

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TCL INDUSTRIES HOLDINGS CO. LTD,
Petitioner,

v.

ATI TECHNOLOGIES ULC,
Patent Owner.

IPR2024-00366
Patent 8,760,454 B2

Before JAMES P. CALVE, BRIAN J. McNAMARA, and
KEVIN W. CHERRY, *Administrative Patent Judges*.

McNAMARA, *Administrative Patent Judge*.

DECISION
Granting Institution of *Inter Partes* Review
35 U.S.C. § 314

I. INTRODUCTION

On January 2, 2024, TCL Industries Holdings Co., Ltd. (“Petitioner”) filed a petition, Paper 1 (“Petition” or “Pet.”), to institute an *inter partes* review (“IPR”) of claims 1–11 (the “challenged claims”) of U.S. Patent No. 8,760,454 B2 (“the ’454 patent”). 35 U.S.C. § 311. Petitioner also filed a Motion For Joinder with *Realtek Semiconductor Corp. v. ATI Technologies ULC*, IPR2023-00922 (“the Realtek IPR”). Paper 3 (“Motion For Joinder” or “Mot.”).¹ Petitioner’s Motion for Joinder states that Petition relies on the identical arguments and grounds and the same expert opinions and testimony as those asserted by the petitioner (Realtek) in the petition filed in the Realtek IPR. Mot. 3–4. We instituted trial in the Realtek IPR on December 1, 2023 and entered a scheduling Order in the Realtek IPR on December 5, 2023. Realtek IPR, Papers 10, 11.

In this proceeding, a Notice of Filing Date Accorded was entered on January 18, 2024, setting the due date for Patent Owner’s Preliminary Response to April 18, 2024. On February 2, 2024, ATI Technologies ULC (“Patent Owner”) filed a Response to Petitioner’s Motion for Joinder in which Patent Owner did not object to the joinder, given Petitioner’s agreement to assume an understudy role in a joined proceeding. Paper 7. We recognize Patent Owner’s acquiescence to joinder as effectively acknowledging that the challenges asserted by TCL are identical to those in

¹ In this Decision, citations to papers in the Realtek IPR are preceded with “Realtek IPR,” e.g., the Decision to Institute in the Realtek IPR is cited as “Realtek IPR Dec. to Inst.,” the Preliminary Response in the Realtek IPR is cited as “Realtek IPR Prelim. Resp.,” and the Petition in the Realtek IPR is cited as “Realtek IPR Pet.”

the Realtek IPR and waiving the filing of a Preliminary Response in this proceeding. Therefore, we proceed to this Decision.

We have jurisdiction under 35 U.S.C. § 6. This Decision on Institution is issued pursuant to 35 U.S.C. § 314, which provides that an *inter partes* review may not be instituted unless the information presented in the Petition “shows that there is a reasonable likelihood that the Petitioner would prevail with respect to at least 1 of the claims challenged in the petition.”

A decision to institute under § 314 may not institute on fewer than all claims challenged in the petition. *SAS Inst., Inc. v. Iancu*, 138 S. Ct. 1348, 1359–60 (2018). In addition, per Board practice, if the Board institutes trial, it will institute “on all of the challenged claims and on all grounds of unpatentability asserted for each claim.” *See* 37 C.F.R. § 42.108(a).

Having considered the arguments and the associated evidence presented in the Petition and in the Realtek IPR, we institute *inter partes* review.

II. REAL PARTIES IN INTEREST

The Petition identifies TCL Industries Holdings Co., Ltd. (“TCL”) and 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; TCL Overseas Marketing Ltd., and TTE Technology, Inc. as real parties-in-interest. Pet. 1–2.

Patent Owner identifies ATI Technologies ULC as its real party-in-interest. Paper 5, 1.

III. RELATED MATTERS

Petitioner and Patent Owner identify the following proceedings as ones that may affect or be affected by a decision in this proceeding:

Advanced Micro Devices, Inc. et al v. TCL Industries Holdings Co., Ltd. et al, C.A. No. 2:22-cv-00134 (E.D. Tex. May 5, 2022) (“the AMD Litigation”); and

Certain Graphics Systems, Components Thereof, and Digital Televisions Containing the Same, Inv. No. 337-TA-1318 (“the ITC Investigation”);

Realtek Semiconductor Corp. v. ATI Technologies ULC, No. IPR2023- 00922;

Pet. 2–3, Paper 5, 1. Petitioner states that “the asserted patent claims were terminated by Order No. 10 on July 14, 2022 upon motion of ATI” and “the target date for completion of the investigation is January 23, 2024.” Pet. 2–3. AMD litigation is stayed pending final resolution of the ITC Investigation, expected on or about November 7, 2023. *Id.*

IV. THE ’454 PATENT

The ’454 patent concerns a graphics processing architecture that employs a single or “unified shader,” i.e., a shader that “is configured to perform both vertex and pixel operations.” Ex. 1001, 1:32–33, 3:10–12. The ’454 patent explains that, in computer graphics, complex shapes and structures are formed by sampling, interconnecting, and rendering simpler objects, e.g., triangles or other suitable polygons, called primitives. *Id.* at 1:38–42. Primitives are formed by interconnecting individual pixels. *Id.* at 1:42–43. In order to render an object for display, based on the location of the pixels within the primitives and the primitives’ orientation with respect

to the desired shape, color and texture are applied to the individual pixels that make up the shape to be generated. *Id.* at 1:42–48.

Graphics processors that interconnect the primitives and apply color and textures to the generated shapes include a series of shaders that specify how a final image is drawn on a display device and its corresponding attributes. *Id.* at 1:49–54. A shader receives shape data in object space (x , y , z), color information, texture information, luminance information, and viewing angle information and produces output data (x' , y' , z') that represent the object with texture and other appearance properties applied to it. *Id.* at 1:55–60. Figs. 2A, 2B of the '454 patent show vertex data V_x , V_y , V_z of a cube applied to a vertex shader that outputs angularly oriented vertices V_x' , V_y' , V_z' and appearance attributes of a corresponding cube. *Id.* at 2:3–7. A pixel shader operating at the pixel level provides the color value associated with each pixel of a rendered object. *Id.* at 2:8–12.

According to the '454 patent, in a conventional graphics processor, the vertex shader and pixel shader are “separate components that are configured to perform only a single transformation or operation. Ex. 1001, 2:12–15. “In conventional graphics processors, the vertex shader and the pixel shader are juxtaposed in a sequential, pipelined fashion, with the vertex shader being positioned before and operating on vertex data before the pixel shader can operate on individual pixel data.” *Id.* at 2:25–29, 4:5–7. Figure 3, reproduced below, is a schematic of such a conventional shader.

