

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTEL CORPORATION,
Petitioner

v.

BiTMICRO LLC,
Patent Owner.

IPR2023-00779

U.S. Patent No. 9,135,190

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 9,135,190**

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	GROUND FOR STANDING.....	1
III.	IDENTIFICATION OF CHALLENGE	1
	A. Prior Art	1
	B. Grounds For Challenge	2
IV.	'190 PATENT	2
	A. Overview	2
	B. Level Of Ordinary Skill In The Art	4
	C. Claim Construction	5
V.	GROUND THE COMBINATION OF DANILAK AND CORNWELL RENDERS CLAIMS 1-2, 5-8, 11, 19, 29-30, 33, 36-39, 42, 94, AND 100 OBVIOUS.	5
	A. Overview of the Combination.....	5
	1. Danilak.....	5
	2. Cornwell.....	6
	3. Combination of Danilak and Cornwell.....	8
	a. Cornwell's Non-Volatile Memory Structure.....	8
	b. NAND and NOR Flash Devices	11
	B. Independent Claim 1	15
	1. Structural Limitations	18
	a. "Memory Device"	19
	b. "Memory Locations"	19
	c. "Memory Store"	21
	d. "Controller"	22
	e. "Memory Bus"	26
	f. "Communication Path"	28

TABLE OF CONTENTS (continued)

	Page
2. “Device Profile” Limitations	29
a. “Device Profile”	30
b. “Attributes”	33
c. “Attribute Qualifier”	35
d. “Device Profile is Optimal ...” [1A.6].	37
3. Preamble and Method Steps	40
a. Preamble [1P]/ “Addressing a First Memory Location ...” [1A].	40
b. “Identifying Command Details ...” [1B].	43
c. “Obtaining the First Set of Attributes” [1C] and “Using Attributes From Said First and Second Device Profiles” [1D].	45
d. “Selecting a Transfer Size for the Memory Transaction” [1E].	48
4. Read-Write Limitations	50
a. Controller Performs the Memory Read or Memory Write Transaction on a Memory Location.	51
b. Write Data Flow Plath	52
c. Read Data Flow Path	53
C. Independent Claim 29	54
1. Preamble [29P].	55
2. Method Limitations	56
D. Independent Claim 100	59
E. Dependent Claims	63
1. First/Second Memory Location Claims 2 and 33	63

TABLE OF CONTENTS
(continued)

	Page
2. “ <i>Attribute Qualifier</i> ” Claims 5-7, 36-38	63
3. “ <i>Attribute</i> ” Claims 8, 11, 39, 42	64
4. NAND/NOR Attribute Claim 19	66
5. “ <i>Memory Space</i> ” Claims 30 and 94	66
VI. DISCRETIONARY DENIAL IS NOT APPROPRIATE	67
A. Parallel Litigation	67
B. Prior Petition	68
VII. MANDATORY NOTICES	69
A. Real Party In Interest	69
B. Related Matters	69
C. Notice of Counsel and Service Information	70
VIII. CONCLUSION	70

EXHIBIT LIST

Exhibit	Reference
1001	U.S. Patent No. 9,135,190 to Bruce et al. (“the ’190 patent”)
1002	File History of the ’190 patent
1003	Declaration of Dr. Donald Alpert, Ph.D. in Support of Petition
1004	CV of Dr. Donald Alpert, Ph.D.
1005	U.S. Patent Publication 2008/0086588 A1 to Danilak et al. (“Danilak”)
1006	U.S. Patent Publication 2007/0180186 A1 to Cornwell et al. (“Cornwell”)
1007	Open NAND Flash Interface Specification, Revision 2.0 (February 27, 2008) (“ONFI specification”)
1008	Joe E. Brewer and Manzur Gill, “Nonvolatile Memory Technologies with Emphasis on Flash: A Comprehensive Guide to Understanding and Using NVM Devices” (IEEE Press, 2008) (“Brewer”)
1009	Peter DiPaolo, “NOR continues to battle NAND flash memory in the Handset” (Design & Reuse, July 12, 2005)
1010	Technical Note NAND Flash 101: An Introduction to NAND Flash and How to Design It In to Your Next Product, Micron Technology Technical Note 29-19 (Micron Technology Inc., November 2006)
1011	NAND vs. NOR Flash Memory Technology Overview (Toshiba America Electronics Components, Inc., April 25, 2006)
1012	G. Campardo, R. Micheloni, and D. Novosel, “VLSI Design of Non-Volatile Memories” (Springer-Verlag Berlin Heidelberg, 2005)
1013	“PoP, SiP, MCM, MCP or SoC? Assessing the Mobile/Embedded Design Tradeoffs” (Technical Article, Embedded, December 15, 2006)
1014	Chul Lee, Sung Hoon Baek, and Kyu Ho Park, “A Hybrid Flash File System Based on NOR and NAND Flash Memories

Exhibit	Reference
	for Embedded Devices (IEEE Transactions on Computers, Vol. 57, No. 7, July 2008)
1015	Rudolf F. Graf, Modern Dictionary of Electronics (Seventh Edition) (Newnes, 1999), pp. 93, 621, 680
1016	Kostas Magoutis, et al., “Structure and Performance of the Direct Access File System,” In Proceedings of the General Track of the annual conference on USENIX Annual Technical Conference (ATEC ‘02). USENIX Association, USA, 1–14
1017	IEEE 610.10-1994 IEEE Standard Glossary of Computer Hardware Terminology (Standards Coordinating Committee of the IEEE Computer Society, June 14, 1994)
1018	David A. Patterson and John L. Hennessy, Computer Organization and Design The Hardware/Software Interface (Third Edition, Morgan Kaufmann Publishers, 2004)
1019	MC68010/MC68012 16-/32-Bit Virtual Memory Microprocessors (Motorola Inc., May 1995)
1020	Pentium® Pro Family Developer’s Manual, Volume 1: Specifications (Intel Corporation, 1997)
1021	“Selecting and Implementing NAND Flash Storage in Embedded Applications” (Technical Article, Embedded, March 4, 2008)
1022	[Reserved]
1023	Numonyx™ StrataFlash® Embedded Memory (P30) Datasheet (Numonyx, B.V., August 2008)
1024	Defendants’ Opening Claim Construction Brief, Case Nos. 6:22-cv-00331-ADA and 6:22-cv-00335-ADA (W.D. Tex.) (Dkt. No. 33)
1025	Public version of Plaintiff BitMicro’s Responsive Claim Construction Brief, Case Nos. 6:22-cv-00331-ADA and 6:22-cv-00335-ADA (W.D. Tex.) (Dkt. No. 44)
1026	Summons and Proof of Service to Intel Corporation, Case No. 6:22-cv-00335-ADA (W.D. Tex.)

Exhibit	Reference
1027	United States District Courts - National Judicial Caseload Profile

I. Introduction

Intel Corporation (“Petitioner”) petitions for *inter partes* review (“IPR”) of U.S. Patent No. 9,135,190 (“the ’190 patent”; INTEL-1001) claims 1-2, 5-8, 11, 19, 29-30, 33, 36-39, 42, 94, and 100 (“the challenged claims”).

II. Grounds For Standing

Intel certifies the ’190 patent is available for IPR and Petitioner is not barred or estopped from requesting IPR on the grounds herein. (*See* INTEL-1026.)

III. Identification Of Challenge

A. Prior Art

The face of the ’190 patent, filed September 4, 2010, claims priority to Provisional Application No. 61/239,794, filed September 4, 2009. (INTEL-1001 (63), 1:16-19.) Each applied reference listed below was filed or published before the earliest possible priority date of September 4, 2009.

1. **U.S. Patent Publication 2008/0086588 to Danilak, et al** (“Danilak”; INTEL-1005), published April 10, 2008, is prior art under §102(b).

2. **U.S. Patent Publication 2007/0180186 to Cornwell, et al** (“Cornwell”; INTEL-1006), published August 2, 2007, is prior art under §102(b).

None of the applied references were cited in or discussed during prosecution of the ’190 patent, weighing heavily against discretionary denial under 35 U.S.C. §325(d).

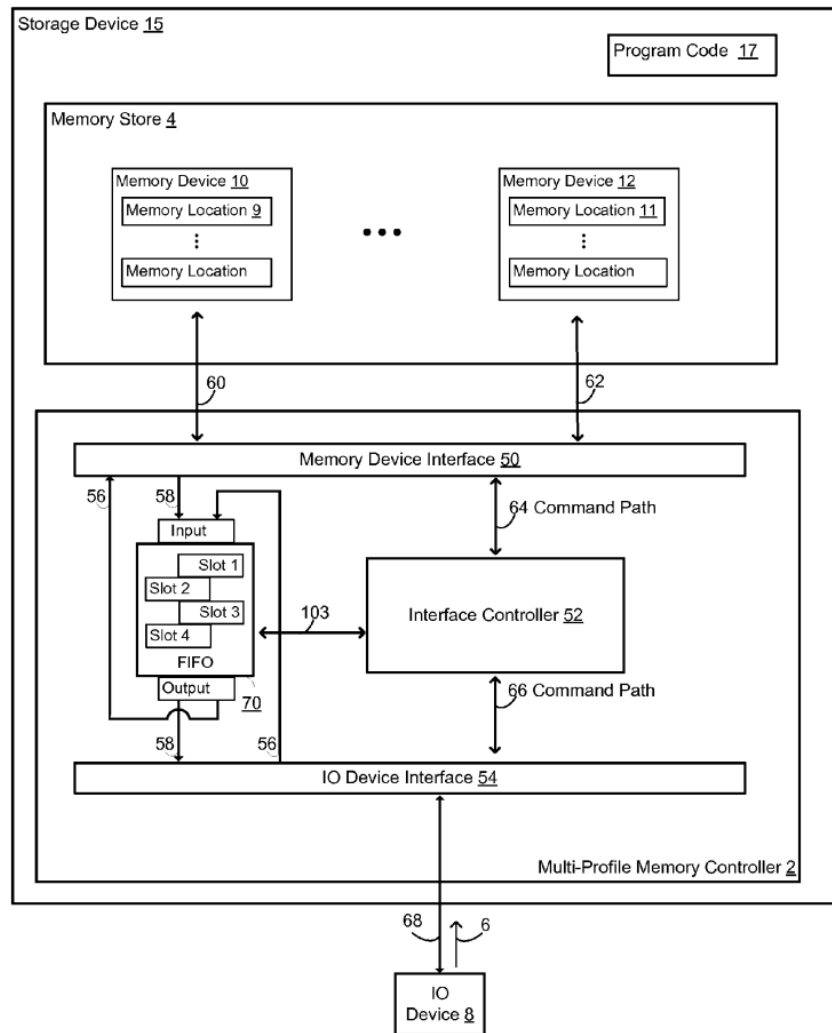
B. Grounds For Challenge

Claims 1-2, 5-8, 11, 19, 29-30, 33, 36-39, 42, 94, and 100 are obvious over the combination of Danilak and Cornwell.

IV. '190 Patent

A. Overview

The '190 patent “relates to multi-profile memory controllers and computing devices, such as solid-state storage devices, that use these multi-profile memory controllers.” (INTEL-1001, 1:17-19.) The multi-profile memory controller 2, illustrated in Figure 1 below, “perform[s] a memory transaction on a memory store 4 in response to receiving a memory transaction request 6” initiated by an IO device 8. (INTEL-1001, 2:50-55.) The memory store “includes addressable memory locations, such as memory locations 9 and 11, that are respectively associated with a set of at least one memory attribute.” (INTEL-1001, 2:63-65.)



'190 Patent, Figure 1

The set of memory attributes for a device is organized into a device profile, as shown in Figure 2A below. The attributes “include a page size 16, a block size 18, a memory protocol 20 supported by the memory device in which the memory location is addressable, a memory device type 22, and a partial page size 23.” (INTEL-1001, 3:25-29.) Figures 2B and 2C illustrate exemplary device profiles having shared attributes (page size, block size, memory protocol, device type, and

partial page size). (*See, e.g.*, INTEL-1001, 4:8-11.) The exemplary profiles differ “by attribute qualifier for their sets of shared attributes.” (INTEL-1001, 4:11-4:14.) For example, page sizes, block sizes and partial page sizes “differ by their attribute values” and memory protocols and memory device types “differ by their attribute types.” (INTEL-1001, 4:14-17.)

24		25	
Page Size	<u>16</u>	(# of pages)	
Block Size	<u>18</u>	(# of blocks)	
Memory Protocol	<u>20</u>	(Protocol Type Supported)	
Mem. Dev. Type	<u>22</u>	(Mem. Type)	
Partial Page Size	<u>23</u>	(# of pages)	

FIG. 2A

26	
Page Size	<u>28</u> 1024
Block Size	<u>30</u> 2048
Memory Protocol	<u>32</u> ONFi
Mem. Dev. Type	<u>34</u> NAND Flash
Partial Page Size	<u>36</u> 512

FIG. 2B

38	
Page Size	<u>40</u> 512
Block Size	<u>42</u> 8192
Memory Protocol	<u>44</u> ONFi
Mem. Dev. Type	<u>46</u> NAND Flash
Partial Page Size	<u>48</u> 256

FIG. 2C

'190 Patent, Figures 2A-2C

B. Level Of Ordinary Skill In The Art

A person of ordinary skill in the art (“POSITA”) would have had at least a bachelor’s degree in electrical or computer engineering and two years of experience working with storage systems (including storage systems made up of different types of storage devices), file management policies, and different data

access patterns, or an equivalent combination of education and experience.

(INTEL-1003, ¶57.)

C. Claim Construction

Petitioner does not believe any terms require construction in this IPR proceeding. Therefore, Petitioner and its expert have applied the plain and ordinary meaning to each claim term.

