

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

REALTEK SEMICONDUCTOR CORP.,

Petitioner

v.

ATI TECHNOLOGIES ULC

Patent Owner

Case No.: IPR2023-00688

Patent 11,184,628

**DECLARATION OF DR. DINESH MANOCHA IN SUPPORT OF
PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO.
11,184,628**

TABLE OF CONTENTS

	Page
I. INTRODUCTION	4
II. QUALIFICATIONS AND PERSONAL EXPERIENCE.....	5
III. legal principles.....	10
IV. OVERVIEW OF THE '628 PATENT	16
A. Technology Background	16
B. File History.....	23
V. PERSON OF ORDINARY SKILL IN THE ART	25
VI. CLAIM CONSTRUCTION	25
A. “unpacked”	25
VII. SUMMARY OF THE PRIOR ART.....	26
A. Sorgard	26
B. FXT1.....	30
C. Reshetov	35
VIII. GROUND 1: SORGARD AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 7-12.....	36
A. Independent Claim 7	36
B. Dependent Claim 8.....	49
C. Dependent Claim 9.....	51
D. Dependent Claim 10.....	53
E. Dependent Claim 11.....	54
F. Dependent Claim 12.....	55
IX. GROUND 2: FXT1 AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 7- 12	58
A. Independent Claim 7	58
B. Dependent Claim 8.....	71
C. Dependent Claim 9.....	72

TABLE OF CONTENTS
(continued)

	Page
D. Dependent Claim 10.....	73
E. Dependent Claim 11.....	74
F. Dependent Claim 12.....	76
X. GROUND 3: FXTI IN VIEW OF RESHETOV AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 12	79
A. Dependent Claim 12.....	79
XI. DECLARATION.....	81

I, Dr. Dinesh Manocha, declare as follows:

I. INTRODUCTION

1. I have been engaged by Realtek Semiconductor Corp. (“Petitioner”) to provide my expert opinion on invalidity related issues in connection with Petitioner’s Petition for *Inter Partes* Review in IPR2023-00688 (“Petition”) before the Patent Trial and Appeal Board (“Board”). I was asked to provide my opinions in this expert declaration (“Declaration”). I intend to provide testimony regarding my opinions if called to testify at a deposition.

2. This Declaration addresses my opinions related to the validity of the Asserted Claims of U.S. Patent Nos. 11,184,628 (the “’628 patent”) (Exhibit 1001). I understand the challenged claims to be claims 7-12 of the ’628 patent. Accordingly, my Declaration addresses these claims.

3. I am being compensated for my time working in connection with this Petition. However, my compensation is not affected by or contingent upon the substance of my opinions or on the ultimate outcome of this Petition, and I have no other interest in this Petition.

4. The opinions expressed in this Declaration are my own and are based on my personal knowledge, my education, experience, and training, and on my understanding of the information and documents referenced in this Declaration. If called as an expert in this case, I anticipate that my testimony will concern the

matters addressed in this Declaration. My testimony may be affected by the positions that ATI Technologies ULC (“Patent Owner”) and/or its expert takes on these and other related topics after this Declaration is issued, either in the form of expert declarations or deposition testimony, or any ruling that the Board may make, and therefore I reserve the right to provide testimony regarding those issues.

II. QUALIFICATIONS AND PERSONAL EXPERIENCE

5. I received my B.E. in Computer Science and Engineering from the Indian Institute of Technology in 1987. I received my M.S. and Ph.D. in Computer Science from the University of California at Berkeley in 1990 and 1992, respectively.

6. After receiving my doctorate degree, I joined the faculty of the Department of Computer Science at the University of North Carolina (“UNC”) at Chapel Hill as an Assistant Professor in 1992. I was promoted to Associate Professor in 1998 and to Professor in 2001. I was named as a Matthew Mason/Phi Delta Theta Distinguished Professor in 2006, a post I held until 2018. In May 2018, I joined the University of Maryland at College Park as the Paul Chrisman Iribe Chair of Computer Science and Electrical & Computer Engineering. I was also appointed the Distinguished University Professor in 2020, a post I currently hold.

7. My academic research focuses on geometric computing and computer graphics with applications to simulation and robotics. As a result of this research, I have published more than 700 papers related to computer graphics, robotics, computational geometry, databases, multimedia, high performance computing and symbolic computation. I have received 19 best paper and time of test awards. Between UNC Chapel Hill and University of Maryland, I have supervised more than 43 Ph.D. students over the last 30 years and am currently supervising more than 20 Ph.D. students. During the same time, I have also supervised more than 45 M.S. students and 10 B.S. students.

8. I have been leading a large research group that has developed algorithms and systems for interactive computer graphics that utilize the latest capabilities of graphics processors and graphics systems. My research group has developed many well-known software packages for real-time rendering, texture compression, collision detection, triangulation, GPU-computing, robotics, multi-agent simulation, geometric modeling, solving algebraic systems, etc. These packages have been downloaded by more than 300,000 users worldwide and licensed to more than 60 industrial organizations including Intel, Microsoft, Disney, Ford, Kawasaki, Siemens, Philips Labs, MSC Software, Lockheed Martin, Raytheon etc. I have served as a Principal Investigator or Co-Principal

Investigator on 84 grants and obtained more than \$59M in funding to support my research at UNC Chapel Hill and University of Maryland.

9. In addition to publishing and guiding my research group, I have also interacted closely with leading industry players in these areas for three decades. My research group has actively collaborated with many industrial organizations including Amazon, ARA, AMD, Boeing, Disney, Dolby, Google, Lockheed, Intel, Microsoft, Nvidia, Oculus, SAIC, Willow Garage, Samsung, etc. My research has been supported by ARO, ARL, Boeing, DARPA, IARPA, NASA, NIH, NRL, NSF, ONR, RDECOM, and many industrial partners.

10. I was a co-founder of Impulsonic, a developer of physics-based audio simulation technologies, which was acquired by Valve Inc. in November 2016. The Phonon technology developed by Impulsonic has been widely released as Steam Audio SDK. I am also a co-founder of Inception Robotics. I am a co-inventor of 16 patents (11 granted and 5 pending), several of which have been licensed to industry.

11. I frequently teach undergraduate and graduate courses in Computer Graphics, Modeling, Rendering, and GPU Computing and Robotics. Most of my teaching and research activities require familiarity with, for example, graphics hardware (including design and features), software libraries, computer graphics

algorithms, programmable graphics, and graphics APIs such as OpenGL and DirectX and related libraries.

12. I have served as program committee member or program chair of more than 120 leading conferences, including many in computer graphics. I have given more than 140 invited or keynote talks at conferences and distinguished lectures at various institutions. I have served on the editorial board, or as a guest editor, on over eleven leading journals in computer graphics, robotics, geometry processing and scientific computing, including the ACM Transactions on Graphics, IEEE Transactions on Visualization and Computer Graphics, IEEE Computer Graphics and Applications, and Graphical Models.

13. I am a Fellow of ACM (Association for Computing Machinery), IEEE (Institute of Electrical and Electronics Engineers), AAAI (Association for the Advancement of Artificial Intelligence), and AAAS (American Association for the Advancement of Science), and a member of ACM SIGGRAPH Academy. I have received the Distinguished Alumni Award from Indian Institute of Technology, Delhi, the Bézier Award from the Solid Modeling Association, and the Distinguished Career in Computer Science Award from Washington Academy of Sciences. I have also won numerous awards throughout my career, including NSF Career Award, ONR Young Investigator Award, Sloan Fellowship, IBM

Fellowship, SIGMOD IndySort Winner, Honda Research Award, and UNC Hettleman Prize.

14. I have conducted research and teaching courses on topics related to computer graphics and GPU computing for more than two decades. My research group has conducted extensive research on real-time rendering on GPUs, explored parallel hardware and software capabilities of GPUs for various applications related to computer graphics, database computation, machine learning, robotics, and geometric computing. My research has also designed many fast and new methods for texture compression/decompression as well as utilized the massive parallelism of GPU architectures. I have also supervised the research of a large number of undergraduate and graduate students, who have taken senior positions in the graphics and hardware industry. I have also interacted (and consulted) with leading researchers and developers at industry for three decades, including senior personnel at Silicon Graphics, Intel, AMD, NVidia, Sony, Microsoft, ATI, ARM, etc. My research group at the University of North Carolina was among the first leading research groups that exploited the parallelism of GPUs (from 1999 onwards) in terms of utilizing a large number of concurrent threads and memory architectures for real-time rendering, ray tracing, physics simulation, geometric computing, database operations, etc. I have organized some of the very first courses and workshops related to GPU computing at ACM SIGGRAPH

Conferences from 2002 to 2007, addressing many issues related to massive parallelism on the GPUs, including exploiting the parallelism and designing libraries and algorithms to exploit the hardware capabilities. I also worked on faster and improved methods for texture compression from 2012 to 2018 and published many papers. My research group has released open-source code for texture compression (<https://gamma.umd.edu/software#texture>), which has been widely used in the gaming and graphics industry.

III. LEGAL PRINCIPLES

15. I am not a lawyer and offer no opinion of the law. Accordingly, the “Legal Principles” herein have been provided to me from counsel, but I have read and adopted it as my own. For example, I have, however, been informed by counsel of the legal principles that apply with respect to the issues for which I was asked to provide an opinion, including patent invalidity. I have applied these legal principles in formulating my opinions and arriving at my conclusions. I have also been informed that the claims of an issued patent are presumed valid, but those claims can be found invalid if it can be proven, by clear and convincing evidence, that the claims fail to comply with one or more requirements of patentability.

16. It is my understanding that a patent may claim the benefit of the filing date of an earlier application only if the earlier application provides adequate disclosure of the patent’s claims. I further understand that to satisfy the disclosure

requirement, the disclosure of the earlier filed application must describe the later claimed invention in sufficient detail that one skilled in the art, as of the earlier filing date sought, could clearly conclude that the inventor invented the claimed subject matter. I understand that while the earlier application need not describe the claimed subject matter in precisely the same terms as found in the claims at issue, the prior application must convey with reasonable clarity to a person of ordinary skill in the art that, as of the filing date sought, the inventor was in possession of the invention. It is also my understanding that claims of a patent that contain new matter that is not adequately disclosed by an earlier filed application do not receive the benefit of the earlier filing date.

17. It is my understanding that a patent claim is invalid if the differences between the patented subject matter and the prior art are such that the subject matter as a whole would have been obvious at the time of the invention to a person of ordinary skill in the art. I understand that the following factors must be evaluated to determine whether the claimed invention is obvious: (1) the scope and content of the prior art, (2) the difference or differences between each claim of the patent and the prior art, (3) the level of ordinary skill in the art at the time the invention of the patent was made, and (4) any secondary considerations or objective indicia of nonobviousness.

18. I understand that the combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results. In other words, when a claim simply arranges prior-art elements with each performing the same function it had been known to perform and yields no more than one would expect from such an arrangement, then such a combination is obvious. Moreover, when a patent claims a structure already known in the prior art that is altered by the mere substitution of one element for another known in the field, the combination is likely to be obvious unless the combination yields an unpredictable result.

19. I understand that a patentee's admissions about the scope and content of the prior art provide a factual foundation as to what a skilled artisan would have known at the time of invention. For example, by characterizing claimed elements as conventional in the specification, the patentee effectively admits that such things would be known to a person of ordinary skill in the art. I understand that I can rely on such admissions and/or knowledge in forming an opinion as to whether claims are obvious. Such reliance could include, for example, furnishing a motivation to combine, or supplying a missing claim limitation. Thus, even though a patentee's admissions are not themselves prior art references, they can establish the background knowledge possessed by a person of ordinary skill in the art.

20. I have been informed that when a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of ordinary skill in the art can implement a predictable variation, such a variation is likely unpatentable. For the same reason, if a technique has been used to improve one device, and one of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond the skill of the person of ordinary skill in the art. One question to consider is whether the improvement is more than the predictable use of prior-art elements according to their established functions.

21. I have been informed that it may often be necessary, in a validity analysis, to look to interrelated teachings of multiple patents or other publications or pieces of prior art; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by one of ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue.

22. I have been informed that a validity analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim. Rather, it is appropriate to take account of the inferences and creative steps that a person of

ordinary skill in the art would employ. I understand that a person of ordinary skill in the art is also a person of ordinary creativity.

23. I understand that a claim composed of several elements is not proved obvious merely by demonstrating that each element was, independently, known in the prior art. I understand that it can be important to identify a reason that would have prompted a person of ordinary skill in the art in the relevant field to combine the elements in the way the claimed invention does. Therefore, I am informed that, in determining whether a claimed combination is obvious, the correct question is whether the combination was obvious to a person of ordinary skill in the art at the time of the alleged invention. I am told that one of the ways in which subject matter can be proved obvious is by noting that there existed at the time of invention a known problem for which there was an obvious solution encompassed by the patent's claims. I understand that any need or problem known in the field of endeavor at the time of alleged invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.

24. I understand that one should not assume that a person of ordinary skill in the art attempting to solve a problem will be led only to those elements of prior art designed to solve the same problem. Instead, I understand that since familiar items may have obvious uses beyond their primary purposes, in many cases a

person of ordinary skill in the art would be able to fit the teachings of multiple prior art references together like pieces of a puzzle.

25. I understand that, when there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions, a person of ordinary skill in the art has good reason to pursue the known options within his or her technical grasp. If this leads to the anticipated success, then it is likely the product is not of innovation but of ordinary skill and common sense. I understand that, in such an instance, the fact that the combination was obvious to try may show that the combination is obvious.

26. I understand that, when determining whether a claimed combination is obvious, the correct analysis is not whether a person of ordinary skill in the art, writing on a blank slate, would have chosen the particular combination of elements described in the claim. Instead, I understand the correct analysis considers whether one of ordinary skill, facing the wide range of needs created by developments in the field of endeavor, would have seen a benefit to selecting the combination claimed.

27. I understand that, when determining whether a claimed invention is obvious, any “secondary considerations” of nonobviousness should also be considered. I further understand that there must be a nexus between any alleged secondary considerations and the patent claims. In particular, when the product

that practices an invention is not co-extensive with the patented invention—for example, if the patented invention is only one component of a commercially successful product—then the patentee must show a sufficient relationship or nexus between the patented invention in a product and the secondary considerations, *e.g.*, commercial success of the product. In addition, I understand that the burden of introducing evidence of secondary considerations generally is on the Patent Owner.

