

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

PARTEC AG,
Patent Owner.

Case IPR2025-00318
Patent 11,537,442 B2

PATENT OWNER'S RESPONSE UNDER 37 C.F.R. § 42.120

Mail Stop "PATENT BOARD"
Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

EXHIBIT LIST

Exhibit	Description
EX2001	Complaint for Patent Infringement, <i>PARTEC AG and BF EXAQC AG vs. MICROSOFT CORPORATION</i> , Civil Action No. 2:24-cv-433 (E.D. Tex. filed June 10, 2024)
EX2002	Second Amended Docket Control Order, Dkt. No. 48, <i>PARTEC AG and BF EXAQC AG vs. MICROSOFT CORPORATION</i> , Civil Action No. 2:24-cv-433 (E.D. Tex. filed April 9, 2025)
EX2003	United States District Courts – National Judicial Caseload Profile through December 31, 2024, https://www.uscourts.gov/sites/default/files/2025-02/fcms_na_distprofile1231.2024.pdf (accessed April 11, 2025)
EX2004	Defendant Microsoft’s October 30, 2024 Invalidity Contentions, <i>PARTEC AG and BF EXAQC AG vs. MICROSOFT CORPORATION</i> , Civil Action No. 2:24-cv-433 (E.D. Tex.)
EX2005	Defendant Microsoft’s March 6, 2025 Invalidity Contentions, <i>PARTEC AG and BF EXAQC AG vs. MICROSOFT CORPORATION</i> , Civil Action No. 2:24-cv-433 (E.D. Tex.)
EX2006	U.S. Patent Publication No. 2017/0262319
EX2007	Dynamic Process Management with Allocation-internal Co-Scheduling towards Interactive Supercomputing, Clauss et al., 1st COSH Workshop on Co-Scheduling of HPC Applications, 2016
EX2008	PDF comparison generated using PDF-XChange Editor of Petition Sections VII.C.1.c and VII.C.9.e against EX1003 Sections X.C.1.c and X.C.9.e
EX2009	Declaration of Dr. David Kaeli in Support of Patent Owner’s Preliminary Response
EX2010	<i>Curriculum Vitae</i> for Dr. David Kaeli
EX2011	List of Cases for Dr. David Kaeli
EX2012	The DEEP-ER project: I/O and resiliency extensions for the Cluster-Booster architecture, Anke Kreuzer et al., 2018
EX2013	NVIDIA GeForce RTX 2080 Ti Specs TechPowerUp GPU Database, https://www.techpowerup.com/gpu-specs/geforce-rtx-2080-ti.c3305 , accessed April 11, 2025

EX2014	Efficient Data-parallel Distributed DNN Training for Big Dataset under Heterogeneous GPU Cluster, Shinyoung Ahn et al., 2024 IEEE International Conference on Big Data
EX2015	Practice and Experience in using Parallel and Scalable Machine Learning with Heterogenous Modular Supercomputing Architectures, Morris Riedel et al., 2021
EX2016	Definition of “portion,” The American Heritage Dictionary of the English Language, Third Edition, Houghton Mifflin Company, 1992
EX2017	Definitions of Research and Development: An Annotated Compilation of Official Sources, https://nces.nsf.gov/pubs/nces25215#definitions , accessed on September 20, 2025
EX2018	ParTec partners with HDRZ research lab to develop 500 petaflops AI supercomputer, https://www.datacenterdynamics.com/en/news/partec-partners-with-hdrz-research-lab-to-develop-500-petaflops-ai-supercomputer/ , accessed on September 22, 2025
EX2019	ParTec AG and HZDR to Build AI Supercomputer Supporting Research in AI, Quantum Computing, and HPC, https://thequantuminsider.com/2024/10/22/partec-ag-and-hzdr-to-build-ai-supercomputer-supporting-research-in-ai-quantum-computing-and-hpc/ , accessed on September 23, 2025
EX2020	Deposition Testimony of Darrell Long, Ph.D., September 17, 2025
EX2021	Declaration of Dr. David Kaeli in Support of Patent Owner’s Response
EX2022	Exhibits to the September 17, 2025 Deposition of Darrell Long, Ph.D.

TABLE OF CONTENTS

I.	Introduction.....	1
II.	Background.....	3
	A. State of the Art	3
	B. The '442 Patent Invention.....	5
	C. Petitioner's References.....	10
	1. Lippert [EX1004].....	10
	2. Budenske [EX1005].....	13
	3. Kambatla [EX1006].....	17
	D. '442 Patent Prosecution History.....	18
III.	Legal Standards	19
IV.	Level of Ordinary Skill in the Art	21
V.	Claim Construction.....	21
VI.	Argument	21
	A. Ground 1 Fails to Teach or Render Obvious Any Challenged Claims.....	23
	1. Claim 1	23
	2. Claim 3	50
	3. Claim 5	54
	4. Claim 9.....	57
	5. The Remaining Claims	58
	B. Ground 2 Fails to Teach or Render Obvious Any Challenged Claims.....	58
	1. Claim 3	58
	2. Claim 5	59
	3. Claim 9.....	61
	4. A POSITA would not have modified the references in the claimed manner	64
	5. The Remaining Claims	64

VII. Conclusion	64
-----------------------	----

I. Introduction

Patent Owner ParTec AG (“ParTec” or “Patent Owner”) respectfully requests that the Board confirm patentability of Claims 1-10 (the “Challenged Claims”) of U.S. Patent No. 11,537,442 (“the ’442 Patent”), because Petitioner Microsoft Corporation (“Microsoft” or “Petitioner”) has not met its burden of demonstrating unpatentability of any of the Challenged Claims.

The Petition grounds are based on a published patent application (US2013/0282787), from Dr. Thomas Lippert, one of the ’442 Patent co-inventors, that was the primary reference analyzed by the Examiner in the original prosecution. As recognized during the prosecution of the ’442 patent, the Lippert ’787 published application failed to teach redistribution of sub-tasks to a mix of computation and booster nodes in subsequent computing iterations based on information learned during the processing of the sub-tasks in prior computing iterations.

Acknowledging that the ’787 published application did not teach this aspect of the ’442 patented invention, the Petition relied upon a secondary reference (Budenske) for its disclosure of “input-data dependent remapping of application sub-tasks ... using a previously stored off-line statically determined mapping.” EX1005 at 387 (emphasis omitted). But Budenske itself expressly states that it cannot (and does not) teach distributing sub-tasks among different types of processors in

subsequent computing iterations based on information learned during prior iterations.

Even if that glaring omission could be overlooked, the Petition's obviousness theories rely on an unworkable combination that a POSITA would not have been motivated to attempt. The Budenske article from 1998 candidly acknowledges its many shortcomings, including (1) its operating software (IOS) "has not been implemented" and to even attempt to do so would represent "a major undertaking," raising serious questions regarding enablement of that teaching; (2) the HC Kernel requires off-line subtask mappings generated *before* the computing iterations occur because of the longer execution times required in Budenske; and (3) the HC Kernel implements subtasks using only "processors ... of the same type." EX1005 at 389-390, 392. Petitioner's proposed combination to modify the Lippert '787 published application with the off-line assignment mappings of Budenske flies in the face of the flexible and dynamic approach disclosed and claimed in the '442 patent of using the information learned during prior computing iterations to re-distribute sub-tasks among different types of processing nodes.

Petitioner's other analysis ignores or misinterprets entire Claim limitations altogether (such as the daemons of Claim 5 and 9 and the dynamically altered further distribution of Claim 3), and also should be rejected.

II. Background

Patent Owner ParTec is a leading provider of modular supercomputers and software that provide massive computing power for purposes such as artificial intelligence solutions, neural networks and quantum computing. *See* EX2017; EX2018. Its services include the design and development of high-performance computers (HPC), as well as consulting and support services in all areas of development, construction, and operation of these advanced systems. Among other projects, ParTec is currently leading the construction of the first exascale supercomputer in Europe: the JUPITER Exascale Supercomputer. ParTec also contributed extensively to the construction of the MareNostrum 5 pre-exascale supercomputer in Barcelona—named in 2023 as the best high-performance computing development of the prior year. A critical aspect of the supercomputers being co-developed by ParTec is the features disclosed and claimed in its patents, including the '442 Patent.

A. State of the Art

A computing cluster generally includes a number of processing units and resources that can be organized to function as a single system. Computing clusters composed of multiple processors can be used to achieve improved data processing throughput and enhanced functionality relative to that available in a single computer.

Initially, these clusters were homogenous and typically required a single type of general-purpose processor. EX2021¹ ¶¶35-37.

Heterogeneous systems, in contrast, enable the integration of multiple different processor architectures within a single system or cluster, recognizing that different processor architectures provide different advantages and disadvantages. *Id.* ¶¶31-37. Traditional central processing units (“CPUs”) typically offer flexibility and high single-thread performance at the expense of, for example, high energy usage and limited parallelization capabilities. *Id.* ¶35. Other computing architectures have their own tradeoffs. For example, traditional graphical processing units (“GPUs”) are more specialized, and may offer better parallel processing capabilities than CPUs, but may have slower clock speeds and less flexible processing architectures. *Id.* Certain computing tasks are more efficiently and/or more quickly handled by CPUs, while others are more efficiently and/or more quickly handled by specialized processors like GPUs. More specialized processors (such as GPUs) that offload processing from a CPU are sometimes referred to as “boosters” or “accelerators” because they provide the capability to perform specialized or more complex

¹ Patent Owner withdraws the previously submitted declaration of Dr. Kaeli supporting Patent Owner’s Preliminary Response (EX2009) in favor of a more comprehensive declaration submitted as EX2021.

operations that would require significant CPU resources and/or more time to perform. *Id.* In such systems, the accelerator operates under the control of a CPU, which will provide the necessary data to the accelerator and manage its operation.

