

**UNITED STATES INTERNATIONAL TRADE COMMISSION
WASHINGTON, D.C.**

**Before the Honorable Cameron R. Elliot
Administrative Law Judge**

In the Matter of

**CERTAIN GRAPHICS SYSTEMS,
COMPONENTS THEREOF, AND
DIGITAL TELEVISIONS CONTAINING
THE SAME**

Investigation No. 337-TA-1318

RESPONDENTS' POST-TRIAL BRIEF

Pursuant to Ground Rule 13.1, Respondents Realtek Semiconductor Corp. (“Realtek”) and TCL Industries Holdings Co., Ltd., TCL Industries Holdings (H.K.) Limited, TCL Electronics Holdings Limited, TCL Technology Group Corporation, TTE Corporation, TCL Holdings (BVI) Limited, TCL King Electrical Appliances (Huizhou) Co., Ltd., Shenzhen TCL New Technology Co., Ltd., TCL MOKA International Limited, TCL Smart Device (Vietnam) Co., Ltd., Manufacturas Avanzadas SA de CV, TCL Electronics Mexico, S de RL de CV, and TCL Overseas Marketing Ltd. (“TCL”) (collectively “Respondents”) hereby submit their Post-Trial Brief.

III. U.S. PATENT NO. 7,742,053

A. Claim Construction

1. “Select a command thread”

Claim 5[b] of the '053 patent recites, in relevant part, “an arbiter ... operable to select a command thread from either of the plurality of pixel command threads and the plurality of vertex command threads.” Under precedent and the intrinsic record, this claim term should be construed to require that the arbiter be “operable to select a single command thread,” which does not preclude the arbiter from *also* be capable of selecting a plurality of command threads from a larger plurality.

However, even if the claim term is construed to mean “operable to select one or more command threads,” as proposed by Complainants, the claim still requires the arbiter to have *the minimum capability of selecting a single command thread*. This is consistent with the claim construction order, which stated that “the claim itself requires selection of ‘a [single] command thread,’” Order No. 31 at 16 (brackets in original), and Order No. 66, which said the foregoing

[REDACTED]

statement was not a binding construction but was “intended to contrast the minimum number of threads required to be selected (one) with the minimum threads available in the selection pool.” Order No. 66 at 2-3. Accordingly, if an accused arbiter is not “operable” (capable) of selecting a single command thread, then it cannot satisfy the limitation “operable to select a command thread” under either of the two possible constructions of the term. Both Respondents’ and Complainants’ experts agree. Tr. (Pfister) at 594:5-19, Tr. (Reinman) at 375:3-15.

a. Complainants Have Agreed That “Operable to Select a Command Thread” Requires the Arbiter to Have the Minimum Capability of Selecting a Single Command Thread

During claim construction, Complainants took the position that the terms “arbiter” and “select a command thread” should be given their plain and ordinary meaning. Specifically, Complainants argued that Respondents’ proposed construction of the term “arbiter”—requiring the arbiter to select “*one* of a plurality of command threads”—had the same meaning as the claim language “operable to select a command thread from either of the plurality of pixel command threads and the plurality of vertex command threads.” EDIS 778052 at 5. On this basis, Complainants argued against Respondents’ construction because it would render the claim language “superfluous.” *Id.* The ALJ agreed with Complainants and held that the claim terms “arbiter” and “select a command thread” should be given their plain and ordinary meaning. Order No. 31 at 15-17. Order No. 31 further stated that “the claim itself requires *selection of ‘a [single] command thread’* from among at least four such threads, two (or more) pixel threads and two (or more) vertex threads.” *Id.* at 16 (brackets in original) (emphasis added). Accordingly, Order No. 31 and Complainants’ positions during claim construction support a construction that the arbiter must have a minimum capability of selecting a single command thread, regardless of how the claim term “select a command thread” is otherwise construed.

Complainants’ experts agreed that the claimed arbiter must have the minimum capability

[REDACTED]

of selecting a single command thread. Dr. Reinman testified that in forming his opinion on infringement, “My understanding was that *the arbiter was required to be able to select a single command thread*, but that there was also nothing to preclude it from also selecting more than a single, as long as it wasn’t selecting everything; otherwise, it would not be arbitrating.” Tr. (Reinman) at 374:19-375:17. Respondents experts also testified that the arbiter must be capable of selecting a single command thread. Tr. (Pfister) at 579:12-580:2, 594:5-19, 597:14-598:17.

b. The Term “Operable to Select a Command Thread” Should Be Construed As “Operable to Select a Single Command Thread” In the Context of Claim 5 and the ’053 Patent.

Complainants rely on the presumptive rule that the plain and ordinary meaning of “a” is “one or more.” *Baldwin Graphic Sys., Inc. v. Siebert, Inc.*, 512 F.3d 1338, 1342 (Fed. Cir. 2008). However, “*Baldwin* ... does not set a hard and fast rule that ‘a’ always means one or more than one.” *Harari v. Lee*, 656 F.3d 1331, 1341 (Fed. Cir. 2011). Here, the claim language, the context of the claim, and analogous case law all demonstrate that the term “operable to select a command thread” means “operable to select a single command thread,” as Order No. 31 stated. This does not preclude the arbiter from *also* selecting a plurality of command threads from a larger plurality, it just requires the capability of selecting a single command thread.

The plain language of claim 5 requires that the arbiter must be operable to select “a command thread” from *either* (i) two or more pixel command threads or (ii) two or more vertex command threads. The word “either” in the clause “from *either* of the plurality of pixel command threads and the plurality of vertex command threads” indicates that the minimum “pool” of command threads is two threads. Otherwise, no meaning would be given to the word “either” in the claim. Accordingly, the arbiter must necessarily be capable of selecting a single command thread. If it picked two command threads from a group of two command threads, it would not have “selected” anything. *See Nat’l Recovery Techs., Inc. v. Magnetic Separation Sys., Inc.*, 166

[REDACTED]

F.3d 1190, 1195 (Fed. Cir. 1999) (“The plain meaning of the term ‘select’ ... implies that some signals are to be picked or chosen and others are to be excluded A specific choice is being made.”). “Reading this otherwise renders any difference between the singular and the plural terms superfluous.” *August Tech. Corp. v. Camtek, Ltd.*, 655 F.3d 1278, 1284 (Fed. Cir. 2011). Claim 5[c] then requires command processing engines “operable to receive and process the command thread.” Put simply, the arbiter must be capable of choosing a single command thread from a group of two or more threads and then send the selected command thread to the command processing engines.

“Although the word ‘a’ generally means ‘one or more’ in open-ended claims containing the transitional phrase ‘comprising,’ that general rule does not apply when the specification or the prosecution history shows that the term was used in its singular sense.” *Norian Corp. v. Stryker Corp.*, 432 F.3d 1356, 1359 (Fed. Cir. 2005) (citing *KCJ Corp. v. Kinetic Concepts, Inc.*, 223 F.3d 1351, 1356 (Fed. Cir. 2000)). As the Federal Circuit explained, “the question whether ‘a’ or ‘an’ is treated as singular or plural depends heavily on the context of its use,” and “[t]he general rule does not apply when the context clearly evidences that the usage is limited to the singular.” *TiVo, Inc. v. EchoStar Commc’ns Corp.*, 516 F.3d 1290, 1303-04 (Fed. Cir. 2008); *Enfish, LLC v. Microsoft Corp.*, 822 F.3d 1327, 1342 (Fed. Cir. 2016). Because the entirety of Claim 5[b] supports the construction of “a command thread” to mean “a single command thread,” the general rule that “a” means “more than one” does not apply.

c. Because Claim Element 5[b] Is Not Open-Ended, it Requires the Arbiter to be Capable of Selecting a Single Command Thread

In addition, even though the indefinite article “a” generally means “one or more” in patent parlance, there is another condition to this general rule. “[A]n indefinite article ‘a’ or ‘an’ in patent parlance carries the meaning of ‘one or more’ in *open-ended claims containing the transitional*

[REDACTED]

phrase ‘comprising.’” *Baldwin*, 512 F.3d at 1342 (citing *KCJ Corp.*, 223 F.3d at 1356) (emphasis added). Here, the general rule does not apply because (i) the limitation in Claim 5[b] is not open-ended, and (ii) the use of “a” in conjunction with a closed group of alternatives (*i.e.*, two or more pixel command threads or two or more vertex command threads) limits its meaning to “one.”

In their Pre-Trial Brief, Complainants suggest that Claim 5[b] is open-ended because it uses the transitional phrase “comprising.” CPTB at 21. To be sure, Claim 5 does use “comprising” in the preamble, but that only means that the list of claim elements is not exclusive; it does not mean that *every limitation* is open-ended. *See In re Varma*, 816 F.3d 1352, 1362 (Fed. Cir. 2016) (“Although the transitional term ‘comprising’ indicates that the claim is open-ended, the term does not render each limitation or phrase within the claim open-ended.”); *Dippin’ Dots, Inc. v. Mosey*, 476 F.3d 1337, 1343 (Fed. Cir. 2007). Claim element 5[b] does **not** use “comprising.” Rather it requires an arbiter to be “operable to select a command thread” from either of two groups (two or more pixel command threads or two or more vertex command threads). Thus, the arbiter must be capable of selecting a single thread; it may be able to select more than one thread, but it must be able to select, at a minimum, just one.

Relatedly, the *Baldwin* rule does not apply where, as here, an indefinite article precedes a closed group.¹

¹ Claim 5[b] limits the selection of a command thread from one of two groups of command threads (pixel and vertex command threads). This limitation resembles a closed group known as a “Markush” group, and it provides further support for the interpretation that the selection of “a command thread” requires the selection of “a *single* command thread.” Like the selection between two specific groups of command threads in Claim 5[b], a Markush claim “incorporates a *closed* list of specific alternative limitations.” *In re Kiely*, No. 2022-1076, 2022 WL 2062163, at *2 (Fed. Cir. June 8, 2022). And like Claim 5[b]’s use of “a” (requiring the selection of “a command thread” from two alternative groups), when an indefinite article such as “a” is used in conjunction with a Markush grouping, it indicates that only one member of that group will be selected at a time. Because “a command thread” must be selected from one of two groups, each

d. Even if “A” is Construed to Mean “One or More,” the Arbiter Must Still Be Capable of Selecting a Single Command Thread

Finally, even if the ALJ construes “a command thread” to mean “one or more command threads,” the arbiter still must have the capability to select a single command thread. This is so even if the arbiter *also* has the capability of selecting more than one command thread. In other words, the ability to select a single command thread is a minimum requirement of the claim. Dr. Reinman expressly admitted that this is the plain meaning of the claim that he applied in his infringement analysis: “My understanding was that the arbiter was required to be able to select a single command thread.” Tr. (Reinman) at 375:3-15. Thus, to the extent AMD is now arguing for a different interpretation of the claim, it has no expert opinion on infringement.

Indeed, the Federal Circuit recently explained that when a claim element requires a system to do “a” thing, it requires the system, at a minimum, to be able to do that “one” thing. *See In re Varma*, 816 F.3d at 1362. In *Varma*, the Federal Circuit held that the PTAB had improperly interpreted the claim term “a statistical analysis *request* corresponding to two or more selected investments.” *Id.* at 1362-63 (emphasis added). The PTAB reasoned that prior art would be covered by the claim even if “two *requests* would be necessary in the [prior-art] system to accomplish an analysis of ‘two or more investments.’” *Id.* at 1362 (emphasis added) (citation omitted). The Federal Circuit rejected this construction, however, and held that the system must be capable of analyzing two or more investments through “[a] *single request*.” *Id.* (emphasis added). The context in *Varma* demonstrated that “a request” meant “one request.” *See id.* at 1363 n.2; *see also CANVS Corp. v. United States*, 126 Fed. Cl. 106, 118 (2016) (“Because the phrase ‘structured to define a line of sight’ is a limitation on the claim element ‘an optical input,’ the

of which may only contain two command threads, it indicates that only one command thread be selected at a time.

[REDACTED]

article ‘a’ contained therein does not become open-ended under the *Baldwin* rule it instructs only a single line of sight.”).

Likewise, here, there is no basis for inferring from the words that come after “select a command thread” that the singular term embraces a plurality of command threads. To the contrary, as discussed above, the claim language and context demonstrate that the arbiter must be able to, *at a minimum*, select a single command thread—even if it is *also* able to select more than one.

2. “At least one memory device ... operative to store a plurality of pixel command threads and ... a plurality of vertex command threads”

Order No. 31 indicated that the term “memory device” should have its plain and ordinary meaning, Order No. 31 at 20-21, but an apparent dispute has arisen as a result of testimony at the hearing regarding what it means for the memory device to be “operative to store a plurality of pixel command threads and ... a plurality of vertex command threads.”

Complainants have taken the position that the memory device does *not* need to store even a single command thread, much less *a plurality* of vertex or pixel command threads as required by the claim. [REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] As a result, under Complainants’ reading of the claim limitation, (1) it does not require any minimum portion of a command thread to be stored in memory, (2) it does not require any particular aspects of a command thread to be stored in memory, and (3) it does not require a command thread to be stored in memory over any specific period of time. Rather, under Complainants’ construction, the claim limitation only requires that the memory device be able to store an undetermined percentage of one or more of a plurality of command threads as little as a few bits or a pointer), undetermined aspects of the command threads (which may or may not

[REDACTED]

include, for example, instructions or a program counter), over some unlimited period of time.

Complainants' position would effectively read out of the claim the requirement that the memory device be "operative to store a plurality of pixel/vertex command threads." Complainants' position is essentially that claim 5 just requires memory, because *any* memory can store a plurality of command threads if there is no limitation whatsoever on how much of the command threads needs to be stored, what part or portions of the command threads need to be stored, and the period of time over which the plurality of command threads must be stored. This would effectively render the entire clause "operative to store a plurality of pixel command threads and ... a plurality of vertex command threads" meaningless, because any memory, with any level of capability, would be able to satisfy Complainants' construction of the claim limitation. Claim 5 does not just require "memory," it requires that the memory device have a certain *capability*, and the capability it must have is to "store a plurality of pixel/vertex command threads," which means nothing unless it requires actually storing a plurality of command threads at once, at a particular snapshot in time. Anything less than that, and any memory would satisfy Complainants' construction, because there would be no minimum capability required of the memory device.

The Federal Circuit has consistently held that claim terms must not be construed in a way that would render any term superfluous, but rather, each term should be construed to have a distinct meaning. *SSI Techs., LLC v. Dongguan Zhengyang Elec. Mech. Ltd.*, 59 F. 4th 1328, 1334-35 (Fed. Cir. 2023); *Akzo Nobel Coatings, Inc. v. Dow Chem. Co.*, 811 F.3d 1334, 1340 (Fed. Cir. 2016). Claims should be construed so as to give "meaning to all the terms of the claim." *In re Power Integrations, Inc.*, 884 F.3d 1370, 1376 (Fed. Cir. 2018); *Agilent Techs., Inc. v. Affymetrix, Inc.*, 567 F.3d 1366, 1377-78 (Fed. Cir. 2009); *Merck & Co.*, 395 F.3d at 1372.

Contrary to Dr. Reinman's testimony at the hearing, in its Pre-Trial Brief, Complainants

[REDACTED]

appeared to agree that this claim limitation requires that the memory device be capable of storing entire command threads “at the same time,” and only argued that the limitation does not require the storage of entire vertex shader programs or pixel shader programs at the same time. CPTB at 20. Complainants argued that the limitation should be construed to include memory that is “operative to store the commands of the vertex/pixel shader program over a period of time, or is *operative to store a sequence of commands at the same time*, albeit not necessarily of an entire vertex/pixel shader program.” *Id.* (emphasis added). It is not clear what distinction Complainants were attempting to draw between, *e.g.*, a “vertex shader program” and a “vertex command thread,” as Dr. Reinman at the hearing acknowledged characterizing a shader program as a command thread. Tr. (Reinman) at 282:2-14. But Complainants’ briefing suggests at least a pre-hearing understanding that the memory device limitation requires the capability of storing entire command threads in memory at the same time, which is consistent with Respondents’ position.

In order to give meaning to the express language of the claim, the “memory device” must be “operable” or “capable” of storing a plurality of vertex/pixel command threads at one point in time. Any other construction of the limitation “operative to store a plurality of pixel command threads and ... a plurality of vertex command threads” would render the term meaningless.

3. “Command processing engine”

A “command processing engine” is an engine that processes commands. *See, e.g.*, JX-0001 at 2:52-58 (stating that the command processing engine “may perform at least one processing command”). The only question left open by Order No. 31 is whether the term “command processing engine” should be limited to hardware. Nothing in the intrinsic evidence suggests such a limitation. “The command processing engine may be any suitable engine as recognized by one having ordinary skill in the art for processing commands.” *Id.* at 2:59-62. Respondents proposed

[REDACTED]

construction, “software or hardware module that processes commands,” is consistent with the plain meaning and should be adopted.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

D. Invalidity

1. Obviousness Under 35 U.S.C. § 103(a)

Stuttard invalidates all Asserted Claims of the '053 Patent in this Investigation. In rebuttal, Dr. Hart only addresses claim elements 5[a] and 5[b]. His argument is premised entirely on his incorrect interpretation (rejected by the PTAB) that the asserted claims require storing a plurality of vertex command threads and a plurality of pixel command threads “at the same time,” and processing both vertex and pixel command threads “at the same time.” Even though this new limitation that Dr. Hart adds to the claim was never raised during claim construction — and AMD did not apply this added limitation in their own infringement analysis — Dr. Hart has no other point of rebuttal. Regardless, the evidence and testimony clearly show that Stuttard renders the asserted claims obvious even under Dr. Hart’s incorrect claim construction.

a. Stuttard

Stuttard renders obvious the Asserted Claims (5, 6, and 9) of the '053 Patent considering

[REDACTED]

the knowledge of a POSITA as confirmed by the intrinsic evidence. *See, e.g.*, RDX-0004C.4-57.