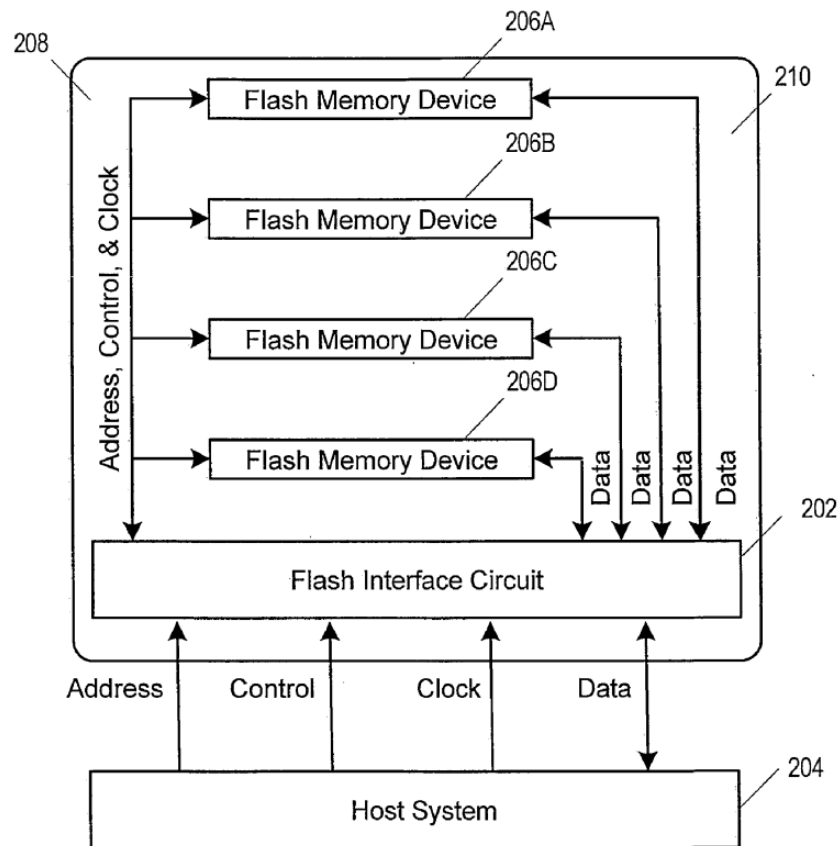
V. GROUND The Combination of Danilak and Cornwell Renders Claims 1-2, 5-8, 11, 19, 29-30, 33, 36-39, 42, 94, and 100 Obvious.

A. Overview of the Combination

1. Danilak

Danilak “relates to memory, and more particularly to enhanced capacity, performance, flexibility, and reliability in multiple flash memory circuit systems.” (INTEL-1005, ¶2.) Danilak’s memory, illustrated in Figure 2 below, includes several flash memory devices 204A-204N connected to host 206 through flash interface circuit 202. The flash interface circuit exposes the attached flash memory devices to the system as a single flash memory device, referred to as a “virtual flash memory device.” (See INTEL-1005, ¶17, *see also*, ¶¶51, 80.) As such, “[o]ne or more attributes of the virtual flash memory device(s) may differ from the attributes of the flash memory devices 104A-140N.” (INTEL-1005, ¶17, *see also*, ¶¶51, 80.) In operation, when the host makes a memory access request, the flash interface circuit “translate[s] the request into one or more physical flash memory

device accesses”, “emulate[s] attributes of the virtual configuration that differ from the physical flash memory devices”, and “return[s] an appropriate response to the request to the host.” (INTEL-1005, ¶78.)

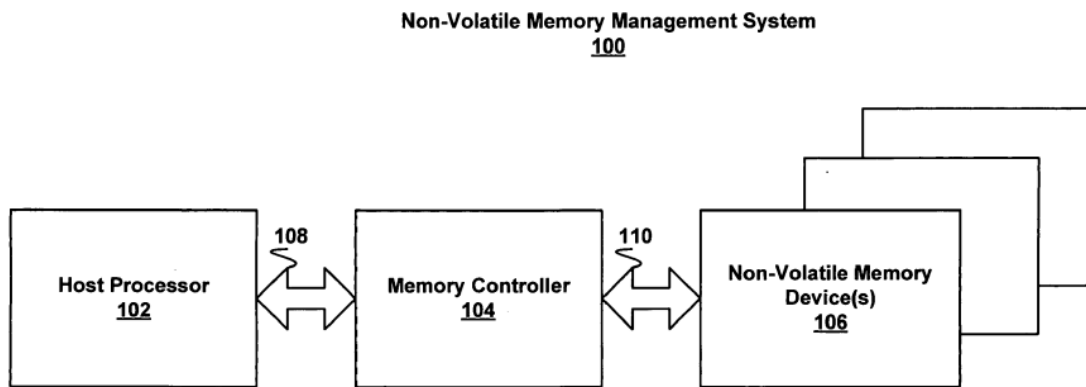


Danilak, Figure 2

2. Cornwell

Cornwell relates to memory management, particularly in the context of non-volatile memory such as flash. (See INTEL-1006, ¶¶2, 5, Title.) Cornwell’s memory management system, illustrated in Figure 1 below, includes host 102, memory controller 104, and one or more non-volatile memory devices 106 such as

“NAND and/or NOR flash media.” (See INTEL-1006, ¶20.) The memory controller “manages memory access” by “recogniz[ing] control, address, and data signals transmitted” by the host and translating these signals into memory access requests on memory devices 106. (INTEL-1006, ¶¶19, 21.)



Cornwell, Figure 1

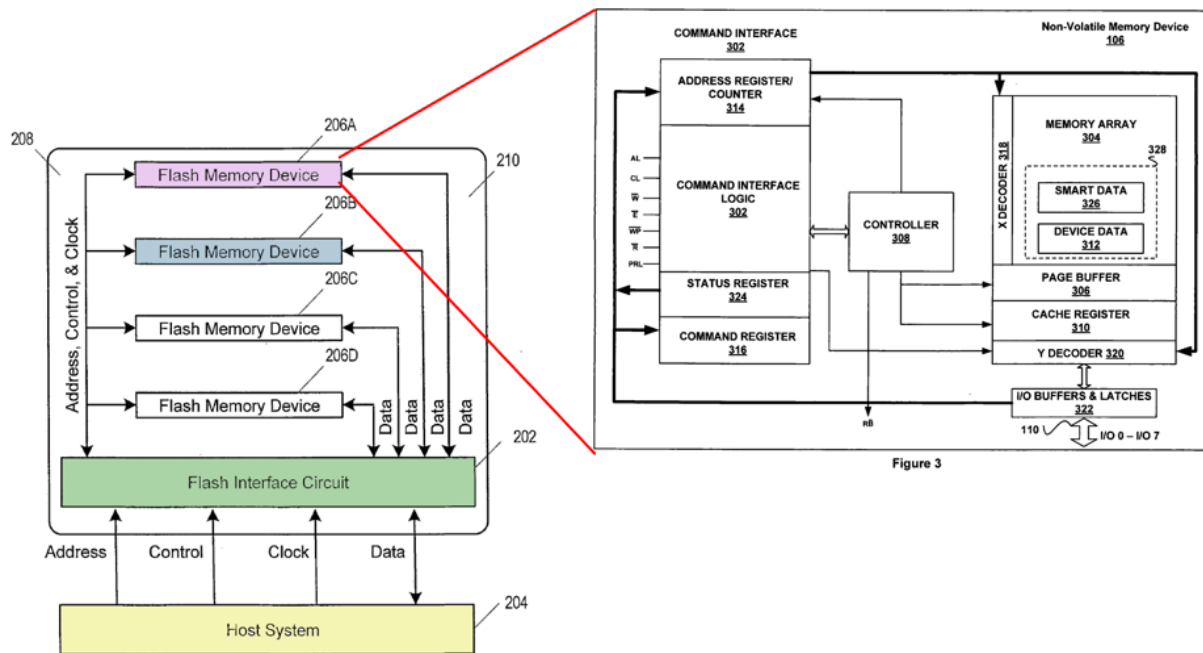
Cornwell’s non-volatile memory device, illustrated in Figure 3 below, includes a memory array, registers for storing received commands and addresses, an address decoder (X-decoder 318), and a controller which carries out operations on the memory array. (See INTEL-1006, ¶¶35-38.) The memory device also includes a spare area 328 in the memory array for storing device data including page size, block size, access time, identification data, and “any other information that is associated with attributes, properties or characteristics of the memory device 106.” (INTEL-1006, ¶42.)

3. Combination of Danilak and Cornwell

A POSITA would have been motivated to combine the teachings of Danilak and the teachings of Cornwell. (INTEL-1003, ¶¶75-86.) Specifically, a POSITA would have been motivated to combine Cornwell's teachings regarding the structure of its non-volatile memory with Danilak's flash memory devices and would have been motivated to use both NAND and NOR flash devices in the combined memory system. (*Id.*)

a. Cornwell's Non-Volatile Memory Structure

A POSITA would have been motivated to combine Cornwell's teachings regarding the structure of non-volatile memory (including the use of device data stored in non-volatile memory) with Danilak's memory devices, as illustrated in the following figure which combines Danilak's Figure 2 with Cornwell's Figure 3. (*See* INTEL-1003, ¶¶76-81.)



Danilak, Annotated Figure 2 (left); Cornwell, Figure 3 (right)

A POSITA would have been motivated to combine Danilak and Cornwell as discussed above for numerous reasons. First, Danilak discloses the use of flash memory devices 206A-206D but provides no details regarding the structure of its flash memory devices. (INTEL-1003, ¶77.) A POSITA would have therefore been motivated to search for references providing these details and would have been led to Cornwell which describes “non-volatile memory management.” (See INTEL-1006, Title; INTEL-1003, ¶77.) Cornwell is in the same field as Danilak and the ’190 patent—memory and memory device management. (INTEL-1003, ¶78.) Danilak relates to “memory, and more particularly to enhanced capacity, performance, flexibility, and reliability in multiple flash memory circuit systems.”

(INTEL-1005, ¶2.) Cornwell similarly “relate[s] to memory management.”

(INTEL-1006, ¶2.) And the claims of the ’190 patent are directed to memory that uses device profiles. (*See, e.g.*, INTEL-1001, claims 1, 29.)

Second, a POSITA would have been motivated to combine the teachings of Cornwell with Danilak to create an operational memory system. (INTEL-1003, ¶79.) For example, Danilak discloses that configuration data for its flash memory devices “is discovered using one or more read operations to the attached flash memory devices.” (INTEL-1005, ¶68.) Danilak however provides no detail regarding where the device-specific data is stored or how the device-specific configuration data is discovered via a read of a flash device. This functionality is taught by Cornwell which describes the storage of device data in a memory device. (*See* INTEL-1006, ¶42.) Danilak also teaches that its memory devices perform read and write operations but provide no details regarding how a non-volatile memory processes these operations. Cornwell describes the general operation of a non-volatile memory device. (*See, e.g.*, INTEL-1006, ¶¶34-43.)

Third, the integration of Cornwell’s non-volatile memory device architecture into Danilak’s system is merely the simple substitution of one known element (Cornwell’s non-volatile memory) for another (Danilak’s flash memory) to obtain predictable results. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 415-21 (2007); INTEL-1003, ¶¶80-81. As discussed in §V.B-E, Danilak contains a memory

device, computer program product, and method which differs from the device and method claimed in the '190 patent simply by the substitution of Cornwell's flash memory device for Danilak's flash memory device. A POSITA would have substituted Danilak's flash memory device for Cornwell's flash memory device as discussed above and the results of the substitution would have been predictable because Cornwell's flash memory devices use standard interfaces and perform flash memory operations in a predictable manner. (INTEL-1003, ¶¶80-81, *citing* INTEL-1007, 1, 91.)

b. NAND and NOR Flash Devices

A POSITA would have also been motivated to use both NAND and NOR Flash devices in the combined memory of Danilak and Cornwell. (*See*, INTEL-1003, ¶¶82-86, 32-40.) Cornwell provides an express teaching that would have led a POSITA to use both NAND and NOR flash devices in the combined memory system, stating that the non-volatile memory devices used in its system “include but are not limited to: **NAND and/or NOR**¹ flash media.” (INTEL-1006, ¶20; INTEL-1003, ¶82.) Cornwell further teaches that its stored device data includes a device type attribute (e.g., **NAND, NOR**, etc.) further indicating an intent to support different types of devices in the same memory. (INTEL-1003, ¶82.)

¹ All emphasis added unless otherwise noted.

Danilak also provides a suggestion to use different types of devices in its memory system. (INTEL-1003, ¶83.) Danilak teaches its “flash interface circuit 102 may provide vendor-specific protocol translation between attached flash memory devices and may present itself to host as a different type of flash, or a different configuration, or as a different vendor’s flash device.” (INTEL-1005, ¶20.) Danilak specifically teaches its flash interface circuit provides support for and translation between NOR and NAND protocols. (*See* INTEL-1005, ¶45.) The support for both protocols and the suggestion that flash memory devices could be either NAND or NOR would have at least suggested to a POSITA the inclusion of both types of memory in a memory system. (INTEL-1003, ¶83.) Additionally, the inclusion of manufacturer and device code by Danilak would have indicated to a POSITA support for different types of devices in the memory system because several vendors offered both NAND and NOR memory devices prior to the ’190 patent. (INTEL-1003, ¶83; *see, e.g.*, INTEL-1011, 2 (noting Toshiba offered both NAND and NOR Flash devices); INTEL-1010, 5 (providing table comparing Micron NAND and NOR Flash devices.)

A POSITA would have been further motivated to use both NAND and NOR memory in a memory system to address a large and growing market in portable devices, particularly those requiring code execution and high-capacity storage, like smartphones. (INTEL-1003, ¶84; *see also*, INTEL-1011, 4 (“When a system, such

as a camera phone, has a requirement for code execution and high capacity data storage, designers may need to consider alternatives and tradeoffs, such as using both types of Flash memory”).) It was well-known that the “strengths of NOR-type Flash memories are fast random-access reads and reliable storage.” (INTEL-1008, 22; *see also*, INTEL-1003, ¶¶33-34.) NOR flash, therefore, works well in any application where fast, reliable random-access reads are needed, like for code execution. (INTEL-1008, 22; INTEL-1003, ¶¶33-34, 36.) In contrast, the “strengths of NAND-type Flash memories are fast writes and high-density arrays”, making it well suited for data storage applications. (INTEL-1008, 23; *see also*, INTEL-1003, ¶¶35-36.) Prior to the ’190 patent, manufacturers, like Toshiba, offered Multi Chip Packages combining both NOR and NAND Flash in a single package. (*See* INTEL-1011, 4; *see also*, INTEL-1003, ¶¶37-38.) Indeed, memory combining multiple flash NOR and NAND with RAM in a single package was “widely” in use in cell phone applications at least 3 years before the ’190 patent. (INTEL-1013, 1.) In 2005, 15% of MCPs available for the handset market incorporated NAND and most retained NOR for code execution including the NEC handset illustrated below which utilized Intel’s NOR flash memory and Toshiba’s NAND flash. (*See* INTEL-1009, 2-3.) Thus, a POSITA would have been motivated and understood how to combine NAND and NOR Flash devices in a

single storage device to take advantage of their individual strengths. (INTEL-1003, ¶84; *see also*, ¶¶32-38.)

