UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

REALTEK SEMICONDUCTOR CORP.,

Petitioner,

v.

ADVANCED MICRO DEVICES, INC.

Patent Owner.

Case No. IPR2023-00687

U.S. Patent No. 8,854,381

PATENT OWNER'S CONTINGENT MOTION TO AMEND

TABLE OF CONTENTS

Page

I. INTRODUCTION 1
II. REQUESTED CONTINGENT AMENDMENT 2
III. A REASONABLE NUMBER OF SUBSTITUTE CLAIMS ARE PROPOSED 2
IV. THE PROPOSED AMENDMENTS RESPOND TO A GROUND OF UNPATENTABILITY INVOLVED IN THIS TRIAL 3
V. THE PROPOSED AMENDMENTS DO NOT SEEK TO ENLARGE THE SCOPE OF THE CHALLENGED CLAIMS 4
VI. THE PROPOSED AMENDMENTS DO NOT INTRODUCE NEW SUBJECT MATTER 4
A. Substitute Claim 28 Is Supported by the Originally-Filed Application 4
1. 15. A computing system comprising: 5
2. a memory; 5
3. a first processing unit; and 5
4. a bus coupled to the memory and the first processing unit, wherein the first processing unit comprises: 5
5. 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 5
6. 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. 6
7. 28. The computing system of claim 15, wherein the shader core comprises: multiple processing elements; 7
8. 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 7

9. 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,.....7

10. wherein each of the plurality of tasks is decomposable into two or more related threads,.....8

11. and wherein at least one processing element is configured to receive tasks from each engine. 8

B. Substitute Claim 29 Is Supported by the Originally-Filed Application 10

1. 29. 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,.....10

2. wherein the plurality of engines are arranged in parallel with one another,.....10

3. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine,.....11

4. wherein each of the plurality of tasks is decomposable into two or more related threads, and.....11

5. wherein each engine is configured to load state data associated with at least one of the plurality of tasks.....11

VII. The proposed substitute claims are not unpatentable 12

A. Claim 28 13

1. Wilt.....13

a. wherein each of the plurality of tasks is decomposable into two or more related threads 13

b. wherein at least one processing element is configured to receive tasks from each engine 16

2. Jiao 17

a. wherein each of the plurality of tasks is decomposable into two or more related threads 17

b. wherein at least one processing element is configured to receive tasks from each engine 18

3. Stuttard 19

a. wherein each of the plurality of tasks is decomposable into two or more related threads 19

b. wherein at least one processing element is configured to receive tasks from each engine 20

B. Claim 29 20

1. Wilt 20

a. wherein the plurality of engines are arranged in parallel with one another 20

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine 21

c. wherein each of the plurality of tasks is decomposable into two or more related threads 21

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks 21

2. Jiao 22

a. wherein the plurality of engines are arranged in parallel with one another 22

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine 22

c. wherein each of the plurality of tasks is decomposable into two or more related threads 23

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks 23

3. Stuttard 23

a. wherein the plurality of engines are arranged in parallel with one another 23

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine 24

c. wherein each of the plurality of tasks is decomposable into two or more related threads 24

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks 24

VIII. Conclusion 25

TABLE OF AUTHORITIES

Page(s)

Cases

Aspex Eyewear, Inc. v. Marchon Eyewear, Inc.,
672 F.3d 1335 (Fed. Cir. 2012) 15

Lectrosonics, Inc. v. Zaxcom, Inc.,
IPR2018-01129, Paper 15 (PTAB Feb. 25, 2019).....3

Statutes

35 U.S.C. § 103 1

35 U.S.C. § 103(a) 1

35 U.S.C. § 316(d) 1

35 U.S.C. § 316(d)(3).....4

Other Authorities

37 C.F.R. § 42.121 1

37 C.F.R. § 42.121(a)(2)(i) 3

37 C.F.R. § 42.121(a)(2)(ii) 4

37 C.F.R. § 42.121(a)(3) 2, 3

84 Fed. Reg. 9,497 2

UPDATED TABLE OF EXHIBITS

Exhibit	Description
2007	Deposition Transcript of Massoud Pedram, Ph.D. (Jan. 5 and 8, 2023)
2008	Declaration of John Hart, Ph.D., Regarding Patent Owner's Contingent Motion to Amend (Jan. 17, 2024)

I. INTRODUCTION

Pursuant to Paper 11, Patent Owner ATI Technologies ULC ("Patent Owner") submits this contingent Motion to Amend ("Motion") to request amendment of challenged claims 16 and 17 of U.S. Patent No. 8,854,381 ("the '381 Patent") to provide further clarification of the invention set forth therein, in the event the Board determines that those claims are unpatentable. Petitioner Realtek Semiconductor Corp. ("Petitioner") has challenged the patentability of claims 15-20 as obvious under 35 U.S.C. § 103, as follows:

- Ground 1: Claims 15-20 are challenged under 35 U.S.C. § 103(a) over U.S. Patent No. 8,151,095 to Wilt ("Wilt") in view of the knowledge of a person of ordinary skill in the art.
- Ground 2: Claims 15-20 are challenged under 35 U.S.C. § 103(a) over U.S. Patent No. 8,319,774 to Jiao *et al.* ("Jiao") in view of the knowledge of a person of ordinary skill in the art.
- Ground 3: Claims 15-20 are challenged under 35 U.S.C. § 103(a) over PCT Pub. No. WO00/62182A2 to Stuttard *et al.* ("Stuttard") in view of the knowledge of a person of ordinary skill in the art.

Pet. at 21.

Patent Owner submits that this motion satisfies the statutory and regulatory requirements for a motion to amend the claims under 35 U.S.C. § 316(d) and 37

C.F.R. § 42.121 because the proposed amendments (i) do not enlarge the scope of the claims or introduce new subject matter, (ii) are responsive to at least one ground of unpatentability instituted in this proceeding, (iii) are supported by the disclosure of the originally-filed application, and (iv) are patentable over the closest known prior art.

Patent Owner requests preliminary guidance from the Board regarding this motion, pursuant to the MTA Pilot Program (84 Fed. Reg. 9,497). This proceeding was instituted on October 25, 2023, *see* Paper 7, and thus qualifies for the MTA Pilot Program effective on March 15, 2019. Patent Owner reserves its right to file a revised motion to amend subject to the Board's preliminary guidance.

II. REQUESTED CONTINGENT AMENDMENT

If during this IPR proceeding, claim 16 is found unpatentable, Patent Owner requests that it be replaced with proposed substitute claim 28, which is recited in Appendix A. If claim 17 is found unpatentable, Patent Owner requests that it be replaced with proposed substitute claim 29, also recited in Appendix A. The differences between the current claims and the proposed substitute claims, which comprise only additional language, is noted with underline formatting.

III. A REASONABLE NUMBER OF SUBSTITUTE CLAIMS ARE PROPOSED

Under 37 C.F.R. § 42.121(a)(3), a motion to amend may propose a reasonable number of substitute claims for each challenged claim. Generally, it is presumed

"that only one substitute claim would be needed to replace each challenged claim," but that presumption may be rebutted upon a showing of need. 37 C.F.R. § 42.121(a)(3); *Lectrosonics, Inc. v. Zaxcom, Inc.*, IPR2018-01129, Paper 15 at 4-5 (PTAB Feb. 25, 2019). Patent Owner submits that this motion complies with this requirement because only a total of two substitute claims are proposed for the six challenged claims.

