 provides instructions to the computer processor unit (CPU) 12," and "I-OCM unit 10 contains a large static random access memory (SRAM) 24 or other data memory device *for instruction* or other *data code storage* which is accessible for *reads from CPU 12*." Ex. 1006, 4:50–53, 7:10–13, Fig. 1 (emphases added). Further, Haswell clarifies that "the term data is to be understood to include instructions to be executed by the computer processor." Ex. 1006, 4:37–39. Based on these disclosures, we are persuaded by Dr. Sechen's testimony that "Haswell's SRAM 24 is an instruction cache because it is a fast and small memory (compared to the main memory) that is accessed before the main memory and that holds instructions," and "because access to it is fast, it is smaller than the main memory, and is accessed before the main memory." Ex. 1020 ¶¶ 23, 29 (Dr. Sechen explaining how SRAM 24 is accessed before the main memory, because Haswell states that the SRAM 24 "is accessible for reads from CPU 12," and that this "provides the best performance path." Ex. 1006, 7:9–19).

In context, claim 1 of the '619 patent requires "fetching a plurality of instructions stored in the instruction cache for execution by the processor." Haswell's SRAM 24 functions as the claimed instruction cache because its "instructions are the raw, uncorrected output of the SRAM, which provides the best performance path," and "[i]n normal operation, raw, uncorrected SRAM data (minus the check bits)," i.e., the instructions from the SRAM

24, “are transferred during a CPU access in order to minimize the number of cycles required for an instruction fetch” and “to minimize the instruction access time.” Ex. 1006, 7:14–19, 7:23–29, Fig. 2; Ex. 1020 ¶ 24. As Dr. Sechen testifies, “the SRAM 24 is fast, and the CPU 12 can directly and quickly access instructions from the SRAM 24.” Ex. 1020 ¶ 24. We find persuasive Petitioner’s, and Dr. Sechen’s, explanations that “a POSITA would have understood that Haswell’s CPU 12 would access instructions from the SRAM 24 first, and only in the event of uncorrectable errors, would access the instructions from a back-up memory, e.g., the main memory.” Reply 9–10 (explaining how “CPU 12 can obtain instructions first from the SRAM 24, i.e., the instruction cache, and then from the main memory, as needed”); Ex. 1020 ¶ 30. Therefore, a POSITA would understand that Haswell’s SRAM 24 is an instruction cache memory according to the meaning of that term set forth above. *Id.*

Even if we were to limit the claim scope of “instruction cache,” to Patent Owner’s proffered definition (requiring an instruction store for storing instructions and a tag store for storing tags), we determine Petitioner has still presented persuasive evidence establishing that the Haswell’s system would store tags.⁴ Specifically, we find persuasive Dr. Sechen’s testimony that “several different locations in the main memory would be mapped to a single location in the SRAM 24,” and “[a] particular scheme of sharing the locations would depend on the cache configuration,” such that

a POSITA would understand that the particular locations of the main memory that are currently copied to the cache can be determined by the combination of the location of that word within the SRAM 24, which is specified by the 15-bit address

⁴ As noted above, independent claims 1, 21, and 29 do not recite tags or storing tags and the Specification does not demand such a requirement.

provided on the address bus “Addr(0:14)” and the remaining higher order bits of the full address provided by the CPU 12 on the 30-bit address bus “ABus(0:29).”

Ex. 1020 ¶¶ 31–32 (citing Ex. 1022, 268, 274, Figs. 7–13; Ex. 1006, Fig. 2).

Based on these disclosures, Dr. Sechen persuasively testifies that “[a] POSITA would understand that these remaining higher order address bits, entirely or in part, depending on the mapping scheme, indeed constitute the tag bits for the SRAM 24.” *Id.* ¶ 32 (citing Ex. 2005, 22:12–23:8) (Dr. Sechen testifying on cross examination that “there is either 15 or 17 address bits, the higher order address bits that would be the tag.”). We have considered Patent Owner’s positions that Figure 2 of Haswell does not specifically show any tag logic, and SRAM 24 would be an on-chip memory that does not use tags (*see* Sur-reply 17–20; Tr. 40:15–41:15), but we find more persuasive Dr. Sechen’s response that “[i]t is not shown explicitly, but it has to be there, because like I said the addresses are 30 or 32 bits. And you are only passing 15 of them to the cache, and yet it is able to successfully extract instructions which means that it necessarily has a tag.” Ex. 2005, 22:12–23:8.

3. *“[1b] each of the instructions being fetched during a respective one of a plurality of instruction cycles”*

Claim 1 further recites “each of the instructions being fetched during a respective one of a plurality of instruction cycles.” Ex. 1001, 10:43–45.

Petitioner asserts that in Haswell “each instruction stored in the instruction cache (*e.g.*, SRAM 24) is transmitted to the processor (*e.g.*, CPU 12) within a single clock cycle—*e.g.*, during a respective one of the processor’s instruction cycles.” Pet. 23 (Ex. 1002 ¶ 83). Petitioner relies on Haswell’s disclosure of “transmitting the raw data signal from the data memory to a computer processor, simultaneously checking the raw data

signal for corruption, and transmitting either the raw or predetermined reserved signal to the computer processor, are all performed within a single clock cycle.” *Id.* (quoting Ex. 1006, 2:61–67). Petitioner notes that “Haswell discloses a similar notion, namely ‘the computer processor **instruction unit is adapted to transmit either the raw or predetermined reserved instruction to the computer processor, within every a [sic] cycle.**’” Pet. 23–24 (quoting Ex. 1006, 3:60–62) (citing Ex. 1002 ¶ 84).

We find each of Petitioner’s contentions persuasive for these limitations. For the reasons set forth above, we have considered and found unpersuasive Patent Owner’s contentions that Haswell is not directed to detecting and correcting errors in instructions retrieved from an instruction cache. Patent Owner otherwise does not present any unique arguments related to this limitation.

Accordingly, based on our review and consideration of the final record, we determine that Petitioner has persuasively shown that Haswell discloses this limitation.

4. *“[1c] performing error detection for each of the instructions concurrently with the fetching of a respective one of the instructions.”*

Claim 1 further recites “performing error detection for each of the instructions concurrently with the fetching of a respective one of the instructions.” Ex. 1001, 10:46–48.

Petitioner asserts that Haswell “discloses that the error detection for an instruction occurs when an instruction is being fetched,” and Petitioner further notes that “claim 1 does not recite whether the detection occurs for the instruction currently being fetched or for a previously fetched instruction.” Pet. 24 (citing Ex. 1002 ¶ 86). Petitioner alleges that “Haswell

discloses that the error detection for an instruction occurs when that instruction is being fetched.” *Id.* Quoting Haswell, Petitioner argues “Haswell discloses that ‘the steps of transmitting the raw data signal [(the instruction)] from the data memory to a computer processor, simultaneously checking the raw data signal for corruption, and transmitting either the raw or predetermined reserved signal to the computer processor, are all performed within a single clock cycle.’” *Id.* (quoting Ex. 1006, 2:62–67) (citing Ex. 1002 ¶ 87). Haswell further discloses “[t]he checking of the raw data signal for presence of data corruption is preferably performed simultaneously with the commencement of transmission of the raw data signal from the data memory to the computer processor,” and “[t]he term data is to be understood to include instructions to be executed by the computer processor.” Ex. 1006, 3:7–10, 4:37–39, 7:37–46; *see also* Ex. 1002 ¶ 87.

We find each of Petitioner’s contentions persuasive for this limitation. Patent Owner does not present any unique arguments related to this limitation. Accordingly, based on the final record, we determine that Petitioner has persuasively shown that Haswell discloses this limitation.

5. Conclusion

We have reviewed the parties’ assertions, prior art, and other evidence, as described above, and find Petitioner has established by a preponderance of the evidence that Haswell discloses each limitation of claim 1. Accordingly, we are persuaded on the final record before us that Petitioner has shown by a preponderance of the evidence that independent claim 1 is anticipated by Haswell.

C. Analysis of Claims 10, 11, 13, 21, 22, 24, 26, and 29–31

1. Independent Claims 21 and 29

As noted above, independent claims 21 and 29 are nearly identical in scope to claim 1, except that they are system claims whereas claim 1 is a method claim. For claim 21, Petitioner relies on its contentions for claim 1. Pet. 28–33. Claim 29 also contains a means plus function limitation. Petitioner’s analysis closely follows that of claim 1 for each of these independent claims. *See* Pet. 28–33 (also providing a comparison of the analysis to claim 1 limitations), 36–40.

Claim 21 is similar to claim 1, but Petitioner notes, and discusses, one unique difference. Claim 1 requires, “performing error detection for,” but claim 21 recites “circuitry configured to detect an error in.” *See* Pet. 31–33. For this limitation of claim 21, Petitioner relies on Haswell’s disclosure in Figure 2, specifically “circuitry with sub-circuitry (ECC GEN/CK unit 26) that tests for data corruption (i.e., performs an error detection) in each of the instructions fetched from the instruction cache (SRAM 24).” Pet. 32. Petitioner contends that “because Haswell discloses that the error detection is ‘performed simultaneously with the commencement of transmission of the raw data signal from the data memory to the computer processor’ (Ex. 1006, 3:7–10, *see also id.* at 7:37–46), Haswell discloses claim element 21[c].” Pet. 33 (citing Ex. 1006, 7:30–34; Ex. 1002 ¶ 109).

For claim 29’s “means for detecting an error” limitation, Petitioner persuasively identifies Haswell’s “sub-circuitry or structure [Fig. 2]—ECC GEN/CK unit 26—configured to detect an error in each of the instructions fetched from the instruction cache (SRAM 24)” as performing the recited function. Pet. 39 (citing Ex. 1002 ¶ 129). Petitioner establishes persuasively that

ECC GEN/CK unit 26 is the same as or equivalent to the comparator 139 (in the error correction circuitry 129) of the '619 patent because both ECC GEN/CK unit 26 and comparator 139 are configured to check ECC generated codes to detect bit errors in each of the instructions concurrently with the fetching of a respective one of the instructions.

Pet. 40 (citing Ex. 1002 ¶ 129). As noted above, we determine Petitioner has shown by a preponderance of the evidence that claim 1 is anticipated by Haswell, and because claims 21 and 29 are nearly identical in scope, Patent Owner's contentions to the contrary apply equally to these claims. Based on the final record, we also find Petitioner's remaining uncontested contentions for claims 21 and 29 persuasive.

2. *Dependent Claims 10, 11, 13, 22, 24, 26, 30, and 31*

Petitioner contends that dependent claims 10, 11, 13, 22, 24, 26, 30, and 31, which depend directly or indirectly from independent claims 1, 21, or 29, are anticipated by Haswell, and provides explanation as to how the prior art discloses each limitation of these claims. Pet. 25–45.

Claim 10 depends from claim 1 and recites “the step of performing error detection for each of the instructions concurrently with the fetching of a respective one of the instructions further comprises the step of detecting an uncorrectable error in the instruction.” Petitioner explains persuasively that Haswell discloses: “If the error was not correctable, retrieve the correct original data or instruction from a back-up memory area attached to the PLB.” Pet. 25 (quoting Ex. 1006, 8:9–11) (citing Ex. 1002 ¶ 90) (emphasis omitted). Thus, according to Petitioner, “Haswell distinguishes between an error that is correctable and an error that is not correctable (e.g., uncorrectable) in the fetched instruction,” and “Haswell discloses claim 10—*i.e.*, claim element 1[c] plus the additional limitation ‘detecting an

uncorrectable error in the instruction.” *Id.* (citing Ex. 1002 ¶ 90). Based on the final record, we find Petitioner’s uncontested contentions for claim 10 persuasive.

Claim 11 depends from claim 10 and recites “the step of replacing the instruction in the instruction cache with a correct version of the instruction obtained from a random access memory.” Petitioner establishes how Haswell discloses that if the error was not correctable, the correct original data or instruction may be retrieved from a back-up memory, and as such, when the error is determined not to be correctable, Haswell discloses that the corrected instruction can be retrieved from the main memory to replace the corrupted instruction in the instruction cache (e.g., SRAM 24). Pet. 25–26 (citing Ex. 1006, 8:9–11; Ex. 1002 ¶ 92); *see also* Ex. 1002 ¶ 94 (explaining when the instruction is in error and not correctable, this will be treated like a cache miss, meaning that the “missing” or “correct” instruction is first passed to the cache, and then the processor accesses it from the cache). Based on the final record, we find Petitioner’s uncontested contentions for claim 11 persuasive.