A POSITA would have been further motivated to use both NAND and NOR Flash devices to support applications using both small data and larger files. (INTEL-1003, ¶85.) For example, in embedded systems, “the major file operations are the updating or appending of small data, yet most of the capacity is occupied by large files.” (*See, e.g.*, INTEL-1014, 1002.) And in a mobile phone, “small records such as text messages and an address book are updated sporadically, while comparatively large images and media files occupy most of the storage area.” (*Id.*) In such systems, applications that frequently update records having a much smaller size than the page size of a NAND Flash “greatly reduce the utilization of the medium” which in turn deteriorates the life span of the memory. (*Id.*) A POSITA would have been motivated to integrate NAND and NOR Flash devices together for a file system to optimize utilization, durability, and life span of the memory devices. (INTEL-1003, ¶85.)

Moreover, the combination is nothing more than combining prior art elements (existing NAND flash and NOR flash devices in a memory system) according to known methods. (INTEL-1003, ¶86.) Danilak expressly teaches that its flash interface circuit performs protocol translations for its attached memory devices. (*See* INTEL-1005, ¶¶20, 45.) And Cornwell teaches that both NAND and

NOR flash devices are supported in a memory system. (INTEL-1006, ¶20.) The '190 patent also admits that protocols for accessing non-volatile memory devices such as ONFI were known prior to its critical date. (*See*, INTEL-1001, 4:47-5:3.) The results of the combination would have been predictable because the interface protocols were known and existing devices successfully integrated both NAND and NOR flash devices and supported the protocols for both simultaneously. (INTEL-1003, ¶86.)

B. Independent Claim 1

Claim 1 includes a set of 5 method steps [1A]-[1E] recited in the context of a set of structural limitations, 11 “*wherein*” clauses, and 2 “*said*” clauses narrowing a prior limitation. The “*wherein*” and “*said*” clauses can be grouped into two general categories: limitations directed to “read-write” or to “device profile[s].” The following table identifies the categorization for each limitation and the section addressing the limitation. Petitioner first addresses the structural limitations of the claims followed by the “device profile” limitations, method steps, and finally the “read-write” limitations.

Claim 1	Section
[1P] A method of performing memory transactions [structural] on a set of memory locations that includes a first memory location and a second memory location, the method comprising:	§V.B.3.a Structure: §V.B.1.b

Claim 1	Section
[1A/Action] performing a memory transaction comprising addressing [structural] a first memory location in a first memory device in a memory store,	§V.B.3.a Structure: §§V.B.1.a-c
[1A.1/Profile] said first memory location and a second memory location respectively associated with a first device profile and a second device profile,	§V.B.2.a
[1A.2/Structural] wherein said second memory location is in a second memory device in the memory store,	§§V.B.1.a-c
[1A.3/Read-Write] wherein a memory read transaction or a memory write transaction is performed on at least one of the first memory location or second memory location,	§V.B.4.a
[1A.4/Read-Write] wherein data is transferred to the first memory location or second memory location from a host for a memory write transaction,	§V.B.4.b
[1A.5/Read-Write] wherein data is transferred from the first memory location or second memory location to the host for a memory read transaction;	§V.B.4.c
[1A.6/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;	§V.B.2.d
[1B/Action] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;	§V.B.3.b
[1B.1/Profile] said first device profile representing a first set of attributes of said first memory location, and said second device profile representing a second set of	§V.B.2.b

Claim 1	Section
attributes of said second memory location, and a difference exists between said first and second device profiles;	
[1C/Action] after identifying the command details, obtaining the first set of attributes;	§V.B.3.c
[1C.1/Profile] wherein each attribute in the first set of attributes and second set of attributes is associated with a respective attribute qualifier that qualifies a respective memory location	§V.B.2.c
[1C.2/Structural] wherein the memory store is directly coupled to at least one memory bus and wherein the at least one memory bus is directly coupled to a controller, and	§V.B.1.e
[1C.3/Read-Write] wherein the controller performs the memory read transaction and memory write transaction on the memory store	§V.B.4.a
[1C.4/Structural] wherein the host is directly coupled to the controller by a communication path as the controller performs the memory read transaction or memory write transaction	§V.B.1.f
[1D/Action] using attributes from said first and second device profiles; and	§V.B.3.c
[1E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.	§V.B.3.d

1. Structural Limitations

Danilak “relates to memory.” (INTEL-1005, ¶2.) Danilak’s memory system, illustrated in Figure 2 below (left), includes flash memory devices 206A-206D coupled to flash interface circuit 202 (shaded green) which is coupled to host 204 (shaded yellow). (INTEL-1005, ¶22.) As shown below in the side-by-side depictions of Figure 2 and Figure 1 of the ’190 patent (right), Danilak’s system and the ’190 patent’s system have the same basic structure. (INTEL-1003, ¶117.)

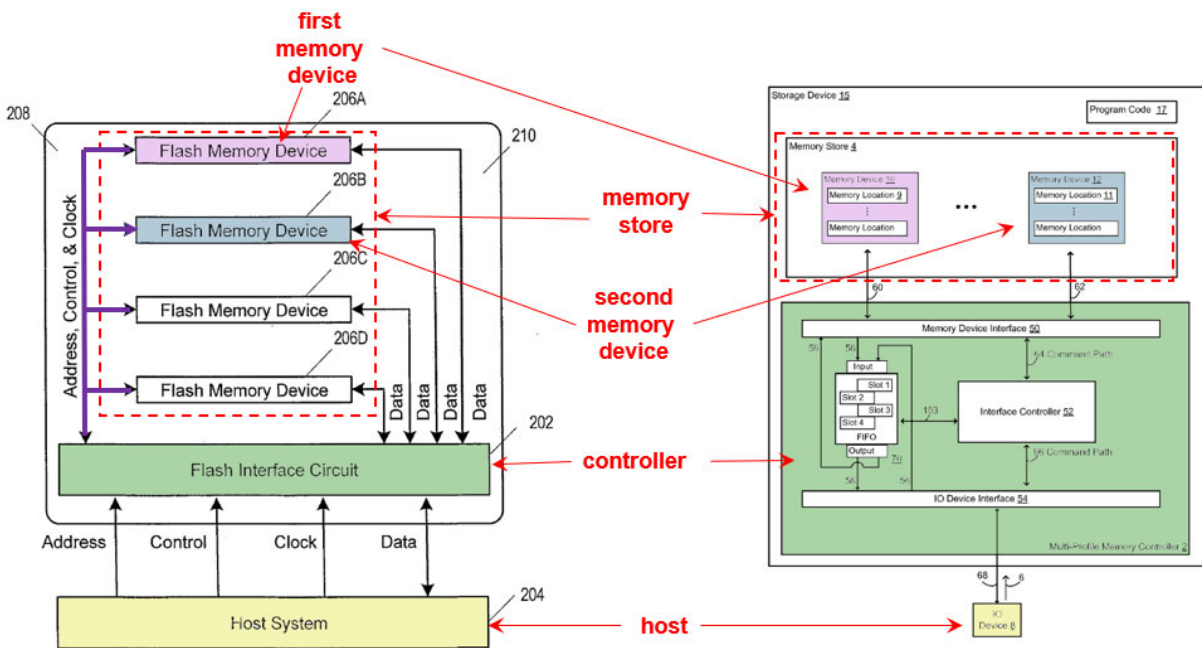


FIG. 1

Danilak, Annotated Figure 2 (left); '190 Patent, Annotated Figure 1 (right)

The combination of Danilak and Cornwell discloses the following structural limitations of claim 1 as discussed in further detail below. (INTEL-1003, ¶¶89-117.)

[1P]	<i>A method of performing memory transactions on a set of memory locations that includes a first memory location and a second memory location ...</i>
[1A]	<i>performing a memory transaction comprising addressing a first memory location in a first memory device in a memory store ...</i>
[1A.2]	<i>wherein a second memory location is in a second memory device in the memory store ...</i>
[1C.2]	<i>wherein the memory store is directly coupled to at least one memory bus and wherein the at least one memory bus is directly coupled to a controller ...</i>
[1C.4]	<i>wherein the host is directly coupled to the controller by a communication path ...</i>

a. “Memory Device”

Danilak’s flash memory devices 206 are the recited “*memory device[s]*” with one being the “*first memory device*” (e.g., flash memory device 206A; shaded pink) and one being the “*second memory device*” (e.g., flash memory device 206B; shaded blue). (INTEL-1003, ¶91; INTEL-1005, ¶¶22, 17-18, Figure 2; *see also*, INTEL-1001, 4:34-38 (describing memory devices can be flash memory devices).)

b. “Memory Locations”

The ’190 patent describes a “memory location” as an addressable portion of a memory device. (*See* INTEL-1001, 2:63; INTEL-1003, ¶92.) Each of Danilak’s flash memory devices includes one or more addressable portions and therefore has multiple “*memory locations.*” (INTEL-1003, ¶¶92-97.) Danilak explains that a “flash memory is typically divided into sub-units, portions, or blocks.” (INTEL-

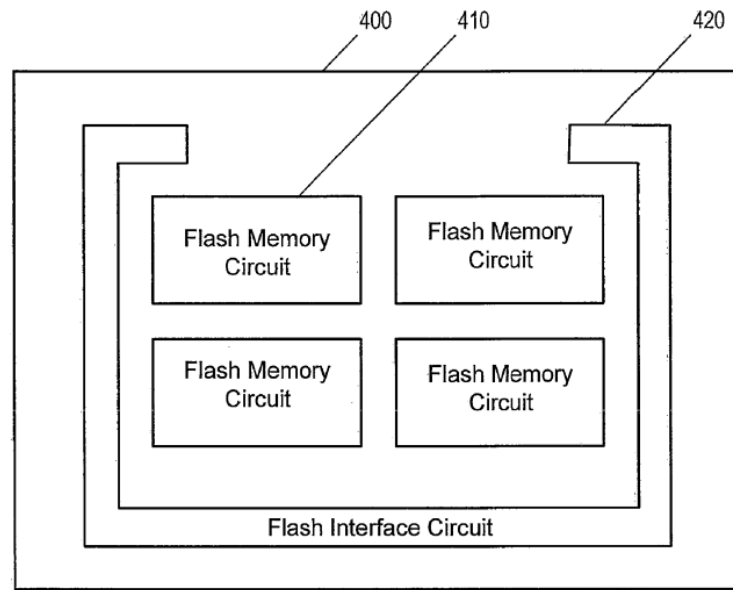
1005, ¶24.) As shown in Figure 2 above, each flash memory device is connected to an address bus over which the address for the memory operation is provided. (INTEL-1003, ¶93, *citing* INTEL-1005, Figure 2; INTEL-1008, 66-67.) Danilak also refers to addressable memory “locations” in its specification. (INTEL-1005, claim 7 (referring to “storage locations in the one or more flash memory devices”); ¶25 (describing accessing “physical memory location”).) Cornwell confirms a flash memory device includes multiple “*memory location[s]*”, teaching that flash memory “can be read or programmed a byte or word at a time in a random-access fashion” but “is usually erased a block at a time.” (INTEL-1006, ¶4.) Indeed, it is inherent that a memory device includes “*memory locations*” because the sole purpose of memory is to store data in an accessible location and provide the stored data upon request. (INTEL-1003, ¶95, *citing* INTEL-1008, 66-67; INTEL-1007, 2.) The ’190 patent also acknowledges this well-known fact stating that NAND flash “includes a set of **addressable non-volatile memory cells**, named flash blocks.” (INTEL-1001, 4:44-45.)

Thus, the combination of Danilak and Cornwell discloses “*a first memory location in a first memory device*” [1A], “*a second memory location is in a second memory device*” [1A.2], and “*a set of memory locations that includes a first memory location and a second memory location*” [1P]. (INTEL-1003, ¶¶92-97.)

c. “Memory Store”

The combination of Danilak and Cornwell discloses a “*memory store*.” (INTEL-1003, ¶¶98-100.) Danilak’s flash memory devices are exposed “as at least one flash memory device” (referred to as a “virtual flash device”) to the rest of the system via flash interface circuit 102. (INTEL-1005, ¶17; *see also*, ¶¶51, 81.) The virtual flash memory device comprised of multiple physical flash memory devices is a “*memory store*.” (INTEL-1003, ¶¶98, 100; INTEL-1001, 2:63 (memory store “includes addressable memory locations”); 10:13-17 (memory store includes memory devices “grouped so that each of the memory devices are associated with only one memory bus controller”); INTEL-1017, 89 (“3.2126 store. (1) A device into which data can be placed, in which they can be retained, and from which they can be retrieved”).) Thus, the combination of Danilak and Cornwell discloses the “*first memory device*” and “*second memory device*” are in “*a memory store*” [1A]/[1A.2]. (INTEL-1003, ¶¶98-100.)

Should PO contend the “*memory store*” must incorporate the memory devices in a single die, Danilak discloses such an embodiment. (INTEL-1003, ¶99.) In Danilak, as illustrated by the embodiment of Figure 4, the flash interface circuit is “integrated with one or more flash devices onto a single monolithic semiconductor die.” (INTEL-1005, ¶54.)



