

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ADVANCED MICRO DEVICES, INC. and PENSANDO SYSTEMS, INC.,

Petitioners,

v.

CONCURRENT VENTURES, LLC and XTREAMEDGE, INC.,

Patent Owners.

Case IPR2025-00486

U.S. Patent 10,873,753

**DECLARATION OF DR. PAUL S. MIN IN SUPPORT OF
PATENT OWNERS' RESPONSE**

TABLE OF CONTENTS

I.	SCOPE AND SUMMARY OF OPINIONS.....	5
V.	PRIORITY DATE OF THE '753 PATENT	14
VI.	LEVEL OF ORDINARY SKILL	15
VII.	TECHNICAL BACKGROUND	16
VIII.	SUMMARY OF THE '753 PATENT	21
	A. Overview of the Patent	21
	B. Prosecution History	24
	C. District Court Allegations	25
IX.	CLAIM CONSTRUCTION	26
	A. “memory superposition” (claims 4, 7, and 15)	26
	B. “interleaved memory” (claims 4, 7, and 15)	27
	C. “swapped memory over a specified address range” (claims 4, 7, and 15)	27
	D. “memory allocated to specified address ranges according to parameterized task needs” (claims 4, 7, and 15)	27
	E. “a module identifier”	28
	F. “sandbox”	30
	G. “selecting a memory superposition” and “selectable arrangement”	32
X.	THE ASSERTED PRIOR ART	34
	A. U.S. Patent Application Publication No. 2013/0346667 A1 (“Stroud”) (EX1004)	34
	B. U.S. Patent No. 9,148,389 (“DaSilva”) (EX1005)	35

C.	“Memory Systems and Pipelined Processors” (“Cragon”) (EX1006).....	35
D.	U.S. Patent Application Publication No. 2002/0087821 (“Saulsbury”) (EX1007).....	36
XI.	PETITIONERS AND DR. ALPERT DO NOT SHOW THAT ANY CHALLENGED CLAIM WOULD HAVE BEEN OBVIOUS	37
A.	Petitioners Fail to Prove their Combination of Stroud and DaSilva (Ground 1) Discloses or Renders Obvious Claims 1, 3, and 7.	37
1.	Stroud Does Not Disclose or Render Obvious a “Module Identifier” or That “Each Module . . . Can Be Addressed Through The Header” As Required By Claims 1[e] and 13[e].	37
2.	Claims 1–3, 13, 14 are Patentable over Stroud and DaSilva.....	47
3.	Stroud Does Not Disclose “At Least One Processor Allocatable for Offloading Hardware Tasks to Software” As Required By Claim 3.	48
4.	Stroud Does Not Disclose a Processor “Executing in Cooperation with a Sandbox of One of the Plurality of Swappable Pods or Cards” As Required by Claim 3.....	66
B.	In my opinion Ground 2 fails because Petitioners have not shown that any of its multiple proposed combinations disclose or render or render obvious the memory limitations of [7d]	72
1.	Neither Stroud nor Cragon teaches, suggests, or renders obvious a single card with each of the four required memory arrangements.....	74
2.	Neither Stroud nor Cragon, or the proposed combination, teaches that memory superpositions can be <i>selected</i>	79
3.	Claims 9-10 are Patentable over Stroud and Cragon.....	81

C.	Grounds 3 and 4 Suffer From the Same Core Problems as Ground 2.....	81
1.	Saulsbury does not teach, alone or in combination, the combination of four distinct memory arrangements on a single card required by limitation [7d].	82
2.	Saulsbury does not teach, alone or in combination, the claimed “selectable arrangement”	84
3.	Claims 9-10 are Patentable over Stroud, Cragon, and Saulsbury.....	85
XII.	SIGNATURE.....	86

I. SCOPE AND SUMMARY OF OPINIONS

1. I, Dr. Paul S. Min, am making this declaration in support of Concurrent Ventures, LLC and XstreamEdge, Inc.’s Patent Owners’ Response in the matter of IPR2025-00486 before the Patent Trial and Appeal Board (PTAB) of the United States Patent and Trademark Office, which involves the Petition for Inter Partes Review (“IPR”) of U.S. Patent No. 10,873,753 (the “’753 patent”) filed by Advanced Micro Devices, Inc. and Pensando Systems, Inc. (“Petitioners”).

2. As explained in detail below, I am qualified to offer opinions from the perspective of a person of ordinary skill in the art (“POSITA”) at the time of the invention and can render an objective opinion regarding the issues raised in this IPR.

3. I have been asked to analyze the Petition (Paper 1) and Dr. Alpert’s declaration (EX1003) in support thereof, as well as the ’753 patent and its file history, and all identified prior art. I have been asked to provide my opinions about certain obviousness theories and opinions in Dr. Alpert’s declaration. I have also been asked to provide opinions about how a POSITA would have understood and interpreted the ’753 patent, including its claims; how a POSITA would have understood and interpreted the Petition’s alleged prior art references; and a POSITA’s knowledge of the state of the art as it relates to addressing these obviousness theories and opinions. My opinions and conclusions are set forth in this

Declaration. I understand that this Declaration is being submitted in connection with Patent Owners' Response to the Petition.

4. For purposes of this Declaration, I was asked to consider claims 1–4, 7–10, and 13–15 of the '753 patent, which I understand are the challenged claims. *See* Pet., 16. I understand that claims 1, 7, and 13 are the independent claims.

5. This Declaration and its exhibits, in addition to my education, training, and experience in technology fields pertinent to the '753 patent, form the basis of my opinion.

6. I am not offering any conclusions as to the ultimate determinations that I understand the Patent Trial and Appeal Board ("Board") will make in this proceeding. I am solely providing my opinions on the technical interpretations, characterizations, theories, and opinions that I understand are involved in this proceeding.

7. I may offer additional opinions based on my further review of the materials in this case, including opinions and/or testimony of other expert declarants.

II. MATERIALS CONSIDERED

8. In forming the opinions in this Declaration, I reviewed the Petition (Paper 1), Dr. Alpert's declaration (EX1003), the materials considered by Dr. Alpert (EX1003, Appx. 3), the prior art and supporting materials cited by Petitioners, the

'753 patent, and the '753 patent's file history. I have also cited various other materials throughout my declaration that I considered as part of my analysis.

III. QUALIFICATIONS AND PROFESSIONAL BACKGROUND

9. I believe that I am well qualified to serve as a technical expert in this matter based upon my educational and work experience. My curriculum vitae ("CV") is submitted as Exhibit 2022.

10. I received a B.S. degree in Electrical Engineering in 1982, an M.S. degree in Electrical Engineering in 1984, and a Ph.D. degree in Electrical Engineering in 1987 from the University of Michigan in Ann Arbor. I received several academic honors, including my B.S. degree with honors, a best graduate student award and a best teaching assistant award during my M.S. study, and a best paper award from a major international conference for reporting results from my Ph.D. thesis.

11. After receiving my Ph.D., I worked at Bellcore in New Jersey from August 1987 until August 1990. At Bellcore, I was responsible for evolving the public switched telephone network (POTS) into a multi-services voice and data network that incorporated packet switches, optical technologies, and wireless technologies.

12. In September 1990, I joined the faculty at Washington University in St. Louis. In July 1996, I was promoted to an Associate Professor of Electrical

Engineering with tenure. I am currently a Senior Professor at Washington University of Electrical and Systems Engineering. I have also served as the Chair of the Graduate Curriculum (2000-2002) and the Chair of the Undergraduate Curriculum (2011-2014) for the Electrical and Systems Engineering department.

13. At Washington University, I have conducted research in communication, computing, and related electronic hardware and software. My research group has pioneered a new paradigm for designing electronic circuits that can alleviate the speed and performance mismatch against optical technology. I have received several grants from the U.S. Federal Agencies, including the National Science Foundation and the Defense Advanced Research Project Agency, and numerous contracts from companies and organizations around the world. As part of my research activities, I have designed and implemented a number of System-on-Chip (SoC) devices, which incorporated multiple processor-cores that communicate and cooperate with others.

14. As a faculty member at Washington University, I have taught a number of courses in electronics, communication, and computing at both the undergraduate and graduate levels. For example, I have taught communication theory (Washington University ESE 471), transmission and multiplexing (Washington University ESE 571), and signaling and control of communication networks (Washington University ESE 572).

15. I have supervised numerous graduate students, many of whom received a doctoral degree under my guidance. A number of doctoral theses that I have supervised relate specifically to the computing and communication technologies related to this proceeding. My students and I have published a number of peer-reviewed articles in these topics.

16. In addition to my responsibilities as a university faculty member, I have founded two companies. In May 1997, I founded MinMax Technologies, Inc., a fabless semiconductor company that developed switch fabric integrated circuit chips for the Internet. In March 1999, I founded Erlang Technology, Inc., a fabless semiconductor company that focused on the design and development of integrated circuit chips and software for the Internet. One of Erlang's products received a best product of the year award in 2004 from a major trade journal for the electronics industry. As part of my company activities, I have prototyped numerous chassis systems, which included multiple modular cards performing various tasks (e.g., switching card, memory card, input/output card, power supply card, etc.). In doing so, I became intimately familiar with various module standards such as CompactPCI and ATCA (Advanced Telecom Computing Architecture).

17. I developed and implemented a number of semiconductor integrated circuit chips. These integrated circuit chips include switch fabrics, network processors, protocol processor high speed interfaces, protocol transposers,

controllers, content search engines, look up table processors, and memory cells and arrays.

18. I am a named inventor on eleven U.S. patents, many of which are directly related to computing and communication technologies. I have extensively published technical papers in international conferences and journals, technical memoranda and reports, and given a number of seminars and invited talks. I have organized several international conferences and served as an international journal editor.

19. I am a member of and have been actively involved in a number of professional organizations. For example, I have served as the Chair of the Saint Louis Section of the IEEE with more than 3,000 members (2014), and a member of the Eta Kappa Nu Honor Society for electrical engineers. I have also been an Ambassador of the McDonnell International Scholars Academy (2007-2013).

20. I was a member of the Presidential Business Advisors Committee (to President George W. Bush) and was named 2002 Businessman of the Year by the Wall Street Journal for the State of Missouri, for my entrepreneurial efforts.

21. I have received multiple awards for my publications and contributions to technical communication systems. In 1988, I received the Best Paper Award at 18th ISATA Award of Technical Excellence in Florence, Italy. I was also a Rockwell Fellow at the University of Michigan in Ann Arbor in 1988 and 1989. I received a

Research Initiation Award in 1993 from the Defense Advance Research Project Agency (DARPA). I received the Best Paper Award at MOBILITY 2011 in Barcelona, Spain. In my nearly 40 years of professional experience, I have acquired significant knowledge about telecommunications, computer, and electronics industry standards and standard setting organizations such as ANSI, IEEE, ITU, IETF, Bellcore, and 3GPP.

22. Additional details of my education and work experience, awards and honors, and publications that may be relevant to the opinions I have formed are set forth in my curriculum vitae, as well as a list of cases in which I have testified as an expert at trial or by deposition. *See* EX2022.

IV. UNDERSTANDING OF LEGAL PRINCIPLES

23. At the outset, I want to clarify that all of my analysis and opinions in this declaration are from the perspective of a POSITA at the time of invention. To the extent that portions of this declaration do not expressly include language indicating that the analysis is from the perspective of a POSITA at the time of invention, it should be understood that my analysis and opinions are from the perspective of a person of ordinary skill in the art at the time of invention. For example, my opinion that a reference does not disclose or teach a claim limitation should be understood to mean that, in my opinion, a POSITA at the time of invention would have interpreted the reference as not disclosing the limitation. Likewise, my

opinions regarding obviousness are from the perspective of a POSITA at the time of the invention.

24. I do not offer opinions of law. The opinions I am expressing in this Declaration involve the application of my knowledge and experience to the evaluation of the challenged patent and certain art with respect to the challenged. The following paragraphs express my understanding of legal principles related to my analysis.

A. Obviousness

25. I understand that a claim is unpatentable under 35 U.S.C. § 103 if the differences between the claimed subject matter and the prior art are such that the claimed subject matter as a whole would have been obvious to a person of ordinary skill in the art at the time of the invention. I also understand that an obviousness analysis is based on the scope and content of the prior art, the differences between the claimed subject matter and the prior art, the level of ordinary skill in the pertinent art at the time of the invention, and objective indicia of nonobviousness, such as commercial success, industry praise, unexpected results, copying, and other such factors.

26. It is my understanding that, to find a claimed invention obvious in light of a combination of prior-art references, a person of ordinary skill in the art at the time of the invention must have had reason to combine the teachings of the prior-art

references to achieve the claimed invention, and that the person of ordinary skill must have had a reasonable expectation of succeeding in combining the teachings to arrive at the claimed invention. I further understand that a reason to combine or expectation of success is undermined if the combination or modification of the technology disclosed in the prior art would interfere with an objective of the technology disclosed in the prior art or if the prior art teaches away from making the combination or modification. I understand that a prior art reference is said to teach away from the claimed invention if a person of ordinary skill in the art at the time of the invention, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken in the claim.

27. I understand that an assessment of what a reference discloses or teaches, for purposes of an obviousness analysis, must be conducted from the perspective of a person of ordinary skill in the art at the time of the invention. In other words, a reference discloses or teaches a claim limitation only if a person of ordinary skill in the art, at the time of the invention, would have interpreted the reference as disclosing the claim limitation. I understand that reliance on common sense, common wisdom, and common knowledge to inform an obviousness analysis must be explained with sufficient reasoning. I understand that assessment of obviousness

from the perspective of someone skilled in the art before the invention was made, without the benefit of hindsight.

28. I understand that it is not permissible to use hindsight after viewing the claimed invention to determine the question of obviousness or to rely on the teachings of the claimed invention in determining whether a POSITA would find the invention obvious.

B. Claim Construction

29. I understand that claims and their terms are construed, that is, interpreted, in accordance with their ordinary and customary meaning as understood by a person of ordinary skill in the art at the time of the invention when read in the context of the patent's specification and prosecution history. I further understand that if the patent's specification reveals a special definition given to a claim term by the inventor, then the inventor's special definition governs, even if it differs from the term's ordinary meaning.

30. I understand that there are certain terms for construction that are at issue in this proceeding. I have discussed these terms and my understanding of them further below.