IV. OVERVIEW OF THE '628 PATENT

A. Technology Background

28. It was well-known at the time of the '628 patent that texture decompression is an integral component of image generating and rendering system in computer graphics. The textures are typically stored in the graphics memory. Generally speaking, decompression methods load a compressed texture, uncompress to compute the color values associated with each texel of the resulting texture image, and map those color values to the corresponding point on a geometric primitive, which could be a triangle or a curved surface. These compression and decompression methods have been integrated with graphics hardware and processors and are well known in the art, now or at the time of the '628 patent.

29. The '628 patent describes a texture decompression method that includes receiving a compressed texture block, determining a partition of pixels

used for the compressed texture block, wherein the partition defines one or more disjoint subsets of pixels, unpacking data for each subset, and then decompressing each of the one or more disjoint subsets to form an approximation of an original texture block. Textures were proposed in the field of computer graphics in the 1970s to enhance the realism and fidelity of synthesized images. Texture mapping has been supported in graphics hardware since the early 1990s for real-time or interactive rendering applications. To achieve a high fidelity, a large amount of video memory needs to be devoted to storing these textures. Furthermore, accessing texture memory at run time comes at a high performance cost. The need for ever increasing realism in interactive applications such as games drives a constant requirement for more and higher resolution art or texture assets. As a result, rendering from a compressed representation of the image has become a key technique for both reducing memory footprint and increasing texture access bandwidth, improving both size and speed metrics of an implementation.

30. In order to store large textures in the GPU memory, texture compression and decompression techniques have been widely used in graphics hardware or processors by most vendors for more than two decades. Many different formats and techniques have been proposed to compress and decompress textures. The notion of using block-based truncation coding is not new; an early paper by Delp and Mitchell was published in 1979. The size of a block together

with the size of its encoding determines the compression ratio of the format. Since the late 1990s, many compression and decompression methods have been proposed for low dynamic range (“LDR”) images as well as high dynamic range (“HDR”) images.

31. The ’628 patent includes a partial survey of the state of the art on texture compression, but missed some key references. It states that an ideal solution “would allow compression of various types of data and would have the flexibility to make the best use of available memory and bandwidth.” Ex. 1001, 2:67-3:3. However, the patent contains several incorrect statements about the art.

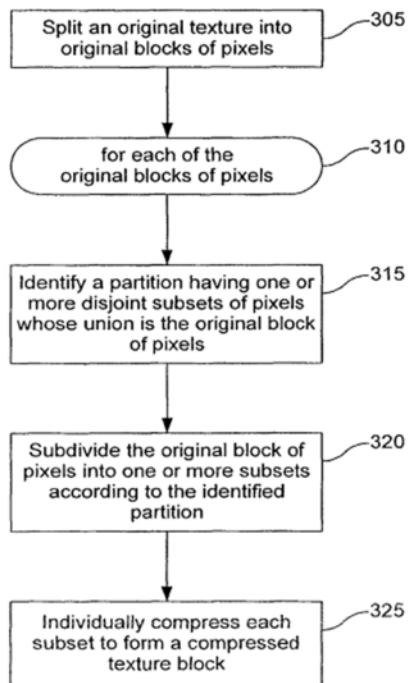
32. For example: The statement “traditional texture compression schemes target a specific type of texture content (e.g., JPEG for photographic images) and perform well within that set, but perform poorly as soon as presented with a type of image outside of the designated set” is incorrect. *Id.*, 3:3-7. Many patents and papers published before the filing of this paper provide various flexible schemes.

33. The ’628 patent incorrectly characterizes FXT 1 as saying “FXT 1 is a competing compression scheme with DXTC. It essentially extends DXTC with some additional block types that can be mixed within any given image, and also provides a 4 bits per pixel compression mode for textures with alpha. The gains in image quality over DXTC were never conclusive and FXT 1 received limited industry support.” *Id.*, 2:46-52. This is incorrect with respect all the details related

to FXT1 and how it has been cited by others. *See, e.g.,*

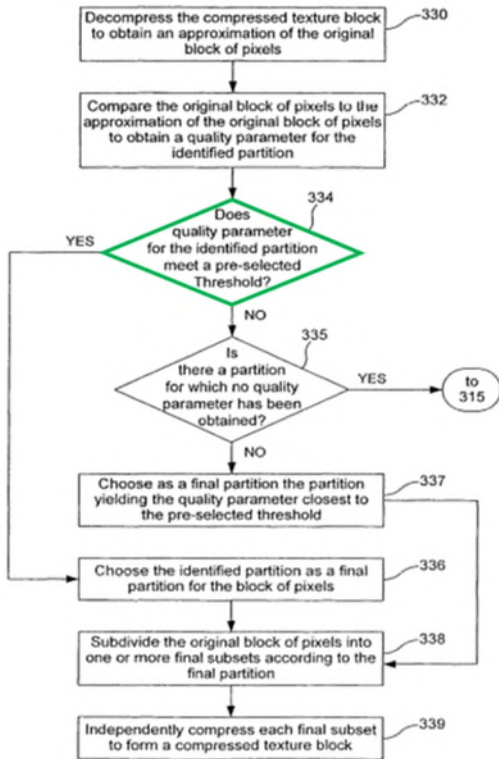
https://www.graphicon.org/html/2000/Texture/Ivanov_Kuzmin.pdf.

34. The '628 patent describes some preferred embodiment for lossy texture compression schemes, which purportedly seek a balance between texture compression ratio and image quality. *Id.*, 4:27-29. It discloses an allegedly flexible method for partitioning of texture blocks, which are used for improving the quality of texture compression in terms of image fidelity. *Id.*, 4:52-54. The supposed motivation is that such pre-compression partitioning can flexibly select partitions that are suitable to the characteristics of the texture blocks being compressed. *Id.*, 4:55-58. The patent alleges that its embodiment is a “further refinement of block-based texture compression schemes such as DXTC,” and uses “discrete blocks of pixels (e.g., 4x4 or 8x8 pixel blocks).” *Id.*, 5:66-6:4. The patent does not, however, define the word *block*. And the 4x4 or 8x8 sizes above are merely exemplary block sizes. The specification discloses subdividing or partitioning a block internally into “subsets” of different shapes and/or sizes before compression. *Id.*, 6:5-9. Each subset is compressed in a manner similar to DXTC. *Id.*, 6:9-10.



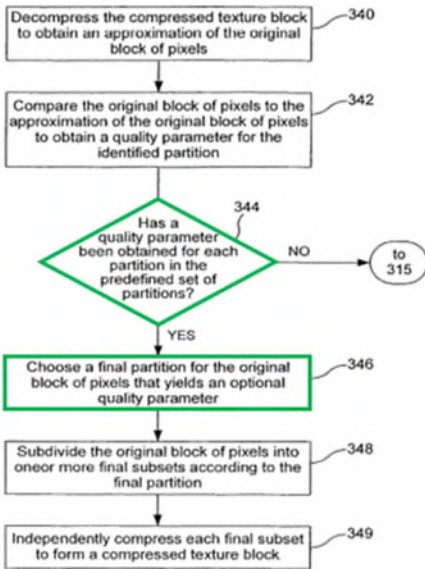
Id., Fig. 3A.

35. The '628 patent discloses that some embodiments could have several sets of partitions or subsets. *Id.*, 6:49-50. After the texels corresponding to each partition or subset have been compressed, the compressed texture block may be evaluated to refine the partition selection using a quality parameter. This quality parameter can be used to ensure that the image fidelity does not degrade significantly after applying the texture compression and decompression steps to each block or subset. In this regard, the patent shows an embodiment in which the quality parameter associated with a partition is compared with a pre-selected threshold to determine whether the partition can be selected as the final partition for compression. If not, further subdivision is performed. *Id.*, 7:66-8:8.



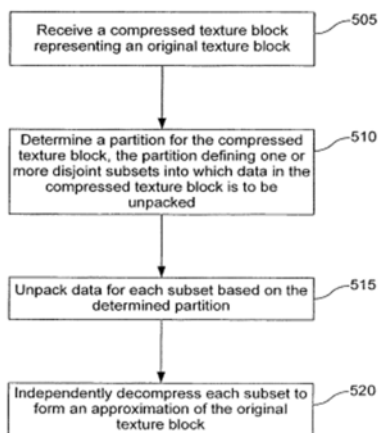
Id., Fig. 3B (annotated).

36. In an alternative embodiment, instead of comparing the quality parameter with a pre-selected threshold, the algorithm appears to iterate over all possible partitions before selecting one having an optimal quality parameter as that corresponding to the final partition. *Id.*, 8:19-29.



Id., Fig. 3C (annotated).

37. The '628 patent also discloses a decompression method, which involves working backward to recover the original texels, and the color values associated with each texel, as a POSITA would have found obvious because compression and decompression involve inverse functionality.



Id., Fig. 5. The decompression method disclosed and claimed in the '628 patent involves, *inter alia*, “determining the partition that was used for the compressed

texture, the partition defining one or more disjoint subsets into which the compressed texture block is to be unpacked,” *Id.*, 8:37-47, and then unpacking the data for each subset.

B. File History

38. Applicant filed the patent application on January 25, 2019. *See Ex. 1001, 1.* On June 11, 2020, the Examiner issued what was entitled a Non-Final Office Action rejecting claims 1-20 as unpatentable on the following grounds: (1) claims 1, 2, 6, 7, 11, 12, 16 and 17 as obvious over U.S. Patent No. 7,388,993 (Govindaswamy) in view U.S. Patent Application No. 20040126031 (Dwyer); (2) claims 3 and 14 rejected as obvious over Govindaswamy in view of Dwyer in further view of U.S. Patent No. 6,891,793 (Reshetov); (3) claims 4, 5, 14 and 15 rejected as obvious over Govindaswamy in view of Dwyer in further view of Stasidski et. al. “A New Class of Fast Shape-Adaptive Orthogonal Transforms and Their Application to Region-Based Image Compression”; (4) claims 8 and 18 over Govindaswamy in view of Dwyer in further view of U.S. Patent No. 5,488,365 (Seroussi); claims 9 and 19 rejected as obvious over Govindaswamy in view of Dwyer in further view of U.S. Patent No. 6,208,273 (Dye); and (5) claims 10 and 20 rejected as obvious over Govindaswamy in view of Dwyer in further view of Eckert et. al., Perceptual quality metrics applied to still image compression, Dec. 1998, Signal Processing 70, 177-200. *Ex. 1002, 84-96.*

39. In response to this Office Action, Applicant cancelled original claims 4-5 and 14-15, amended the other claims, and added two new claims. *Id.*, 123-36. However, on January 12, 2021, the Examiner issued a Final Office Action still rejecting all claims as obvious over combinations of Govidaswamy, Dwyer, Stasidski, Seroussi, Dye, and Eckert. *Id.*, 138-56.

40. Thus, Applicant amended the claims again and cancelled original claims 3, 7, 10, 13, 17, and 20. *Id.*, 168-181. In Applicant's accompanying remarks, Applicant argues that the cited references, used either alone or in combination, fail to teach or suggest "the claimed use of a compressed texture block including two or more disjoint subsets of data and individually decompressing each of the two or more disjoint subsets as recited in the amended dependent claims." *Id.*, 177-78.

41. On August 5, 2021, the Examiner issued a Notice of Allowance allowing the amended claims. *Id.*, 191. The Examiner noted that "the features of the explicitly claimed limitations claim 1 [sic] and similar limitations of other independents of the currently pending application only make sense when taken into account the claim as a whole, so therefore the limitations as cited/argued are not the exclusive determination of allowability, but rather the limitations as cited/argued above when taken into account with the claim language as a whole does the above limitations determine allowability." *Id.*, 192. The claims were

renumbered as claims 1-12 in the patent issued on November 23, 2021. *See id.*, 198; Ex. 1001.

V. PERSON OF ORDINARY SKILL IN THE ART

42. I understand the ALJ has adopted the following definition of a person of ordinary skill in the art (“POSITA”): “a POSITA would have a bachelor degree in electrical engineering, computer engineering, computer science, or a related field, and at least two years of experience in ‘parallel computing or graphics processing including developing, designing, or programming software or hardware for graphics processing or parallel processing units, graphics accelerators[,] or other graphics processing systems,’ with additional education substitution for experience and vice versa.” Ex. (Order No. 31) at 9 (alteration in original). That is the definition that I have applied in my analysis.

VI. CLAIM CONSTRUCTION

43. I understand that the Board gives claim terms their ordinary and customary meaning as a POSITA would understand them in light of the patent’s specification and prosecution history. I also understand that several of the claim terms of the ’628 patent have already been construed by the ITC or argued by Petitioners.

A. “unpacked”

44. Claim 7 recites that “data in the compressed texture block is to be unpacked.” The ALJ in the related ITC matter has considered the construction of

this term, finding that “unpacked” means “extracted.” Ex. 1010, 42-43. I will adopt this more limiting construction for the purposes of this declaration.

45. [Paragraph intentionally left blank]

46. [Paragraph intentionally left blank]

47. [Paragraph intentionally left blank]

48. [Paragraph intentionally left blank]

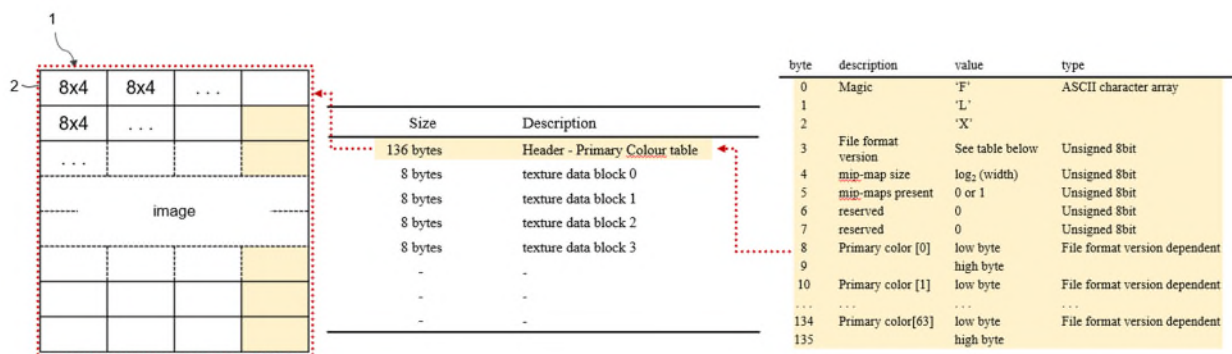
49. [Paragraph intentionally left blank]

VII. SUMMARY OF THE PRIOR ART

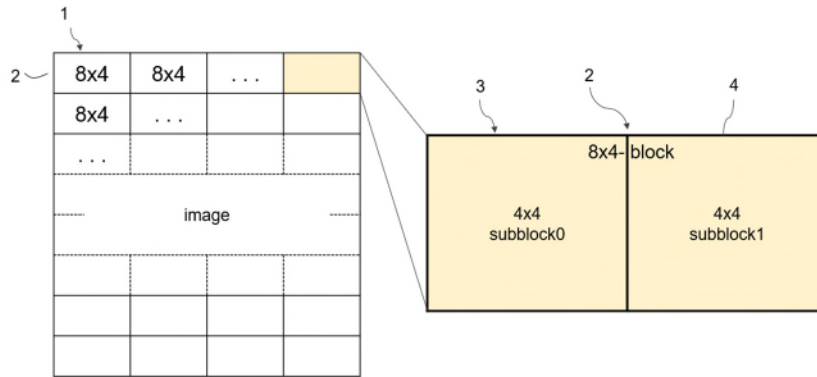
A. Sorgard

50. Sorgard claims priority to a PCT application filed on March 3, 2006. The PCT application was published as WO2006/092616 in English on September 8, 2006. Sorgard is prior art to the '628 patent under 35 U.S.C. § 102(e).

