

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION
("Microsoft"),
Petitioner

v.

THROUGHPUTER, INC.
("ThroughPuter"),
Patent Owner

Patent No. 10,963,306

DECLARATION OF DR. HENRY H. HOUH

TABLE OF CONTENTS

I.	INTRODUCTION	4
II.	QUALIFICATIONS	5
III.	LEVEL OF ORDINARY SKILL IN THE ART	10
IV.	MATERIALS CONSIDERED AND RELIED UPON	11
V.	LEGAL STANDARDS	13
A.	LEGAL STANDARDS FOR PRIOR ART.....	13
B.	LEGAL STANDARD FOR PRIORITY DATE	14
C.	LEGAL STANDARD FOR OBVIOUSNESS	14
VI.	OVERVIEW OF THE '306 PATENT	18
A.	SUBJECT MATTER OVERVIEW	18
B.	FILE HISTORY OF THE '306 PATENT	27
C.	INTERPRETATION OF THE '306 PATENT CLAIMS AT ISSUE	29
VII.	OVERVIEW OF THE CITED REFERENCES.....	30
A.	NOLLET (EX1004)	30
B.	TUAN (EX1005).....	34
C.	MILLER (EX1006)	39
D.	REDAELLI (EX1007)	41
VIII.	ANALYSIS.....	42
A.	[GROUND 1A & 1B] – NOLLET IN VIEW OF TUAN AND MILLER OR NOLLET IN VIEW OF TUAN, MILLER, AND REDAELLI RENDERS CLAIMS 1-24 OBVIOUS	42
1.	The Nollet-Tuan-Miller Combination	42
2.	The Nollet-Tuan-Miller-Redaelli Combination.....	50

Claim 1.....	52
Claim 2.....	81
Claim 3.....	83
Claim 4.....	84
Claim 5.....	85
Claim 6.....	88
Claim 7.....	89
Claim 8.....	90
Claim 9.....	92
Claim 10.....	92
Claim 11.....	93
Claim 12.....	94
Claim 13.....	100
Claim 14.....	101
Claim 15.....	102
Claim 16.....	102
Claim 17.....	102
Claim 18.....	103
Claim 19.....	103
Claim 20.....	104
Claim 21.....	104
Claim 22.....	104
Claim 23.....	104
Claim 24.....	105
IX. CONCLUSION.....	105

I, Dr. Henry H. Houh, of Burlington, MA, declare that:

I. Introduction

1. I have been retained on behalf of Microsoft Corp. (“Microsoft” or “Petitioner”) to offer technical opinions related to U.S. Patent No. 10,963,306 (“The ’306 patent”) (Microsoft-1001). I understand that Microsoft is requesting that the Patent Trial and Appeal Board (“PTAB” or “Board”) institute an *inter partes* review (“IPR”) proceeding of the ’306 patent. Specifically, I have been asked to consider the validity of claims 1–24 of the ’306 Patent (the “Challenged Claims”) in view of prior art, anticipation and obviousness considerations from the perspective of a person of ordinary skill in the art at the time of the invention (“POSITA”) as it relates to the ’306 Patent.

2. I am being compensated for my time at my standard consulting rate. I am also being reimbursed for expenses that I incur during the course of this work. My compensation is not contingent upon the results of my study, the substance of my opinions, or the outcome of any proceeding involving the challenged claims. I have no financial interest in the outcome of this matter or on the pending litigation between Petitioner and Patent Owner.

3. My analysis here is based on my years of education, research and experience, as well as my investigation and study of relevant materials, including those cited herein.

4. I may rely upon these materials, my knowledge and experience, and/or additional materials to rebut arguments raised by the Patent Owner. Further, I may also consider additional documents and information in forming any necessary opinions, including documents that may not yet have been provided to me.

5. My analysis of the materials produced in this proceeding is ongoing and I will continue to review any new material as it is provided. This declaration represents only those opinions I have formed to date. I reserve the right to revise, supplement, and/or amend my opinions stated herein based on new information and on my continuing analysis of the materials already provided.

II. Qualifications

6. I am over the age of 18 and am competent to write this declaration. I have personal knowledge, or have developed knowledge of these technologies based upon education, training, or experience, of the matters set forth herein.

7. My qualifications and professional experience are described in my curriculum vitae, a copy of which is attached as Appendix A.

8. I received a Ph.D. in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology (“MIT”) in 1998. Beforehand, I received a Master of Science degree in Electrical Engineering and Computer Science in 1991, a Bachelor of Science degree in Electrical Engineering and

Computer Science in 1989, and a Bachelor of Science degree in Physics in 1990, all from MIT.

9. I am currently self-employed as an independent technical consultant. I am also president of a company that provides supplemental science, technology, engineering, and mathematics (“STEM”) education to children of all ages.

10. I first entered telecommunications in 1987 when I worked as a summer intern at AT&T Bell Laboratories as part of a five-year dual degree program at MIT. During my first summer, I worked designing digital hardware including using programmable logic devices. I continued to work at AT&T Bell Laboratories as part of this MIT program. While at MIT, I was a teaching assistant (“TA”) in the Electrical Engineering and Computer Science Department’s core Computer Architectures course. I first was a TA as a senior for a role typically reserved for graduate students. I later became head TA. The course covered various topics in computer architectures, including programmable logic arrays, multiprocessor systems and parallel processing. As a TA, I helped write homework assignments, lab assignments, and exams. I also taught in the recitation sections.

11. Later, as part of my doctoral research at MIT from 1991-1998, I was a research assistant in the Telemedia Network Systems (“TNS”) group at the Laboratory for Computer Science. The TNS group built a high-speed gigabit network and created applications that ran over the network. Example applications

included ones for remote video capture, processing, and display of video on computer terminals. In addition to working on the design of core network components, designing and building the high-speed links, and designing and writing the device drivers for the interface cards, I also helped design the switch, which was composed of an array of FPGAs at the core.

12. I also set up the group's web server and helped to build the web pages that initiated the above-mentioned video sessions via a web interface. Vice President Al Gore visited our group in 1996 and received a demonstration of—and remotely drove—a radio-controlled toy car with a wireless video camera mounted on it that was built by our group. This toy car device received commands transmitted over a network from a remote computer, and video data from the toy car was transmitted wirelessly then over a computer network back to the user controller. On occasion, we allowed users visiting our web site to drive the toy car from their remote computer while they watched the video on their computer. The video stream was encoded by TNS-designed hardware, streamed over the TNS-designed network, and displayed using TNS-designed software.

13. I defended and submitted my Ph.D. thesis, titled “Designing Networks for Tomorrow's Traffic,” in January 1998. As part of my thesis research, I analyzed local area and wide area flows to show a more efficient method for routing packets in a network, based on traffic patterns at the time.

14. From 1997 to 1999, I was a Senior Scientist and Engineer at NBX Corporation, a start-up that made business telephone systems for streaming packetized audio over data networks instead of using traditional telephone lines. NBX was later acquired by 3Com Corporation and the phone system is still used today by numerous businesses.

15. As part of my work at NBX, I designed the core audio reconstruction algorithms for the telephones, as well as the packet transmission algorithms. I also designed and validated the core packet transport protocol used by the phone system. The protocol was used for all signaling in the phone system, including for the setup of conference calls. The NBX system also featured a computer interface for initiating phone calls, which could also initiate conference calls. The NBX system also supported the Telephony Application Programming Interface (“TAPI”) that allowed other computer programs to integrate with our system telephony features. We obtained U.S. Patent No. 6,697,963, entitled “Telecommunication method for ensuring on-time delivery of packets containing time-sensitive data,” as part of this work.

16. From 1999-2004, I was employed by Empirix or its predecessor company, Teradyne. Empirix was a leader in test tools for telecommunications protocols and systems, providing functional testing tools as well as load testing tools. From 2000-2001, I conceived and built a test platform for testing Voiceover-

IP (VoIP) based on highly parallel network processors. The first application on this new test platform was a cloud emulator for simulating the effects of transmitting VoIP over a busy network. The platform was based on a multi-core high performance network processor chip.

17. In 2006, as part of my role at BBN Technologies, I helped found PodZinger Inc., now known as RAMP Inc. PodZinger utilized BBN's speech recognition algorithms to search through the spoken words in audio and video segments. While I was Vice President of Operations and Technology, PodZinger followed its initial prototype with a full streaming audio and video search solution. I also created a social networking web site, which BBN sold to a venture-funded startup company. In the process of creating the web site, I designed and specified the authentication and authorization protocols.

18. I have been awarded several United States patents, and I have several patent publications including the following examples:

- U.S. Patent No. 7,975,296, "Automated security threat testing of web pages";
- U.S. Patent No. 7,877,736, "Computer language interpretation and optimization for server testing";
- U.S. Patent No. 7,801,910, "Method and apparatus for timed tagging of media content";

- U.S. Patent 7,590,542, “Method of Generating Test Scripts Using a Voice-Capable Markup Language”;
- U.S. Patent No. 6,967,963, “Telecommunication method for ensuring on-time delivery of packets containing time-sensitive data”;

19. Additional details concerning my professional qualifications, experience, and publications are set forth in my curriculum vitae (“CV”), which is attached as Appendix A.

III. Level of Ordinary Skill in the Art

20. In rendering the opinions set forth in this declaration, I was asked to consider the patent claims and the prior art through the eyes of a POSITA at the time of the alleged invention, which I understand is asserted to be November 4, 2011—the earliest possible priority date of the ’306 Patent. I understand that the factors considered in determining the ordinary level of skill in a field of art include the level of education and experience of persons working in the field; the types of problems encountered in the field; the teachings of the prior art, and the sophistication of the technology at the time of the alleged invention. I understand that a POSITA is not a specific real individual, but rather is a hypothetical individual having the qualities reflected by the factors above. I understand that a POSITA would also have knowledge from the teachings of the prior art, including

the art cited below.

21. Taking these factors into consideration, on or before November 4, 2011, a POSITA relating to the technology of the '306 Patent would have had at least a combination of a Bachelor's Degree (or a similar Master's Degree, or higher degree) in an academic area emphasizing computer science, or a related field, and two or more years of work experience with parallel processing technologies. Additional education or industry experience may compensate for a deficit in one of the other aspects of the requirements stated above.

22. Before November 4, 2011, my level of skill in the art was at least that of a POSITA. I am qualified to provide opinions concerning what a POSITA would have known and understood at that time, and my analysis and conclusions herein are from the perspective of a POSITA as of that date.

IV. Materials Considered and Relied Upon

23. In reaching the conclusions described in this declaration, I have relied on the documents and materials cited herein as well as those identified in this declaration, including the '306 Patent, the prosecution history of the '306 Patent, and prior art references cited herein. These materials comprise patents, related documents, and printed publications. Each of these materials is a type of document that experts in my field would have reasonably relied upon when forming their opinions.

24. I have also relied on my education, training, research, knowledge, and personal and professional experience in the relevant technologies and systems that were already in use prior to, and within the timeframe of the earliest possible priority date of the claimed subject matter in the '306 Patent, which is November 4, 2011.

- EX1001: U.S. Patent No. 10,963,306 to Sandstrom (“the ’306 patent”)
- EX1002: Excerpts from the Prosecution History of the ’306 patent (“the Prosecution History”)
- EX1004: U.S. Patent Application Publication No. 20090187756A1 to Nollet et al. (“Nollet”)
- EX1005: Vu Manh Tuan. “A Study on a Multitasking Environment for Dynamically Reconfigurable Processors.” (2009) (“Tuan”)
- EX1006: U.S. Patent No. 8,296,434 to Miller et al. (“Miller”)
- EX1007: Redaelli, Francesco, et al. “Reconfiguration aware task scheduling for multi-fpga systems.” 4th HiPEAC Workshop on Reconfigurable Computing (2010) (“Redaelli”)
- EX1008: U.S. Patent No. 8,136,153 to Zhang et al. (“Zhang”)
- EX1009: U.S. Patent No. 7,178,145 to Bono (“Bono”)
- EX1010: Lety, Emmanuel, Laurent Gautier, and Christophe Diot, “Mimaze, a 3d multi-player game on the internet,” Proceedings of the 4th International Conference on Virtual System and Multimedia, Gifu, Japan. Vol. 1. 1998 (“Lety”)
- EX1011: Kyprianos Papadimitriou, Apostolos Dollas, and Scott Hauck, “Performance of partial reconfiguration in FPGA systems: A survey and a

cost model,” ACM Transactions on Reconfigurable Technology and Systems (TRETs) 4, no. 4 (2011) (“Papadimitriou”)

V. Legal Standards

25. I am not a lawyer and do not provide any legal opinions, but I have been advised that certain legal standards are to be applied by technical experts in forming opinions regarding meaning and validity of patent claims. I have applied the legal standards described below, which were provided to me by counsel for the Patent Owner.

26. It is my understanding that assessing the validity of a U.S. patent based on a prior art analysis involves two steps. First, one must assess what meaning one of ordinary skill in the art would have given the terms. Second, after assessing the meaning of the claim terms, one may then assess validity by comparing a patent claim to the “prior art.” I understand that the teaching of the prior art is viewed through the eyes of a POSITA at the time the invention was made. My analysis as to what constitutes a relevant POSITA is set forth above.

A. Legal Standards for Prior Art

27. I understand that a patent or other publication must first qualify as prior art before it can be used to invalidate a patent claim. For purposes of this declaration, counsel for Petitioner has instructed me to assume that each prior art

reference cited in the combinations described below (Nollet, Tuan, Miller and Redaelli) qualify as prior art to the '306 Patent.

28. [RESERVED]

29. [RESERVED]

30. [RESERVED]

31. [RESERVED]

32. [RESERVED]

B. Legal Standard for Priority Date

33. I understand that the “priority date” (or “earliest effective filing date” or “Critical Date”) of a patent is the date on which it is filed, or the date on which an earlier application was filed if the patentee properly claims the benefit of the earlier application’s filing date. For purposes of my analysis in this declaration, I have assumed that the '306 Patent is entitled to a priority date of November 4, 2011.

C. Legal Standard for Obviousness

34. My understanding is that a patent claim is invalid as obvious only if the subject matter of the claimed invention “as a whole” would have been obvious to a POSITA at the time the invention was made. To determine the differences between a prior art reference (or a proposed combination of prior art references) and the claims, the question of obviousness is not whether the differences

themselves would have been obvious, but whether the claimed invention as a whole would have been obvious. Also, obviousness grounds cannot be sustained by mere conclusory statements. Rather, it is necessary to provide some articulated reasoning with rational underpinning to support the legal conclusion of obviousness.

35. I understand that a patent claim that comprises several elements is not proved obvious by simply showing that each of its elements was independently known in the prior art. In my evaluation of whether any claim of the '306 Patent would have been obvious, I considered whether the Petition, or any evidence submitted in this proceeding, presented an articulated reason with a rational basis that would have motivated a POSITA to combine the elements or concepts from the prior art in the same way as in the claimed invention.

36. It is my understanding that there is no single way to define the line between true inventiveness on one hand—which is patentable—and the application of common sense and ordinary skill to solve a problem on the other hand—which is not patentable. For instance, factors such as market forces or other design incentives may be the source of what produced a change, rather than true inventiveness.

37. I understand that the decision-maker may consider whether the change was merely the predictable result of using prior art elements according to their

known functions, or whether it was the result of true inventiveness. And, the decision-maker may also consider whether there is some teaching or suggestion in the prior art to make the modification or combination of elements recited in the claim at issue. Also, the decision-maker may consider whether the innovation applies a known technique that had been used to improve a similar device or method in a similar way. The decision-maker may also consider whether the claimed invention would have been obvious to try, meaning that the claimed innovation was one of a relatively small number of possible approaches to the problem with a reasonable expectation of success by those skilled in the art.

38. I have been instructed by counsel that if any of these considerations are relied upon to reach a conclusion of obviousness, the law requires that the analysis of such a consideration must be made explicit. I understand that the decision-maker must be careful not to determine obviousness using the benefit of hindsight and that many true inventions might seem obvious after the fact. I understand that the decision-maker should consider obviousness from the position of a POSITA at the time the claimed invention was made, and that the decision-maker should not consider what is known today or what is learned from the teaching of the patent.

39. I understand that in order to determine whether a patent claim is obvious, one must make certain factual findings regarding the claimed invention

and the prior art. Specifically, I understand that the following factors must be evaluated to determine whether a claim is obvious: the scope and content of the prior art; the difference or differences, if any, between the claim of the patent and the prior art; the level of ordinary skill in the art at the time the claimed invention was made; and, if available, the objective indicia of non-obviousness, also known as “secondary considerations.”

40. I understand that the secondary considerations include: commercial success of a product due to the merits of the claimed invention; a long felt need for the solution provided by the claimed invention; unsuccessful attempts by others to find the solution provided by the claimed invention; copying of the claimed invention by others; unexpected and superior results from the claimed invention; acceptance by others of the claimed invention as shown by praise from others in the field or from the licensing of the claimed invention; teaching away from the conventional wisdom in the art at the time of the invention; independent invention of the claimed invention by others before or at about the same time as the named inventor thought of it; and other evidence tending to show obviousness.

41. I understand that, to establish a secondary consideration, the evidence must demonstrate a nexus between that secondary consideration and the claimed invention.

VI. Overview of the '306 Patent

A. Subject Matter Overview

42. The '306 Patent is titled “Managing resource sharing in a multi-core data processing fabric” and relates to “an extensible, multi-stage, realtime application program processing load adaptive, manycore data processing architecture shared dynamically among instances of parallelized and pipelined application software programs” EX1001 at Title, Abstract.

43. Claim 1 of the '306 patent is reproduced below:

1. A method for task-switching on a multi-user parallel processing hardware architecture comprising a plurality of reconfigurable logic-based processing units, the method comprising:
linking, through a first set of inter-task communication paths of the multi-user parallel processing hardware architecture, a first set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a first multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a first program, wherein each of the first set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of first tasks of the first program corresponding to a respective processing stage of the first multi-stage program instance;
linking, through a second set of inter-task communication paths of the multi-user parallel processing hardware architecture, a second set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a second multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a second program, wherein each of the second set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of second tasks of the second program corresponding to a respective processing stage of the second multi-stage program instance;

maintaining, in a storage, a first location for a first reconfigurable logic-based processing unit of the first set of reconfigurable logic-based processing units executing the first multi-stage program instance such that a first one or more users and/or programs are enabled to communicate directly with the first multi-stage program instance;

maintaining, in the storage, a second location for a second reconfigurable logic-based processing unit of the second set of reconfigurable logic-based processing units executing the second multi-stage program instance such that a second one or more users and/or programs are enabled to communicate directly with the second multi-stage program instance; and

in response to an increased demand for the second program, reallocating, by a controller comprising software and/or hardware logic configured to implement a load-adaptive allocation policy, at least one processing unit of the first set of reconfigurable logic-based processing units, the reallocating resulting in

- (1) switching the at least one of the first set of processing units from performing a task of the plurality of first tasks to performing one task of the plurality of second tasks, wherein switching comprises matching a first programming configuration of the at least one of the first set of processing units to a programming configuration demanded by the one task, and
- (2) adjusting, in storage, at least one of the first location or the second location to enable, through the respective location, direct communication to the other multi-stage program instance of the first multi-stage program instance or the second multi-stage program instance;

wherein the load-adaptive allocation policy is configured to facilitate minimizing reconfiguring the plurality of reconfigurable logic-based processing units.

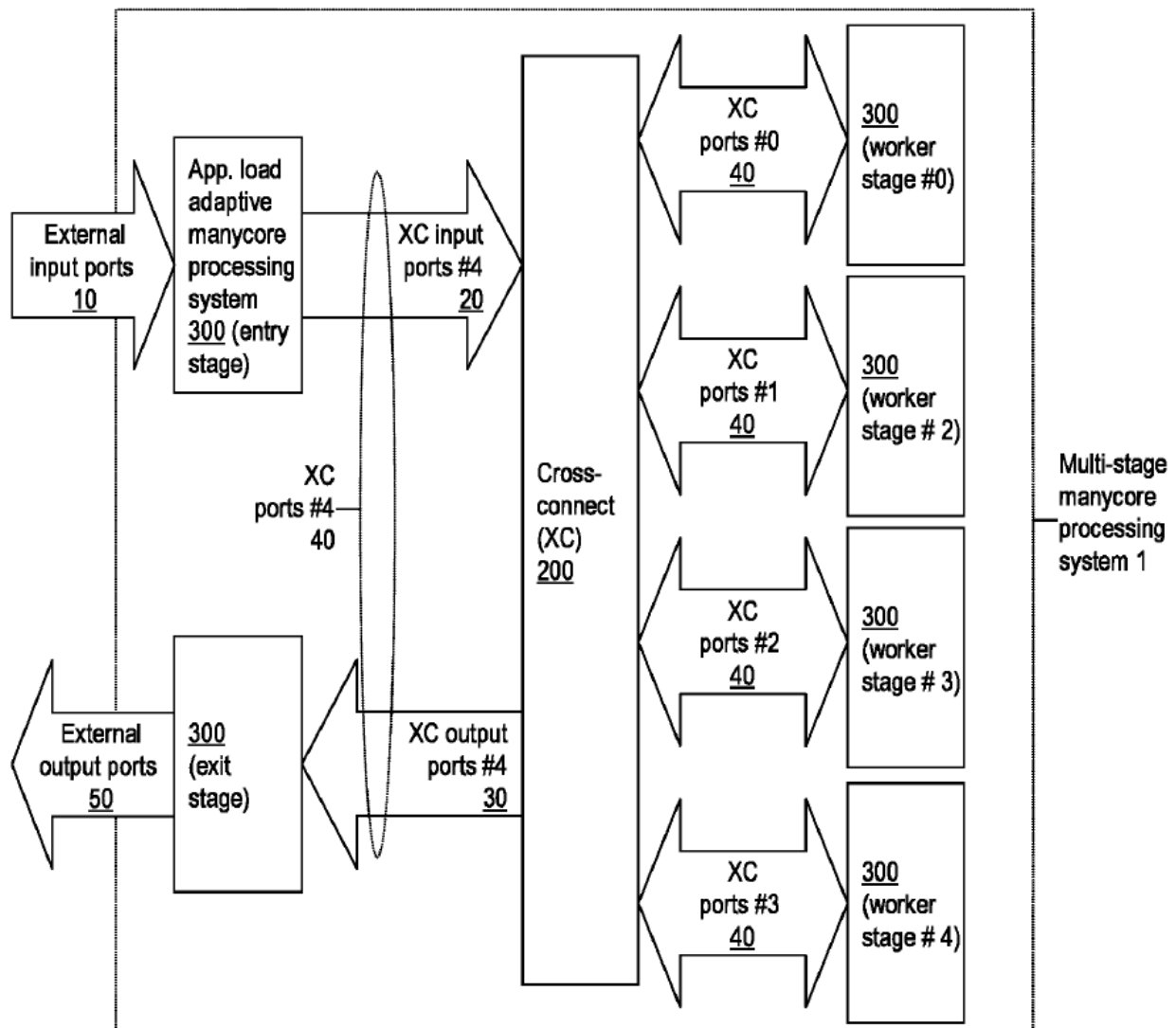
Id., Claim 1.