V. PRIORITY DATE OF THE '753 PATENT

31. Dr. Alpert states that the time of the invention is July 15, 2018, which is the filing date of the '753 patent. For the purpose of my opinions in this

Declaration, I have been asked to assume the '753 patent is entitled to that priority date. Therefore, when I refer to a POSITA at the time of the invention in my Declaration, I am referring to a person of ordinary skill in the art as of July 15, 2018.

VI. LEVEL OF ORDINARY SKILL

32. Dr. Alpert set forth a level of ordinary skill as follows:

I came to the conclusion that a person of ordinary skill in the field of art of the '753 patent would have been a person with a bachelor's degree in electrical engineering, computer engineering, or a similar major with at least two years of experience in the field (described above) or a person with a master's degree in electrical engineering, computer engineering, or a similar major with a specialization in the field. A person with less education but more relevant practical experience may also meet this standard.

EX1003, ¶37.

33. I understand that the following factors may be considered in determining the appropriate level of skill in the art: (1) the educational level of the inventor; (2) type of problems encountered in the art; (3) prior art solutions to those problems; (4) rapidity with which innovations are made; (5) sophistication of the technology; and (6) educational level of active workers in the field. I understand that the person of ordinary skill in the art is not an actual individual, but instead is a construct for a hypothetical person who is presumed to be aware of all the pertinent prior art. I have addressed relevant information regarding these factors and applied this understanding below.

34. As summarized above with regards to my qualifications and experience, I possessed at least the level of a POSITA at the time of the invention of the '753 patent because I had a Ph.D., M.S., and B.S. in Electrical Engineering. EX2021. I am currently a Senior Professor in the Department of Electrical Systems Engineering at Washington University, St. Louis, and have worked in the field of electrical engineering and data processing, streaming architecture, and data networks for more than thirty years. *Id.*

VII. TECHNICAL BACKGROUND

35. At the time of the '753 patent's invention, most high-performance computing and data-processing systems followed a processor-centric architectural model. In these systems, one or more general-purpose CPUs controlled system operation, executed software under an operating system, which controlled and coordinated peripheral devices, task accelerators, and I/O components. Communication between components would typically occur through shared memory, memory-mapped registers, interrupts, or buses. Even when systems handled packetized data externally, such as Ethernet or IP traffic, those packets were often terminated at the network interface and converted into software data structures managed by the CPU.

36. By contrast, internal packet-based communication fabrics within a computing platform were not well understood or widely adopted. Designers did not

commonly treat internal hardware modules as independently addressable endpoints communicating through a messaging interface network. Instead, accelerators like field programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs) were usually invoked indirectly and operated under the control of a CPU. Packetization was viewed as adding latency and complexity inside a system chassis, making it unsuitable for fine-grained coordination of internal components.

37. Relatedly, the art did not generally contemplate uniform packet headers capable of identifying and routing messages to specific hardware or software elements within a system. While external networks used layered protocols and addressing schemes—like the Ethernet layers discussed by Dr. Alpert (EX1003, ¶¶7–13)—internal systems relied on static wiring, bus arbitration, or memory addresses rather than message headers carrying hierarchical identifiers. As a result, internal communication was typically rigid, difficult to reconfigure, and closely tied to a particular hardware layout.

38. Dr. Alpert’s recitation of the technical background in this case does not sufficiently address this gap in the prior art concerning internal packet-based communication. He discusses that “[p]acket-based networks have been the dominant form of computer networking for decades,” but the rest of his discussion only concerns server-to-server connections in a blade server system. EX1003, ¶¶7–8. Even though prior art servers sent higher-level packets over Ethernet, most

commonly using the TCP/IP, designers did not commonly treat internal hardware modules as independently addressable endpoints inside the blade servers.

39. Including a module identifier in a packet enables a level of flexibility that conventional mechanisms could not provide. By decoupling communication from fixed wiring or static memory addresses, modules could be added, removed, replaced, or reconfigured without requiring system-wide redesign or hard-coded routing assumptions. A module identifier also supports scalable system composition where packets can be dynamically directed to the appropriate module based on configuration or workload needs. This enables load balancing, redundancy, and scaling. As system components evolve over time, the packet structure remains the same, insulating data flows from changes in the hardware layout.

40. Embedding the module identifier in the packet also allows the system to treat hardware and software elements uniformly as addressable endpoints. This unification simplifies orchestration across diverse processing resources and enables flexible data-flow architectures in which functionality can be redistributed across modules as requirements change. In this way, the module identifier is a key enabler of adaptability, extensibility, and long-term system evolution, and ensures that packets can be routed and addressed with maximum flexibility and accuracy.

41. Reconfigurable hardware, including those using sandboxes, further illustrates a gap in the art. Although programmable logic devices were known, they

were most often deployed as fixed-function accelerators or controlled through unique interfaces. The idea that reconfigurable hardware regions could be dynamically instantiated, isolated, and treated as peers in a message-based internal network—communicating directly with other hardware and software elements without CPU mediation—was not part of the conventional design approach. Instead, flexibility was achieved primarily in software, not through reconfigurable, network-addressable hardware modules.

42. At the same time, offloading from hardware to software was not a common or expected design pattern in data-processing systems. The prevailing paradigm was the exact opposite: software tasks were offloaded to hardware accelerators to improve performance, reduce latency, or lower CPU utilization. Hardware was viewed as the preferred execution environment for specialized functions, while software orchestrated those functions. Once a task was committed to hardware—like an FPGA or ASIC—it was generally expected to remain there for the duration of processing.

43. Designs in the art did not typically contemplate dynamic migration or delegation of work from hardware back to software based on workload characteristics or resource availability. Instead, hardware and software occupied relatively rigid roles, with clear boundaries and one-directional control flow. This rigidity reflected both architectural assumptions and practical constraints, including

the lack of flexible communication mechanisms and the overhead associated with transferring state or context from hardware into software execution environments.

44. As a result, systems generally lacked the infrastructure needed to support efficient hardware-to-software offloading, such as standardized messaging, specific addressing of processing elements, or mechanisms for transferring intermediate results without excessive latency. The notion that a processor could initiate or trigger moving a hardware task from a hardware element to a software processing element as part of an integrated data flow ran counter to conventional design principles. This underscores why architecture that supports bidirectional offloading between hardware and software represented a departure from the prevailing art rather than a routine or obvious extension of existing systems.

45. In short, prevailing systems emphasized centralized control, static interconnects, software-mediated coordination, and unidirectional dataflows between hardware and software elements. The concept of an internal, packet-based communication fabric that directly interconnects hardware and software components, allowing them to be addressed, orchestrated, and reconfigured through messages instead of processor-centric control, was not well-known or established in the art.

VIII. SUMMARY OF THE '753 PATENT

A. Overview of the Patent

46. The '753 patent recognized a problem in the art relating to data flow processing architectures as of the patent's effective filing date. Software-based systems offered flexibility and ease of use but were often inferior to hardware-based systems with respect to processing power. EX1001, 1:8–11. Pure hardware-based systems, in contrast, offered superior processing speeds but were not a desirable alternative because “redesigning dedicated hardware each time a new data processing application or data formatting requirement comes up is an expensive proposition.” *Id.*, 1:11–14. Further, “designing, debugging and manufacturing dedicated electronic hardware is time-consuming, especially when revision cycles are taken into account.” *Id.*, 1:14–16

47. To address this problem in the art, the inventors of the '753 patent developed an innovative hybrid data-processing platform that offers the processing speed advantage of hardware while retaining much of the flexibility of software-based systems. Specifically, the inventors conceived of an architecture that utilizes one or more swappable pods or cards that include a portion for user-definable hardware modules or user-definable software modules, with the pods or cards communicatively linked via a messaging interface network. The goal of the system was maximum flexibility. For example, the invention taught a module identifier such

that each individual module inside the system could be uniquely addressable by the management switch, and sandboxes within FPGAs so users could install custom logic tailored to their system. For another example, the instance identifier also allowed each module to be identifiable and addressable, this time through a unique value assigned to that instance out of all modules of that type. These innovations are reflected in the claims of the '753 patent.

48. Dr. Alpert opines that “[t]he purportedly novel feature of the '753 patent is the use of several identifiers in the packet header.” EX1003, ¶¶21–24. It is my opinion that this understates the invention. Prior art systems were lacking in flexibility where they involved sending packets to multiple pods and cards on multiple chassis. The packet-based messaging system claimed by the '753 system is tailored to allow maximum flexibility for users building a data flow system, such as swappable pods with modules containing unique identifiers.

49. It is my opinion that the novel, hardware-defined anything system comprising this invention is distinct from a standard blade server system and was not disclosed in the prior art. Dr. Alpert contends that it was common for a blade server chassis to include swappable cards or pods connected through a backplane using Ethernet, and the user of packet headers and memory types was disclosed in the prior art, but this is not seen in the references submitted by Petitioner.

50. The '753 patent also discloses substantial innovations in the use of memory superposition. As the '753 patent explains, a variety of memory types were known as of the priority date, such as “such as SRAM (static random access memory) 144, DRAM (dynamic random access memory) 146, FRAM (ferroelectric random-access memory) 148, RLDRAM (reduced latency dynamic random access memory) 150, [and] flash 152 (electrically erasable programmable read-only memory or EEPROM).” EX1001, 7:2-8. “Memory types” refer to different forms of memory, such as RAM and ROM. The '753 patent recognizes that “[e]ach type of memory as an optimal use” (EX1001, 7:21) and contemplates a system of memory superposition in which multiple types of memory are “made available for a data flow or other data processing operation.” EX1001, 6:65-7:15. The claims require selection of a memory superposition to occur not during the design of the platform, but during runtime (configuration and operation).

51. Through the process of memory superposition, diverse types of memory are utilized together in different ways with the goal of optimizing memory performance for particular tasks—creating distinct memory arrangements. “Memory arrangements” are different ways of arranging those forms of memory, such as “interleaving” the RAM and ROM together. A wide variety of memory arrangements were known as of the priority date of the '753 patent, but the '753 patent describes and claims a memory arrangement configuration that was not

present in the prior art: a “selectable arrangement” of four different memory arrangements, each of which is present and available for use on a single card. EX1001, 14:25-32. The four specific memory arrangements identified are “multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.” *Id.*; *see also* 8:26-47. The inventors of the ’753 patent determined that it was particularly advantageous to combine these four memory arrangements on a single pod or card and allow them to be selected. As the ’753 patent further explains, the end result of the use of the memory arrangements contemplated by the ’753 patent is complementary, overlapping integration of distinct memory types that optimizes performance for a particular processing task: “[a]nalogous to the overlapping of waves to form a final observed wave in wave superposition, where each wave contributes to the whole, in this case different types of memory are being overlapped, each with benefits and drawbacks, to form a final memory system which performs optimally in a given video stream or other data flow/data processing task.” EX1001, 7:40-46.

B. Prosecution History

52. Dr. Alpert discusses that, during prosecution, the as-filed dependent claims 2, 9, and 16 were allowable because “the closest prior art of record, either alone or in combination with other prior art of record, does not teach, suggest, or

disclose packets each having *a header with a chassis identifier, a board identifier, a module identifier, an instance identifier, and a type identifier, so that each type of module, each instance of a type of module, and each module on each board in each chassis can be addressed through the header.*” EX1002, 000090. The Office also indicated that as-filed dependent claims 5, 12, and 18 are allowable because “prior art of record, does not teach, suggest, or disclose at least one of the one or more swappable pods or cards having a plurality of types of memory and memory superposition with selectable arrangement of the plurality of types of memory as multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.” EX1002, 000091.

53. Dr. Alpert uses this prosecution history to suggest that the Applicant “simply incorporated the subject matter indicated as allowable by the Examiner into the independent claims,” but this prosecution history only supports my opinion that the Office viewed the subject matter of these claims as patentable. EX1003, ¶¶25–27.

C. District Court Allegations

54. Dr. Alpert repeats certain infringement allegations from the related district court litigation, but he does not tether these claims to his opinion on the prior art. Based on my reading, I disagree with the limited scope of his summary, namely

his summarizing hundreds of pages of infringement contentions into no more than five paragraphs.

IX. CLAIM CONSTRUCTION

55. In my opinion, a POSITA would have interpreted the following claim terms, when read in the context of the '753 patent specification and prosecution history, to have their plain and ordinary meaning. Except for the specific instances discussed below where I have applied certain construction proposed by Petitioner for purposes of evaluating the Petition, I have applied the plain meaning of the claims in forming my opinions. In my opinion, all claim construction positions that Patent Owner and I have proposed are consistent with and reflect the plain meaning of the terms in question.

56. I understand that there are certain terms for construction that are at issue in this proceeding. I have discussed these terms and my understanding of them below.

A. “memory superposition” (claims 4, 7, and 15)

57. Dr. Alpert states:

“In the parallel district court proceeding, Patent Owner has proposed the construction for this term as ‘two or more types of memory that can be made available for a data flow or other data processing operation.’ Ex-1036, 000003. It is my understanding that Petitioners have adopted this construction for the purposes of this Petition. Therefore, I have applied this construction to my analysis of the prior art.

58. I also apply this construction to my analysis of the prior art.

B. “interleaved memory” (claims 4, 7, and 15)

59. Dr. Alpert states that “a POSITA would have understood that ‘interleaved memory’ means an arrangement of different types of memory that are alternately accessed to achieve an average access bandwidth.” EX1003, ¶43.

60. It is my understanding that Patent Owners have adopted this construction for the purposes of this Response. Therefore, I have applied this construction to my analysis of the prior art.

C. “swapped memory over a specified address range” (claims 4, 7, and 15)

61. Dr. Alpert states that “a POSITA would have understood that ‘swapped memory over a specified address range’ means memory with a specified address range where information is moved between the memory and non-volatile storage.” EX1003, ¶47.

62. It is my understanding that Patent Owners have adopted this construction for the purposes of this Response. Therefore, I have applied this construction to my analysis of the prior art.

D. “memory allocated to specified address ranges according to parameterized task needs” (claims 4, 7, and 15)

63. Dr. Alpert states that “a POSITA would have understood that ‘memory allocated to specified address ranges according to parameterized task needs’ means different types of memory allocated to specified address ranges according to parameters for task needs.” EX1003, ¶52.