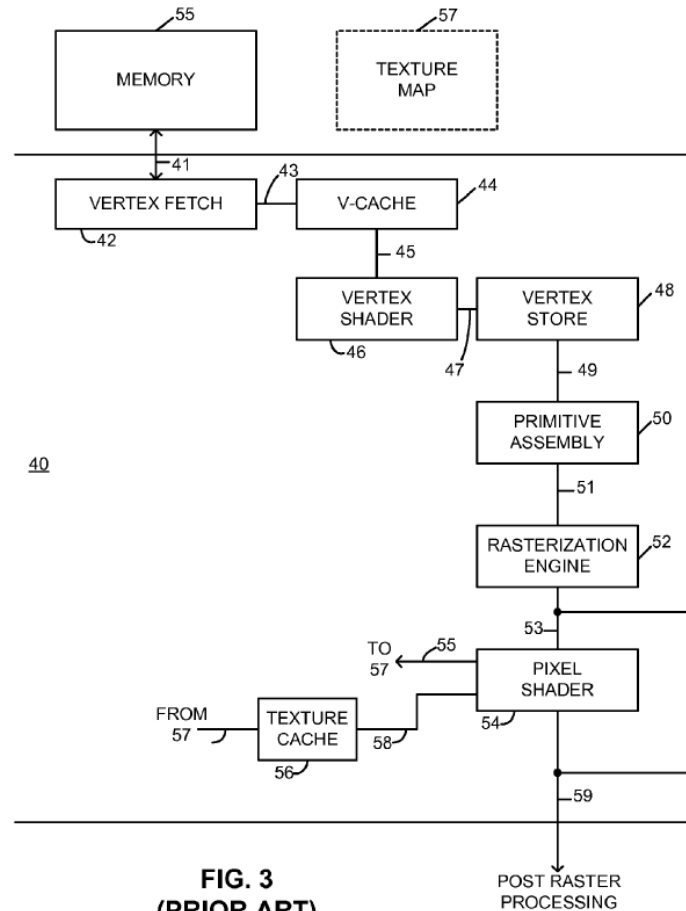


FIG. 3
(PRIOR ART)

In graphics processor 40 of Figure 3, on line 41 vertex fetch block 42 receives from off-chip memory 55 vertex information relating to primitives to be rendered. *Id.* at 3:23–27. The fetched vertex data (e.g., object shape, color and texture information, viewing angle) is stored in vertex cache (V-cache) 44 and, upon request, is transmitted on line 45 to vertex shader 46. *Id.* at 3:26–30. Vertex shader 46 typically is programmed to apply a transformation position matrix to the input position information supplied from V-cache 44 and thereby provide data representing a perspective corrected image of the object to be rendered, as well as texture and color coordinates for the image. *Id.* at 3:30–39. Vertex shader 46 transmits the transformed vertices to vertex store 48, which then transmits the modified

vertex information to primitive assembly block 50, where the input vertex information is converted to primitives using well known techniques. *Id.* at 3:40–49. Rasterization engine 52 receives and converts the previously assembled primitives into pixel data that is then transmitted to pixel shader 54. *Id.* at 3:49–53. Pixel shader 54 generates color and additional appearance attributes for each pixel and applies the attributes to respective pixels. Pixel shader 54 can fetch texture data from texture map 57, as indexed by pixel data from rasterization engine 52, and apply logical or arithmetic operations on the received texture data to generate the pixel color or other appearance attributes of interest. *Id.* at 3:54–66. The pixel appearance attribute is also combined with a base color provided by rasterization engine 54 to provide a pixel color corresponding to the position of interest. *Id.* at 3:66–4:4.

To avoid the computational inefficiency and space requirements of separate vertex and pixel shaders, the '454 patent incorporates into the graphics processor a unified shader that performs both pixel and vertex operations, as shown in Figures 4A and 4B. Ex. 1001, 2:45–46, 4:5–28. Figure 4A, reproduced below, is an exemplary embodiment of a graphics processor with a unified shader. *Id.* at 3:47–48, 4:13.

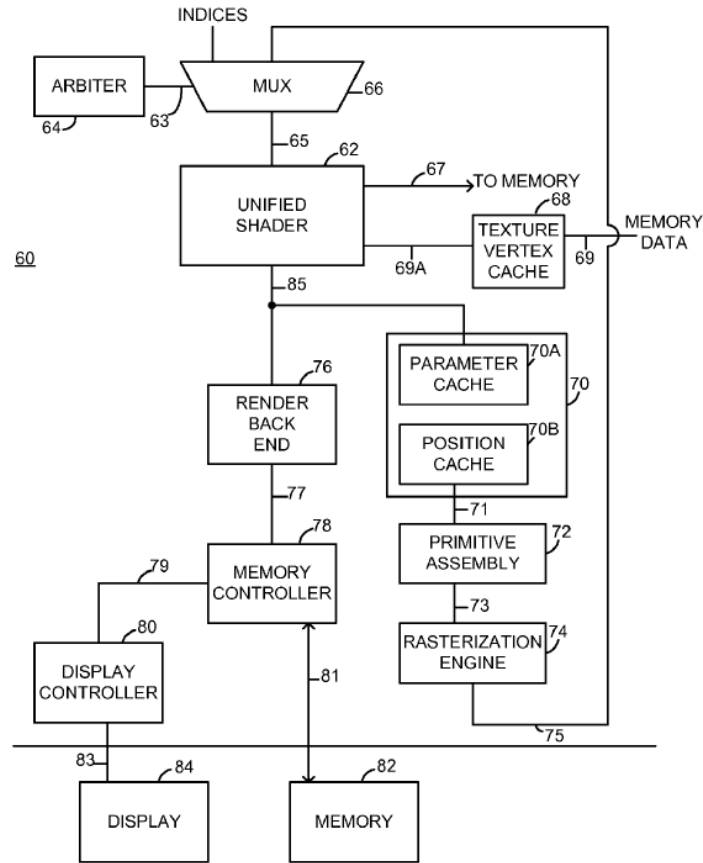


FIG. 4A

Ex. 1001, Fig. 4A. As Figure 4A illustrates, graphics processor 60 includes arbiter 64, multiplexer (mux) 66, and unified shader 62. A control signal on line 63 from arbiter 64 to mux 66 determines which of two inputs mux 66 supplies to unified shader 62. *Id.* at 4:19–21. Mux 66 sends unified shader 62 vertex data (e.g., indices) from a first mux input if there are enough resources available in the shader to operate on the vertex data; otherwise, mux 66 sends unified shader 62 interpolated pixel parameter data from rasterization engine 74 on line 75. *Id.* at 4:13–28.

Figure 5, reproduced below, is a schematic of unified shader 62.
 Ex. 1001, 4:31–32.