44. Based on the specification, the '306 patent is directed to “the field of data processing and networking, particularly to techniques for connecting tasks of parallelized programs running on multi-stage manycore processor with each other as well as with external parties with high resource efficiency and high data

processing throughput rate.” EX1001, 1:65-2:3.

45. Based on the specification, the '306 patent explains that two main objectives exist in advancing computing technology: 1) “**maximizing the processing speed** of one given computationally intensive program running on a dedicated hardware comprising a large number of parallel processing elements” and 2) “**most efficiently shar[ing] a given pool of computing hardware resources** among a large number of user application programs.” *Id.*, 2:7-17. As such, the '306 patent alleges that “there is a need for a new manycore processor system data flow and IO architecture whose operation, while providing IO data throughout performance, causes little or no overhead in terms of usage of the computation units of the processor.” *Id.*, 3:18-22; EX1002, 1.

46. As allegedly addressing these issues, the '306 patent describes a “multi-stage manycore processor system,” shown in the functional block diagram in Figure 1 (reproduced below). EX1001, 6:19-21. The block diagram includes “a set of application processing load adaptive manycore processing stages interconnected by a packet destination app-task-inst controlled cross connect.” *Id.*, 8:44-48. For example, Fig. 1 shows interconnected worker stages #0, #2, #3 and #4. *Id.*



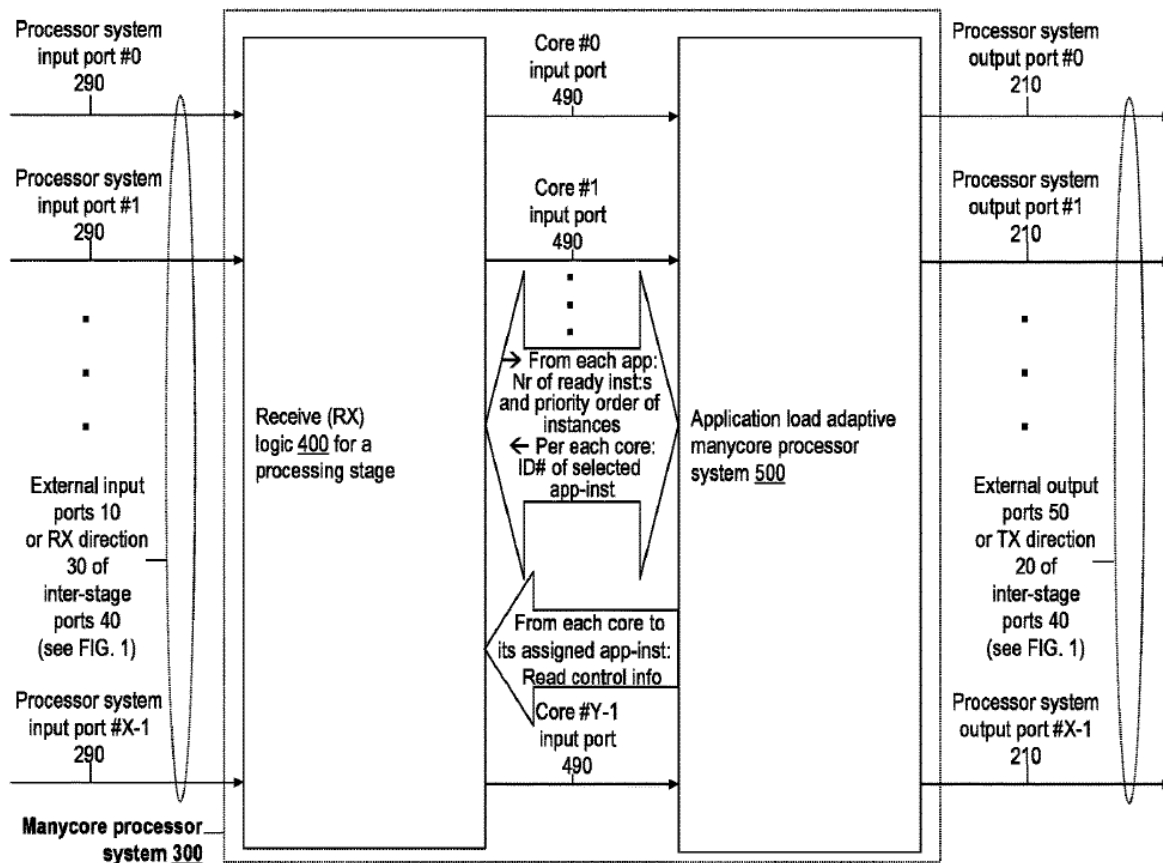
Id., FIG. 1.

47. As explained in the specification of the '306 Patent, in Figure 1, the “system 1 provides data processing services to be used by external parties (e.g. client portions of programs whose server portions run on the system 1) over networks,” “receives data units (e.g. messages, requests, data packets or streams to be processed) from its users through its RX network ports 10, and transmits the processing results to the relevant parties through its TX network ports 50.” *Id.*,

8:55-64. Data can be passed in the system through the cross-connect (XC) 200.

Id., 9:21-36. The XCs 200 include first-in first out buffers (FIFOs) 260, as illustrated in Figure 2. *Id.*, 10:45-11:12, FIG. 2.

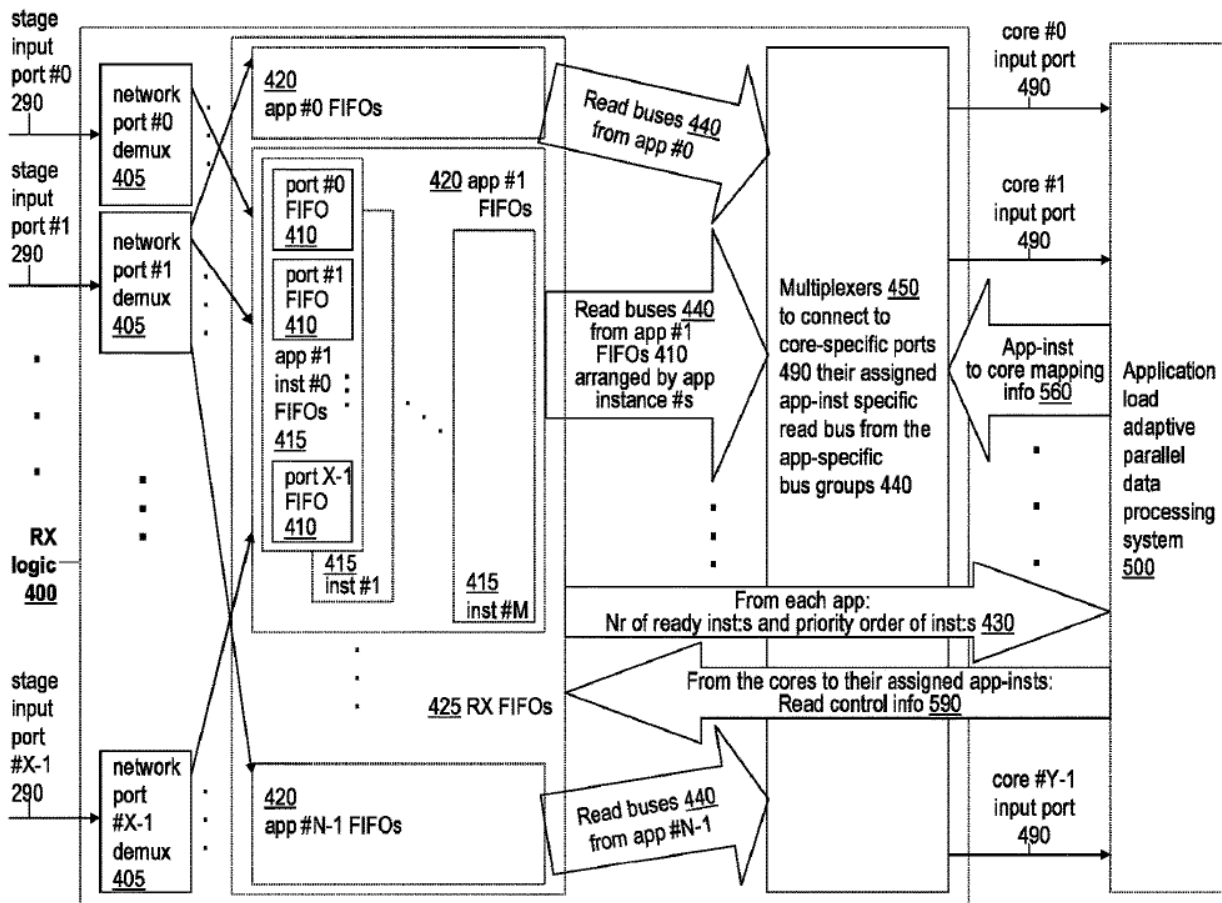
48. As explained in the specification of the '306 Patent, Figure 3 (reproduced below) shows a “high-level functional block diagram for any of the manycore processor systems in the multi-stage processor system.” *Id.*, 6:25-28, 12:23-29. The processing systems 300 include a “manycore processor system 500,” “an RX logic subsystem 400, which connects input data units (packets) from any of the input ports 290 to any of the processing cores of the manycore processor 500, according at which core their indicated destination app-task-instance may be executing at any given time.” *Id.*, 12:30-37.



Id., FIG. 3.

49. As explained in the specification of the '306 Patent, Figure 4 (reproduced below) shows “a functional block diagram for the input data receive logic subsystem for the manycore processor system.” *Id.*, 6:29-32, 12:46-53. “The RX logic connecting the input packets from the input ports 290 to the local processing cores arranges the data from all the input ports 290 according to their indicated destination applications and then provides for each core of the manycore processor 500 read access to the input packets for the app-task instance executing on the given core at any given time.” *Id.*, 12:54-60. In operation, input packets

arrive over the network input ports 290 and “are grouped to a set of destination application specific FIFO modules 420, whose fill levels (in part) drives the allocation and assignment of cores at the local manycore processor 500 among instances of the app-tasks hosted on that processor, in order to maximize the total (value-add, e.g. revenue, of the) data processing throughput across all the application programs configured for the manycore processor system.” *Id.*, 13:1-10.

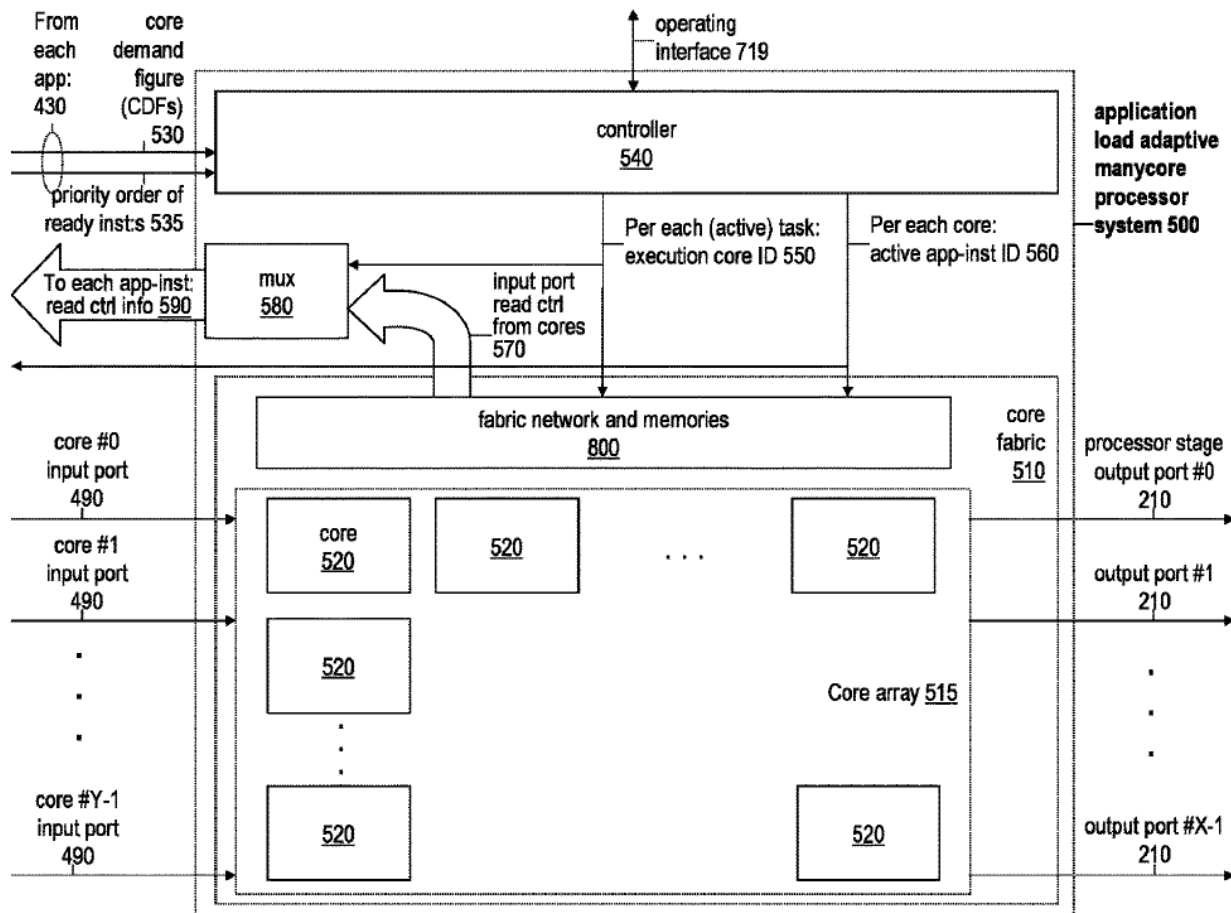


Id., FIG. 4.

50. The input packets are “connected 450 to specific cores of the processor 500 where their associated app-inst:s are executing at a given time.” *Id.*,

13:10-15. “Logic at each application scope FIFO module 420 signals 430 to the manycore processor system 500 the present processing load level of the application as a number of the ready to execute instances of the given app-task and, as well as the priority order of such instances.” *Id.*, 13:46-50. The “signals 430,” as referred to as “info 430 from the applications,” “expressed demands,” “capacity demand expressions,” and “info flow,” are used to map processor cores to tasks. 13:52-57, 14:6-15 (“For the info flow 430 (FIGS. 4 and 5), which is used for optimally allocating and assigning the cores of the processor 500 among the locally hosted app inst:s...”), 15:4-11, 15:16-18, 15:26-32.

51. As explained in the specification of the ’306 Patent, Figure 5 (reproduced below) shows a functional block diagram for the application load adaptive parallel data processing subsystem 500 for a given manycore processing system 300. *Id.*, 6:33-37. The processor system 500 “comprises an array 515 of processing cores 520, which are dynamically shared among a[sic] the locally hosted tasks of a set of application programs configured to run on the system 1.” *Id.*, 15:12-15. The demand information 430 (from Figure 4) issued to write “associated applications’ capacity demand indicators 530,” also referred to as “core-demand-figures (CDFs),” to the controller. *Id.*, 15:16-22. The CDFs “express how many cores 520 their associated app-task is presently able to utilize for its ready to execute instances.” *Id.*



Id., FIG. 5.

“A hardware logic based controller module 540 within the processor system 500, through a periodic process, allocates and assigns the cores 520 of the processor 500 among the set of applications 610 (FIG. 6) and their instances, at least in part based on the CDFs 530 of the applications.” *Id.*, 15:33-37.

52. The '306 further describes a process 700 of “app-inst to core mapping” for “maximizing the (value-add of the) application program processing throughput of the manycore fabric 510 shared among a number of software programs.” *Id.*, 16:36-42. The first step involves “allocating 710 the array 515 of

cores among the set of applications 610, based on CDFs 530 and CEs 717 of the applications, to produce for each application 620 a number of cores 520 allocated to it 715 (for the time period in between the current and the next run of the process 700).” *Id.*, 16:47-52. The second step involves two sub-steps that are performed “based at least in part on the allocating 710, for each given application that was allocated at least one core.” *Id.*, 16:53-64. The first sub-step is “selecting 720, according to the app-inst priority list 535, the highest priority instances of the given application for execution corresponding to the number of cores allocated to the given application,” and the second sub-step is “mapping 730 each selected app-inst to one of the available cores of the array 515, to produce, i) per each core of the array, an identification 560 of the app-inst that the given core was assigned to, as well as ii) for each app-inst selected for execution on the fabric 515, an identification 550 of its assigned core.” *Id.*

B. File History of the '306 Patent

53. As part of my preparation of this declaration, I reviewed the file history of the '306 Patent (EX1002). On September 28, 2020, the '306 Patent was filed as U.S. Patent Application No. 17/034/404 (“the '404 application”), which is a continuation of series of other patent applications and claims priority to a number of provisional applications dating back to the earliest date of November 4, 2011. EX1002, 309-310, 350-354.

54. Substantively, the '306 patent prosecution history is relatively short with no rejection issued by the Office. *See, generally, id.* The '404 application was filed with a single claim that Applicant canceled in a concurrently-filed preliminary amendment, adding new claims 2-11. *Id.*, 377-381. Applicant also filed a Track One request, which the Office granted thereafter. *Id.*, 304, 293. Two weeks after the Track One request was granted, Applicant filed yet another preliminary amendment, canceling claims 2-11 and adding another new set of claims, claims 12-37, alleging that the added claims “introduce no new matter.” *Id.*, 277-286.

55. The Office subsequently issued a Notice of Allowance with an examiner interview summary, both of which indicated that the examiner contacted Applicant about potentially allowable subject matter and that the parties agreed to an Examiner’s Amendment to move the case to allowance. *Id.*, 72-91. The Notice of Allowance identified Nollat as one of five references deemed the “closest art of record,” while not identifying the specific features that the art failed to teach or suggest. *Id.*, 87-89. Instead, the Notice stated only that the “prior arts of record ... do not expressly teach or render obvious the limitations recited in claims 12 and 14, when taken in the context of the claim as a whole.” *Id.*, 87 (emphasis omitted). Notably, however, the Notice stated that Nollat “teaches a plurality of *FPGAs*/reconfigurable devices being connected together in different topologies

and communicate to one another through the *buses* of the topologies; the FPGAs are used to execute *tasks of applications*, that have a plurality of tasks that are linked together; the tasks can be *relocated* on different FPGAs based on *monitored* execution status of the tasks.” *Id.* (emphasis in original). Applicant never substantively disputed this. While the Office considered Nollet, it never considered Tuan, Miller, or Redaelli, each of which is presented herein in a new invalidity ground that the Office never contemplated.

C. Interpretation of the '306 Patent Claims at Issue

56. For purposes of my analysis in this IPR proceeding, I understand that the terms that appear in the claims of the '306 Patent should be interpreted according to their plain and ordinary meaning. In determining the ordinary and customary meaning, I understand that the words of a claim are first given their plain meaning that those words would have had to a POSITA at the time of the alleged invention. I also understand that the structure of the claims, the specification and file history also may be used to better construe a claim insofar as the plain meaning of the claims cannot be understood. Moreover, I understand that even treatises and dictionaries may be used, albeit under limited circumstances, to determine the meaning attributed by a POSITA to a claim term at the time of filing. I have followed this approach in my analysis, and for all of the claim terms considered in this declaration, I have applied the plain and ordinary meaning of

those terms.

57. I also understand that the words of the claims should be interpreted as they would have been interpreted by a POSITA at the time the invention was made (not today). For purposes of my analysis here, I have used the November 4, 2011 priority date of the '306 Patent as the date of invention. Without exception, however, my analysis of the proper meaning of the recited claim elements in this declaration would be correct if the date of invention was anywhere in the early-to-mid 2010s.

VII. Overview of the Cited References

A. Nollet (EX1004)

58. Nollet is a patent publication titled “System and method for hardware-software multitasking on a reconfigurable computing platform.” EX1004, Title. Nollet is directed to a multi-processor computing platform that includes a reconfigurable device (e.g., Field Programmable Gate Array (FPGA)) that is adapted for dynamic hardware and software multitasking. EX1004, Abstract, [0058]. Nollet explains that operating systems (OS) that can “provide an environment where several tasks can run concurrently, with minimal of interference between them, but with support for safe data sharing” is known. *Id.*, [0006]. In these systems, the OS “should manage the available hardware resources in a consistent, efficient and fair way.” *Id.* If “multiple resources are available on

a computing platform, a scheduling problem can be defined” as “how to decide what task to run and where.” *Id.*, [0008]. These scheduling decisions “can be based on parameters like task priority, processor affinity,” and if pre-emptive scheduling is used, it is desirable to save the context of a current task in order for the task to continue when resources are later available. *Id.*

59. To address these issues, Nollet discloses a multi-tasking environment that is “capable of executing at least one application, composed of a plurality of tasks.” *Id.*, [0054]-[0055]. This environment includes a “reconfigurable hardware device, compris[ing] a plurality of small configurable logic blocks and an interconnect structure for interconnecting the configurable logic blocks,” where the hardware device “can be a logic gate array, e.g. a Field Programmable Gate Array (FPGA).” *Id.*, [0057]. When the hardware device is reconfigured, the “configuration or function of the logic blocks and/or the interconnection between the configurable logic blocks” are changed. *Id.*

60. The individual logic blocks are often referred to as “tiles” when they are “placed edge-to-edge in a tiled arrangement,” or array. *Id.*, [0058]-[0059]; *see also id.*, [0013] (“a reconfigurable matrix (for instance a gate array, especially an FPGA”). Each tile is “configurable in a variety of configurations,” where each configuration is “capable of running a hardware component or task” so that the “functionality of the hardware reconfigurable device” can be modified from

running a first task in a first configuration to running a “second (different) task” in a “second (different) configuration.” *Id.*, [0059].

61. In an example of such task switching and hardware reconfiguration, Nollet describes a scenario where a user of a portable device wishes to run two separate applications at the same time. *Id.*, [0107]. The user first starts playing a movie on a movie player, causing a movie decoder to begin performing hardware tasks. *Id.* “When the user is starting his movie decoder the first time, the reconfigurable hardware is free and completely available” so that “the movie player can be started on it, as a hardware task,” and the “computational power of the reconfigurable hardware allows playing the movie at full quality.” *Id.* However, later, user decides to play a 3D game on the device while still playing the movie, but “there are not enough resources available on the reconfigurable hardware to run both applications.” *Id.* As such, “the movie can be downscaled in resolution and frame rate, resulting in reduced computational requirements for the player,” which “frees up some hardware resources and allows the 3D game to be started as a hardware task.” *Id.* Nollet explains that, because these applications are run on a reconfigurable hardware, this problem can be solved without requiring different architectures for different multimedia content. *Id.* Specifically, Nollet discloses:

As an example, a user powers on a portable device to watch a movie. He presses the start button and begins enjoying the show. Some time

later, a scene is boring him. He wants to play a 3D game for a while, while the scene is going on. He will resume watching the movie after the scene. He therefore wants to have the movie being displayed in a small part of the screen. The computing platform is used in the portable device. It is composed of one or more ISP(s), one or more application specific integrated circuit(s) (ASIC) for functions like wireless communication or LCD management, and reconfigurable hardware. When the user is starting his movie decoder the first time, the reconfigurable hardware is free and completely available. As a result, the movie player can be started on it, as a hardware task. The computational power of the reconfigurable hardware allows playing the movie at full quality. **When the user decides to start the 3D game, there are not enough resources available on the reconfigurable hardware to run both applications. However the movie can be downscaled in resolution and frame rate, resulting in reduced computational requirements for the player. The movie player can therefore be rescheduled into one of the ISPs, as a software task. This frees up some hardware resources and allows the 3D game to be started as a hardware task.** For different multimedia content, different architectures are required. Consequently the two architectures would have to be present on the device to enable the same scenario. By running these applications on a reconfigurable hardware this problem is avoided. An architecture can be downloaded that fits best the application running on the device.

