

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTEL CORPORATION,
Petitioner

v.

BiTMICRO LLC,
Patent Owner.

IPR2023-00782

U.S. Patent No. 8,010,740

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 8,010,740**

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.	STANDING	1
III.	CHALLENGE	1
A.	Prior Art	1
1.	U.S. Patent No. 6,000,006 to Bruce et al. (“Bruce-006”; INTEL-1005), issued December 7, 1999, is prior art under pre-AIA §102(b).	2
2.	U.S. Patent No. 5,822,251 to Bruce et al. (“Bruce-251”; INTEL-1006), issued October 13, 1998, is prior art under pre-AIA §102(b).	2
3.	“ <i>American National Standard for Information Technology - SCSI-3 Block Commands (SBC)</i> ” (“SBC”; INTEL-1007), published in 1998 is prior art under pre-AIA §102(b).....	2
B.	Challenge Grounds.....	2
IV.	’740 PATENT OVERVIEW	2
A.	Specification.....	3
B.	POSITAs	5
C.	Claim Construction	5
V.	GROUND 1: CLAIMS 1 AND 14 ARE OBVIOUS OVER BRUCE-006.....	6
A.	Bruce-006 Overview	6
B.	Independent Claim 1	8
1.	Preamble [1P].....	8
2.	“ <i>Set of Logical Fields</i> ” [1A]	14
3.	“ <i>Set of PBA Fields</i> ” [1B]	18
4.	“ <i>Causes Optimized Memory Operations</i> ” [1C]	22
C.	Dependent Claim 14.....	27
VI.	GROUND 2: CLAIMS 1 AND 14 ARE OBVIOUS OVER BRUCE-006 COMBINED WITH SBC	28

TABLE OF CONTENTS (continued)

	Page
A. SBC Overview	28
B. Motivation to Combine Bruce-006 with SBC	30
C. Independent Claim 1	32
1. Preamble [1P].....	32
2. “Set of Logical Fields” [1A] / “Set of PBA Fields” [1B].....	35
3. “Causes Optimized Memory Operations” [1C]	35
D. Dependent Claim 14.....	39
VII. GROUND 3: CLAIMS 1, 9-15, 32, AND 34 ARE OBVIOUS OVER THE COMBINATION OF BRUCE-006 AND BRUCE- 251	39
A. Combination Overview	39
1. Bruce-251 Overview	39
2. Motivation to Combine Bruce-006 and Bruce-251	41
B. Independent Claim 1	46
1. Preamble [1P].....	46
2. “Set of Logical Fields” [1A] / “Set of PBA Fields” [1B] / “Causes Optimized Memory Operations” [1C]	49
C. Dependent Claim 9.....	49
D. Dependent Claim 10.....	55
E. Dependent Claim 11.....	57
F. Dependent Claims 12 and 13	57
G. Dependent Claim 14.....	58
H. Dependent Claim 15.....	58
1. “Memory Store” [15A].....	58
2. “Group Identifier for Identifying a Portion of Memory Devices Controlled by a FDE” [15B]	61
I. Independent Claim 32	63

TABLE OF CONTENTS (continued)

	Page
1. Preamble [32P].....	63
2. “ <i>First and Second LBA Sets Mapped to First and Second PBAs</i> ” [32A]	67
3. “ <i>First and Second Physical Memory Locations Addressable by First and Second PBAs</i> ” [32B]	68
4. “ <i>Memory System</i> ” [32C]	68
5. “ <i>One Difference Between Said First and Second PBAs</i> ” [32D]	70
6. “ <i>Optimally Distributed</i> ” [32E].....	70
J. Dependent Claim 34.....	71
VIII. GROUND 4: CLAIMS 1, 9-15, 32, AND 34 ARE OBVIOUS OVER BRUCE-006, BRUCE-251 AND SBC COMBINED	72
A. Motivation to Combine Bruce-006, Bruce-251, and SBC.....	72
B. Independent Claim 1	74
1. Preamble [1P].....	74
2. “ <i>Set of Logical Fields</i> ” [1A] / “ <i>Set of PBA Fields</i> ” [1B].....	76
3. “ <i>Causes Optimized Memory Operations</i> ” [1C]	76
C. Dependent Claims 9-15.....	77
D. Independent Claim 32	77
1. Preamble [32P].....	77
2. “ <i>First and Second LBA Sets Mapped to First and Second PBAs</i> ” [32A] / “ <i>First and Second Physical Memory Locations Addressable by First and Second PBAs</i> ” [32B] / “ <i>Memory System</i> ” [32C] / “ <i>One Difference Between Said First and Second PBAs</i> ” [32D]	78
3. “ <i>Optimally Distributed</i> ” [32E].....	79
E. Dependent Claim 34.....	79
IX. DISCRETIONARY DENIAL IS NOT APPROPRIATE	80

TABLE OF CONTENTS
(continued)

	Page
A. 35 U.S.C. §325(d)	80
B. 35 U.S.C. §314(a)	82
X. MANDATORY NOTICES	84
A. Real Party In Interest.....	84
B. Related Matters	84
C. Notice Of Counsel And Service Information.....	85
XI. CONCLUSION.....	85

EXHIBIT LIST

Exhibit	Reference
1001	U.S. Patent No. 8,010,740 to Arcedera et al. (“the ’740 patent”)
1002	File History of the ’740 patent
1003	Declaration of Dr. Carl Sechen, Ph.D. in Support of Petition
1004	CV of Dr. Carl Sechen, Ph.D.
1005	U.S. Patent No. 6,000,006 to Bruce et al. (“Bruce-006”)
1006	U.S. Patent No. 5,822,251 to Bruce et al. (“Bruce-251”)
1007	“ <i>American National Standard for Information Technology - SCSI-3 Block Commands (SBC)</i> ,” American National Standards Institute (1998) (“SBC”)
1008	United States District Courts — National Judicial Caseload Profile
1009	The IEEE Standard Dictionary of Electrical and Electronics Terms (6th Ed. 1996)
1010	U.S. Patent No. 5,519,844 to Stallmo (“Stallmo”)
1011	Microsoft Press Computer Dictionary (3rd Ed. 1999)
1012	U.S. Patent No. 7,299,316 to Chou et al. (“Chou”)
1013	Summons with Proof of Service, <i>BiTMICRO LLC v. Intel Corp.</i> , No. 6:22-cv-00335-ADA (W.D. Tex.), Dkt. 9
1014	Declaration of June Munford in Support of Petition
1015	Claim Construction Order and Memorandum In Support Thereof, <i>BiTMICRO LLC v. Kioxia America, Inc., Kioxia Corporation</i> , No. 6:22-cv-00331-ADA (W.D. Tex.), Dkt. 54
1016	U.S. Patent App. Pub. No. 2003/0163630 to Aasheim et al. (“Aasheim”)
1017	Wu et al., <i>An Efficient B-Tree Layer for Flash-Memory Storage Systems</i> , RTCSA 2003: Real-Time and Embedded Computing Sys. and Applications: 9th Int’l. Conf., Revised Papers, Vol. 2968, pp. 409-430 (February 18-20, 2004) (“Wu”)

Exhibit	Reference
1018	Gal et al., <i>Algorithms and Data Structures for Flash Memories</i> , ACM Computing Survs., Vol. 37, No. 2, pp. 138-163 (June 2005) (“Gal”)
1019	Kim et al., <i>A Space-Efficient Flash Translation Layer for CompactFlash Systems</i> , IEEE Transactions on Consumer Elecs., Vol. 48, No. 2, pp. 366-375 (May 2002) (“Kim”)
1020	Pavan et al., <i>Flash Memory Cells—An Overview</i> , Proc. of the IEEE, Vol. 85, No. 8, pp. 1248-1271 (1997) (“Pavan”)
1021	Yim, <i>A Novel Memory Hierarchy for Flash Memory Based Storage Systems</i> , JSTS: Journal of Semiconductor Technology and Science, Vol. 5, No. 4, pp. 262-269 (2005) (“Yim”)
1022	U.S. Patent No. 5,479,638 to Assar et al. (“Assar”)
1023	U.S. Patent No. 7,000,063 to Friedman et al. (“Friedman”)
1024	Huffman et al., <i>A Standard Interface for NAND Flash</i> , Intel Developer Forum (March 7, 2006) (“Huffman”)
1025	Chang et al., <i>An Adaptive Striping Architecture for Flash Memory Storage Systems of Embedded Systems</i> , Proceedings. Eighth IEEE Real-Time and Embedded Tech. and Applications Symp. pp. 187-196 (September 2002) (“Chang”)
1026	Park et al., <i>A High Performance Controller for NAND Flash-based Solid State Disk (NSSD)</i> , 2006 21st IEEE Non-Volatile Semiconductor Memory Workshop, pp. 17-20 (May 15, 2006) (“Park”)
1027	U.S. Patent No. 7,506,098 to Arcedera et al. (“Arcedera”)
1028	Tsur, <i>Trends in COTS Storage Solutions for Data Acquisition Systems</i> , Airborne Intelligence, Surveillance, Reconnaissance (ISR) Systems and Applications III, Vol. 6209, pp. 45-55 SPIE (May 5, 2006) (“Tsur”)
1029	Ryu et al., <i>Improvement of Space Utilization in NAND Flash Memory Storages</i> , Embedded Software and Systems: Second

Exhibit	Reference
	International Conference, ICESS 2005, Xi'an, China, Proceedings 2, pp. 766-775 (December 16-18, 2005) (“Ryu”)
1030	Chung et al., <i>LSTAFF: System Software For Large Block Flash Memory</i> , Lecture Notes in Computer Science, 3398 pp. 704-710 (2005) (“Chung”)
1031	U.S. Patent No. 5,404,485 to Ban (“Ban”)
1032	Peterson, <i>Designing A VME-To-SCSI Adapter</i> , The System Engineers Handbook, pp. 311-319 (1992) (“Peterson”)
1033	American National Standards Institute (ANSI) X3.131-1986, <i>Small Computer System Interface (SCSI)</i> (1986) (“SCSI-1986”)
1034	U.S. Patent No. 6,381,677 to Beardsley et al. (“Beardsley”)

I. Introduction

Intel Corporation seeks *inter partes* review of U.S. Patent No. 8,010,740 claims 1, 9-15, 32, and 34. The claimed subject matter was obvious in view of patents that the original assignee filed almost nine years before the earliest possible priority date for the '740 patent. Those prior art patents were never disclosed to the USPTO during the prosecution of the '740 patent. The Board should accordingly institute a trial and cancel the challenged claims.

II. Standing

Intel certifies the '740 patent is available for IPR and Intel is not barred or estopped from requesting IPR on the grounds herein.

III. Challenge

A. Prior Art

The '740 patent was filed November 25, 2008, as a continuation-in-part ("CIP") of Application No. 11/450,005, filed June 8, 2006. (INTEL-1001, (63), 1:7-9.) Each reference below was published before the earliest possible priority date of June 8, 2006 ("Earliest Priority Date").

1. **U.S. Patent No. 6,000,006 to Bruce et al.** (“Bruce-006”; INTEL-1005), issued December 7, 1999, is prior art under pre-AIA §102(b).
2. **U.S. Patent No. 5,822,251 to Bruce et al.** (“Bruce-251”; INTEL-1006), issued October 13, 1998, is prior art under pre-AIA §102(b).
3. **“American National Standard for Information Technology - SCSI-3 Block Commands (SBC)”** (“SBC”; INTEL-1007), published in 1998 is prior art under pre-AIA §102(b).

B. Challenge Grounds

Ground		Claims	Prior Art
1	103	1, 14	Bruce-006
2	103	1, 14	Bruce-006 combined with SBC
3	103	1, 9-15, 32, and 34	Bruce-006 combined with Bruce-251
4	103	1, 9-15, 32, and 34	Bruce-006 combined with Bruce-251 and SBC

IV. '740 Patent Overview

The '740 patent describes optimizing memory operations in response to a host-initiated I/O transaction request by “optimally distribut[ing]” operational load across memory system resources, thereby “increasing operational efficiency, reducing memory operation latency or both.” (INTEL-1001, 1:16-24, 3:15-21.)

A. Specification

The '740 patent describes a “[m]apping table 9 [that] associates LBAs from th[e] host logical addressing system to the device addressing system used by storage device 2.” (INTEL-1001, 6:42-44.) In annotated Figure 2, the table

“includes a set of fields, named logical fields 26 [highlighted blue], and a set of PBA fields 28 [highlighted red].”

(INTEL-1001, 5:29-30.)

“[E]ach logical field [(40, 42, 44, 45)] in set of

logical fields 26 ... represent[s] a LBA set” and each LBA set is comprised of a “set[] of consecutive LBAs [highlighted green].” (INTEL-1001, 6:46-51.)

Each PBA field (46, 48, 50, 51) “represent[s] a unique addressable physical memory location, named PBA, in memory store 6 [highlighted purple in annotated Figure 1 below].” (INTEL-1001, 5:30-33.) “For each PBA field, th[e] physical memory location may be represented by using a set of access parameters fields 30

9 ↙

Logical Fields 26	PBA Fields 28			
	Access Parameter Fields 30			Memory Index Fields 32
	Bus Id Fields 34	FDE Id Fields 36	Group Id Fields 38	
40 LBAs 00-07	8-0	22-01	15-0	0x00 46
42 LBAs 08-15	8-0	22-02	15-1	0x08 48
44 LBAs 16-23	8-1	22-11	15-0	0x00 50
45 LBAs 24-31	8-1	22-12	15-1	0x08 51
⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮

and a memory index field 32.” (INTEL-1001, 5:33-35.) “The set of access parameter[] fields 30 includes a bus identifier field 34, a FDE identifier field 36, and a group identifier field 38.” (INTEL-1001, 5:36-38.)