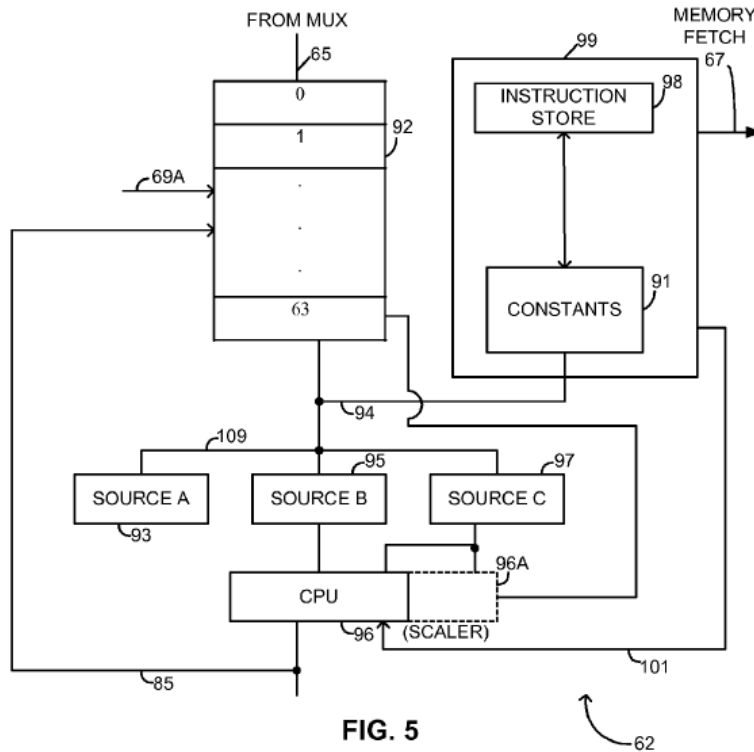


FIG. 5

Id. at Fig. 5. Shader 62 includes (i) register block 92 (comprising 64 general purpose registers), (ii) source registers A, B, and C (reference designators 94, 95, and 97, respectively), (iii) CPU 96 (adapted to perform 32-bit floating point arithmetic operations and logical operations on corresponding operands and, in section 96A scalar operations, e.g., log, exponent), and (iv) sequencer 99. *Id.* at 4:40–52. Sequencer 99, which includes instruction store 98, determines whether the next instruction to be executed from instruction store 98 on data maintained in general purpose register 92, as provided by the source registers, is an arithmetic instruction, or a logical instruction or a memory instruction, e.g., a texture fetch. *Id.* at 4:52–66. Sequencer 99 also includes constants block 91—constants block 91 includes transformation matrices used in connection with vertex manipulation operations. *Id.* at 4:53–55. Instruction store 98 maintains both vertex manipulation and pixel manipulation instructions, enabling the shader to

perform both vertex and pixel operations, and to execute memory fetch operations based on the information received from mux 66. *Id.* at 5:19–27.

In application, when vertex data (e.g., indices) to be processed is transmitted to general purpose register block 92 from mux 66, instruction store 98 passes corresponding control signals on line 101 to processor 96 to perform vertex operations. If general purpose block 92 does not have enough space available to store the incoming vertex data, arbiter 64 will not instruct mux 66 to transmit vertex data. *Id.* at 5:31–44. Instead, any pixel calculations currently being performed by processor 96 are continued based on instructions from instruction store 98, until enough registers in block 92 become available to accommodate the vertex data. *Id.* at 5:45–49. Unified shader 62 can switch from executing vertex to pixel instructions, regardless of the degree of completion, reducing the down time of processor 96. *See id.* at 5:32–52.

Pixel based output from the unified shader on line 85 is sent to cache block 70 that includes parameter cache 70A and position cache 70B. Ex. 1001, 5:54–58, Fig. 4A. Pixel information in cache block 70 is transmitted to primitive assembly block 72, where the information is assembled into a series of triangles or other primitives and transmitted to rasterization engine block 74. Rasterization block 74 converts the primitives to individual pixel data, e.g., through a walking process, and interpolated pixel data is transmitted to the second input of the mux on line 75. *Id.* at 5:63–6:4.

Vertex data output from shader 62 on line 85 is transmitted to back end block 76, where the vertex data is converted to a format suitable for display on display device 84, and transmitted to memory 82 and display controller 80, where the formatted information is routed to display 84. Ex. 1001, 6:5–18.

V. ILLUSTRATIVE CLAIMS

Independent claim 1 is drawn to a method carried out by a unified shader. Ex. 1001, 6:52–7:11. Independent claims 2, 3, 4, 5, and 11 are drawn to a “unified shader,” a term in the preambles that limits the claims to shaders that can perform both vertex and pixel processing. Ex. 1001, 7:13–8:30. *See* Section VIII.B herein. Claims 6–9 depend from claim 5. Claim 10 depends from claim 7.

Independent claim 2, reproduced below using Petitioner’s paragraph designations, is illustrative:

- 2. [pre] A unified shader, comprising:
 - [a] a general purpose register block for maintaining data;
 - [b] a processor unit;
 - [c] a sequencer, coupled to the general purpose register block and the processor unit, the sequencer maintaining instructions operative to cause the processor unit to execute vertex calculation and pixel calculation operations on selected data maintained in the general purpose register block; and
 - [d] wherein the processor unit executes instructions that generate a pixel color in response to selected data from the general purpose register block and generates vertex position and appearance data in response to selected data from the general purpose register block.

Ex. 1001, 6:65–7:11. Independent claim 2 recites that the processor unit executes instructions in response to selected data, but does not include a limitation reciting the selection of operations based on the amount of space available in a data store. Independent claim 5, reproduced below, is illustrative of a claim with such an express limitation.

- 5. A unified shader comprising:
 - a processor unit;
 - a sequencer coupled to the processor unit, the sequencer maintaining instructions operative to cause the processor unit to execute vertex calculation and pixel calculation operations

on selected data maintained in a store depending upon an amount of space available in the store.

Ex. 1001, 8:1–8; *see also* claims 1, 3, 4.

In the Realtek IPR, Patent Owner characterizes the '454 patent as having several salient features discussed below that are embodied in the claims. Realtek IPR Prelim. Resp. 9–17.

First, Patent Owner asserts that all of claims 1–11 require a unified shader, i.e., a shader the Specification defines as one that is capable of executing both vertex and pixel threads to transform both vertex and pixel data, respectively². *Id.* at 9–11 (citing Ex. 1001, 2:58–61; Ex. 2001, Mangione-Smith Decl. ¶ 29), 15. The embodiment of the unified shader in the '454 patent includes a register that stores vertex and pixel data that are to be transformed by the unified shader, a sequencer that stores individual instructions that form vertex and pixel threads that the unified shader uses to manipulate and transform the store vertex and pixel data, and a processor that can execute the threads. *See id.* at 10–11.

Second, Patent Owner asserts that the unified shader in the '454 patent allocates the unified shader's resources to switch between unfinished threads to prevent downtime and data backlogs. *See id.* at 11–13; Ex. 1001, 8:22–30 (claim 11).

Third, Patent Owner states that, unlike prior art systems that use pre-determined priorities or balance workloads based on how many of a particular task is in the queue to execute, the '454 patent selects specific data based on the availability of space in the data storage for particular types of data. Realtek IPR Prelim. Resp. 13, 15–16. For example, Patent Owner

² We more fully address the construction of the term “unified shader” in Section VIII.B herein.

notes that if there is insufficient room in the data storage for vertex data, the processor continues pixel operations until enough registers within the general purpose register block become available to store vertex data. *Id.* at 13–14 (citing, *e.g.*, Ex. 1001, code (57)) 15 (asserting that claims 1 and 3–11 require performing operations based on this dynamically-monitored data storage capacity).

Fourth, Patent Owner asserts that, unlike the prior art, in which a shader performs tasks based on received instructions, the unified shader of the '454 patent is data-driven, such that the receipt of particular selected data dictates the execution of instructions, *e.g.*, the unified shader performs one of the vertex operations or pixel operations based on the selected one of the plurality of inputs. *Id.* at 14 (citing Ex. 1001, 2:58–3:17; Ex. 2001, Mangione-Smith Decl. ¶ 39), 17 (noting that claim 2 recites executing instructions to generate a pixel color and vertex instructions that generate vertex and appearance data in response to selected data).