IV. THE PROPOSED AMENDMENTS RESPOND TO A GROUND OF UNPATENTABILITY INVOLVED IN THIS TRIAL

Under 37 C.F.R. § 42.121(a)(2)(i), the amendment(s) to the claims must be responsive to at least one ground of unpatentability involved in the trial. In the Institution Decision, the Board determined that Petitioner had established a reasonable likelihood that each of the challenged claims, claims 15-20, is invalid on the basis of each of the three asserted grounds, on the basis of Wilt, Jiao, and Stuttard. Paper 7. As explained below, each of the proposed substitute claims adds narrowing limitations that clarify that the invention is patentable over the closest known prior art, which is Wilt, Jiao, and Stuttard. Concurrently with this motion, Patent Owner is filing an information disclosure statement in the prosecution file of the '381 patent. The references cited therein, which are also listed in Appendix B here, were alleged by Petitioner to be relevant to the '381 patent and other patents of Patent Owner during the ITC Investigation between the parties, and Patent Owner is disclosing those references out of an abundance of caution, but Patent Owner does

not concede any are prior art. Copies of the non-patent-literature references that were obtainable by Patent Owner are being filed with the IDS. Patent Owner is not aware that any of the cited references is closer prior art than Wilt, Jiao, and Stuttard to the inventions of the '381 patent.

V. THE PROPOSED AMENDMENTS DO NOT SEEK TO ENLARGE THE SCOPE OF THE CHALLENGED CLAIMS

As explained above in Section II, the proposed substitute claims are strictly narrower versions of existing, challenged claims. No subject matter is proposed to be removed from the claims. Thus, no aspect of the proposed substitute claims expands the scope of any challenged claim.

VI. THE PROPOSED AMENDMENTS DO NOT INTRODUCE NEW SUBJECT MATTER

Under 35 U.S.C. § 316(d)(3) and 37 C.F.R. § 42.121(a)(2)(ii), a motion to amend may not introduce new subject matter. Patent Owner submits this requirement is satisfied because each of the proposed amendments, and each substitute claim in its entirety, has support in the application that led to the '381 Patent, as filed, as well as in the specification of the '381 Patent as issued, which is the same as the originally-filed application.

A. Substitute Claim 28 Is Supported by the Originally-Filed Application

The entirety of substitute claim 28 is supported by the originally-filed application. Each limitation is separately addressed below under different headings.

1. 15. A computing system comprising:

Claim 15, from which claim 28 depends, recites "[a] computing system," which is disclosed at least in Paragraphs 2, 3, 6, 13-17, 33-42, and Figs. 1-5, *e.g.*, "computing system 100," of the originally-filed application, among other places. Ex. 1002 at 6-35; Ex. 2008 at ¶ 25.

2. a memory;

This limitation is disclosed at least in Paragraphs 15, 37-38, 56, Fig. 1 (items 104, 106), and Fig. 2 (system memory) of the originally-filed application, among other places. Ex. 1002 at 6-35; Ex. 2008 at ¶ 26.

3. a first processing unit; and

This limitation is disclosed at least in Paragraphs 13, 15, 33-45 (GPU 110) Fig. 1 (GPU 110), and Fig. 2 of the originally-filed application, among other places. Ex. 1002 at 6-35; Ex. 2008 at ¶ 27.

4. a bus coupled to the memory and the first processing unit, wherein the first processing unit comprises:

This limitation is disclosed at least in Paragraphs 15, 37-38, and Fig. 1 (bus 114) of the originally-filed application, among other places. Ex. 1002 at 6-35; Ex. 2008 at ¶ 28.

5. 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

This limitation is disclosed at least in the Abstract, Paragraphs 13, 15, 16, 27-

31 (virtual engines), 41, 44, 47, 58-64, Fig. 2 (engines 202), Fig. 3A (engines 312 are a "plurality of engines"; OS Scheduler 310 is a "scheduling module"; application 302 runs on CPU 102, which is a "second processing unit"), Fig. 3B (engines 332 are a "plurality of engines"; OS Scheduler 310 is a "scheduling module" and runs on CPU 102, which is a "second processing unit"), and Fig. 4 (engines 432 are a "plurality of engines"; OS Scheduler 310 is a "scheduling module" and runs on CPU 102, which is a "second processing unit") of the originally-filed application, among other places. Ex. 1002 at 6-35; Ex. 2008 at ¶ 29.

6. 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.

This limitation is disclosed at least in Paragraphs 13, 15, 27-32, 35, 41-42, 50-54, 58-64, and Figs. 2-4 of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, paragraphs 27, 30, 41, 50-54, and 58-64 describe the ability of the shader core 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. Ex. 2008 at ¶ 30.

7. **28. The computing system of claim 15, wherein the shader core comprises: multiple processing elements;**

This limitation is disclosed at least in Paragraphs 30, 50-54, and Fig. 2 (depicting processing elements 220 of shader core 214) of the originally-filed application, among other places. Ex. 1002 at 6-35.

8. **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**

This limitation is disclosed at least in Paragraphs 30, 50-54, and Fig. 2 (depicting processing elements 220) of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, Paragraph 30 explains that "resources of the shader core are partitioned in space and/or time." The specification further states that "[t]o achieve spatial partitioning, for example, a first (e.g., low-latency) task from a first virtual engine is issued to a first subset of processing elements (SIMDs) of the shader core, and a second (e.g., standard graphics) task from a second virtual engine is issued to a second subset of processing elements (SIMDs) of the shader core." Paragraphs 8 and 16, among others, explain that such execution is based on state data associated with each of the plurality of tasks. Ex. 2008 at ¶ 32.

9. **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,**

This limitation is disclosed at least in 30, 50-54, and Fig. 2 (depicting processing elements 220) of the originally-filed application, among other places. Ex.

1002 at 6-35. The citations and explanation in the previous element apply here too. Ex. 2008 at ¶ 33.

10. wherein each of the plurality of tasks is decomposable into two or more related threads,

This limitation is disclosed at least in Paragraphs 5-8 of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, paragraph 5 states that "[t]o perform a single graphics-processing task, a GPU may execute a plurality of different threads (sequence of instructions)." Paragraph 6 similarly refers to "the various threads (e.g., shader programs) of a single graphics-processing task." Paragraph 8 also refers to "the threads of the first task." The fact that the threads are "of" a task, for example, indicates that the task is decomposable into those threads and that they are related. Dr. Pedram's deposition testimony demonstrates the same understanding, that in general tasks are decomposable into related threads. Ex. 2007 at 13:5-25:4, 29:22-31:4. To the extent it is argued that the '381 Patent specification leaves open the possibility that a task may decompose into only one thread, Patent Owner disagrees for the reasons set forth in the concurrently-filed Patent Owner's Response. However, it does not add new matter to claim a clearly-disclosed embodiment as done here. Ex. 2008 at ¶ 34.

11. and wherein at least one processing element is configured to receive tasks from each engine.

This limitation is disclosed at least in Paragraphs 27-32, 35, 50-54, and Fig. 2

(depicting processing elements 220) of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, Paragraph 35 states "GPU 110 includes a plurality of virtual engines that share resources of a single shader core." This alone indicates that in general the processing elements are configured to accept tasks from all engines. Paragraph 30 explains that "resources of the shader core are partitioned in space and/or time." The specification further states that "[t]o achieve temporal partitioning, for example, the first and second tasks share a percentage of time of the processing elements (SIMDs) of the shader core." Paragraph 43 explains, for example, that the engines can process different types of tasks. Combined, these all indicate that one way to execute tasks simultaneously in the patented invention is for the processing elements of the shader core to receive and execute tasks from one engine at one time, and to receive and execute tasks from another engine at another time, and that over time the processing elements will receive tasks from each engine. Notably, the specification explains that this functionality does not conflict with spatial partitioning because the "resources of the shader core are partitioned in space ***and***/or time." Ex. 1002 at 12 (emphasis added); Ex. 2008 at ¶ 35. This could be simultaneous partitioning of both types, *i.e.*, some subset of elements employ temporal partitioning and some employ spatial partitioning, or the partitioning could be the same across all elements such that all elements are first partitioned in time and then in space, or vice versa. Ex. 2008 at ¶ 35; *see also id.* at ¶¶ 35-40.