64. It is my understanding that Patent Owners have adopted this construction for the purposes of this Response. Therefore, I have applied this construction to my analysis of the prior art.

E. “a module identifier”

65. Based on my review of the claims and specification of the ’753 patent, it is my opinion that a POSITA would have understood that “a module identifier” means “a value or field in the packet header that allows each module to be identified.” This is apparent from the structure of the claims.

66. Claims 1 and 13 require a “type identifier,” “instance identifier,” “module identifier,” “board identifier,” and “chassis identifier.” Correspondingly, the claims require that these identifiers are provided “so that each type of module, each instance of a type of module, and each module on each board in each chassis can be addressed through the header.” Thus, it is clear from the claim language itself that the “module identifier” must enable “each module” to “be address[able]”—just as the “instance identifier” must enable “each instance” to be “be address[able]” and so forth.

67. It is my opinion that the plain language of claim 1 requires that the module identifier allows each module to be identified. Petitioners and Dr. Alpert, however, appear to assume that a module identifier need only identify some modules rather than all. Pet., 42–45; EX1003, ¶¶117–19. I believe that is incorrect.

68. Patent Owners’ construction is directly from the claim language, which requires a packet-based communication system where “each module on each board in each chassis can be addressed through the header.” EX1001, c. 1. Thus, the claim defines that the “module identifier” must be capable of addressing each module within the system, not simply a certain type of module within the system. This is a key purpose of the invention.

69. The specification of the ’753 patent also explains that “[a] header on Module Message Interface packets specifies chassis, board, module, instance and type identifiers, so that packets in the MMI network are routed to a hardware or software module in a pod or card in the chassis or one of multiple chassis.” EX1001, 3:4–8. It states that the module identifier in each packet “could be static (typically, for hardware), or dynamic (typically, for software), indicating that the module identifier is specific to each module itself. *Id.*, 9:19–20. If the module identifier did not identify each module, packets could not be addressable to each module in the system.

70. From my review of his deposition, Dr. Alpert agreed with this construction. He testified that the module identifier refers to each hardware or software module on the pod or card. EX2024, 60:20–22 (“My understanding is that the module identifier would -- would identify a specific one of the hardware or software modules.”); 70:8–10 (“Yes, it should. Yeah, it does”). He said that each

module on the card must be addressable, even those that are not user definable. *Id.*, 78:9–12 (“No, as I’m -- as I’m reading the claim, it would refer to the hardware modules and software modules of 1B, not only to the user-definable hardware modules or user-definable software modules of 1C.”); 61:22–62:3 (“And the module ID in the header would identify a module and in particular so that the -- the module can be addressed through the header”).

71. The module identifier must identify each module. If the Board were to adopt Petitioners’ reading of Stroud, it would substantially limit the full scope of the claim.

F. “sandbox”

72. Based on my review of the claims and specification of the ’753 patent, it is my opinion that a POSITA would have understood that “sandbox” means “a partitioned, designated region of a programmable device.”

73. As a POSITA, I believe that “sandbox” as a term of art with a settled meaning, particularly when viewed in light of the intrinsic record. The plain language of the claims makes clear that it denotes a customizable region inside of a configurable circuit that is different than a general FPGA. The claims elsewhere refer to “user-definable hardware modules or user-definable software modules,” thus a “sandbox” is distinct element within the system. EX1001, cl. 1.

74. The ’753 patent’s specification distinguishes a PLD from a sandbox:

A programmable device for data flow processing in a user-configurable server with swappable pods or cards is provided. The device includes a field programmable logic device (PLD) ***having a first region and a second region***. The first region comprises a hardware-based or firmware-based router with a port enabling communication between the router and the second region, and the ***second region*** comprises one or more sandboxes with user-definable electronic circuits.

Id., 1:36–44 (emphasis added). “In some versions, the bridge 128 is locked to prevent user access, in an FPGA 122 that supports partial reconfiguration, so that a user can configure or reconfigure one or more modules in one or more sandboxes 132, but cannot reconfigure the bridge 128.” *Id.*, 5:30–34. Thus, each PLD (*i.e.*, CLD) in the system may have a “region,” the second comprising a sandbox with user-definable electronic circuits. *Id.*

75. Dr. Alpert did not propose a definition of “sandbox” in this proceeding. However, I understand that he admitted that he did not apply a POSITA’s definition of sandbox to the claims, rather he used a preliminary definition discussed by Patent Owners when exchanging proposed claim construction terms in district court. EX2024, 124:13–17.

76. The plain and ordinary meaning of “sandbox” best aligns with Patent Owners’ definition. Patent Owners suggested that “sandbox” should be limited to, at the very least, meaning a “portion designated for programmable electronic circuits.” EX1036, 000004. The parties ultimately agreed that “sandbox” need not

be construed in district court and did not include the term in their claim construction brief.

77. Dr. Alpert’s opinion on Stroud would read out the sandbox limitation. I observe that Petitioners contend that Stroud discloses a processor in cooperation with a sandbox simply because it discloses that the system contains a CLD, meaning that a CLD, without isolated regions, is a sandbox simply because it is configurable. Pet., 59.

78. As I discuss in more detail below, I do not agree with this argument. The ’753 patent states that the sandbox is *a region* of the PLD—not the PLD as a whole, and this is how a POSITA would understand the term “sandbox” in the art. PLDs are always configurable, so Dr. Alpert’s position amounts to arguing that every PLD in the world is a sandbox.

79. Because “sandbox” is a term of art denoting a partitioned, designated region of a programmable device reserved as a user-reconfigurable region for implementing user-definable electronic circuits and hardware or software modules, a POSITA would not equate it with a CLD/PLD/FPGA that does not have isolated regions.

G. “selecting a memory superposition” and “selectable arrangement”

80. In my opinion, the terms “selecting a memory superposition” and “selectable arrangement” require that selection occur during the “method for data

flow processing in the platform.” That understanding is consistent with the plain meaning a POSITA would give the terms in the context of the claims and specification.

81. Claim 7 is a “method for data flow processing in a platform” and includes the step of “communicating a data flow” and “selecting a memory superposition.” EX1001, cl. 7. Thus a POSITA would understand that it is directed to the *use*, not original design, of a data processing “platform.” This is consistent with the specification, which indicates that memory superposition selection is an active process that occurs during system configuration or operation. *See, e.g.*, EX1001, 8:22-46 (explaining that memory superposition permits memory structures to be “reconfigured for subsequent tasks”), 7:21-56 (explaining that in memory superposition “different types of memory are being overlapped, each with benefits and drawbacks, to form a final memory system which performs optimally in a given video stream or other data flow/data processing task”). In other words, a POSITA would understand that selection of a memory superposition occur not during the design of the platform, but during runtime (*i.e.*, configuration and operation).

82. A broad interpretation of the claims that would encompass the design phase is inconsistent with the language, specification, and common sense. A POSITA would not understand “selecting” to occur during the design phase, because that would mean that a “selectable arrangement” is just an arrangement that was

available to select during the design phase. Further, all memory arrangements would be “selected” because at some point a designer choose them. Such a broad reading would essentially nullify the patentable significance of the claim language.

X. THE ASSERTED PRIOR ART

A. U.S. Patent Application Publication No. 2013/0346667 A1 (“Stroud”) (EX1004)

83. I have reviewed the Stroud reference.

84. Stroud is a patent application directed to “systems and methods for testing communications networks, services, and devices, e.g., testing the traffic-handling performance and/or security of the network, network accessible devices, cloud services, and data center services.” EX1004, ¶0001.

85. Stroud discusses an increased cyber security threat environment and the importance of network security testing, EX1004, ¶¶2–3, and notes the importance of “content-aware and software-driven packet inspection,” *id.*, ¶5.

86. To address these issues, Stroud proposes a “network testing system 16” that “may be configured to test the performance (e.g., traffic-handling performance) of devices 14, the security of a test system 18 (e.g., from security attacks), or both the performance and security of a test system 18.” *Id.*, ¶81. Stroud does not discuss or contemplate large-scale data storage or suggest that its teachings would be relevant to data center design.

87. Communication between certain internal components in Stroud's system occurs via "backplane 56." *Id.*, ¶101. This communication can occur via an internal Ethernet that is not connected to external networks and thus does not require that each component have a "globally unique" MAC address. *Id.*, ¶366. Stroud discloses that, "in some embodiments, each network processor 105 and CLD in system 16 automatically derives a 6-byte MAC address for itself" utilizing a specific format. *Id.*, ¶366. As discussed below, this six-field format for internal Ethernet communications does not include the "module identifier" required by the claims.

B. U.S. Patent No. 9,148,389 ("DaSilva") (EX1005)

88. I have reviewed the DaSilva reference.

89. I understand that the Petition relies on DaSilva for its teachings regarding a "chassis ID." Pet., 19 (citing EX1005, Abstract).

C. "Memory Systems and Pipelined Processors" ("Cragon") (EX1006)

90. I have reviewed the Cragon reference.

91. Cragon is a 1996 computer textbook that provides an overview of a variety of general concepts and considerations relating to memory systems as of that date.

92. I understand that the Petition relies on Cragon to supply the "selectable arrangement" of "types of memory" required by claim element 7[d], as well as the specific memory arrangements enumerated in the claim. Pet., 73-78.

93. It is my opinion that Cragon discusses the advantages and disadvantages of a variety of memory arrangements at a high level, but does not disclose using any of the arrangements together on a single card in a selectable arrangement.

94. To the contrary, Cragon states that the principal portion of Cragon that Dr. Alpert and Petitioners rely upon for a disclosure of memory arrangements is “discussed in terms of a peripheral device.” EX1006, 000558. This is not surprising as Cragon is focused on the memory hardware that existed in 1996 and before, which was dominated by large external disk systems that seem archaic now (and would have to a POSITA as of 2018). *See, e.g.*, EX1006, 000534 (using a “5-1/4” floppy” as example of “disk organization hierarchy”). A POSITA in 2018 looking to develop a flexible, modular information processing system would have seen relatively little value in Cragon’s discussion of obsolete memory hardware from decades past, and in my opinion would not have viewed Cragon as of more historical interest than as a resource for system development.

D. U.S. Patent Application Publication No. 2002/0087821 (“Saulsbury”) (EX1007)

95. I have reviewed the Saulsbury reference.

96. I understand that Petitioner relies on Saulsbury in Ground 3 to supply the “selectable arrangement” of “types of memory” required by claim element 7[d]. Pet., 84-85.

97. It is my opinion that Petitioners rely on Saulsbury because of the shortfalls I discuss of their Cragon-based theory for this limitation.

98. As discussed below, it is also my opinion Saulsbury does not teach any capability to select memory arrangements, much less selecting between the specific memory arrangements required by the claims, and does not resolve the defects in the Stroud/Cragon combination.

XI. PETITIONERS AND DR. ALPERT DO NOT SHOW THAT ANY CHALLENGED CLAIM WOULD HAVE BEEN OBVIOUS

A. Petitioners Fail to Prove their Combination of Stroud and DaSilva (Ground 1) Discloses or Renders Obvious Claims 1, 3, and 7.

1. Stroud Does Not Disclose or Render Obvious a “Module Identifier” or That “Each Module . . . Can Be Addressed Through The Header” As Required By Claims 1[e] and 13[e].

99. In my opinion, Petitioners have not shown that Stroud discloses a “module identifier” or that “each module . . . can be addressed through the header” as required by claim elements 1[e] and 13[e]. Petitioners argue that Stroud’s disclosure of the “6th Byte” of the six-byte Ethernet MAC address—the “interface identifier”—discloses the “module identifier,” but each module in Stroud does not have a unique interface with Management Switch 110, so it cannot be uniquely identified. Pet., 42–43, 46 (citing EX1004, ¶372).

100. Petitioners acknowledge that Stroud’s interface identifier only “identifies modules by their interface to the management switch,” but each module in Stroud does not have a unique interface with Management Switch 110—Stroud

discloses multiple that do not. Pet., 42; EX1004, ¶372. Stroud discloses multiple modules that do not have direct interfaces to Management Switch 110. At most, the interface identifier that Petitioners point to only identifies *some* modules.

101. Stroud teaches that the 6th byte in the packet header “indicates processor interface (each interface to the management switch has its own MAC address).” EX1004, ¶372. Petitioners do not identify any other sentence in Stroud that describes the 6th byte, but this sentence does not indicate that it corresponds to each module on each board that packets will use to reach their correct location. EX1001, 8:65–67.

102. The ’753 patent makes clear that the module identifier must be capable of directly identifying each software or hardware module on the chip to achieve the flexible, swappable pod architecture of the system. EX1001, 8:61–65. The ’753 patent uses the example of an MMI packet “destined to chassis number two, board number three in that chassis, module number seven on that board, instance number three of that module on that board, and the module being a hardware type.” *Id.*

103. Nearly every computer system has modules that communicate with management switches, but for this system, if the identifier was only capable of identifying a processor that is directly connected to Management Switch 110, configuration packets would not be able to directly reach a module behind another module. Other systems typically rely on implicit addressing, such as ports, buses, or

control paths, where modules are reachable only through predefined connections rather than being independently and uniformly addressable entities.