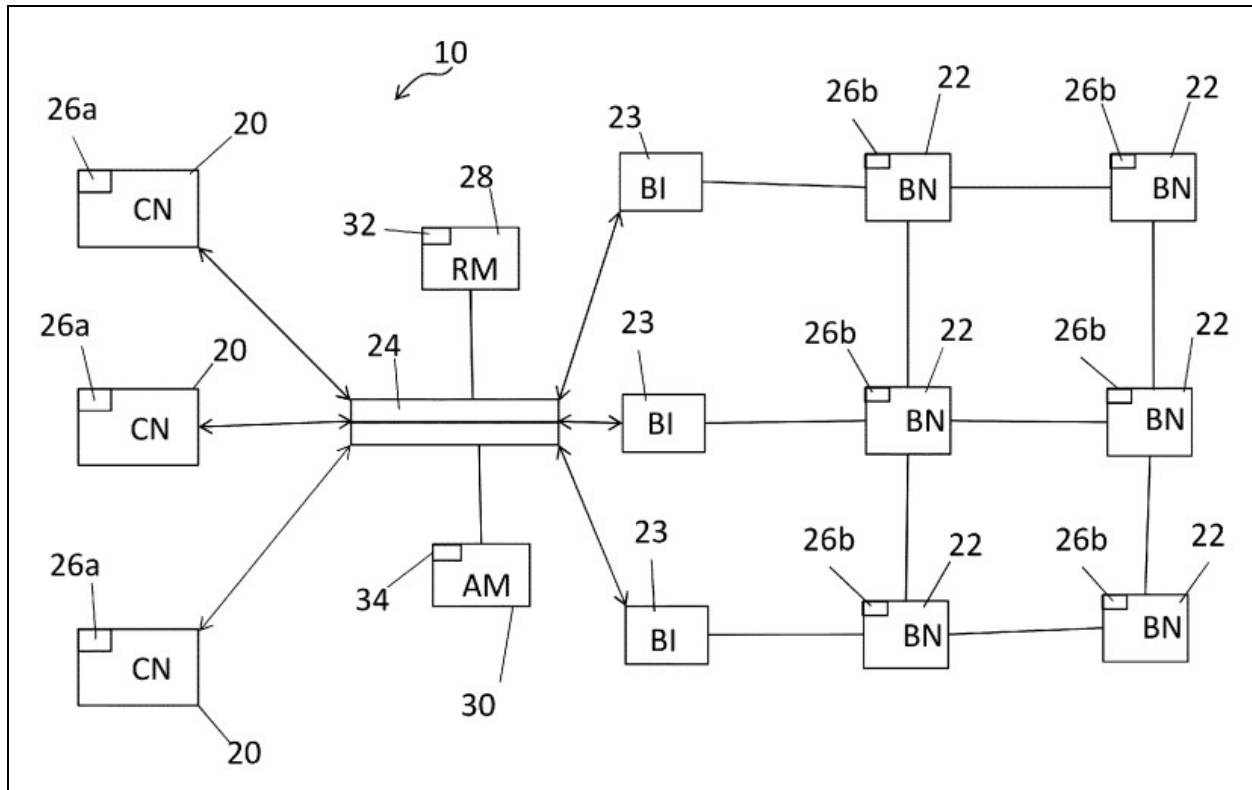
When dealing with these heterogeneous systems, historically the question was to which type of processor within the system a given task should be assigned because distribution of subtasks between different types of processors at the same time was not feasible. *See id.* ¶37. The '442 Patent inventors recognized that simply choosing one processor type over the other was insufficient to fully optimize operation of a cluster, and the far more useful (and difficult) question was how to efficiently utilize a combination of processor types (*e.g.*, a combination of computation nodes and booster nodes) dynamically over the course of several computing iterations to capitalize on the relative advantages of both. *See* EX1001 at 1:23-55, 4:64-5:24. The concept of dynamically redistributing sub-tasks to a mixture of CPU's and accelerators over several computing iterations based on information during prior iterations provided a much more powerful computing approach. It is this problem and solution that the '442 Patent addresses. EX2021 ¶¶33-35.

B. The '442 Patent Invention

The '442 Patent is directed to a heterogeneous computing system arrangement capable of providing more efficient computation of tasks due—in part—to how it distributes sub-tasks across computing iterations based on information learned

during prior computing iterations. Heterogeneous computing systems incorporate different types of processors, including multiple general purpose processors along with, more specialized “accelerator-type” processors. *See* EX1001 at 3:24-38.

An example embodiment is shown in Figure 1:



EX1001 Fig. 1.

As shown in Figure 1, the example system described in the '442 Patent includes multiple computation nodes 20 (“CN”) in communication with booster nodes 22 (“BN”) using communication infrastructure 24. *Id.* at 2:66-3:11. “The system 10 also includes a resource manager 28 shown connected to the communication infrastructure 24 and an application manager 30.” *Id.* at 3:19-21.

The '442 Patent teaches how such heterogeneous computing systems can be adapted for parallel processing of large and extremely complex computation jobs. *See* EX1001 at 1:15-20. “A job to be computed by the system may comprise a number of tasks some of which or all may be repeated a number of times during the execution of the job.” *Id.* at 3:43-46. “The tasks may comprise a number of sub-tasks ...[that] may be more or less suitable for processing by one or more of the computation nodes or one or more of the boosters. In particular, the scalability of the sub-task may indicate whether it is more appropriate for the sub-task to be processed by a computation node or a booster.” *Id.* at 3:50-55. The '442 Patent gives the example of “a ‘Monte-Carlo’ based simulation where an effect is modelled using a random number, the calculations being repeated many times in succession.” *Id.* at 3:44-49; EX2021 ¶36.

The '442 Patent teaches that efficiency can be improved by using a specific configuration capable of dynamically updating the distribution of sub-tasks between the computation nodes and booster nodes over the course of multiple computing iterations. EX1001 at 3:59-4:3. For example, even though a sub-task might appear suitable for a particular booster node based on initial analysis, the sub-task might be better suited for a different distribution of computing resources on subsequent iterations. The '442 Patent teaches techniques to perform the monitoring and

distribution based on “information learned about the processing of the sub-task and any need to call further sub-tasks during the processing.” *See id.* at 4:48-5:9.

Independent Claims 1 and 9 are reproduced below. The last limitation of both Claim 1 and Claim 9 are similar and address providing an updated distribution of the sub-tasks among computation nodes and booster nodes for processing a further computing iteration based on the claimed information:²

[1.1] A method of operating a heterogeneous computing system comprising a plurality of computation nodes and a plurality of booster nodes, at least one of the plurality of computation nodes and a plurality of booster nodes being arranged to compute a computation task, the computation task comprising a plurality of sub-tasks, the method comprising:

[1.2] in a first computing iteration, assigning and processing the plurality of sub-tasks by at least a portion of the plurality of computation nodes and at least a portion of the plurality of booster nodes in a first distribution; and

² Patent Owner will refer to the same “bracketed headings” utilized in the Board’s Institution Decision. *See* Institution Decision at 4-5.

[1.3] generating, using information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes, a further distribution of the plurality of sub-tasks between the plurality of computation nodes and the plurality of booster nodes for processing thereby in a further computing iteration.

[9.1] A heterogeneous computing system comprising:

[9.2] a plurality of computation nodes and a plurality of booster nodes for computing one or more tasks comprising multiple sub-tasks;

[9.3] a communication interface connecting the plurality of computation nodes with each other and the plurality of booster nodes;

[9.4] a resource manager for assigning at least a portion of the plurality of booster nodes and at least a portion of the plurality of computation nodes to each other for the computing of the one or more tasks in a first computing iteration; and

[9.5] an application manager configured to receive information from daemons operating in at least the portion of the plurality of

computation nodes and at least the portion of the plurality of booster nodes to update a distribution of the multiple sub-tasks between the plurality of computation nodes and the plurality of booster nodes in a further computing iteration.

C. Petitioner's References

Petitioner asserts two grounds based on three references as shown in the table below. Pet. at 2. These different grounds are addressed separately in this Response.

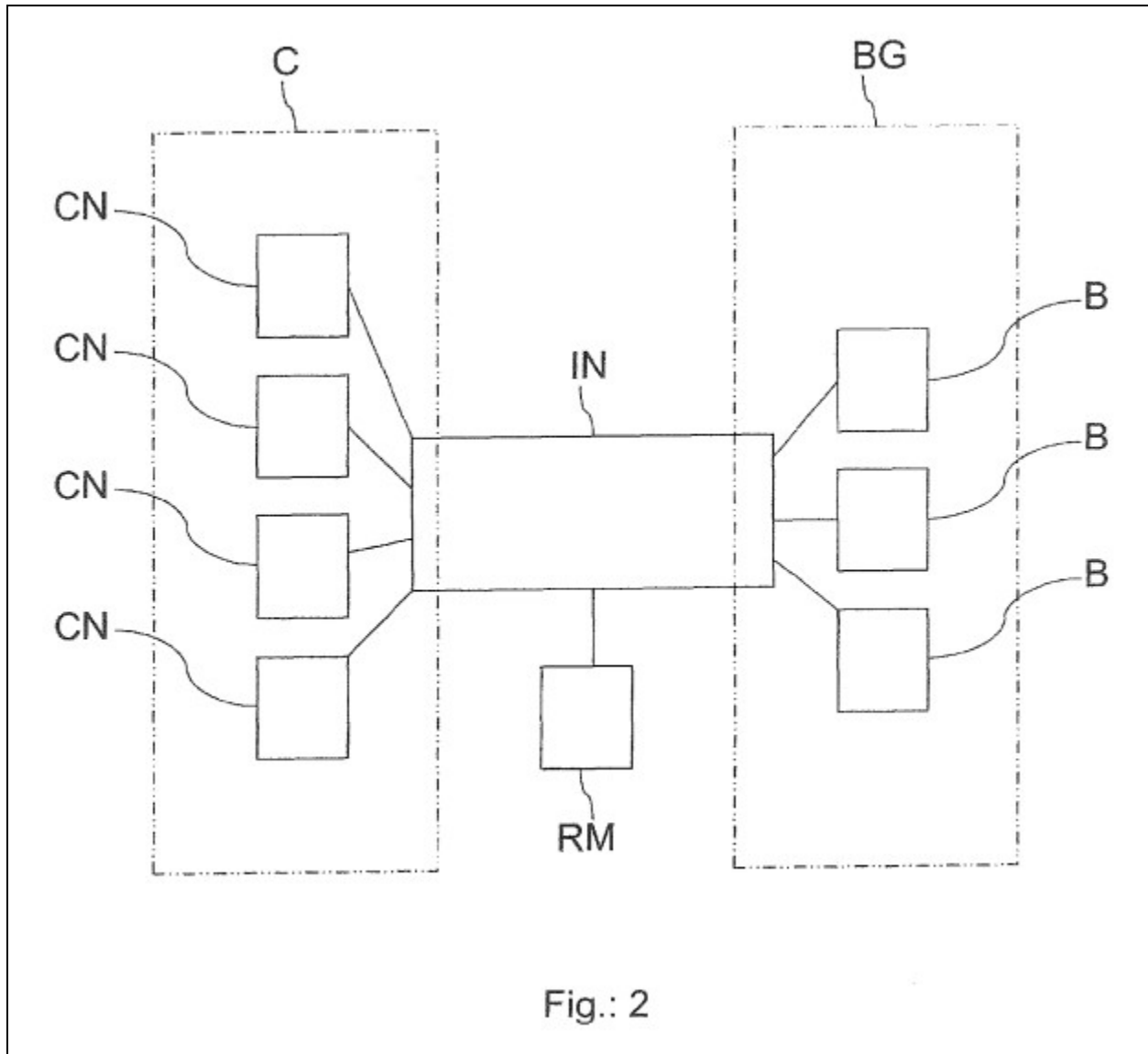
Ground	References	Statute	Claims
Ground 1	Lippert (EX1004) and Budenske (EX1005)	35 U.S.C. § 103	1-10
Ground 2	Lippert, Budenske, and Kambatla (EX1006)	35 U.S.C. § 103	2-5, 8-10

1. Lippert [EX1004]

“Lippert” is a U.S. Patent Application (Pub. No. 2013/0282787) to Thomas Lippert titled “Computer Cluster Arrangement For Processing A Computation Task And Method For Operation Thereof.” EX1004 at 1. Dr. Thomas Lippert is one of the inventors of the '442 Patent challenged in this proceeding, and the patent that issued from the 2013/0282787 patent application (10,142,156) is also assigned to ParTec. The Lippert patent application used as the primary reference in the Petition was addressed extensively during prosecution of the '442 Patent, and its disclosure

is incorporated by reference into the '442 Patent as “WO 2012/049247.” *See, e.g.*, EX1002 at 156-166, 173-175; EX1001 at 1:23-24; Section II.D.