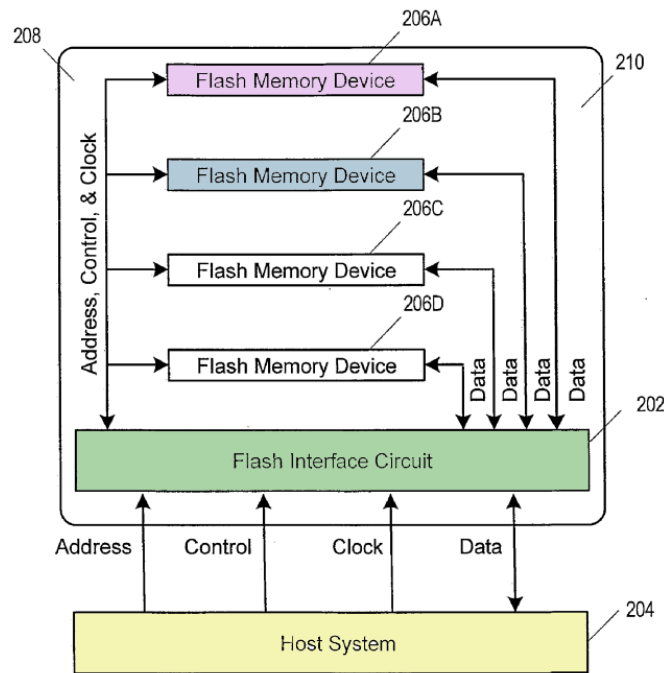
Danilak, Figure 4

d. “Controller”

The flash interface circuit in the combined system of Danilak and Cornwell is a “*controller*.” (INTEL-1003, ¶¶101-112.) A controller in the ’190 patent “manage[s], control[s] or both the flow of data between the IO device 8 and memory store 4.” (INTEL-1001, 2:60-62.) Danilak’s flash interface circuit performs these same functions as well as other memory management functions typically performed by flash memory controllers. (INTEL-1003, ¶¶101-112.)

Flash interface circuit 202² manages and controls access to the flash memory devices by the host. (INTEL-1003, ¶¶102-105.) Specifically, “all signals from the host system are received by the flash interface circuit before presentation to the flash memory devices” and “all signals from the flash memory devices are received by the flash interface circuit before being presented to the host system 204.” (INTEL-1005, ¶22.)

² Danilak’s Figure 1 includes a separate component labeled “flash memory controller” not present in the Figure 2 embodiment. A POSITA would have therefore understood the functions performed by the “flash memory controller” of Figure 1 are performed by the flash interface circuit as discussed throughout Danilak. (INTEL-1003, ¶102; *see also, e.g.*, INTEL-1005, ¶¶29-41.)



Danilak, Annotated Figure 2

The flash interface circuit exposes the set of flash memory devices to the host as a single “virtual flash memory device.” (INTEL-1005, ¶¶17, 51, 80.) As such, the flash interface circuit translates a memory transaction request from the host “into one or more physical flash memory device accesses”, “emulate[s] attributes of the virtual configuration that differ from the physical flash memory devices”, and “return[s] an appropriate response to the request to the host.” (INTEL-1005, ¶78; *see also*, ¶23 (references to “flash interface circuit” include flash interface circuit 202).)

Additionally, Danilak’s flash interface circuit performs functions of a flash driver —making “flash memory appear to the host system as a read/write memory

array” and supporting “basic file system functions (e.g., read, write, file open, file close, etc.) and directory operation (e.g., create, open, close, copy, etc.).” (INTEL-1005, ¶¶29-30.) A POSITA would have been motivated to include flash driver functionality in the flash interface circuit of Figure 2 based on the express teaching of Danilak that “[m]oving some or all of the flash driver functionality into a flash interface circuit enables the use of standard flash devices that do not have integrated flash driver capability and/or standard flash memory controllers that do not have integrated flash driver capability.” (See INTEL-1005, ¶30; INTEL-1003, ¶105.)

The flash interface circuit performs other memory management functions that control flow of data between host and memory devices. (INTEL-1003, ¶106.) For example, Danilak’s flash interface circuit performs functions including increasing erase and modify bandwidth, reducing read latency by prefetching, increasing write bandwidth, and increasing bus bandwidth. (INTEL-1005, ¶¶35-41; INTEL-1003, ¶106.)

The flash interface circuit also performs memory management functions typically performed by a memory controller such as bad block detection, error detection and collection, garbage collection, and wear leveling. (See INTEL-1005, ¶¶24-34; INTEL-1003, ¶¶107-111.) A POSITA would have understood that the flash interface circuit of Figure 2 without a separate flash memory controller

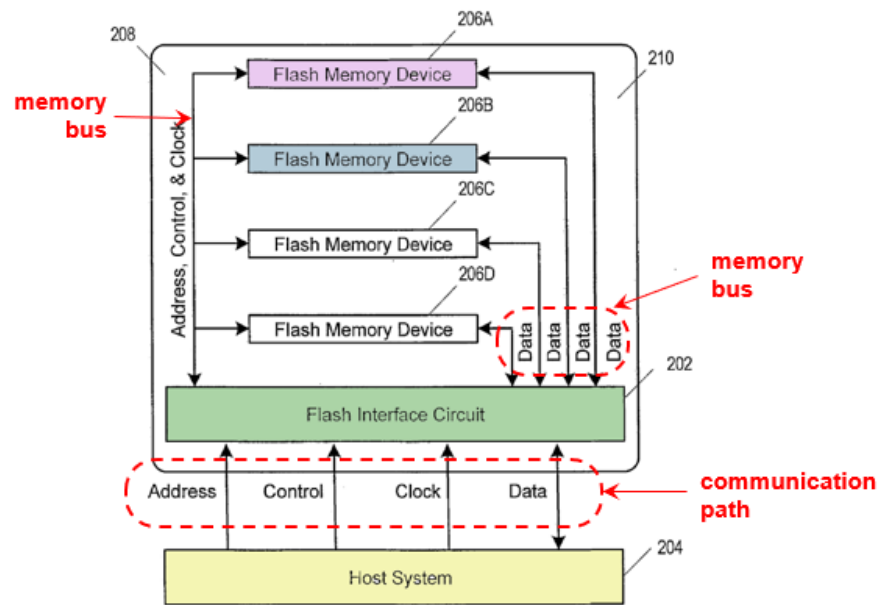
performed these memory management functions. (INTEL-1003, ¶107) Regardless, a POSITA would have been motivated to include each management function in the flash interface circuit based on the express teachings in Danilak. (See INTEL-1005, ¶25 (disclosing the flash interface circuit performs bad block relocation), ¶28 (generating and checking ECC in flash interface circuit “enables the use of MLC flash memory devices in applications designed for the lower error rate of” SLC flash memory devices”), ¶¶32, 34 (performing garbage collection in flash interface devices “enables the use of standard flash devices” that do not have integrated garbage collection capability or integrated wear leveling capabilities); INTEL-1003, ¶¶107-111.) A POSITA would have been further motivated to include these functions in the flash interface control to improve system performance by offloading these functions from the host and to provide flexibility to select various types of memory devices to use while simplifying host software by hiding detailed characteristics of wear-leveling, error-correction, and other functions that require management. (INTEL-1003, ¶111.)

Accordingly, the flash interface circuit in the combination of Danilak and Cornwell is a “*controller*” [1C.2]. (INTEL-1003, ¶¶101-112.)

e. “*Memory Bus*”

As shown in Danilak’s Figure 2 below, flash interface circuit 202 is directly connected to each flash memory device to communicate address, control, and clock

signals (left) and data (right). (INTEL-1005, ¶22.) Although not explicitly labeled a bus in Figure 2, the connection carrying these signals between the flash interface controller and the flash memory devices is a “*memory bus*” in the combined system of Danilak and Cornwell. (INTEL-1003, ¶113, *citing, e.g.*, INTEL-1017, 13 (definition of “bus”), 57 (definition of “memory bus”).) Danilak refers to use of a bus between the flash interface circuit and the memory devices. (*See* INTEL-1005, ¶41 (section on “increasing bus bandwidth” describing 8-bit bus between each memory device and the flash memory circuit), ¶18 (describing “connections of data bus & control signals between flash memory devices 104A-104N and a flash interface circuit 102”), ¶80 (referring to the “buses of the one or more flash memory devices”).) Cornwell confirms that a controller (like Danilak’s flash interface circuit) is connected to a memory device via a bus. (*See* INTEL-1006, ¶24 (“one or more memory devices 106 receive signals from the memory controller 104 over Input/Output (I/O) bus 110”); ¶34 (describing bus used to input address, output data during a read, and input data during a program operation).) Further, the ’190 patent admits it was known to use a memory bus as the connection with a memory device. (*See* INTEL-1001, 6:25-27 (“[t]he means and method of selecting a memory device or addressing a memory location from a set of memory devices by using a single memory bus are known.”))



Danilak, Annotated Figure 2

Accordingly, in the combination of Danilak and Cornwell, the set of flash memory devices (“the memory store”) “is directly coupled to at least one memory bus” and “the at least one memory bus is directly coupled to a controller” (Danilak’s flash memory circuit) [1C.2]. (INTEL-1003, ¶¶113-115.)

f. “Communication Path”

As shown in Figure 2 above, address, control, clock and data signals are communicated over a connection between host 204 and flash interface circuit 202. (See also, INTEL-1005, ¶22; see also, ¶18 (“Also shown are exemplary data bus & control signals between the flash interface circuit 102 and a host system 106”); INTEL-1003, ¶116.) Cornwell teaches the communication path between the host and a controller, like the flash interface circuit of Danilak, is a bus. (See INTEL-

1006, ¶21 (“the memory controller 104 recognizes control, address, and data signals transmitted on bus 108”).) Thus, the combination of Danilak and Cornwell discloses “*the host is directly coupled to the controller by a communication path*” [1C.4]. (INTEL-1003, ¶116.) This direct coupling exists throughout the operation of the memory system. (*Id.*)

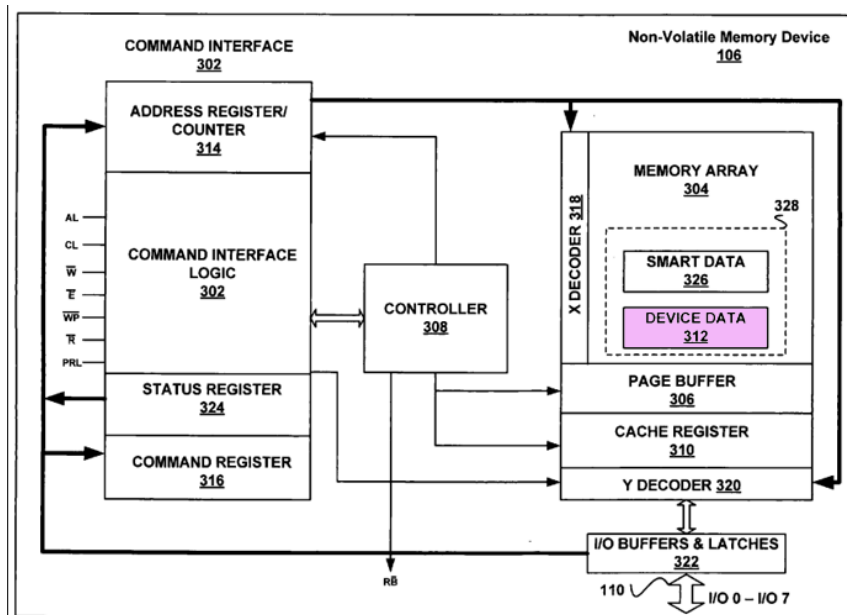
2. “Device Profile” Limitations

Claim 1 further recites a memory location is associated with a “*device profile*.” The combination of Danilak and Cornwell discloses the following device profile limitations claim 1:

[1A.1]	<i>said first memory location and a second memory location respectively associated with a first device profile and a second device profile</i>
[1A.6]	<i>wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type</i>
[1B.1]	<i>said first device profile represents a first set of attributes of said first memory location and said second device profile representing a second set of attributes of said second memory location, and a difference exists between said first and second device profiles</i>
[1C.1]	<i>wherein each attribute in the first set of attributes and second a set of attributes is associated with a respective attribute qualifier that qualifies a respective memory location</i>
[1C.5]	<i>wherein the first device profile and second device profile are each stored in the memory store</i>

a. ***“Device Profile”***

Device data stored in the non-volatile (flash) memory in the combination of Danilak and Cornwell is a “*device profile*.” (INTEL-1003, ¶¶119-122; See §V.A.3.) Cornwell’s device data 312 (shaded pink) is stored in a spare area 328 of the memory array 304. (INTEL-1006, Figure 3 below.) Cornwell’s device data specifies the attributes of the memory device including device identification information, device type, page size, block size, sequential access time, and device density. (See INTEL-1006, ¶42.)



Cornwell, Annotated Figure 3

The stored device data in the combined system further includes configuration data used by Danilak’s flash interface circuit, illustrated in Danilak’s Figure 9 below. (INTEL-1003, ¶120.) Danilak’s configuration data includes a set

of overlapping device data with Cornwell's device data such as manufacturer, device, and chip identification information, page size, device density/plane size, sequential access time, and block size. A POSITA would have been motivated to store the non-overlapping configuration data of Danilak (e.g., cell type, number of simultaneously programmed pages, interleave between multiple chips, cache program) in the device data of a memory device. (INTEL-1003, ¶120.) Cornwell expressly suggests this, teaching device data includes "any other information that is associated with attributes, properties or characteristics of the memory device 106." (INTEL-1006, ¶42.) Danilak also suggests storage of the configuration data in the memory device, disclosing its configuration data is discovered **from** the attached memory devices. (INTEL-1005, ¶59; INTEL-1003, ¶120.)