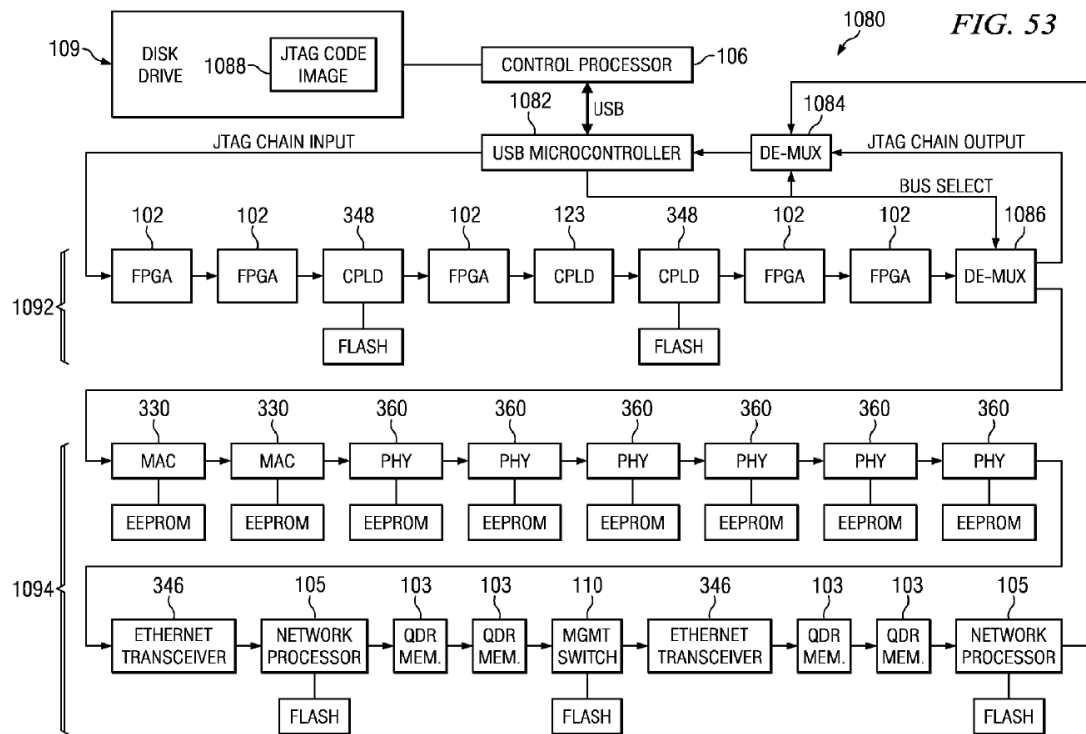
104. A module identifier is important because it names the actual functional destination of the message, allowing packets to be routed directly to a specific hardware or software module even when multiple modules share the same physical interface or port to Management Switch 110. An interface identifier, by contrast, only identifies a point of attachment to the switch, which requires additional logic to determine which downstream module should receive the message and limits flexibility as modules are added, removed, or reconfigured.

105. I do not believe that Stroud discloses a module identifier. Paragraph 372 says nothing about modules, much less that they have a unique MAC address (or otherwise disclose a “module identifier”). Stroud only shows that it is the interface to Management Switch 110 that has its own MAC address. EX1004, ¶372

106. There are several modules in Stroud that do not have a direct connection to Management Switch 110. XLR 105, CPLD Power Control Reset Timing 348, and FPGA Routing 102B are all modules. However, not all have a direct connection back to Management Switch 110. Because they do not have a direct connection to Management Switch 110, they would not be identified by the interface identifier. XLR 105 and CPLD 348 are not directly connected to Management Switch 110 in Figure 14B:

refer to the hardware modules and software modules of 1B, not only to the user-definable hardware modules or user-definable software modules of 1C.”).

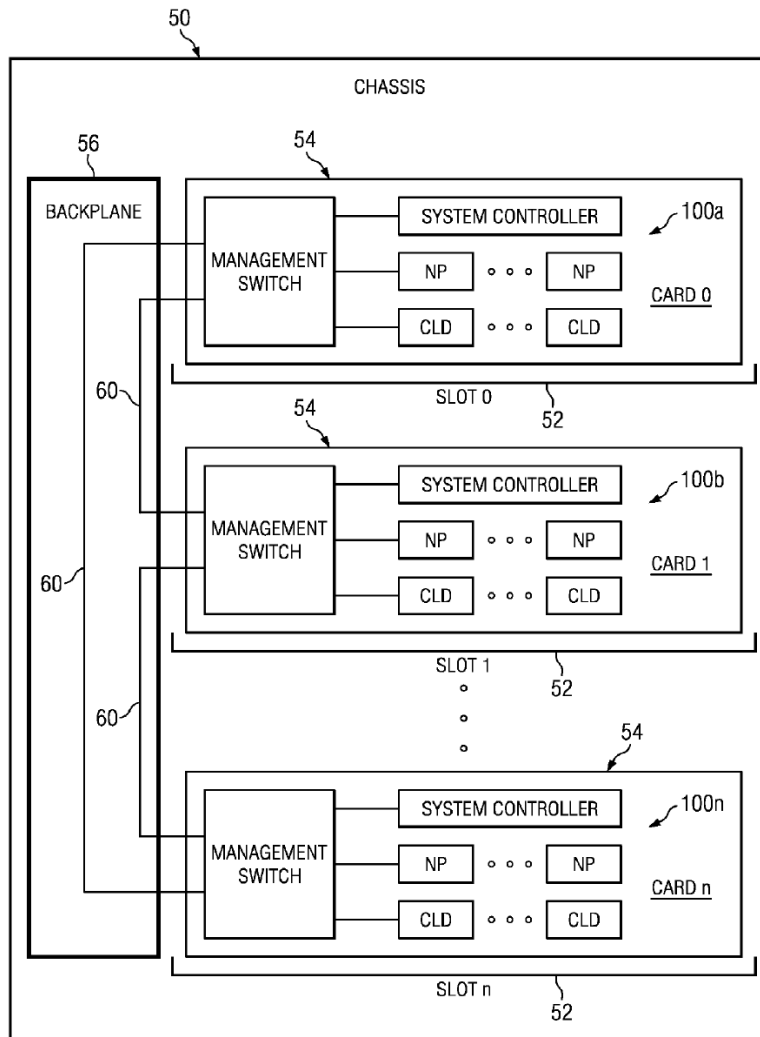
108. Upon my review of Stroud, I observed that Figure 53 shows numerous modules chained together sitting between each other without a direct connection to Management Switch 110:



Id., FIG. 53; *see also id.*, Fig. 23 (showing the Capture/Offload CLD 102A, Routing CLD 102B, Network Processor 106 and Control Processor 105—all of which Petitioners identify as modules—between the Test Ports 101 and the disk drive 109, without interfaces to the Management Switch 110). CPLD 348 may also provide reset and power sequencing for network processor 105.”).

109. It is clear to me that CPLD Power Control Reset Timing, depicted in Figure 14B, is an important module in the system. Stroud describes that “CPLD 348 may provide glue logic between network processor 205 and boot flash 326 (e.g., because the network processor may be capable of interfacing flash memory directly)” and “may also provide reset and power sequencing for network processor 105.” EX1004, ¶309. Its responsibilities includes providing coordinated power-up, reset sequencing, and fault recovery for the hardware on a blade or card. For example, CPLD 348 can be configured to reset power-sequencing parameters. Stroud itself is clear that FPGAs and CPLDs are all modules. FPGAs and CPLDs are both hardware modules in Stroud.

110. There are many examples in Stroud showing that modules are not directly connected to Management Switch 110, such that the interface identifier is insufficient to identify each module. Figure 3 is another example which shows that the first layer of NPs and CLDs are directly connected to Management Switch 110, but there can be more than one NP and CLD chained together:



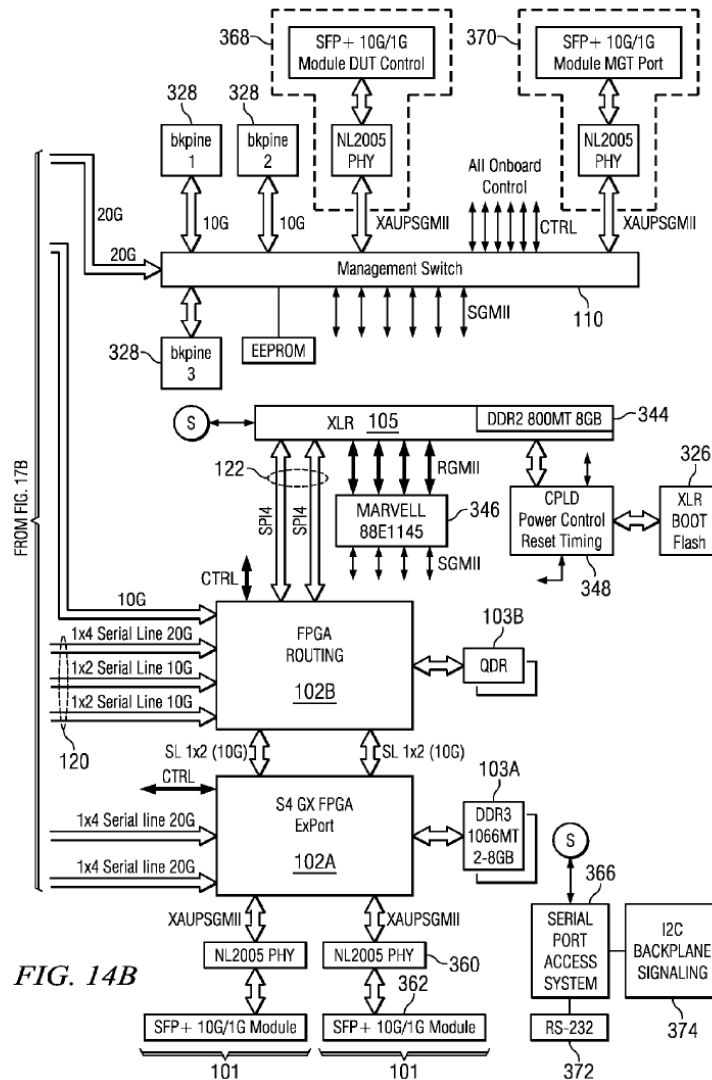
EX1004, FIG. 3. Similarly, in Figure 15 the Nehalem Control is a module, Pet., 26 (identifying “controller 106” as a software module), that does not have an interface with the Management Switch 110.

111. Stroud also discusses that network processors can connect to other devices. In my opinion, this configuration is common and shows that there may be other modules downstream from the network processors that cannot be routed simply by the interface identifier. EX1004, ¶310 (“Network processor 105 provides

four parallel Ethernet ports, e.g., RGMII ports, for communicating with other devices via the Ethernet protocol.”); *see also id.*, Fig. 20 (showing that the “network processor 105”—which Petitioners identify as a module—does not interface with the Management Switch 110 but goes through desk drive 109 and in particular, DHCP server 300).

112. I do not find any of Petitioners’ citations to indicate otherwise. Pet., 43-44 (citing EX1004, ¶100, ¶101, ¶103, ¶113, ¶¶117-118, ¶134, ¶¶146-147, ¶¶156-157, ¶167, ¶¶174-176, ¶189, ¶310, ¶¶314-319, ¶354, ¶366, Figs. 3-5, 6A-9A). For each of these citations, Petitioners appear to merely demonstrate that certain modules within Stroud have some level of connection, and some modules may be connected to Management Switch 110. *See, e.g.*, EX1004, ¶¶117–118. None of their citations say anything about interface connections or how an interface connection is synonymous with a module identifier. For example, ¶¶100, 101, 103, and 113 merely list different components that may be included in the system without regard to whether they can be addressed by the module identifier. Paragraphs 117–118 do the same; they merely discuss that there are multiple CLDs within the system which have relationships to other components, but do not discuss anything about interface connections. EX1004, ¶¶117–118. This is true with every citation listed above. Even where Stroud discusses that the management connects to the CLD, it does not teach that this is a direction such that an interface identifier is akin to a module identifier.

113. Petitioners and Dr. Alpert do not reconcile that a module identifier and interface identifier are different. The '753 patent disclose a flexible, modular system, requiring each module to be directly addressable in case cards are swapped out, while Stroud's interface identifier only identifies modules that are directly connected to Management Switch 110. EX1001, 13:10–13 (“As protocols 10 change, components in the system can be swapped, upgraded or reconfigured but will still provide an advantage in hardware over software-only or software-centric systems”). The interface identifier in Stroud specifically allows for configuration packets that flow into the system via the backplane and progress to modules via Management Switch 110, but not directly to modules downstream of Management Switch 110:



EX1004, FIG. 14B (depicting that certain modules, like CPLD power control reset timing 348, sit behind other modules, such that an interface identifier could not identify that specific module).

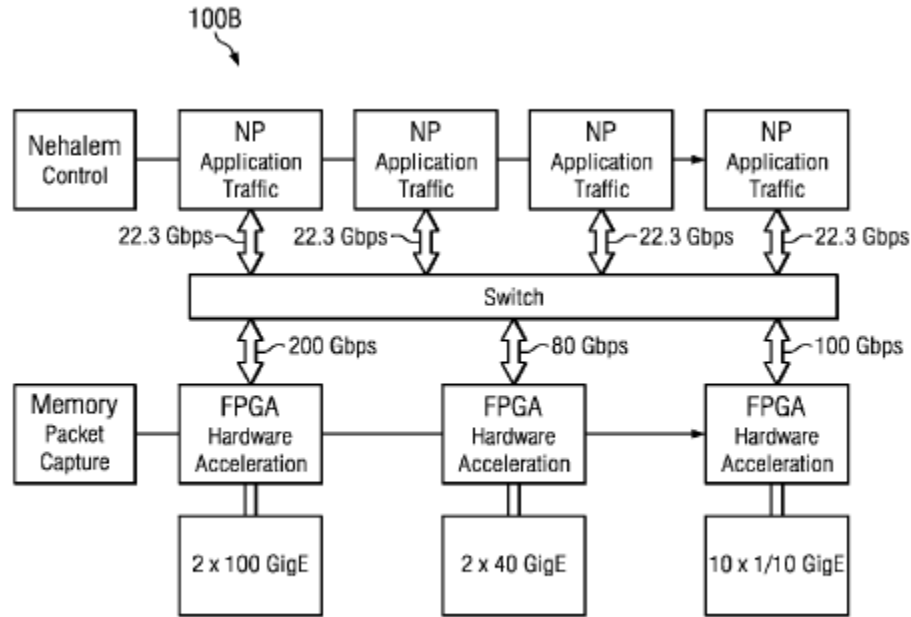


FIG. 15

Id., FIG. 15; *see also id.*, Fig. 20 (showing that the “network processor 105”—which Petitioners identify as a module—does not interface with the Management Switch 110 but goes through desk drive 109 and in particular, DHCP server 300).

114. Petitioners contend that Stroud’s interface identifier is interchangeable with the module identifier because the interface identifier is “unique to the module within each card,” Pet., 42–43 (citing EX1004, ¶372), but Stroud makes clear that this is simply incorrect. Nothing in Stroud indicates that the 6th byte in its internal Ethernet MAC format identifies a module.

2. Claims 1–3, 13, 14 are Patentable over Stroud and DaSilva.

115. Because independent claims 1 and 13 are patentable over Stroud and DaSilva, the more specific dependent claims are also patentable for the same reasons.