B. Substitute Claim 29 Is Supported by the Originally-Filed Application

Claim 15 has support in the originally-filed application, as explained in the previous section. Ex. 2008 at ¶ 41.

1. 29. 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,

This limitation is disclosed at least in Paragraphs 30, 50-54, and Fig. 2 (depicting processing elements 220 of shader core 214) of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, as Paragraphs 30 and 50-54 explain, the tasks are received by the engines and then "issued" from the engines to the shader core. Further, as a POSITA would know, the engines would or could use a queue, a common computer element, to receive and then issue the tasks. Ex. 2008 at ¶ 42. Claim 3 of the original application also discloses this limitation. Ex. 1002 at 25.

2. wherein the plurality of engines are arranged in parallel with one another,

This limitation is disclosed at least in the Abstract, Paragraphs 13, 15, 29, and Fig. 2 of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, paragraph 13 states that "The plurality of virtual engines is configured to (i) receive, from an operating system (OS), a plurality of tasks substantially in parallel with each other." Further, paragraph 29 states that "[t]asks from each virtual

engine are then issued to a single shader core substantially in parallel with each other." Moreover, in Figures 2 (elements 202), 3A (elements 312), 3B (elements 312), and 4 (elements 432) the engines are depicted as being arranged in parallel with one another. Ex. 2008 at ¶ 43.

3. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine,

This limitation is disclosed at least in the Abstract, Paragraphs 7, 29, 39, 43, 63-64, and Fig. 4 of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, paragraph 64 states that "GPU 110 includes a plurality of virtual engines 432A-N, each configured to service one of command buffers 420A-N," which is depicted in Fig. 4, in which each engine receives tasks via a distinct command buffer. Petitioner's expert, Dr. Pedram, admitted that is what Fig. 4 depicts. Ex. 2007at 81:2-5; Ex. 2008 at ¶ 44.

4. wherein each of the plurality of tasks is decomposable into two or more related threads, and

This limitation is disclosed for the same reasons as in claim 28 above. *See* Section VI.A.10 *supra*; Ex. 2008 at ¶ 45.

5. wherein each engine is configured to load state data associated with at least one of the plurality of tasks.

This limitation is disclosed at least in the Abstract, Paragraphs 13, 16, 27, 53, and Figs. 2, 3A, 3B, and 4 of the originally-filed application, among other places. Ex. 1002 at 6-35. For example, paragraph 13 states that "[t]he plurality of virtual

engines is configured to (i) receive, from an operating system (OS), a plurality of tasks substantially in parallel with each other and (ii) load a set of state data associated with each of the plurality of tasks." As a POSITA would understand this statement is intended to convey that each engine loads the state data associated with the task it receives and transmits to the shader core. Ex. 2008 at ¶ 46. At a minimum, a POSITA would know that was one possible design based on the disclosure. *Id.* at ¶ 47. Moreover, as depicted in the figures, each of the engines receives tasks, *i.e.*, item "(1)"; for the same reason, in the citation above, each engine loads state data with those tasks, *i.e.*, item "(2)."

VII. THE PROPOSED SUBSTITUTE CLAIMS ARE NOT UNPATENTABLE

Although it is Petitioner's burden to demonstrate invalidity of the substitute claims, Patent Owner provides the below exemplary reasons why the closest known prior art, the three references relied on by Petitioner in the IPR proceeding, Wilt, Jiao, and Stuttard, do not invalidate the substitute claims.

The below arguments are directed only to the additional claim limitations provided in the contingent amendments. The grounds further fail to invalidate the current claims for reasons explained in the concurrently-filed Patent Owner's Response. Those arguments are not repeated here.

Patent Owner reserves the right to respond with additional arguments as the briefing schedule permits.

A. Claim 28

1. Wilt

As explained in the concurrently-filed POR, Petitioner has alleged two theories for what constitute the claimed plurality of engines and the claimed shader core. Under what Patent Owner will call the "Single PPU Theory," the work distribution unit 210, the front-end unit 212, and the core interface unit 303 of a single PPU are each an engine, such that they are the plurality of engines, and a single PPU is the shader core. The Institution Decision explained that another theory could be that the cores 208 within one PPU together are the shader core. Ex. 2008 at ¶ 51.

Under Petitioner's second theory, the "Multiple PPU Theory," the work distribution unit 210, the front-end unit 212, and the core interface unit 303 of a single PPU are together a single engine, and that combination of components within each PPU taken together are the claimed plurality of engines. The Petitioner argued under this theory that all PPUs taken together are the claimed shader core. The Institution Decision explained that another theory could be that all cores 208 within all PPUs together are the shader core. *Id.* at ¶ 52.

Under either theory, Wilt fails to disclose claim 28. *Id.* at ¶ 53.

a. wherein each of the plurality of tasks is decomposable into two or more related threads

Claim 28 requires that engines receive from a scheduling module a plurality

of tasks; those tasks must then be received by the shader core, which must execute a first of those tasks ***while*** executing a second of those tasks. Wilt fails to disclose that the alleged shader core executes first and second such tasks wherein each of the plurality of tasks is decomposable into two or more threads. *Id.* at ¶ 54.

A "thread" is a sequence of instructions. Ex. 1001 at 1:47-51. A "task," in this claim limitation, and in keeping with the ordinary meaning of the term and its use in the '381 Patent, is a higher-level construct that is decomposable into two or more threads, and thus comprises ***multiple distinct*** sequences of instructions. *Id.* Wilt operates at the instruction level, or at best the thread level, not the task level. That is, in Wilt, shader core receives and allegedly simultaneously executes at best different instructions or threads, not tasks, at the same time. And there is nothing indicating that any such threads would correspond to different parent tasks. Ex. 2008 at ¶ 55.

Petitioner has alleged that the application software running on the CPU is the claimed scheduling module that transmits tasks that received by the alleged engines and the alleged shader core. Pet. at 26. But in Wilt, the CPU only issues "commands" and a "command stream" that according to Petitioner are ultimately received by the alleged engines and alleged shader core:

> In particular, CPU 102 issues ***commands*** that control the operation of PPUs 202. In some embodiments, 65 CPU 102 writes ***a stream of commands*** for each PPU 202 to a pushbuffer (not explicitly shown in FIG. 1), which may be

> located in system memory 104, PP memory 204, or another storage location accessible to both CPU 102 and PPU 202. PPU 202 reads ***the command stream*** from the pushbuffer and executes commands asynchronously with operation of CPU 102.

Ex. 1004 (Wilt) at 4:63-5:4 (emphases added); *see also* Ex. 2008 at ¶ 56. There is no disclosure of modification of these command streams into tasks before transmitting them to the alleged shader core, and thus, at best, this also only discloses that a shader core can receive and execute (allegedly simultaneously) multiple command streams. Ex. 2008 at ¶ 57.

But without more, executing multiple command streams does not indicate executing multiple tasks simultaneously, because each command stream is only one thread, not two or mroe. This is true in both the Single PPU and Multiple PPU Theories. There is no evidence that threads from two different tasks would ever be executed at the same time, and there is no evidence that what the PPU or any subcomponent executes comprises more than one thread. As Dr. Pedram has admitted and Dr. Hart has explained, in general, looking just at a thread or multiple threads, it is not knowable without more information (that is lacking in Wilt, Jiao, and Stuttard) which task or tasks they were decomposed from; threads could have been decomposed from, and thus correspond to, the same task, for example. Ex. 2008 at ¶ 58; Ex. 2007 at 21:6-27:4. It is insufficient to invalidate the claims to argue that something might, could, or should happen; Petitioner must prove that the

claim language is performed, and it cannot here. *Aspex Eyewear, Inc. v. Marchon Eyewear, Inc*., 672 F.3d 1335, 1349 (Fed. Cir. 2012).