Id., [0107] (emphasis added).

62. In order to manage the different applications running on the hardware/software multitasking platform, Nollet provides for an operating system that manages tasks. *Id.*, [0086]. Each application running on the platform contains a number of tasks, each of which is run on an available hardware/software resource. *Id.* “The operating system preferably keeps track of the available resources, meaning location of all the tasks running. For example, the OS can keep track of which tile of an FPGA is being used to execute a task.” *Id.* In this way,

Nollet provides for a way of keeping track of which task is being run where (i.e., which task is being run on which FPGA tiles). *Id.* Specifically, Nollet discloses:

To create a true hardware/software multitasking environment, an operating system is provided by one aspect of the invention that manages the different applications running on the platform. The main function of the operating system is to manage tasks. An application running on the platform will contain a combination of tasks. Each of these tasks is to run on one of the available resources. When a new task is started or an old task is deleted, some of the existing tasks might have to be rescheduled onto another resource (e.g. to provide more computational power to the new task or to decrease the power consumption). In accordance with an aspect of the invention, the task creation and deletion consist in the usual task handling for a software task, and in reconfiguring the FPGA for a hardware task. **The operating system preferably keeps track of the available resources, meaning location of all the tasks running. For example, the OS can keep track of which tile of an FPGA is being used to execute a task.** Task rescheduling is also provided by one embodiment of the invention. State equivalence is identified between a software task and a hardware task. Indeed, when a task is rescheduled from hardware to software or vice-versa, the state information is transferred in order to resume the task where it has stopped.

Id., [0086] (emphasis added).

63. Nollet also describes a specific implementation of its reconfigurable multitasking computing platform that includes FPGAs as the reconfigurable hardware device “having a packet-switched **interconnection network**” provided by “a 2D torus using wormhole routing,” for example. *Id.*, [0110] (emphasis added); *see also id.*, [0034], [0035], FIG. 6.

B. Tuan (EX1005)

64. Tuan is a PhD thesis titled “A study on a multitasking environment for

dynamically reconfigurable processors.” Tuan, Cover. Tuan seeks to build “a true multitasking execution environment” on dynamically reconfigurable processors.

Tuan, Preface. Specifically, Tuan states:

Efficiently mapping applications partitioned into multiple threads of control or tasks onto a coarse-grained dynamically reconfigurable processing array is crucial for further exploiting the parallelism of applications. Basically, a certain speedup of reconfigurable devices over traditional microprocessors can be achieved by exploiting the inherent parallelism of a target application at a larger scale apart from instruction-level parallelism. However, in many cases, the parallelism of an application is not enough for reconfigurable array to be utilized efficiently. In such a condition, one of the methods to improve performance is to make the best use of stream-level pipelined execution; or in other words, to exploit the task-level parallelism. That is, a total process is partitioned into small independent **sequential processes, which are separately mapped onto a coarse-grained underlying reconfigurable array architecture comprising of multiple hardware execution units.** By investigating the trade-off between the size of a tile group and delay as well as execution time, an optimized mapping algorithm is proposed and evaluated compared with the single-process execution model and other mapping versions.

Tuan, Preface (emphasis added).

65. Like the '306 patent and Nollet, Tuan is directed to “multitasking on dynamically reconfigurable processors” and a parallel processing hardware architecture. EX1005, 2, 25-26, 102-103, FIG. 6.1. Like the '306 patent. Tuan considers “how to improve the throughput in data flow drive applications by using an appropriate application model and a mapping algorithm to map a target application onto a[n] underlying coarse-grained reconfigurable device consisting of multiple hardware executing units.” *Id.*, 2. It also considers “how to extend the

multi-thread execution model in a tile-based architecture into a multicore architecture.” *Id.*

66. Figure 6.1 of Tuan (reproduced below), for example, shows a “tile-based architecture” with 4x2 tiles and first-in first-out (FIFO) memory mechanisms “to exchange data between tasks mapped” to different groups of tiles. *Id.*, 72 (explaining that the FIFO “employs VMEMs between Tiles” and “is used as an inter-task communication method”), 102 (“Embedded memory modules within the reconfigurable array are used to form a communication mechanism between tasks assigned to two tiles...”), 103 (“A FIFO is for one-way communication and acts like a pipe.”). “A hardware task[] can be mapped to a tile for execution,” and “several neighboring tiles can be joined together to form a tile group.” *Id.*, 102.

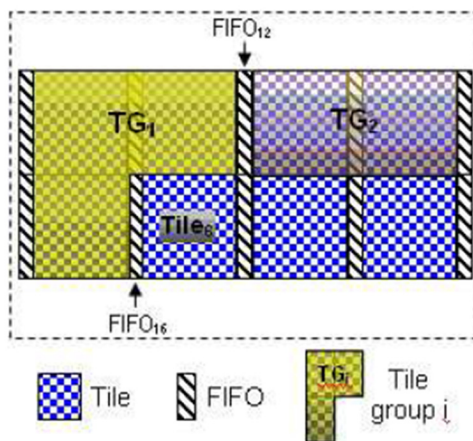


Fig. 6.1: Tile-based architecture

Id., Fig. 6.1.

67. While Tuan discloses that tasks can be mapped to a tile in its

discussion of Figure 6.1, it further explains that the tasks assigned to the tiles can be switched. *Id.*, 30-32, FIG. 2.8. In particular, with reference to Figure 2.8 (reproduced below), Tuan shows “task-switching” by the reconfigurable array, explaining that [w]hen a task completes or is preempted, the area it occupies is freed, and a new task may take over the area; so, when that task is resumed for continuing execution, it is likely to be assigned to a new area.” *Id.*, 31.

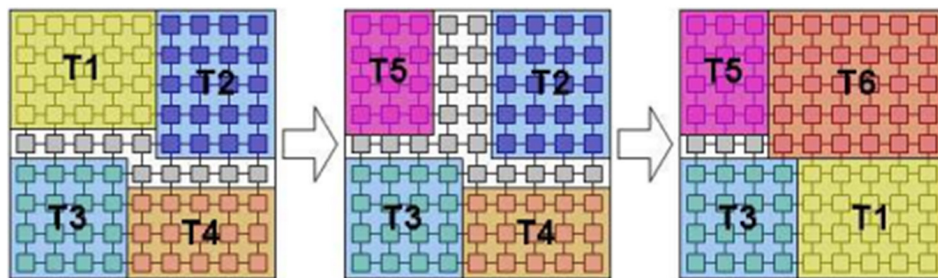


Fig. 2.8: Time-sharing, two-dimensional partitioning model

Id., FIG. 2.8.

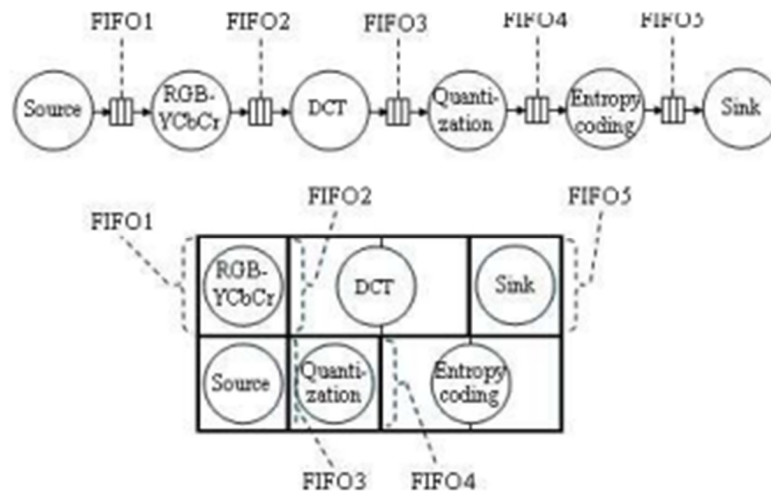
68. With reference to Figure 2.8, Tuan explains:

[A]t the beginning, four tasks *T1*, *T2*, *T3* and *T4* are allocated for execution. **After some times, task *T1* is preempted and removed, task *T5* comes and takes over a part of area that was occupied by *T1* previously.** Then, when tasks *T2* and *T4* complete, task *T6* is scheduled and *T1* is resumed. This time, task *T1* is allocated to a new area. As shown, the external fragmentation, which is fragmentation of area outside the rectangular boundary of all tasks, changes from time to time.

Id., 32 (emphasis added). Task switching in Tuan occurs when task *T1* is preempted and removed and task *T5* occupies the processing unit—that is, task *T1*

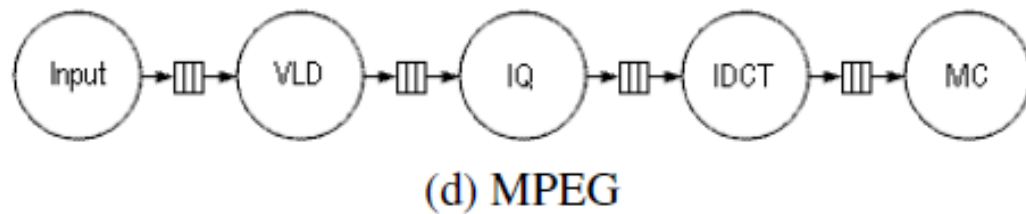
is switched out for task T5. *Id.* Similarly, task switching occurs when task T6 is scheduled and T1 resumed after tasks T2 and T4 are complete. *Id.*

69. To implement its parallel processing platform, Tuan specifies a mapping step where target applications are “mapped onto a candidate architecture for evaluation.” EX1005, 65. In this mapping process, “a total job” of an application “is represented with multiple processing tasks that can be executed in a pipelined manner.” *Id.*, 66. For example, a JPEG encoder is mapped in Figure 4.3 to include multiple processing tasks: a source stage, a sink stage, and multiple intermediate processing stages, including RGB-YCbCr conversion, two-dimensional DCT, quantization, and entropy coding stages. *Id.*, 67, 73-74. Each task, or stage, is executed by a set of reconfigurable logic-based processing units (e.g., tiles), and separated from a neighboring task by a FIFO. *Id.*, 73, 75.



Id., FIG. 4.3.

70. In another mapping example, Tuan maps an MPEG decoder, which is a type of movie decoder, in Figure 4.9(d) (reproduced below). *Id.*, 74-75, FIG. 4.9. Like the JPEG encoder, the MPEG decoder is mapped to multiple processing tasks: “input, variable length decode (VLD), inverse quantization (IQ), inverse discrete cosine transform (IDCT), and motion compensation (MC),” each of which is separated from a neighboring task by a FIFO. *Id.*, 75.



EX1005, FIG. 4.9.

71. With respect to its system architecture, Tuan lists various components, including user interface components, OS service provider, hardware abstraction layer, interconnection network, and underlying configurable device. EX1005, 27. Tuan further describes an interface between multiple users, the OS, and the hardware, specifying that “users can input commands and execute applications via the interface.” *Id.*

C. Miller (EX1006)

72. Miller is a patent publication titled “Providing dynamically scaling computing load balancing” and directed at “providing load balancing functionality among multiple computing nodes.” EX1006, Title and Abstract.

73. As explained in the specification, Miller relates to managing groups of computing nodes, where each group of computing nodes executes either a single application or a set of related applications in a multi-tiered architecture. EX1006, 3:8-13. Miller further discloses that its system can support multiple users: “at least some of the computing nodes of a group for a user or other entity are part of one or more physical computer networks in one or more geographical locations and under control of the user or other entity, and at least some of the dynamic scaling of the group uses one or more other computing nodes that are part of a remote computer network (e.g., a virtual computer network provided under the control of a network-accessible service available to remote users).” *Id.*, 2:6-14.

74. At a high level, Miller is directed to “providing load balancing functionality among multiple computing nodes.” EX1006, Abstract. More specifically, Miller provides for a Dynamic Computing Scaling (DCS) Load Balancer module that “dynamically scale[s] the computing nodes of a group when defined criteria are satisfied,” for example, by increasing the number of computing nodes in a group based on demand. *Id.*, 2:19-25, 3:23-25, 4:7-26. Miller also teaches a broad variety of such criteria that the DCS Load Balancer module may use, including measures of demand such as volume of requests received or handled by each computing node. *Id.*, 3:30-55. The DCS Load Balancer module monitors the load of a group of computing nodes and allocates more computing nodes when

the load is high or when there is an increased demand (e.g., greater than a threshold). *Id.*, 22:19-67, 28:64-29:17. For example, Miller discloses that in response to an increased demand for the functionality of the group computing nodes, another computing node can be “repurpose[ed]” to provide the necessary functionality. *Id.*, 29:3-17.

75. Miller additionally teaches that the computing nodes in its system “may comprise any combination of hardware or software,” including computers, database servers, network storage devices, and the like. *Id.*, 30:14-32.

Additionally, the systems described in Miller can be implemented in firmware and/or hardware, including field-programmable gate arrays (FPGAs), complex programmable logic devices (CPLDs), and the like. *Id.*, 30:60-31:2

D. Redaelli (EX1007)

76. Redaelli is a paper titled “Reconfiguration Aware Task Scheduling for Multi-PFPGA Systems.” EX1007, 65 (Title). More particular, Redaelli discusses “sharing the computational environment and task placement in partially and dynamically reconfigurable FPGA clusters deployed for data intensive high performance computing applications” using a “scheduling solution [that] suggest[s] the number of FPGA that an online scheduler should later use for a known situation or for a situation similar to a known one.” *Id.*, 65 (Abstract) . When scheduling tasks, Redaelli’s system “looks for module reuse opportunities

and places all the tasks that can be reused,” and its schedule ensures that “[m]odules are left intact on the FPGA until other tasks require their space, in order to increase the probability of reuse.” *Id.*, 70.

VIII. ANALYSIS

77. For the reasons articulated in detail below, and based on my review of the ’306 Patent, the file history, and the prior art references cited here, it is my opinion that a POSITA would have readily understood that the Nollet-Tuan-Miller Combination provides all the elements of claims 1-24. Further, it is my opinion that the Nollet-Tuan-Miller-Redaelli Combination provides all the elements of claims 1-24.

A. [GROUND 1A & 1B] – Nollet in view of Tuan and Miller or Nollet in view of Tuan, Miller, and Redaelli Renders Claims 1-24 Obvious

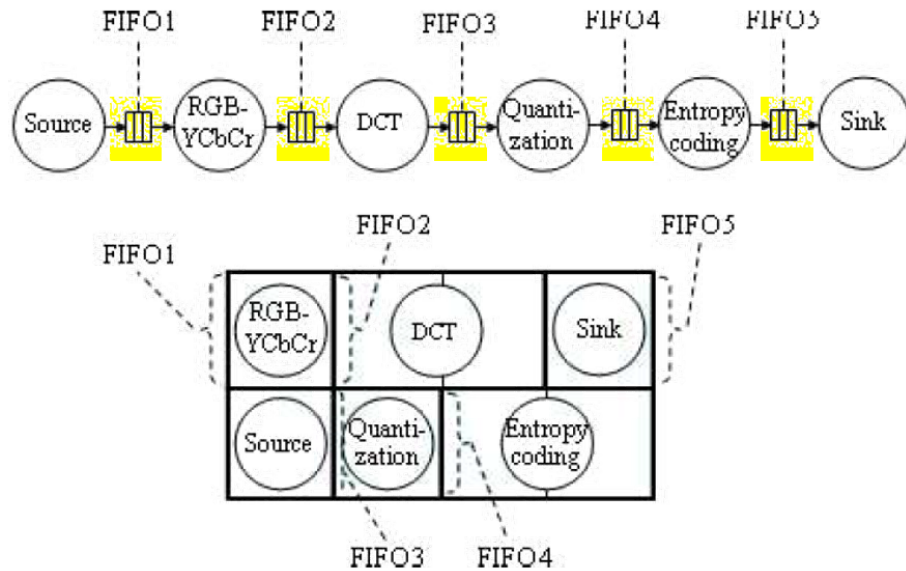
1. The Nollet-Tuan-Miller Combination

78. As discussed above, both Nollet and Tuan are directed to multitasking using reconfigurable logic-based processing units. *See supra* Section VII. Nollet is directed to a multi-processor computing platform with a reconfigurable device (e.g., Field Programmable Gate Array (FPGA)) that is adapted for dynamic hardware and software multitasking. EX1004, Abstract, [0017], [0058], FIG. 1. Nollet provides a method for execution of one or more applications by a reconfigurable matrix that includes individually reconfigurable tiles that can be configured to perform hardware tasks. *Id.*, [0017], FIG. 1. The hardware tasks can

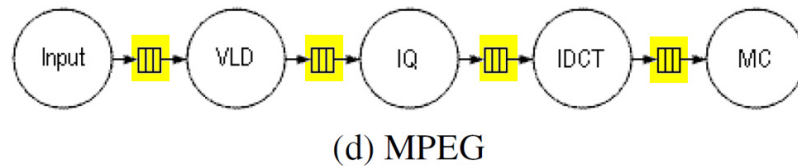
communicate via the paths illustrated in Figure 1. *Id.*, [0129]-[0133], FIG. 1. In an example implementation, the reconfigurable hardware on a portable device responds to a user first watching a movie on the device and later wanting to play a 3D game while the movie is still playing. *Id.*, [0107]. In this scenario, the reconfigurable hardware is first used to decode the movie for the movie player, and later, at least some of the hardware is freed up to perform 3D game-related tasks to facilitate play by the user. *Id.*

79. Similarly, Tuan is directed to “multitasking on dynamically reconfigurable processors” and a parallel processing hardware architecture with reconfigurable logic-based processing units, or tiles. EX1005, 2, 25-26, 102-103, FIG. 6.1. The individual tiles can be configured to execute particular tasks, where an task of an application is mapped onto a tile or tile group, and, as needed, the tiles can be reconfigured and switched to perform other tasks. *Id.*, 30-32, 73, FIG. 2.8. Tuan further explains that to perform the mapping of tasks to tiles, “a total job is represented with multiple processes tasks that can be executed in a pipelined manner.” *Id.*, 66. Tuan provides examples of representing the tasks of JPEG encoding (FIG. 4.3 (reproduced below)) and MPEG decoding (FIG. 4.9(d) (reproduced below)). *Id.*, 67, 73-75, FIGS. 4.3, 4.9. Each stage, represented by a circle, is executed by one or more reconfigurable tiles, and neighboring stages are connected via a first-in first-out (FIFO) memory mechanism (highlighted in FIGS.

4.3 and 4.9(d) below), “which employs VMEMs between Tiles, is used as an inter-task communication method.” *Id.*, 66 (“Neighboring Tile groups can be communicated with each other through FIFOs formed with the distributed memory modules.”), 72-73, 75.



EX1005, FIG. 4.3.



(d) MPEG

EX1005, FIG. 4.9 (highlighted).

80. Tuan provides further details about its system. First, Tuan explains that its system includes “an interface between the users, the OS and the hardware” and that “users can input commands and execute applications via the interface.” *Id.*, 27. Next, Tuan discloses an “network manager” and memory that collectively

support inter-task communication and manages “information about where tasks are located.” *Id.*, 15-16, 27-29. These components assist in keeping track the location of tiles where a task is mapped. *Id.*

81. Nollet and Tuan both relate to parallel processing on dynamically-reconfigurable hardware, including assigning individually-reconfigurable tiles to tasks of an application. EX1004, Abstract, [0017], [0058], FIG. 1; EX1005, 2, 25-26, 30-32, 73, 102-103, FIG. 6.1. Tuan explicitly teaches representing the processes of an application as multiple stages or tasks, and Nollet explains that hardware tasks communicate, which would suggest to a POSITA that those tasks are part of a multi-stage program. EX1004, [0132], FIG. 1; EX1005, 67, 73-75, FIGS. 4.3, 4.9; *see also* EX1004, [0148]-[0149]. A POSITA would have found it obvious to implement the reconfigurable multitasking computing platform of Nollet with the task mapping architecture, network manager, and memory of Tuan. A system based on the teachings of both Nollet and Tuan would thus facilitate mapping tiles to tasks of an application by representing the tasks of an application in a pipelined manner, which “makes it easier to partition an application into a set of parallel communicating tasks,” which is “suitable to be implemented on reconfigurable arrays,” such as Nollet’s reconfigurable device. EX1005, 66.

82. The combined system would further include FIFOs, network manager, and memory to facilitate Nollet’s teaching of inter-task communication and to help

manage “information about where tasks are located” after tasks are mapped to tiles or cores. EX1004, [0129]-[0133], FIG. 1; EX1005, 15-16, 27 (“In a reconfigurable system, several communication mechanisms can be exploited to transfer data between tasks such as memory, non-shared direct hardware channels, or an interconnection network. One of the simplest methods to transfer data between two hardware tasks is to use embedded memory modules with a suitable arbitrator.”); *see also* EX1005, 28-29, 66, 72-73, 75. A POSITA would have been motivated to provide such a combination to implement a mapping approach and application structure “to improve throughput and reduce fragmentation” and to “support[] inter-task communication and input/output operations.” EX1005, 28-29.

83. Miller relates to managing groups of computing nodes, where each group of computing nodes executes either a single application or a set of related applications in a multi-tiered architecture, supporting remote users over a network. EX1006, 2:6-14, 3:8-13. Like Nollet and Tuan, Miller further contemplates the use of reconfigurable devices (e.g., FPGAs) in its system. *Id.*, 30:50-31:2. To manage the multiple computing nodes, Miller provides for a Dynamic Computing Scaling (DCS) Load Balancer module that “dynamically scale[s] the computing nodes of a group when defined criteria are satisfied,” for example, by increasing the number of computing nodes in a group based on

demand (volume of requests). *Id.*, 2:19-25, 3:23-25, 4:7-26. For example, the DCS Load Balancer module monitors the load of a group of computing nodes and allocates more computing nodes, potentially by repurposing the functionality of those nodes for a new task, in response to increased load or increased demand. *Id.*, 22:19-67, 28:64-29:17.