The '740 patent teaches that “[l]ogical fields in mapping table 9 may be initialized to represent LBA sets that are serially arranged in mapping table 9 so that LBAs from adjacent LBA sets are contiguous”

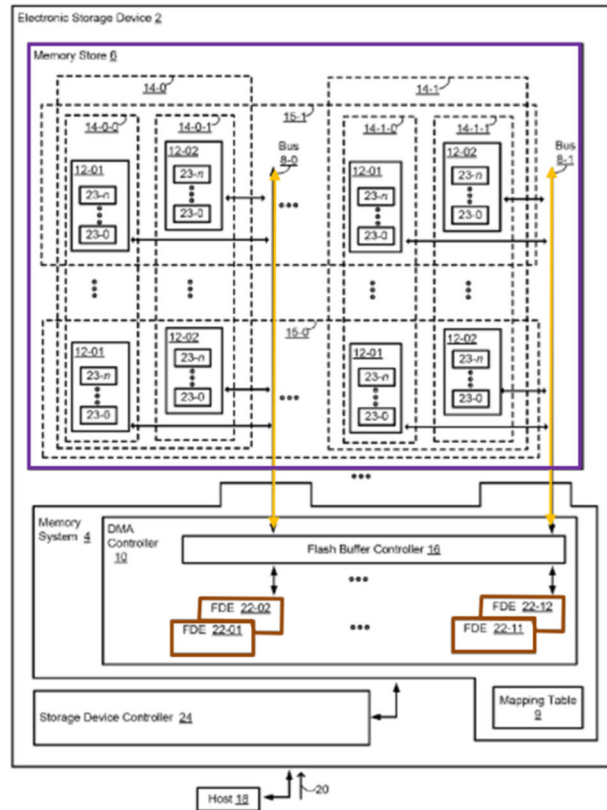


FIG. 1

and that “any two adjacent PBAs differ by at least one access parameter.” (INTEL-1001, 6:55-57, 7:5-6.) The ’740 patent explains that for “a sequential I/O transaction ... associated with consecutive LBAs,” “[s]ince mapping table 9 has been initialized to map these LBAs 00-07 and LBAs 08-15 with LBA sets 40 and 42, respectively, [the] DMA controller would more than likely be able to perform memory operations ... in an optimal manner by interleaving these memory operations across different buses, different FDEs, different flash memory devices, or any combination of these.” (INTEL-1001, 9:28-37.)

B. POSITAs

Persons of ordinary skill in the art would have had at least a bachelor's degree in electrical engineering or computer engineering and two years' experience in memory system design, including solid-state memory, or equivalent combined education, work, and/or experience. (INTEL-1003, ¶22.)

C. Claim Construction

Under 37 C.F.R. § 42.100(b), claims are construed according to *Phillips v. AWH Corp.*, which holds that “words of a claim are generally given their ordinary and customary meaning,” and “claims must be read in view of the specification, of which they are a part.” 415 F.3d 1303, 1312-13, 15 (Fed. Cir. 2005) (citations omitted). If the specification defines a claim term, “the inventor’s lexicography governs.” *Id.* at 1316. The ’740 patent includes express lexicography for “LBA” and “LBA set” that this Petition identifies, while applying plain meaning for the other claim terms.¹

¹ Intel reserves the right to argue in co-pending district court litigation that “optimized memory operations” and “optimally distributed” are indefinite under *Nautilus, Inc. v. Biosig Instruments, Inc.*, 572 U.S. 898, 901 (2014).

LBA	“The term ‘LBA’, which may also be referred to herein as a logical block address, is intended to represent an address that is part of a logical addressing system (not shown) used by a host.” (INTEL-1001, 6:37-40)
LBA set	“These LBAs may be grouped into sets of consecutive LBAs, named ‘LBA sets.’” (INTEL-1001, 6:46-48)

Although “block” appears in “LBA” (or “logical block address”), the ’740 patent teaches that sizes of the claimed LBAs or LBA sets are not tied to sizes of a physical flash block and may be smaller, the same as, or larger than a physical block. (INTEL-1001, 7:25-27, 7:31-40; INTEL-1003, ¶79.)

V. GROUND 1: Claims 1 and 14 Are Obvious Over Bruce-006

A. Bruce-006 Overview

Bruce-006 issued on December 7, 1999, making it prior art under pre-AIA 35 U.S.C. §102(b). Bruce-006 was never disclosed during prosecution of the ’740 patent.

Bruce-006 discloses a “re-map table ... used to arbitrarily re-map all logical addresses from a host system to physical addresses of flash-memory devices.” (INTEL-1005, Abstract, 6:19-20.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

90

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Annotated-Figure 7A, 8:3-5.) Bruce-006 explains that “host requests contain[ing] a logical address which includes a logical-block address (LBA) [highlighted blue]” “are first re-mapped by unified re-map table 20.” (INTEL-1005, 6:12-17.) “The re-map table includes a physical block address (PBA) [highlighted red] for each possible logical block address” (INTEL-1005, 5:42-43) and “two write-counters for wear-leveling” (INTEL-1005, 3:15). Once an entry in the re-map table is determined to be ready for wear-leveling, “[t]he wear-leveling swap [] moves the data for the heavily-used LBA 011 to the lightly-used PBA 101, while the existing data in [the lightly-used] PBA 101 for [the lightly-used] LBA 010 is transferred to the cache and then to the heavily-written PBA 111.” (INTEL-1005, 8:29-39.)

Lastly, Bruce-006 discloses that “[i]nterleaving can be combined with the invention.”² (INTEL-1005, 11:20.) “Interleaving is desirable since multiple accesses can occur in parallel to the flash devices, improving performance.” (INTEL-1005, 11:20-22.) Such performance improvement can result “when sequential blocks are accessed.” (INTEL-1005, 11:39-40.)

B. Independent Claim 1

1. Preamble [1P]

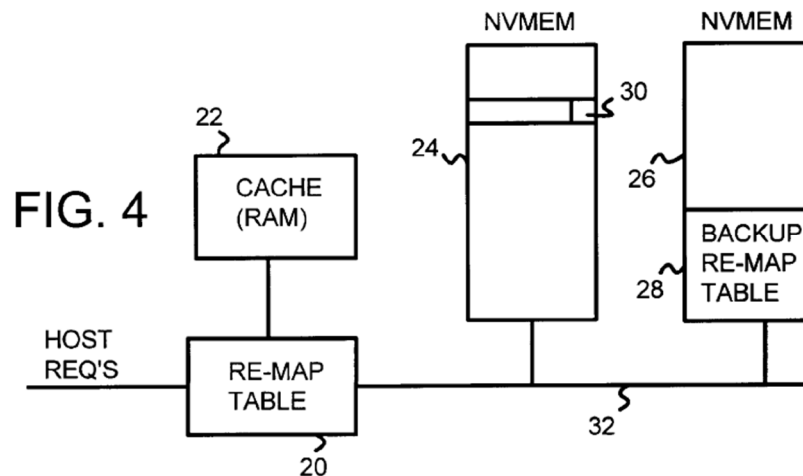
If the preamble is limiting, Bruce-006 discloses and renders obvious “[a] *mapping table for optimizing memory operations performed by an electronic storage device in response to receiving an I/O transaction request initiated by a host.*” (INTEL-1003, ¶¶86-111.)

Entitled “Unified **Re-Map** and Cache-Index **Table** ... for Wear-Leveling of Non-Volatile Flash RAM Mass Storage,” Bruce-006 discloses a mapping table and teaches that “[h]ost requests for access to flash storage are first re-mapped by [a] unified **re-map table**.” (INTEL-1005, [54], 6:12-13, 5:15-17, 12:40-43.) “A single lookup of the **re-map table** for a logical address produces the physical block

² “Interleaved memory,” generally, refers to “[a] type of memory in which two or more separate arrays are used to fill alternate accesses in such a way as to speed the average access time of the memory.” INTEL-1009 at 544.

address (PBA) of the data in flash memory.” (INTEL-1005, 5:33-35.) Thus, Bruce-006 discloses the claimed “*mapping table*.” (INTEL-1001, 6:42-44 (“Mapping table 9 associates LBAs from this host logical addressing system to the device addressing system used by storage device 2.”); INTEL-1003, ¶¶87-89.)

Bruce-006 further teaches an electronic storage device—specifically, a flash-memory system as in Figure 4.



(INTEL-1005, Figure 4.) The “flash-memory system provides solid-state [] **storage** as a replacement to a hard disk.” (INTEL-1005, Abstract, 1:7-8 (“This invention relates to non-volatile [] **storage** system”), 6:8-9 (“Such a system is useful as a **mass-storage** replacement to a hard disk on a personal computer.”)) The Bruce-006 flash-memory system thus corresponds to the claimed “*electronic storage device*.” (INTEL-1003, ¶¶90-92.)

Bruce-006 further teaches a flash-memory system receiving host-initiated I/O transaction requests and performing memory operations in response. (INTEL-1005, 5:40-42 (“Logical block addresses (LBA’s) **from** a user or **host system** are **received**.”), 6:12-13 (“**Host requests** for access to flash storage are first re-mapped by unified re-map table 20.”)) Host requests include both reads and writes. (INTEL-1005, 6:12-42, 8:51-52, 9:30-32.) Because Bruce-006 teaches that the flash-memory system “is useful as a mass-storage replacement to a hard disk,” POSITAs would understand Bruce-006’s host requests to be “*I/O [input/output] transaction request[s] initiated by a host.*” (INTEL-1005, 6:8-9; INTEL-1003, ¶93; *see also*, INTEL-1011, 252 (“A disk drive is an example of an input/output device”).)) Furthermore, Bruce-006’s host requests match the I/O transaction requests in the ’740 patent. (INTEL-1001, Figure 1 (showing I/O transaction requests as requests between an electronic storage device and a host), 9:26-27 (providing “a write transaction” as an example “I/O transaction request”).))

Bruce-006 also teaches that “[t]hese host requests contain a logical address which includes a logical-block address (LBA) and an offset address that identifies a page within a flash block and perhaps a byte within a page for a read access.” (INTEL-1005, 6:13-17.) For each host request received by the flash-memory system, “[t]he entry for the current LBA is read from unified re-map table 20, and a cache-valid bit is checked to determine if the LBA’s data is in cache 22.”

(INTEL-1005, 6:12-13, 6:20-22; *see also*, INTEL-1005, 7:26-31.) “When the cache valid bit indicates that the data is not stored in cache 22, then a PBA field in the entry retrieved from unified re-map table 20 is used **to access the data in flash memory.**” (INTEL-1005, 6:35-38; *see also*, INTEL-1005, 7:31-33.) Thus, Bruce-006 discloses a flash-memory system performing “*memory operations*” (i.e., accesses to data in flash memory) “*in response to receiving I/O transaction requests initiated by a host.*” (INTEL-1003, ¶¶94-95.)

Bruce-006 then teaches that multiple memory operations are performed in response to a single I/O transaction request initiated by a host. When the host’s write request causes the total-write count and incremental-write count associated with a LBA to exceed the total and incremental-write thresholds, wear-leveling, which requires multiple memory operations as illustrated in Figure 7A, becomes necessary. (INTEL-1005, 8:29-9:12.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

90

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Figure 7A.) Bruce-006 explains that “[e]ntry 90, for LBA 011, is ready for wear-leveling, since its total writes [count] has exceeded the total threshold of 250,000, and its incremental writes [count] has also exceeded the incremental threshold of 25,000.” (INTEL-1005, 8:29-32.) “Each time a block is written, its total-write count is compared to a total-write threshold, while the block’s incremental-write count is compared to an incremental-write threshold” (consisting of the writes since the data in the block was last swapped out). (INTEL-1005, 4:53-5:5.) Because the threshold comparisons happen after each write request, POSITAs would understand that determining whether entry 90 is ready for wear-leveling is “*in response to*” a host’s write request to LBA 011. (INTEL-1005, 8:45-50; INTEL-1003, ¶99.) “The wear-leveling controller searches for a least-used flash block” which is “PBA 101, with 20,000 total writes.” (INTEL-1005, 8:32-34.) As “[t]his block is currently allocated to LBA 010,” “[t]he wear-leveling swap thus moves the data for the heavily-used LBA 011 to the lightly-used PBA 101, while the existing data in PBA 101 for LBA 010 is transferred to the cache and then to the heavily-written PBA 111.” (INTEL-1005, 8:35-39.) Figure 7B below shows the revised mapping table “after the wear-leveling swap of LBA’s 010 and 011.” (INTEL-1005, 8:40-41.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	111	250,002	1
011	101	20,001	1
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	313,001	25,001

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7B

92

(INTEL-1005, Figure 7B.) Therefore, Bruce-006 teaches a flash-memory system performing multiple memory reads or writes in response to an I/O transaction request initiated by a host (i.e., a write request when wear-leveling is determined to be necessary). (INTEL-1003, ¶¶96-101.)

Finally, Bruce-006 discloses a mapping table for “*optimizing memory operations.*” (INTEL-1003, ¶¶102-110.) Bruce-006 teaches that “[t]he **interleaving used** can be flexible by multiplexing different logical address bits to **the unified re-map table.**” (INTEL1005, 11:66-67.) Thus, Bruce-006 teaches that “interleaving can be combined with the re-map table.” (See also, INTEL-1005, 11:20.) As detailed further below (p.22-27), Bruce-006’s re-map table causes multiple wear-leveling memory operations to be performed on different flash devices in response to an I/O transaction request initiated by a host. (INTEL-1005, 7:66-9:37, Figures 7A-7E.) POSITAs would understand that memory

operations on different flash devices can be interleaved when block-interleaving is used. (INTEL-1003, ¶108.) Such “[i]nterleaving is desirable since multiple accesses can occur in parallel to the flash devices, improving performance.” (INTEL-1005, 11:20-22; *see also*, INTEL-1005, 11:37-40 (“With such block interleaving, ... [p]erformance is improved.”).) As detailed below (p.24-26), Bruce-006 also teaches wear-leveling operations associated with sequential LBAs. With Bruce-006’s teaching of performance improvement from interleaving “when sequential blocks are accessed,” POSITAs would understand Bruce-006 to teach interleaving and hence, optimizing, the corresponding memory operations. (INTEL-1005, 11:39-40; INTEL-1003, ¶104.) These disclosures in Bruce-006 match how the ’740 patent describes optimizing memory operations. (INTEL-1001, 2:14-19 (“These optimizing memory operations include increasing the likelihood that, in response to an I/O transaction initiated by a host, ... these memory operations will be optimally distributed across different storage device resources, such as by **interleaving** or **parallel memory operations**.”), 9:26-37 (“perform memory operations ... in an optimal manner by interleaving these memory operations across ... different flash memory devices”).)