Lippert discloses a computer cluster arrangement including computation nodes (“CN”) coupled to boosters (“B”) using a communication infrastructure (“IN”). EX1004 at [0067]. A Resource Manager (“RM”) can be used in Lippert to assign boosters to computation nodes. An embodiment of this arrangement can be seen in Lippert’s Figure 2:



It is not surprising that Lippert and the '442 Patent share some similarities given the '442 Patent notes that it is a further development of Lippert. EX1001 at 1:23-28. Nor is it surprising that the '442 Patent inventors continued to work to significantly improve on Dr. Thomas Lippert's earlier work, leading to the claimed advances in the '442 Patent. *See* EX1001 at 1:35-38. Petitioner's use of Lippert as the primary reference in this proceeding (despite being the central focus during the

original prosecution) demonstrates just how groundbreaking the Lippert disclosure was and how deficient the other available prior art was in comparison to ParTec's own work.³ However, as the Patent Office concluded, Lippert's earlier patent does not teach or render obvious the improvements of the '442 Patent, as addressed in Section II.D.

2. Budenske [EX1005]

"Budenske" is an article listing authors John R. Budenske, Ranga S. Ramanujan and Howard Jay Siegel titled "A Method for the On-Line Use of Off-Line Derived Remappings of Iterative Automatic Target Recognition Tasks onto a Particular Class of Heterogeneous Parallel Platforms." EX1005 at iv. Budenske identifies itself as being included in the October 1998 Edition of The Journal of Supercomputing. *Id.* at i-iii.

"Th[e] paper concentrates on the operation of the [heterogeneous computing ("HC")] Kernel" which it claims "differs from other real-time HC mapping techniques in that it allows on-line real-time use of off-line precomputed mappings." *Id.* at 388-389. The paper states that it "focuses on (1) the application and hardware platform characteristics that are needed to enable the use of the HC Kernel, (2) the

³ Notably, Microsoft has not sought *inter partes* review of U.S. Patent No. 10,142,156 that issued from the Lippert application.

techniques that comprise the HC Kernel, and (3) how the information needed for the HC Kernel to operate is collected.” *Id.* The authors emphasize that the IOS itself has never been implemented, doing so would be a “major undertaking,” and the paper relates only to “design concepts for the HC Kernel.” *Id.* at 389.

Petitioner has relied on Budenske to address distributing sub-tasks between computation nodes and booster nodes in a further computing iteration based on information related to the processing of the plurality of sub-tasks. *See, e.g.,* Pet. at 29-32. While Budenske may be directed to a “heterogeneous” system to the extent the system can include different types of processors, Budenske expressly notes that its system was not designed to distribute tasks to a mixture of processor types. Instead, tasks requiring multiple processors are assigned to processors of the same type. EX1005 at 392 (“For simplicity, it is assumed that if an implementation of a given subtask uses multiple processors, *all processors will be of the same type.*”) (emphasis added). Budenske, unlike the ’442 Patent, neither teaches nor enables distribution of subtasks to different types of processors. It is expressly limited to assigning subtasks to the same processor types. *See* EX2021 ¶¶50-53.

Further, in contrast to “real-time HC mapping techniques,” Budenske is limited to use of off-line precomputed mappings, which require “much longer execution times to search for a good solution than what is practical for an on-line heuristic.” EX1005 at 389-393. Thus, Budenske suggests implementing a static, pre-

generated assignment of tasks, rather than a dynamic assignment based on the collection of information generated in real-time.

More specifically, Budenske describes theories for a type of automatic target recognition (ATR) system and explains that “an ATR system takes a set of images from a group of sensors and produces some type of description of the scene.” *Id.* at 387, 390. Budenske’s process for remapping subtasks relies on “off-line precomputed mappings.” *Id.* at 398. These mappings are generated prior to any task processing based on finite sets of “dynamic parameter” values (for example, the number of objects in a prior image frame). *Id.* at 393, 398. Budenske describes using an “MDDG Generator to use genetic algorithms (or other off-line heuristics) to determine a matching and scheduling (i.e., MDDG) off-line for each scenario associated with a production ATR application task.” *Id.* at 398. These offline generated mappings “allow the HC Kernel Monitor to select a new MDDG during execution if desired based on the actual values of the dynamic parameters at the end of an iteration through the corresponding DDG.” *Id.* at 398; *see also id.* at 394 (“Scenario Generator constructs S scenarios, where each scenario corresponds to a particular set of representative values for the D dynamic parameters (one representative value for each dynamic parameter.”).

During online processing, “the HC Kernel Monitor receives updated actual values of the dynamic parameters at the end of each iteration through the

corresponding DDG.” *Id.* at 402. “For each dynamic parameter, the representative value included in the allowable choices for that parameter (as specified by the Scenario Generator) is found that is closest in absolute difference to the current actual value for that parameter.” *Id.* “The resulting vector of D representative values is used as the approximation of the set of current actual dynamic parameters in terms of a precomputed scenario.” *Id.* Budenske relies on these off-line generated mappings based on the theory that using purely off-line generated mappings will “produce better mappings than on-line heuristics.” *Id.* at 389. However, Budenske’s reliance on “off-line statically determined mapping” greatly limits its flexibility in distributing sub-tasks at runtime. *Id.* at 387 (emphasis changed).

Petitioner’s suggestion that Budenske somehow suggests and would motivate a skilled artisan to dynamically assign sub-tasks among different types of processors on subsequent computing iterations based on information learned during a prior computing iteration is nothing more than attorney argument and would require a wholesale redesign of the Budenske disclosure. As explained in more detail below, distributing sub-tasks between processors of different types is difficult, and a POSITA at the applicable time would have understood Budenske to (at best) be determining which processor type to use for the full task based on precomputed mappings—not dynamically distributing sub-tasks between disparate processor types based on information learned during a prior processing iteration.

3. Kambatla [EX1006]

“Kambatla” is a U.S. Patent Application (Pub. No. 2018/0074855) to Karthik Kambatla titled “Utilization-Aware Resource Scheduling in a Distributed Computing Cluster.” EX1006 at 1. Kambatla relates to computing clusters, but does not teach anything directed to heterogeneous computing using different types of processors. *See, e.g., id.* at [0002]. Kambatla never discusses the use of different processor types at the same time, distribution of tasks to different processor types, or anything else that might make it relevant to an actual heterogeneous computing architecture. Thus, Kambatla cannot remedy any of the deficiencies in Budenske related to distribution of sub-tasks between heterogeneous computation nodes and booster nodes.

Petitioner asserts that “Kambatla explains that its teachings can be successfully implemented in heterogeneous computing systems comprising ‘CPU and memory resources’ (i.e., computation nodes) and ‘GPU[s] (graphical processing [units])’ (i.e., boosters).” Pet. at 61 (citing EX1006 [0020], [0023], [0059]). However, Kambatla does not disclose any embodiments where both computation nodes and booster nodes are used together to compute a computation task. Kambatla paragraph 20 merely states that various different types of resources can be allocated under different frameworks, but not that different processor types will be used within the same framework to process the same computation task. Kambatla paragraph 23

merely notes that the overall environment can include various resources, but not that tasks will be distributed to any specific resources. Kambatla paragraph 59 similarly does not disclose that different processor types will be used to process the same computation task.

During his deposition, Petitioner's expert also indicated that paragraphs 89 and 94 of Kambatla related to heterogeneous computing systems. EX2020 at 78:14-79:2. But yet again, these paragraphs—at best—indicated that the system could potentially be implemented with one type of processor or another, and not that different types of processors would be utilized for the same computation task. EX2021 ¶¶54-56. Furthermore, Dr. Long conceded during his deposition that even though Kambatla gives examples of potential processors that the system could use, it “does not require that the processors be different.” EX2020 at 82:10-15.

D. '442 Patent Prosecution History

As Petitioner concedes, the USPTO has already extensively evaluated its primary reference (Lippert) during prosecution of the '442 Patent. *See* Pet. at 6-7. The Examiner initially rejected all pending '442 Patent Claims over Lippert in an April 22, 2022 Office Action. EX1002 at 154-167.

In response to rejection of the original claims, ParTec amended the claims and traversed the rejection, noting Lippert failed to teach redistribution of sub-tasks based on information related to the processing of the plurality of sub-tasks across

multiple computing iterations. EX1002 at 171-175. ParTec argued that, “Lippert fail[ed] to disclose the claimed combination ‘in a first computing iteration, assigning and processing the plurality of sub-tasks by at least a portion of the plurality of computation nodes and at least a portion of the plurality of booster nodes in a first distribution,’ and ‘generating, using information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes, a further distribution of the plurality of sub-tasks between the plurality of computation nodes and the plurality of booster nodes for processing thereby in a further computing iteration,’ as recited in amended claim 1 and as similarly recited in amended claim 9.” *Id.* at 175.

The Examiner agreed, allowing the amended claims over the Lippert ’787 published application. *Id.* at 184.

III. Legal Standards

An IPR challenge is limited to the invalidity grounds *presented in the petition*. *See In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1381 (Fed. Cir. 2016) (“[T]he Board must base its decision on arguments that were advanced by a party, and to which the opposing party was given a chance to respond.”).

“In an IPR, the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (petitions must identify “with

particularity . . . the evidence that supports the grounds for the challenge to each claim”); 35 U.S.C. § 312(a)(3). This burden of persuasion never shifts to the patent owner. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015).

“To satisfy its burden of proving obviousness, a petitioner cannot employ mere conclusory statements. The petitioner must instead articulate specific reasoning, based on evidence of record, to support the legal conclusion of obviousness.” *In re Magnum Oil Tools Int’l*, 829 F.3d at 1380. The obviousness inquiry requires considering whether one of skill in the art “would have been motivated to combine the prior art to achieve the claimed invention.” *In re NuVasive, Inc.*, 842 F.3d 1376, 1381 (Fed. Cir. 2016) (quoting *In re Warsaw Orthopedic, Inc.*, 832 F.3d 1327, 1333 (Fed. Cir. 2016)). “[T]he factual inquiry whether to combine references must be thorough and searching...” *Id.*

“[I]f an obviousness case is based on a non-self-enabled reference, and no other prior art reference or evidence would have enabled a skilled artisan to make the claimed invention, then the invention cannot be said to have been obvious.” *Raytheon Techs. Corp. v. GE*, 993 F.3d 1374, 1377 (Fed. Cir. 2021). “[E]ven though a non-enabling reference can play a role in an obviousness analysis, the evidence of record must still establish that a skilled artisan could have made the claimed invention.” *Id.* at 1381.