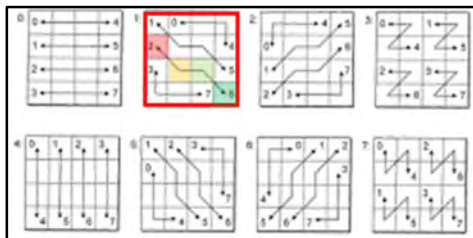
51. Sorgard discloses a texture compression/decompression scheme for graphics systems. A global color table of a suitable size, e.g., 64 colors, is used to represent an image. Ex. 1004, 6:48-7:10.



The image is partitioned into 8x4 blocks, each of which is further partitioned into two 4x4 sub-blocks. *Id.*, 4:59-5:3.

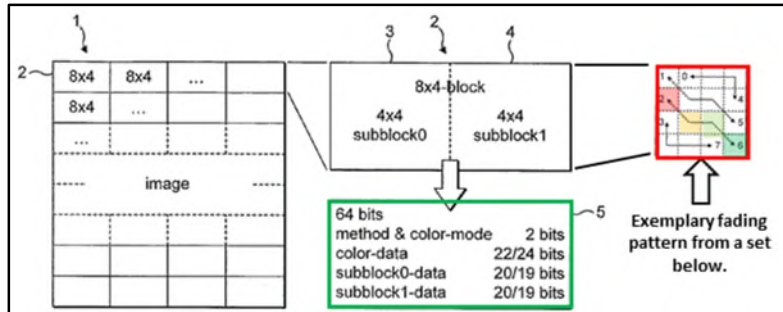


In an embodiment, each sub-block is further divided into several fading paths or interpolation paths that together form a fading pattern. The fading pattern can be selected from a set of patterns to match the image.



Ex. 1004, Figs. 1, 5 (partial and annotated).

52. Each compressed 8x4 block is represented by 64 bits (green box), which includes a set of four colors derived from the global color table used to compress/decompress the block. *Id.*, 24:61-25:4, 28:9-19, 30:9-31.



53. Each interpolation path comprises two endpoints and several interior points. The colors of the endpoints are selected from the set of four colors. The colors of the interior points are calculated from the color endpoints using interpolation. *Id.*, 30:9-31, 31:30-36, 31:52-58, 33:53-34:22.

54. Sorgard was filed on June 23, 2008 and issued on January 24, 2012. Sorgard claims priority to a Great Britain application filed on March 4, 2005 and a PCT application filed on March 3, 2006. The PCT application was published as WO2006/092616 in English on September 8, 2006. Sorgard is prior art to the '628 patent under at least 35 U.S.C. § 102(e).

55. Sorgard is related to compressing textures that are used in graphics systems. Textures are widely used to generate high-quality images for real-time rendering and have been supported in graphics hardware and processors since the early 1990s. Sorgard addressed two main problems related to the use of high-resolution textures: the storage or memory overhead of texture data can be high; moreover, accessing textures from the graphics memory can lead to performance loss due to high memory bandwidth requirement. To solve these problems,

Sorgard described new methods that can achieve a higher texture compression rate, yet maintain good image fidelity. Sorgard provided an accurate overview of the state of the art at that time (2005), in terms of use of block truncation code (BTC) method as well as good compression methods proposed by S3 Inc., which is widely known as S3TC texture compression. Sorgard also proposed a block-based data encoding approach including data that indicates how to generate a set of data values for each specific block as well as data that indicates a method that can be used to generate the data value for each texture element represented by the block.

56. Sorgard described multiple embodiments. In one of the embodiments, the patent describes a method to subdivide a set of texture data elements into a plurality of subsets of texture data elements. Each subset is encoded according to the block-based compression scheme. It describes many ways to divide a block into these subsets. In this process of subdividing the image into smaller blocks, any order of encoding of these texture blocks could be chosen (e.g., in Morton order). Furthermore, set of texture elements encoded as a single block of texture data can be further subdivided within the texture block, e.g., into texture data “sub-blocks.” In this embodiment, Sorgard supported the use of adaptive scheme to generate adaptive sub-blocks of a larger block. The resulting sub-blocks could be of different sizes. Sorgard also described methods that can be used to ensure that

the texture obtained after the decompression step satisfy some criteria in terms of image quality.

57. These subblocks may include data (e.g., some sort of endpoints) and the method used to generate the data values for texture elements. The data values during the decompression step can be computed independently for each block or sub-block. Sorgard also described multiple embodiments that describe how the data values (i.e., the colors of the texels of the decompressed texture) can be generated.

B. FXT1

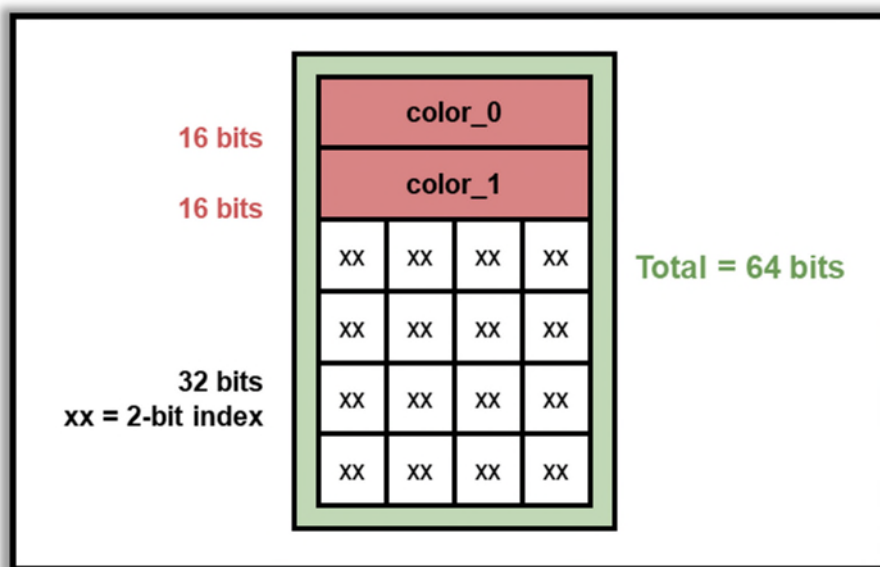
58. FXT1 is a texture compression technology publicly distributed by 3dfx Interactive, Inc. in 1999. FXT1 supported “an advanced and flexible technological foundation” for increasing demands of texture rendering. Ex. 1005, 2. *FXT1 Texture Compression Technology White Paper* (“FXT1”) was published in September 1999 and is prior art to the ’628 patent under at least 35 U.S.C. §§ 102(a), (b).

59. FXT1 discloses a texture compression technology that, like the later ’628 patent, divides an image into portions. Each portion has two color endpoints from which additional, intermediate colors are computed using interpolation. The endpoint and intermediate colors form a set of colors analogous to the color ramp of the ’628 patent. Each texel is represented using an index that

indicates which color from the set of colors is used to render the texel during decompression. FXT1 uses 4 algorithms with different portion sizes and different compression techniques to increase the quality of texture reproduction. Ex. 1005, 5.

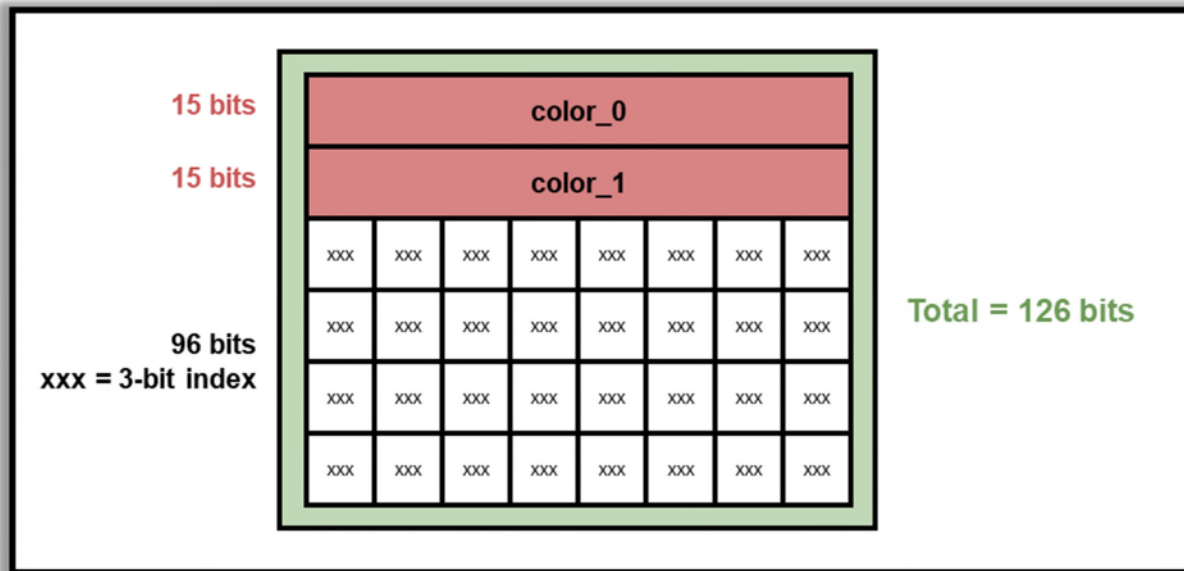
60. More specifically, during compression, an image is divided into 4x4 and/or 4x8 texel portions. *Id.* Each portion is compressed using four different compression algorithms. *Id.* The highest-quality compressed portion produced by the four algorithms is selected as the compressor output. *Id.* Two of the four algorithms are relevant to invalidity of the '628 patent: CC_MIXED and CC_HI.

61. The CC_MIXED algorithm operates on 4x4 texel portions. Each portion “has two 16-bit colors stored in an RGB 565 format.” *Id.* In other words, the three basic colors, red, green, and blue, used to generate colors on electronic devices, are respectively represented by 5, 6, and 5 bits. Two additional colors are “created by interpolating between the two RGB 565 colors.” *Id.* The four colors are stored in a lookup table. *Id.* A 2-bit index is used to select one of the four colors to render the corresponding texel. *Id.*



62. The CC_HI algorithm operates on 4x8 texel portions. Each portion “stores two 15-bit colors in an RGB 555 format.” *Id.* Five additional colors are created by interpolating between the two 15-bit colors. *Id.* These seven colors and an eighth, transparent color are stored in a lookup table. *Id.* A 3-bit index is used to select one of the eight colors to render the texel. *Id.*

63. FXT1 details its compression algorithm. *See id.*, 5. Decompression algorithm follows an inverse process of, and can be inferred from, the compression algorithm. Accordingly, what FXT1 discloses about compression applies to decompression, and FXT1 only briefly discusses its decompression processing. *See Ex. 1005, 6.*



64. During decompression, a 2-bit field stored in each portion is used to determine which of the four algorithms generated the highest-quality compressed data for each portion. For example, the 2-bit field of a first portion might have a value 0, indicating that CC_MIXED generated the compressed data for the first portion and thus is to be used to decompress the data. The 2-bit field of a second portion might have a value 1, indicating that CC_HI generated the compressed data for the second portion and thus is to be used to decompress the data. See Ex. 1005, 6.

65. FXT1 is a texture compression technology that supported an advanced and flexible technological foundation for increasing demands of texture rendering. Ex. 1005, 2. It “allow[ed] content developers to create textures at higher resolutions and use more textures in a given scene while simultaneously reducing

the memory bandwidth required for texturing.” *Id.* FXT1 was developed by 3dfx Interactive, Inc. (“3dfx”). *Id.*

66. Many contemporaneous publications cite FXT1.

67. For example, a paper in the Proceedings of International Conference Graphicon 2000 (Ivanov and Kuzmin, *Color Distribution for Compression of Textural Data*) has devoted a full subsection (Section 2.2.3) to give an overview of FXT1 and its characteristics. An October 2001 publication in *Visual Computer* cites FXT1 and notes that it was available online at <http://www-dev.3dfx.com/fxt1/fxt1whitepaper.pdf>. Ex. 1011, *4160 (Chen et al., *A JPEG-like texture compression with adaptive quantization for 3D graphics application*, the *Visual Computer*, October 2, 2001). Similarly, a February 2006 conference paper cites FXT1 and notes its availability online and its publication date of 1999. Ex. 1012, *7000 (Stachera et al., *Fractal-based hierarchical mip-pyramid texture compression*, *Machine Graphics and Vision*, February 2006). An online technology site also noted FXT1 and reproduced several figures from the paper. Ex. 1013, (https://web.archive.org/web/20041120003604/http://www3.sharkyextreme.com/hardware/articles/99/3dfx_fxt/wp/4.shtml).