2. “*Set of Logical Fields*” [1A]

Bruce-006 discloses and renders obvious a “*mapping table comprising... a set of logical fields, including a first logical field and a second logical field, and*

said logical fields respectively disposed for representing a plurality of LBA sets, including said first logical field disposed for representing a first LBA set and said second logical field disposed for representing a second LBA set, said first and second LBA sets each representing a set of consecutive LBAs.” (INTEL-1003, ¶¶112-123.)

The ’740 patent defines “LBA” as “an address that is part of a logical addressing system [] used by a host,” and “LBA set” as a “group[ing] ... of consecutive LBAs.” (INTEL-1001 6:37-40, 46-48.) Thus, in the context of claim 1, the first and second LBA sets represent groupings of consecutive logical addresses in the host’s addressing system, and the first and second logical fields of the mapping table represent the first and second LBA sets. (INTEL-1003, ¶112.)

Although Bruce-006 uses somewhat different terminology, POSITAs would understand it to disclose or render obvious the “LBAs,” “LBA sets,” and “logical fields” claimed in the ’740 patent. (INTEL-1003, ¶¶113-22.)

’740 term	Disclosure in Bruce-006
“LBA,” or “logical block address” (INTEL-1001, 6:37-40)	“logical address” (INTEL-1005, 6:12-16)
“LBA set” (INTEL-1001, 6:46-48)	“LBA” (INTEL-1005, 6:12-16)

“logical fields” (INTEL-1001, 6:48-50)	“unified re-map table ... entries ... for all LBAs” (INTEL-1005, 6:19-20, Fig. 7A)
---	---

In Bruce-006, “[h]ost requests for access to flash storage are first re-mapped by unified re-map table 20. These host requests contain a logical address which includes [what Bruce-006 calls] a logical-block address (LBA) and an offset address that identifies a page within a flash block.” (INTEL-1005, 6:12-17.) However, only “blocks rather than pages are re-mapped by the [Bruce-006] re-map table.” (INTEL-1005, 5:63-64.) Thus, an LBA entry in Bruce-006’s re-map table corresponds to and represents the grouping of logical addresses for the consecutive pages in the re-mapped physical block. (INTEL-1005, 6:19-20 (“Unified re-map table 20 contains entries preferably for all LBA’s.”); INTEL-1003, ¶116-17.) As such, the Bruce-006 LBAs qualify as the claimed LBA sets (or groupings of consecutive logical addresses). (INTEL-1003, ¶116-17.) Annotated Figure 7A below shows a re-map table with “7 sequential logical block addresses (LBA’s).” (INTEL-1005, 8:1-6.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

90

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Annotated-Figure 7A.) Bruce-006 thus discloses first and second logical fields (highlighted **orange** and **purple**, respectively) that represent first and second LBAs. (INTEL-1003, ¶¶114-19.)

That conclusion is reinforced by Bruce-006's teaching of combining the re-map table with block-interleaving. (INTEL-1005, 11:19-12:10.) "With . . . block interleaving, a sequence of four blocks are interleaved among four flash devices." (INTEL-1005, 11:37-38.) For block-interleaving, "[t]he LBA is the address bits above the page within a block bits." (INTEL-1005, 11:67-12:8.) Because the four bits in positions 3:0 correspond to pages within a block, Bruce-006 explains that only "[b]its 31:4 are routed to the re-map table for block interleaving." (INTEL-1005, 12:4-8.) Here again, Bruce-006 discloses LBAs in the logical fields of the re-map table that correspond to and represent the logical addresses for 16 consecutive pages within the remapped flash block. (INTEL-1003, ¶118.)

Bruce-006's disclosures also match the illustrative sizes for LBAs (i.e., logical addresses) and LBA sets (i.e., groupings of logical addresses) in the '740 patent. In Bruce-006, "[a] block is typically composed of 16 pages[, and a] page is usually 512 bytes in size." (INTEL-1005, 5:64-65.) With the block-based re-mapping of Bruce-006, the disclosed logical addresses for pages (which correspond to LBAs in the '740 patent) would be 512 bytes each in size and the Bruce-006 LBAs (which correspond to LBA sets in the '740 patent) would be 8 Kilobytes in size (512 bytes per page x 16 pages in a block = 8KB). (INTEL-1003, ¶120.) In one example, the '740 patent suggests that each LBA (Bruce-006's logical address) could have a size of 512 bytes and that an LBA set could be associated with 8 LBAs, yielding an LBA set size of 4 Kilobytes. (INTEL-1001, 7:22-25.) The '740 patent goes on to explain that the LBA set size could alternatively be set to any integer multiple of a native page size. (INTEL-1001, 7:22-25.) The '740 patent therefore teaches that there could be more than 8 LBAs in an LBA set, such as 16 LBAs. (INTEL-1003, ¶121.) Thus, an LBA set size of 8 Kilobytes as taught in Bruce-006 is within the set sizes the '740 patent discloses. (INTEL-1003, ¶121.)

3. "Set of PBA Fields" [1B]

Bruce-006 discloses and renders obvious "*a set of PBA fields, including a first PBA field and a second PBA field, said set of PBA fields respectively disposed*

for representing a set of PBAs, including a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters, said PBAs each associated with a physical memory location in a memory store, said set of logical fields and said set of PBA fields disposed to associate said first and second LBA sets with said first and second PBAs.” (INTEL-1003, ¶¶124-133.)

“Each entry in [Bruce-006’s] unified re-map table contains a physical block address (PBA) of the flash memory allocated to the logical address.” (INTEL-1005, Abstract.) As shown in annotated Figure 7A below, Bruce-006 discloses an exemplary re-map table with “7 LBA’s [that] are mapped to various physical blocks in flash memory by the PBA fields.” (INTEL-1005, 8:5-6.) Thus, Bruce-006 discloses a mapping table with multiple PBA fields [highlighted **red**], including a first PBA field [highlighted **brown**] and a second PBA field [highlighted **light blue**], that are mapped to or associated with (respectively) the first LBA set [highlighted **orange**] and the second LBA set [highlighted **purple**] as discussed above (p.15-18). (INTEL-1003, ¶¶124-27.)

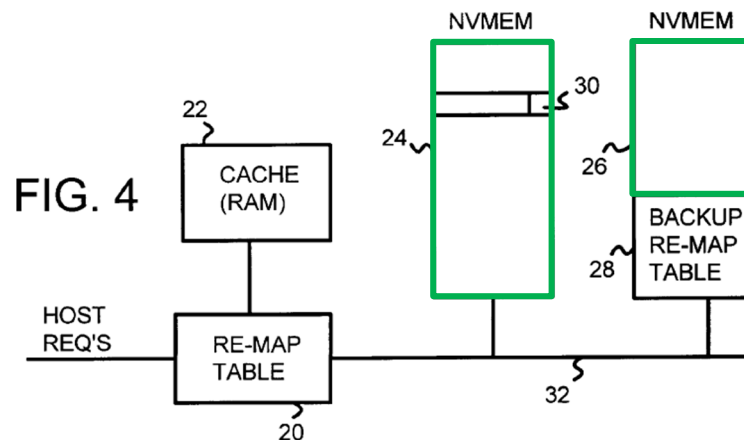
LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

TOT_THRESH = 250,000
 INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Annotated-Figure 7A.)

Bruce-006 also discloses first and second PBAs that are “associated with a physical memory location in a memory store.” (INTEL-1003, ¶¶125-26.) Figure 4 is a block diagram of Bruce-006’s flash-memory system with “several non-volatile (NV) flash memories 24, 26” (highlighted green) that “can be individual flash-memory chips.” (INTEL-1005, 6:6-8, 6:38-41.)



(INTEL-1005, Annotated-Figure 4.) The non-volatile flash memories 24, 26 correspond to the claimed “memory store.” (INTEL-1003, ¶¶125-26.)

In Bruce-006, “a **PBA field** in the entry retrieved from unified re-map table 20 is used to **access the data in flash memory.**” (INTEL-1005, 6:36-38.) The “physical block address **PBA field** [in the re-map table] contains the physical address of the **flash memory device** containing the data of the LBA.” (INTEL-1005, 6:60-62.) “This physical address includes a portion of the address that identifies a **flash memory chip or card**, and another portion that identifies a **block within a flash chip.**” (INTEL-1005, 6:62-65.) Thus, POSITAs would understand that the PBAs disclosed in Bruce-006 are associated with physical memory locations in a memory store. (INTEL-1003, ¶¶125-26.)

Bruce-006 also discloses first and second PBAs each “*disposed for representing a [] set of access parameters*” in its disclosures about “[i]nterleaving of blocks across four devices.” (INTEL-1005, 11:19-26, 11:31-40, 11:67-12:10.) Specifically, Bruce-006 discloses that for four-device block-interleaving, the physical addresses are assigned so that bits 31 to 18 denote the “device within a bank; bank within the system, etc.”; bits 17 to 6 denote the “block within a flash device”; and bits 5 to 4 denote the “interleave among [the] four devices.” (INTEL-1005, 11:19-26, 11:31-40.) Bruce-006 further teaches that each entry in the re-map table also “contains a physical block address PBA field which ... includes a portion of the address that identifies a **flash memory chip or card**, and another portion that identifies a **block** within a flash chip.” (INTEL-1005, 6:60-65; *see*

also, INTEL-1005, 2:65-67, 16:3-9.) Bruce-006 also teaches that when block-interleaving is used, “each entry contains a physical-block-address for an interleave of at least four physical blocks, the at least four physical blocks being in at least four of the flash memory chips.” (INTEL-1005, 15:53-59, 17:6-12.) POSITAs would therefore understand from these teachings that bits [31:4] of the physical addresses for block-interleaving are included in the PBAs in the re-map table. (INTEL-1003, ¶¶127-30.) POSITAs would further understand that at least bits 31 to 18 and bits 5 to 4 of the PBAs are parameters for accessing the physical flash device that the re-map table of Bruce-006 associates with the corresponding LBAs. (INTEL-1005, 11:19-26, 11:31-40, 11:67-12:10; INTEL-1003, ¶131.)

4. “Causes Optimized Memory Operations” [1C]

Bruce-006 discloses and renders obvious “*wherein, in response to receiving the I/O transaction request, said mapping table causes the electronic storage device to perform optimized memory operations on memory locations respectively associated with said first PBA and said second PBA, if the I/O transaction request is associated with said first and second LBA sets.*” (INTEL-1003, ¶¶134-40.)

As explained (p.11-12), Bruce-006 teaches using a re-map table to perform multiple memory operations in response to receiving a write request initiated by a host that triggers wear-leveling. At least three memory operations are required to swap the data between PBAs 101 and 11 shown in Figure 7A : 1) a read memory

operation to transfer the data from the lightly-used PBA 101 (highlighted **brown**) to the cache; 2) a write memory operation to “move[] the data for the heavily-used LBA 011 to the lightly-used PBA 101 [(highlighted **brown**)]”; and 3) a write memory operation to move the data from cache to the heavily-written PBA 111 (in **light blue**). (INTEL-1005, 8:35-39; INTEL-1003, ¶¶135-36.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

90

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Annotated-Figure 7A.) These memory operations are on memory locations associated with a first PBA (highlighted **brown**) and a second PBA (highlighted **light blue**). Figure 7B below illustrates the revised mapping table “after the wear-leveling swap of LBA’s 010 [(highlighted **orange**)] and 011 [(highlighted **purple**)].” (INTEL-1005, 8:40-41.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	111	250,002	1
011	101	20,001	1
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	313,001	25,001

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7B

92

(INTEL-1005, Annotated-Figure 7B.) Thus, the write request initiated by a host to LBA 011 is associated with a first LBA set (highlighted orange) and a second LBA set (highlighted purple). (INTEL-1003, ¶¶135-36.)

Moreover, as explained (p.13-14), Bruce-006 discloses combining the re-map table with block-interleaving. “With such block-interleaving, a **sequence** of four blocks are interleaved among four flash devices,” and “[p]erformance is improved when **sequential blocks** are accessed and the flash devices are slow.” (INTEL-1005, 11:38-40.) As shown in Figure 7B above, the wear-leveling operations described are in response to a write request initiated by a host that is associated with adjacent LBA sets 010 and 011. Given Bruce-006’s teaching that interleaving sequential block accesses can lead to performance improvement by allowing parallel memory accesses, POSITAs would understand Bruce-006 to teach that interleaving the corresponding wear-leveling memory operations among

different flash devices increases the likelihood that those operations can be distributed across different resources and proceed in parallel. (INTEL-1005, 11:20-26, 15:53-59; INTEL-1003, ¶¶137-39.) This is how optimizing memory operations is described in the '740 patent. (INTEL-1001, 1:16-24, 3:12-21.)

