

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

APPLE INC., HTC CORPORATION, HTC AMERICA, INC.,
SAMSUNG ELECTRONICS CO. LTD,
SAMSUNG ELECTRONICS AMERICA, INC., AMAZON.COM, INC.,
SONY CORP., SONY ELECTRONICS INC.,
SONY MOBILE COMMUNICATIONS AB,
SONY MOBILE COMMUNICATIONS (USA) INC.,
LG ELECTRONICS, INC., LG ELECTRONICS USA, INC., and
LG ELECTRONICS MOBILECOMM USA, INC.,
Petitioner,

v.

MEMORY INTEGRITY, LLC,
Patent Owner.

Case IPR2015-00159¹
Patent 7,296,121 B2

Before JENNIFER S. BISK, NEIL T. POWELL, and KERRY BEGLEY,
Administrative Patent Judges.

BEGLEY, *Administrative Patent Judge.*

FINAL WRITTEN DECISION
35 U.S.C. § 318(a) and 37 C.F.R. § 42.73

¹ Sony Corp., Sony Electronics Inc., Sony Mobile Communications AB, Sony Mobile Communications (USA) Inc., LG Electronics, Inc., LG Electronics USA, Inc., and LG Electronics Mobilecomm USA, Inc., who filed a Petition in IPR2015-01376, have been joined as petitioners in the instant proceeding.

Apple Inc., HTC Corporation, HTC America, Inc., Samsung Electronics Co. Ltd., Samsung Electronics America, Inc.,² and Amazon.com, Inc. (collectively, “Initial Petitioners”) filed a Petition requesting *inter partes* review of claims 1–3, 8, and 11–25 of U.S. Patent No. 7,296,121 B2 (Ex. 1001, “the ’121 patent”). Pet. Pursuant to 35 U.S.C. § 314(a), we determined the Petition showed a reasonable likelihood that Petitioner would prevail in establishing the unpatentability of claims 1–3, 8, 11, and 15–25, and instituted an *inter partes* review of these claims. Paper 12 (“Inst. Dec.”). We, however, did not institute review of claims 12–14, because we determined the Petition did not show a reasonable likelihood that Petitioner would prevail with respect to these claims. *Id.* at 23–30.

After institution, Sony Corp., Sony Electronics Inc., Sony Mobile Communications AB, Sony Mobile Communications (USA) Inc., LG Electronics, Inc., LG Electronics USA, Inc., and LG Electronics Mobilecomm USA, Inc. (collectively, “Subsequent Petitioners”; and with Initial Petitioners, “Petitioner”) filed a Petition in IPR2015-01376, requesting *inter partes* review of claims 1–3, 8, 11, 12,³ and 15–25 of the

² The Petition also lists Samsung Telecommunications America, LLC (“STA”) as a petitioner. Paper 6 (“Pet.”), 1. After the filing of the Petition, however, STA merged with and into Samsung Electronics America, Inc. Paper 10. Thus, STA no longer exists as a separate corporate entity. *Id.*

³ Subsequent Petitioners represented that they included claim 12 “merely to conform” to the Petition and motion for rehearing of the Institution Decision, regarding claim 12, that was pending before the Board and that if the motion was denied, they requested joinder on “all claims except claim 12.” IPR2015-01376, Paper 3 (“IPR2015-01376 Pet.”), 1 n.1, 33 n.5. Because we denied the motion, we understood Subsequent Petitioners to no longer maintain their challenge of claim 12 and to the extent they did, we determined they had not shown the IPR2015-01376 Petition warranted institution of review of the claim. IPR2015-01376, Paper 12, at 12–15.

IPR2015-00159

Patent 7,296,121 B2

'121 patent on the same grounds as those instituted in this proceeding.

IPR2015-01376 Pet. Subsequent Petitioners also filed a motion for joinder with this proceeding, which we granted. IPR2015-01376, Papers 4, 12.

Patent Owner Memory Integrity, LLC ("Patent Owner") filed a Patent Owner Response (Paper 25 ("PO Resp.")) and a Motion to Amend (Paper 26 ("Mot.")). Petitioner filed a Reply to Patent Owner's Response (Paper 35, "Reply") and an Opposition to Patent Owner's Motion to Amend (Paper 36, "Opp."). Patent Owner then filed a Reply in support of its Motion to Amend (Paper 37, "Mot. Reply"). Petitioner also filed a Motion for Observations on the deposition testimony of Patent Owner's expert. Paper 41. An oral hearing was held before the Board. Paper 45 ("Tr.").

We issue this Final Written Decision pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73. Having considered the record before us, we determine Petitioner has shown by a preponderance of the evidence that claims 1–3, 8, and 15–25 of the '121 patent are unpatentable. See 35 U.S.C. § 316(e). Petitioner, however, has not demonstrated by a preponderance of the evidence that claim 11 is unpatentable.

I. BACKGROUND

A. RELATED PROCEEDINGS

The parties indicate Patent Owner has asserted the '121 patent in numerous cases filed in the U.S. District Court for the District of Delaware. Pet. 1–2; Paper 8, 1–2. In addition, the '121 patent was the subject of several petitions for *inter partes* review before the Office—IPR2015-00158, IPR2015-00161, IPR2015-00163, IPR2015-00172, and IPR2015-01353. See Paper 8, 4; IPR2015-00163, Paper 34. Of these proceedings, only IPR2015-00163 is ongoing and a final written decision in IPR2015-00163 is being issued concurrently with this Decision.

B. THE '121 PATENT

The '121 patent relates to techniques to reduce memory transaction traffic and to improve data access and cache coherency in systems with multiple processors connected using point-to-point links. Ex. 1001, 1:22–25, 2:39–51. The '121 patent explains that cache coherency problems can arise in a system with multiple processors, each with an individual cache memory, because the system may contain multiple copies of the same data. *Id.* at 1:26–45.

The '121 patent discloses a computer system with processing nodes, each with a cache memory, connected by a point-to-point architecture. *Id.* at [57], 2:48–62. The system also includes a “probe filtering unit” that can receive a probe from a processing node. *Id.* at [57], 2:52–65, 5:45–47. The '121 patent defines a probe as “[a] mechanism for eliciting a response from a node to maintain cache coherency in a system.” *Id.* at 5:45–47.

The probe filtering unit then can evaluate the probe based on probe filtering information and transmit the probe to selected processing nodes. *Id.* at [57], 2:52–3:5, 14:50–52; *see id.* at 28:29–58, 29:43–46. The '121 patent explains that probe filtering information is “[a]ny criterion that can be used to reduce the number of clusters or nodes probed.” *Id.* at 14:50–52.

The probe filtering unit also may be operable to accumulate responses from the selected processing nodes and to respond to the node from which the probe originated. *Id.* at 3:5–8, 28:59–67, 29:46–51. Figure 18 of the patent is reproduced below.

Figure 18

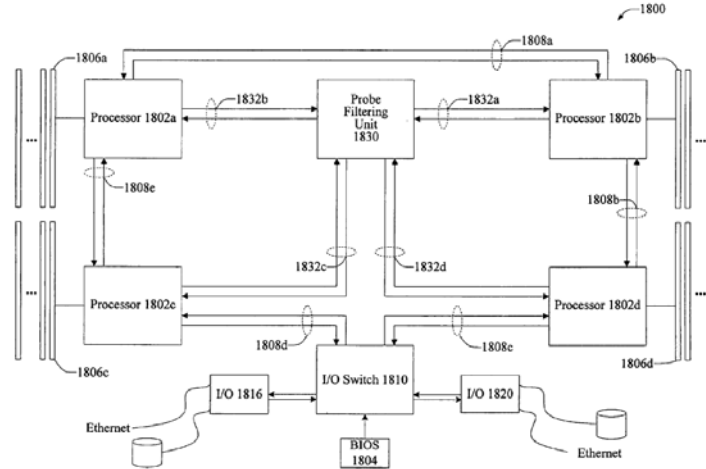


Figure 18 is a diagrammatic representation of a multiple processor system with a probe filtering unit. *Id.* at 3:61–63, 26:58–27:20, Fig. 18. Specifically, Figure 18 depicts multiple processor system 1800 with processing nodes 1802a–d interconnected by point-to-point communication links 1808a–e. *Id.* at 26:58–27:1. System 1800 also includes probe filtering unit 1830. *Id.* at 3:61–63, 26:58–27:20, Fig. 18.

Claims 1, 16, and 25 of the '121 patent are independent claims.

Claim 1 is illustrative of the claimed subject matter and recites:

1. A computer system comprising a plurality of processing nodes interconnected by a first point-to-point architecture, each processing node having a cache memory associated therewith, the computer system further comprising a probe filtering unit which is operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories.

Id. at 30:65–31:7 (line breaks added).

C. INSTITUTED GROUNDS OF UNPATENTABILITY

We instituted *inter partes* review of the '121 patent on the following grounds of unpatentability asserted in the Petition. Inst. Dec. 30.

Claims	Basis	Reference[s]
1–3, 8, 11, 15, 16, 25	§ 102 ⁴	U.S. Patent Application Pub. No. 2002/0053004 A1 (published May 2, 2002) (Ex. 1003, “Pong”)
17–24	§ 103	Pong and MICHAEL JOHN SEBASTIAN SMITH, APPLICATION-SPECIFIC INTEGRATED CIRCUITS (1997) (Ex. 1008, “Smith”)

Petitioner supports its challenge with Declarations executed by Dr. Robert Horst on October 28, 2014 (Ex. 1014) and on December 1, 2015 (Ex. 1025). Patent Owner relies on a Declaration executed by Dr. Vojin Oklobdzija on August 11, 2015 (Ex. 2016).

II. UNPATENTABILITY ANALYSIS

A. LEVEL OF ORDINARY SKILL IN THE ART

We begin our analysis by addressing the level of ordinary skill in the art. Dr. Horst and Dr. Oklobdzija agree that a person of ordinary skill in the art would have had at least a “bachelor’s degree in electrical engineering, computer engineering, or computer science” and “two years of experience in the design of multiprocessor systems.” Ex. 1014 ¶ 8; Ex. 2016 ¶ 8. We adopt this proposal as the level of ordinary skill in the art, based on the testimony of the parties’ experts as well as our review of the '121 patent and the prior art involved in this proceeding.

⁴ The Leahy-Smith America Invents Act (“AIA”), Pub. L. No. 112–29 (2011), revised 35 U.S.C. §§ 102–103, effective March 16, 2013. Because the '121 patent has an effective filing date before this date, we refer to the pre-AIA versions of §§ 102 and 103.

B. CLAIM INTERPRETATION

We next address the meaning of the claims. We interpret claims in an unexpired patent using the “broadest reasonable construction in light of the specification of the patent.” 37 C.F.R. § 42.100(b); *see Cuozzo Speed Techs., LLC v. Lee*, No. 15–446, slip op. at 12–20 (S.C. June 20, 2016) (holding that 37 C.F.R. § 42.100(b) “represents a reasonable exercise of the rulemaking authority that Congress delegated to the . . . Office”). Under this standard, we presume a claim term carries its “ordinary and customary meaning,” which “is the meaning that the term would have to a person of ordinary skill in the art” at the time of the invention. *In re Translogic Tech., Inc.*, 504 F.3d 1249, 1257 (Fed. Cir. 2007). This presumption is rebutted when the patentee acts as a lexicographer by giving the term a particular meaning in the specification with “reasonable clarity, deliberateness, and precision.” *In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

Petitioner and Patent Owner each proffer proposed constructions of various claim terms. Pet. 4–15; PO Resp. 1–17; Reply 1–7. In the Institution Decision, we construed several terms, including “states associated with selected ones of the cache memories.” Inst. Dec. 5–14. The parties’ post-institution briefing presents arguments regarding our construction of “states” (independent claims 1, 16, and 25), in addition to the proper construction of “programmed” (claim 11), “accumulate responses to each probe” (claim 15), and “accumulating probe responses” (claim 25). *See* PO Resp. 1–17; Reply 1–7, 23. We address below the parties’ arguments regarding these terms. We otherwise maintain our constructions from the Institution Decision.

1. “[S]TATES” (CLAIMS 1, 16, AND 25)

Independent claims 1, 16, and 25 recite “probe filtering information” “representative of *states* associated with selected ones of the cache memories.” Ex. 1001, 31:5–7, 32:14–15, 32:51–55 (emphasis added). Before our Institution Decision, each party proposed a construction of the term “states associated with selected ones of the cache memories.” Pet. 9–10; IPR2015-00159, Paper 11 (“Prelim. Resp.”), 13–24. Petitioner argued that the term is “broad enough to encompass ‘any modes or conditions of selected ones of the cache memories.’” Pet. 10. Patent Owner proposed that the term means “‘cache coherence protocol states associated with data blocks stored in selected ones of the cache memories’” and that a “‘cache coherence protocol state’ means ‘the current state of a data block in a protocol used to maintain the coherency of caches, in which a data block can only be in one current state at a time, and in which the current state can transition to a different state upon one or more triggering events or conditions.’” Prelim. Resp. 13–14. In the Institution Decision, we did not adopt either party’s proposed construction, but found that “the term is not limited to cache coherence protocol states and is broad enough to include the condition of presence—i.e., what is stored in cache memory.” Inst. Dec. 10.

In its Response, Patent Owner continues to argue that “the appropriate construction of states is limited to cache coherence states, and does not include mere presence.” PO Resp. 2. Petitioner does not agree that the term should be so limited. Reply 1–4. In particular, Petitioner asserts that the broadest reasonable construction of the term “states” is not limited to cache coherency states, *id.* at 2–4, and is “broad enough to encompass the condition of presence,” *id.* at 4.

a. CACHE COHERENCE STATES

The language of the independent claims “states associated with selected ones of the cache memories” plainly links the “states” to “cache memories.” Ex. 1001, 31:5–7, 32:14–15, 32:52–55. In addition, in these claims, the term “representative of states associated with selected ones of the cache memories” modifies “*probe filtering information*” (*id.* (emphasis added)), which the patent defines as “[a]ny criterion that can be used to reduce the number of clusters or nodes probed” (*id.* at 14:50–52). Thus, the recited “states” relate, not just to any aspect of the cache memory, but to the contents of that memory. See Inst. Dec. 7–8.

For the reasons discussed below, however, despite the arguments and evidence in Patent Owner’s Response, we remain unpersuaded that the ’121 patent supports limiting the broadest reasonable construction of “states” solely to cache coherence protocol states. A claim term will be interpreted more narrowly than its ordinary and customary meaning only under two circumstances: (1) the “patentee sets out a definition and acts as [its] own lexicographer,” or (2) the “patentee disavows the full scope of a claim term either in the specification or during prosecution.” *Aventis Pharma S.A. v. Hospira, Inc.*, 675 F.3d 1324, 1330 (Fed. Cir. 2012). To disavow claim scope, the specification or prosecution history must “make[] clear that the invention does not include a particular feature” and the feature is then “deemed to be outside the reach of the claims of the patent, even though the language of the claims, read without reference to the specification” or prosecution history, “might be considered broad enough to encompass the feature in question.” *SciMed Life Sys., Inc. v. Advanced Cardiovascular Sys., Inc.*, 242 F.3d 1337, 1341 (Fed. Cir. 2001); see *Aventis*, 675 F.3d at 1330. To disavow claim scope, the patentee may “includ[e] in the

specification expressions of manifest exclusion or restriction, representing a clear disavowal of claim scope.” *Aventis*, 675 F.3d at 1330 (internal quotations omitted). In this context, it is not sufficient “that the only embodiments, or all of the embodiments, contain a particular limitation.” *Id.*

Here, the relevant language in the independent claims, “states associated with selected ones of the cache memories,” expressly recites “states” alone—not cache coherency states, to which Patent Owner seeks to limit the term. As Petitioner points out, Patent Owner’s proposed construction seeks to add additional narrowing descriptive language to the term “states.” *See* Reply 2.

Moreover, the claims do not recite “cache coherence states” or “cache coherence protocol states.” Dependent claim 3, which depends indirectly from claim 1, however, recites “a cache coherence controller” and “a cache coherence directory.” Ex. 1001, 31:12–14. Similarly, claim 5, another claim that depends indirectly from claim 1, requires a “cache coherence controller.” *Id.* at 31:24. Thus, had the patentees intended to limit “states,” as recited in the independent claims of the ’121 patent, to cache coherence states, they demonstratively could have done so by explicitly modifying the disputed term with “cache coherence”—but did not.⁵ *See* Reply 2–3.

⁵ Patent Owner notes that the Institution Decision “preliminarily determined that ‘states’ in the claims of the ’121 Patent are not limited to ‘cache coherence protocol states,’ “despite the fact that the Board determined that the term ‘probe’ . . . should be construed as a ‘mechanism for eliciting a response from a node to maintain cache coherency in a system.’”

PO Resp. 1–2. We do not agree with Patent Owner’s implication that our construction of “probe” conflicts with our construction of “states.” The two words recite different parts of the claimed invention. Also, the ’121 patent expressly defines “probe” (*see* Ex. 1001, 5:45–47; Inst. Dec. 6), but not “states.” If “states” were intended to be limited to cache coherency protocol

Turning to the written description, we do not find persuasive Patent Owner's arguments that the remainder of the specification supports limiting "states" to cache coherency states. Rather, we agree with Petitioner that the '121 patent uses broad language in describing "states," explaining that "particular implementations may use a different set of states" and "[t]he techniques of the present invention can be used with a variety of different possible memory line states." Ex. 1001, 14:30–36; *see* Inst. Dec. 9; Pet. 9; Reply 2.

Patent Owner asserts that the teachings of the '121 patent make clear that its inventions are directed to the specific field of cache coherency and the term "state" has "a specific meaning in the field of cache coherency—a cache coherency state." PO Resp. 3–4; *see* Tr. 63:13–64:2. As to the field of the '121 patent, we find that it is directed, generally, to "data access and cache coherency in systems having multiple processors." *E.g.*, Ex. 1001, 2:39–42. The '121 patent explains that data access, and the disclosed invention, involve techniques for reducing probe traffic as well as cache coherency techniques. *See, e.g., id.* at 1:21–27 ("The present invention relates to accessing data in a multiple processor system. More specifically, the present invention provides techniques for reducing memory transaction traffic in a multiple processor system. Data access in multiple processor systems can raise issues relating to cache coherency."); *see also, e.g., id.* at [54] (title) ("Reducing Probe Traffic in Multiprocessor Systems"); *id.* at 2:45–48 ("According to the present invention, various techniques are

states, the '121 patent could have provided an express definition for "states," as it does for "probes." Patent Owner does not provide evidence or reasoning persuading us that one of ordinary skill would find the express definition of "probes" as somehow limiting the term "states."

provided for reducing traffic relating to memory transactions in multi-processor systems.”). Although we agree with Patent Owner that the field of the ’121 patent *includes* cache coherency, we are not persuaded that this fact alone limits the term “state” to “cache coherence states.”