It is undisputed that PCT Publication No. WO 00/62182 (“Stuttard”) was published on October 19, 2000, RX-0416.00002, and is therefore prior art under 35 U.S.C. § 102(b), and cannot be sworn behind. *See GoPro, Inc. v. Contour IP Holding LLC*, 908 F.3d 690, 693 (Fed. Cir. 2018).

(1) Overview of Stuttard

Stuttard discloses a graphics processing system that contains an embedded processing unit, processing core, local memory and a video output. RX-0416.0127; RDX-0004C.18. Stuttard’s processing core is where the relevant graphic processing takes place. Tr. (Pedram) at 896:18-22. It contains a thread manager, two controllers, and processing blocks. RX-0416.0131-32; RDX-0004C.19-24. The thread manager stores and selects command threads for processing using at least a cache memory, a scheduler, thread processors, and buses. *Id.*; Tr. (Pedram) at 899:3-901:4. The scheduler and thread processor can select from among a plurality of vertex command threads and a plurality of pixel command threads. Tr. (Pedram) at 900:2-21. The selected command threads are passed to the processing blocks, which can processes both pixel and vertex command threads. Tr. (Pedram) at 899:3-901:4. Moreover, the processing of pixel and vertex command threads can occur at the same time. Tr. (Pedram) at 892:2-15, 902:4-905:5, 906:4-15, 914:8-25, 990:6-12. Additional detail is provided below on an element-by-element basis.

The sole dispute between the parties regarding Stuttard turns on a two-part argument advanced by AMD: (1) that claim 5 requires that pixel command threads and vertex command threads must be stored and processed ***at the same time***, which is a limitation AMD improperly seeks to add to the claim, and (2) that Stuttard’s system is limited to a “phased system” that is unable to store and process both vertex and pixel command threads ***at the same time***. AMD is incorrect on both issues.

[REDACTED]

(2) Stuttard is NOT limited to a phased system

(a) “at the same time” is not a limitation

Every rebuttal put forth by Complainant is premised on the assumption that the claims require both storing and processing vertex and pixel command threads “at the same time.” Complainant’s expert, Dr. Hart, testified that he agreed Stuttard’s cache memory device stores both a plurality of vertex command threads and a plurality of pixel command threads. Tr. (Hart) at 1300:2-9. His only basis for disputing that claim element 5[a] is not disclosed by Stuttard is that both types of pluralities are not stored “at the same time.” *Id.* at 1300:10-14. Likewise, Dr. Hart agrees that Stuttard’s arbiter (as identified by Dr. Pedram) will “select between command threads from either pixel command threads or vertex command threads,” and his only basis for disputing that claim element 5[b] is not met is that the arbiter allegedly does not select between the two types of threads at the same time. *Id.* at 1252:13-22. In fact, as shown in more detail below with respect to the claim elements, Dr. Hart agreed that every element was present if the claims do not require “at the same time.”

As an initial matter, “at the same time” is a limitation that does not appear in the claims or specification, which AMD fabricated to try and avoid invalidity based on Stuttard. None of the asserted claims recite “at the same time.” The specification does not even use the phrase “at the same time.” Further, AMD never raised this issue during claim construction for any recited limitation. For example, Dr. Hart confirmed that during claim construction, his testimony was that the memory device limitation should be given its plain and ordinary meaning. RX-0099C.00015; Tr. (Hart) at 1301:14-21. During claim construction, neither AMD nor Dr. Hart ever raised the issue of “at the same time” as part of the plain and ordinary meaning of any term. RX-0099C.015; Tr. (Hart) at 1301:22-1302:2. It is merely a made up limitation, applied only to avoid the prior art.

Because none of the asserted claims require storing or processing of vertex and pixel

[REDACTED]

command threads “at the same time,” the ALJ could stop there and find the claims rendered obvious by Stuttard, because AMD has no other basis for distinguishing Stuttard if this is not required by the claims.

(b) Even if required, Stuttard discloses storing and processing both vertex and pixel command threads “at the same time”

Even if the claims required storing and processing vertex and pixel command threads “at the same time”—and they do not—Stuttard does in fact disclose this. At a minimum, Dr. Pedram testified that storing and processing both vertex and pixel command threads “at the same time” would have been rendered obvious by Stuttard in light of the knowledge of a POSITA. Moreover, the PTAB previously rejected AMD’s argument because AMD failed to apply the proper legal standard when considering Stuttard.

Dr. Hart completely ignores Stuttard’s teachings that at any particular moment in time the system can be assigned to any number of tasks, including different vertex and pixel threads. For example, Dr. Hart failed to address Stuttard’s statement:

“At any particular moment in time, each thread may be assigned to any one of a number of tasks. For example, thread zero may be assigned for general system control, thread 1 assigned to execute 2D (two dimensional) activities, and threads 2 to 7 assigned to executing 3D activities (such as calculating vertices, primitives or rastering).”

RX-0416.0014 at 11:16-25; *see also* RX-0416.0013-14 at 10:7-11.1. Instead, Dr. Hart improperly focuses on Stuttard’s teachings about how it processes a single thread, rather than considering Stuttard “as a whole.” As with ANY graphics processing, including that done in the ‘053 patent, you must process the vertex component before you process the pixel component. For example, if the task is to draw a triangle, you must calculate the triangle vertices (endpoints) to transform them into the screen-space coordinates before you rasterize the triangle into individual pixels and calculate the final color values of the pixels that will be displayed on the screen (create the triangle

[REDACTED]

image). *See* Tr. (Reinman) at 105:3-106:17. By focusing on the individual phases of processing a single task and how each processing block works standing alone, Dr. Hart fails to consider Stuttard *as a whole* and incorrectly concludes that different tasks cannot be done in parallel. Dr. Hart then jumps to the conclusion that if the system does not process any vertex and pixel command threads in parallel, then there will be no need or reason to select between vertex and pixel command threads. Dr. Hart's overly narrow interpretation of Stuttard improperly ignores Stuttard's overall teachings of executing multiple types of tasks at the same time.

"A reference must be considered for everything it teaches by way of technology and is not limited to the particular invention it is describing and attempting to protect." *EWP Corp. v. Reliance Universal Inc.*, 755 F.2d 898, 907 (Fed. Cir. 1985). Further, for an obvious analysis, one must consider the reference, "as a whole" and in light of the knowledge of a POSITA. *In re Merck & Co.*, 800 F.2d 1091, 1097 (Fed. Cir. 1986). The PTAB correctly held that the exact argument presented here—that "Stuttard's system performs graphics-processing operations in discrete, sequential phases, i.e., a phased system"—"fail to consider Stuttard, as a whole, including the disclosure of a thread manager and scheduler," and violate the principles of law set forth in *EWP Corp.* and *Merck*. Failing to consider the teachings of Stuttard as a whole, including the teachings that multiple processing blocks can be assigned multiple vertex and pixel activities "as any particular moment in time," is again fatal to Dr. Hart's opinions and AMD's position here.

(3) Independent Claim 5

(a) 5[preamble]: A graphics processing system comprising:

The preamble is not a limitation. Regardless, Stuttard explicitly discloses a graphics processing system. *See, e.g.*, RX-0416.0127 at Figure 1 (labeling its system as a "graphics system"); RX-0416.0007 at 4:21-22 ("Figure 1 is a system level block diagram illustrating a

graphics data processing system 3.”); *see also*, RX-0416.0007 and .0008 at 4:15-17 and 5:8-9; Tr. (Pedram) at 901:19-902:2.

- (b) 5[a]: at least one memory device comprising a first portion operative to store a plurality of pixel command threads and a second portion operative to store a plurality of vertex command threads;**

Stuttard discloses a cache (memory) unit 1024, which is the claimed at least one memory device. RDX-0004C.31; RX-0416.0132; Tr. (Pedram) at 902:4-11. Dr. Pedram testified that Stuttard both discloses (Tr. (Pedram) at 903:11-22) and renders obvious (*Id.* at 903:23-905:5) that its cache memory device comprises a first portion operative to store a plurality of pixel command threads and a second portion operative to store a plurality of vertex command threads. RDX-004C.32-33; RX-0416.014 at 11:3-5, 16-25. Dr. Hart agreed that Stuttard’s cache memory device “stores a plurality of vertex command threads” and “stores a plurality of pixel command threads” and that those command threads will be “executed on the processing blocks.” Tr. (Hart) at 1254:11-18; 1300:2-14.

Stuttard discloses that it fetches and stores in its cache memory command threads associated with general system control, a 2D activity, and/or a 3D activity. RX-0416.0013-14; Tr. (Pedram) at 902:13-903:22. A POSITA would know that threads associated with these activities comprise both a plurality of vertex command threads and a plurality of pixel command threads. Tr. (Pedram) at 902:13-903:22. Further, a POSITA would have understood that Stuttard is processing both a plurality of each of vertex and pixel command threads (due to the number of processing blocks in Stuttard’s system, which is at least eight) and that it is at least obvious they are being stored in the cache memory device. *Id.* at 903:23-905:5. Moreover, a POSITA would also have understood the advantages of having a memory device operative to store the pluralities of command threads as claimed and been motivated to operate with such a memory device. *Id.*;

[REDACTED]

RDX-0004C.33.

Again, Dr. Hart's *only* rebuttal to this limitation (found only at roughly a single page of the transcript from 1254:10-1255:19) is that Stuttard's memory would not store both vertex and pixel command threads "at the same time." Tr. (Hart) at 1300:10-14; 1254:19-1255:8. But, storing at the same time is not a claim limitation, was never raised as part of the claim construction briefing, contradicts Dr. Hart's testimony that the claim limitation should be given its plain and ordinary meaning, and was not a limitation AMD asserted or showed in their alleged infringement case. If the claim does not have this requirement, then Dr. Hart has no rebuttal to this claim limitation.

Even if the claims did require storing and processing both vertex and pixel command threads "at the same time," AMD and Dr. Hart are simply wrong, as they intentionally ignore the teachings of Stuttard that disclose and/or at a minimum render obvious storing and processing both vertex and pixel command threads at the same time. Dr. Pedram testified that when rendering two different objects in the same 2-D screen, it is possible for a first object to be in its vertex processing stage while the second object is in its pixel processing stage. In other words, supporting a phased pipeline (system) does not conflict with storing or processing vertex and pixel command threads simultaneously. Tr. (Pedram) at 1018:6-1019:12.

Moreover, Stuttard *does* disclose processing of both vertex and pixel activities (*e.g.*, calculating vertices, rastering) that would require storing both types of command threads at the same time. This is discussed in more detail with respect to the arbiter limitation below. And, even if, for the sake of argument, Stuttard's teachings were not found to be an express or inherent disclosure, Dr. Pedram testified that it would have been obvious to store and process both vertex and pixel command threads at the same time. Tr. (Pedram) at 903:23-905:5.

First, the system described in Stuttard is capable of storing and processing both vertex and

[REDACTED]

pixel command threads at the same time. These are system claims in which the memory device, arbiter, and command processing engines are only required to be “operative” or “operable” to perform the recited functions. “Operable” is construed to mean “capable.” *See, e.g., In re Imes*, 778 F.3d 1250, 1253-54 (Fed. Cir. 2015). When claim language recites “capability,” a prior art reference may anticipate or render obvious the claim if it discloses an apparatus that is “reasonably capable of operating so as to meet the claim limitations, even if it does not meet the claim limitations in all modes of operation.” *ParkerVision, Inc. v. Qualcomm Inc.*, 903 F.3d 1354, 1361 (Fed. Cir. 2018). This is precisely the case here, where Dr. Hart’s analysis focuses only on one moment in time, when graphics processing first commences, and does not consider the capability of Stuttard’s system during its full operation.

Second, the disclosures discussed above (RX-0416.014 at 11:3-5, 16-25 — which were ignored by AMD — would, at a minimum, suggest to a POSITA that one could store and process both vertex and pixel command threads in cache (memory) unit 1024 at the same time. Tr. (Pedram) at 902:905:5.

Third, the express disclosures in Stuttard together with the known advantages of storing and processing both vertex and pixel threads at the same time would further motivate a POSITA to practice Stuttard as claimed, including efficiencies such as run time efficiencies, which were identified in the Stuttard reference itself. Tr. (Pedram) at 904:23-905:5; RX-0416.0015 at 12:1-7. More precisely, a POSITA studying the Stuttard’s system would be motivated by Stuttard’s explicitly stated goals of (i) “effectively mask[ing]” the latency of the memory I/O operations from the system “by the use of different threads,” RX-0416.0013 at 10:19-31 (ii) having a “faster response time to external events,” RX-0416.0013-14 at 10:35-11:1 (iii) maintaining “the processing of simultaneously active program threads, so that each [of] the processing blocks 106

[REDACTED]

can be active for as much time as possible,” RX-0416.0014 at 11:16-33 and (iv) reducing “the need for context switching when changing the active thread, and thereby reducing memory accesses and hence increasing the speed of operation,” RX-0416.0015 at 12:1-7 to practice “at the same time” when working with the Stuttard’s system. Dr. Pedram concluded that this limitation was obvious. In addition to the reasoning above, Dr. Pedram based his obvious conclusion, in part, on the fact that this limitation, storing a plurality of vertex command threads and a plurality of pixel command threads in different portions at the same time was well known in the art to a POSITA. Tr. (Pedram) at 891:19-892:1; RDX-0004C.9-10; RX-0416 at 1:24-28, 42-64, 2:39-41.

- (c) **5[b]: an arbiter, coupled to the at least one memory device, operable to select a command thread from either of the plurality of pixel command threads and the plurality of vertex command threads; and**

Stuttard discloses a thread scheduler 1025 and thread processors 1026, which collectively comprise the claimed arbiter. RDX-0004C.35-38; RX-0416.0132 at Figure 4; Tr. (Pedram) at 905:7-16. The thread scheduler 1025 and thread processors 1026 are coupled to the cache unit via a bus and operable to select a pixel command thread or a vertex command thread. RDX-0004C.37-40; RX-0416.0132 at Figure 4; Tr. (Pedram) at 905:19-906:17-907:7. None of this is disputed by AMD. For example, Dr. Hart admits that Stuttard’s arbiter will “select between command threads from either pixel command thread or vertex command threads.” *Id.* at 1252:12-17.

The only dispute, again, is whether Stuttard’s arbiter can select between a plurality of vertex command threads and a plurality of pixel command threads “at the same time.” Dr. Hart asserts this is a limitation of the claims based his interpretation that the claim requires at least four command threads during selection, *i.e.*, at least two vertex command threads and at least two pixel command threads. Tr. (Hart) at 1249:15-21. However, Dr. Hart’s interpretation of the claim is wrong because it ignores the plain language of the claim and reads out the phrase “either of.” The

[REDACTED]

claim recites “an arbiter ... operable to select a command thread from either of the plurality of pixel command threads and the plurality of vertex command threads.” JX-0001.0010 (emphasis added). Requiring a minimum of four command threads makes the phrase “either of” meaningless as the claim would have the exact same scope if “either of” was deleted. The Federal Circuit has consistently warned against any interpretation that “renders [terms] void, meaningless, or superfluous.” *AMP Plus, Inc. v. DMF, Inc.*, No. 2021-1595, 2022 WL 16844516, at *6 (Fed. Cir. Nov. 10, 2022) (citing *Intel Corp. v. Qualcomm Inc.*, 21 F.4th 801, 810 (Fed. Cir. 2021)). Thus, Dr. Hart has applied (1) something more than the plain and ordinary meaning to avoid the prior art, and (2) an incorrect interpretation of the claim – that the arbiter must select between at least two vertex command threads AND two pixel command threads (for a total of four). Once that incorrect claim requirement is removed, Dr. Hart has no rebuttal.

Regardless, Stuttard stores and processes both a plurality of vertex and pixel command threads at the same time. Dr. Pedram testified that Stuttard stores, for example, eight different command threads in its cache and has eight or more thread processors with a corresponding number of processing blocks (command processing engines). Tr. (Pedram) at 897:11-898:14; 899:4-15; 900:2-901:9; 906:7-15; RDX-0004C.21-26; RX-0416.0011 at 8:9-13; RX-0416.0014 at 11:16-25. The thread scheduler 1025 works with the thread processors 1026 to schedule and activate the highest priority thread.

“Within the thread manager 102, the status block 1030 receives status information 1036 from each of the threads. The status information is transferred to the thread scheduler 1025 by the status block 1030. The status information is used by the thread scheduler 1025 to determine which thread should be active at any one time. Core instructions 1032 issued by the thread manager 102 are sent to the array controller 104 and the channel controller 108 (FIG. 3).”

RX-0416.0015 at 12:13-22; RDX-0004C.40; Tr. (Pedram) at 906:19-907:7.

Dr. Pedram further testified that Stuttard discloses that it can process both vertex and pixel

[REDACTED]

command threads at the same time. Tr. (Pedram) at 907:8-908:17. Stuttard expressly states:

“At any particular moment in time, each thread may be assigned to any one of a number of tasks. For example, thread zero may be assigned for general system control, thread 1 assigned to execute 2D (two dimensional) activities, and threads 2 to 7 assigned to executing 3D activities (such as calculating vertices, primitives or rastering).”

RX-0416.0014 at 11:16-25; RDX-0004C.42. 3D activities “include both vertex processing as well as pixel processing.” Tr. (Pedram) at 907:8-908:2. A POSITA would know that “calculating vertices requires vertex command thread processing and doing rasterization at least requires doing pixel command thread processing.” *Id.* at 902:23-903:6. Dr. Pedram further testified that a POSITA would have understood that there could be combinations of vertex and pixel command threads, including two or more vertex command threads and two or more pixel command threads being processed at the same time in Stuttard’s multiple processing blocks. *Id.* at 900:2-21; 905:7-908:2. This teaching in Stuttard is fatal to Dr. Hart’s interpretation.

