Trials@uspto.gov
571-272-7822

Paper 33
Entered: September 18, 2023

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

THROUGHPUTER, INC.,
Patent Owner.

IPR2022-00527
Patent 9,632,833 B2

Before MICHAEL R. ZECHER, JOHN A. HUDALLA, and MATTHEW S. MEYERS, *Administrative Patent Judges*.

MEYERS, *Administrative Patent Judge*.

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

I. INTRODUCTION

A. Background and Summary

Microsoft Corporation ("Petitioner") filed a Petition (Paper 2, "Pet.") requesting an *inter partes* review of claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 ("the challenged claims") of U.S. Patent No. 9,632,833 B2 (Ex. 1001, "the '833 patent"). ThroughPuter, Inc. ("Patent Owner") filed a Preliminary Response. Paper 7. We determined that the information presented in the Petition established that there was a reasonable likelihood that Petitioner would prevail with respect to its unpatentability challenges. Pursuant to 35 U.S.C. § 314, we instituted this proceeding on September 19, 2022, as to all challenged claims and all grounds of unpatentability presented in the Petition. Paper 12 ("Dec. on Inst.").

During the course of trial, Patent Owner filed a Patent Owner Response (Paper 17, "PO Resp."), and Petitioner filed a Reply to the Patent Owner Response (Paper 20, "Pet. Reply"). Patent Owner also filed a Sur-reply (Paper 28, "PO Sur-reply"). An oral hearing was held on June 20, 2023, and a transcript of the hearing is included in the record. Paper 32 ("Tr.").

We have jurisdiction under 35 U.S.C. § 6. This decision is a Final Written Decision under 35 U.S.C. § 318(a) as to the patentability of claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the '833 patent. For the reasons discussed below, Petitioner has demonstrated by a preponderance of the evidence that these claims of the '833 patent are unpatentable.

B. Real Parties-in-Interest

Petitioner identifies itself as only the real party-in-interest. Pet. 73. Patent Owner identifies itself as the only real party-in-interest. Paper 11, 2.

C. Related Proceedings

The parties identify that the '833 patent is involved in *ThroughPuter, Inc. v. Microsoft Corporation*, No. 2:22-cv-00344 (W.D. Wash). Paper 10, 1; Paper 11, 2. Patent Owner indicates that, "[a]s of August 18, 2022, the related district court litigation has been stayed pending resolution of the present *inter partes* review proceeding." Paper 11, 2.

We also note that Petitioner has challenged other patents owned by Patent Owner in IPR2022-00528, IPR2022-00574, IPR2022-00757, and IPR2022-00758. We denied institution of *inter partes* review in IPR2022-00757 and IPR2022-00758. Final Written Decisions in IPR2022-00528 and IPR2022-00574 are being issued concurrently with this Decision.

D. The '833 Patent

The '833 patent is titled "Scheduling Application Instances to Processor Cores Over Consecutive Allocation Periods Based on Application Requirements." Ex. 1001, code (54). The '833 patent pertains to "techniques for improving information processing efficiency and performance through dynamically adapting processing resource types to match processing task types." *Id.* at 1:37–41. In this regard, the '833 patent discloses that

> [o]bjectives for the core 120 allocation algorithm 310 include maximizing the processor 100 core utilization (i.e., generally minimizing, and so long as there are ready [application-task/instances], eliminating core idling), while ensuring that each application 220 gets at least up to its entitled (e.g. a contract based minimum) share of the processor core capacity whenever it has processing load to utilize such amount of cores.

Id. at 13:1–7.

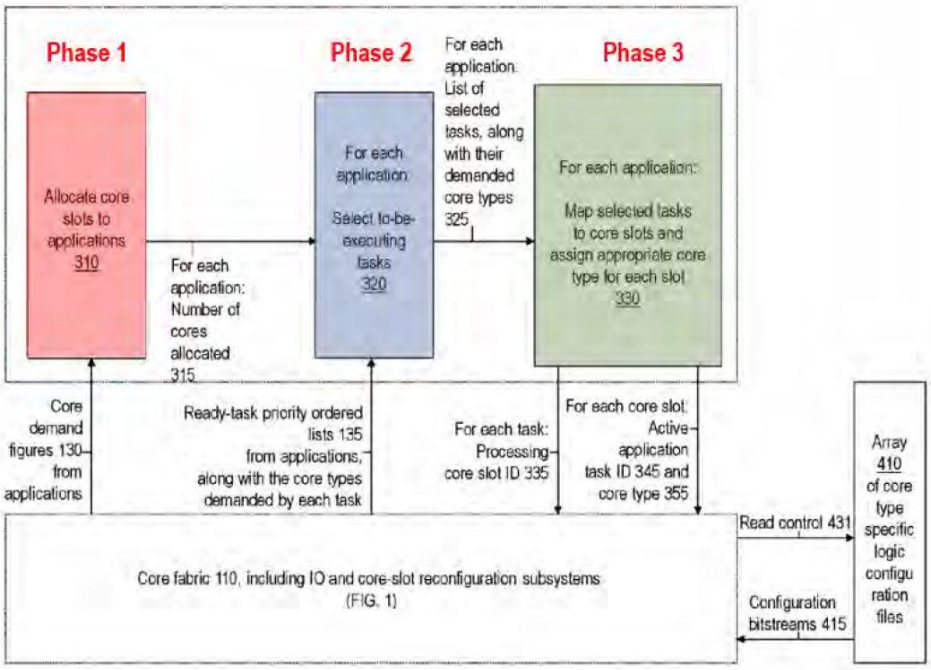
The '833 patent describes that "the range of processing jobs for a given instance of processing hardware (e.g. a server blade used for computing Infrastructure-as-a-Service), especially over the lifetime of the given hardware instance, may comprise several types of jobs and their tasks, with each type best suited for its corresponding, distinct, processor type." Ex. 1001, 1:59–64. The '833 patent further describes an array of hardware processing units, including different processor types, will likely run several types of jobs, each best suited to different processor types. *Id.* at 1:64–2:3. The '833 patent describes that it is quite impractical to predict "when a given instance of processing (having a hardware with a processor core or array of them) is deployed for service, to know what would be the optimal type of core for any given processor instance," and as such, the '833 patent identifies that conventional scheduling techniques often lead to jobs processed "by suboptimal types of processing cores." *Id.* at 2:4–17. The '833 patent "provides a processing task type adaptive manycore processor system for executing time variable sets of processing software program tasks of differing types on their assigned cores of matching types." *Id.* at 4:1–9. The '833 patent thus provides "application processing load adaptive allocation of the cores among the software applications configured for the system, as well as for dynamically reconfiguring the core slots according to the types of the processing tasks assigned to any given core slot of the multi-core fabric." *Id.* at 8:44–53.

To accomplish this, the '833 patent utilizes for each application "capacity demand indicators 130" or "core-demand-figures (CDFs)" that "express how many cores 120 their associated app-task is presently able [to] utilize for its ready to execute instances." Ex. 1001, 9:1–7. The system

> repeatedly selects and places the to-be-executing instances 240 of the set of locally hosted app-tasks 210 to their assigned target core slots 120, as well as determines, for the core-slot reconfiguration subsystem of the processor 100 (FIGS. 3 and 4), the demanded core types for the core slots of the array 115, along with notifications of changes in the demanded core type for each given core slot.

Id. at 10:14–21.

Petitioner has supplied an annotated version of Figure 3, reproduced below, which is a block diagram of the system of the '833 patent.



Pet. 9 (reproducing Figure 3 of the '833 patent). Figure 3 shows algorithm 310, which allocates core slots to applications, application selection step 320, and core mapping step 330. Ex. 1001, 13:29–14:44.

E. The Challenged Claims

Petitioner challenges claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the '833 patent. Pet. 2. Challenged claims 10, 15, 19, 29, and 34 are independent claims. Claims 15 and 19 are illustrative of the challenged

claims, and are reproduced below, with Petitioner's bracketing, labels, and formatting.

> [15.P] A method for assigning instances of software programs to an array of processor cores comprising:
>
> [15.1] for each of a series of successive core allocation periods (CAPs), selecting, from a group of executable instances of a set of software programs, a subset of the executable instances, referred to as selected instances, for execution on the cores of the array for an upcoming CAP,
>
> [15.2] wherein the selection of the selected instances is based, at least in part, on a respective capacity demand indication of each of the set of software programs, with said indication of a given program (a) being based at least in part on a number of its executable instances that presently have input data available for processing and (b) indicating processor core types demanded by its executable instances;
>
> [15.3] assigning each of the selected instances for execution on a processor core in the array of processor cores based, at least in part, on matching the respective demanded processor core types associated with the selected instances with types of processor cores available for assignment; and
>
> [15.4] executing the selected instances on their assigned cores over the next CAP, at least in part, to process the input data.

Ex. 1001, 21:60–22:16; Pet. vii.

> [19.P]. A process for computing resource management, the process comprising:
>
> [19.1] a sub-process configured to periodically, once for each of a series of core allocation periods (CAPs), to carry out an allocation of an array of processor cores among a set of software programs, said sub-process comprising:
>
> [19.2] (i) a first round of the allocation, in which a subset of the cores are allocated among the software programs so that any actually materialized demands for the cores by each of the

software programs up to their respective entitled shares of the cores are met, and

[19.3] (ii) a second round of the allocation, in which any of the cores that remain unallocated after the first round are allocated among the software programs whose materialized demands for the cores had not been met by amounts of the cores so far allocated to them by the present invocation of the allocation; and

[19.4] a sub-process configured to assign individual instances of the set of software programs to individual cores of the array so that each such instance of the software programs, which was selected for execution on the array of cores on consecutive CAPs, is assigned to same one of the cores for execution on each of such consecutive CAPs.

Ex. 1001, 22:6–50; Pet. viii.

F. Instituted Grounds of Unpatentability

We instituted trial based on all asserted claims and grounds of unpatentability as follows:

Claims Challenged	35 U.S.C. §[1]	References
15, 17, 18, 34, 37	103(a)	Agrawal[2], Brent[3]
10, 13, 14, 19–22, 24, 25, 27, 29, 32, 33	103(a)	Agrawal, Brent, Feitelson[4]

[1] The Leahy-Smith America Invents Act ("AIA"), Pub. L. No. 112-29, 125 Stat. 284 (2011), amended 35 U.S.C. §§ 102 and 103 effective March 16, 2013. Because the '833 patent claims priority as a continuation to an application filed before March 16, 2013, we apply the pre-AIA versions of the statutory bases for unpatentability. *See* Ex. 1001, code (63).

[2] Kunal Agrawal et al., "Adaptive Scheduling with Parallelism Feedback," Proceedings of the 2006 ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2006 (Ex. 1004, "Agrawal").

[3] US 2010/0131955 A1, published May 27, 2010 (Ex. 1005, "Brent").

[4] Dror G. Feitelson, "Job Scheduling in Multiprogrammed Parallel Systems," IBM Research Report RC 19790 (87657), 2nd Rev., August 1997 (Ex. 1006, "Feitelson").

Pet. 2; Dec. on Inst. 40–41.

Petitioner submits a declaration of Jeffrey S. Chase, Ph.D., with its Petition (Ex. 1003) and second declaration of Dr. Chase with its Reply (Ex. 1032) in support of its contentions. Patent Owner submitted a Declaration of Melissa C. Smith, Ph.D. (Ex. 2003) in support of its Response. Ex. 2003. Dr. Chase was cross-examined. *See* Ex. 2005 (deposition transcript of Dr. Chase).[5] Dr. Smith was cross-examined. *See* Ex. 1033 (deposition transcript of Dr. Smith).

II. ANALYSIS

A. Principles of Law

"In an [*inter partes* review], 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) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review petitions to identify "with particularity . . . the evidence that supports the grounds for the challenge to each claim")). Petitioner bears the burden of persuasion to prove unpatentability of each challenged claim by a preponderance of the evidence. 35 U.S.C. § 316(e). This burden does not shift to Patent Owner, except in limited circumstances not present here. *Dynamic Drinkware, LLC v. Nat'l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015).

A patent claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are such that

[5] We refer to the page numbers inserted by Patent Owner at the bottom, right portion of each page in Exhibit 2005.

the subject matter, as a whole, "would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) when in evidence, any objective evidence of obviousness or non-obviousness.[6] *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). "While the sequence of these questions might be reordered in any particular case" (*KSR*, 550 U.S. at 407), the U.S. Court of Appeals for the Federal Circuit has explained that an obviousness determination can be made only after consideration of all of the *Graham* factors. *See, e.g.*, *Kinetic Concepts, Inc. v. Smith & Nephew, Inc.*, 688 F.3d 1342, 1360 (Fed. Cir. 2012).

In analyzing the obviousness of a combination of prior art elements, it can be important to identify a reason that would have prompted one of skill in the art "to combine . . . known elements in the fashion claimed by the patent at issue." *KSR*, 550 U.S. at 418. A precise teaching directed to the specific subject matter of a challenged claim is not necessary to establish obviousness. *Id.* Rather, "any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed." *Id.* at 420. Accordingly, a party who petitions the Board for a determination of unpatentability based on obviousness must show that "a skilled artisan

[6] The parties do not present objective evidence of non-obviousness.

would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success in doing so." *In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1381 (Fed. Cir. 2016) (quotations and citations omitted).

B. Level of Ordinary Skill in the Art

We review the grounds of unpatentability presented in the Petition in view what a person of ordinary skill in the art would have understood at the time of invention. *Graham*, 383 U.S. at 17. Petitioner asserts that a person of ordinary skill in the art at the time of the invention "would have had a Master's degree in computer science, computer engineering, or a related field, and 2–3 years of practical computer programming or engineering experience, including experience designing or researching parallel processing systems." Pet. 10–11 (citing Ex. 1003 ¶¶ 18–20). Petitioner adds that "[a]dditional graduate education could substitute for professional experience, or significant experience in the field could substitute for formal education." *Id.* at 11. Patent Owner does not dispute or otherwise address the level of ordinary skill in the art. *See generally* PO Resp.; PO Sur-reply.

We adopt Petitioner's definition of the level of ordinary skill in the art and we apply it in our obviousness evaluations below. We are satisfied that this definition comports with the level of skill necessary to understand and implement the teachings of the '833 patent and the asserted prior art.

C. Claim Construction

In an *inter partes* review, we construe patent claims using the same claim construction standard that would be used to construe the claim in a

civil action under 35 U.S.C. § 282(b). 37 C.F.R. § 42.100(b) (2021). This rule adopts the same claim construction standard used by Article III federal courts, which follow *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (*en banc*), and its progeny. Under that standard, the words of a claim are generally given their "ordinary and customary meaning," which is the meaning the term would have to a person of ordinary skill at the time of the invention, in the context of the entire patent including the specification. *See Phillips*, 415 F.3d at 1312–13.

Petitioner asserts that "[a]ll claim terms should be construed according to the *Phillips* standard. Pet. 11. In addition, Petitioner submits that the term "instance" should be construed to mean "a program 'task,'" and that in the case of the plural "instances," "there is no requirement in the claims for the 'instances' to be instances of the *same* 'task.'" *Id.* at 11–13 (citing Ex. 1003 ¶¶ 48–50). Patent Owner does not comment on Petitioner's proposed construction, or otherwise address claim construction. *See generally* PO Resp.; PO Sur-reply.

We determine that no aspects of the challenged claims require explicit construction. *See, e.g.*, *Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) ("The Board is required to construe 'only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy.'" (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

D. Obviousness over Agrawal and Brent (Ground 1A)

Petitioner contends that claims 15, 17, 18, 34, and 37 would have been unpatentable as obvious over Agrawal and Brent. Pet. 13–38. Petitioner

also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003). Patent Owner responds to Petitioner's assertions. *See* PO Resp. 1–29; PO Sur-reply 1–23.

1. Overview of Agrawal (Ex. 1004)

Agrawal is titled "Adaptive Scheduling with Parallelism Feedback." Ex. 1004, 1.[7] Agrawal relates to multiprocessor computer system scheduling in a shared multiprogramming environment. *Id.* Agrawal presents "an adaptive task scheduler for multitasked jobs with dependencies that provides continual parallelism feedback to the job scheduler in the form of requests for processors." *Id.* Agrawal focuses on "space-sharing" for parallel jobs, using a two-level strategy: "a kernel-level job scheduler which allots processors to jobs, and a user-level task scheduler which schedules the tasks belonging to a given job onto the allotted processors." *Id.*

Using "parallelism feedback," Agrawal's scheduler "adaptively changes the allotment of processors according to the availability of processors in the current system environment and the job scheduler's administrative policy." Ex. 1004, 2. Agrawal's adaptive task scheduler does not focus on job scheduling, but rather on task scheduling, and, "[i]nstead of using instantaneous parallelism," Agrawal's A-Greedy algorithm "provides parallelism feedback to the job scheduler based on a single summary statistic and the job's behavior on the previous quantum." *Id.* Agrawal describes that its A-Greedy algorithm "provides parallelism feedback using the past

[7] Exhibit 1004 has no original page numbers in the filed version. We rely on the page numbers inserted by Petitioner in the bottom, middle portion of each page in Exhibit 1004.

behavior of the job" and does "not assume that the job's future parallelism is correlated with its history of parallelism." *Id.* According to Agrawal:

> Between quanta $q - 1$ and q, the task scheduler determines its job's ***desire*** d_q, which is the number of processors the job wants for quantum q. The task scheduler provides the desire d_q to the job scheduler as its parallelism feedback. The job scheduler follows some processor allocation policy to determine the ***processor availability*** p_q, which is the number of processors the job is entitled to get for the quantum q.

Id. Agrawal discloses that "[t]he number of processors the job receives for quantum q is the job's ***allotment*** $a_q = \min\{d_q, p_q\}$, which is the smaller of its desire and the processor availability." *Id.*

Agrawal also describes "instantaneous parallelism," which relies on "the number of processors the job can effectively use at the current moment, as the parallelism feedback to the job scheduler." Ex. 1004, 2. Agrawal discloses that, "[a]lthough using instantaneous parallelism for parallelism feedback is simple, it can cause gross misallocation of processor resources." *Id.* According to Agrawal,

> the parallelism of a job may change substantially during a scheduling quantum, alternating between parallel and serial phases. The sampling of instantaneous parallelism at a scheduling event between quanta may lead the task scheduler to request either too many or too few processors depending on which phase is currently active, whereas the desirable request might be something in between. Consequently, the job may systematically waste processor cycles on the one hand or take too long to complete on the other.

Id. Agrawal acknowledges that A-Greedy "operates nearly as efficiently" as a "task scheduler that uses instantaneous parallelism as feedback," and that a scheduler using instantaneous parallelism wastes no processor cycles. *Id.* at 3.

2. Overview of Brent (Ex. 1005)

Brent is titled "Highly Distributed Parallel Processing on Multi-Core Device." Ex. 1005, code (54). Brent relates to multi-core processing systems, and discloses a highly distributed multi-core system with an adaptive scheduler. *Id.* ¶¶ 3, 8.

Figure 1 of Brent, reproduced below, shows an exemplary multi-core system with an adaptive scheduler.

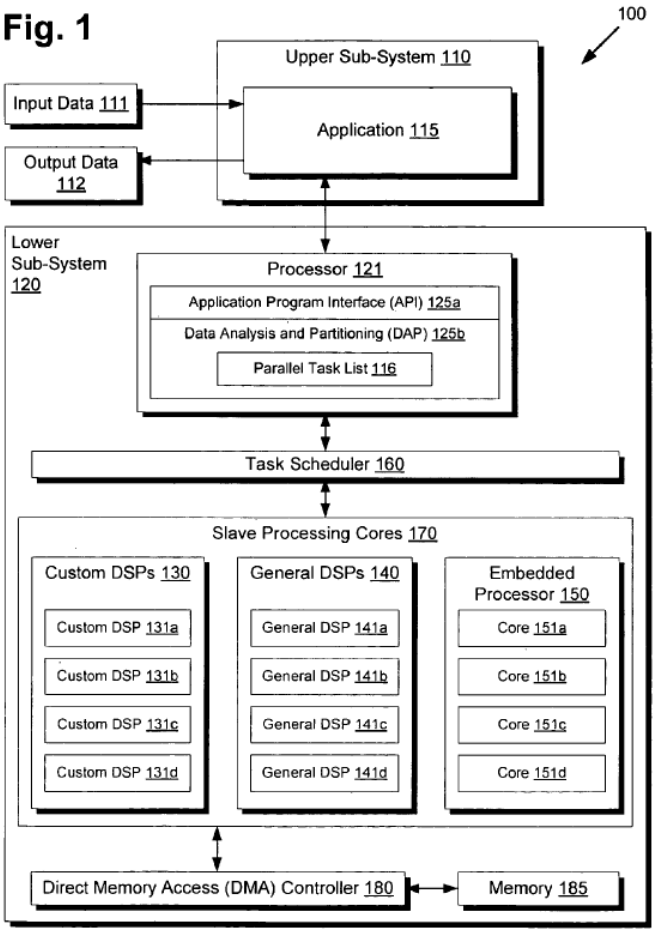


Figure 1 shows, among other things, input data 111, application 115, and processor 121 which creates task list 116, which is passed to task scheduler 160 for execution on slave processing cores 170. Brent discloses that "slave processing cores 170 may include several different types of processing

cores. Custom digital signal processors (DSPs) 130 include custom DSPs 131*a*–131*d*, which may have limited instruction sets optimized for specific processing patterns." *Id.* ¶ 16. Brent describes that "each task may be assigned a preferred core type for optimal execution." *Id.* ¶ 18.