Patent Owner, in fact, concedes that the term “state” “may have many broad and different meanings . . . in the general field of computers.”

PO Resp. 3 (citing Ex. 2016 ¶ 15). Indeed, in our Institution Decision, we relied on a dictionary definition of “state” from the MICROSOFT COMPUTER DICTIONARY (5th ed. 2002): “[t]he condition at a particular time of any of numerous elements of computing—a device, a communications channel, a network station, a program, a bit, or other element—used to report on or to control computer operations.” Ex. 3001, 497–98. Patent Owner agrees that this dictionary is directed “to the entire field of computing.” PO Resp. 3; Reply 4. And as Petitioner points out, Patent Owner relies on this same dictionary when proposing a construction for another term in the ’121 patent—“programmed.” See PO Resp. 13; Reply 4. We disagree with Patent Owner’s contention that a person of ordinary skill in the art would not base its definition of the term “states” on the field of computers generally, but instead would rely on a meaning specific to the “field of cache coherency.”

To begin with, Patent Owner agrees that a person of ordinary skill in the art would have a degree in electrical engineering, computer engineering, or computer science and at least two years of experience in the design of multiprocessor systems. Ex. 1014 ¶ 8; Ex. 2016 ¶ 8. Nothing in this definition points to a specific field, known as cache coherency, with its own terminology displacing the more general terminology used by those in the field of computing.

Patent Owner relies on a few excerpts of the '121 patent, which allegedly “demonstrate that the use of the term ‘state’ in the patent is directed to cache coherence protocol states.”⁶ PO Resp. 5–6. For example, Patent Owner points to the following passage as “mak[ing] it clear that the relevant state is a cache coherence protocol state” (*id.* at 5):

It should be noted that a *coherence protocol* can contain several types of messages. In one example, a coherence protocol includes four types of messages; data or cache access requests, probes, responses or probe responses, and data packets. Data or cache access requests usually target the home node memory controller. Probes are used to query each cache in the system. The probe packet can carry information that allows the caches to properly *transition the cache state for a specified line*.

Ex. 1001, 9:21–29 (emphases added); see PO Resp. 5 (quoting Ex. 1001, 9:21–29). Similarly, Patent Owner cites to the specification’s statement that “[b]y using a *coherence directory*, global *memory line state information* (with respect to each cluster) can be maintained and accessed by a memory controller or a cache coherence controller in a particular cluster,” asserting

⁶ We note that in IPR2015-00163, Patent Owner’s arguments addressing Exhibit 1009 (“Koster”) belie its position that “state” necessarily refers to a cache coherency state in the context of the '121 patent. In IPR2015-00163, Patent Owner argues that the “mere fact” that Koster refers to his shadow tag memory as ‘local state memory’ does not mean that it contains ‘information representative of states associated with selected ones of the cache memories,’” as used in the '121 patent. IPR2015-00163, Paper 31, 23. Patent Owner concludes that the tags “are not representative of cache coherency states.” *Id.* Essentially, Patent Owner argues in its proposed construction of “states” in IPR2015-00163 and this case that a person of ordinary skill in the art would understand the term “state” alone means “cache coherency state” in the context of the '121 patent. PO Resp. 2–6; IPR2015-00163, Paper 31, 2–7. Yet, in IPR2015-00163, when analyzing Koster, which involves the same field as the '121 patent, Patent Owner asserts that the same person would understand the term “state” alone does *not* mean cache coherency state, but instead means something broader.

that this statement only makes sense if the coherence directory concerns solely cache coherence states. PO Resp. 5–6 (quoting Ex. 1001, 13:4–7) (emphases added by Patent Owner).

We are not persuaded that these passages of the '121 patent limit the term “states” as asserted by Patent Owner. Neither of the passages relied upon by Patent Owner actually uses the term “state” as recited in the challenged claims. Instead, the first passage uses the term “*cache state*” and the second uses the term “*global memory line state information*.” Thus, even if the passages describe a concept narrower than “states associated with selected ones of the cache memories,” as recited in the challenged claims, this difference can be attributed to the fact that different terms are used. More importantly, these passages do not expressly disclaim or disavow the broader scope of the claim language, particularly given the expansive language used elsewhere in the specification allowing states to include “a variety of different possible memory line states.” Ex. 1001, 14:30–36.

Patent Owner also points to Figures 7 and 8 of the '121 patent as allegedly “strongly illustrative that the '121 patent uses ‘state’ to mean cache coherence protocol states.” PO Resp. 6. According to Patent Owner, the specification equates the word “state” with cache coherence states by disclosing that “*the coherence directory 701 [of Figure 7] includes state information 713*” and by stating “[i]n some embodiments, the memory line states are modified, owned, shared, and invalid.” *Id.* (quoting Ex. 1001, 13:55–59) (emphasis added by Patent Owner). In other words, Patent Owner argues that because Figure 7 shows a column labeled “state,” and describes this column as including in some embodiments the states used in common cache coherence protocols, such as MOESI and MOSI, the term

“state” must be equivalent to cache coherence protocol states. *See* Tr. 77:20–78:18.

Figures 7 and 8, however, also are not persuasive as defining or limiting the term “state” because they are clearly described as exemplary embodiments. *See, e.g.*, Ex. 1001, 3:15–18, 4:11–35, 13:44–59 (describing Figure 7 using the term “example” or “embodiment” at least five times), 14:48–50, 30:57–64. We are not persuaded that Figure 7 shows anything more than what it purports to show—one example with a column labeled “state,” that may refer to “memory line states” of “modified, owned, shared, and invalid.” Again, nothing in this example expressly disclaims or disavows the broad claim language, particularly in light of other statements in the specification allowing states to include “a variety of different possible memory line states.” *Id.* at 14:30–36.

Patent Owner also proffers extrinsic evidence to support its proposed construction of the term “state.” Extrinsic evidence is “less significant than the intrinsic record in determining the legally operative meaning of claim language.” *Phillips v. AWH Corp.*, 415 F.3d 1303, 1317 (Fed. Cir. 2005) (internal citations and quotations omitted). For example, Patent Owner points to “one of the treatises on cache coherency,” DANIEL J. SORIN ET AL., A PRIMER ON MEMORY CONSISTENCY AND CACHE COHERENCE (2011) (Ex. 2010, “Sorin”), as equating the term “state” with cache coherence protocol states. PO Resp. 4 (citing Ex. 2010, 88–89; Ex. 2016 ¶ 15). We do not find this evidence persuasive. Evidence of the use of shorthand within one section in one publication does not indicate that that same shorthand will be recognized, by a person of ordinary skill, as necessarily having the same meaning in other contexts.

Patent Owner also points to another article, where the authors—three of whom are the authors of Sorin, discussed above—state that “[a] processor’s access to a cache block is determined by the state of that block in its cache, and this state is *generally* one of the five MOESI (Modified, Owned, Exclusive, Shared, Invalid) states.” PO Resp. 4–5 (citing Ex. 2003, 1) (emphasis added); see Reply 3–4. Again, this isolated use of a shorthand of “states” in a publication, with several of the same authors as the other cited publication, does not persuade us that this shorthand is universally accepted to have a particular meaning whenever a particular type of software is being discussed. Moreover, the use of the term “generally” in this statement shows that it is not limiting the term “states” to cache coherency states, or more specifically, MOESI states.

Dr. Oklobdzija’s testimony, which cites and relies on this article and Sorin, also does not persuade us that “states” necessarily means cache coherency states. See Ex. 2016 ¶¶ 15–16 (citing Ex. 2010, 88–91; Ex. 2003, 1). Thus, we are not persuaded that Patent Owner’s extrinsic evidence regarding ordinary meaning overcomes the intrinsic record of this case. See *Phillips*, 415 F.3d at 1318 (“[A] court should discount any expert testimony that is clearly at odds with the claim construction mandated by the claims themselves, the written description, and the prosecution history, in other words, with the written record of the patent.”) (internal citations and quotations omitted).

In summary, we conclude that the “states” in the claim language “states associated with selected ones of the cache memories,” as recited in independent claims 1, 16, and 25 of the ’121 patent, relate to the contents of cache memory, but are not limited to cache coherence protocol states.

b. PRESENCE

Even if we agreed with Patent Owner’s assertion that the term “states” is limited to cache coherence states, we are persuaded by Petitioner’s assertions that within the field of cache coherency, a “state” may refer to a lack of presence, *see* Pet. 10; Reply 4, and that the ’121 patent and extrinsic evidence cited by Patent Owner demonstrate that “not present” is a cache coherency state and the invalid cache coherency state may signify a lack of presence, *see* Reply 4; Tr. 51:5–52:17; Ex. 1025 ¶ 25. We find unpersuasive Patent Owner’s arguments to the contrary. *See* PO Resp. 6–10.

Patent Owner asserts that the ’121 patent teaches “‘state’ provides additional information about ‘a particular cached line’ that is known to already be ‘somewhere’ (i.e. it is already known to be present).” *Id.* at 7. Patent Owner bases this assertion on a passage of the ’121 patent stating “because the cache coherence directory provides information about *where*” “*memory lines are cached as well as their states*, probes only need be directed toward the clusters in *which the requested memory line is cached*” and “[t]he state of a particular cached line will determine what type of probe is generated.” *Id.* (quoting Ex. 1001, 19:36–43 (emphases added by Patent Owner)). According to Patent Owner, this passage “plainly indicates” that the state of a memory line is different from where it is located and that a “state” only exists for a cached line. *Id.*

Patent Owner also points to the ’121 patent’s discussion of an “occupancy vector” as demonstrating that the patent does not consider presence to be a state. *Id.* at 8–9. According to Patent Owner, the ’121 patent’s statement that “[a]ny mechanism for tracking what clusters hold a copy of the relevant memory line in cache is referred to herein as an occupancy vector” (Ex. 1001, 14:2–4), and its treatment of the “occupancy

vector” differently than the “state” field in Figure 7, indicate that the ’121 patent “understands presence and ‘state’ to be different.” PO Resp. 8–9 (citing Ex. 1001, 13:55–57, 13:67–14:2, Fig. 7); Ex. 2016 ¶ 24.

Intrinsic and extrinsic evidence contradicts Patent Owner’s and Dr. Oklobdzija’s suggestion that a cache coherency state necessarily needs to indicate more than whether or not a memory line is present in cache memory. Starting with the intrinsic evidence, as Petitioner argues, the ’121 patent itself directly touches on the subject of presence when discussing cache coherency states. *See* Reply 4. For example, when describing certain embodiments associated with Figure 7, including cache coherency memory line states of “modified, owned, shared, and invalid,” the ’121 patent declares “[i]n the invalid state, a memory line is not currently available in cache associated with any remote cluster.” Ex. 1001, 13:58–61 (emphasis added); *see* Reply 4. Patent Owner argues that, contrary to the ordinary meaning of these words, the disclosed “invalid” state does not refer only to a lack of presence, but instead means that the memory line “can be present but invalid,” because it cannot be “rel[ied] on,” or is “not available to use because it is invalid.” Tr. 80:20–81:13. Patent Owner, however, does not explain sufficiently or persuasively why one of ordinary skill would understand the statement in this way, as opposed to the plain meaning of the “currently available” language in the specification. *See id.* at 80:13–84:8.

Moreover, none of the passages of the ’121 patent cited by Patent Owner clearly limits the scope of “states.” *See, e.g.*, Ex. 1001, 13:44–55 (stating multiple times that the embodiment depicted in Figure 7, with an occupancy vector, is an “example”). Regarding Patent Owner’s citation of the ’121 patent’s discussion of states providing information beyond whether specific data is present in a cache, we again note that the ’121 patent

expressly discloses that its invention encompasses use of states other than the examples discussed therein. *Id.* at 14:30–36.

Turning to extrinsic evidence, as Petitioner argues, extrinsic evidence cited by Patent Owner and Dr. Oklobdzija explains that not present is a cache coherency state and consistent with the '121 patent's discussion of the "invalid" state, one meaning of the invalid cache coherency state is that the block is not present or available in the cache. *See* Reply 4; Ex. 1025 ¶ 25; Ex. 1001, 13:58–61. For example, Patent Owner and Dr. Oklobdzija cite Sorin as well as PARALLEL COMPUTER ARCHITECTURE, a textbook authored by David E. Culler (1999) (Exs. 1028, 2002, 2011, "Culler").⁷ *E.g.*, PO Resp. 4 (citing Ex. 2010, 88–91); *id.* at 10 n.1 (citing Ex. 2011, 302). Dr. Oklobdzija testified that Sorin is a "competent" and Culler is a "very highly regarded" source regarding cache coherency. Ex. 1026, 38:3–39:11.

Culler explains that "if a block is not present in a cache we can assume" or "view[]" "it to be in a special 'not present' state or even in the invalid state." Ex. 1028, 279–280. Culler elsewhere "treat[s] the not-present state as invalid." *Id.* at 296; *id.* at 280 ("[T]he 'not present' state is assumed to be the same as invalid."); *see id.* at 292, 294–95. Specifically with respect to a Dragon protocol it discusses, Culler states that "there is no explicit invalid (I) state" but that "if a block is not present in a cache at all, it can be imagined in a special invalid or not-present state." Ex. 1028, 302. Later, Culler discusses using the "state NP (not present)" in connection with the

⁷ Patent Owner submitted two pages of Culler as Exhibit 2002 and nine pages of Culler as Exhibit 2011. Because these pages also are included in Exhibit 1028, submitted by Petitioner, for consistency and clarity, our citations to Culler in this Decision are only to Exhibit 1028.

“MESI protocol.” *Id.* at 307–310. As Dr. Horst testifies, this discussion of Culler expressly “refers to ‘not present’ as a state.” Ex. 1025 ¶ 25.

Sorin explains the cache coherence state of “I(nvalid)” as follows:

The block is invalid. The cache either does not contain the block or it contains a potentially stale copy that it may not read or write. In this primer, we do not distinguish between these two situations, although sometimes the former situation may be denoted as the “Not Present” state.

Ex. 2010, 89.

Thus, the ’121 patent and the extrinsic evidence of record demonstrate that not present is a cache coherency state and that one meaning of the invalid cache coherency state is that the block is not present or available in the cache. Accordingly, we conclude that even if “state,” as recited in independent claims 1, 16, and 25 of the ’121 patent, requires a cache coherency state, it encompasses the cache coherency states of invalid, meaning the block is not present or available in the cache, and not present.

2. “PROGRAMMED” (CLAIM 11)

Claim 11 recites “wherein each of the processing nodes is *programmed* to complete a memory transaction after receiving a first number of responses to a first probe.” Ex. 1001, 31:49–52 (emphasis added). We did not construe “programmed” in the Institution Decision. In the Response, Patent Owner asserts “programmed” should be construed as “configured by a sequence of instructions.” PO Resp. 11. According to Patent Owner, this construction is consistent with the ’121 patent and various dictionary definitions. *Id.* at 11–14 (citing Ex. 1001, 27:67–28:2, 28:8–24, 29:1–3; Ex. 2012, 359; Ex. 2013, 590–91; Ex. 2014, 931). Patent Owner adds that, at a minimum, the broadest reasonable interpretation of

“programmed” must exclude hardwired logic. *Id.* at 15–17 (citing Ex. 2012, 214; Ex. 2015, 15/3; Ex. 2016 ¶¶ 38–39).

Petitioner asserts Patent Owner’s interpretation of “programmed” is too narrow and proposes a construction of “designed to perform a sequence of operations, regardless of whether this design is in hardware or software.” Reply 5–7. Petitioner argues Patent Owner does not cite to any language in the ’121 patent that limits the nature of programming and contends the term should be given its ordinary meaning. *See id.*; Tr. 43:17–24. According to Petitioner, the patent does not use the words “instruction” or “execute” and makes no distinction between hardwired logic and the relevant portion of the claimed processors. Reply 5. Petitioner also states that dictionaries to which Patent Owner cites include “reasonable definitions for programming that defy [Patent Owner’s] proposal.” *Id.* (quoting Ex. 2014, 931).

We agree with Petitioner’s assertions that the ’121 patent does not define or use the term “programmed” in a manner inconsistent with the plain and ordinary meaning. None of the passages of the ’121 patent specification to which the parties cite define or limit the scope of the term. *See* Ex. 1001, 7:49–52, 27:67–28:24; PO Resp. 12–13, 32 (citing Ex. 1001, 27:64–28:24, 29:1–3); Reply 5–6 (citing Ex. 1001, 7:49–52, 28:2–5).

The heart of the dispute regarding this limitation, however, lies in whether the disclosure of processors completing transactions in a system necessarily discloses that those processors are “programmed” to do so. Because the ’121 patent does not illuminate this issue, but instead the parties rely on expert testimony and other extrinsic evidence to support their respective positions, we consider this issue to more properly fall within the category of unpatentability as opposed to claim construction. *See* Pet. 30–31; PO Resp. 11–17, 30–33; Reply 5–7. Thus, we do not further construe

the term “programmed” other than to note that the ’121 patent uses the term in the ordinary and customary manner. *See* Ex. 1001, 7:49–52, 27:64–28:24, 29:1–3; Ex. 2012, 359; Ex. 2013, 590–91; Ex. 2014, 931.

3. “[A]CCUMULATE RESPONSES TO EACH PROBE” (CLAIM 15) AND
“ACCUMULATING PROBE RESPONSES” (CLAIM 25)

Claim 15 recites “accumulate responses to each probe.” Ex. 1001, 32:3–5. Claim 25 similarly recites “accumulating probe responses.” *Id.* at 32:58–61. We did not expressly construe these terms in the Institution Decision. Patent Owner argued in the Preliminary Response that these terms should be construed as “gather two or more responses to a probe,” Prelim. Resp. 24, and indicated at oral hearing that it continues to maintain that this is the proper construction of the terms, Tr. 116:10–18. Patent Owner asserts that this construction is supported by the plain language of the claims as well as the dictionary definition of “accumulate.” Prelim. Resp. 24 (citing Ex. 2004 (MERRIAM-WEBSTER’S COLLEGIATE DICTIONARY (10th ed. 1999), 8). Petitioner has not offered a construction of the term but in its Reply, agrees that Patent Owner’s proposed construction is “reasonable.” Reply 23.

Neither Petitioner nor Patent Owner argues that the specification or prosecution history of the ’121 patent defines or otherwise limits the scope of the claim language to something other than its ordinary and customary meaning. *See, e.g.*, Prelim. Resp. 24; Reply 23. Based on our review, we conclude that the patent uses the term “accumulate” consistent with its ordinary and customary meaning, and the patentee did not define or otherwise narrow the scope of the claim language. *See, e.g.*, Ex. 1001, 9:38–42, 10:17–20, 11:14–18, 12:15–20, 12:37–38, 28:63–67, 29:46–50, 30:7–9.

Turning to extrinsic evidence, Dr. Oklobdzija testified at his deposition that there is no special meaning of the claim language in the art.