Dr. Hart completely ignores Stuttard’s teachings that at any particular moment in time the system can be assigned to any number of tasks, including different vertex and pixel activities. For example, he failed to address the quote from Stuttard above. Instead, Dr. Hart narrowly and improperly focused on Stuttard’s teachings about how it processes a single thread. As with ANY graphics processing, including that done in the ‘053 patent, you must process the vertex component before you process the pixel component. For example, if the task is to draw a triangle, you must calculate the triangle vertices (endpoints) before you rasterize the triangle into individual pixels (create the triangle image). *See* Tr. (Reinman) at 105:3-106:17. By focusing on the phases of processing a single task and how each processing block works, Dr. Hart fails to consider Stuttard as a whole and incorrectly infers that two tasks can be done in parallel.

As discussed in more detail below with respect to the command processing engines, Dr. Hart’s very narrow interpretation of Stuttard that conveniently ignores Stuttard’s overall teachings

[REDACTED]

of executing multiple types of tasks at a time is wrong. “A reference must be considered for everything it teaches by way of technology and is not limited to the particular invention it is describing and attempting to protect.” *EWP Corp.*, 755 F.2d at 907. Further, for an obvious analysis, one must consider the reference, “as a whole” and in light of the knowledge of a POSITA. *In re Merck & Co.*, 800 F.2d at 1097. Dr. Hart’s exact argument of Stuttard being limited to a phased system was completely and unequivocally rejected by the PTAB as being contrary to this legal principle. RX-0519.0098-99. Failing to consider the teachings of Stuttard as a whole, including the teachings that multiple processing blocks can be assigned multiple vertex and pixel activities “as any particular moment in time,” is again fatal to Dr. Hart’s opinions and AMD’s position. Dr. Pedram further testified that this limitation was obvious because an arbiter was well known in the art to select a command thread between different types of command threads. Tr. (Pedram) at 911:21-25; 892:17-893:11; RDX-0004C.12-13; RX-0416.0008 at 1:32-64, 2:48-52.

(d) 5[c]: a plurality of command processing engines, coupled to the arbiter, each operable to receive and process the command thread.

Dr. Hart did not offer any rebuttal opinions at the hearing regarding this limitation. As a result, its disclosure by Stuttard is not in dispute.

Dr. Pedram testified that Stuttard’s processing blocks 106 comprise the claimed plurality of command processing engines. Tr. (Pedram) at 908:23-909:17; RDX-0004C.44-48. These processing blocks perform the actual graphics processing. Tr. (Pedram) at 910:13-15. Stuttard’s processing blocks 106 are coupled to the arbiter (thread scheduler 1025 and thread processors 1026 within the thread manager 102) via the array controller 104 and channel controller 108. Tr. (Pedram) at 909:1-21; RDX-0004C.44-46; RX-0416.0131 at Figure 3, 0012 at 9:11-17. The processing blocks receive the command threads from the arbiter through the controllers (102 and 104). Tr. (Pedram) at 909:23-910:7; RDX-0004C.46; RX-0416.0012 at 9:11-17. The processing

[REDACTED]

blocks then process “the command thread” received from the arbiter. Tr. (Pedram) at 910:9-21; RDX-0004C.47-48; RX-0416.0011 at 8:5-7, 0020 at 17:29-32, 0023-24 at 20:32-21:7. Further, the command thread may be a vertex command thread or a pixel command thread. Tr. (Pedram) at 910:22-911:20; RDX-0004C.48; RX-0416.0023-24 at 20:32-21:7.

Dr. Pedram explained that different processing blocks in Stuttard’s system are independent of each other, with each processing block organized as a single-instruction multiple-data (SIMD) array of PE’s. With 8 processing blocks, Stuttard’s system can support 8 tasks running at the same time, including general I/O, 2-D activity and 3-D activity tasks. Tr. (Pedram) at 995:9-996:5; RX-0416.0014 at 11:16-33. Dr. Pedram further testified that 3-D activity tasks involve processing of both vertex and pixel command threads. Tr. (Pedram) at 907:8-908:2; RX-0416.0015 at 11:16-25.

Based on the testimony and evidence identified above, Stuttard renders obvious claim 5 of the ’053 patent. Tr. (Pedram) at 911:21-25; RDX-0004C.49.

(4) Dependent Claim 6: The graphics processing system of claim 5, wherein the plurality of command processing engines comprises at least one arithmetic logic unit

Stuttard expressly discloses a processing block (command processing engine) comprising at least one arithmetic logic unit (ALU). Tr. (Pedram) at 913:9-914:4; RDX-0004C.52-54; RX-0416.0131 at Figure 3, 0050 at 47:24-30, 0139 at Figure 11. Specifically, processing block 106 includes a processor unit 1061, which “includes an arithmetic logic unit (ALU) 214.” *Id.*

Dr. Hart did not dispute that Stuttard discloses an ALU as part of its processing blocks.

(5) Dependent Claim 9: The graphics processing system of claim 5, wherein a single memory device comprises the first portion and the second portion of the at least one memory device.

As discussed earlier with respect to claim element 5[a], Stuttard discloses a memory device in the form of a cache unit 1024. RDX-0004C.31; RX-0416.0132; Tr. (Pedram) at 902:4-11.

[REDACTED]

Because this cache unit comprises a single device, this claim is satisfied for the same reasons as element 5[a]. RDX-0004C.56, 57; RX-0416.0132 at Figure 4; Tr. (Pedram) at 914:8-25.

Based on the testimony and evidence discussed above, dependent claims 6 and 9 are rendered obvious by Stuttard. RDX-0004C.57; Tr. (Pedram) at 915:2-5.

2. Lack of Secondary Considerations of Non-Obviousness

At the hearing, AMD failed to provide any evidence probative of secondary considerations of non-obviousness with respect to the '053 patent. With respect to the '053 patent, AMD relies on “unified shader” technology for its alleged evidence of secondary considerations—which AMD generally defines as a processing engine that “unifies pixel shading and vertex shading.” Tr. (Reinman) at 235:20-25. There are several fundamental flaws in AMD’s theory.

First, it is not disputed that “unified shaders” were already known in the art. For example, Dr. Hart did not dispute that Stuttard’s processing blocks are capable of processing both vertex and pixel command threads (*i.e.*, “unified shaders”). Further, under Dr. Hart’s opinion provided at the hearing, the claim requires every command processing engine to be configured to process both vertex and pixel command threads. Tr. (Hart) at 1248:19-1249:9. Claim 6 requires that the plurality of processing engines comprise at least one arithmetic logic unit. It is undisputed that arithmetic logic units were disclosed as part of Stuttard’s prior art command processing engines. Further, the '053 patent specification discloses that “[t]he command processing engine may be ***any suitable engine as recognized by one having ordinary skill in the art*** for processing commands, such as a texture engine, an arithmetic logic unit, or any other suitable processing engine.” JX-0001 at 2:59-62 (emphasis added). Thus, the '053 patent itself expressly identifies that the claimed “command processing engines” were within the knowledge of a person of ordinary skill in the art.

[REDACTED]

It is simply indisputable that unified shaders were known in the art at the time of the invention.⁴ In order for evidence of non-obviousness to be relevant, there must be a nexus between the merits of the claimed invention and objective evidence of secondary considerations. *In re GPAC Inc.*, 57 F.3d 1573, 1580 (Fed. Cir. 1995); *Demaco Corp. v. F. Von Langsdorff Licensing Ltd.*, 851 F.2d 1387, 1392-93 (Fed. Cir. 1988). “Evidence of commercial success, or other secondary considerations,” is not relevant if it is “due to *an unclaimed feature of the device*,” or “*if the feature . . . was known in the prior art*.” See *Ormco Corp. v. Align Tech., Inc.*, 463 F.3d 1299, 1311-12 (Fed. Cir. 2006) (emphasis added); see also *id.* at 1311-13 (holding claims obvious despite evidence of commercial success because success was due to unclaimed or non-novel features of the device); *Campbell Soup Co. v. Gamon Plus, Inc.*, 10 F.4th 1268, 1278-79 (Fed. Cir. 2021), *cert. denied*, 142 S. Ct. 1129 (2022) (patentee failed to show secondary considerations evidence was due to “unique characteristics” separating the product from the prior art); *Ethicon Endo-Surgery, Inc. v. Covidien LP*, 812 F.3d 1023, 1034-35 (Fed. Cir. 2016) (upholding Board finding that patent was obvious in view of prior art where patentee did not show commercial success of

⁴ This was one of several reasons the PTAB rejected Complainants’ alleged evidence of secondary considerations of non-obviousness with respect to the ’053 patent. Order No. 68 precludes Respondents from relying on the PTAB’s Final Written Decision (RX-0519) “to support their obviousness case” because it was not cited in the claim charts that were part of Respondents’ final invalidity contentions. *Id.* at 5. However, Order No. 68 recognizes the PTAB’s FWD can be cited for other purposes, such as claim construction “or some other disputed matter in this investigation.” *Id.* This would include secondary considerations of non-obviousness, which is an issue where Complainants bear the burden of proof. Moreover, while the ALJ struck references to the FWD because it was not referenced in Respondents’ claim chart served with final invalidity contentions, Respondents *did* cite extensively to the PTAB’s FWD in its invalidity contentions with respect to the lack of secondary indicia of non-obviousness, which is why Complainants did not move to strike reliance on the FWD for this purpose. See EDIS 785260, Ex. 2 at 17 (final invalidity contentions) (“In finding the ’053 patent invalid in its final written decision, the Patent Trial and Appeal Board rejected ATI’s argument of non-obviousness. Final Written Decision, IPR2015-00325 Paper 62 at 77-79”).

[REDACTED]

product and evidence of unmet needs were due to patented features rather than unclaimed or non-novel features); *Richdel, Inc. v. Sunspool Corp.*, 714 F.2d 1573, 1580 (Fed. Cir. 1983) (claim was obvious despite purported showing of commercial success where patentee failed to show that “such commercial success . . . was due to anything disclosed in the patent in suit which was not readily available in the prior art”).

As previously explained, it is not disputed by Dr. Hart that “unified shaders” were known in the art. The PTAB independently reached the same conclusion in rejecting Complainants’ evidence of secondary considerations:

More importantly, the feature—a “unified shader”—that is allegedly adopted by the industry stems from what was known in the prior art. *See, e.g.*, Ex. 1005; Ex. 3003. Contrary to ATI’s argument and expert testimony, the named inventors of the ’053 patent were not the first in the art to disclose a single graphics processing unit to perform both vertex and pixel operations. As ATI acknowledges, such a feature was known in the art. PO Resp. 53; Ex. 2151 ¶ 199 (“Stuttard’s system processes vertex operations (i.e., geometry) and the so-called pixel operations (i.e., rasterization) . . . across all processing blocks of the whole system.”). Indeed, both Stuttard and the ’182 international publication, which was published on October 19, 2000,⁵ more than one year prior to the ’053 patent’s effective filing date, disclose a plurality of processor blocks, each capable of processing both vertex and pixel operations. Ex. 1005, 8:34–10:47, Fig. 3; Ex. 3003, 17:26– 22:32, Fig. 3.

RX-0519 at 78. In fact, the term “unified shader” does not appear anywhere in the ’053 patent.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

⁵ The international publication referenced by the FWD is the prior art referenced referred to as “Stuttard” herein, which Respondents rely upon for obviousness.

[REDACTED]

IV. U.S. PATENT NO. 8,854,381

A. **Claim Construction**

1. **“Configured to ... execute a first task ... *while executing* a second task”**

The plain and ordinary meaning of the term “execute a first task . . . *while executing* a second task” is to execute the first task and the second task at the same time, *i.e.*, simultaneously. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1313 (Fed. Cir. 2005) (en banc) (a claim term should be given its “ordinary and customary meaning” as understood by a POSITA at the time of the invention).⁶ The language of the claim, the specification, and the prosecution history all confirm this plain and ordinary meaning, which is further supported by expert testimony on both sides. The ALJ should construe the claim term accordingly.

In addition, claim 15 requires that the shader core be “configured to” (designed to) execute the first task and second task simultaneously. Thus, even if the first task and second task are sometimes executed simultaneously, if this occurs only as a result of chance or probability, rather than by design, this fails to satisfy the claim.

a. **The ordinary meaning of the term “while executing” refers to executing a first task and a second task simultaneously**

Claim 15(e) recites “a shader core *configured to* . . . execute a first task . . . *while executing* a second task.” To say that an apparatus can perform one function *while* performing another is necessarily to say it can perform both functions at the same time. *See Phillips*, 415 F.3d at 1314 (“In some cases, the ordinary meaning of claim language as understood by a person of skill in the art may be readily apparent even to lay judges, and claim construction in such cases involves little more than the application of the widely accepted meaning of commonly understood words.”); *see*

⁶ AMD first identified this term for construction on February 9, 2023, without proposing a construction, and briefed this issue for the first time in its Pre-Hearing Brief.

[REDACTED]

also Tr. (Pfister) at 714:4-16 (“‘while’ means at the same time”).

Complainants’ position as to what constitutes “while executing” is broader than “simultaneous” or “at the same time.” For example, Complainants stated in their Pre-Hearing Brief that “the term ‘while’ to a POSITA . . . would only require the shader core to be configured to execute the first task for some period of time during which the second task is being executed.” CPrHB at 124. Dr. Reinman confirmed that “while executing” means “the first and second tasks share a percentage of time,” that is, the execution of the tasks “will be overlapped.” Tr. (Reinman) at 157:10-158:6. However, as will be discussed in the non-infringement section below, Complainants argue that just having two tasks present in the shader core at the same time constitutes “executing” the tasks at the same time, *i.e.*, if instructions associated with the first task and instructions associated with the second task are being executed on different clock cycles (*e.g.*, interleaving), that constitutes executing the tasks at the same time. Respondents disagree. Alternating between two tasks is not executing them at the same time or simultaneously, under the plain meaning of “while executing.” *See* Tr. (Pfister) at 714:10-16 (claim 15 requires operation at the same time and on the same clock cycle).

b. As a result of AMD’s amendment of the claim during prosecution, “while executing” must be construed to mean executing the first task and second task simultaneously

The prosecution history can “often inform the meaning of the claim language by demonstrating how the inventor understood the invention.” *Phillips*, 415 F.3d at 1317. Here, the Examiner initially rejected claim 15 as indefinite, finding that the description of “a shader core configured to execute the plurality of tasks *substantially in parallel*” was inadequate because the term “substantially” was “not definite.” JX-0009.124. In response, AMD amended the claim to strike the claim term “substantially in parallel” and replace it with “execute a first task . . . *while executing* a second task.” JX-0009.141, 146 (emphasis added). This is strong evidence of how

[REDACTED]

AMD understood the invention it sought to patent. *See Phillips*, 415 F.3d at 1317. Since the term “substantially in parallel” was not sufficiently definite, AMD chose to replace with “while executing,” which in light of the prosecution, specification, and the plain meaning of “while” must be construed to mean “simultaneous” execution of the first task and second task.

Applying the usual rules of claim construction—the plain meaning of “while executing,” the prosecution history, and corresponding embodiments in the specification (described in the next section)—the term “while executing” should be construed as requiring the first task and the second task to be executed “simultaneously.” But this result is also required by the legal doctrines of prosecution disclaimer and estoppel, as AMD clearly surrendered the claim scope of “substantially in parallel”—the original language of the claim—by amending its claim during prosecution to overcome the patent examiner’s rejection.

The Supreme Court has long held that “claims that have been narrowed in order to obtain the issuance of a patent ... cannot be sustained to cover that which was previously ... eliminated from the patent.” *Graham v. John Deere Co.*, 383 U.S. 1, 33 (1966). Courts must “exclude any interpretation that may have been disclaimed or disavowed during prosecution in order to obtain claim allowance.” *Standard Oil Co. v. Am. Cyanamid Co.*, 774 F.2d 448, 452 (Fed. Cir. 1985). This is necessary to “protect[] the public’s reliance on definitive statements made during prosecution.” *Omega Eng’g, Inc. v. Raytek Corp.*, 334 F.3d 1314, 1324 (Fed. Cir. 2003). Prosecution disclaimer may arise either from statements made during the prosecution or, as here, from claim amendments. *See Edwards Lifesciences LLC v. Cook Inc.*, 582 F.3d 1322, 1330-31 (Fed. Cir. 2009) (reviewing claim amendment and accompanying remarks). The focus is on the statements made by the applicant, not on whether the examiner relied on the statement to allow the claim. *Springs Window Fashions LP v. Novo Indus., L.P.*, 323 F.3d 989, 994 (Fed. Cir. 2003).

[REDACTED]

Subject matter surrendered during prosecution cannot be reclaimed in litigation to support a claim of infringement, either through application of the doctrine of equivalents or otherwise. *See Cloud Farm Assocs. LP v. Volkswagen Grp. of Am., Inc.*, 674 F. App'x 1000, 1007 (Fed. Cir. 2017) (disclaimer during prosecution prevents the patentee from “reclaiming subject matter that was surrendered through statements or claim amendments”) (citing *Omega Eng'g*, 334 F.3d at 1324); *Greenliant Sys., Inc. v. Xicor LLC*, 692 F.3d 1261, 1271 (Fed. Cir. 2012) (rule against recapture prevents a patentee from reclaiming surrendered subject matter); *Loral Fairchild Corp. v. Sony Corp.*, 181 F.3d 1313, 1326 (Fed. Cir. 1999) (when a claim is amended for purposes of patentability, prosecution history estoppel precludes the patentee from reclaiming the subject matter that was surrendered by the amendment).

Complainants clearly disclaimed “substantially in parallel” by replacing it with “while executing” to obtain allowance of the claim. The definiteness rejection explained that the limitation “a shader core configured to execute the plurality of tasks ***substantially in parallel***” was inadequate because the term “substantially” was “not definite.” JX-0009.124. Accordingly, this language was changed to more specifically clarify that the shader core is “configured to . . . execute a first task . . . ***while executing*** a second task.” JX-0009.141. A claim term is indefinite if it “might mean several different things and no informed and confident choice is available among the contending definitions.” *Interval Licensing LLC v. AOL, Inc.*, 766 F.3d 1364, 1371 (Fed. Cir. 2014) (citation omitted). By editing the claim to be more definite, AMD chose one of several contending definitions: “while executing,” *i.e.*, simultaneously. In so doing, it “indicat[ed] what the claims do not cover, [and] . . . by implication surrender[ed] such protection.” *Ekchian v. Home Depot, Inc.*, 104 F.3d 1299, 1304 (Fed. Cir. 1997) (applying rules of estoppel).