Additional support comes from Bruce-006's teachings about address bit assignment for block interleaving. Although Figure 7A and other figures have LBA sets and PBAs with only three or four bits, Bruce-006 explains that "[i]n actual systems, the LBA's and PBA's have more address bits." (INTEL-1005, 6:3-5; INTEL-1003, ¶137.) For PBAs with more physical address bits, Bruce-006 provides the following bit assignment:

3:0 Page within a block
5:4 Interleave among four devices
17:6 Block within a flash device
31:18 Device within a bank; bank within the system, etc.

(INTEL-1005, 11:31-37.) As explained above (p.21-22), for block interleaving, POSITAs would understand Bruce-006 to include physical address bits [31:4] in the PBAs of the re-map table. (INTEL-1003, ¶¶137-39.) The lower two bits of the PBA fields correspond to bits "5:4" of the physical address, which identify one of the four interleaved flash devices that is associated with each LBA set. (INTEL-1003, ¶¶137-39.) Referencing Figure 7A below, for the wear-leveling memory operations described above, the write memory operation to move the data from

cache to the heavily-written PBA 111 would be routed to a flash device identified such as “11,” (the lower two bits of 111) while the write memory operation to write the data for the heavily-used LBA from the heavily-used PBA to the lightly-used PBA 101 would be routed to a different flash device such as “01” (the lower two bits of 101). (INTEL-1003, ¶¶137-39.)

LBA	PBA	TOT_WR'S	INCR_WR'S
001	001	50,000	50,000
010	101	20,000	20,000
011	111	250,001	250,001
100	011	350,000	15,000
101	100	125,000	125,000
110	000	200,000	200,000
111	010	300,000	12,000

90

TOT_THRESH = 250,000
INCR_THRESH = 25,000

FIG. 7A

(INTEL-1005, Annotated-Figure 7A.) Bruce-006 teaches that by having the two memory operations directed at different flash devices, “multiple accesses can occur in parallel to the flash devices, improving performance.” (INTEL-1005, 11:21-22.) This again is how optimized memory operations are described in the ’740 patent. (INTEL-1001, 2:14-21.)

Lastly, Bruce-006 teaches that it is the re-map table that causes the optimized memory operations to be performed in response to a single I/O transaction request initiated by a host. Bruce-006’s re-map table includes the “dual

write counters ... along with address translation information.” (INTEL-1005, 5:14-16.) Bruce-006 then explains that “[e]ach time a block is written, its total-write count is compared to a total-write threshold, while the block’s incremental-write count is compared to an incremental-write threshold.” (INTEL-1005, 4:53-5:5.) When both the total-write count and incremental-write count exceed the system-wide thresholds, wear-leveling is performed. (INTEL-1005, 7:34-54.) Thus, the re-map table in Bruce-006 causes multiple memory operations to be performed in response to receiving an I/O transaction request initiated by a host. (INTEL-1003, ¶¶133-140.) Thus, Bruce-006 discloses and renders obvious claim limitation [1C]. (*Id.*)

C. Dependent Claim 14

Claim 14 recites “*The mapping table of claim 1, wherein said first and second PBAs respectively include a first group identifier and a second group identifier.*”

As explained for claim limitation [1B] (p.21-22), Bruce-006 discloses a mapping table with “*a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters.*” As further explained above (p.20-22), Bruce-006 discloses a flash-memory system with “several non-volatile (NV) flash memories [], which can be individual flash-memory chips” and teaches using bits of the PBA fields as access

parameters for identifying one of the four interleaved flash devices (bits [5:4] for block-interleaving) and “[d]evice within a bank, bank within the system, etc.” (bits [31:8] for block-interleaving). (INTEL-1005, 6:6-8, 6:38-41, 11:19-26, 11:31-40.) POSITAs would understand Bruce-006’s teaching of using bits in the PBAs for identifying a “[d]evice within a bank” and “bank within the system” to teach that each bank includes multiple flash devices. (INTEL-1003, ¶¶141-43.) Therefore, the bits of the PBAs used for identifying a “bank within the system” in Bruce-006 correspond to the claimed “group identifier” as they are used to identify a group of flash devices in a bank. (*Id.*) As such, because Bruce-006 discloses a mapping table with “a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters,” Bruce-006 also discloses and renders obvious “*wherein said first and second PBAs respectively include a first group identifier and a second group identifier.*” (*Id.*)

VI. GROUND 2: Claims 1 and 14 Are Obvious Over Bruce-006 Combined With SBC

A. SBC Overview

SBC is an American National Standards Institute (“ANSI”) standard that was published in 1998, making it prior art under pre-AIA 35 U.S.C. § 102(b). SBC was not disclosed to the USPTO during prosecution of the ’740 patent.

SBC relates to “SCSI” or the “Small Computer System Interface,” which is a standardized interface that allows an application running on a small computer system (e.g., a personal computer) to communicate with peripheral devices. (INTEL-1003, ¶¶145-46.) The SBC “standard defines the command set extensions to facilitate operation of SCSI block devices.” (INTEL-1007, 1.) A “block device” in the SBC standard is “[a] device that is capable of containing data stored in blocks that have a unique logical block address.” (INTEL-1007, 3.) One of the objectives for the standard is to “[p]ermit an application client [which is part of a host] to communicate with a logical unit that declares itself to be a direct-access device” (INTEL-1007, 1, 4.) SBC identifies flash to be a typical variation of the direct-access block device. (INTEL-1007, 15-16.) SBC defines numerous “commands unique to the type of SCSI block devices,” including READ(10) and WRITE(10) commands for direct-access block devices. (INTEL-1007, 1, 28-29, 40-41, 56-57.) Both the READ(10) and WRITE(10) commands allow for large data transfer by specifying the number of contiguous logical blocks of data in the TRANSFER LENGTH field. (INTEL-1007, 40-41, 56-58; INTEL-1003, ¶¶147-49.)

B. Motivation to Combine Bruce-006 with SBC

POSITAs were motivated to configure Bruce-006's electronic storage device to receive host requests formatted as SBC's SCSI block commands. (INTEL-1003, ¶¶150-61.)

SBC is a 1998 ANSI standard that defines the SCSI command set extensions to facilitate I/O requests from a host to a "SCSI block device"—i.e., one that conforms to SBC and addresses data "in groups referred to as logical blocks." (INTEL-1007, 1, 4, 6-7; INTEL-1003, ¶151.) Block addressable flash-memory systems like Bruce-006 are expressly supported. (INTEL-1007, 15-16.)

Bruce-006 teaches a flash-memory system "useful as a mass-storage replacement to a hard disk on a personal computer" and identifies as an example "SCSI hard disks," which happen to have 512-byte sector size matching typical flash page size. (INTEL-1005, 1:21-25, 5, 64-65, 6:7-10.) Thus, to implement the Bruce-006 system as a SCSI hard disk replacement, POSITAs would recognize the need to select the SCSI format/interface for host requests. (INTEL-1003, ¶¶153-54.) As it was well-known at the Earliest Priority Date that SCSI storage devices implemented the SBC block commands, a POSITA would have been motivated to combine SBC's disclosure of SCSI block commands with Bruce-006. (INTEL-1003, ¶157.)

POSITAs would also have been motivated to supplement Bruce-006's known device (flash-memory system) with the interface logic needed to use SBC's SCSI command set because SBC expresses the benefit of doing so (providing **"efficient peer-to-peer operation"** between host and flash storage). *KSR Intern. Co. v. Teleflex Inc.*, 550 U.S. 398, 418-419 (2007). POSITAs would have known the combined system would realize the benefit. (INTEL-1003, ¶¶152-58.)

Moreover, Bruce-006 depicts its flash-memory system receiving host I/O transaction requests without specifying the request format/interface. (INTEL-1005, Abstract, 2:62-64, 5:39-42, 6:7-13, 8:45-47, Claims 1, 10, 17.) POSITAs would therefore be motivated to look to SBC for needed format/interface details. (INTEL-1003, ¶¶152-58.)

Further motivation to combine exists because the combination involves applying a known technique or method (SBC's SCSI command set) to Bruce-006's known device (flash-memory system) that is ready for improvement (efficient peer-to-peer operation by an OS by block transfer) to yield predictable results. *KSR*, 550 U.S. at 415-417.

POSITAs would have had a reasonable expectation of success with predictable results implementing SBC's SCSI block commands with Bruce-006's flash-memory system. SBC is specifically designed for use with block devices such as Bruce-006's flash-memory system. (INTEL-1007, 1, 15-16 (SBC

identifies flash as a typical variation of a direct-access block device).) SBC demonstrates that performing I/O transactions between a host and flash-memory system was well known in the art. (INTEL-1003, ¶¶151-52.) Moreover, Bruce-006 requires that the host requests include a logical address but otherwise imposes no restriction on the protocol used. (INTEL-1005, 5:39-42, 6:12-17.) SBC teaches host requests that include a logical address. (INTEL-1007, viii, 1, 8; INTEL-1003, ¶¶151-57.) POSITAs would have thus recognized that there are no technical barriers to incorporate SBC's command set into Bruce-006's flash-memory system. (INTEL-1003, ¶161.) Thus, POSITAs would have had a reasonable expectation of success in making the combination. (INTEL-1003, ¶¶159-60.)

C. Independent Claim 1

1. Preamble [1P]

As ground 1 explains (p.8-14), Bruce-006 alone discloses and renders obvious “[a] *mapping table for optimizing memory operations performed by an electronic storage device in response to receiving an I/O transaction request initiated by a host.*” Bruce-006 combined with SBC also renders obvious the portion of the preamble (if limiting) that refers to optimizing more than one memory operation performed in response to receiving an I/O transaction request initiated by a host. (INTEL-1003, ¶¶162-71.)

As explained (p.30-32), POSITAs would have been motivated to combine Bruce-006 with SBC. For block devices (like Bruce-006's flash-memory system), SBC teaches that "[a]n application client issues WRITE commands to store the blocks of data (write operations) and READ commands to retrieve the blocks of data (read operations)." (INTEL-1007, 7.) SBC's READ(10) command includes the fields in Table 15. (INTEL-1007, 28-29, 40.)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (28h)							
1	Reserved			DPO	FUA	Reserved		RELADR
2	(MSB)							
3								
4								
5	LOGICAL BLOCK ADDRESS							(LSB)
6	Reserved							
7	(MSB)							
8	TRANSFER LENGTH							(LSB)
9	CONTROL							

(INTEL-1007, 40.) Setting the "relative address (RELADR) bit [highlighted orange]" to "zero indicates that the LOGICAL BLOCK ADDRESS field [highlighted green] specifies the first logical block of the range of logical blocks [of data] for this command." (INTEL-1007, 37, 41.) The "TRANSFER LENGTH field [highlighted blue] specifies the number of contiguous logical blocks of data that shall be transferred." (INTEL-1007, 41.) Given that the TRANSFER LENGTH field is two bytes (or 16 bits) wide, SBC teaches that a READ(10) command can request a data transfer up to 64K (2^{16}) contiguous logical blocks. (INTEL-1003, ¶¶167-68.) SBC further teaches that when the FUA bit (highlighted

red) is set to one, the data is read from “the specified logical blocks from the media (i.e., the data is not directly retrieved from the cache).” (INTEL-1007, 40.)

As detailed below (p.37), a READ(10) command for reading more than 16 contiguous logical blocks would necessarily result in the flash-memory system traversing multiple sequential rows of the re-map table taught in Bruce-006 and require multiple read operations—at least one for each set of sixteen logical blocks requested as indicated by the SBC command transfer length. (INTEL-1007, 28-29, 39-41; INTEL-1003, ¶¶169-70.) A WRITE(10) command with comparable transfer length would likewise require multiple memory operations to the flash blocks of the system taught in Bruce-006. (INTEL-1007, 28-29, 56-57; INTEL-1003, ¶¶169-70.) Thus, performing more than one memory operation in response to receiving an I/O transaction request initiated by a host was obvious over Bruce-006 combined with SBC. (INTEL-1003, ¶¶163-71.) Other documentary evidence confirms that POSITAs would have known how to convert a single host command for transfer size larger than the size of an entry in the re-map table. (INTEL-1012, 11:16-60, Figure 15; INTEL-1003, ¶171.)

Moreover, with Bruce-006’s teachings that (i) interleaving improves performance through parallel accesses (detailed at p.13-14); (ii) sequential block accesses improve performance; and (iii) using interleave bits in the PBA to interleave “a sequence of four blocks ... among four flash devices” (INTEL-1005,

11:20-26, 11, 39-40, 15:53-59), POSITAs would understand that interleaving memory operations performed in response to a READ(10) or WRITE(10) command that spans multiple sequential LBAs in Bruce-006 among different flash devices would increase the likelihood of distributing memory operations across different resources. (INTEL-1003, ¶¶165-70.) This is how the '740 patent describes optimizing memory operations. (INTEL-1001, 1:16-24, 3:12-21.) Thus, optimizing more than one memory operation performed in response to receiving an I/O transaction request initiated by a host was obvious over Bruce-006 combined with SBC. (INTEL-1003, ¶¶163-71.)

2. “Set of Logical Fields” [1A] / “Set of PBA Fields” [1B]

As explained above (p.14-18 and p.18-22), Bruce-006 discloses and renders these claim limitations obvious.

3. “Causes Optimized Memory Operations” [1C]

Ground 1 explains (p.22-27) why Bruce-006 alone discloses and renders obvious “*wherein, in response to receiving the I/O transaction request, said mapping table causes the electronic storage device to perform optimized memory operations on memory locations respectively associated with said first PBA and said second PBA, if the I/O transaction request is associated with said first and second LBA sets.*” This claim limitation also was obvious over Bruce-006 combined with SBC. (INTEL-1003, ¶¶174-82.)