See Ex. 1026, 142:8–12; Reply 23. In addition, the dictionary definition of “accumulate” to which Patent Owner cites—“to gather or pile up esp[ecially] little by little”—supports Patent Owner’s proposed construction as the ordinary and customary meaning of the claim language. See Prelim. Resp. 24 (citing Ex. 2004, 8); Ex. 2004, 8.

In sum, we agree that Patent Owner’s proposed construction is the ordinary and customary meaning of the claim language. We construe “accumulate responses to each probe,” as recited in claim 15, as gather two or more responses to each probe, and “accumulating probe responses,” as recited in claim 25, as gathering two or more responses to a probe.

C. INSTITUTED GROUNDS OF UNPATENTABILITY

1. *ANTICIPATION BY PONG*

We turn to the instituted ground asserting that claims 1–3, 8, 11, 15, 16, and 25 of the ’121 patent are unpatentable as anticipated by Pong. Pet. 20–44; Inst. Dec. 14–23. Pong, a U.S. patent application, was filed on November 19, 1999 and published on May 2, 2002. Ex. 1003, [22], [43]. The ’121 patent was filed on October 15, 2004 as a continuation-in-part of U.S. Patent Application No. 10/288,347 (“’347 Application”), filed on November 4, 2002. Ex. 1001, [22], [63]. Thus, we agree with Petitioner’s uncontested assertions that Pong is prior art to the ’121 patent at least under 35 U.S.C. § 102(a) and 102(e). Pet. 4.

a. OVERVIEW OF PONG

Pong discloses a multiprocessor system implementing an asynchronous cache coherence protocol. Ex. 1003 ¶ 12. When a processor “propagates a read or write request” and the receiving processor, upon probing its cache, determines that it “does not have [the requested] data block . . . , it simply drops the request without responding.” *Id.* ¶ 24.

Therefore, “the processors do not have to synchronize a response to a request for a data block.” *Id.*

In addition, Pong explains that “write invalidation and write update” are the “two primary protocols for cache coherence.” *Id.* ¶ 48. “Either of these protocols may be used to implement” the disclosed system. *Id.* ¶ 69; *see id.* ¶ 48. These protocols respond differently to write operations. Specifically, in response to a write operation, “[t]he write invalidation protocol invalidates other copies of a data block,” whereas the write update protocol “updates all of the cached copies of a data block.” *Id.* ¶ 48.

Pong discusses an implementation of the disclosed multiprocessor system using “point-to-point links to communicate memory requests.” *Id.* ¶ 12; *see id.* ¶¶ 15, 29–30. Figure 2 of Pong is reproduced below.

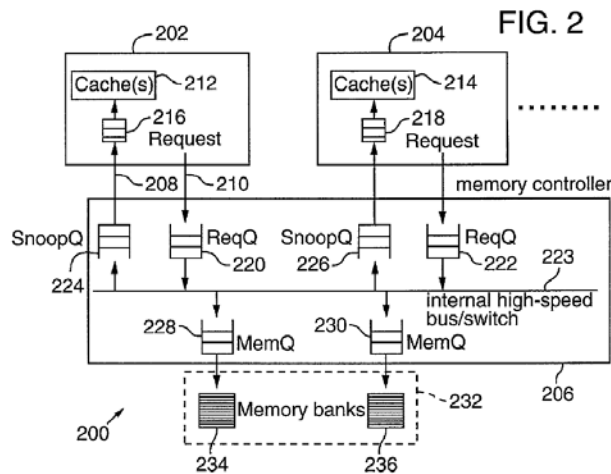


Figure 2 depicts multiprocessor 200, with processors 202, 204 and memory controller 206. *Id.* ¶¶ 18, 30. Each processor includes “one or more caches” 212, 214. *Id.* ¶ 31. Each processor communicates with memory controller 206 through “two dedicated and unidirectional links.” *Id.* ¶ 30.

Memory controller 206 has one request queue 220, 222 and one snoop queue 224, 226 per processor. *Id.* ¶¶ 31–33. Pong explains that the queues in the multiprocessor system process requests in first in, first out order. *Id.*

¶ 35. The request queue receives requests from the processor; the snoop queue issues requests to the processor. *Id.* ¶¶ 32–35. Memory controller 206 also includes internal address bus 223, through which these queues communicate. *See id.* ¶ 33, Fig. 3. “When a processor issues a request for a block of data,” the request enters a request queue (220 or 222) in memory controller 206. *Id.* ¶ 32; *see id.* ¶ 43. These request queues may be designed either to “broadcast the request to all other processors and the memory” or to “target the request to a specific processor or set of processors known to have a copy of the requested block.” *Id.* ¶ 32.

Pong describes, with reference to Figure 3, a “possible implementation of the data path for the architecture” shown in Figure 2, in which the memory controller also includes an incoming queue and an outgoing queue for each processor. *See id.* ¶¶ 39–43, Fig. 3. Pong explains that when a processor “is responding to a request for a data block,” it transfers the data block to the memory controller, particularly the outgoing queue that corresponds to the processor. *Id.* ¶¶ 41, 43. From the outgoing queue, the data block enters the bus of the memory controller. *Id.* ¶ 41. From the bus, the data block is buffered in the incoming queue that corresponds to the processor to which the data block is destined. *Id.*

Pong also discloses the use of a directory to “reduce traffic in the control path.” *Id.* ¶ 51. “A directory, in this context, is a mechanism for identifying which processors have a copy of a data block.” *Id.* The directory can be implemented with a “presence bit vector,” with one bit per processor. *Id.* “When the bit corresponding to a processor is set in the bit vector, the processor has a copy of the data block.” *Id.*

Figure 4 of Pong is reproduced below. Pong explains that “unless otherwise noted, the description of the components in [Figure 4] is the same” as discussed for Figures 2 and 3. *Id.* ¶ 56.

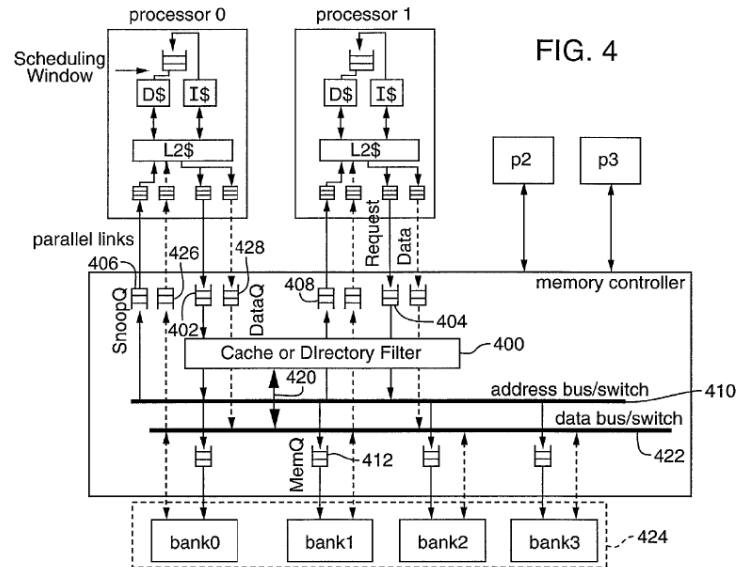


Figure 4 “is a block diagram of a multiprocessor with a memory controller.” *Id.* ¶ 20. As shown in Figure 4, one option for implementing the directory is to store the directory “in a memory device that is . . . integrated into the memory controller.” *Id.* ¶ 57. “In this implementation, the memory controller directs a request from the request queue to the directory, which filters the request and addresses it to the appropriate processors (and possibly memory devices).” *Id.* ¶ 56. Specifically, “directory filter 400 receives requests from the request queues (e.g., 402, 404) in the memory controller, determines which processors have a copy of the data block of interest, and forwards the request to the [snoop queue](s) (e.g., 406, 408) corresponding to these processors.” *Id.* ¶ 57.

b. ALLEGED LACK OF ENABLEMENT OF PONG

Patent Owner, in its Response, argues that Pong does not enable a person of ordinary skill in the art to practice the inventions recited in the

independent claims of the '121 patent without undue experimentation.

PO Resp. 18, 24. In particular, Patent Owner alleges three deficiencies in Pong's disclosure: (1) a failure "to teach how to practice a point-to-point architecture," (2) "unexplained contradictions" in Pong's disclosures and a lack of clarity as to "what combinations of embodiments can be practiced with each other," and (3) a failure to disclose adequately a cache coherence protocol. *Id.* at 18, 23–24. As a result of these alleged deficiencies, Patent Owner, with supporting testimony from Dr. Oklobdzija, asserts that "it would take in excess of two years to design, verify, and implement a computer system with a point-to-point architecture and supporting cache coherency based on the disclosures of Pong and the background knowledge of one skilled in the art." *Id.* at 24 (citing Ex. 2016 ¶ 83).

Petitioner responds, arguing that Pong would have enabled a person of ordinary skill to make or use the claimed invention, particularly when considering Pong's disclosures in light of information known in the art, such as Culler, a textbook that Dr. Horst and Dr. Oklobdzija agree is well known and commonly used in college courses. Reply 8–14; *see* Ex. 1025 ¶ 10. According to Petitioner, Dr. Horst's testimony and Culler demonstrate that the disclosures of Pong alleged to be deficient were well known in the art and one of ordinary skill would have readily understood and could have implemented Pong's alternative embodiments, including the specific embodiment on which Petitioner's unpatentability arguments rely. *See* Reply 10–14. In addition, Petitioner faults Dr. Oklobdzija's declaration testimony as lacking adequate explanation and support for his two-year estimate for design and implementation. *Id.* at 8–10. Moreover, Petitioner argues that Dr. Oklobdzija's deposition testimony evidences that his

enablement opinions are based on Patent Owner's unsupported claim constructions and relate to unclaimed features. *Id.* at 9–10.

To be anticipatory, the allegedly anticipatory disclosure must “enable[] the subject matter of . . . the patented invention.” *Elan Pharm., Inc. v. Mayo Found. for Med. Educ. & Research*, 346 F.3d 1051, 1052 (Fed. Cir. 2003). A prior art disclosure is enabling if it teaches a person of ordinary skill in the art to make or use the claimed invention “without undue experimentation.” *Id.* at 1054; *see United States v. Telectronics, Inc.*, 857 F.2d 778, 785 (Fed. Cir. 1988) (“The test of enablement is whether one reasonably skilled in the art could make or use the invention from the disclosures in the [reference] coupled with information known in the art without undue experimentation.”). In other words, enablement “is not precluded by the necessity for some experimentation”; rather, the required experimentation “must not be undue.” *In re Wands*, 858 F.2d 731, 736–37 (Fed. Cir. 1988). “The determination of what constitutes undue experimentation in a given case requires the application of a standard of reasonableness, having due regard for the nature of the invention and the state of the art.” *Id.* at 737. The determination is not “merely quantitative,” because even a “considerable amount of experimentation is permissible, if it is merely routine.” *Id.* “Factors to be considered in determining whether a disclosure would require undue experimentation” include:

- (1) the quantity of experimentation necessary, (2) the amount of direction or guidance presented, (3) the presence or absence of working examples, (4) the nature of the invention, (5) the state of the prior art, (6) the relative skill of those in the art, (7) the predictability or unpredictability of the art, and (8) the breadth of the claims.

Id.

Here, as an initial matter, we note that Dr. Oklobdzija testified at his deposition that his enablement opinions are based on Patent Owner’s proposed constructions of “states”—which we have found in § II.B.1.a to be improper as narrower than the broadest reasonable construction. Ex. 1026, 131:25–132:18; Reply 10. We also rejected Patent Owner’s proposed narrow construction of “states” in our Institution Decision. *See* Inst. Dec. 7–10. Dr. Oklobdzija’s reliance on this proposed construction, which has not been accepted by the Board, renders his opinion that Pong does not enable the independent claims of the ’121 patent misplaced.

Beginning with the first of the deficiencies Patent Owner alleges, Patent Owner argues Pong fails to enable the “point-to-point architecture” recited in the independent claims of the ’121 patent. *See* Ex. 1001, 30:66, 32:9, 32:42–43. Patent Owner argues that Pong treats a point-to-point architecture as largely interchangeable with a bus, but known bus-based cache coherence schemes were not readily adapted to a point-to-point architecture in 2002.⁸ PO Resp. 18–20 (citing, *inter alia*, Ex. 2016 ¶¶ 75–76). According to Patent Owner, Pong lacks “meaningful explanation” of required implementation issues, such as “what are the messages, how are

⁸ The ’121 patent was filed on October 15, 2004 as a continuation-in-part of the ’347 Application, filed on November 4, 2002. Ex. 1001, [22], [63]. The parties’ enablement arguments address the state of the art in 2002 and, thus, assume that the independent claims of the ’121 patent are entitled to priority to the ’347 Application. *See* PO Resp. 19; Ex. 2016 ¶ 75; Reply 12; Ex. 1025 ¶ 11. In addressing the enablement arguments, we likewise assume that the independent claims of the ’121 patent are entitled to priority to the filing date of the ’347 Application, November 4, 2002, as we determined in the Institution Decision in related IPR2015-00163. IPR2015-00163, Paper 18, 15–17. As explained below, we conclude that the claims are enabled as of this earlier priority date. Even if the claims were not entitled to this earlier priority date, we would reach the same conclusion.

messages generated, what structures handle processing of which messages, [and] how are messages disseminated through the point-to-point architecture.” *Id.* at 20; Ex. 2016 ¶ 77.

We, however, agree with Petitioner that the record before us shows that Pong’s disclosures would have enabled a person of ordinary skill in the art to implement the recited point-to-point architecture without undue experimentation. *See* Reply 11–12; Ex. 1025 ¶¶ 7–11, 18. Petitioner—with persuasive supporting testimony from Dr. Horst—demonstrates that point-to-point architectures, particularly the architecture disclosed in Pong, as well as directory-based cache coherence schemes using such point-to-point architectures, were well known before 2002. Reply 11–12; Ex. 1025 ¶¶ 10–11. Dr. Horst agrees with Dr. Oklobdzija’s testimony that Culler is a “very highly regarded” textbook that was taught in college courses at the time the ’121 patent was invented, and thus opines that Culler is a reliable indication of the information available to one of ordinary skill in the art at the relevant time period. Ex. 1025 ¶ 10; Ex. 1026, 38:15–40:11. Moreover, Dr. Horst explains that the point-to-point architecture in Pong is known as a “dancehall” approach, which Culler discloses includes processors with their own cache connected to main memory by a “scalable point-to-point network rather than a bus.” Ex. 1025 ¶ 11; Ex. 1028, 270–71, Fig. 5.2(c); *see* Ex. 1003 ¶¶ 29–30, Figs. 2, 4. Culler identifies this dancehall approach as a “common” architecture. Ex. 1028, 270–71, Fig. 5.2(c); Ex. 1025 ¶ 11. In addition, Petitioner and Dr. Horst point out that Culler devotes an entire chapter—more than 120 pages in length—to directory-based cache coherence schemes, which “rely on point-to-point network transactions,” thereby demonstrating that the use of such schemes using point-to-point architectures was well known by 2002. Reply 12, 14; Ex. 1025 ¶ 11;

Ex. 1028, 553–677 (chapter 8). Dr. Horst’s testimony, together with Culler as evidence of the knowledge of a person of ordinary skill in the art, demonstrates that the point-to-point architecture disclosed in Pong was well within the understanding and ability of one of ordinary skill.

As a specific example, Dr. Oklobdzija views as “impractical” and lacking adequate explanation Pong’s statement that “[t]he topology of a point-to-point architecture may be made transparent to the devices utilizing it by emulating a shared bus type of protocol.” Ex. 1003 ¶ 29; Ex. 2016 ¶ 76. Yet, as Petitioner notes, Dr. Oklobdzija testified at his deposition that such “emulati[on of] a shared bus-type protocol” becomes “impractical” only in a system with “more than 16” processors. Ex. 1026, 140:1–141:8. The ’121 patent claims, however, recite only a “*plurality* of processing nodes interconnected by a first point-to-point architecture” and, therefore, do not require that more than sixteen processors be connected by the recited point-to-point architecture. Ex. 1001, 30:65–66, 32:8–9, 32:41–43 (emphasis added). Thus, Dr. Oklobdzija’s alleged concern with the practicality of Pong’s disclosures is not directed to an actual requirement of the claims. Moreover, we find persuasive Dr. Horst’s testimony that one of ordinary skill would have understood Pong’s statement regarding emulation of a shared bus protocol to refer to Pong’s disclosure, in its discussion of the point-to-point architecture in Figure 2, that the queues in the memory controller “may be designed to broadcast the request to all other processors and the memory” because in a shared bus protocol, all requests are sent to all processors. Ex. 1025 ¶ 9; Ex. 1003 ¶ 32. Accordingly, we are not persuaded that Pong’s reference to emulating a shared bus type of protocol would have interfered with a person of ordinary skill’s implementation of the point-to-

point architecture disclosed in Pong and recited in the independent claims of the '121 patent.

As to the implementation details Patent Owner faults Pong for failing to provide, we agree with Petitioner and Dr. Horst that such “low-level implementation choices . . . were well within the ability and knowledge of a person of ordinary skill in the art.” Ex. 1025 ¶ 17; Reply 9, 14; see PO Resp. 20. Indeed, Culler includes a chapter spanning more than 120 pages directed to directory-based cache coherence systems that rely on point-to-point networks, including various case studies as well as testing and simulation of such systems. See Ex. 1028, 553–555; see *generally id.* at 553–677 (chapter 8). Therefore, we do not agree with Patent Owner’s implication that Pong’s failure to explain these details would have impeded a person of ordinary skill from building an operable system with a point-to-point architecture based on Pong’s disclosures.

Turning to the second alleged deficiency, Patent Owner argues that Pong contains contradictory disclosures regarding how messages are addressed to processors, specifically paragraph 56 of Pong refers to a *directory* in the memory controller “filter[ing a] request and address[ing] it to the appropriate processors,” whereas paragraph 47 refers to “the *requesting processor* specifically address[ing a] read request to the processor that has the valid copy.” PO Resp. 21 (quoting Ex. 1003 ¶¶ 47, 56) (emphasis added). According to Patent Owner and Dr. Oklobdzija, these disclosures make it “unclear” whether Pong’s “requests are specifically addressed by the requesting processor to the processor which possesses the cache line, or . . . whether the addressing is performed by the memory controller.” *Id.*; Ex. 2016 ¶ 78. In addition, Patent Owner criticizes Pong for disclosing a “large number of different ‘embodiments’” without teaching

how these embodiments come together into an “operable” apparatus that “satisfies the limitations of the independent claims of the ’121 [p]atent.”

PO Resp. 21–22; *see* Ex. 2016 ¶ 79.