Claim 13 depends from claim 1 and further requires “a first one of the instructions is fetched during a first one of the instruction cycles,” and “each of the instructions being fetched during a respective one of a plurality of instruction cycles.” Petitioner establishes how each of these limitations is disclosed by Haswell. Pet. 28 (citing Ex. 1002 ¶¶ 97, 98). First, Haswell discloses the instructions fetched during a one of a plurality of instruction cycles and Dr. Sechen testifies persuasively that a POSITA would understand that Haswell also discloses that a first one of the instructions can be fetched during a first one of the instruction cycles. Ex. 1002 ¶ 97. Next, Dr. Sechen explains persuasively why Haswell discloses claim 13’s

requirement that the error detection is performed during “the first one of the instruction cycles,” similar to the reasons set forth above for claim element 1[c]. *Id.* ¶ 98. Based on the final record, we find Petitioner’s uncontested contentions for claim 13 persuasive.

Claim 22 depends from claim 21 and recites “wherein the circuitry is further configured to generate a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction.” Petitioner establishes that once an error is detected in the fetched instruction from Haswell’s SRAM 24, ECC GEN/CK unit 26 determines whether the error is correctable, and if the error is correctable, ECC GEN/CK unit 26 proceeds to correct the error and store the corrected instruction (a corrected version of the instruction) to the I-OCM corrected data register (DCR) 14—*i.e.*, generate a corrected instruction. Pet. 33 (citing Ex. 1002 ¶ 112; Ex. 1006, 7:40–54). Further, Haswell discloses that one of the steps performed while writing the corrected instruction back to SRAM 24 includes the step of waiting for ECC-corrected data 31 to be generated and stored in the corrected data register. *See* Ex. 1006, 8:45–46; Ex. 1002 ¶ 113. Based on the final record, we find Petitioner’s uncontested contentions for claim 22 persuasive.

Claim 24 depends from claim 22 and recites “the circuitry is further configured to write the corrected instruction back to the instruction cache.” Petitioner establishes that Haswell uses I-OCM controller 10 as part of the circuitry that is responsible for writing the corrected instruction back to instruction cache SRAM 24. Pet. 34–35; Ex. 1006, 7:62–67 (“Then write the corrected data or instruction 31 back to SRAM 24, to substitute for the corrupted raw data 29, via a PLB 18 write.”). We find persuasive Petitioner’s showing that the PLB 18 write process writes the corrected data

or instruction, and I-OCM controller 10 provides the capability to load code into SRAM 24 via PLB 18, such that I-OCM controller 10 would be the structure responsible for writing the corrected instruction back to instruction cache SRAM 24. Pet. 35; Ex. 1002 ¶ 117. Based on the final record, we find Petitioner's uncontested contentions for claim 24 persuasive.

Claim 26 depends from claim 21 and recites “the circuitry is further configured to determine whether the error in the instruction is correctable.” Petitioner establishes that “Haswell discloses a circuitry configured to detect an error in each of the instructions,” and “the circuitry determines whether the error is correctable.” Pet. 35–36 (citing Ex. 1002 ¶ 120). Petitioner relies on Haswell's disclosure that “[a]n attempt is also made at the same time by ECC GEN/CK 26 to determine if the error is correctable, and then to correct the corrupted data by known correction techniques,” and then the system may “1. Determine if the error was correctable by reading the I-OCM status register. 2. If the error was correctable, read the corrected data or instruction 31 from the I-OCM corrected data register (DCR) 14 via the DCR bus or interface 32.” *Id.* (quoting Ex. 1006, 7:48–51, 7:60–64 (emphasis omitted)); Ex. 1002 ¶ 120. Petitioner provides further support about the details of the PLB 18 write process and explains how Haswell identifies I-OCM controller 10 as the structure responsible for writing the corrected instruction back to instruction cache SRAM 24. Pet. 35 (citing Ex. 1002 ¶ 117). We find persuasive Petitioner's contentions that “Haswell discloses that I-OCM controller 10 may ‘[i]nitiate an SRAM write, which will regenerate the ECC check bits and store the new 72 bit double word into the SRAM.’” *Id.* (citing Ex. 1006, 8:51–53; Ex. 1002 ¶ 117). Based on the final record, we find Petitioner's uncontested contentions for claim 26 persuasive.

Claim 30 depends from claim 29 and recites a means plus function claim requirement. Specifically, claim 30 requires “means for generating a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction.” Claim 30 is otherwise similar to claim 22 discussed above.

As noted previously in our claim construction analysis, Petitioner identifies the corresponding structure from the '619 patent as the error correction logic 143 or circuit 129 for performing the generating a corrected instruction function for claim 30's means element. Pet. 41–42. Petitioner establishes that “Haswell identifies ECC GEN/CK unit 26 as the structure generating the corrected instruction if an error is detected in the instruction being fetched,” and that “ECC GEN/CK unit 26 shown in Figure 2 of Haswell is the same as, or equivalent to, the respective structure described in the '619 patent, because both ECC GEN/CK unit 26 and error correction logic 129 generate ECC, check or compare ECC, and make corrections based on ECC.” Pet. 42–43 (citing (Ex. 1002 ¶ 135). Based on the final record, we find Petitioner's uncontested contentions for claim 30 persuasive.

Claim 31 depends from claim 30, and recites “means for writing the corrected instruction back to the instruction cache.” As noted above in our claim construction analysis, the corresponding structure is a multiplexer, or multiplexers. Petitioner establishes that “Haswell discloses equivalent structure,” specifically, “I-OCM controller 10 as the structure performing the function of ‘writing the corrected instruction back to the instruction cache’ as discussed in claim 24, which is identical to claim 31.” Pet. 44 (citing Ex. 1002 ¶¶ 139–140). Petitioner also directs us to its contentions for claim 24, which contains the same limitation. *Id.* Based on the final record, we find Petitioner's uncontested contentions for claim 30 persuasive.

Patent Owner does not specifically respond to Petitioner's arguments related to the additional dependent claims discussed above. *See generally* PO Resp. As such, Patent Owner has waived any arguments related to these claims. *See* Paper 10, at 8 (Scheduling Order) ("Patent Owner is cautioned that any arguments not raised in the response may be deemed waived.").

Based on our review and consideration of the final record, we determine that Petitioner has established by a preponderance of the evidence that Haswell discloses the limitations of claims 10, 11, 13, 21, 22, 24, 26, and 29–31 and that these claims are anticipated by Haswell.

VI. OBVIOUSNESS OVER HASWELL

Because Petitioner has persuasively established that claims 1, 10, 11, 13, 21, 22, 24, 26, and 29–31 are anticipated by Haswell, we decline to address Petitioner's alternative contention that these same claims would have also been obvious over Haswell to the person of ordinary skill in the art. *See* Pet. 21 ("Haswell Anticipates and Renders Obvious Claims 1, 10, 11, 13, 21, 22, 24, 26, and 29-31"). *See Boston Sci. Scimed, Inc. v. Cook Grp. Inc.*, 809 F. App'x 984, 990 (Fed. Cir. 2020) ("[T]he Board need not address issues that are not necessary to the resolution of the proceeding.").

VII. OBVIOUSNESS OVER HASWELL AND ITOU

Petitioner asserts dependent claims 2–9, 14–20, 23–25, 27, and 32 would have been obvious over the combination of Haswell and Itou. Pet. 45–82. Having now considered the evidence in the complete record established during trial, we are persuaded that, based on this record, Petitioner has demonstrated by a preponderance of the evidence that these claims would have been obvious over those references in combination.

A. Overview of Itou

Itou is titled “Control System and Memory Control Method Executing a Detection of an Error in a Formation in Parallel With Reading Operation.” Ex. 1007, code (54). The invention generally relates to control of access to a memory in order “to provide a memory control technology capable of ensuring a high reliability by having a data error correcting function and exhibiting the same performance as an architecture containing no error correcting mechanism.” *Id.* at 1:9–10, 1:40–44. Itou discloses a memory control device which, among other components, includes a memory module and an error detecting unit executing an error detection operation in parallel with a reading operation of the memory module. *Id.* at code (57). Itou explains that when an error is detected, the error detecting unit corrects the error in the information containing the detected error. *Id.*

The memory control device (circuit) of Itou is shown below in Figure 1:

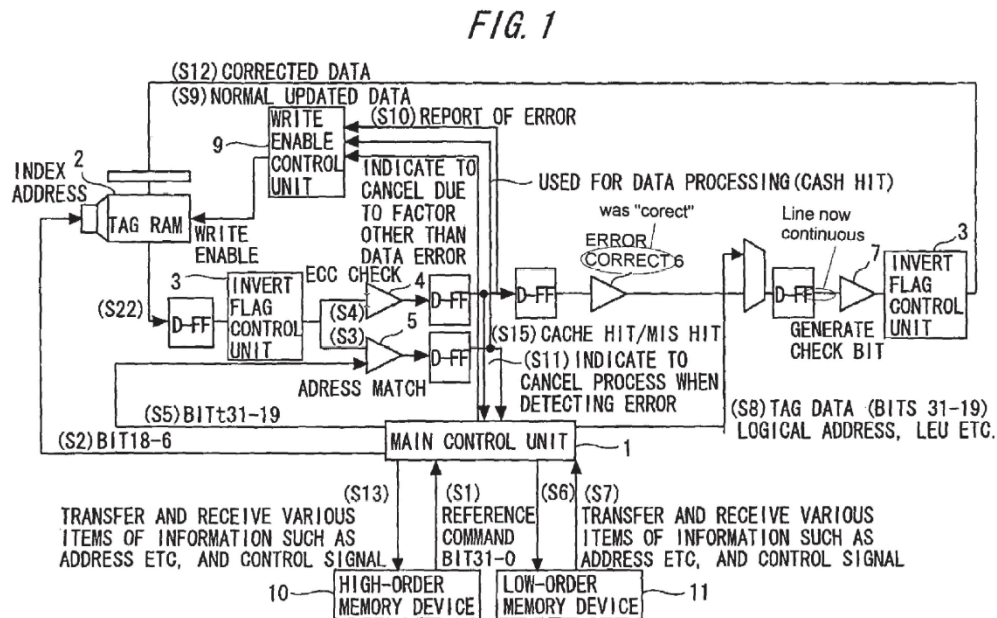


Figure 1 of Itou depicts a block diagram showing a hardware architecture of a memory control circuit. *Id.* at 2:48–50. “This memory control circuit

includes tag RAM 2 serving as a component of a cache memory, a data write circuit for writing data to tag RAM 2, a data read circuit for reading from tag RAM 2, and main control unit 1 that controls the data write and read to and from tag RAM 2.” *Id.* at 3:9–15. As shown in Figure 1, the CPU (high-order memory device 10) is coupled to tag RAM 2 via main control unit 1.