As detailed (p.30-32), POSITAs would have been motivated to combine Bruce-006 with SBC to use SCSI interface commands like READ(10) and WRITE(10) to achieve large data transfers to or from Bruce-006's flash-memory system. SBC teaches that a host may send a READ(10) or WRITE(10) to a connected block device with a LOGICAL BLOCK ADDRESS field to "specif[y] the logical block where the read [or write] operation shall begin." (INTEL-1007, 39, 40-41, 56-57.) SBC therefore uses the term "logical block address" to refer to the host's logical address. Bruce-006, on the other hand, discloses a host's "logical address which includes a logical-block address (LBA) and an offset address that identifies a page within a flash block." (INTEL-1005, 6:13-17.) Bruce-006 further teaches that each LBA is mapped to a physical block in flash memory and that "[a] [flash] block is typically composed of 16 pages." (INTEL-1005, 5:64-65, 8:5-6.) Thus, SBC's logical block address corresponds to Bruce-006's logical address and 16 logical block addresses in SBC correspond to a LBA in Bruce-006. (INTEL-1007, 39, 41, 57; INTEL-1005, 6:13-17; INTEL-1003, ¶175.)

Bruce-006	SBC
logical address	logical block address
LBA	16 logical block addresses

SBC further teaches that a READ(10) or WRITE(10) command can be used to retrieve or store up to 64K contiguous logical blocks of data (given that the transfer length field is 2-bytes (or 16 bits) wide; and $2^{16} = 64K$). (INTEL-1007, 40-41, 56-57; INTEL-1003, ¶¶176-77.) Since each group of 16 logical block addresses in SBC corresponds to a LBA in Bruce-006, in the combined SBC-Bruce-006 flash-memory system, each transfer of 16 contiguous logical blocks of data in a READ(10) or WRITE(10) command would necessarily result in having the flash-memory system use one row of the re-map table taught in Bruce-006 and require one read or write operation to be performed. (INTEL-1003, ¶¶178-80.) Consequently, in the combined SBC-Bruce-006 flash-memory system, receiving a READ(10) or WRITE(10) command with the RELADR (relative address) bit set to 0, the FUA (force unit access) bit set to 1, and the TRANSFER LENGTH set to greater than 16 would necessarily result in having the flash-memory system use two or more sequential LBA entries in Bruce-006's re-map table. (INTEL-1007, 40-41, 56-57; INTEL-1003, ¶¶178-81.) As Bruce-006 teaches using the re-map table to lookup the PBA for each LBA and using each PBA to access the data in flash memory, a READ(10) or WRITE(10) command that spans two or more sequential LBAs in Bruce-006 would therefore cause more than one memory operation to be performed on memory locations associated with the PBAs mapped to the two or more LBAs in Bruce-006's re-map table. (INTEL-1005, 5:33-35,

5:42-43, 6:35-42; INTEL-1003, ¶¶178-81.) Thus, Bruce-006 combined with SBC renders obvious the claim limitation requiring “*wherein, in response to receiving the I/O transaction request, said mapping table causes the electronic storage device to perform ... memory operations on memory locations respectively associated with said first PBA and said second PBA, if the I/O transaction request is associated with said first and second LBA sets.*” (INTEL-1003, ¶¶174-82.)

As explained above in claim 1’s preamble (p.32-35), Bruce-006 combined with SBC also renders obvious “*perform[ing] optimized memory operations.*” More specifically, when block-interleaving across four flash devices as taught in Bruce-006 and the flash-memory system receives a WRITE(10) command to store more than 16 pages of data, POSITAs would have known that each succeeding PBA in the re-map table would be mapped to the next of four flash devices in repeating order through the lower two bits of the PBA fields—i.e., bits “5:4” of the above physical address bit assignment. (INTEL-1005, 11:31-40, 15:48-59; INTEL-1003, ¶¶180-81.) This mapping is required to realize the performance improvement taught in Bruce-006. (INTEL-1005, 11:20-26; INTEL-1003, ¶¶180-81.) It would have been equally obvious to POSITAs that a similarly configured re-map table would be used in response to a READ(10) command in the combined SBC-Bruce-006 flash-memory system. (INTEL-1003, ¶¶180-81.) Thus, in response to receiving an I/O transaction request (i.e., a READ(10) or WRITE(10)

command with transfer length greater than 16), the combined SBC-Bruce-006 flash-memory system renders obvious a mapping table that causes the flash-memory system to perform optimized memory operations on memory locations respectively associated with two PBAs, if the I/O transaction is associated with two LBA sets. (INTEL-1003, ¶¶174-82.)

D. Dependent Claim 14

As explained (p.27-28), Bruce-006 discloses “*wherein said first and second PBA respectively include a first group identifier and a second group identifier.*” (INTEL-1003, ¶183.) Thus, Bruce-006 combined with SBC also renders this claim obvious.

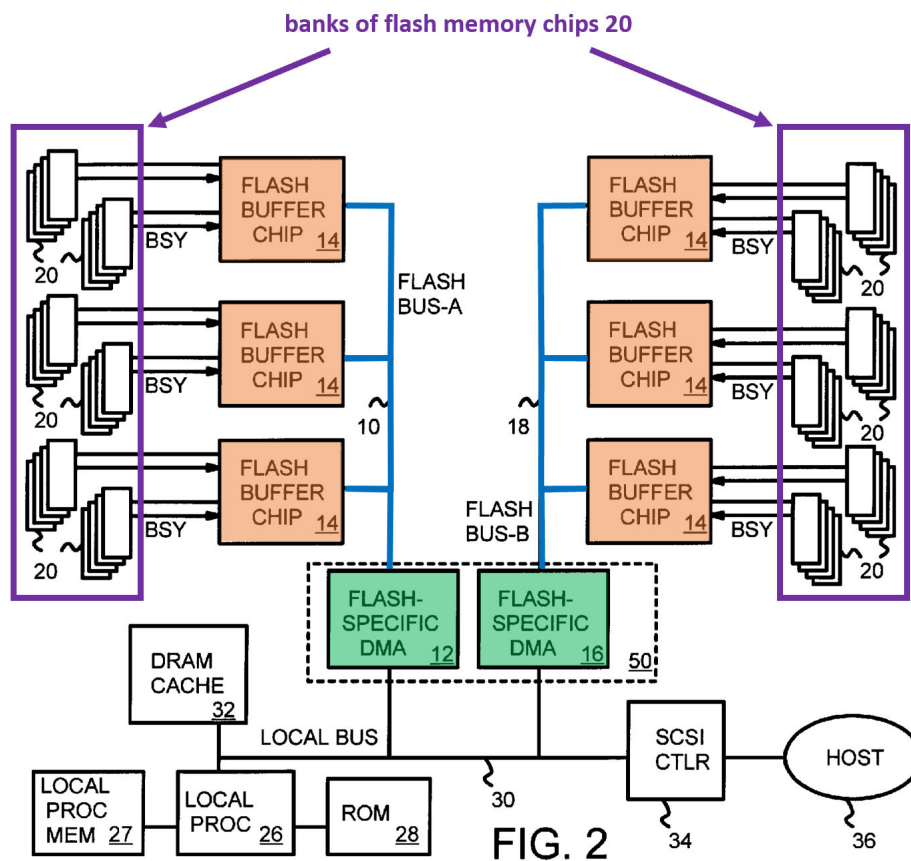
VII. GROUND 3: Claims 1, 9-15, 32, and 34 Are Obvious Over the Combination of Bruce-006 and Bruce-251

A. Combination Overview

1. Bruce-251 Overview

Bruce-251 is a CIP of Bruce-006 with three overlapping inventors. As in annotated Figure 2 below, Bruce-251 teaches “an expandable flash-disk system using dual DMA and dual flash busses.” (INTEL-1006, Abstract, 1:9-13, 5:52-53.) “Banks of flash-memory chips 20 [highlighted purple] connect to flash buffer chips 14 [highlighted orange], which are connected to flash busses 10, 18 [highlighted blue].” (INTEL-1006, 5:53-55.) “A great many flash-memory chips can be added since each flash buffer chip can connect to four banks of 8 flash-

memory chips, a total of 32 flash-memory chips per flash buffer chip.” (INTEL-1006, 5:58-61.) Bruce-251 further teaches connecting dual flash busses 10 and 18 to local bus 30 via dual flash-specific DMA controllers 12 and 16 (highlighted green). (INTEL-1006, 5:7-9, 5:18-19, 5:62-67, 6:13-15, Figs. 1 & 2.) The DMA controllers perform memory operations by “sending command and address bytes to the flash-memory chips through the flash buffer chips.” (INTEL-1006, 6:13-22.)



(INTEL-1006, Annotated-Figure 2.)

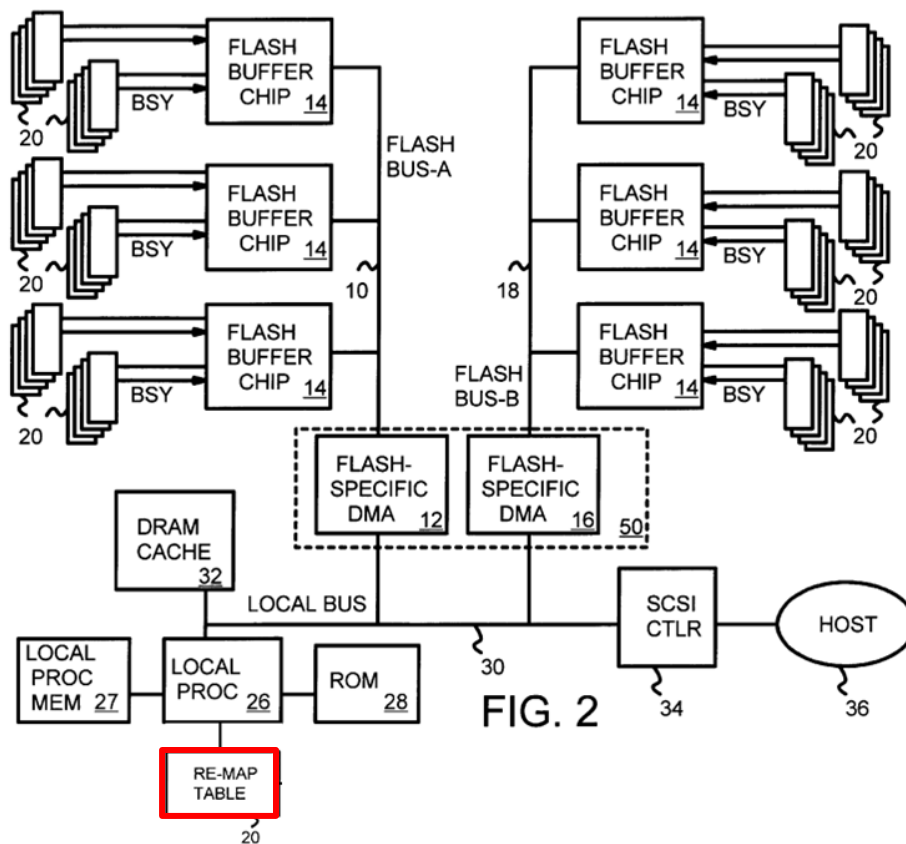
Moreover, Bruce-251 explains that DMA controllers 12 and 16 each “contain[] four state machines 40, one for each of four banks of flash-memory

chips[,] ... allow[ing] four flash-memory chips to be accessed at once for each flash bus, for a total of eight flash accesses.” (INTEL-1006, 6:53-57.) With this expandable flash-memory system, “[t]he data can be interleaved among the four banks for each flash buffer chip, and also interleaved among chips connected to the two flash busses.” (INTEL-1006, 6:57-60.) Additionally, Bruce-251 teaches that “additional flash busses can be added along with additional DMA controllers” and that “DMA controllers could be dynamically assigned to flash busses or to interleaves rather than fixed.” (INTEL-1006, 14:9-12.)

Lastly, Bruce-251 teaches that the expandable flash-disk system includes a local processor 26 “that controls the flash-memory storage peripheral.” (INTEL-1006, 3:48-50; *see also, e.g.*, 5:51-6:15.) The local processor 26 is coupled to a ROM 28, which “contains firmware code of routines that execute on local processor 26 to respond to host requests. Other system-maintenance routines are stored on ROM 28, such as wear-leveling and copy-back routines.” (INTEL-1006, 6:5-9.)

2. Motivation to Combine Bruce-006 and Bruce-251

POSITAs would have been motivated to combine Bruce-006’s unified re-map table (highlighted **red**) with Bruce-251’s expandable flash-memory system as shown in annotated Figure 2 below. (INTEL-1003, ¶¶187-204.)



(INTEL-1006, supplemented-Figure 2.)

First, POSITAs would have been motivated to supplement Bruce-251's flash-memory system with Bruce-006's re-map table to add the table's wear-leveling benefits to Bruce-251. Bruce-251 discloses a local processor for executing firmware routines to perform maintenance, including "wear-leveling routines for re-mapping data from over-used or faulty memory blocks to under-used or unused memory blocks." (INTEL-1006, 3:48-55, 5:51-6:12, 16:59-64.) Bruce-006 discloses a unified re-map table for a flash-memory system that stores "dual write counters ... along with address translation information" for

determining when wear-leveling is required. (INTEL-1005, 2:60-3:14, 4:53-5:20, 7:22-44, 7:58-63). Bruce-006 describes specific advantages of its unified re-map table, including providing a “more efficient and exact wear-leveling scheme [that] minimizes excess writes to flash-memory while re-mapping addresses to pages of flash-memory.” “The total-write counter allows the wear-leveling firmware to determine which blocks have had the most writes and are most likely to wear out,” and “[t]he incremental-write counter prevents thrashing, where a block is continually swapped back and forth during successive wear-leveling operations.” (INTEL-1005, 4:43-61, 5:3-20, 12:12-15, 12:22-24, 12:26-29). By storing dual write counters in the unified re-map table, along with address translation information, “[a] single lookup of the re-map table for a logical address produces the physical block address (PBA) of the data in flash-memory [and] the two write-counter values for wear-leveling,” eliminating the need for separate table accesses to obtain address translation information and wear-leveling information. (INTEL-1005, 12:44-47, 6:6-18.) “This one-step re-mapping is faster for bad pages than the prior-art technique of mechanical mapping followed by reading re-map pointers of bad pages.” (INTEL-1005, 5:18-20.) POSITAs would have been motivated by those expressly suggested benefits to incorporate Bruce-006’s unified re-map table in Bruce-251’s system.