As Petitioner points out, however, Petitioner’s anticipation arguments rely on the embodiment depicted in Pong’s Figure 4—in which, as disclosed in paragraphs 56 and 57, a directory integrated into a memory controller filters requests and addresses them to processors—implementing either a write update or write invalidate cache coherency protocol. *See, e.g.*, Pet. 23–30, 36–44, 56; Ex. 1014 ¶¶ A-5–A-14; Inst. Dec. 16–19; Reply 10, 13–14; Ex. 1003 ¶¶ 48, 55–57, 69, Fig. 4. Accordingly, the other embodiments disclosed in Pong to which Patent Owner’s arguments refer, including the embodiment disclosed in paragraph 47, are irrelevant to the issue of whether Pong enables one of ordinary skill to make and use the embodiment in Figure 4, implementing either a write update or invalidate protocol, without undue experimentation. *See* Reply 10, 13–14.

In addition, with respect to the embodiment depicted in Figure 4, Pong states that “[u]nless otherwise noted, the description of the components is the same as provided above” with regard to Figures 2 and 3. Ex. 1003 ¶ 56; Reply 13. Thus, we agree with Petitioner that Pong makes clear that its description of the implementation in Figure 4 takes precedence over any contradictory description in previous embodiments. Reply 13; Ex. 1025 ¶ 15. We also find persuasive Petitioner’s argument and Dr. Horst’s opinion that one of ordinary skill in the art could readily distinguish this embodiment in Figure 4 from the alternative implementations discussed throughout Pong. Ex. 1025 ¶¶ 7–16; Reply 10–14.

As to the specific alleged contradiction between the disclosures in paragraphs 47 and 56 regarding how requests are addressed to processors,

we agree with Petitioner and Dr. Horst that one of ordinary skill would have understood these paragraphs to disclose alternative means by which a directory may filter requests. Reply 12–13; Ex. 1025 ¶ 13. We credit Dr. Horst’s testimony that one of ordinary skill would have readily distinguished these “well-known alternative embodiments” and “would not have viewed them as conflicting.” Ex. 1025 ¶ 13. Indeed, Patent Owner appears to concede that the disclosures in paragraphs 47 and 56, regarding “processors which specifically address vs. memory controllers which filter,” are merely “different ‘embodiments.’” PO Resp. 21.

Pong, in paragraph 47, discloses a “[d]irectory [b]ased [f]ilter,” specifically “for [r]ead [m]isses,” in which “state information can be extended to include the ID of the processor that currently has a particular data block” such that “when a requesting processor makes a read request and finds that its cache does not have a valid copy of the requested data block,” the requesting processor then can “[u]se[] the processor ID” to “specifically address[] the read request to the processor that has the valid copy.” Ex. 1003 ¶¶ 46–47. We credit Dr. Horst’s explanation that a person of ordinary skill would have understood this disclosure to describe an “embodiment in which a requesting processor uses information contained in a directory to specifically address requests to a processor storing a valid copy.” Ex. 1025 ¶ 13. In paragraph 56, however, Pong refers to a “directory” of a memory controller, “which filters [a] request and addresses it to the appropriate processors.” Ex. 1003 ¶ 56. We credit Dr. Horst’s testimony that one of ordinary skill would have understood this to be an “alternative approach” to that disclosed in paragraph 47 in which the directory itself, rather than the requesting processor, addresses the request. Ex. 1025 ¶ 13.

We agree with Petitioner and Dr. Horst that Culler supports Dr. Horst’s testimony on this point. *Id.* Specifically, similar to the approach disclosed in paragraph 47, Culler describes a protocol in which after a “read miss,” “the requestor” is given the “identity of the owner node” and then “sends a request to the owner.” Ex. 1028, 585; *see id.* at 560, 564; Ex. 1025 ¶ 13. In addition, Culler explains that the number of network transactions in this approach can be reduced by instead using “intervention forwarding,” in which the home, including a directory, “forwards the request as an intervention transaction to the owner, asking it to retrieve the block from its cache”—which Dr. Horst testifies corresponds to the approach disclosed in Pong’s paragraph 56. Ex. 1028, 585; Ex. 1025 ¶ 13.

Accordingly, we do not view paragraphs 46 and 57 of Pong as contradictory, as Patent Owner suggests, but instead as alternative embodiments that would have been readily understood and distinguished by one of ordinary skill. Nor do we agree with Patent Owner’s suggestion that the alleged contradiction between these paragraphs would have interfered with one of ordinary skill’s ability to implement a directory that filters and addresses requests, as disclosed in paragraph 56 of Pong, on which Petitioner’s anticipation arguments rely. *See, e.g.,* Pet. 23–30, 36–44, 56; Inst. Dec. 16–19; Reply 10, 13–14; Ex. 1014 ¶¶ A-6, A-14.

Moving to Patent Owner’s final alleged deficiency in Pong’s disclosure, Patent Owner contends Pong fails to disclose “how cache coherency is maintained, i.e., [its cache coherence protocol.” PO Resp. 22. According to Patent Owner and Dr. Oklobdzija, because the ’121 patent defines “probe” as “[a] mechanism for eliciting a response from a node to maintain cache coherency in a system,” “the claims are directed to a computer system where ‘cache coherency’ is ‘maintain[ed]’”—which

necessarily requires a cache coherence protocol. *Id.* (quoting Inst. Dec. 6); Ex. 2016 ¶ 80 & n.3; *see* Ex. 1001, 5:45–47, Inst. Dec. 6. Patent Owner, with supporting testimony from Dr. Oklobdzija, argues that Pong does not identify with particularity or “teach what cache coherence protocol it uses in its embodiments which use point-to-point links.” PO Resp. 23; Ex. 2016 ¶ 80. Patent Owner also faults Pong for failing to disclose “any cache coherence protocol states,” which allegedly are an essential part of a cache coherence protocol. PO Resp. 23; Ex. 2016 ¶ 82.

Patent Owner and Dr. Oklobdzija do not explain or support adequately their assertion and opinion, respectively, that the claims necessarily require a cache coherence protocol. *See* PO Resp. 22; Ex. 2016 ¶ 80 n.3. A cache coherence protocol is not recited in any claim, and Dr. Oklobdzija’s unsupported and unexplained testimony does not show that the definition of “probe” in the ’121 patent necessitates such a requirement. *See* Ex. 2016 ¶ 80 n.3; 37 C.F.R. § 42.65(a) (“Expert testimony that does not disclose the underlying facts or data on which the opinion is based is entitled to little or no weight.”).

Even if we were to accept Patent Owner’s position that a cache coherence protocol is required to meet the claim language in light of the definition of “probe,” we disagree with Patent Owner’s assertions that Pong does not disclose sufficiently a cache coherency protocol that could be implemented by one of ordinary skill in the art. Pong is expressly directed to cache coherence protocols. *See, e.g.,* Ex. 1003, [54], ¶¶ 1, 69; *see also* Tr. 85:21–23 (Patent Owner stating that “any cache-coherency system is going to use some cache-coherency protocol”). Moreover, Pong explains directory-based protocols implemented using a presence bit vector. Ex. 1003 ¶¶ 51, 69; Reply 13–14; Ex. 1025 ¶ 16. In addition, Pong

discusses implementing the disclosed invention with “two types of *cache coherence protocols: write invalidate and write update*,” and explains that the “write invalidation protocol invalidates other copies of a data block in response to a write operation” whereas a “write update . . . protocol updates all of the cached copies of a data block when it is modified in a write operation.” Ex. 1003 ¶¶ 48, 69 (emphasis added). Notably, Patent Owner’s arguments and Dr. Oklobdzija’s opinion regarding Pong’s alleged failure to disclose a cache coherence protocol do not address specifically Pong’s disclosures regarding the implementation of the write update and write invalidate protocols, or a presence bit vector as a directory. See PO Resp. 22–23; Ex. 2016 ¶¶ 80–82. In light of these disclosures of Pong, we credit and find persuasive Dr. Horst’s testimony that, as further evidenced by Culler, one of ordinary skill “would have readily known how to implement the embodiment shown in [Figure] 4 of Pong, in which a centralized directory utilizes intervention forwarding and a presence bit vector to implement a cache coherence protocol.” Ex. 1025 ¶ 16 (citing Ex. 1028, 560–64); see Reply 12–14.

In addition, regarding Pong’s alleged failure to disclose cache coherence protocol states, as discussed above in § II.B.1.a, we do not agree with Patent Owner’s proposed construction of “states” as limited to cache coherence protocol states. For the same reasons, we disagree with Patent Owner’s enablement arguments based on this construction. See *supra* § II.B.1.a; Ex. 1026, 131:25–132:18; Reply 10. Moreover, Patent Owner’s enablement arguments regarding cache coherence states merely refer to its arguments disputing that Pong discloses “states.” PO Resp. 23. For the reasons explained below in § II.C.1.c.i.a, Pong sufficiently discloses the recited “states” to one of ordinary skill under our construction and Patent

Owner’s proposed narrower construction requiring cache coherence states—and Patent Owner has not offered persuasive evidence that these disclosures are not enabling.

Finally, Patent Owner argues, and Dr. Oklobdzija opines, that “[i]n light of” the three alleged deficiencies in Pong’s disclosure, discussed above, Pong “does not enable one of skill in the art [to] implement a cache-coherent, point-to-point architecture without undue experimentation.” *Id.* at 23–24; Ex. 2016 ¶ 83. Dr. Oklobdzija also states his opinion that “it would take in excess of two years to design, verify, and implement a computer system with a point-to-point architecture and supporting cache coherency.” Ex. 2016 ¶ 83. Petitioner and Dr. Horst, however, contest Patent Owner’s position. Reply 7–14; Ex. 1025 ¶ 18. In addition to opining that Pong is enabling, Dr. Horst expressly disagrees with Dr. Oklobdzija’s two-year estimate and further opines that even assuming this estimate was correct, this “would not constitute undue experimentation” because “[i]n [his] experience, it usually takes more than two years to ‘design, verify, and implement’ a multiprocessor system.” Ex. 1025 ¶ 18.

As outlined above, we disagree with Patent Owner regarding the three deficiencies it alleges in Pong’s disclosures and instead find more persuasive Petitioner’s showing that one of ordinary skill in the art would have understood and been able to implement Pong’s disclosures on each issue. Given that Patent Owner’s argument and Dr. Oklobdzija’s opinion that Pong would not have enabled one of ordinary skill to implement the invention recited in the independent claims of the ’121 patent are based on these alleged deficiencies, we disagree with Patent Owner’s position for the reasons given above. Instead, we credit and find persuasive Dr. Horst’s testimony that Pong “would have enabled one of ordinary skill in the art to

make or use the claimed invention without undue experimentation.”

Ex. 1025 ¶ 18; *see id.* ¶ 7. We agree with Petitioner that the record evidence—including Pong as well as evidence of the knowledge of a person of ordinary skill in the art, including Dr. Horst’s testimony and Culler—demonstrate that Pong’s disclosures would have enabled one of ordinary skill to make and use the system, apparatus, and method recited in the independent claims of the ’121 patent without undue experimentation.

Moreover, we agree with Petitioner that Dr. Oklobdzija’s testimony stating his opinion as to a two-year time period to implement Pong’s disclosures fails to explain sufficiently what and how much experimentation would have been required, or why this experimentation should be considered undue under the circumstances. Reply 8–9; *see* Ex. 2016 ¶ 83. For example, the testimony is devoid of any details regarding which tests would have had to be performed, whether such tests would have been routine in the art, whether the extent of testing would have been unusual for the art, and how such tests relate to the ability of one of ordinary skill in the art. *See Wands*, 858 F.2d at 737 (explaining that factors to be considered in determining whether experimentation is undue include “the quantity of experimentation necessary,” “the state of the prior art,” “the relative skill of those in the art,” and “the predictability or unpredictability of the art”). Therefore, we do not credit or find persuasive Dr. Oklobdzija’s two-year estimate. *See* 37 C.F.R. § 42.65(a).

In addition, even if we were to accept this two-year estimate, there is insufficient evidence in the record to support a determination that this would constitute undue experimentation. We do not find persuasive Patent Owner’s assertion, with a citation to *White Consolidated Industries, Inc. v. Vega Servo-Control, Inc.*, 713 F.2d 788, 791 (Fed. Cir. 1983), that two years

of experimentation is “clearly” undue. PO Resp. 24. The technology at issue in this proceeding is distinct from the numerical control system for machine tools that was at issue in *White* and, thus, *White* is not informative of the reasonableness of such a length of experimentation based on the “nature of the invention and the state of the art” for the ’121 patent. *See id.* at 789, 791; *Wands*, 858 F.2d at 737. Nor is the length of experimentation alone sufficient to show that it would be undue. *Wands*, 858 F.2d at 737 (“The test is not merely quantitative, since a considerable amount of experimentation is permissible, if it is merely routine . . .”). To the contrary, Dr. Horst opines that “[i]n [his] experience,” more than two years for designing, verifying, and implementing a multiprocessor system is standard and, therefore, would not constitute undue experimentation. Ex. 1025 ¶ 18.

In sum, the preponderance of the evidence before us demonstrates that Pong enables one of ordinary skill to make and use the system, apparatus, and method recited in the independent claims of the ’121 patent without undue experimentation.

c. INDEPENDENT CLAIMS 1, 16, AND 25

We turn to address Petitioner’s showing regarding whether Pong, particularly its embodiment in Figure 4, implementing either the write update or write invalidate protocol, discloses the limitations of independent claims 1, 16, and 25 of the ’121 patent, as required for anticipation. *See* Pet. 23–30, 35–44, 56; Inst. Dec. 17–23; Reply 15–25. Petitioner contends that Pong’s memory controller corresponds to the recited “probe filtering unit” and the bits in its presence bit vector correspond to the recited “probe filtering information” “representative of states associated with selected ones of the cache memories.” Pet. 25–27, 36–37, 38–44. Further, Petitioner

argues that both read requests and write requests in Pong correspond to the “probe” or “probes” recited in claims 1, 16, and 25. *Id.*

i. Common Disputed Limitations

We first address the limitations common to independent claims 1, 16, and 25 that Patent Owner disputes Pong discloses.

(a) “probe filtering information” “representative of states associated with selected ones of the cache memories”

Petitioner argues that—with either the write update protocol or the write invalidate protocol implemented—Pong’s presence bit vector is equivalent to the “probe filtering information” “representative of states associated with selected ones of the cache memories” recited in independent claims 1, 16, and 25. *Id.* at 26–27, 36–37, 40–42. Petitioner proffers supporting testimony from Dr. Horst, opining that—for both the write update protocol and the write invalidate protocol—a set bit in the presence bit vector indicates not only that the corresponding processor has a copy of the data block but also that the copy is valid. *Id.* at 26 (citing Ex. 1014 ¶¶ A-10–A-13); Ex. 1014 ¶¶ A-9–A-13. In other words, according to Petitioner, a set bit in the presence bit vector represents states of both “presence and validity” of the copy of the data block in the cache of the corresponding processor. Ex. 1014 ¶ A-13; *see* Pet. 26–27; Ex. 1014 ¶¶ A-10–A-13; Reply 15; Ex. 1025 ¶ 19.

Patent Owner contests Petitioner’s assertions. PO Resp. 25–27. With respect to the bits in Pong’s presence bit vector representing presence, Patent Owner contends Petitioner’s argument is foreclosed by Patent Owner’s proposed claim construction of “states” as limited to cache coherency states. *Id.* at 25. Next, as to the bits in Pong’s presence bit vector representing validity, or a “valid” state, Patent Owner asserts Petitioner does not rely on

Pong's disclosures and instead improperly relies on extrinsic evidence, including cache coherence protocols disclosed in other references, to inform Pong's teachings. *Id.* at 25–26. In addition, Patent Owner asserts Pong's presence bit vector does not maintain validity information and Pong's system instead separately tracks validity. *Id.* Specific to the write update protocol, Patent Owner also argues, with supporting testimony from Dr. Oklobdzija, that it is well known in the art that write update protocols do not have a valid/invalid state. *Id.* at 27; Ex. 2016 ¶ 90.

Under the construction of “states” we adopted in the Institution Decision and maintain in this Decision—in which “states” is not limited to cache coherency states—we determine that Pong's presence bit vector discloses the recited “probe filtering information” “representative of states associated with selected ones of the cache memories,” where either the write update or the write invalidate protocol is implemented. *See* Inst. Dec. 7–10; *supra* § II.B.1.a. First, under this construction, there is no dispute that “states associated with selected ones of the cache memories” includes presence. *See* Inst. Dec. 9–10; PO Resp. 25 (arguing that Petitioner's argument that the presence bit vector represents the “state” of presence allegedly is foreclosed only by Patent Owner's proposed “construction of state as a cache coherence protocol state”); *id.* at 1–11, 25–27; Reply 15; Ex. 3001, 497–48 (definition of “state”); Ex. 2016 ¶ 87. Nor is there any dispute that the bits in Pong's presence bit vector represent this state of presence. Pong discloses that when a bit in the presence bit vector is set, the processor that corresponds to the bit “has a copy of the data block,” i.e., a copy of the data block is present. Ex. 1003 ¶ 51. Accordingly, we agree with Petitioner that in both the write update and write invalidate protocols, a set bit represents the state of presence of a cached copy of the data block in a

particular processor. *See id.* ¶¶ 51, 69; Pet. 10, 26–27; Reply 15; Ex. 1014 ¶¶ A-9–A-13; Ex. 1025 ¶ 19.

Second, for the write update protocol, we conclude that Pong discloses the recited “probe filtering information” “representative of states associated with selected ones of the cache memories” for the additional reason that the bits in Pong’s presence bit vector are representative of validity, or a valid state—as Petitioner argues.⁹ *See* Pet. 26–27; Reply 15, 17–18. In addition to disclosing that a set bit in the presence bit vector reflects that the corresponding “processor has a copy of the data block,” Ex. 1003 ¶ 51, Pong explains that the write update protocol “updates all of the cached copies of a data block when it is modified in a write operation,” *id.* ¶ 48; *see id.* ¶ 69; Pet. 26–27 (citing Ex. 1003 ¶¶ 48, 51); Ex. 1014

⁹ We note that for the write invalidate protocol, there is insufficient evidence in the record to support Petitioner’s position that a set bit in the presence bit vector necessarily represents validity, or a valid state, because the directory clears the presence bit corresponding to a processor when it sends that processor an invalidation request. *See* Ex. 1014 ¶ A-12; Pet. 26; PO Resp. 26; Ex. 2016 ¶¶ 91, 93; Reply 16–19; Ex. 1025 ¶ 22. Unlike the write update protocol, Pong itself lacks adequate disclosures to support this finding. *See* Ex. 1003 ¶¶ 48–49 (explaining only that in a write invalidate, other copies of a data block are invalidated in response to a write invalidation). Additionally, particularly given this lack of relevant disclosure in Pong, we agree with Patent Owner that Dr. Horst’s reliance on the disclosures of Exhibit 1004, an article by David Chaiken (“Chaiken”), are insufficient to show how Pong necessarily operates. *See* Ex. 1014 ¶ A-12 (citing Ex. 1004, 50); PO Resp. 26; Ex. 2016 ¶¶ 91, 93; *Cont’l Can Co. USA v. Monsanto Co.*, 948 F.2d 1264, 1268 (Fed. Cir. 1991). For example, even assuming that Chaiken implements a write invalidate protocol, which Dr. Horst’s testimony does not address, his testimony lacks sufficient explanation to show that write invalidate protocols, by definition, clear the presence bit when sending an invalidation request or that Pong necessarily implements the same procedures as Chaiken. *See* Ex. 2016 ¶¶ 91, 93; Ex. 1014 ¶ A-12; 37 C.F.R. § 42.65(a).