IV. Level of Ordinary Skill in the Art

A person of ordinary skill in the art at the time of invention (“POSITA”) would have had (1) a bachelor’s degree in computer science or computer engineering and approximately two years of experience in the field of heterogeneous computing systems, or (2) a master’s degree in computer science or computer engineering with a focus on heterogeneous computing systems. EX2021 ¶¶42-44.

V. Claim Construction

Claims are construed according to *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). *See* 37 C.F.R. § 42.100(b). “Any prior claim construction determination concerning a term of the claim in a civil action...will be considered.” *Id.*

VI. Argument

The Petition should be denied for multiple independent reasons. As shown below, all Petition Grounds fail to teach or render obvious any Challenged Claims.

First, the Petition Grounds fail to teach or render obvious a further distribution of sub-tasks between computation nodes and booster nodes during a further computing iteration using information obtained during the processing of sub-tasks, at least because Budenske expressly disclaims such a disclosure and would further fail to enable such a disclosure (Claims 1 and 9).

Second, the Petition Grounds fail to teach or render obvious generating, using information relating to the processing of the plurality of sub-tasks, a further distribution in a further computing iteration, at least because all purported further distributions in the asserted grounds are generated prior to the beginning of processing of the sub-tasks (Claim 1).

Third, the Petition Grounds fail to teach or render obvious the claimed information relating to the processing of the plurality of sub-tasks, at least because the information relied upon in the asserted grounds relates, at best, to the results of the execution of the sub-tasks (Claim 1).

Fourth, the Petition Grounds fail to teach or render obvious the resource manager dynamically altering further distribution during the computing of the computation task, at least because Petitioner has misinterpreted the claim language and failed to address the concept of a further distribution of the plurality of sub-tasks (Claim 3).

Fifth, the Petition Grounds fail to address the specific claimed computation node daemons and booster node daemons, and at best improperly relied on generic disclosure of daemons (Claims 5 and 9).

Finally, a POSITA would not have been motivated to make the proposed combination of Lippert, Budenske, and Kambatla, as discussed below (All Claims).

A. Ground 1 Fails to Teach or Render Obvious Any Challenged Claims

1. Claim 1

(a) Distribution of Sub-Tasks to Both Computation Nodes and Booster Nodes in Both a First and Further Computing Iteration

Claim 1 requires generating a “further distribution” of the plurality of sub-tasks between the plurality of computation nodes and booster nodes in a “further computing iteration,” using “information relating to the processing of the plurality of sub-tasks” by at least the portion of the plurality of computation nodes and booster nodes from a “first distribution.” *See* ’442 Patent Claim 1. Neither Lippert nor Budenske disclose this requirement. Furthermore, Budenske is not enabled on its face, much less for the teachings for which Petitioner has relied on it, and a POSITA would not have made the proposed combination between Lippert and Budenske.

(i) The Claims Require Distribution of Sub-Tasks to Both Booster Nodes and Computation Nodes In Both a First and Further Computing Iteration

First, as indicated above, the Challenged Claims require that, in **both a first and further** computing iteration, sub-tasks are distributed to **both** computation nodes and booster nodes. This can be seen in the plain language of, for example, ’442 Patent Claim 1 at 5:33-36:

in a first computing iteration, assigning and processing the plurality of sub-tasks by **at least a portion** of the plurality of computation

nodes and **at least a portion** of the plurality of booster nodes in a first distribution;

As shown in the claim language, in the first computing iteration there is a first distribution of sub-tasks. In that distribution, there must be a plurality (*i.e.*, more than one) of sub-tasks. EX2021 ¶¶58-61; EX2020 at 27:11-21. Those sub-tasks must be assigned to (1) a portion (*i.e.*, at least one) of the plurality of computation nodes, and (2) a portion (*i.e.*, at least one) of the plurality of booster nodes.⁴ Assigning all sub-tasks within a single computing iteration to **just** computation nodes or **just** booster nodes is not sufficient to render obvious the claim language. Petitioner's expert agreed with that assessment, noting that assignment of sub-tasks to just booster nodes would not satisfy this limitation. EX2020 at 46:18-47:5 (agreeing that "if sub-tasks are assigned only to booster nodes" the limitation would not be met

⁴ The plain meaning of the term "portion" requires at least part of the plurality. *See, e.g.*, EX2016 at 1412 (defining portion as "[a] section or quantity within a larger thing; a part of a whole."); EX2020 at 47:7-23 (Petitioner's expert agreeing that "portion" meant "at least" one or more). For example, if you offer someone a portion of a cake and then do not give them anything, you have not given them a portion of the cake.

because “the wording requires me to use at least one computation node, even if I didn’t need it”).

The same is true of subsequent computing iterations as described in, for example, ’442 Patent Claim 1:

generating, using information relating to the processing of the plurality of sub-tasks by at least **the portion** of the plurality of computation nodes and at least **the portion** of the plurality of booster nodes, a further distribution of the plurality of sub-tasks **between** the plurality of computation nodes and the plurality of booster nodes for processing thereby in a further computing iteration.

’442 Patent Claim 1 at 5:37-43 (emphasis added).

This again shows that the sub-tasks are distributed between both the computation nodes **and** booster nodes, and not simply distributed to one or the other.

This interpretation is further confirmed by the specification, which repeatedly discusses seeking an ideal sub-task distribution between both computation nodes and booster nodes, rather than simply selecting one type or the other for processing of all sub-tasks. *See, e.g.*, EX1001 at 2:5-13, 2:15-18 (“The information provided allows an application manager to determine if the distribution of sub-tasks between computation nodes and booster nodes can be **adapted or improved upon for a further computing iteration.**” (emphasis added)), 2:19-25 (“A resource manager

may determine the assignment of tasks and sub-tasks to the computation nodes **and** booster nodes for the first iteration...” (emphasis added)), 2:45-50 (“[T]he application manager being arranged to receive information from daemons operating in the computation nodes and booster nodes to update a distribution of the sub-tasks between the computation nodes **and** the booster nodes between a first computing iteration and a further computing iteration.” (emphasis added)), 4:9-13 (teaching “us[ing] the information provided by the daemons...to determine whether the distribution of sub-tasks to computation nodes **and** boosters could be adjusted to optimize or adapt the computation of the task for a subsequent iteration (emphasis added)), 4:59-63.

Correspondingly, the ’442 Patent describes “a first **division** of sub-tasks **between** a computation node and a booster” in a computing iteration. *Id.* at 3:59-60 (emphasis added). It teaches that “it is possible that such a division is not an optimal division for the computation of the task,” because “**lcertain** sub-tasks which were assigned to a booster on the first iteration might in fact not be suitable for processing by the booster such that a processing of the sub-tasks by a computation node rather than a booster might optimize the computation of the task as a whole.” *Id.* at 3:60-66 (emphasis added). Thus, the ’442 Patent specifies that it may be best to redistribute only certain sub-tasks to a different node type while other sub-tasks remain with that node type. “Accordingly, for a second and possible further

iterations, if needed, of the task [using] an altered second and/or further sub-task distribution might improve the efficiency of the computation of the task.” *Id.* at 3:66-4:3.

(ii) Ground 1 Fails To Teach Or Render Obvious Distribution of Sub-Tasks To Both Computation Nodes and Booster Nodes

As discussed above, all Claims require assigning sub-tasks to both computation nodes **and** booster nodes during a further computing iteration. Petitioner does not contend that Lippert teaches limitations [1.3] or [9.5]. *See* Pet. at 29-34, 52 (citing only Budenske). Nor could Petitioner—as noted during prosecution (and agreed to by the examiner), Lippert does not disclose generating further distributions of sub-tasks between computation nodes and booster nodes in further iterations using information relating to the processing of the plurality of sub-tasks. EX1002 at 173-175, 184.

Instead, Petitioner is wholly reliant on Budenske for these limitations. However, Budenske does not teach distributing sub-tasks between different types of processors, much less performing a redistribution of sub-tasks among different processor types in a further computing iteration based on information learned during prior iterations. In fact, Budenske expressly states that it is not implementing such an arrangement, noting “it is assumed that if an implementation of a given subtask uses multiple processors, **all processors will be of the same type.**” EX1005 at 392 (emphasis added); EX2021 ¶¶62-66. Further reinforcing this restriction on

Budenske's design, Petitioner's Expert testified that he could not think of any situation in the examples given by Budenske that would result in moving a sub-task from one processor type to another processor type. EX2020 at 136:23-137:4.

Subsequent publications discussing Budenske confirm that same limitation. Petitioner asserts that EX1007 ("A semi-static approach to mapping dynamic iterative tasks onto heterogeneous computing systems") allegedly discloses a "successful implementation of Budenske's teachings." Pet. at 22. This reference reiterates that "for simplicity" data-parallel implementations (*i.e.*, assigning subtasks to different processors to be processed in parallel, such as would occur for multiple subtasks processed during the same computing iteration) were limited to using processors of the same type. EX1007 at 81-82; *see also id.* ("[T]he expected execution time of a particular multiprocessor implementation of a sub-task is independent of which fixed-size subset of the processors of a given type are assigned to execute the subtask."). Thus, the Kwok reference, EX1007, confirms that Budenske necessarily excludes the possibility of assigning subtasks to both computation nodes and booster nodes in a single computing iteration. EX2021 ¶¶67-68.

Petitioner also points to EX1008 as another allegedly "successful implementation[] of Budenske's teachings," Pet. at 22. But yet again, EX1008 emphasizes that if multiple processors are used for subtask processing, they must be

“processors of the same type.” EX1008 at 156. The message to a skilled artisan is clear and unmistakable from these collective teachings: Budenske’s offline generated arrangement should be restricted to use with processors of a single type. EX2021 ¶¶67-68. Thus, Budenske (and the subsequent references addressing Budenske) make it clear that it was expressly not teaching the limitations for which Petitioner relies on it. As a result, the Petition grounds cannot teach or render obvious any Challenged Claims.

The Institution Decision suggests that “the assumption of a simplified circumstance for purposes of providing an example or illustration does not disclaim or nullif[y] **disclosures of Budenske on using different types of processors to perform a subtask.**” Institution Decision at 22 (emphasis added). However, Budenske does not have any such disclosures. Budenske does not disclose **any** embodiments wherein subtasks are distributed to both computation nodes and booster nodes in the same computing iteration (let alone a further computing iteration). There are no disclosures to “disclaim or nullify.” Instead, the Petition is wholly reliant on a disclosure that not only does not teach the concept for which Petitioner relies on it, but which expressly states it is not teaching the concept for which Petitioner relies on it.