Itou recognizes that prior methods of data protection include a parity check, but Itou states that “[t]he parity check is, though capable of detecting an error, incapable of correcting the error.” *Id.* at 1:10–17. Itou explains that according to the parity check method, “an access time is delayed to a great degree due to the error correcting process,” and invalidating the data causes the logic to become intricate in such a method. *Id.*

B. Petitioner’s Contentions

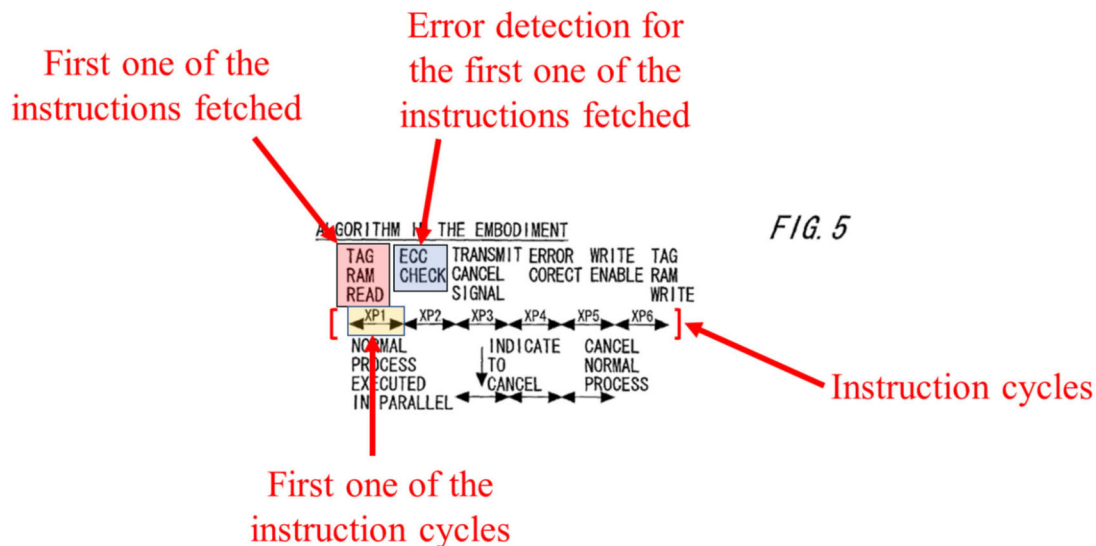
1. Claim 2

Claim 2 depends from claim 1 and additionally requires, “a first one of the instructions is fetched during a first one of the instruction cycles; and the error detection is performed for the first one of the instructions during a second one of the instruction cycles that is subsequent to the first one of the instruction cycles.”

The first portion of claim 2 is identical to claim 13, and Petitioner relies on its prior analysis based on Hawell for these limitations, which we found persuasive. *See* Pet. 45. Petitioner also relies on Itou and argues “[a] POSITA would understand that the data read from tag RAM memory during a first one (XP1) of the instruction cycles correspond to ‘a first one of the instructions [] fetched during a first one of the instruction cycles,’” such that “a POSITA would understand that fetching information or data from a tag RAM memory as described by Itou can be equally applied to fetching

instructions from an instruction cache—e.g., as disclosed by Haswell.”
Pet. 45–46 (Ex. 1002 ¶ 144).

The second portion of claim 2 “requires that the error detection (for the first one of the instructions) occurs during a cycle that is subsequent to the first one of the instruction cycles—e.g., during a second one of the instruction cycles.” Pet. 46. Petitioner contends that “Haswell discloses ‘performing error detection for each of the instructions’ fetched from instruction cache SRAM 24.” Pet. 47 (citing Ex. 1002 ¶ 147). Petitioner relies on the testimony of Dr. Sechen, and contends that “Itou further discloses that the error detection occurs ‘for the first one of the instructions during a second one of the instruction cycles that is subsequent to the first one of the instruction cycles.’” *Id.* (citing Ex. 1002 ¶ 147). Petitioner contends that “[b]ecause Itou’s teachings are not exclusive to a tag RAM (which is part of the cache memory) and can be applied to other portions or types of the cache . . . such as the instruction portion of the instruction cache,” Petitioner reasons that “a POSITA would understand that fetching data from a tag RAM memory, as disclosed by Itou, encompasses fetching instructions from an instruction cache.” *Id.* (citing Ex. 1002 ¶ 148). Petitioner further relies on Figure 5 of Itou showing “data (the first one of the instructions) from tag RAM 2 are fetched during a first cycle XP1 (a first one of the instruction cycles) and undergo an Error Check Code (ECC) check (error detection) during a subsequent second cycle XP2,” as depicted below in annotated and highlighted Figure 5 of Itou. *Id.* at 47–48.



Petitioner's highlighted and annotated Figure 5 of Itou showing instruction cycles and a first one of the instruction cycles fetched with subsequent error detection for the first one of the instructions fetched. Pet. 48. Itou explains that "the data are read from the tag RAM (XP1), the ECC check is executed (XP2), and the cancel signal is transmitted when detecting the error (XP3)." Ex. 1007, 8:65–67; Ex. 1002 ¶ 148.

Petitioner contends "that an 'ECC check' as described by Itou, corresponds to 'the error detection' recited in claim limitation 2[b]," because "[w]hen 'using the error correcting circuit . . . the data read from the memory undergo an execution of the error detection and error correction and, after removing the error from the data, undergo an execution of the essential processing.'" Pet. 48 (quoting Ex. 1007, 1:26–29) (citing Ex. 1002 ¶ 149) (emphasis omitted). Petitioner argues that a POSITA would deem this approach "beneficial in situations where the frequency of the error is low and performing the error detection in a separate clock cycle would enable a shorter clock cycle period." Pet. 49 (citing Ex. 1002 ¶ 50).

2. *Claim 3*

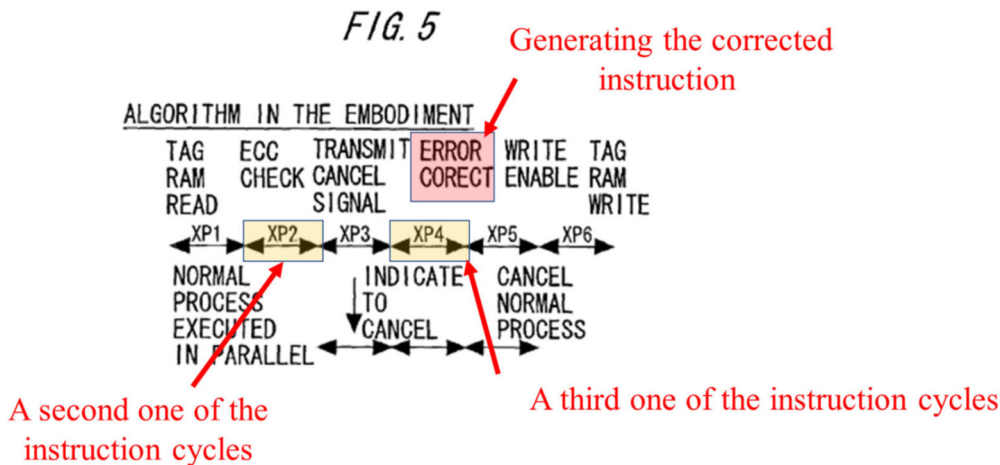
Claim 3 depends from claim 2, and recites that “the step of generating a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction.” Petitioner contends that Haswell and Itou teach claim 3, which is similar in scope to claim 22 examined above. Pet. 49 (citing Ex. 1002 ¶ 151). Petitioner first relies on its analysis of claim 22 and Haswell’s disclosure applied to the limitations of that claim. *Id.*

Petitioner also contends that “Itou discloses that ‘when detecting a correctable error, this memory control circuit executes the error correction in a short period of time without rereading the data from the low-order memory device 11.’” Pet. 50 (quoting Ex. 1007, 9:15–18) (citing Ex. 1002 ¶ 154). Thus, according to Petitioner, “Itou discloses ‘the step of generating a corrected instruction if the error is detected in the instruction.’” *Id.* Further, “Itou explains that the generated corrected instruction is ‘a corrected version of the instruction’ because it ultimately replaces the instruction where the error is detected.” *Id.* (citing Ex. 1002 ¶ 155; Ex. 1007, 8:13–15 (“The bits containing the error occurred undergo an execution of the error correction.”)). Petitioner relies on Itou’s disclosure that “[i]f the error data is data used for process and if the processing is not canceled due to the factor other than the data error, the memory control circuit in this embodiment enables the error-corrected data to be written.” *Id.* (quoting Ex. 1007, 4:63–66) (citing Ex. 1002 ¶ 155).

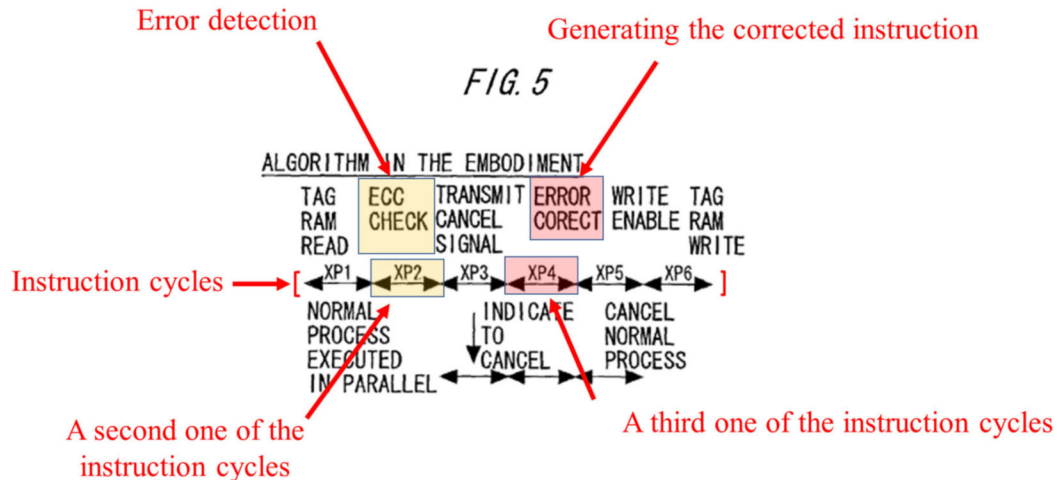
3. *Claim 4*

Claim 4 depends from claim 3 and recites “wherein the step of generating the corrected instruction is performed during a third one of the instruction cycles, the third one of the instruction cycles being subsequent to

the second one of the instruction cycles.” Petitioner contends that “Itou in combination with Haswell discloses claim 4.” Pet. 50–51 (citing Ex. 1002 ¶ 157). Petitioner relies on annotated Figure 5 of Itou, depicted below, which “shows that the error check (error detection) is performed in cycle XP2, and the error correction process is initiated and requested (the step of generating the corrected instruction) in cycle XP4, which is **after** cycle XP2 in which the error check (error detection) is performed.” Pet. 51 (citing Ex. 1002 ¶ 158).



Petitioner’s annotated Figure 5 of Itou showing error correction (generating the corrected instruction) and block XP4 as a third one of the instruction cycles. Pet. 51. Petitioner relies on another annotated Figure 5, depicted below, to show “the data are read from the tag RAM (XP1), **the ECC check is executed (XP2)**. . . . Moreover, after transmitting the cancel signal, **the error correction (XP4) is executed**, the write enable signal is asserted (XP5), and the corrected data are written to the tag RAM 2 (XP6).” Pet. 52 (quoting Ex. 1007, 8:65–9:8) (citing Ex. 1002 ¶ 158).



Petitioner’s annotated Figure 5 of Itou further highlighting the ECC check (error detection) being executed (XP2). Pet. 52.

According to Petitioner, “[a] POSITA would understand that the error correction (generating the corrected instruction) occurs in a subsequent cycle (XP4) from the error check (error detection), which occurs in cycle XP2,” and as such, “the error correction occurs during an instruction cycle that is after the second one of the instruction cycles—e.g., during ‘a third one of the instruction cycles’ shown in Figure 5.” *Id.* (citing Ex. 1002 ¶ 159).

4. Claim 5

Claim 5 depends from claim 3, and recites “the step of stalling the processor during the generation of the corrected instruction.” Petitioner contends that Haswell and Itou teach this additional limitation of claim 5. Pet. 53. Petitioner first relies on the description in the ’619 patent that the processor “is stalled while it waits for the corrected instruction,” and based on these descriptions reasons that “a POSITA would understand that when a processor is stalled, the processor can be executing a no operation (NOP) command or an error handling routine.” *Id.* (citing Ex. 1002 ¶ 163).