900

Byte	Bits	Mnemonic	Description
0	0	—	Reserved
	1	ONFI	ONFI standard support (No, Yes)
	3:2	CHIPS	Number of stacked chips (1, 2, 4, 8)
	7:4	—	Reserved
1	8	MKR	External maker code reported by flash interface circuit to host.
2	8	DEV	External device code reported by flash interface circuit to host.
3	1:0	CIPN	Internal chip number (1, 2, 4, 8)
	3:2	CELL	Cell type (2, 4, 8, 16 levels)
	5:4	SIMP	# of simultaneously programmed pages (1, 2, 4, 8)
	6	INTRL	Interleave between multiple chips (No, Yes)
	7	CACHE	Cache program (No, Yes)
4	1:0	PAGE	Page size (1, 2, 4, 8 KB)
	2	RSIZE	Redundant area size (8, 16 bytes / 512 bytes)
	7&3	SMIN	Serial access minimum (50/30, 25 ns)
	5:4	BSIZE	Block size (64, 128, 256, 512 KB)
	6	ORG	Organization (x8, x16)
5	1:0	—	Reserved
	3:2	PLANE	Plane number
	6:4	PLSIZE	Plane size (64, 128, 256, 512, 1Gb, 2Gb, 4Gb, 8Gb)
	7	—	Reserved
6	7:0	—	Reserved

FIG. 9
Danilak, Figure 9

Each non-volatile memory device in Cornwell includes device data 312. This is consistent with Danilak’s disclosure teaching that “multiple instances of the configuration data may be stored in the discovery block.” (INTEL-1005, ¶68.) That is, every memory device has stored device data stored which is discovered by the flash interface circuit and stored locally in a configuration block. (INTEL-1003, ¶121, *citing, e.g.*, INTEL-1007, 91-106 requiring devices store a retrievable data

structure describing “organization, features, timings and other behavioral parameters”).)

The device data describes attributes of the device and applies to each of the device’s memory locations. (INTEL-1003, ¶122; *see also*, INTEL-1006, ¶42.) Therefore, each memory location in a first memory device is associated with the device profile for the first memory device and each memory location in a second memory device is associated with the device profile for the second memory device. (INTEL-1003, ¶122.) The combination of Danilak and Cornwell therefore discloses “*said first memory location and a second memory location respectively associated with a first device profile and a second device profile*” [1A.1]. (INTEL-1003, ¶¶119-122.) Because each device profile is stored in its associated non-volatile memory in the memory store, as disclosed by Cornwell, the combination of Danilak and Cornwell also discloses “*wherein the first device profile and second device profile are each stored in the memory store*” [1C.5]. (*Id.*)

b. “Attributes”

Cornwell refers to the device data stored in the memory device as “**attributes**” of the device: “Chip data 312 can also include but is not limited to: device type (e.g., NAND, NOR, etc.) ... (collectively, **referred to herein as ‘attributes’**).” (INTEL-1006, ¶42.) Each device profile therefore has a “*set of attributes*” that applies to each of its memory locations. The attributes used to

describe the memory devices in the combined memory system of Danilak and Cornwell are consistent with the attributes disclosed in the '190 patent, illustrated in Figure 2A below, which include page size 16, block size 18, memory protocol 20 supported by the memory device in which the memory location is addressable, and memory device type 22. (INTEL-1001, 3:25-29.)

24		25		14
Page Size	<u>16</u>	(# of pages)		
Block Size	<u>18</u>	(# of blocks)		
Memory Protocol	<u>20</u>	(Protocol Type Supported)		
Mem. Dev. Type	<u>22</u>	(Mem. Type)		
Partial Page Size	<u>23</u>	(# of pages)		

'190 Patent, Figure 2A

Accordingly, the combination of Danilak and Cornwell discloses “*said first device profile representing a first set of attributes of said first memory location, and said second device profile representing a second set of attributes of said second memory location*” [1B.1]. (INTEL-1003, ¶¶123-125.)

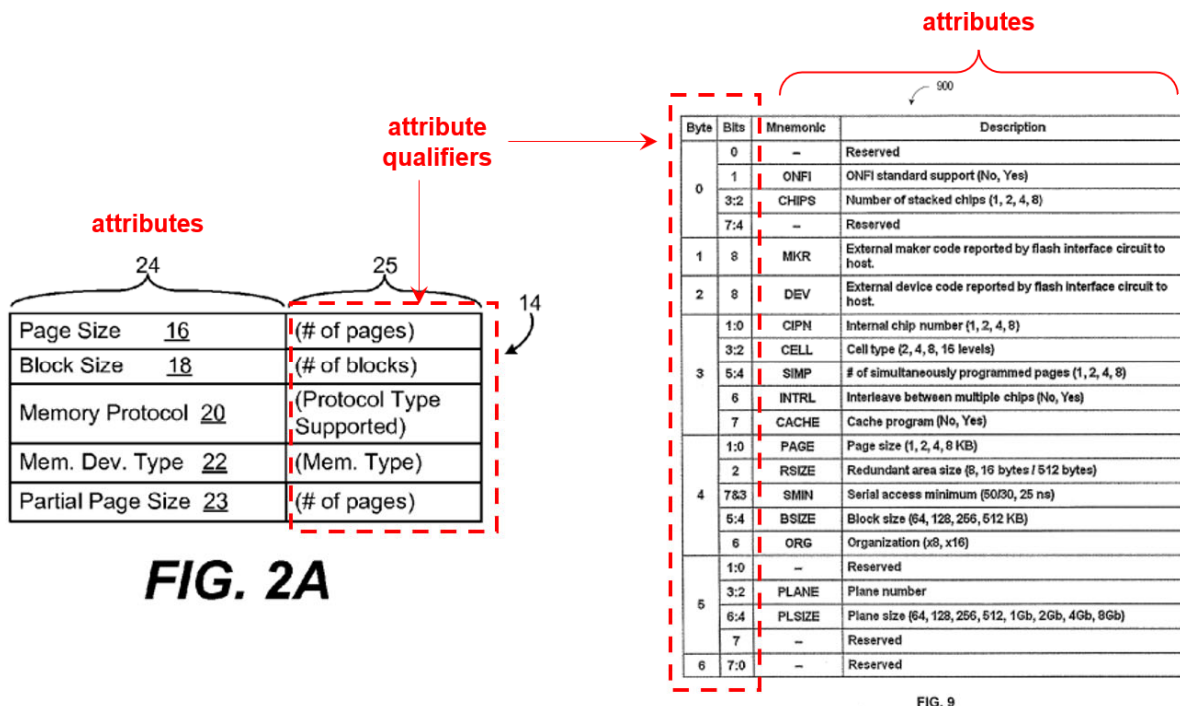
Danilak teaches that “various attached flash memory devices may have differing attributes.” (INTEL-1005, ¶68.) The combined system of Danilak and Cornwell includes both NAND and NOR flash devices. (See §V.A.3.b.) The device profiles for NAND and NOR devices differ because the devices have different values and types for specified attributes (e.g., type (NOR vs NAND), density, page

size, access time, etc.). (INTEL-1003, ¶126, *citing* INTEL-1008, 26 (“Flash memory possesses its own set of attributes that define the product category and each specific product implements those attributes in a slightly different way”); INTEL-1011, 3 (Table 4) (showing difference in density and speed between NOR and NAND Flash); INTEL-1010, 5 (Table 5) (showing difference in speed between NOR and NAND Flash).) Cornwell also teaches, for example, that “[f]lash is available in a variety of block sizes”, indicating that block sizes vary among attached devices. (INTEL-1006, ¶48; INTEL-1003, ¶126.) Other attributes can similarly vary within the same type of Flash including, e.g., number of levels and density. (See INTEL-1011, 3 (showing differences between SLC and MLC NAND Flash).) The combination of Danilak and Cornwell therefore discloses that “*a difference exists between said first and second device profiles*” [1B.1]. (INTEL-1003, ¶¶126-127.)

c. “Attribute Qualifier”

In the ’190 patent, “[e]ach attribute kept in a device profile is associated with data, named ‘attribute qualifier,’ that qualifies the attribute, such as an attribute value or attribute type.” (INTEL-1001, 3:16-18, Figure 2A.) The combination of Danilak and Cornwell discloses certain attributes are associated with numerical values such as page size and block size. (See INTEL-1005, ¶64, Figure 9 below left; INTEL-1006, ¶42.) These attributes have “value” attribute

qualifiers, consistent with the '190 patent. (See INTEL-1001, 3:30-39; INTEL-1003, ¶¶128-129.) Danilak and Cornwell also disclose attributes specified using non-numeric data. These attributes have “type” attribute qualifiers, consistent with the '190 patent. (See INTEL-1001, 3:30-39; INTEL-1003, ¶¶128-129.)



'190 Patent, Annotated Figure 2A (left); Danilak, Annotated Figure 9 (right)

The combination of Danilak and Cornwell therefore discloses “each attribute in the first set of attributes and the second set of attributes is associated with a respective attribute qualifier that qualifies a respective memory location” [1C.1]. (INTEL-1003, ¶¶128-130.)

d. “Device Profile is Optimal ...” [1A.6].

Claim 1 recites the limitation “*said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type*” [1A.6]. The ’190 patent mentions the word “optimal” in a single sentence in the specification but does not disclose what makes a target memory location “optimal” for a specific data type. (See INTEL-1001, 9:7-16; INTEL-1003, ¶131.) In the co-pending district court litigation, PO pointed to disclosures in the specification which imply a large block size is desirable for sequential memory access and a small block size is desirable for random access as examples supporting this claim limitation. (See INTEL-1025, 27-29; INTEL-1003, ¶132; *see also*, INTEL-1024.) The combination of Danilak and Cornwell discloses or at least suggests “*memory device profiles*” that are “*optimal*” for sequential and random data accesses consistent with the examples identified by PO. (INTEL-1003, ¶¶131-140.)

In the combination of Danilak and Cornwell, the memory includes both NAND and NOR flash memory devices. (See §V.A.3.b.) It was well-known that NOR flash is best suited for code storage because NOR flash has fast “random-access” reads. (See INTEL-1003, ¶¶134, 33-34, 36, *citing* INTEL-1008, 22, 38-39; INTEL-1009, 1-2; INTEL-1010, 4; INTEL-1011, 3.) It was also well-known that NAND flash is “strongly suited” to data storage applications because, although

NAND flash “has poor random-access read performance”, its “sequential read performance is good” and it has superior write speed. (INTEL-1003, ¶¶135, 35-36, *citing* INTEL-1008, 23; INTEL-1009, 1-3; INTEL-1010, 3, 5; INTEL-1011, 3).)

A POSITA would have been further motivated to incorporate a memory management function that (1) selects NOR flash when a transaction is received requiring writing or reading code because NOR flash is “*optimal*” for code execution which requires fast random-access reads and (2) selects NAND flash when a transaction is received for writing and reading data (e.g., audio or image storage) which requires faster sequential write and read accesses. (INTEL-1003, ¶137.) A POSITA would have understood that the request from the host included an indication of the data type associated with the request (e.g., code or data) or alternatively would have been motivated to include such an indication to achieve the optimal and desired performance of the dual NAND/NOR memory system. (*Id.*) Microprocessor buses in existence more than a decade before the ’190 patent included such an indication. (*See, e.g.*, INTEL-1019, 4-5 (identifying function code assignments of program or data); INTEL-1020, 3-14 (indicating “data or code”).) A POSITA would have been motivated to include this selection of memory device type to optimize application and device performance. (INTEL-1003, ¶137.) Indeed, mobile phones existing prior to the ’190 patent used both NOR and NAND flash to

achieve the optimizations offered by both in a single system, allowing the vendor “to get the best of both worlds.” (INTEL-1003, ¶137, *citing* INTEL-1009, 3.)

Accordingly, the combined memory of Danilak and Cornwell includes a “*device profile*” [NOR device type] that “*is optimal for a data type subject to the memory transaction* [e.g., reading or writing code]” where the “*data type*” is “*a random data type*” and a “*device profile*” [NAND device type] that “*is optimal for a data type subject to the memory transaction* [e.g., writing data]” where the “*data type*” is a “*sequential data type*.” (INTEL-1003, ¶¶133-138.)

It was further known that the “ideal block size depends on the nature of the application.” (INTEL-1016, 8.) For example, large blocks and accordingly large page sizes³ are “preferable for applications that do long sequential reads.” (*Id.*) The ’190 patent acknowledges this known fact, admitting in the background section “[f]lash memory devices that have a relatively large block size generally provide better sequential memory access performance than flash memory devices that have a relatively small block size.” (INTEL-1001, 1:27-31.) Consistent with this knowledge, Cornwell teaches “[f]lash is available in a variety of block sizes” and

³ Page size is the relevant parameter for reading data, such as code, and block size is the relevant parameter for erasing data. (INTEL-1003, ¶139, *See, e.g.*, INTEL-1007, 2-3.)

both Cornwell and Danilak provide “block size” and “page size” as configurable attributes of a memory device. (INTEL-1006, ¶48, *see also*, ¶42; INTEL-1005, ¶59, Figure 9.) For example, NAND flash was available with small block sizes and large block sizes. (INTEL-1003, ¶139.) Accordingly, based on the teachings of Cornwell and knowledge of a POSITA, a POSITA would have been motivated to use a flash memory with a large block size to optimize sequential access (e.g., to select a NAND flash memory with large block size over a NAND flash memory with small block). (INTEL-1003, ¶139.)

Accordingly, the combined memory of Danilak and Cornwell includes a “*device profile*” that “*is optimal [e.g., has large block size/page size] for a data type subject to the memory transaction*” where the “*data type*” is a “*sequential data type*.” (INTEL-1003, ¶¶131-140.)

3. Preamble and Method Steps

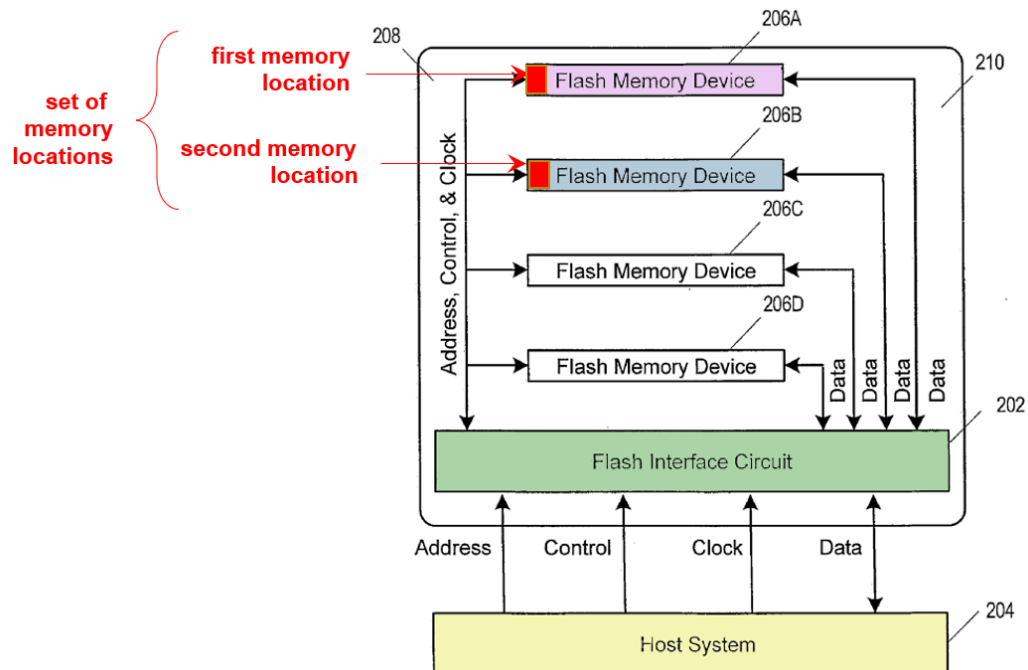
a. Preamble [1P]/ “Addressing a First Memory Location ...” [1A].