3. Claim 15

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 15 as set forth in the Petition. Pet. 24–37 (citing Exs. 1003, 1004, 1005). We use Petitioner's notations to identify the claim limitations.

i. [15.P] A method for assigning instances of software programs to an array of processor cores comprising:[8]

Petitioner asserts that, if the preamble is treated as limiting, the combination of Agrawal and Brent discloses its contents. Pet. 24 (citing Ex. 1003 ¶¶ 71–74). More particularly, Petitioner asserts that "Agrawal discloses 'two-level scheduling' in which, at each scheduling quantum, a 'job scheduler . . . allots processors to jobs,' and a 'task scheduler [] schedules the tasks belonging to a given job onto the allotted processors.'" *Id.* at 24–25 (citing Ex. 1004, 1–2). Petitioner further asserts that Brent discloses task schedulers "that 'distribut[e]' a selected 'subset of tasks' to a[] collection of 'slave processing cores 270.'" *Id.* at 25 (citing Ex. 1005 ¶¶ 29, 16–21, Figs. 1–3). According to Petitioner, in the asserted combination, "task schedulers for a set of jobs that were allocated cores in a scheduling quantum assign selected 'tasks' (***instances***) of the jobs (***software programs***)

[8] The issue of whether the preamble is limiting need not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

to an array of processing cores, e.g., cores 270." *Id.* (citing Ex. 1003 ¶ 72, n.1).

ii. *[15.1] for each of a series of successive core allocation periods (CAPs), selecting, from a group of executable instances of a set of software programs, a subset of the executable instances, referred to as selected instances, for execution on the cores of the array for an upcoming CAP;*

Petitioner asserts that Agrawal's two-level scheduling operations "periodically repeat for each of a series of successive scheduling quanta (***core allocation periods (CAPs)***)." Pet. 26 (citing Ex. 1004, 2). Petitioner further asserts that, during each quanta, Agrawal's "task scheduler 'schedules the tasks belonging to a given job onto the allotted processors.'" *Id.* (citing Ex. 1004, 1). Petitioner further asserts that, "[a]s Brent explains, scheduling tasks of a job, e.g., from an application 115, onto an array of processor cores involves selecting a subset of tasks for execution on the cores." *Id.* (citing Ex. 1005 ¶¶ 15, 20, 27, Figs. 1–3; Ex. 1003 ¶ 75).

According to Petitioner, the combination resulting from Agrawal and Brent "would provide a job scheduler that received parallelism feedback from each of the jobs' respective task schedulers," "allot a set of cores (processors) to each job based on the job's desired number of cores indicated in its parallelism feedback," and for each job, the task scheduler "would then assign the job's ready-to-execute task(s) to respective ones of the cores allotted to the job, and based on Brent, seek to match tasks to their preferred core types according to the tasks' type-affinity indicators." Pet. 21 (citing Ex. 1003 ¶ 65; Ex. 1004, 1–2; Ex. 1005 ¶¶ 21–23, 27–29, Figs. 1–3). Petitioner explains, that in Agrawal,

> "desire" is "the number of processors the job wants for quantum *q*"; "allotment" is "[t]he number of processors the job

> receives for quantum *q* …, which [is] the smaller of its desire and the processor availability"), [and] ("***instantaneous parallelism***" is "the number of processors the job can effectively use at the current moment").

Id. at 28 (citing Ex. 1004, 2). Petitioner reasons

> [t]o accommodate an allotment of fewer cores to the jobs than desired (e.g., fewer than the number of "ready-to-execute" tasks that the "job [could] effectively use at the current moment"), the respective task schedulers of Jobs 1 and 2 select a subset of their tasks for execution during the scheduling quantum.

Id. Relying on the declaration testimony of Dr. Chase, Petitioner concludes, that because

> this same process is repeated for each successive scheduling quantum (***CAP***) based on the teachings of Agrawal, the Agrawal-Brent combination would select in each CAP, from a group of executable instances of a set of software programs . . . , a subset of the executable instances . . . for execution on the cores of the array for an upcoming quantum (***CAP***).

Id. at 28–29 (citing Ex. 1003 ¶ 78).

iii. [15.2] wherein the selection of the selected instances is based, at least in part, on a respective capacity demand indication of each of the set of software programs, with said indication of a given program (a) being based at least in part on a number of its executable instances that presently have input data available for processing and (b) indicating processor core types demanded by its executable instances;

Petitioner asserts that, in Agrawal, "[a] task becomes ***ready*** to be executed when all its parents have been executed." Pet. 29 (citing Ex. 1004, 1). Petitioner explains that Agrawal describes "instantaneous parallelism" as "feedback provided to the job scheduler from each job's respective task scheduler [that] indicates 'the number of processors the job can effectively use at the current moment.'" *Id.* (citing Ex. 1004, 2). According to

Petitioner, one of ordinary skill in the art "would have understood, including for long-task jobs (*see* [Pet. 13–17]), that 'the number of processors the job can effectively use at the current moment'—i.e., the measure for 'instantaneous parallelism'—corresponds and equates to the number of tasks that the job has ready to execute." *Id.* at 29–30 (citing Ex. 1003 ¶ 79). Petitioner concludes that "[t]he parallelism feedback (***capacity demand indication***) of a given job (***given program***) in the Agrawal-Brent combination is therefore based at least in part on a number of the job's tasks (***executable instances***) that presently have input data available for processing." *Id.* at 30 (citing Ex. 1003 ¶¶ 54, 57). Petitioner also notes that both Agrawal and Brent describe that all dependent tasks must be resolved before a dependent task may become ready to execute. *Id.* (citing Ex. 1004, 1; Ex. 1005 ¶¶ 20, 26–27, Figs. 2–3).[9]

With respect to "processor core types," Petitioner asserts that Brent discloses "an indication for an application (***a given software program***) of the processor core types demanded by its tasks (***executable instances***)." Pet. 31 (citing Ex. 1005 ¶¶ 16–18, 22, 27, 29, Fig. 2). According to Petitioner, "[i]t would have been obvious to apply Brent's teachings for indicating demanded processor core types in combination with Agrawal." *Id.* (citing Pet. 19–24; Ex. 1003 ¶ 81).

> More specifically, Petitioner asserts that one of ordinary skill in the art would have found it obvious to combine the teachings of Agrawal and Brent, thereby implementing a two-level

[9] Alternatively, Petitioner asserts that this aspect of the limitation would have been obvious "in view of Brent to count a task as ready to execute in the resulting combination based on the task having input data available to process." Pet. 30–31 (citing Ex. 1003 ¶ 80).

> scheduling strategy like that disclosed in Agrawal in the context of a system that provides an array of processing cores of various types (e.g., embedded CPUs, custom DSPs, and general DSPs) as taught in Brent.

Pet. 21 (citing Ex. 1003 ¶ 65; Ex. 1005 ¶¶ 16–17). Petitioner provides several reasons why one of ordinary skill in the art would have been motivated to combine the teachings of Agrawal and Brent. *Id.* at 21–24 (citing Ex. 1003 ¶¶ 66–70). For example, Petitioner asserts that both references are directed to "scheduling tasks in a parallel processing environment, including scheduling techniques that account for task dependencies and application demands that change over time." *Id.* at 21–22 (citing Ex. 1004, 1–3; Ex. 1005 ¶¶ 13, 20, 26). Moreover, Petitioner asserts that Brent offers additional implementation details that Agrawal lacks, including the use of heterogeneous processor cores and type-affinity scheduling to match tasks to specific core types. *Id.* at 22 (citing Ex. 1005 ¶¶ 18, 21–23). According to Petitioner, "[i]t would have been obvious to apply these concepts from Brent in combination with Agrawal to achieve predictable benefits, including an improved ability to optimize task execution on a processor whose instruction set matches the instruction types provided in a task." *Id.* (citing Ex. 1003 ¶ 67; Ex. 1005 ¶¶ 16, 18–19).

iv. [15.3] assigning each of the selected instances for execution on a processor core in the array of processor cores based, at least in part, on matching the respective demanded processor core types associated with the selected instances with types of processor cores available for assignment; and

> Petitioner asserts that the combination of Agarwal and Brent discloses task schedulers for each job (***software program***) that collectively assign each of the selected tasks (***selected instances***) for execution on a processor core in the array (e.g., array 170)

> based, at least in part, on matching the respective demanded processor core types associated with the selected tasks (***selected instances***) with types of processor cores available for assignment.

Pet. 34–35 (citing Ex. 1003 ¶¶ 85–86; Ex. 1004, 1; Ex. 1005 ¶¶ 16–23, 27–29; Pet. 19–24, 29–34).

v. [15.4] executing the selected instances on their assigned cores over the next CAP, at least in part, to process the input data.

Petitioner asserts that the combination of Agrawal and Brent "discloses executing the selected tasks (***selected instances***) on their assigned cores over the next scheduling quantum (***CAP***), at least in part, to process the input data." Pet. 36 (citing Ex. 1003 ¶ 87; Ex. 1004, 2; Ex. 1005 ¶¶ 14, 18, 20, 26–27, Fig. 2; Pet. 19–24).

4. Patent Owner's Arguments

At the outset, Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with respect to limitations 15.P–15.4. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that the combination Agrawal and Brent teaches these limitations of claim 15.

Instead, Patent Owner's arguments are based primarily on its contention that Petitioner impermissibly modifies Agrawal's task scheduler to provide feedback to its job scheduler in the form of instantaneous parallelism. PO Resp. 1–2; PO Sur-reply 1–2. More particularly, Patent Owner argues that Agrawal teaches away from the use of instantaneous parallelism because "Agrawal expressly considers and rejects the use of instantaneous parallelism feedback because it 'can cause gross misallocation

of processor resources' in a system of the type claimed and disclosed in the '833 patent." PO Resp. 7–8 (quoting Ex. 1004, 2).

Patent Owner asserts that Agrawal's disclosure is directed to an "A-GREEDY task scheduler [that] will outperform the use of instantaneous parallelism with respect to scheduling efficiency." *Id.* at 13 (citing Ex. 1004, 3; Ex. 2003 ¶¶ 46–47). And, according to Patent Owner, any suggestion in Agrawal "that adaptive scheduling based on instantaneous parallelism can be performed" is limited to "the 'special case' of unit quanta" because "the whole point of Agrawal was to design a robust scheduler that operates beyond the 'special case.'" *Id.* at 9.

To support its position, Patent Owner points to a thesis by Siddhartha Sen[10, 11], cited in Agrawal (Ex. 1004, 2 n.48), that "weighs the pros and cons of various scheduling approaches and determines that core allocation based on instantaneous parallelism is 'inherently misleading.'" PO Resp. 8 (citing Ex. 2004, 3). According to Patent Owner, "Sen discusses the inaccuracy of instantaneous parallelism with reference to a two-level scheduling system including an OS that allocates processor (akin to Agrawal's job scheduler) and a job scheduler that assigns tasks to processors (akin to Agrawal's task scheduler)." *Id.* at 12 (citing Ex. 2004, 30; Ex. 2003 ¶ 34). Patent Owner concludes that "Sen's determination that instantaneous parallelism can be 'inherently misleading' is consistent with Agrawal's own study." *Id.* at 13

[10] Siddhartha Sen, "Dynamic Processor Allocation for Adaptively Parallel Work-Stealing Jobs," Master's thesis, Massachusetts Institute of Technology, Sept. 2004 (Ex. 2004, "Sen").
[11] We note that Sen was not cited in any of the grounds of unpatentability presented by Petitioner, but rather, Sen was cited by Agrawal as evidence of earlier research in the field of task scheduling. *See generally* Ex. 1004.

(citing Ex. 2003 ¶ 35). Consequently, Patent Owner argues that "Agrawal teaches away from a design based solely on the 'special case.'" *Id.* at 9.

In addition, Patent Owner argues that modifying Agrawal to operate in this manner would fundamentally change Agarwal's principle of operation, and that Petitioner fails to provide adequate rationale to support the suggested change in Agrawal's principle of operation. PO Resp. 24–29.

5. Discussion

Based on the fully developed trial record, we do not agree with Patent Owner's arguments. At the outset, we agree with Petitioner that Patent Owner's arguments are based improperly on its "notion that Agrawal—i.e., the primary reference in both [asserted grounds]—must somehow be 'modified' to implement the instantaneous parallelism techniques that the Petition cites and relies upon." Pet. Reply 3. As we stated in the Decision on Intuition (Dec. on Inst. 23), we do not understand the Petition to suggest modifying Agrawal to use instantaneous parallelism, but rather, as relying on Agrawal's disclosure of a two-level scheduler that provides feedback to a job scheduler in the form of instantaneous parallelism. *See* Pet. 29–30 (citing Ex. 1004, 2 ("Agrawal further explains that the 'instantaneous parallelism' feedback provided to the job scheduler from each job's respective task scheduler indicates 'the number of processors the job can effectively use at the current moment.'")). We credit the declaration testimony of Dr. Chase who testifies that "it was not necessary to describe a modification to Agrawal's system to use instantaneous parallelism feedback because it was already part of certain systems described in Agrawal." Ex. 1032 ¶ 14. More particularly, Dr. Chase states that he "expressly

addressed in [his] First Declaration how Agrawal itself discloses instantaneous parallelism feedback." *Id.*

We acknowledge that much of Agrawal's disclosure focuses on improvements to the type of feedback used by a job scheduler, i.e., A-GREEDY, but nonetheless find that Agrawal teaches that it was well-known in the art at the time of the invention to use instantaneous parallelism in a two-level scheduler. *See* Pet. 15 (citing Ex. 1004, 2). For example, Agrawal identifies that "[v]arious researchers . . . ha[d] used the notion of instantaneous parallelism,[] the number of processors the job can effectively use at the current moment, as the parallelism feedback to the job scheduler." Ex. 1004, 2. Agrawal also identifies that related work on adaptive task scheduling demonstrates that "instantaneous parallelism feedback is 4-competitive with respect to makespan for batched jobs with multiple phases, where the parallelism of the job remains constant during the phase and the phases are relatively long compared with the length of a scheduling quantum." *Id.* at 8. Based on these disclosures, Dr. Chase testifies—and we agree—that "Agrawal repeatedly identifies instantaneous parallelism feedback as a suitable solution for some systems—particularly systems that run what I referred to as 'long-task jobs.'" Ex. 1032 ¶ 14 (citing Ex. 1003 ¶¶ 56–57).

Although Agrawal and Sen describe some drawbacks to using instantaneous parallelism, we credit Dr. Chase's declaration testimony, which is consistent with the disclosure of Agrawal itself, that the use of instantaneous parallelism is still a workable alternative to the historical-based feedback used in Agrawal's A-GREEDY algorithm. For example, Dr. Chase testifies that

> Agrawal undoubtedly advocates for A-GREEDY and describes some comparative advantages of A-GREEDY over instantaneous parallelism for some use cases, but as described further below, Agrawal never suggests that A-GREEDY always outperforms instantaneous parallelism, that A-GREEDY is the only acceptable solution for all systems, or that instantaneous parallelism would not have been known and obvious for [one of ordinary skill in the art] to use long before the '833 [p]atent.

Ex. 1032 ¶ 14; *see, e.g.*, Ex. 1004, 3 ("[E]ven without up-to-date information on instantaneous parallelism, A-GREEDY operates nearly as efficiently."), 7 ("If the task scheduler can communicate with the job scheduler before every vector operation, then using instantaneous parallelism as feedback works fine."). In fact, Agrawal concludes, in light of Sen, that "it is not apparent which strategy is superior." Ex. 1004, 8 ("These studies use different strategies for parallelism feedback, and all report better system performance with parallelism feedback than without, but it is not apparent which strategy is superior."), 7 ("A-GREEDY-like adaptive strategy for parallelism feedback *should* outperform a strategy based on instantaneous parallelism." (emphasis added)). Thus, Agrawal supports Dr. Chase's testimony.

Based on the evidence of record, we are not persuaded by Patent Owner's contention that "Petitioner proposed a modification to Agrawal or suggested a 'design choice' based on Agrawal." PO Sur-reply 22. Instead, as we explain above, we credit the declaration testimony of Dr. Chase who testifies that "it was not necessary to describe a modification to Agrawal's system to use instantaneous parallelism feedback because it was already part of certain systems described in Agrawal." Ex. 1032 ¶ 14. *See In re Fritch*, 972 F.2d 1260, 1264 (Fed. Cir. 1992) ("It is well settled that a prior art reference is relevant for all that it teaches to those of ordinary skill in the

art."); *see also Merck & Co. Inc. v. Biocraft Labs., Inc.*, 874 F.2d 804, 807 (Fed. Cir. 1989) (holding that all the prior art's disclosures must be considered in a § 103(a) inquiry).

We agree with Petitioner that "[n]othing in Agrawal obligates [one of ordinary skill in the art] to start from Agrawal's other embodiments that employ A-GREEDY or requires [one of ordinary skill in the art] to modify these embodiments to use instantaneous parallelism either in lieu of or together with A-GREEDY." Pet. Reply 7. Here, Patent Owner's teaching away arguments are not directed to whether it would have been obvious to one of ordinary skill in the art to combine Agrawal and Brent, but rather, directed to the teachings of Agrawal itself. *See* Pet. Reply 2–3 ("[Patent Owner] does not appear to dispute that the combinations based on Agrawal, Brent, and Feitelson disclose and render obvious each limitation of the Challenged Claims."). Thus, we find Patent Owner's teaching away arguments to be an improper individual attack on Agrawal, not the combined teachings of Agrawal and Brent, which, by itself, does not rebut the obviousness of Petitioner's proposed combination. *Bradium Techs. LLC v. Iancu*, 923 F.3d 1032, 1050 (Fed. Cir. 2019). ("A finding of obviousness, however, cannot be overcome 'by attacking references individually where the rejection is based upon the teachings of a combination of references.'").

We also find Patent Owner's arguments based on *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 567 F.3d 1314 (2009) (PO Resp. 9) and *Polaris Indus. Inc. v. Artic Cat Inc.*, 882 F.3d 1056 (Fed. Cir. 2008) (PO Resp. 17 n.3) to be unavailing. *DePuy Spine* and *Polaris Industries* both involved an actual modification of a primary reference in a combination that proposed to integrate a feature from a secondary reference into the primary

reference that would have "deterred" a person of ordinary skill in the art from combining references in the manner proposed. *DePuy Spine*, 567 F.3d at 1328. We agree with Petitioner that "[n]either of these cases is remotely analogous to the case at hand where the combinations proposed in the Petition include features expressly found in the base reference and where those features are used as the starting point for the combinations." Pet. Reply 9.[12] Patent Owner neither provides any argument against the

[12] In the present case, Petitioner's proposed combination is premised primarily on its acknowledgment that "Agrawal does not prescribe any particular type(s) of processors that must be employed to implement its scheduling solutions." Pet. 20 (citing Ex. 1003 ¶¶ 63–70); *see also id.* at 31 ("It would have been obvious to apply Brent's teachings for indicating demanded processor core types in combination with Agrawal as described above."). And, to address this deficiency, Petitioner contends that one of ordinary skill in the art would have looked to Brent's disclosure of "an array of processing cores 170 configured to support parallel execution of multiple tasks of a software application 115." *Id.* at 20 (citing Ex. 1003 ¶ 63; Ex. 1005 ¶¶ 15–16). Petitioner provides several persuasive reasons why one of ordinary skill in the art would have been motivated to combine Agrawal and Brent, with a reasonable expectation of success. *See* Pet. 19–24 (citing Ex. 1003 ¶¶ 63–70). For example, Petitioner sufficiently sets forth that Agrawal and Brent are both directed to systems for scheduling tasks in a parallel processing environment (Pet. 21–22 (citing Ex. 1003 ¶ 66; Ex. 1004, 1–3; Ex. 1005 ¶¶ 13, 20, 26)) and how it would have been obvious to apply Brent's disclosure of an array of heterogeneous processor cores "with Agrawal to achieve predictable benefits, including an improved ability to optimize task execution on a processor whose instruction set matches the instruction types provided in a task." *Id.* at 22 (citing Ex. 1003 ¶¶ 67; Ex. 1005 ¶¶ 16, 18, 19, Abstract). Petitioner contends, and we agree, that matching a given task to a core having its demanded core type may result in better efficiency "thereby reducing the tasks's time to completion and ordinarily allowing the task to run on a more efficient processor." *Id.* (citing Ex. 1003 ¶ 67). And, given Agrawal and Brent both describe scheduling tasks in a parallel processing environment (Ex. 1004, 1–3; Ex. 1005 ¶¶ 13,

combined teachings of Agrawal and Brent, nor explains why the combination would have been beyond the level of ordinary skill, nor disputes that one with ordinary skill in the art would have had a reasonable expectation of success in combining the teachings.

We also do not agree with Patent Owner's argument that one of ordinary skill in the art "would not be motivated to architect a system based on instantaneous parallelism feedback to practice the claimed inventions because such a system would result in gross misallocation of processor resources." PO Resp. 7 (emphases omitted). Patent Owner asserts

> Agrawal's "special case" is the case of "unit quanta," which are quanta that consist of "only a single time step." [Ex. 1004, 3]. Dr. Smith explains that a job's parallelism will not change during a quantum with only a single time step. Ex. 2003, ¶ 46. In that situation, there will be no fluctuation of processor usage during the scheduling period, which mitigates the risk of inaccurate sampling. *Id.* But where a job's parallelism fluctuates during a quantum, Agrawal is clear that instantaneous parallelism "can cause gross misallocation of processor resources." Ex. 1004 at 2; Ex. 2003, ¶54.

PO Resp. 13. And, relying on its declarant, Patent Owner asserts that "Dr. Smith explains that [one of ordinary skill in the art] would read claim 15 as a whole and recognize that the job's parallelism and processor usage would fluctuate during the claimed core allocation periods." *Id.* at 22

20, 26), we agree with and credit Dr. Chase's declaration testimony that it would have been obvious and well within the skill of one of ordinary skill in the art to augment Agrawal's disclosure of existing parallelism feedback mechanisms, e.g., instantaneous parallelism, "with an indication of how many cores of each type the job desires (especially since Brent already confirms this information was available to the task scheduler)." Pet. 23 (citing Ex. 1003 ¶ 68).

(citing Ex. 2003 ¶ 57). Consequently, Patent Owner argues that one of ordinary skill in the art would have been "actively discouraged" from relying on "instantaneous parallelism path in the context of the inventions claimed in the '833 patent." *Id.* (citing Ex. 2003 ¶ 57).

In response, Petitioner asserts that, "[e]ven if A-GREEDY were the only legitimate starting point for reliance on Agrawal as urged by [Patent Owner]" (Pet. Reply 10), one of ordinary skill in the art would have been motivated to use instantaneous parallelism as a feedback mechanism based on the teachings of Agrawal. Pet. Reply 10–17. Relying on the declaration testimony of Dr. Chase, Petitioner contends that one of ordinary skill in the "would have recognized the existence of tradeoffs implicated by the choice of A-GREEDY or instantaneous parallelism." Pet. Reply 11 (citing Ex. 1032 ¶ 16). Petitioner asserts

> Agrawal explains that A-GREEDY may achieve better processor utilization efficiency than instantaneous parallelism when the quantum length is long (L >>1) and when the system executes certain jobs whose parallelism can change sharply over time (e.g., by alternating between serial and parallel phases). [Ex. 1004], 2, 7. It is only in this limited context that Agrawal notes how instantaneous parallelism may yield a significant "misallocation of processor resources." *Id.*, 2.