Petitioner argues that “Haswell discloses: ‘If an ECC error is detected in the data fetched from the SRAM, . . . **I-OCM unit 10 swaps in a predetermined error-indicating signal or instruction 30 in place of the retrieved corrupted data** . . . within the same CPU clock cycle.’” *Id.* (quoting Ex. 1006, 7:40–45) (citing Ex. 1002 ¶ 164). Petitioner contends that “when an error is corrected, the transmission of the error-containing data (corrupted data) from the instruction cache to the processor is stopped,” and “[i]nstead, a predetermined error-indicating signal or instruction 30 is transmitted to the processor in place of the retrieved corrupted data.” Pet. 53–54. Haswell discloses that “an interrupt is taken” when the Haswell’s process attempts to execute the predetermined reserved error indicating signal, and “[t]he interrupt causes the processor to execute an error handling routine instead of executing instructions from the instruction cache.” Pet. 54 (citing Ex. 1002 ¶ 164). According to Petitioner, “the processor is stalled as would be understood by a POSITA in view of the ’619 patent.” *Id.* Petitioner further relies on Haswell’s disclosure that “when the substituted instruction is executed, it preferably causes processor 12 to execute an error handling routine which can retrieve the corrected instruction stored by IOCM unit 10, update SRAM 24, **and continue execution of the instructions.**” *Id.* (quoting Ex. 1006, 8:22–27) (citing Ex. 1002 ¶ 164). “Therefore,” according to Petitioner, “the processor is stalled—e.g., instructions from the instruction cache are not executed—until the error handling routine is fully executed.” *Id.*

Petitioner next relies on Itou. Petitioner contends that “Itou is not exclusive to a tag RAM (which is part of the cache) and can be applied to other portions or types of the . . . such as the instruction portion of the instruction cache.” Pet. 55. Petitioner argues that “a POSITA would

understand that fetching data from a tag RAM memory, as disclosed by Itou, encompasses fetching instructions from an instruction cache,” and “Itou explains that when an error is detected, the ECC check unit transmits a **process cancel signal** to the main control unit to prevent the main control unit from reading data from the cache memory.” *Id.* (citing Ex. 1007, 8:5–10; Ex. 1002 ¶ 167). Petitioner reasons that “[d]uring the time the main control unit is stopped from reading (and therefore executing) data from the tag RAM memory, there is no data output to the processor from the main control unit—*e.g.*, the processor can no longer execute instructions from the cache, and it is stalled,” and “[a]t the same time, the corrected instruction is being generated.” *Id.* (citing Ex. 1007, 8:11–15; Ex. 1002 ¶ 167). Based on additional disclosures in Itou, Petitioner concludes that “a POSITA would understand that while the process cancel signal is being transmitted, the processor stops receiving the requested data via the main control unit, and is stalled.” Pet. 56 (citing Ex. 1002 ¶ 168).

5. Claim 6

Claim 6 depends from claim 3 and recites “the step of writing the corrected instruction back to the instruction cache.” Petitioner first notes that claim 6 is similar in scope with claim 24, and Petitioner relies on its prior analysis showing how Haswell discloses the limitations of claim 24. Pet. 57.

Petitioner also relies on Itou as teaching this limitation. *Id.* Petitioner relies on Itou’s disclosure of an error correction unit used for correcting the bit error of the data of which the error has been detected by the ECC check unit and the disclosure “that ‘the corrected instruction’ is written—‘[i]f the error data is data used for process and if the processing is not canceled due to the factor other than the data error, **the memory control circuit in this**

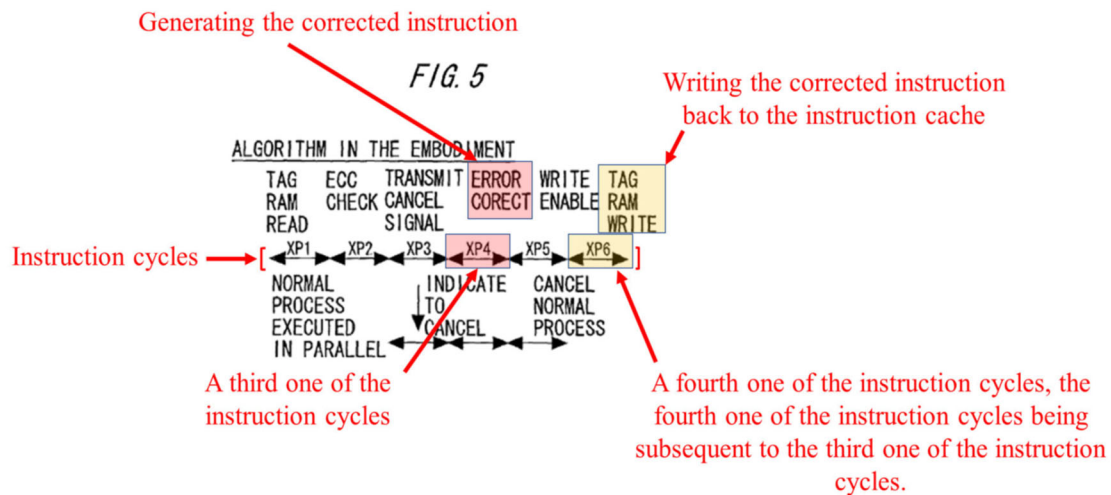
embodiment enables the error-corrected data to be written.” Pet. 58 (citing Ex. 1007, 4:63–66; Ex. 1002 ¶ 176). Next, “Itou explains that **the corrected instruction is written back to the instruction cache.**” *Id.* (citing Ex. 1002 ¶ 177; Ex. 1007, 9:5–8 (“[A]fter transmitting the cancel signal, the error correction (XP4) is executed, the write enable signal is asserted (XP5), and **the corrected data are written to the tag RAM 2 (XP6).**”)).

6. *Claim 7*

Claim 7 depends from claim 6, and recites “the step of generating the corrected instruction is performed during a third one of the instruction cycles, the third one of the instruction cycles being subsequent to the second one of the instruction cycles,” (7[a]) and “the step of writing the corrected instruction back to the instruction cache is performed during a fourth one of the instruction cycles, the fourth one of the instruction cycles being subsequent to the third one of the instruction cycles” (7[b]). Petitioner first notes that the first portion of the claim, “[e]lement 7[a], is identical to claim 4, which Itou and Haswell discloses.” Pet. 59.

As for the second portion of the claim, element 7[b], Petitioner contends that the combination of Itou and Haswell teach this element, which “requires that **the step of writing the corrected instruction** back to the instruction cache **occurs after the step of generating the corrected instruction.**” *Id.* (citing Ex. 1002 ¶ 181). Petitioner relies on an annotated and highlighted Itou Figure 5, depicted below, to show how “Itou discloses that the error correction (generating the corrected instruction) is executed in cycle XP4 (*e.g.*, the claimed third one of the instruction cycles) and the corrected data is written to tag RAM 2 (writing the corrected instruction back to the instruction cache) in cycle XP6—*e.g.*, a cycle subsequent to XP4

cycle as required by claim element 7[b].” Pet. 59–60 (citing Ex. 1002 ¶ 182).



Petitioner’s partial annotated and highlighted Figure 5 of Itou demonstrating how after transmitting the cancel signal, the error correction (XP4) is executed and the corrected data are written to the tag RAM 2 (XP6).

Pet. 60. According to Petitioner, “[a] POSITA would understand that cycle XP6, which is subsequent to cycle XP4, can be any cycle after cycle XP4,” and “cycle XP6 can be the claimed “fourth cycle” as shown in Figure 5 above.” *Id.* (citing Ex. 1002 ¶ 183).

7. Claim 8

Claim 8 depends from claim 6 and recites “the step of writing the corrected instruction back to the instruction cache further comprises the step of writing the corrected instruction to the instruction cache over the instruction stored in the instruction cache.” Petitioner relies on Haswell’s disclosure “that ‘if the error was correctable, read the corrected data or instruction 31 from the I-OCM corrected data register (DCR) 14 via the DCR bus or interface 32. Then **write the corrected data or instruction 31**

back to SRAM 24, to substitute for the corrupted raw data 29, via a PLB 18 write.” Pet. 61 (quoting Ex. 1006, 7:62–67; Ex. 1002 ¶ 185).

Petitioner also asserts that Itou teaches this limitation. Petitioner argues

Itou explains: “[t]he error correcting unit 6 corrects the bit error of the data of which the error has been detected by the ECC check unit 4” (Ex. 1007, 5:19-21; Ex. 1002, ¶187) and that “[t]he **data in the write target line** (e.g., the line 0 or 1 contained in the way 0) **are updated**, while the data in the write non-target line are written intact again to the tag RAM 2.” (Ex. 1007, 8:42-45; Ex. 1002, ¶187.)

Pet. 61. Petitioner concludes that “[a] POSITA would understand that Itou replaces error-containing information with corrected information in the same address of the tag RAM.” *Id.* (citing Ex. 1002 ¶ 188).

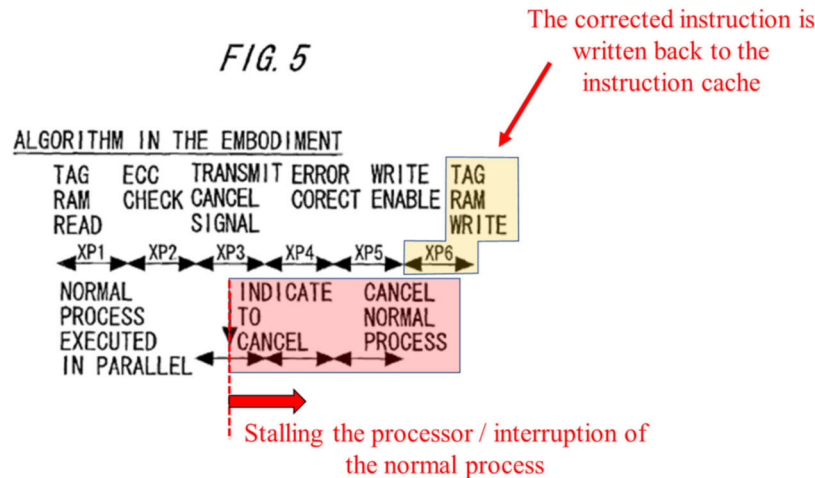
8. *Claim 9*

Claim 9 depends on claim 6 and recites “the step of stalling the processor while the corrected instruction is written back to the instruction cache.” Petitioner asserts “[c]laim 9 is similar in scope to claim 5, which is disclosed by Haswell and Itou.” Pet. 62. Petitioner explains that “[t]he difference between claims 5 and 9 is that in claim 9 ‘the step of stalling the processor’ occurs ‘while the corrected instruction is written back to the instruction cache.’” *Id.* Petitioner relies on Haswell’s teaching “that while the processor is stalled, the corrected instruction is being generated and stored in the I-OCM corrected data register (DCR) 14.” *Id.* (citing Ex. 1002 ¶ 191). Further, Petitioner notes “that following the step of generating the corrected instruction, and while the processor remains stalled, the corrected instruction is written back to SRAM 24 (e.g., the instruction cache).” *Id.* Petitioner quotes a portion of Haswell explaining how if an error is correctable, the corrected data may be read from the I-OCM corrected data

register and then write the corrected data back to SRAM 24, to substitute for the corrupted raw data 29. Pet. 62–63 (citing Ex. 1006, 7:62–67; Ex. 1002 ¶ 191). Petitioner contends that “[s]ubsequently, the processor (CPU 12) processes the corrected instruction, at which point the processor resumes executing instructions from the instruction cache (SRAM 24) and is no longer stalled.” Pet. 63 (citing Ex. 1002 ¶ 191).

Petitioner next relies on Itou. *Id.* According to Petitioner, “Itou discloses that when an error is detected in the data fetched from the cache memory, the ECC check unit 4 transmits a process cancel signal to the main control unit that prevents the main control unit from reading data from the cache memory.” *Id.* (citing Ex. 1007, 8:5–9; Ex. 1002 ¶ 193). Petitioner argues that “[t]his signal transmission continues until corrected data (corrected instruction) is provided (*e.g.*, while the corrected instruction is written back to the instruction cache).” *Id.* (citing Ex. 1007, 8:13–15; Ex. 1002 ¶ 193). Petitioner quotes portions of Itou that explain when detecting the error, the ECC check unit transmits a process cancel indication to the main control unit and the main control unit stops reading the data from the line of the data RAM that corresponds to the cache hit signal, and executes a rereading process while the bits containing the error occurred undergo an execution of the error correction. Pet. 63–64 (quoting Ex. 1007, 8:5–28) (citing Ex. 1002 ¶ 193).