The combination of Danilak and Cornwell discloses “*performing memory transactions*” [1P]. (INTEL-1003, ¶¶141-146, *citing* (INTEL-1018, 582 (describing a transaction).) In the combination of Danilak and Cornwell, the flash interface circuit (“*controller*”) receives a memory transaction request (e.g., read or write request) from the host formatted according to a defined protocol such as IDE/ATA. (INTEL-1005, ¶78; *see also*, e.g., INTEL-1006, ¶¶25-27.) The memory

transaction request from the host specifies the transaction type (e.g., read or write) and a virtual address on which the transaction should be performed. (INTEL-1003, ¶141, *citing, e.g.*, INTEL-1007, 2-3, 111-112, 117-118.) For write transactions, the request also includes the data to be written to the specified address. (INTEL-1003, ¶141, *citing, e.g.*, INTEL-1007, 117-118.) The flash interface circuit performs the requested transaction by “translat[ing] the request into one or more physical flash memory device accesses.” (INTEL-1005, ¶78.) The types of physical flash memory device accesses performed by Danilak’s flash interface circuit include read, program (write), and erase operations. (*See, e.g.*, INTEL-1005, ¶¶33-40; INTEL-1001, 5:9-11 (indicating that a read or write transaction is a memory transaction).) Cornwell similarly teaches that its memory controller “includes circuitry for enabling read, program and erase operations to one or more memory devices 106.” (INTEL-1006, ¶31.) Additional discussion of read and write transactions in the combination of Danilak and Cornwell is provided in §V.B.4.

Each operation required to perform the “*memory transaction*” requested by the host is performed on a “*memory location*” in a physical memory device. (INTEL-1003, ¶¶143-144.) As discussed in §V.B.1.b, each memory device in the memory store includes multiple addressable “*memory locations*” including “*a first memory location and a second memory location*” as illustrated in annotated Figure 2 below. In Danilak, while performing the requested memory transaction, the flash

interface circuit translates the received address “to an address to be transmitted to the attached flash devices.” (INTEL-1005, ¶72; *see also*, ¶68.) The address transmitted to the flash memory device indicates the physical memory location on which the requested transaction is to be performed. (INTEL-1003, ¶143; INTEL-1006, ¶35 (an X-decoder in a non-volatile memory “decodes input addresses” received over the bus “to determine a memory line to be accessed for the read or write operation”).)



Danilak, Annotated Figure 2

Moreover, it would have been well-known to a POSITA that a memory system performed memory transactions by issuing read or write (program) operations to a memory location—a fact admitted by the '190 patent. (INTEL-

1003, ¶145; INTEL-1001, 4:47-50 (“A flash memory device permits memory operations, such as a write or read operation, to be performed on these flash blocks according to a protocol supported by the flash memory device”).) The ONFI specification, referenced in the ’190 patent, also discloses that read and program (write) operations are made to an addressable memory location. (See e.g., INTEL-1007, 3 (“page: The smallest addressable unit for read and program operations”), 2 (“The row address identifies the page, block, and LUN to be accessed. The column address identifies the byte or word within a page to access”).)

The combination of Danilak and Cornwell discloses “*performing memory transactions on a set of memory locations that includes a first memory location and a second memory location*” [1P]. (INTEL-1003, ¶¶141-146.) A memory location, such as the “*first memory location in a first memory device in a memory store*”, is an addressable portion of memory on which an operation (e.g., read or write) is performed. (See §V.B.1.b.) Accordingly, the combination of Danilak and Cornwell also discloses “*addressing a first memory location in a first memory device in a memory store*” [1A] during the performance of a memory transaction. (INTEL-1003, ¶¶148-152; see also, ¶¶141-146.)

b. “Identifying Command Details ...” [1B].

The combination of Danilak and Cornwell discloses that “performing the memory transaction” also includes “*identifying command details for causing the*

memory transaction to be performed, wherein said command details comprise the first memory device” [1B]. (INTEL-1003, ¶¶153-156.)

The ’190 patent describes “command details” as the “details necessary to cause the interpreted memory transaction to be performed by multi-profile memory controller 2 through its components.” (INTEL-1001, 7:60-63.) At a minimum, the command details required for an operation on a memory device are the command type (read or write) and address. (See INTEL-1006, ¶¶34-38; INTEL-1007, 111-112 (read), 117-118 (program (write)).) Danilak’s flash memory circuit (“*controller*”) identifies the command details required to perform the memory transaction requested by a host. (INTEL-1003, ¶¶154-155.)

In Danilak, during a memory transaction, translation units 508/509 in the flash interface circuit “translate host flash memory access and configuration requests into requests to one or more flash memory devices.” (INTEL-1005, ¶56.) To perform this translation, the flash interface circuit identifies the address provided by the host. The address translation unit then translates this address into an internal address. (INTEL-1005, ¶71.) This internal address includes, among other fields, “a device field identifying the addressed flash memory device.” (INTEL-1005, ¶70.) The internal address is subsequently translated by the flash interface circuit “to an address to be transmitted to the attached flash devices.” (INTEL-1005, ¶72.) Therefore, the received address and its translated internal

address are both command details necessary to cause flash interface circuit to perform the requested memory transaction. (INTEL-1003, ¶155.)

Accordingly, the combination of Danilak and Cornwell discloses that “performing the memory transaction” includes “*identifying command details for causing the memory transaction to be performed, wherein said command details comprise the first memory device [device field of internal address]*” [1B]. (INTEL-1003, ¶¶153-156.)

c. “*Obtaining the First Set of Attributes*” [1C] and “*Using Attributes From Said First and Second Device Profiles*” [1D].

The combination of Danilak and Cornwell discloses “*after identifying the command details, obtaining the first set of attributes*” [1C] and “*using attributes from said first and second device profiles*” [1D]. (INTEL-1003, ¶¶157-163.)

In Danilak, the flash interface circuit “expos[es] the virtual configuration” of the memory store “to the host rather than the physical configuration.” (INTEL-1005, ¶77.) Therefore, when performing a memory transaction, the flash interface circuit must modify the transaction request “provided from the host system based on differences between the virtual configuration presented by the interface circuit [] to the host system and the physical configuration of the flash memory devices.” (INTEL-1005, ¶56.)

Danilak describes an example of a translation required based on differences between the virtual and physical configuration of the memory devices. In this example, “the configured page size” for the virtual memory reported to the host “is larger than the discovered page size[s]” for the physical memory devices. (*See* INTEL-1005, ¶73.) This difference results in a memory transaction request from a host specifying a data size consistent with the virtual page size but larger than can be handled by a single memory device. (INTEL-1003, ¶160.) To account for this difference, Danilak’s flash interface circuit is configured to translate a host write request with a large data size into multiple write operations of smaller sizes. (*Id.*) As explained by Danilak, in this circumstance, the memory transaction requested by the host is “performed to multiple flash memory devices to form a page of the configured size.” (INTEL-1005, ¶73.) That is, the flash interface circuit identifies the page sizes for available flash memory devices and selects two or more memory devices having page sizes which sum to the data size of the memory write transaction. (INTEL-1003, ¶160.)

To identify the page sizes for the available flash memory devices during this translation, the translation units must obtain attributes for the physical memory devices. These attributes, discovered from the memory devices, are stored locally in the configuration files at the flash interface circuit. (*See* §§V.B.2.b, V.A.1.) Because the virtual address received from the host is associated with the “*first*

memory device” by the translated internal address (“*command detail*”), the attributes obtained by the translation unit include the “*first set of attributes*” associated with the first memory device. (INTEL-1003, ¶161.) Thus, Danilak teaches or suggests “*obtaining the first set of attributes*” [1C]. (INTEL-1003, ¶¶158-162.)

However, because the page size associated with the “*first memory device*” (and all individual memory devices) is less than the size needed to complete the memory write transaction, the translation units must obtain additional device profiles “to form a page of the configured size.” (INTEL-1005, ¶73.) For example, if the memory write request specified a data size of 4K and the first memory device has a page size of 2k, the translation units must identify one or more additional memory devices with page sizes summing to 2k. (INTEL-1003, ¶162.) To do this, the translation units obtain the attributes of additional memory devices. (*Id.*) These attributes are then used together with the attributes of the identified “*first memory device*” to select the set of memory devices having page sizes summing to the needed size to perform the requested memory transaction. (*Id.*)

Accordingly, Danilak teaches or suggests the flash interface circuit “*us[es] attributes [e.g., page size] from said first and second device profiles*” when processing the memory transaction. (INTEL-1003, ¶¶157-163.)

d. “Selecting a Transfer Size for the Memory Transaction” [1E].

The combination of Danilak and Cornwell discloses “*selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes*” [1E]. (INTEL-1003, ¶¶164-167.) A host memory transaction has a “data size” which is the amount of data to be read or written by the transaction. (INTEL-1003, ¶164.) This host memory transaction is translated by a memory controller into one or more operations sent to the physical memory devices. (*Id.*) The amount of data read or written by an operation from/to an individual memory device is the “transfer size.” (*Id.*) It was well-known that the transfer size of a memory operation to a memory device is dictated by the size of the largest addressable unit for that device (e.g., page size for flash memory). (*Id.*) It was also well-known that the transfer size of a memory operation is a function of the data size of the transaction requested (e.g., transfer size is either equal to or a fraction of the data size). (*Id.*) Danilak teaches these well-known relationships.

As Danilak explains, in the circumstance where the virtual page size (and hence the data size) is larger than the discovered page size, “the memory request may be performed to multiple flash memory devices to form a page of the configured size.” (INTEL-1005, ¶73; INTEL-1003, ¶165.) That is, in Danilak’s size translation example, the data size for the requested memory transaction is

larger than the transfer size any individual memory device can accept in a single operation, requiring multiple memory operations to be performed. (See INTEL-1005, ¶73.) Danilak's flash interface circuit therefore selects a set of available memory devices to perform the requested transaction based on their page sizes (transfer size) and the data size of the transaction. (INTEL-1003, ¶165.) Therefore, the transfer size of each individual memory operation needed to perform the requested memory transaction is a function of the page size of the memory device (e.g., "*first set of attributes*") and the "*data size of the memory transaction*" (e.g., a fraction of the data size of the transaction requested by the host). (*Id.*)

The size translation function performed by Danilak's flash interface circuit is consistent with the example of transfer size selection described in the '190 patent. (INTEL-1003, ¶166.) In the '190 patent's example, "if the data size is greater than the page size attribute value in device profile 26, the multi-profile memory controller 2 uses a transfer size equal to at least the page size attribute value for each memory operation transfer for the memory transaction requested until the remaining data not yet transferred from a requesting IO device is less than the page size attribute." (INTEL-1001, 8:51-57.)

Accordingly, the combination of Danilak and Cornwell therefore discloses "*selecting a transfer size for the memory transaction, wherein the transfer size is a*

function of a data size of the memory transaction and the first set of attributes”

[1E]. (INTEL-1003, ¶¶164-167.)

4. Read-Write Limitations

Claim 1 recites the following limitations directed to well-known aspects of memory read and write transactions/operations. Indeed, the '190 patent itself admits these were known, stating “[a] flash memory device permits memory operations, such as a write or read operation, to be performed on these flash blocks according to a protocol supported by the flash memory device.” (INTEL-1001, 4:47-50.) The combination of Danilak and Cornwell also discloses these limitations. (INTEL-1003, ¶¶169-184.)

[1A.3]	<i>wherein a memory read transaction or a memory write transaction is performed on at least one of the first memory location or second memory location</i>
[1A.4]	<i>wherein data is transferred to the first memory location or second memory location from a host for a memory write transaction</i>
[1A.5]	<i>wherein data is transferred from the first memory location or second memory location to the host for a memory read transaction</i>
[1C.3]	<i>wherein the controller performs the memory read transaction and memory write transaction on the memory store</i>
[1C.4]	<i>wherein the host is directly coupled to the controller by a communication path as the controller performs the memory read transaction or memory write transaction</i>

a. *Controller Performs the Memory Read or Memory Write Transaction on a Memory Location.*

As discussed in §V.B.3.a, the combination of Danilak and Cornwell discloses “*performing memory transactions on a set of memory locations.*” Both Danilak and Cornwell disclose the memory transactions performed by the flash interface circuit (controller) include read and write transactions. For example, Danilak’s flash interface circuit performs the function of a flash driver, including “mak[ing] the flash memory appear to the host system as a **read/write** memory array” and “support[ing] basic file system functions (e.g., **read**, **write**, file open, file close, etc.).” (INTEL-1005, ¶29.) Danilak also repeatedly references read and write operations in its discussion of the memory management functions performed by the flash interface circuit. (*See, e.g.*, INTEL-1005, ¶¶31, 33, 35; INTEL-1003, ¶170.) Cornwell similarly discloses that a memory controller performs read and write operations, explaining “one or more memory devices 106 receive signals from the memory controller 104 over Input/Output (I/O) bus 110, which enables the memory devices 106 to perform memory access requests (e.g., **read or write operations**).” (INTEL-1006, ¶24.)

For the reasons discussed in §V.B.3.a, read and write transactions are each performed on a “*memory location*” by the flash interface circuit (“*controller*”).)

Thus, the combination of Danilak and Cornwell discloses that “*a memory read transaction or a memory write transaction is performed on at least one of the first memory location or second memory location*” [1A.3] and “*the controller performs the memory read transaction and memory write transaction on the memory store*” [1C.3]. (INTEL-1003, ¶¶170-173.)