Although Wilt states in some portions that cores 208 execute "tasks," given that the only thing the alleged engines upstream of the shader core ever receive are at best threads, and there is no disclosure of creation of tasks from those threads, the only reasonable interpretation is that the cores actually execute what the '381 Patent would call threads, not tasks as the '381 Patent and the claim language in claim 28 describe that term. Ex. 1004 at 4:1-45. That understanding is confirmed by other statements in Wilt in the same portion of the reference clarifying that "[e]ach processing core 208 is capable of executing a large number (e.g., tens or hundreds) of ***threads*** concurrently. . . ." *Id*. at 4:2-7 (emphasis added). Ex. 2008 at ¶ 59.

b. wherein at least one processing element is configured to receive tasks from each engine

In Wilt, commands enter the PPU via host interface 206, are sent to front end unit 212, then from there to work distribution unit 210, then according to Petitioner and its expert from there to core interface 303, which transmits the commands or command streams (the alleged tasks) to cores 208. Ex. 1004 at 3:49-4:32; Ex. 2007 at 63:24-67:23; Ex. 2008 at ¶ 60.

Thus, as to the Single PPU Theory, the cores 208 only receive alleged tasks from core interface 303 according to Petitioner's expert and no processing element in a PPU receives tasks from each of the alleged engines. Ex. 2008 at ¶ 61.

As to the Multiple PPU Theory, as Dr. Pedram conceded, any core only receives commands from the alleged combined engine within that same PPU. Ex. 2007 at 86:24-88:7. Therefore, no processing element receives commands from more than one engine, let alone is any configured to receive such commands from each engine. Ex. 2008 at ¶ 62.

2. Jiao

Petitioner has recited two theories as to what constitute the engines in Jiao. Under the Shader Theory, the vertex shader 304, geometry shader 306, and pixel shader 310 are the plurality of engines. Pet. at 39. Under the Thread Slot Theory, the plurality of engines are those items along with the "thread slots" within each of the execution units 420 of the EU pool 414 that constitutes the alleged shader core. *Id.* at 40. *See also* Ex. 2008 at ¶ 63.

a. wherein each of the plurality of tasks is decomposable into two or more related threads

As explained in the concurrently-filed Patent Owner's Response, the alleged shaders do not receive instructions or threads, much less tasks; they only receive data, in the form of primitives and constants. To the extent the Board believes the shaders receive instructions, there is no disclosure of a task that is decomposable into two or more related threads. Ex. 2008 at ¶ 64.

Moreover, although Jiao refers to the shaders generating "tasks" that are then received by the execution units in the alleged shader core, the other descriptions

make clear that these so-called "tasks" are what the '381 Patent would refer to as a thread. For example, although Jiao states that "[a]s 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," the further detail regarding how that happens clarifies that those "tasks" are in fact threads. Ex. 1005 at 9:35-38. Jiao states in the next sentence that "the EU pool control unit 206 handles the assigning of those tasks ***to available threads*** within the various execution units 420." *Id*. at 9:39-41. Moreover, Jiao states "EU pool control unit 206 functions as a global scheduler for the tasks and assigns appropriate threads in the execution units 420 using a data-driven approach (e.g., vertex, pixel, geometry packets in the input). For instance, the EU pool control unit 206 assigns a thread to one of the empty thread slots in the respective execution unit 420 of the EU pool 414." *Id.* at 9:15-21. Thus, what the EU pool control does is assign ***threads*** to open ***thread slots*** (using the meaning of that term from the '381 Patent). Ex. 2008 at ¶ 65.

At best, this might disclose receipt and execution of threads, but as with Wilt, there is no disclosure of simultaneous execution of different tasks because there is nothing to indicate any two threads correspond to different parent tasks or that each of the alleged tasks is decomposable into more than one thread. *Id.* at ¶ 66.

b. wherein at least one processing element is configured to receive tasks from each engine

Petitioner has not attempted to demonstrate that Jiao discloses this

functionality and Patent Owner reserves the right to respond if it does.

3. Stuttard

a. wherein each of the plurality of tasks is decomposable into two or more related threads

Stuttard operates at the thread level, from the alleged scheduling module, EPU 8, to the alleged engines, to the alleged shader core, the processing blocks 106—not the task level—and so fails to meet this claim requirement. Stuttard explains that in the thread manager 102 (which Petitioner incorrectly argues sends tasks to the processing blocks 106, the alleged shader core), fetches for execution, "[i]nstructions for ***a thread*** . . . from local memory or the EPU 8 by the fetch unit 1023." Ex. 1006 at 11:12-14 (emphasis added). The cache unit 1024 stores threads, not tasks. *Id.* at 23:5-6. The thread scheduler 1025 operates on threads as well: "First, it stops the active ***thread*** from running, and waits a cycle for any semaphore decrements to propagate." *Id.* at 25:1-3 (emphasis added). Then, thread scheduler activates ***a thread*** based on parameters should as the thread's priority, whether the thread is yielding, etc. *Id*. at 25:5-17. Further, each thread has a "thread processor." *Id.* at 25:26-29. And what is supplied to the processing blocks is an "instruction stream" (*i.e.*, a thread): "In this particular example the same instruction stream is supplied to all of the processing blocks in the system." *Id*. at 11:33-35. Thus, as with Wilt and Jiao, the alleged simultaneous execution by the processing blocks is only of, at best, threads, not tasks, and there is nothing indicating the threads

correspond to different parent tasks, as explained above, nor that any of the alleged tasks is decomposable into more than one thread. Ex. 2008 at ¶ 68.

b. wherein at least one processing element is configured to receive tasks from each engine

Petitioner has alleged that "[t]he binning unit 1069, the channel controller 108, the array controller 104, and components of the thread manager 102 (including the cache memory unit 1024, the thread processors 1026, and the status block 1030) constitute the claimed 'plurality of engines.'" Pet. at 56. *See also* Ex. 2008 at ¶ 69.

Petitioner alleges that the array controller 104 transmits the alleged tasks to the processing blocks. Pet. at 59. None of the other alleged engines are alleged to transmit tasks to the to the alleged shader core, and thus no processing elements is configured to receive tasks from each engine. Ex. 2008 at ¶ 70..

B. Claim 29

1. Wilt

a. wherein the plurality of engines are arranged in parallel with one another

In the Single PPU Theory, the plurality of engines Petitioner has pointed to are arranged in series with one another, not in parallel, as Dr. Pedram conceded. Ex. 2007 at 67:24-68:2. Front end 212 sends threads to work distribution unit 210, which Dr. Pedram alleges sends them to core interface 303, which sends them to core 208 for execution. Ex. 2008 at ¶ 71.

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine

Under either Wilt theory (the Single PPU Theory or the Multiple PPU Theory), the alleged engines in Wilt do not each have a distinct command buffer for each engine, as Dr. Pedram conceded. In the Single PPU Theory, front end 212, work distribution unit 210, and core interface 303 are not described as each having a distinct command buffer, much less one from which the alleged engines receive the alleged tasks. In the Multiple PPU Theory, the combination of those units, one in each PPU, are not described as each having a distinct command buffer, much less one from which the alleged engines receive the alleged tasks. *Id.* at ¶ 72.

c. wherein each of the plurality of tasks is decomposable into two or more related threads

As discussed above with respect to claim 28, Wilt's alleged engines and the alleged shader core only receive and execute at best individual threads, and so the shader core does not simultaneously execute different tasks. *Id.* at ¶ 73.

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks

Only core interface 303 is alleged to load state data. Front end unit 212 and work distribution unit 210 do not load state data, as Dr. Pedram admitted. Ex. 2007 at 73:14-74:3; *id.* at ¶ 74.