(iii) **Budenske Does Not Enable Distributing Sub-tasks to Both Computation Nodes and Booster Nodes In a Computing Iteration**

In addition to not disclosing or teaching a system in which tasks are dynamically assigned to different types of nodes, Budenske also fails to include any disclosure that would enable such a limitation. To the contrary, by its own express statements Budenske acknowledges that it did not disclose or teach how to implement the claimed mixed node arrangement for reasons of “simplicity.” EX1005 at 392. Budenske acknowledges that the authors never tried to implement its theories, because to do so would have been a “major undertaking.” *Id.* at 389. Instead, the authors of Budenske acknowledged that they were merely providing “design concepts” for one aspect of the disclosures: the “HC Kernel.” *Id.* “[O]ther components [were] discussed only to the extent needed for describing the HC Kernel.” *Id.* at 388; EX2020 at 161:8-23. A POSITA would have understood from this that Budenske is not enabling on its face. EX2021 ¶¶69-81.

“[I]f an obviousness case is based on a non-self-enabled reference, and no other prior art reference or evidence would have enabled a skilled artisan to make the claimed invention, then the invention cannot be said to have been obvious.” *Raytheon Techs. Corp. v. GE*, 993 F.3d 1374, 1377 (Fed. Cir. 2021). Here, Ground

1 relies exclusively on Budenske for this limitation,⁵ and Petitioner has provided no evidence that the reference (which expressly states it did not consider whether distributing sub-tasks to different types of processors was even possible for “simplicity”) is an enabling disclosure for the limitations for which it is relied on.

The Federal Circuit found a similar proposed invalidity argument non-enabling in the *Raytheon* case. There, Raytheon owned a patent “directed to gas turbine engines” with a specific “power density” range. *Raytheon*, 993 F.3d at 1377. The petitioner had sought *inter partes* review based in part on a “1987 NASA technical memorandum” referred to as “Knip.” *Id.* at 1378. The “advanced [turbofan] engine” design discussed by Knip would have required composite materials that were “unattainable at that time.” *Id.* Despite the fact that such materials were not disclosed, for simplicity the authors of Knip contemplated “an imagined application of these ‘revolutionary’ composite materials to a turbofan engine.” *Id.*

The PTAB in the *Raytheon* case had initially found Knip to be an enabling disclosure because “the claims at issue do not require the advanced materials recited

⁵ Petitioner’s citations to EX1007 and EX1008 (ostensibly discussing Budenske) cannot resolve these shortcomings. As discussed in the previous section and further below, EX1007 and EX1008 also both expressly state that they did not consider situations where sub-tasks were assigned to more than one type of processor.

by Knip” and therefore “whether Knip’s advanced engine had been or could be implemented is not the proper consideration.” *Id.* at 1379-82 (internal quotes and citations omitted). The Federal Circuit disagreed, “conclud[ing] that GE failed to provide an evidence-based case for how the turbofan engine claimed in the ’751 patent having a particular power density is enabled by Knip’s disclosure. Thus, the Board’s finding that Knip is ‘enabling’ is legal error.” *Id.* at 1382.

Here, the lack of enablement is even more egregious. In *Raytheon* it was an unclaimed element—the “unattainable” composite material assumed to be theoretically possible for purposes of simplicity—that was not enabled by the disclosure. Here it is a specifically claimed element—redistribution of sub-tasks to both computation nodes and booster nodes in the further computing iteration—that Budenske did not even attempt to enable for purposes of “simplicity.”

While it may be true that “[e]ven if a reference discloses an inoperative device, it is prior art for all that it teaches,” here Budenske expressly admitted that it was not even considering the distribution of tasks to multiple different processor types in a single computing iteration. *Id.* at 1380 (quoting *Beckman Instruments Inc. v. LKB Produkter AB*, 892 F.2d 1547, 1551 (Fed. Cir. 1989)); EX1005 at 392. “[E]ven though a non-enabling reference can play a role in an obviousness analysis, the evidence of record must still establish that a skilled artisan could have made the claimed invention.” *Raytheon*, 892 F.2d at 1381. The evidence of record fails to meet

that mark. Thus, the Petition Grounds cannot demonstrate that the Budenske reference is enabling for the very teaching upon which the Petition relies.

By restricting execution to a single processor type, Budenske avoids addressing how different nodes would be selected in further iterations, what metrics and information would be needed to redistribute tasks, as well as what interconnect configurations would be needed for interfacing computation nodes and booster nodes. Instead, by selecting a single processor type “for simplicity,” Budenske treats “the expected execution time of a particular multiprocessor implementation of a subtask [as] independent of which fixed-size subset of the processors **of a given type** are assigned to execute the subtask.” *Id.* (emphasis added); EX2021 ¶¶37-41, 73. Petitioner’s expert noted the same, stated that by restricting sub-tasks to a single type of processor, sub-task assignment would be simplified because sub-tasks would “all complete at about the same time.” EX2020 at 149:12-25.

Dr. Kaeli’s analysis further confirms Budenske’s lack of enablement. As Dr. Kaeli explains, in addition to the express statements from Budenske that it is limited to processors of a single type, a POSITA would also have understood the same restriction applied relative to Budenske from the discussion of its architecture. EX2021 ¶¶52-53. For example, Petitioner refers to the SHARC DSPs of Budenske as the accelerators/boosters. Pet. at 21. Budenske explains that, under its architecture, those particular DSPs will “physically share a DRAM.” EX1005 at 392.

Dr. Kaeli notes that in the context of Budenske, a POSITA would understand that those DSPs “all share the same working memory, and if five processors were needed instead of four for the next iteration (such as when the number of objects of interest in the last frame changed from four to five—*see* example on page 393), all DSP processors would have access to the same data set and could take over the extra processing load. EX2021 ¶52; *see also* EX2020 at 107:12-108:8 (Dr. Long agreeing that the shared DRAM would be where data being processed by the DSP processors, making it “so communication between SHARC processors is very fast.”), 109:6-12 (Dr. Long stating that the ability to transfer data would be important when mapping tasks), 154:3-15 (noting that sharing a DRAM would make data sharing “quicker” and “make memory coherence issues simpler.”).

“The same would not be true of different processor types (such as RISC processors) which would not have direct access to that shared memory pool.” EX2021 ¶52. Stated more simply, the Budenske architecture implements a shared memory that requires the processor types to be the same in order to allow these nodes to work cooperatively. The net effect is that a skilled artisan would conclude from Budenske that mixing processor types for particular sub-tasks would be difficult to implement and would require undue experimentation to develop further innovations.

Confirming its focus on a single, uniform processor set, Budenske emphasizes that processors of the same type will have symmetric and conflict-free

intercommunication. EX1005 at 392. Budenske also emphasizes that expected execution times can be assumed for any assigned processor because of the predicate requirement that assignment is restricted to a single type of processor. *Id.* (“[T]he expected execution time of a particular multiprocessor implementation of a subtask is independent of which fixed-size subset of the processors **of a given type** are assigned to execute the subtask.”). As Dr. Kaeli explains, all of this would indicate to a POSITA that Budenske was incapable of assigning a subtask to more than one type of processor. EX2021 ¶¶51-53.

While Budenske suggests that its disclosure addresses “how to assign resources (e.g., processors) to the subtasks” generally, the fact that the authors felt the need to expressly limit their discussion to using only one type of processor for any given sub-task speaks volumes of the difficulty of distributing sub-tasks between disparate types of processors. *See* EX1005 at 391. This difficulty would have led a POSITA to conclude that distributing subtasks between different types of processors in both a first and further computing iteration was not taught by Budenske, and it certainly was not enabled. EX2021 ¶¶50-53, 62-81.

Furthermore, there is no evidence that the system described by Budenske (which did not even assign sub-tasks to processors of different types at the same time), could be implemented without undue experimentation based on the Budenske disclosure. For example, Petitioner asserts that EX1007 and EX1008 represent

“successful implementations of Budenske’s teachings.” Pet. at 22. On the contrary, those exhibits show that further experimentation into the “design concepts” disclosed by Budenske continued for years after Budenske’s publication, and neither one identifies any successful implementation of the high-level concepts purportedly taught by Budenske of using processing nodes of different types (much less the specific distribution of tasks relevant to the current analysis). *See* EX2020 at 156:11-25 (Petitioner’s expert testifying “I know that they worked on this project for a number of years, so clearly it was not trivial.”); EX2021 ¶¶74-80. These references highlight the extensive work necessary to implement even the simplified version of Budenske, which only utilized a single processor type. It follows that Budenske did not provide an enabling disclosure for implementing mixed processor types in its design.

Additionally, EX1007 notes that “[t]he implementation of the On–Off methodology entails addressing the **fundamental research issue** of how to select representative mappings off-line for on-line use.” EX1007 at 80 (emphasis added). As Dr. Long explained, if an issue is a fundamental research issue that means there is not yet a solution to the problem. EX2020 at 122:20-123:22 (“If you already have a solution, then it’s not really considered fundamental.”). The fact that the topic that

Petitioner claims Budenske discloses is still a “fundamental research issue”⁶ at least 7 years after Budenske was published demonstrates that Budenske was not an enabling disclosure. EX1007 confirms that several aspects of Budenske’s dynamic parameters were “n[ever] evaluated in any way, as this was not the focus of Budenske.” EX1007 at 79. And even seven years after Budenske, EX1007 still noted that its experiments were still forced to maintain the limitation that subtasks would be assigned to processors “of the same type.” EX1007 at 81 (“For simplicity, it is assumed that if a data-parallel implementation of a given subtask uses a virtual machine of processors, all processors will be of the same type.”). If Budenske was an enabling disclosure, why would such a simplification still be necessary in experiments carried out subsequently? There is absolutely no evidence presented in the Petition that Budenske included an enabling disclosure relative to assigning subtasks within a computing iteration to different types of processors, and the subsequent references cited in the Petition confirm this lack of enablement.

⁶ EX2021 ¶¶75-78; EX2020 at 122:20-123:3 (Dr. Long explaining that a “fundamental research issue” is an “interesting research question”), 123:19-22 (Dr. Long explaining that under the NSF definition “fundamental research” is “curiosity-driven research for unsolved problems”).

Other details provided by EX1007 further confirm that Budenske could not teach or render obvious distributing tasks between both computation nodes and booster nodes in the same computing iteration. EX1007 purports to teach how to use dynamic parameters to predict the “execution time expression for subtask.” EX1007 at 82-83. The equation provided for predicting subtask execution time makes clear that this prediction logic can only function if the subtasks are assigned to a single processor type. EX2021 ¶¶80-81. For example, the equation includes only a single “heterogeneity factor” h_{iu} , meaning it would only be applicable to a single type of processor. EX1007 at 83. EX1007 further confirms this single equation will only work for the system of EX1007 because “different subtasks will have affinities for different machines,” which a POSITA would have understood to mean that those subtasks will be assigned only to the type of machine for which it has an affinity. *Id.*; EX2020 at 120:10-122:16. These repeated limitations of only processing subtasks on a single type of processor fall far short of the teachings of the ’442 Patent, which explains that it is often necessary to redistribute subtasks from one node type to another in subsequent computing iterations. *See* section VI.A.1.a.i.