68. An article from Electronic News Online dated November 22, 1999 notes that 3dfx launched its graphics add-in boards with FXT1 texture compression

support. Ex. 1018, (Hesseldahl, *Do You Voodoo Like I Do?*, Electronic News Online, November 22, 1999). Additionally, PC Gamer Magazine reported that 3dfx announced its Voodoo4 and 5 graphics add-in boards at Comdex in Las Vegas. Ex. 1014, *7017 (*The New Voodoos*, PC Gamer, Vol. 7 No. 2, February 2000). The report also noted the support of FXT1. *Id.* Similarly, Computer Games Magazine reported on 3dfx' new open-source texture compression. Ex. 1015, *6991 (Gizmopolis, Computer Games Magazine, No. 109, December 1999).

69. Contemporaneous patent publications also acknowledged FXT1 to be prior art. Indeed, the '628 patent itself does so, though it does not provide an accurate characterization of FXT1 as explained above in paragraph 33. See Ex. 1001, 2:46-52. U.S. Patent Pub. No. 2004/0151372 ("Reshetov") also does so and describes the operation of FXT1. Ex. 1008, ¶¶ [0004]-[0015].

C. Reshetov

70. Reshetov was filed on January 23, 2004 and was published on August 5, 2004. Reshetov claims priority to an application filed on June 30, 2000 and is prior art to the '628 patent under at least 35 U.S.C. § 102(a), (b), and (e).

71. Reshetov described a block decomposition-based method for texture compression. During the compression step, the method computes a first color value is stored for the first block, and the second color value is stored for the second block. The overall compression method reduces the size by storing for

each texture block bits representing a color value and the bits that are used to index into each of the texels to a color. Some of the embodiments provide the color palette for a block of the image, where at least one color in the palette is associated with another block. It also describes an appropriate decompression method. Reshetov also disclosed using an error function to quantify the ability of one or more color palettes to represent the original color values of various texels in a texture map.

VIII. GROUND 1: SORGARD AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 7-12

A. Independent Claim 7

72. To the extent that the preamble of claim 7 is limiting, it is my opinion that Sorgard disclosed **a computer processing system**.

73. For example, Sorgard disclosed an “apparatus to compress (and decompress) texture data in computer graphics systems.” Ex. 1004, 1:9-12.

74. It is my opinion that Sorgard discloses **a processor and a memory**.

75. For example, Sorgard disclosed that a preferred embodiment of its invention is a graphics processor. Ex. 1004, 19:10-14. It was well known by the time of the '628 patent that such a graphics processor comprised internal memory and/or coupled to external memory. *See, e.g., id.*, 26:30-36, 35:23-26.

76. Alternatively, the invention of Sorgard may be implemented using a computer program executed on a computer system. *Id.*, 23:37-39. It was well known by the time of the '628 patent that such a computer system comprised a processor such as the central processing unit ("CPU") to execute instructions. Accordingly, Sorgard disclosed that its invention may be implemented in computer readable instructions stored in memory such as CD-ROM, ROM, or hard disk. *Id.*, 23:39-42; *see also id.*, 23:50-64 ("such instructions may be stored using any memory technology . . .").

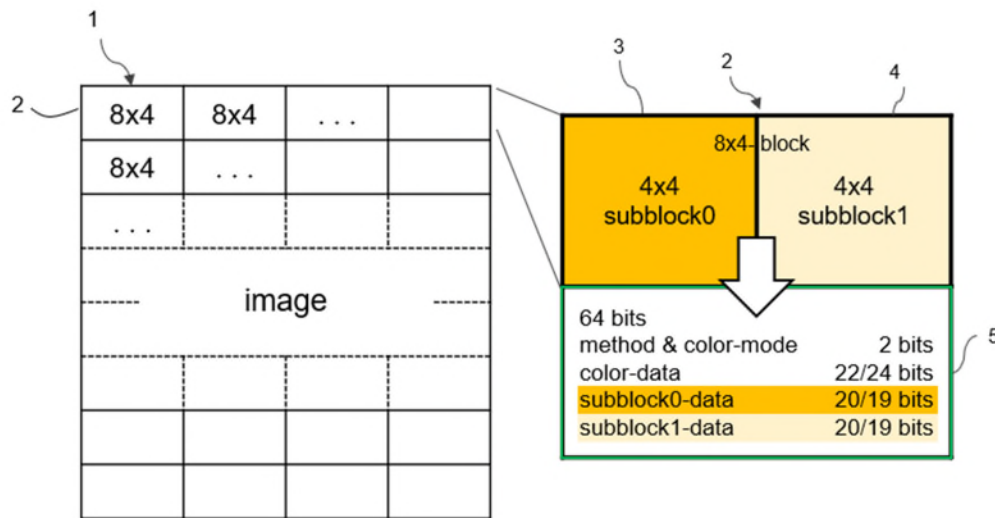
77. It is my opinion that Sorgard discloses a processor configured to receive a compressed texture block, wherein the compressed texture block includes two or more disjoint subsets into which data in the compressed texture block is to be unpacked, wherein the two or more disjoint subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints.

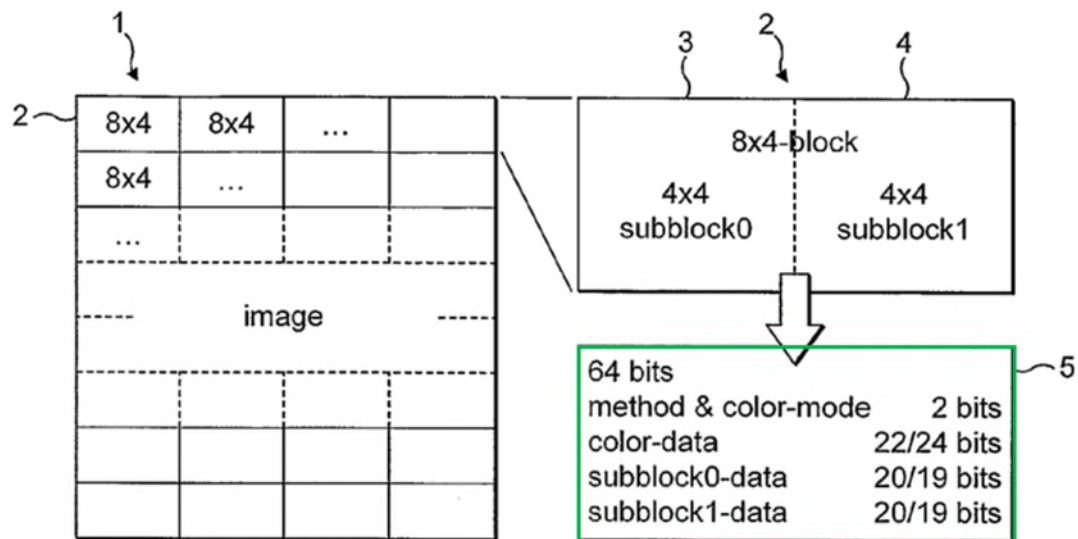
78. Whether the method of Sorgard ran on a graphics processor or via software running on a CPU of a conventional computer system, my opinion is that a POSITA would have understood that the processor would be configured to execute the method of Sorgard. Ex. 1004, 19:7-10.

79. Sorgard disclosed texture compression and decompression, which was also referred to as encoding and decoding. *See, e.g., id.*, 1:37-39; 2:3-5, 18:11-14.

80. To the extent Sorgard's disclosure addresses texture compression, a POSITA would have understood that the disclosure applies to decompression because decompression is the inverse of the compression operation. *Id.*, 16:61-17:1.

81. Sorgard disclosed that its processor is configured to **receive a compressed texture block**. For example, Sorgard disclosed a compressed texture block. *Id.*, 24:41-47:

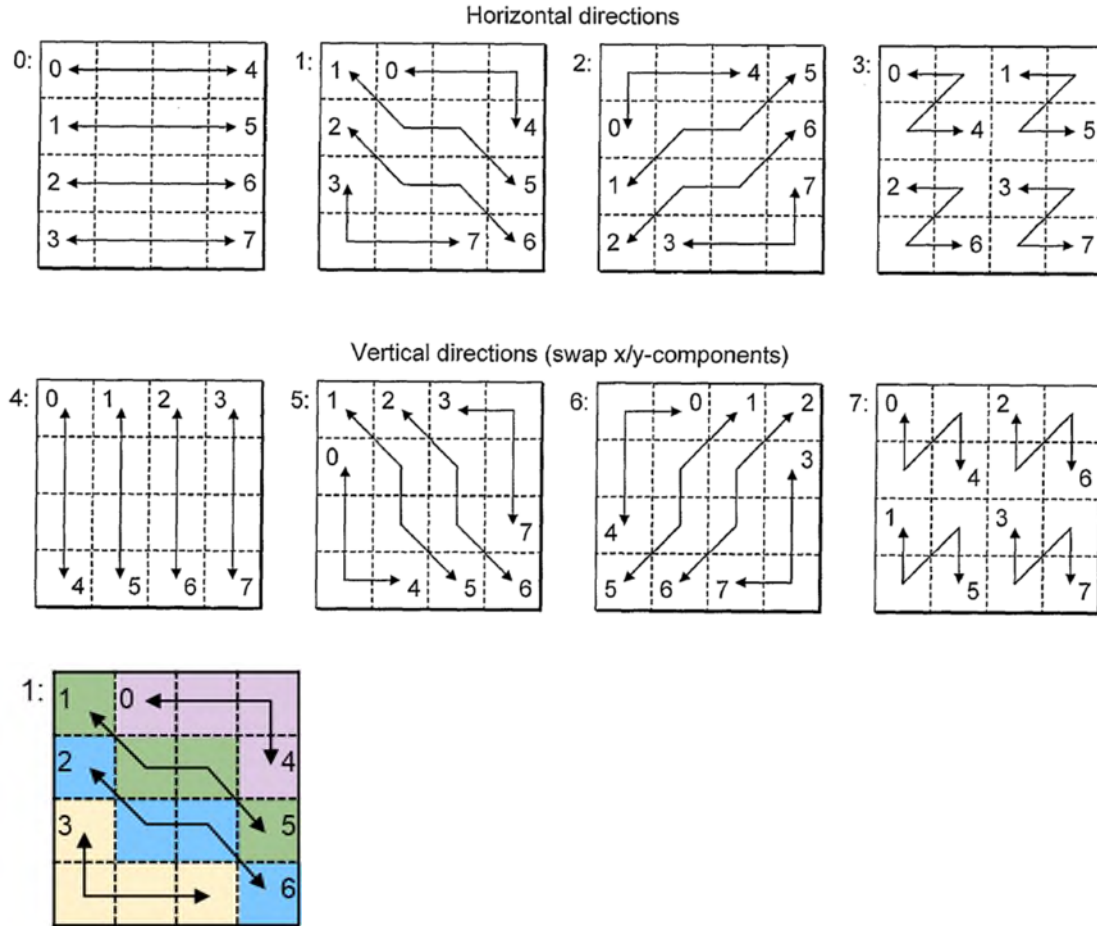




Id., Fig. 1 (annotated).

82. Sorgard further disclosed decoding or decompressing the compressed texture block. *See, e.g., id.*, 18:11-21. This necessarily meant that the processor was configured to receive the compressed texture block so that it could decompress it. *Id.*, 17:52-56.

83. Sorgard comprises **a compressed texture block including two or more disjoint subsets**. Specifically, Sorgard disclosed dividing each 4x4 texture block into a plurality of interpolation paths. *Id.*, 33:23-28. As shown in Figure 5 below, these interpolation paths are nonoverlapping or disjoint:



Id., Fig. 5 (annotated). Each interpolation path covers four texture elements, and four interpolation paths cover all 16 texture elements in the 4x4 block because the paths are nonoverlapping or disjoint. *See id.*, 33:23-28.

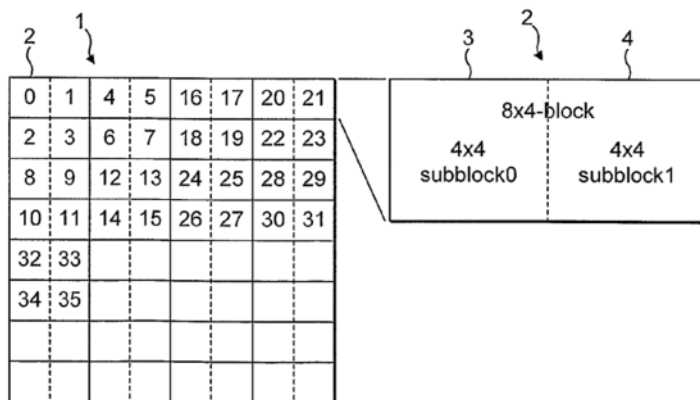
84. The “fading pattern [of an image] defines a set of interpolation paths to be applied to texture elements that the block represents.” *Id.*, 13:7-9. Sorgard disclosed that shaping the interpolation paths after the fading pattern achieved “particularly good colour reproduction for photo-realistic textures.” *Id.*, 11:43-47. Next, the claim limitation requires some data in the compressed texture block to be

unpacked. In their infringement claim chart mapping this limitation, Patent Owner interpreted as follows:

The two or more disjoint subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints. Furthermore, the compressed texture block includes two or more disjoint subsets into which data in the compressed texture block is to be unpacked. For example, each color gradient comprises a first and second endpoint (“Endpoint 1” and “Endpoint 2”). Furthermore, the “color data for a block is encoded as a gradient between two color endpoints.” “Each texel selects a position along that gradient, which is then interpolated during decompression.”

See Ex. 1019. Accordingly, Patent Owner has argued, for purposes of infringement, that claim element 7(b)—including the limitation “two or more disjoint subsets into which data in the compressed texture block is to be unpacked”—is satisfied when a compressed texture block contains two or more disjoint subsets, color data is encoded as a color gradient between two color endpoints, each subset has its own set of color endpoints, and the texel colors for each subset are interpolated during decompression. Under this interpretation by Patent Owner, Sorgard discloses claim element 7(b) as discussed below here and with respect to the next limitation.

85. For example, compressed data are partitioned into disjoint interpolation paths. See *id.*, Fig. 5. Because the compressed data comprise an ordered set of data to represent an image, and the image would be reconstructed by decompressing the compressed data, it would have been at least obvious that unpacking data in the compressed texture block would maintain the order of the data set and keep the data in their same respective disjoint subsets in which they reside in the compressed texture data block, based in part on color endpoints and interpolation them, as described in more detail below. Sorgard disclosed such ordered arrangements at both the block level and the texel level. *Id.*, 27:44-52 (“The individual texture data blocks are further arranged in the present invention such that the 4×4 texture element sub-blocks that are encoded by the texture data blocks are in Morton order. Thus, if the texel location is x, y , and $x_{15}-x_0$ and $y_{15}-y_0$ are the bits in x, y , the block position (in bytes) is $y_{15}x_{15}y_{14}x_{14} \dots y_3x_3y_2x_2$, and the texel position inside the block is $x_2y_1x_1y_0x_0$. An example of this block layout for a 32×32 array (map) is shown in FIG. 2.”).