VI. ASSERTED GROUNDS

Petitioner asserts the same challenge grounds as those asserted in the Realtek IPR. Pet. 18–85. Petitioner asserts that claims 1–11 would have been unpatentable on the following grounds:

Claim(s) Challenged	35 U.S.C. §	Reference(s)
1–11	103	Lindholm '685 ³ , Lindholm '913 ⁴

³ U.S. Patent No. 7,038,685 to Lindholm et al. issued May 2, 2006 (Ex. 1005).

⁴ U.S. Patent No. 7,015,913 to Lindholm et al. issued Mar. 21, 2006 (Ex. 1006).

Claim(s) Challenged	35 U.S.C. §	Reference(s)
1–11	103	Amanatides ⁵ , Kohn ⁶
1–11	103	Selzer ⁷ , Fiske ⁸

VII. LEVEL OF ORDINARY SKILL IN THE ART

Petitioner describes a person of ordinary skill (“ordinarily skilled artisan” or “POSITA”) as having “at least a four-year degree in electrical engineering, computer engineering, computer science, or a related field and two years relevant experience in the graphics processing field including developing, designing, or programming hardware for graphics processing units.” Pet. 8. In the Realtek IPR Patent Owner does not comment on the level of ordinary skill. *See* Dec. to Inst. 13.

The level of ordinary skill in the art usually is evidenced by the references themselves. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001); *In re GPAC Inc.*, 57 F.3d 1573, 1579 (Fed. Cir. 1995); *In re Oelrich*, 579 F.2d 86, 91 (CCPA 1978). As Petitioner’s description of a person of ordinary skill is consistent the subject matter of the challenged

⁵ John Amanatides and Edward Szurkowski, *A Simple, Flexible, Parallel Graphics Architecture*, Proceedings of Graphics Interface at 155–160 (Canadian Information Processing Society 1993) (Ex. 1007).

⁶ Les Kohn and Neal Margulis, *Introducing the Intel i860 64-bit Microprocessor*, IEEE, Volume 9, Issue 4, pages 15–30, August 1989 (Ex. 1008).

⁷ Harald Selzer, *Dynamic Load Balancing within a High Performance Graphics System*, In Proceedings of Rendering, Visualization and Rasterization Hardware (Eurographics' 91 Workshop) at 37–53 (Springer-Verlag 1993) published in 1993 (“Selzer”) (Ex. 1009).

⁸ Stuart Fiske and William J. Dally, *Thread prioritization: A Thread Scheduling Mechanism for Multiple-Context Parallel Processors*, Proceedings of First Symposium on High-Performance Computer Architecture, 1995 at 210–221 (IEEE 1995) (Ex. 1010).

claims and the references, we apply Petitioner’s description for purposes of this Decision.

VIII. CLAIM CONSTRUCTION

A. *Claim Construction Principles*

We interpret claim terms using “the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b).” 37 C.F.R. § 42.100(b) (2019). In this context, claim terms “are generally given their ordinary and customary meaning” as understood by a person of ordinary skill in the art in question at the time of the invention. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (citing *Vitronics*, 90 F.3d at 1582) (en banc). “In determining the meaning of the disputed claim limitation, we look principally to the intrinsic evidence of record, examining the claim language itself, the written description, and the prosecution history, if in evidence.” *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 469 F.3d 1005, 1014 (Fed. Cir. 2006) (citing *Phillips*, 415 F.3d at 1312–17). Extrinsic evidence is “less significant than the intrinsic record in determining ‘the legally operative meaning of claim language.’” *Phillips*, 415 F.3d at 1317 (citing *C.R. Bard, Inc. v. U.S. Surgical Corp.*, 388 F.3d 858, 862 (Fed. Cir. 2004)).

Any special definition for a claim term must be set forth in the specification with reasonable clarity, deliberateness, and precision. *In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

We construe only those claim terms that require analysis to determine whether to institute inter partes review. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) (“The Board is required to construe ‘only those terms . . . that are in controversy, and only to the extent

necessary to resolve the controversy.’” (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

B. Realtek IPR Constructions

In the Realtek IPR, neither party proposed a special definition or explicit construction of any claim term. Underlying Patent Owner’s arguments, however, was a construction of the term “unified shader.” Noting that the limiting term “unified shader” appears in the preamble of all the claims, we construed the claimed “unified shader” to mean a shader that is configured to perform both vertex and pixel operations. Ex. 1001, 3:10–12. Realtek IPR Dec. to Inst. 15–17. As we did not perceive a need to construe other terms, we applied the plain and ordinary meaning to the remaining claim language. *Id.* at 17. We apply the same reasoning in this Decision and do not repeat the analysis here.

IX. ANALYSIS

A. Legal Principles

“In an [inter partes review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (requiring inter partes review petitions to identify “with particularity . . . the evidence that supports the grounds for the challenge to each claim”)). This burden of persuasion never shifts to patent owner. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (discussing the burden of proof in inter partes review).

The question of obviousness is resolved on the basis of underlying factual determinations including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art;

(3) the level of ordinary skill in the art; and (4) objective evidence of nonobviousness. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

Additionally, the obviousness inquiry typically requires an analysis of “whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue.” *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 418 (2007) (citing *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (requiring “articulated reasoning with some rational underpinning to support the legal conclusion of obviousness”)); see *In re Warsaw Orthopedic, Inc.*, 832 F.3d 1327, 1333 (Fed. Cir. 2016) (citing *DyStar Textilfarben GmbH & Co. Deutschland KG v. C. H. Patrick Co.*, 464 F.3d 1356, 1360 (Fed. Cir. 2006)).

An obviousness analysis “need not seek out precise teachings directed to the specific subject matter of the challenged claim, for a court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ.” *KSR*, 550 U.S. at 418; accord *In re Translogic Tech., Inc.*, 504 F.3d 1249, 1259 (Fed. Cir. 2007). Petitioner cannot satisfy its burden of proving obviousness by employing “mere conclusory statements.” *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1380 (Fed. Cir. 2016). Instead, Petitioner must articulate a reason why a person of ordinary skill in the art would have combined the prior art references. *In re NuVasive*, 842 F.3d 1376, 1382 (Fed. Cir. 2016).

A reason to combine or modify the prior art may be found explicitly or implicitly in market forces; design incentives; the “interrelated teachings of multiple patents”; “any need or problem known in the field of endeavor at the time of invention and addressed by the patent”; and the background knowledge, creativity, and common sense of the person of ordinary skill.

Perfect Web Techs., Inc. v. InfoUSA, Inc., 587 F.3d 1324, 1328–29 (Fed. Cir. 2009) (quoting *KSR*, 550 U.S. at 418–21).

In determining whether a claim is obvious in light of the prior art, when in evidence, we consider any relevant objective evidence of non-obviousness. *See Graham*, 383 U.S. at 17. Notwithstanding what the teachings of the prior art would have suggested to one of ordinary skill in the art at the time of the invention, the totality of the evidence submitted, including objective evidence of non-obviousness, may lead to a conclusion that the challenged claims would not have been obvious to one of ordinary skill. *In re Piasecki*, 745 F.2d 1468, 1471–72 (Fed. Cir. 1984). At this stage of the proceeding Patent Owner does not present evidence of such objective considerations.