EX1008 reinforces the same conclusions. Yet again, it notes that sub-tasks are limited to processors **of the same type**. EX1008 at 156 (“Each of these subtasks may allow the use of multiple processors **of the same type** for efficient execution.” (emphasis added)), 158 (“an off-line mapping heuristic...uses the following

information to create a mapping...the expected execution time of each subtask on a set of processors (**of the same type**) assigned, as a function of the type and number of processors” (emphasis added)), 159 (“Because each subtask can be mapped to multiple processors **of the same type**...” (emphasis added)); EX2020 at 126:13-127:1 (“They’re saying that if I have a sub-task, all the things in the sub-task are doing the same thing, in which case, using the same type of processor makes sense. If you’re asking if there’s anything in the paper that teaches away from that, where doing the same thing should be done with multiple types of processors, I’m unaware of it.”).

Notably, despite the strong statements in all three documents (Budenske, EX1007, and EX1008) that Budsenke’s theories were not implemented, that the authors were still performing “fundamental research” to figure out how the theories could be implemented, and/or that the disclosures were limiting subtasks to only one type of processor, the Petition failed to even address those topics, much less offer analysis on enablement.

Further confirming the difficulty of mapping subtasks to different processor types, Dr. Kaeli cites several publications that discuss the difficulty of mapping to multiple types of processors, including an article addressing the difficulty of mapping to multiple different types of GPUs resulting in what it refers to as a “straggler problem” that results in “lower GPU utilization”, EX2021 ¶38 (citing

EX2014 at 179-180); EX2020 at 152:10-15 (“We would always like to be able to use our processors to their full capacity...”); EX2020 at 163:7-166:2 (agreeing that task distribution problems still existed even in 2024 based on EX2014). He also cites another article which notes that code that must utilize two different architectures (e.g., general purpose processor as well as a booster) “introduces significantly more complexity than was seen under [] other [] historical models, ‘leading to a dissonance with these traditional HPC system workloads.’” EX2021 ¶39 (quoting EX2015 at 1-2). These publications further confirm that a POSITA would not have understood the Budenske disclosure to teach mapping of subtasks to different types of processors at the same time.

(iv) A POSITA would not have modified the references in the claimed manner

As discussed above, the Petition Grounds fail to teach or render obvious at least ’442 Patent Claim limitations [1.3] and [9.5]. Furthermore, even if Budenske had taught limitations [1.3] and [9.5], a POSITA still would not have combined the references in the claimed manner. EX2021 ¶¶106-115.

As discussed above, Budenske would not have motivated a POSITA to distribute sub-tasks between processor types, suggesting that for its teachings to

work all processors utilized for a given task should be of the same type.⁷ EX1005 at 392. Budenske explained that for its solution to work it must be “assumed” that “if an implementation of a given subtask uses multiple processors, all processors will be of the same type” which would discourage a POSITA dealing with distributing tasks to disparate types of processors for different subtasks from “following the path set out in the reference.” EX1005 at 392; *Allergan*, 796 F.3d at 1305.

Budenske’s teachings would have strongly weighed against motivating a skilled artisan to develop a system according to the ’442 Patent claims. For example, a POSITA would have understood that the Budenske disclosure is limited to

⁷ In fact, when taken in combination with the subsequent teachings of those who purportedly attempted to implement the teachings of Budenske, *see Section VI.A.1.a.ii-iii* above discussing EX1007 and EX1008, a POSITA likely would have viewed Budenske as teaching away from using different types of processors during a computing iteration. *See Allergan, Inc. v. Sandoz Inc.*, 796 F.3d 1293, 1305 (Fed. Cir. 2015) (quoting *In re Gurley*, 27 F.3d 551, 553 (Fed. Cir. 1994)) (“A reference may be said to teach away when a person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken by the applicant.”).

processing new data each time Budenske starts a new “iteration” as Budenske refers to them. *See* EX1005 at 391, 395 (“After each execution iteration through the set of subtasks, the values of certain dynamic parameters of the application may change, such as the number of objects detected in the current frame of a real-time image stream being processed.”). This is consistent with Budenske’s focus on ATR tasks, which would continuously process new image data each “iteration” at the expense of being able to redistribute and reprocess previously processed data. *Id.* at 390-91, 395. In contrast, the ‘442 Patent allows evaluation of information relating to the processing of subtasks to generate a further distribution of subtasks, supporting raw data feedback loops for deeper analysis (*e.g.*, refining prior frames via boosters). *See* EX1001 at 2:44-50, 3:66-4:3. The approach taught by Budenske would not allow for the kind of dynamic, iterative refinement taught by the ’442 Patent, and a POSITA would have understood the limited approach taught by Budenske to not be analogous to Lippert, and also to teach away from combining Lippert with Budenske.

That conclusion is further reinforced by subsequent publications addressing the Budenske reference, which confirm a POSITA would not have had a reasonable expectation of success of combining its dynamic parameter teachings with other references, at least because “fundamental research” was still necessary on the topic for years after Budenske was published. *See, e.g.*, EX1007 at 80; EX1008.

To demonstrate obviousness, Petitioner must show “that a skilled artisan would have had reason to combine the teaching of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success from doing so.” *Redline Detection, LLC v. Star Envirotech, Inc.*, 811 F.3d 435, 449 (Fed. Cir. 2015) (internal quotation marks and citations omitted). One of the authors of Budenske (Howard Jay Siegel) noted in a subsequent publication that the purported methods of mapping tasks using “dynamic parameter values” in Budenske were not “ever evaluated in any way, as this was not the focus of Budenske et al.” EX1007 at 79. Thus, the authors of Budenske recognized and stated publicly that the “dynamic parameter values” and purported mappings that Petitioner relies on were not evaluated or proven to be usable, and a POSITA would have had no expectation of success in using them as taught by Budenske, much less in modifying a combination of Lippert and Budenske to use them in a manner that no reference of record actually taught was possible. The lack of an expectation of success is further reinforced by references cited by Dr. Kaeli that emphasized the difficulty of mapping tasks to processors of different types. *See* Section VI.A.1; EX2021 ¶¶38-39, 62-81.

The lack of flexibility in Budenske’s approach would also have taught away from combining the much later Lippert reference with Budenske. As discussed above in Section II.C.2, Budenske is reliant on an “off-line statically determined

mapping” approach that greatly limits its flexibility in distributing sub-tasks at runtime. *Id.* at 387 (emphasis changed). Thus, a POSITA looking to achieve the improvements taught by the ’442 Patent (*e.g.*, generating further distributions based on information relating to the **processing** of the plurality of sub-tasks and **dynamically** altering further distributions), would not have been motivated to try to implement Budenske’s static approach to mapping. *See, e.g.*, EX1001 at 5:37-44, 6:1-5; Section II.B.

The non-obviousness of the proposed combination is further reinforced by Kambatla. EX2021 ¶¶54-56, 113-114. Kambatla does not address the challenges of distributing sub-tasks in a more complicated heterogeneous architecture. And even in the homogeneous computing architecture contemplated by Kambatla, the reference noted that “[t]he amount of computing resources required to process a given task can be difficult to predict. It is inevitably difficult to accurately estimate the resource requirements of a job or its constituent tasks because: (i) resource usage of a task varies over time, and (ii) resource usage can vary across tasks of the same job based on the input they process.” EX1006 at [0005]. Those challenges would be compounded even further in the context of a heterogeneous architecture, wherein resource usage of a task varies not just by time and input, but based on the varied capabilities of different processor types as well. Thus, even years after Budenske, Petitioner’s other references continued to reinforce the difficulty of estimating

resource usage and distributing tasks (let alone distributing sub-tasks across disparate processor types during subsequent computing iterations). This further reinforces that a POSITA would not have considered it to have been obvious to combine Lippert or Kambatla with the teachings of Budenske.

As Dr. Kaeli notes, at the time Budenske was purportedly published, a heterogeneous system was typically customized for a very specific task (automatic target recognition in the case of Budenske). EX2021 ¶¶32-33, 50-53. This meant a task or subtask would (at best) be assigned to the most appropriate processor type, rather than to a mix of processors. *Id.* That again conforms with Budenske’s express teachings that a subtask would only be assigned to processors of a single type. EX1005 at 392.

Secondary Considerations further demonstrate the nonobviousness of the proposed combination. For example, Petitioner’s expert testified that the ’442 Patent claims enabled “dynamic reassignment” elements which “enables fault tolerance.” EX2020 at 71:1-19 (noting that “if I have one of the, say, booster nodes fail, then I could do a dynamic reassignment”). Dr. Long noted that this ability was “actually kind of cool.” *Id.* Such praise of the ’442 Patent invention (from the opposing party’s expert no less) is strong evidence of the nonobviousness of the ’442 Patent Claims. *See Apple Inc. v. Samsung Elecs. Co., Ltd.*, 839 F.3d 1034, 1053 (Fed. Cir. 2016) (“Evidence that the industry praised a claimed invention or a product that embodies

the patent claims weighs against an assertion that the same claimed invention would have been obvious. Industry participants, especially competitors, are not likely to praise an obvious advance over the known art.”).

(b) Generating, Using Information Relating to the Processing of the Plurality of Sub-Tasks...a Further Distribution

’442 Patent Claim 1 recites “**generating**, using information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes, a further distribution ... in a further computing iteration.” EX1001 Claim 1 at 5:37-43 (emphasis added). Petitioner relies on Budenske as allegedly teaching this limitation in Ground 1. Pet. at 29-32. However, Budenske fails to teach or render obvious this limitation, at least because all purported “distributions” of Budenske are determined offline **before** any processing of sub-tasks even begins.

Budenske is titled “A Method for the On-Line Use of Off-Line Derived Remappings of Iterative Automatic Target Recognition Tasks onto a Particular Class of Heterogeneous Parallel Platforms.” EX1005 at 387. Consistent with that, it computes all of its purported task mappings “off-line.” *Id.* at 387-89, 404. These mappings are not generated using information relating to the processing of the sub-tasks, at least because no sub-tasks have been processed when the mappings are created. As Petitioner’s expert explained, the “offline” calculation of Budenske “doesn’t count against the computation time for the task. You can let this run all

weekend or a month or however long you want to come up with a bunch of possible mappings that you can choose.” EX2020 at 147:8-13. On the other hand, he explained that “[a]n online heuristic is one you would have to do on the fly during computation.” *Id.* at 147:14-15; *see* EX1005 at 387. Thus, the off-line heuristics of Budenske that are not done during computation are not using information relating to the processing of the sub-tasks, at least because no processing of the sub-tasks has occurred yet. Thus, Budenske fails to teach or render obvious this limitation. *See* EX2021 ¶¶87-88.

(c) Information Relating to the Processing of the Plurality of Sub-Tasks

The Petitioner Grounds also fail against Claim 1 at least because they fail to teach or render obvious the claimed information relating to the processing of the plurality of sub-tasks. Claim 1 recites:

[G]enerating, using **information relating to the processing of the plurality of sub-tasks** by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes, a further distribution of the plurality of sub-tasks...