84. Combining the teachings of Nollet with Tuan and Miller would have merely involved combining prior art elements according to known methods to yield predictable results. Nollet and Tuan are directed to similar systems that assign reconfigurable tiles of a hardware device to tasks of an application. A POSITA would have been motivated to modify Nollet's platform with Tuan's explicit inclusion of FIFOs (which are equivalent to the queues disclosed in Nollet, a network manager, and memory to facilitate communications between hardware tasks, keep track of the locations of tiles where each application task is assigned, and managing distribution of tasks to hardware resources. EX1004, [0104], EX1005, 15-16, 27-29, 66, 72-73, 75. The use of these components in Tuan would align with Nollet's teaching or suggestion that its platform supports each of these functions. EX1004, [0081] (discussing "manag[ing] the applications by distributing the different tasks on the available resources"), [0086] (discussing "keep[ing] track of which tile of an FPGA is being used to execute a task"), FIG. 1 (showing communication paths between tasks, suggesting multi-stage programs).

A POSITA would have understood Nollet’s teaching of communication paths between tasks as suggesting the inclusion of multi-stage programs and naturally looked to related references such as Tuan that describe how programs can be partitioned into multiple “tasks that can be executed in a pipelined manner,” which “allows for parallelism and pipelined processing, which are suitable to be implemented on reconfigurable arrays,” such as those taught in Nollet. EX1005, 66. Moreover, a POSITA would have understood Nollet’s discussion of keeping track of which FGPA tile is being used to execute a task as requiring the use of a system for managing this information and thus naturally looked to related references such as Tuan that describe how “information about where tasks are located” for execution by FPGA tiles is stored and updated using FIFOs for inter-task communication, a network manager, and memory. EX1005, 15-16, 27-29, 66, 72-73, 75.

85. Nollet and Miller are both directed to efficient resource sharing among applications, including with the use of reconfigurable devices, such as FPGAs. A POSITA would have been motivated to modify Nollet’s computing platform with Miller’s DCS Load Balancer module to monitor the load of Nollet’s reconfigurable processing units, and, in response to increased load or demand, repurpose or reconfigure processing units for other tasks. EX1006, 22:19-67, 28:64-29:17. A POSITA would have been further motivated to apply Miller’s load

balancing system to Nollet's computing platform to create "a virtual computer network provided under the control of a network-accessible service available to remote users." *Id.*, 2:6-14. Indeed, Miller explains that virtualization technologies for hardware "allow[] various computing resources to be efficiently and securely shared by multiple customers." *Id.*, 1:26-42. This approach is consistent with Nollet's teaching of "manag[ing] the available hardware resources in a consistent, efficient and fair way." EX1004, [0006].

86. The combination of Nollet, Tuan, and Miller would have been predictable and foreseeable because 1) Nollet and Tuan both teach parallel processing on dynamically-reconfigurable hardware by assigning individually-reconfigurable tiles to tasks of an application, 2) Nollet and Miller both focus on efficient resource sharing among applications, and 3) the combination merely involves incorporating known features that are explicitly disclosed in Tuan and/or Miller and would have been understood by a POSITA as likely present but not explicitly disclosed by Nollet. *see, e.g.*, EX1004, [0087], [0159], FIG.1 (showing communication paths between tasks, suggesting multi-stage programs). The combination is predictable, at least in part, because elements of the combined system would perform similar functions that had been known to perform prior to the combination. For example, the computing platform of Nollet would still perform the same or similar functions, including assigning tasks of applications to

FPGA tiles, as it did prior to the combination with Tuan and Miller. Applying and combining these concepts would have been obvious from a POSITA's perspective, and a POSITA would have had a reasonable expectation of success in doing so. The combined teachings of Nollet, Tuan and Miller would have rendered obvious claims 1-24 of the '306 patent, and the combination discloses each limitation of the claims as described below.

2. The Nollet-Tuan-Miller-Redaelli Combination

87. As discussed *supra* Section VIII.A.1, the Nollet-Tuan-Miller combination provides a computing platform that allows for parallel processing of applications using a dynamically reconfigurable device with individually programmable units that can be assigned to tasks of the applications and switched between tasks depending on program demand.

88. Similarly, Redaelli is directed to “sharing the computational environment and task placement in partially and dynamically reconfigurable FPGA clusters deployed for data intensive high performance computing applications” using a “scheduling solution [that] suggest[s] the number of FPGA that an online scheduler should later use for a known situation or for a situation similar to a known one.” EX1007, 65 (Abstract). In particular, Redaelli teaches “look[ing] for module reuse opportunities and plac[ing] all the tasks that can be reused” and leaving modules “intact on the FPGA until other tasks require their space, in order

to increase the probability of reuse.” *Id.*, 70.

89. A POSITA would have found it obvious to implement the Nollet-Tuan-Miller combination in light of Redaelli’s scheduling solution for reconfigurable FPGAs. Notably, Nollet stresses the importance of “manag[ing] the available hardware resources in a **consistent, efficient and fair way.**” EX1004, [0006] (emphasis added). Given the desire to allocate hardware resources efficiently, a POSITA would have naturally looked to Redaelli’s teaching of a scheduler that looks for “reuse opportunities” and “increase[s] the probability of reuse.” EX1007, 70. Such a combination would have resulted in a system where the configurable tiles of the Nollet-Tuan-Miller combination are efficiently allocated in a manner that minimizes the necessity of reconfiguring the tiles because reusing tiles (without reconfiguration) is prioritized. Minimizing reconfiguration would further “minimize execution latency,” which Nollet teaches is desirable for the system. EX1004, [0085]; *see also id.*, [0089] (“reconfiguration speed/latency”), [0140] (“switch latency”).

90. Combining the teachings of Nollet, Tuan, and Miller with Redaelli would have merely involved combining prior art elements according to known methods to yield predictable results. Like Nollet, Redaelli is directed to partially and dynamically reconfigurable FPGAs. A POSITA would have been motivated to implement the Nollet-Tuan-Miller platform with Redaelli’s scheduler that looks

for “reuse opportunities” and “increase[s] the probability of reuse.” The combination of Nollet, Tuan, Miller, and Redaelli would have been predictable and foreseeable because 1) Nollet, Tuan, Miller, and Redaelli all teach efficient resource allocation of hardware devices, and 2) the combination merely involves incorporating known features that are explicitly disclosed by Redaelli and would have been understood by a POSITA as likely present, but not explicitly disclosed by Nollet. *See* EX1004, 6 (“consistent, efficient and fair way”), [0085] (“minimize execution latency”), [0089] (“reconfiguration speed/latency”), [0140] (“switch latency”). The combination is predictable, at least in part, because elements of the combined system would perform similar functions that they had been known to perform prior to the combination. For example, the computing platform of Nollet would still perform the same or similar functions, including assigning and executing parallel tasks to FPGA tiles, as it did prior to the combination with Tuan, Miller, and Redaelli.

91. The combined teachings of Nollet, Tuan, Miller, Redaelli would have rendered obvious claims 1-24 of the ’306 patent, and the combination discloses each limitation of the claims as described below.

Claim 1

[1.0] A method for task-switching on a multi-user parallel processing hardware architecture comprising a plurality of reconfigurable logic-based processing units, the method comprising:

92. It is my opinion that the Nollet-Tuan-Miller discloses a method for task-switching on a multi-user parallel processing hardware architecture comprising a plurality of reconfigurable logic-based processing units.

93. Nollet discloses a parallel processing hardware architecture comprising a “reconfigurable matrix [that] can be subdivided into smaller processing units or “tiles.” EX1004, [0017]. Nollet explains, “Preferably, the reconfigurable device, comprises a plurality of tiles, each tile being configurable in a variety of configurations, each of the tile configurations being capable of running a hardware task.” *Id.*, [0020]. Nollet further discloses task-switching:

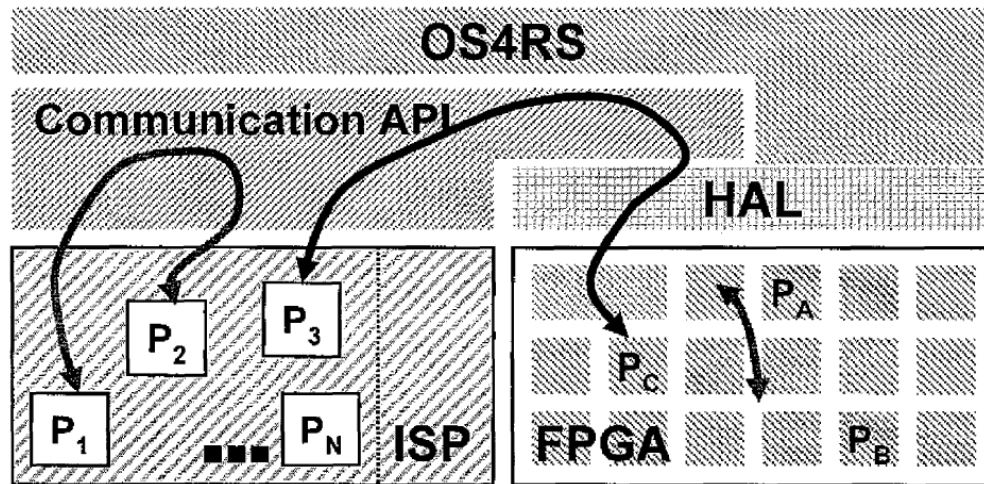
[0020]The configuring of the reconfigurable device is preferably carried out at run-time of the application. The configuring of the reconfigurable device is preferably transparent to a user running the application. The hardware tasks can be written in a description language and the hardware tasks written in the description language can be automatically converted into a hardware bitstream description. **Preferably, the reconfigurable device, comprises a plurality of tiles, each tile being configurable in a variety of configurations, each of the tile configurations being capable of running a hardware task and wherein the configuring of the reconfigurable device such that the at least one new hardware task other than the first plurality of hardware tasks can be executed, changes at least one of the tiles from one configuration to another configuration.**

Id., [0020].

94. Figure 1 (reproduced below) shows an exemplary hardware

architecture comprising an FPGA array having a plurality of reconfigurable tiles.

Id., [0037], FIG. 1. Because each tile is “configurable in a variety of configurations,” a POSITA would have understood that each tile is a reconfigurable logic-based processing unit. Nollet also discloses task-switching by disclosing switching from movie tasks to 3D game tasks. Nollet, [0107].



EX1004, FIG. 1.

95. Like Nollet, Tuan discloses a parallel processing hardware architecture. EX1005, 25-26, 102-103, FIG. 6.1. In particular, Figure 6.1 of Tuan discloses a tile-based parallel processing hardware architecture where reconfigurable logic-based processing units (specifically, DRPA tiles) are connected using a FIFO model. EX1005, 25-26, 102-103, FIG. 6.1.

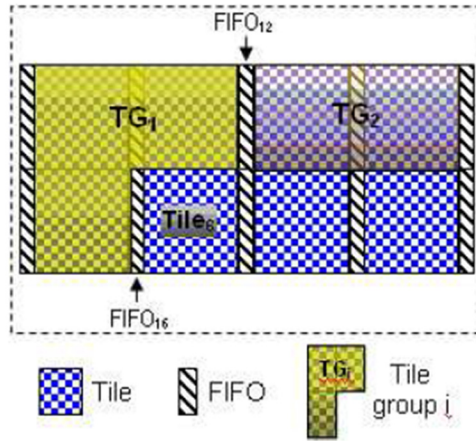


Fig. 6.1: Tile-based architecture

Id., FIG. 6.1.

96. Figure 2.8 of Tuan shows “task-switching” on this parallel processing hardware architecture using a plurality of reconfigurable logic-based processing units. *Id.*, 30-32, FIG. 2.8.

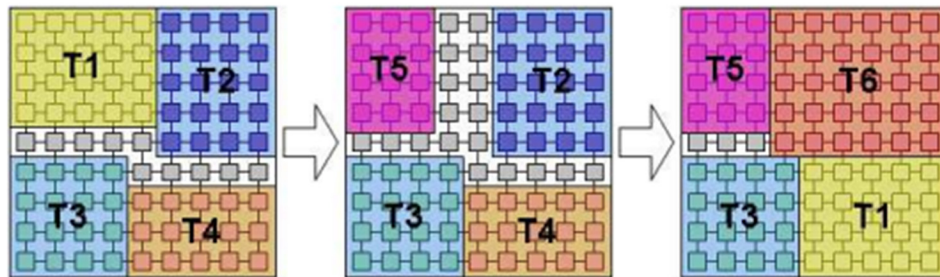


Fig. 2.8: Time-sharing, two-dimensional partitioning model

Id., FIG. 2.8.

97. Tuan explains:

[A]t the beginning, four tasks $T1$, $T2$, $T3$ and $T4$ are allocated for execution. **After some times, task $T1$ is preempted and removed, task $T5$ comes and takes over**

a part of area that was occupied by T1 previously. Then, when tasks *T2* and *T4* complete, task *T6* is scheduled and *T1* is resumed. This time, task *T1* is allocated to a new area. As shown, the external fragmentation, which is fragmentation of area outside the rectangular boundary of all tasks, changes from time to time.

Id., 32 (emphasis added). Task switching in Tuan occurs when task *T1* is preempted and removed and task *T5* occupies the processing unit—that is, task *T1* is switched out for task *T5*. Similarly, task switching occurs when task *T6* is scheduled and *T1* resumed after tasks *T2* and *T4* are complete.

98. Additionally, Tuan discloses “an interface between the **users**, the OS and the hardware” and that “**users** can input commands and execute applications via the interface.” EX1005, 27 (emphasis added). Because Tuan explicitly acknowledges the involvement of multiple “users” using the disclosed hardware, a POSITA would have understood that Tuan discloses a multi-user system. Furthermore, it would have been obvious to a POSITA that multiple users share Tuan’s system to improve the efficient use of multiple expensive resources.

99. Miller further discloses multi-user systems. EX1006, 2:6-14. In particular, Miller discloses “a virtual computer network provided under the control of a network-accessible service available to remote **users**.” *Id.* (emphasis added). Moreover, the use of parallel processing systems—such as the systems of Nollet or Tuan—by multiple users was well-known in the art before the Critical Date.

100. Alternatively, with respect to Ground 1B, Redaelli teaches multi-user parallel processing hardware architecture comprising a plurality of reconfigurable logic-based processing units. Specifically, Redaelli teaches, “partially and dynamically reconfigurable systems can also be effective when they serve in a **multi-user** environment for application domains that are relatively well defined (e.g., a high performance computing node for scientific computing).” *Id.*, Abstract (emphasis added).

[1.1] linking, through a first set of inter-task communication paths of the multi-user parallel processing hardware architecture, a first set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a first multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a first program,

101. It is my opinion that the Nollet-Tuan-Miller combination discloses linking, through a first set of inter-task communication paths of the multi-user parallel processing hardware architecture, a first set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a first multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a first program.

102. A POSITA would have understood that a multi-stage program includes a set of tasks (stages) that work cooperatively to accomplish a computing goal. The tasks cooperate by exchanging data through a communication path.

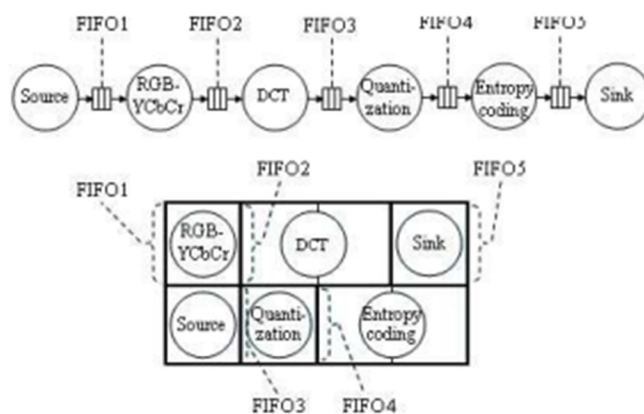
103. Nollet provides “a method for **execution of** an application or a

plurality of applications, the execution requiring running substantially simultaneously a **plurality of tasks** (multi-tasking), on the platform.” EX1004, [0017] (emphasis added). This method provides for dynamic instantiation of hardware tasks “by exploiting partial reconfiguration” by subdividing the reconfigurable device into **tiles that are individually reconfigurable**. *Id.* (“hence fine-grain configurability is exploited”) (emphasis added). That is, Nollet’s platform allows individual tiles of a hardware device to be reconfigurable to perform different hardware tasks of one or more applications. *Id.*, [0017], FIG. 1 (showing different tiles executing different hardware tasks P_A, P_B, and P_C on an FPGA).

104. In an example scenario, Nollet discloses configuring a first set of FPGA tiles to execute a first set of tasks (movie decoder tasks) when a user starts watching a movie. *Id.*, [0107]. Nollet also discloses interconnecting the logic blocks of its reconfigurable hardware device (e.g., FGPA) to change the configuration or function of the logic blocks. *Id.*, [0057]. Nollet further explains that “[m]essages 5 between tasks, both scheduled in hardware 8, are routed inside the interconnection network,” where example communication paths between tasks are shown in Figure 1. *Id.*, [0132], FIG. 1. A POSITA would have understood that because the hardware tasks communicate, then the tasks are likely part of a multi-stage program instance. Moreover, Nollet discloses an example video decoder

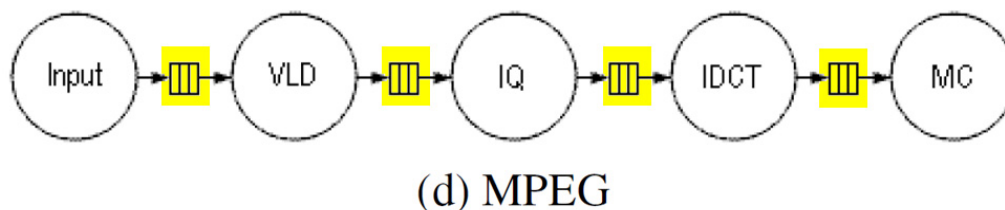
where “a send thread 100 passes the coded frames one by one to the decoder thread 102, 106,” which “decodes the frames and sends them, one macroblock at a time, to a receive thread 104 that reconstructs the images and displays them,” and which a POSITA would have found further suggests the multi-stage nature of the programs in Nollet. EX1004, [0148]-[0149].

105. To the extent that Nollet does not explicitly state that its movie decoder is a multi-stage program instance, Tuan discloses this. Preliminarily, Tuan discloses linking a first set of reconfigurable tiles into a multi-stage program instance by reconfiguring the communication paths. EX1005, 66, FIGS. 4.3 and 4.9, Table 4.2. For example, Tuan shows that a JPEG encoder in Figure 4.3, which includes a source stage, a sink stage, and multiple intermediate processing stages, including RGB-YCbCr conversion, two-dimensional DCT, quantization, and entropy coding stages. *Id.*, 67, 73-74. Each of these stages is executed by a set of reconfigurable logic-based processing units (e.g., tiles). *Id.*, 73 (“An application is mapped onto the [reconfigurable] architecture in a way that each task is mapped onto a tile or a Tile Group according to the mapping algorithm and each task’s FIFO is mapped one-to-one onto [a corresponding] FIFO.”), 75.



EX1005, FIG. 4.3.

106. In another example of interconnecting reconfigurable tiles to create a multi-stage applications (i.e., program instance)—like JPEG encoding, Tuan discusses movie decoders. *Id.*, 74-75. In particular, Figure 4.9(d) of Tuan (reproduced below) shows an example of a movie decoder—such as the movie decoder of Nolllet—in the form of an MPEG decoder. *Id.*, 74-75, FIG. 4.9. Like the JPEG encoder, the MPEG decoder of Tuan is a multi-stage program instance that includes multiple computation stages including “input, variable length decode (VLD), inverse quantization (IQ), inverse discrete cosine transform (IDCT), and motion compensation (MC).” *Id.*, 75. Thus, a POSITA would have understood that Nolllet’s movie decoder is a multi-stage program instance, at least based on Tuan’s more detailed teachings involving the multiple tasks of movie decoders.



EX1005, FIG. 4.9 (highlighted).

107. The Nollet-Tuan-Miller combination further discloses that the linking of the processing units (stages) is performed through a first set of inter-task communication paths. EX1005, 66-67, 73-75, FIGS. 4.3 and 4.9. In particular, the MPEG decoder includes a number of stages, and though not explicitly labeled in Figure 4.9(d), each of the stages of the MPEG decoder is connected to an adjacent stage via a FIFO (highlighted above), through which communication is performed. *Id.*, 66-67, 73, 75, FIG. 4.9(d). Regarding the FIFOs, Tuan explains that “[a] first-in first-out (FIFO) memory mechanism, which employs VMEMs between Tiles, is used as an inter-task communication method,” and “[n]eighboring Tile groups can be communicated with each other through FIFOs formed with the distributed memory modules.” EX1005, 66, 72. Thus, a POSITA would have understood that the stages of the MPEG decoder are connected via a first set of inter-task communication paths (FIFO memory mechanism) in view of Tuan’s Figure 4.3, which includes labels for each of the FIFOs. EX1004, FIG. 1 (showing communication paths between tasks).

108. For at least the reasons discussed above in Section VIII.A.1, a

POSITA would have been motivated combine Nollet and Tuan.

[1.2] wherein each of the first set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of first tasks of the first program corresponding to a respective processing stage of the first multi-stage program instance;

109. It is my opinion that the Nollet-Tuan-Miller combination discloses that each of the first set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of first tasks of the first program corresponding to a respective processing stage of the first multi-stage program instance.

110. As discussed above for element [1.1], Nollet discloses configuring a first set of reconfigurable logic-based processing units (e.g., FPGA tiles) to perform a first set of tasks (movie decoder tasks). EX1004, [0107], FIG. 1. In that configuration, each of the reconfigurable units is programmed to perform a respective task of the first set of tasks (movie decoder tasks). *Id.* As further discussed in element [1.1], it would have been obvious to a POSITA that the movie decoder of the Nollet-Tuan-Miller combination is a multi-stage program instance, so the movie decoder tasks would thus correspond to a respective processing stage (e.g., VLD, IQ, IDCT) of the movie decoder. EX1004, [0107]; EX1005, 73-75, FIG. 4.9(d).

[1.3] linking, through a second set of inter-task communication paths of the multi-user parallel processing hardware architecture, a second set of reconfigurable logic-based processing units of the plurality of reconfigurable

logic-based processing units into a second multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a second program,

111. It is my opinion that the Nollet-Tuan-Miller combination discloses linking, through a second set of inter-task communication paths of the multi-user parallel processing hardware architecture, a second set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a second multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a second program.

112. Nollet discloses linking a second set of reconfigurable logic-based processing units (e.g., FPGA tiles) to perform a second set of tasks (e.g., 3D gaming tasks) of a second program (3D game). EX1004, [0017], [0107]; *see supra* discussion of element [1.1]. Specifically, Nollet teaches “dynamically reconfiguring a computing platform in a process for executing **at least one application** on the platform.” EX1004, [0019] (emphasis added). Nollet discloses reconfiguring one or more processing units to run 3D gaming tasks of a 3D game. *Id.*, [0105], [0107] (“This frees up some hardware resources and allows the 3D game to be started as a hardware task.”).