Id., Fig. 2.

86. Further, it is my opinion that a POSITA would have understood that such ordered arrangement, based on the color endpoints and interpolating between them (as discussed more fully below with respect to the next limitation), meant that data in the compressed texture block is to be unpacked into the two or more disjoint subsets within which the data is positioned in the compressed texture data. *See also id.*, 34:28-31.

87. Sorgard further disclosed that **the two or more disjoint subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints**. Specifically, each interpolation path “cover[ing] a plurality of texture elements is defined, together with its endpoint data values (e.g. colours).” *Id.*, 11:29-31; 12:12-15 (“Thus, for example, the data indicating a path that covers a number of texture elements preferably defines or indicates a path that covers a plurality of texture elements, and the endpoint data values for the path”); 33:40-41 (“This encoding consists of including in the texture data block the colour at each end point in the fade pattern”).

88. Because there is a set of end points for each of a plurality of interpolation paths, there is a first subset having a first set of color endpoints and a second subset having a second set of color endpoints.

89. To the extent that it is argued that Sorgard does not disclose the unpacking requirement of claim element 7[b], such an argument falls short. Sorgard discloses deriving a 4-color table for each 4x4 sub-block from the 64-color global color table and further determining the colors of interpolation path endpoints from the 4-color table using 2-bit indices. Ex. 1004, 33:53-61. Claim element 7[b] merely requires a disjoint subset to include a set of color endpoints. An interpolation path includes two color endpoints (e.g., endpoints 2 and 6 above) and satisfies the requirement. Sorgard discloses determining the colors of the endpoints for each interpolation path. In the ITC action, AMD argues that determining a color endpoint using a 2-bit index is somehow outside the scope of the requirement that the disjoint subsets include subsets having their respective set of color endpoints. Put differently, AMD appears to construe the claim to require the actual color data (e.g., RGB data) for the endpoints to be included in the disjoint subsets in the compressed texture block. But the plain and ordinary meaning of the claim places no such restriction.

90. Sorgard discloses a processor configured to **individually decompress each of the two or more disjoint subsets to form texels.**

91. For the same reasons I explained above with respect to claim limitation 7[b], either the graphics processor or a processor of a conventional

computer system running graphics software is a processor configured to be performed the step required by this claim limitation 7[c].

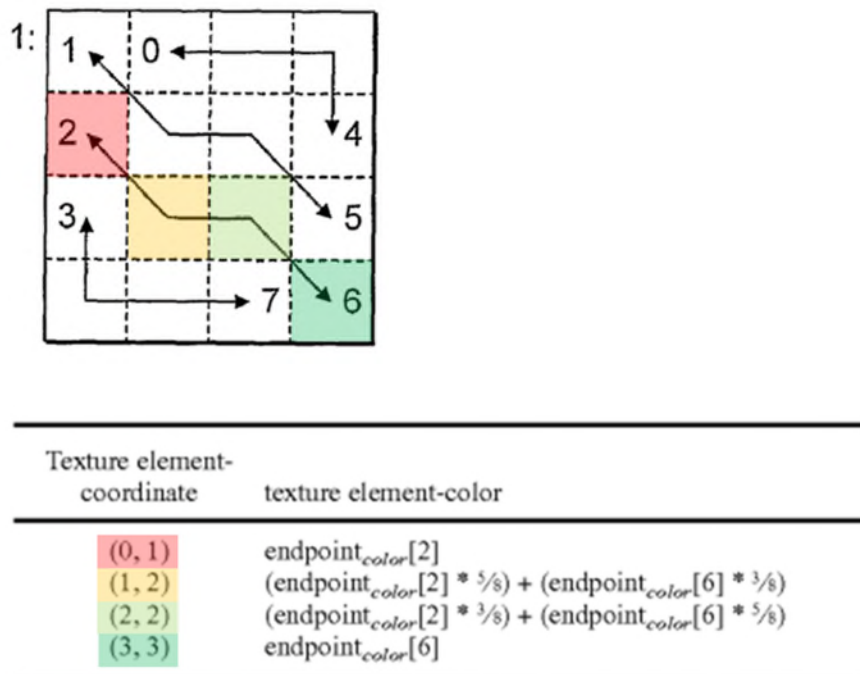
92. Sorgard discloses that each interpolation path (disjoint subset) is reproduced or decompressed independently. Ex. 1004, 13:39-45. A POSITA would have understood that decompressing the interpolation paths independently meant decompressing them separately or individually because the opposite, e.g., processing them together, would not be decompressing them independently. See Ex. 1001, claim 10 (claim dependent from the current claim 7 and hence presumably narrower in scope requiring independently decompressing the disjoint subsets in addition to requiring individually decompressing the disjoint subsets).

93. Sorgard's processing of interpolation paths also taught that each of the two or more disjoint subsets (interpolation paths) is individually decompressed to form texels. For example, all texels of an interpolation path are decompressed to a combination of the colors of its two endpoints:

$$\begin{bmatrix} \text{texel0: } endpoint_{color}[x] \\ \text{texel1: } \left(endpoint_{color}[x] * \frac{5}{8} \right) + \left(endpoint_{color}[y] * \frac{3}{8} \right) \\ \text{texel2: } \left(endpoint_{color}[x] * \frac{3}{8} \right) + \left(endpoint_{color}[y] * \frac{5}{8} \right) \\ \text{texel3: } endpoint_{color}[y] \end{bmatrix}$$

Ex. 1004, 33:62-34:8.

94. Sorgard illustrates this general decompression formula with a specific example of endpoints 2 and 6 of direction 1 in Figure 5:



Id., Fig. 5 (partial and annotated); *id.*, 34:15-22 (annotated).

95. As shown, all four texels in the direction-1 interpolation path are decompressed to form texels using endpoint color [2] (in light red) and endpoint color [6] (in light green). This set of four texels of one disjoint subset (interpolation path) is decompressed individually, separate from other interpolation paths. And this process would repeat for each of the interpolation paths within the block to form texels covering the entire block.

96. In my opinion, decompression in Sorgard comprises **determining a first texel color for a first texel that is part of the first disjoint subset by**

interpolating between color endpoints of the first set of color endpoints based on a first index value for the first texel.

97. The interpolation is shown in Sorgard's formulas cited above with respect to claim element 7[c] meets this claim limitation. Specifically, Sorgard disclosed determining the texel colors at coordinates (1, 2) and (2, 2) (shaded yellow and light green) that is part of the direction-1 interpolation path (first disjoint subset) by interpolating between color endpoints at coordinates (0, 1) and (3, 3) (shaded red and green). *See also* Ex. 1004, 33:16-22; 11:25-42; 12:24-30; claims 13, 14.

98. Sorgard discloses that such determination is based on an index value for the texel whose color is to be determined. *Id.*, 16:10-17 ("In another arrangement, only the data values for a selected number of the texture elements that the block encodes are indicated (e.g., the "endpoint" texture elements, where a fading pattern or interpolation path is to be used to derive data values for individual texture elements). This data could again be provided in the form of a data value index for each such texture element, or at least allow such an index to be derived.").

99. Sorgard uses an index to determine the fading pattern of the interpolation paths. Ex. 1004, 33:43-47. For example, a 3-bit index is used to select one of the eight fading patterns in Figure 5. For each texel, this index

determines the interpolation formula used to derive its color. For example, an index value of 0, 1, 2, or 3 (shown in green boxes below) determines that the texel at coordinate (1, 2) (yellow) is to be interpolated between endpointcolor[2] and [6], with endpointcolor[2] weighed more heavily, while the texel at coordinate (2,1) (purple) is to be interpolated as between endpointcolor[5] and [1], with endpointcolor[5] weighed more heavily.

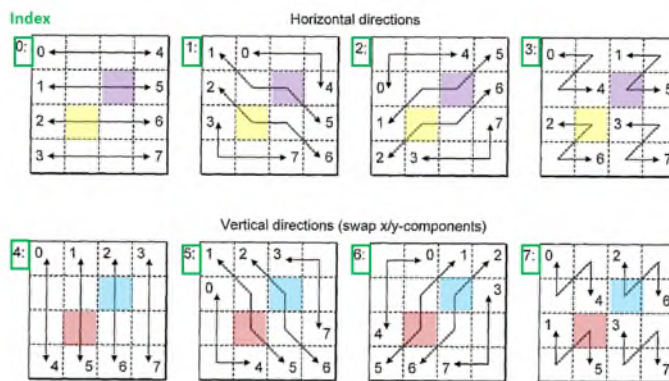


FIG. 5

100. Similarly, an index value of 4, 5, 6, or 7 determines that the texel at coordinate (1, 2) (red) is to be interpolated as $(\text{endpoint}_{\text{color}}[5] * 5/8) + (\text{endpoint}_{\text{color}}[1] * 3/8)$, while the texel at coordinate (2,1) (blue) is to be interpolated as $(\text{endpoint}_{\text{color}}[2] * 5/8) + (\text{endpoint}_{\text{color}}[6] * 3/8)$.

101. Additionally or alternatively, Sorgard disclosed an index that is used to determine the direction of the fading pattern or interpolation path. Ex. 1004, 33:43-47. To the extent that the determination of the texel color is applied to a texel based on this index indicating the direction of the interpolation path, such determination is based on an index value for the texel whose color is to be

determined. Put differently, it would have been obvious that texel position (or equivalent, e.g., texel location in the compressed data representation) together with the 3-bit index constitute the index value of the '628 patent.

102. It is my opinion that Sorgard includes **determining a second texel color for a second texel that is part of the second disjoint subset by interpolating between color endpoints of the second set of color endpoints based on a second index value for the second texel.**

103. It is my opinion that the reasons above in claim limitation 7[d] apply to this claim limitation, with the addition that Sorgard's decompression algorithm is repeated for a second texel that is part of a second disjoint subset, which a POSITA would have understood to be inherent in Sorgard.

B. Dependent Claim 8

104. In my opinion, Sorgard discloses the computer processing system of claim 7 as I explained above. It is also my opinion that Sorgard further discloses the additional limitation of claim 8.

105. The requirement of claim 8 that **the compressed texture block represents the original texture block** is inherent in the key objective of image or texture compression, namely, to reduce the data size of an image without distorting the image unacceptably. Thus, it is inherent in working texture compression

algorithms including Sorgard's that the compressed texture block represents the original texture block. *Id.*

106. Sorgard confirms this conclusion. For example, Sorgard taught that the encoded texel data values represent the original texture data values because, after decompression, a representation of the original data is reproduced. Ex. 1004, 3:9-15 ("As will be appreciated by those skilled in the art, the actual data values accorded to the texture elements, both in their original, unencoded raw form, and in their encoded form (or at least when the encoded data has been decoded to reproduce a representation of the original data) will depend on the form of 'texture' that the texture elements are intended to define or represent."). Sorgard further taught that the compressed data, upon decompression, generate a set of texel colors corresponding to the original texture data. *Id.*, 3:16-22 ("Thus, for example, where the texture elements define colours to be used, the texture data elements in their original, unencoded form will each comprise a set of colour values (Red, Green, Blue (RGB), or a set of colour and transparency values (Red, Green, Blue, Alpha (RGBA)), and the encoded data, when decoded (reproduced), will generate a corresponding set of colour values."). As a further example, Sorgard disclosed that "a plurality of such individual texture data blocks will be needed to encode the overall set of original texture data (texture data elements)." *Id.*, 4:31-34.

107. Indeed, Sorgard taught that it was known in the art that the compressed data represented the original texture data. *Id.*, 6:9-14

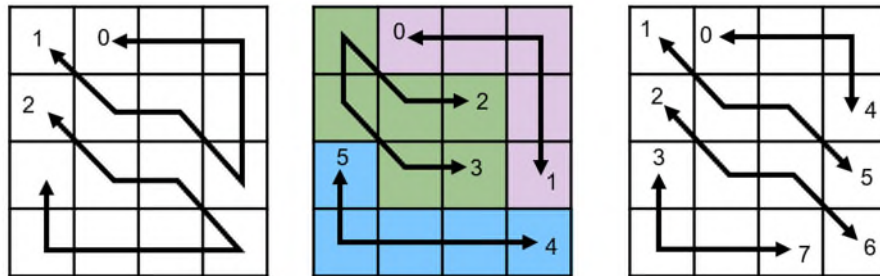
C. Dependent Claim 9

108. It is my opinion that Sorgard discloses the computer processing system of claim 7 as I explained above. It is further my opinion that Sorgard discloses or at least rendered obvious the additional limitation of claim 9.

109. Sorgard discloses that “[t]he number of texture elements that an interpolation path [a disjoint subset] covers can be selected as desired.” Ex. 1004, 12:47-48. Based on Sorgard’s teaching of matching the pattern of interpolation paths to the fading pattern of the image, it is my opinion that a POSITA would have understood that it would be desired to vary the number of texels in an interpolation path, i.e., the size of the disjoint subset, based on the fading pattern of the image. Such variation would produce more realistic textures. *See id.*, 11:43-47; 11:11-17; 13:22-29.

110. Further, Sorgard taught that an interpolation path of a fixed size of four texels is a preferred embodiment. *Id.*, 12:50-51 (“In a preferred embodiment, each interpolation path covers four texture elements of the block . . .”). It is my opinion that a POSITA would have understood from that statement that Sorgard permitted embodiments of different-sized interpolation paths or disjoint subsets.

111. Additionally, Sorgard discloses that the number of interpolation paths in a 4x4 block can be varied, e.g., a block can have 2 paths of 8 texels each, or 4 paths of 4 texels each. Thus, for example, a 4x4 block can have 3 interpolation paths. See Ex. 1001, 7:9-15 (3 subsets per block). These paths would necessarily cover different numbers of texels (e.g., 5, 5, 6 texels for the 3 paths) because the together they must cover 16 texels. Two paths of 5 texels and a path of 6 texels have **different sizes**.