Id. Petitioner points out that Dr. Smith acknowledges that one of ordinary skill in the art would "understand that the claimed system may from time to time schedule a unit quantum" (*Id.* at 10 (citing PO Resp. 22 n.4)), and argues that

> [f]or at least these unit quanta, it is undisputed that instantaneous parallelism provides the most efficient parallelism feedback to the job scheduler—even more so than A-GREEDY—and hence [one of ordinary skill in the art] would have been motivated to

> use instantaneous parallelism to increase processing efficiency when scheduling in unit quanta."

Id.

Based on the fully developed trial record, we do not agree with Patent Owner's arguments. Instead, we agree with Petitioner that "A-GREEDY and instantaneous parallelism provide alternative solutions for the task scheduler to express its desire d_q to the job scheduler." Pet. Reply 12 (citing Ex. 1004, 3–4). Although Agrawal's disclosure focuses primarily on the benefits of A-GREEDY, we credit Dr. Chase's declaration testimony stating that one that of ordinary skill in the art would have understood that "Agrawal itself discloses instantaneous parallelism feedback" (Ex. 1032 ¶ 14), that Agrawal's "proposal for A-GREEDY is not the only suitable solution for implementing parallelism feedback in Agrawal's scheduling framework," and that "A-GREEDY [would not] always be the most optimal solution." *Id.* ¶ 16.

Patent Owner responds that "[t]he differences identified by Petitioner demonstrate that there is no 'tradeoff' between two equally viable design options. Instead, this is a situation where one option is plainly inferior to the other on the most important issues to [one of ordinary skill in the art]." PO Sur-reply 15. Patent Owner argues that there would be no "tradeoff" in efficiency or overhead. *Id.* at 15–18. According to Patent Owner, "Sen empirically demonstrates that instantaneous parallelism will result in processor misallocation" (*id.*), and "truncating core allocation periods so that they are short enough to ensure long-task jobs will be scheduled on the system" would result in increased overhead. *Id.* at 16. In addition, Patent

Owner contends that "Agrawal explains that A-GREEDY outperforms instantaneous parallelism in every single real world use case." *Id.* at 9.

However, just because better alternatives exist in the prior art does not mean that an inferior combination is inapt for obviousness purposes. *In re Mouttet*, 686 F.3d 1322, 1333 (Fed. Cir. 2012) (citing *In re Gurley*, 27 F.3d 551, 553 (Fed. Cir. 1994)). Here, Dr. Chase opines that

> [t]hese tradeoffs would have led [one of ordinary skill in the art] to select one scheme or the other (or both) for use in a given system depending on applicable processing demands and design constraints, and the competing advantages and disadvantages between A-GREEDY and instantaneous parallelism certainly would not have precluded use of the latter under appropriate conditions.

Ex. 1032 ¶ 16. We find Dr. Chase's opinion credible and supported by the record. For example, Agrawal teaches that it was well-known in the art at the time of the invention to use instantaneous parallelism in a two-level scheduler. Ex. 1004, 2. In addition, Dr. Chase points out that, for certain scenarios, "Agrawal specifically acknowledges that instantaneous parallelism feedback yields no wasted processor cycles for unit quanta, and explains that '[i]f the task scheduler can communicate with the job scheduler before every vector operation, then using instantaneous parallelism feedback works fine.'" Ex. 1032 ¶ 21 (citing Ex. 1004, 3). In contrast, we find Dr. Smith's supporting declaration testimony that one of ordinary skill in the art would have been "actively discouraged" (*see* Ex. 2003 ¶ 57) to be predicated on the notion that Petitioner's asserted grounds involved modifying Agrawal's system to implement instantaneous parallelism feedback as a substitute for parallelism feedback based on A-GREEDY. However, as we explain above and below, Petitioner has persuasively shown

that Agrawal discloses that instantaneous parallelism as a workable alternative, with certain tradeoffs, when compared against Agrawal's A-GREEDY algorithm. Therefore, we find the record demonstrates persuasively that one of ordinary skill in the art would have been motivated to select instantaneous parallelism as the feedback mechanism to a job scheduler under certain operating conditions. Ex. 1004, 1–2; Ex. 1032 ¶¶ 14–15.

In making this determination, we see nothing in the claims that would prevent one of ordinary skill in the art from relying on Agrawal's disclosure of instantaneous parallelism based on "unit quanta" or CAPs of a "single time step." Patent Owner argues that "the claims recite 'core allocation periods' that, in operation, would include multiple time steps." PO Resp. 9; *see also id.* at 17 (citing Ex. 2003 ¶¶ 52–56) ("the challenged claims recite 'core allocation periods' during which processor usage will fluctuate."). To support its argument, Patent Owner contends that "the claims generally recite: 1) hosting multiple programs on a computer processing system; 2) the multiple programs including tasks; and 3) the tasks demanding different processor core types." PO Resp. 19. According to Patent Owner

> an "aspect of the invention involves, in a processing task load and type adaptive processing core array, a method of executing ***time variable*** sets of information processing tasks of differing types on their assigned cores of matching types, with a matching type of a core for a given task referring to processing core deemed optimal for executing the given task."

Id. at 19 (citing Ex. 1001, 4:24–33). And, relying on the declaration testimony of Dr. Smith, Petitioner contends that "[t]ime variable sets of processing tasks will involve multiple time steps and would not be executed during unit quanta." *Id.* at 19–20 (citing Ex. 2003 ¶ 26). Thus, Patent

Owner concludes that one of ordinary skill in the art "would recognize that a job's parallelism and processor usage would fluctuate during core allocation periods executing time variable sets of information processing tasks." *Id.* at 20 (citing Ex. 2003 ¶ 26); *see also* PO Sur-reply 19 ("[T]he claims are directed to a situation in which there is variation in a job's parallelism during a quantum.").

Petitioner responds that

> [n]either the claims nor the [S]pecification once mentions the concept of defining CAP lengths by time steps, let alone restricting CAPs to multiple time steps. The [S]pecification broadly describes CAPs in the context of a "core assignment process 300 … [that] is exercised periodically, e.g.[,] at intervals such as once per a defined number (for instance 64, 256 or 1024, or so forth) of processing core clock or instruction cycles." [Ex. 1001], 9:33–38. But a time step can encompass multiple core clock or instruction cycles, and thus a time step defined as 64, 256, or 1024 core clock or instruction cycles would conceivably result in CAPs consisting of a unit quanta in the '833 [p]atent.

Pet. Reply 18 (citing Ex. 1032 ¶ 9; Ex. 1033, 11:17–20). In addition, Petitioner argues that the asserted grounds do not need to rely on "unit quanta to render the claims of the '833 [p]atent obvious. Pet. Reply 17. According to Petitioner, "[t]he Petition and Dr. Chase's initial declaration both explained that the proposed combinations are applicable for any 'long-task jobs,' which can occur in the context of both short ***and long quanta*** comprised of multiple time steps." *Id.* (citing Ex. 1003 ¶¶ 56–57, 79, 110, 146; Pet. 16, 29–30, 45).

Based on the fully developed trial record, we do not agree with Patent Owner that the claims recite 'core allocation periods' that, in operation,

would include multiple time steps." *See* PO Resp. 9. Instead, we agree with Petitioner, relying on the declaration testimony of Dr. Chase that

> [t]he plain language of the claims say nothing about requiring processor usage to fluctuate within a given CAP. Such fluctuations can instead occur over time steps that span multiple CAPs, especially since the '833 [p]atent is clear that tasks commonly execute across multiple CAPs (and tasks need not be fully and completely executed in the course of a single CAP).

Pet. Reply. 18 (citing Ex. 1032 ¶ 10). We find Dr. Chase's opinion to be credible and supported by the evidence. For example, the '833 patent discloses "[t]asks whose execution carries across successive CAPs are referred to as 'continuing' tasks (or app-task-instances more precisely)." *Id.* (citing Ex. 1001, 15:1–5). Dr. Smith opines that one of ordinary skill in the art would understand "processor usage will fluctuate as its tasks are executed during an allocation period." Ex. 2003 ¶ 51. However, we find the '833 patent does not require a task to complete execution within a single CAP, but rather supports Petitioner's position that "tasks that are executed across multiple time steps will be executed across one or more CAPs depending on the length of the CAP." Pet. Reply 19 (citing Ex. 1032 ¶ 11). Therefore, we agree with Petitioner that there is nothing in the either the plain language of the challenged claims or the '833 patent's Specification that would preclude one of ordinary skill in the art from relying on Agrawal's disclosure of instantaneous parallelism when designing a two-level scheduling system.

We also do not agree with Patent Owner's arguments that one of ordinary skill in the art would not rely on Agrawal's disclosure of feedback using instantaneous parallelism because "doing so would increase scheduling overhead." PO Resp. 24–25 (citing Ex. 2003 ¶¶ 58–59); *see also* PO Sur-reply 16 ("Petitioner's proposal of truncating core allocation periods

so that they are short enough to ensure long-task jobs will be scheduled on the system, will result in more overhead than a system with more realistically bounded core allocation periods."). To support its argument, Patent Owner relies on the declaration testimony of Dr. Smith who "explains that [one of ordinary skill in the art] would seek as little overhead as possible while noting that Agrawal's A-GREEDY system is more efficient in scheduling in the vast majority of use cases and without the overhead required to avoid the inherent inaccuracy involved in instantaneous parallelism." PO Resp. 25 (citing Ex. 2003 ¶¶ 58–59).

Petitioner responds by arguing that

> [t]he selection of either A-GREEDY or instantaneous parallelism for use in a particular design would implicate predictable differences in overhead costs and processor efficiency—i.e., factors that are impacted differently in each scheme based on the quantum (CAP) length and characteristics of the tasks to be executed by a given job

Pet. Reply 14 (citing Ex. 1032 ¶¶ 17–19). Petitioner points out that Dr. Smith agrees with Dr. Chase that there would be predictable differences or tradeoffs implicated by the use of A-GREEDY or instantaneous parallelism. Pet. Reply 14 n.3 (citing Ex. 1033, 67:18–68:24, 91:1–17, 94:14–95:5, 96:1–97:9). For example, when asked whether "a designer has to make a trade-off between how often there's going to be core allocation versus how much time there's going to be spent processing," Dr. Smith answered that it was "a fair statement." Ex. 1033, 67:18–22; *see also id.* at 68:9–12 (Dr. Smith agreed that "it is definitely a balancing between what -- how efficient is my execution during that time period versus should I make changes to make it more efficient."). Dr. Smith also acknowledged that "it may be appropriate to have short core allocation periods or relatively longer

core allocation periods" depending on the different types of tasks, cores, and length of programs. *Id.* at 68:13–22. And, when asked whether "A-GREEDY takes more overhead to run than instantaneous parallelism takes to run" (*id.* at 112:7–9) in terms of "the amount of processing resources it takes to do the actual instantaneous parallelism feedback or the actual A-GREEDY counting and comparison work" (*id.* at 112:1–3), Dr. Smith answered that A-GREEDY required more overhead, "but not significantly more" (*id.* at 112:10–11).

We find Petitioner's arguments and evidence persuasive for the reasons set forth by Petitioner. In particular, a given course of action may have simultaneous advantages and disadvantages. *See Medichem, S.A. v. Rolabo, S.L.*, 437 F.3d 1157, 1165 (Fed. Cir. 2006) (stating that "a given course of action often has simultaneous advantages and disadvantages, and this does not necessarily obviate motivation to combine"); *see also Winner Int'l Royalty Corp. v. Wang*, 202 F.3d 1340, 1349 n.8 (Fed. Cir. 2000) ("The fact that the motivating benefit comes at the expense of another benefit . . . should not nullify its use as a basis to modify the disclosure of one reference with the teachings of another. Instead, the benefits, both lost and gained, should be weighed against one another."). Here, we credit Dr. Chase's declaration testimony that identifies the competing advantages and disadvantages between A-GREEDY and instantaneous parallelism in the context of Agrawal's two-level scheduling framework. *See* Pet. Reply 11–17 (citing Ex. 1032 ¶¶ 16–21). From this, we find that one of ordinary skill in the art would have had reason to, and as such, been motivated to design a two-level scheduling system based on Agrawal's disclosure of instantaneous parallelism in light of the evidence provided by the parties.

Even if a system based on instantaneous parallelism feedback would result in some misallocation of processor resources, as Patent Owner contends, we find that one of ordinary skill in the art would have understood the advantages of utilizing a task scheduler that provides feedback in the form of instantaneous parallelism under certain operating conditions.

We also are not persuaded by Patent Owner's argument that "the system suggested in the Petition would fundamentally change Agrawal's principle of operation by implementing the very instantaneous parallelism feedback Agrawal set out to avoid." PO Resp. 26. More particularly, Patent Owner argues that "[t]he entire point of Agrawal is not to use instantaneous parallelism, yet somehow, modifying Agrawal to do exactly that is the trial ground." *Id.* at 25. According to Patent Owner, the only way one of ordinary skill in the art "could have hoped to avoid gross misallocation of processors resources while implementing instantaneous parallelism feedback would be to have near continuous communication between the job scheduler and the task scheduler." *Id.* (citing Ex. 2003 ¶¶ 58–60).

Based on the fully developed trial record, we do not agree with Patent Owner's contention that providing feedback to Agrawal's job scheduler in the form of "instantaneous parallelism" as opposed to "A-GREEDY" would fundamentally change Agrawal's principle of operation or otherwise render it inoperable. Instead, we agree with Petitioner that Agrawal's principle of operation could not "be violated or its intended purpose negated by the use of instantaneous parallelism" because, as we explain above, "this feature is already present in the original system." Pet. Reply 1. Patent Owner argues

> [i]f A-GREEDY were operated based on instantaneous parallelism feedback, A-GREEDY's three-way classification would be rendered superfluous. Ex. 2003, ¶¶60–61. Indeed,

> there would no longer be a need for A-GREEDY's desire estimation, rendering the three-way classification inoperable and highlighting the fundamental change in principle of operation that would be required for Agrawal to effectively use instantaneous parallelism feedback.

PO Resp. 28–29 (citing Ex. 2003 ¶¶ 60–61). However, Patent Owner's and Dr. Smith's position discounts that it was well-known in the art at the time of the invention to use instantaneous parallelism in a two-level scheduler. *See* Pet. 15 (citing Ex. 1004, 2). Instead, we are persuaded by the declaration testimony of Dr. Chase who states that "it was not necessary to describe a modification to Agrawal's system to use instantaneous parallelism feedback because it was already part of certain systems described in Agrawal." Ex. 1032 ¶ 14. We also credit the declaration testimony of Dr. Chase who opines that "[a]ccording to Agrawal, A-GREEDY and instantaneous parallelism provide alternative or otherwise different solutions for the task scheduler to express its desire d_q to the job scheduler. Ex. 1032 ¶ 18 (citing Ex. 1004, 3–4).

We find Dr. Chase's opinion credible and supported by the record. For example, Agrawal discloses that "[i]nstead of using instantaneous parallelism, A-GREEDY provides parallelism feedback to the job scheduler based on a single summary statistic and the job's behavior on the previous quantum." Ex. 1004, 2. We acknowledge that Agrawal presents a three-way classification system for its A-GREEDY's desire estimation, but also find that that Agrawal discloses determining a job's desire using instantaneous parallelism, i.e., "the number of processors the job can effectively use at the current moment." *Id.*

Based on the complete record, we find Petitioner has persuasively shown that one of ordinary skill in the art would have been motivated to design a two-level scheduler that provides feedback to a job scheduler based on Agrawal's disclosure of instantaneous parallelism. *See* Ex. 1003 ¶ 56 (Dr. Chase testifying that one of ordinary skill in the art "[would have recognized 'instantaneous parallelism' feedback to be an especially useful form of parallelism feedback for jobs whose task durations are long relative to the length of a quantum."); *see also* Ex. 1032 ¶ 21 ("The use of instantaneous parallelism feedback over A-GREEDY therefore would have been obvious in many cases, particularly where the designer prefers a simple feedback solution and where the system will be used to run long-task jobs in which the quantum length is short relative to task length."). "It's not necessary to show that a combination is 'the *best* option, only that it be a *suitable* option." *Intel Corp. v. Qualcomm Inc.*, 21 F.4th 784, 800 (Fed. Cir. 2021) (emphasis in original); *see also In re Fulton*, 391 F.3d 1195, 1200 (Fed. Cir. 2004) ("[A] finding that the prior art as a whole suggests the desirability of a particular combination need not be supported by a finding that the prior art suggests that the combination claimed by the patent applicant is the preferred, or most desirable, combination.").

a) Conclusion for Claim 15

Petitioner has persuasively shown that the combination of Agrawal and Brent teaches all limitations of claim 15. Petitioner also has put forth persuasive reasons for combining these references. *See* Pet. 19–24 (citing Ex. 1003 ¶¶ 65–70). Thus, regarding claim 15, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention

as a whole would have been obvious over the combination of Agrawal and Brent.

6. Claim 34

Petitioner provides an element-by-element analysis of where each element in the challenged claims is disclosed in, or would have been obvious in view of, the combination of Agrawal and Brent. *See* Pet. 38 (mapping limitations 34.P, 34.1, 34.2, 34.3, and 34.4 to limitations 15.P, 15.1, 15.2, 15.3, and 15.4, respectively). Petitioner also relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 92–97).

Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with respect to limitations 34.P, 34.1, 34.2, 34.3, and 34.4. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that the combination of Agrawal and Brent discloses these limitations. *See supra* §§ II.D.3–5.

Petitioner also has put forth persuasive reasons for combining these references with a reasonable expectation of success. *See* Pet. 19–24 (citing Ex. 1003 ¶¶ 63–70). We credit Dr. Chase's opinion that one of ordinary skill in the art "would have understood two-level scheduling techniques like Agrawal's to be fully compatible with implementation on a heterogeneous array of cores like Brent's." *Id.* at 23 (citing Ex. 1003 ¶ 68). We also credit Petitioner's explanation, relying on Dr. Chase's declaration testimony, that Agrawal "describes conventional feedback channels by which a task scheduler provides feedback to a job scheduler about the desired core allocation for a job, which Agrawal's original job scheduler already uses to inform allocation decisions" (*id.* at 23 (citing Ex. 1004, 1–2), and

> [g]iven these existing feedback capabilities, it would have been obvious and well within the skill of [one of ordinary skill in the art] to augment this parallelism feedback with an indication of how many cores of each type the job desires (especially since Brent already confirms this information was available to the task scheduler).

Id. (citing Ex. 1003 ¶ 68). Thus, regarding claim 34, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Agrawal and Brent.

7. Dependent Claims 17, 18, and 37

Petitioner contends that the additional limitations of dependent claims 17, 18, and 37 are rendered obvious by the combination of Agrawal and Brent. Pet. 37–38. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003 ¶¶ 88–91, 98, 99).

i. [17] The method of claim 15, wherein one of the executable instances of a given one of the programs corresponds to a task, a process, an actor, a thread, a procedure or a function of the given program.

Petitioner asserts that the combination of Agrawal and Brent renders obvious claim 17. Pet. 37. More particularly, Petitioner asserts that the combination of Agrawal and Brent "discloses that one of the executable instances of a given of the jobs (program) corresponds to a "task" of the given job." *Id.* (citing Ex. 1004, 1; Ex. 1005 ¶ 15, Figs. 1–2; Ex. 1003 ¶¶ 88–89; Pet. 19–24).

ii. [18] The method of claim 15, wherein said indication of the given program is further based on a measure of input data that said one program presently has available for processing.

Petitioner asserts that the combination of Agrawal and Brent renders obvious claim 18. Pet. 37. More particularly, Petitioner asserts that, as discussed above with respect to limitation 15.2, "said capacity demand indication of a given job (***program***) is based at least in part on a number of its tasks (***executable instances***) that presently have input data available for processing." *Id.* (citing Ex. 1003 ¶¶ 90–91). Petitioner further asserts that Agrawal discloses that "an increase in the number of tasks that have input data available for processing will increase the job's number of ready-to-execute tasks, and thus increase the job's reported 'desire' in its parallelism feedback (e.g., capacity demand indication)" and "[a]s additional input data becomes available for additional tasks, the parallelism feedback for the job increases to reflect the additional number of ready tasks *Id.* (citing Ex. 1004, 1–2; Pet. 13–17). Relying on the declaration testimony of Dr. Chase, Petitioner explains that "[t]he capacity demand indication is therefore based on a measure of input data that the program presently has available for processing." *Id.* (citing Ex. 1003 ¶ 91).

iii. [37] The system of claim 34, wherein said indication of the given program is further based on a measure of input data that said one program presently has available for processing.

Petitioner asserts that the combination of Agrawal and Brent renders obvious claim 37 for the same reasons discussed above with respect to claim 18. Pet. 38 (citing *id.* at 37; Ex. 1003 ¶¶ 98–99).

iv. Conclusion for Dependent Claims 17, 18, and 37

Patent Owner does not challenge Petitioner's contentions with respect to the limitations of dependent claims 17, 18, and 37, and Petitioner's contentions are supported by the cited evidence. For the reasons given by Petitioner and summarized above, we are persuaded that Petitioner has shown by a preponderance of the evidence that the limitations of each of claims 17, 18, and 37 are rendered obvious by the combination of Agrawal and Brent. Therefore, we determine that Petitioner has shown, by a preponderance of the evidence, that the additional limitations of claims 17, 18, and 37 would have been obvious over the combination of Agrawal and Brent.

8. Conclusion for Ground 1A

For the foregoing reasons, we conclude that Petitioner has shown by a preponderance of the evidence that claims 15, 17, 18, 34, 37 are unpatentable under § 103(a) over Agrawal and Brent.

E. Obviousness over Agrawal, Brent, and Feitelson (Ground 1B)

Petitioner asserts that claims 10, 13, 14, 19–22, 24, 25, 27, 29, 32, and 33 would have been unpatentable as obvious over Agrawal, Brent, and Feitelson. Pet. 38–73. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003). Patent Owner responds to Petitioner's assertions regarding claim 19. *See* PO Resp. 29–33; PO Sur-reply 23–25.

1. Overview of Feitelson (Ex. 1006)

Feitelson is titled "Job Scheduling in Multiprogrammed Parallel Systems." Ex. 1006, i.[13] Feitelson is an IBM Research Report and concerns scheduling on parallel computer systems shared by a number of users. *Id.*, Abstr. Feitelson "describe[s] the many different approaches within a unified framework based on the mechanisms used to achieve multiprogramming," such as single-level and two-level scheduling, time slicing, space slicing, and hybrid approaches involving both. *Id.* In Feitelson, which is primarily concerned with "interactive use," "the full range of schemes developed for multiprogramming on parallel machines are reviewed. This includes both paper designs from academia and practical ones that are commercially available." *Id.* at 3. Feitelson does not discuss scheduling involving interdependencies between processors, scheduling with real-time constraints, or scheduling on "heterogeneous networks" of different computer hardware types. *Id.* at 3–4.