¶¶ A-10–A-11 (same); Inst. Dec. 20. We credit and find persuasive Dr. Horst’s testimony that as a result, from the perspective of the presence bit vector, “every processor indicated as storing a copy of the memory line in its cache is storing a valid copy.” Ex. 1014 ¶ A-11; see Pet. 26–27. In other words, as Dr. Horst testifies, “because a write update protocol always updates caches to keep them valid,” “when a cache line is present[,] it is necessarily valid.” Ex. 1025 ¶ 23; see Reply 17–18. Indeed, Dr. Oklobdzija agrees that because the write update protocol “force[s the caches] to update their value for a cache line every time it changes”—as Pong discloses—*“every processor that has the cache line will have the line in a valid state.”* Ex. 2016 ¶ 90 (emphasis added); Ex. 1003 ¶ 48; see also Tr. 115:1–23 (Patent Owner conceding that in a write update protocol, any cached copies could be “considered . . . to be valid” because they are “kept usable” and “updated”). Accordingly, based on Pong’s disclosures, as well as the explanatory testimony of Dr. Horst in addition to Dr. Oklobdzija, we determine that in the write update protocol, a set bit in Pong’s presence bit vector represents that the corresponding cached copy is present and valid.

We are not convinced by Patent Owner’s arguments disputing that in the write update protocol, a set bit in Pong’s presence bit vector represents validity, or a valid state. Patent Owner contends Petitioner’s argument that the presence bit vector conveys whether a cache line is in a valid state “does not rely on the express disclosures of Pong” because Pong does not discuss validity with respect to the vector. PO Resp. 25; Ex. 2016 ¶ 89 (opining that “Pong itself does not disclose the presence bit vector as indicating a valid or invalid state”). Instead, according to Patent Owner, Petitioner and Dr. Horst “rel[y] on the teachings of other references to conclude that Pong’s presence bit vector reflects a validity state.” PO Resp. 26.

We recognize that Pong does not explicitly state the word “valid” regarding a bit in the presence bit vector. Yet—as outlined above—Pong’s disclosures that a write update protocol “updates all of the cached copies of a data block when it is modified in a write operation” and a set bit in the presence bit vector indicates that the corresponding “processor has a copy of the data block” are alone sufficient to make clear that when a bit is set in the vector, the corresponding cached copy of the data block is necessarily valid. Ex. 1003 ¶¶ 48, 51, 69. Dr. Horst’s explanatory testimony, discussed above, supports this conclusion, providing further evidence that a person of ordinary skill in the art readily would recognize and understand this from Pong’s disclosures. See Pet. 26–27; Ex. 1014 ¶¶ A-9–A-11, A-13; Reply 17–18; Ex. 1025 ¶¶ 19, 23, 26; see *Cont’l Can*, 948 F.2d at 1268 (holding that if an allegedly anticipatory reference “is silent about the asserted inherent characteristic, such gap in the reference may be filled with recourse to extrinsic evidence” which “must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill”).

Moreover, as Petitioner contends, even Dr. Oklobdzija’s testimony and the extrinsic evidence proffered by Patent Owner evidence that a set bit in Pong’s presence bit vector necessarily represents validity, or a valid state, within the write update protocol. PO Resp. 27 (citing Ex. 2011, 302; Ex. 2016 ¶ 90); Ex. 2016 ¶ 90 (citing Ex. 2011, 302) (opining that because a write update protocol “force[s the caches] to update their value for a cache line every time it changes,” “*every processor that has the cache line will have the line in a valid state*”) (emphasis added); Ex. 1028, 302 (“[A]n update-based protocol . . . always keeps the blocks in the cache up-to-date, so it is always okay to use the data present in the cache”); Reply 17–18;

Ex. 1025 ¶ 23. In addition, at the oral hearing, Patent Owner conceded that in a write update protocol, any cached copies could be “considered . . . to be valid” because they are “kept usable” and “updated.” Tr. 115:1–23.

Patent Owner also argues that Pong, as evidenced by paragraphs 13 and 61, tracks validity separately from presence and targets requests to processors based on presence—not validity. PO Resp. 25–26; Ex. 2016 ¶¶ 91–92. Paragraph 61 of Pong refers to Figure 6, illustrating “an example of a data block that incorporates a presence bit vector,” in which the data “block is associated with state information 602, such as a bit indicating whether the block is valid or invalid, and the processor ID of a processor currently having a valid copy of the block.” Ex. 1003 ¶ 61. As Petitioner responds, however, Petitioner does not rely on the embodiment illustrated in Figure 6, in which the presence bit vector is incorporated into a data block, and instead relies on the embodiment illustrated in Figure 4, in which the directory, or presence bit vector, is “integrated into the memory controller.” *Id.* ¶¶ 51, 56–57, 61; *see* Pet. 23–30, 36–44, 56; Inst. Dec. 16–19; Reply 18; Ex. 1014 ¶¶ A-5–A-14. Accordingly, this alternative embodiment is distinct from, and does not contradict Petitioner’s assertions regarding, the embodiment in Figure 4. In addition, we credit Dr. Horst’s explanation that the purpose of the state information bit in Figure 6 is distinguishable from that of the bits of the presence bit vector. *See* Ex. 1025 ¶ 24; Reply 18.

Paragraph 13, in turn, states that “[e]ach of the processors and memory that receive the request independently check to determine whether they have a valid copy of the requested data block based on the state information. Only the processor or memory having a valid copy of the requested data block responds to the request.” Ex. 1003 ¶ 13. This paragraph is in a “Summary” section and, as Petitioner argues, is not clearly

tied to the embodiment in Figure 4, including a directory in the form of a presence bit vector, implementing the write update protocol—as opposed to, for example, alternative embodiments employing the snooping or broadcast-based protocol or the write invalidate protocol discussed elsewhere in Pong. *See id.*; Reply 15–16; Ex. 1025 ¶ 20.

Even if paragraph 13 of Pong applies to the embodiment in Figure 4, we agree with Petitioner that a separate validity check by a processor—after receiving a request but before responding with the data block in its cache—does not undermine Petitioner’s showing that Pong’s presence bit vector represents validity. Reply 16; Ex. 1025 ¶¶ 21–22; Tr. 55:4–10. Rather, both parties acknowledge possible imperfections in a directory’s knowledge of the contents of the cache of processors; for example, a processor may flush or evict a memory line from its cache without alerting the directory or there may be a time lag between when a cache is modified and notice of the modification reaches the directory. *See* Reply 16; Ex. 1025 ¶ 21 (citing Ex. 1028, 562, 604); Ex. 1028, 562 (explaining that “[t]he directory information for a block is simply main memory’s view of the cache state of that block in different caches” and “there will be periods when a directory’s knowledge of a cache state is incorrect . . . [; d]uring this time, the directory may send a message to the cache based on its old (no longer valid) knowledge”); Tr. 55:4–10, 73:4–74:15. As a result, even if Pong’s processors tracked and checked the validity of the copy of the data block in its cache before responding to a request, this would not contradict Petitioner’s showing that the bits in the presence bit vector represent validity.

Such potential imperfections in the directory, specifically the presence bit vector, regarding the status of a processor’s cache does not undermine

Petitioner’s showing that the presence bit vector corresponds to the recited “probe filtering information,” because a set bit in the vector represents validity, or a valid state, of the corresponding processor’s cache—to the best information available to the presence bit vector. Reply 17; Ex. 1025 ¶¶ 21–22. As Patent Owner acknowledges, the claim language “probe filtering information” “representative of states associated with selected ones of the cache memories,” does not require completely accurate information—only that the information be representative of the state of cache memory. See Tr. 73:7–74:15 (Patent Owner); *id.* at 126:3–9 (Petitioner); Reply 17.

In sum, under our construction of “states,” Pong’s presence bit vector discloses the recited “probe filtering information” “representative of states associated with selected ones of the cache memories” because its bits represent the state of presence where either the write update or the write invalidate protocol is implemented and validity, i.e., a valid state, where the write update protocol is implemented.

In addition—even if we were to adopt Patent Owner’s proposed construction of “states” limiting the term to cache coherency states, we would still determine that Pong discloses the recited “probe filtering information” “representative of states associated with selected ones of the cache memories.” See PO Resp. 11; *see, e.g.*, Tr. 64:5–7, 85:8–86:8. As outlined above in § II.B.1.b, we agree with Petitioner that even under this narrower construction proposed by Patent Owner, the recited “states” include the cache coherency states of invalid, meaning the block is not present or available in the cache, and not present. Reply 4, 15, 19; *see* Pet. 10; Tr. 51:5–52:17; Ex. 1025 ¶ 25. Moreover, according to Petitioner, presence of a valid copy represents a “valid” cache coherency state. See Reply 15; Tr. 51:5–52:17; Ex. 1025 ¶ 23.

Patent Owner, with supporting testimony from Dr. Oklobdzija, disputes Petitioner's assertion that the bits in the presence bit vector represent a "valid" state where the write update protocol is implemented, arguing that it is well known in the art that "write-update protocols do not have a valid/invalid state." PO Resp. 27; Ex. 2016 ¶ 90; *see* Tr. 71:4–9, 72:1–73:3, 115:4–23.¹⁰ As support, Patent Owner and Dr. Oklobdzija cite to Culler's description of the Dragon protocol, identified by Dr. Oklobdzija as one of the most well-known write update protocols, in which Culler explains that there "is no *explicit* invalid (I) state" "because Dragon is an update-based protocol; the protocol always keeps the blocks in the cache up-to-date, so it is always okay to use the data present in the cache if the tag match succeeds." PO Resp. 27 (quoting Ex. 2011, 302) (emphasis added); Ex. 2016 ¶ 90 (quoting Ex. 2011, 302) (emphasis added); Ex. 1028, 302 (emphasis added). As a result, according to Patent Owner, even though the cached copies of data blocks in a write update protocol could be "consider[ed] . . . to be valid" because they are "usable" and "updated"—there is no express valid or invalid state. *See* Tr. 115:1–23. Moreover, Patent Owner argues and Dr. Oklobdzija opines that Pong's memory controller "cannot selectively filter and forward 'probes' to various processors based on validity in a write update protocol because every processor that has the cache line will have the line in a valid state." Ex. 2016 ¶ 90 (emphasis added); *see* Tr. 72:1–73:3.

¹⁰ From Patent Owner's arguments in the cited portions of the Response and hearing, we understand these arguments to be directed to Patent Owner's proposed construction of "states" as limited to cache coherency states. To the extent Patent Owner intends these arguments to apply to the broader construction of "states" that we have adopted, our analysis of these arguments below also applies to this construction.

Here, as we explain above, Pong is explicitly directed to cache coherence protocols, including the directory-based protocol implemented in the embodiment of Figure 4 on which Petitioner’s anticipation arguments rely. *See, e.g.*, Ex. 1003, [54], ¶¶ 1, 48, 51, 56–57, 69; Pet. 26; Reply 13–14; Ex. 1025 ¶ 16. Pong also discusses implementing the disclosed invention using a write update protocol, which it explains to be a “type[] of cache coherence protocol[.]” Ex. 1003 ¶ 69; *see id.* ¶ 48.

We determine that in Pong’s write update protocol, the bits in the presence bit vector represent cache coherency states—a valid state as well as an invalid state, meaning the data is not present or available in the cache, or a not present state.¹¹ First, as explained above in § II.B.1.b, even under Patent Owner’s construction of “states” as cache coherency states, the term encompasses these cache coherency states of invalid and not present. We agree with Petitioner that a bit in the presence bit vector—particularly when

¹¹ Patent Owner argues that even if Pong’s presence bit vector represents validity, it would fail to meet the recited “probe filtering information” because presence and validity are not sufficient to “practically” maintain cache coherency in a multi-processor system. PO Resp. 27; Ex. 2016 ¶ 94. Patent Owner’s argument is misplaced because whether presence and validity maintain cache coherency is not relevant to the “probe filtering information” claim language. Moreover, although the ’121 patent defines “probe” as “[a] mechanism for eliciting a response from a node to maintain cache coherency in a system,” Ex. 1001, 5:45–47; Inst. Dec. 6, the “practical[ity]” of cache coherency maintenance is not a claim requirement. We analyze, and find unpersuasive, similar arguments from Patent Owner in the section addressing its enablement arguments. *See supra* § II.C.1.b. We also note that we determine that the bits in Pong’s presence bit vector represent both valid and invalid states where the write update protocol is implemented, and Dr. Oklobdzija admitted, based on disclosures in Sorin, that it is “possible to implement” a multiprocessor system with a cache coherence protocol with only the states of valid and invalid. *See* Ex. 1026, 56:11–57:23; *see* Ex. 2010, 88.

not set—represents such an invalid, or not present, state. *See* Reply 4, 19; Tr. 52:4–17; Ex. 1025 ¶ 25; Pet. 10. In particular, Pong explains that “[w]hen the bit corresponding to a processor is set in the bit vector, the processor has a copy of the data block.” Ex. 1003 ¶ 51. In other words, when the bit is not set, the data block is not present in the processor’s cache. *See id.*; Tr. 123:10–15.

Second, regarding the “valid” state, there is no dispute on the record before us that a “valid” state is a cache coherency state, particularly given that Patent Owner itself repeatedly refers to a “valid” state throughout its briefing under its proposed construction of “states” as a cache coherency state. *See* PO Resp. 25 (contesting Petitioner’s argument regarding presence as improper under Patent Owner’s proposed construction of “states” but omitting comparable argument for Petitioner’s “‘valid’ state” argument); Ex. 2016 ¶¶ 86–88 (same); PO Resp. 25–27 (referring to a “‘valid’ state” in arguments under proposed construction of “states” as limited to cache coherency states); Ex. 2016 ¶¶ 88–90 (referring to a “‘valid’ state” and equating “validity” with “‘valid’ state”); *see also* Ex. 1025 ¶ 26; Ex. 1028, 280, 292, 628, 652 (referring to a valid state); Ex. 2010, 88, 163 (same). Indeed, Pong itself refers to “valid” as a “state.” Ex. 1003 ¶¶ 23, 61.

The “states” limitation requires only that the “probe filtering information” be “*representative of states* associated with selected ones of the cache memories.” Ex. 1001, 31:5–7, 32:14–15, 32:51–55 (emphasis added). Indeed, Patent Owner conceded at oral hearing that the recited probe filtering information need not “directly” be a cache coherency state and instead, could be “representative of one of those states.” Tr. 77:2–9; *see id.* at 74:11–15. Accordingly, the claim language does not require that the

“probe filtering information” include an express or explicit state, as Patent Owner’s arguments assume.

As discussed in our analysis above under our construction of “states,” we agree with Petitioner that Pong’s disclosures, as well as the supporting evidence of record, demonstrate that within the write update protocol, a set bit in Pong’s presence bit vector necessarily represents that a copy of the data block in the corresponding processor is present and valid. *See, e.g.*, Ex. 1003 ¶¶ 48, 51, 69; Ex. 1014 ¶ A-11; Pet. 26–27; Ex. 1025 ¶ 23; Reply 15, 17–18; Ex. 2016 ¶ 90; Tr. 51:17–52:18. We determine that this is sufficient for the bit to be “representative” of a “valid” cache coherency state—regardless of whether Pong expressly uses the term “state” to refer to the bits in the presence bit vector, or whether other write update protocols explicitly contain a “valid” state. *See In re Gleave*, 560 F.3d 1331, 1334 (Fed. Cir. 2009) (explaining that an anticipatory reference “need not satisfy an *ipsissimis verbis* test”); *Cont’l Can*, 948 F.2d at 1268.

Additionally, as to Patent Owner’s arguments that Pong does not filter probes based on a “valid” state where the write update protocol is implemented, *see* Ex. 2016 ¶ 90 (emphasis added); Tr. 72:1–73:3, independent claims 1 and 16 require that the “probe filtering unit” “transmit the probes only to selected ones of the processing nodes with reference to probe filtering information representative of states associated with selected ones of the cache memories,” Ex. 1001, 31:1–7, 32:10–15. Independent claim 25 requires that this transmission be performed based on an “evaluat[ion]” of whether any of the cache memories have a “valid copy of the memory line.” *Id.* at 32:48–55. Pong meets this claim language because the directory, i.e., the presence bit vector—the bits of which, as we determined above, represent whether each processor’s cache has the data

block in a “valid” state—is used to “filter[]” requests and “address[them] to the appropriate processors.” Ex. 1003 ¶ 56; *see id.* ¶¶ 48, 51, 57; *see* Pet. 40–41; Reply 25; Ex. 1014 ¶¶ A-9–A-13; Ex. 1025 ¶¶ 19–23. Pong meets the language of claims 1 and 16 for the additional reason that Pong’s directory does not forward the request to processors that lack of “copy of the data block of interest”—i.e., are in the invalid, or not present, state. Ex. 1003 ¶ 57.

In conclusion, even under Patent Owner’s proposed construction of “states” as limited to cache coherency states, Pong discloses “probe filtering information” “representative of states associated with selected ones of the cache memories,” as recited in claims 1, 16, and 25, because in Pong’s write update protocol, the bits in the presence bit vector represent cache coherency states, namely a valid state as well as an invalid state, meaning the data is not present or available in the cache, or a not present state.

(b) “probe”

Each of independent claims 1, 16, and 25 recites “probes” or a “probe.” Specifically, claims 1 and 16 recite a “probe filtering unit . . . operable to receive probes corresponding to memory lines from the processing nodes and to transmit the probes only to selected ones of the processing nodes.” Ex. 1001, 31:1–5, 32:11–14. Claim 25, in turn, recites “transmitting a probe from a first one of the processing nodes only to a probe filtering unit” and “transmitting the probe from the probe filtering unit only to the selected ones of the processing nodes.” *Id.* at 32:45–57. In the Institution Decision, we adopted the ’121 patent’s definition of “probe” as our construction of the term: “[a] mechanism for eliciting a response from a node to maintain cache coherency in a system.” *Id.* at 5:45–47; Inst. Dec. 6.

The parties dispute whether Pong discloses “probes” under this construction. Pet. 25–26; PO Resp. 28–30; Reply 19–21.