112. Further, it is my opinion that a POSITA would have understood that Sorgard's method supported different-sized interpolations paths. For example, a preferred embodiment indicated the texels covered by an interpolation path. Ex 1004, 12:17-20 ("Where such an interpolation arrangement is to be used, then the encoded texture data block preferably indicates the texture elements that the interpolation path or paths to be used covers."). A POSITA would have understood that such indication could cover interpolation paths of different sizes. More specifically, in a more preferred embodiment, an interpolation path is

indicated by the two endpoints and the paths between the two endpoints. *Id.*, 12:20-23 (“This is preferably done by indicating the endpoints of the interpolation path and the path to be followed between these endpoints (e.g., whether it is a straight line or zig-zag, etc.).”) A POSITA would have understood that interpolation paths of different sizes could all be represented in this manner.

D. Dependent Claim 10

113. It is my opinion that Sorgard discloses the computer processing system of claim 7 as I explained above. It is further my opinion that Sorgard discloses the additional limitation of claim 10.

114. Sorgard discloses that each interpolation path (disjoint subset) is reproduced or decompressed independently. Ex. 1004, 13:39-45 (“Where, as discussed above, the texture data block encodes sub-sets of texture data elements that are to be reproduced independently, then the fading pattern(s) (or interpolation scheme) preferably covers a single sub-set of the texture elements, i.e. such that a fading pattern (or interpolation scheme) will need to be applied to each sub-set of texture elements independently.”); *see also id.*, 20:20-25 (“two or more sets of data, each set of data representing a sub-set of the set of data elements that is to be reproduced independently of the other sub-sets of data elements.”).

E. Dependent Claim 11

115. Sorgard disclosed the computer processing system of claim 7 as I explained above. It is also my opinion that Sorgard disclosed the additional limitation of claim 11.

116. The requirement of claim 11 is inherent in texture compression, namely, to reduce the data size of an image without distorting the image unacceptably. Thus, the decompressed image may be considered an approximation as recited in claim 11. *Id.* Sorgard disclosed that its algorithm resulted in “little loss of quality.” Ex. 1004, 35:17-22 (“Notwithstanding the relatively high compression rate and smaller compact nature of the data compression format of the present invention, it is still very capable of compressing, for example, different kinds of images, and in particular both real-world images and drawings, with little loss of quality.”). In other words, the decompressed or reconstructed image is a close approximation of the original image.

117. Furthermore, applying a decompressed image to a rendered surface for display was the intended application of image/texture compression/decompression because images were meant to be displayed — being shown on a display is the reason why they were created in the first place. Specifically, texture is used to project a 3D image onto a 2D surface, to create on a 2D surface the appearance of a 3D object. This was a common POSITA

knowledge by the time of the '628 patent. *See, e.g.*, Ex. 1001, 1:13-26 (“It is common in computer graphics systems to generate textures and colours for pixels in the image to be displayed by applying so-called textures or texture data to the surfaces to be drawn. For example, in 3-dimensional (3D) computer graphics, surface detail on objects is commonly added by applying a “texture”, such as an image of a piece of fabric, to a set of polygons representing the object, to give the rendered image of the object the appearance of the ‘texture’ (e.g. fabric). Such textures are typically applied by storing an array of texture elements or ‘texels,’ each representing a given texture (e.g. colour, luminance, and/or light/shadow, etc.) value, and then mapping the texels onto the corresponding elements, such as (and, indeed, typically) the pixels, in the image to be displayed.”). Thus, it is inherent that the processor running the Sorgard algorithm is **configured to apply the approximation to a rendered surface for display.**

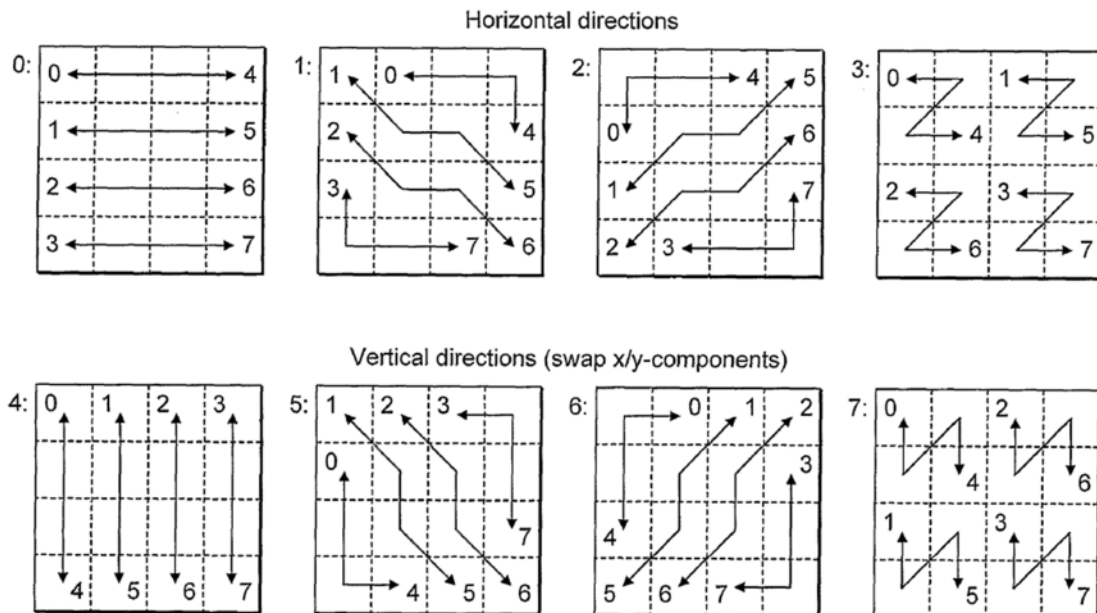
F. Dependent Claim 12

118. Sorgard disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is further my opinion that Sorgard rendered obvious the additional limitation of claim 12.

119. For example, Sorgard disclosed “using error comparison techniques to determine the reduced set of data values that best matches the original set of data values” to provide a “base set of data values to be used for the texture data block or

blocks.” Ex. 1004, 6:19-24. These error comparisons are used to evaluate the quality, which is specified using given parameters. If the resulting error is within the quality parameter specification such as below a pre-selected threshold, that can provide some guarantees on the image quality retained by texture compression.

120. Sorgard further disclosed that “[i]n a preferred embodiment, eight fading patterns, each having a different combination of interpolation paths are used. This has been found, in the case of colour data at least, to facilitate particularly good reproduction of, e.g., photo-realistic images.” *Id.*, 13:25-29. These eight fading patterns are shown in Figure 5. *Id.*, 33:29-35.



Id., Fig. 5.

121. In my opinion, a POSITA would have understood or at least found it obvious that Sorgard determined this set of eight fading patterns, out of all possible

fading patterns in a 4x4 blocks, as some examples that could facilitate particularly good reproduction of photo-realistic images through selecting a set of photo-realistic images, selecting a quality parameter such as an error value of a possible fading pattern, and selecting a threshold to sift or filter out fading patterns that perform poorly against the set of photo-realistic images based on a comparison of the quality parameter with the pre-selected threshold. This is an extensive search effort because each image typically comprised many 4x4 texture blocks, and even a desirable number of fading patterns in the set (eight) was not known in the beginning. Thus, to narrow the possible sets of fading patterns, a POSITA would have at least found it obvious to discard a fading pattern that performed particularly poorly against an image, and that an obvious criterion of a particularly poor performance was a pre-selected error threshold. Such a use of threshold was well known to a POSITA by the time of the '628 patent. *See, e.g.*, U.S. Patent Pub. No. 2004/0228527 ("Iourcha '527") ¶ [0039] ("According to one embodiment, the decompression of each original block of the image is optimized so that the loss in image quality is reduced to a desired amount, or meets some threshold."); *id.*, ¶ [0058] ("If the error generated is not greater than the threshold value, at block 750, the encoding is acceptable. If the error generated is greater than the threshold value, at block 760, the values in the compressed blocks are altered to reduce the error."). Thus, to the extent that Sorgard's method used this set of eight fading

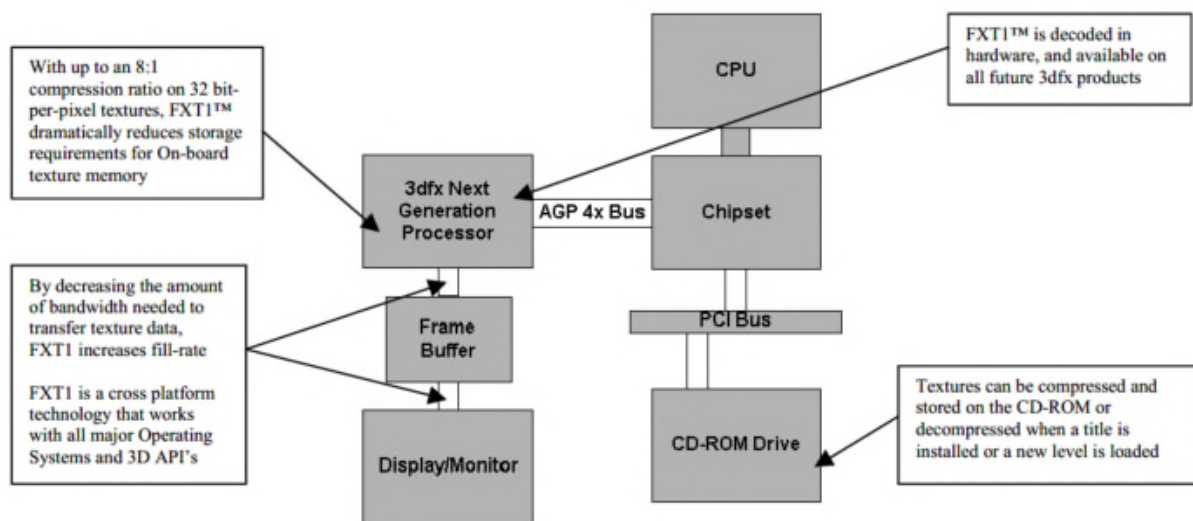
patterns or interpolation paths, each of the one or more interpolation paths (disjoint subsets) is associated with a quality parameter that met a pre-selected threshold.

IX. GROUND 2: FXT1 AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 7-12

A. Independent Claim 7

122. I have been informed that the preamble may not be limiting. To the extent that the preamble of claim 7 is limiting, it is my opinion that FXT1 disclosed a **computer processing system**.

123. For example, FXT1 discloses a computer processing system comprising a CPU, a 3dfx next-generation processor, a frame buffer, a CD-ROM drive, and a display/monitor:



Textures have a long way to travel before they are rendered on your screen. FXT1™ reduces their size considerably, thereby reducing bandwidth requirements making the transfer more efficient

Ex. 1005, 5.

124. Alternatively, FXT1 discloses software tools including a command-line utility and an Adobe PhotoShop™ plug-in, both of which ran on a computer and were capable of supporting compression and decompression using the FXT1 algorithm. *Id.*, 7. The computer that these software tools ran on constitutes a computer processing system.

125. FXT1 expressly disclosed these two implementations. *Id.*, 8 (“Decoding FXT1™ textures can be done transparently in the 3D hardware at run-time, or decoded in software . . .”).

126. In my opinion, FXT1 disclosed **a processor and a memory**.

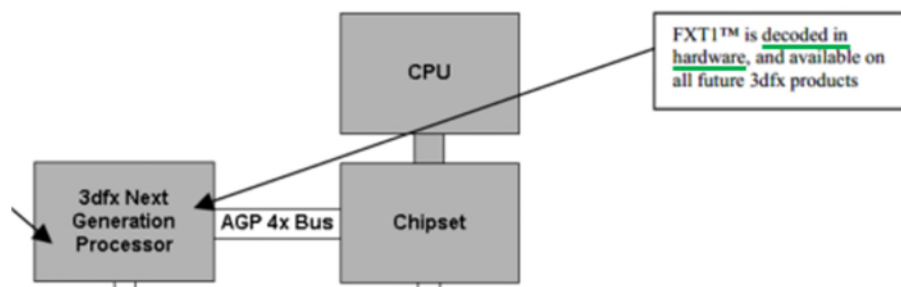
127. For example, the 3dfx next-generation processor satisfies the requirements for a processor in this claim. Ex. 1005, 5 (figure cited above with respect to 7[preamble]). Either the frame buffer or a CD-ROM in the drive constitutes a memory. Further, texture compression was known to be a memory-intensive application, and FXT1 discussed memory size and bandwidth considerations in texture compression. *Id.*, 3. Thus, a computer processing system running FXT1 algorithm comprised a memory in addition to a memory in processors like the CPU or the 3dfx next-generation processor that was well known by the time of the '628 patent.

128. Alternatively, a POSITA would have understood that the computer that the 3dfx software tools ran on comprised a processor such as a CPU and a

memory such as RAM and/or a hard drive, as such computer systems were conventional by the time of the '628 patent.

129. In my opinion, FXT1 disclosed or at least rendered obvious a processor configured to receive a compressed texture block, wherein the compressed texture block includes two or more disjoint subsets into which data in the compressed texture block is to be unpacked, wherein the two or more disjoint subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints.

130. Whether the FXT1 algorithm ran on the 3dfx next-generation processor or via a 3dfx software tool running on a processor of a conventional computer system, it is my opinion that a POSITA would have understood that that the processor would be configured to execute the FXT1 algorithm. For example, the figure showing an FXT1 system has an arrow pointing to the 3dfx next-generation processor and states that FXT1 is “decoded in hardware”:

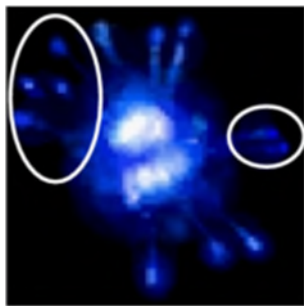


Ex. 1005, 5 (partial and annotated).

131. Based on this figure, it is my opinion that a POSITA would have understood that the 3dfx next-generation processor implemented the FXT1 texture decoding or decompression algorithm.

132. Alternatively, FXT1 texture decoding or decompression may be done using 3dfx's free software tools running on a conventional computer system. *Id.*, 7.

133. FXT1 disclosed how its texture compression algorithm works. *Id.*, 5 (“FXT1™ texture compression works by . . .”). A POSITA would have understood that the compression algorithm outputs a compressed image comprising a compressed texture block. For example, the FXT1 compression algorithm can output a compressed image that, after decompression, comprises this image represented by a texture block:



Id., 7.