Second, Bruce-251 discloses an expandable flash-memory system that performs wear-leveling and re-mapping of data but does not disclose the implementation details. (INTEL-1006, 3:48-55, 5:51-6:12, 6:5-9.) POSITAs would be motivated to look to the Bruce-006 teachings for implementation details. POSITAs would have been further motivated to turn to Bruce-006 because Bruce-251 is a CIP of Bruce-006 and shares three named inventors as Bruce-006. (INTEL-1006, [63], 1:9-13.) POSITAs would have seen the prominent reference to Bruce-006 in Bruce-251, including Bruce-006's title: "**Unified Re-Map and Cache-Index Table with Dual Write-Counters for Wear-Leveling of Non-Volatile Flash RAM Mass Storage.**" (INTEL-1006, 1:9-13.)

Third, this combination merely amounts to using a known technique (Bruce-006's unified re-map table) to improve a similar system (Bruce-251's expandable flash-memory system) in the same way (to provide faster re-mapping, as well as more efficient and exact wear-leveling). *KSR*, 550 U.S. at 417.

Fourth, this combination also merely amounts to combining prior art elements (Bruce-006's unified re-map table and Bruce-251's expandable flash-memory system) according to known methods (as evidenced by Bruce-006, which shows that POSITAs would have known how to implement Bruce-006's unified re-map table in a similar expandable flash-memory system) to yield predictable results. *KSR*, 550 U.S. at 416-418.

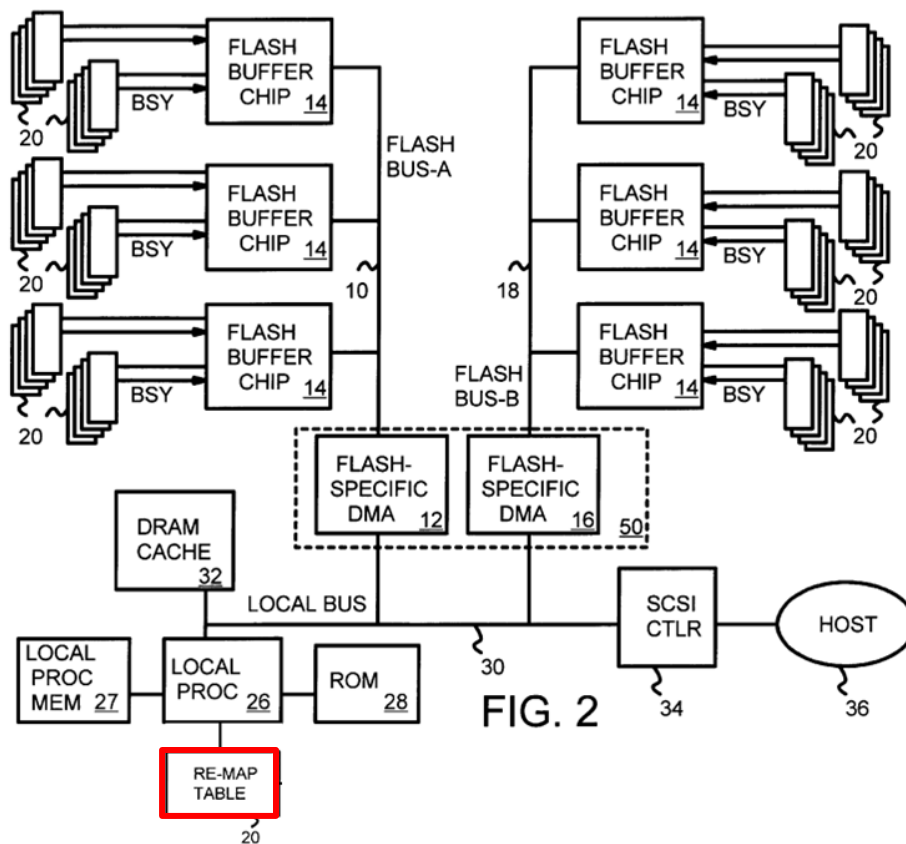
Fifth, POSITAs would reasonably have expected success incorporating Bruce-006's unified re-map table into Bruce-251's expandable flash-memory system. (INTEL-1003, ¶¶203-204.) Bruce-006 demonstrates that POSITAs knew how to implement a re-map table for wear-leveling and data re-mapping in a flash-memory system similar to the expandable flash-memory system disclosed by Bruce-251. (INTEL-1005, 2:60-3:14, 4:53-5:20.) Both Bruce-251 and Bruce-006 implement wear-leveling through firmware routines. (INTEL-1005, 5:3-12, 7:48-51, 8:45-60; INTEL-1006, 3:48-55, 6:5-9.) And Bruce-006 discloses that the unified re-map table is stored in SRAM (INTEL-1005, 6:43-44), which is compatible with storing the unified re-map table in any of a number of memories available to the local processor 26 of Bruce-251's system, including the local processor's local memory 27, DRAM cache 32, or other memories internal to the local processor (INTEL-1006, 6:8-12.) The contents of Bruce-006's unified re-map table further underscore compatibility of the combination. Bruce-006's unified re-map table includes, in its PBA fields, physical address bits used to identify the physical memory location where data may be found, including the bank within a system, and device within a bank. (INTEL-1005, 6:55-65, 11:22-47.) And as described above, Bruce-251's expandable flash-memory system includes multiple banks of flash-memory chips, connected to different flash busses, which receive commands from different DMA controllers. (INTEL-1006, Abstract, 1:9-

13, 5:7-9, 5:18-19, 5:52-61, 6:13-15.) POSITAs would have recognized that bits of the PBA fields of Bruce-006's unified re-map table could be used, once combined with Bruce-251's expandable flash-memory system, to identify physical memory locations in the expandable flash-memory system, just as they operate in Bruce-006. POSITAs would have thus recognized that no technical barriers would impede incorporating Bruce-006's unified re-map table into Bruce-251's expandable flash-memory system. (INTEL-1003, ¶204.)

B. Independent Claim 1

1. Preamble [1P]

If limiting, the preamble is obvious in view of Bruce-251 combined with Bruce-006. As explained in detail above, POSITAs would have been motivated to combine Bruce-006 with Bruce-251 as shown below. (INTEL-1003, ¶¶205-207.)



(INTEL-1006, supplemented-Figure 2.) Bruce-251 teaches that “[t]he invention relates to non-volatile **memory storage** system, and more particularly to expansion of **flash-memory** systems.” (INTEL-1006, 1:18-20; *see also*, INTEL-1006, 16:34-17:35, 18:28-59, 19:13-45.) Thus, the Bruce-006-251 flash-memory system corresponds to the claimed “*electronic storage device*.” (INTEL-1003, ¶205.)

Bruce-251 further teaches a flash-memory system receiving I/O transaction requests that a host initiates and performing read/write memory operations in response. (INTEL-1006, 2:49-55 (“An expandable flash-memory system has a **host interface** that receives **requests** for access of flash-memory. ... The flash-

specific DMA controller generates a sequence of command bytes followed by address bytes to **initiate a data transfer** from the flash-memory.”), 6:1-7 (“**Requests from host** 36 are received by SCSI controller 34”), 6:13-19 (“**Access** of flash-memory chips 20 ... are initiated by ... sending a high-level command to one of flash-specific DMA controllers 12, 16. ... **Many sequences** may be needed, such as for **block reads and writes.**”)) Bruce-251 teaches that its flash-memory system can be a “peripheral.” (INTEL-1006, 3:48-50, 16:54.) Bruce-251 further teaches that “flash-memory systems are frequently used as a mass-storage replacement to a hard disk on a personal computer.” (INTEL-1006, 1:31-34.) POSITAs would therefore understand host read/write requests in Bruce-251 to be “*I/O [input/output] transaction request[s] initiated by a host.*” (INTEL-1003, ¶206.) The host requests in Bruce-251 match the I/O transaction requests described in the ’740 patent. (INTEL-1001, Figure 1, 9:26-27.)

As in ground 1 (p.8-9, 13-14), Bruce-006 discloses and renders obvious “[a] *mapping table for optimizing memory operations*” in response to a host request that triggers wear-leveling operations. In the combined Bruce-006-251 system, Bruce-006’s re-map table would be used to perform memory operations in response to a host-initiated I/O transaction request. Therefore, when a request triggers wear-leveling based on the counters in Bruce-006’s re-map table, the table would cause the combined Bruce-006-251 system to perform optimized memory operations for

the same reasons explained for ground 1 (p.13-14, 22-27.) (INTEL-1003, ¶207.)

Thus, Bruce-006 combined with Bruce-251 renders obvious this preamble, if limiting. (INTEL-1003, ¶¶205-207.)

**2. “Set of Logical Fields” [1A] / “Set of PBA Fields” [1B] /
“Causes Optimized Memory Operations” [1C]**

Bruce-251 discloses a memory store as it teaches an expandable flash-memory system with “[b]anks of flash-memory chips 20.” (INTEL-1006, 5:53.) As in ground 1 (p.14-27), Bruce-006 alone discloses and renders the remaining claim limitations obvious.

C. Dependent Claim 9

Claim 9 recites “*The mapping table of claim 1, wherein said set of access parameters includes a bus identifier, an FDE identifier and a group identifier.*”

As explained for ground 1, limitation [1B] (p.18-21), Bruce-006 discloses a mapping table with “*a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters.*” As further explained above for ground 1, claim 14 (p.27-28), Bruce-006 also discloses that “*said set of access parameters includes a ...group identifier.*” Bruce-006 does not, however, explicitly disclose that said set of access parameters includes a bus identifier or an FDE identifier. But as explained above (p.41-46), it would have been obvious to combine Bruce-006 with Bruce-251 and

the combined Bruce-006-251 system renders this obvious. (INTEL-1003, ¶¶209-217.)

Bruce-251 discloses that “[r]ather than directly connecting individual flash-memory chips to a controller, flash buffer chips are used.” (INTEL-1006, Abstract.) More specifically, “each flash buffer chip can connect to four banks of 8 flash-memory chips.” (INTEL-1006, 5:59-60.) That is, Bruce-251 discloses grouping eight flash-memory chips into one bank and connecting four banks of memory devices to each flash buffer chip. (INTEL-1003, ¶211.) Bruce-251 further describes a flash bus protocol where bits [1:0] of an 8-bit address/data/command signal are used to identify one of the four banks of memory devices associated with each memory operation as shown in the annotated table below:

bit 7	normal = 1, initialization = 0
bits 5:4	00 = data, 01 = command, 10 = address, 11 = disconnect
bit 3	1 = lock sequence
bits 1:0	Interleave selection, one of four banks

(INTEL-1006, 6:13-24, 9:51-57, 10:6-15; *see also*, INTEL-1006, Figure 1, 4:55-5:10.) These two bits correspond to the claimed “group identifier” as they are used to identify a group of eight flash-memory devices in a bank. (INTEL-1003, ¶212.) As explained above (p.21-22), Bruce-006 teaches including physical address bits

[31:18] for denoting “[d]evices within a bank; bank within the system, etc.” in the re-map table’s PBA fields for block-interleaving. (INTEL-1005, 6:60-65.) In the combined Bruce-006-251 system, Bruce-006’s re-map table will use the two-bit group identifier taught in Bruce-251 as the bits in the PBAs for identifying the “bank within the system” taught in Bruce-006. (INTEL-1005, 11:30, 11:36, 11:47; INTEL-1003, ¶212.) As such, Bruce-006 combined with Bruce-251 also renders obvious “*wherein said set of access parameters includes ... a group identifier.*” (INTEL-1003, ¶212.)

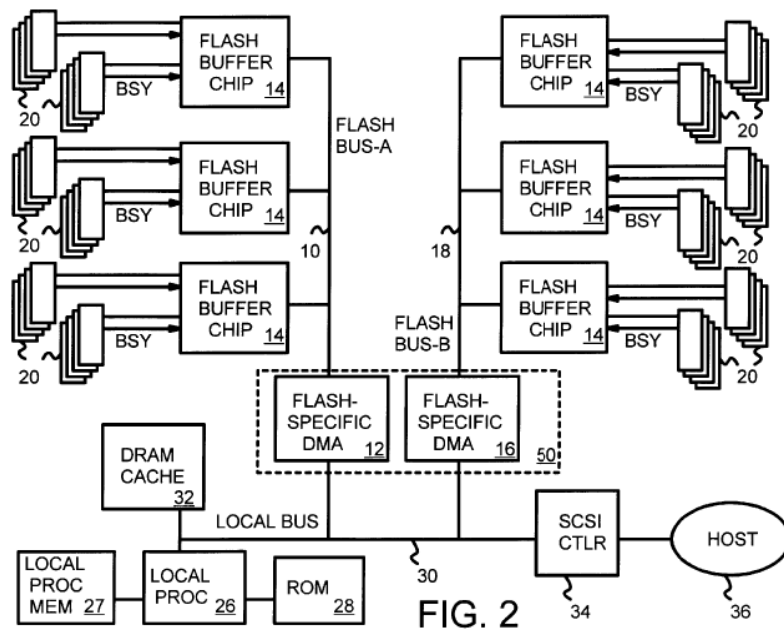
Bruce-251 further teaches “an expandable flash-disk system using ... **dual flash busses.**” (INTEL-1006, 5:52-53.) More specifically, Bruce-251 teaches having “[b]anks of flash-memory chips 20 connect to flash buffer chips 14, which are connected to **flash busses 10, 18.**” (INTEL-1006, 5:53-55.) The use of elements 10 and 18 to identify the two flash busses disclosed in Bruce-251 correspond to the claimed bus identifier in the ’740 patent. (INTEL-1003, ¶213.) Bruce-251’s use of flash busses 10 and 18 for each representing a bus interface mirrors the ’740 Patent’s uses of bus identifiers 8-0 and 8-1. (INTEL-1001, Figure 1, 5:39-41 (“Bus identifier ... is an identifier that is used for representing a bus interface”).)