2. Claim 10

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 10 as set forth in the Petition. Pet. 41–58 (citing Exs. 1003, 1004, 1005, 1006). We use Petitioner's notations to identify the claim limitations.

[13] We refer to the page numbers inserted by Petitioner at the bottom, middle portion of each page in Exhibit 1006.

i. [10.P] A method for controlling execution of a set of software programs on an array of processor cores, with a given core among the array referred to as a core slot, the method comprising:[14]

Petitioner asserts that, if the preamble is treated as limiting, the combination of Agrawal, Brent, and Feitelson discloses its contents. Pet. 41–43 (citing Ex. 1003 ¶¶ 105–106; Ex. 1004, 1–2; Ex. 1005 ¶¶ 14, 16–17, Fig. 1; Ex. 1006, 8; Ex. 1008, 261–262). Petitioner asserts that "Agrawal discloses a method for controlling execution of a set of jobs (***software programs***) on a set of processors." *Id.* at 41 (citing Ex. 1004, 1–2; Pet. 13–17). Petitioner acknowledges that Agrawal does not disclose using any particular type of processors or describe its processors as an "array of processor cores," but argues that such arrays were known in the art at the time of the invention, as evidenced by Brent. *Id.* at 42 (citing Ex. 1005 ¶¶ 14, 16–17, Fig. 1; Ex. 1003 ¶ 106). According to Petitioner, it would have been obvious to implement Agrawal's "two-level scheduling" using Brent's "array of processor cores" "to achieve predictable benefits associated with core arrays." *Id.* at 43 (citing Ex. 1003 ¶ 106).

ii. [10.1] monitoring capacity demand indicators of one or more programs among said set of programs, with said indicator of a given one of the programs expressing a number of instances that the given program has available for execution for a succeeding Core Allocation Period (CAP);

Petitioner asserts that "Agrawal discloses, and the combination provides, a 'job scheduler' that monitors 'parallelism feedback' (***capacity demand indicators***) received from the respective task schedulers of each

[14] The issue of whether the preamble is limiting need not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

managed job (***one or more programs among said set of programs***)." Pet. 43 (citing Ex. 1003 ¶¶ 107–110; Ex. 1004, 1–2). Petitioner explains that Agarwal's adaptive scheduling strategy utilizes a task scheduler that provides "***parallelism feedback***" to a "job scheduler so that when a job cannot use many processors, those processors can be reallotted to jobs with ample need." *Id.* at 43–44 (citing Ex. 1004, 2). Petitioner further explains that Agarwal's "'parallelism feedback' for a job indicates the 'job's ***desire*** d_q, which is the number of processors the job wants for quantum q." *Id.* at 44 (citing Ex. 1004, 2). According to Petitioner, in Agarwal,

> one method for computing the job's "desire" is to employ "instantaneous parallelism," which indicates "the number of processors the job can effectively use at the current moment[] . . . ." [Ex. 1004, 2]. As jobs seek to execute all of their "ready" tasks, the number of processors that a job would desire and effectively use at a given time relates to the number of processors (cores) that the job would use to execute its "ready" tasks.

Id. (citing Ex. 1004, 1–2).

Relying on the declaration testimony of Dr. Chase, Petitioner asserts that "Agrawal and Brent both teach scheduling concepts that assign just one task to a given core at a time," and as such, "Agrawal's 'instantaneous parallelism' feedback provides a '***capacity demand indicator***' for a job (***program***) that expresses a number of tasks (***instances***, *see supra*, [Pet. 11–13]) that the job (***program***) has available for execution in a succeeding scheduling quantum (***core allocation period***). Pet. 44 (citing Ex. 1003 ¶ 108); *see also* Ex. 1004, 3 ("Between quanta $q - 1$ and q, the task scheduler determines its job's desire … for quantum q.").

iii. [10.2] allocating the array of cores among the set of programs for the succeeding CAP at least in part based on said capacity demand indicators for that CAP, to produce a new allocation of the cores among the programs to replace a present allocation of the core slots among the programs; and

Petitioner asserts

> Agrawal discloses, and the combination provides, a "job scheduler" that allocates processors (e.g., allocates cores in an "array of cores" based on Brent) among a set of "jobs" (***programs***) for a succeeding scheduling quantum (***succeeding CAP***) at least in part based on the "desires" expressed by the jobs' respective task schedulers through parallelism feedback (***capacity demand indicators***) for that quantum.

Pet. 46 (citing Ex. 1004, 1–2; Ex. 1003 ¶ 111; Pet. 13–17, 39–41). More particularly, Petitioner asserts that Agarwal discloses that "[b]etween quantum *q-1* (***present CAP***) and *q* (***succeeding CAP***), the job scheduler produces a new allocation of the cores among the jobs to replace a present allocation of the core slots among the jobs." *Id.* (citing Ex. 1004, 2). Relying on the declaration testimony of Dr. Chase, Petitioner provides an example consistent with the teachings of Agarwal and Brent. *See* Pet. 47 (citing Ex. 1003 ¶ 112). In the example, Petitioner explains that "Job 1 may be allotted additional cores in the new allocation if its 'desire' increased for the succeeding quantum to accommodate additional ready-to-execute tasks," and alternatively, "Job 2 may be allotted fewer cores in the new allocation if its "desire" decreased for the succeeding quantum." *Id.* (citing Ex. 1004, 1–2; Ex. 1003 ¶ 112).

iv. [10.3.i] based at least in part on the allocating, placing instances of the programs to the array of cores through sub-steps of: (i) identifying the following:

Petitioner asserts that "Agrawal discloses that '[e]ach job has its own task scheduler,' and the task scheduler 'schedules the tasks belonging to a given job onto the allotted processors' for a scheduling quantum." Pet. 48 (citing Ex. 1004, 1–2; Pet. 13–17). Petitioner explains that "[s]cheduling tasks involves placing tasks (***instances***) of the various jobs (***programs***) to the respective processors allotted to the job for a given quantum." *Id.* (citing Ex. 1005 ¶¶ 18–23, 29, Fig. 2). According to Petitioner, the combination of Agrawal, Brent, and Feitelson would "schedule tasks for cores (core slots) in an array of processor cores by placing tasks for execution on individual cores." *Id.* at 48–49 (citing Ex. 1003 ¶ 113; Ex. 1005 ¶¶ 18–23, 29). Petitioner further asserts that, "[a]s with Agrawal, the placement of tasks to cores in the resulting combination is 'based at least in part on the allocating.'" *Id.* at 49 (citing Ex. 1003 ¶ 113).

v. [10.3.ii] a) a set of instances of the programs from the new allocation that were not included in the present allocation, with this set referred to as activating program instances;

Petitioner asserts that "[i]n placing tasks to cores, task schedulers in the Agrawal-Brent-Feitelson combination identify a set of activating tasks (***activating program instances***) of the jobs (***programs***) from the new allocation for a succeeding quantum (***succeeding CAP***) that were not included in the present allocation of a preceding quantum (***preceding CAP***)." Pet. 49 (citing Ex. 1003 ¶¶ 114–115). Petitioner identifies that "Brent discloses an identification of activating tasks from 'parallel task list 216' and 'analyz[ing] the state of slave processing cores 270 to determine free cores

to distribute tasks.'" *Id.* (citing Ex. 1005 ¶¶ 15–23, Figs. 2–3). Petitioner notes that "Agrawal likewise identifies tasks for execution via a 'dynamically unfolding' DAG [directed acyclic graph]." *Id.* (citing Ex. 1004, 1). According to Petitioner,

> it would have been obvious to identify activating tasks in the Agrawal-Brent-Feitelson combination to enable the respective task schedulers of each job to assign the activating tasks to the "available cores" for that job, i.e., those cores that have been allotted to the job and that are available based on the deactivation of tasks from a preceding quantum (CAP).

Id. at 49–50 (citing Ex. 1003 ¶ 114; Pet. 50–58).

vi. [10.3.iii] b) a set of instances of the programs from the present allocation that are not in the new allocation, with this set referred to as deactivating program instances; and

Petitioner asserts that similar to identifying activating tasks, as recited in limitation 10.3ii, the combination of Agrawal, Brent, and Feitelson discloses that task schedulers "identify a set of deactivating tasks (***deactivating program instances***) of the jobs (***programs***) from the new allocation for a succeeding quantum (***succeeding CAP***) that were not included in the present allocation of a preceding quantum (***preceding CAP***)." Pet. 50–51 (citing Ex. 1003 ¶ 116; Ex. 1005 ¶¶ 20–23, 27–29, Fig. 2). According to Petitioner, one of ordinary skill in the art

> would have appreciated that identifying deactivating tasks and their corresponding cores would be advantageous to distinguish which allotted cores were available for activating tasks and which allotted cores were not, especially since the resulting system would predictably keep tasks that continue execution across succeeding quanta on their same cores based on the teachings of Feitelson.

Id. at 51 (citing Ex. 1003 ¶ 116; Pet. 53–58).

vii. [10.3.iv] c) a set of core slots among the array that were assigned to the set of deactivating program instances in the present allocation, with this set referred to as available cores; and

Petitioner asserts that "the teachings of Agrawal, Brent, and Feitelson provide for identification of a set of available core slots (***available cores***) in an array that were assigned to a set of deactivating tasks (***deactivating program instances***) in the present allocation for a present scheduling quantum (***CAP***)." Pet. 52 (citing Ex. 1003 ¶ 117; Ex. 1005 ¶¶ 20–22, Figs. 2–3; Pet. 13–19, 39–41, 48–52).

viii. [10.4] (ii) assigning the array of cores among the instances of the programs by placing each of the activating program instances to one of the available cores, while keeping each such program instance, which was included both in the present and the new allocation, assigned for the CAP corresponding to the new allocation to the same core as it was assigned on the CAP corresponding to the present allocation.

Petitioner asserts that "Agrawal discloses that '[e]ach job has its own task scheduler,' and the task scheduler 'schedules the tasks belonging to a given job onto the allotted processors' for a scheduling quantum." Pet. 53 (citing Ex. 1004, 1–2). Petitioner further asserts that, in Agarwal, "[s]cheduling tasks involves assigning tasks (***instances***) of the various jobs (***programs***) to the respective processors allotted to the job for a given quantum." *Id.* (citing Ex. 1004, 1–2; Ex. 1005 ¶¶18–23, 29, Fig. 2). Petitioner adds that Brent also discloses a task scheduler that similarly assigns selected tasks "for execution on different cores for a 'present time frame,'" but also allows tasks to continue executing on their previously assigned cores. *Id.* at 53–54 (citing Ex. 1005 ¶¶ 18–23, 29; Ex. 1003 ¶ 120).

To the extent Agarwal and Brent may not disclose the elements of limitation 10.4, Petitioner contends that Feitelson teaches these elements.

Pet. 54 (citing Ex. 1003 ¶¶ 122–128). Petitioner argues that Feitelson demonstrates that "an important 'consideration in the scheduling of threads . . . is matching threads with the most appropriate PE [processing entity] for them to run on.'" *Id.* (citing Ex. 1006, 59). In this regard, Feitelson describes an "affinity scheduling" policy that "tries to schedule threads on the same PE on which they ran most recently, under the assumption that this particular PE might still have some relevant data in its cache." *Id.* According to Petitioner, one of ordinary skill in the art

> would have found it obvious to apply Feitelson's suggestion for keeping threads (e.g., tasks) executing in successive scheduling quanta on their same processors (e.g., cores) across the successive quanta (thereby "increas[ing] the effective scheduling quantum" for the tasks), and predictable reasons would have motivated the combination.

Id. at 55 (citing Ex. 1003 ¶ 112). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that, in contrast, one of ordinary skill in the art

> would have understood that switching threads or tasks from one processor to another ordinarily requires overhead and additional clock cycles to interrupt ongoing execution of the thread, store data that characterizes the thread's current execution context, and transfer the instructions, data, and other necessary information for executing the thread to the new processor.

Id. at 56 (citing Ex. 1003 ¶ 123; Ex. 1006, 8). And, to avoid unnecessary switching costs, Petitioner contends that one of ordinary skill in the art

> would have sought to implement the Agrawal-Brent-Feitelson combination to keep tasks that are included both in the present and new allocations of successive scheduling quanta (CAPs) assigned to their same respective cores. [Ex. 1003 ¶ 124; Ex. 1006 ¶¶ 59–60]. Likewise, it would have been obvious to place each of the activating tasks (***activating program instances***) in the resulting combination to one of the "available cores" (*see*

> *supra*, Element [1.3.iv]) since the other allotted cores would all be occupied by the continuing tasks.

Pet. 57 (citing Ex. 1003 ¶ 124).

3. Patent Owner's Arguments

Patent Owner disputes neither Petitioner's analysis of limitations 10.P, 10.1, 10.2, 10.3i, 10.3ii, 103iii, 10.3iv., and 10.4, as it pertains to Ground 1B, nor Petitioner's rationale for combining the teachings of Feitelson with those of Agrawal and Brent. Otherwise, Patent Owner relies on the same arguments we have discussed with respect to the combination of Agrawal and Brent, as set forth in Ground 1A. *See supra* § II.D.5. As we explain above in our analysis of Ground 1A, we do not agree with Patent Owner's arguments.

4. Conclusion for Claim 10

Petitioner has persuasively shown that the combination of Agrawal, Brent, and Feitelson teaches all limitations of claim 10. Petitioner also has put forth persuasive reasons for combining these references with a reasonable expectation of success. *See* Pet. 39–41 (citing Ex. 1003 ¶¶ 102–103; *see also id.* ¶¶ 63–70, 119–128). We credit Dr. Chase's opinion that one of ordinary skill in the art "would have found it obvious to apply Feitelson's suggestion for keeping threads (e.g., tasks) executing in successive scheduling quanta on their same processors (e.g., cores) across the successive quanta (thereby "increas[ing] the effective scheduling quantum" for the tasks). *Id.* at 55 (citing Ex. 1003 ¶ 122). We also credit Petitioner's explanation, relying on Dr. Chase's declaration testimony, that one of ordinary skill in the art

> would have understood that switching threads or tasks from one processor to another ordinarily requires overhead and additional

> clock cycles to interrupt ongoing execution of the thread, store data that characterizes the thread's current execution context, and transfer the instructions, data, and other necessary information for executing the thread to the new processor.

Id. at 56 (citing Ex. 1003 ¶ 123). Thus, regarding claim 10, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Agrawal, Brent, and Feitelson.

5. Independent Claim 19

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 19 as set forth in the Petition. Pet. 59–67. We use Petitioner's notations to identify the claim limitations.

i. [19.P] A process for computing resource management, the process comprising:[15]

Petitioner asserts that, if the preamble is treated as limiting, the combination of Agrawal, Brent, and Feitelson discloses its contents. Pet. 59 (citing Ex. 1003 ¶¶ 133–134; Ex. 1004, 1–2; Ex. 1005 18–23, Figs. 2–3; Ex. 1006, Abstract, ¶¶ 9–13). More particularly, Petitioner asserts that the asserted combination "provide a process for computing resource management." *Id.*

[15] The issue of whether the preamble is limiting need not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

ii. *[19.1] sub-process configured to periodically, once for each of a series of core allocation periods (CAPs), to carry out an allocation of an array of processor cores among a set of software programs, said sub-process comprising:*

Petitioner asserts that "Agrawal discloses, and the combination provides, a sub-process configured to periodically, once for each of a series of scheduling quanta (***core allocation periods (CAPs)***), carry out an allocation of processors among a set of jobs (***software programs***)." Pet. 60 (citing Ex. 1003 ¶ 135; Pet. 13–17). More particularly, Petitioner asserts that Agrawal discloses equal-sized scheduling quanta, and "its techniques relate to 'adaptive' (also referred to as 'dynamic') scheduling, in which 'the job scheduler can change the number of processors allotted to a job while the job is executing.'" *Id.* (citing Ex. 1004, 2). According to Petitioner, "it would have been obvious to apply two-level scheduling based on Agrawal in the context of an 'array of processor cores' as taught in Brent." *Id.* (citing Ex. 1003 ¶ 135; Ex. 1005 ¶¶ 16–17, Fig. 2; Pet. 39–43).

iii. *[19.2] (i) a first round of the allocation, in which a subset of the cores are allocated among the software programs so that any actually materialized demands for the cores by each of the software programs up to their respective entitled shares of the cores are met, and*

Petitioner asserts that

> Agrawal discloses that "[b]etween quanta $q-1$ and q, the task scheduler determines its job's ***desire*** d_q, which is the number of processors the job wants for quantum q." [Ex. 1004, 2]. "The job scheduler follows some processor allocation policy to determine the ***processor availability*** p_q, which is the number of processors the job is entitled to get for the quantum q." *Id.* "The number of processors the job receives for quantum q is the job's ***allotment*** $a_q = \min\{d_q, p_q\}$, which is the smaller of its desire and the processor availability." *Id.*

Pet. 60–61. In light of this disclosure, Petitioner contends that Agrawal discloses "a first round of allocation for each scheduling quantum in which a 'job scheduler' allocates a number of processors among the jobs (***software programs***) so that any actually materialized demands for processors by each of the jobs up to their entitled processor availability p_q (***entitled shares***) are met." *Id.* at 61 (citing Ex. 1003 ¶ 136–141; Ex. 1004, 1–2).

According to Petitioner, "Agrawal invites use of any 'processor allocation policy' to identify the respective entitlements p_q of the jobs, and although claim 19 requires no particular policy, one predictable option would be to use a pre-defined number, e.g., a number that prescribes a 'minimum partition size' per program," as suggested by Feitelson. *Id.* (citing Ex. 1006, 44). Based on Feitelson, Petitioner concludes that one of ordinary skill in the art

> would have found it obvious to define a minimum partition size for each job or class of jobs in the resulting combination to "guarantee a certain level of service" to the jobs. [Ex. 1003 ¶ 137; Ex. 1006, 44]. Other techniques could also be used to define entitlements, such as equipartitioning in which each job is entitled to an equal number of processors. [Ex. 1006], 38.

Id.

iv. [19.3] (ii) a second round of the allocation, in which any of the cores that remain unallocated after the first round are allocated among the software programs whose materialized demands for the cores had not been met by amounts of the cores so far allocated to them by the present invocation of the allocation; and

Petitioner asserts that "Agrawal's technique for allotting the minimum of a job's desired number of processors (e.g., cores) d_q and entitled share of processors p_q can leave a subset of processors (cores) un-allotted for the next scheduling quantum (***CAP***)." Pet. 61 (citing Ex. 1003 ¶¶ 136–141;

Ex. 1004, 1–2; Pet. 60–63). Petitioner acknowledges that Agrawal may not "expressly describe what to do with the unallotted processors," but notes that "Feitelson addresses a number of issues that would be considered when determining how many processors (e.g., cores) to allocate to a job (i.e., the job's 'partition size')." *Id.* at 63–64 (citing Ex. 1006, 38). For example, Feitelson suggests "'allocat[ing] all the PEs [processing entities] if there is sufficient demand.'" *Id.* at 64 (citing Ex. 1006, 16; *see also id.* at 15–18, 28, 31–32, 37–46). Based on these teachings, Petitioner concludes

> it would have been obvious to configure the Agrawal-Brent-Feitelson system to provide a second round of allocation in which any cores that remained unallotted after the first round would be allocated among the jobs whose materialized demands for the cores had not been met by amounts of the cores so far allocated to them by the present invocation of the allocation.

Id. (citing Ex. 1003 ¶¶ 142–148).

Petitioner explains that one of ordinary skill in the art "would have followed Feitelson's suggestion to allocate additional cores to mitigate [occurrences] in which 'resources are explicitly wasted by the system.'" *Id.* (citing Ex. 1006, 38). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that "[u]n-allotted cores would otherwise be inaccessible to the jobs running on the array for the scheduling quantum, thereby leaving those cores 'idle' and a 'wasted' resource." *Id.* (citing Ex. 1003 ¶ 144). In addition, Petitioner contends that

> [p]roviding separate rounds of allocation in which the second follows the first also would have been obvious since performing the "first round" earlier would provide a simplified method of ensuring that each requesting job was allocated a number of cores up to its entitled share, and would readily yield a determination of the surplus number of cores that could be allocated in the second round.

Id. at 64–65 (citing Ex. 1003 ¶¶ 144–145 (citing corroboration Exs. 1016, 1019, 1021)). According to Petitioner, one of ordinary skill in the art "would have expected that the second round of allocation would improve performance of those jobs that are allocated additional cores from the un-allotted pool." *Id.* at 65 (citing Ex. 1003 ¶ 146; Ex. 1006, 38). Petitioner notes that "[a]llocating additional cores to the job to meet or move closer to the number that the job could 'effectively use' would predictably improve performance by allowing more of the job's tasks to be executed sooner" and "was exceedingly common" at the time of the invention. *Id.* (citing Ex. 1003 ¶ 146–148 (citing corroboration Exs. 1016, 1017, 1018, 1019, 1021)).

v. [19.4] a sub-process configured to assign individual instances of the set of software programs to individual cores of the array so that each such instance of the software programs, which was selected for execution on the array of cores on consecutive CAPs, is assigned to same one of the cores for execution on each of such consecutive CAPs.

Petitioner asserts that the combination of Agrawal, Brent, and Feitelson

> provide[s] a sub-process configured to assign individual tasks (***instances***) of the set of jobs (***software programs***) to individual cores of the array so that each such task (***instance***), which was selected for execution on the array of cores on consecutive scheduling quanta (***CAPs***), is assigned to same one of the cores for execution on each of such consecutive CAPs.

Pet. 67 (citing Ex. 1003 ¶ 149; Pet. 53–58).

6. Patent Owner's Arguments

At the outset, Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with

respect to limitations 19.P–19.2, and 19.4. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that the combination Agrawal, Brent, and Feitelson teaches these limitations of claim 19.

Patent Owner argues that Petitioner's ground of unpatentability based on Agarwal, Brent, and Feitelson is deficient with respect to claim 19. PO Resp. 29–33; PO Sur-reply 23–25. More particularly, Patent Owner argues that Petitioner's "argument that this claim would be obvious fundamentally violates Agrawal's core allocation policy according to which a processor's allotment is capped at its entitlement." PO Resp. 29 (citing Ex. 2003 ¶ 64).