2. Jiao

a. wherein the plurality of engines are arranged in parallel with one another

Jiao's shaders, which are the alleged engines are not arranged in parallel with one another; they are arranged serially, such that input of one relies on the direct output of the other, as depicted in Fig. 3, described in Jiao at column 6, lines 1-48, and conceded by Dr. Pedram. Ex. 2007 at 93:3-16 (referencing "transfer of tasks from the software driver, through the API, to the graphics pipeline that goes through the input assembler, the vertex shader, the geometry shader, the rasterizer, the pixel shader and, finally, generates these set of threads that get executed on the shader core, the computational core of Jiao").

The addition of the "thread slots" within the alleged shader core, EUP 414, Petitioner argues in an alternative theory, does not remedy the flaw. The shaders are serially connected to one another and are also serially connected to the shader core, including the thread slots therein, because according to Petitioner's theory, the alleged tasks flow from the shaders to the thread slots within the alleged shader core. Ex. 2008 at ¶ 76.

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine

The shaders in Jiao, the alleged engines, are "not associated with distinct command buffers," as Dr. Pedram testified. Ex. 2007 at 122:4-21. Even less are

they shown to receive tasks via distinct command buffers for each engine. Ex. 2008 at ¶ 77.

c. wherein each of the plurality of tasks is decomposable into two or more related threads

As discussed above in claim 28, Jiao's alleged engines do not receive instructions or threads, much less a task that is decomposable into two or more related threads. Similarly, EU Pool 414 does not simultaneously execute two or more different such tasks; at best Petitioner points to the elements 420 receiving only threads, which does not demonstrate simultaneous execution of multiple different tasks each task decomposable into multiple treads. *Id.* at ¶ 78.

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks

Petitioner does not allege that the thread slots in the Jiao load state data and thus, at least Petitioner's theory in which the thread slots in the shader core are engines fails to disclose this element. Pet. at 39-47. The Petition also does not specifically identify what action by the shaders allegedly meets the requirement to load state data, and thus its primary argument fails as well. Ex. 2008 at ¶ 79.

3. Stuttard

a. wherein the plurality of engines are arranged in parallel with one another

Petitioner has argued that "[t]he binning unit 1069, the channel controller 108, the array controller 104, and components of the thread manager 102 (including the

cache memory unit 1024, the thread processors 1026, and the status block 1030) constitute the claimed 'plurality of engines.'" Pet. at 56. These items are not in parallel with one another, as seen in Figs. 3 and 4. Ex. 1006 at Figs. 3-4. The components of the thread manager are serially linked, and those as a group are serially linked with the array controller and channel controller, which is serially linked to processing blocks 106, which contains binning unit 1069. Ex. 2008 at ¶ 80.

b. wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine

The items alleged to be engines by Petitioner are not connected to a distinct command buffer for each engine, as Dr. Pedram admitted, much less do they receive tasks via any such command buffers. Ex. 2007 at 165:4-8; Ex. 1006 at Figs 3-4; Ex. 2008 at ¶ 81.

c. wherein each of the plurality of tasks is decomposable into two or more related threads

As discussed above with respect to claim 28, Stuttard's processing blocks, the alleged shader core does not receive tasks, let alone simultaneously execute more than task, each task being decomposable into two or more related threads, as at best it is alleged to simultaneously execute threads, which may have come from the same parent task. Ex. 2008 at ¶ 82.

d. wherein each engine is configured to load state data associated with at least one of the plurality of tasks

Petitioner only alleges that the array controller, of all alleged engines, loads

state data; as Dr. Pedram admitted, none of the units in the thread manager, nor the binning unit, load state data. Ex. 2007 at 163:21-164:10; Ex. 2008 at ¶ 83.

VIII. CONCLUSION

For the above reasons, Patent Owner requests that if claims 16 and 17 of the '381 Patent are determined to be unpatentable, they be replaced by proposed substitute claims 28 and 29 and that those claims be found patentable.

Dated: January 17, 2024

/s/Reza Dokhanchy
Reza Dokhanchy (Reg. No. 62,795)
Mintz, Levin, Cohn, Ferris, Glovsky and Popeo, P.C.
3580 Carmel Mountain Road, Suite 300
San Diego, CA 92130
Phone: 858-314-1596
Fax: 858-314-1501
Email: RDokhanchy@mintz.com

CERTIFICATE OF SERVICE

I certify that a copy of Patent Owner's Motion to Amend is being served by electronic mail on the following counsel of record:

Lead Counsel
Jeffrey Johnson
Reg. No. 53,078
ORRICK, HERRINGTON & SUTCLIFFE, LLP
609 Main Street, 40th Floor
Houston, TX 77002-3106
Telephone: (713) 658-6400
Facsimile: (713) 658-6401
Email: 3J6PTABDocket@orrick.com; Realtek-AMD_OHS@orrick.com

Backup Counsel
Christopher J. Higgins
Reg. No. 66,422
ORRICK, HERRINGTON & SUTCLIFFE, LLP
1152 15th Street, N.W.
Washington, DC 20005-1706
Telephone: (202) 339-8400
Facsimile: (202) 339-8500
Email: OCHPTABDocket@orrick.com; Realtek-AMD_OHS@orrick.com

Steve Baik
Reg. No. 42,281
WHITE HAT LEGAL
P. O. Box 382
San Jose, CA 95002
Telephone: (650) 618-5282
Email: sbaik@whitehat.legal

Dated: January 17, 2024

/s/Reza Dokhanchy/
Reza Dokhanchy (Reg. No. 62,795)

APPENDIX A

Patent Owner's amendments in proposed substitute claims 28-29 are set forth below.

Current Claim	Proposed Substitute Claim
15. A computing system comprising: a memory; a first processing unit; and a bus coupled to the memory and the first processing unit, 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 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.	No substitute claim proposed.
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.	28. The computing system of claim 15, wherein the shader core comprises: <u>multiple processing elements;</u> 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,

Current Claim	Proposed Substitute Claim
	wherein each of the plurality of tasks is decomposable into two or more related threads; and wherein at least one processing element is configured to receive tasks from each engine.
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.	29. 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, wherein the plurality of engines are arranged in parallel with one another; wherein each engine is configured to receive at least one of the plurality of tasks via a distinct command buffer for each engine; wherein each of the plurality of tasks is decomposable into two or more related threads; and wherein each engine is configured to load state data associated with at least one of the plurality of tasks.
18. The computing system of claim 15, wherein the first task comprises a low-latency task and the second task comprises a regular-latency task.	No substitute claim proposed.
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.	No substitute claim proposed.
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.	No substitute claim proposed.