3. Stroud Does Not Disclose “At Least One Processor Allocatable for Offloading Hardware Tasks to Software” As Required By Claim 3.

116. In my opinion, Petitioners have not shown that Stroud discloses “at least one processor allocatable for offloading hardware tasks to software.” EX1001, cl. 3. Petitioners contend that Stroud teaches this limitation because its control processor 106 configures routing CLD 102B to perform “load balancing” between network processors. Pet., 57.

117. Petitioners do not demonstrate that Stroud discloses that the claimed “processor” is allocatable for this claimed “offloading” because Stroud only shows a system for routing software packets to software processor. It is my opinion that “load balancing” is substantially different than moving a task to software that was originally intended to be executed in hardware.

118. Based on my review of the ’753 patent, claim 3 requires that the claimed “processor” can optimize packet processing by offloading packets that are originally assigned to hardware units can instead be processed by software, based on available processing power. EX1001, 2:67–3:3. This process is captured in claim 3’s requirement that the claimed “processor” is “allocatable for offloading hardware tasks to software.” *Id.*, cl. 3.

119. The ’753 patent teaches that “the workload can be offloaded from a programmable logic device [which is hardware] to a processor [running software]

for tasks that can be done serially and/or slower.” EX1001, 2:67–3:3; 10:1–3 (“One or more software processing elements 174 may optionally be deeply embedded within a data flow path in some embodiments.”). This process is captured in claim 3’s requirement that the claimed “processor” is “allocatable for offloading hardware tasks to software.” *Id.*, cl. 3.

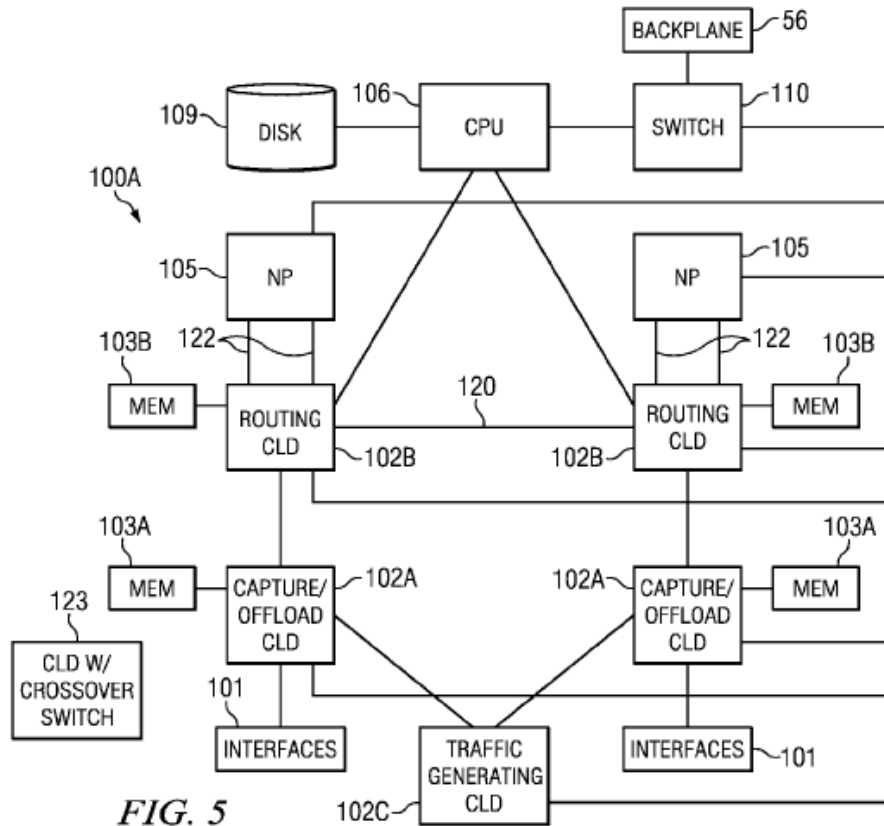
120. First, I note that the Petition’s arguments regarding the “offloading” element are internally inconsistent. Petitioners and Dr. Alpert appear to allege that the claimed “processor” in Stroud is the “controller 106 which comprises a ‘management processor.’” Pet., 57 (“The controller runs software to control and configure the CLDs and NPs.”). But when alleging that Stroud discloses the claimed “offloading,” the Petition points to actions performed by a *different* device altogether: CLD 102B, also referred to as the “routing CLD.”

121. Stroud’s disclosure of “[i]ncoming packet routing and load balancing” does not meet the claimed “offloading.” Pet., 57 (citing EX1004, ¶¶245-248, ¶321, Fig. 15). Stroud clearly discloses that that “routing and load balancing” is performed by CLD 102B—not what Petitioners identify as the claimed “processor” (“controller 106 which comprises a ‘management processor’”). *See, e.g.*, EX1004, ¶241, ¶¶245–248, ¶¶410–11, ¶459; *compare* ¶157 (discussing the “controller 106” *without* disclosing anything about routing and load balancing”) *with* ¶¶128–143 (explaining

that is it the “Router CLDs 102B” that exclusively perform the routing and load balancing); EX2024, 110:12-111:6.

122. Stroud shows that CLD 102B is explicitly referred to as the “*router* CLD”—demonstrating Stroud’s consistent disclosure that it is the “router CLD,” not “controller 106” or any subparts that perform the routing and load balancing. EX1004, ¶241, ¶¶245–248, ¶¶410–11, ¶459.

123. Stroud does not teach a system where tasks are executed by both CLDs and NPs—tasks are only ever executed by network processors. Stroud discloses a “networking testing system” that includes a control processor 106, network processors 105, routing CLDs 102B, capture/offload CLDs 102A, capture buffers 103A, and test ports/interfaces 101. Figure 5 illustrates the packet processing and routing system of the network testing system:

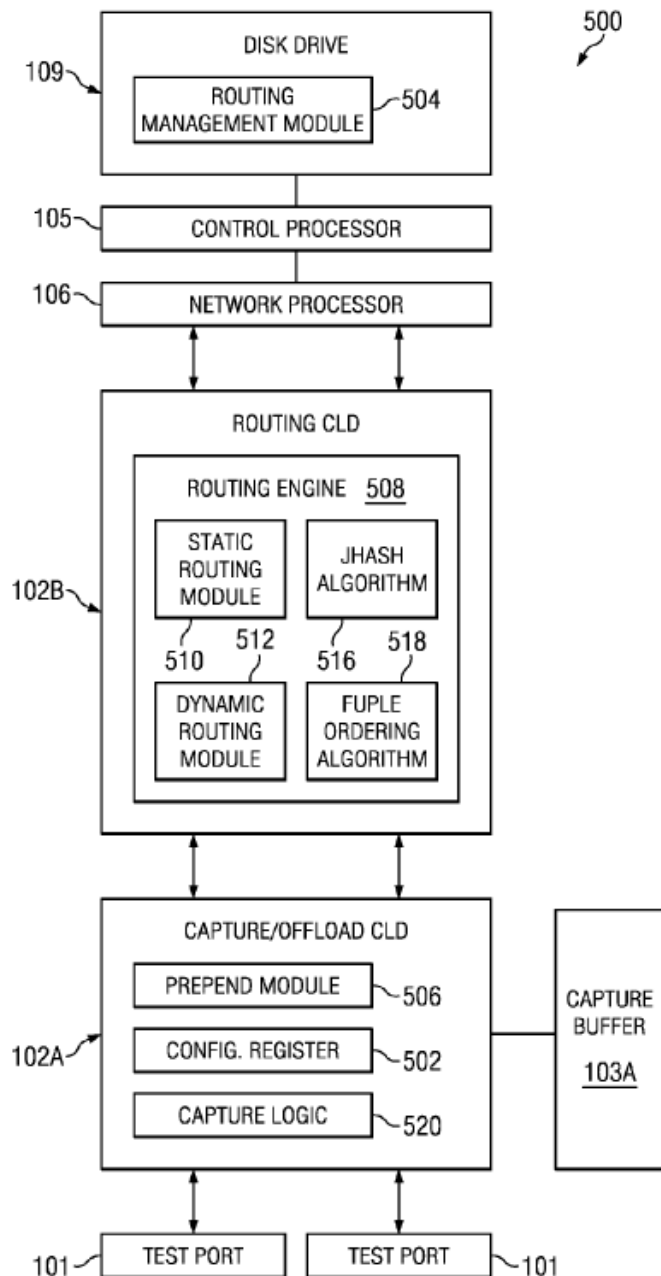


EX1004, FIG. 5. Packets flow into the system via test interfaces 101 that connect the internal testing architecture to the external system to be tested. *Id.*, ¶¶114–16. These interfaces connect directly to capture/offload CLDs 102A within the system which “are the first and last device in the packet-processing pipeline” and may save a copy of the packet to memory or forwarding the packet onto a different CLD. *Id.*, ¶¶119–124.

124. After the packets are captured, capture/offload CLDs 102A may forward the packet to routing CLD 102B for dispersal throughout the system, namely “for sending traffic to and from the test interfaces 101” to a network processor 105. *Id.*, ¶131. These network processors 105 perform a variety of tasks on the packets,

including “user and application-layer simulation.” *Id.*, ¶¶144–53. Controller 106 sits outside this packet-flow pathway and serves as the user interface for managing the system or an auxiliary network processor. *Id.*, ¶¶154–60.

125. Petitioners identify control processor 106 as the “at least one processor” required by the claim limitation, but control processor 106 does not offload tasks from hardware to software when a hardware element becomes overloaded. Control processor 106 only configures CLDs and NPs during system design time and serves as the user interface for managing the system. EX1004, ¶¶245–248. Figure 23 illustrates a simplified view of the system where packets flow in from a test port 101 to their final destination at the network processor:



Id., FIG. 23. Control processor 106 does not participate in assigning the packet to a particular destination, rather this is done based on an algorithm by the routing CLD 102B. In one embodiment, control processor 106 may act as an auxiliary processor to execute tasks, but it never assumes the routing functionality of the hardware elements. *Id.* at ¶154. A POSITA would understand that a controller functioning as

an auxiliary network processor simply means that it acts as a backup network processor—that is, it would only take on the overflow tasks that would originally be assigned to a network processor.

126. Figure 24 also illustrates that no hardware element originally tasked with processing a packet will redirect that packet to software at the instruction of control processor 106:

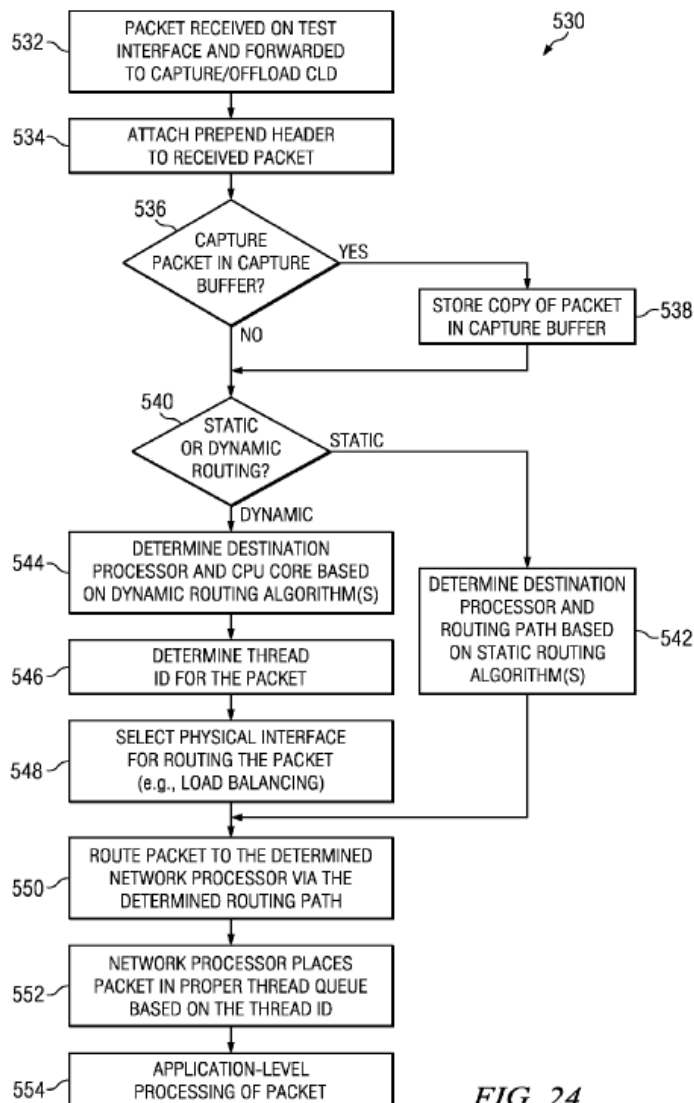


FIG. 24

Id., FIG. 24. Packets are received on the test interface and automatically forwarded to the capture/offload CLD, which is a hardware element that attaches headers to assist with traffic management. *Id.* Forwarding a packet from a processor to a router with the aim of it being processed by software is not a form of offloading a hardware task to software.

127. Routing CLD 102B is not capable of resolving tasks in the first place, rather the packets are always destined for execution by a network processor. In fact, Stroud does not disclose any embodiment in which packets are processed by hardware; the entire system is designed to forward packets to network processors. EX1004, ¶¶409–10. For example, capture/offload CLD 102A attaches a header to the packet, and routing CLD 102B applies a routing algorithm, but it does not have any ability to resolve the task. *Id.*, FIG. 24. The reason why Stroud teaches this implementation of load balancing instead of offloading is because other modules within the system prohibited the system from using conventional techniques to route packets. Stroud teaches that “each network processor 105 and CLD in system 16 automatically derives a 6-byte MAC address for itself” where each byte indicates a location like slot number and chip type. EX1004, ¶¶366–372. Because “[c]onventional load-balancing systems rely on some characteristic of each incoming packet to be unique, such as the IP or Ethernet address,” when those bytes are used for another purpose, the system cannot make efficient use of multiple

processors. *Id.*, ¶245. The role of hardware in Stroud is to compensate for this unique identification, not to process packets.

128. Nowhere does Stroud show a processor that offloads hardware tasks from a hardware module to a software processor, rather each of the FPGAs within the system play a capture or routing role with the ultimate goal of sending the packets to network processors for execution.