In Danilak, “all signals from the host system are received by the flash interface circuit before presentation to the flash memory devices” and “all signals from the flash memory devices are received by the flash interface circuit before being presented to the host system 204.” (INTEL-1005, ¶22.) The signals between the host and flash interface circuit (“*controller*”) are carried over a “*communication path*” as shown in Figure 2. (See also, §V.B.1.f.) When the host requests a memory access, the flash interface circuit “return[s] an appropriate response to the request to the host” after the transaction is performed. (INTEL-1005, ¶78.) Thus, the combination of Danilak and Cornwell discloses “*the host is directly coupled to the controller by a communication path as the controller performs the memory read transaction or memory write transaction*” [1C.4]. (INTEL-1003, ¶174.)

b. Write Data Flow Plath

Both Danilak and Cornwell disclose the well-known write data flow recited in claim 1: “*data is transferred to the first memory location or second memory*

location from a host for a memory write transaction” [1A.4]. (INTEL-1003, ¶¶175-179.) For example, in Danilak’s technique for improving bandwidth, the flash interface circuit holds write requests received from the host in local memory “and write[s] them into the multiple flash memory chips in an interleaved fashion (e.g., alternating, etc.).” (INTEL-1005, ¶40.) Cornwell similarly discloses that the host sends a program (write) request to a memory controller using a specified protocol (e.g., ATA). (See INTEL-1006, ¶¶25-27.) The memory controller then sends a page program command to the non-volatile memory which “programs the page into the memory array 304 at the address” provided. (INTEL-1006, ¶39.) The ONFI specification further confirms this well-known write data flow specifying its “[d]ata input cycle timing describes the timing of data transfers from the host to the device (i.e. data writes).” (INTEL-1007, 69.)

Danilak and Cornwell therefore both disclose “*data is transferred to the first memory location or second memory location from a host for a memory write transaction*” [1A.4]. (INTEL-1003, ¶¶175-179.)

c. Read Data Flow Path

Both Danilak and Cornwell disclose the well-known read data flow recited in claim 1: “*data is transferred from the first memory location or second memory location to the host for a memory read transaction*” [1A.5]. (INTEL-1003, ¶¶180-184.) In its prefetch technique, Danilak’s flash interface circuit “read[s] data from

the flash memory ahead of a request by the system.” (INTEL-1005, ¶38.) The prefetched data is stored and when a read request is received for the prefetched data, the prefetched data is “returned by the flash interface circuit to the system.” (INTEL-1005, ¶39.) Cornwell similarly discloses that the host sends a read request to a memory controller which sends a page read command to the non-volatile memory which in turn reads data out to the memory controller. (INTEL-1006, ¶¶25-27, 40.) The ONFI specification further confirms this well-known read data flow defining a “read request” as a “request from the host that results in a data transfer from the device to the host.” (INTEL-1007, 3.)

Danilak and Cornwell therefore both disclose “*data is transferred from the first memory location or second memory location to the host for a memory read transaction*” [1A.5]. (INTEL-1003, ¶¶180-184.)

C. Independent Claim 29

Independent claim 29 is directed to a “*computer readable medium comprising computer executable instructions*” which “*cause a method of performing memory transactions*” to occur. The limitations recited in claim 29 are substantially identical to limitations in claim 1. The primary difference between the claims is the preamble. For the reasons discussed below in §V.C.1 and in §V.B, the combination of Danilak and Cornwell renders claim 29 obvious.

1. Preamble [29P]

The combination of Danilak and Cornwell discloses a “*non-transitory computer readable medium comprising executable instructions adapted to cause a method of performing memory transactions on a first memory location and a second memory location*” [29P]. (INTEL-1003, ¶¶187-189.)

The combination of Danilak and Cornwell discloses “*a method of performing memory transactions ...*” as discussed in §V.B.3.a. Translations units 508/509 of Danilak’s flash interface circuit perform the recited “action” steps of the method claims [29A]-[29E] as discussed in §§V.B.3.b-V.B.3.d.

Danilak discloses translation units 508/509 “may be implemented in any desired fashion (combinatorial logic circuitry, pipelined circuitry, **processor-based software**, state machines, various other circuitry, and/or any combination of the foregoing).” (INTEL-1005, ¶55.) Because a flash device powers on in a ‘read’ state, it was well-known that Flash devices required software to place the memory devices into another state such as program or erase. (See INTEL-1008, 32; INTEL-1003, ¶188.) A POSITA would have therefore been motivated to implement the translation units and other functions performed by the flash interface circuit using “processor-based software” as taught by Danilak. (INTEL-1003, ¶188.)

Additionally, a POSITA would have been motivated to use software to implement translation units 508/509 to permit updates to management functions

performed by the flash interface controller. (*Id.*) A POSITA would have understood that using software to implement some of the flash management functions provided a strong advantage compared to a hardware-only implementation, because the algorithms could be changed to accommodate differences between flash vendors, technology evolution to denser chips, and characteristics for wear leveling and error correcting. (*Id.*) Cornwell further confirms that it was known to implement memory controller functions using software, teaching its controller is implemented with “firmware.” (INTEL-1006, ¶29.)

A POSITA would have understood that “processor-based software” is stored on “*non-transitory computer readable medium*” (e.g., stored in memory while the software is running) and therefore comprises “*executable instructions.*” (INTEL-1003, ¶189, *citing* INTEL-1022, 20.)

Accordingly, the combination of Danilak and Cornwell discloses the preamble. (INTEL-1003, ¶¶187-189.)

2. Method Limitations

The following table identifies claim 1 limitations corresponding to claim 29 limitations and identifies the differences between the claims. The primary difference between the claims is in limitations [1A] and [29A]. Limitation [1A] recites that the “*first memory location*” is “*in a first memory device in a memory*

store.” Limitation [29A] does not include this language, making limitation [29A] broader. Addressing a first memory location in a first memory device in a memory store meets the limitation of “*addressing a first memory location.*” (INTEL-1003, ¶191.) The remaining differences between the limitations are non-material.

Claim 1	Claim 29	
[1A/Action] performing a memory transaction comprising addressing a first memory location in a first memory device in a memory store,	[29A/Action] performing a memory transaction comprising addressing a first memory location,	§V.B.3.a, INTEL-1003, ¶¶191, 141-146 §V.B.1.a-c INTEL-1003, ¶¶191, 91-100
[1A.1/Profile] said first memory location and a second memory location respectively associated with a first device profile and a second device profile,	[29A.1/Profile] said first memory location and a second memory location <u>are</u> respectively associated with a first device profile and a second device profile,	§V.B.2.a, INTEL-1003, ¶¶190, 119-122 §V.B.1.a-b INTEL-1003, ¶¶190, 91-97
[1A.6/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;	[29A.2/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;	§V.B.2.d INTEL-1003, ¶¶190, 131-140
[1B/Action] said performing the memory transaction further	[29B/Action] said performing the memory transaction further	§V.B.3.b INTEL-1003, ¶¶191, 153-156

comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;	comprising identifying command details for causing the memory transaction to be performed	
[1B.1/Profile] said first device profile representing a first set of attributes of said first memory location, and said second device profile representing a second set of attributes of said second memory location, and a difference exists between said first and second device profiles;	[29B.1/Profile] said first device profile representing a first set of attributes of said first memory device, and said second device profile representing a second set of attributes of said second memory device, and a difference exists between said first and second device profiles;	§V.B.2.b INTEL-1003, ¶¶190, 123-127
[1C/Action] after identifying the command details, obtaining the first set of attributes;	[29C/Action] after identifying the command details, obtaining the first set of attributes;	§V.B.3.c INTEL-1003, ¶¶191, 157-162
[1D/Action] using attributes from said first and second device profiles; and	[29D/Action] using attributes from said first and second device profiles; and	§V.B.3.c INTEL-1003, ¶¶191, 157-163
[1E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory	[29E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory	§V.B.3.d INTEL-1003, ¶¶191, 164-167

transaction and the first set of attributes.	transaction and the first set of attributes.	
--	--	--

D. Independent Claim 100

Claim 100 recites overlapping subject matter with claim 1, as shown in the table below. There are 6 differences between the claims, each of which makes claim 100 broader than claim 1, as highlighted by the underlines in the table.

- The preamble of claim 100 only recites a “method performing memory transactions”, making it broader than claim 1.
- Limitation [1B.1] recites a “difference exists between said first and second device profiles” which is not included in limitation [100B.1], making limitation [100B.1] broader.
- Limitation [1C.2] recites the memory store is coupled to a memory bus which is coupled to the controller, whereas limitation [100C.1] recites the memory store is coupled to the memory controller. Limitation [1C.2] meets limitation [100C.1] because the memory store is directly coupled to the memory controller by the bus.
- Limitations [1C.3] and [1C.4] recite “memory read transaction [and/or] memory write transaction” whereas limitations [100C.2] and [100C.3] recite broadly “the memory transaction.” Read and Write are memory transactions and therefore meet limitations [100C.2] and [100C.3].

- Limitation [1C.4] recites that the host is coupled to the controller by a communication path, and [100C.3] does not recite a communication path, making [100C.3] broader.
- Claim 100 recites a “memory controller” whereas claim 1 recites “a controller.” These are equivalent terms.

(INTEL-1003, ¶¶237-240.) For the reasons discussed in the sections cited in the table, the combination of Danilak and Cornwell also teaches or suggests the preamble and limitations of claim 100.

Claim 1	Claim 100	
[1P] A method of performing memory transactions <u>on a set of memory locations that includes a first memory location and a second memory location</u> , the method comprising:	[100P] A method for performing memory transactions, the method comprising	§V.B.2.a INTEL-1003, ¶¶236, 141-146
[1A/Action] performing a memory transaction comprising addressing a first memory location in a first memory device in a memory store,	[100A/Action] performing a memory transaction comprising addressing a first memory location in a first memory device in a memory store,	§V.B.3.a INTEL-1003, ¶¶239-240, 141-146, 148-152 §V.B.1.a-c INTEL-1003, ¶¶237, 91-100
[1A.1/Profile] <u>said</u> first memory location and a second memory location respectively associated with a first device profile and a	[100A.1/Profile] <u>the</u> first memory location and a second memory location respectively associated with a first device profile	§V.B.2.a, INTEL-1003, ¶¶238, 119-122

Claim 1	Claim 100	
second device profile,	and a second device profile;	
[1A.6/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;	[100A.2/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type	§V.B.2.d INTEL-1003, ¶¶238, 131-140
[1B/Action] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;	[100B/Action] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;	§V.B.3.b INTEL-1003, ¶¶239-240, 153-156
[1B.1/Profile] <u>said</u> first device profile representing a first set of attributes of said first memory location, and said second device profile representing a second set of attributes of said second memory location, <u>and a difference exists between said first and second device profiles</u> ;	[100B.1/Profile] <u>the</u> first device profile representing a first set of attributes of the first memory location, and <u>the</u> second device profile representing a second set of attributes of the second memory location;	§V.B.2.b INTEL-1003, ¶¶238, 123-127
[1C/Action] after identifying the command details,	[100C/Action] after identifying the command	§V.B.3.c INTEL-1003, ¶¶239-

Claim 1	Claim 100	
obtaining the first set of attributes;	details, obtaining the first set of attributes;	240, 157-162
[1C.2/Structural] wherein the memory store is directly coupled to at least one <u>memory bus</u> and wherein the <u>at least one memory bus is directly coupled to a controller, and</u>	[100C.1/Structural] wherein the memory store is directly coupled to a <u>memory controller, and</u>	§V.B.1.a-e INTEL-1003, ¶¶237, 91-115
[1C.3/Read-Write] wherein the controller performs the <u>memory read transaction and memory write transaction on the memory store</u>	[100C.2/Transaction] wherein the <u>memory controller performs a memory transaction;</u>	§V.B.4.a INTEL-1003, ¶¶239-240, 170-174
[1C.4/Structural] wherein <u>the host is directly coupled to the controller by a communication path as the controller performs the memory read transaction or memory write transaction</u>	[100C.3/Structural] wherein a host is directly coupled to the <u>memory controller [transactional] as the memory controller performs the memory transaction;</u>	§§V.B.1.f, V.B.4.a INTEL-1003, ¶¶237, 239-240, 116, 170-174
[1D/Action] using attributes from said first and second device profiles; and	[100D/Action] using attributes from the first and second device profiles; and	§V.B.3.c INTEL-1003, ¶¶239-240, 157-163
[1E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.	[100E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.	§V.B.3.d INTEL-1003, ¶¶239-240, 164-167

E. Dependent Claims

1. First/Second Memory Location Claims 2 and 33

The combination of Danilak and Cornwell discloses that “*said first memory location is located in a first memory device, and said second memory location is located in a second memory device*” [2]/[33]. (INTEL-1003, ¶¶241-242.)

These claims merely restate limitations present in limitation [1A] (“*first memory location in a first memory device*”) and [1A.2] (“*second memory location is in a second memory device*”). For the reasons discussed in §§V.B.3.a, V.B.1.b, the combination of Danilak and Cornwell renders claims 2 and 33 obvious.

2. “Attribute Qualifier” Claims 5-7, 36-38

Both Danilak and Cornwell teach that each “*attribute*” has an “*attribute qualifier*” (numerical value or non-numerical type). (See §V.B.2.a) The “*attribute qualifiers*” vary for memory devices in the combination of Danilak and Cornwell. (See INTEL-1005, ¶68 (“various attached flash memory devices may have different attributes”).) As discussed in §V.A.3.b, the combination of Danilak and Cornwell includes both NOR Flash devices and NAND Flash devices. The device profile for a NOR Flash device has different “*attribute qualifiers*” than the device profile for a NAND Flash device. (INTEL-1003, ¶246.) For example, the non-numerical qualifiers for the device type attribute (NAND or NOR) differ between NAND and NOR devices. (*Id.*) The numerical qualifiers for page size, block size,

access time, and device density also differ between NOR and NAND devices. (*Id.*, *citing* INTEL-1011, 3 (noting differences (e.g., access time and density) between NAND and NOR devices); INTEL-1010, 5 (noting differences (e.g., access time) between NAND and NOR devices).)