Accordingly, reading the term “while executing” to mean “substantially in parallel” or

[REDACTED]

“(substantially) simultaneously”—terms used in the specification—would undermine “the public’s reliance on definitive statements made during prosecution.” *See Omega Eng’g*, 334 F.3d at 1324. It would also render the claim indefinite. *See C.R. Bard, Inc. v. M3 Sys., Inc.*, 157 F.3d 1340, 1363 (Fed. Cir. 1998) (“Claims must be interpreted the same way for determining infringement as was done to sustain their validity.”). Whether applying legal principles of disclaimer, estoppel, or surrendered subject matter, Complainants should not be permitted to recapture in litigation the claim scope that they surrendered during prosecution to obtain allowance of the claim. The Examiner correctly determined that a POSITA would not be sure what “substantially” in parallel or “substantially” simultaneously means. The only remaining way in which “while executing” can be construed is to apply its plain meaning, which is “simultaneously executing.”

c. Embodiments in the specification support Respondents’ proposed construction of “while executing”

The specification supports Respondents’ construction of the term “while executing” as requiring simultaneous processing. In particular, the specification discloses embodiments in which the shader core of the invention is configured to execute two tasks simultaneously (*e.g.*, on the same clock cycle) by executing the first and second task on different processing units or different subsets of processing elements, which allows the execution of the tasks to proceed exactly in parallel. For example, the specification explains that the scheduling module may “divide the resources of shader core 214 between concurrent tasks.” JX-0003.0011 5:34-39; 10:47-50. Under this approach, “a first ... task from a first virtual engine is issued to a first subset of processing elements (SIMDs) of the shader core, and a second ... task from a second virtual engine is issued to a second subset of processing elements (SIMDs) of the shader core,” which allows the tasks to be processed simultaneously. *Id.* at 5:17-22. The specification elsewhere describes processing multiple threads or shader programs “*at the same time*” by executing them “in parallel” using

[REDACTED]

“different processing elements of the shader core.” *Id.* at 1:58-2:1.

Accordingly, the plain meaning of the claim language, the specification, and the prosecution history all support construing the term “execute a first task . . . while executing a second task” to mean “simultaneously executing” the first task and the second task.

d. The shader core must be “configured to” execute the first task and the second task simultaneously, not as the result of chance or probability, which only requires capability

Claim 15 provides that the shader core must be “**configured to** . . . execute a first task from the plurality of tasks **while executing** a second task from the plurality of tasks.”

The Federal Circuit has held that the phrase “configured to” means “designed to” or “made to,” which is narrower and more specific than “capable of” or “suitable for.” *Aspex Eyewear, Inc. v. Marchon Eyewear, Inc.*, 672 F.3d 1335, 1349 (Fed. Cir. 2012). An apparatus or system that is “configured to” perform a specified function does not mean it is merely “capable of” the recited function, which would eliminate any meaningful limit to the words “configured to,” but means that it is specifically “arranged or set-up to perform a specified function,” “set up for operation especially in a particular way,” or “actually programmed or equipped with software to” perform the recited function. *Certain Infotainment Sys., Components Thereof, and Autos. Containing the Same*, Inv. No. 337-TA-1119, Final Initial Determination at 237 (Nov. 13, 2019). Thus, in *Certain High-Density Fiber Optic Equip. & Components Thereof*, the claim term “configured to support” was construed to refer to **the design** of the fiber optic equipment, “not its use at any given point in time.” Inv. No. 337-TA-1194, Final Initial Determination at 65 (Mar. 23, 2021).

In the context of claim 15, the claim language “a shader core **configured to** . . .” means it is insufficient that the shader core is simply “capable of” executing a first task and a second task simultaneously. Rather, the shader core must be specifically programmed or designed to cause a first task and a second task to be executed simultaneously—and not just any two tasks, but a first

[REDACTED]

task and a second task from *the same plurality of tasks* received by the plurality of engines. If a first task and a second task from a plurality of tasks are occasionally executed at the same time, and it happens as a matter of probability or chance, this is nothing more than a shader core that is *capable of* executing a first task and a second task simultaneously. Claim 15 clearly requires something more—that the shader core be designed or programmed to cause the simultaneous execution of two tasks from the same plurality of tasks.

2. “Engine / Plurality of Engines”

a. The plain and ordinary meaning of the term “engine” is that it processes data

The plain and ordinary meaning of the term “engine” is that it *processes data*. Tr. (Pfister) at 609:3-13; 706:5-23; 801:19-21. Dr. Pfister’s testimony is unrebutted, as Dr. Reinman expressly stated that he did not form an opinion as to “whether the claimed engine must be able to perform some type of processing.” Tr. (Reinman) at 389:16-21.

The basic understanding of “engine” articulated by Dr. Pfister is supported throughout the specification of the ’381 patent. The “plurality of engines” is consistently described in the specification as comprising the *command processor* of the GPU. JX-0003.0003-7,11 at Fig. 2, 3, 4, 6:65-7:21. The specification further discloses that “[t]he different virtual engines of command processor 230 process different types of tasks.” *Id.* at 7:2-21. For example, “Primary 3D engine 202D processes standard graphics-processing tasks, and low-latency 3D engine 202E processes high-priority graphics-processing tasks.” *Id.* The specification discloses that a “graphics processing task performed by a GPU typically involves complex mathematical computations,” and mathematical computations necessarily involve data. *Id.* at 1:47-49. Moreover, the ALJ has construed “task” to include a “unit of processing work,” which necessarily includes data, as processing necessarily includes performing mathematical computations.

[REDACTED]

The fundamental objective of claim construction is to ascertain “the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention.” *Phillips*, 415 F.3d at 1313. Dr. Pfister’s testimony that a POSITA would understand “engine” in the context of the ’381 patent to have the basic requirement of *processing data* is unrebutted, supported by the specification, and should be adopted as part of any construction of “engine.”

b. The term “plurality of engines” should be construed to exclude a hardware-only embodiment, which is not supported by the specification

In Order No. 31, the ALJ determined that the term “plurality of engines” or “engine” in the ’381 patent was not *limited* to either a hardware-only or a software-only implementation, but otherwise left the term for construction after the hearing. As explained herein, a “hardware-only” implementation of the “plurality of engines” is not supported by the specification, and thus, the term should not be construed to encompass a hardware-only embodiment.⁷

When holding that the term “plurality of engines” can include *a combination* of hardware and software, the ALJ relied on a statement in the specification that “[e]mbodiments of the present invention may be implemented using hardware, software or a combination thereof.” Order No. 31 at 39-41, citing JX-0003.0013 at 10:59-60, 10:62-12:43. However, nothing in this section of the patent (“Example Computer Implementation”) describes or enables any embodiment of *the “plurality of engines”* that is comprised *only* of hardware. While this section of the specification

⁷ As expressed herein, the specification does not provide any written description for a hardware-only embodiment of “plurality of engines” or enable a hardware-only embodiment. “Claims amenable to more than one construction should, when it is reasonably possible to do so, be construed to preserve their validity.” *Karsten Mfg. Corp. v. Cleveland Golf Co.*, 242 F.3d 1376, 1384 (Fed. Cir. 2001); *see, e.g., Cabinet Vision v. Cabnetware*, No. 99-1050, 230 F.3d 1382, 2000 WL 158463, at *6 (Fed. Cir. Feb. 14, 2000) (construing “user-defined standards” narrowly because a broader construction would not satisfy 35 U.S.C. § 112).

[REDACTED]

states that “[e]mbodiments *of the present invention* may be implemented using hardware, software, or a combination thereof,” what is referred to as “the present invention” is a reference to *the entirety of what is claimed as the invention, i.e.*, claim 15 of the patent. Claim 15 recites many components that may be implemented in hardware, such as the “memory,” the “first processing unit,” the “second processing unit,” the “bus,” and the “shader core.” However, there is no written description of *the plurality of engines* being implemented only in hardware. In fact, the term “engine” is never used in this section of the patent, and nothing in this section even refers to the functions performed by the plurality of engines according to claim 15, *i.e.*, receiving a plurality of tasks, loading associated state data, and providing the tasks to the shader core. *See id.* at 10:59-12:43. Rather, the section mentions “computer programs” that “when executed, enable the computer system 500 to perform the features of the present invention, as discussed herein,” *id.* at 11:64-66, which a POSITA would understand as including the “plurality of engines” as being software, which the rest of the specification describes as “virtual engines.” *See, e.g., id.* at Abstract, 3:1-6, 5:29-32, 6:1-4, 6:67-7:4, 8:8-17, 9:45-52, 10:33-39. Even the limited references to “engines” without the modifier “virtual” are clearly identified as “virtual engines” in context. *Id.* at 7:1-19, 7:34-41, 8:10-14. Accordingly, there are no disclosures in the specification of the “plurality of engines” as being anything other than “virtual” engines.

Consistent with the specification, Dr. Pfister testified at the hearing that while some of the claimed components can be implemented in hardware and the claimed plurality of engines can be a combination of software running on hardware, the claimed plurality of engines cannot be “purely hardware” read in the context of the specification of the ’381 patent, because it only discloses “virtual engines” as “software that runs on top of the GPU.” Tr. (Pfister) at 609:3-611:8. As Dr. Pfister explained, not only does the term “virtual” denote software, but the engines must be

[REDACTED]

software in order to use the resources of the GPU as described in the specification. *Id.* Dr. Pedram likewise testified that the term “virtual” “to a person skilled in the art would really mean software, and Dr. Pedram further testified that there is not even one embodiment in the specification that has an engine implemented in a hardware-only embodiment. Tr. (Pedram) at 920:18-922:6. This understanding is supported by the specification, which describes the “virtual engines” as having “access to,” “sharing” and “using” hardware resources of the GPU. JX-0003.0012 at 7:1-19.

The testimony from Dr. Pfister and Dr. Pedram on this issue was effectively unrebutted. Dr. Reinman testified that he did not form an opinion as to whether the claimed “plurality of engines” requires a particular structure, such as hardware or software. Tr. (Reinman) at 389:22-390:2. And while Dr. Lastra discussed the possibility of “hardware being virtualized by using hardware components,” Tr. (Lastra) at 1294:6-1295:5, Dr. Lastra was unable to point to any part of the specification to support his speculation that hardware was being “virtualized” in the context of the ’381 Patent. Extrinsic evidence, such as technical dictionaries or expert testimony, cannot be used to contradict the scope and meaning of a term that a POSITA would ascertain by reading the specification and the intrinsic record. *Grace Instrument Indus., LLC v. Chandler Instruments Co.*, 57 F.4th 1001, 1010 (Fed. Cir. 2023).

Accordingly, since the specification only describes “virtual engines” and never discloses or provides written description support for a hardware-only embodiment of the “plurality of engines”—and Dr. Lastra failed to cite any support from the ’381 patent that is inconsistent with the opinions of Dr. Pfister and Dr. Pedram—the term “engine” or “plurality of engines” cannot be construed so broadly as to encompass *a hardware-only* embodiment or implementation.

c. Complainants’ proposed requirement of “servicing a command buffer” lacks any basis in the claim

During the hearing, Dr. Reinman testified that “an engine must be configured to service a

[REDACTED]

command buffer.” Tr. (Reinman) at 132:1-24. *See also* CPrHB at 121-122. Dr. Reinman and Complainants based their position on citations to the specification that refer to “a plurality of command buffers.” *Id.* (citing to JX-0003.0009 at 2:10-19, 10:25-35). But the first citation at 2:8-19 is describing the prior art, and 10:25-35 merely describes one embodiment. This is made more explicit elsewhere in the specification, where it states that “[i]n **embodiments**, the GPU processes a plurality of command buffers.” *Id.* at 5:4-5. Moreover, the specification specifically teaches that the use of “command buffers” is an optional embodiment: “In an embodiment ... In **another** embodiment, CPU 102 sends instructions ... to respective command buffers.” *Id.* at 6:28-32 (emphasis added). Moreover, in yet another embodiment, the specification discloses that the “command processor” (which “includes a plurality of virtual engines”) “receives tasks ... from **one or more** command buffers.” *Id.* at 6:65-7:4. Insofar as the specification expressly discloses that there may be only one command buffer servicing the plurality of engines, this directly contradicts Complainants’ proposed construction.

Moreover, Complainants’ proposed requirement that each engine “service its own command buffer” violates numerous cardinal rules of claim construction. Nothing in claim 15 recites or requires a “command buffer.” Accordingly, Complainants are seeking to impermissibly add an entirely new structural limitation to the claim. The Federal Circuit has called it “one of the cardinal sins of patent law—reading a limitation from the written description into the claims.” *Phillips*, 415 F.3d at 1320 (citation omitted). “[A] claim construction must not import limitations from the specification into the claims.” *Douglas Dynamics, LLC v. Buyers Prods. Co.*, 717 F.3d 1336, 1342 (Fed. Cir. 2013). Moreover, courts “normally do not interpret claim terms in a way that excludes embodiments disclosed in the specification,” and Complainants have not provided probative evidence for departing from that general principle. *SIMO Holdings Inc. v. Hong Kong*

[REDACTED]

uCloudlink Network Tech. Ltd., 983 F.3d 1367, 1378 (Fed. Cir. 2021) (citing *Oatey Co. v. IPS Corp.*, 514 F.3d 1271, 1276 (Fed. Cir. 2008)). Adding an additional structural limitation to claim 15, that each “engine” must have its own “command buffer,” violates each of these cardinal rules of claim construction. *See also* Tr. (Pfister) at 612:6-11 (disagreeing that each of the plurality of engines must service its own command buffer).

Complainants also argue that an “engine” “must be independently configured to (i) “receive at least one task” and to (ii) “load state data of the received tasks.” Tr. (Reinman) at 131:8-25; CP rHB at 120-121. However, claim 15 already recites “a plurality of engines configured to receive, from a scheduling module associated with a second processing unit, a plurality of tasks and to load state data associated with each of the plurality of tasks.” These two proposed “requirements” for the term “engine” merely paraphrase other language of the claim. “Construing a claim term to include features of that term already recited in the claims would make those expressly recited features redundant.” *Apple, Inc. v. Ameranth, Inc.*, 842 F.3d 1229, 1237-38 (Fed. Cir. 2016). Moreover, Complainants improperly seek to rewrite the rest of the claim limitation to require that each engine only receive and load state data for a *single* task, rather than a plurality of tasks, which contradicts other language in the claim, as explained *infra* in the next section.

3. “Each of the plurality of tasks”

Claim 15 recites in relevant part, “a plurality of engines configured to receive ... a plurality of tasks and to load state data associated with each of the plurality of tasks.” As Complainants have argued, the article “a” is generally construed to mean “one or more” “unless the language of the claims themselves, the specification, or the prosecution history necessitate[s] a departure from the rule.” *SanDisk Corp. v. Kingston Tech. Co.*, 695 F.3d 1348, 1360 (Fed. Cir. 2012) (citation omitted). In this context, the claim term “configured to receive ... a plurality of tasks” should be construed to mean “configured to receive ... [one or more pluralities] of tasks,” as nothing in the

[REDACTED]

intrinsic evidence necessitates a different construction.

Accordingly, when reading the next clause of the claim, “... and to load state data associated with **each** of the plurality of tasks,” it is clear that each engine must be configured to load state data for **each** of the pluralities of tasks that are received. If the plurality of engines were intended to only receive a single “plurality of tasks” and load state data associated with that single plurality of tasks, the claim would have read “load state data associated with each of the tasks in the plurality of tasks.” But it is a well-settled principle that courts must “construe the claim as written, not as the patentees wish they had written it. *Ecolab, Inc. v. FMC Corp.*, 569 F.3d 1335, 1344-45 (Fed. Cir. 2009). The claim as written requires that the plurality of engines be configured to load state data for each plurality of tasks. Thus, each engine must be configured to load state data for at least two tasks (*i.e.*, there must be two pluralities of tasks present in the plurality of engines). If each engine is **only** “configured to” or “designed to” receive and load state data for **a single** task and is not also designed to receive and load state data for a plurality of tasks, it does not satisfy the literal requirements of the claim.

Complainants point to an example embodiment in the specification that refers to the OS scheduler issuing a first task to one virtual engine and a second task to a second virtual engine. CPrHB at 122-123, citing 9:34-36. Complainants argue that Respondents’ construction would improperly “read out” this embodiment from the scope of the claim. To the contrary, under Respondents’ construction, the claimed plurality of engines would still have the capability to receive and load state data for a single task. Nothing in Respondents’ proposed construction excludes such a scenario. However, under the plain language of the claim, each engine must *also* be configured to receive a plurality of tasks and to load state data for each plurality. It is this configuration that the applicants chose to claim and must be reflected in any construction of claim

[REDACTED]

15. An engine that is not configured to receive and load state data for a plurality of tasks (as well as a single task, depending on what is issued to it by the scheduling module) does not meet the plain language of the claim. Moreover, while the single line of the specification relied upon in the specification refers to one task being issued to each engine, it certainly does not preclude the possibility of a second task being issued to each engine, such that each engine must be able to receive and load state data for the plurality of tasks issued to it by the scheduling module.

As the ALJ recognized at the conclusion of the hearing, it does not appear that the parties should actually have a dispute on many of the claim terms at issue. That appears to be the case with this construction, because Complainants' expert, Dr. Reinman, expressly agreed with Respondents' construction. Dr. Reinman was asked:

Q. In addition to receiving a plurality of tasks, each engine has to also load state data for a plurality of tasks, correct?

A. Yes, sir.

Tr. (Reinman) at 398:21-24. Thus, the intrinsic evidence provides only one possible construction of claim 15 and that is affirmed by the fact that both experts agree on that construction.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

D. Invalidity

1. Anticipation Under 35 U.S.C. § 102(a)

Two prior art references, Jiao and Stuttard, each invalidate all Asserted Claims of the '381 Patent in this Investigation. Jiao is a compelling reference that anticipates every element of the Asserted Claims. Stuttard was discussed above with respect to the '053 Patent and anticipates every claim limitation.