7. Discussion

Patent Owner argues that "[c]laim 19 recites two rounds of allocation," which requires a first round that "allocates cores to programs to fulfill 'actually materialized demands' up to a program's entitlement" and a second round that "allocates 'any of the cores that remain unallocated after the first round.'" PO Resp. 29 (citing Ex. 2003 ¶ 63). Relying on the declaration testimony of Dr. Smith, Patent Owner asserts that "disregarding Agrawal's capped-at-entitlement allocation policy would fundamentally change Agrawal's principle of operation while raising the risk of conflicts between jobs for processor demand." *Id.* (citing Ex. 2003 ¶¶ 64–69).

Patent Owner explains that "Agrawal allocates processors to jobs based on the lesser of desire and processor entitlement" (PO Resp. 30 (citing Ex. 1003 ¶¶ 42–43, 64; Ex. 1004, 2), and "[t]he Petition acknowledges that the first round of allocation would operate according to this principle," but "in the second round of allocation, the Petition pitches this allocation policy and suggests that Agrawal's job scheduler would allocate cores to Job 2

above Job 2's entitlement." *Id.* at 31–32. And, relying on the declaration testimony of Dr. Smith, Patent Owner contends that "given the importance of allotment in Agrawal, [one of ordinary skill in the art] would not be inclined [to] modify Agrawal in a way that would render the classification system inoperable." *Id.* at 33 (citing Ex. 2003 ¶ 70).

Petitioner responds that "Agrawal renders obvious a first round of allocation in which a job scheduler allocates a number of processors among the jobs (***software programs***) so that any actually materialized demands for processors by each of the jobs up to their entitlement p_q are met." Pet. Reply 21 (citing Ex. 1004, 2; Pet. 60). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that one of ordinary skill in the art

> would have understood that Agrawal's practice of initially allocating processors up to the lesser of entitlement or desire would commonly leave some processors unallocated, e.g., when the system includes a sufficient number of processors to meet each job's entitlement but some jobs express a desire for fewer processors than their entitlement (or even express no desire for processors) for a given quantum.

Id. at 21–22 (citing Ex. 1003 ¶¶ 137–141; Pet. 60–61). When some processors are left unallocated after the first round, Petitioner asserts that "the prior art confirms that it would have been obvious to allocate in a second round processing cores that remained unallocated in the first round to jobs whose desire had not initially been fully met in the first round." *Id.* at 22 (citing Ex. 1003 ¶¶ 136–147; Pet. 63–67).

For example, Petitioner points out that Feitelson discloses that "'all the PEs [processing entities]' should be allocated 'if there is sufficient demand'" (Pet. Reply 22 (citing Ex. 1006, 38)), and contends that one of ordinary skill in the art

> would have been motivated to follow Feitelson's suggestion to allocate additional cores to jobs in the face of sufficient demand (e.g., unfulfilled desire), especially where doing so would beneficially increase the amount of work that could be accomplished by the system in each quantum and would mitigate the possibility of "resources [being] explicitly wasted by the system."

Id. (citing Ex. 1003 ¶ 144). Petitioner adds that there is no disclosure in Agrawal that "implements a strict 'capped-at-entitlement allocation policy.'" *Id.* at 23. Thus, Petitioner contends that the proposed modification to Agrawal would not violate Agrawal's core allocation policy, but rather, "improves Agrawal by putting its unallocated cores to beneficial use." *Id.* at 24.

Based on the fully developed trial record, we do not agree with Patent Owner's arguments. Instead, we agree with Petitioner that "nothing in Agrawal suggests that cores that remain ***unallocated*** after carrying out Agrawal's initial round of allocation must or even should remain unallocated for a quantum, e.g., rather than being assigned to jobs whose demand remains unfulfilled ***once all jobs' entitlements have already been met***." Pet. Reply 24 (citing Ex. 1032 ¶¶ 28–29). We also agree with Petitioner that "Agrawal explicitly indicates that the job scheduler can follow any suitable 'processor allocation policy to determine the processor availability p_q,'" and credit the declaration testimony of Dr. Chase who explains that one of ordinary skill in the art would have understood such a teaching "to encompass policies that would leave some cores unallocated if $d_q < p_q$ for at least some jobs." *Id.* at 23 (citing Ex. 1003 ¶¶ 141–148; Ex. 1032 ¶ 27).

Relying on the declaration testimony of Dr. Smith, Patent owner argues that "disregarding Agrawal's capped-at-entitlement allocation policy

would fundamentally change Agrawal's principle of operation while raising the risk of conflicts between jobs for processor demand." PO Resp. 29 (citing Ex. 2003 ¶¶ 64–69); *see also id.* at 32 ("Agrawal's capped-at-entitlement allocation policy . . . ensure[s] that a particularly processor intensive program does not dominate the limited processor resources."). According to Patent Owner, "modifying Agrawal's allotment to essentially collapse into demand would render Agrawal's classification system inoperable." PO Resp. 32 (citing Ex. 2003 ¶ 70).

We do not find Patent Owner's arguments persuasive. Neither Patent Owner nor Dr. Smith adequately explains how a second round of allocation following a first round of allocation according to Agrawal's core allocation policy, i.e., each job is allotted at least its desired number of cores d_q up to its entitled processor availability p_q, would result in allotment collapsing into demand such that it would render Agrawal inoperable. *See* Ex. 1004, 2 ("allotment . . . is the smaller of its desire and the processor availability). Instead, Petitioner is simply modifying Agrawal to include a second round of allocation, as suggested by Feitelson, and as such, we agree with Petitioner that its ground of unpatentability "explicitly preserves in the first round of allocation Agrawal's scheme that ensures each job is allotted at least its desired number of cores d_q up to its entitled processor availability p_q." Pet. Reply 24 (citing Pet. 60–62). We credit Dr. Chase's declaration testimony that one of ordinary skill in the art

> would not have understood Agrawal to have a hard "capped-at-entitlement allocation policy" that would preclude a second round of allocation after the jobs' desire up to their entitled share of cores is met. Such a policy would have been substantially out of step with ordinary practices and expectations in the field since [one of ordinary skill in the art] would have desired to maximize

> the amount of work performed over a period of time while avoiding wasted or unutilized resources.

Ex. 1032 ¶ 26 (citing Ex. 1003 ¶ 148; Ex. 1006, 38). We find Dr. Chase's testimony to be credible and supported by the record. *See, e.g.*, Ex. 1003 ¶ 148 (citing corroboration Exs. 1016–1019, 1021).

Based on the complete record, we do not agree with Patent Owner that the addition of a second round of allocation, as suggested by Feitelson, would render Agrawal inoperable or violate its core allocation policy. We find Petitioner has persuasively shown that one of ordinary skill in the art would have "been familiar with many systems configured to re-allocate spare capacity or unused resources to jobs with higher demands" (Pet. 65–66 (citing Ex. 1003 ¶¶ 146–148)), and he/she would have "expected that the second round of allocation would improve performance of those jobs that are allocated additional cores from the un-allotted pool" (*id.* at 65 (citing Ex. 1003 ¶ 146)).

a) Conclusion for Claim 19

Petitioner has persuasively shown that the combination of Agrawal, Brent, and Feitelson teaches all limitations of claim 19. Petitioner also has put forth persuasive reasons for combining these references. *See* Pet. 39–41 (citing Ex. 1003 ¶¶ 63–70, 102–103), 64–67 (citing Ex. 1003 ¶¶ 142–148). Thus, regarding claim 19, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Agrawal, Brent, and Feitelson.

8. Independent Claim 29

Petitioner provides an element-by-element analysis of where each element in the challenged claims is disclosed in, or would have been obvious

in view of, the combination of Agrawal, Brent, and Feitelson. *See* Pet. 72–73 (mapping limitations 29.P, 29.1, 29.2, 29.3, 29.4.i–29.4.iv, and 29.5 to limitations 10.P, 10.1, 10.2, 10.3.i–10.3.iv, 10.4, respectively). Petitioner also relies on the testimony of Dr. Chase for evidentiary support. *Id.* (citing Ex. 1003 ¶¶ 164–173).

Other than by relying on its arguments presented for Ground 1A, as discussed above, Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with respect to limitations 29.1, 29.2, 29.3, 29.4.i–29.4.iv, and 29.5. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that the combination of Agrawal, Brent, and Feitelson discloses these limitations. *See supra* §§ II.E.2–4.

As discussed above with respect to claim 10, Petitioner has put forth persuasive reasons for combining Agrawal, Brent, and Feitelson with a reasonable expectation of success. *See* Pet. 39–41 (citing Ex. 1003 ¶¶ 102–103; *see also id.* ¶¶ 63–70, 119–128). Thus, regarding claim 29, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Agrawal, Brent, and Feitelson.

9. Dependent Claims 13, 14, 20–22, 24, 25, 27, 32, and 33

Petitioner contends that the additional limitations of dependent claims 13, 14, 20–22, 24, 25, 27, 32, and 33 are would have been unpatentable as obvious over the combination of Agrawal, Brent, and Feitelson. Pet. 58–59,

68–72, 73. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003 ¶¶ 129–132, 150–161, 174–177).

i. [13] The method of claim 10, wherein the assigning produces, for a given one of the core slots, identification of a core type demanded by the program instance assigned for execution on the given core slot for a given CAP.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 13 obvious. Pet. 58–59 (citing Ex. 1003 ¶¶ 129–130; Ex. 1004, 1–2; Ex. 1005 ¶¶ 16–18, 22, 27, 29). More particularly, Petitioner asserts that, for each of the cores of array 270, Brent discloses "that the assigning produces identification of a core type (e.g., cDSP, gDSP, eCPU) demanded by the program task (***instance***) assigned for execution on the given core slot for a given time frame (e.g., a scheduling quantum (***CAP***) as disclosed in Agrawal)." *Id.* at 58.

ii. [14] The method of claim 10, wherein one of the instances of a given one of the programs corresponds to a task, a process, an actor, a thread, a procedure or a function of the given program.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 14 obvious. Pet. 59 (citing Ex. 1003 ¶¶ 131–132; Ex. 1004, 1; Ex. 1005 ¶¶ 15, Figs. 1–2). More particularly, Petitioner asserts that "[t]he teachings of Agrawal, Brent, and Feitelson disclose that one of the executable instances of a given of the jobs (program) corresponds to a 'task' of the given job." *Id.*

iii. *[20] The process of claim 19, wherein the materialized demand for the cores by a given one of the programs is expressed as a number of schedulable instances that the given program has ready for execution for a CAP following a present invocation of the allocation.*

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 20 obvious. Pet. 68 (citing Ex. 1003 ¶¶ 150–151; Ex. 1004, 1–2). More particularly, Petitioner asserts that "Agrawal discloses, and the combination provides, that "between quanta *q-1* and *q*, the task scheduler determines its job's ***desire*** d_q, which is the number of processors the job wants for quantum *q*." *Id.* (citing Ex. 1004, 2). Petitioner explains that

> [t]he *desire* d_q provides the materialized demand of a given job (***program***) which, as explained above (Element [10.2]), indicates how many cores the job is initially allotted (if the job's entitled share p_q is not lower), and expresses a number of ready-to-execute tasks (***schedulable instances***) that the given program has ready for execution for a quantum (***CAP***) following a present invocation of the allocation.

Id. (citing Ex. 1003 ¶ 151; Ex. 1004, 1–2).

iv. *[21] The process of claim 20, wherein the number of schedulable instances that the given program has ready for execution for the CAP following the present invocation of the allocation is formed: (A) independently of (1) the respective numbers for other programs of the set, (2) the other programs' utilizations of any cores allocated to them, and (3) utilization of the cores across the array, as well as (B) so that said number, for at least some of the CAPs, is different than the number of the cores allocated to the given program for a CAP preceding the present invocation of the allocation.*

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 21 obvious. Pet. 68–69 (citing Ex. 1003 ¶¶ 152–153; Ex. 1004, 1–2). More particularly, Petitioner asserts that "Agrawal

discloses, and the combination provides, for 'model[ing] the execution of a parallel job as a dynamically unfolding [dag] of tasks, where each node in the dag represents a unit-time task' and '[a]n edge represents a serial dependency between tasks.'" *Id.* at 68 (citing Ex. 1004, 1). Petitioner explains that "[a] task becomes ready to be executed when all its parents have been executed" and

> [t]he materialized demand of a job, as previously described, corresponds to the job's *desire* d_q (e.g., a job will obtain d_q number of processors for a quantum if $p_q >= d_q$), and for instantaneous parallelism feedback is based on the number of schedulable tasks (***instances***) that the job has ready to execute for an upcoming quantum (CAP).

Id. (citing Ex. 1003 ¶ 153; Ex. 1004, 1–2). Petitioner further asserts that "[t]he number of tasks that a job has ready to execute in Agrawal is simply based on the resolution of dependencies (e.g., 'all its parents have been executed')." *Id.* at 69 (citing Ex. 1004, 1). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that "[t]he number of schedulable tasks (***instances***), for at least some quanta (***CAPs***), will also be different than the number of the cores allocated to the given job (***program***) for a quantum preceding the present invocation of the allocation." *Id.* (citing Ex. 1004, 1–2; Ex. 1003 ¶ 153).

v. [22] The process of claim 19, wherein any given program gets allocated at least its entitled share of the cores following such invocations of the allocation for which it demanded at least such entitled share, wherein the entitled share of the cores for a given program is one of: i) an even division of an amount of the cores within the array of cores, or ii) a contract based amount of cores.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 22 obvious. Pet. 69–70 (citing Ex. 1003 ¶¶ 154–157;

Ex. 1004, 1–2; Ex. 1006, 44–45). More particularly, Petitioner asserts that the combination Agrawal, Brent, and Feitelson "provides a first round of allocation in which any given job (***program***) gets allocated at least its entitled share of the cores following such invocations of the allocation for which it demanded at least such entitled share." *Id.* at 69 (citing Ex. 1004, 1–2; Ex. 1006, 44). Petitioner identifies that Feitelson "discloses that a partition size, e.g., a 'minimal partition size' to which a job is entitled, can be an even division of an amount of cores within a set of processors (e.g., an array of cores as Brent teaches)." *Id.* at 69–70 (citing Ex. 1006, 44–45). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that

> [i]t would have been obvious to entitle jobs to an even division share of cores, e.g., based on a folding policy, to achieve any of the advantages of such an approach that Feitelson expressly describes—including benefits for "load balancing" and achieving "communication locality" among the tasks in a job's allotted cores.

Id. at 70 (citing Ex. 1003 ¶ 157).

vi. [24] The process of claim 19, wherein, on invocations of the allocation when there is no materialized demand for the cores by a given one of the programs, the subset of the cores allocated by first round for the given program comprises zero cores, even when the entitled share of the cores of the given program is a positive number of cores.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 24 obvious. Pet. 70–71 (citing Ex. 1003 ¶¶ 158–159; Ex. 1004, 2). More particularly, Petitioner asserts

> Agrawal discloses, and the combination provides, that a given job (***program***) would be allocated zero cores in the first round when the job has no materialized demand (e.g., such that d_q =0 and the job has no ready tasks that would be counted under

> instantaneous parallelism), even when the entitled share of the cores p_q for the job is a positive number of cores.

Id.

vii. *[25] The process of claim 19, wherein the sub-process configured to carry out an allocation further comprises a third round of the allocation, by which round all the cores that remain unallocated after the second round are allocated among the programs so that any programs with no existing allocated cores are prioritized in getting cores allocated.*[16]

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 25 obvious. Pet. 71–72 (citing Ex. 1003 ¶¶ 160–161; Ex. 1004, 1–2; Ex. 1006, 31–32, 38). Petitioner explains that

> [a]llocating additional cores to any jobs whose materialized demands had not been met by the first-round of allocation would have been obvious to increase resource utilization and performance of the jobs' execution in a quantum—particularly by allocating additional cores up to each job's desire d_q (since the job could "effectively use" each processor in this case).

Id. at 71. Petitioner explains, however, that "Feitelson notes that '[i]t is well known that adding more and more PEs suffers from diminishing returns, and might even cause a degradation in performance.'" *Id.* (citing Ex. 1006, 31–32). Relying on the declaration testimony of Dr. Chase, Petitioner asserts that, "in many cases, rather than allotting even more cores to the jobs whose materialized demands had been met, [one of ordinary skill in the art] instead would have sought to purpose any remaining un-allotted cores after the second round of execution for another purpose." *Id.* (citing Ex. 1003 ¶ 161).

[16] On September 18, 2018, a certificate of correction was entered that changed the scope of dependent claim 25 post-issuance. *See* Ex. 1003 ¶ 160.

Also, relying on the declaration testimony of Dr. Chase, Petitioner asserts that

> it would have been obvious to provide such a third round of allocation in the resulting combination to increase utilization (e.g., avoid wasting idle cores), and to allow certain jobs whose tasks may become ready to execute before a quantum has expired to begin executing immediately during the current quantum rather than waiting for the next scheduling interval.

Id. at 72 (citing Ex. 1003 ¶ 161; Ex. 1004, 1–2).

viii. [27] The process of claim 19, wherein one of the instances of the given program corresponds to a task, a process, an actor, a thread, a procedure or a function of the given program.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 27 obvious for the same reasons discussed above with respect to claim 14. Pet. 72 (citing Ex. 1003 ¶¶ 162–163).

ix. [32] The system of claim 29, wherein the assigning produces, for a given one of the core slots, identification of a core type demanded by the program instance assigned for execution on the given core slot for a given CAP.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 32 obvious for the same reasons discussed above with respect to claim 13. Pet. 73 (citing Ex. 1003 ¶¶ 174–175).

x. [33] The system of claim 29, wherein one of the instances of a given one of the programs corresponds to a task, a process, an actor, a thread, a procedure or a function of the given program.

Petitioner asserts that the combination Agrawal, Brent, and Feitelson renders claim 33 obvious for the same reasons discussed above with respect to claim 14. Pet. 73 (citing Ex. 1003 ¶¶ 176–177).

xi. Conclusion for Dependent Claims 13, 14, 20–22, 24, 25, 27, 32, and 33

Patent Owner does not challenge Petitioner's contentions with respect to the limitations of dependent claims 13, 14, 20–22, 24, 25, 27, 32, and 33, and Petitioner's contentions are supported by the cited evidence. For the reasons given by Petitioner and summarized above, we determine that Petitioner has shown, by a preponderance of the evidence, that the additional limitations of each of claims 13, 14, 20–22, 24, 25, 27, 32, and 33 would have been obvious over the combination of Agrawal, Brent, and Feitelson. Therefore, Petitioner has established by a preponderance of the evidence that claims 13, 14, 20–22, 24, 25, 27, 32, and 33 are unpatentable under § 103(a) over Agrawal, Brent, and Feitelson.

10. Conclusion for Ground 1B

For the foregoing reasons, we conclude that Petitioner has shown by a preponderance of the evidence that claims 10, 13, 14, 19–22, 24, 25, 27, 29,

32, and 33 are unpatentable under § 103(a) over Agrawal, Brent, and Feitelson.

III. CONCLUSION

In summary:

Claims	35 U.S.C. §	References/Basis	Claims Shown Upatentable	Claims Not Shown Unpatentable
15, 17, 18, 34, 37	103(a)	Agrawal, Brent	15, 17, 18, 34, 37	
10, 13, 14, 19–22, 24, 25, 27, 29, 32, 33	103(a)	Agrawal, Brent, Feitelson	10, 13, 14, 19–22, 24, 25, 27, 29, 32, 33	
Overall Outcome			10, 13–15, 17–22, 24, 25, 27, 29, 32–34, 37	

IV. ORDER

In consideration of the foregoing, it is hereby

ORDERED that Petitioner has established based on a preponderance of evidence that claims 10, 13–15, 17–22, 24, 25, 27, 29, 32–34, and 37 of the '833 patent are unpatentable as set forth above; and

FURTHER ORDERED that, because this is a Final Written Decision, parties to the proceeding seeking judicial review of the Decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.[17]

[17] Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this Decision, we draw Patent Owner's attention to the April 2019 Notice Regarding Options for Amendments by Patent Owner Through Reissue or Reexamination During a Pending AIA Trial Proceeding. *See* 84 Fed. Reg. 16,654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. *See* 37 C.F.R. § 42.8(a)(3), (b)(2).

IPR2022-00527
Patent 9,632,833 B2

For PETITIONER:

Walter Renner
David Holt
Nicholas Stephens
Jennifer Huang
FISH & RICHARDSON P.C.
axf-ptab@fr.com
holt2@fr.com
nstephens@fr.com
jjh@fr.com

For PATENT OWNER:

Scott McKeown
ROPES & GRAY LLP
scott.mckeown@ropesgray.com

W. Cook Alciati
Michael Dorfman
GARDELLA GRACE P.A.
calciati@gardellagrace.com
mdorfman@gardellagrace.com

Trials@uspto.gov
571-272-7822

Paper 27
Entered: September 18, 2023

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

THROUGHPUTER, INC.,
Patent Owner.

IPR2022-00528
Patent 9,424,090 B2

Before MICHAEL R. ZECHER, JOHN A. HUDALLA, and MATTHEW S. MEYERS, *Administrative Patent Judges*.

MEYERS, *Administrative Patent Judge*.

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

I. INTRODUCTION

A. Background and Summary

Microsoft Corporation ("Petitioner") filed a Petition (Paper 2, "Pet.") requesting an *inter partes* review of claims 1–7 and 9–15 ("the challenged claims") of U.S. Patent No. 9,424,090 B2 (Ex. 1001, "the '090 patent"). ThroughPuter, Inc. ("Patent Owner") filed a Preliminary Response. Paper 7. We determined that the information presented in the Petition established that there was a reasonable likelihood that Petitioner would prevail with respect to its unpatentability challenges. Pursuant to 35 U.S.C. § 314, we instituted this proceeding on September 19, 2022, as to all challenged claims and all grounds of unpatentability presented in the Petition. Paper 12 ("Dec. on Inst.").

During the course of trial, Patent Owner filed a Patent Owner Response (Paper 17, "PO Resp."), and Petitioner filed a Reply to the Patent Owner Response (Paper 18, "Pet. Reply"). Patent Owner also filed a Sur-reply (Paper 19, "PO Sur-reply"). An oral hearing was held on June 20, 2023, and a transcript of the hearing is included in the record. Paper 26 ("Tr.").

We have jurisdiction under 35 U.S.C. § 6. This decision is a Final Written Decision under 35 U.S.C. § 318(a) as to the patentability of claims 1–7 and 9–15 of the '090 patent. For the reasons discussed below, Petitioner has demonstrated by a preponderance of the evidence that claims 1–7 and 9–15 of the '090 patent are unpatentable.