We analyze the asserted grounds of unpatentability in accordance with these principles to determine whether Petitioner has met its burden to establish a reasonable likelihood of success at trial.

B. Petitioner’s Challenge to Claims 1–11 As Obvious Over Lindholm ’685 and Lindholm ’913 (collectively, “the Lindholm References”)

1. *Lindholm ’685 (Ex. 1005)*

Lindholm ’685 discloses a graphics processor that includes a multithreaded processing unit. Ex. 1005, 1:44–45.

The multithreaded processing unit includes a thread control unit configured to store pointers to program instructions associated with threads, each thread processing a sample type of vertex, pixel, or primitive. The multithreaded processing unit also includes at least one programmable computation unit configured to process data under control of the program instructions.

Id. at 1:45–51. Figure 2 of Lindholm ’685 is reproduced below.

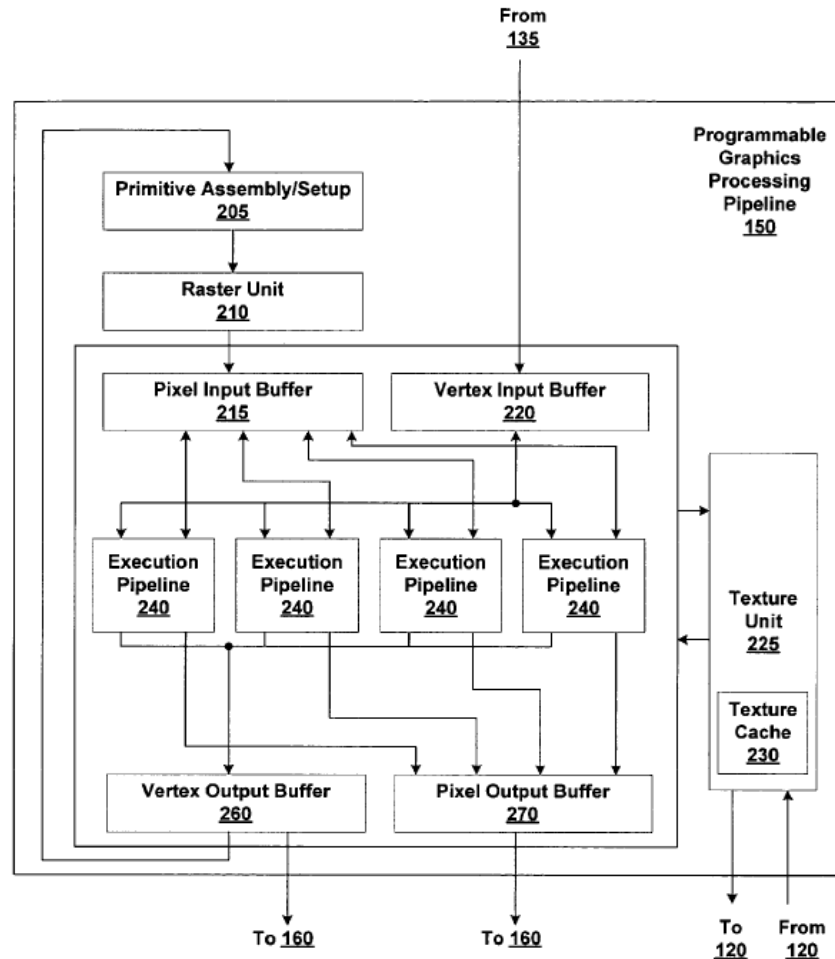


FIG. 2

Ex. 1005, Fig. 2. Figure 2 illustrates that Lindholm '685 includes execution pipelines 240, each containing a multithreaded processing unit that accepts and processes vertex and pixel samples from vertex and pixel input buffers 210, 220, respectively, when a thread is available. *Id.* at 4:31–36, 5:23–25. Figure 4 of Lindholm '685, showing details of a multithreaded processing unit, is reproduced below.

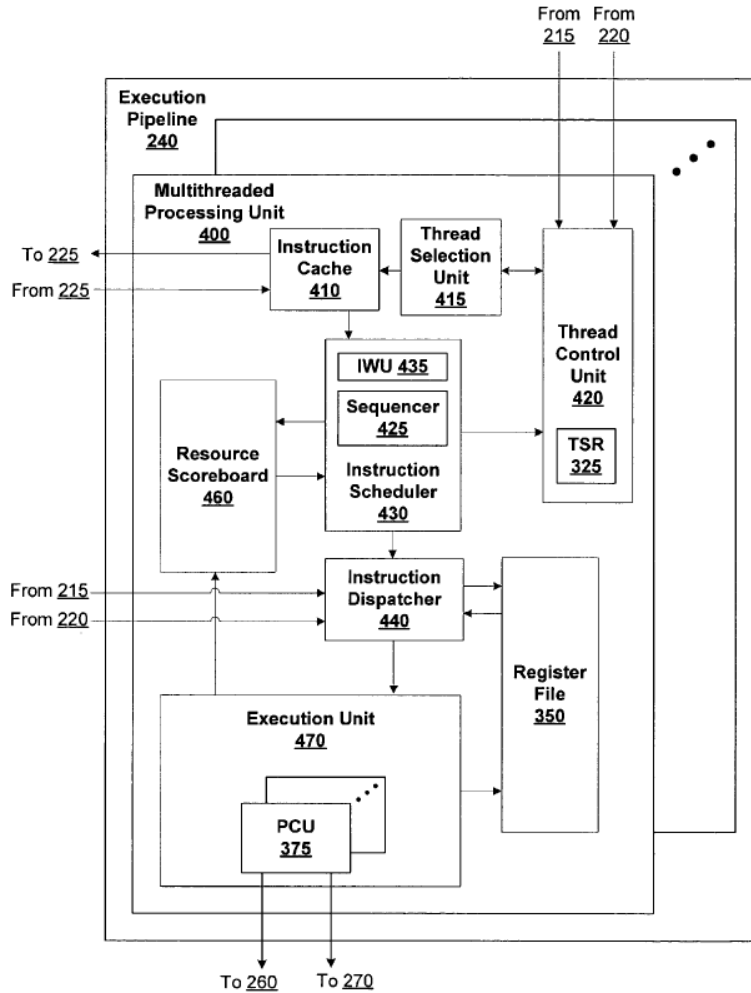


FIG. 4

Id., Fig. 4. Each thread processing unit can process vertex or pixel data—Instruction Dispatcher 440 gathers source data from pixel input buffer 215 and vertex input buffer 220 or Register file 350 specified in an instruction and outputs the instruction and source data to Execution unit 470. *Id.* at 9:34–38. When program instructions associated with a thread have completed execution, the storage resources allocated to retain intermediate data during thread execution become available to another thread. *Id.* at 9:57–63.