AMD's primary disputes are based on contrived claim constructions that belie the plain and ordinary meaning.

As discussed below, at least according to the interpretations and constructions proposed by AMD in their infringement contentions and/or expert reports, the prior art discloses and/or renders obvious every asserted claim.

a. Jiao

(1) Overview of Jiao

It is undisputed that U.S. Patent No. 8,319,774 qualifies as prior art under 35 U.S.C. § 102(b). Jiao was issued on November 27, 2012, and identifies the inventors as Yang Jiao, Yijung Su, and John Brothers. RX-0423.00002.

[REDACTED]

Jiao discloses a computing system for graphics processing, which comprises a system memory, a computational core, and a bus coupling the two. RX-0423.00013–14; RX-0423.00002; Tr. (Pedram) at 928:20-929:2; RDX-0004C.85. Jiao's computing system for graphics processing comprises a display device and local memory 106 coupled to a graphics processing unit (GPU) 114. RX-0423.00013–14 at 4:49–5:67; Tr. (Pedram) at 928:20-929:18. This GPU includes a graphics processor 202, which comprises the compiler 115, a multiple execution unit (EU) computational core 204, an EU pool control unit 206, and a graphics pipeline with fixed function logic 208. RX-0423.00013 at 5:33-57; Tr. (Pedram) at 928:20-929:18. The computational core 204 comprises an execution unit pool (EUP) 414, with one embodiment comprising one or more execution units (Eus) 420a through 420h. RX-0423.00015 at 8:15-18. Further, Jiao discloses that the EUP 414 is configured to receive a plurality of tasks from at least a vertex shader, a geometry shader, and/or a pixel shader in the graphics pipeline 208. *Id.* at 5:33-57.

Jiao discloses a plurality of 512-bit buses coupled to the computational core 204 and the memory in graphics pipeline 204 and/or the vertex/stream caches in the EU pool control unit 206. RX-0423.00013 at 4:49-5:6; RX-0423.00015 at 7:35-55. Jiao discloses that a plurality of engines including vertex, geometry, and pixel shaders is configured to receive a plurality of tasks from the EU pool control unit 206 and that the shaders load constants, rendering context information, and graphics data before commencing execution of a task. RX-0423.00016 at 9:25-38. Constants are provided by a constant buffer of a constant buffer system 200. RX-0423.00014 at 6:24-27; Tr. (Pedram) at 928:3-16; 936:17-938:1 RDX-0004C.85.

In an embodiment, each EU is configured for simultaneous multi- threaded operations running multiple contexts. RX-0423.00016 at 9:33-38. Jiao also teaches that a low-latency task is associated with a slow access path to constants using the system's memory hierarchy, whereas

[REDACTED]

a regular-latency task is associated with a fast access path to constants using the constant buffers in each EU 420. *Id.* at 10:13-44. Finally, Jiao discloses a graphic processor system, configurable as a computer system, comprising components such as arithmetic logic units and other logic and performs functionalities suitable for graphics-processing tasks and general- compute tasks. RX-0423.00017 at 11:4-13; Tr. (Pedram) at 949:12-950:12; RDX-0004C.117.

A POSITA would have understood that the concepts, components and features apply equally across each embodiment, that is, a POSITA would not have understood them to be mutually exclusive and they could be “mixed and matched” within the spirit of the disclosure. Moreover, a POSITA would have been motivated to mix and match them as they all are disclosed in the same reference and are directed to the same field of invention and solving the same problems. *See* Tr. (Pedram) at 929:3-25 (the computational core “acts as the claimed shader core.”).

(2) Independent Claim 15

(a) 15[preamble]: A computing system comprising:

All parties agree that the preamble not limiting. Regardless, AMD does not dispute that Jiao discloses this element. Tr. (Pedram) at 931:12-17; RDX-0004C.84; Tr. (Hart) at 1267:5-1277:16; CDX-0008C.68-91. Jiao discloses graphics processing/processor systems. Tr. (Pedram) at 931:12-17; RDX-0004C.84. A POSITA would understand Jiao’s graphics processing systems (*e.g.*, graphics processor system 100) to be a computing system. RX-0407C ¶216. Indeed, Jiao expressly discloses that the graphics processor system is implemented as a “computer system.” RX-0423.00013 at 4:54-55. Figure 1 shows a “computer system” embodiment of Jiao’s graphics processor system. RX-0423.00013. at Fig. 1; Tr. (Pedram) at 931:12-17; RDX-0004C.84.

(b) 15[a]: a memory;

Jiao discloses this element, and AMD does not dispute that Jiao discloses this element. Tr. (Pedram) at 931:20-932:3; RDX-0004C.85; *see* Tr. (Hart) at 1267:5-1277:16; *see also* CDX-0008C.68-91. Specifically, Jiao discloses system memory 124, local memory 106 (*e.g.*, which may comprise a display buffer, frame buffer, texture buffer, command buffer, etc.), memory 306 and various caches (*e.g.*, depth or Z-cache, color or D-cache, texture or T-cache, etc.) in the logical graphics pipeline of Figure 3, a vertex/stream cache in EU pool control and vertex/stream cache 206, and L2 cache 422 in computational core 204. *See, e.g.*, RX-0423.0003-7, 13-15 at 4:49-5:6, 33-57, 6:1-48, 6:65-7:18-27, 35-55, 8:15-41, Figures 1-4; RX-0407C ¶219; Tr. (Pedram) at 931:20-932:3; RDX-0004C.85.

(c) 15[b]: a first processing unit; and

Jiao discloses this element, and AMD does not dispute that Jiao discloses this element. Tr. at 1267:5-1277:16; *see* Tr. (Pedram) at 932:4-8; RDX-0004C.85, 87-90; Tr. (Hart) at 1267:5-1277:16; *see also* CDX-0008C.68-91. Jiao discloses a computational core 204 (also known as a software programmable core processing unit) within a graphic processing unit (GPU 114). RX-0423.0003-7, 13-15 at 4:49-5:57, 5:47-51, 6:1-48, 6:49-7:55, Figures 1-4; Tr. (Pedram) at 932:4-8; RDX-0004C.85, 87-90.

(d) 15[c]: a bus coupled to the memory and the first processing unit,

Jiao discloses this element, and AMD does not dispute that Jiao discloses this element. Tr. at 932:9-24; RDX-0004C.85-86; *see* Tr. (Hart) at 1267:5-1277:16; *see also* CDX-0008C.68-91. Specifically, Jiao discloses a bus coupled to local memory 106 and the GPU via a memory interface unit (MIU 110), and a plurality of 512-bit buses coupled to memory 306 and graphics processor 202 (hence, computational core 204), and 512-bit buses coupled to vertex/stream cache and

[REDACTED]

computational core 204.). RX-0423.00003, 13-15 at 4:49- 5:6, 7:19-55; *see also* Figure 1. Alternatively, Jiao also discloses a bus coupled to system memory 124 and the GPU (hence, computational core 204) via PCI-E BIU 118. RX-0423.00013 at 4:49-5:6 and Figure 1; Tr. (Pedram) at 932:9-24; RDX-0004C.85-86.

- (e) **15[d]: wherein the first processing unit comprises: a plurality of engines configured to receive, from a scheduling module associated with a second processing unit, a plurality of tasks and to load state data associated with each of the plurality of tasks; and**

Jiao discloses this element. Tr. (Pedram) at 932:25-938:17; RDX-0004C.87-101. For example, Jiao's computational core 204 comprises vertex, geometry, and pixel shaders. Tr. (Pedram) at 933:13-934:8; RX-0423.00014 at 5:47-51 ("The computational core 204 comprises a pool of multiple execution units...with various shader programs, including a vertex shader, geometry shader, and/or pixel shader"). A POSITA would have understood the vertex shader 304, geometry shader 306, and pixel shader 310 in graphics pipeline 208 of GPU 114 to be "engines." *See* Tr. (Pedram) at 934:1-8. These shaders are configured to receive and process graphics data (e.g., graphics primitives) to produce pixel data. RX-0423.00014 at 6:1-48, 49-64. At the hearing, Dr. Hart refused to provide an opinion as to whether the vertex, geometry, and pixel shaders were part of the computation core. Tr. (Hart) at 1321:8-12.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] Jiao teaches that there exist "thread slots" within each of the execution units 420 of the EU pool 414 and that "the EU pool control unit 206 assigns a thread to one of the empty thread slots in the respective execution unit 420." Tr. (Pedram) at 948:25-949:3; RX-0423.00016 at

[REDACTED]

9:15-21. Further, Jiao discloses that each execution unit 420 of the EU pool 414 may store a plurality of tasks or threads (*e.g.*, in some embodiments up to 32) at the same time. Tr. (Pedram) at 949:8-11; RX-0423.00016 at 9:11-15.

Jiao discloses that the plurality of vertex, geometry, and pixel shaders (“engines”) are configured to receive a plurality of tasks from the EU pool control unit 206. Tr. (Pedram) at 935:14-936:16.; RDX-0004C.97-99; *see* RX-0423.00003, 6, 14 at 5:17-23 and 6:65-7:27 and Fig. 1 and 4. The shaders load constants, rendering context information, and graphics data before commencing execution of any tasks. *See, e.g.*, RX-0423.00003, 13-15 at 6:13- 44; 65-7:27, 8:35-41, 9:15-24, 9:39-54, Fig. 3 and 4; Tr. (Pedram) at 936:17-938:1; RDX-0004C.100-101.

In an effort to avoid Jiao’s disclosure, Dr. Hart admitted that he did not apply the ’court’s construction of “task,” (“a unit of processing work”), when opining whether Jiao discloses vertex, geometry, and pixel shaders configured to receive tasks. Tr. (Hart) at 1328:2-20. Thus, Dr. Hart’s ultimate opinion must be disregarded, as Dr. Hart instead used his own construction based on “how [Dr. Hart interprets] it’s being used in a claim.” *Id.*

Dr. Hart also opined that the “engine” requirement is not met because the plurality of shader programs do not receive the “plurality of tasks,” arguing instead that Jiao’s shader programs generate tasks. Tr. (Hart) at 1326:14-1327:15. Dr. Hart based his opinion on the excerpt: “As various shader tasks are generated by the vertex shader 304, geometry shader 306, and pixel shader 310, they are delivered to the respective execution units 420 to be carried out.” RX-0423.00016 at 9:35-38. Similar to how AMD is reading the claim for infringement, Jiao’s vertex shader 304, geometry shader 306, and pixel shader 310, receive graphics (high level) tasks from the driver software 128 residing in the system memory 124 via the EU pool control unit 206 and produce (low level) shader tasks that are then mapped to threads of the various execution units 420 by the

[REDACTED]

EU pool control unit 206. *Id.* at 9:43-49, at 5:47-51. A POSITA would have understood that Jiao discloses a high-level task that has given rise to the low-level task that is mapped to a thread. *See* Tr. (Pedram) at 936:17-938:1; RDX-0004C.100-101.

Based on the ALJ's construction, Jiao discloses the claimed tasks. *See, e.g.,* RX-0423.00016 at 9:42-47 ("...the EU pool control unit 206 is responsible for assigning vertex shader, geometry shader, and pixel shader tasks to threads of the various execution units 420, and also performs an associated 'bookkeeping' of the tasks and threads."). *See, e.g.,* RX-0423.00015-16 at 8:65-67 and 9:10-13 (sometimes using task and thread interchangeably). Thus, a POSITA would have understood Jiao's inherent disclosure of a high-level task to be the construed "task." To the extent not considered an express disclosure, for the same reasons explained above, Jiao's disclosure would have also rendered this claim limitation obvious. *See* Tr. (Pedram) at 936:17-938:1; RDX-0004C.100-101.

The '381 Patent uses the term "command" as an example of a graphics task. The '381 Patent states: "To pass tasks to the GPU, an operating-system (OS) scheduler stores the tasks in a command buffer...processes one command buffer at a time....[and] serially places tasks in the command buffer, and the GPU typically processes the tasks in the order in which they are placed in the command buffer," and "Command processor 230 receives tasks (*e.g.*, graphics processing and general-compute tasks) from one or more command buffers filled by the OS scheduler." *See e.g.,* JX-0003.0009,11 at 2:8-14 and 6:65-67. "Commands" in the '381 Patent, satisfies the ALJ's construction of the term "tasks". Jiao uses "command" similarly: "The system memory 124...communicates instruction sets or commands through the use of the CPU 126 to registers in the GPU 114, and an applications programming interface (API) space 130, from which constants are either memory mapped or constant buffer mapped by a compiler 115 in the GPU 114." RX-

[REDACTED]

0423.00014 at 5:17-23. “Responsive to a command from the driver software 128..., vertex information is passed to vertex shader 304 in the computational core 204 to implement vertex transformations.” RX-0423.00015 at 7:57-62. In another instance, Dr. Hart refused to opine regarding whether a single command could be a task. Tr. (Hart) at 1331:2-4.

Jiao’s EU pool control unit 206 is the “scheduling module.” Tr. (Pedram) at 934:9-21; RDX-0004C.93-94. The EU pool control unit 206 sends to the vertex, geometry, and pixel shaders (“engines”) various tasks. For example, the EU pool control unit 206 “functions as a global scheduler for the tasks” (while also being responsible for assigning vertex shader, geometry shader, and pixel shader tasks to threads of the various execution units 420”). *See, e.g.*, RX-0423.00005-6, 14-16 at 6:65-7:27, 9:15-24, 9:39-54, Fig. 3 and 4. Further, Jiao’s EU pool control unit 206 is associated with the command stream processor 406, which is the claimed “second processing unit.” Tr. (Pedram) at 934:22-935:13; RDX.0004C.95-96; *see, e.g.*, RX-0423.00015 at 7:19-27 (“The command stream processor 406 provides triangle vertex indices to the EU pool control unit 206....The EU pool control unit 206 assembles vertex shader inputs from the stream cache and sends data to the computational core 204 (input E)...[and] assembles geometry shader inputs and provides those inputs to the computational core 204 (input F)... control[ling] the respective inflow and outflow to the computational core 204.”). The command stream processor 406 is a processing unit associated with EU pool control unit 206 at least because it performs “new constant update[s]” for use by the EU pool control logic, which assembles inputs for shader tasks. RX-0423.00005-6, 14-16 at 3:48-60, 4:10-23, 10 55:11-46, 14:13-23; RDX-0004C.101.

Alternatively, the driver software 128 constitutes the claimed “scheduling module.” Tr. (Pedram) at 935:15-936:4; RDX-0004C.93-94. Jiao discloses that the driver software “communicates instruction sets or commands through the use of the CPU 126 to registers in the

[REDACTED]

GPU 114.” RX-0423.00014 at 5:17-23. Responsive to a command from the driver software 128, vertex information is passed to the vertex shader 304 in the computational core 204 to implement vertex transformations, and after processing in the fixed-function graphics engine 208, are passed to the pixel shader 310 in the computational core 204. RX-0423.00015 at 7:57-8:14. And, Jiao’s driver software 128 is associated with the CPU 126, which is the claimed “second processing unit.” *See, e.g.*, RX-0423.00014 at 5:8-13; Tr. (Pedram) at 934:24-935:13; RDX-0004C.95-96.

Dr. Hart reiterated that he applied “plain and ordinary” meaning for all terms not construed, including the term “associated with.” Hr. Tran (Hart) at 1310:13-17, 1311:15-1312:22. But then AMD construes, through Dr. Hart, the term “associated with” to mean “part of” to avoid the prior art. Tr. (Hart) at 1311:15-1312:22. AMD’s construction is not the plain and ordinary meaning.

A POSITA would have understood the plain and ordinary meaning of “associated with” to simply be “has a relationship with,” either physical or functional. *See* Tr. (Pedram) at 936:17-938:1. AMD’s construction restricts “associated with” to physical relation, and further restricts the physical association to mean it must be “part of,” excluding obvious physical associations like dedicated bus connections. Tr. (Hart) at 1311:15-1312:22. Captive components, whose sole purpose or function is to be responsive to or provide functionality to another component, are not technically “part of” but are certainly “associated with.” *See id.*

AMD’s construction also ignores components having more than one “association.” Tr. (Hart) at 1312:8-15. Nothing in the claims prevents the first and second processing units from having additional associations that overlap or being associated with each other. *See* JX-0003.0015 at 14:1-37. Claim 15 has no requirements for any associations of the first processing unit. *Id.* It only requires that the scheduling module be associated with a second processing unit. *Id.*

Dr. Hart’s moves away from “plain and ordinary meaning” by basing his construction of

[REDACTED]

“associated with” on statements in the file history. *See, e.g.*, Tr. (Hart) at 1310:13-1312:14 But even that reliance is flawed. First, the patentee’s statements are directed to a claim limitation not recited in the Asserted Claims. JX-0009.0185 (arguing the prior art does not teach or suggest “a *shader core associated with the first processing unit and configured to receive the plurality of tasks from at least one of the plurality of engines,*” in which the plurality of engines are “associated with a first processing unit and configured to receive, *from a scheduling module associated with a second processing unit*). *Id.* (emphasis in original by patentee).¹⁷ The Asserted Claims have no “association” requirement for the “first processing unit.” *Id.*

Second, the focus of the argument in the file history is what constitutes a processing unit—not “associated with.” *See* JX-0009.0185 (“Thread scheduler 210 and the hardware units are not associated with different processing units; rather, they are associated with the same processing unit — namely shader core 130.”) (emphasis in original).

Third, nothing in the claim or the patentee’s statements rises to the level of clear disavowal, which would prevent a single processor from having multiple different processing units (*i.e.*, a first and second) embedded within it. *See id.* The examiner only identified a single processing unit, but Respondents identified two, with their own associations. Tr. (Pedram) at 1311:15-1312:22. And nothing, in the claims or the patentee’s statements, prevent the first and second processing unit from having more than one “associations,” even if they overlap. *Id.* For example, both the first and second processing units may be associated with a GPU, and simultaneously have respective separate associations (*e.g.*, the first processing unit with a shader core while the second processing unit is a scheduling module).