129. I understand that Dr. Alpert made clear during his deposition that CLD 102B is solely responsible for the identified routing and load balancing. *Id.* (Q. “Stroud generally discloses that its routing and load balancing is done by that routing CLD 102B, right?” A. “Yeah, these paragraphs of Stroud do say that the CLD 102B is doing the load -- the routing and load balancing.” Q. “And that’s true for Stroud generally, right?” A. “Yes. That’s true for Stroud, and there’s just a number of different places that 102B is identified in the figures.”), 116:14–117:2. I agree with this because the routing and load balancing in Stroud is performed by CLD 102.

130. Claim 3 of the ’753 patent requires a **single** processor that is allocatable for all four claimed functions; alleging that one device performs three functions and another device performs the fourth function is not consistent with the invention. Petitioners and Dr. Alpert do this by accusing both controller 106 and routing CLD 102B. Pet., 57, 58–59; EX2024, 115:15–116:13 (admitting the “[i]ncoming packet routing and load balancing” is performed by the CLD 102B).

131. Another reason I disagree with Petitioners is that Stroud only discloses routing and load balancing *software* tasks. It does not disclose any *offloading* of *hardware* tasks *to* software. This is required by the '753 patent and, as discussed above, offloading from hardware to software was a key part of this invention that was not common in the prior art.

132. Stroud makes clear that “[i]ncoming packet routing and load balancing enable multiple *network processors 105* to be used efficiently in a single system.” EX1004, ¶245. Stroud discloses “[i]ncoming packet routing and load balancing,” but that the routing is *not* between both hardware and software devices. *Id.* Petitioners only show routing tasks between software elements—Stroud’s “[i]ncoming packet routing and load balancing” has nothing to do with offloading from hardware to software devices. EX1004, ¶245.

133. Routing between software elements does not disclose offloading hardware tasks from hardware elements to software. In my opinion, “offloading” means reassigning from a first component meant to perform the task to a second component meant to perform the task. Dr. Alpert appears to agree with this definition, testifying that offloading hardware tasks to software “means transferring a task that’s been previously assigned to hardware to be performed by software instead.” EX2024, 108:2–12 (adding that it is meant “to remove the burden from the hardware”). The claims require offloading from a hardware element capable of

executing the task to a software element capable of executing the task—load balancing between software elements does not disclose that functionality.

134. In my opinion, it would not have been obvious to offload hardware tasks to software when the hardware elements were at capacity because this is the opposite direction that tasks are typically offloaded. I disagree with Petitioners that “when were [sic] CLDs near capacity, it was obvious to offload hardware tasks to a NP, including the controller which acts ‘as an auxiliary network processor.’ Pet., 57 (citing EX1004, ¶154).

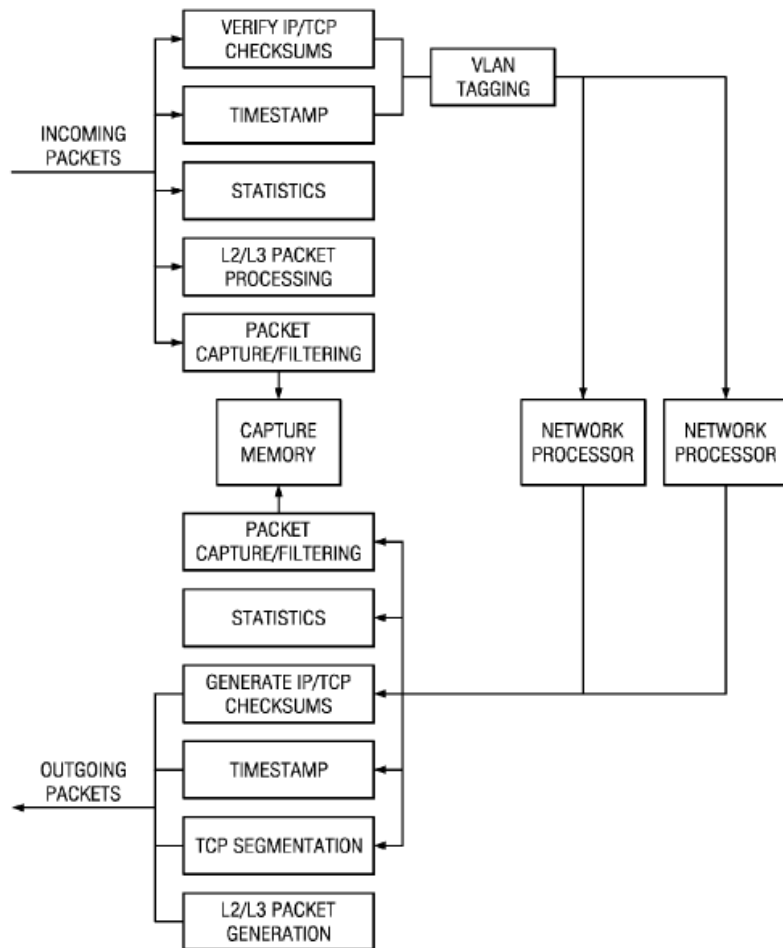
135. Stroud’s suggestion that workloads are portable between NPs and CLDs does not show offloading. Tasks can be routed between different elements on a card, but offloading means moving a task from one element to another to alleviate the burden on that element. I do not see any evidence from Petitioners that this was common or obvious, nor do I see discussion from Dr. Alpert beyond his general and unsupported assertion that “[i]t was common to offload complicated tasks from hardware to avoid clogging the pipeline.” *Id.* (citing EX1003, ¶151).

136. Paragraph 154 of Stroud does not demonstrate that “it was obvious to offload hardware tasks to a NP” Pet., 57. This paragraph only describes that “[c]ontroller 106 may compare [sic] any suitable controller programmed to control various functions of system architecture 100A.” EX1004, ¶296. There is no discussion in this paragraph about offloading from hardware to software, only that

“controller 106 may be a general purpose CPU with multiple cores” and “can also be used for other tasks in the system 100A.” *Id.* Stroud lists all of the tasks of the control processor 106 in ¶¶154–163, none of which include offloading to itself or to other network processors based on hardware load. EX1004, ¶¶154–163.

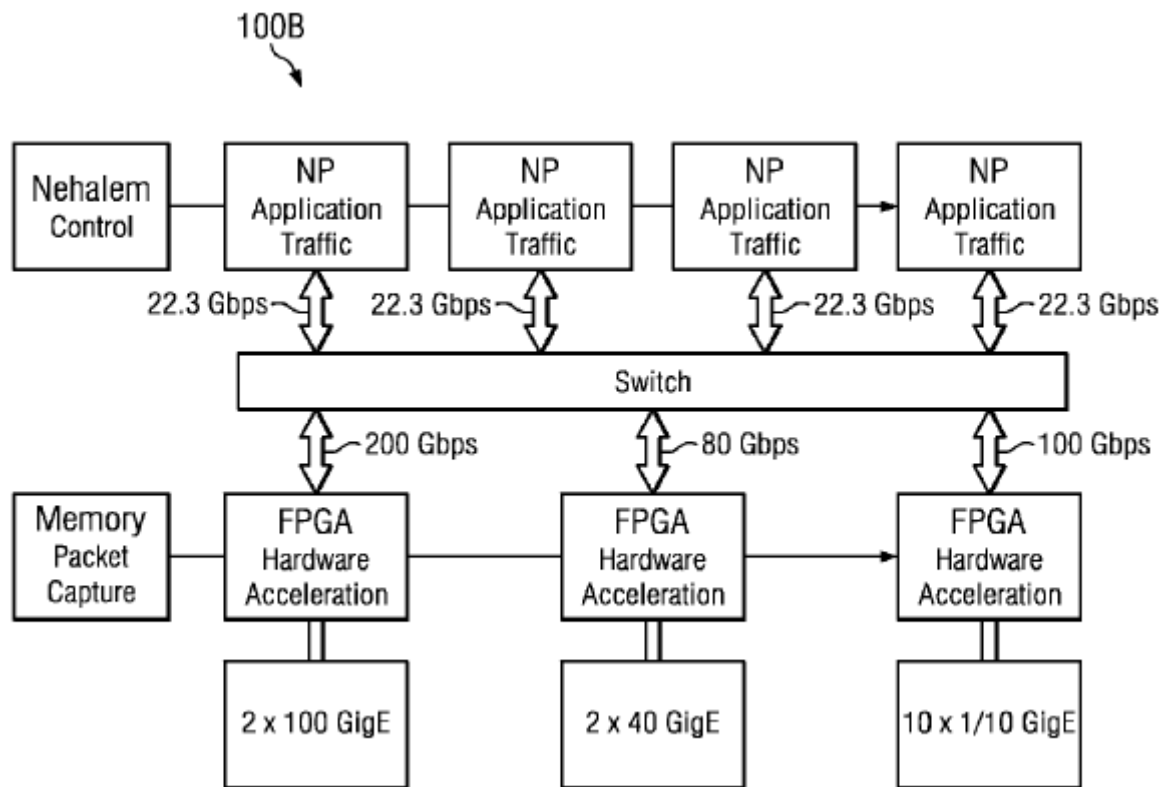
137. Claim 3 requires that a processor orchestrates offloading from hardware to software to remove the burden from hardware, not simply that tasks can move between modules on a board. EX1001, cl. 1. Stroud teaches that “the capture CLD and L2/L3 CLD may be configured to offload a portion of traffic, e.g., 25%, from the network processors,” but this does not disclose or suggest offloading hardware tasks to software. EX1004, ¶296. If anything, it shows the opposite as a portion of traffic is offloaded to other hardware and is not in response to another element becoming overloaded.

138. I do not see evidence anywhere in Stroud supporting the claimed offloading. The paragraphs describing Figure 12 that Petitioners cite only suggest that “[i]ncoming packets, also called ingress packets, arriving on external interfaces may be processed by one or more of several core functional modules in high-speed configurable logic devices,” and their output “may be fed into one or more network processors along with the ingress packet.” *Id.*, ¶¶282–94. Figure 12 makes clear that incoming packets are received and forwarded to a set of network processors for completion, only using hardware elements to facilitate this routing. *Id.*, ¶406.

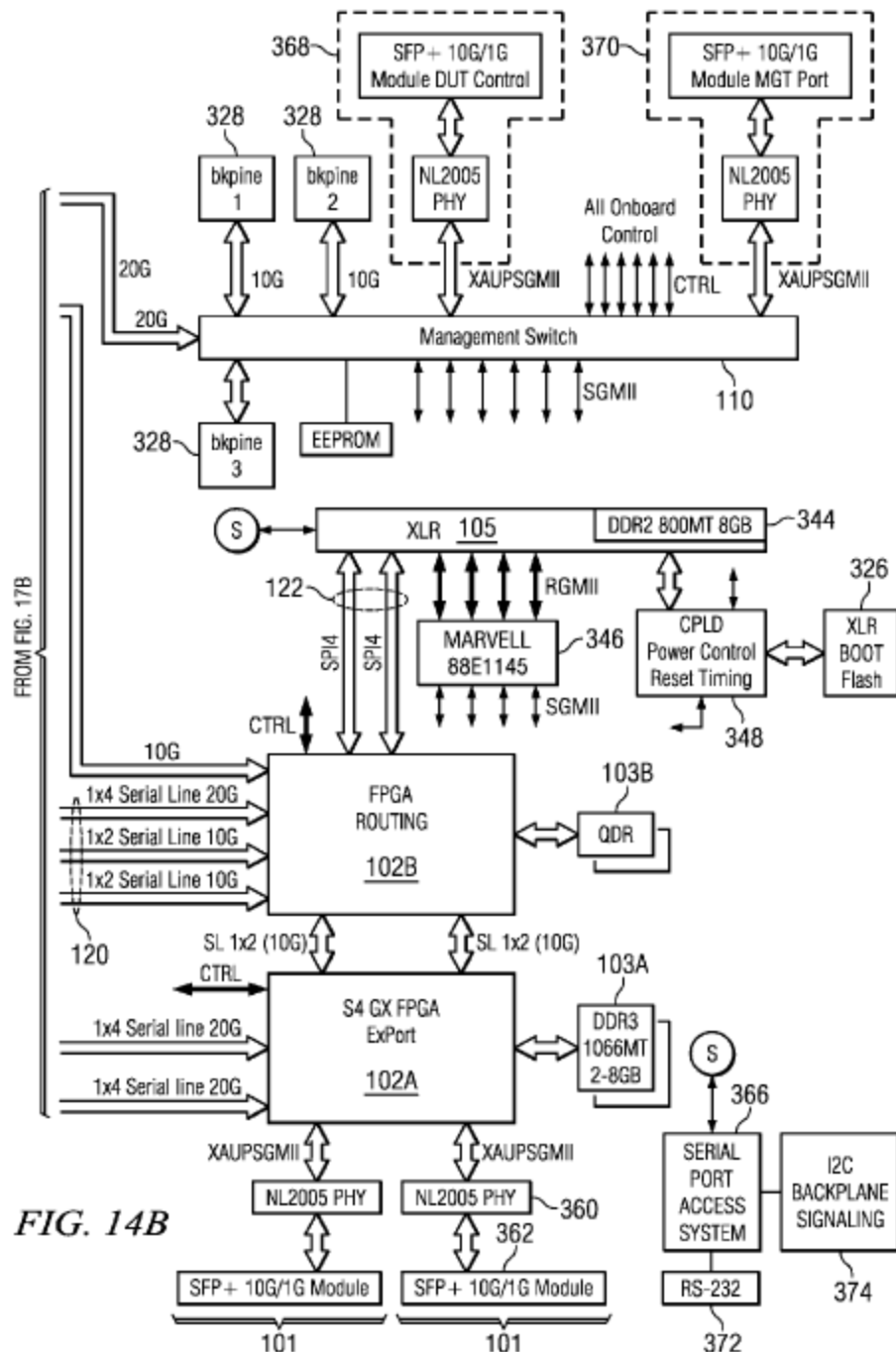


Id., FIG. 12. The packet may be buffered by a CLD, but this is only a temporary storage location and distinct from executing the task of the packet as the system “may then proceed to the process for routing the packet to a network processor 105.” *Id.*, ¶¶408, 412 (“Once the static or dynamic routing determinations are made as discussed above, routing CLD 102B may then route packet to the determined network processor 105 over the determined routing path (e.g., physical interface(s)) at step 550.”).