EX1001 Claim 1 at 5:37-41 (emphasis added); *see also* EX1001 Claim 5 ’442 Patent Claim 5 at 6:12-14 (“The method according to claim 1, wherein daemons operate in the plurality of computation nodes and the plurality of booster nodes to **generate the information**.” (emphasis added)).

The '442 Patent explains the nature of this information. “During a first iteration of the task, [1] the results of the execution of the sub-tasks are collected together with [2] the information from the daemons concerning the processing of the sub-tasks and the status of the nodes.” EX1001 at 4:23-27 (bracketed numbers added); *see also id.* at 4:32-37 (daemons “report status information ... enabling the calculation of subsequent iterations to be optimized by [] further adjustments to the allocation of sub-tasks to computation nodes and boosters.”). It is important to note that the '442 Patent makes a distinction between (1) the results of the execution of the sub-tasks, and (2) the information from the daemons concerning the processing of the sub-tasks and the status of the nodes. A POSITA would have understood from these specification teachings that the results of the execution of the sub-tasks (*e.g.*, the resulting values of the data processing) are **not** the claimed information relating to the processing of the plurality of subtasks generated by the daemons. EX2021 ¶¶83-86.

Petitioner relies on Budenske’s “dynamic parameters” as the “information relating to the processing of sub-tasks” recited in the Challenged Claims. Pet. at 30; EX2020 at 129:10-130:22. However, the dynamic parameters of Budenske relied on by Petitioner relate to the exact thing the '442 Patent distinguished as being separate from the claimed information—“the results of the execution of the sub-tasks.” EX1001 at 4:23-27; Pet. at 31 (“Therefore, the result of processing the sub-tasks—

the values of the dynamic parameters—is information relating to the processing of the sub-tasks.”).

For example, Petitioner’s expert testified that the dynamic parameters discussed in the last full paragraph of Budenske page 393 were the dynamic parameters that Petitioner was relying on for this limitation. EX2020 at 129:10-130:22. However, all of those parameters refer to the results of the execution of the sub-tasks analyzing the underlying data. The task being discussed in Budenske is “automatic target recognition (*ATR*).” EX1005 at 387; EX2020 at 132:21-25. The example given in Budenske is trying to identify a “tank” as an “object of interest.” EX1005 at 393. Relevant to that task, Budenske states that tracking of tanks would involve the use of:

[F]our dynamic parameters: (1) number of objects of interest being tracked in the last image frame, (2) average size of the objects currently being tracked (in terms of percent of the total number of pixels in the image), (3) whether the lighting conditions in the last image frame required requantization for better contrast for display, and (4) whether there was cloud cover detected in the last image frame that required special processing.

Id. at 393.

Notably, all of those dynamic parameters relate specifically to the results of the execution of the sub-tasks. Petitioner's expert confirmed the same, agreeing that all "refer to the data being processed." EX2020 at 130:1-22. Thus, those dynamic parameters cannot be the claimed "information relating to the processing of the plurality of sub-tasks" as asserted in the Petition.

2. Claim 3

(a) Dynamically Alter Further Distribution

Ground 1 also fails for Claim 3, at least because it fails to teach or render obvious "the resource manager dynamically alter[ing] further distribution during the computing of the computation task." EX1001 Claim 3 at 6:3-5. As discussed previously, Claim 1 recites:

generating, using information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes, **a further distribution** of the plurality of sub-tasks between the plurality of computation nodes and the plurality of booster nodes for processing thereby in a further computing iteration.

EX1001 Claim 1 at 5:37-43 (emphasis added).

Claim 3 depends from Claim 2 which depends from Claim 1. Claim 3 recites that "the application manager receives the information and processes the information

as input to the resource manager such that the resource manager **dynamically alters further distribution** during the computing of the computation task.” EX1001 Claim 3 at 6:1-5 (emphasis added). The “further distribution” referred to is the further distribution of the plurality of sub-tasks as shown in Claim 1.

The Petition fails to teach or render obvious the Claim 3 limitation “wherein the application manager receives the information and processes the information as input to the resource manager such that the resource manager dynamically alters further distribution during the computing of the computation task,” at least because the Petition Grounds for Claim 3 fail to even address distribution of sub-tasks.

Petitioner first relies on Lippert for this limitation, arguing that Lippert teaches “establish[ing] a dynamic assignment at runtime, which means during processing of the computation task.” Pet. at 39 (citing EX1004 [0013]) (internal quotes omitted). However, Lippert paragraph [0013] is addressing “assignment of a booster to a computation node,” not a further distribution of sub-tasks between computation nodes and booster nodes as claimed. As discussed above, Lippert does not teach redistribution of sub-tasks based on information related to the processing of the plurality of sub-tasks across multiple computing iterations. *See, e.g.*, Section II.D. Therefore, it could not teach or render obvious the claimed “further distribution” in a further computing iteration of Claim 3.

It is Claim 4 (which depends from Claim 3) that recites “dynamically alter[ing] assignment of the plurality of computation nodes and the plurality of booster nodes to each other.” EX1001 Claim 4 at 6:8-10. Petitioner improperly conflates these two limitations, and as a result the Petition ground is defective on its face. *See* EX2020 at 140:22-141:9 (confirming that Petitioner was treating the limitations of Claim 3 and Claim 4 as “essentially the same”). Even if the plain language of the claim did not rebut that assumption, the idea that dependent Claim 4 would simply repeat the same limitation as Claim 3 is not credible. *See World Class Tech. Corp. v. Ormco Corp.*, 769 F.3d 1120, 1125 (Fed. Cir. 2014) (“The doctrine of claim differentiation creates a presumption that distinct claims, particularly an independent claim and its dependent claim, have different scopes.”).

Regarding this issue, Petitioner asserts:

A POSITA would have understood that “the resource manager dynamically alters further distribution” as recited in this claim element encompasses “the resource manager dynamically alters assignment of the plurality of computation nodes and the plurality of booster nodes to each other,” because claim 4 depending from claim 3 specifies this, and further because the ’442 patent describes the resource manager adjusting booster assignments according to updated assignment information from the application manager.

Pet. at 40 (citing EX1001 at 4:27-31).

Petitioner’s conclusion that Claim 3’s “alter[ing] further [sub-task] distribution” encompasses Claim 4’s assignment of computation nodes and booster nodes to each other is unsupported. The portion of the ’442 Patent that Petitioner cites for support actually confirms that Petitioner’s theory is incorrect. ’442 Patent at 4:27-31 (emphasis added) is an embodiment wherein “[t]he application manager [] performs [1] reassignment of the sub-tasks for a subsequent iteration of the task and passes this updated assignment information to the resource manager which [2] **may also** adjust the assignment of boosters to nodes accordingly.” This confirms that, in that embodiment, assignment of sub-tasks in a subsequent iteration is a separate action from assigning booster nodes to computation nodes. In the cited embodiment, the application manager performs a reassignment of sub-tasks and passes this information to the resource manager. The resource manager then “may also”—that is, optionally and separately—adjust the assignment of boosters to nodes. The two actions are separate, with the action of adjusting the assignment of boosters to nodes being optional.

While Petitioner does not appear to rely on Budenske as teaching or rendering obvious this limitation, it should be noted that Budenske would also fail to do so. As discussed in Section VI.A.1.a, Budenske never distributes sub-tasks to both computation nodes and booster nodes in the same computing iteration. Furthermore,

all of Budenske’s mapping of subtasks is done offline, and not dynamically during the computing of the computation task. *See, e.g.*, Budenske at 387 (“The contribution of this paper is...using a previously stored off-line statically determined *mapping* (i.e., a matching of application subtasks to processors and a scheduling for the execution order of these subtasks), 389-390, 392-395, 398.

The testimony of Petitioner’s expert regarding this limitation should also be given little weight, at least because it simply repeats Petitioner’s conclusory analysis. *See Xerox Corp. et al. v. Bytemark, Inc.*, IPR2022-00624, Paper 9 at 15 (precedential). Furthermore, when asked to explain his awkward wording stating that one claim limitation “encompasses” another limitation, his response was “[t]hat’s not how I would write it.” EX2020 at 138:22-139:11. He did not explain why there is language in his declaration that is “not how [he] would write it,” and it reinforces that his declaration is merely parroting attorney argument.

3. Claim 5

(a) Computation Nodes Daemons and Booster Node Daemons

The Petition Grounds fail to teach or render obvious the computation node daemons and booster node daemons as claimed by Claim 5.

Claim 5 recites “[t]he method according to claim 1, wherein daemons operate in the plurality of computation nodes and the plurality of booster nodes to generate the information.” EX1001 at 6:12-14. The antecedent basis for the claimed

“information” comes from Claim 1’s “information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes.” *See id.* at 5:37-40. The Petition grounds fail to teach or render obvious the claimed daemons operating in computation nodes and daemons operating in booster nodes.

For Ground 1, Petitioner does not assert that either Lippert or Budenske discloses the concept of daemons, much less daemons operating in a plurality of computation nodes and a plurality of booster nodes to generate information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes. Pet. at 41-43.

Despite the absence of such a disclosure in its Ground 1, Petitioner relies on POSITA knowledge as a basis to render obvious the missing limitation, asserting that “a POSITA thus would have implemented Lippert-Budenske with daemons at each computation node and each booster node configured to collect and send [the] information.” Pet. at 42. Petitioner provides no evidence that the specific novel claimed arrangement (daemons operating in a plurality of computation nodes and a plurality of booster nodes to generate information relating to the processing of the plurality of sub-tasks by at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes) was known to a POSITA,

and as a result such reliance is improper. *See Shockwave Med., Inc. v. Cardiovascular Sys.*, 142 F.4th 1371, 1380 (Fed. Cir. 2025) (suggesting that claim limitations novel to the invention may not be supplied by “general background knowledge”); *see also* https://www.uspto.gov/sites/default/files/documents/aapa_memo_final_signed.pdf.

Regardless, Petitioner cannot succeed on this ground, at least because it has failed to show that the specific claimed arrangement is “unusually simple” and the technology is “particularly straightforward.” *See Koninklijke Philips N.V. v. Google LLC*, 948 F.3d 1330, 1338 (Fed. Cir. 2020). The mere existence of daemons generally does not mean the specific implementation of daemons in the claimed heterogeneous node types providing the information relating to the processing of the plurality of sub-tasks would be “unusually simple” or “particularly straightforward.” Furthermore, Petitioner’s expert confirmed that there were other ways to implement the proposed combination of Lippert-Budenske without utilizing daemons, and Petitioner failed to demonstrate why a POSITA would have selected the claimed daemon arrangement over those other possibilities. EX2020 at 127:18-128-7; EX2021 ¶¶89-92.

4. Claim 9

(a) Distribution of Sub-Tasks To Both Computation Nodes and Booster Nodes in Both a First and Further Computing Iteration

For the same reasons discussed above for Claim 1 in Section VI.A.1.a, the Petition Grounds fail for Claim 9. '442 Patent Claim 9 recites:

an application manager configured to receive information from daemons operating in at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes to **update a distribution** of the multiple sub-tasks **between** the plurality of computation nodes **and** the plurality of booster nodes in a further computing iteration.