113. Tuan discloses a number of different applications that are partitioned and modeled as multi-stage program instances. EX1005, 72-75. For example, Tuan discloses a Viterbi decoder, a JPEG encoder, an MPEG decoder, a Turbo

encoder, a G721 encoder, and a G721 decoder. *Id.*, 73-75. It would have been obvious to a POSITA to partition and model the 3D gaming tasks of Nollet as multi-stage program instances on dynamically reconfigurable processors, as taught in Tuan, at least because such an approach “enables efficient sharing of resources due to adequate task scheduling, allows dynamic changes of the application due to introduction of new tasks during run-time and greatly simplifies development since tasks can be developed quite independent from each other.” *Id.*, 16.

114. Partitioning 3D gaming into tasks has long been a common programming practice. For example, Lety describes “MiMaze, a distributed 3D multiplayer game on the Internet” and states, “[w]e have developed a multi-threaded Java applet to (Figure 1 right) (i) receive avatar positions and orientations from the UDP socket, (ii) receive “join and leave” information from the TCP socket, (iii) select the navigation type (the user is outside the maze, or it is in its own avatar), (iv) provide access to the VRML nodes through the EAI, and to (v) modify the 3D scene. We have also designed a specific scheduling algorithm to maximize the real-time quality of the application.” EX1010, Abstract, 3 (emphasis added).] A POSITA would have understood that multi-threaded Java applications consist of cooperating stages, which, in this example, are stages (i)-(v).

115. The Nollet-Tuan-Miller combination further discloses that the linking of the processing units is performed through a second set of inter-task

communication paths. EX1005, 66-67, 73-75, FIGS. 4.3 and 4.9. In particular, the 3D gaming tasks of the Nollet-Tuan-Miller combination includes a number of stages, where each of the stages of is connected to an adjacent stage via a FIFO, through which communication would be performed. *Id.*, 66-67, 73, 75, FIG. 4.9(d). Regarding the FIFOs, Tuan explains that “[a] first-in first-out (FIFO) memory mechanism, which employs VMEMs between Tiles, is used as an inter-task communication method,” and “[n]eighboring Tile groups can be communicated with each other through FIFOs formed with the distributed memory modules.” EX1005, 66, 72. A POSITA would have understood that the stages of the 3D gaming tasks would be connected via a second set of inter-task communication paths (FIFO memory mechanism) in view of Tuan’s Figure 4.3, which includes labels for each of the FIFOs.

116. For at least the reasons discussed above in Section VIII.A.1, a POSITA would have been motivated combine Nollet and Tuan.

[1.4] wherein each of the second set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of second tasks of the second program corresponding to a respective processing stage of the second multi-stage program instance;

117. It is my opinion that the Nollet-Tuan-Miller combination discloses that each of the second set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of second tasks of the second program corresponding to a respective processing stage of the second

multi-stage program instance.

118. As discussed above for element [1.3], the Nollet-Tuan-Miller combination discloses configuring a second set of reconfigurable logic-based processing units (e.g., FPGA tiles) to perform a second set of tasks (3D gaming tasks). EX1004, [0105], [0107]; EX1005, 66-67, 73-75, FIGS. 4.3 and 4.9(d). In that configuration, each of the reconfigurable units is programmed to perform a respective task of the second set of tasks (3D gaming tasks). *Id.* As further discussed in element [1.3], it would have been obvious to a POSITA that the 3D gaming tasks of the Nollet-Tuan-Miller combination is a multi-stage program instance, so the 3D gaming tasks would thus correspond to a respective processing stage. EX1004, [0105], [0107]; EX1005, 66-67, 73-75, FIGS. 4.3 and 4.9(d).

[1.5] maintaining, in a storage, a first location for a first reconfigurable logic-based processing unit of the first set of reconfigurable logic-based processing units executing the first multi-stage program instance such that a first one or more users and/or programs are enabled to communicate directly with the first multi-stage program instance;

119. It is my opinion that the Nollet-Tuan-Miller combination discloses maintaining, in a storage, a first location for a first reconfigurable logic-based processing unit of the first set of reconfigurable logic-based processing units executing the first multi-stage program instance such that a first one or more users and/or programs are enabled to communicate directly with the first multi-stage program instance. In fact, a POSITA would have recognized that maintaining the

location of a stage in memory is important for the efficient operation of any multi-stage program since, as described above, the stages communicate. Since stage locations are not fixed at a particular location (e.g., address) within the parallel processing hardware, before communicating with another stage, a source stage must determine the current location (e.g., address) of the destination stage. As an alternative, a stage could broadcast messages to every location within the parallel processing hardware, but that approach increases the number of messages that must be processed by each stage (i.e., a stage receives both messages intended for the stage and messages intended for other stages), which decreases efficiency, so both Nollet and Tuan reject that approach.

120. Nollet discloses keeping track of which task is being run where (i.e., which FPGA tile(s)). EX1004, [0029] (“means for deciding on which tile a hardware task is executed”), [0066]-[0068] (discussing the use of “hardware task routing tables”), [0086]. Nollet explains that the operating system “keeps track of the available resources, meaning location of all the tasks running,” for example, by “keep[ing] track of which tile of an FPGA is being used to execute a task.” *Id.*, [0086]. Specifically, Nollet discloses:

[0086] To create a true hardware/software multitasking environment, an operating system is provided by one aspect of the invention that manages the different applications running on the platform. The main function of the operating system is to manage tasks. An application running on the platform will contain a combination of tasks. Each of these tasks isq to run on one of the available resources. When a new

task is started or an old task is deleted, some of the existing tasks might have to be rescheduled onto another resource (e.g. to provide more computational power to the new task or to decrease the power consumption). In accordance with an aspect of the invention, the task creation and deletion consist in the usual task handling for a software task, and in reconfiguring the FPGA for a hardware task. **The operating system preferably keeps track of the available resources, meaning location of all the tasks running. For example, the OS can keep track of which tile of an FPGA is being used to execute a task.** Task rescheduling is also provided by one embodiment of the invention. State equivalence is identified between a software task and a hardware task. Indeed, when a task is rescheduled from hardware to software or vice-versa, the state information is transferred in order to resume the task where it has stopped.

Id., [0086].

121. Nollet discloses that by keeping track of which tasks are being run where, the operating system can facilitate direct communication to the hardware tasks. EX1005, [0087] (discussing “communication between two hardware tasks, communication between a hardware task and a software task and communication between two software tasks”), [0097] (explaining that an “interconnection network” allows “fast synchronous communication between hardware tasks implemented on the same matrix”), [0099] (“A fixed communication network between the tiles allows them to communicate together...”), [0159]; *see also id.*, [0104], [0113], [0114], [0132], FIG. 6. Nollet further discloses that if a user wishes to watch a movie, “[h]e presses the start button and begins enjoying the show.” *Id.*, [0107] (“When the user is starting his movie decoder the first time, the reconfigurable hardware is free and completely available. As a result, the movie

player can be started on it, as a hardware task.”). This permits the user and the program to directly communicate with the movie decoder. Thus, Nollet discloses, for example, maintaining a first location for a first FPGA tile executing the first multi-stage program instance, or the movie decoder, such that the operating system facilitates direct communication between a user/program with the first multi-stage program instance (movie decoder). EX1004, [0087], [0107], [0159].

122. Tuan similarly discloses keeping track of which task is being run where (i.e., which task is being run at which reconfigurable tile) to enable inter-task communications. EX1005, 15-16. For example, Tuan discloses that a “network manager” supports inter-task communication and input/output operations:

The **network manager** plays an important role in supporting inter-task communication and input/output operations. In this thesis, I encourage to use an interconnection network to connect cores and form a multicore reconfigurable architecture. Although the network itself is fixed, tasks are dynamically mapped onto any core, and once the mapping process has been done, **information about where tasks are located is updated to the network manager in order to allow related tasks to communicate to one another.**

Id. (emphasis added). As shown in Figure 2.5 (reproduced below), the network manager maintains a “task address allocator” and a “routing support.” EX1005, 28.

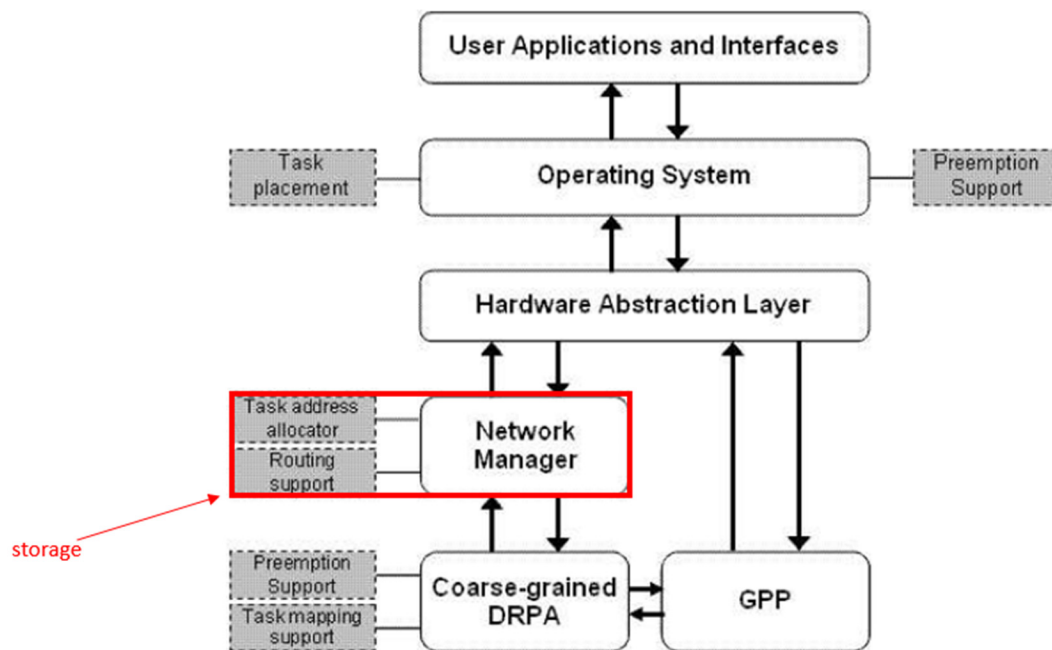


Fig. 2.5: System architecture

Id., FIG. 2.5 (annotated).

123. Tuan further teaches that its system includes memory, for example, for transferring data and inter-process communication. *Id.*, 27 (“several communication mechanisms can be exploited to transfer data between tasks such as memory...”), 28 (“The services provided include embedded memory reading and writing...”). A POSITA would have understood that the network manager of Tuan maintains the location information of where tasks are being executed in a storage, or memory.

124. To the extent that it is argued that “communicate directly” requires something or different than the teachings of Nollet and Tuan, Petitioner submits that the direct communications disclosed in Nollet and Tuan are at least as direct as

such communications disclosed in the '306 patent. With reference to Figure 4, the '306 patent states, "The input packets arriving over the input ports are demuxed by individual RX network port specific demultiplexers (demux:s) 405 to their indicated (via overhead bits) destination app-inst and input port specific FIFO buffers 410." EX1001, 13:18-22. That is, the "direct communication" of the '306 patent flow through at least a demux 405, FIFO 410, and a mux 450. *Id.*

[1.6] maintaining, in the storage, a second location for a second reconfigurable logic-based processing unit of the second set of reconfigurable logic-based processing units executing the second multi-stage program instance such that a second one or more users and/or programs are enabled to communicate directly with the second multi-stage program instance; and

125. For at least the same reasons as discussed in element [1.5], it is my opinion that the Nollet-Tuan-Miller combination discloses maintaining, in the storage, a second location for a second reconfigurable logic-based processing unit of the second set of reconfigurable logic-based processing units executing the second multi-stage program instance such that a second one or more users and/or programs are enabled to communicate directly with the second multi-stage program instance.

126. Both Nollet and Tuan disclose keeping track of which task is being run where (i.e., which FPGA tile(s)) and storing the location information in memory such that a second program (3D game program) can communicate directly with a second multi-stage program instance (3D game tasks). EX1004, [0067],

[0087], [0107], [0159]; EX1005, 15-16, 27-29, FIG. 2.5.

[1.7] in response to an increased demand for the second program, reallocating, by a controller comprising software and/or hardware logic configured to implement a load-adaptive allocation policy, at least one processing unit of the first set of reconfigurable logic-based processing units, the reallocating resulting in

127. It is my opinion that the Nollet-Tuan-Miller combination discloses, in response to an increased demand for the second program, reallocating, by a controller comprising software and/or hardware logic configured to implement a load-adaptive allocation policy, at least one processing unit of the first set of reconfigurable logic-based processing units.

128. As discussed above for elements [1.1]-[1.6], Nollet discloses dynamically reconfiguring a computing platform in a program for executing at least one application, including an example where a user is watching a movie on a portable device until he decides he wants to play a 3D game while a scene in the movie is playing. EX1004, [0107]. When the user first starts watching the movie, therefore increasing the demand for the movie player, the movie decoder starts and reconfigurable hardware is used for this task. *Id.* “When the user decides to start the 3D game, there are not enough resources available on the reconfigurable hardware to run both [the movie and 3D game] applications,” but “the movie can be downscaled in resolution and frame rate” to reduce computational requirements, “free[ing] up some hardware resources and allow[ing] the 3D game to be started as

a hardware task.” *Id.*

129. At a higher level, Nollet discloses reallocating at least one processing unit “based on parameters like task priority and processor affinity.” *Id.*, [0008] (“how to decide what task to run and where”), [0082]. Nollet further discloses task preemption, or reallocation of at least one processing unit, and explains that the applications concerns “multiple, possible independent, tasks running in parallel on the reconfigurable matrix.” *Id.*, [0082]. When preemption occurs, “the context of the current task can be saved, in order for this task to continue whenever additional computing resources are assigned to it.” *Id.* Nollet also discloses monitoring the state of the computing resource through a number of variables, which “mainly includes **the load of the computing unit.**” *Id.*, [0160] (emphasis added). In the example above, when the processing load in the FPGA is too high to accommodate an additional program, the movie tasks are reallocated to the CPU to make room for the 3D gaming tasks that have just been started, thus experiencing increased demand. *Id.*, [0107].

130. Nollet further discloses that reallocation of the reconfigurable processing units is performed by “an operating system that transparently manages the complexity of hierarchical reconfiguration.” *Id.*, [0158] (“the invention provides a filtering application on top of a microcontroller and a parameterizable filter block, both instantiated inside a reconfigurable hardware device such as an

FPGA”), [0159] (“the operating system will have to keep track of the available computing resources as well as the executing OS4RS tasks.”), [0086] (“The operating system preferably keeps track of the available resources, meaning location of all the tasks running.”). Nollet explains that for managing preemption, “a kind of memory controller/programmer is provided” that “takes care of saving/reloading context and configuration of the ... tiles.” *Id.*, [0085].

131. With respect to implementing a load-adaptive allocation policy, Miller teaches this feature. At a high level, Miller is directed to “providing load balancing functionality among multiple computing nodes” in a system that can be implemented with FPGAs. EX1006, Abstract, 30:60-31:2. More specifically, Miller provides for a Dynamic Computing Scaling (DCS) Load Balancer module that “dynamically scale[s] the computing nodes of a group when defined criteria are satisfied,” for example, by increasing the number of computing nodes in a group based on demand. *Id.*, 2:19-25, 3:23-25, 4:7-26. Miller also teaches a broad variety of such criteria that the DCS Load Balancer module may use, including measures of demand such as volume of requests received or handled by each computing node. *Id.*, 3:30-55. The DCS Load Balancer module monitors the load of a group of computing nodes and allocates more computing nodes when the load is high or when there is an increased demand (e.g., greater than a threshold). *Id.*, 22:19-67, 28:64-29:17. For example, Miller discloses that in response to an

increased demand for the functionality of the group computing nodes (e.g., a second program), another computing node can be “repurpose[ed]” to provide the necessary functionality (e.g., execute the program). *Id.*, 29:3-17. In the Nollet-Tuan-Miller combination, “[w]hen the user decides to start the 3D game” and “there are not enough resources available on the reconfigurable hardware to run both [the movie player and 3D game] applications,” the DCS Load Balancer module of would reallocate a reconfigurable processing unit in response to the increased demand from the 3D game application.

132. For at least the reasons discussed above in Section VIII.A.1, a POSITA would have found it obvious to implement task preemption (e.g., reallocation of FPGA tiles) in a load-adaptive manner as load-adaptive reallocation as taught by Miller, including reallocating tasks in response to increased demand by other processes, and POSITA would have been motivated combine Nollet and Miller.

[1.8] (1) switching the at least one of the first set of processing units from performing a task of the plurality of first tasks to performing one task of the plurality of second tasks, wherein switching comprises matching a first programming configuration of the at least one of the first set of processing units to a programming configuration demanded by the one task, and

133. The Nollet-Tuan-Miller combination discloses switching the at least one of the first set of processing units from performing a task of the plurality of first tasks to performing one task of the plurality of second tasks, wherein

switching comprises matching a first programming configuration of the at least one of the first set of processing units to a programming configuration demanded by the one task.

134. A POSITA would have understood that for reconfigurable hardware (e.g., an FPGA), switching a processing unit (e.g., an FPGA tile) from performing a first task to performing a second (different) task requires changing the programming configuration (a “bitstream”) loaded by the processing unit to a programming configuration that matches the needs of the second task. For example, Nollet explains, “The hardware tasks can be written in a description language and the **hardware tasks written in the description language can be automatically converted into a hardware bitstream description**” and “a software task is usually an object file, while the **hardware task is represented by a bitstream.**” EX1004, [0020], [0094] (emphasis added).

135. As discussed above for element [1.7], the Nollet-Tuan-Miller combination teaches switching a first set of FPGA tiles from running movie tasks (a first programming configuration) to running 3D gaming tasks (a (second) programming configuration demanded by the one task). Nollet further discloses “reconfiguring one of the tiles from a first configuration, capable of running a first task, into a second different configuration, capable of running a second different task.” EX1004, [0036]. Specifically, Nollet discloses:

Another aspect of the invention also provides a method for reconfiguring a reconfigurable hardware device having tiles such that a set of hardware tasks the device is capable to run is changed, the method comprising **reconfiguring one of the tiles from a first configuration, capable of running a first task, into a second different configuration, capable of running a second different task**; the reconfiguring of the one tile being such that the interface of the second configuration satisfies the interface position constraints.

Id., [0036] (emphasis added).

136. Furthermore, as discussed above for element [1.7], the Nollet-Tuan-Miller combination discloses reallocating one more reconfigurable processing units in response to an increased demand for a second program (e.g., 3D game) to implement a load-adaptive policy. Nollet explains that when preemption occurs, a reconfigurable processing unit ceases to perform one task and is reconfigured to perform a second task. EX1004, [0026], [0082], [0107] (“The movie player can therefore be rescheduled into one of the ISPs, as a software task. This frees up some hardware resources and allows the 3D game to be started as a hardware task.”). Thus, A POSITA would have understood that the reallocation of the Nollet-Tuan-Miller combination involves switching a tile from performing one task of the plurality of first tasks (e.g., a movie decoder task) to performing one task of the plurality of second tasks (e.g., a 3D game task).

[1.9] (2) adjusting, in storage, at least one of the first location or the second location to enable, through the respective location, direct communication to the

other multi-stage program instance of the first multi-stage program instance or the second multi-stage program instance;

137. The Nollet-Tuan-Miller combination discloses adjusting, in storage, at least one of the first location or the second location to enable, through the respective location, direct communication to the other multi-stage program instance of the first multi-stage program instance or the second multi-stage program instance.

138. As described above, maintaining an accurate location for a stage is necessary for enabling efficient communication with the stage. Since processing units are being reallocated from one task (stage) to a different task, a POSITA would have recognized that adjusting the stored locations (e.g., addresses) for these stages is similarly necessary, as described further below with respect to Nollet and Tuan.

139. As discussed above with respect to element [1.6], Nollet and Tuan disclose keeping track of which task is being run where (i.e., which task is being run at which reconfigurable processing unit). EX1004, [0086]; EX1005, 15-16. Nollet further discloses that when the FPGA tiles are reconfigured to handle different tasks, the routing tables are updated to keep track of which task is being run where. EX1004, [0087], [0114] (“Routing tables transform a destination logical address into the number of X and Y hops on the network. The routing tables are updated by the RT-OS to match the position of the IP-block in the

network. Therefore, a task does not need to know where it is situated in the network and whether the tasks it communicates with are running in hardware or software.”).

140. Similarly, Tuan discloses that tasks are dynamically mapped onto cores in an interconnection network, and “once the mapping process has been done, *information about where tasks are located is updated to the network manager* in order to allow related tasks to communicate to one another.” EX1005, 15-16 (emphasis added).

[1.10] wherein the load-adaptive allocation policy is configured to facilitate minimizing reconfiguring the plurality of reconfigurable logic-based processing units.

141. The Nollet-Tuan-Miller combination discloses that the load-adaptive allocation policy is configured to facilitate minimizing reconfiguring the plurality of reconfigurable logic-based processing units.

142. Reconfiguring reconfigurable logic-based processing unit necessarily imposes a delay as the system performs the reconfiguration, and delays reduce computing efficiency. As of the effective date of the patent, reconfiguring an FPGA could take from a few milliseconds to hundreds of milliseconds. EX1011, Table I. To improve system efficiency, a POSITA would have recognized the importance of minimizing such reconfiguration, as recognized by Nollet, Tuan, and Redaelli.

143. Nollet discloses allocating reconfigurable processing units (e.g., FPGA tiles) to corresponding tasks based on “processor affinity.” EX1004, [0008], [0082]. Though not explicitly defined in Nollet, a POSITA would have understood that processor affinity refers to the preference that a task or application has for being executed on a particular hardware device or processor. *See, e.g.*, EX1008, 2:49-66 (“[E]ach process in the queue has a mask indicating its preferred/kin processor. At allocation time, each process is allocated to its kin processor in preference to others.”); EX1009, 5:57-6:2 (“A code thread instance having a ‘hard’ processor affinity is executed only by a specified [processor]. A code thread instance having a ‘soft’ processor affinity should be executed by a specified one of the processors unless the specified processor is heavily loaded and another processor is less heavily loaded and is available to process the code thread instance.”). Because Nollet discloses the use of processor affinity in its allocation process, a POSITA would have understood Nollet as minimizing reconfiguration of a processing unit assigned to a task that has a preference for that processing unit when the processing unit is already executing the task and has not yet completed it. Additionally, a POSITA would have found it obvious to implement the load adaptive policy in the Nollet-Tuan-Miller combination in a manner that minimized the reconfiguring of the plurality of reconfigurable logic-based processing units. Notably, a POSITA would have been familiar with this known approach to

increase the probability of reusing a configuration. *See, e.g.*, EX1007, 65, 69-70 (“tries to parallelize the execution as much as possible and evaluate the performance for a given system configuration”; “Modules are left intact on the FPGA until other tasks require their space, in order to increase the probability of reuse”; “First of all it looks for module reuse opportunities and places all the tasks that can be reused.”).