B. Real Parties-in-Interest

Petitioner identifies itself as the only real party-in-interest. Pet. 74. Patent Owner identifies itself as the only real party-in-interest. Paper 11, 2.

C. Related Proceedings

The parties identify that the '090 patent is involved in *ThroughPuter, Inc. v. Microsoft Corporation*, No. 2:22-cv-00344 (W.D. Wash). Paper 10, 1; Paper 11, 2. Patent Owner indicates that, "[a]s of August 18, 2022, the related district court litigation has been stayed pending resolution of the present *inter partes* review proceeding." Paper 11, 2.

We also note that Petitioner has challenged other patents owned by Patent Owner in IPR2022-00527, IPR2022-00574, IPR2022-00757, and IPR2022-00758. We denied institution of *inter partes* review in IPR2022-00757 and IPR2022-00758. Final Written Decisions in IPR2022-00527 and IPR2022-00574 are being issued concurrently with this Decision.

D. The '090 Patent

The '090 patent is titled "Scheduling Tasks to Configurable Processing Cores Based on Task Requirements and Specification." Ex. 1001, code (54). The '090 patent pertains to "techniques for improving information processing efficiency and performance through dynamically adapting processing resource types to match processing task types." *Id.* at 1:34–38.

According to the '090 patent, "the range of processing jobs for a given instance of processing hardware (e.g., a server blade used for computing Infrastructure-as-a-Service), especially over the lifetime of the given hardware instance, may comprise several types of jobs and their tasks, with each type best suited for its corresponding, distinct, processor type." Ex. 1001, 1:56–61. The '090 patent describes that an array of hardware processing units, including different processor types, will likely run several types of jobs, each best suited to different processor types. *Id.* at 1:61–67.

The '090 patent notes that it is "infeasible to predict ahead of time . . . what would be the optimal type of core for any given processor instance, or the optimal breakdown of core types for a given array of processors," and as such, conventional scheduling techniques may lead to jobs "processed often by suboptimal types of processing cores." *Id.* at 2:1–13.

The '090 patent "provides a processing task type adaptive manycore processor system for executing time variable sets of processing software program tasks of differing types on their assigned cores of matching types." Ex. 1001, 3:64–4:5. The '090 patent thus provides "application processing load adaptive allocation of the cores among the software applications configured for the system, as well as for dynamically reconfiguring the core slots according to the types of the processing tasks assigned to any given core slot of the multi-core fabric." *Id.* at 8:40–49.

To accomplish this, the '090 patent utilizes for each application "capacity demand indicators 130 to the controller 140. Each of these indicators 130, referred to herein as core-demand-figures (CDFs), express how many cores 120 their associated app-task is presently able [to] utilize for its ready to execute instances." Ex. 1001, 8:64–9:3. The system

> repeatedly selects and places the to-be-executing instances 240 of the set of locally hosted app-tasks 210 to their assigned target core slots 120, as well as determines, for the core-slot reconfiguration subsystem of the processor 100 (FIGS. 3 and 4), the demanded core types for the core slots of the array 115, along with notifications of changes in the demanded core type for each given core slot.

Id. at 10:10–17.

Figure 3, reproduced below, is a block diagram of the system of the '090 patent.

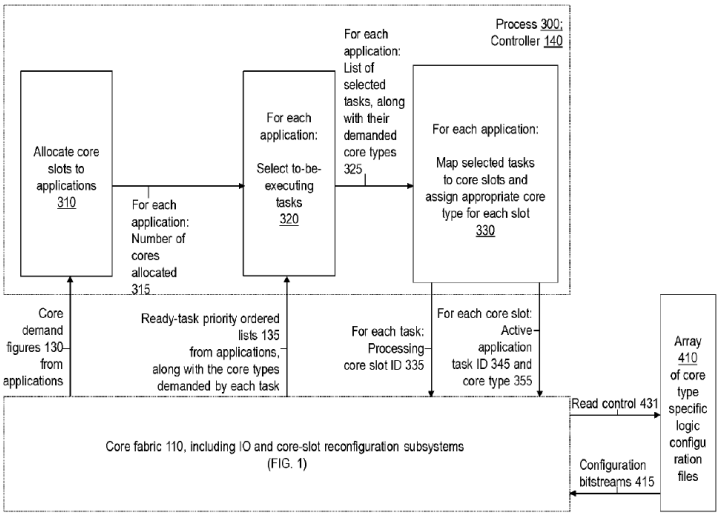


Figure 3 shows algorithm 310, which allocates core slots to applications, application selection step 320, and core mapping step 330. Ex. 1001, 14:5–67.

E. The Challenged Claims

Petitioner challenges claims 1–7 and 9–15 of the '090 patent. Pet. 2. Challenged claims 1 and 9 are independent. Claim 1 is illustrative, and is reproduced below, with Petitioner's bracketing, labels, and formatting.

> [1.P] A method for assigning a set of processing tasks to an array of processor cores of configurable types, the method comprising:
>
> [1.a] executing time variable subsets of the processing tasks of differing types on their assigned processor cores of matching types, wherein the matching type for the assigned processor core for a given processing task of the set corresponds to a type of a processor core demanded by the given processing task;
>
> [1.b] for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks;
>
> [1.c] assigning the selected tasks for execution on the processor cores for the next CAP in a manner to maximize, within the array, a number of processor cores whose assigned

processing tasks for a present and the next CAP demands the same type of processor core; and

[1.d] configuring the array such that the type of any given processor core in the array matches the type of processing task assigned for execution on the given processor core for the next CAP.

Ex. 1001, 19:59–20:15; Pet. 31–38.

F. Instituted Grounds of Unpatentability

We instituted trial based on all asserted claims and grounds of unpatentability as follows:

Claims Challenged	35 U.S.C. §[1]	References
1–5, 7	103(a)	Chen[2], Agrawal[3]
3, 4	103(a)	Chen, Agrawal, Compton[4]
1, 2, 5–7, 9–15	103(a)	Chen, Agrawal, Brent[5]

Pet. 1–2; Dec. on Inst. 27. To support its Petition, Petitioner proffers a Declaration of Jeffrey S. Chase, Ph.D. Ex. 1003.

---

[1] The Leahy-Smith America Invents Act ("AIA"), Pub. L. No. 112-29, 125 Stat. 284 (2011), amended 35 U.S.C. §§ 102 and 103 effective March 16, 2013. Because the '090 patent claims priority as a continuation to an application filed before March 16, 2013, we apply the pre-AIA versions of the statutory bases for unpatentability. *See* Ex. 1001, code (63).

[2] G. Chen et al., "Configuration-Sensitive Process Scheduling for FPGA-Based Computing Platforms," Proceedings Design, Automation and Test in Europe Conference and Exhibition, Vol. 1, pp. 486–93 (2004) (Ex. 1004, "Chen").

[3] Kunal Agrawal et al., "Adaptive Scheduling with Parallelism Feedback," Proceedings of the 2006 ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2006 (Ex. 1005, "Agrawal").

[4] Katherine Compton & Scott Hauck, "Reconfigurable Computing: A Survey of Systems and Software," ACM Computing Surveys, Vol. 34, No. 2, June 2002, pp. 171–210 (Ex. 1006, "Compton").

[5] US 2010/0131955 A1, published May 27, 2010 (Ex. 1007, "Brent").

II. ANALYSIS

A. Principles of Law

"In an [*inter partes* review], 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) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review petitions to identify "with particularity . . . the evidence that supports the grounds for the challenge to each claim")). Petitioner bears the burden of persuasion to prove unpatentability of each challenged claim by a preponderance of the evidence. 35 U.S.C. § 316(e). This burden does not shift to Patent Owner, except in limited circumstances not present here. *Dynamic Drinkware, LLC v. Nat'l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015).

A patent claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, "would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) when in evidence, any objective evidence of obviousness or non-obviousness.[6] *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). "While the sequence of these questions might be

[6] The parties do not present objective evidence of non-obviousness.

reordered in any particular case" (*KSR*, 550 U.S. at 407), the U.S. Court of Appeals for the Federal Circuit has explained that an obviousness determination can be made only after consideration of all of the *Graham* factors. *See, e.g.*, *Kinetic Concepts, Inc. v. Smith & Nephew, Inc.*, 688 F.3d 1342, 1360 (Fed. Cir. 2012).

In analyzing the obviousness of a combination of prior art elements, it can be important to identify a reason that would have prompted one of skill in the art "to combine . . . known elements in the fashion claimed by the patent at issue." *KSR*, 550 U.S. at 418. A precise teaching directed to the specific subject matter of a challenged claim is not necessary to establish obviousness. *Id.* Rather, "any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed." *Id.* at 420. Accordingly, a party who petitions the Board for a determination of unpatentability based on obviousness must show that "a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success in doing so." *In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1381 (Fed. Cir. 2016) (quotations and citations omitted).

B. Level of Ordinary Skill in the Art

We review the grounds of unpatentability presented in the Petition in view of what a person of ordinary skill in the art would have understood at the time of invention. *Graham*, 383 U.S. at 17. Petitioner asserts a person of ordinary skill in the art at the time of the invention "would have had a Master's degree in computer science, computer engineering, or a related

field, and 2–3 years of practical computer programming or engineering experience, including experience designing or researching parallel processing systems." Pet. 19 (citing Ex. 1003 ¶ 18). Petitioner adds that "[a]dditional graduate education could substitute for professional experience, or significant experience in the field could substitute for formal education." *Id.* at 20. Patent Owner does not dispute or otherwise address the level of ordinary skill in the art. *See generally* PO Resp.; PO Sur-reply.

We adopt Petitioner's definition of the level of ordinary skill in the art and we apply it in our obviousness evaluations below. We are satisfied that this definition comports with the level of skill necessary to understand and implement the teachings of the '090 patent and the asserted prior art.

C. Claim Construction

In an *inter partes* review, we construe patent claims using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. § 282(b). 37 C.F.R. § 42.100(b) (2021). This rule adopts the same claim construction standard used by Article III federal courts, which follow *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (*en banc*), and its progeny. Under that standard, the words of a claim are generally given their "ordinary and customary meaning," which is the meaning the term would have to a person of ordinary skill at the time of the invention, in the context of the entire patent including the specification. *See Phillips*, 415 F.3d at 1312–13. "[W]here a party believes that a specific term has meaning other than its plain meaning, the party should provide a statement identifying a proposed construction of the particular term and

where the disclosure supports that meaning." Consolidated Trial Practice Guide,[7] 44.

Petitioner submits that the term "an array of processor cores of configurable types" (independent claims 1 and 9) should be construed to mean "an indexed set of discrete, reconfigurable hardware elements for processing." Pet. 20 (citing Ex. 1003 ¶¶ 62–66). Patent Owner does not comment on Petitioner's proposed construction, or otherwise address claim construction. *See generally* PO Resp.; PO Sur-reply.

We determine that no aspects of the challenged claims require explicit construction. *See, e.g.*, *Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) ("The Board is required to construe 'only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy.'" (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

D. Obviousness over Chen and Agrawal (Ground 1)

Petitioner contends that claims 1–5 and 7 would have been obvious over Chen and Agrawal. Pet. 22–44. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003). Patent Owner responds to Petitioner's assertions. *See* PO Resp. 1–7; PO Sur-reply 1–12.

1. Overview of Chen (Ex. 1004)

Chen is titled "Configuration-Sensitive Process Scheduling for [Field-Programmable Gate Array ("FPGA")] FPGA-Based Computing Platforms."

[7] *Available at* https://www.uspto.gov/TrialPracticeGuideConsolidated.

Ex. 1004, 1[8]. Chen introduces an operating system "process scheduler, and demonstrate[s] how it can be customized considering the needs of reconfigurable hardware," with a characteristic that the "process scheduler is configuration sensitive, that is, it reuses the current FPGA configuration as much as possible." *Id.*

Chen models its "reconfigurable system as a rectangular array of configurable logic blocks (say NxM) – called CLBs." Ex. 1004, 2. Chen describes that "[e]ach process to be scheduled in this architecture uses a 'rectangular portion' of this array." *Id.* Figure 1, reproduced below, "illustrates a 6x8 array and three processes scheduled on it (they occupy spaces of 2x3, 2x4, and 6x2 CLBs)." *Id.*

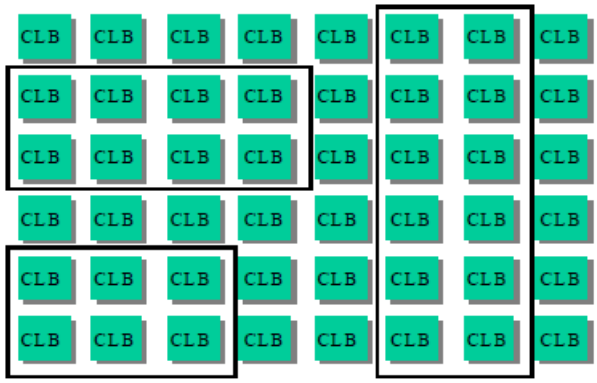


Figure 1 of Chen, illustrates a 6x8 array of configurable logic blocks (such as FPGAs), with rectangles around three running processes occupying different rectangular segments of the array.

Chen discloses that each process to be scheduled is represented by a subtask graph ("STG") and "[e]ach node of this graph represents a process

[8] Exhibit 1004 has no original page numbers in the filed version. We rely on the page numbers inserted by Petitioner in the bottom, middle portion of each page in Exhibit 1004.

code portion (subtask) that will be executed in a single quantum of time once it gets scheduled." Ex. 1004, 2. Chen explains that, "[s]ince one of the objectives of any FPGA-based system is to maximize FPGA utilization, the OS [operating system] scheduler should be able to schedule nodes from the STGs of different applications." *Id.* Chen's "approach is oriented towards maximizing FPGA space utilization by parallel execution of multiple processes." *Id.*

In Chen, "[t]he crucial step in [its] configuration-aware scheduling strategy is assigning an FPGA space for a process that is about to start executing." Ex. 1004, 3. To that end, Chen describes that its "OS scheduler has two important tasks." *Id.* at 3. The first is "[d]etermining a suitable schedule for the processes to be scheduled," and the second, is "[w]hen a process need[s] to start executing, determining where (to which FPGA region(s)) its subtasks should be assigned." *Id.* Chen's "scheduling algorithm is a variant of list scheduling, an algorithm frequently employed by optimizing compilers." *Id.* at 3. More particularly, Chen discloses

> [i]n this scheduling, the next STG node to be scheduled (and to be assigned an FPGA space) is the one that leads to minimum cost (considering the four cases above) among all schedulable nodes. In this way, at each step, this greedy heuristic selects the next STG node to be scheduled.

Id.

2. Overview of Agrawal (Ex. 1005)

Agrawal is titled "Adaptive Scheduling with Parallelism Feedback." Ex. 1005, 1.[9] Agrawal relates to multiprocessor computer system scheduling in a shared multiprogramming environment. *Id.* Agrawal presents "an adaptive task scheduler for multitasked jobs with dependencies that provides continual parallelism feedback to the job scheduler in the form of requests for processors." *Id.* Agrawal focuses on "space-sharing" for parallel jobs, using a two-level strategy: "a kernel-level job scheduler which allots processors to jobs, and a user-level task scheduler which schedules the tasks belonging to a given job onto the allotted processors." *Id.*

Using "parallelism feedback," Agrawal's scheduler "adaptively changes the allotment of processors according to the availability of processors in the current system environment and the job scheduler's administrative policy." Ex. 1004, 2. Agrawal's adaptive task scheduler does not focus on job scheduling, but rather on task scheduling, and, "[i]nstead of using instantaneous parallelism," Agrawal's A-Greedy algorithm "provides parallelism feedback to the job scheduler based on a single summary statistic and the job's behavior on the previous quantum." *Id.* Agrawal describes that its A-Greedy algorithm "provides parallelism feedback using the past behavior of the job" and does "not assume that the job's future parallelism is correlated with its history of parallelism." *Id.* According to Agrawal:

> Between quanta $q - 1$ and q, the task scheduler determines its job's ***desire*** d_q, which is the number of processors the job wants for quantum q. The task scheduler provides the desire d_q to the

[9] Exhibit 1005 has no original page numbers in the filed version. We rely on the page numbers inserted by Petitioner in the bottom, middle portion of each page in Exhibit 1005.

> job scheduler as its parallelism feedback. The job scheduler follows some processor allocation policy to determine the ***processor availability*** p_q, which is the number of processors the job is entitled to get for the quantum q.

Id. Agrawal discloses that "[t]he number of processors the job receives for quantum q is the job's ***allotment*** $a_q = \min\{d_q, p_q\}$, which is the smaller of its desire and the processor availability." *Id.*

Agrawal also describes "instantaneous parallelism," which relies on "the number of processors the job can effectively use at the current moment, as the parallelism feedback to the job scheduler." Ex. 1004, 2. Agrawal discloses that, "[a]lthough using instantaneous parallelism for parallelism feedback is simple, it can cause gross misallocation of processor resources." *Id.* According to Agrawal,

> the parallelism of a job may change substantially during a scheduling quantum, alternating between parallel and serial phases. The sampling of instantaneous parallelism at a scheduling event between quanta may lead the task scheduler to request either too many or too few processors depending on which phase is currently active, whereas the desirable request might be something in between. Consequently, the job may systematically waste processor cycles on the one hand or take too long to complete on the other.

Id. Agrawal acknowledges that A-Greedy "operates nearly as efficiently" as a "task scheduler that uses instantaneous parallelism as feedback," and that a scheduler using instantaneous parallelism wastes no processor cycles. *Id.*

3. Claim 1

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 1 as set forth in the Petition. Pet. 26–39 (citing Exs. 1003, 1004, 1005). We use Petitioner's notations to identify the claim limitations.

i. [1.P] A method for assigning a set of processing tasks to an array of processor cores of configurable types, the method comprising:[10]

Petitioner asserts that, if the preamble is treated as limiting, Chen discloses its contents. Pet. 31–33 (citing Ex. 1003 ¶¶ 71–74). More particularly, Petitioner asserts that "Chen teaches a method for assigning processes and their tasks ('subtasks' or 'STG nodes') to spaces/portions/regions (hereinafter, 'regions') of an FPGA, where the regions of the FPGA are configurable for the different processes." *Id.* at 31–32 (citing Ex. 1004, 1–3, Fig. 1). According to Petitioner, "Chen proposes 'a configuration-sensitive execution strategy; i.e., at each scheduling step, we select (among the executable process) the one that fits well in one or more regions, without needing reconfiguration (which is typically a costly operation).'" *Id.* at 32 (citing Ex. 1004, 3).

ii. Element [1.a] executing time variable subsets of the processing tasks of differing types on their assigned processor cores of matching types, wherein the matching type for the assigned processor core for a given processing task of the set corresponds to a type of a processor core demanded by the given processing task;

Petitioner asserts that Chen discloses "a 'configuration-aware scheduling strategy' that includes 'assigning an FPGA space for a process that is ***about to start executing***.'" Pet. 33 (citing Ex. 1004, 3). According to Petitioner, "Chen teaches 'at each scheduling step,' 'select[ing] (among the executable process) ***the one that fits well in one or more regions***, without

[10] The issue of whether the preamble is limiting needs not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

needing reconfiguration (which is typically a costly operation).'" *Id.* at 34 (citing Ex. 1004, 3).

iii. Element [1.b] for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks

Petitioner asserts

> [t]he combination of Chen and Agrawal discloses for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks.

Pet. 34 (citing Ex. 1003 ¶¶ 94–105; Ex. 1004, 2–3, Fig. 7; Ex. 1005, 2–3). According to Petitioner, one of ordinary skill in the art "would have understood that a CAP is an interval or period of time for which core allocation decisions are made" (*id.* at 35 (citing Ex. 1003 ¶ 94)), and "Chen, explains that '[e]ach node of [a subtask] graph represents a process code portion (subtask) that will be executed ***in a single quantum of time*** once it gets scheduled.'" *Id.* (citing Ex. 1004, 2). Petitioner asserts that, in Chen,

> when a new incoming process needs to be scheduled, the OS needs to find a (rectangular) space for it in the reconfigurable device (this is called space allocation). This space can be one of the spaces that have already been allocated to some other process that could not run concurrently with this incoming process (this is called configuration reuse in this paper).

Id. (quoting Ex. 1004, 2). Petitioner further asserts that Chen discloses that "'each process to be scheduled [is represented] by a subtask graph (henceforth referred to as STG),' and '[e]ach node of this graph represents a process code portion (subtask) that will be executed in a single quantum of

time once it gets scheduled.'" *Id.* at 33 (quoting Ex. 1004, 2). Petitioner argues that, in Chen's configuration-sensitive execution strategy, Chen describes "'select[ing] (among the executable process) the one that fits well in one or more regions, without needing reconfiguration (which is typically a costly operation).'" *Id.* (quoting Ex. 1004, 3). When selecting tasks for scheduling, Chen's scheduler "tak[es] into account, for a given 'STG_{ij} for potential execution on the FPGAconfig-$desc_{ij}$,' the 'configuration required by STG_{ij} and $\{(s_1,s_2),(e_1,e_2)\}_{ij}$,' which defines the region requested for the STG." *Id.* at 35–36 (citing Ex. 1004, 3).

Petitioner also contends that, to the extent "Chen does not teach the use of core allocation periods," Agrawal discloses "scheduling tasks to processors where time is broken into a sequence of equal-size scheduling quanta of length L, which is a CAP." Pet. 36 (citing Ex. 1003 ¶ 96; Ex. 1005, 2). Petitioner asserts that, in Agrawal's scheduling system, "time is broken into a sequence of equal-size scheduling quanta 1, 2, . . . of length L, and the job scheduler is free to reallocate processors between quanta." *Id.* (citing Ex. 1005, 2). According to Petitioner, one of ordinary skill in the art "would have been motivated to combine the mapping of tasks to regions (cores) of Chen with the scheduling quanta of Agrawal to realize the benefits of selecting and using an appropriate scheduling quantum length." *Id.* at 36–37 (citing Ex. 1003 ¶ 102; Ex. 1005, 2). Petitioner explains that one of ordinary skill in the art "would have recognized that, like Chen, Agrawal is directed to a space-sharing system for scheduling tasks to hardware by means of Greedy heuristics," and as such "would have been motivated to modify Chen's method with Agrawal's use of scheduling quanta for making

favorable choices for space allocation across the array." *Id.* at 37 (citing Ex. 1003 ¶ 103; Ex. 1005, 3).

iv. Element [1.c] assigning the selected tasks for execution on the processor cores for the next CAP in a manner to maximize, within the array, a number of processor cores whose assigned processing tasks for a present and the next CAP demands the same type of processor core; and

Petitioner asserts that Chen discloses "a configuration-sensitive execution strategy" that, at each scheduling step, selects the executable process "***that fits well in one or more regions, without needing reconfiguration*** (which is typically a costly operation)." Pet. 38 (citing Ex. 1004, 3). Petitioner further asserts that "Chen indicates that its scheduling algorithm '***tries to reuse the current configuration as much as possible***" (*id.* (citing Ex. 1004, 1)), and notes that Chen's "'***main objective is to maximize configuration reuse, that is, . . . to minimize the number of reconfigurations.***'" *Id.* (quoting Ex. 1004, 2). According to Petitioner, one of ordinary skill in the art

> would have understood that by assigning tasks to cores in a manner that maximizes configuration reuse, or equivalently that minimizes the number of reconfigurations, Chen teaches assigning the selected tasks for execution on the processor cores for the next CAP in a manner to maximize, within the array, a number of processor cores whose assigned processing tasks for a present and the next CAP demands the same type of processor core.