Petitioner argues that Pong’s read requests are “probes” because Pong discloses, as part of its cache coherent system in the embodiment of Figure 4, that these requests are forwarded by the memory controller (“probe filtering unit”) to other processors containing the requested data block, the processors respond with valid and cache coherent copies of the block, and the memory controller updates its presence bit vector. Pet. 25; Reply 19–21 (citing Ex. 1025 ¶¶ 28–29). According to Petitioner and Dr. Horst, these disclosed responses to a read request maintain cache coherency in Pong’s system. Pet. 25; Reply 19–21; Ex. 1014 ¶ A-8; Ex. 1025 ¶¶ 28–29. In addition, Petitioner asserts that in the write invalidate protocol, Pong’s write requests are “probes” because in response to a write request, the directory sends write invalidation requests to processors storing cached copies of the invalidated data block and these processors invalidate the data block. See Pet. 25–26; Reply 20–21; Ex. 1014 ¶ A-8; Ex. 1025 ¶ 30.

Patent Owner disputes Petitioner’s position that Pong’s read requests constitute “probes.” PO Resp. 28–30. Patent Owner argues that Pong lacks an “express teaching that its read requests are a ‘mechanism for eliciting a response from a node to maintain cache coherency.’” *Id.* at 28. Moreover, Patent Owner asserts that Pong’s disclosures regarding “how read requests are processed” and “how cache coherency is maintained during read requests” are effusive and contradictory, largely as a result of Pong’s alleged failure to specify its cache coherence protocol and cache coherence states. *Id.* Patent Owner also points to an alleged contradiction between Pong’s disclosures in paragraph 47, which discusses a requesting processor using a processor ID to address a read request to the processor with a valid copy of

the block, and paragraph 13, which refers to targeting a request to processors having a copy of the requested block. *Id.* at 29–30. According to Patent Owner, as a result of these alleged deficiencies, one of ordinary skill in the art cannot reliably determine whether and conclude that Pong’s read requests are mechanisms for maintaining cache coherency. *Id.* at 28–30.

Here, for the reasons given in § II.C.1.b, we are not persuaded that Patent Owner’s assertions relying on paragraph 47 of Pong undermine Petitioner’s showing that Pong discloses the recited “probes.” As we explain above, Petitioner’s anticipation arguments rely on the embodiment in Pong’s Figure 4—the discussion of which Pong explains takes precedence over any contradictory disclosures in previous embodiments—wherein the directory in the memory controller forwards requests to processors with a copy of the relevant data block, and we credit Dr. Horst’s testimony that one of ordinary skill would have understood paragraph 47 to be an alternative embodiment readily distinguishable from that in Figure 4. *See supra* § II.C.1.b; Reply 12–14, 19; Ex. 1025 ¶ 13; Ex. 1003 ¶¶ 55–57, Fig. 4.

Similarly, for reasons discussed above, we are not convinced by Patent Owner’s arguments that Pong fails to disclose adequately a cache coherence protocol or states and additionally, that these alleged deficiencies in Pong contribute to any failure to disclose the recited “probes.” As we explain above in § II.C.1.b, Pong explicitly is directed to cache coherence protocols, explains directory-based filtering approaches using a presence bit vector, and discloses implementing its system using either of “two types of cache coherence protocols: write invalidate and write update.” *See supra* § II.C.1.b; Ex. 1003, [54], ¶¶ 1, 48, 51, 69; Reply 13–14; Ex. 1025 ¶¶ 16, 28. In addition, as we explain in §§ II.B.1 and II.C.1.c.i.a, Pong discloses the

recited “states” under our construction of the term and under Patent Owner’s narrower proposed construction requiring cache coherency states.

Rather, we agree with Petitioner that Pong’s read requests meet the construction of “probe” because within Pong’s disclosed cache coherence protocols, Pong discusses several responses to a read request from other processors and the memory controller, which maintain cache coherency. *See* Pet. 25–26; Reply 19–21. First, upon receiving a “request[,]” including a read request, Pong’s memory controller “determines which processors have a copy of the data block of interest[] and forwards the request” to the appropriate processor or processors. Ex. 1003 ¶ 57; *see id.* ¶¶ 24, 32, 56; Inst. Dec. 17–18; Pet. 25; Reply 19, 22; Ex. 1014 ¶ A-9; Ex. 1025 ¶ 28. Second, each processor that has a copy of the requested data block responds with a valid, cache coherent copy. *See* Ex. 1003 ¶¶ 24, 43; Inst. Dec. 18; Pet. 25; Reply 20; Ex. 1014 ¶ A-8; Ex. 1025 ¶ 28. As explained above in § II.C.1.c.i.a, particularly in the write update protocol, every processor that has a copy of a data block has a valid copy. *See supra* § II.C.1.c.i.a; Ex. 1003 ¶¶ 48, 69.

Third, the memory controller updates the presence bit vector to reflect that the requesting processor now has a copy of the data block. *See* Ex. 1003 ¶ 51; Pet. 25–26; Reply 20; Ex. 1025 ¶¶ 28–30. Specifically, Pong explains that “[w]hen the bit corresponding to a processor is set in the [presence] bit vector, the processor has a copy of the data block.” Ex. 1003 ¶ 51. Based on this disclosure, we agree with Petitioner that upon a completed read request, the memory controller must update the presence bit vector by setting the corresponding bit to reflect that the requesting processor now has a copy of the relevant data block; otherwise, the presence bit vector would not reflect accurately which processors have a copy of the block. *See id.*;

Reply 20; Ex. 1025 ¶¶ 28–30; Pet. 25–26; Ex. 1014 ¶ A-9. In addition, we credit the testimony of Dr. Horst that based on this disclosure in Pong, one of ordinary skill would have understood that Pong’s memory controller updates the directory after a read request. Ex. 1025 ¶¶ 28–29; *see* Reply 20.

We agree with Petitioner’s assertions and credit Dr. Horst’s supporting testimony that these responses to read requests maintain, and are directed to maintaining, cache coherency by supplying valid, cache coherent copies of data blocks to requesting processors and maintaining a current presence bit vector. Pet. 25–26; Reply 19–20; Ex. 1025 ¶ 28; *see* Ex. 1014 ¶¶ A-8–A-9, A-11. For example, for the write invalidate protocol, we agree with Petitioner and Dr. Horst that the update to the presence bit vector resulting from a read request maintains cache coherency because the directory uses the presence bit vector to direct write invalidations to the processors storing cached copies of the relevant data block. *See* Ex. 1003 ¶¶ 32, 48–49, 52; Reply 20–21; Ex. 1025 ¶ 30; Pet. 25–26; Ex. 1014 ¶ A-8. In other words, the updated presence bit vector ensures that the write invalidations are sent to the proper processors. Reply 20–21; Ex. 1025 ¶ 30.

In sum, Pong’s read requests constitute the recited “probes” where either the write update or write invalidate protocol is implemented.

Turning to write requests, we also agree with Petitioner’s argument that write requests meet the ’121 patent’s definition of a “probe” where the write invalidate protocol is implemented.¹² *See* Pet. 25–26; Reply 20–21. Pong explains that in response to a write request, the directory sends a “write invalidation [request] only to the processors that have a copy of the data

¹² We note that we conclude below that although write requests meet the definition of a “probe,” they do not satisfy the “accumulating probe responses” limitation of claim 25. *See infra* § II.C.1.c.ii.b.

block.” Ex. 1003 ¶¶ 48–49, 52; Pet. 25–26; Reply 20–21. As a result, these processors invalidate their copy of the block. See Ex. 1003 ¶¶ 48, 52; Pet. 25; Ex. 1014 ¶ A-8. We credit Dr. Horst’s testimony that these responses to a write request, invalidating cached copies of the requested data block, serve to maintain cache coherency. Ex. 1014 ¶ A-8; see Pet. 25.

In conclusion, we determine that Pong’s read requests constitute the recited “probes” where either the write update or write invalidate protocol is implemented and Pong’s write requests constitute the recited “probes” where the write invalidate protocol is implemented.

ii. Disputed Limitations of Claim 25

In addition to limitations common to independent claims 1, 16, and 25, Patent Owner contests Petitioner’s showing regarding certain limitations specific to claim 25. See PO Resp. 34–37.

(a) “valid copy”

Patent Owner contests Petitioner’s showing that Pong discloses “evaluating the probe with the probe filtering unit to determine whether a valid copy of the memory line is in any of the cache memories, the evaluating being done with reference to probe filtering information,” as recited in claim 25. *Id.* at 37. Patent Owner relies on its argument—with respect to the “probe filtering information” limitation discussed above—that the bits in Pong’s presence bit vector do not represent a valid state. See *id.*; Ex. 2016 ¶ 111. In our analysis of the “probe filtering information” limitation in § II.C.1.c.i.a, we have addressed these arguments as well as this limitation of claim 25, and found Patent Owner’s argument unpersuasive where Pong’s write update protocol is implemented. See *supra* § II.C.1.c.i.a; Ex. 1003 ¶¶ 48, 51, 56–57; Pet. 40–41; Reply 25; Ex. 1014 ¶¶ A-9–A-13; Ex. 1025 ¶¶ 19–23.

(b) “*accumulating probe responses*”

The parties dispute whether Pong discloses “accumulating probe responses from the selected processing nodes with the probe filtering unit,” as recited in claim 25. Ex. 1001, 32:58–61; Pet. 42–43; PO Resp. 34–36; Reply 21–24. Petitioner, with supporting testimony from Dr. Horst, argues that Pong discloses, in the embodiment in Figure 4, that its memory controller (“probe filtering unit”) forwards requests to multiple processors having a copy of the relevant data block and the processors that have the requested data block in their cache respond with a copy. See Pet. 42–43 (citing Ex. 1014 ¶ A-17); Ex. 1014 ¶ A-17; Reply 21–23; Ex. 1025 ¶¶ 31–34. Moreover, Petitioner asserts, and Dr. Horst opines, that based on Pong’s disclosures regarding the memory controller buffering incoming and outgoing data blocks in queues, one of ordinary skill would have understood that the memory controller accumulates responses to a read request. Reply 23–24; Ex. 1025 ¶¶ 36–37; see Pet. 42–43 (citing Ex. 1014 ¶ A-17); Ex. 1014 ¶ A-17.

Patent Owner contests Petitioner’s position. PO Resp. 34–36. First, Patent Owner contends that paragraphs 13 and 47 of Pong demonstrate that Pong “does not disclose accumulating responses to a probe because in Pong’s system there is, at most, only one response to any request.” *Id.* at 35–36 (quoting Ex. 1003 ¶¶ 13, 47) (emphasis omitted). Second, Patent Owner, with supporting testimony from Dr. Oklobdzija, argues that even if Pong has multiple responses to a single probe, Pong does not disclose accumulating responses, because “there is nothing in Pong guaranteeing that responses from two different nodes will happen in such immediate succession that they will . . . both be in the same queue at the same time”—

i.e., “the first response could exit the queue before the second response enters.” *Id.* at 36; Ex. 2016 ¶ 110.

We first address Petitioner’s showing regarding write requests. We agree with Patent Owner that to the extent Petitioner asserts that responses to write requests satisfy the “accumulating probe responses” claim language, Petitioner has not demonstrated that Pong discloses the memory controller (“probe filtering unit”) accumulating, or gathering, responses to a write request. *See, e.g.*, Tr. 111:18–112:1. In addressing the “probe” limitation, Petitioner identifies only “invalidating cached copies of the requested data block” as a response to a write request. Pet. 25, 39–40; *see* Ex. 1014 ¶ A-8; *supra* § II.C.1.c.i.b. Furthermore, in analyzing the “accumulating probe responses” limitation, Petitioner never clearly explains or identifies any response to a write request that returns to or is gathered by the memory controller. *See, e.g.*, Pet. 42–43; Reply 23 (arguing only that a “read request” results in processors responding to memory controller with multiple copies of the requested data block); Ex. 1025 ¶ 34.

In addition, Dr. Horst never states an opinion that responses to a write request are gathered, or accumulated, with the memory controller. He testifies only that responses to “read requests” are accumulated with Pong’s memory controller. Ex. 1014 ¶ A-17 (“[T]he memory controller accumulates read responses”); Ex. 1025 at Heading D (“Memory Controller Accumulates Multiple Read Responses”); *id.* ¶¶ 36–37 (opining regarding “read requests” and “read responses”). In light of these deficiencies in Petitioner’s evidence, to the extent Petitioner argues that Pong’s write requests meet the “accumulating probe responses” limitation of claim 25, Petitioner has not shown that Pong’s write requests meet the claim language.

Turning to Pong's read requests, we agree with Petitioner on each of the disputed issues. First, Petitioner has shown that Pong's discussion, relevant to the embodiment of Figure 4, discloses multiple responses to a read request. See Ex. 1014 ¶ A-17; Pet. 42–43 (citing Ex. 1014 ¶ A-17); Reply 21–23; Ex. 1025 ¶¶ 31–34. Specific to the embodiment in Figure 4 in which the directory is integrated into the memory controller, Pong explains that the memory controller, particularly the directory, “filters [a] request and addresses it to the appropriate *processors*.” Ex. 1003 ¶ 56 (emphasis added); see Reply 22; Ex. 1025 ¶ 22. More specifically, Pong discloses that the directory, upon receiving a request, “determines which *processors* have a copy of the data block of interest, and forwards *the request* to . . . *these processors*.” Ex. 1003 ¶ 57 (emphases added); see Reply 22; Ex. 1025 ¶ 22; see also Ex. 1003 ¶ 32 (“The memory controller . . . may target *the request* to a *specific processor or set of processors* known to have a copy of the requested block.”) (emphases added); Ex. 1014 ¶ A-17. In sum, Pong describes the directory of the memory controller addressing and forwarding a read request to processors—i.e., more than one processor—with a copy of the relevant data block. See Ex. 1014 ¶ A-17; Ex. 1025 ¶ 32; Reply 22; Pet. 42–43 (citing Ex. 1014 ¶ A-17).

In addition, elsewhere in the description of the invention, Pong explains that a processor, upon receiving a read request, probes its cache and “if the processor has the requested block, it proceeds to provide it to the requesting processor.” Ex. 1003 ¶ 24; see Ex. 1014 ¶ A-17; Pet. 42–43 (citing Ex. 1014 ¶ A-17); Reply 22; Ex. 1025 ¶ 33. We agree with and credit Dr. Horst's testimony that this disclosure is consistent with the description of the embodiment in Figure 4. Ex. 1025 ¶ 33.

Accordingly, because Pong discloses that its memory controller sends a read request to multiple processors, determined to have a copy of the requested data block, and that each processor responds if it has a copy of the requested data block, we agree with Petitioner and Dr. Horst that Pong discloses multiple responses to a read request. *See* Ex. 1014 ¶ A-17; Reply 21–23; Ex. 1025 ¶¶ 31–34.

Patent Owner’s references and citations to paragraphs 13 and 47 of Pong do not convince us to the contrary, namely, that Pong discloses at most one response to a read request. *See* PO Resp. 35–36; Ex. 2016 ¶ 109. Paragraph 47 states that upon a read miss, “the requesting processor” can use a processor ID to “specifically address[] the read request to the processor that has the valid copy.” Ex. 1003 ¶ 47. Paragraph 13, in turn, explains that “only the processor or memory having a valid copy of the requested data block responds to the request.” *Id.* ¶ 13.

With respect to paragraph 47, as we explain above in §§ II.C.1.b and II.C.1.c.i.b, we credit Dr. Horst’s testimony that one of ordinary skill would have known and understood that paragraph 47 is an alternative, distinguishable embodiment to that disclosed in Pong’s Figure 4. *See supra* §§ II.C.1.b, II.C.1.c.i.b; *see also* Reply 12–14, 19; Ex. 1025 ¶ 13; Ex. 1003 ¶¶ 55–57, Fig. 4. Regarding paragraph 13, as we explain above in § II.C.1.c.i.a, this paragraph is in a “Summary” section of Pong and, as Petitioner argues, is not clearly tied to the embodiment in Figure 4, including a directory in the form of a presence bit vector—on which Petitioner’s anticipation arguments rely. *See supra* § II.C.1.c.i.a; Ex. 1003 ¶ 13; Reply 15–16; Ex. 1025 ¶ 20.

In addition, as detailed in § II.C.1.c.i.a, for the write update protocol, in which all cached copies are updated when a data block is updated in a

write operation, we agree with and have credited Dr. Horst’s testimony that Pong’s disclosures make clear that if a data block is present in cache, it is valid. *See supra* § II.C.1.c.i.a; Ex. 1003 ¶¶ 48, 69; Ex. 1014 ¶¶ A-10–A-11; Ex. 2016 ¶ 90 (“[E]very processor that has the cache line will have the line in a valid state.”). Thus, we are not convinced that paragraphs 13 and 47 of Pong demonstrate that in all embodiments, including, for example, that in Figure 4 implementing the write update protocol, only one processor has a valid copy of a data block and only that processor responds to the requesting processor. *See* PO Resp. 35–36; Tr. 113:11–16, 114:10–12.

In sum, paragraphs 13 and 47 neither clearly apply to the embodiment in Figure 4 nor do they negate the other disclosures of Pong, outlined above, demonstrating that a read request generates multiple responses.

Second, Petitioner’s evidence shows that such responses to a read request in Pong are accumulated by the memory controller. As Petitioner and Dr. Horst explain, Pong details how a responsive data block moves from the processors to the memory controller, through an outgoing queue and an incoming queue, to the requesting processor. *See* Pet. 34–35, 42–43; Ex. 1014 ¶ A-17; Reply 24; Ex. 1025 ¶ 35; Inst. Dec. 16. Pong explains that “[w]hen [a] processor is responding to a request for a data block,” it transfers, in first in, first out order, the data block to the outgoing queue of the memory controller that corresponds to that processor. Ex. 1003 ¶¶ 41, 43, Figs. 3–4; Inst. Dec. 16; Reply 24. The data block then is “buffered” in the outgoing queue before entering the bus of the memory controller. Ex. 1003 ¶ 41, Figs. 3–4. From the bus, the data block is “buffered” in the incoming queue that corresponds to the processor to which the data block is destined—i.e., the requesting processor. *Id.*

In addition, Dr. Horst's testimony, cited in the Petition and Reply, opines that in light of the manner in which Pong's processors respond to the read request and the memory controller buffers the responses in queues, Pong's memory controller accumulates responses to a read request. Ex. 1014 ¶ A-17; Ex. 1025 ¶¶ 36–37; Pet. 42–43 (citing Ex. 1014 ¶ A-17); Reply 24 (citing Ex. 1025 ¶¶ 36–37). Dr. Horst testifies that “the memory controller accumulates read responses from multiple responding processors in response to requests received from the requesting processor.” Ex. 1014 ¶ A-17. Dr. Horst also testifies that a person of ordinary skill in the art would have understood that Pong's system “would have simultaneously stored multiple read responses to a single request” in the memory controller. Ex. 1025 ¶ 36. Dr. Horst further explains that such a person would have understood and known that “during normal operation” of Pong's system, “at least in some cases,” “processors responding to a request in Pong's system . . . will respond in close enough succession that those responses will be simultaneously stored in the queues of Pong's memory controller” or, in other words, each response would not have been “forwarded through the memory controller's queues before the next response is received by the memory controller.” *Id.* Dr. Horst opines that “for at least a subset of the read requests, the responses to each read request would be stored simultaneously in the queues, as the queues would be unable to pass the individual responses through to the requesting processor without some delay.” *Id.* ¶ 37. Dr. Horst testifies that “[t]his is not simply a possibility, but a reality of normal operation of multiprocessor systems.” *Id.*

Inherency may not be established by “probabilities or possibilities.” *King Pharms., Inc. v. Eon Labs, Inc.*, 616 F.3d 1267, 1275 (Fed. Cir. 2010). The prior art, however, sufficiently shows inherency if the disclosure “is

sufficient to show that the natural result flowing from the operation as taught would result in the performance of the questioned function.” *Id.* (quoting *In re Oelrich*, 666 F.2d 578, 581 (CCPA 1981)); *Cont’l Can*, 948 F.2d at 1269 (quoting *Oelrich*, 666 F.2d at 581). Inherency also is established where the prior art, in its “normal and usual operation,” “will perform” the claimed function. *King Pharms.*, 616 F.3d at 1275 (quoting *In re Ackenbach*, 45 F.2d 437, 439 (CCPA 1930)); *Ackenbach*, 45 F.2d at 439; *see also King Pharms.*, 616 F.3d at 1276 (quoting *Hewlett-Packard Co. v. Mustek Sys., Inc.*, 340 F.3d 1314, 1326 (Fed. Cir. 2003)) (“[A] prior art product that sometimes, but not always, embodies a claimed method nonetheless teaches that aspect of the invention.”). To establish inherency, extrinsic evidence “must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill.” *Cont’l Can*, 948 F.2d at 1268.