APPENDIX B

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
4725831	A	1988-02-16	Coleman, Terrence J.
4805127	A	1989-02-14	Hata et al.
5388206	A	1995-02-07	Poulton et al.
5394524	A	1995-02-28	DiNicola et al.
5485559	A	1996-01-16	Sakaibara et al.
5500939	A	1996-03-19	Kurihara, Katsuhide
5671401	A	1997-09-23	Harrell, Chandlee Bryant
5745125	A	1998-04-28	Deering, et al.
5781199	A	1998-07-14	Oniki et al.
5790134	A	1998-08-04	Lentz, Derek
5808690	A	1998-09-15	Rich, Henry H.
5815165	A	1998-09-29	Blixt, Stefan
5815166	A	1998-09-29	Baldwin, David Robert
5821950	A	1998-10-13	Rentschler, et al.
5938756	A	1999-08-17	Van Hook, et al.
5968167	A	1999-10-19	Whittaker, et al.
5999196	A	1999-12-07	Storm, et al.
6002409	A	1999-12-14	Harkin, Patrick A.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
6057850	A	2000-05-02	Kichury, John J.
6088044	A	2000-07-11	Kwok, et al.
6105127	A	2000-08-15	Kimura, et al.
6201545	A	2001-03-13	Wong, et al.
6222550	A	2001-04-24	Rosman, et al.
6249288	B1	2001-06-19	Campbell, Paul W.
6259460	B1	2001-07-10	Gossett, et al.
6289369	B1	2001-09-11	Sundaresan, Neelakantan
6292194	B1	2001-09-18	Powell, Ill, William Chambers
6331856	B1	2001-12-18	Van Hook, et al.
6340972	B1	2002-01-22	Fox, Thomas Winters
6344852	B1	2002-02-05	Zhu, et al.
6384833	B1	2002-05-07	Denneau, et al.
6388671	B1	2002-05-14	Yoshizawa, et al.
6452595	B1	2002-09-17	Montrym, et al.
6473089	B1	2002-10-29	Wei, et al.
6476807	B1	2002-11-05	Duluk, et al.
6501480	B1	2002-12-31	Maclnnis, et al.
6525737	B1	2003-02-25	Duluk, et al.
6526583	B1	2003-02-25	Auld, et al.
6573893	B1	2003-06-03	Naqvi,et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
6597356	B1	2003-07-22	Moreton, et al.
6597363	B1	2003-07-22	Duluk, et al.
6614444	B1	2003-09-02	Duluk, et al.
6630935	B1	2003-10-07	Taylor, et al.
6640299	B1	2003-10-28	Mang, et al.
6650327	B1	2003-11-18	Alrey, et al.
6707453	B1	2004-03-16	Rossin, et al.
6717577	B1	2004-04-06	Cheng, et al.
6747660	B1	2004-06-08	Olano, et al.
6809732	B2	2004-10-26	Zatz, et al.
6828975	B2	2004-12-07	Noyle, Jeff M. J.
6919894	B2	2005-07-19	Emmot, et al.
6919896	B2	2005-07-19	Sasaki, et al.
6931641	B1	2005-08-16	Davis, et al.
6952214	B2	2005-10-04	Naegle, et al.
6963347	B1	2005-11-08	Selvaggi, et al.
6980209	B1	2005-12-27	Donham, et al.
7015913	B1	2006-03-21	Lindholm, et al.
7038685	B1	2006-05-02	Lindholm
7038686	B1	2006-05-02	Lindholm, John Erik
7102646	B1	2006-09-05	Rubinstein, et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
7103720	B1	2006-09-05	Moy, et al.
7116332	B2	2006-10-03	Boyd, et al.
7233335	B2	2007-06-19	Moreton, et al.
7733349	B1	2010-06-08	Tannenbaum, David C.
7847803	B1	2010-12-07	Van Hook, Timothy J.
8619077	B1	2013-12-31	Cote, et al.
8933945	B2	2015-01-13	Leather, et al.
7477260	B1	2009-01-13	Nordquist, Bryon S.
7750915	B1	2010-07-06	Acocella, et al.
8200947	B1	2012-06-12	Nickolls, et al.
8214625	B1	2012-07-03	Nickolls, et al.
7925860	B1	2011-04-12	Juffa, et al.
6975321	B1	2005-12-13	Lindholm, et al.
8448145	B2	2013-05-21	Ganai, et al.
8139069	B1	2012-03-20	Molnar, et al.
7584342	B1	2009-09-01	Nordquist, et al.
7788468	B1	2010-08-31	Nickolls, et al.
8108625	B1	2012-01-31	Coon, et al.
8108659	B1	2012-01-31	Le Grand, Scott M.
8335892	B1	2012-12-18	Minkin, et al.
7290121	B2	2007-10-30	Kromer, Stephen Charles

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
6507903	B1	2003-01-14	Beatty, et al.
6832378	B1	2004-12-14	Beatty, et al.
7140018	B1	2006-11-21	Beatty, et al.
6839748	B1	2005-01-04	Allavarpu, et al.
7861060	B1	2010-12-28	Nickolls, et al.
6496925	B1	2002-12-17	Rodgers, et al.
5968160	A	1999-10-19	Saito, et al.
7493615	B2	2009-02-17	Gewirtz, et al.
8327378	B1	2012-12-04	Vasanthakumari, et al.
7571301	B2	2009-08-04	Kejariwal, et al.
7477255	B1	2009-01-13	Lindholm, et al.
8677106	B2	2014-03-18	Nickolls, et al.
7117496	B1	2006-10-03	Ramesh, et al.
6857122	B1	2005-02-15	Takeda, et al.
5805470	A	1998-09-08	Averill, Gregory S.
5313622	A	1994-05-17	Truchard et al.
8666940	B2	2014-03-04	Roxburgh et al.
7730501	B2	2010-06-01	Kumar et al.
5613136	A	1997-03-18	Casavant et al.
7562366	B2	2009-07-14	Pope et al.
7779163	B2	2010-08-17	Franchuk et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
7735093	B2	2010-06-08	Gruber et al.
7444641	B1	2008-10-28	Diepstraten et al.
9552206	B2	2017-01-24	Johnson et al.
9189233	B2	2015-11-17	Sasanka, et al.
8484516	B2	2013-07-09	Giannini et al.
8379033	B2	2013-02-19	Molnar et al.
8085272	B1	2011-12-27	Molnar et al.
8788732	B2	2014-07-22	Zhu et al.
7881321	B2	2011-02-01	Deneroff et al.
7752123	B2	2010-07-06	Brookfield et al.
8099452	B2	2012-01-17	Chkodrov et al.
7447873	B1	2008-11-04	Nordquist, Bryon S.
5086386	A	1992-02-04	Islam, Nayeem
7185338	B2	2007-02-27	Chamdani et al.
9459941	B2	2016-10-04	Zwaal, Frederik Hugo
9298498	B2	2016-03-29	Gosalia et al.
8756350	B2	2014-06-17	Blumrich et al.
5222216	A	1993-06-22	Parish et al.
9280444	B2	2016-03-08	Scheerer et al.
6606704	B1	2003-08-12	Adiletta et al.
7594095	B1	2009-09-22	Nordquist, Bryon S.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
8564604	B2	2013-10-22	Jiao, Yang (Jeff)
6784889	B1	2004-08-31	Radke, William
9098336	B2	2015-08-04	Adachi et al.
8589942	B2	2013-11-19	lsani, Tarik
7904704	B2	2011-03-08	Kang et al.
8190826	B2	2012-05-29	Lefebvre et al.
8438370	B1	2013-05-07	Budge, Brian C.
7526634	B1	2009-04-28	Duluk et al.
9158575	B2	2015-10-13	Smith, Graeme Roy
8319774	B2	2012-11-27	Jiao et al.
7634637	B1	2009-12-15	Lindholm et al.
8633936	B2	2014-01-21	Du et al.
8223158	B1	2012-07-17	Lindholm et al.
8151095	B1	2012-04-03	Wilt, Nicholas Patrick
8542247	B1	2013-09-24	Hakura et al.
8144149	B2	2012-03-27	Jiao et al.
8141088	B2	2012-03-20	Morishita et al.
7605825	B1	2009-10-20	Lau et al.
7324520	B2	2008-01-29	Lakshmanamurthy et al.
7673304	B2	2010-03-02	Gosalia et al.
8345053	B2	2013-01-01	Jiao et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
7325232	B2	2008-01-29	Liem, Clifford
8489857	B2	2013-07-16	Hobson et al.
8321870	B2	2012-11-27	Messier et al.
7629978	B1	2009-12-08	Diard, Franck R.
7616207	B1	2009-11-10	Diard, Franck R.,et al.
7685371	B1	2010-03-23	Duncan, etl. al
8368701	B2	2013-02-05	Paltashev et al.
6693639	B2	2004-02-17	Duluk et al.
8102402	B2	2012-01-24	Sorgard et al.
6819793	B1	2004-11-16	Reshetov et al.
6683978	B1	2004-01-27	lourcha et al.
5956431	A	1999-09-21	lourcha et al.
7873212	B2	2011-01-18	Roimela et al.
7961195	B1	2011-06-14	Rogers et al.
7385611	B1	2008-06-10	Toksvig et al.
8401082	B2	2013-03-18	Ye et al.
7388993	B2	2008-06-17	Govindaswamy et al.
7266150	B2	2007-09-04	Demos, Gary A.
7787691	B2	2010-08-31	Str6m, Jacob
6339428	B1	2002-01-15	Fowler et al.
8165195	B2	2012-04-24	Song, Byung-cheol