2. *Lindholm '913 (Ex. 1006)*

Lindholm '913 discloses a “programmable graphics processor that supports processing of graphics data elements in an order independent from the order in which the graphics data elements are received by the programmable graphics pipeline within the programmable graphics processor.” Ex. 1006, 1:39–44. The graphics processor includes at least one multithreaded processing unit that receives samples in a first order to be processed by program instructions associated with at least one thread. *Id.* at 1:48–52. Each multithreaded processing unit includes a scheduler. *Id.* at 1:53. The scheduler is configured to receive program instructions, determine the availability of source data, and schedule instructions for execution in a second order that is independent of the first order. *Id.* at 1:54–56. Each multithreaded processing unit includes a resource tracking unit that tracks the availability of source data, and a dispatcher configured to output program instructions in the second order to be executed by the multithreaded processing unit. *Id.* at 1:56–61. In one embodiment, when first source data required to process a program instruction associated with a first thread is not available, and second source data required to process a program instruction associated with a second thread are available, the program instructions associated with the second thread are dispatched prior to dispatching program instructions for the first thread. *Id.* at 2:7–22.

3. *Reasons to Combine the Teachings of Lindholm '685 and Lindholm '913*

Noting that Lindholm '685 and Lindholm '913 are both directed to a multithreaded shader and much of their Specifications and figures are the same, in the Realtek IPR Petitioner contends that a person of ordinary skill would have recognized the Lindholm references disclose different details of

the same unified shader architecture. Realtek IPR Pet. 21–22. In the Realtek IPR, Petitioner also states that an ordinarily skilled artisan would have looked to the additional disclosure in Lindholm ’913 for additional details about scheduling threads for execution independent of other threads. *Id.* at 22. In the Realtek IPR, we agreed that a person of ordinary skill would have had reason to combine the teachings of the Lindholm references. We apply the same reasoning to this Decision do not repeat that analysis here.

4. *Analysis in the Realtek IPR*

In the Realtek IPR we were persuaded that Patent Owner antedated the Lindholm references and therefore they are not prior art. Therefore, in this proceeding we are not persuaded that Petitioner will succeed on its challenge based on the Lindholm references. Realtek IPR Dec. to Inst. 23–41. We apply the same reasoning in this Decision and do not repeat our analysis here.

In the Realtek IPR, we also noted that if the Lindholm references were considered applicable prior art, we are persuaded that Petitioner has cited sufficient evidence to demonstrate that a person of ordinary skill would have been motivated to combine the teachings of Lindholm ’685 and Lindholm ’913, and that in combination, the references would have taught, or at least suggested, all the limitations of claims 1–11 to a person of ordinary skill in the art. We apply the same reasoning to this Decisions and do not repeat that analysis here. *Id.*

C. *Petitioner’s Challenge to Claims 1–11 As Obvious Over Amanatides and Kohn*

1. *Amanatides (Ex. 1007)*

Amanatides discloses a graphics architecture in which a host sends graphics primitives to a graphics pipeline front end. Ex. 1007, 156, Fig. 1.

The front end performs geometric (vertex) operations (G) using floating-point ALUs. *Id.* Transformed/shaded primitives are sent to a back end that performs rasterization and visibility operations (R). *Id.* Amanatides proposes an architecture, as shown in Figure 3 reproduced below.

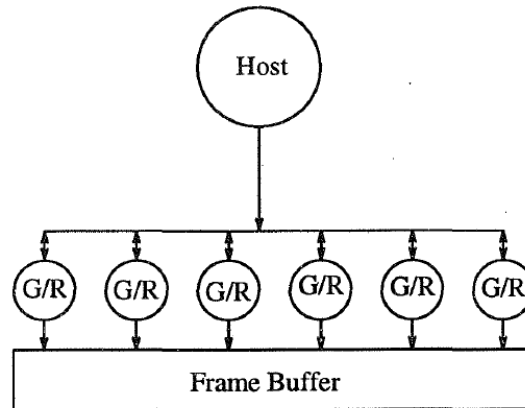


Figure 3

Ex. 1007, Fig. 3. In Amanatides' architecture, the functions of front ends (G) and back ends (R) are collapsed onto the same processors (G/R). *Id.* at 157. Each processor performs front end tasks, e.g., transforming/clipping/lighting/perspective operations, for the primitives and broadcasts the results to the other processors to perform packet and rasterization operations. *Id.* According to Amanatides, the design is "auto load balancing," such that, if there is a lot of texture mapping, the processors spend most of their time on that, but if most of the primitives are simple, the processors spend the majority of their time on the front end computation. *Id.* Amanatides notes that the approach employs powerful general purpose processors, instead of special purpose graphics hardware and, because the hardware pipeline is not optimized for a particular algorithm, capabilities can be changed and enhancements can be made freely. *Id.*

Figure 4, reproduced below, is a general outline of the design proposed by Amanatides, as implemented with Intel i860 general purpose microprocessors optimized for graphics. *Id.* at 158.

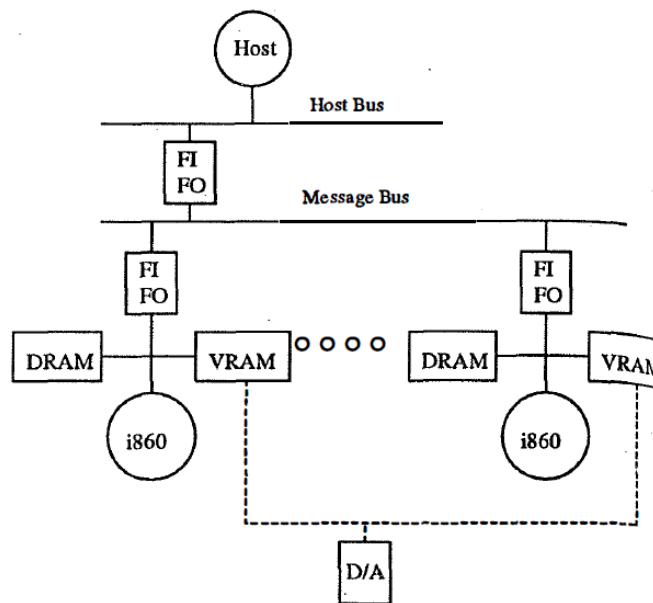


Figure 4

Id. at Fig. 4. In Figure 4, processors are connected to each other and the host via a bidirectional FIFO connected to a message bus that handles both high and low priority packets. *Id.* at 159. The FIFOs help keep each processor busy and help perform the necessary multicasting without processor intervention. *Id.* Each processor waits for packets in its input FIFO. *Id.* Packets containing polygons can be geometry messages (GM) and rendering messages (RM). *Id.* GMs come from the host—the polygons are transformed, clipped, shaded, and then sent out as RMs via an output FIFO to the other processors. *Id.* When a processor receives an RM, the processor scan converts the polygon and draws it in its frame buffer. *Id.*

GMs are treated as low priority packets that cannot access the bus if high priority packets, i.e., RMs, are waiting. *Id.* A worst case scenario occurs when every input queue is full of GMs. *Id.* As a processor turns one

of these GMs into an RM, it shuts down further introduction of GMs from the host. *Id.* at 160. As GMs turn into RMs, they wait in output FIFOs. *Id.* Whenever space appears at any input FIFO, a high priority RM is delivered to it. Eventually all the GMs are turned into RMs and delivered to FIFOs. *Id.* When all the output FIFOs are empty and the processors are working on the remaining RMs in their input FIFOs, the host can begin to deliver GMs again. *Id.* Eventually, oscillating between working on GMs and RMs, each processor is doing useful work. *Id.*