134. In my opinion, a POSITA would have understood that decompression involves **receiving a compressed texture block** as required by this claim limitation. Further, a POSITA would have understood that decompression is the

inverse of the compression operation, and that what FXT1 disclosed about its compression algorithm applied to FXT1 decompression.

135. Reproduced a description of FXT1 compression/decompression algorithm that this Petition will refer to repeatedly:

F -Compression (also known as Encoding):

FXT1™ texture compression works by dividing an image into multiple 4x4 and/or 4x8 texel blocks. Each texture (including opaque and transparent textures) is compressed to an average of four bits-per-texel. In the FXT1™ texture compression scheme, four different compression algorithms are used, with the best one chosen per block to generate the highest quality result. The four algorithms used are:

-Compression (also known as Encoding):

FXT1™ texture compression works by dividing an image into multiple 4x4 and/or 4x8 texel blocks. Each texture (including opaque and transparent textures) is compressed to an average of four bits-per-texel. In the FXT1™ texture compression scheme, four different compression algorithms are used, with the best one chosen per block to generate the highest quality result. The four algorithms used are:

1. *CC_MIXED* (similar to other texture compression schemes): A 4x4 texel block is represented by two bits-per-texel for opaque textures. Additionally, each block has two 16-bit colors stored in an RGB 565 format. The two RGB 565 colors and two additional colors (created by interpolating between the two RGB 565 colors) form the primary colors for this texel block and its associated four color lookup table. A 2-bit index is used to determine which color from the lookup table will be used for each texel in the 4x4 block. Transparent textures are created by making one of the four colors transparent.

2. *CC_HI* (best for spatial resolution): A 4x8 texel block is represented by three bits-per-texel for opaque and transparent textures. Each block stores two 15-bit colors in an RGB 555 format. The two RGB 555 colors and five additional colors (created by interpolating between the two RGB 555 colors) form the primary colors for this texel block. Additionally, an eighth color is defined to be the transparent color. A 3-bit index is used to determine which color from the 8-entry lookup table will be used for each texel in the 4x8 block.

Id., 5.

136. FXT1 comprises **a compressed texture block including two or more disjoint subsets**. FXT1 partitions a texture block into nonoverlapping or disjoint subsets. *Id.*, 5 (“FXT1™ texture compression works by dividing an image into multiple 4x4 and/or 4x8 texel blocks.”). More specifically, FXT1 partitions a texture block into 4x4 or 4x8 nonoverlapping or disjoint subsets in the *CC_MIXED* and *CC_HI* mode, respectively. *Id.* (“1. *CC_MIXED* . . . A 4x4 texel block is represented by two bits-per-texel for opaque textures . . . 2. *CC_HI* (best for spatial resolution): A 4x8 texel block is represented by three bits-per-texel for opaque and transparent textures”). Whether a 4x4 or 4x8 subset is selected (or whether the *CC_CHROMA* or *CC_ALPHA* scheme is used) is determined on a subset-by-subset basis. *Id.* (“In the FXT1™ texture compression scheme, four different compression algorithms are used, with the best one chosen per block to generate the highest quality result.”). FXT1 selects the best one of the four schemes for each subset to generate the final compressed data. *Id.* Accordingly, each image block is typically partitioned into a mixture of 4x4 and 4x8 nonoverlapping or disjoint subsets.

137. [Paragraph intentionally left blank]

138. FXT1 discloses that **data in the compressed texture block is to be unpacked into the two or more disjoint subsets.**¹ For example, in CC_MIXED and CC_HI schemes, FXT1 compressed data are partition into 4-texel-by-4-texel or 4-texel-by-8-texel disjoint subsets with color endpoints. *Id.*, 5. Because the compressed data comprise an ordered set of data to represent an image, and the image would be reconstructed by decompressing the compressed data, it would have been at least obvious that unpacking data in the compressed texture block, based in part on the color endpoints as described in more detail below with respect to the next limitation, would maintain the order of the data set and keep the data in their same respective disjoint subsets in which they reside in the compressed texture data block.

139. FXT1 further disclosed that **the two or more disjoint subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints.** For example, in the CC_MIXED scheme, each disjoint 4x4 subset has “two 16-bit colors stored in an RGB 565 format.” *Id.*, 5. Two additional colors are “created by interpolating between the two RGB 565 colors.” *Id.* Thus, these two 16-bit colors constitute a set of color endpoints.

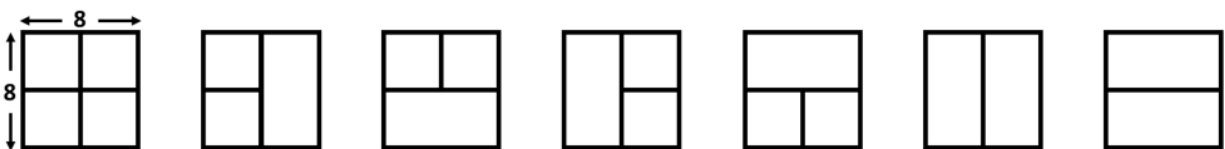
¹ Petitioner is applying the interpretation of Patent Owner on this limitation as described with respect to Sorgard above.

Additionally, in the CC_HI scheme, each disjoint 4x8 subset comprises “two 15-bit colors in an RGB 555 format.” *Id.* Five additional colors are “created by interpolating between the two RGB 555 colors.” *Id.* Thus, these two 15-bit colors constitute a set of color endpoints. Because each 4x4 or 4x8 subset comprises its own set of color endpoints, two or more disjoint 4x4 and/or 4x8 subsets include a first subset having a first set of color endpoints and a second subset having a second set of color endpoints.

140. To the extent it is argued that FXT1 did not disclose or render obvious the claim requirement **the compressed texture block includes two or more disjoint subsets**, for example, to the extent it is argued that the claimed compressed texture block would read on a 4x4 or 4x8 texel block of FXT1 and each 4x4 or 4x8 texel block does not include two or more disjoint subsets, Petitioner disagrees for the following reasons. First, the claimed “compressed texture block” in its plain and ordinary meaning can read on a block comprising an entire image or a part of an image, and the claim does not specify a size of the block. Second, the claim does not specify partitioning an image into a plurality of blocks but merely recites a single “compressed texture block.” Thus, restricting the claimed compressed texture block to FXT1’s 4x4 or 4x8 texel blocks is overly narrow. Third, “block” is not a term of art that is limited to something equivalent to a 4x4 or 4x8 texel block of FXT1. For example, at the time of the ’628 patent, a

POSITA would have broadly interpreted the word *block* in the context of texture compression. See Ex. 1008, ¶ [0075] (“In the broadest sense a texture block is a plurality of texels.”). And neither is the plain and ordinary meaning of the claimed “block” limited to, e.g., a 4x4 or 4x8 block.

141. Additionally or alternatively, to the extent it is argued that the claimed compressed texture block would read on a 4x4 or 4x8 texel block of FXT1 and each 4x4 or 4x8 texel block does not include two or more disjoint subsets, a POSITA would have found it obvious to partition an image into a plurality of texture blocks, each of which including two or more disjoint 4x4 and/or 4x8 texel blocks. For example, an image may be partitioned into a plurality of 8x8 texture blocks, each of including two or more disjoint 4x4 and/or 4x8 texel blocks. The following figure shows some of such 8x8 texture blocks further partitioned into 4x4 and/or 4x8 blocks:



142. Partitioning an 8x8 block into smaller subsets was known in the art. See Ex. 1017.

143. In my opinion, a POSITA would have found it obvious that other block sizes are available. For example, an image may be partitioned into square

blocks with each side having a length that is a multiple of 8 (8, 16, 24, etc.). And FXT1 disclosed texture sizes such as 64x64, 128x128, . . . , 2048x2048, etc.:

Texture Size	8-bit	16-bit	24-bit	32-bit	4-bit FXT1™
64x64	6 KB	9 KB	13 KB	16 KB	2KB
128x128	18 KB	33 KB	49 KB	64 KB	8KB
256x256	66 KB	129 KB	193 KB	256 KB	32KB
512x512	258 KB	513 KB	769 KB	1024 KB	128KB
1024x1024	1,026 KB	2,049 KB	3,073 KB	4096 KB	512KB
2048x2048	4,098 KB	8,193 KB	12,289 KB	16384 KB	2048KB

This table shows the memory requirements for various sizes and color depths of textures without compression. With FXT1™ texture compression, 32-bit images are reduced by a ratio of 8:1 without a perceivable loss in image quality.

Ex. 1005, 2. Each such a square block may include two or more disjoint 4x4 and/or 4x8 texel blocks. Further, it would have been obvious to partition a 2048x2048 image into, for instance, 1,024 64x64 blocks, each of which may be further partitioned into a plurality of 4x4 and 4x8 disjoint subsets. Making such an additional level of partitioning would have been nothing more than applying a well-understood, conventional image decomposition or partitioning technique with predictable results. A POSITA would have been motivated to make an additional level of partitioning to, for example, break up a large image into sets of data such that performing compression on each block would preserve the image fidelity of the texture.

144. In my opinion, FXT1 disclosed a processor configured to **individually decompress each of the two or more disjoint subsets to form texels.**

145. For the same reasons I explained above with respect to claim limitation 7[b], either the 3dfx next-generation processor or a processor of a conventional computer system running a 3dfx software tool is a processor configured to be performed the step required by this claim limitation 7[c].

146. In the CC_MIXED scheme, a two-bit index represents each texel. Ex. 1005, 5. The index is “used to determine which color from the lookup table will be used for each texel” in the disjoint 4x4 subset. *Id.* Using a per-texel index to a per-disjoint-subset lookup table to determine the color for each texel satisfies the limitation to individually decompress each of the two or more disjoint subsets to form texels because each disjoint subset is decompressed individually or separately using its own lookup table.

147. Similarly, in the CC_HI scheme, a three-bit index represents each texel. *Id.* The index is “used to determine which color from the 8-entry lookup table will be used for each texel” in the disjoint 4x8 subset. *Id.* Using a per-texel index to a per-disjoint-subset lookup table to determine the color for each texel satisfies the limitation to individually decompress each of the two or more disjoint subsets to form texels because each disjoint subset is decompressed individually or separately using its own lookup table.

148. FXT1 decompression algorithm repeats the decompression process for each of the disjoint subsets to form texels. For example, during compression, the

best of the four schemes is chosen on a per-disjoint-subset basis. *Id.* (“In the FXT1™ texture compression scheme, four different compression algorithms are used, with the best one chosen per block to generate the highest quality result.”). Each disjoint subset would be evaluated based on the quality of the outputs of the four schemes. During decompression, for each disjoint subset, a 2-bit field is “used to determine which of the 4 compression schemes was utilized by the compression algorithm for best visual quality” and thus which of the 4 schemes should be used for decompression. *Id.*, 6. This process would be repeated to decompress the two or more disjoint subsets to form texels.

149. It is my opinion that FXT1 includes **determining a first texel color for a first texel that is part of the first disjoint subset by interpolating between color endpoints of the first set of color endpoints based on a first index value for the first texel.**

150. In the CC_MIXED scheme, each disjoint 4x4 subset has “two 16-bit colors stored in an RGB 565 format.” Ex. 1005, 5. Two additional colors are “created by interpolating between the two RGB 565 colors.” *Id.* This results in a set of 4 colors, 2 color endpoints and 2 interpolated colors. A two-bit index is “used to determine which [of the 4] color[s] from the lookup table will be used for each texel” in the disjoint 4x4 subset. *Id.*

151. In the CC_HI scheme, each disjoint 4x8 subset comprises “two 15-bit colors in an RGB 555 format.” *Id.* Five additional colors are “created by interpolating between the two RGB 555 colors.” *Id.* A three-bit index is “used to determine which color from the 8-entry lookup table will be used for each texel” in the disjoint 4x8 subset. *Id.*

152. Accordingly, in both schemes, FXT1 decompression algorithm determines a texel color for a texel that is part of a disjoint subset by interpolating between two color endpoints based on a 2- or 3-bit index. The determined color is one of the two color endpoints or one of the two or five interpolated colors.

153. This process is repeated for each disjoint 4x4 or 4x8 subset. Thus, FXT1 determines a first texel color for a first texel that is part of the first disjoint subset by interpolating between color endpoints of the first set of color endpoints based on a first index value for the first texel.

154. It is my opinion that FXT1 includes **determining a second texel color for a second texel that is part of the second disjoint subset by interpolating between color endpoints of the second set of color endpoints based on a second index value for the second texel.**

155. The reasons above in claim limitation 7[d] apply to this claim limitation, with the addition that FXT1’s decompression algorithm is repeated for a second texel that is part of a second disjoint subset, which a POSITA would have

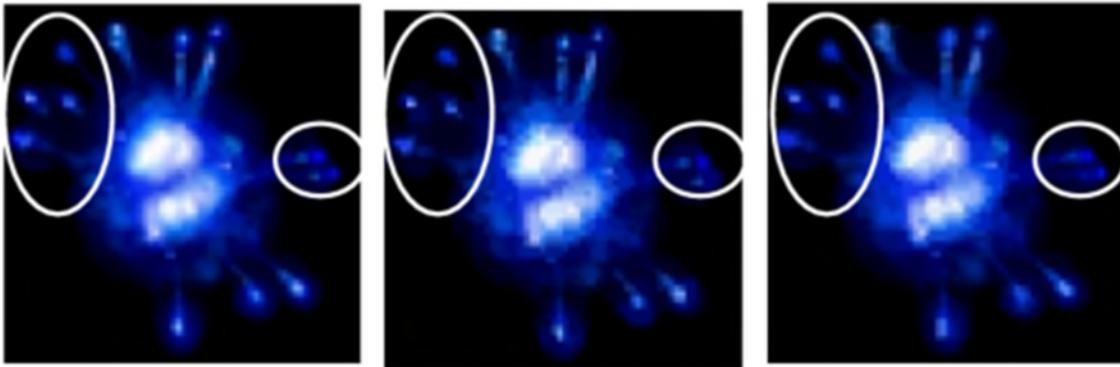
understood to be inherent in FXT1. For example, FXT1 showed complete decompressed images, which could be generated only if both claim limitations 7[d] and 7[e] are satisfied, i.e., the determining step is repeated for a plurality of texels.

B. Dependent Claim 8

156. FXT1 disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is further my opinion that FXT1 disclosed the additional limitation of claim 8.