¹⁷ The Jiao reference cited in JX-0009 is not the Jiao reference asserted herein.

[REDACTED]

Fourth, AMD is essentially rewriting the term “associated with” to mean “comprising.” *See* Tr. (Hart) at 1311:15-1312:22. The Asserted Claims do not require any “association” with the first processing unit or preclude overlapping associations of any other elements. *See id.* Dr. Hart’s rebuttal relies on this this untimely construction. *See* Tr. (Hart) at 1311:15-1312:22.

Lastly, “state data” refers to “texture handles, shader constants, transform matrices, or the like” associated with tasks. *See, e.g.*, JX-0003.0009, 10 at 1:53-57, 3:3-7. Jiao uses the phrase “rendering context” to refer to “specific visual properties related to the appearance of surfaces (*e.g.*, materials, reflectance, shape, textures, etc.) and/or operations related to the same (*e.g.*, required transformations) which may collectively be referred to as a rendering context for this object or group of objects.” RX-0423.00012 at 1:50-55. “When rendering an object or group of objects for a given rendering context, a shader utilizes a constant data pool (*e.g.*, context-specific constants stored in a constant buffer).” *Id.* at 1:55-58.

Jiao discloses that its shaders load constants, rendering context information, and graphics data before commencing execution of a task. Tr. (Pedram) at 936:17-938:17; RDX-0004C.100-101. Further, constants for performing one or more of the shader computations are provided by a constant buffer of a constant buffer system 200. RX-0423.00014-16 at. RX-0423.0005-6, 16-18 at 6:1- 7:55, 8:65-9:14; 9:15-24, 12:33-52, 12:63-13:6, Figs. 3 and 4; Tr. (Pedram) at 936:17-938:17; RDX-0004C.100-101.

- (f) **15[e]: a shader core configured to receive the plurality of tasks from at least one of the plurality of engines and to execute a first task from the plurality of tasks while executing a second task from the plurality of tasks based on respective state data associated with each of the first and second tasks.**

Jiao’s computational core 204 comprises an execution unit pool (EUP) 414, which comprises one or more execution units (EUs) 420 a through 420 h (collectively referred to herein

[REDACTED]

as EUs 420). Tr. (Pedram) at 939:22-942:11; RX-0423.00015 at 8:15-18. Each of the EUs 420 includes control logic and local memory, and each EU 420 is capable of processing multiple instructions within a single clock cycle. RX-0423.00015 at 8:18-21. The EUP is configured to receive the plurality of tasks from at least a vertex shader, a geometry shader, and/or a pixel shader (i.e., the “engines”) in graphics pipeline processing. *See, e.g.*, RX-0423.00016 at 9:35-38 (“As various shader tasks are generated by the vertex shader 304, geometry shader 306, and pixel shader 310, they are delivered to the respective execution units 420 to be carried out.”). A POSITA would have understood Jiao’s EU pool (EUP) 414 to be a “shader core” as recited in claim 15. *See* Tr. (Pedram) at 938:21-939:3; RDX-0004C.102.

Jiao’s EUP is configured to execute multiple, simultaneous tasks based on respective state data including constants, rendering context information, and graphics data. *See, e.g.*, RX-0423.00013-15 at 3:48-60, 6:49-7:18. Further, the EUP is configured to execute a first task while executing a second task. *See, e.g.*, RX-0423.00015 at 8:21-23 (“Thus, the EU pool 414, at its peak, can process multiple threads simultaneously or substantially simultaneously.”). Each task requires its corresponding context-specific constants to be able to be mapped or executed on some execution unit of the computational core. *See, e.g.*, RX-0423.00016 at 1:43-58, 6:49-7:18, 8:15-9:14, 9:25-54, 9:65-10:8; *see also* Tr. (Pedram) at 936:17-938:1; RDX-0004C.104.

Dr. Hart argues that “executing multiple tasks received from one or more of the engines, including executing a first task while executing a second task based on their respective state data” is not disclosed. Tr. (Hart) at 1275:16-1276:19. Although Jiao teaches that multiple threads can be executed in parallel, Dr. Hart believes that these threads are not associated with the tasks required by the Asserted Claims. Tr. (Hart) at 1275:16-1277:12. However, consistent with AMD’s infringement theory, a POSITA would know that Jiao’s high-level tasks (commands) are processed

[REDACTED]

by the vertex shader 304, geometry shader 306, and pixel shader 310 and mapped to low level tasks running on threads of the execution units of the computational core by the EU pool control unit.

- (3) **Dependent Claim 16: The computing system of claim 15, wherein the shader core comprises:
a first plurality of processing elements configured to execute the first task based on a first set of state data associated with the first task; and
a second plurality of processing elements configured to execute the second task based on a second set of state data associated with the second task.**

Jiao discloses this element. Tr. (Pedram) at 946:6-948:3; RDX-0004C.110-112. Jiao discloses that the EU pool (EUP) 414 (which is the claimed “shader core”) comprises a plurality of EUs 420. RX-0423.00015 at 8:15-18. “Each of the EUs 420 includes control logic and local memory, as described further below, and each EU 420 is capable of processing multiple instructions within a single clock cycle,” and “can process multiple threads simultaneously or substantially simultaneously.” *Id.* at 8:18-23. Jiao’s EUP is configured to execute multiple threads simultaneously or substantially simultaneously based on respective state data including constants, rendering context information, and graphics data. *See, e.g.*, RX-0423.00014 at 6:1-64. Further, “[e]ach of these execution units 420 is identical in design and configurable for programmed operation . . . [and], each execution unit 420 is capable of multi-threaded operations simultaneously. As various shader tasks are generated by the vertex shader 304, geometry shader 306, and pixel shader 310, they are delivered to the respective execution units 420 to be carried out.” *See, e.g.*, at 9:25-38. A POSITA would have understood Jiao’s plurality of EUs 420 to constitute the claimed “first and second pluralities of processing elements.” Tr. (Pedram) at 946:6-949:3; RX-0407C ¶235.

[REDACTED]

[REDACTED]

(4) Dependent Claim 17: The computing system of claim 15, wherein the plurality of engines comprises a first queue configured to receive the first task and a second queue configured to receive the second task.

Jiao discloses this element. Tr. (Pedram) at 948:4-949:11; RDX-0004C.113-116. Jiao discloses that each execution unit 420 of the EU pool 414 may store a plurality of tasks or threads (*e.g.*, in some embodiments up to 32) at the same time. Tr. (Pedram) at 949:9-11; RDX-0004C.114; RX-0423.00016 at 9:11-15. Jiao also teaches that there exist “thread slots” within each of the execution units 420 of the EU pool 414 and that “the EU pool control unit 206 assigns a thread to one of the empty thread slots in the respective execution unit 420.” RX-0423.00016 at 9:15-21; *see also* Tr. (Pedram) at 948:22-949:3; RDX-0004C.114-115. Jiao further explains that as individual tasks are generated, the EU pool control unit 206 handles assignment of these tasks to available threads within the various execution units 420 and that as tasks are completed, the EU pool control unit 206 further manages the release of the relevant threads. Tr. (Pedram) at 948:22-949:11. In this regard, the EU pool control unit 206 is responsible for assigning vertex shader, geometry shader, and pixel shader tasks to threads of the various execution units 420. RX-0423.00016 at 9:39-47. Therefore, Jiao teaches that there are buffers (*e.g.*, queues) with multiple thread slots (up to 32 in some embodiments) in each of its execution units. Tr. (Pedram) at 948:4-949:11; RDX-0004C.113-116.

Alternatively, where the claimed “engines” include the vertex shader 304, geometry shader 306, and pixel shader 310 in the graphics pipeline 208 of GPU 114 plus the “thread slots” within the execution units 420 of the EU pool 414, Jiao teaches that engines comprise queues, one per a group of (up to 32) threads. Tr. (Pedram) at 948:4-949:11. As previously explained, Jiao’s EUP is configured to execute multiple, simultaneous tasks based on respective state data including constants, rendering context information, and graphics data. *See, e.g.*, RX-0423.00013-15 at 3:48-

[REDACTED]

60, 6:49-7:18; *see also* Tr. (Pedram) at 948:4-949:11; RDX-0004C.113-116. Therefore, Jiao teaches that its “engines” comprise a first queue configured to receive the first task and a second queue configured to receive the second task. Tr. (Pedram) at 948:4-949:11; RDX-0004C.114.

(5) Dependent Claim 20: The computing system of claim 15, wherein the shader core is configured to receive the plurality of tasks based on a priority associated with each of the plurality of tasks.

Jiao discloses this element. Tr. (Pedram) at 950:4-952:3; RDX-0004C.118-119. Jiao discloses that the rendering contexts associated with tasks processing have priority levels. Tr. (Pedram) at 950:4-952; *see, e.g.*, RX-0423.00002. “Embodiments of the present disclosure are directed to graphics processing systems, comprising..., wherein one of the execution units is configurable to process a thread corresponding to a rendering context, wherein the rendering context comprises a plurality of constants with a priority level; a constant buffer control unit providing a translation table for the rendering context to map the corresponding constants into the slots of the physical storage space.” RX-0423.00012 at 2:24-36. For example, Jiao explains that the constant blocks may be loaded to the CB system 200 using a fast access path 506 or a slow access path 508 based on the high or low priority level 504 associated with the respective rendering context. RX-0423.00016 at 10:13-54; Tr. (Pedram) at 950:13-952:3; RDX-0004C.118.

a. Stuttard

(1) Independent Claim 15

Stuttard anticipates all of the limitations of claim 15. RDX-0004C.145; Tr. (Pedram) at 972:9-14.

(a) Claim 15[preamble]

The preamble is not limiting. Regardless, Stuttard discloses a computing system. Precisely, Stuttard describes a graphics system that can be a “stand alone processing system,” which a POSITA would understand is a computing system. RDX-004C.124; RX-0416.0127 at

Figure 1, RX-0416.0127 at 5:1-4; Tr. (Pedram) at 963:21-964:1.

(b) Claims 15[a], 15[b], and 15[c]

Stuttard discloses the claimed memory (local memory 12) and first processing unit (processing core 10). RDX-004C.125; RX-0416.0127 at Figure 1; Tr. (Pedram) at 964:4-10. Primary bus 4 and secondary bus 6 couples the local memory 12 and processing core 10 together. RDX-004C.126-127; RX-0416.0127 at Figure 1; RX-0416.0009 at 6:6-19; Tr. (Pedram) at 964:11-20. Either of the primary bus 4 or secondary bus 6 satisfy the claimed bus. *Id.* None of these claim elements were disputed by Dr. Hart.

(c) Claim 15[d]

(i) Plurality of engines

Stuttard provides additional detail about its processing core 10 (first processing unit) in Figures 3 and 4. RDX-004C.128; RX-0416.0131, 132; Tr. (Pedram) at 964:23-965:17. Dr. Pedram testified that processing core 10 includes a number of engines identified as array controller 104, channel controller 108, and components of the thread manager 102 including cache memory unit 1024, status block 1030, and thread processors 1026. RDX-004C.128; RX-0416.0131-132 at Figures 3 and 4; RX-0416.0009 at 6:6-19; Tr. (Pedram) at 965:19-25.

(ii) Configured to receiving from a scheduling module associated with a second processing unit, a plurality of tasks

Stuttard discloses that the plurality of engines are configured to receive a plurality of tasks from a scheduling module associated with a second processing unit. Stuttard discloses a scheduling module (instruction fetch unit 1023 and thread scheduler 1025, working together) associated with a second processing unit (identified as embedded processor unit EPU 8). RDX-0004C.130; RX-0416.0131-132 at Figures 3 and 4; Tr. (Pedram) at 966:1-11. Dr. Pedram explained that Stuttard discloses that the instructions for a thread are fetched from the local external

[REDACTED]

memory 103 or from the EPU 8 by the fetch unit 1023. RDX-0004C.132; RX-0416.0126 at 23:13-28; Tr. (Pedram) at 966:13-25. These instructions are supplied to the cache memory 1024 (which is one of the identified engines in Stuttard) via connecting logic. RDX-0004C.132; RX-0416.0126 at 23:13-28. The thread scheduler 1025 then schedules the execution of each thread, ensuring that each processor block 106 is kept busy as much as possible. *Id.* Thus, each of the fetch unit 1023 and/or thread scheduler 1025, are associated with the EPU, which directly feeds the fetch unit 1023 the plurality of tasks. Tr. (Pedram) at 966:1-25. As Dr. Pedram showed at trial, a task may comprise one or more threads. RDX-0004C.107 For example, the “Multi Task” drawing illustrates a scenario where two tasks are active, a first task comprising 5 threads and a second task comprising 3 threads. As explained above, the threads for these two tasks are supplied to the cache memory 1024 by the fetch unit 1023. When all 8 threads associated with the two tasks are received by the cache memory 1024, the corresponding tasks will also have been received by the identified engines (which include the cache memory 1024.) Similarly, when the 8 threads are sent to the processing blocks 106 by the thread processors 1026, the corresponding tasks will also have been sent to the processing blocks 106. Finally, when the processing of these threads is completed, the processing of the corresponding tasks will also have been completed.

AMD expert, Dr. Hart, incorrectly states that Stuttard’s channel and array controllers, and components of the thread manager do not receive tasks because the thread manager receives control signals, and the channel and array controllers receive instructions. Tr. (Hart) at 1280:14-1281:13. Stuttard’s teachings that “the thread manager 102 is connected to receive control signals from the EPU 8,” and “the control signals inform the thread manager as to when instructions are to be fetched and where the instructions are to be found,” and “the thread manager 102 is connected to provide these instructions to the array controller 104 and to the channel controller 108,” (Stuttard

[REDACTED]

RX-0416.0011 at 8:17-24) are consistent with and supports the fact the claimed “engines” receive tasks from the fetch unit 1023 and/or thread scheduler 1025. More precisely, the instruction streams or threads are received from local memory or the EPU by the fetch unit 1023 inside the thread manager. RX-0416.00014 at 11:12-14. Moreover, it is the thread manager 102 which, with the aid of the thread processors 1026, manages the issuance of instruction streams or threads to the processing blocks of Stuttard’s system to carry out. RX-0416.0013-0014 at 10:7-17 and 11:27-35. Finally, consistent with AMD’s infringement position, a task comprises multiple threads, and a thread comprises multiple instructions. When all instruction associated with a thread are executed, the thread will have been executed and when all threads associated with a task are executed, the task will have been executed.

Dr. Hart also incorrectly states that the Stuttard’s “engines” depend on each other and work in a serial fashion, that is, they are not independent of each other as allegedly required by the ‘381 patent. Tr. (Hart) at 1281:24-1282:8. But the independence of engines is not required by the claims of the ‘381 Patent.

Finally, Dr. Hart falsely states that the Stuttard’s fetch unit and the scheduler in its thread manager cannot be the claimed “scheduling module” because claim 15 of ‘381 requires that the scheduling module be associated with a second processing unit whereas the Stuttard’s fetch unit and the scheduler in its thread manager are on the first processing unit, 1282:16-21. This incorrect statement is driven by Dr. Hart’s equating “associated with” with being “part of,” which is wrong for the same reasons explained above as to Jiao. Tr. (Pedram) at 1020:15-1021:13.

(iii) Loading state data associated with each of the plurality of tasks.

Stuttard expressly states that the status block 1030 receives status information 1036 from each of the threads associated with a task, which is then transferred to the thread scheduler 1025

[REDACTED]

(“by the status block” – one of the claimed engines), which uses the status information to schedule the threads. Tr. (Pedram) at 899:19-901:9, 965:19-966:25; RX-0416.0026 at 23:13-28. Dr. Pedram explained that the status information is state data. Tr. (Pedram) at 967:18-23.

Stuttard therefore discloses a plurality of engines configured to receive, from a scheduling module associated with a second processing unit, a plurality of tasks and to load state data associated with each of the plurality of tasks.

(d) Claim 15[e]

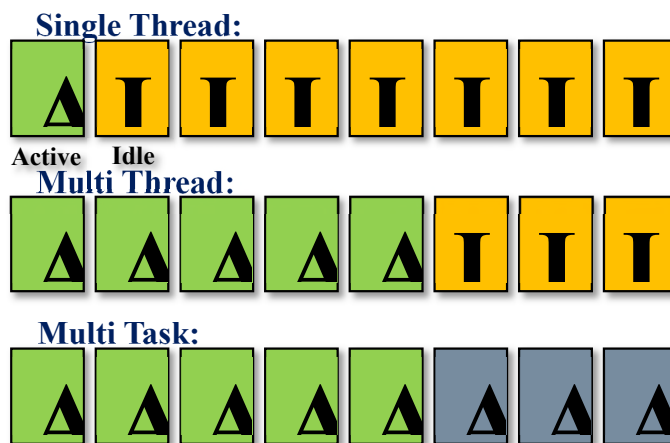
Dr. Pedram identified Stuttard’s plurality of processing blocks 106 as the claimed shader core. RDX-0004C.136; RX-0416.0131 at Figure 3; Tr. (Pedram) at 967:25-968:9. Stuttard discloses that the processing blocks receive tasks from the thread manager. RDX-0004C.138; RX-0416.0014 at 11:27-12:7; Tr. (Pedram) at 968:11-25. The thread processors, which is one of the identified engines, control the issuance of core instructions from the thread manager so that the system can maintain the simultaneous active program threads within the processing blocks. The processing blocks receive the corresponding instructions, which compose the corresponding tasks, from the engines. RDX-0004C.138; RX-0416.0014 at 11:27-12:7; Tr. (Pedram) at 968:11-25.