Petitioner relies on a highlighted and annotated Figure 5 of Itou, depicted below, and argues that “Itou shows that when an error is detected during the ECC check, a cancel signal is transmitted thereafter that disrupts the normal process (*e.g.*, the address matching, or the data reading process), and the corrected instructions are provided while the normal process is interrupted by the cancel signal.” Pet. 64.



Petitioner's partial annotated and highlighted Figure 5 of Itou shows "TAG RAM WRITE" highlighted at XP6 (cycle) and also how the corrected instruction is written back to the instruction cache and the processor is stalled to interrupt the normal process. Pet. 64. Petitioner argues that "Itou discloses that the cancel signal is transmitted when detecting the error (XP3), and the corrected data are written to tag RAM 2." *Id.* (citing Ex. 1002 ¶ 194). Petitioner concludes, "POSITA would understand that once the cancel signal is transmitted the processor is stalled," and [d]uring the time the normal process is stalled, the corrected instruction is written to the tag RAM as shown in Figure 5." *Id.*

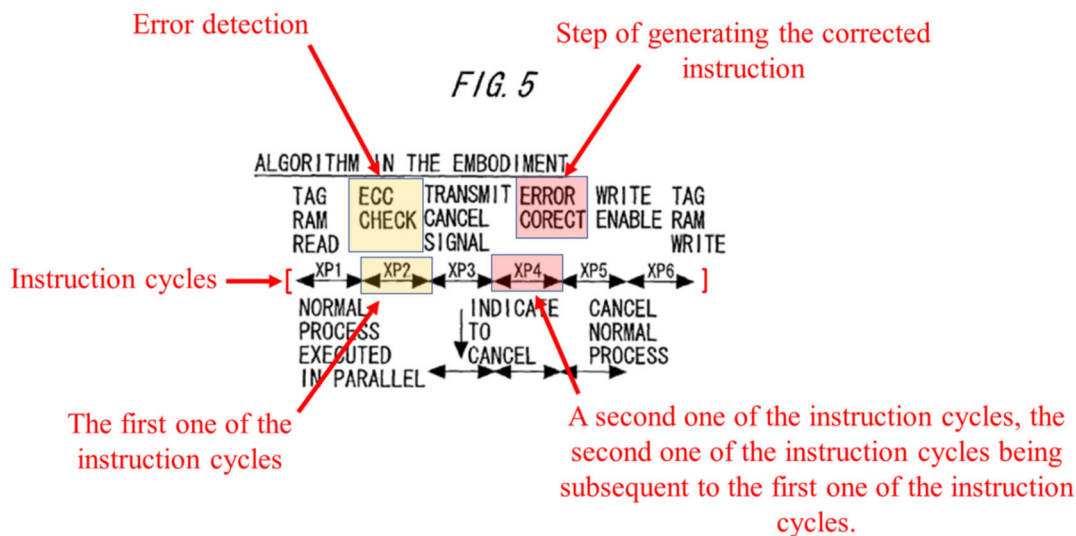
9. Claim 14

Claim 14 depends from claim 13 and recites "the step of generating a corrected instruction if the error is detected in the instruction, the corrected instruction being a corrected version of the instruction." Claim 14 is identical to claim 3, and Petitioner relies on its previous analysis of the combination of Haswell and Itou applied to claim 3 as also teaching the limitations of claim 14. Pet. 65 (citing Ex. 1002 ¶ 197).

10. *Claim 15*

Claim 15 depends from claim 14 and recites “wherein the step of generating the corrected instruction is performed during a second one of the instruction cycles, the second one of the instruction cycles being subsequent to the first one of the instruction cycles.” According to Petitioner, “[c]laim 15 recites that the ‘the step of generating the corrected instructions is performed’ in a cycle **after** the step of ‘the error detection.’” Pet. 65 (citing Ex. 1002 ¶ 198).

Petitioner relies on Itou Figure 5, depicted below, and argues “Itou shows that for its memory control unit, the error check (error detection) is performed in cycle XP2 and the error correction process is initiated and requested in cycle XP4, which is **after** cycle XP2.” Pet. 66



Petitioner’s partial annotated and highlighted Figure 5 of Itou shows highlighted XP2 and XP4 instruction cycles with XP4 labeled by Petitioner as “[a] second one of the instruction cycles, the second one of the instruction cycles being subsequent to the first one of the instruction cycles.” Pet. 65 (citing Ex. 1007, Fig. 5; Ex. 1002 ¶ 199). According to Petitioner, and Dr. Sechen, “[a] POSITA would understand that Itou, in combination with

Haswell, discloses that ‘the step of generating the corrected instruction is performed’ in a cycle subsequent to that of ‘the error detection.’” *Id.* at 66 (citing Ex. 1002 ¶ 200).

11. Claim 16

Claim 16 depends from claim 14 and recites “the step of stalling the processor during the generation of the corrected instruction.” Claim 16 is identical to claim 5, and Petitioner relies on its previous analysis of the combination of Haswell and Itou applied to claim 5 as also teaching the limitations of claim 16. Pet. 66–67 (citing Ex. 1002 ¶ 201).

12. Claim 17

Claim 17 depends from claim 14 and recites “the step of writing the corrected instruction back to the instruction cache.” Claim 17 is identical to claim 6, and Petitioner relies on its previous analysis of the combination of Haswell and Itou applied to claim 6 as also teaching the limitations of claim 17. Pet. 67 (citing Ex. 1002 ¶ 203).

13. Claim 18

The scope of claim 18 is nearly identical to that of claim 7, examined above. Petitioner notes “the only difference being the recited cycles.” Pet. 67–68 (noting claim 18 uniquely recites “a second one of the instruction cycles” and “a third one of the instruction cycles” as compared to a third and fourth cycle for claim 7). Petitioner contends that “the temporal sequence and the content of the operations in claims 7 and 18 remain the same,” thus, “because Haswell and Itou disclose claim 7, Haswell and Itou disclose claim 18.” Pet. 69.

Petitioner additionally argues that Itou in combination with Haswell discloses claim element 18[a] because that element is identical to claim 15, which is disclosed by Itou, in combination with Haswell, as Petitioner set

forth above. *Id.* Petitioner further argues that “[a] POSITA would understand that Itou and Haswell discloses ‘the step of writing the corrected instruction back to the instruction cache is performed’ in a cycle subsequent to ‘the step of generating the corrected instruction’ because, as discussed in claim 7, Itou discloses this temporal sequence of events.” Pet. 70 (citing Ex. 1002 ¶¶ 209–211).

14. Claim 19

Claim 19 depends from claim 17 and is identical to claim 8, which Petitioner contends “is disclosed by Haswell and Itou” as discussed above. Pet. 70–71. Petitioner relies on its previous analysis of the combination of Haswell and Itou applied to claim 8 as also teaching the limitations of claim 19. *Id.* (citing Ex. 1002 ¶ 212).

15. Claim 20

Claim 20 depends from claim 17 and is identical to claim 9, which Petitioner contends “is disclosed by Haswell and Itou as discussed above.” Pet. 71. Because claim 20 is identical to claim 9, Petitioner relies on its previous analysis of the combination of Haswell and Itou applied to claim 9 as also teaching the limitations of claim 20. *Id.* (citing Ex. 1002 ¶ 212).

16. Claim 23

Claim 23 depends from claim 22, and has a substantially similar scope to claim 5, which Petitioner alleges “is disclosed by Haswell and Itou.” Pet. 71 (citing Ex. 1002 ¶ 217). Claim 23 recites, “the circuitry stalls the processor during the generation of the corrected instruction.”

Petitioner contends that the combination of Haswell and Ito “disclose the additional limitation of claim 23 because their respective circuitries are configured to perform the operation recited in claim 5.” *Id.* at 72. Specifically, Petitioner contends

Haswell teaches that “I-OCM unit 10 swaps in a predetermined error indicating signal or instruction 30 in place of the retrieved corrupted data” (Ex. 1006, 7:43-45; Ex. 1002, ¶218), which stalls the processor—*i.e.*, forces the processor to execute an error handling routine instead of executing the corrupted instruction. (Ex. 1002, ¶218.) Therefore, Haswell discloses that part of the disclosed circuitry shown in Figure 2 (e.g., I-OCM unit 10) “stalls the processor during the generation of the corrected instruction” as recited in claim 23. (Id.)

Pet. 72. Petitioner also relies on Itou’s disclosure “that ECC check unit 4 is responsible for transmitting a process cancel indication to stall the processor,” and, as such, “Itou discloses that part of the disclosed circuitry shown in Figure 1 (e.g., ECC check unit 4) ‘stalls the processor during the generation of the corrected instruction’ as recited in claim 23.” *Id.* (citing Ex. 1002 ¶ 219).

17. Claim 24

Claim 24 depends from claim 22 and has substantially similar scope to claim 6, but uniquely recites “the circuitry is further configured to write the corrected instruction back to the instruction cache.” Petitioner contends that “Itou’s circuitry is configured to perform the operation recited in claim 6—*e.g.*, ‘writing the corrected instruction back to the instruction cache,’” because “Itou explains that **the corrected instruction is written back to the instruction cache.**” Pet. 73 (citing Ex. 1002 ¶ 223; Ex. 1007, 8:42–45 (Itou discloses that “[t]he data in the write target line . . . are updated, while the data in the write non-target line are written intact again to the tag RAM 2.”)).

18. Claim 25

Claim 25 depends from claim 24 and has similar scope to claims 9 and 20, which Petitioner contends “are both disclosed by Haswell and Itou,”

but claim 25 uniquely recites “the circuitry stalls,” as opposed to claim 9 and 20 requiring “the step of stalling.” Pet. 73–74 (citing Ex. 1002 ¶ 225).

Petitioner contends “Haswell and Itou, alone and in combination, disclose claim 25 because their respective circuitries are configured to perform the operation recited in claims 9 and 20.” Pet. 74 (citing Ex. 1002 ¶ 227). First, Petitioner notes that “Haswell discloses that the I-OCM unit 10 swaps in an error-indicating signal 30, and stalls the processor, then writes the corrected instruction back to the instruction cache,” and “I-OCM controller 10 is part of Haswell’s circuitry.” *Id.* (citing Ex. 1006, 7:43–45, 7:62–67; Ex. 1002 ¶ 228).

Next, Petitioner relies on Itou’s disclosure “that when an error is detected in the fetched data from the cache memory, the ECC check unit 4 transmits a process cancel signal to the main control unit that prevents the main control unit from reading data from the cache memory.” *Id.* (citing Ex. 1007, 8:5–9; Ex. 1002 ¶ 230). Petitioner contends that “[t]his signal transmission continues until corrected data (*e.g.*, the “corrected instruction”) is provided (*e.g.*, while the corrected instruction is written back to the instruction cache),” and “ECC check unit 4 is part of Itou’s memory control circuitry shown in Figure 1.” *Id.* (citing Ex. 1007, 8:13–15; Ex. 1002 ¶ 230).

19. Claim 27

Claim 27 depends from claim 26 and recites “the circuitry is further configured to replace the instruction in the instruction cache with a correct version of the instruction obtained from a random access memory if the error is determined not to be correctable.” Petitioner contends that “Haswell and Itou, alone and in combination, disclose claim 27.” Pet. 75 (citing Ex. 1002 ¶ 231).

Petitioner relies first on Haswell's disclosure "that '[i]f the error was not correctable, retrieve the correct original data or instruction from a back-up memory area attached to the PLB.'" Pet. 75 (quoting Ex. 1006, 8:9–11). Based on this disclosure, Petitioner contends "[a] POSITA would understand that Haswell's back-up memory area is the main memory (*e.g.*, a random access memory) commonly found in processor-related architectures," and "that when the instruction has a non-correctable error, this will be treated like a cache miss." *Id.* (citing Ex. 1002 ¶ 232). Further, in situations "when the error is determined not to be correctable, Haswell discloses that the corrected instruction can be retrieved from the main memory to replace the corrupted instruction in the instruction cache (*e.g.*, SRAM 24)." *Id.*

Petitioner also relies on Itou arguing that it "discloses that 'when detecting a correctable error, this memory control circuit executes the error correction in a short period of time without rereading the data from the low-order memory device 11.'" Pet. 75–76 (citing Ex. 1007, 9:15–18; Ex. 1002 ¶ 233). Petitioner argues that "[w]hen the memory control circuit determines that the error is not correctable, it resorts to reading the data from low-order device 11—*e.g.*, replace the instruction in the instruction cache with a correct version of the instruction from the main memory," and that "Itou discloses that low-order device 11 is the main memory of a computer." Pet. 76 (citing Ex. 1002 ¶ 233; Ex. 1007, 5:44–45).