Bruce-251 also discloses connecting multiple flash buffer chips to “[t]wo flash-specific DMA controllers.” (INTEL-1006, Abstract.) More specifically,

Bruce-251 teaches that “[a]ccesses of flash-memory chips 20 by local processor 26 are initiated by local processor 26 sending a high-level command to one of flash-specific DMA controllers 12, 16. These DMA controllers then generate sequences of command and address bytes, and transfer data.” (INTEL-1006, 6:13-17.) That is, Bruce-251 discloses the use of two flash-specific DMA controllers for accessing different flash-memory chips in an expandable flash-memory system. (INTEL-1003, ¶214.) Since the ’740 patent describes “[a]n FDE [to be] a device that is capable of controlling a flash-memory device and performing DMA operations on the flash-memory device,” the two flash-specific DMA controllers disclosed in Bruce-251 correspond to the claimed FDEs in the ’740 patent. (INTEL-1001, 3:35-39; INTEL-1003, ¶215.)

Bruce-251 additionally discloses alternative embodiments in which “DMA controllers could be dynamically assigned to flash busses or to interleaves rather than fixed.” (INTEL-1006, 14:10-12.) That is, Bruce-251 teaches connecting each flash-specific DMA controller to more than one of the flash busses and configuring the relationship between the DMA controllers and the flash busses on a dynamic basis. (INTEL-1003, ¶216.) POSITAs would understand from the above disclosure that Bruce-251 teaches that both the flash-specific DMA controllers 12 and 16 in Figure 2 below can be connected to flash bus-A 10 and flash bus-B 18. (*Id.*) POSITAs would further understand Bruce-251 to teach that flash-specific

DMA controller 12 can be dynamically configured to communicate via flash bus-A 10 or flash bus-B 18 and that flash-specific DMA controller 16 can also dynamically communicate via flash bus-A 10 or flash bus-B 18. (*Id.*)



(INTEL-1006, Figure 2.) Bruce-251's use of DMA controllers 12 and 16 to identify one DMA controller from another DMA controller in a set of DMA controllers that are coupled to the same bus interface mirrors the '740 Patent's uses of FDE identifiers 8-0 and 8-1. (INTEL-1001, Figure 1, 5:41-44.)

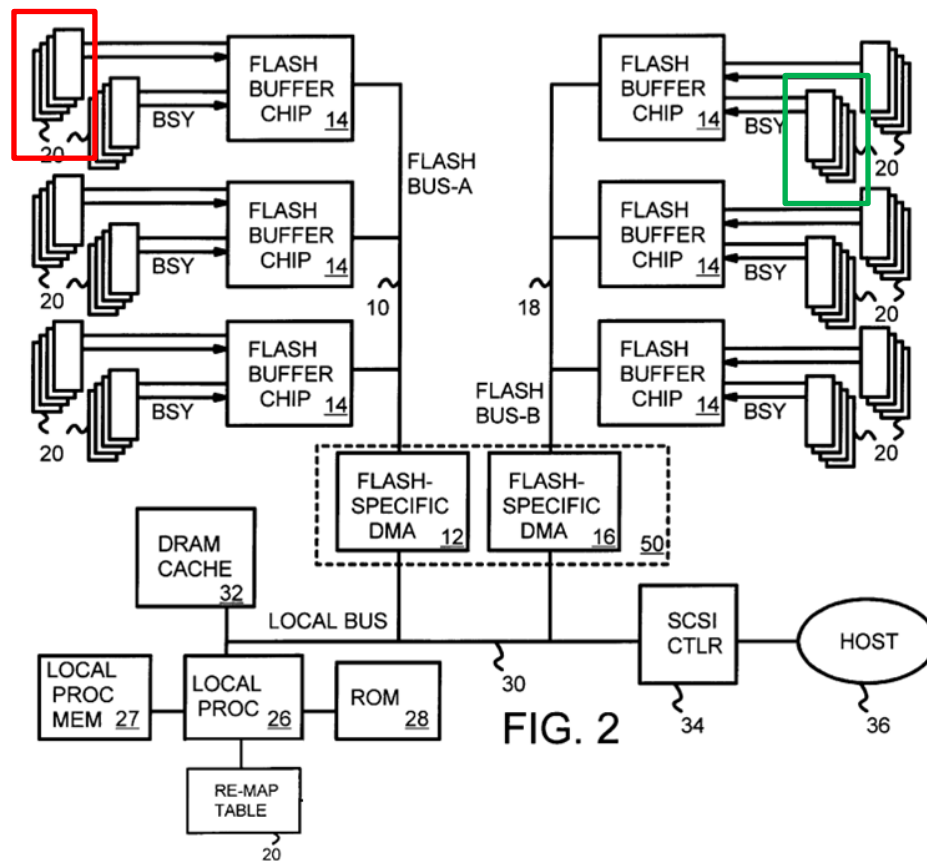
Moreover, given that there is more than one DMA controller 12, 16 on local bus 30 and that more than one flash bus is dynamically assigned to each DMA controller as described above, POSITAs would understand that both the DMA controller number (i.e., 12 or 16) and the flash bus number (i.e., 10 or 18) are needed to identify how a specific buffer chip can be accessed. (INTEL-1006,

Figure 1; INTEL-1003, ¶¶216-17.) Stallmo confirms this understanding by illustrating use of a Controller Number and a Channel (bus) Number to access a storage unit on the identified bus using the identified controller. (INTEL-1010, 4:56-60, 7:44-48, 5-9; *Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1372-73 (Fed. Cir. 2019) (general teachings in reference demonstrate POSITAs' understanding of prior art). Therefore, POSITAs would understand that in the combined Bruce-006-251 system, the high-level command sent by local processor 26 over local bus 30 to the DMA controllers 12, 16 in response to host requests would have contained access parameters including, in addition to the group identifier described above, a DMA controller (FDE) identifier and a flash bus identifier to access particular flash-memory chips 20. (INTEL-1006, 6:1-5, 6:13-15; INTEL-1003, ¶217.) And those access parameters would have necessarily been included in, or at least represented by, the PBAs in the re-map table, consistent with the use of PBA bits in Bruce-006 to identify a “[d]evice within a bank; bank within the system, etc.” (INTEL-1005, 11:30, 11:36, 11:47; *see supra* ground 1, [1B] at p.21-22.) Thus, Bruce-006 combined with Bruce-251 renders obvious “*wherein said set of access parameters includes a bus identifier, an FDE identifier and a group identifier.*” (INTEL-1003, ¶¶209-217.)

D. Dependent Claim 10

Claim 10 recites “*The mapping table of claim 9, wherein said first and second PBA include different bus identifiers, different FDE identifiers and different group identifiers.*”

As explained above for claim 9 (p.49-54), in the Bruce-006-251 system, the PBAs in the re-map table would necessarily include access parameters, including at least a group identifier, a DMA controller (FDE) identifier and a flash bus identifier. POSITAs would therefore understand that in the Bruce-006-251 system, the PBAs used for accessing memory locations in a first bank of chips on flash bus 10 using DMA controller 12 (highlighted **red**) would necessarily include a different group identifier, DMA controller identifier, and bus identifier from the PBAs used for accessing a second bank of chips on flash bus 18 using DMA controller 16 (highlighted **green**). (INTEL-1003, ¶218-19.)



(INTEL-1006, supplemented-Figure 2.)

Bruce-006 further teaches that “[u]nified re-map table 20 contains entries preferably for all LBA’s” and that “logical addresses can be arbitrarily translated into physical addresses for the flash devices.” (INTEL-1005, 6:19-20, 5:44-46.)

“The physical addresses do not have to correspond to the logical addresses”

(INTEL-1005, 5:46-48.) “Since flash blocks can be swapped and frequently-written addresses allocated to different flash blocks over time without regard to the logical address,” POSITAs would understand that there must exist two LBAs in the re-map table in the Bruce-006-251 system with different bus, FDE, and group

identifiers in the corresponding PBAs. (INTEL-1005, 5:48-51; INTEL-1003, ¶219.) Thus, Bruce-006 combined with Bruce-251 renders this claim obvious. (*Id.*)

E. Dependent Claim 11

Claim 11 recites “[t]he mapping table of claim 9, wherein said first and second PBA respectively include **different combinations** of said bus identifiers, said FDE identifiers, and said group identifiers.” Claim 11 is broader than claim 10 which requires different bus, FDE, **and** group identifiers. For the same reasons described above for claim 10, Bruce-006 combined with Bruce-251 renders claim 11 obvious. (INTEL-1003, ¶220.)

F. Dependent Claims 12 and 13

Claims 12 and 13 recite “[t]he mapping table of claim 1, wherein said first and second PBA include different bus (claim 12) / FDE (claim 13) identifiers.” The recited claim limitations here are a part of claim 10 which requires “**wherein said first and second PBA include different bus identifiers, different FDE identifiers and different group identifiers.**” For the same reasons described above for claim 10, Bruce-006 combined with Bruce-251 renders claims 12 and 13 obvious. (INTEL-1003, ¶221-222.)

G. Dependent Claim 14

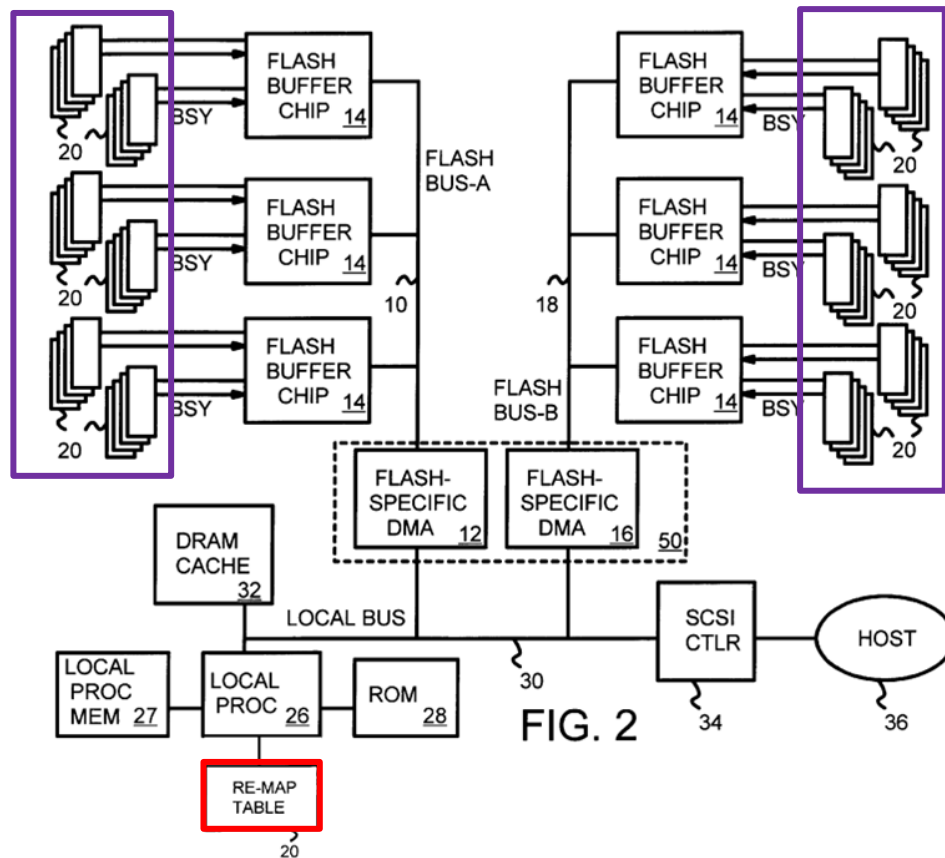
Claim 14 recites “[t]he mapping table of claim 1, wherein said first and second PBAs respectively include a first group identifier and a second group identifier.” The recited claim limitation here is broader than the claim limitation in claim 10 requiring “said first and second PBA include ... **different** group identifiers.” For the same reasons described for claim 10, Bruce-006 combined with Bruce-251 renders claim 14 obvious. (INTEL-1003, ¶223.)