2. *Kohn (Ex. 1008)*

Kohn discloses that the i860 CPU microprocessor, a 64-bit Reduced Instruction Set Computer (RISC) has chip area divided evenly between blocks for integer operations, floating point operations, and instruction and data cache memories. Ex. 1008, 15. The i860 contains separate register files for integer and floating point units to support parallel execution. *Id.* at 16. In addition, there are six control registers and four special purpose registers. *Id.* The i860 operates on standard integer and floating point data, as well as pixel data formats for graphics operations. *Id.* The floating point hardware performs transformation calculations and advanced lighting calculations required for 3D graphics. *Id.* at 24. Kohn also describes the basis of graphics processing, noting that in 3D graphics, polygons represent the points on the surface of a solid object, that the graphics unit calculates only the vertices of the polygons, that the unit knows the colors of the polygons and must calculate the points, i.e., the pixels, between the vertices and that intensity interpolation provided smoother linear changes in pixel intensity and color between vertices than flat shading of the polygons. *Id.*

3. *Reasons To Combine The Teachings of Amanatides and Kohn*

In the Realtek IPR, Petitioner states that a person of ordinary skill in the art would have had reason to combine Amanatides with Kohn because both references use the Intel i860 processor and both references describe processing graphics with that same processor. Realtek IPR Pet. 44 (citing Ex. 1003, Pfister Decl. ¶ 165). Petitioner notes that Amanatides states “[a]fter exploring several DSP microprocessors, we decided on the Intel i860. It is a powerful general-purpose microprocessor optimized for graphics.” *Id.* at 45 (citing Ex. 1007, 4). Petitioner also notes that Kohn states that the designers of the i860 processor “recognized a synergistic opportunity to incorporate a 3D graphics unit that supports interactive visualization of results.” *Id.* (citing Ex. 1008, 15).

In the Realtek IPR we agreed with Petitioner that an ordinarily skilled artisan would have recognized that the i860 processor would be useful to implement the graphics architecture described in Amanatides and would have had reason to combine the teachings of the references. Realtek IPR Dec. to Inst. 44–45. We apply the same reasoning in this Decision and do not repeat the analysis here.

4. *Analysis in the Realtek IPR*

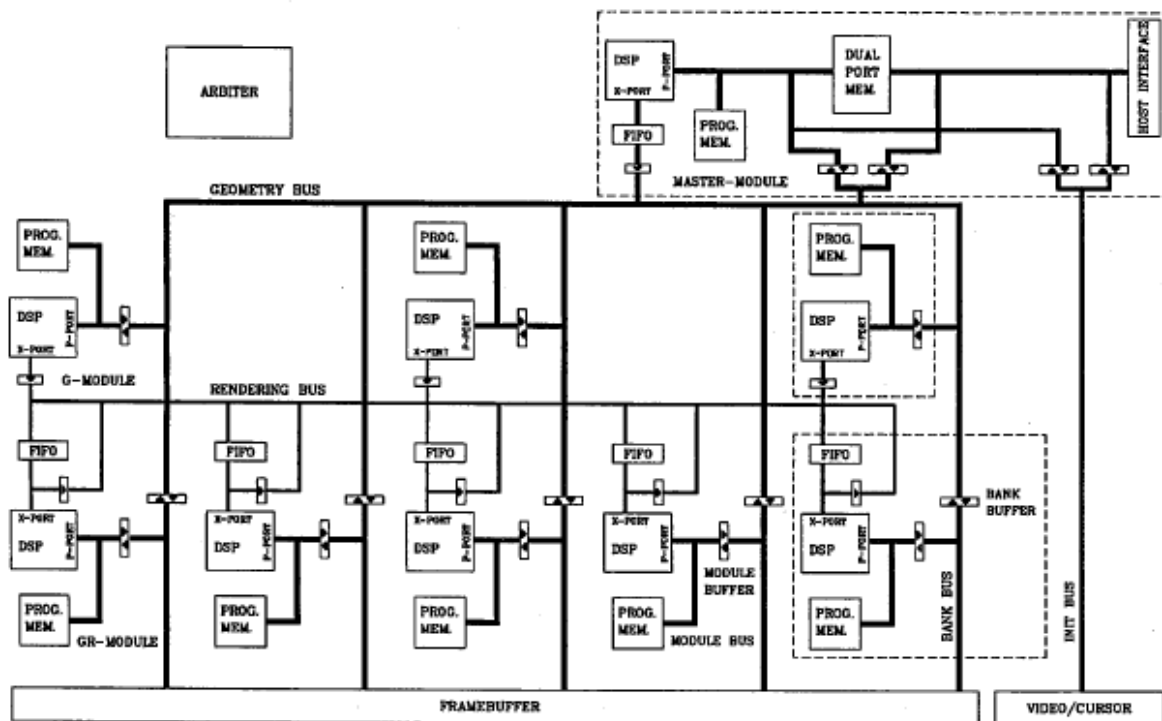
Having reviewed the arguments and evidence of record, in the Realtek IPR were persuaded Petitioner has demonstrated that a person of ordinary skill would have had reason to combine the teachings of Amanatides and Kohn, and that in combination, Amanatides and Kohn would have taught or suggested these limitations of claims 1 and 3–11 to an ordinarily skilled artisan. As the Petition in this proceeding advances the same arguments and relies on the same evidence as that considered in the Realtek IPR, we do not

repeat our analysis here and determine that Petitioner has demonstrated a reasonable likelihood it will succeed on its challenges based on Amanatides and Kohn. Realtek IPR Dec. to Inst. 45–58.

D. Petitioner’s Challenge to Claims 1–11 As Obvious Over Selzer and Fiske

1. *Selzer (Ex. 1009)*

Selzer discloses a graphics processor with an arbitrary number of high-level programmable processors able to perform both rendering tasks and geometry calculations operating in parallel. Ex. 1009, 37, 39. An example digital signal processor (DSP) has an on-chip cache, fast instruction cycle, floating point unit, and two independent parallel bus interfaces. *Id.* at 40. Figure 4.4 of Selzer, reproduced below, is a block diagram of Selzer’s system.