20. Claim 32

Claim 32 depends from claim 31 and recites "means for stalling the processor during the generation of the corrected instruction and during the writing of the corrected instruction back to the instruction cache." Petitioner notes that "[c]laim 32 is substantially the same as the limitations in claims 5 and 9." Pet. 76.

Petitioner relies on its analysis of claims 5 and 9 and also shows how “Haswell discloses that the I-OCM unit 10 swaps in an error-indicating signal, and stalls the processor,” and “[t]herefore, Haswell identifies I-OCM unit 10 as the component that causes the processor to stall during the steps of generating the corrected instruction and writing the corrected instruction back to SRAM 24.” Pet. 78 (citing Ex. 1002 ¶ 239; Ex. 1006, 7:43–45).

Petitioner also relies on Itou disclosure that “an ECC check unit 4 which transmits a process cancel signal (S11) to stall the processor—that is, ECC check unit 4 performs the function of ‘stalling the processor,’” thus the “ECC check unit 4 is the same as, or equivalent to, the respective structure described in the ’619 patent, Itou discloses claim 32.” Pet. 78–79 (citing Ex. 1002 ¶ 242; Ex. 1007, Fig. 1).

21. Reasons for Combining Haswell and Itou

Petitioner contends that “[a] POSITA would have been motivated to combine Haswell and Itou for” several reasons we examine below. Pet. 79–82 (citing Ex. 1002 ¶¶ 243–249).

Petitioner first contends that “both Haswell and Itou are directed to systems and methods for addressing bit errors in data stored within a cache memory operatively coupled to a processor.” *Id.* (citing Ex. 1002 ¶ 244). Petitioner argues that “[t]o achieve this, Haswell and Itou use error correcting code (ECC)-based methods to detect and correct bit errors when data is fetched from the cache memory.” *Id.* Petitioner relies on Haswell’s disclosure that:

In its preferred embodiment, the present invention provides a method and apparatus to allow an instruction on-chip memory unit (I-OCM) associated with a computer processor to implement error correcting code (ECC) to maximize data integrity while minimizing performance overhead.

Pet. 80 (quoting Ex. 1006, 4:29–34) (citing Ex. 1002 ¶ 244). Petitioner relies on Itou’s similar disclosure that “a memory control technology capable of ensuring a high reliability by having a data error correcting function and exhibiting the same performance as an architecture containing no error correcting mechanism.” *Id.* (quoting Ex. 1007, 1:41–44; Ex. 1002, ¶ 244).

Petitioner next contends that “Haswell and Itou offer different ECC-based options for error detection and correction in a cache,” and thus “[a] POSITA could therefore combine teachings from both references to achieve improved error detection and recovery.” *Id.* (citing Ex. 1002 ¶ 245). Petitioner relies on Haswell’s teaching that:

Preferably the computer processor operates on timed, uniform clock cycles, and simultaneously checking the raw data signal for corruption, and transmitting either the raw or predetermined reserved signal to the computer processor, are all performed within a single clock cycle.

Pet. 80 (quoting Ex. 1006, 2:61–67) (citing Ex. 1002 ¶ 245). According to Petitioner, “[t]he error detection for an instruction being fetched occurs **concurrently** with the fetching of the instruction.” *Id.* (citing Ex. 1002 ¶ 245).

Petitioner argues that “Itou discloses that even though the ECC error check occurs in parallel with the fetching of **an** instruction, the ECC error check is performed for the instruction fetched in a prior processing cycle.” Pet. 80–81 (citing Ex. 1002 ¶ 246). Petitioner points to Itou’s disclosure that “[o]n the other hand, according to the memory control circuit in this embodiment, the data are read from the tag RAM (XP1), the ECC check is executed (XP2), and the cancel signal is transmitted when detecting the error (XP3).” Pet. 81 (quoting Ex. 1007, 8:64–67) (citing Ex. 2002 ¶ 246).

“Thus,” according to Petitioner, “a POSITA could choose between the two methods or combine the methods on different situations or cycles according to the system’s needs.” *Id.* Petitioner reasons that “Itou’s method of fetching an instruction in one cycle and checking it in a later cycle may be used where the clock speed is too fast to allow both a cache fetch and error detection of that same instruction in a single clock cycle,” and such “fetching would enable a faster clock—*e.g.*, a shorter clock period.” *Id.*

Petitioner notes that “Itou explains that the disclosed method is not reserved exclusively for tag RAM memory, but can be applied to other types of memories,” and thus, “a POSITA would understand that the methods of Itou can be applicable to the instruction cache of Haswell and vice versa.” Pet. 81 (citing Ex. 1002 ¶ 247).

Finally, Petitioner argues that “applying the teachings of Itou in Haswell, or vice versa, does not require substantial changes,” this is “because both references use known hardware components and architectures.” Pet. 81. According to Petitioner, “the combination of Haswell and Itou would yield predictable results.” Pet. 81–82.

C. Patent Owner’s Contentions

Patent Owner makes several arguments related to the combination of Haswell and Itou, and also presents arguments related to certain limitations within the dependent claims. *See* PO Resp. 3–4, 43–52. We examine each of these arguments in more detail below.

1. A POSITA Would Not Have Combined Haswell and Itou and Had a Reasonable Expectation of Success

Patent Owner first argues that a POSITA would not have combined Haswell and Itou. The first reason given by Patent Owner is that “Itou does not describe detecting errors in instructions retrieved from an instruction

cache,” but instead, “Itou’s ECC is configured to detect errors in tags stored in Itou’s tag RAM, not the instructions stored in Itou’s (unillustrated) data RAM, which are stored in a different memory structure.” PO Resp. 43 (citing Ex. 2003 ¶ 87 (Przybylski Declaration)).

Patent Owner recognizes that Itou states that its error detection process does not require the specific memory control circuit illustrated in Figure 2 of Itou, but instead could be used with other types of memory besides simply tag RAM 2, but Patent Owner asserts that Petitioner is wrong in applying Itou’s methods to the instruction cache of Haswell. *Id.* at 43–44 (citing Ex. 1007, 9:37–40; Ex. 2003 ¶ 88). Although “Itou acknowledges that ‘there have been proposed a multiplicity of techniques for protecting the data on the memory,’ and that ECC can be configured to detect errors in instructions retrieved from an instruction cache,” Patent Owner contends that Itou disparages such a method. *Id.* at 44 (quoting Ex. 1007, 1:23–35). Patent Owner argues that “Itou specifically states that configuring ECC to detect errors in instructions is an inferior method because ‘[t]he memory access time is delayed,’ and [‘][t]his delay is a **serious problem** especially in a cache memory that need high-speed accessing.’” *Id.* (quoting Ex. 1007, 1:31–35). Patent Owner alleges that Itou’s “process is superior because ‘the ECC based data check and the normal operation [*i.e.*, providing instructions from the unillustrated data RAM] are performed in parallel, thereby reducing a cache access time in a way that enhances the reliability.’” *Id.* (quoting Ex. 1007, 9:10–12) (citing Ex. 2003 ¶¶ 89–91).

Patent Owner further contends that “Haswell does not even mention a process for retrieving instructions that incorporates tag data, much less explain how such a process could be applicable to its ECC configuration.” *Id.* Patent Owner argues that “[e]ven if a POSITA sought to modify

Haswell's process, the POSITA would not look to Itou because of the differences in domain and implementation between Haswell and Itou.” *Id.* at 44–45.

Patent Owner next alleges that a POSITA would not have reasonably expected to successfully combine Haswell and Itou. PO Resp. 45. Patent Owner argues that “by failing to provide any detail on how Haswell and Itou would be combined, Petitioner does not meet its burden to show that a POSITA would have reasonably expected to achieve the combination without undue experimentation.” *Id.* (citing Ex. 2003 ¶ 93); *see also* Sur-reply 23 (“By failing to sufficiently explain how a POSA could have modified Haswell to delay error checking, Petitioner failed to demonstrate why a POSA would have had a reasonable expectation of success.”). Patent Owner argues that Petitioner’s expert acknowledged the combination “would require making a new circuit,” and “that it would have been difficult for a POSITA to combine Haswell and Itou using just those references alone.” *Id.* at 45–46 (citing Ex. 2005, 131:5–20, 160:21–25 (Dr. Sechen Deposition); Ex. 2003 ¶ 96). Patent Owner contends that the “combination would have required a complete overhaul of Haswell’s I-OCM circuit—faster clock speed, new processing pipeline, faster processor—in order to incorporate the purportedly advantageous features of Itou.” *Id.* at 47 (citing Ex. 2005, 124:11–23).

Responding to Petitioner’s Reply, Patent Owner further alleges that Petitioner presented improper new arguments explaining how specifically the proposed modification to Haswell would occur, and Patent Owner is prejudiced by its inability to respond. Sur-reply 24.

2. *Itou Does Not Correct Instructions During Certain Claimed Instruction Cycles.*

Patent Owner presents several arguments related to the Itou's purported failure to teach correcting instructions during cycles claimed in specific dependent claims. PO Resp. 48–50. Patent Owner points out that “[d]ependent claims 4, 7, 15, and 18 require generating a corrected instruction in either the third instruction cycle (claims 4 and 7) or the second instruction cycle (claims 15 and 18) after the cycle in which an instruction is fetched.” *Id.* at 48. Patent Owner argues that Figure 5 of Itou only “illustrates the error correction taking place in the *fourth* instruction cycle (XP4).” *Id.* at 48–49. Patent Owner argues that “[a]lthough acknowledging that Itou's error correction process is initiated in cycle XP4, Petitioner does not explain how the error correction process being generated in cycle XP4 would meet the limitations of claims 4, 7, 15 and 18, which require error correction to be performed in either the second or third cycle.” *Id.* at 49–50 (citing Ex. 2003 ¶¶ 102, 103) (quoting Ex. 2005, 68:20–69:10). According to Patent Owner, Petitioner's own expert contradicts Petitioner's assertion that Itou discloses limitations of these dependent claims. *Id.*

3. *Haswell Teaches Away from Stalling a Processor.*

Patent Owner first notes that “[c]laims 5, 9, 16, 23, 25, and 32 require stalling the processor while a corrected instruction is generated.” PO Resp. 50. Patent Owner contends that “Haswell avoids stalling the processor, even if that means adding instruction cycles,” and “[i]n the case of either a correctable or uncorrectable error, instead of stalling the CPU, Haswell modifies the instruction being presented to the CPU (via Mux 28) and causes the CPU to take an interrupt instead.” *Id.* at 51 (citing Ex. 1006, 4:39–42, 1:17–19, 7:37–54; Ex. 2003 ¶¶ 106–108). “Thus,” according to

Patent Owner, “Haswell operates the processor without pausing, stalling, or shutting down.” *Id.*

Patent Owner contrasts Itou, which allegedly “takes the opposite approach and stalls the processor when an error is detected in the tag RAM.” *Id.* (citing Ex. 1007, 5:5–21, 7:65–8:39). Patent Owner relies on Dr. Przybylski’s testimony that an interrupt is an entirely different response to an error than a stall. *Id.* (citing Ex. 2003 ¶ 109). “Thus,” according to Patent Owner, “a POSITA would not have sought to combine a reference like Haswell, which seeks to avoiding stalling the processor if there is an error, with Itou.” *Id.* Patent Owner further argues that Dr. Przybylski does not discuss Haswell in terms of “idling” and “[b]ecause Haswell does not stall the processor, and instead seeks to avoid doing so, Petitioner cannot establish that a POSA would have combined Haswell and Itou to meet the stalling’ limitation.” Sur-reply 24–25.