Dr. Pedram also testified that Stuttard’s shader core is configured to execute a first task from the plurality of tasks while executing a second task from the plurality of tasks based on its disclosure of multithreading and multitasking. RDX-0004C.140-143; RX-0416.0013 at 10:7-17; RX-0416.0014 at 11:16-25; Tr. (Pedram) at 969:2-971:14, 972:1-7, 973:12-974:20. Stuttard discloses that multithreading refers to the use of several threads to perform a single task, whereas multitasking refers to the use of several threads to perform multiple tasks simultaneously. *Id.* Stuttard expressly discloses using at least eight threads. RX-0416.0014 at 11:16-25 (“Of course, although the example described here has eight threads, any number of threads can be controlled in this manner.”); RDX-0004C.142. With at least eight threads, Stuttard can perform both



multithreading and multitasking. Tr. (Pedram) at 969:16-18.

Dr. Pedram explained that performing multithreading and multitasking is how Stuttard can achieve maximum performance, maximum resource utilization, and all of the benefits known to a person of ordinary skill in the art that come about from high speed processing and good utilization



of the resources. Tr. (Pedram) at 969:19-970:8; RDX-0004C.141-143. This is illustrated below.

RDX-0004C.141. The green and blue-gray boxes above represent active processing blocks and the yellow boxes represent idle processing blocks. Tr. (Pedram) at 970:12-13, 21-24. If you have a single thread and eight processing blocks, one of them could be active, whereas the remaining seven sit idle. *Id.* at 969:23-25. With multithreading, more processing blocks can be active because more threads can be run at the same time as shown in the second row above. *Id.* at 970:1-3. But, if you employ multitasking, each task creating its own set of threads, then you can achieve maximum performance with essentially no idle resources. *Id.* at 970:4-24. This is represented in the third row above, where the green boxes represent multiple threads of one task running on multiple processors in Stuttard, while the blue-gray boxes represent multiple threads of a second task running on multiple processor in Stuttard. *Id.* Again, Stuttard expressly discloses using both multithreading and multitasking. RDX-0004C.140-143; RX-0416.0013 at 10:7-17; RX-

[REDACTED]

0416.0014 at 11:16-25; Tr. (Pedram) at 969:2-971:14, 972:1-7, 973:12-974:20.

Dr. Hart also argues that Stuttard's processing blocks are configured in a SIMD fashion where every processing block receives a single instruction. As explained by Dr. Pedram, this is not true. More precisely, although each of the Stuttard's processing blocks is configured as a SIMD unit, different processing blocks (of which there can be 8 or more) are independent of each other and supported by its own dedicated thread processor 1026, meaning that each processing block can run its own instruction stream. Tr. (Pedram) at 995:9-996:5, 997:5-9, 1000:10-20.

Dr. Hart, incorrectly states that Stuttard's multithreading is not defined as running multiple threads concurrently and although Stuttard defines multitasking, Stuttard's system does not perform multitasking. He provides no support for these false statements that ignore explicit disclosures in Stuttard. As discussed earlier with respect to Stuttard and the '053 patent, Dr. Hart is cherry-picking what he chooses to consider in Stuttard and ignoring the express disclosure of Stuttard that multiple threads doing multiple tasks can be assigned to multiple processing blocks at the same time. Stuttard expressly states:

"At any particular moment in time, each thread may be assigned to any one of a number of tasks. For example, thread zero may be assigned for general system control, thread 1 assigned to execute 2D (two dimensional) activities, and threads 2 to 7 assigned to executing 3D activities (such as calculating vertices, primitives or rastering)."

RX-0416.0014 at 11:16-25; RDX-0004C.42. 3D activities "include both vertex processing as well as pixel processing." Tr. (Pedram) at 907:8-908:2. A POSITA would know that "calculating vertices requires vertex command thread processing and doing rasterization at least requires doing pixel command thread processing." *Id.* at 902:23-903:6. Dr. Pedram further testified that a POSITA would have understood that there could be combinations of vertex and pixel command threads, including two or more vertex command threads and two or more pixel command threads. *Id.* at 900:2-21; 905:8-908:2. This teaching in Stuttard is fatal to Dr. Hart's interpretation. Stuttard

[REDACTED]

also teaches that multiple threads can be simultaneously active and running on different processing blocks at the same time: “The thread processors 1026 control the issuance of core instructions from the thread manager so as to maintain processing of simultaneously active program threads, so that each [of] the processing blocks 106 can be active for as much time as possible.”

Faced with the irrefutable disclosure in Stuttard that multiple threads (which can correspond to multiple tasks) can run simultaneously on Stuttard’s processing blocks, Dr. Hart comes up with a new theory whereby all of Stuttard’s processing blocks receive and execute the same instruction at the same time, where the thread manager provides a stream (list) of thread interleaved instructions to all of the processing blocks to carry out. Tr. (Hart) at 1283:20-1284-22. In other words, Stuttard’s thread manager is designed to interleave instructions from different threads and not to execute multiples tasks concurrently. Tr. (Hart) at 1287:1-7. This incorrect and unsupported theory was not presented before the trial. Regardless, even under this theory and consistent with AMD’s infringement theory, the original threads (whose instructions are interleaved to form a stream of instructions) execute at the same time, meaning that the execution time of the said threads overlap in time.

As another attempt to refute Stuttard’s explicit teachings about multi-threading and multi-tasking, Dr. Hart, again for the first time, brings up an unfounded theory that the need for coherent (shared) memory access in graphics systems necessitates that all of Stuttard’s processing blocks execute the same instruction at a time. Tr. (Hart) at 1285:15-1286:6. First, there is no discussion of coherent memory access requirement in the ‘381 Patent, which claims multi-tasking itself. Second, cache coherency is not required in many cases, for example, if some memory locations are read only or are only accessed by a single thread, then no coherence is required. If caches are small and coherence is not always required but only at specific instructions of the program, then it

[REDACTED]

is possible to achieve correct behavior of the program using cache flushes before or after specific memory accesses.

Stuttard therefore discloses a shader core configured to receive the plurality of tasks from at least one of the plurality of engines and to execute a first task from the plurality of tasks while executing a second task from the plurality of tasks based on respective state data associated with each of the first and second tasks.

(2) Claim 16

As explained above, Stuttard discloses a shader core comprised of processing blocks 106. RDX-0004C.136, 146; RX-0416.0131 at Figure 3; Tr. (Pedram) at 967:25-968:9. Each processing block 106 constitutes a “processing element.” RDX-0004C.148; Tr. (Pedram) at 973:12-22. Dr. Pedram again explained that Stuttard discloses multiple tasks can be run at any point in time. Tr. (Pedram) at 973:12-22. One task could use a first plurality of processing blocks, while the remaining processing blocks perform the second task. *Id.* Again, Stuttard expressly discloses have different threads (each assigned to a different task) running at the same time. RDX-0004C.150; RX-0416.0014 at 11:16-25; Tr. (Pedram) at 974:3-20.

Dr. Hart offered no rebuttal for claim 16.

(3) Claim 17

Stuttard discloses that its cache memory can be replaced by a series of first-in/first-out buffers (FIFO), one per thread. RDX-0004C.151; RX-0416.0026 at 23:3-11; Tr. (Pedram) at 975:3-14. With different tasks running on the Stuttard system, each task would be using queues, i.e., the FIFO buffers, for storage. *Id.* FIFO is just another term for a queue, as known by a person of ordinary skill in the art. Tr. (Pedram) at 975:17-19.

Dr. Hart offered no rebuttal for claim 17.

[REDACTED]

[REDACTED]

(4) Claim 20

Stuttard expressly discloses selecting and assigning (i.e., receiving) threads associated with tasks based on “priorities relative to one another.” Tr. (Pedram) at 976:25-977:15; RDX-0004C.153; RX-0416.0014 at 11:16-25. Indeed, under the Court’s construction of the term “task” as “a unit of processing work or portion of an application to be executed” and based on the disclosure of the ’381 Patent, a POSITA would understand that a task would comprise one or more threads and that a high-priority task would give rise to high-priority threads. Conversely, once all high-priority threads associated with a task are completed based on their assigned priority levels, the task has received a high priority processing by the computing system.

2. Obviousness Under 35 U.S.C. § 103(a)

a. Jiao

(1) Claims 15, 16, 17 and 20

As discussed above, Jiao discloses the elements of the asserted claims. To the claims are not deemed anticipated, Jiao’s disclosure as expressed above (and taken as a whole) would have been understood by a POSITA to render the claims obvious. Tr. (Pedram) at 931:2-8; RDX-0004C.108, 120.

For example, a POSITA would have understood Jiao’s computational core 204 to be a [first] processing unit that is comprised of a pool of multiple execution units to meet the computing requirement imposed by shader tasks associated with various shader programs, including a vertex shader, geometry shader, and/or pixel shader, processing data for the graphics pipeline 208. Tr. (Pedram) at 929:3-930:2; RDX-0004C.87-90; *see, e.g.*, RX-0423.00003-6, 13-15 at 4:49-5:57, 6:1-48, 6:49-7:55, Figures 1-4. Also, a POSITA would have found it obvious that Jiao discloses a bus coupled to the memory and the GPU. *See* Tr. (Pedram) at 929:3-930:2; RDX-0004C.85-86. A POSITA would have understood Jiao to disclose buses to couple the various processing

[REDACTED]

components and memory. *See* Tr. (Pedram) at 929:3-930:2; RDX-0004C.85-86; *see, e.g.*, RX-0423.00015 at 8:60-64. Indeed, a POSITA would have found it obvious that Jiao discloses a bus coupled to system memory 124 and the GPU (hence, computational core 204) via PCI-E BIU 118, a bus coupled to local memory 106 and the GPU via a memory interface unit (MIU 110), and a plurality of 512-bit buses coupled to memory 306 and graphics processor 202 (hence, computational core 204), and 512-bit buses coupled to vertex/stream cache and computational core 204. *Id.* at 4:49-5:6, 7:35-55; *see also* Figure 1. Tr. (Pedram) at 932:9-24; RDX-0004C.85-86.

A POSTIA would have found it obvious that Jiao's vertex shader 304, geometry shader 306, and pixel shader 310 in graphics pipeline 208 of GPU 114 plus the "thread slots" could be the configured "engines," as described above. *See also* RX-0423.00016 at 9:15-24.

For all of these reasons and those provided with respect to the anticipation of Claims 15, 16, 17, and 20 with respect to Jiao above, and to the extent not expressly disclosed, a POSITA would have found Claims 15, 16, 17 and 20 obvious in light of Jiao. Tr. (Pedram) at 931:2-8; RDX-0004C.108, 120.

(2) Dependent Claim 19: The computing system of claim 15, wherein the first task comprises a graphics-processing task and the second task comprises a general-compute task.

As explained previously, Jiao discloses a graphic processor system, which may be configured as a computer system. *See* Tr. (Pedram) at 931:12-17; RDX-0004C.84. Further, the system comprises components such as arithmetic logic units and other logic and performs functionalities suitable for graphics-processing tasks and general-compute tasks. RX-0423.0014 at 6:1-48, 11:4-13. Further, Jiao teaches that the graphics processor 202 utilizes programmable vertex 304, geometry 306, and pixel shaders 310 and that each of these execution units 420 is identical in design and configurable for programmed multi-threaded operations. RX-0423.00014, 17 at 9:25-38. A POSITA would have understood and found it obvious that such programmable

[REDACTED]

execution units can be used to do both graphics-processing and general-compute tasks. Tr. (Pedram) at 949:12-950:12; RDX-0004C.117.

b. Stuttard

As discussed above, Stuttard discloses the elements of the asserted claims. To the extent any claim element for any of the asserted claims 15, 16, 17, and 20 is not deemed anticipated, Stuttard's disclosure as expressed above (and taken as a whole) would have been understood by a POSITA to render the claims obvious. Tr. (Pedram) at 977:20-24; RDX-0004C.145, 155.

(1) Claim 19

Stuttard discloses scheduling and executing general system control, 2D activities, and 3D activities, at the same time. Tr. (Pedram) at 976:4-15, 902:2-903:6, 907:8-908:2, 969:19-970:8; RDX-0004C.42, 141-143; RX-0416.0014 at 11:16-25. 3D activities would correspond to graphics processing tasks, and at least the general system control would be considered a general compute tasks.

Dr. Hart's sole rebuttal is that Stuttard describes a graphics processor and is focused on graphics processing tasks. Tr. (Hart) at 14-18. Again, Dr. Hart simply cherry picks what he wants to see in Stuttard and ignores other Stuttard disclosures. Dr. Hart offers no rebuttal to Dr. Pedram's testimony that threads assigned to "general system control" would be considered general compute tasks. Tr. (Pedram) at 976:4-15.

3. Lack of Secondary Considerations of Non-Obviousness

Dr. Hart's secondary considerations fail at the outset because the asserted claims do not require asynchronous task dispatch and because asynchronous task dispatch was already known in the art. *See* RDX-0004C.156; Tr. (Pedram) at 978:15-980:5.

In order for evidence of non-obviousness to be relevant, there must be a nexus between the

[REDACTED]

merits of the claimed invention and objective evidence of secondary considerations. *In re GPAC Inc.*, 57 F.3d 1573, 1580 (Fed. Cir. 1995); *Demaco Corp. v. F. Von Langsdorff Licensing Ltd.*, 851 F.2d 1387, 1392-93 (Fed. Cir. 1988). “Evidence of commercial success, or other secondary considerations,” is not relevant if it is “due to *an unclaimed feature of the device*,” or “*if the feature . . . was known in the prior art*.” See *Ormco Corp. v. Align Tech., Inc.*, 463 F.3d 1299, 1311-12 (Fed. Cir. 2006) (emphasis added). See also *supra*, cases cited at Section III.D.2.

The claims of the '381 patent do not recite asynchronous task dispatch. *Id.* The '381 patent provides no definition of asynchronous task dispatch and AMD never argued that asynchronous task dispatch was a limitation associated with any particular claim limitation during claim construction. Dr. Hart's testimony related to whether or not the claims require asynchronous task dispatch was to make a conclusory statement: “Yes, they do.” Tr. (Hart) at 1288:13-15. He offers no identification of which claim and claim element(s) is the basis of his conclusion. He offers no explanation or claim construction analysis of why or how any alleged claim element requires asynchronous task dispatch. Indeed, Dr. Hart's only testimony about what may or may not constitute asynchronous task dispatch is that each engine must be served by its own command buffer. Tr. (Hart) at 1262:20-1263:1. The claims do not recite or require any command buffers. See *supra* at 72-73.

In addition, and also fatal to Dr. Hart's position, is Dr. Pedram's testimony that asynchronous task dispatch was known in the art. Tr. (Pedram) at 979:16-980:5. Dr. Hart offered no rebuttal to Dr. Pedram's testimony that asynchronous task dispatch was known in the art. Because asynchronous task dispatch was known in the art, AMD's secondary considerations based on asynchronous task dispatch fail. Moreover, AMD has not and cannot provide any analysis or proof that the AMD products that were the subject of the alleged objective indicia of non-

[REDACTED]

obviousness practiced any asserted claim of the '381 patent, as its purported evidence of secondary considerations is not tied to AMD's DI Products or any other products as to which any AMD expert opined practiced the claims of the '381 patent. Dr. Hart admitted that he did not perform any analysis of the products to determine if they practiced any of the asserted claims. Tr. (Hart) at 1308:19-1309:3. Dr. Hart admitted that he merely relied on the conclusion of Dr. Reinman but did not review or perform any actual analysis of the claims to test its veracity or form the bases of his opinions. *Id.* at 1310:5-12. Moreover, Dr. Hart testified that he relied on Dr. Reinman's analysis of the domestic industry products, Tr. (Hart) at 1255:24-1256:8, but Dr. Reinman did not offer any expert opinion or claim-by-claim analysis that the products to which Complainants have tied their evidence of secondary considerations practiced any of the claims of the '381 patent.

Finally, any evidence of commercial success regarding the AMD DI Products is not relevant to secondary considerations of non-obviousness for the additional reason that those products are not "co-extensive" with the alleged invention of the '053 patent. Those products have many features other than the alleged "asynchronous task dispatch" in the GPU and AMD has not offered any evidence tying commercial success to the "asynchronous task dispatch" feature of those products. A patentee is not entitled to a presumption of nexus between asserted evidence of secondary considerations and a patent claim unless the patentee has shown that "the product 'is the invention disclosed and claimed,'" that is, where the product "embodies the claimed features, and is coextensive with them." *Teva Pharms. Int'l GmbH v. Eli Lilly & Co.*, 8 F.4th 1349, 1360 (Fed. Cir. 2021) (citations omitted); *Fox Factory, Inc. v. SRAM, LLC*, 944 F.3d 1366, 1373 (Fed. Cir. 2019). Conversely, if the patented invention—as here—is only one component of a commercially successful product, "the patentee is not entitled to a presumption of nexus." *Teva Pharms.*, 8 F.4th at 1360 (citing *Fox Factory*, 944 F.3d at 1373). There is no colorable argument

[REDACTED]

that any AMD DI Product has no features other than “asynchronous task dispatch” within the shader core, and AMD has not otherwise shown that any commercial success of its products is attributable to the “asynchronous task dispatch” feature.

E. Section 101

The Asserted Claims of the '381 Patent are invalid under 35 U.S.C. § 101 because they are directed to the abstract idea of task scheduling and impart no particular technological improvement to a computer or other technology. Tr. (Pedram) at 922:8-927:15; RDX-0004C.70, 77. As construed by the ALJ, the claims are completely untethered to any purported improvement over the prior art as recited in the specification and the foundation of the claims is nothing more than generic computer hardware (e.g., a memory, a bus, and processing units). There can be no dispute that the idea of scheduling tasks is abstract, much less when that scheduling has no particular constraints recited in the claims. Thus, the only issue for the ALJ to decide is whether Step 2 of *Alice* – do the claims transform the abstract idea into something patent-eligible – is met. But with nothing in the claims aside from generic hardware that processed the generically scheduled tasks, the resounding answer to Step 2 is a failure under *Alice*.

The *Alice* framework for determining if patent claims are valid under 35 U.S.C. § 101 requires a two-step analysis: (1) whether the patent claims are directed toward a patent-ineligible concept and (2) whether the claim’s elements, considered both individually and as an ordered combination, transform the nature of the claim into a patent-eligible application. *Alice Corp. Pty. Ltd. v. CLS Bank Int’l*, 573 U.S. 208, 217-18 (2014).