Thus, combination of Danilak and Cornwell discloses that “*said difference includes different attribute qualifiers for at least one set of shared attributes from said first and second device profiles*” [5]/[36], “*said different attribute qualifiers [e.g., page size, block size, access time, and density] include different attribute values*” [6]/[37] and “*said different attribute qualifiers [e.g., device type and ONFI support] include different attribute types*” [7]/[38]. (INTEL-1003, ¶¶243-247 (5/36), ¶¶248-251 (6/37), and ¶¶252-253 (7/38).)

3. “*Attribute*” Claims 8, 11, 39, 42

In the combination of Danilak and Cornwell, each memory device has a device profile which specifies the same set of attributes for each memory device. (*See* §V.B.2.a.) The device data stored in the profile includes device identification information, **device type**, **page size**, block size, sequential access time and device density as taught by Cornwell and additional data disclosed by Danilak (and illustrated in Figure 9 below) including **ONFI support**, cell type, number of simultaneously programmed pages, cache program, etc.). (INTEL-1006, ¶42; INTEL-1005, Figure 9, ¶59.) These attributes, included in each data profile, are

therefore “shared attributes” and “include a memory device type attribute” [8]/[39] and “include[] any one of the following memory device attributes: partial page size, page size, memory protocol, and memory device type” [11]/[42]. (INTEL-1003, ¶¶254-255 (8/39) and ¶¶256-257 (11/42).)

900

Byte	Bits	Mnemonic	Description
0	0	—	Reserved
	1	ONFI	ONFI standard support (No, Yes)
	3:2	CHIPS	Number of stacked chips (1, 2, 4, 8)
	7:4	—	Reserved
1	8	MKR	External maker code reported by flash interface circuit to host.
2	8	DEV	External device code reported by flash interface circuit to host.
3	1:0	CIPN	Internal chip number (1, 2, 4, 8)
	3:2	CELL	Cell type (2, 4, 8, 16 levels)
	5:4	SIMP	# of simultaneously programmed pages (1, 2, 4, 8)
	6	INTRL	Interleave between multiple chips (No, Yes)
	7	CACHE	Cache program (No, Yes)
4	1:0	PAGE	Page size (1, 2, 4, 8 KB)
	2	RSIZE	Redundant area size (8, 16 bytes / 512 bytes)
	7&3	SMIN	Serial access minimum (50/30, 25 ns)
	5:4	BSIZE	Block size (64, 128, 256, 512 KB)
	6	ORG	Organization (x8, x16)
5	1:0	—	Reserved
	3:2	PLANE	Plane number
	6:4	PLSIZE	Plane size (64, 128, 256, 512, 1Gb, 2Gb, 4Gb, 8Gb)
	7	—	Reserved
6	7:0	—	Reserved

FIG. 9
Danilak, Figure 9

4. NAND/NOR Attribute Claim 19

The memory of the combination of Danilak and Cornwell includes both NAND and NOR devices. (See §V.A.3.b.) As discussed in §V.E.2, the device profile for a NOR Flash device differs from the device profile for a NAND Flash device. For example, the non-numerical qualifiers for the device type attribute (NAND or NOR) and ONFI support (Yes or No) differ. (INTEL-1003, ¶¶259-260, 253.)

Thus, the combination of Danilak and Cornwell discloses “*wherein said first memory device type attribute corresponds to a NAND flash memory device and said second memory device type attribute corresponds to a NOR flash memory device*” [19]. (INTEL-1003, ¶¶258-260.)

5. “Memory Space” Claims 30 and 94

As discussed in §V.B.1.b, each memory device has a set of addressable locations. This set of addressable locations is the “*memory space*” for the device. The combination of Danilak and Cornwell discloses “*said first memory location is part of a memory space associated with a first memory device*” [30A] and “*said second memory location is part of a memory space associated with a second memory device*” [30B]. (INTEL-1003, ¶262.)

The “*memory store*” includes multiple memory devices. (See §V.B.1a, c.) Accordingly, the combination of Danilak and Cornwell discloses “*said first and*

second memory devices comprise at least a portion of a memory store” [30C].

INTEL-1003, ¶263.)

Each memory location in the attached memory devices is addressable. (*See* §V.B.1.b.) Accordingly, the combination of Danilak and Cornwell discloses “*the first memory location and the second memory location are addressable memory locations*” [94]. (INTEL-1003, ¶¶264-265.)

VI. Discretionary Denial Is Not Appropriate

A. Parallel Litigation

In addition to the compelling merits of the Petition, the *Fintiv* factors weigh heavily against discretionary denial. The ’190 patent was asserted against Intel in the Western District of Texas in early April 2022. The case was transferred to the Northern District of California (“NDCA”) in February 2023. The median-time-to-trial of all civil cases in NDCA is 32.9 months. (See INTEL-1027, 66.) Assuming that timing from the original filing date, the NDCA trial date is February 2025, over 2 months after the final written decision. Moreover, the co-pending district court litigation is in its early stages. The court has not issued a scheduling order and claim construction, expert discovery, summary judgment and trial all remain in the future.

B. Prior Petition

The *General Plastic* factors weigh heavily against discretionary denial. Although the '190 patent was challenged in a prior IPR proceeding on March 23, 2023 by Kioxia America, Inc. ("Kioxia IPR"), these are Intel's first Petitions challenging the '190 patent. Intel has no significant relationship with the prior petitioner, Kioxia. *Cf. Valve Corp. v. Electronic Scripting Products, Inc.*, IPR2019-00062, Paper 11, 2 (April 2, 2019) (precedential) (denying institution where significant business relationship existed between the Petitioners and Petitioners were accused of infringement based on same product). Unlike the parties in *Valve*, Kioxia and Intel are being accused of infringement based on **different** products in different litigations and no significant business relationship exists between Intel and Kioxia. *Id.* Additionally, Intel was not involved in the preparation or filing of the Kioxia IPR. Merely being accused of infringing the same patent does not create a relationship that justifies discretionarily denying institution. *See, e.g., Siemens Gamesa Renewable Energy, Inc. v. General Electric Co.*, IPR2022-01479, Paper 11, 11-13 (March 7, 2023).

The remaining factors similarly weigh against denial. Factors 2 and 4 relate to prior art withheld from the first petition and Factor 5 relates to explanation of timing between multiple petition filings. Because Intel did not file the Kioxia IPR, these factors are inapplicable, weighing against denial. Intel's IPRs are being filed

only 9 days after the filing of the Kioxia IPR, no substantive responses have been filed by PO and no decisions have been issued by the Board. Moreover, the present Petitions by Intel asserts different grounds than the Kioxia IPR and challenge additional claims. Road mapping concerns are therefore not present and Factor 3 weighs against denial. Because the Petition was filed only 9 days after the Kioxia IPR and challenges some of the same claims and asserts one of the same references (Cornwell), the petitions can be efficiently considered by the Board. Factors 6 and 7 also weigh against denial. *See Western Digital Techs., Inc. v. Viasat, Inc.*, IPR2022-01126, Paper 13, 13-15 (November 29, 2022).

VII. Mandatory Notices

A. Real Party In Interest

Intel is the real party in interest.

B. Related Matters

To the best of Intel's knowledge, the '190 patent has been involved in the following district court litigations:

- *BiTMICRO LLC v. KIOXIA America, Inc.*, 6-22-cv-00331 (WDTX)
- *BiTMICRO LLC v. Intel Corp.*, 6-22-cv-00335 (WDTX)
- *KIOXIA America v. BiTMICRO LLC*, IPR2023-00741 (PTAB)

C. Notice of Counsel and Service Information

Pursuant to 37 C.F.R. §§ 42.8(b)(3), 42.8(b)(4) and 42.10(a), Intel appoints **Lori A. Gordon** (Reg. No. 50,633) at the address of Perkins Coie LLP, 700 Thirteenth Street, N.W., Suite 800, Washington, DC 20005-3960, phone number 202-654-6200, as its lead counsel; and **Sarah Fowler** (Reg. No. 54,854) at the address of 3150 Porter Dr., Palo Alto, CA 94304, and **Tyler Bowen**, (Reg. No. 60,461) at the address of 2901 North Central Avenue, Suite 2000, Phoenix, AZ 85012, and **Chad Campbell** (To Be Admitted *Pro Hac Vice*), 2901 North Central Avenue, Suite 2000, Phoenix, AZ 85012, as its backup counsel.

Intel consents to electronic service by email at the email addresses:
gordon-ptab@perkinscoie.com; stahnke-ptab@perkinscoie.com;
Bowen-ptab@perkinscoie.com; and Campbell-ptab@perkinscoie.com.

VIII. Conclusion

Inter partes review of the challenged claims is respectfully requested.

Respectfully submitted,

/Lori A. Gordon/

Lori A. Gordon
Reg. No. 50,633
Attorney for Petitioner

PERKINS COIE LLP
700 Thirteenth Street, NW, Suite 800
Washington, DC 2005

Date: April 1, 2023

APPENDIX OF INDEPENDENT CLAIMS

[1P] A method of performing memory transactions **[structural]** on a set of memory locations that includes a first memory location and a second memory location, the method comprising:

[1A/Action] performing a memory transaction comprising addressing a first memory location **[structural]** in a first memory device in a memory store,

[1A.1/Profile] said first memory location and a second memory location respectively associated with a first device profile and a second device profile,

[1A.2/Structural] wherein said second memory location is in a second memory device in the memory store,

[1A.3/Read-Write] wherein a memory read transaction or a memory write transaction is performed on at least one of the first memory location or second memory location,

[1A.4/Read-Write] wherein data is transferred to the first memory location or second memory location from a host for a memory write transaction,

[1A.5/Read-Write] wherein data is transferred from the first memory location or second memory location to the host for a memory read transaction;

[1A.6/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;

[1B/Action] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;

[1B.1/Profile] said first device profile representing a first set of attributes of said first memory location, and said second device profile representing a second set

of attributes of said second memory location, and a difference exists between said first and second device profiles;

[1C/Action] after identifying the command details, obtaining the first set of attributes;

[1C.1/Profile] wherein each attribute in the first set of attributes and second set of attributes is associated with a respective attribute qualifier that qualifies a respective memory location;

[1C.2/Structural] wherein the memory store is directly coupled to at least one memory bus and wherein the at least one memory bus is directly coupled to a controller, and

[1C.3/Read-Write] wherein the controller performs the memory read transaction and memory write transaction on the memory store;

[1C.4/Structural] wherein the host is directly coupled to the controller by a communication path **[Read-Write]** as the controller performs the memory read transaction or memory write transaction;

[1C.5/Profile] wherein the first device profile and second device profile are each stored in the memory store; and

[1D/Action] using attributes from said first and second device profiles; and

[1E/Action] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.

[29P] A non-transitory computer readable medium comprising computer executable instructions adapted to cause a method of performing memory transactions on a first memory location and a second memory location, said method comprising:

[29A] performing a memory transaction comprising addressing a first memory location,

[29A.1/Profile] said first memory location and a second memory location are respectively associated with a first device profile and a second device profile;

[29A.2/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;

[29B] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;

[29B.1/Profile] said first device profile representing a first set of attributes of said first memory device, and said second device profile representing a second set of attributes of said second memory device, and a difference exists between said first and second device profiles;

[29C] after identifying the command details, obtaining the first set of attributes; and

[29D] using attributes from said first and second device profiles; and

[29E] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.

[100P] A method for performing memory transactions, the method comprising:

[100A] performing a memory transaction comprising addressing a first memory location **[structural]** in a first memory device in a memory store,

[100A.1/Profile] the first memory location and a second memory location respectively associated with a first device profile and a second device profile;

[100A.2/Profile] wherein said first device profile is optimal for a data type subject to the memory transaction, wherein said data type comprises one of a random data type or a sequential data type;

[100B] said performing the memory transaction further comprising identifying command details for causing the memory transaction to be performed, wherein said command details comprising the first memory device;

[100B.1/Profile] the first device profile representing a first set of attributes of the first memory location, and the second device profile representing a second set of attributes of the second memory location;

[100C] after identifying the command details, obtaining the first set of attributes;

[100C.1/Structural] wherein the memory store is directly coupled to a memory controller, and

[100C.2/Transaction] wherein the memory controller performs a memory transaction;

[100C.3/Structural] wherein a host is directly coupled to the memory controller **[transaction]** as the memory controller performs the memory transaction; and

[100D] using attributes from the first and second device profiles; and

[100E] selecting a transfer size for the memory transaction, wherein the transfer size is a function of a data size of the memory transaction and the first set of attributes.

CERTIFICATE OF WORD COUNT UNDER 37 CFR §42.24(d)

Pursuant to 37 C.F.R. §42.24(a), Intel hereby certifies that portions of the above-captioned Petition for *Inter Partes* Review of U.S. Patent No. 9,135,190, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 12,624. Pursuant to 37 C.F.R. §42.24(a), this word count is in compliance and excludes the table of contents, table of authorities, mandatory notices under §42.8, certificate of service, certificate of word count, an appendix of exhibits, and any claim listing. This word count was prepared using Microsoft Word.

Respectfully submitted,

/Lori A. Gordon/

Lori A. Gordon
Attorney for Petitioner

Date: April 1, 2023

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of the Petition for *Inter Partes* Review of U.S. Patent No. 9,135,190 and supporting materials (Explanation for and Ranking of Two Petitions Challenging U.S. Patent No. 9,135,190, Exhibits and Power of Attorney) were served via overnight delivery on the Patent Owner at the correspondence address of record as listed on PAIR:

Jackson Walker LLP
100 Congress Avenue, Suite 1100
Austin, TX 78701

and via email to the attorneys of record for Plaintiff in the concurrent *BiTMICRO LLC v. Intel Corp.*, 6-22-cv-00335 litigation:

Corey A Johanningmeier - cjohanningmeier@bdiplaw.com

Denise M De Mory - ddemory@bdiplaw.com

Michael E Flynn-O'Brien - mflynnobrien@bdiplaw.com

Richard C Lin - rlin@bdiplaw.com

B Russell Horton - rhorton@gbkh.com

Respectfully submitted,

/Lori A. Gordon/

Lori A. Gordon
Attorney for Petitioner

Date: April 1, 2023