144. Alternatively, with respect to Ground 1B, Redaelli teaches minimizing reconfiguring the plurality of reconfigurable logic-based processing units. Redaelli is directed to “Reconfiguration Aware Task Scheduling for Multi-FPGA Systems.” EX1007, 65. In particular, Redaelli teaches a scheduling approach that “take[s] advantage of all the features of the multi-FPGA architecture and ... tries to parallelize the execution as much as possible and evaluate the performance for a given system configuration.” *Id.*, 69-70. In Redaelli’s approach, “[m]odules are left intact on the FPGA until other tasks require their space, in order to increase the probability of reuse” by “look[ing] for module reuse opportunities and plac[ing] all the tasks that can be reused.” *Id.*, 70. It would have been obvious to a POSITA to combine the teachings of Nollet with the scheduling approach of Redaelli to minimize reconfiguration and maximize reuse of configurations for at least the reasons discussed above in Section VIII.A.2.

Claim 2

[2.0] The method of claim 1, wherein at least one of the first program and the second program is a data processing service.

145. It is my opinion that the Nollet-Tuan-Miller combination discloses that at least one of the first program and the second program is a data processing service.

146. A POSITA would have understood that the term “data processing service” encompassed a broad range of computing activities, including receiving, storing, processing and producing data. Nollet discloses running a movie decoder as a first program and a 3D game as a second program. EX1004, [0107]. Both the movie decoder program and the 3D game are data processing services. That is, a movie decoder is used to decode or decompress encoded video, including, for example, adjusting resolution and frame rate, which a POSITA would have understood is a data processing service. EX1004, [0107]. Similarly, a 3D game involves receiving inputs from a user/player and producing, rendering, and displaying audio/visual outputs, which a POSITA also would have understood as a data processing service. EX1004, [0107].

147. Moreover, Tuan teaches explicit tasks performed by a movie decoder (MPEG decoder), such as a variable length decode task, an inverse quantization task, and inverse discrete cosine transfer task, and a motion compensation task. EX1005, 75. A POSITA would have understood any one of these tasks as being a data processing task, so the MPEG decoder is a data processing service.

Claim 3

[3.0] The method of claim 2, wherein the data processing service is available for use by an external user or program of at least one of a) the first one or more users and/or programs or b) the second one or more users and/or programs communicating with the respective program via a network connection with the multi-user parallel processing hardware architecture.

148. It is my opinion that the Nollet-Tuan-Miller combination discloses that the data processing service is available for use by an external user or program of at least one of a) the first one or more users and/or programs or b) the second one or more users and/or programs communicating with the respective program via a network connection with the multi-user parallel processing hardware architecture. I note that element [1.5] states that “a first one or more users and/or programs are enabled to communicate directly with the first multi-stage program instance,” and element [2.0] states that the first (or second) multi-stage program instance is a data processing service. A POSITA would have understood that if users and/or programs are enabled to communicate directly with a data processing service, the data processing service is available for use by the users and/or programs.

149. In one example, Miller discloses that multiple computing nodes are “part of one or more physical computer networks in one or more geographical locations and under control of the user or other entity,” so they are connected to form a network that can be shared by remote users through the internet. EX1006,

2:6-14, FIGS. 1A-1B. For example, multiple computing nodes are connected to form a provided computer network 120a, which is accessible by client 1. *Id.*, FIG. 1B and 13:1-21.

150. In the Nollet-Tuan-Miller combination, the multitasking platform of Nollet would provide various data processing services (e.g., movie decoder, 3D game) that are available for use by an external user or program in the computer network of which Nollet's platform would be a part. A POSITA would have been motivated to combine Nollet with Miller for at least the reasons discussed in Section VIII.A.1.

Claim 4

[4.0] The method of claim 1, wherein the at least one processing unit comprises at least one Field Programmable Gate Array (FPGA).

151. It is my opinion that the Nollet-Tuan-Miller combination discloses that the at least one processing unit comprises at least one Field Programmable Gate Array (FPGA).

152. It would have been obvious to a POSITA to modify Nollet in view of Tuan to include an array reconfigurable devices, as in Tuan, such that each processing unit is an FPGA, rather than a FPGA tile. EX1004, [0012], [0013] (“a reconfigurable matrix (for instance a gate array, especially an FPGA), adapted for (dynamic) hardware/software multitasking”); EX1005, FIG. 2.4. Further, Tuan explicitly teaches that computing cores can be either entire reconfigurable units, or

partitions of a reconfigurable unit: “In a homogeneous system, cores are identical in architecture. For example, a multi-FPGA system containing multiple FPGA chips can be seen as a multicore system, where each core corresponds to an FPGA. In a reconfigurable system where the reconfigurable array can be divided into hardware parts, each of which can be assigned to a task for execution, and all parts can be reconfigured and executed independently, cores can be considered as equivalent to such parts.” EX1005, 25-26.

153. Thus, a POSITA would have understood that the system of Nollet, which includes as cores at least one FPGA that is divided into hardware parts, could be modified by Tuan to use multiple FPGAs as cores. Such a modification would have resulted in a more powerful device with more computing resources and improve throughput. EX1005, 25-26, 29. Tuan and Miller also disclose the use of FPGAs. EX1005, 36, 64, 100; EX1006, 30:67-31:1.

Claim 5

[5.0] The method of claim 1, wherein the plurality of reconfigurable logic-based processing units is included in an array of processing units.

154. It is my opinion that the Nollet-Tuan-Miller combination discloses that the plurality of reconfigurable logic-based processing units is included in an array of processing units.

155. Figure 6 of Nollet (reproduced below) shows that the plurality of reconfigurable logic-based processing units (e.g., FPGA tiles) is included in an

array of processing units (e.g., FPGA). EX1004, [0113]-[0114]; *see also id.*, [0102] (“In order to do multitasking, the FPGA is partitioned into an array of identical tiles, each tile running a hardware task...”).

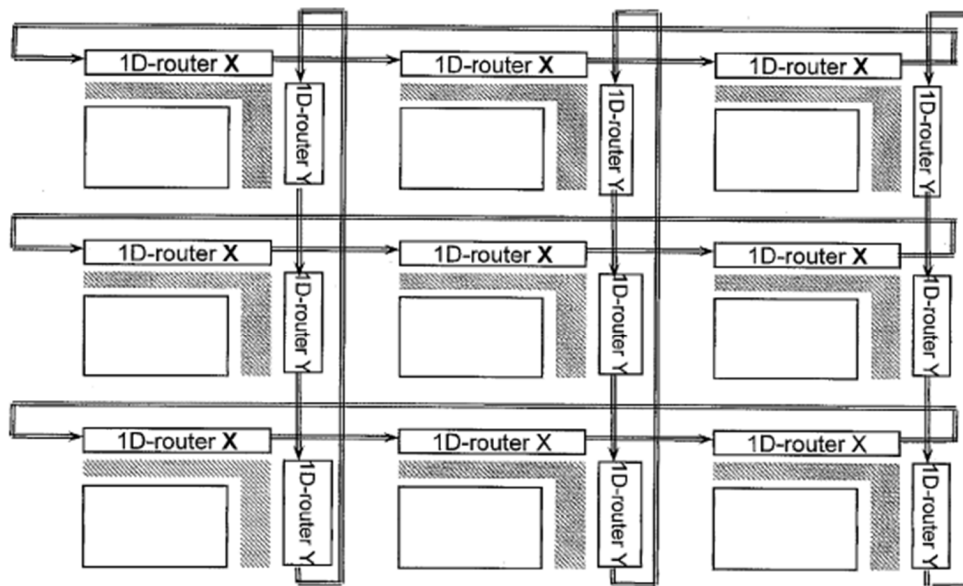


Fig. 6

Id., FIG. 6.

156. Specifically, Nollet discloses:

[0113] On a 2D torus, rows are equivalent to columns, therefore the torus can be decomposed into rows of 1D toruses connected to columns of 1D toruses as shown in FIG. 6. Two simple 1D routers 28 can be chained to obtain a 2D router and produce a more modular network. A 1D-router has two input/output channels as shown in FIG. 8. A message entering a router 28 can either be forwarded along the current direction or sent orthogonally, either on a column if the Y header-flit is non-zero, or into a hardware task 29. Each channel is composed of a 16-bit data path and of 3-bit control signals. Two control-bits are used to signal the presence of a message and its destination and the third is a back-pressure signal (NACK) used to block a message entering a busy router or interface. The routers 28 handle two time-multiplexed VCs to avoid deadlocking. These VCs are interleaved, with one clock cycle each. For

efficiency, the router is fully pipelined. Because the data channels are interleaved the control signals are also interleaved and precede data by one clock cycle. The NACK signal, used for blocking, is back-propagated along the message path. It takes two ticks for a flit to get out of a router, therefore at each clock cycle data is transmitted on, alternating VCs.

[0114] A 1D-router is composed of one input controller 31 per channel, one arbiter 35 and an output-controller 36 as shown in FIG. 9. The input controllers 31a, 31b issue output requests to a round-robin arbiter 35 and a unit 32a, 32b decrements header-flits when routing. Each output channel has a unit 33a, 33b with a 2-flit deep buffer 34a, 34b to be able to resume blocked messages. The output controller 36 is composed of a 2-input crossbar-switch and the NACK. An interface decouples the hardware task 29 from the network using DP-RAMs as message buffers. The hardware task 29 (FIG. 6) can then use independent data-width and clock rates, allowing easy IP integration. The interfaces are designed to cope with the sustained bit-rates required by the ICN and perform multiple-message buffering to reduce network congestion. Moreover the interface, called a “net-cell”, provides the hardware task (or IP-block) with high-level communication by means of routing tables. Hardware tasks 29 within an application communicate through a tuple composed of a logical address and a port number, similar to the IP address and UDP port number for the UDP protocol. Routing tables transform a destination logical address into the number of X and Y hops on the network. The routing tables are updated by the RT-OS to match the position of the IP-block in the network. Therefore, a task does not need to know where it is situated in the network and whether the tasks it communicates with are running in hardware or software. The RT-OS adapts the routing tables after an IP-block reconfiguration. They can also be modified at any time without having to reset or stop the hardware task. This is very useful if some other task from the same application is switched in or out of the network as a function of the available resources and the QoS policy. The net-cell buffers two input and two output messages on a Virtex I and eight on the Virtex II. Each message buffer is complemented by a control register bank giving the length and origin/destination of the message. This high-level protocol information such as port number or net-cell origin is piggy-backed in the Y-header flit and does not require extra bandwidth.

Id., [0113]-[0114].

Claim 6

[6.0] The method of claim 5, wherein the load-adaptive allocation policy is configured to facilitate load balancing across the array of processing units.

157. It is my opinion that the Nollet-Tuan-Miller combination discloses that the load-adaptive allocation policy is configured to facilitate load balancing across the array of processing units.

158. As discussed above for element [5.0], Nollet teaches that the plurality of reconfigurable processing units (e.g., FPGA tiles) is included in an array. EX1004, [0113]-[0114], FIG. 6. Nollet discloses allocating processing units to tasks in an efficient manner. *Id.*, [0006] (“provide an environment where several tasks can run concurrently, with minimal of interference between them”), [0008], [0159] (“The OS4RS’s main duty is to divide the available computing resources among all executing tasks **in an efficient and fair way.**”). Nollet also discloses that the “operating system is also able **to monitor the state of the computing resource** through a number of variables,” which “mainly includes the **load of the computing unit.**” *Id.*, [0160] (emphasis added). Nollet discloses that when the processing load in the FPGA is too high to accommodate a second program (3D game), the tasks from the first program (movie player) are downscaled in resolution and reallocated to make room for the 3D gaming tasks. *Id.*, [0107].

159. As discussed above for element [1.7], Miller discloses a load-adaptive allocation policy that “provid[es] load balancing functionality among multiple

computing nodes” in a system that can be implemented with FPGAs. EX1006, Abstract, 30:60-31:2. It would have been obvious to a POSITA thus to implement the multitasking platform of Nollet with Miller’s load-adaptive allocation policy, which would facilitate load balancing across the array of reconfigurable units of Nollet. A POSITA would have understood that load balancing results in the efficient use of computing resources, such as computing cores. Since the systems of both Nollet and Tuan include arrays of processing units as computing resources, a POSITA would have understood that load balancing, as taught by Miller, across the arrays of processing units of both Nollet and Tuan would result in the efficient use of those resources. A POSITA would have been motivated to combine Nollet, Tuan, and Miller for at least the reasons discussed in Section VIII.A.1.

Claim 7

[7.0] The method of claim 1, wherein the controller is configured to monitor availability of the plurality of reconfigurable logic-based processing units.

160. It is my opinion that the Nollet-Tuan-Miller combination discloses that the controller is configured to monitor availability of the plurality of reconfigurable logic-based processing units.

161. Nollet discloses that the controller is configured to monitor availability of the plurality of reconfigurable logic-based processing units.

EX1004, [0086] (“The operating system preferably keeps track of the available resources, meaning location of all the tasks running.”), [0159] (“the operating

system will have to keep track of the available computing resources as well as the executing OS4RS tasks.”).

162. Tuan also discloses that “once the mapping process has been done, information about where tasks are located is updated to the network manager in order to allow related tasks to communicate to one another.” EX1005, 28-29. Tuan further teaches allocating unused tiles based on the execution time of tile groups. *Id.*, 70. Thus, it would have been obvious to a POSITA that Tuan’s controller is configured to monitor the availability of tiles, including keeping track of “unused tiles.”

163. Miller further teaches that its DCS Load Balancer module takes actions based on information about the status of computing nodes and demand for computing nodes. EX1006, 28:64-29:24.

Claim 8

[8.0] The method of claim 1, further comprising monitoring a first characteristic corresponding to at least one of the first program and the first set of reconfigurable logic-based processing units, wherein the increased demand is determined at least in part by the monitoring of the first characteristic.

164. It is my opinion that the Nollet-Tuan-Miller combination discloses monitoring a first characteristic corresponding to at least one of the first program and the first set of reconfigurable logic-based processing units, wherein the increased demand is determined at least in part by the monitoring of the first characteristic.

165. The controller of Nollet monitors the availability of the plurality of reconfigurable logic-based processing units, including the load of the processing units. EX1004, [0086], [0159]-[0160]. Nollet states that the “operating system is also able to monitor the state of the computing resource through a number of variables,” which “mainly includes the load of the computing unit.” *Id.*, [0160].

166. For example, when a user first presses the start button to watch a movie, Nollet explains that “the reconfigurable hardware is free and completely available.” *Id.*, [0107]. However, “[w]hen the user decides to start the 3D game, there are not enough resources available on the reconfigurable hardware to run both applications,” so the movie is “downscaled in resolution and frame rate, resulting in reduced computational requirements for the player,” freeing up hardware resources and allowing the 3D game to be started as a hardware task. *Id.* Thus, Nollet teaches using the monitoring of the load of the hardware to identify an increased demand. *Id.*

167. Moreover, a POSITA would have understood that a load balancer naturally must determine the load of a system before that load can be balanced. Further, the load of a system can be measured based on various characteristics of the system. A POSITA would have recognized that many characteristics can be used to determine the load, especially in light of Miller, which discloses at least ten such characteristics in a “non-exclusive list.” EX1006 3:30-55.

Claim 9

[9.0] The method of claim 8, wherein the first characteristic comprises a data processing load.

168. It is my opinion that the Nollet-Tuan-Miller combination discloses that the first characteristic comprises a data processing load.

169. For at least the reasons discussed in element [8.0], Nollet and Miller both disclose monitoring availability of processing units, including the load of the processing units. EX1004, [0086], [0107], [0159]-[0160]; EX1006, 3:23-61.

Claim 10

[10.0] The method of claim 8, wherein the first characteristic comprises a readiness to execute of one or more tasks of the plurality of first tasks.

170. It is my opinion that the Nollet-Tuan-Miller combination discloses that the first characteristic comprises a readiness to execute of one or more tasks of the plurality of first tasks.

171. In the movie player/3D game example, Nollet explains that “[w]hen the user is starting his movie decoder the first time, the reconfigurable hardware is free and completely available,” so the movie decoding task is “ready” to execute. EX1004, [0107]. After the user decides to start the 3D game, the movie is “downscaled in resolution and frame rate, resulting in reduced computational requirements for the [movie] player,” which “frees up some hardware resources.” The “freed up resources” are thus now ready to execute tasks. Thus, Nollet teaches

monitoring the readiness of one or more tasks.

Claim 11

[11.0] The method of claim 1, wherein matching the first programming configuration of the at least one of the first set of processing units to the programming configuration demanded by the one task comprises, when the first programming configuration is different than the programming configuration demanded by the one task, reconfiguring the first programming configuration to the programming configuration demanded by the one task.

172. It is my opinion that the Nollet-Tuan-Miller combination discloses that matching the first programming configuration of the at least one of the first set of processing units to the programming configuration demanded by the one task comprises, when the first programming configuration is different than the programming configuration demanded by the one task, reconfiguring the first programming configuration to the programming configuration demanded by the one task.

173. As discussed above for elements [1.7]-[1.8], the Nollet-Tuan-Miller combination teaches switching a first set of FPGA tiles from running movie tasks (a first programming configuration) to running 3D gaming tasks (a (second) programming configuration demanded by the one task) and “reconfiguring one of the tiles from a first configuration, capable of running a first task, into a second different configuration, capable of running a second different task.” EX1004, [0107], [0036].

174. A POSITA would have recognized that to match a set of processing

units to a demanded configuration requires reconfiguring the processing units, except in the specific case where the processing units are already configured according to the demand. In fact, Tuan states, “The hardware circuit of reconfigurable architectures is not static but adapted to the target applications at the run time,” but, “in reconfigurable computing, a hardware task consists of a two-dimensional logic circuit that is loaded in a reconfigurable array for execution.” EX1005, 2, 11-12. A POSITA would have recognized that loading a two-dimensional logic circuit in a reconfigurable array is “reconfiguring the first programming configuration.”

Claim 12

[12.0] A system for task-switching on a multi-user parallel processing hardware architecture comprising a plurality of reconfigurable logic-based processing units, the system comprising:

175. For at least the same reasons as discussed above for element [1.0], the Nollet-Tuan-Miller combination discloses this element.

[12.1] the plurality of reconfigurable logic-based processing units;

176. For at least the same reasons as discussed above for element [1.0], the Nollet-Tuan-Miller combination discloses this element.

[12.2] a reconfigurable cross-connect for establishing communication links between pairs of processing units of the plurality of reconfigurable logic-based processing units; and

177. It is my opinion that the Nollet-Tuan-Miller combination discloses a reconfigurable cross-connect for establishing communication links between pairs

of processing units of the plurality of reconfigurable logic-based processing units.

178. With respect to this element, the '306 patent states that its system includes “a hardware implemented data packet switching cross-connect (XC) connecting data packets from an output port of a processing stage to an input port of a given processing stage if a destination software program task of the data packet is hosted at the given processing stage.” EX1001, 3:43-59.

179. Nollet discloses such a reconfigurable cross-connect for establishing communication links between pairs of processing units of the plurality of reconfigurable logic-based processing units. EX1005, [0110]-[0112]. Specifically, Nollet discloses a “reconfigurable hardware device, compris[ing] typically a plurality of small configurable logic blocks and **an interconnect structure for interconnecting the configurable logic blocks,**” and states that “[r]econfiguring the hardware device means changing the configuration or function of the logic blocks and/or the **interconnection between the configurable logic blocks.**” *Id.*, [0057] (emphasis added). Nollet also describes a specific implementation of its reconfigurable multitasking computing platform that includes FPGAs as the reconfigurable hardware device “having a packet-switched **interconnection network**” provided by “a 2D torus using wormhole routing,” for example. *Id.*, [0110] (emphasis added); *see also id.*, [0034] (“The communication network can be adapted for supporting packet-switched interconnection network,

in particular for supporting message passing communication between tasks, executed by the tiles. ... The communication network may comprise a plurality of routers. Interfaces, for decoupling the tasks from the network may be provided.”), [0035] (“A routing structure for interconnecting the configurable logic blocks (CLB 's) may be provided. A communication network defined on the routing structure for providing communication between the tiles can also be provided. The communication network can comprise a plurality of routers.”). Specifically, Nollet discloses:

[0110]As an example of an implementation of the invention, a board is provided containing a Xilinx Virtex-II™ XC2V6000 FPGA, a Compaq iPaq™ handheld device and some glue logic. **The FPGA serves as the reconfigurable hardware device, having a packet-switched interconnection network, and runs hardware tasks.** The iPaq contains the rest of the platform. There is only one ISP, which is a StrongARM SA-1110 processor, for example. The iPaq and the FPGA board are connected together via the Expansion Bus of the iPaq. **The communication between the processor and the network on the FPGA is performed by buffering messages in a Random Access Memory, e.g. a DPRAM.** The glue logic on the FPGA board allows the processor to perform a partial reconfiguration of the FPGA. The operating system is based on Linux™, for example. A port of Linux is indeed available for the iPaq. Real-time services to the Linux OS (RT-Linux), running on the processor (206 Mhz) is added. Extensions enabling the management of hardware tasks are added. **The interconnection network is a 2D torus using wormhole routing, for example.** This implementation achieves transfer rates up to 77.6 MB/s between two adjacent routers, when clocked at 40 MHz. **Interconnection networks on FPGAs allow fine-grain dynamic partial reconfiguration and make hardware multi-tasking a reality.**

Nollet, [0110] (emphasis added).

180. Figure 6 of Nollet further illustrates such a reconfigurable cross-connect. Nollet explains that “[o]n a 2D torus, rows are equivalent to columns, therefore the torus can be decomposed into rows of 1D toruses connected to columns of 1D toruses as shown in FIG. 6.” *Id.*, [0113]. “Two simple 1D routers 28 can be chained to obtain a 2D router and produce a more modular network.” *Id.*

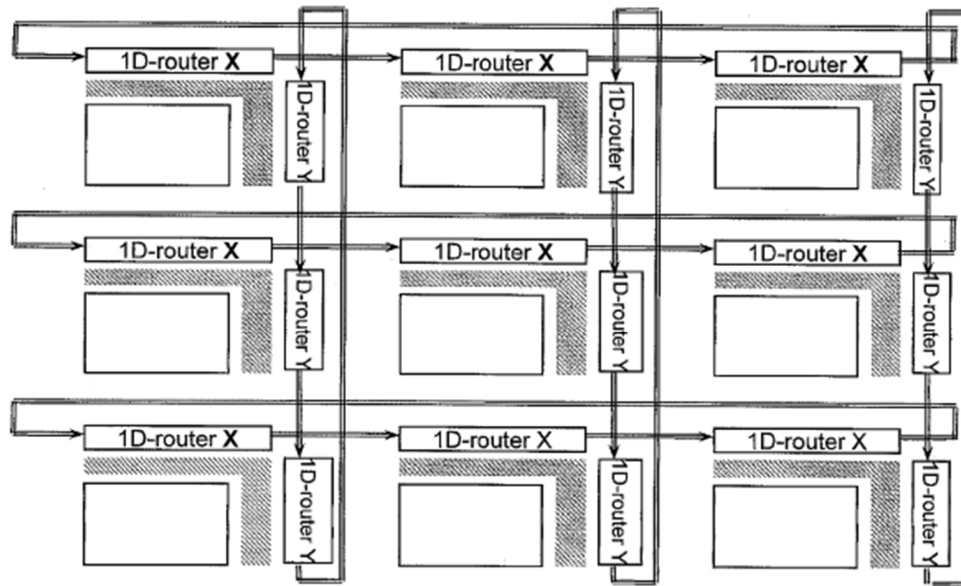


Fig. 6

Id., FIG. 6.