D. Analysis

Having now considered the evidence in the complete record established during trial, we are persuaded that, based on this record, Petitioner has demonstrated by a preponderance of the evidence that each of claims 2–9, 14–20, 23–25, 27, and 32 would have been obvious over the combination of Haswell and Itou. As a preliminary matter, we have examined Petitioner’s uncontested contentions with respect to these claims, and find them persuasive for the reasons set forth by Petitioner, and as outlined above. *See* Pet. 45–82. Below, we address each of Patent Owner’s arguments and provide our analysis based on the final record.

*1. Reasonable Expectation of Success in Combining
Haswell and Itou*

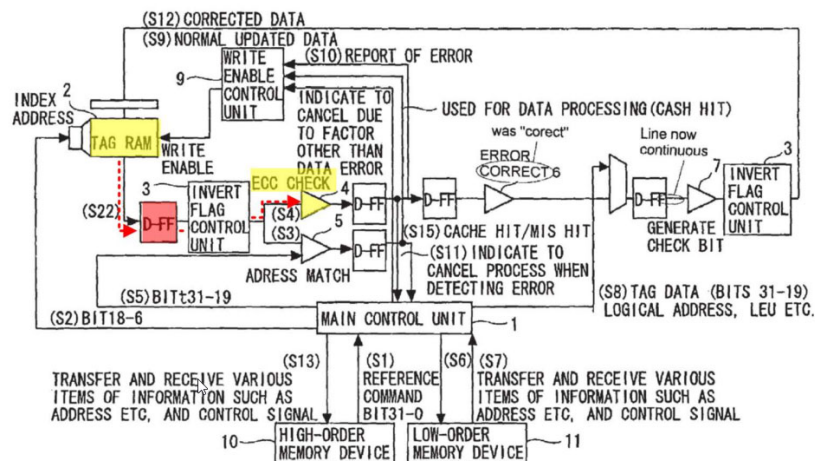
Petitioner's stated rationales and basis for combining Haswell and Itou are supported by a preponderance of the evidence in the final record. *See* Pet. 79–82; Ex. 1002 ¶¶ 243–249. Petitioner has shown, as supported by the testimony of Dr. Sechen, that “delaying checking for an error in data (instruction or tag) to a subsequent cycle, as Itou teaches, as opposed to checking for errors in the same cycle as Haswell teaches can provide the predictable benefit of higher clock speeds and improved performance.” Reply 19 (citing Ex. 1020 ¶¶ 54–55; Ex. 1002 ¶¶ 245–256).

We consider Patent Owner's contention that Itou conveys that configuring ECC to detect errors in instructions is an inferior process and would not work as alleged by Petitioner. PO Resp. 44–45. Despite the drawbacks pointed out by Patent Owner, we determine that a POSITA would reasonably have expected the combination of Haswell and Itou to succeed because Itou first states that its technique is not limited to the tag RAM and can be applied to “a cache memory.” Ex. 1007, 9:37–40. This indication would have reasonably conveyed to the POSITA that Itou's technique can be applied to Haswell's instruction cache SRAM 24 and error detection. *See* Ex. 1020 ¶¶ 56–57. Further, Itou explains that in the case of using the error correcting circuit, the data read from the memory undergo an execution of the error detection and error correction, and after removing error from the data, undergo an execution of the essential processing. Ex. 1007, 1:25–29. We agree with Petitioner that this is the same as what Haswell is doing. Tr. 34:14–26. We also agree with Petitioner that inserting a delay between the fetching and the error detection, as taught by Itou,

would be well within the ability of a person of skill in the art, as explained below by Dr. Sechen.

Dr. Sechen testifies persuasively that “a POSITA would have known that delaying error checking in Haswell’s system by one cycle can be achieved without a significant change to Haswell’s system, e.g., by inserting a register between the SRAM 24 and the ECC GEN/CK unit 26.” *Id.* ¶ 57. Dr. Sechen provides a specific example testifying that “in Itou’s circuitry shown in FIG. 1, a D flip-flop (D-FF), i.e., a register, is inserted between the tag RAM 2 and the ECC check unit 4.” *Id.* (citing Ex. 1007, 7:37–46, Fig. 1).

FIG. 1



The foregoing figure is Dr. Sechen's annotated and highlighted Figure 1 of Itou highlighting a register (D-FF) between the tag RAM 2 and the ECC check unit 4. *Id.* Based on this proposed example configuration, Dr. Sechen testifies persuasively that "a POSITA would have been motivated to combine Haswell and Itou, and would have expected the combination to succeed." *Id.*; *see also* Tr. 35:8–10 ("Itou delays its error checking from fetching by that one clock cycle using a D flip-flop, which is a fairly straightforward logic device used in most architectures.").

We have considered Patent Owner’s contention that Dr. Sechen’s alleged “new testimony” violates our rules. Sur-reply 24 (“presenting a new technical argument that prejudices Patent Owner by foreclosing the opportunity for Dr. Przybylski to assess Dr. Sechen’s new proposed modification to Haswell”). Based on the entire record, however, we find that Dr. Sechen’s testimony was directly responsive to Patent Owner’s Response and to the questions asked to Dr. Sechen during cross examination (Ex. 2005, 117:6–118:17 (“Q. How does a . . . person of ordinary skill in the art combine two circuits?”)). These are questions that Patent Owner also relies upon in its Response. *See* PO Resp. 46 (citing Ex. 2005, 117:24–25). Finally, Patent Owner had an opportunity to address Dr. Sechen’s testimony in its Sur-reply and during the oral hearing.

Patent Owner contends that according to Itou “configuring ECC to detect errors” can introduce a memory access delay, which can be “a *serious problem*.” PO Resp. 44; Ex. 2003 ¶¶ 89–90. We disagree, however, with the Patent Owner’s assessment of Itou and the alleged difficulty of configuring the ECC to detect errors in instructions. We find Dr. Sechen’s testimony persuasive that per Itou, correcting errors, not merely detecting them, can cause delays which can be a serious problem. Ex. 1020 ¶¶ 59–60 (citing Ex. 1007, 1:25–35). Dr. Sechen further explains that Haswell also recognized the same problem, and as one solution, separated error detection from error correction. *Id.* ¶¶ 60–61 (citing Ex. 1006, 7:38–46, 7:54–8:13). Thus, we agree with Petitioner and Dr. Sechen that “in the Haswell-Itou combination, the problem that PO describes does not exist.” Reply 21 (Ex. 1020 ¶¶ 62–63).

We determine that Petitioner has established that Haswell and Itou would be combined without undue experimentation and with a reasonable

expectation of success. *See* Pet. 81–82; Ex. 1002 ¶ 248. Petitioner has established persuasively that “the combination, which requires delaying Haswell’s error checking by one clock cycle, would have been straightforward to a POSITA, using known hardware, e.g., a register.” Reply 21 (citing Ex. 1020 ¶ 64). As explained by Dr. Sechen, the modification requires inserting a register between the SRAM 24 and the ECC GEN/CK unit 26, resulting in new circuitry. Ex. 1020 ¶ 64; Ex. 2005, 117:24–25.

We do not find persuasive Patent Owner’s contentions that combining Haswell and Itou would require more than what is disclosed in those references based on Dr. Sechen’s testimony. PO Resp. 45–48. We understand Dr. Sechen’s testimony cited by Patent Owner to refer to the need for more information in order to produce an actual chip. *See id.* Specifically, Dr. Sechen explained that to generate a chip layout for fabrication more information would be needed, but not for the design of the modified Haswell system. Reply 22 (citing Ex. 1020 ¶ 65; Ex. 2005, 131:4–132:12). It is not necessary that a blueprint be created that allows physical combinability of two references in order to construct a commercial product; instead, the issue is whether a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention. We determine Petitioner has met that standard by a preponderance of the evidence and has explained how Haswell’s and Itou’s teachings could have been combined. *See* Pet. 81–82; Ex. 1020 ¶¶ 57, 66; Ex. 1002 ¶ 248.

2. *Error Correction in Subsequent Cycles – Claims 4, 7, 15, and 18*

As noted above, Patent Owner contends that Itou’s teachings are focused on error correction taking place in the fourth instruction cycle (XP4), and this teaching does not suggest the requirements for claims 4, 7, 15, and 18. PO Resp. 48–49. We have considered Patent Owner’s contentions, but we determine that Petitioner’s position that “Itou teaches the limitations recited in claims 4, 7, 15, and 18, because it teaches performing error detection in one cycle and performing error correction is a **subsequent** cycle,” is more persuasive. Reply 23 (citing Ex. 1020 ¶ 68; Ex. 1007, 8:65–9:8, Fig. 5). More specifically, we find persuasive the position that “Itou’s ‘XP4’ cycle, in which error correction is performed, is the ‘third’ cycle per claims 4 and 7 and the ‘second’ cycle per claims 15 and 18, because the ‘XP4’ cycle is subsequent to the ‘XP2’ cycle in which error detection is performed.” *Id.* at 24 (citing Ex. 1020 ¶ 71).

Petitioner reasons that

In patent claims, the “terms ‘first, second, and third’ are terms to distinguish different elements of the claim, not terms supplying a numerical limit.” *Gillette Co. v. Energizer Holdings, Inc.*, 405 F. 3d 1367, 1373 (2005). The ’619 patent does not state or suggest that “second” means immediately following the “first,” or that “third” means immediately following the “second.”

Id. at 23. Thus, we agree with Petitioner that because claims 4, 7, 15, and 18 recite either that “the third one of the instruction cycles being subsequent to the second one,” or, “the second one of the instruction cycles being subsequent to the first one,” the limitations “subsequent to the second one” and “subsequent to the first one” would be superfluous if the terms “first,”

“second,” and “third” imposed a strict numerical sequence that Patent Owner alleges. *Id.*

3. *Haswell Discloses Stalling a Processor – Claims 5, 9, 16, 23, 25, and 32*

Based on the final record, we agree with Petitioner that Haswell and Itou teach stalling a processor as required by dependent claims 5, 9, 16, 23, 25, and 32.

Petitioner contends that Haswell’s processor is stalled when instructions from the instruction cache are not executed until the error handling routine is fully executed. *See* Pet. 53–54. Petitioner reasons, and we agree, that Haswell’s process is the same as the claimed stalling because “when a particular instruction to be executed is corrupt, that particular instruction cannot be executed, and instead, the processor must wait for the corrupt instruction to be corrected or for the correct instruction to be obtained from a back-up memory.” Reply 24 (citing Ex. 1020 ¶ 72). Specifically, Haswell teaches that “an interrupt is taken” when Haswell’s process attempts to execute the predetermined reserved error indicating signal, and “[t]he interrupt causes the processor to execute an error handling routine instead of executing instructions from the instruction cache.” Pet. 54 (citing Ex. 1002 ¶ 164); Tr. 35:19–26 (“by creating an interrupt, as Haswell has done, it allows the processor to process other instructions or do other things, but it will wait for the instruction that it was expecting through this fetch process”).

We do not agree with Patent Owner’s arguments to the contrary, because the contentions seemingly conflate processor idling and processor stalling. PO Resp. 50–52; Ex. 2003 ¶¶ 105–109. Dr. Sechen explains

persuasively how stalling is different from idling. Ex. 1020 ¶ 73. We find persuasive, Dr. Sechen's contention that:

A processor idles, e.g., pauses, as Dr. Przybylski describes, *see* [Ex. 2003] ¶108, when the processor has no work to do. *See* Ex.1026 at 2. In contrast, a processor is said to be stalled when the processor "has work to do but cannot because of any of a variety of reasons." *Id.* Haswell's processor operates in this manner when it is supposed to execute an instruction, but cannot because the instruction was corrupt, and hence, is replaced by the reserve code 30. Ex.1006, 7:38-46 (stating that "I-OCM unit 10 swaps in a predetermined error-indicating signal or instruction 30 in place of the retrieved corrupted data," i.e., instruction). Upon encountering the reserve code 30, Haswell's processor invokes an error handling routine, *id.*, 7:55-59, but it cannot do the work it was supposed to do, i.e., execute the instruction.