139. Paragraphs 245–48 only show how control processor 106 configures CLDs and NPs during system design time and serves as the user interface for managing the system, nothing related to offloading from hardware to software. EX1004, ¶¶245–248. The same is true for Figure 15 and ¶321, which only show that the system contains a multitude of different CLDs and NPs. EX1004, FIG. 15, ¶321. These sections describe an alternative system architecture encompassing “two additional network processors and a total of 14 FPGAs coordinated with a single high-bandwidth Ethernet switch.” EX1004, FIG. 15, ¶321. Petitioners use this alternative embodiment to suggest that the presence of both NPs and FPGAs means that the controller processor can perform offloading, Pet., 57, but they ignore that this alternative embodiment still does not disclose moving tasks from FPGAs to NPs when they FPGAs are overloaded. In fact, when asked about this during his deposition, Dr. Alpert admitted that the only difference he could identify in this embodiment was the presence of more hardware and processor elements. EX2024, 111:2–6.



EX1004, FIG. 15. This is bolstered by the fact that the additional FPGA circuits are all connected to a memory packet capture module, showing that these FPGAs are still responsible for packet capturing and routing. Stroud further illustrates the limited routing functionality in its most detailed figure:



Id., FIG. 14B.

140. The '753 patent discloses something fundamentally different than the load balancing described in Stroud. The purpose of “offloading” in the '753 patent is to “maximize the total system performance” by moving “tasks that can be done

serially and/or slower” *from* hardware to software. EX1001, 2:67–3:3; 6:44–45. When an FPGA or CLD is overloaded, the system can free hardware resources by moving simpler or slower operations to a software element. *Id.*, 2:67–3:3. The patent also describes that “[s]ystem performance is maximized with hybrid hardware and software data processing,” reflecting that a purpose of the invention is resource management. *Id.*, 6:44–45.

141. Software may be used to configure hardware modules by defining their parameters, control registers, or logic states so the hardware performs a desired function, but the actual data processing remains executed by the hardware itself. This differs from offloading, where a task that would otherwise be performed by hardware is instead executed by software at runtime, shifting the operational workload rather than merely setting up or orchestrating the hardware. This understanding also makes sense because defining a hardware task as one that simply has the “potential” to be assigned to hardware would mean that effectively *every* computer task is a hardware task.

142. As a POSITA, I do not believe it would be obvious to offload from hardware to software. Systems typically offload packets in the opposite direction—from software to hardware—to conserve software resources for more complex computational tasks. Modular, swappable, flexible dataflow systems like the ’753 patent need to operate at optimal efficiency, so having the option to offload slow

tasks to software is important. Typical systems seek to offload software tasks to hardware to free up software resources, but the inventors of the '753 patent appear to have understood that its system would benefit from tasks flowing in each direction. This is why the '753 patent discloses a system of “deeply embedding processing elements in hardware.” EX1001, 6:28–30.

143. Stroud’s system addresses a problem in the prior art that is different from the '753 patent. Execution-level processors were used unevenly, thereby reducing processing efficiency in multi-processor systems. EX1004, ¶245. Stroud does not discuss offloading tasks from hardware to software because that was not the focus of the application. Instead, it explains that conventional load-balancing mechanisms routed packets based on unique packet characteristics, but when packets lacked unique attributes, conventional systems tended to create bottlenecks by routing all traffic to a single NP. *Id.* Stroud’s solution was to have control processor 106 configure routing CLD 102B to expose packet features that were not normally accessible in conventional load-balancing systems. *Id.* Despite this routing functionality, Stroud does not show that control processor 106 “offloads” packets from a hardware element to a software element. Stroud’s form of packet routing is substantially different than the '753 patent’s design of “deeply embedding processing elements in hardware” so that a processing element could identify over-

provisioned hardware elements and dynamically shift those tasks to available software elements. EX1001, 6:28–45.

144. In my opinion, Stroud does not teach moving tasks from hardware to software that were originally assigned for execution in hardware. Instead, it describes a different resource management approach where tasks are balanced across multiple network processors. Hardware elements merely facilitate packet routing and are not capable of actually executing the tasks. EX1004, ¶¶245–248. Stroud does not disclose an architecture in which software processors can assume the routing functionality of the hardware elements when those elements become overloaded. *Id.*

4. Stroud Does Not Disclose a Processor “Executing in Cooperation with a Sandbox of One of the Plurality of Swappable Pods or Cards” As Required by Claim 3.

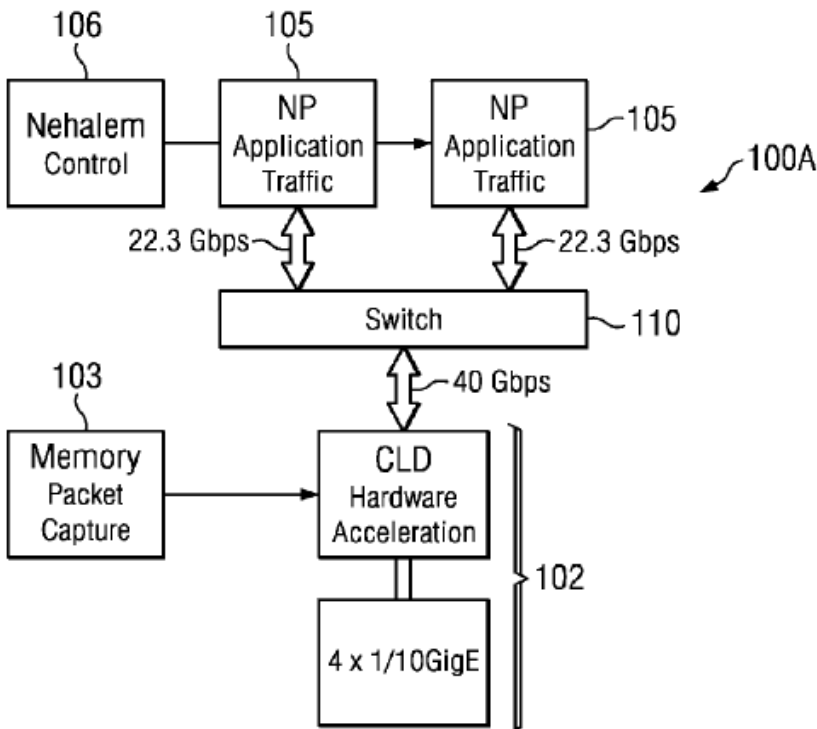
145. In my opinion, Stroud does not teach that the same “processor” that is allocatable for the other three claimed functions (including “offloading”) is also allocatable for “executing in cooperation with a sandbox of one of the plurality of swappable pods or cards.” EX1001, cl. 3. Stroud does not disclose that any of the “swappable pods or cards” have “a sandbox,” and thus does not disclose this limitation.

146. Stroud’s CLDs do not include sandboxes. Petitioners mistakenly argue that Stroud’s CLDs include sandboxes simply because they are PLDs/FPGAs with portions for user-definable hardware. Pet., 60. A POSITA would understand that a

sandbox is a partitioned region within a CLD/PLD/FPGA where users can install custom logic, not any kind of user-configurable device.

147. Nothing in Stroud mentions sandboxes. Petitioners appear to cite portions of Stroud that disclose a generic CLD comprising a FPGA. EX1004, ¶107, ¶¶109–112, ¶117 ¶¶311–313 (each disclosing only that Stroud’s CLDs can be PLDs/FPGAs, but not mentioning sandboxes), ¶462; EX1003, ¶161 (discussing the operation of CLDs without mentioning sandboxes). But none of these citations discuss an isolated, partitioned region of the FPGA equating to a sandbox.

148. For example, Petitioners point to Figure 4 of Stroud, which teaches that the architecture of a card in the system may contain CLDs coupled to a management switch for “high-performance, specialized computation, data transfer, and data analysis capabilities to process certain data or computation intensive tasks at or near the network line rates.” Pet., 60 (citing EX1004, ¶106).



EX1004, FIG. 4. But while Stroud may contain a CLD hardware accelerator “that includes a set of programmable logic units, internal memory, and high-speed internal and external interconnections,” there is no evidence that the CLD contains a sandbox. *Id.*, ¶107. Stroud does not discuss or even mention sandboxes, much less specify that Stroud’s CLDs “include sandboxes.”

149. Petitioners make the same mistake with their other evidence. Stroud discusses that the system may contain a CPU, GPU, NP, or ASIC programmed to implement an instruction set, but it is silent on whether there are sandboxes within these devices. Pet., 60 (citing EX1004, ¶¶110–112). And while Stroud defines CLDs as a “device that can be modified to simulate many types of hardware,” ¶117 only

discusses that CLDs are reprogrammable to support testing and simulation architecture. *Id.* (citing EX1004, ¶117).

150. Stroud actually suggests that its CLDs do not contain sandboxes. Petitioners point to ¶¶311–13, Pet., 60, but in those paragraphs, Stroud references the ALTERA STRATIX 4, which does not come pre-programmed with a sandbox. EX2023. The ALTERA STRATIX 4 was a widely available FPGA and one that I am familiar with as a POSITA. It did not come with a sandbox region accessible to the control processor. *Id.* If a user sought to use ALTERA STRATIX 4 as a sandbox, they would need to reconfigure the system themselves to create a sandbox, which would not be obvious to do in Stroud. *Id.*

151. I do not identify any evidence that “Stroud’s CLDs include sandboxes.” I also disagree that Stroud inherently discloses sandboxes because anything that includes “programmable electronic circuits” is a sandbox because it is contrary to the plain and ordinary meaning of “sandbox” and effectively reads out the limitation from the ’753 patent. Pet., 60. Since the sandbox is a partitioned region, it enhances security, enables testing, and prevents harmful processes from accessing critical resources. Sandboxes provide isolated environments that allow systems to safely test, debug, and execute processes without risking the integrity or security of the main architecture. Additionally, they help contain crashes or faults, so they don’t impact the rest of the system.

152. It would be incorrect for Petitioners to adopt a construction of “portion designated for programmable electronic circuits” without acknowledging that the “portion” refers to an isolated region in the device. I understand Dr. Alpert did not analyze the meaning of sandbox to a POSITA; rather, he testified that he only based his “sandbox” opinion on a proposed construction circulated between Petitioners and Patent Owners. EX2024, 124:13–17; 130:3–6 (“[The analysis is] not in my declaration because I didn’t rely on that information. I relied instead on this construction proposed by the Patent Owner, which I thought was supported by the specification.”). I disagree with this analysis because it ignores “portion” from the construction and contends that a sandbox can be anything, even the entire FPGA. The result is that sandbox has no meaning. Dr. Alpert’s report does not include any discussion on the meaning of a sandbox, which it could not because there is no support in the art for his understanding of this claim limitation.

153. My understanding of a sandbox is confirmed by the ‘753 patent specification. The ‘753 patent discloses a PLD for data flow processing in a user-configurable server that contains a second *region* with “one or more sandboxes with user-definable electronic circuits.” EX1001, 1:36–44. Claim 3 makes clear that one or more of the swappable pods within the system contain a sandbox, and the sandbox is implemented as a configurable *region* in the hardware module. *Id.*, cl. 3. The ‘753 patent explains that, in some embodiments, the programmable device is an FPGA, and

the MMI network connection links to sandboxes *within* the FPGA. *Id.*, 4:62–65. The sandboxes are not limited to functioning as MMI network connectors, but “[m]any features could be implemented in the sandboxes . . . [u]ser-definable hardware modules and user-definable software modules are possible [which] could include dynamic port interfaces to processing elements or other modules, configurable as to data width and/or speed, or direct connect (e.g., serial link or parallel bus) versus hardware queue (e.g., FIFO) or software queue.” *Id.*, 5:50–56.

154. My opinion is also informed by the ‘753 teaching that “a user can configure or reconfigure one or more modules in one or more sandboxes 132.” EX1001, 5:30–34. Sandboxes give users the opportunity to install custom hardware logic. Consistent with the “hardware defined anything” purpose of the invention, the sandbox provides users with the ability to install custom logic that supports routing, dataflow processing, hardware acceleration, or memory access. EX1001, 13:14–17 (“The use of FPGAs 122, and especially with the introduction of sandboxes 132, support the “hardware defined everything” concept, and allow systems to be built for what’s next, i.e., systems not yet designed.”). By writing custom logic and loading it onto the PLD, the user can flexibly and safely integrate new logic into the system without interfering with fixed logic on the device. *Id.*

155. Stroud does not disclose that any of the “swappable pods or cards” have “a sandbox,” and thus does not disclose this limitation.

B. In my opinion Ground 2 fails because Petitioners have not shown that any of its multiple proposed combinations disclose or render or render obvious the memory limitations of [7d]

156. I understand that Ground 2 is focused on claim 7 and claims that depend on it, and relies on the combination of Stroud and Cragon to supply the required claim limitations. As I explain below, in my opinion Petitioners' proposed combination of Stroud and Cragon do not render obvious Claim 7 or any of its challenged dependent claims. My analysis below focused on what I understand has been labelled limitation [7d], because in my opinion it clearly is not present in Stroud or Cragon, and the proposed combination.

157. I note at the outset that Limitation [7d] has several requirements. The first requirement is the step of "selecting a memory superposition in at least one of the plurality of swappable pods or cards." EX1001, 18:64-65. The claim also specifies that the swappable pod or card¹ at issue must contain a "plurality of types of memory with selectable arrangement of the plurality of types of memory." *Id.*, 18:66-67. Finally, the card must have a "selectable arrangement" that includes each of the following memory arrangements: (1) "multilevel cache;" (2) "interleaved

¹ I understand the Petition uses "card" to refer to both "cards" and "pods." Pet., 23. Accordingly, I will do the same in this declaration, and my references to "card" or "cards" in this section apply equally to "pods."

memory;” (3) “swapped memory over a specified address range;” and (4) “memory allocated to specified address ranges according to parameterized task needs” (which I refer to collectively as the “claimed memory arrangement”). Thus, in my opinion, Limitation [7d] requires a single card that has multiple types of memory that can be arranged in each of the four enumerated memory arrangements, such that one of the memory arrangements can be selected.

158. Below I focus on two core reasons that Petitioners prior art combination does not disclose or render obvious this limitation.