Id. (citing Ex. 1003 ¶ 108).

v. Element [1.d] configuring the array such that the type of any given processor core in the array matches the type of processing task

> *assigned for execution on the given processor core for the next CAP.*

Petitioner asserts that Chen's configuration-sensitive execution strategy "configure[s] regions (cores) of the array in advance . . . based on the scheduling information it has." Pet. 38–39 (citing Ex. 1004, 1, 4). According to Petitioner, Chen "explains that the tasks of a process are assigned to regions (cores) and that it selects and executes a process that 'fits well in the one or more regions'" (*id.* at 39 (citing Ex. 1004, 3)), and "Chen indicates that ['] preconfiguration for the STG node scheduled for time t starts at time t-1.'" *Id.* (citing Ex. 1004, 4).

4. Patent Owner's Arguments

At the outset, Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with respect to limitations 1.P, 1.a, and 1.d. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that Chen teaches these limitations of claim 1.

Patent Owner argues that Petitioner's ground of unpatentability based on Chen and Agarwal is deficient. *See* PO Resp. 1–7; PO Sur-reply 1–12. Patent Owner argues independent claim 1 recites "a very specific method of 'selecting' and 'assigning' 'selected tasks' (i.e., at least two tasks) for execution on an 'array of processor cores of configurable types.'" PO Resp. 1. According to Patent Owner, Petitioner relies on "Chen and Chen alone" as disclosing "the claimed steps of 'assigning' and 'selecting' [limitations 1.b and 1.c, respectively]" (*id.*), but "[n]either the Petition nor the Chase Declaration explain what specifically in Chen discloses selecting

tasks (i.e., at least two tasks) for execution during a single core allocation period," as required by claim 1. *Id.* at 3; *see also* PO Sur-reply 1 ("Chen unambiguously discloses that only one task is selected and assigned for the next core allocation period at a time.").

5. Discussion

After considering both parties' arguments, we are persuaded that the arguments and evidence provided by Petitioner demonstrates that the prior art teaches limitations 1.b and 1.c of claim 1 by a preponderance of the evidence.

Patent Owner contends that "Petitioner's reliance on Chen as disclosing the 'selecting' limitation" relies on Agrawal "only '[t]o the extent that Patent Owner argues that Chen does not teach the use of core allocation periods.'" PO Resp. 5 (citing Pet. 36). And because "Patent Owner does not dispute that Chen discloses core allocation periods," Petitioner's reliance on "Agrawal in connection with the 'selecting' limitation is of no moment and cannot save the Petition's failure to identify a disclosure in Chen of selecting 'tasks' (i.e., at least two) for assignment." *Id.*; *see also id.* at 6 ("[B]y the Petition's own design, Agrawal does not factor into the ground.").

Based on the fully developed trial record, we do not agree with Patent Owner's arguments. Instead, as we stated in our Decision on Institution, we understand Petitioner to be relying "on '[t]he combination of Chen and Agrawal' to address limitation [1.b]." Dec. on Inst. 19 (quoting Pet. 34). In this regard, when addressing limitation [1.b], the Petition identifies that

> The combination of Chen and Agrawal discloses for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least

> in part based on core capacity demand expressions associated with the processing tasks.

Pet. 34 (citing Ex. 1003 ¶¶ 94–105; Ex. 1004, 2–3, Fig. 7; Ex. 1005, 2–3); *cf.* Pet. 33 (citing Ex. 1003 ¶¶ 91–93; Ex. 1004, 2–3, 6, Fig. 2) (identifying that "Chen discloses this element" when mapping Chen to the subject matter of limitation [1.a]"). Mapping the teachings of Chen to limitation [1.b], the Petition states

> To the extent that Patent Owner argues that Chen does not teach the use of core allocation periods, Agrawal discloses, for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks. [Ex. 1003] ¶ 96. In particular, Agrawal teaches scheduling ***tasks*** to processors where time is broken into a sequence of equal-size scheduling quanta of length L, which is a CAP. *Id.*; [Ex. 1005], 2.

Pet. 36 (emphasis added). And, relying on the declaration testimony of Dr. Chase, Petitioner asserts that one of ordinary skill in the art "would have been motivated to combine the mapping of tasks to regions (cores) of Chen with the scheduling quanta of ***Agrawal to realize the benefits of selecting*** and using an appropriate scheduling quantum length." *Id.* at 36–37 (emphasis added) (citing Ex. 1003 ¶ 102; Ex. 1005, 2).

In addition, relying on the declaration testimony of Dr. Chase, Petitioner asserts that one of ordinary skill in the art "would have been motivated to combine the mapping of tasks to regions (cores) of Chen with the scheduling quanta of Agrawal to realize the benefits of selecting and using an appropriate scheduling quantum length." Pet. 36–37 (citing Ex. 1003 ¶ 102; Ex. 1005, 2). Also relying on the declaration testimony of

Dr. Chase, Petitioner asserts that one of ordinary skill in the art "would have recognized that, like Chen, Agrawal is directed to a space-sharing system for scheduling tasks to hardware by means of its greedy task scheduler. *Id.* at 37 (citing Ex. 1003 ¶ 103; Ex. 1005, 3).

Based on the fully developed trial record, Petitioner shows persuasively that Agrawal's task scheduler schedules, i.e., selects, tasks for execution, and we credit Dr. Chase's declaration testimony that one of ordinary skill in the art "would have been motivated to modify Chen's method with Agrawal's use of scheduling quanta for making favorable choices for space allocation across the array." *Id.* (citing Ex. 1003 ¶ 103). Petitioner provides several persuasive reasons why one of ordinary skill in the art would have been motivated to combine Chen and Agrawal, with a reasonable expectation of success. *See* Pet. 26–31 (citing Ex. 1003 ¶¶ 96–102).[11] Thus, Patent Owner's arguments that the Petitioner relies on Chen, and Chen alone to address limitation [1.b] are unavailing.

Patent Owner next argues that Chen fails to disclose "'selecting' and 'assigning' more than one task at a time for execution during the 'next' core allocation period" because "Chen unambiguously discloses that only one task is selected and assigned for the next core allocation period at a time." PO Sur-reply 1. According to Patent Owner,

> Chen's selection of a single subtask at a time for assignment makes sense given Chen's stated purpose of maximizing configuration reuse. *See* Ex. 1004 at 2. Chen

[11] Patent Owner neither provides any argument against the combined teachings of Chen and Agrawal, nor explains why the combination would have been beyond the level of ordinary skill, nor disputes that one with ordinary skill in the art would have had a reasonable expectation of success in combining the teachings.

> discloses reconfigurable FPGA processors and explains that reconfiguration "is typically a costly option." *Id.* at 3. To avoid that cost, Chen prioritizes configuration reuse over all else. As such, the subtask Chen selects for assignment will be the one that avoids processor reconfiguration. Contrast this with a system that prioritizes the avoidance of idle processor resources. Such a system would select multiple subtasks for execution at a time to ensure all processor resources are utilized.

PO Resp. 4. Patent Owner also asserts that "Petitioner raises several arguments for the first time in its Reply attempting to backfill the Petition." PO Sur-reply 3. Based on the fully developed trial record, we do not agree with Patent Owner's arguments.

Instead, we agree with Petitioner that "the Petition explains how Chen teaches or at least renders obvious selecting and assigning more than a single subtask." Pet. Reply 6. For example, the Petition identifies that Chen discloses that "each 'process to be scheduled [is represented] by a subtask graph (henceforth referred to as STG),' and '[e]ach node of this graph represents a process code portion (subtask) that will be executed in a single quantum of time once it gets scheduled.'" Pet. 35 (quoting Ex. 1004, 2). Chen describes that, because "one of the objectives of any FPGA-based system is to maximize FPGA utilization, the OS scheduler should be able to schedule nodes from the STGs of different applications." Ex. 1004, 2. Chen also makes clear that its approach "is oriented towards maximizing FPGA space utilization by parallel execution of multiple processes." *Id.* Thus, we agree with Petitioner that "Chen's disclosure makes clear that multiple subtasks would be selected and assigned at a time." Pet. Reply 7.

In making this determination, we credit Dr. Chase's unrebutted declaration testimony that "[b]oth Chen and Agrawal represent the tasks of a

given parallel program (running as a job or a process) using a dependence graph of tasks to capture time-varying sets of unit-time tasks that are ready to execute in parallel at each point in time." Ex. 1003 ¶ 101 (citing Ex. 1004, 2, Fig. 2). Patent Owner argues that Dr. Chase's declaration testimony should be discounted because the fact that "tasks are able to execute in parallel does not mean that Chen selects and assigns more than one task at a time for execution during the next CAP." PO Sur-reply 5. Patent Owner asserts that rather than selecting and assigning more than one task at a time,

> Chen discloses selecting and assigning a single task at a time for the next CAP based on the principle of maximizing configuration reuse. According to Chen, "the one" task that is selected and assigned can then execute in parallel with other tasks that were selected and assigned one at a time.

Id. (citing Ex. 1004, 3). Patent Owner further asserts that "selection and assignment of a single subtask for execution for the next CAP" makes perfect sense in the content of Chen "[b]ecause Chen implements a configuration reuse approach to scheduling above all else, individual selection and assignment of tasks." *Id.* at 7.

Based on the fully developed trial record, we disagree with Patent Owner's contention that one of ordinary skill in the art would have understood Chen's configure-sensitive scheduling algorithm as being limited to the selection and assignment of a single task for execution in the next CAP. In this regard, Chen discloses that its "OS scheduler should be able to schedule nodes from the STGs of different applications" and "is oriented towards maximizing FPGA space utilization by parallel execution of multiple processes." Ex. 1004, 2. Based on these cited disclosures in Chen,

we find that the scheduler is not limited to scheduling a single task for an upcoming period, but instead the scheduler optimizes scheduling by scheduling more than one task to execute in parallel if more than one task can execute in the next period. *See* Ex. 1003 ¶ 101 (citing Ex. 1004, 2–3; Ex. 1005, 1–2 ("The strategies of Chen and Agrawal both use space sharing techniques and greedy heuristics to achieve efficient space utilization.").

Patent Owner argues that "[s]electing and assigning multiple tasks at a time would fundamentally alter the process of determining a 'suitability factor' [in Chen,] and result in an unmanageable and overly complex system." PO Sur-reply 10. However, Chen discloses that "the next STG node to be scheduled (and to be assigned an FPGA space) is the one that leads to minimum cost . . . among all schedulable nodes. In this way, at each step, this greedy heuristic selects the next STG node to be scheduled," because this indicates Chen does not stop scheduling after it schedules a single task for an upcoming period. *Id.* at 3; *see also id.* at 2 (Chen describing that its "approach is oriented towards maximizing FPGA space utilization by parallel execution of multiple processes."). As such, we disagree with Patent Owner's suggestions that selecting and assigning multiple tasks at the same time would result in Chen becoming "an unmanageable and overly complex system." Rather, we agree with Petitioner that one of ordinary skill in the art would have understood that, in Chen, "multiple subtasks would be selected and assigned at a time in order to maximally utilize available space while also prioritizing reuse."). Pet. Reply 7; *see also id.* Ex. 1004, 2 ("Therefore, our approach is oriented towards maximizing FPGA space utilization by parallel execution of multiple processes.").

Having reviewed all of Petitioner's assertions regarding these limitations, as well as all supporting evidence, we determine on this complete record presented that Petitioner has persuasively shown that the combination of Chen and Agrawal teaches "'selecting' and 'assigning' more than one task at a time for execution during the 'next' core allocation period" (PO Resp. 1), as called for by limitations [1.b] and [1.c].

a) Conclusion for Claim 1

Petitioner has persuasively shown that the combination of Chen and Agrawal teaches all limitations of claim 1. Petitioner also has put forth persuasive reasons for combining these references. *See* Pet. 26–31 (citing Ex. 1003 ¶¶ 101–104). Thus, regarding claim 1, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Chen and Agrawal.

6. Dependent Claims 2–5 and 7

Petitioner contends that the additional limitations of dependent claims 2–5 and 7 are rendered obvious by the combination of Chen and Agrawal. Pet. 39–44. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003).

i. [2] The method of claim 1 wherein the configuring is done based at least in part on a type of processor core demanded by the processing task assigned for execution on the given core.

Petitioner asserts that the combination Chen and Agrawal renders claim 2 obvious. Pet. 39–40 (citing Ex. 1003 ¶¶ 115–116; Ex. 1004, 3). More particularly, Petitioner asserts that "Chen teaches assigning '(among the executable process) the one that fits well in one or more regions.'" *Id.* at 39 (citing Ex. 1004, 3). According to Petitioner, Chen's process

"consider[s] a subtask graph 'STG_{ij} for potential execution on the FPGA,' where 'config-$desc_{ij}$ [is] the configuration required by STG_{ij} and $\{(s_1,s_2),(e_1,e_2)\}_{ij}$ [is] the region requested.'" *Id.* at 40 (citing Ex. 1004, 3). Petitioner explains that "[a]ll of this information, including the configuration required by the subtask graph (which represents a process to be scheduled), is taken into account in the scheduling algorithm to minimize reconfiguration latency." *Id.* (citing Ex. 1004, 3).

ii. [3] The method of claim 1 wherein configuring the array involves configuring a processor core to provide a direct hardware logic implementation of the processing task assigned for processing by the given core.

Petitioner asserts that the combination Chen and Agrawal renders claim 3 obvious. Pet. 40–41 (citing Ex. 1003 ¶¶ 119–120 (citing Ex. 1004)). More particularly, Petitioner asserts that one of ordinary skill in the art "would have understood that configuration of an FPGA device region to execute a task, as in Chen, would have involved a direct hardware logic implementation." *Id.* at 40 (citing Ex. 1003 ¶ 119). Petitioner explains, that, "if a single region (core) is assigned to a single subtask of a process and configured to execute the subtask, as in Chen, then there is necessarily a direct hardware logic implementation of the subtask's function." *Id.* (citing Ex. 1003 ¶ 119).

iii. [4] The method of claim 3 wherein the direct hardware logic implementation performs an information processing function of the processing task without executable program instructions.

Petitioner asserts that the combination Chen and Agrawal renders claim 4 obvious. Pet. 41 (citing Ex. 1003 ¶¶ 123–124; Ex. 1004, 1). More particularly, Petitioner points out that Chen identifies that "[r]econfigurable computing systems have shown the ability to greatly accelerate program

execution, thereby providing a high-performance alternative to software-only implementations and a programmable alternative to ASICs." *Id.* (citing Ex. 1004, 1). In this regard, Petitioner asserts that one of ordinary skill in the art "would have understood that configuring a region (core) of an FPGA to execute a task of an application, as in Chen, would have resulted in a direct hardware logic implementation that performs various processing, including an information processing function, of the processing task." *Id.* (citing Ex. 1003 ¶ 123). Petitioner further notes that one of ordinary skill in the art "would have understood that once configured to implement a function, the FPGA implements the function without executable program instructions, as in an application-specific integrated circuit or ASIC." *Id.* (citing Ex. 1003 ¶ 123).

iv. [5] The method of claim 1 wherein the core capacity demand expressions, of at least one of the processing tasks, includes an indication of a type of a processor core associated with the at least one processing task.

Petitioner asserts that the combination Chen and Agrawal renders claim 5 obvious. Pet. 42–43 (citing Ex. 1003 ¶¶ 127–128; Ex. 1004, 3). More particularly, Petitioner asserts that "Chen considers a subtask graph (representing a process) 'STG_{ij} for potential execution on the FPGA,' where the configuration required by the STG (config-desc) and the region requested by the STG are taken into account." *Id.* at 42 (citing Ex. 1004, 3). Petitioner further asserts that "Chen discloses that each task (subtask or STG node) has a corresponding configuration description (config-desc) that matches against the configuration descriptions contained within the context entries that describe the candidate regions (cores) of the FPGA." *Id.* (citing Ex. 1004, 3). Petitioner explains that one of ordinary skill in the art would

have understood "that the config-desc corresponding to a task represents a core type demanded by that task" and known "that a config-desc corresponding to an FPGA region represents the configured type of the FPGA region and is equivalent to a configured type of a reconfigurable processor core." *Id.* (citing Ex. 1003 ¶ 127). Petitioner last asserts that "Chen discloses assigning FPGA regions (cores) based on suitability of a particular process for a particular FGPA region." *Id.* at 42–43 (citing Ex. 1004, 3).

v. *[7] The method of claim 1 wherein a processing task of the set is one of: a function, a procedure, an actor, a thread, or an instance of a software program.*

Petitioner asserts that the combination Chen and Agrawal renders claim 7 obvious. Pet. 43–44 (citing Ex. 1003 ¶¶ 131–133; Ex. 1004, 2, 4–5; Ex. 1012 ¶ 3; Ex. 1013, Abstract). More particularly, Petitioner asserts that "the processes in Chen are described as being at least functions of an application" (*id.* at 43 (citing Ex. 1004, 2, 4–5)), and as such, one of ordinary skill in the art "would have understood that the tasks in Chen are at least functions." *Id.* (citing Ex. 1003 ¶ 131). Petitioner further asserts that "Chen discloses that a task (subtask) is an executable code portion." *Id.* (citing Ex. 1004, 2). According to Petitioner, one of ordinary skill in the art "would have understood that a task in execution is an instance of a program, and that a task or code portion executing on a general-purpose processor runs a sequence of instructions on a core" (*id.* at 44 (citing Ex. 1003 ¶ 132)), and known that "[a] sequence of instructions that can be scheduled on a core is a thread." *Id.* (citing Ex. 1012 ¶ 3; Ex. 1013, Abstract). Thus, Petitioner concludes that one of ordinary skill in the art "would have understood that

the tasks in Chen may also be understood to comprise threads." *Id.* (citing Ex. 1003 ¶ 132).

a) Conclusion for Dependent Claims 2–5 and 7

Patent Owner does not challenge Petitioner's contentions with respect to the limitations of dependent claims 2–5 and 7, and Petitioner's contentions are supported by the cited evidence. For the reasons given by Petitioner and summarized above, we are persuaded that Petitioner has shown by a preponderance of the evidence that the limitations of each of claims 2–5 and 7 are rendered obvious by the combination of Chen and Agrawal. Therefore, we are persuaded that Petitioner has established by a preponderance of the evidence that claims 2–5 and 7 are unpatentable under § 103(a) over Chen and Agrawal.

7. Conclusion for Ground 1

For the foregoing reasons, we conclude that Petitioner has shown by a preponderance of the evidence that claims 1–5 and 7 are unpatentable under § 103(a) over Chen and Agrawal.

E. Obviousness over Chen, Agrawal, and Compton (Ground 2)

Petitioner contends that claims 3 and 4 would have been unpatentable as obvious over Chen, Agrawal, and Compton. Pet. 44–50. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003).

1. Overview of Compton (Ex. 1006)

Compton is titled "Reconfigurable Computing: A Survey of Systems and Software," and "presents a survey of current research in hardware and software systems for reconfigurable computing, as well as techniques that

specifically target run-time reconfigurability." Ex. 1006, 171, 173. Compton provides "an introduction to this rapidly evolving field, bringing interested readers quickly up to speed on developments from the last half-decade." *Id.* at 173. Compton discusses systems that incorporate "some form of hardware programmability—customizing how the hardware is used using a number of physical control points. These control points can then be changed periodically in order to execute different applications using the same hardware." *Id.* at 173–74. Compton explains that "[r]econfigurable computing systems use FPGAs or other programmable hardware to accelerate algorithm execution by mapping compute-intensive calculations to the reconfigurable substrate." *Id.* at 175.

i. [3] The method of claim 1 wherein configuring the array involves configuring a processor core to provide a direct hardware logic implementation of the processing task assigned for processing by the given core.

Petitioner asserts that the combination Chen, Agrawal, and Compton renders claim 3 obvious. Pet. 48–49 (citing Ex. 1003 ¶¶ 136–141; Ex. 1006, 2–4). More particularly, Petitioner asserts that "Compton states that reconfigurable devices include FPGAs, such as the ones disclosed in Chen and Agrawal." *Id.* at 48 (citing Ex. 1006, 2). Petitioner asserts that "Compton explains that FPGAs 'contain an array of computational elements whose functionality is determined through multiple programmable configuration bits,' where '[t]hese elements, sometimes known as logic blocks, are connected using a set of routing resources that are also programmable.'" *Id.* at 48–49 (citing Ex. 1006, 2).

ii. [4] The method of claim 3 wherein the direct hardware logic implementation performs an information processing function of the processing task without executable program instructions.

Petitioner asserts that the combination Chen, Agrawal, and Compton renders claim 4 obvious. Pet. 49–50 (citing Ex. 1003 ¶¶ 144–145; Ex. 1006, 5). More particularly, Petitioner asserts that "Compton explains that '[r]econfigurable computing systems use FPGAs or other programmable hardware to accelerate algorithm execution by mapping compute-intensive calculations to the reconfigurable substrate.'" *Id.* at 50 (citing Ex. 1006, 5).

iii. Conclusion for Dependent Claims 3 and 4

Patent Owner does not challenge Petitioner's contentions with respect to these limitations, and Petitioner's contentions are supported by the cited evidence. Petitioner also has put forth persuasive reasons for combining these references. *See* Pet. 45–48 (Ex. 1003 ¶¶ 139–140). For the reasons given by Petitioner and summarized above, we are persuaded that Petitioner has shown by a preponderance of the evidence that the limitations of each of claims 3 and 4 are rendered obvious by the combination of Chen, Agrawal, and Compton. Therefore, we are persuaded that Petitioner has established by a preponderance of the evidence that claims 3 and 4 are unpatentable under § 103(a) over Chen, Agrawal, and Compton.