Here, we credit and are persuaded by Dr. Horst’s testimony, outlined above. Ex. 1014 ¶ A-17; Ex. 1025 ¶¶ 36–37. We find Dr. Horst’s testimony sufficient to show that, when Pong’s system is operating in its “normal” or usual fashion, the memory controller “will” gather responses to a read request, by simultaneously storing them, and that this would naturally and necessarily result from the disclosed operation of Pong’s system. Ex. 1025 ¶¶ 36–37. Dr. Horst’s testimony further demonstrates that a person of ordinary skill in the art would recognize as such. *Id.* Accordingly, Petitioner’s evidence shows that Pong discloses that its memory controller (“probe filtering unit”) “accumulat[es],” or gathers, responses to a read request (“probe responses”) from processors the directory determines to have a copy of the requested data block (“selected processing nodes”), as claim 25 requires. *See, e.g., King Pharms.*, 616 F.3d at 1275–76; *Cont’l*

Can, 948 F.2d at 1268; *see also In re Preda*, 401 F.2d 825, 826 (CCPA 1968) (holding, in context of anticipation analysis, that “in considering the disclosure of a reference it is proper to take into account not only specific teachings of the reference but also the inferences which one skilled in the art would reasonably be expected to draw therefrom”).

Patent Owner does not offer persuasive evidence to the contrary. Patent Owner argues, and Dr. Oklobdzija opines, that Pong’s disclosures are insufficient to show “that responses from two different nodes” to the same probe “will happen in such immediate succession that they will . . . both be in” or necessarily be stored in “the *same queue at the same time.*” PO Resp. 36 (emphasis added); Ex. 2016 ¶ 110 (emphasis added). Yet Patent Owner has not provided sufficient reasoning, and we see none, that the ordinary meaning of the claim language, which Patent Owner proposed and we adopted—gathering two or more responses to a probe—requires that the responses be in the *same queue* of the memory controller at the same time.

In conclusion, Petitioner has shown, by a preponderance of the evidence before us, that Pong’s discussion of read requests discloses “accumulating probe responses from the selected processing nodes with the probe filtering unit,” as recited in claim 25.

iii. Remaining Limitations of Claims 1, 16, and 25

We have reviewed the evidence and arguments presented in the Petition as to the remaining limitations of claims 1, 16, and 25. Pet. 23–27, 35–44. Patent Owner does not contest that Pong discloses these limitations. *See* PO Resp. 25–37. Based on our review of the Petition, we find persuasive Petitioner’s arguments and supporting evidence, including citations to Pong’s disclosures and Dr. Horst’s testimony, and we adopt them

as the basis for our determination that Pong discloses these limitations of claims 1, 16, and 25. Pet. 23–27, 35–44.

iv. Conclusion

In conclusion, based on our review of the arguments and evidence in the Petition, Response, and Reply, we determine that Petitioner has demonstrated, by a preponderance of the evidence, that Pong discloses each limitation of claims 1, 16, and 25 and, therefore, anticipates these claims.

d. DEPENDENT CLAIMS 2, 3, 8, 11, AND 15

We next consider the parties’ arguments regarding whether Pong discloses the additional limitations of dependent claims 2, 3, 8, 11, and 15, each of which depends directly or indirectly from independent claim 1.

i. Claim 11 – “programmed”

Claim 11 recites “wherein each of the processing nodes is programmed to complete a memory transaction after receiving a first number of responses to a first probe, the first number being fewer than the number of processing nodes.” Ex. 1001, 31:49–53. Patent Owner disputes Petitioner’s showing that Pong discloses this limitation. PO Resp. 30–34.

Petitioner argues that Pong inherently discloses this limitation. See Pet. 30–31. Petitioner asserts that Pong describes an example in which the memory controller in a four-processor system sends a request from one processor to only two of the processors and, thus, the requesting processor “could only receive, at most, two responses.” *Id.* Therefore, according to Petitioner, the requesting processor in Pong “*must necessarily be programmed* to complete a memory transaction after receiving a first number of responses to a first probe . . . , where the first number is fewer than the number of processing nodes (two is fewer than four).” *Id.* at 31 (emphasis added); see Ex. 1014 ¶ A-21; Reply 21.

We, however, agree with Patent Owner that Petitioner has not demonstrated sufficiently that Pong inherently discloses the limitation because its processors are necessarily “programmed” to complete a memory transaction. Inherency is established where the “prior art necessarily functions in accordance with, or includes, the claimed limitations.” *In re Cruciferous Sprout Litig.*, 301 F.3d 1343, 1349 (Fed. Cir. 2002).

Patent Owner does not dispute that Pong discloses that its processors “complete a memory transaction after receiving” a number of responses fewer than the number of processing nodes, as recited in claim 11. *See* PO Resp. 30–34. Patent Owner, however, contests Petitioner’s showing that Pong’s processors are “programmed” to do so—as claim 11 requires—arguing that “nothing in Pong requires that [its processors] be configured to complete memory transactions using programming” “as opposed to using any other mechanism for implementing the operation described by Pong.” *Id.* at 31, 33. For example, according to Patent Owner, the functionality of Pong’s processors could be implemented via hardwired logic, which does “not meet the claim requirement that ‘each of the processing nodes is *programmed* to complete a memory transaction.’” *Id.* at 31; *see id.* at 32–33. Patent Owner supports these arguments with testimony from Dr. Oklobdzija, who opines that “there is no reason why [Pong’s] processors must necessarily be configured using programming as opposed to using any other mechanism for implementing the operation described by Pong” and that the functionality of Pong’s processors in completing a memory transaction “could be implemented via hardwired logic as opposed to programming.” Ex. 2016 ¶¶ 101–03.

More specifically, Patent Owner—with support from Exhibit 2015, ELECTRICAL ENGINEER’S REFERENCE BOOK (16th ed. 2003), cited in its

Response—argues that there are “three main classes of digital systems,” including “traditional hard-wired systems,” “truly programmable systems,” and a “middle class . . . programmable logic.” Tr. 93:4–25, 99:21–23; *see* Ex. 2015, 15/3; PO Resp. 15–17. Patent Owner concedes that both truly programmable systems *and* the middle class of programmable logic, which includes field programmable gate arrays (“FPGAs”), qualify as “programmed.” Tr. 95:11–22, 100:3–4, 101:3–102:8, 107:20–22; Ex. 2015, 15/3; *see* Ex. 1001, 7:49–52 (referring to a FPGA as a “programmable chip”). According to Patent Owner, however, traditional hard-wired circuitry is not programmed. Tr. 95:22–25, 99:21–23, 102:5–8, 107:23–25; *see* PO Resp. 15–17, 31–33; Ex. 2016 ¶¶ 38–39; Ex. 2015, 15/3.

Patent Owner provides additional evidence to support this position. According to Dr. Oklobdzija, the difference between traditional hardwired logic and programmable systems is that in a hardwired system, the “sequence of operations is governed by the physical interconnection of the digital processing elements,” which are “inflexible because the design is specific to a particular processing function: if the processing function is changed, then the processing elements and their interconnections have to be altered.” Ex. 2016 ¶ 39 (citing Ex. 2015, 15/3). Programmable systems, on the other hand, can be controlled using “a prescriptive program of instructions (i.e. software).” *Id.* (citing Ex. 2015, 15/3). In other words, the difference is that devices falling within the middle category, programmable logic, can be “configured” or “designed” using software, while traditional hardwired logic cannot. *See id.* (citing Ex. 2015, 15/3); PO Resp. 15–17,

31–33 (citing Ex. 2015, 15/3); Ex. 2015, 15/3; Ex. 1008, 16, 170, 299¹³ (explaining that a “key element[]” of an FPGA is that a designer “uses custom software, tailored to each programming technology and FPGA architecture, to design and implement the programmable connections” and that one component of an FPGA is “design software” that allows it to be “program[ed]”).

Patent Owner also cites to the MICROSOFT COMPUTER DICTIONARY, which distinguishes hardwired from programming. PO Resp. 15 (citing Ex. 2012, 214). Specifically, the dictionary defines “hardwired” as “[b]uilt into a system using hardware such as logic circuits, *rather than accomplished through programming.*” Ex. 2012, 214 (emphasis added); *see id.* at 359 (defining programming).

Petitioner fails to proffer evidence sufficient to rebut Patent Owner’s evidence that Pong’s disclosure of a processor completing a memory transaction does not necessarily disclose that the processor is programmed to do so. *See* Pet. 30–31; Reply 5–17, 21. Dr. Horst, in his original declaration, asserts that because Pong discloses an example with a four-processor system in which a requesting processor could receive, at most, two responses, “the requesting processor *must necessarily be programmed* to complete a memory transaction after receiving a first number of responses to a first probe.” Ex. 1014 ¶¶ A-19–A-21 (emphasis added). This assertion, however, is conclusory, and does not sufficiently explain why the processor must be programmed or exclude possibilities other than programming. *See id.* ¶¶ A-21–A-22; PO Resp. 32; Ex. 2016 ¶ 103. In particular, Dr. Horst’s testimony provides no discussion or explanation as to whether a processor

¹³ In our citations to Exhibit 1008, Smith, we use the original page numbers of the book, rather than the exhibit page numbers added by Petitioner.

composed solely of hardwired logic would necessarily be programmed and whether the functionality described in Pong could not be implemented using hardwired logic. *See* Ex. 1014 ¶¶ A-21–A-22; PO Resp. 32.

In his reply declaration, Dr. Horst testifies that “the term ‘programmed’ is commonly used to describe the design and configuration of hardwired logic.” Ex. 1025 ¶ 2. Dr. Horst, however, supports this conclusory statement only by reference to FPGAs, which fall within Patent Owner’s described middle class of devices, programmable logic devices, which Patent Owner concedes are programmed. *Id.* ¶¶ 3–5; *see* Tr. 95:11–22, 100:3–4, 101:3–102:8, 107:20–22; Ex. 2015, 15/3 (explaining that FPGAs are “programmable . . . logic devices . . . that can be programmed” to implement certain functions); Ex. 1001, 7:59–52 (referring to a FPGA as a “programmable chip”); Ex. 1008, 16, 170, 299. Thus, Dr. Horst’s testimony does not address all possible means by which the processor disclosed in Pong could be implemented or even all hardwired logic—only a subset of hardware devices. Ex. 1025 ¶ 3.

Dr. Horst does not provide testimony sufficient to support Petitioner’s position that all processors, even those that are purely hardwired, are necessarily programmed to complete memory transactions. *See, e.g.*, Tr. 99:21–100:2; Ex. 1014 ¶¶ A-21–A-22; Ex. 1025 ¶¶ 2–6. Nor does Dr. Horst’s testimony exclude the possibility that Pong’s processors implement traditional hardwired logic to complete a memory transaction upon receiving responses. *See, e.g.*, Ex. 1014 ¶¶ A-21–A-22; Ex. 1025 ¶¶ 2–6. Petitioner does not point to any other evidence adequate to show that Pong’s disclosure of a processor completing memory transactions within its system, by itself, necessarily discloses that the processor is programmed to do so. *See, e.g.*, Pet. 30–31; Reply 5–7, 21.

For the reasons given, we agree with Patent Owner that Petitioner has not demonstrated, by a preponderance of the evidence, that Pong discloses “each of the processing nodes is programmed to complete a memory transaction after receiving a first number of responses to a first probe,” as recited in claim 11, and thus that Pong anticipates claim 11.

ii. Claim 15 – “operable to accumulate responses to each probe”

Patent Owner also disputes that Pong discloses the additional limitation of claim 15 reciting “wherein the probe filtering unit is operable to accumulate responses to each probe.” PO Resp. 34–26. Petitioner and Patent Owner proffer substantially the same arguments for this limitation of claim 15 as they make regarding the similar “accumulating probe responses” limitation of independent claim 25. Ex. 1001, 32:58–61; see Pet. 34–35, 42–43; PO Resp. 34–36; Reply 21–24. For the reasons given in § II.C.1.c.ii.b in our analysis of this limitation of claim 25, we determine that Petitioner has shown, by a preponderance of the evidence, that Pong discloses “wherein the probe filtering unit is operable to accumulate responses to each probe.” See *supra* § II.C.1.c.ii.b. In addition, we determine Pong discloses this limitation of claim 15 for the additional reason that Pong’s memory controller is “operable to,” or capable of, gathering responses to a read request in the incoming queue for the requesting processor before the responses are sent to the requesting processor. See Ex. 1003 ¶¶ 41, 43, Figs. 3–4; Pet. 34–35; Ex. 1014 ¶¶ A-17–A-18; see also Ex. 1025 ¶¶ 36–37.

iii. Remaining Limitations of Dependent Claims 2, 3, 8, and 15

Petitioner asserts that Pong discloses the additional limitations of dependent claims 2, 3, 8, and 15, and Patent Owner does not dispute these assertions. See Pet. 27–30, 34–35; PO Resp. 30–37. Based on our review of the Petition’s arguments and evidence, including citations to Pong and

Dr. Horst’s supporting declaration, we find Petitioner’s reasoning and evidence persuasive and adopt them as the basis for our determination that Petitioner has demonstrated that Pong discloses each of the additional limitations of claims 2, 3, 8, and 15. Pet. 27–30, 34–35.

iv. Conclusion

For the reasons given, in addition to our reasoning above regarding the anticipation of independent claim 1, we conclude Petitioner has shown by a preponderance of the evidence that claims 2, 3, 8, and 15 are anticipated by Pong. Petitioner, however, has not shown by a preponderance of the evidence that claim 11 is unpatentable as anticipated by Pong.

2. OBVIOUSNESS OVER PONG AND SMITH

We turn to the instituted obviousness ground alleging that claims 17–24 of the ’121 patent would have been obvious over Pong and Smith. *Id.* at 55–59; Inst. Dec. 28–30. We agree with Petitioner’s undisputed assertion that Smith was published in 1997 and, therefore, is prior art to the ’121 patent under 35 U.S.C. § 102(b). *See* Pet. 4; Ex. 1008, copyright page.

a. OVERVIEW OF SMITH

Smith explains the development and design of application-specific integrated circuits (“ASICs”). Ex. 1008, 1–38. Smith provides examples of ASICs, including “a chip designed to handle the interface between memory and a microprocessor for a workstation.” *Id.* at 4. Smith further explains “the sequence of steps to design an ASIC,” which includes “[e]ntering the design into an ASIC design system, either using a *hardware description language (HDL)* or schematic entry”; “produc[ing] a *netlist*”; and “[p]relayout” and “[p]ostlayout simulation.” *Id.* at 16–18. Smith also states that masks are “the tooling . . . used to manufacture [an] ASIC.” *Id.* at 28.

b. ALLEGED LACK OF ENABLEMENT OF PONG AND SMITH

Patent Owner argues the asserted ground of obviousness over Pong and Smith must fail because Pong does not enable the independent claims of the '121 patent and Smith does not remedy the alleged deficiencies in Pong. PO Resp. 17, 24. In § II.C.1.b above, we analyze Patent Owner's arguments that Pong does not enable the independent claims of the '121 patent and find them unpersuasive. For the same reasons, the preponderance of the evidence before us does not support Patent Owner's enablement arguments directed to Pong in combination with Smith.

c. OBVIOUSNESS DISCUSSION

Claims 17, 19, and 24 depend directly from independent claim 16, and claims 18 and 20–23 depend indirectly from claim 16. Ex. 1001, 32:16–39. Claim 17 recites “[a]n integrated circuit comprising the probe filtering unit of claim 16.” *Id.* at 32:16–17. Claim 19 adds the following limitation to claim 16: “[a]t least one computer-readable medium having data structures stored therein representative of the probe filtering unit of claim 16.” *Id.* at 32:21–23. Claim 24 recites the limitation “[a] set of semiconductor processing masks representative of at least a portion of the probe filtering unit of claim 16.” *Id.* at 32:37–39.

Petitioner contends that Pong's memory controller is equivalent to the “probe filtering unit” recited in claim 16, yet concedes “Pong does not explicitly describe the hardware and/or software used to implement the memory controller.” Pet. 56; Ex. 1014 ¶ A-34. Petitioner, with supporting testimony from Dr. Horst, argues that it would have been obvious to one of ordinary skill in the art to implement Pong's memory controller as an ASIC in view of Smith's teachings. Pet. 56–57; Ex. 1014 ¶¶ A-36–A-37.

According to Petitioner, ASICs can “reduce cost and improve reliability” of

a microelectronic system and “a chip designed to handle the interface between memory and a microprocessor,” such as Pong’s memory controller, “is appropriately designed as an ASIC.” Pet. 56–57 (quoting Ex. 1008, 3–4); Ex. 1014 ¶¶ A-35–A-36. Dr. Horst opines that “anyone designing cache-coherent multiprocessors at the time of the [’]121 patent’s priority date would have considered designing the logic using ASICs because it was one of a small, limited number of options that the designer could have tried in order to implement circuits for cache coherence.” Ex. 1014 ¶ A-37; *see* Pet. 57; Inst. Dec. 29. Dr. Horst further explains that one of ordinary skill would have understood Smith’s ASIC “design software and the design it produces are data structures stored on one or more computer readable mediums.” Ex. 1014 ¶ A-39; *see* Pet. 58. Moreover, Petitioner asserts and Dr. Horst opines that Smith teaches that implementing Pong’s memory controller as an ASIC would require a set of semiconductor processing masks representative of at least a portion of the memory controller. Pet. 59; Ex. 1014 ¶ A-41.