[12.3] hardware logic and/or software logic configured to

181. It is my opinion that the Nollet-Tuan-Miller combination discloses hardware logic and/or software logic.

182. Nollet discloses a parallel processing hardware/software architecture comprising a hardware device with a matrix of reconfigurable processing units (e.g., FPGA tiles). EX1004, [0017], [0082], Abstract (“Dynamic

hardware/software multitasking methods for a reconfigurable computing platform including reconfigurable hardware devices such as gate arrays, especially FPGA's, and software, such as dedicated hardware/software operating systems and middleware, adapted for supporting the methods, especially multitasking, are described"). "Preferably, the reconfigurable device, comprises a plurality of tiles, each tile being configurable in a variety of configurations, each of the tile configurations being capable of running a hardware task." *Id.*, [0020]. Each of the reconfigurable units is a piece of hardware logic. *See id.*, [0123] (explaining that an example reconfigurable system (OS4RS) is "responsible for scheduling tasks, both on the ISP and on the reconfigurable logic"), [0136], [0140], [0143].

[12.4] link, through the reconfigurable cross-connect, a first set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a first multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a first program,

183. For at least the same reasons as discussed above for element [1.1], the Nollet-Tuan-Miller combination discloses this element.

[12.5] wherein each of the first set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of first tasks of the first program corresponding to a respective processing stage of the first multi-stage program instance,

184. For at least the same reasons as discussed above for element [1.2], the Nollet-Tuan-Miller combination discloses this element.

[12.6] link, through the reconfigurable cross-connect, a second set of reconfigurable logic-based processing units of the plurality of reconfigurable logic-based processing units into a second multi-stage program instance configured to execute a corresponding set of interdependent processing stages of a second program,

185. For at least the same reasons as discussed above for element [1.3], the Nollet-Tuan-Miller combination discloses this element.

[12.7] wherein each of the second set of reconfigurable logic-based processing units is programmed to perform a respective task of a plurality of second tasks of the second program corresponding to a respective processing stage of the second multi-stage program instance,

186. For at least the same reasons as discussed above for element [1.4], the Nollet-Tuan-Miller combination discloses this element.

[12.8] maintain, in a storage, a first location for a first reconfigurable logic-based processing unit of the first set of reconfigurable logic-based processing units executing the first multi-stage program instance such that a first one or more users and/or programs are enabled to communicate directly with the first multi-stage program instance,

187. For at least the same reasons as discussed above for element [1.5], the Nollet-Tuan-Miller combination discloses this element.

[12.9] maintain, in the storage, a second location for a second reconfigurable logic-based processing unit of the second set of reconfigurable logic-based processing units executing the second multi-stage program instance such that a second one or more users and/or programs are enabled to communicate directly with the second multi-stage program instance, and

188. For at least the same reasons as discussed above for element [1.6], the Nollet-Tuan-Miller combination discloses this element.

[12.10] in response to an increased demand for the second program, reallocate, according to a load-adaptive allocation policy, at least one processing unit of the

first set of reconfigurable logic-based processing units, the reallocating resulting in

189. For at least the same reasons as discussed above for element [1.7], the Nollet-Tuan-Miller combination discloses this element.

[12.11] (1) switching the at least one of the first set of processing units from performing a task of the plurality of first tasks to performing one task of the plurality of second tasks, wherein switching comprises matching a first programming configuration of the at least one of the first set of processing units to a programming configuration demanded by the one task, and

190. For at least the same reasons as discussed above for element [1.8], the Nollet-Tuan-Miller combination discloses this element.

[12.12] (2) adjusting, in storage, at least one of the first location or the second location to enable, through the respective location, direct communication to the other multi-stage program instance of the first multi-stage program instance or the second multi-stage program instance;

191. For at least the same reasons as discussed above for element [1.9], the Nollet-Tuan-Miller combination discloses this element.

[12.13] wherein the load-adaptive allocation policy is configured to facilitate minimizing reconfiguring the plurality of reconfigurable logic-based processing units.

192. For at least the same reasons as discussed above for element [1.10], the Nollet-Tuan-Miller combination and/or the Nollet-Tuan-Miller-Redaelli combination discloses this element.

Claim 13

[13.0] The system of claim 12, wherein the plurality of reconfigurable logic-based processing units is included in an array of processing units.

193. For at least the same reasons as discussed above for element [5.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 14

[14.0] The system of claim 13, wherein the array of processing units comprises a plurality of central processing units (CPUs) and/or graphics processing units (GPUs).

194. It is my opinion that the Nollet-Tuan-Miller combination discloses wherein the array of processing units comprises a plurality of central processing units (CPUs) and/or graphics processing units (GPUs).

195. Nollet discloses a multi-processor computing platform containing “one or more instruction set processors (ISP)” and a reconfigurable matrix (FPGA), adapted for dynamic hardware/software multitasking. EX1004, Abstract. Along with the FPGA, Nollet discloses a CPU enabled to communicate with the FPGA. *Id.*, [0112]; *see also id.*, [0152]. With respect to CPUs and ISPs, Nollet explains that “software bytecode 68 is interpreted by the software virtual machine (SVM) 69, and turned into native application software code 70 that runs on the native CPU of a programmable processor 71, e.g. an ISP as shown in FIG. 17.” *Id.*, [0079]. Thus, it would have been obvious to a POSITA that, because Nollet can include one or more ISPs and that the native CPU of a programmable processor can be an ISP, the array of Nollet comprises a plurality of CPUs.

Claim 15

[15.0] The system of claim 12, wherein at least one of the first program and the second program executes a data processing service.

196. For at least the same reasons as discussed above for element [2.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 16

[16.0] The system of claim 12, further comprising a network connection for providing communication between an external user or program of at least one of a) the first one or more users and/or programs or b) the second one or more users and/or programs and the data processing service.

197. For at least the same reasons as discussed above for element [3.0], the Nollet-Tuan-Miller combination discloses this element.

198. Additionally, Miller discloses a network connection for providing communication between an external user or program. EX1006, 4:50-5:31 (“After configuring such a computer network, the user and/or a DCS Load Balancer module may interact from one or more remote locations with the computer network being provided by the network-accessible service, such as to execute programs on the computing nodes of the provided computing network...”), 28:7:12 (“the computing system 300 has components that include a CPU 305, various I/O components 310 The illustrated I/O components include a display 311, a network connection 312...”).

Claim 17

[17.0] The system of claim 16, wherein providing communication comprises providing communication to the external user or program without awareness of the first location or the second location by the external user or program.

199. It is my opinion that the Nollet-Tuan-Miller combination discloses that providing communication comprises providing communication to the external user or program without awareness of the first location or the second location by the external user or program.

200. In particular, Miller discloses that “one client may configure the provided computer network for that client to ***provide access to one or more network-accessible remote services*** that are provided by other of the clients using other of the provided computer networks at that geographical location, such as ***without the one client being aware that those remote resources are physically provided*** by other computing nodes at the same geographical location.” EX1004, 27:8-15. It would have been obvious to a POSITA that if the one client was not aware that remote resources were provided, that client would further not be aware of the first location or the second location in the system storage.

Claim 18

[18.0] The system of claim 12, wherein the at least one processing unit comprises at least one Field Programmable Gate Array (FPGA).

201. For at least the same reasons as discussed above for element [4.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 19

[19.0] The system of claim 12, wherein the plurality of reconfigurable logic-based processing units is included in an array of processing units.

202. For at least the same reasons as discussed above for element [5.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 20

[20.0] The system of claim 19, wherein the load-adaptive allocation policy is configured to facilitate load balancing across the array of processing units.

203. For at least the same reasons as discussed above for element [6.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 21

[21.0] The system of claim 12, further comprising monitoring a first characteristic corresponding to at least one of the first program and the first set of reconfigurable logic-based processing units, wherein the increased demand is determined at least in part by the monitoring of the first characteristic.

204. For at least the same reasons as discussed above for element [8.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 22

[22.0] The system of claim 21, wherein the first characteristic comprises a data processing load.

205. For at least the same reasons as discussed above for element [9.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 23

[23.0] The system of claim 21, wherein the first characteristic comprises a readiness to execute of one or more tasks of the plurality of first tasks.

206. For at least the same reasons as discussed above for element [10.0], the Nollet-Tuan-Miller combination discloses this element.

Claim 24

[24.0] The system of claim 12, wherein matching the first programming configuration of the at least one of the first set of processing units to the programming configuration demanded by the one task comprises, when the first programming configuration is different than the programming configuration demanded by the one task, reconfiguring the first programming configuration to the programming configuration demanded by the one task.

207. For at least the same reasons as discussed above for element [11.0], the Nollet-Tuan-Miller combination discloses this element.

IX. CONCLUSION

208. The findings and opinions set forth in this declaration are based on my work and examinations to date.

209. I may continue my examinations. I may also receive additional documentation and other factual evidence over the course of this IPR that will allow me to supplement and/or refine my opinions. I reserve the right to add to, alter, or delete my opinions and my declaration upon discovery of any additional information. I reserve the right to make such changes as may be deemed necessary.

210. In signing this declaration, I recognize that the declaration will be filed as evidence in an IPR before the PTAB. I also recognize that I may be

subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross-examination within the United States during the time allotted for cross-examination.

211. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: 3/30/2022

By: 

Dr. Henry H. Houh
Burlington, MA

APPENDIX A

Henry H. Houh

15 Ballard Ter.
Lexington, MA 02420
henry.houh@gmail.com
cell: 781-325-8273

Education

Massachusetts Institute of Technology, Cambridge, MA

- PhD in Electrical Engineering and Computer Science, February 1998. “Designing Networks for Tomorrow's Traffic,” thesis supervised by Professor David Tennenhouse and Professor John Guttag. GPA 4.7/5.0
- Master of Science in Electrical Engineering and Computer Science, February 1991. “Demonstration of a laser repetition rate multiplier,” thesis. GPA 4.5/5.0
- Bachelor of Science in Electrical Engineering and Computer Science, June 1989. “Boundary element analysis of arbitrarily shaped dielectric structures,” thesis. GPA 4.7/5.0
- Bachelor of Science in Physics, February 1990. GPA 4.7/5.0

MITx, Cambridge, MA

- September 2020 – present: Enrolled in Statistics and Data Science MicroMasters program, a sequence of five MIT-offered graduate-level courses in statistics and machine learning, culminating in a MicroMasters certificate, expected 2022. Have completed four of five required courses.

Experience

BlocksCAD Inc. (H3XL spin-out)

- 2014 - present: Founder and member of Board of Directors. BlocksCAD is a visual Computer-Aided Design (CAD) program, designed to teach children mathematics and geometric visualization. BlocksCAD is being used in U.S. schools and worldwide in maker spaces, with 350,000+ users. BlocksCAD was selected to be part of the LearnLaunch 2017 Boost Accelerator, the MIT Play Labs 2018 Summer Accelerator, and the MassChallenge 2018 Boston Cohort. Recipient of DARPA DSO grant in 2014, and SBIR grant from the USDA in 2019.

H3XL Inc. d/b/a Einstein's Workshop (formerly Lexington Robotics)

- 2009 - present: Founder and President. Started local league providing science and engineering education programs based on LEGO Mindstorms, LEGO WeDo, and FIRST LEGO League. In 2012, grew program into a full science, technology, engineering and math enrichment program and creative/maker space, in 7,000 square feet of space. Serve 2,000+ kids and families annually. As of end of 2021, have delivered an estimated 220,000 student-instruction-hours of

STEM courses. Principle Investigator for 2-year DARPA grant to improve 3D Computer Aided Design tool developed in-house for the purposes of teaching 8 year olds and up 3D CAD.

Houh Consulting Inc. / Independent Consultant

- 2009 - present: Technical consultant specializing in Social Networking, Web 2.0, Web Site Development, Data Networking, Optical Networking, Telecommunications, Media Streaming and Voice Over IP. Clients include: Akin Gump Strauss Hauer & Feld LLP, Arnold & Porter Kaye Scholer LLP, BBN, Baker Botts LLP, Barnes & Thornburg LLP, BBN Technologies, Beus Gilbert McGroder PLLC, Bracewell LLP, Christensen O'Connor Johnson Kindness PLLC, Cooley LLP, Covington & Burling LLP, Cozen O'Connor P.C., DLA Piper, Davis Polk & Wardwell LLP, Desmarais LLP, Duane Morris LLP, Erise IP, P.A., Faegre Baker Daniels LLP, Finnegan, Henderson, Farabow, Garrett & Dunner, LLP, Fish & Richardson P.C., Goldman Ismail Tomaselli Brennan & Baum LLP, Haynes and Boone, LLP, Kaye Scholer, Kellogg Huber Hanson Todd Evans & Figel PLLC, Kirkland & Ellis LLP, Klarquist Sparkman, LLP, McCarthy Tetrault LLP, McGuireWoods LLP, Morgan, Lewis & Bockius LLP, Munger Tolles & Olson LLP, Orrick, Herrington & Sutcliffe LLP, O'Melveny & Myers LLP, Orrick, Herrington & Sutcliffe LLP, Paul Hastings LLP, Paul, Weiss, Rifkind, Wharton & Garrison LLP, Perkins Coie LLP, Quinn Emanuel Urquhart & Sullivan, LLP, Sidley Austin LLP, Smith Baluch LLP, Sterne, Kessler, Goldstein & Fox PLLC, Troutman Sanders LLP, Venable LLP, Weil, Gotshal & Manges LLP, Wiley Rein LLP, Wilmer Cutler Pickering Hale and Dorr LLP, and Winston & Strawn LLP.

Eons

- 2008 - 2009: Chief Technology Officer. Created product that Eons acquired from BBN Technologies. Integrated BBN product with Eons social networking platform and significantly increased the Eons group creation rate. Helped evaluate advertisement platform offerings and rolled out the "Boom Network" advertising network. Eons raised \$32 million from General Catalyst Partners, Charles River Ventures, Sequoia Capital, and Intel Capital.

BBN Technologies

- 2007 - 2008: Delta Division, Vice President of Technology. Grew "Boomerang" counter-sniper project engineering team and significantly de-risked \$10 million worth of product deliveries. Boomerang was a significant asset leading to the acquisition of BBN by Raytheon in 2009. Created new business plan and grew team; launched new fully-featured social networking web site in 5 months. Served as lead expert witness in patent infringement lawsuit, resulting in \$58 million jury award to client; verdict for patents I testified on were upheld on appeal which resulted in a \$120 million settlement.
- 2004 - 2007: Delta Division, Director of Technology, responsible for commercializing IP and creating new businesses. Hired and grew division's initial engineering team. Wrote three business plans, two of which are funded and active. For call center business plan, acted as general manager, hiring and managing engineering team, inside sales team, and identifying and recruiting a new general manager. Identified and recruited other key employees to Delta Division, including senior members of team leading to successful internal sales growth and spin-outs of projects. Contracted by BBN to BBN spin-out PodZinger as VP of Operations and

Technology. Identified sales team for counter-sniper system, leading to \$10 million dollars of new sales within 6 months and \$100+ million in additional orders in the following two years.

PodZinger Inc. (BBN spin-out, also known as EveryZing and now RAMP)

- 2006: Vice President of Operations and Technology. Significantly upgraded capability of consumer-facing search site and redeployed web site from company to co-location facility. Identified key portions of infrastructure for upgrading and cost reduction. Helped write business plan, evaluating advertisement models of revenue. Hired in operations replacement and phased back to BBN.

Commonwealth Capital

- 2004: Entrepreneur-in-residence (informally), performed technical due diligence on business plans, brainstormed ideas for new businesses with venture partner. With venture partner, left for portfolio company BBN to form core of commercialization team.

Empirix, Inc./Teradyne, Inc.

- 2001 - 2004: Chief Technologist, Engineering Manager, Web Application Test Group. Researched potential new product areas; developed product plan and prototype. Responsible for three new and existing products. Managed off-shore development team. Chief architect for all web testing products. Re-architected core testing product, helped write javascript interpreter. Provided technical vision for core product.
- 2000 - 2001: Chief Technologist, Communication Infrastructure Test Group. Responsible for incorporating new technology internally, tracking new technologies, technical evaluation of partnerships and potential acquisitions. Helped develop division strategy. Developed plans which formed core capabilities for successful new products introduced in 2004-5.
- 2000 - 2001: Engineering Manager, Communications Infrastructure Test Group. Execution of new product plan developed in prior role. Grew team from four existing engineers to team of over 30 on immediate team and over 40 on project. Delivered new platform in one year. Platform and derivatives accounted for large portion of booked products for the division within 2 years and is currently (2008) a key portion of new product offerings.
- 2000: Empirix was formed as a spin-out of Teradyne in January 2000. Reported to CEO in carve-out of Empirix from Teradyne.
- 1999: (Teradyne) Director of Business Development, Software Test Units. Reported directly to Chairman of the Board/Founder and then to general manager of software test unit (6 divisions of Teradyne). Evaluated and researched acquisition and partnership candidates. Internally assessed technology position in market and gaps in product lines. Worked with senior division staff to develop new product strategies. Attended internal Teradyne divisional board meetings. Chairman served as my mentor.

3Com Corporation/NBX Corporation

- 1999: Software Engineer 5. Continued work after 3Com acquisition of NBX. Built cross-division relationships for new products and research directions. NBX was acquired by 3Com in March 1999.

- 1997 - 1999: (NBX) Senior Scientist and Engineer. Work in IP Telephony. Architected next-generation product. Protocol design and validation for core protocol now used tens of millions of times daily. Led team in integration of IP protocols into current product. Designed audio reconstruction algorithms. Developed applications for bug analysis and diagnosis of system problems. Implementation of network simulator. Work on collaborative projects with external partners. Worked to identify gaps in product. Representative at numerous trade shows. Innovated novel methods of using product.

MIT Department of Electrical Engineering and Computer Science, Cambridge, MA

- 1991 - 1998: Research assistant, Telemedia Network Systems Group. Design, development, and implementation of Gigabit ATM network for distributed multimedia system. Studied host interface design issues. ATM network simulation.
- Spring 1989, Fall 1990, Spring 1995: Teaching assistant, Computation Structures digital systems course. (Spring 1995 Head TA)
- 1988 - 1989: Head laboratory teaching assistant for Computation Structures. Responsibilities included writing and revising lab assignments, and maintaining the lab.
- 1987: Laboratory teaching assistant for Computation Structures.
- 1987: Design, construction, and programming of 16-bit computer.

Agora Technology Group, Incorporated

- 1994 - 1996: Founder and CEO. Conceived and oversaw development of targeted advertising-supported Web sites. Responsible for company's vision and direction. Attended the first two WWW Conferences, presenting a workshop and paper at the first, and appearing on the "Commercialization and Economics of the Web" panel and chairing the "Where Commercial Services and the Web Are Headed" panel at the second. Sold company intact; is currently an operating stand-alone company.

AT&T Bell Laboratories, Holmdel, NJ

- 1989 - 1990: Implementation of cascadable all-optical fiber logic gate. Modelocking of all-fiber erbium laser. Construction of modelocked laser repetition rate booster. Strong optics laboratory and fiber optic experience.
- Summer 1988, 1987: Research in integrated optics. Analysis of rectangular waveguides using microwave modeling. Fabrication of integrated optical components.

Honors

- MIT Alumni Association Great Dome Award, 2010, Baker 60th Anniversary Reunion Co-Chair (highest group award given by MIT Alumni Association)
- MIT Alumni Association Presidential Citation Award (now known as Great Dome), 2008, Member of MIT Chairman's Salon committee
- MIT Alumni Association Bronze Beaver Award, 2007 (highest individual award given by MIT Alumni Association)
- MIT Alumni Association Volunteer Honor Roll, February 2004
- MIT Alumni Association Lobdell Award, 1999

- Boston Museum of Science Gold Pin for 1000 hours of Volunteer Service, April 1999
- MIT Alumni Association Presidential Citation Award (now known as Great Dome), 1997, Member of Alumni Online Communications Committee

Patents and Patent Publications

- U.S. Patent #9,697,231, H. Houh, J. Stern, R. Spina, M. Meteer, “Methods and apparatus for providing virtual media channels based on media search,” July 4, 2017.
- U.S. Patent #9,697,230, H. Houh, J. Stern, “Methods and apparatus for dynamic presentation of advertising, factual, and informational content using enhanced metadata in search-driven media applications,” July 4, 2017. See also WO2007056485.
- U.S. Patent #7,975,296, L. Apfelbaum, H. Houh, T. Mayberry and G. Friedman, “Automated security threat testing of web pages,” July 5, 2011. See also US20030159063, WO2003067405.
- U.S. Patent #7,877,736, H. Houh and J. N. Stern, “Computer language interpretation and optimization for server testing,” January 25, 2011. See also US20050138104, WO2005043300.
- U.S. Patent #7,801,910, H. Houh and J. N. Stern, “Method and apparatus for timed tagging of media content,” September 21, 2010. See also US20070112837, US20090222442, WO2007056535.
- U.S. Patent #7,590,542, D. C. Williams, W. C. Hand, H. Houh, A. R. Seeley, “Method of Generating Test Scripts Using a Voice-Capable Markup Language,” September 15, 2009. See also EP1530869, US20030212561, WO2003096663.
- U.S. Patent #6,967,963, H. H. Houh, P. Anderson, C. Gadda, “Telecommunication method for ensuring on-time delivery of packets containing time-sensitive data,” November 22, 2005. See also EP1060400, WO2000033092, CA2318774.
- U.S. Patent #5,144,375, M. C. Gabriel, H. H. Houh, N. A. Whitaker, “Sagnac Optical Logic Gate,” September 1, 1992. Also issued as European Patent # EP0456422, July 23, 1997, German Patent #DE69126913, August 28, 1997
- U.S. Patent Application Publication No. 20020015387, “Voice Traffic Packet Capture and Analysis Tool for a Data Network” (Abandoned in 2007)
- U.S. Patent Application Publication No. 20020016708, “Method and Apparatus for Utilizing a Network Processor as Part of a Test System” (Abandoned in 2007)
- U.S. Patent Application Publication No. 20020016937, “Method and Apparatus for Utilizing a Network Processor as Part of a Test System.” (Abandoned in 2007) See also WO2002011413.
- U.S. Patent Application Publication No. 20070106646, “User-directed navigation of multimedia search results” (Abandoned in 2009)
- U.S. Patent Application Publication No. 20070106660, “Method and apparatus for using confidence scores of enhanced metadata in search-driven media applications” (Abandoned in 2009)
- U.S. Patent Application Publication No. 20070106685, “Method and apparatus for updating speech recognition databases and reindexing audio and video content using the same” (Abandoned in 2015)
- U.S. Patent Application Publication No. 20070118873, “Methods and apparatus for merging media content” (Abandoned in 2015)
- U.S. Patent Application Publication No. 20090222442, “User-directed navigation of multimedia search results” (Abandoned in 2016)

- U.S. Patent Application 11/395,732, “Search snippet creation for audio and video data” (Abandoned in 2009)
- U.S. Patent Application 11/774,931, “Methods and apparatus for managing a social networking web site”
- U.S. Patent Application 11/774,947, “Methods and apparatus for organizing media files”
- U.S. Patent Application 11/774,956, “Methods and apparatus for managing an online event”
- U.S. Provisional Patent Application 61/086,909, “Measuring and ranking relationship activity”
- U.S. Provisional Patent Application 61/086,914, “Detecting media object commonality”
- U.S. Provisional Patent Application 61/086,904, “Message categorization based on message characteristics”
- U.S. Provisional Patent Application 61/086,905, “Photo tagging to request action”

Trials, Testimony and Depositions

- Case No. 1:06CV682 (CMH/BRP), Verizon vs. Vonage, U.S. District Court for the Eastern District of Virginia, filed expert report, was deposed and testified at trial.
- Case No. 1:08CV157 (CMH/TRJ), Verizon vs. Cox, U.S. District Court for the Eastern District of Virginia, was deposed and testified at trial.
- Case No. 5:09-cv-476, Two-Way Media vs. AT&T, U.S. District Court for the Western District of Texas, filed expert report, testified at trial.
- Case No. 2:10-cv-248 (RAJ/FBS), ActiveVideo Networks vs. Verizon, U.S. District Court for the Eastern District of Virginia, filed expert report and was deposed as an expert witness.
- Case No. 1:11-cv-00880-TSE-JFA, Bear Creek Technologies, Inc. vs. Verizon Services Corp., et al, U.S. District Court for the Eastern District of Virginia, was deposed as an expert witness.
- Case No. 3:10-CV-298-BBC, AlmondNet, Inc. vs. Microsoft Corp., U.S. District Court for the Western District of Wisconsin, filed expert report.
- Case No. 6:10-cv-00597, Guardian Media Technologies, Ltd. Vs. AT&T Operations, Inc. et al., U.S. District Court for the Eastern District of Texas, Tyler Division, filed expert report.
- Case No. ESCV2010-02282C, The Octopus Solution LLC v. Gary Brown et al., Essex, MA Superior Court, testified at trial.
- Investigation No. 337-TA-882, In the matter of Certain digital media devices, including televisions, Blu-ray disc players, home theater systems, tablets and mobile phones, components thereof and associated software, U.S. International Trade Commission, filed expert reports, was deposed and testified at hearing.
- Investigation No. 337-TA-995, In the matter of Certain communications or computing devices, and components thereof, U.S. International Trade Commission, filed expert reports, and was deposed.
- Case No. 8:12-cv-122-LES-TDT, Prism Technologies LLC v. AT&T Mobility LLC, U.S. District Court for the District of Nebraska, filed expert report, was deposed and testified at trial.
- Case No. 11-2684 (JWL), Sprint Communications Co., L.P., v. Comcast Cable Communications, LLC, et al., U.S. District Court for the District of Kansas, filed expert report and was deposed.
- Case No. 11-2686 (JWL), Sprint Communications Co., L.P., v. Time Warner Cable, Inc., et. al., LLC, et al., U.S. District Court for the District of Kansas, filed expert report, was deposed, and testified at trial.
- Case No. 12-487 (SLR), Cox Communications, Inc., et al., v. Sprint Communications Company L.P., et al., U.S. District Court for the District of Delaware, filed expert report and was deposed.