2. Conclusion for Ground 2

For the foregoing reasons, we conclude that Petitioner has shown by a preponderance of the evidence that claims 3 and 4 are unpatentable under § 103(a) over Chen, Agrawal, and Compton.

F. Obviousness over Chen, Agrawal, and Brent (Ground 3)

Petitioner contends that claims 1, 2, 5–7, and 9–15 would have been obvious over Chen, Agrawal, and Brent. Pet. 50–74. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003).

1. Overview of Brent (Ex. 1007)

Brent is titled "Highly Distributed Parallel Processing on Multi-Core Device." Ex. 1007, code (54). Brent relates to multi-core processing systems, and discloses a highly distributed multi-core system with an adaptive scheduler. *Id.* ¶¶ 3, 8.

Figure 1 of Brent, reproduced below, shows an exemplary multi-core system with an adaptive scheduler.

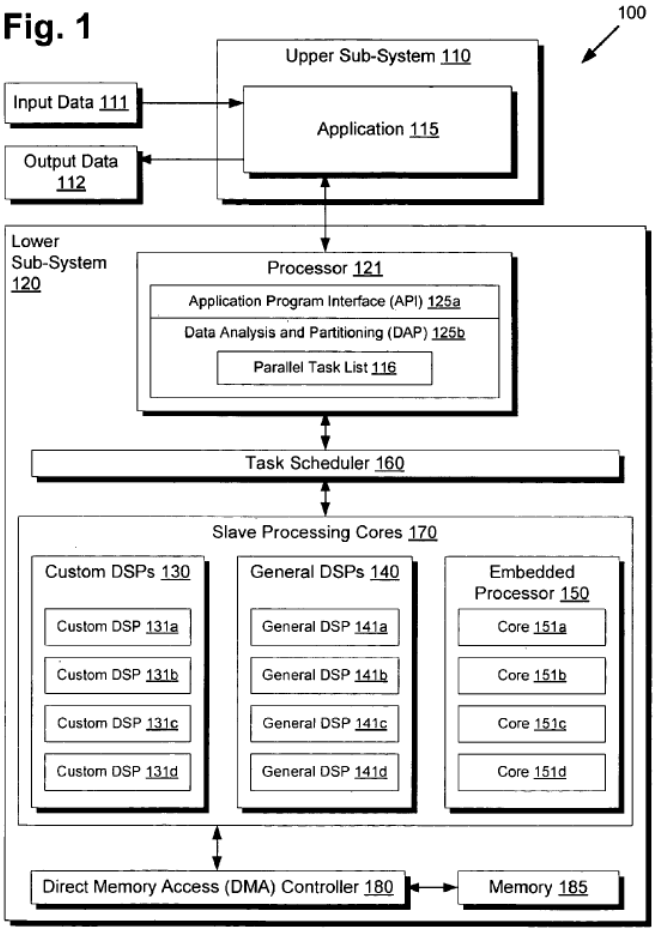


Figure 1 shows, among other things, input data 111, application 115, and processor 121 which creates task list 116, which is passed to task scheduler 160 for execution on slave processing cores 170. Brent discloses that "slave processing cores 170 may include several different types of processing cores. Custom digital signal processors (DSPs) 130 include custom DSPs 131*a*–131*d*, which may have limited instruction sets optimized for specific processing patterns." Ex. 1007 ¶ 16. Brent describes that "each task may be assigned a preferred core type for optimal execution." *Id.* ¶ 18.

2. Petitioner's Assertions for Claim 1

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 1 as set forth in the Petition. Pet. 58–60

(citing Exs. 1003, 1004, 1005, 1007). We use Petitioner's notations to identify the claim limitations.

i. [1.P] A method for assigning a set of processing tasks to an array of processor cores of configurable types, the method comprising:[12]

Petitioner asserts that Chen discloses the contents of the preamble of claim 1 for the reasons discussed above with respect to Ground 1. Pet. 58 (citing Pet. 31–33).

ii. Element [1.a] executing time variable subsets of the processing tasks of differing types on their assigned processor cores of matching types, wherein the matching type for the assigned processor core for a given processing task of the set corresponds to a type of a processor core demanded by the given processing task;

Petitioner asserts that Chen discloses limitation [1.a] for the reasons discussed above with respect to Ground 1. Pet. 59 (citing Pet. 33–34). However, to the extent Chen may not disclose limitation [1.a], Petitioner asserts that Brent does. *Id.* (citing Ex. 1003 ¶¶ 151–157; Ex. 1007 ¶¶ 18, 23, 24, 27, 29–30, Figs. 2–3). More particularly, Petitioner identifies that "Brent states that 'each task may be assigned a preferred core type for optimal execution,'" and "teaches the use of a parallel task list 216 that shows a number of processing tasks and the type of processor core demanded by each processing task. *Id.* (citing Ex. 1007 ¶ 18).

iii. Element [1.b] for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores

[12] The issue of whether the preamble is limiting needs not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks

Petitioner asserts that the combination of Chen and Agrawal discloses limitation [1.b] for the reasons discussed above with respect to Ground 1. Pet. 60 (citing Pet. 34–37).

iv. Element [1.c] assigning the selected tasks for execution on the processor cores for the next CAP in a manner to maximize, within the array, a number of processor cores whose assigned processing tasks for a present and the next CAP demands the same type of processor core; and

Petitioner asserts that the combination of Chen and Agrawal discloses limitation [1.c] for the reasons discussed above with respect to Ground 1. Pet. 60 (citing Pet. 37–38).

v. Element [1.d] configuring the array such that the type of any given processor core in the array matches the type of processing task assigned for execution on the given processor core for the next CAP.

Petitioner asserts that the combination of Chen and Agrawal discloses element [1.d] for the reasons discussed above with respect to Ground 1. Pet. 60 (citing Pet. 38–39).

a) Patent Owner's Arguments

Patent Owner disputes neither Petitioner's analysis of limitations 1.P, 1.a, 1.b, 1.c, and 1.d, as it pertains to Ground 3, nor Petitioner's rationale for combining Brent with Chen and Agrawal. Otherwise, Patent Owner relies on the same arguments we have discussed with respect to the combination of Chen and Agrawal, as set forth in Ground 1. As we explain above in our analysis of Ground 1, we do not agree with Patent Owner's arguments. *See supra* § II.D.5.

b) Conclusion for Claim 1

Petitioner has persuasively shown that the combination of Chen, Agrawal, and Brent teaches all limitations of claim 1. Petitioner also has put forth persuasive reasons for combining these references with a reasonable expectation of success. *See* Pet. 55–58 (citing Ex. 1003 ¶¶ 155–157). We credit Dr. Chase's opinion that one of ordinary skill in the art "would have found it obvious to implement the Chen-Agrawal system with Brent's use of a 'parallel task list' for selecting and assigning tasks to appropriate cores for execution." *Id.* at 56 (citing Ex. 1003 ¶ 155). We also credit Petitioner's explanation, relying on Dr. Chase's declaration testimony, that one of ordinary skill in the art "would have naturally looked to Brent for an example of a system where tasks include explicit indications regarding the best-fitting or more suitable region or core type for the respective task." *Id.* at 56–57 (citing Ex. 1003 ¶ 155). Thus, regarding claim 1, we determine that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Chen, Agrawal, and Brent.

3. Dependent Claims 2 and 5–7

Petitioner contends that the additional limitations of dependent claims 2 and 5–7 are rendered obvious by the combination of Chen, Agrawal, and Brent. Pet. 61–69. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003).

i. Element [2] The method of claim 1 wherein the configuring is done based at least in part on a type of processor core demanded by the processing task assigned for execution on the given core.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 2 obvious. Pet. 61 (citing Ex. 1003 ¶¶ 166–167; Ex. 1007

¶¶ 18–19, Fig. 2). Relying on the declaration testimony of Dr. Chase, Petitioner states the following:

> Brent explains that "Tasks Task1, Task2, Task3, Task4, and Task5 are ordered by priority in the parallel task list 216, where "each task [is] assigned a preferred core type for optimal execution." [Ex. 1007 ¶ 18]. For example, Task1 and Task4 are assigned to custom DSP 230 (cDSP), and Task2 and Task5 are assigned to general DSP 240 (gDSP). *Id.* The "parallel task list 216 may be used as base guidelines for task scheduler 260." [*Id.* ¶ 19].

Pet. 61 (citing Ex. 1003 ¶ 166).

ii. Element [5] The method of claim 1 wherein the core capacity demand expressions, of at least one of the processing tasks, includes an indication of a type of a processor core associated with the at least one processing task.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 5 obvious. Pet. 61–63 (citing Ex. 1003 ¶¶ 170–171; Ex. 1007 ¶¶ 18, 29, Fig. 2). More particularly, Petitioner asserts that "Brent teaches a 'Parallel Task List 216' that includes a series of core capacity demand expressions for five different tasks." *Id.* at 62 (citing Ex. 1007 ¶¶ 18, 29, Fig. 2). Petitioner explains that "[e]ach of the core capacity demand expressions indicate a type of processor associated with the respective processing task" and "each task may be assigned a preferred core type for optimal execution." *Id.* (citing Ex. 1007 ¶ 18).

iii. Claim 6

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 6 obvious. Pet. 63–69 (citing Ex. 1003 ¶¶ 173–180; Ex. 1004, 2, 4–5; Ex. 1007 ¶¶ 15, 18–19, 25–27, Figs. 1–3). We use Petitioner's notations to identify the limitations of claim 6.

a. Element [6.P] The method of claim 1 wherein:

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 1 obvious. Pet. 63 (citing Pet. 59–60).

b. Element [6.a] each of the processing tasks belongs to one of a group of programs;

Petitioner asserts that Chen discloses the subject matter of limitation [6.a]. Pet. 63–64 (citing Ex. 1003 ¶ 174). More particularly, Petitioner asserts that "the processes in Chen are described as being 'extracted from a given embedded application,' and Chen includes an example with tasks from 'two large, real-life embedded applications: *encr* and *usonic*.'" *Id.* (citing Ex. 1004, 2, 4–5). Petitioner further asserts that "Brent teaches an adaptive scheduler that assigns parallel tasks to execute across several types of processing cores" and "[t]he parallel tasks belong to various applications." *Id.* at 64 (citing Ex. 1003 ¶ 175; Ex. 1007, Abstract, ¶ 30, Fig. 1).

c. Element [6.b] the selecting is done at least in part based on respective capacity demand indications by individual programs among the group; and

Petitioner asserts that Brent discloses the subject matter of limitation [6.b]. Pet. 64–68 (citing Ex. 1003 ¶¶ 177–178; Ex. 1007 ¶¶ 15, 18–19, 25–26, 27, Figs. 1–3). More particularly, Petitioner asserts that "Brent teaches that selecting of tasks for execution is based on task priorities from a 'Parallel Task List 216,' where the selected tasks are determined to have currently accessible data inputs." *Id.* at 64 (citing Ex. 1007 ¶¶ 18–19, 25–26, 27, Figs. 1–3). Petitioner explains that one of ordinary skill in the art "would have understood that the Parallel Task List 216 relates to a particular application (e.g., application 115)" (*id.* at 65 (citing Ex. 1003 ¶ 177)), and "Brent discloses that an application 115 can 'direct the generation of the

parallel task list 116,' by invoking a data analysis and partitioning (DAP) module 125b that executes on processor 121 via an application program interface (API) 125a." *Id.* at 65–66 (citing Ex. 1007 ¶ 15).

> *d. Element [6.c] the core capacity demand indications, of at least one of the programs, are based on: i) an amount of input data that the at least one program presently has available for processing, ii) a number of tasks of the at least one program that presently have input data available for processing, or iii) a number of instances of the at least one program that presently have input data available for processing.*

Petitioner asserts that Brent discloses the subject matter of limitation [6.c]. Pet. 68–69 (citing Ex. 1003 ¶¶ 179–180; Ex. 1007 ¶¶ 26–27, Figs. 2–3). More particularly, Petitioner asserts that "Brent teaches selecting tasks to execute based at least on an amount of input data that the at least one program (application) has available for processing, *e.g.*, all of its input data or less than all of its input data." *Id.* at 68 (citing Ex. 1007 ¶¶ 26–27).

iv. Element [7] The method of claim 1 wherein a processing task of the set is one of: a function, a procedure, an actor, a thread, or an instance of a software program.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 7 obvious. Pet. 69. More particularly, Petitioner asserts that Chen discloses the subject matter of limitation [7.0] for the reasons discussed above with respect to Ground 1. *Id.* (citing Pet. 43–44; Ex. 1003 ¶ 183).

vi. Conclusion for Dependent Claims 2 and 5–7

Patent Owner does not challenge Petitioner's contentions with respect to the limitations of dependent claims 2 and 5–7, and Petitioner's contentions are supported by the cited evidence. For the reasons given by Petitioner and summarized above, we are persuaded that Petitioner has

shown by a preponderance of the evidence that the limitations of each of claims 2 and 5–7 are rendered obvious by the combination of Chen, Agrawal, and Brent. Therefore, we are persuaded that Petitioner has established by a preponderance of the evidence that claims 2 and 5–7 are unpatentable under § 103(a) over Chen, Agrawal, and Brent.

4. Petitioner's Assertions for Claim 9

We begin by reproducing, for context, Petitioner's assertions regarding the limitations of claim 9 as set forth in the Petition. Pet. 69–71 (citing Ex. 1003). We use Petitioner's notations to identify the claim limitations.

i. [9.P] A method for assigning a set of processing tasks to an array of processor cores of configurable types comprising:[13]

Petitioner asserts that Chen discloses the contents of the preamble of claim 1 for the reasons discussed above with respect to [1.P]. Pet. 69 (citing Pet. 58; Ex. 1003 ¶ 187).

ii. Element [9.a] executing time variable subsets of the processing tasks of differing types on their assigned processor cores of matching types, wherein the matching type for the assigned processor core for a given processing task of the set corresponds to a type of a processor core demanded by the given processing task;

Petitioner asserts that Chen discloses limitation [9.a] for the reasons discussed above with respect to claim limitation [1.a]. Pet. 69 (citing Pet. 59–60; Ex. 1003 ¶ 189).

[13] The issue of whether the preamble is limiting needs not be resolved because, regardless of whether the preamble is limiting, Petitioner has persuasively shown that the cited art teaches the preamble.

iii. Element [9.b] for each of a series of core allocation periods (CAPs), selecting, from the set of processing tasks, specific tasks, referred to as selected tasks, for execution on the processor cores for a next CAP at least in part based on core capacity demand expressions associated with the processing tasks; and;

Petitioner asserts that the combination of Chen and Agrawal discloses limitation [9.b] for the reasons discussed above with respect to claim limitation [1.b]. Pet. 70 (citing Pet. 60; Ex. 1003 ¶ 191).

iv. Element [9.c] assigning the selected tasks for execution on the processor cores for the next CAP in a manner to maximize, within the array, a number of processor cores whose assigned processing tasks for a present and the next CAP demands the same type of processor core, wherein:

Petitioner asserts that the combination of Chen and Agrawal discloses limitation [9.c] for the reasons discussed above with respect to claim limitation [1.c]. Pet. 70 (citing Pet. 60; Ex. 1003 ¶ 193).

v. Element [9.c.i] each of the processing tasks belongs to one of a group of programs;

Petitioner asserts that the combination of Chen, Agrawal, and Brent discloses limitation [9.c.i] for the reasons discussed above with respect to claim limitation [6.a]. Pet. 70 (citing Pet. 63–64; Ex. 1003 ¶ 195).

vi. Element [9.c.ii] the selecting is done at least in part based on respective capacity demand indications by individual programs among the group; and

Petitioner asserts that the combination of Chen, Agrawal, and Brent discloses limitation [9.c.ii] for the reasons discussed above with respect to claim limitation [6.b]. Pet. 70 (citing Pet. 64–68; Ex. 1003 ¶ 197).

vii. Element [9.c.iii] the core capacity demand indications, of at least one of the programs, are based on: i) an amount of input data that the at least one program presently has available for processing, ii) a number of tasks of the at least one program that presently have

input data available for processing, or iii) a number of instances of the at least one program that presently have input data available for processing.

Petitioner asserts that the combination of Chen, Agrawal, and Brent discloses limitation [9.c.iii] for the reasons discussed above with respect to claim limitation [6.c]. Pet. 71 (citing Pet. 68–69; Ex. 1003 ¶ 199).

a) Patent Owner's Arguments

Patent Owner does not present arguments in its Response or Sur-reply addressing the merits of Petitioner's assertions with respect to limitations 9.P, 9.a, 9.b, and 9.c.i–iii. *See generally* PO Resp.; PO Sur-reply. Based on our review and consideration of the fully developed trial record, we determine that Petitioner has shown that the combination of Chen Agrawal, and Brent discloses these limitations. *See supra* §§ II.F.2–3.

b) Conclusion for Claim 9

Petitioner has persuasively shown that the combination of Chen, Agrawal, and Brent teaches all limitations of claim 9. Petitioner also has put forth persuasive reasons for combining these references with a reasonable expectation of success. *See* Pet. 55–58 (citing Ex. 1003 ¶¶ 155–157). We credit Dr. Chase's opinion that one of ordinary skill in the art "would have found it obvious to implement the Chen-Agrawal system with Brent's use of a 'parallel task list' for selecting and assigning tasks to appropriate cores for execution." *Id.* at 56 (citing Ex. 1003 ¶ 155). We also credit Petitioner's explanation, relying on Dr. Chase's declaration testimony, that one of ordinary skill in the art "would have naturally looked to Brent for an example of a system where tasks include explicit indications regarding the best-fitting or more suitable region or core type for the respective task." *Id.* at 56–57 (citing Ex. 1003 ¶ 155). Thus, regarding claim 9, we determine

that Petitioner has shown, by a preponderance of the evidence, that the claimed invention as a whole would have been obvious over the combination of Chen, Agrawal, and Brent.

5. Dependent Claims 10–15

Petitioner contends that the additional limitations of dependent claims 10–15 are rendered obvious by the combination of Chen, Agrawal, and Brent. Pet. 71–74. Petitioner also relies on the testimony of Dr. Chase to support its arguments. *Id.* (citing Ex. 1003).

i. Element [10] The method of claim 9, further comprising configuring the array such that the type of any given processor core in the array matches the type of processing task assigned for execution on the given processor core for the next CAP.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 10 obvious for the reasons discussed above with respect to limitation [1.d]. Pet. 71 (citing Pet. 38–39; Ex. 1003 ¶ 203).

ii. Element [11] The method of claim 10 wherein the configuring is done based at least in part on a type of processor core demanded by the processing task assigned for execution on the given core.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 11 obvious for the reasons discussed above with respect to limitation [2]. Pet. 72 (citing Pet. 61; Ex. 1003 ¶ 207).

iii. Element [12] The method of claim 10 wherein configuring the array involves configuring a processor core to provide a direct hardware logic implementation of the task assigned for processing by the given core.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 12 obvious for the reasons discussed above with respect to limitation [3]. Pet. 72 (citing Pet. 40–41; Ex. 1003 ¶ 211).

iv. Element [13] The method of claim 12 wherein the direct hardware logic implementation performs an information processing function of the processing task without executable program instructions.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 13 obvious for the reasons discussed above with respect to limitation [4]. Pet. 73 (citing Pet. 41; Ex. 1003 ¶ 215).

v. Element [14] The method of claim 9 wherein the core capacity demand expressions, of at least one of the processing tasks, includes an indication of a type of a processor core associated with the at least one processing task.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 14 obvious for the reasons discussed above with respect to limitation [5]. Pet. 73 (citing Pet. 61–63; Ex. 1003 ¶ 219).

vi. Element [15] The method of claim 9 wherein a processing task of the set is one of: a function, a procedure, an actor, a thread, or an instance of a software program.

Petitioner asserts that the combination of Chen, Agrawal, and Brent renders claim 15 obvious for the reasons discussed above with respect to limitation [7]. Pet. 74 (citing Pet. 43–44; Ex. 1003 ¶ 223).

vii. Conclusion for Dependent Claims 10–15

Patent Owner does not challenge Petitioner's contentions with respect to these limitations, and Petitioner's contentions are supported by the cited evidence. For the reasons given by Petitioner and summarized above, we are persuaded that Petitioner has shown by a preponderance of the evidence that the limitations of each of claims 10–15 are rendered obvious by the combination of Chen, Agrawal, and Brent. Therefore, we are persuaded that Petitioner has established by a preponderance of the evidence that claims 10–15 are unpatentable under § 103(a) over Chen, Agrawal, and Brent.

6. Conclusion for Ground 3

For the foregoing reasons, we conclude that Petitioner has shown by a preponderance of the evidence that claims 1, 2, 5–7, 9–15 are unpatentable under § 103(a) over Chen, Agrawal, and Brent.

III. CONCLUSION

In summary:

Claims	35 U.S.C. §	References	Claims Shown Upatentable	Claims Not Shown Unpatentable
1–5, 7	103(a)	Chen, Agrawal,	1–5, 7	
3, 4	103(a)	Chen, Agrawal, Compton	3, 4	
1, 2, 5–7, 9–15	103(a)	Chen, Agrawal, Brent	1, 2, 5–7, 9–15	
Overall Outcome			1–7, 9–15	

IV. ORDER

In consideration of the foregoing, it is hereby

ORDERED that Petitioner has established based on a preponderance of evidence that claims 1–7 and 9–15 of the '090 patent are unpatentable as set forth above; and

FURTHER ORDERED that, because this is a Final Written Decision, parties to the proceeding seeking judicial review of the Decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.[14]

[14] Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this Decision, we draw Patent Owner's attention to the April 2019 Notice Regarding Options for Amendments by Patent Owner Through Reissue or Reexamination During a Pending AIA Trial Proceeding. *See* 84 Fed. Reg.

IPR2022-00528
Patent 9,424,090 B2

For PETITIONER:
Walter Renner
David Holt
Nicholas Stephens
Jennifer Huang
FISH & RICHARDSON P.C.
axf-ptab@fr.com
holt2@fr.com
nstephens@fr.com
jjh@fr.com

For PATENT OWNER:

Scott McKeown
ROPES & GRAY LLP
scott.mckeown@ropesgray.com

W. Cook Alciati
Michael Dorfman
GARDELLA GRACE P.A.
calciati@gardellagrace.com
mdorfman@gardellagrace.com

16,654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. *See* 37 C.F.R. § 42.8(a)(3), (b)(2).