Ex. 1020 ¶ 73.

Patent Owner's expert, Dr. Przybylski cites scenarios in Haswell that involve avoiding stalling. *See* Ex. 2003 ¶ 108 ("Haswell avoids stalling the processor, even if that means adding instruction cycles. . . . Thus, Haswell operates the processor without pausing, stalling, or shutting down."). These scenarios are, however, consistent with Dr. Sechen's testimony.

Specifically, Petitioner shows persuasively how "the erroneous instruction may never be executed due to an immediately preceding jump instruction . . . or the ECC GEN/CK unit 26 may correct the error before the processor encounters the reserved op-code 30," and "[i]n both cases, stalling is avoided." Reply 25 (citing Ex. 1006, 8:18–21; Ex. 1020 ¶¶ 74–75). Thus, we agree with Petitioner that Haswell does teach stalling a processor and a POSITA would combine Haswell and Itou in the manner proposed by Petitioner. Ex. 1020 ¶ 76.

VIII. OBVIOUSNESS OVER HASWELL AND LEMPEL

Petitioner asserts dependent claims 12 and 28 would have been obvious over the combination of Haswell and Lempel. Pet. 82–85. Based on the final record, Petitioner has not shown by a preponderance of the evidence that claims 12 and 28 would have been obvious over the combination of Haswell and Lempel.

A. Overview of Lempel

Lempel is titled “System and Method for Soft Error Handling” and it relates to detecting and clearing a soft error in a cache memory. Ex. 1008, codes (54), (57). Lempel explains that “[s]oft error’ is a term . . . used to describe random corruption of data in computer memory [and that] [s]uch corruption may be caused, for example, by particles [e.g., alpha particles] in normal environmental radiation.” *Id.* at 1:12–17. As depicted below in the system diagram of Figure 1, “the cache 102 may be coupled to or include a soft error detection logic 104. The soft error detection logic may be configured to detect soft errors in data loaded into the cache, for example by parity checking.” *Id.* at 2:22–26.

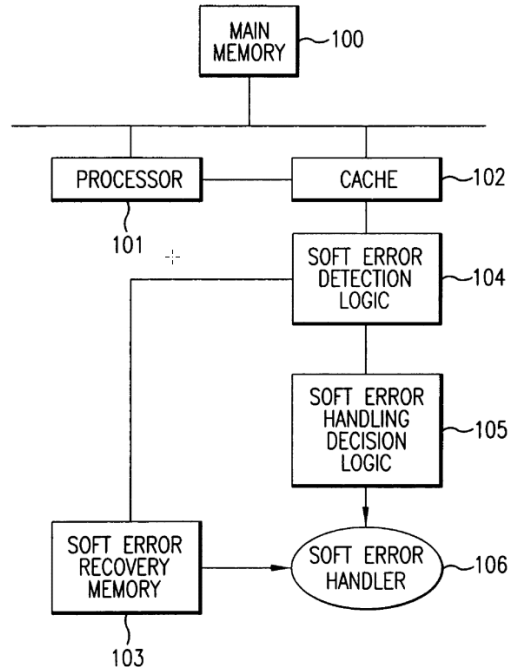


FIG. 1

Figure 1 of Lempel depicts a system diagram showing main memory 100 coupled to processor or CPU 101 and cache 102, which may also be coupled to processor 101. *Id.* at 2:13–16.

In the Background of Lempel, it discusses known methods of handling soft errors in caches, including shutting down operations and also not shutting down operations but instead using error correction circuitry (“ECC”). Ex. 1008, 1:18–45. With regard to ECC methods, Lempel describes how

ECC is additional hardware logic built into a cache; the logic is able to detect soft errors and execute a hardware algorithm to correct them. However, a disadvantage of ECC is that the additional hardware takes up space on the silicon and requires time to perform the needed computations, imposing further area and timing constraints on the overall design.

Id. Further, Lempel notes that “an additional cycle is usually added to the cache access time in order to accommodate the ECC’s soft error correction

logic, *adversely impacting processor performance even when no soft errors are detected.*” *Id.* (emphasis added). Lempel notes that a distinct “approach is needed for handling soft errors in view of the foregoing considerations.” *Id.*

In Lempel’s process, when a soft error is detected, the “normal operations . . . such as instruction fetch, may be paused while microcode . . . executes to clear the soft error.” *Id.* at 2:4–12, 2:40–3:17 (“operations of the front end of the processor, which is typically involved in such pre-execution operations as instruction fetch, may be paused”). While the processor is paused, microcode is executed to clear (i.e., remove) the error from the instruction cache. *Id.*

B. Analysis

Because we determine that a POSITA would have recognized that Haswell and Lempel provide markedly different (and likely incompatible) solutions to the same problem, we determine that a POSITA would not have combined these two references as Petitioner proposes to arrive at the invention of claims 12 and 28. We focus our analysis below on this singular dispositive issue.

Claim 12 depends from claim 10 and recites “the step of replacing the instruction in the instruction cache with the correct version of the instruction obtained from the random access memory is initiated by mimicking a cache miss condition.” Claim 28 depends from claim 26 and recites “the operation of the portion of the circuitry that replaces the instruction in the instruction cache with a correct version of the instruction obtained from a random access memory if the error is determined not to be correctable is initiated by mimicking a cache miss condition.” Petitioner relies on Lempel as teaching

the claim requirements of “mimicking a cache miss condition.” *See* Pet. 82–83.

Petitioner provides three reasons why a person of ordinary skill in the art would have combined Haswell and Lempel: (1) “both Haswell and Lempel are directed to systems and methods for addressing bit errors in data stored within a cache memory operatively coupled to a processor;” (2) “a POSITA could use the error detection and recovery method of Haswell (e.g., an ECC (Error Check Code)-based method) when a correctable error is detected, and the error detection and recovery method of Lempel . . . when a non-correctable error is detected;” and (3) “applying the teachings of Haswell and Lempel, or vice versa, does not require substantial changes.” Pet. 84–85.

Patent Owner contends that “Lempel . . . teaches away from using error correction circuitry (referenced in Lempel as ‘ECC’),” because “Lempel advised avoiding ECC approaches to error detection altogether, due to the timing, performance, space, and cost limitations of then-existing ECC hardware. (Ex. 1008 at 1:31-44, 1:58-61).” PO Resp. 52. Patent Owner notes that Haswell set out to improve upon the known strategy of adding an error-correcting circuit (ECC) to the main memory by performing error detection on instructions prior to them being fetched by the processor, and replacing detected errors with a predetermined data signal to avoid stalling the processor, but Lempel discourages such techniques. *Id.* at 52–53 (citing Ex. 1006, 2:29–36).

Patent Owner contends, and we agree, that “Lempel sought to improve upon the known method of using parity checking to detect errors by avoiding the need to shut down the processor when errors are detected. (Ex. 1008 at 1:20-30).” *Id.* Further, “Lempel did not accomplish this by

performing error detection in parallel with the processor's operation, but rather by pausing processor operations when an error is detected and allowing microcode to clear the error from the instruction cache.” *Id.* at 53.

Lempel distinguishes its parity based method of handling errors in caches from that process used in Haswell known as error correction circuitry (ECC). Ex. 1008, 1:15–43. According to Lempel, ECC involves “additional hardware logic,” that is a disadvantage because it “takes up space on the silicon and requires time to perform the needed computations, imposing further area and timing constraints on the overall design.” *Id.* at 1:33–39. Further, Lempel describes the “additional cycle” needed to execute an ECC method and “to accommodate the ECC’s soft error correction logic” as a further drawback because this cycle would adversely impact the processor performance even when no soft errors are detected. *Id.* at 1:39–43.

Based on the record before us, including Lempel’s clear desire to distinguish its parity method from ECC based approaches, we agree with Patent Owner that “[a] POSITA would have recognized that Haswell and Lempel provide markedly different . . . solutions to the same problem: detecting errors caused by bit flipping.” PO Resp. 53 (citing Ex. 2003 ¶ 114). A POSITA would not have looked to Lempel at the time of invention for solutions in improving Haswell’s ECC based approach. This is so because “[a] POSITA would have recognized that Lempel discourages ECC approaches such as Haswell’s ECC circuitry altogether by stating several reasons why parity checking is superior—namely, that ECC imposes area and timing constraints on the circuit design, adversely impacts processor performance, and costs more.” *Id.* (citing Ex. 1008, 1:31–44, 1:58–61; Ex. 2003 ¶ 114). Indeed, a POSITA would have been further discouraged from looking to Lempel’s parity checking approach because

such approaches are generally seen as incapable of correcting an error, whereas the data is invalidated and data is read afresh from a data source, which incidentally causes significant delay. Ex. 2003 ¶ 114; *see also* Tr. 36:20–25 (Petitioner’s counsel stating “with Lempel, they don’t do a correction of the data. Every single time there is a problem with the data, it will go back to the backup memory to get the original instruction -- or the original data.”).

We have considered Petitioner’s contentions, and the testimony of Dr. Sechen, that “Lempel discloses the same sequence of operations,” and “[i]n particular, upon detecting a ‘soft error’ at block 201, the ‘processor may be stalled . . . a microcode sequencer may be invoked, and certain microcode may be executed.” Reply 26 (citing Ex. 1020 ¶¶ 77–79; Ex. 1008, 2:40–57, Fig. 2). Petitioner also contends that “Lempel’s stalling of the processor is similar to Haswell’s behavior of not executing the instruction that was supposed to be executed.” *Id.* Petitioner provides additional reasons why a POSITA would purportedly not be dissuaded from combining Haswell and Lempel based on the incompatibilities described. *Id.* at 27–28. We find Petitioner’s arguments and testimony unpersuasive.

More specifically, “Petitioner relies on functionality that Haswell already provides, such as error detection and correcting a correctable error in hardware,” yet, “Lempel has no role in providing these capabilities to the proposed Haswell-Lempel combination.” Sur-reply 25. “Thus,” we agree with Patent Owner that, “Lempel offers nothing that would have motivated a POSA to ignore the incompatibilities between Haswell and Lempel’s approaches to error detection.” *Id.*

Accordingly, Petitioner has not established by a preponderance of the evidence that claims 12 and 28 would have been obvious over the combination of Haswell and Lempel.

IX. CONCLUSION

In summary:⁵

Claims	35 U.S.C. §	Reference(s)/ Basis	Claims Shown Unpatentable	Claims Not Shown Unpatentable
1, 10, 11, 13, 21, 22, 24, 26, 29– 31	102	Haswell	1, 10, 11, 13, 21, 22, 24, 26, 29–31	
1, 10, 11, 13, 21, 22, 24, 26, 29– 31	103 ⁶	Haswell		
2–9, 14–20, 23–25, 27, 32	103	Haswell, Itou	2–9, 14–20, 23–25, 27, 32	
12, 28	103	Haswell, Lempel		12, 28
Overall Outcome			1–11, 13–27, 29–32	12, 28

⁵ Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this decision, we draw Patent Owner's attention to the April 2019 *Notice Regarding Options for Amendments by Patent Owner Through Reissue or Reexamination During a Pending AIA Trial Proceeding*. See 84 Fed. Reg. 16654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. See 37 C.F.R. § 42.8(a)(3), (b)(2).

⁶ As explained above, because we conclude that the challenged claims are unpatentable on other grounds, we do not reach the merits of this ground.

X. ORDER

Upon consideration of the record before us, it is:

ORDERED that, based on a preponderance of the evidence, claims 1–11, 13–27, and 29–32 of the '619 patent have been shown to be unpatentable;

ORDERED that, based on a preponderance of the evidence, claims 12 and 28 of the '619 patent have not been shown to be unpatentable; and

FURTHER ORDERED that, because this is a final written decision, parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

IPR2021-01042
Patent 7,302,619 B1

FOR PETITIONER:

Sanjeet Dutta
Suhrid Wadekar
GOODWIN PROCTER LLP
sdutta@goodwinlaw.com
swadekar@goodwinlaw.com

FOR PATENT OWNER:

Justin Oliver
Sarah Brooks
Seth Boeshore
VENABLE LLP
joliver@venable.com
ssbrooks@venable.com
sboeshore@venable.com