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
6959110	B1	2005-10-25	Danskin et al.
5892847	A	1999-04-06	Johnson, Stephen G.
6285796	B1	2001-09-04	Tinku et al.
7693337	B2	2010-04-06	Strom et al.
7171051	B1	2007-01-30	Moreton et al.
7116335	B2	2006-10-03	Pearce et al.
6940511	B2	2005-10-06	Akenine-Moller et al.
7242811	B2	2007-07-10	Fenney, Simon
7050503	B2	2006-05-23	Prakash et al.
7599975	B1	2006-10-06	Donovan et al.
5585944	A	1996-12-17	Rodriguez, Arturo A.
7050639	B1	2006-05-23	Barnes et al.
8208564	B2	2012-06-26	Bossen et al.
7382917	B2	2008-06-03	Barone et al.
5856829	A	1991-01-05	Gray et al.
6778181	B1	2004-08-10	Kilgariff et al.
5943058	A	1999-08-24	Nagy, Michael B.
6795581	B1	2004-09-21	Nomura, Yoshiro
5764802	A	1998-06-09	Simon, Al
7714873	B2	2010-05-11	Flavell et al.
7742053	B2	2010-06-22	Lefebvre, et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
7196710	B1	2007-03-27	Fouladi
6975773	B1	2005-12-13	Govindaswamy
20010024205	Al	2001-09-27	Kishi, et al.
20020033817	Al	2002-03-21	Boyd, et al.
20030076320	Al	2003-04-24	Collodi, et al.
20040100471	Al	2004-05-27	Leather, et al.
20050195210	Al	2005-09-08	Demers, et al.
20120017062	Al	2012-01-19	Goel, et al.
20030140179	Al	2003-07-24	Wilt, et al.
20110078381	Al	2011-03-31	Heinrich, et al.
20090240860	Al	2009-09-24	Coon, et al.
20110078692	Al	2011-03-31	Nickolls, et al.
20110078417	Al	2011-03-31	Fahs, et al.
20090259832	Al	2009-10-15	Grover, et al.
20100251240	Al	2010-09-30	Matesan, Cristian M.
20060236136	Al	2006-10-19	Jones, Darren
20090312856	Al	2009-12-17	Jong, et al.
20070285286	Al	2007-12-13	Hussain, et al.
20050188177	Al	2005-08-25	Gao, et al.
20060146864	Al	2006-07-06	Rosenbluth et al.
20070214343	Al	2007-09-13	Lindholm et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
20070220525	Al	2007-09-20	State et al.
20090240931	Al	2009-09-24	Coon et al.
20080010640	Al	2008-01-10	Foo, Yoong Chert
20100005277	Al	2010-01-07	Gibert et al.
20110032257	Al	2011-02-10	Peterson et al.
20070273699	Al	2007-11-29	Sasaki et al.
20130226909	Al	2013-08-29	Katsunuma et al.
20030182503	Al	2003-09-25	Leong et al.
20080184211	Al	2008-07-31	Nickolls, et al.
20090150654	Al	2009-06-11	Oberman et al.
20090073168	Al	2009-03-19	Jiao et al.
20120204183	Al	2012-08-09	Bayer et al.
20100123717	Al	2010-05-20	Jiao, Yang (Jeff)
20070091097	Al	2011-03-08	Zhang, Guofeng
20100153686	Al	2011-04-19	Frank, Michael
20110050712	Al	2011-03-03	Jackson, Adam D.
20090189909	Al	2009-07-30	Jiao et al.
20060279577	Al	2006-12-14	Bakalash et al.
20090153570	Al	2009-06-18	Jiao et al.
20080201716	Al	2008-08-21	Du et al.
20040151372	Al	2004-08-05	Reshetov et al.

U.S. PATENTS			
Patent Number	**Kind Code**	**Issue Date**	**Name of Patentee or Applicant of Cited Document**
20040228527	Al	2004-11-08	lourcha et al.
20040252894	Al	2004-12-16	Miyanohara, Makoto
20040081356	Al	2004-04-29	Shimizu et al.
20030138152	Al	2003-07-24	Fenney, Simon
20050002579	Al	2005-01-06	Mitchell et al.
20020171644	Al	2002-11-21	Reshetov et al.
20030227462	Al	2003-12-11	Akenine-Moller et al.
20030048955	Al	2006-01-24	Pardas et al.

FOREIGN PATENT DOCUMENTS				
Foreign Document Number	**Country Code**	**Kind Code**	**Alleged Publication Date**	**Name of Patentee Applicant of Cited Document**
1029269	EP	Al	2020-08-23	International Business Machines Corp
1291765	EP	B1	2009-12-30	Panasonic Corp
2348971	GB	B	2000-10-18	Pixelfusion Ltd ClearSpeed Technology Ltd
2000062182	WO	A2	2000-10-19	Stuttard et al.
2001016883	WO	A2	2001-03-08	Blythe et al.
2009113034	WO	Al	2009-09-17	Terechko et al.
1369985	CN	A	2002-09-18	Silicon Integrated Systems Corp
1993038	EP	Al	2008-11-19	Sony Interactive Entertainment Inc
2001073618	WO	A2	2001-10-04	Ussery et al.
1912119	EP	B1	2010-05-19	RHF GBR
2042994	EP	A2	2009-04-01	Fortent Americas Inc.
2274754	GB	A	1994-08-03	Samsung Electronic Ltd
2007021227	WO	Al	2004-12-16	Strom et al.
2006006915	WO	Al	2006-01-19	Strom et al.
2004104621	JP	A	2004-04-02	Toshiba Corp
1997006512	WO	A2	1997-02-20	Myhrvold et al.
2000057649	WO	Al	2000-09-28	Hodgson, Julian
2005348410	JP	A	2005-12-15	Prokop et al.
2006092556	JP	A	2006-04-05	Chiu et al.
2001223913	JP	A	2001-08-17	Watanabe et al.
2004159311	JP	A	2004-06-03	Monobe et al.