Id. at Fig. 4.4. Architecturally, Selzer’s system is arranged as a master module, a geometry module, and a rendering module. The system is

organized as a pipeline with additional parallelism on a functional level using multiple geometry and rendering modules. *Id.* at 43, Fig. 4.4 (showing a master module, three geometry modules (G), and five rendering modules (G-R)). All modules are connected to a geometry bus that acts as a system bus for transferring system, graphics, and update data. *Id.* A rendering bus is designed to convey only rendering primitives to data FIFOs in rendering modules. *Id.*

The master module communicates with a host processor via a dual port memory, allowing data to be transferred and processed in parallel. Ex. 1009, 41. All system resources are accessible by the master. *Id.* at 43. The master module traverses the graphics data structure and feeds graphics data, e.g., high order primitives like splines, polygons, meshes, or triangles, into the data FIFO for delivery to appropriate processors. *Id.* If a geometry module has finished the last task, it accesses the geometry bus and fetches the next primitive or task automatically. *Id.*

A graphics module performs transformation, clipping, polygon and patch subdivision, normal interpolation, and renormalization and lighting operations; the graphics module delivers the processed graphic primitives to the rendering module data FIFOs. *Id.* A rendering module fetches data from its data FIFO, performs rendering calculations, e.g., shading and scan conversion, and conveys the calculated pixel values to the frame buffer. *Id.*

The architecture achieves peak performance when all rendering processor power is exploited for rendering calculations and the rendering processors are working continuously. Ex. 1009, 48. Recognizing that frequently used features, e.g., small triangles or short vectors parallel to the x and y axis, can be rendered quickly, but that geometric calculations for interpolating shading methods cannot be accelerated because they are

independent of the resulting size of the primitive, Selzer's system uses "dynamic load balancing" or "task switching" to speed up geometry calculations. *Id.* at 45–46. The structure of Selzer's rendering module is similar to that of a geometry module, allowing a rendering module to act as a geometry module by fetching graphics data from the master data FIFO and delivering processed data to the FIFOs of an appropriate rendering module. *Id.* After rendering an object, if there is no rendering data in a rendering processor's input buffer, rather than remain idle, the processor switches to geometry processing by requesting an unprocessed geometry object and continuing geometry calculations. *Id.* at 47.

2. *Fiske (Ex. 1010)*

Fiske discloses a parallel processor system using multiple context processors that provide multiple register sets, including multiple instruction pointers, that allow multiple threads to be loaded and ready to run at the same time. Ex. 1010, 210. When a currently executing thread misses in the cache or fails a synchronization test, the processor can begin executing one of the other threads loaded in one of the other hardware contexts. *Id.* Fiske refers to a "context switch" as occurring when the processor switches from one loaded thread to executing another loaded thread, and a "thread swap" as involving swapping a loaded thread with an unloaded thread from a software queue. *Id.*

Figure 1 of Fiske, reproduced below, illustrates a multiple context processor with N contexts.

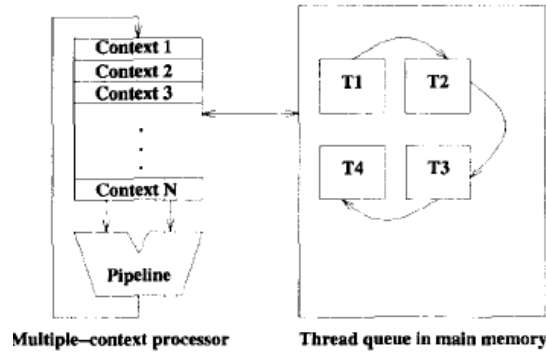


Figure 1: Multiple-context processor with N contexts.

Ex. 1010, Fig. 1. Fiske discloses allowing rapid context switching between several threads as a means of tolerating communication and synchronization latencies. *Id.* at 210. Fiske addresses scheduling such threads, i.e., first determining which threads should have their state loaded in the multiple contexts, and second, deciding which loaded thread to execute at any given time. *Id.* Fiske recognizes that if critical threads are loaded along with non-critical threads and the scheduler treats all loaded threads equally, runtime performance can suffer. *Id.* at 217. Fiske addresses this issue with thread prioritization in which each thread's priority indicates the importance of the thread to the overall problem and changes dynamically. *Id.* at 216–217. First, at each node a software scheduler chooses the highest priority threads as the loaded threads, with threads of equal priority scheduled in a round-robin fashion. *Id.* at 217. Second, at a context switch, the hardware scheduler chooses the loaded thread with the highest priority as the next thread; but if a thread is waiting for a memory reference to be satisfied, it is stalled and is not considered for scheduling until the memory reference is satisfied. *Id.*

3. *Reasons to Combine*

Petitioner contends that a person of ordinary skill would have had reason to combine the flexible graphics processing teachings of Selzer and the multithreaded architecture of Fiske because they both address load balancing within a processing system. Pet. 64. Petitioner notes that Fiske incorporates multiple hardware thread contexts to allow for rapid switching among threads, Selzer utilizes a scalable approach to parallelism by multiplying geometry and rendering modules. *Id.* (citing Ex. 1003, Pfister Decl. ¶¶ 221–226). Noting that Selzer explains how three geometry modules and five rendering modules can be used to switch from performing pixel tasks to performing vertex tasks, Petitioner states that Fiske condenses the approach into a single processing module that itself includes multiple thread contexts. *Id.* at 64–65. According to Petitioner, a person of ordinary skill would have recognized that implementing Fiske’s multithreaded processing architecture as the rendering modules of Selzer would have resulted in performing the same operations as Selzer’s rendering models but would save chip space. *Id.* at 65 (citing Ex. 1003, Pfister Decl. ¶ 223). Petitioner also notes that an ordinarily skilled artisan would recognize that both Selzer and Fiske are directed to dynamic load balancing and the scheduling approach of either reference could be utilized in their combination. *Id.* at 65.

In the Realtek IPR, Patent Owner did not argue explicitly that a person of ordinary skill would not have had reason to combine the teachings of Selzer and Fiske. Based on the arguments and evidence currently of record, in the Realtek IPR we were persuaded that Petitioner has demonstrated a person of ordinary skill would have had reason to combine the teachings of Selzer and Fiske. Realtek IPR Dec. to Inst. 62–63. We

apply the same reasoning on this Decision and do not repeat the analysis here.

4. *Analysis in the Realtek IPR*

In the Realtek IPR, we determined that Petitioner had demonstrated a reasonable likelihood it will succeed on its challenge to claim 2 as obvious over combined teachings of Selzer and Fiske. Realtek IPR Dec. to Inst. 62–72. We also determined that Petitioner had not demonstrated a reasonable likelihood of success on its challenge to claims 1 and 3–11 on this ground. *Id.* We apply the same reasoning in this Decision and do not repeat the analysis here.

X. CONCLUSION

For the reasons discussed above, we determine that Petitioner has demonstrated a reasonable likelihood that it will succeed on the following challenges to patentability:

Claims 1–11 as obvious over Amanatides and Kohn and claim 2 as obvious over Selzer and Fiske.

XI. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that, pursuant to 35 U.S.C. § 314(a) an *inter partes* review of the '454 patent is hereby instituted, commencing on the entry date of this Order, and pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial;

FURTHER ORDERED that the trial is authorized on all grounds set forth in the Petition.

IPR2024-00366
Patent 8,760,454 B2

FOR PETITIONER:

John Schnurer
Yun (Louise) Lu
Kevin J. Patari
Kyle R. Canavera
PERKINS COIE LLP
PerkinsServiceTCL-AMDIPRs@perkinscoie.com

FOR PATENT OWNER:

William A. Meunier
Michael T. Renaud
Adam S. Rizk
Reza Dokhanchy
MINTZ, LEVIN, COHN, FERRIS,
GLOVSKY AND POPEO, P.C.
WAMeunier@mintz.com
MTRenaud@mintz.com
ARizk@mintz.com
RDokhanchy@mintz.com