'442 Patent Claim 9 at 6:38-43 (emphasis added).

Petitioner does not contend that Lippert teaches any of limitation [9.5]. *See* Pet. at 29-34, 52 (citing only Budenske). For the same reasons discussed above for Claim 1 in Section VI.A.1.a (incorporated by reference), the Petition Grounds fail for Claim 9. EX2021 ¶¶93-94.

(b) Computation Node Daemons and Booster Node Daemons

Similar to Claim 5, Claim 9 recites “receiv[ing] information from **daemons operating in at least the portion of the plurality of computation nodes and at least the portion of the plurality of booster nodes** to update a distribution of the multiple sub-tasks between the plurality of computation nodes and the plurality of

booster nodes in a further computing iteration.” EX1001 at 6:38-43 (emphasis added). For the same reasons discussed above in Section VI.A.3 for Claim 5, the Petition Grounds fail to teach or render obvious the computation node daemons and booster node daemons as recited by Claim 9. EX2021 ¶¶95-96.

5. The Remaining Claims

All claims not individually addressed in Section VI.A depend from one or more claims that are not taught or rendered obvious by the Petition grounds for the reasons listed above. As a result, patentability of all claims challenged under Ground 1 should be confirmed.

B. Ground 2 Fails to Teach or Render Obvious Any Challenged Claims

For Ground 2 Petitioner challenges only ’442 Patent Claims 2-5 and 8-10. Pet. at 53. Petitioner’s Ground 2 merely adds Kambatla to its Ground 1 Lippert and Budenske theories; however Kambatla fails to remedy the shortcomings identified above in Section VI.A for Ground 1. Therefore, Petitioner’s Ground 2 challenges should be rejected for the same reasons as identified for Ground 1.

1. Claim 3

(a) Dynamically Alter Further Distribution

As discussed above for Ground 1 in Section VI.A.2.a (incorporated here by reference), the Petition fails to teach or render obvious the Claim 3 limitation “wherein the application manager receives the information and processes the

information as input to the resource manager such that the resource manager dynamically alters further distribution during the computing of the computation task.” EX2021 ¶¶97-100.

While Petitioner ostensibly relies on Kambatla for its Ground 2 challenge to ’442 Patent Claim 3, Petitioner cites only generic statements that “Kambatla discloses that its central resource manager includes various modules configured to handle different tasks and feed information to other modules ‘to aid in resource allocation’” and “that ‘more or fewer components, organized in various configurations ... remain[] within the scope’ of Kambatla.” Pet. at 66 (citing EX1006 [0030]-[0036]). Petitioner has failed to demonstrate that Kambatla’s teachings are even applicable to heterogeneous computing systems, much less capable of dynamically altering further distributions between computation nodes and booster nodes in such systems. *See, e.g.*, Section II.C.3. Nothing cited in Kambatla teaches or renders obvious dynamically altering further distribution during the computing of the computation task as claimed, and therefore Petitioner’s Ground 2 fails for the same reasons as Ground 1.

2. Claim 5

(a) Computation Nodes Daemons and Booster Node Daemons

As discussed above for Ground 1 in Section VI.A.3.a (incorporated here by reference), the Petition fails to teach or render obvious the Claim 5 limitation

“wherein **daemons operate in** the plurality of **computation nodes** and the plurality of **booster nodes** to generate the information.” EX1001 Claim 5 at 6:12-14 (emphasis added).

For Ground 2, Petitioner relies on a general disclosure from Kambatla of a “resource manager daemon.” Pet. at 68-69. However, Kambatla fails to rectify the deficiencies in Ground 1. EX2021 ¶¶101-104.

First, as discussed above in Section II.C.3 (incorporated here by reference), Kambatla does not even address the concept of heterogeneous computing for processing of tasks, much less the inclusion of daemons in different types of computing nodes (e.g., the claimed computation nodes and booster nodes). For example, Dr. Long conceded during his deposition that even though Kambatla gives examples of potential processors that the system could use, it “does not require that the processors be different.” EX2020 at 82:3-8. Thus, it could not teach or render obvious daemons for the different node types.

The Petition analysis ignores the fact that ’442 Patent Claim 5 does not recite daemons in the abstract, but rather specifically daemons operating in computation nodes and daemons operating in booster nodes. EX1001 at 6:11-13, 6:38-41. As Petitioner’s expert noted, the daemons in computation nodes and booster nodes would “almost certainly” be different in the case of a “truly heterogeneous system.” EX2020 at 53:16-21.

Even if Kambatla’s generic disclosure of daemons in the abstract was enough to teach or render obvious the claimed daemons operating in the claimed computation nodes and booster nodes, Petitioner still fails to demonstrate why a POSITA would have made the proposed modifications based on Kambatla. Petitioner argues that a POSITA would have made the combination because (1) Kambatla discloses a heterogeneous computing system, (2) “a POSITA implementing Lippert-Budenske would have recognized the need to facilitate inter-element communication,” and (3) “the particular teachings of Kambatla follow the same principles as Lippert-Budenske.” Pet. at 58-61. As discussed above, point (1) is not supported by Kambatla. Point (2) is purely hindsight based—Petitioner’s own expert conceded that there were other ways to facilitate such information gathering. EX2020 at 127:18-128:21. Finally, point (3) is simply a restatement of point (1) based on the misinterpretation that Kambatla discloses a system capable of processing a computation task with different types of processors.

3. Claim 9

(a) Ground 2 fails to teach or render obvious the “further computing iteration” limitations

Petitioner’s Ground 2 challenge to Claim 9 suffers the same deficiencies as Ground 1, and therefore also fails. Ground 2 relies on the same Lippert and Budenske references as Ground 1 (frequently citing back to the same Ground 1 analysis), but also adds conclusory analysis of Kambatla for certain limitations. Pet. at 71-72.

Ground 2 does not address independent Claim 1, and thus fails to remedy the Claim 1 shortcomings in Ground 1. For limitation [9.5] of independent Claim 9, Petitioner does not directly cite anything from Kambatla, but instead vaguely refers back to “Sections VII.C.1, VIII.B.1, VIII.B.4.” Pet. at 72. None of those sections remedy the deficiencies in Petitioner’s Ground 1 analysis.⁸ EX2021 ¶105.

As an initial matter, as discussed above in Section II.C.3 (incorporated here by reference), Kambatla does not even address heterogeneous computing concepts for processing computation tasks, thus there is no disclosure addressing distribution of sub-tasks to different processor types in any context. Petitioner incorrectly characterizes Kambatla as relating to the “field of endeavor of executing computation tasks within a heterogeneous computing environment.” However,

⁸ Petition Section VII.C.1 is Petitioner’s Ground 1 analysis for Claim 1, and as discussed above that fails to teach or render obvious the “further computing iteration” limitations. Petition Section VIII.B.1 is Petitioner’s Ground 2 analysis of Claim 2, and Petition Section VIII.B.4 is Petitioner’s Ground 2 analysis of Claim 5. Pet. at 62-65, 68-70. Both cite to Kambatla for support regarding certain limitations in dependent claims, but nothing cited in either section (or anywhere else) could teach or render obvious updating distributions of sub-tasks to computation nodes and booster nodes in further computing iterations.

Petitioner cites nothing in Kambatla which demonstrates relevant heterogeneous task processing. *See* Pet. at 53.

For example, Petitioner cites Kambatla paragraphs [0002] and [0004] in support of its statement, but both paragraphs address only cluster computing concepts with nothing specifically related to heterogeneous computing or assigning tasks to different types of processors. Pet. at 53. Petitioner's expert defines cluster computing simply as "a collection of interconnected computing devices ("nodes") utilized as a single, unified computing resource." EX1003 ¶27. While a cluster can be implemented with multiple types of processors for processing tasks, there is no expectation that a cluster will necessarily include multiple types of processors. *See* EX2021 ¶¶31-33, 37. Nor is there an expectation that concepts related in general to cluster computing would be applicable to heterogeneous computing systems. As the above discussion of Budenske illustrates, using different types of processors to work on the same sub-tasks raises complex issues regarding system architecture, which designers like Budenske had to simplify or otherwise avoid. Even assuming for the sake of argument that some of Kambatla's techniques were applicable to heterogeneous computing, it certainly does not address distributing sub-tasks between different types of processors during further computing iterations. EX2021 ¶¶54-56.

4. A POSITA would not have modified the references in the claimed manner

As discussed above in Section VI.A.1.a.iv (incorporated her by reference), a POSITA would not have combined Lippert and Budenske in the claimed manner.

In addition, a POSITA would not have been motivated to further combine Lippert and Budenske with Kambatla, at least because Kambatla does not disclose processing a computation task using a heterogenous computing system. EX2021 ¶¶106-115. As discussed above in Section II.C.3, Kambatla never discusses the use of different processor types at the same time, distribution of tasks to different processor types, or anything else that might make it relevant to an actual heterogeneous computing architecture. As a result, a POSITA would not have been motivated to combine Lippert or Budenske with Kambatla.

5. The Remaining Claims

All claims not individually addressed in Section VI.B depend from one or more claims that are not taught or rendered obvious by the Petition grounds for the reasons listed above. As a result, patentability of all claims challenged under Ground 2 should be confirmed.

VII. Conclusion

Patent Owner respectfully requests that the Board confirm patentability of the Challenged Claims for the reasons stated herein.

Dated: September 30, 2025 Respectfully submitted,

By: /Michael F. Heim /
Michael F. Heim (Reg. No. 32,702)
Attorney for Patent Owner
ParTec AG

CERTIFICATE OF SERVICE

The undersigned certifies that pursuant to 37 C.F.R. § 42.6(e), a copy of the foregoing was served via email to lead and backup counsel of record for Petitioner as follows:

Lead Counsel for Petitioner
Andrew M. Mason (Reg. No. 64,034) andrew.mason@klarquist.com
Back-Up Counsel for Petitioner
Cameron D. Clawson, Reg. No. 73,509 cameron.clawson@klarquist.com
Sarah Jelsema, Reg. No. 70,804 sarah.jelsema@klarquist.com
Frank Morton-Park, Reg. No. 80,750 frank.morton-park@klarquist.com
KLARQUIST SPARKMAN, LLP 121 SW Salmon Street, Suite 1600 Portland, Oregon, 97204 503-595-5300 (phone) 503-595-5301 (fax) Msft-Partec@klarquist.com

Dated: September 30, 2025 Respectfully submitted,

By: /Michael F. Heim /
Michael F. Heim (Reg. No. 32,702)
Attorney for Patent Owner
ParTec AG

CERTIFICATE OF COMPLIANCE

Pursuant to 37 C.F.R. § 42.24(d), the undersigned hereby certifies that this brief complies with the type-volume limitation of 37 C.F.R. § 42.24 because this brief contains 13,653 words.

Dated: September 30, 2025 Respectfully submitted,

By: /Michael F. Heim/
Michael F. Heim (Reg. No. 32,702)
Attorney for Patent Owner
ParTec AG