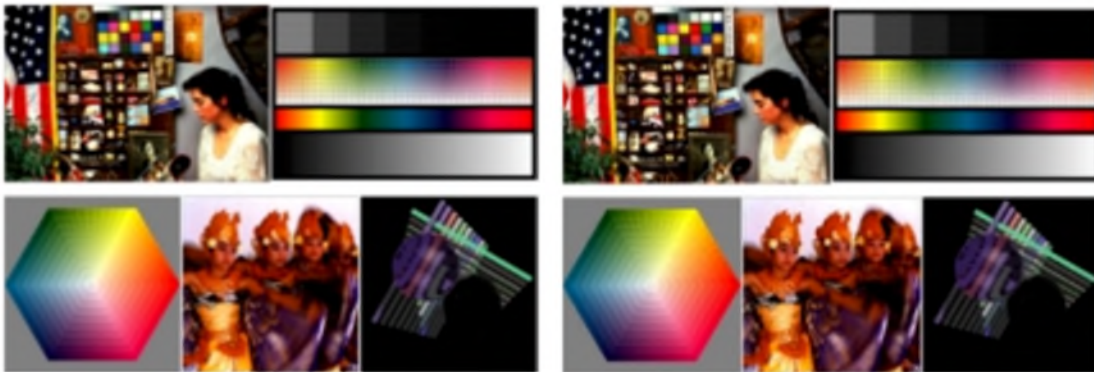
157. The requirement of claim 8 that **the compressed texture block represents the original texture block** is inherent in the key objective of image or texture compression, namely, to reduce the data size of an image without distorting the image unacceptably. Thus, it is inherent in working texture compression algorithms including FXT1 that the compressed texture block represents the original texture block.

158. FXT1 confirms this. For example, it disclosed reproducing textures “for entire rooms and all of the objects or characters in a given scene.” Ex. 1005, 3. This is achievable because the compressed texture block represents the original texture block. As another example, the uncompressed, original images juxtaposed with FXT1-processed images also confirms that the compressed texture block represents the original texture block.



Uncompressed image (left) compared to the same image compressed using S3TC (center) compared to FXT1 (right). Note how much more accurate FXT1 is at resolving the finer details of the images.

Id., 7.



High-resolution image – Uncompressed (Left) vs. same high-resolution image – Compressed (Right). While the compressed image on the right suffers no loss in visual quality, it is represented using 6 times less data than the one on the right.

Id., 8.

C. Dependent Claim 9

159. FXT1 disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is further my opinion that FXT1 disclosed the additional limitation of claim 9. Specifically, as I explained above with respect to claim 7, the CC_MIXED scheme comprises disjoint 4x4 subsets and the CC_HI scheme comprises disjoint 4x8 subsets. These have different sizes.

D. Dependent Claim 10

160. FXT1 disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is further my opinion that FXT1 disclosed the additional limitation of claim 10.

161. In the CC_MIXED scheme, a two-bit index represents each texel. Ex. 1005, 5. The index is “used to determine which color from the lookup table will be used for each texel” in the disjoint 4x4 subset. *Id.* Using a per-texel index to a per-disjoint-subset lookup table to determine the color for each texel satisfies the limitation to independently decompress each of the one or more disjoint subsets because texels in each disjoint subset are decompressed using a lookup table specific to the disjoint subset that is not contingent on another subset. *See id.* (stating that the decompression scheme is chosen from the best of the four “**per block**,” i.e., per disjoint subset).

162. Similarly, in the CC_HI scheme, a three-bit index represents each texel. *Id.* The index is “used to determine which color from the 8-entry lookup table will be used for each texel” in the disjoint 4x8 subset. *Id.* Using a per-texel index to determine the color for each texel satisfies the limitation to independently decompress each of the one or more disjoint subsets because texels in each disjoint subset are decompressed using a lookup table specific to the disjoint subset that is

not contingent on another subset. *See id.* (stating that the decompression scheme is chosen from the best of the four “*per block*,” i.e., per disjoint subset).

E. Dependent Claim 11

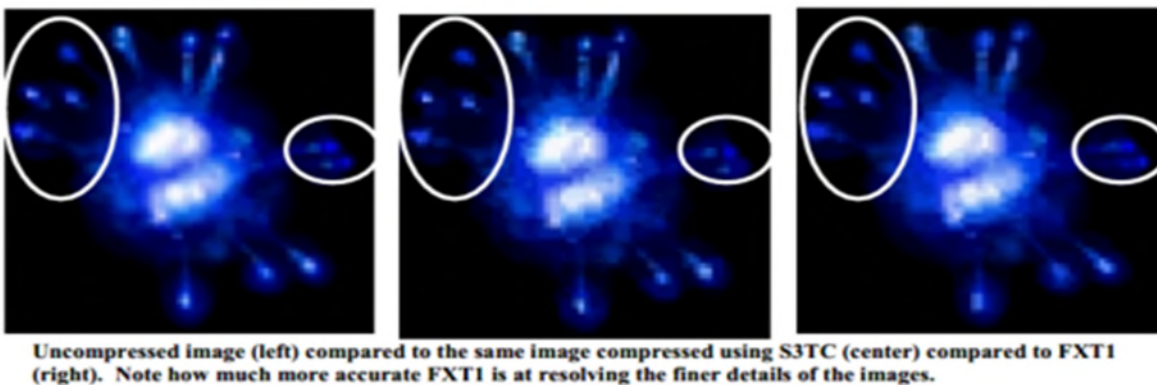
163. FXT1 disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is also my opinion that FXT1 disclosed the additional limitation of claim 11.

164. The requirement of claim 11 is inherent in texture compression, namely, to reduce the data size of an image without distorting the image unacceptably. Thus, the decompressed image may be considered an approximation as recited in claim 11. Furthermore, applying a decompressed image to a rendered surface for display was the intended application of image/texture compression/decompression because images were meant to be displayed — being shown on a display is the reason why they were created in the first place. Specifically, texture is used to project a 3D image onto a 2D surface, to create on a 2D surface the appearance of a 3D object. This was a common POSITA knowledge by the time of the '628 patent. *See, e.g.*, Ex. 1008, ¶ [0052] (“Texture Mapping: Texture mapping incorporates a texture into the surface of an object in an image. Typically, the mapping involves incorporating a two dimensional array of texels into a group pixels representing a three-dimensional surface in the

image.”). Thus, it is inherent that the processor running the FXT1 algorithm is **configured to apply the approximation to a rendered surface for display.**

165. FXT1 confirms this. For example, it disclosed reproducing textures for a rendered scene. Ex. 1005, 2 (“more textures can be used in a rendered scene which can substantially improve overall visual quality. By reducing the memory bandwidth needed for texture transfers and processing, the overall fill-rate of a 3D accelerator is increased, which increases frame-rates and also generates a much more realistic, immersive 3D experience.”); *id.*, 3 (“With limited on-board memory, first and second generation consumer 3D accelerators were limited to texture maps with a maximum resolution of 256x256 texels . . . By utilizing texture compression, higher resolution textures can now be utilized to offer significantly improved visual realism in economical amounts of texture memory . . . Utilizing larger texture maps can significantly enhance the overall visual quality of a rendered scene.”). FXT1 achieves “[h]igher resolution textures for better image quality” in processing 3D images to “create smooth 3D rendering” for display. *Id.*, 4. It further discloses a display/monitor for displaying the rendered textures. *Id.*, 5 (system diagram). Moreover, FXT1 shows decompressed images that are the results of applying the approximation to a rendered surface for display. *See id.*, 7, 8. For example, FXT1 produces a “more accurate” image showing “the finer

details of the images” than S3TC, another texture compression and decompression algorithm:

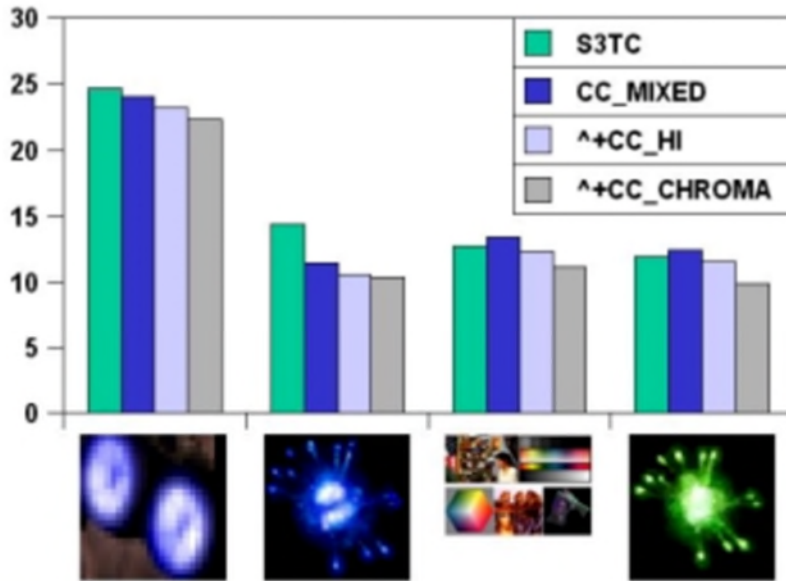


Id., 7. In other words, FXT1 produces closer approximations than S3TC.

F. Dependent Claim 12

166. FXT1 disclosed or at least rendered obvious the computer processing system of claim 7 as I explained above. It is further my opinion that FXT1 rendered obvious the additional limitation of claim 12 to the extent it is argued that claim 12.

167. For example, FXT1 disclosed computing quality using root mean square error of the four compression schemes to select the best of the four per disjoint subset. Ex. 1005, 5 (“In the FXT1™ texture compression scheme, four different compression algorithms are used, with the best one chosen per block to generate the highest quality result.”).



With up to 4 different techniques used to compress each image, FXT1™ provides the most accurate image reproduction when measuring the Root Mean Square error of each encoding algorithm.

Id., 6.

168. The root mean square error constitute a quality parameter, e.g., one that measures “visual quality.” *Id.*, 6.

169. Further, FXT1 disclosed performing real-time compression on larger texture maps. *Id.*, 7 (“All major 3D APIs support FXT1™ texture compression. Textures can be pre-compressed and stored on CD or they can be compressed in real-time by using the provided tools and source code for FXT1™ encoding.”); *id.* (“For developers wishing to do real-time compression in the FXT1™ format, free source code is available for writing an encoder as part of an application.”); *id.*, 2 (“Current titles are using two or more textures per object at resolutions up to 2048x2048 per texture and these requirements will continue to increase as 3D content developers strive for increased realism.”); *id.*, 3 (“Utilizing larger texture

maps can significantly enhance the overall visual quality of a rendered scene.”). A POSITA would have understood that low processing latency is desirable for real-time compression of texture maps. A POSITA would have further found it obvious that lower processing latency could be achieved if, instead of computing 2 to the power of 32 or more than 4 billion combinations of color endpoints and choosing the best one, the algorithm computes a set of color endpoints and its root mean square error, compares the root mean square error of the resulting compressed data with a pre-select threshold, and chooses the set of color endpoints for compression if the error is less than the threshold or performs compression using the next scheme otherwise. In this way, **each of the one or more disjoint subsets is associated with a quality parameter (root mean square error) that met a pre-selected threshold.** Moreover, processing latency can be reduced because not all four billion color combinations need to be computed for each disjoint subset.

170. Doing so would have involved a well-known application of a threshold with predictable results. Further, the threshold may be adjusted based on desired visual quality to ensure a likelihood of success in achieving the desired visual quality. In my opinion, a POSITA would have been motivated to do so to decrease processing latency for real-time compression of large texture maps.

X. GROUND 3: FXTI IN VIEW OF RESHETOV AND THE KNOWLEDGE OF A PERSON OF ORDINARY SKILL IN THE ART RENDER OBVIOUS CLAIMS 12

A. Dependent Claim 12

171. As I explained above, FXT1 rendered obvious claim 12. Additionally or alternatively, it is my opinion that FXT1 in view of Reshetov rendered obvious the additional requirement of claim 12.

172. Reshetov, like FXT1, disclosed using square error as an error function to measure the visual quality of the compressed texture data to represent the original image. Ex. 1008, ¶ [0042] (“Error Function: An error function may be used to quantify the ability of one or more color palettes to represent the original color values for a group of texels. The error function may be composed of a sum of individual texel error contributions. For example, an error function may be the sum (over all texels in a texture block) of the squared minimum difference between each original texel color value and one of the color values from the color palette. The squared minimum difference is used to avoid positive differences negating negative differences, and an absolute value or other means may also be used.”). The sum of square error of Reshetov and mean square error of FXT1 are related and substantively identical in that the mean is simply the sum divided by the number of pixels represented in the sum.

173. Reshetov further disclosed that a color palette, that is, the set of color endpoints used to regenerate the colors of a disjoint texture subset, is generated “typically by iteratively changing color values from the color palette to reduce the error function. *Id.*, ¶ [0045]. An iterative process to generate color endpoints may result in an extended processing latency, especially if the number of iterations is large. *See id.*, ¶ [0085] (“Color values may be changed repeatedly (beyond the initial color values) to further reduce the error function. In one embodiment, a brute-force method such as a Monte Carlo method is used to achieve the optimization, however this method may be computationally demanding relative to other methods.”). In view of FXT1’s real-time compression application (as I explained with respect to claim 12 in the FXT1 ground), a POSITA would have been motivated to apply a pre-select error threshold to terminate the iterative process when the error is at or below an acceptable level to reduce the processing latency. *See id.* (“In one embodiment an optimization technique may be used to reduce the error function. The optimization may minimize the error function or reduce it to an acceptable level by changing one or more color values assigned to texture blocks.”).

174. In my opinion, a POSITA would have been motivated to apply the teaching of Reshetov to FXT1 in the manner above because these teachings of Reshetov applies generally to texture compression. Further, Reshetov cites FXT1

as prior art of which the invention of Reshetov is an extension. *Id.*, ¶¶ [0012]-[0017]. Thus, a POSITA would have understood that these teachings of Reshetov to be applicable to FXT1.

XI. DECLARATION

175. In signing this declaration, I understand that this declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the USPTO.

176. I am aware that willful false statements and the like made by me in connection with this declaration are punishable by fine or imprisonment, or both (18 U.S.C. § 1001) and may jeopardize the validity of the Petition. All statements herein made by me of my own knowledge are true and all statements made on information and belief are believed to be true. I declare under penalty of perjury that the foregoing is true and correct.

Executed this 7 day of March, 2023 at , Washington, DC.

Respectfully Submitted,



Dinesh Manocha, Ph.D.