- Case IPR 2014-00039, Microsoft Corporation v. B.E. Technology LLC, U.S. Patent No. 6,628,314. Submitted declaration and cross-examined in deposition.
- Case IPR 2014-00086, Apple, Inc. v. Evolutionary Intelligence, LLC, U.S. Patent No. 7,010,536. Submitted declaration and cross-examined in deposition.
- Case No. 6:11-CV-421, Stragent, LLC v. Intel Corporation, U.S. District Court for the Eastern District of Texas, Tyler Division, deposited and testified at trial as a fact witness.
- IPR2014-01366, IPR2014-01367, IPR2014-01368, Samsung et al. v. Straight Path, submitted IPR declarations and was deposited.
- IPR2015-1006, IPR2015-1007, Cisco v. Straight Path, submitted IPR declarations.
- IPR2014-01457, IPR2014-01459, Microsoft v. Biscotti, submitted IPR declaration and was deposited.
- IPR2016-00302, Apple v. Nonend, submitted IPR declaration.
- IPR2014-00812, Yelp and Twitter v. Evolutionary Intelligence, submitted IPR declaration.
- IPR2015-00307, Cisco v. AIP, submitted IPR declaration.
- Microsoft v. Acacia, submitted IPR declaration.
- IPR2015-00342, AT&T Mobility LLC and Cellco Partnership d/b/a Verizon Wireless v. Solocron, submitted IPR declaration.
- Mediatube v. Bell Canada, Canadian Federal Court, submitted expert report and testified at trial.
- CBM2016-00036, Google v. At Home Bondholder's Liquidating Trust, submitted CBM declaration.
- IPR2016-01198, IPR2016-01201, Apple, Inc. v. VoIP-Pal.com, Inc., submitted IPR declaration and was deposited.
- Case No. 3:14-cv-2824-VC, Trend Micro Incorporated v. Rpost Holdings, Inc., et al, submitted expert report and was deposited
- Akamai v. Limelight Networks, IPR2016-01631, IPR2016-01711, IPR2017-00249, IPR2017-00349, submitted IPR declarations and was deposited
- Hyperlync Technologies v. Verizon Sourcing, No. 650151/2015 New York Supreme Court, submitted declaration and was deposited
- Cascades Streaming Technologies v. Big Ten Networks, submitted expert reports and was deposited
- Klaustech, Inc. v. Google LLC., Case No. 4:10-CV-05899-JSW (N.D. Cal.), submitted expert report and was deposited
- Cisco v. Uniloc, IPR2017-00198, IPR2017-00058, submitted IPR declarations and was deposited.
- Cisco v. Uniloc, IPR2017-00597, submitted IPR declaration and was deposited
- Twitter v. Youtoo Technologies, LLC, IPR2017-01133, IPR2017-01131, submitted IPR declarations and was deposited
- Taser International Inc. (now Axon Enterprise, Inc.) v. Digital Ally Inc., IPR2017-00375, submitted IPR declaration and was deposited
- Taser International Inc. (now Axon Enterprise, Inc.) v. Digital Ally Inc., IPR2017-00376, IPR2017-00515, IPR2017-00775, submitted IPR declarations
- HTC Corporation v. IPA Technologies, IPR2018-00306, IPR2018-00307, submitted IPR declaration
- Hulu v. Soundview, IPR2018-00864, IPR2018-01023, submitted IPR declarations and was deposited
- XpertUniverse v. Cisco, 3:17-cv-3848, U.S. District Court for Northern District of California, submitted declaration in support of claim construction and was deposited
- Improved Search v. Microsoft, C.A. No. 16-cv-650-JFB-SRF (Del.), submitted declaration in support of claim construction

- Apple v. Uniloc, IPR2018-00361, IPR2018-00394, IPR2018-00395, IPR2019-00700, IPR2019-00701, IPR2019-01667, submitted IPR declarations
- LG Electronics v. Uniloc, submitted IPR declaration
- SRC Labs and Saint Regis Mohawk Tribe v. Microsoft, 2:18-cv-00321-JLR, U.S. District Court for Western District of Washington at Seattle, submitted declaration in support of claim construction and was deposed
- Verizon Wireless v. Barkan Wireless IP Holdings LP, submitted IPR declarations
- Verizon Wireless and Samsung Electronics America Inc. v. Barkan Wireless IP Holdings LP, IPR2019-199, IPR2019-200, IPR2019-234, IPR2019-631, IPR2019-632, submitted IPR declarations
- Samsung Electronics America Inc. v. Barkan Wireless IP Holdings LP, IPR2019-100, submitted IPR declaration
- Cisco Systems Inc. v. Meetrix IP LLC, IPR2019-539, IPR2019-540, IPR2019-541, IPR2019-542, IPR2019-543, IPR2019-544, submitted IPR declarations
- GMG Products LLC v. Traeger Pellet Grills LLC, PGR2019-24, PGR2019-34, PGR2019-35, PGR2019-36, submitted PGR declaration and was deposed
- Uniloc 2017 LLC v. Google LLC, 2:18-cv-00502-JRG-RSP, U.S. District Court for the Eastern District of Texas, Marshall Division, submitted declaration in support of claim construction
- Microsoft Corporation v. Uniloc 2017 LLC, IPR2020-00102, IPR2020-00103, submitted IPR declarations
- Microsoft Corporation and HP Inc. v. Syncloud Technologies, LLC, IPR2020-00316, IPR2020-01031, IPR2020-01032, IPR2020-01269, IPR2020-01270, IPR2020-01271, submitted IPR declarations
- Seven Networks, LLC v. Apple Inc, 2:19-cv-115-JRG, U.S. District Court for the Eastern District of Texas, Marshall Division, submitted declaration in support of claim construction
- Unified Patents Inc. v. Ortiz & Associates Consulting, LLC, IPR2019-00743, submitted IPR declaration
- Uniloc 2017 LLC v. Apple Inc., 3:19-cv-01905-JD, U.S. District Court for the Northern District of California, San Francisco Division, submitted declaration in support of claim construction and was deposed
- ClearPlay Inc. v. Dish Network LLC et al.; 2:14-cv-00191, U.S. District Court for the District of Utah, Central Division, submitted expert report and re-examination declarations
- AMC Networks, Inc. et al v. Sound View Innovations, LLC, IPR2020-00482, submitted IPR declaration
- Case No. 1:17-cv-01734-RGA, Sprint Communications Co., L.P., v. Charter Communications, Inc., et al., U.S. District Court for the District of Delaware, filed expert report and was deposed.
- CommScope Technologies LLC et al. v. Barkan Wireless IP Holdings LP et al., IPR2020-00827, IPR2020-00829, IPR2020-00831, IPR2020-00833, IPR2020-00835, IPR2020-00838, submitted IPR declarations
- Cisco Systems Inc. v. Monarch Networking Solutions LLC, IPR2020-01226, submitted IPR declaration
- CoolTV Network.com v. Blackboard, Inc., Facebook, Inc., International Business Machines Corporation, Kaltura, Inc., Limelight Networks, Inc., Microsoft Corporation, Ooyala, Inc., Snap, Inc., and Trapelo Corp., submitted declaration in support of claim construction and was deposed
- Facebook, Inc. v. Onstream Media Corporation, IPR2020-01507, IPR2020-01508, IPR2020-01525, IPR2020-01527, IPR2020-01528, submitted IPR declarations
- Hisense Co LTD. v. Maxell, LTD, IPR2020-01598, IPR2020-01600, submitted IPR declarations
- Microsoft Corp. v. O'Brien, IPR2021-00015, submitted IPR declaration and was deposed

- Sable Networks Inc. v. Cisco Systems, Inc., WA:20-CV-00288-ADA, U.S. District Court for the Western District of Texas, Waco Division, submitted declaration in support of claim construction
- Peloton Interactive, Inc. v. Icon Health and Fitness Inc et al., IPR2021-00342, submitted IPR declaration
- Hulu, LLC v. SITO Mobile R&D IP, LLC et al., CBM2020-00028, IPR2020-00158, IPR2020-00206 , IPR2020-00219 , IPR2020-00265 , IPR2020-00298 , IPR2020-00304 , IPR2020-00308 , submitted CMB and IPR declarations and was deposed
- Twitter, Inc. and Google LLC v. B.E. Technology, LLC, IPR2021-00482, IPR2021-00483, IPR2021-00484, IPR2021-00485, submitted IPR declarations
- MG Freesites LTP v. ScorpCast, LLC, IPR2021-00511, submitted IPR declaration
- Avionte, LLC v. Sandeep Acharya, Samar Basnet, and Aqore LLC, Arbitration, Hon. Jeffrey Keyes (Ret.)
- Microsoft Corporation v. WSOU Investments, LLC d/b/a Brazos Licensing and Development, IPR2021-00751, submitted IPR declaration
- SAP America, Inc. v. Express Mobile, Inc., IPR2021-01144, IPR2021-01145, IPR2021-01146, submitted IPR declarations
- StratosAudio Inc. v. Hyundai Motor America, 6:20-cv-01125-ADA, U.S. District Court for the Western District of Texas, Waco Division, submitted claim construction declaration and was deposed
- Peloton Interactive, Inc. v. Ifit Health and Fitness Inc, IPR2022-00029, IPR2022-00030, submitted IPR declarations and was deposed
- Investigation No. 337-TA-1263, In the matter of Certain televisions, remote controls, and components thereof, U.S. International Trade Commission, filed expert reports and was deposed
- Alexander Stross v. Netease, Inc., et al., U.S. District Court (Central District of California), Case 2:20-cv-00861-AB(JC), filed expert reports and was deposed
- Ralph Ledergerber v. Netease, Inc., et al., U.S. District Court (Central District of California), Case 2:20-cv-00862-AB(JC), filed expert reports and was deposed
- John Walmsley v. Netease, Inc., et al., U.S. District Court (Central District of California), Case 2:20-cv-00863-AB(JC), filed expert reports and was deposed
- Bernhard Kuhmstedt v. Netease, Inc., et al., U.S. District Court (Central District of California), Case 2:20-cv-02044-AB(JC), filed expert reports and was deposed
- F45 Training Pty Ltd. v. Body Fit Training USA Inc., C.A. No. 20-1194 (LPS) (D. Del.), filed expert report

Publications

- “IP switching: server driven flow classification,” H. H. Houh, *Proceedings of the Washington University Workshop on Integration of IP and ATM*, November 1996.
- “Aurora at MIT,” D. D. Clark, H. H. Houh, and D. L. Tennenhouse, Editors, *MIT Laboratory for Computer Science Technical Report 673*, December 1995.
- “ViewStation Applications: Implications for Network Traffic,” C. J. Lindblad, D. Wetherall, W. Stasior, J. F. Adam, H. H. Houh, M. Ismert, D. Bacher, B. Phillips, and D. L. Tennenhouse, *IEEE Journal of Selected Areas in Communications*, 1995.
- “The VuNet Desk Area Network: Architecture, Implementation, and Experience,” H. H. Houh, J. F. Adam, M. Ismert, C. J. Lindblad, and D. L. Tennenhouse, *IEEE Journal of Selected Areas in Communications*, 13 (4), May, 1995.
- “Reducing the Complexity of ATM Host Interfaces,” H. H. Houh and D. L. Tennenhouse, Hot Interconnects II Symposium Proceedings, Stanford, August 11-12, 1994.

- “Media-intensive data communications in a ‘desk-area’ network,” J. F. Adam, H. H. Houh, M. Ismert, and D. L. Tennenhouse, IEEE Communications, August 1994.
- “ViewStation Applications: Intelligent Video Processing Over A Broadband Local Area Network,” C. J. Lindblad, D. J. Wetherall, W. Stasior, B. Phillips, D. Bacher, J. Adam, H. Houh, M. Ismert, and D. L. Tennenhouse, Proceedings of the 1994 USENIX Symposium on High-Speed Networking, Oakland, CA, August 1994.
- “The Media Gateway: Live Video on the World Wide Web,” H. H. Houh, C. J. Lindblad, J. Soo, D. L. Tennenhouse, and D. J. Wetherall, Workshop at the 1994 World Wide Web Conference, Geneva, Switzerland, May 1994.
- “Active Pages: Intelligent Nodes on the World Wide Web,” H. H. Houh, C. J. Lindblad, and D. J. Wetherall, Proceedings of the 1994 World Wide Web Conference, Geneva, Switzerland, May 1994.
- “Wavelength Division vs. Code Division Access Methods for Optical Networks,” H. H. Houh, Area Exam Paper, May 1993.
- “Experience with the VuNet: A Network Architecture for a Distributed Multimedia System,” J. F. Adam, H. H. Houh, D. L. Tennenhouse, Proceedings of the 18th Conference on Local Computer Networks, pp. 70-76, September 1993
- “The VudBoard: A Simple DMA Interface,” H. H. Houh, Proceedings of the 4th Gigabit Minijam, January 1994.
- “A Software-Oriented Approach to the Design of Media Processing Environments,” D. L. Tennenhouse, J. Adam, D. Carver, H. Houh, M. Ismert, C. Lindblad, W. Stasior, D. Weatherall, D. Bacher, and T. Chang., submitted to the International Conference on Multimedia Computing and Systems, May 1994.
- “A Network Architecture for Distributed Multimedia Systems,” J. F. Adam, H. H. Houh, M. Ismert, and D. L. Tennenhouse, submitted to the International Conference on Multimedia Computing and Systems, May 1994.
- “The Viewstation Collected Papers,” D. L. Tennenhouse, J. Adam, C. Compton, A. Duda, D. Gifford, H. Houh, M. Ismert, C. Lindblad, W. Stasior, R. Weiss, D. Wetherall, D. Bacher, D. Carver, and T. Chang, MIT Laboratory for Computer Science Technical Report, MIT/LCS/TR-590, November 1993.
- “A System's Perspective of the Sagnac Fiber Logic Gates and Their Possible Applications,” A. Huang, N. Whitaker, C. Gabriel, H. Avramopoulos, P. M. W. French, H. H. Houh, and I. Chuang, Applied Optics, September 10, 1994
- “Complete Switching in a Three-Terminal Sagnac Switch,” H. Avramopoulos, P. M. W. French, M. C. Gabriel, H. H. Houh, N. A. Whitaker, T. Morse, IEEE Phot. Tech. Lett. **3** (3), 235
- “Complete Switching in a Three-Terminal Sagnac Switch,” H. Avramopoulos, P. M. W. French, M. C. Gabriel, H. H. Houh, N. A. Whitaker, IEEE/LEOS Annual Meeting, Paper PDP-13, November 1990
- “All-optical phase-locked oscillator,” N. A. Whitaker, Jr., H. H. Houh, H. Avramopoulos, T. F. Morse, IEEE/LEOS Annual Meeting, Paper ELT2.4/MOO3, November 1990
- “Passive modelocking of an all-fiber erbium laser,” H. Avramopoulos, H. H. Houh, N. A. Whitaker, M. C. Gabriel, T. F. Morse, IEEE/LEOS Conference on Optical Amplifiers and their Applications, Paper PDP-8, August 1990
- “Transverse modes, waveguide dispersion, and 30ps recovery in submicron GaAs/AlAs microresonators,” J. L. Jewell, S. L. McCall, A. Scherer, H. H. Houh, N. A. Whitaker, A. C. Gossard, and J. H. English, Appl. Phys. Lett. **55** (1), July 3, 1989

Leadership, Activities and Interests

- Leadership
 - Discovery Museums (Acton, MA)
 - Science and Technology Advisory Council, 2012 - present
 - MIT Alumni Association Board of Directors
 - National Selection Committee, 3 year term, 2020 – 2023
 - K-12 STEM Initiatives Co-chair, 2013 – 2017
 - Awards Committee Chair, 2012 – 2014
 - Awards Committee, 3 year term, 2011 – 2014
 - Vice President, 2 year term, 2004 – 2006
 - Alumni Ad-Hoc Committee, Chair, 2005-2006
 - Board Member, 2 year term, 1997 – 1999
 - MIT Club of Boston
 - Board of Directors, 2006 - 2011
 - K-12 Initiatives Chair, 2009 - 2012
 - VP of Communications, MIT Club of Boston, 2003 - 2006
 - Past-President, MIT Club of Boston, 2002 - 2003
 - President, MIT Club of Boston, 2001 - 2002
 - President-Elect, MIT Club of Boston, 2000 - 2001
 - VP of Programs, MIT Club of Boston, 1999 - 2000
 - Activities Super-Chair, MIT Club of Boston, 1998 - 1999
 - MIT Enterprise Forum of Cambridge, Inc.
 - Past Chair, 2009 – 2011
 - In-NOW-vation Co-chair, 2010
 - Chair, 2007 - 2009
 - Vice Chair, 2005 - 2007
 - Executive Board Member, 2002 - 2011
 - Winter Workshop Co-Chair, February 2007 - conceived idea for conference, which sold-out and produced largest attendance numbers in recent memory
 - Spring Workshop Co-Chair, Spring 2004
 - Membership Committee Chair, Fall 2003 - 2006
 - 25th Anniversary Dinner Chair, Fall 2003
 - As Membership Chair and Board Member, started Special Interest Groups in 2004; a SIG won the MIT Presidential Citation award, the MIT Alumni Association's highest award for organizations, in 2006
 - Estabrook Elementary School PTA
 - Advisory committee to the superintendent on PCB issue, 2010-2011
 - 4th Grade after-school science program co-organizer, 2010-2012
 - 4th and 5th Grade before-school Math Olympiad co-organizer, 2009-2013
 - 5th Grade BBQ and Yearbook Committee, 2011, 2013
 - Family Math Night volunteer, 2008-2012
 - Tau Beta Pi National Engineering Honor Society
 - Advisor, MA B Chapter at MIT, 2003 - present
 - District Director (National Officer), Tau Beta Pi, New England Area, 1991 - 2003
 - President, MA B Chapter at MIT, Fall 1988 - Spring 1989
 - Laureate award, 1989

- MIT Class of 1989
 - Secretary, five consecutive 5 year terms, 1989 - 2014
 - 30-year Reunion Committee, 2019
 - 25-year Reunion Committee, 2014
 - 20-year Reunion Committee and Gift Committee, 2009
 - 15-year Reunion Committee and Gift Committee, 2004
 - 10-year Reunion Committee and Gift Committee, 1999
 - 5-year Reunion Committee, 1994
 - Interim Treasurer, 1993 - 1994
 - Instituted annual senior class career fair, now raising over \$100,000 annually for senior class activities, Fall 1988
- Strong, consistent record of leadership dating to high school
- Acting
 - ‘21’ (Sony Pictures), credited as “Chinatown Dealer,” 2007, Kevin Spacey's movie about the MIT Blackjack Team inspired by “Bringing Down the House” by Ben Mezrich, opened nationwide on March 28, 2008. 21 was the number one movie in US for two weeks and number one globally for one week. 21 also topped the DVD sales, Blu-ray sales and DVD rental charts.
 - Spring Lake Theater Company, first New York-area off-broadway production of “A Chorus Line,” played role of Mark, Summer 1990
- Former member of the MIT Blackjack team
- Producer for 10,000 Maniacs’ 2013 album “Music from the Motion Picture”
- Executive Producer for 10,000 Maniacs’ 2015 album “Twice Told Tales”
- Violist, violist, harpist, guitarist, singer, actor: played in many amateur/semi-professional groups including Reading Symphony Orchestra, MIT Summer Philharmonic Orchestra, Freisinger Chamber Orchestra, Merrimack Valley Philharmonic, Longwood Symphony, MIT Symphony, and Somerville Community Chorus
- Violist, “Coming Home” album by Karen Phillips, 2014
- Google Scholar citations: 1914, h-index: 21, i10-index: 26 (since 2016: 649/13/13)
- Erdos-Bacon number of 7
- Erdos-Bacon-Sabbath number of 11