Scheduling tasks has been done by humans since humans first existed and is an abstract idea. The Asserted Claims of the '381 Patent merely recite the receipt of several “tasks” and then sending two or more of these tasks to a computing unit to be executed.. Tr. (Pedram) at 927:2-15;

[REDACTED]

RDX-0004C.70, 77; JX-0003.0015 at 14:1-38 Claim 15 (“engines configured to *receive*....a plurality of tasks...; and a shader core configured to *receive* the plurality of tasks”), Claim 16 (processing elements configured to *execute* the first task.....processing elements *execute* a second task”), Claim 17 (“...a first queue to *receive* the first task and a second queue configured to *receive* the second task”), Claim 19 (“first task comprises a graphics-processing task and the second task comprises a general-compute task”), Claim 20 (“shader core is configured to *receive* the plurality of tasks”) (emphasis added). Put simply, the ’381 Patent is describing receiving and executing two pieces of information. *See id.* That is inherently abstract. *Braemar Mfg., LLC v. ScottCare Corp.*, 816 F. App’x 465, 468 (Fed. Cir. 2020) (holding claims patent ineligible as they were directed to “collecting, classifying, or otherwise filtering data into different groups based on identifying characteristics”) (citation omitted); *see also Interval Licensing LLC v. AOL, Inc.*, 896 F.3d 1335, 1338 (Fed. Cir. 2018) (finding that processes collecting data and recognizing data within the set, or other data management functions, constitute abstract ideas) (citation omitted).

The specification of the ’381 Patent confirms the abstract nature of the claims. The specification identifies task scheduling as conventional and long- employed by processors. *Tr. (Pedram)* at 927:2-931:1; RDX-0004C.73-76; RX-0149.00010 at 1:15-2:59. In describing the history of the task scheduling, the ’381 Patent described the use of processors in the prior art including CPUs and GPUs, which may be used separately or together. *See JX-0003.0009* at 1:23-32. Prior to the ’381 Patent, GPUs could already execute different threads or shader programs, “typically associated with a set of state data” that is locally stored in the GPU. *Id.* at 1:33-57. The GPUs had “an array of processing elements, called a shader core,” which could be organized into multiple SIMD devices capable of executing different threads, shader programs, or “tasks” at the same time, thereby performing operations in parallel. *Id.* at 1:58-2:7. Additionally, at that time,

[REDACTED]

the CPUs could hand off “tasks” to the GPU, and “an operating-system (OS) scheduler [that] stores the tasks,” which might be processed by the GPU in the order they were received or out of order. *Id.* at 2:4-19. GPUs in the prior art were also able to recognize priorities of tasks, and could “perform the more-important (*e.g.*, low-latency) task before the first task has finished completing in the shader core.” *Id.* at 2:20-37. The fact that tasks would be scheduled based on their associated “state data” was routine and conventional – any tasks executed in a processor must have state associated with them. *Id.* at 1:53-57, 2:20-37. Nothing in the ’381 patent changes the fundamental fact that scheduling tasks is just determining which task to pass on followed by another. Few claimed inventions are as abstract as that, which is why the Federal Circuit routinely finds such “scheduling” claims abstract. *See, e.g., PersonalWeb Techs. LLC v. Google LLC*, 8 F.4th 1310, 1317 (Fed. Cir. 2021) (reiterating that processes which “can be performed in the human mind, or by a human using a pen and paper, ... [has been] repeatedly found unpatentable”) (citation omitted); *Solutran, Inc. v. Elavon, Inc.*, 931 F.3d 1161, 1166 (Fed. Cir. 2019) (“claims performed in a particular order or sequence are directed to abstract ideas”).

The Asserted Claims fare no better under Step Two of *Alice*’s framework, because the claim’s elements, considered both individually and as an ordered combination, do not transform the nature of the claim into a patent-eligible application. *Alice*, 573 U.S. at 223-26. Tr. (Pedram) at 923:1-927:15; RDX-0004C.72. Beyond task scheduling, the Asserted Claims are simply generic computer components doing what generic computer components do. Tr. (Pedram) at 923:1-924:13; RDX-0004C.72; JX-0003.0015 at 14:1-38 (“[a] computing system comprising: a memory; a first processing unit; and a bus”). AMD does not argue that the recited components were not well-known components in the art. *See* Tr. (Hart) at 1267:5-1277:16; *see also* CDX-0008C.68-91. Further, the components are not arranged in a novel or unconventional manner (*e.g.*, a GPU

[REDACTED]

receiving tasks from a CPU and using an array of processing elements in the GPU to execute tasks in parallel), nor is there technological improvement resulting from their particular use or combination. Tr. (Pedram) at 923:1-927:15; RDX-0004C.77; *see also Mayo Collaborative Servs. v. Prometheus Lab 'ys, Inc.*, 566 U.S. 66, 79-80 (2012); *cf. BASCOM Glob. Internet Servs. v. AT&T Mobility LLC*, 827 F.3d 1341, 1350 (Fed. Cir. 2016).

As construed, the claims recite generic hardware components and generic/functional-based software components that do *not* require any specific novel hardware or software, simply what can perform the recited functions in the claim. *See* Order No. 31 at 38-41; Tr. (Pedram) at 922:1-6, 923:2-924:13; *see also* Tr. (Hart) at 1264:13-19. Such functional claiming cannot impart any technological improvement directed to computer hardware. Even the shader core can be a first and second “plurality of processing elements” (such as SIMD units) as recited in dependent claim 16, capable of performing parallel operations, which the specification discloses as prior art. *See* Order No. 31 at 36. All of the components are claimed only according to their historical use, which is nothing more than scheduling tasks to be processed at the same time. Tr. (Pedram) at 923:1-927:15; RDX-0004C.77.

As an ordered combination, the Asserted Claims are not directed to any inventive concept. Tr. (Pedram) at 923:1-925:21; RDX-0004C.77. Each of the functional aspects of the generic hardware components were well-known and clearly understood at the time of the invention. Tr. (Pedram) at 923:1-927:21; RDX-0004C.69-77. And for each step, the inputs, outputs, and logic/hardware were well-known and commonly used architectures, which is commonly found by the Federal Circuit to fail Step 2 of *Alice*. *See id.*

For example, in *Content Extraction & Transmission LLC v. Wells Fargo Bank, National Ass'n*, the Federal Circuit held patent ineligible claims that were directed to “the abstract idea of

[REDACTED]

1) collecting data, 2) recognizing certain data within the collected data set, and 3) storing that recognized data in a memory.” 776 F.3d 1343, 1347 (Fed. Cir. 2014). “The concept of data collection, recognition, and storage is undisputedly well-known” and “humans have always performed these functions.” *Id.* The patent claims also failed to satisfy Step Two of *Alice*, because there was no “inventive concept” in using “existing scanning and processing technology ... to perform well-understood, routine, and conventional activities commonly used in industry.” *Id.* at 1348; *see also In re TLI Commc’ns LLC Pat. Litig.*, 823 F.3d 607, 610-11 (Fed. Cir. 2016) (finding claims patent ineligible that were “drawn to the concept of classifying an image and storing the image based on its classification” and that did not add any inventive concept by using generic computer components such as a “telephone unit,” a “server,” an “image analysis unit” and a “control unit” to take, transmit, and organize digital images); *see also Certain Smart Thermostat Sys., Smart HVAC Sys., Smart HVAC Control Sys., & Components Thereof*, Inv. No. 337-TA-1258, Initial Determination at 67-76, 113-17 (Apr. 4, 2022) (finding ineligible claims where “the only hardware needed to accomplish the [abstract] idea comprise generic temperature sensors, digital memory, and a processor” and the claim did not represent an improvement to the technology); *Certain Wearable Elec. Devices with ECG Functionality & Components Thereof*, Inv. No. 337-TA-1266, Initial Determination at 141-46 (June 27, 2022) (finding ineligible claims directed to data analysis algorithms performed by a processor, using instructions saved in memory, which was performed with generic hardware components including various “sensors,” “mobile computing device,” “processor,” and “computer readable medium,” which did not constitute an improvement to the underlying computing elements).

AMD’s expert testimony regarding patent eligibility spans an entire 23 lines of the Hearing transcript. With respect to Step 2, Dr. Hart was asked a single question regarding whether the

[REDACTED]

claims of the '381 patent were directed to a specific improvement in computer technology, but all Dr. Hart could muster was a generic reference to a GPU and the “shader core being able to slip in tasks into those lost cycles in order to improve its performance.” Tr. (Hart) at 1292:21-1293:3. But admittedly, context switching is not present, recited, or required by the claims of the '381 patent. *Id.* at 1234:17-21. Even more notable, Dr. Hart provided no rebuttal to the seven pages of testimony that Dr. Pedram provided regarding what was conventional, well-known, and failed to impart any specific improvement to computer technology. *See, e.g.*, Tr. (Pedram) at 921-927. Therefore, Dr. Hart’s limited testimony fails to even attempt to rebut Dr. Pedram, and the single Q&A that Dr. Hart provided is irrelevant.

And Dr. Hart’s testimony is further irrelevant because in the '381 Patent, the abstract idea of scheduling tasks to be performed at the same time is capable under pre-existing, conventional computer technology, such as a CPU handing-off a plurality of tasks to a GPU and the GPU using an array of processing elements to execute those tasks in parallel. Tr. (Pedram) at 923:1-927:21; RDX-0004C.69-77. And while dependent claims 18-20 “group” the plurality of tasks into two buckets, such as more important and less important, so that a task from each bucket might be performed at the same time, this again is an abstract idea (*e.g.*, “recognizing certain data within the collected data set”) and the claims do nothing more than process the two types of data in parallel using conventional and well-known computer components. Tr. (Pedram) at 923:1-927:21; RDX-0004C.77. No context switching or multithreaded processing is required by the claims.

Accordingly, the claim elements follow the natural progression of a GPU receiving a task from a CPU and scheduling a plurality of tasks to be executed. The steps, inputs, outputs, and logic/hardware needed to carry out each step, were routine and conventional, and were commonly used in all architectures that existed at the time. Tr. (Pedram) at 923:1-927:15; RDX-0004C.77.

[REDACTED]

Nothing *in the claims* imparts any inventive concept beyond the abstract idea of scheduling tasks, nor does anything *in the claims* directly improve the pre-existing computing system. Even to the extent that Claims 18-20 require the first task and second task of different nature or priority, processing tasks in parallel in a shader core and recognizing priority was all well-known in the prior art. Tr. (Pedram) at 923:1-927:15; RDX-0004C.74-76. Claims 16 and 17 similarly are directed to the same abstract idea and do not improve the functioning of a computer or the underlying technology, which only recite different processing elements in the shader core executing the first and second tasks and having queues to receive the tasks. Tr. (Pedram) at 923:1-927:15; RDX-0004C.74-76. Thus, the claims also fail Step Two of *Alice*. Tr. (Pedram) at 927:2-15; RDX-0004C.77.

The Asserted Claims of the '381 Patent should be found invalid under 35 U.S.C. § 101.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

Johannes Hsu
Jana Zaidan
ORRICK, HERRINGTON & SUTCLIFFE LLP
2050 Main Street, Suite 1100
Irvine, CA 92614
Telephone: (949) 567-6700
Facsimile: (949) 567-6710
Email: Realtek-1318-Service@orrick.com

Jeffrey L. Johnson
ORRICK, HERRINGTON & SUTCLIFFE LLP
609 Main Street, 40th Floor
Houston, TX 77002-3106
Telephone: (713) 658-6400
Facsimile: (713) 658-6401
Email: Realtek-1318-Service@orrick.com

Counsel for Respondent Realtek Semiconductor Corp.

/s/ John P. Schnurer

John P. Schnurer
Kevin J. Patariu
Yun Louise Lu
Vinay P. Sathe
Kyle R. Canavera
Hayden C. Householder
PERKINS COIE LLP
11452 El Camino Real, Suite 300
San Diego, CA 92130-2080
Phone: (858) 720-5700
Email:
PerkinsServiceTCLAMDITC@perkinscoie.com

James B. Coughlan
Perkins Coie LLP
700 Thirteenth Street N.W., Suite 800
Washington, DC 20005-3960
Phone: (202) 654-6200
Email:
PerkinsServiceTCLAMDITC@perkinscoie.com

*Counsel for Respondents
TCL Industries Holdings Co., Ltd., TCL
Industries Holdings (H.K.) Limited, TCL
Electronics Holdings Limited, TCL Technology*



Group Corporation, TTE Corporation, TCL Holdings (BVI) Limited, TCL King Electrical Appliances (Huizhou) Co., Ltd., Shenzhen TCL New Technology Co., Ltd., TCL MOKA International Limited, TCL Smart Device (Vietnam) Co., Ltd., Manufacturas Avanzadas SA de CV, TCL Electronics Mexico, S de RL de CV, and TCL Overseas Marketing Ltd.

CERTIFICATE OF SERVICE

I hereby certify that the foregoing was served on April 7, 2023 upon the following parties as indicated:

The Honorable Katherine Hiner Acting Secretary to the Commission U.S. INTERNATIONAL TRADE COMMISSION 500 E Street, S.W., Room 112 Washington, DC 20436	☐ Via First Class Mail ☐ Via Overnight Courier (2 copies) ☐ Via Hand Delivery (2 copies) ☒ Via EDIS Electronic Filing
Honorable Cameron R. Elliot Administrative Law Judge U.S. INTERNATIONAL TRADE COMMISSION 500 E Street S.W., Room 317 Washington, DC 20436 Email: michael.turner@usitc.gov	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail
<i>Counsel for Complainants Advanced Micro Devices, Inc. and ATI Technologies ULC</i> Adam S. Rizk MINTZ, LEVIN, COHN, FERRIS, GLOSKY, AND POPEO PC One Financial Center Boston, MA 02111 Email: AMD-ITC@mintz.com Steven J. Torres Ryan A. Rucki TORRES, SCAMMON, HINCKS & DAY, LLP 119 High Street Boston, MA 02110 Email: storres@tshdlegal.com rrucki@tshdlegal.com	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail

<p><i>Counsel for Respondents TCL Industries Holdings Co., Ltd.; TCL Industries Holdings (H.K.) Limited; TCL Electronics Holdings Limited f/k/a TCL Multimedia Technology Holdings, Ltd.; TCL Technology Group Corporation; TTE Corporation; TTE Technology, Inc.; TCL Holdings (BVI) Limited; TCL King Electrical Appliances (Huizhou) Co., Ltd.; Shenzhen TCL New Technologies Co., Ltd.; TCL MOKA International Limited; TCL Smart Device (Vietnam) Co., Ltd.; Manufacturas Avanzadas SA de CV; TCL Electronics Mexico, S de RL de CV; and TCL Overseas Marketing Ltd.</i></p> <p>John P. Schnurer PERKINS COIE LLP 700 Thirteenth Street N.W., Suite 800 Washington, DC 20005-3960 Email: PerkinsServiceTCLAMDITC@perkinscoie.com Individual Email: JSchnurer@perkinscoie.com</p>	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail
<p><i>Co-Counsel for Respondent Realtek Semiconductor Corp.</i></p> <p>Theodore J. Angelis K&L GATES LLP 925 Fourth Avenue, Suite 2900 Seattle, WA 98104-1158 Email: realtek1318@klgates.com</p>	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail

/s/ Karen Johnson
Karen Johnson

CERTIFICATE OF SERVICE

I hereby certify that the foregoing was served on April 10, 2023 upon the following parties as indicated:

The Honorable Katherine Hiner Acting Secretary to the Commission U.S. INTERNATIONAL TRADE COMMISSION 500 E Street, S.W., Room 112 Washington, DC 20436	☐ Via First Class Mail ☐ Via Overnight Courier (2 copies) ☐ Via Hand Delivery (2 copies) ☒ Via EDIS Electronic Filing
Honorable Cameron R. Elliot Administrative Law Judge U.S. INTERNATIONAL TRADE COMMISSION 500 E Street S.W., Room 317 Washington, DC 20436 Email: michael.turner@usitc.gov	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail
<i>Counsel for Complainants Advanced Micro Devices, Inc. and ATI Technologies ULC</i> Adam S. Rizk MINTZ, LEVIN, COHN, FERRIS, GLOSKY, AND POPEO PC One Financial Center Boston, MA 02111 Email: AMD-ITC@mintz.com Steven J. Torres Ryan A. Rucki TORRES, SCAMMON, HINCKS & DAY, LLP 119 High Street Boston, MA 02110 Email: storres@tshdlegal.com rrucki@tshdlegal.com	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail

<p><i>Counsel for Respondents TCL Industries Holdings Co., Ltd.; TCL Industries Holdings (H.K.) Limited; TCL Electronics Holdings Limited f/k/a TCL Multimedia Technology Holdings, Ltd.; TCL Technology Group Corporation; TTE Corporation; TTE Technology, Inc.; TCL Holdings (BVI) Limited; TCL King Electrical Appliances (Huizhou) Co., Ltd.; Shenzhen TCL New Technologies Co., Ltd.; TCL MOKA International Limited; TCL Smart Device (Vietnam) Co., Ltd.; Manufacturas Avanzadas SA de CV; TCL Electronics Mexico, S de RL de CV; and TCL Overseas Marketing Ltd.</i></p> <p>John P. Schnurer PERKINS COIE LLP 700 Thirteenth Street N.W., Suite 800 Washington, DC 20005-3960 Email: PerkinsServiceTCLAMDITC@perkinscoie.com Individual Email: JSchnurer@perkinscoie.com</p>	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail
<p><i>Co-Counsel for Respondent Realtek Semiconductor Corp.</i></p> <p>Theodore J. Angelis K&L GATES LLP 925 Fourth Avenue, Suite 2900 Seattle, WA 98104-1158 Email: realtek1318@klgates.com</p>	☐ Via First Class Mail ☐ Via Express Mail ☐ Via Hand Delivery ☒ Via Electronic Mail

/s/ Karen Johnson
Karen Johnson