FOREIGN PATENT DOCUMENTS				
Foreign Document Number	**Country Code**	**Kind Code**	**Alleged Publication Date**	**Name of Patentee Applicant of Cited Document**
2006121645	JP	A	2006-05-11	Fujifilm Holdings Corp
2004289274	JP	A	2004-10-14	Seiko Epson Corp
2000196460	JP	A	2000-07-14	Canon Inc
2004054268	WO	Al	2004-06-24	Rim, Kyung-Soo
2006082276	WO	Al	2006-08-10	Eskelinen, Paavo
H08329260	JP	A	1996-12-13	HP Inc
H11168726	JP	A	1999-06-22	Panasonic Holdings
2002512490	JP	A	2002-04-23	Heidelberger Druckmaschinen AG
2005531833	JP	A	2005-10-20	Stroem et al.
2006133839	JP	A	2006-05-25	Yamada
2003134344	JP	A	2003-05-09	Oyama
2000063770	WO			Williams

NON-PATENT LITERATURE DOCUMENTS
Intel, i860 64 - BIT Microprocessor Hardware Design Guide (February 1989)
Intel, i860 Microprocessor Family Programmer's Reference Manual (1991)
Keckler et al., Processor Coupling: Integrating Compile Time and Runtime Scheduling for Parallelism (1992)
Eggers et al., Simultaneous Multithreading: A Platform For Next-Generation Processors (1997)
3dfx Interactive, FXT1 Texture Compression Technology White Paper
3dfx's New Open-Source Texture Compression, Computer Games Magazine, No. 109, December 1999
Akeley, RealityEngine Graphics (1993)
Amanatides et al., A Simple, Flexible, Parallel Graphics Architecture (1993).
AMD, Radeon R6xx/R7xx Acceleration, Rev. 1.0, (April 2009).
AMD's Single-Slot Superstars, Computer Power User, September 2007, at 24-25; *First Look: iLife and iWork '08*, Macworld, October 2007.
ARB texture compression, https://registry.khronos.org/OpenGL/extensions/ARB/ARB_texture_compression.txt (Last July 19, 2022)
Arik Hesseldahl, *Do you voodoo like I do?*, Electronic News, Vol. 45, No. 47, November 22, 1999
Arik Hesseldalh, Do You Doodoo Like I do?, Electronic News, Nov. 12, 1999; The New Voodoo, PC Gamer, February 2000
Bakhoda, Analyzing CUDA Workloads Using a Detailed GPU Simulator (April 2009)
Beers et al, Rendering from Compressed Textures (August 1996)
Behrens, Rendering with Poxels A software architecture for programmable shading systems (1999).
Chan et al., Efficient Partitioning of Fragment Shaders for Multipass Rendering on Programmable Graphics Hardware, Graphics Hardware (2002).

NON-PATENT LITERATURE DOCUMENTS
Chen & Lee, A JPEG-like texture compression with adaptive quantization for 3D graphics (2001)
Compressing Dynamically Generated Textures on the GPU by Alexanderson et al. (2006)
Cox et al., Multi-Level Texture Caching for 3D Graphics Hardware (1998)
Domonkos et al., A programming model for GPU-based parallel computing with scalability abstraction, Proceedings of the 25th Spring Conference on Computer Graphics.
Delp et al., Image Compression Using Block Truncation Coding (October 1979)
Eggers et al, Simultaneous Multithreading: A Platform for Next-Generation Processors (1997).
Ellsworth, Polygon Rendering for Interactive Visualization on Multicomputers (1996).
ETC1 (Ericsson Texture Compression)
Fenney, Texture Compression using Low-Frequency Signal Modulation (2003)
Fernando et al., Programming Graphics Hardware, Eurographics (2004).
Fiske et al., Thread Prioritization: A Thread Scheduling Mechanism for Multiple-Context Parallel Processors (January 1995).
Fout et al, High Quality Rendering of Compressed Volume Data Formats (2005)
Gaster, OpenCLTM - Parallel computing for CPUs and GPUs (2010).
H. Selzer, *Dynamic Load Balancing Within a High Performance Graphics System,* Proceedings of the Sixth Eurographics Conference on Advances in Computer Graphics Hardware: Rendering, Visualization and Rasterization Hardware pp. 37-53, September 1, 1991
Hakura et al, The Design and Analysis of a Cache Architecture for Texture Mapping (1997)
Heywood et al., Data-parallel agent-based microscopic road network (2017)
Hirata et al., An Elementary Processor Architecture with Simultaneous Instruction Issuing from Multiple Threads (1992).
https://techre-port.com/review/12458/amds-radeon-hd-2900-xt-graphics-processor/
https://web.archive.org/web/20041120003604/http://www3.sharkyextreme.com/hardware/articles/99/3dfx_fxt/wp/4.shtml

NON-PATENT LITERATURE DOCUMENTS
https://www.amd.com/en/support/graphics/amd-radeon-hd/ati-radeon-hd-2000-series/ati-radeon-hd-2600-xt
Igehy, Scalable Graphics Architectures: Interface & Texture (May 2000).
Intel, i860 64 - BIT Microprocessor Hardware Design Guide (February 1989).
Intel, i860 Microprocessor Family Programmer's Reference Manual (1991).
Ivanov et al, Color Distribution - A New Approach to Texture Compression (December 24, 2001)
Jacob Ström et.al, iPACKMAN - High-Quality, Low-Complexity Texture Compression for Mobile Phones (July 30, 2005)
Khronos OpenCL Working Group, The OpenCL Specification, Version: 1.0, Document Revisions; 48, Editor: Aaftab Munshi (October 2009)
Kung, Synchronized and Asynchronous Parallel Algorithms for Multiprocessors, (June 1976).
L.L. Winger, Linearly Constrained Generalized Lloyd Algorithm for Reduced Codebook Vector Quantization (July 2001)
Lalonde et al., Shader-driven compilation of rendering assets, Proceedings of the 29th annual conference on Computer graphics and interactive techniques (2002)
Levkovich-Maslyuk et al., Texture compression with adaptive block partitions (2000)
Mark et al., Cg, A system for programming graphics hardware in a C-like language, ACM SIGGRAPH 2003 Papers (2003).
Marziale et al., Massive threading: Using GPUs to increase the performance of digital forensics tools (September 2007)
McCool, SMASH: A Next-Generation API for Programmable Graphics Accelerators (August 1, 2000).
McCool, SMASH: A Next-Generation API for Programmable Graphics Accelerators (April 20, 2001)
Mike Andrawes, 3dfx Interview (September 1999), available at https://www.anandtech.com/show/375/5
Mullis et al, 3DFX texture compression FXT1 (Apr. 2000)

NON-PATENT LITERATURE DOCUMENTS
NVIDIA, Nvidia Compute, Ptx: Parallel Thread Execution ISA Version 1.4 (March 2009)
NVIDIA, Nvidia Compute, Ptx: Parallel Thread Execution ISA Version 2.2 (August 2010)
Nvidia, NVIDIA GeForce 8800 GPU Architecture Overview (November 2006)
OpenCL, ATI Stream Computing, Programming Guide (August 2010)
Owens et al., A Survey of General-Purpose Computation on Graphics Hardware (May 2005)
Owens et al., Polygon Rendering on a Stream Architecture (2000)
Owens, Computer Graphics on a Stream Architecture (November 2002)
PACK-MAN - High-Quality, Low-Complexity Texture Compression for Mobile Phones (July 30, 2005)
Proudfoot et al., A real-time procedural shading system for programmable graphics hardware, Proceeding of the 28th annual conference on Computer graphics and interactive techniques (2001)
Raasch et al., Applications of Thread Prioritization in SMT Processors (1999)
Seiler et al., Larrabee: A Many-core x86 Architecture for Visual Computing (August 2008)
Smartshader, Smartshader Technology White Paper (2001)
Stachera, et al., Fractal-based Hierarchical mip-pyramid texture compression, February 2006
Stokes, Introduction to Multithreading, Superthreading and Hyperthreading (October 2, 2002)
Torborg et al., Talisman: Commodity Realtime 3D Graphics for the PC (1996)
Tullsen et al., Simultaneous Multithreading: Maximizing On-Chip Parallelism (June 1995)
Ungerer et al., Multithreaded Processors (December 20, 2001)
Wang et al., High Throughput and Low Memory Access Sub-Pixel Interpolation Architecture For H.264/AVC HDTV Decoder (August 2005)
Weber et al., Exploring the Benefits of Multiple Hardware Contexts in a Multiprocessor / Results (1989)
Witheiler, ATI SMARTSHADER Technology - Powering the next generation Radeon (July 18, 2001)

NON-PATENT LITERATURE DOCUMENTS
Zhao et al., A parallel computing approach to viewshed analysis (2013)
Zhu, Efficient synchronization for a large-scale multi-core chip architecture, University of Delaware ProQuest Dissertations Publishing (2007)