159. The first reason is that the prior art does not disclose or render obvious the feature of having a single card that includes each of the four claimed memory arrangements. None of the prior art of record teaches or even suggests combining these four memory arrangements in particular on a single card. To the contrary, as discussed both above and below in more detail, Cragon teaches that core memory would typically be housed in an external disk and not on a single modular card. EX1006, 000558. Further, I have seen no teaching or suggestion in any of the prior art, or other credible evidence, that a POSITA would have been motivated to choose the specific memory arrangements required by the claims, among the extremely high number of arrangements that were available.

160. The second reason is that the prior art does not disclose or render obvious a card that has selectable memory arrangements. Neither Stroud or Cragon discloses anything like this functionality.

1. Neither Stroud nor Cragon teaches, suggests, or renders obvious a single card with each of the four required memory arrangements.

161. I understand that Stroud is the primary reference in Ground 2. As a general matter Stroud is largely indifferent to both memory types and memory configurations. *See* EX1004, ¶127 (“Capture memory device 103A may comprise any suitable type of memory device, e.g., DRAM, SRAM, or Flash memory, hard drive, or any other memory device with sufficient bandwidth.”). The concept of memory superposition is never discussed in Stroud. Nor does it disclose any memory arrangements at all, much less the specific memory arrangements required by the claims.

162. I understand that this is not in dispute as the Petition does not rely on Stroud to disclose the four specific memory arrangements required by limitation [7d]. *See* Pet., 70-78. And I understand that Dr. Alpert agreed that “Stroud does not disclose all four of those memory arrangements.” EX2024, 92:13-16.

163. Cragon includes some generalized discussions of memory arrangement concepts, but it not does teach that they should be used together, much less housed on a single card. Once again, I understand that this is not in dispute as Dr. Alpert

testified that “Cragon does not disclose all four arrangements being used in one card.” EX2024, 104:10-14.

164. Instead, Cragon teaches away from such a configuration. As discussed above, Cragon contemplates that memory systems will largely be housed on “peripheral systems”—not on modular cards like those described and claimed in the ’753 patent. EX1006, 000558. The discussion in Cragon that Dr. Alpert relies on for teachings regarding “allocating memory to a task, and therefore the specified address range for the task, according to parameterized task needs” stem from a discussion of such peripheral memory systems. EX1003, ¶232. There is no teaching or suggestion in Cragon of using four memory arrangements required by the claims together on a single card.

165. Absent such a teaching, a POSITA would have no objective reason or motivation to create the specific memory arrangement required by the claims. I agree with Dr. Alpert that, as of 2018, there were at least “hundreds of memory arrangements out there.” EX2024, 103:17-104:6. Even if there were only 100, as a matter of pure statistics, that means that there were close to four million possible sets of four memory arrangements available. If larger set sizes are considered, the quantity of potential permutations reaches the billions. And I am aware of no teaching in the prior art that it was advantageous to use sets of memory

arrangements, whether of four or some other number, and house them on the same card.

166. In my opinion, Petitioners' and Dr. Alpert's proposed reasons why a POSITA would create the claimed arrangement are unpersuasive, inconsistent with the prior art, and very much influenced by hindsight. Petitioners contend that "a POSITA would have been motivated to provide flexible memory arrangements to support Stroud's flexible architecture and the variety of potential functions on Stroud's cards," and consequently "would have been motivated to apply Cragon's memory arrangements for their known benefits." Pet., 78-79. Dr. Alpert says something similar: "POSITA would have been motivated to provide flexible memory arrangements to support Stroud's flexible architecture." EX1003, ¶234. And Petitioners and Dr. Alpert go on to argue that POSITA would have seen Cragon as a means of "further[ing] Stroud's stated goals of scalability and flexibility by teaching different ways for memory to be utilized." EX1003 ¶235; *see also id.* ¶¶236-38; Pet., 78-81.

167. This reasoning suffers from a number of core defects. First, generalized goals like a "scalability and flexibility" are too abstract, in my opinion, to provide a concrete motivation to pursue any particular configuration, much less the highly specific configuration required by the claims. Further, if a POSITA were motivated by those goals, a POSITA would not identify Cragon as a resource for

achieving them. In 2018 Cragon would have been seen as a dated textbook that is interesting for its discussion of the history of memory development and antiquated memory structures, such as floppy disks, but not as a resource for teachings regarding new systems. Further, Petitioners and Dr. Albert do not actually identify any teachings regarding scalability and flexibility. Nor have I. To the contrary, Cragon's discussion of memory arrangements suggests not they are flexible, scalable, or selectable, but they are largely fixed by hardware decisions made at the design stage. *See generally*, EX1006 000536-000548, 000558.

168. Further, neither Petitioners or nor Dr. Alpert ever explain why a POSITA would have been motivated to select the particular four memory arrangements required by the claims and put them together on a card. There is no suggestion of such a configuration in Cragon, or even of a suggestion that different memory arrangements are preferably used together on a single card. Instead, the portions of Cragon that Petitioners principally rely on simply discuss, at a high level, the characteristics, advantages, and disadvantages of various memory configurations that are available at the design stage. EX1006, 000536-548. I note, moreover, that Dr. Albert's declaration did not even include a discussion whether the four claimed arrangements would even be compatible, much less identify any teaching in Cragon of compatibility. EX2024, 105:2-6 ("Q. Okay. Can you combine all types of memory arrangements together successfully? A. It's not something I thought about. ***I don't***

think I have an opinion whether all types can be combined.”). Even if it had, that would not explain why a POSITA would pursue the particular combination of arrangements of the claims—which no prior art of record discloses. I have reviewed all of portions of Cragon that Dr. Alpert cites in his mapping of the four arrangements to Cragon and I see no teaching that they should be used together. EX1003, ¶¶220-233. There is simply no disclosure, teaching, or suggestion of using the distinct memory arrangements required by the claims together on the same card.

169. Dr. Alpert and Petitioners likewise do not explain why a POSITA would be motivated to choose the particular disclosures in Cragon relied upon. *See* Pet., 73-78; EX1003, ¶¶220-233. As discussed above, Cragon is a textbook that provides a broad overview of a variety of memory structures and processes as they existed in 1996 (and also devotes substantial space to the history of prior memory systems), including many memory arrangements other than the four claimed. Cragon does not teach or suggest that everything it discloses should or can be combined together, and a POSITA would understand that a textbook cannot be interpreted that way. In essence, however, Dr. Alpert and Petitioners’ obviousness theory relies on an assumption that a POSITA would treat all teachings in Cragon as combinable. A POSITA would not make such an unreasonable assumption regarding the teachings of a textbook.

170. Further, Petitioners and Dr. Alpert never grapple with the technical complexity of combining four distinct memory arrangements on a single card. For example, it would typically require an adjustment to the physical arrangement of the memory to shift from a multi-level cache to an interleaved memory. Petitioners never address how this could be technically achievable on a single card, much less why a POSITA would be motivated to take on the effort of making it technically achievable.

2. Neither Stroud nor Cragon, or the proposed combination, teaches that memory superpositions can be *selected*.

171. In my opinion, Ground 2 also fails because there is no disclosure in Stroud or Cragon of “selecting a memory superposition” or of the “selectable arrangement” required by limitation [7b]. As discussed above, Cragon teaches a variety of memory structures that existed and that system designers could consider. It never teaches or suggests that they can be configured in a selectable arrangement. To the contrary, Cragon’s teachings suggest the opposite—that memory arrangements are heavily and influence by hardware decisions and must be selected at the design phase. *Accord*, EX1006, 000536-000548, 0000558.

172. Petitioners and Dr. Alpert’s contention the claimed arrangement is somehow suggested because Cragon teaches “selecting the arrangement of memory based on workload parameters” is off base. Pet., 73; EX1003, 217. Petitioners and Dr. Alpert cite two portions of Cragon for this assertion and neither supports it. The

first, pages 00413-414, is focused on techniques for protecting memory systems from unauthorized access, not selection of different memory arrangements. In particular, the pages deal with “bound registers.” To be clear, registers are part of a processor (e.g., CPU, DSP, GPU) and is not part of the claimed “multilevel cache,” which is part of the addressable memory. Registers are used during the execution of instructions by the processor and is not part of the memory provided by the “swappable pods or cards.” Nothing in the cited pages or in the understanding a POSITA would have of bound registers suggests selecting between various memory arrangements comprising multiple memory types on a single card.

173. The second citation is to pages 000536-548 of Cragon. As I have explained above, these pages are merely are directed to general aspects of disk design. As Cragon makes clear, moreover, they relate to the design of “peripheral [memory] devices.” EX1007, 000558. Nothing in these pages suggests memory arrangements that can be selected during operation or even configuration of a system. Further, the fact that Cragon states that “[t]here is very little solid information published on the question of the disk bandwidth and storage requirements based upon workloads” shows that a POSITA relying on Cragon would have had no reason to pursue a selectable arrangement or to believe it was reasonably likely to succeed. EX1006, 000536.

174. Finally, as noted above, Petitioners and Dr. Alpert never address the technical complexity of combining four distinct memory arrangements on a single card in a selectable arrangement. For example, it would typically require an adjustment to the physical arrangement of the memory to shift from a multi-level cache to an interleaved memory. Petitioners never address how this could be accomplished when the distinct memory arrangements must be selectable on a single card. Further, I have reviewed all of portions of Cragon that Dr. Alpert cites in his mapping of the four arrangements to Cragon and I see no teaching that they should be used together, much less together in a selectable arrangement. EX1003, ¶¶220-233.

3. Claims 9-10 are Patentable over Stroud and Cragon.

175. Because independent claim 7 is patentable over Stroud and Cragon, the more specific dependent claims are also patentable for the same reasons.

C. Grounds 3 and 4 Suffer From the Same Core Problems as Ground 2.

176. Implicitly recognizing that Ground 2 includes no disclosure of a “selectable arrangement,” I understand that in Ground 3, Petitioners propose a different obviousness combination that adds Saulsbury purely for what Petitioners

call its “additional teachings of selecting between memory modes.”² Pet., 82. I also understand that Ground 4 is substantially identical to Ground 3 for purposes of my analysis because in Ground 4 Petitioners apply the obviousness theories of Ground 3 as to memory-related limitations. In my opinion, these grounds do not establish obviousness for at least the reasons below.

1. Saulsbury does not teach, alone or in combination, the combination of four distinct memory arrangements on a single card required by limitation [7d].

177. Saulsbury does not rectify the core defects with Ground 2 because it, like Stroud and Cragon, does not disclose the four required memory arrangements. I understand this is not disputed. EX2024, 93:11-20 (“Saulsbury doesn’t disclose all four of the claimed memory arrangements”). Indeed, in my opinion, Saulsbury does not disclose any memory arrangements (as opposed to memory types).

178. Petitioners rely on Saulsbury for its disclosure of “*switch[ing] the memory mode* between physical memory and cache memory, depending on the particular application of the processor and memory chip.” Pet., 85 (emphasis and brackets in original; citing EX1007 ¶¶3, 65, Fig. 1). But as Dr. Alpert testified,

² Grounds 3 and 4 are substantively identical for purposes of this analysis, since in Ground 4 Petitioners simply apply the obviousness theory of Ground 3 for memory limitations. Pet., 93-94.

Saulsbury teaches only “switching between cache memory and physical memory.” EX2024, 95:16-96:2. The control mechanism for Saulsbury’s switching is a bit, which can only have two values. I understand this is undisputed. EX2024, 96:13-14 (“Saulsbury teaches using a control bit to select the arrangement of the memory”); *id.* (“[a] bit can have two values”); *see also* Pet., 85 (“The arrangement is selected using a control bit.”).

179. Such a control bit could not be used for the four-way selectability required by limitation 7[d]. The claim specifically calls for the “memory superposition,” which requires more than one type of memory. What Saulsbury discloses is selecting the use of one of two memory types, *i.e.*, one that is included within the chip and the one located outside the chip. This clearly do not satisfy the claimed “memory superposition.” I note that nothing in the Petition or Dr. Alpert’s declaration explains how a binary control bit that can switch between two options could be modified by a POSITA to achieve the required four-way selectability of limitation [7d].

180. Further, Ground 3 relies on the same disjointed disclosures from Cragon that a POSITA would not be motivated to combine (as I’ve explained above with respect to Ground 2) and fails to provide any reason that a POSITA would choose the four specific memory arrangements from among the hundreds of arrangements (and millions of sets of four arrangements) available. Accordingly,

Ground 3 lacks a credible motivation to combine for all of the reasons I explained above for Ground 2.

2. Saulsbury does not teach, alone or in combination, the claimed “selectable arrangement”

181. As discussed in the prior section, Saulsbury at most discloses switching between memory types—not memory arrangements. As I explained in my description of the ’753 patent memory types and memory arrangements are different. Memory types are different memory structures, while memory arrangements are the methods used to integrate and harmonize them to achieve design objectives in the most efficient way possible. EX1001, 7:40-46 (“Analogous to the overlapping of waves to form a final observed wave in wave superposition, where each wave contributes to the whole, in this case different types of memory are being overlapped, each with benefits and drawbacks, to form a final memory system which performs optimally in a given video stream or other data flow/data processing task.”).

182. Dr. Alpert and Petitioners never engage with this distinction. Instead, they treat switching between memory types in Saulsbury as amounting to a general teaching that any aspect of a memory system can or should be selectable. *See, e.g.,* Pet., 85, 90-91. A POSITA would not interpret Saulsbury in this way, and would recognize that mere switching between memory types, where only one type of memory is selected, is fundamentally different from switching between different memory arrangements that combines multiple types of memory.

183. In addition, as noted in the prior section, the Petition and Dr. Alpert's declaration do not explain how Saulsbury's control bit could be used to control the four-way selection required by the claims. At most, Saulsbury introduces the abstract concept of selectability, without providing any teaching, or explanation of how to achieve, selection among memory arrangements.

3. Claims 9-10 are Patentable over Stroud, Cragon, and Saulsbury.

184. Because independent claim 7 is patentable over Stroud, Cragon, and Saulsbury the more specific dependent claims are also patentable for the same reasons.

XII. SIGNATURE

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed on this 19th day of December, 2025.

A handwritten signature in black ink, appearing to read "Paul S. Min", is written over a horizontal line.

Dr. Paul S. Min