In responding to Petitioner’s challenge of claims 17–24, Patent Owner relies on its arguments that Pong does not disclose “states” and “probes,” as recited in independent claim 16. PO Resp. 37; *see* Ex. 2016 ¶ 112. As discussed above in addressing the ground of anticipation by Pong, however, we have determined that Petitioner has shown by a preponderance of the evidence that Pong discloses all limitations of claim 16, despite Patent Owner’s arguments to the contrary. *See supra* § II.C.1.c.

We have reviewed the evidence and arguments presented in the Petition and Response regarding the instituted ground that claims 17–24 would have been obvious over Pong and Smith. *See* Pet. 55–59; PO Resp. 37. With respect to claims 17–24, we find persuasive Petitioner’s

arguments and evidence, including citations to Pong, Smith, and Dr. Horst's testimony, and we adopt them as the basis for our decision. Pet. 55–59. We determine that Petitioner has demonstrated that Smith teaches the additional limitations of claims 17–24, and that the combination of Pong and Smith renders these claims obvious.

For these reasons, in addition to our reasoning outlined above regarding the anticipation of independent claim 16 by Pong, we conclude Petitioner has shown by a preponderance of the evidence that claims 17–24 are unpatentable as obvious over Pong and Smith.

III. PATENT OWNER'S MOTION TO AMEND

Because we determine that claims 16–24 are unpatentable, we turn to Patent Owner's contingent request to enter proposed substitute claims 26–34. Mot. 1. Patent Owner states that proposed substitute claim 26 recites original claim 16 and adds the following new limitations:

wherein said states comprise cache coherency states of a cache coherence protocol, and wherein said cache coherence protocol includes at least a modified state, an exclusive state, a shared state, and an invalid state, and wherein said probe filtering unit is coupled to a coherent protocol interface and a non-coherent protocol interface

Id. Patent Owner further explains that “[e]ach of the proposed substitute claims 27–34 were drafted by first converting the respective original claim 17–24 to independent form, and then adding the same proposed new limitations.” *Id.* at 1–2.

As the moving party, Patent Owner bears the burden of proof to establish that it is entitled to the relief requested. *See* 37 C.F.R. § 42.20(c). Entry of proposed amendments is not automatic, but occurs only upon the patent owner having met the requirements of 37 C.F.R. § 42.121 and

demonstrating, by a preponderance of the evidence, the patentability of the proposed substitute claims. *Nike, Inc. v. Adidas AG*, 812 F.3d 1326, 1332–34 (Fed. Cir. 2016); *see Idle Free Sys., Inc. v. Bergstrom, Inc.*, IPR2012-00027, slip op. at 7–8 (PTAB June 11, 2013) (Paper 26) (informative). For the reasons explained below, we conclude that Patent Owner has not met its burden with respect to proposed substitute claims 26–34.

A motion to amend claims must clearly identify the support for the proposed substitute claims. In particular, 37 C.F.R. § 42.121(b)(1) requires the patent owner to set forth the support in the original disclosure of the patent for each proposed substitute claim. Similarly, under 37 C.F.R. § 42.121(b)(2), the patent owner must set forth the support in an earlier-filed disclosure for each claim for which benefit of the filing date of the earlier filed disclosure is sought. 35 U.S.C. § 112 ¶ 1¹⁴ sets forth an enablement requirement, providing that the specification shall describe “the manner and process of making and using [the invention], in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the [invention].”¹⁵

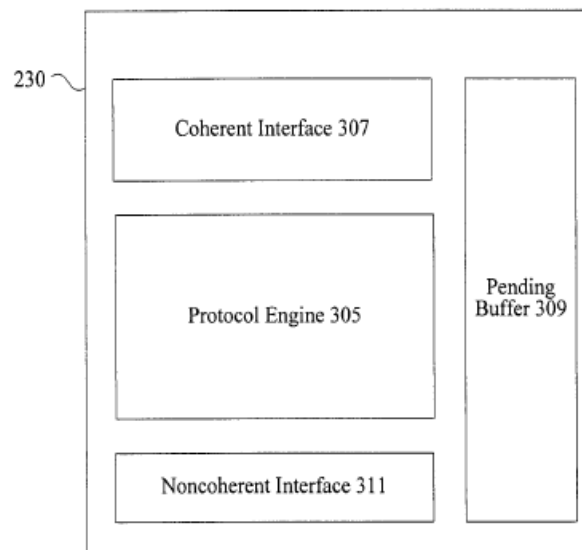
¹⁴ Section 4(c) of the AIA re-designated 35 U.S.C. § 112 ¶ 1, as 35 U.S.C. § 112(a). Because the ’121 patent has an effective filing date before September 16, 2012, the effective date of the relevant section of the AIA, we will refer to the pre-AIA version of 35 U.S.C. § 112.

¹⁵ 35 U.S.C. § 316(d)(3) provides that an amendment under 35 U.S.C. § 316(d) may not enlarge the scope of the claims of the patent or introduce new matter. Similar to 35 U.S.C. § 132’s prohibition on adding new matter to claim amendments, the prohibition of new matter in proposed substitute claims places a burden on the moving party to show those claims adhere to the requirements of 35 U.S.C. § 112, ¶ 1. *See Ariad Pharms., Inc. v. Eli Lilly & Co.*, 598 F.3d 1336, 1348 (Fed. Cir. 2010) (en banc) (“[P]rohibiting adding new matter to the claims has properly been held enforceable under § 112, first paragraph.”).

Automotive Techs. Int'l v. BMW of N. Am., Inc., 501 F.3d 1274, 1282 (Fed. Cir. 2007). This requirement is satisfied when “one skilled in the art, after reading the specification, could practice the claimed invention without undue experimentation.” *Id.* (internal quotation marks omitted).

Patent Owner asserts that support for the proposed substitute claims, including the newly proposed limitations, is found in the originally filed application leading to the '121 patent, U.S. Patent Application No. 10/966,161 (Ex. 2020, “the '161 Application”), and the '347 Application (Ex. 2006),¹⁶ to which the '121 patent claims priority. Mot. 2–10.

Specifically, Patent Owner asserts that the new limitation “wherein said probe filtering unit is coupled to a coherent protocol interface and a non-coherent protocol interface” (“the protocol interface limitation”) is supported by both the '347 and '161 Applications. *Id.* at 9–10. Patent Owner points to Figure 3, reproduced below, which is substantively identical in both the '347 and '161 Applications and in the '121 patent. *Id.* at 9.



¹⁶ Exhibit 2006 has two sets of page numbers. We refer to the original page numbers at the bottom of each page.

Figure 3 shows an example interconnection controller 230, including protocol engine 305 “configured to handle packets such as probes and requests received from processors in various clusters of a multiprocessor system.” Ex. 2020, 12:28–31; Ex. 2006, 11:20–23; Ex. 1001, 7:53–58. Interconnection controller 230 also includes coherent interface 307 and non-coherent interface 311. Ex. 2020, 13:11–16; Ex. 2006, 12:4–10; Ex. 1001, 8:5–11. In addition, Patent Owner relies on several lines of text associated with Figure 3 in the ’347 and ’161 Applications. Mot. 9–10 (citing, *inter alia*, Ex. 2020, 13:11–17; Ex. 2006, 11:13–14, Fig. 3).

Petitioner argues that the disclosure Patent Owner relies upon does not enable the protocol interface limitation, but instead “simply illustrates the broad proposition that the cache coherence controller ‘can also include other interfaces such as a non-coherent protocol interface 311 for communicating with I/O devices.’” Opp. 5 (quoting Ex. 1001, 8:8–11). According to Petitioner, the disclosure “identifies where the non-coherent protocol interface is located, but provides no details for how it is implemented.” *Id.*

We agree with Petitioner that Patent Owner has not met its burden to show that its proposed substitute claims meet the enablement requirement of § 112 ¶ 1. See Tr. 140:3–18. Specifically, Patent Owner has not shown that the specifications of the ’347 or ’161 Applications enable the protocol interface limitation recited by each of the proposed substitute claims.

Considering first the specifications of the ’347 and ’161 Applications, although Patent Owner points to several lines and one figure as describing the limitation, these disclosures simply identify the existence of a cache coherency controller with both a coherent and non-coherent protocol interface without providing any detail of how such a cache coherency controller would be implemented. See Ex. 2020, 12:20–23, 13:11–17;

Ex. 2006, 11:12–15, 12:4–10; Ex. 1001, 7:44–48, 8:5–14. Figure 3 represents a concept of a cache coherence controller with two interfaces—but lacks any details that would show one skilled in the art how to make or use a cache coherence controller with two interfaces. See Ex. 2020, Fig. 3; Ex. 2006, Fig. 3. The specifications describe Figure 3 as a “diagrammatic representation” of a cache coherence controller, and does not purport to show details of how the two interfaces would be implemented together in such a controller. See Ex. 2020, 5:10, 12:20–13:21; Ex. 2006, 5:9, 11:14–12:14; Ex. 1001, 3:23–24, 7:53–8:19. This is supported by the testimony of Dr. Oklobdzija, who testified that Figure 3 (along with Figures 2 and 17 of the ’121 patent) “point[s] to a non-coherent interface and the interconnections” but does not provide details regarding how to implement a coherent and non-coherent interface and instead “le[aves] someone . . . with . . . ordinary skill in the art to . . . figure it out.” Ex. 1026, 90:5–93:1, 95:14–96:10. Dr. Oklobdzija also agreed that “one of ordinary skill would not have already had in their knowledge how to build a system with [both] interface[s].” *Id.* at 90:5–17.

Moreover, the textual discussion of Figure 3, which is the only description of the protocol interfaces relied upon by Patent Owner, provides scant detail concerning how a cache coherence controller with both interfaces is built or operated. The specifications state the following:

The cache coherence controller 230 can also be configured to handle a non-coherent protocol to allow communication with I/O devices. In one embodiment, the cache coherence controller 230 is a specially configured programmable chip such as a programmable logic device or a field programmable gate array.

Figure 3 is a diagrammatic representation of one example of a cache coherence controller 230. According to various

embodiments, the cache coherence controller includes a protocol engine 305 configured to handle packets such as probes and requests received from processors in various clusters of a multiprocessor system.

....

The cache coherence controller has an interface such as a coherent protocol interface 307 that allows the cache coherence controller to communicate with other processors in the cluster as well as external processor clusters. According to various embodiments, each interface 307 and 311 is implemented either as a full crossbar or as separate receive and transmit units using components such as multiplexers and buffers. The cache coherence controller can also include other interfaces such as a non-coherent protocol interface 311 for communicating with I/O devices. It should be noted, however, that the cache coherence controller 230 does not necessarily need to provide both coherent and non-coherent interfaces. It should also be noted that a cache coherence controller in one cluster can communicate with a cache coherence controller in another cluster.

Ex. 2006, 11:14–12:14; *see* Ex. 2020, 12:22–13:21; Ex. 1001, 7:44–8:19.

This general description fails to provide a structure or description of how a person having ordinary skill in the art would make or use a cache coherence controller with both types of interfaces. The most detailed language— “[i]n one embodiment, the cache coherence controller 230 is a specially configured programmable chip such as a programmable logic device or a field programmable gate array” and “[a]ccording to various embodiments, each interface 307 and 311 is implemented either as a full crossbar or as separate receive and transmit units using components such as multiplexers and buffers”—does not specifically describe how to implement the system. Instead, as confirmed by Dr. Oklobdzija, the sentence related to the cache coherence controller 230 “gives kind of [a] suggestion to – that this can be implemented as a programmable chip” and the sentence describing

implementation of the interfaces is simply an “implementation hint.”

Ex. 1026, 100:18–102:20. According to Dr. Oklobdzija, the implementation details are “up to the engineers to – to figure out further.” *Id.* at 101:10–20.

Indeed, Dr. Oklobdzija testified that prior to the filing date of the ’121 patent, a person of ordinary skill in the art would not have known to use *both* a coherent interface *and* a non-coherent interface at once. Ex. 2019 (Dr. Oklobdzija Decl. for Mot. to Amend) ¶¶ 11–12. Specifically, Dr. Oklobdzija testifies that, a reference that “reflects the state of the art of coherent and non-coherent interfaces” as of the alleged effective filing date of the ’121 patent indicated that a system could use only one or the other of a coherent or non-coherent interface, but not both. *Id.* ¶ 12 (citing Ex. 2038, 9). Noticeably absent in Patent Owner’s Motion is reference to any detailed discussion in the ’121 patent or the ’347 and ’161 Applications of the how the cache coherence controller would operate with both interfaces at once, let alone such full, clear, concise, and exact terms as to enable any person skilled in the art, to make and use the system. Given the testimony of Dr. Oklobdzija, we cannot determine that Figure 3 and the few corresponding lines of description apprise one of ordinary skill how to make and use a cache coherence controller with both a coherent and non-coherent protocol interface. *See Automotive Techs.*, 501 F.3d at 1281–85 (holding that electronic sensors were not enabled because the specification’s general description failed to provide a structure or description of how a person having ordinary skill in the art would make or use the sensor).

This is true despite Dr. Oklobdzija’s conclusory statement to the contrary. Ex. 2042 (Dr. Oklobdzija Reply Decl. for Mot. to Amend) ¶ 7 (“[B]ased simply on my review during the deposition itself, I believed, and I continue to believe, that the specification of the ’121 Patent contains a

number of ‘implementation hint[s]’ which would enable one of skill in the art to practice the ‘coherent protocol interface’ and ‘non-coherent protocol interface’ limitations without undue experimentation.”). “It is the specification, not the knowledge of one skilled in the art, that must supply the novel aspects of an invention in order to constitute adequate enablement.” *Genentech, Inc. v. Novo Nordisk A/S*, 108 F.3d 1361, 1366 (Fed. Cir. 1997). Although the knowledge of one skilled in the art is relevant, the novel aspect of an invention must be enabled in the patent. As noted above, according to Dr. Oklobdzija, prior to filing of the ’121 patent, it was suggested that a system could use only one or the other of a coherent or non-coherent interface, but not both. Ex. 2019 ¶¶ 11–12. Given that the allegedly novel aspect of the invention is a cache coherence controller that uses both interfaces, which was unknown to one of ordinary skill at the time, it is insufficient to merely provide “implementation hints” using known technologies to create that novel cache coherence controller. As was the case in *Genentech*, the portions of the specifications of the ’347 and ’161 Applications relied upon by Patent Owner here provide “only a starting point, a direction for further research” on using a probe filtering unit with both protocol interfaces; they do not provide guidance to a person of ordinary skill in the art on how to make or use such system. 108 F.3d at 1366. Patent Owner, thus, fails to show that the specifications of the ’347 and ’161 Applications provide “reasonable detail” sufficient to enable use of a probe filtering unit coupled to a coherent protocol interface and a non-coherent protocol interface. *Id.*

In conclusion, we determine Patent Owner has not met its burden of showing that the protocol interface limitation, recited in each of proposed substitute claims 26–34, is properly enabled under 35 U.S.C. § 112 ¶ 1.

Furthermore, because of the manner in which Patent Owner’s Motion addresses Koster—a prior art U.S. patent of record as Exhibit 1009 that was filed on July 13, 2004 and issued on April 13, 2004—Patent Owner’s failure to demonstrate enablement of the proposed substitute claims results in a failure to demonstrate novelty and non-obviousness of the proposed substitute claims over the prior art of record. When arguing how the limitations of the proposed substitute claims distinguish them from the disclosures of the references of record, Patent Owner’s Motion addresses Koster only by asserting that “[a]s to . . . Koster . . . , because, as discussed above, all of the substitute claims find support in the [’]347 Application, . . . Koster . . . is not prior art to any of the proposed substitute claims.” Mot. 22. Because Patent Owner does not persuade us that the proposed substitute claims are enabled by the disclosure in the ’347 Application, we find unpersuasive Patent Owner’s suggestion that the proposed substitute claims are entitled to the filing date of the ’347 Application. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (“For a patent to claim priority from the filing date of its provisional application, . . . the specification of the *provisional* must . . . enable an ordinarily skilled artisan to practice the invention *claimed* in the *non-provisional* application.”) (internal citation and quotations omitted); *Hyatt v. Boone*, 146 F.3d 1348, 1352 (Fed. Cir. 1998) (“The earlier application must contain a written description of the subject matter of the interference count, and must meet the enablement requirement.”); *see also Novo Nordisk Pharm., Inc. v. Bio-Tech. Gen. Corp.*, 424 F.3d 1347, 1359 (Fed. Cir. 2005) (“[If] the 1983 PCT application was . . . not enabled, Novo would not be able to rely upon the application’s priority date to overcome the Brewer patent.”). Accordingly, we also find unpersuasive Patent Owner’s assertion

that Koster is not prior art to the proposed substitute claims. With no other basis for asserting that the proposed substitute claims are patentable over Koster (by itself or in combination with other references of record), we conclude that Patent Owner has not met its burden of demonstrating patentability of the proposed substitute claims over the references of record. *See Microsoft Corp. v. Proxyconn, Inc.*, 789 F.3d 1292, 1307 (Fed. Cir. 2015) (upholding Board’s requirement that patent owner “show patentable distinction [of the substitute claims] over the prior art of record”); Opp. 1–4.

Thus, Patent Owner has not met its burden for proposed substitute claims 26–34.

IV. CONCLUSION

In conclusion, Petitioner has shown by a preponderance of the evidence that claims 1–3, 8, and 15–25 of the ’121 patent are unpatentable: (1) Pong anticipates claims 1–3, 8, 15, 16, and 25 and (2) Pong and Smith render obvious claims 17–24. Petitioner, however, has not demonstrated that claim 11 is unpatentable as anticipated by Pong. As to Patent Owner’s motion to amend, Patent Owner has not shown that its proposed substitute claims 26–34 are enabled or patentable over the prior art.

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that Petitioner has shown by a preponderance of the evidence that claims 1–3, 8, and 15–25 of the ’121 patent are unpatentable;

FURTHER ORDERED that Patent Owner’s motion to amend is denied; and

FURTHER ORDERED that, because this is a Final Written Decision, the 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.

IPR2015-00159

Patent 7,296,121 B2

PETITIONER:

W. Karl Renner

Roberto J. Devoto

Michael Rueckheim

FISH & RICHARDSON P.C.

axf@fr.com

IPR39521-007IP1@fr.com

Walter E. Hanley, Jr.

Zaed M. Billah

KENYON & KENYON LLP

whanley@keyon.com

zbillah@kenyon.com

PATENT OWNER:

Jonathan D. Baker

Gurtej Singh

Michael D. Saunders

FARNEY DANIELS PC

jbaker@farneydaniels.com

tsingh@farneydaniels.com

msaunders@farneydaniels.com

MemoryIntegrityIPR@farneydaniels.com