H. Dependent Claim 15

1. “Memory Store” [15A]

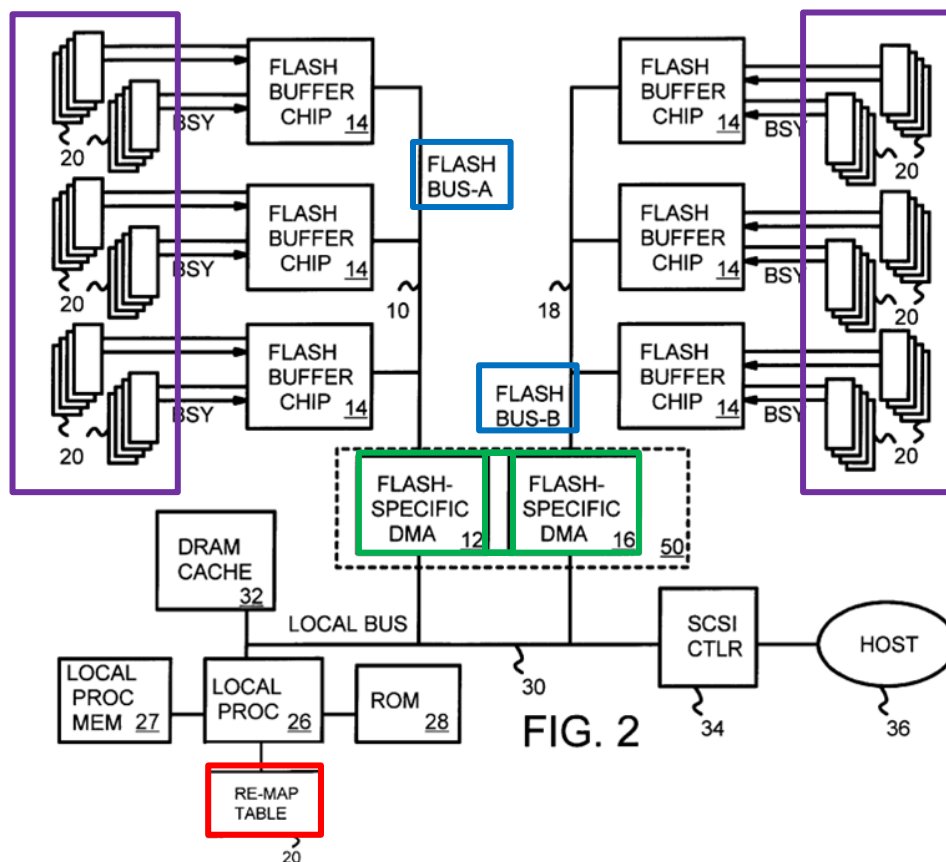
Bruce-006 combined with Bruce-251 renders obvious “[t]he mapping table of claim 1: wherein said memory store includes a set of solid state memory devices that are coupled to a set of buses and that are controlled by a set of FDEs.” (INTEL-1003, ¶224-226.)

As explained above (p.41-46), POSITAs would have been motivated to combine Bruce-006 with Bruce-251 as shown in the figure below. (INTEL-1003, ¶¶187-88.)



(INTEL-1006, supplemented-Figure 2.) And as further referenced above (p.49), Bruce-251 discloses a memory store with “[b]anks of flash-memory chips 20 (highlighted **purple**).” (INTEL-1006, 5:53.) POSITAs would understand all the banks of flash-memory chips (i.e., both purple boxes) taught in Bruce-251 to be “a set of solid state memory devices.” (INTEL-1003, ¶224; *see also*, INTEL-1005, Abstract (“A flash-memory system provides solid-state mass storage.”)) Thus, Bruce-006 combined with Bruce-251 renders obvious “[t]he mapping table of claim 1: wherein said memory store includes a set of solid state memory devices.” (*Id.*)

Bruce-251 further discloses “an expandable flash-disk system using **dual DMA** and **dual flash busses.**” (INTEL-1006, 5:52-53.) As shown in the figure below, the flash-memory chips (highlighted **purple**) are coupled to flash buses 10,18 (highlighted **blue**), and flash-specific DMA controllers 12,16 (highlighted **green**). (INTEL-1006, 5:53-55; *see also*, INTEL-1006, 5:7-9, Figure 1.)



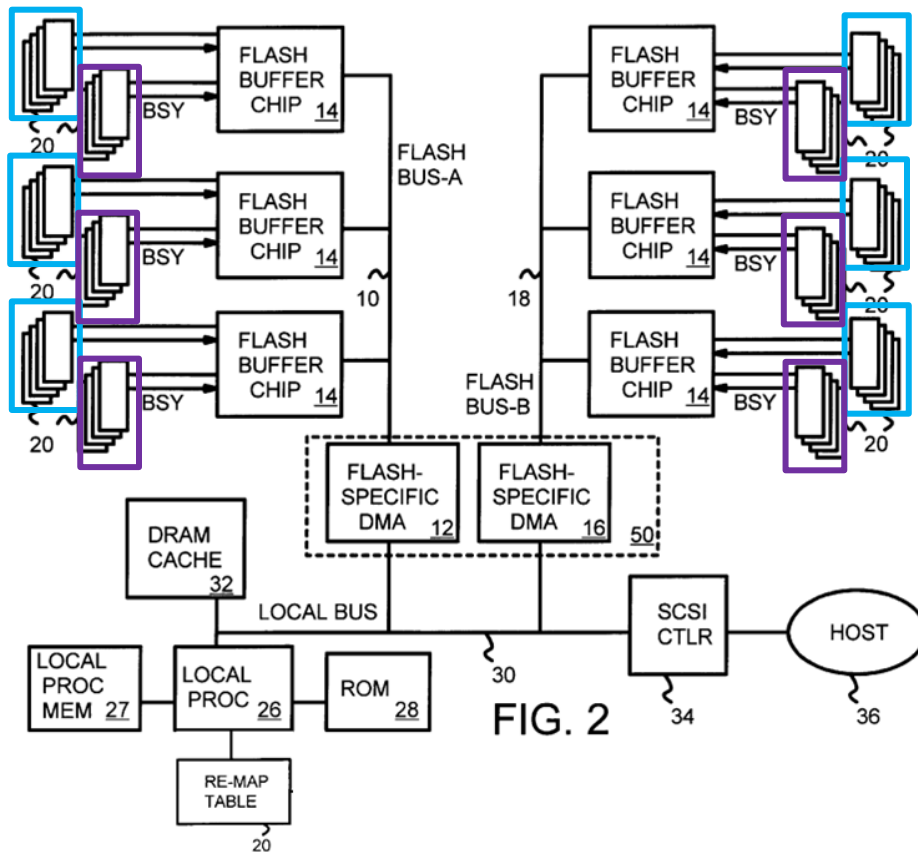
(INTEL-1006, supplemented-Figure 2.) Bruce-251 additionally discloses that “**additional flash busses** can be added along with **additional DMA** controllers.” (INTEL-1006, 14:9-12.) Bruce-251 further teaches the DMA controllers

controlling the flash-memory chips by “sending command and address bytes to the flash-memory chips through the flash buffer chips.” (INTEL-1006, 6:13-22.)

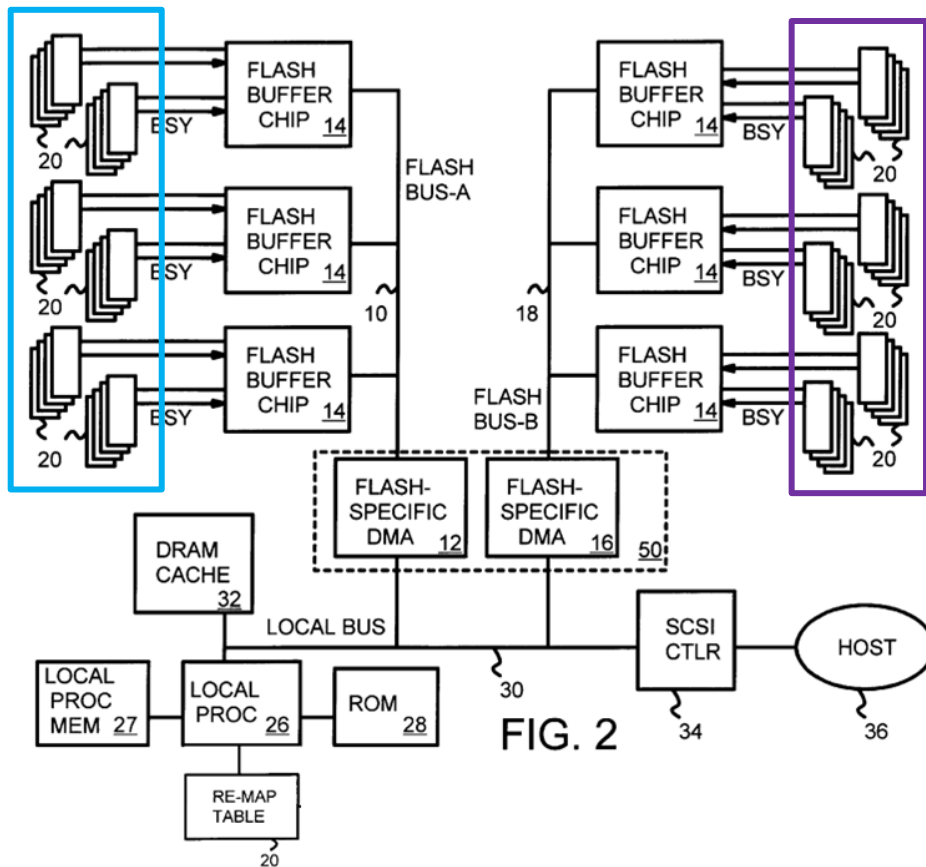
Thus, Bruce-006 combined with Bruce-251 renders this claim obvious. (INTEL-1003, ¶¶225-26.)

2. “Group Identifier for Identifying a Portion of Memory Devices Controlled by a FDE” [15B]

For the same reasons provided above for claim 9 (p.49-54), Bruce-006 combined with Bruce-251 renders obvious “*wherein said access parameter includes a group identifier.*” (INTEL-1003, ¶¶227-28.) Bruce-251 further teaches that each group identifier identifies “*at least a portion of said set of memory devices that are respectively controlled by different FDEs from said set of FDEs.*” As shown in Figure 2 below in which two banks (i.e., two groups) of flash-memory chips are attached to each flash-buffer chip 14 (*see, e.g.*, INTEL-1006, 5:28-29), one bank identifier is used to identify half of all the flash-memory chips (highlighted **light blue**) that are controlled by the two DMA controllers, and a second bank identifier is used for the other half of the flash-memory chips (highlighted **purple**) controlled by the two DMA controllers. (INTEL-1003, ¶228.)



(INTEL-1006, supplemented-Figure 2.) Alternatively, as shown in the figure below, using DMA controllers 12 and 16 to each identify a set of flash-memory chips that are respectively controlled by the two DMA controllers (i.e. DMA controller 12 identifies memory devices highlighted **light blue** while DMA controller 16 identifies memory devices highlighted **purple**) also correspond to the claimed “group identifier” here for the reasons provided above for claim 10 (p.55-57). (INTEL-1003, ¶228.)



(INTEL-1006, supplemented-Figure 2.) Thus, Bruce-006 combined with Bruce-251 renders this claim limitation obvious. (INTEL-1003, ¶¶228-29.)

I. Independent Claim 32

1. Preamble [32P]

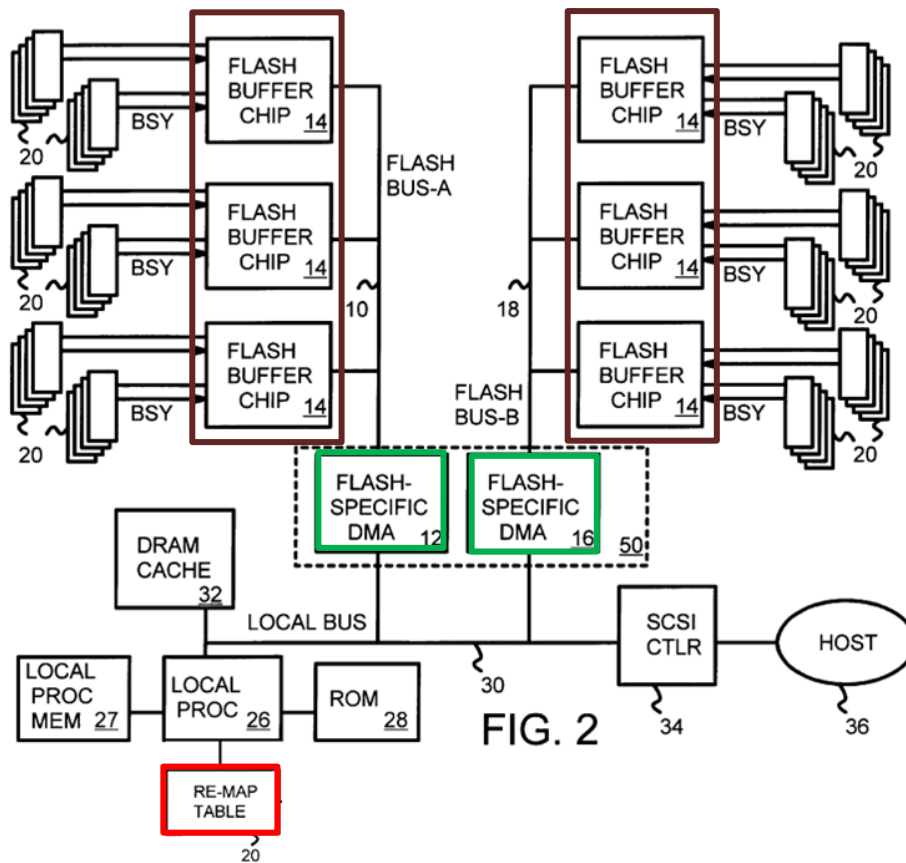
If the preamble is limiting, the combination of Bruce-251 and Bruce-006 renders obvious “[a]n electronic storage device that includes a memory system that uses a memory table for increasing the likelihood that an operational load imposed on the storage apparatus during the processing of an I/O transaction request will be optimally distributed across storage device resources, said I/O

transaction request received by said storage device from a requesting host.”

(INTEL-1003, ¶¶231-34.)

As explained (p.41-46), POSITAs would have been motivated to combine Bruce-006 with Bruce-251 as shown below and the combined Bruce-006-251 flash-memory system corresponds to the claimed “*electronic storage device.*”

(INTEL-1003, ¶¶231-32.)



(INTEL-1006, supplemented-Figure 2.) The re-map table (highlighted **red**) together with the flash-specific DMA controllers 12 and 16 (highlighted **green**) and flash buffer chips 14 (highlighted **brown**) in the combined Bruce-006-251

system correspond to the claimed “*memory system.*” (INTEL-1003, ¶¶231-32.)

This matches the memory system described in the ’740 patent as comprised of a mapping table and a DMA controller that includes a flash buffer controller. (E.g., INTEL-1001, 3:25-31.)

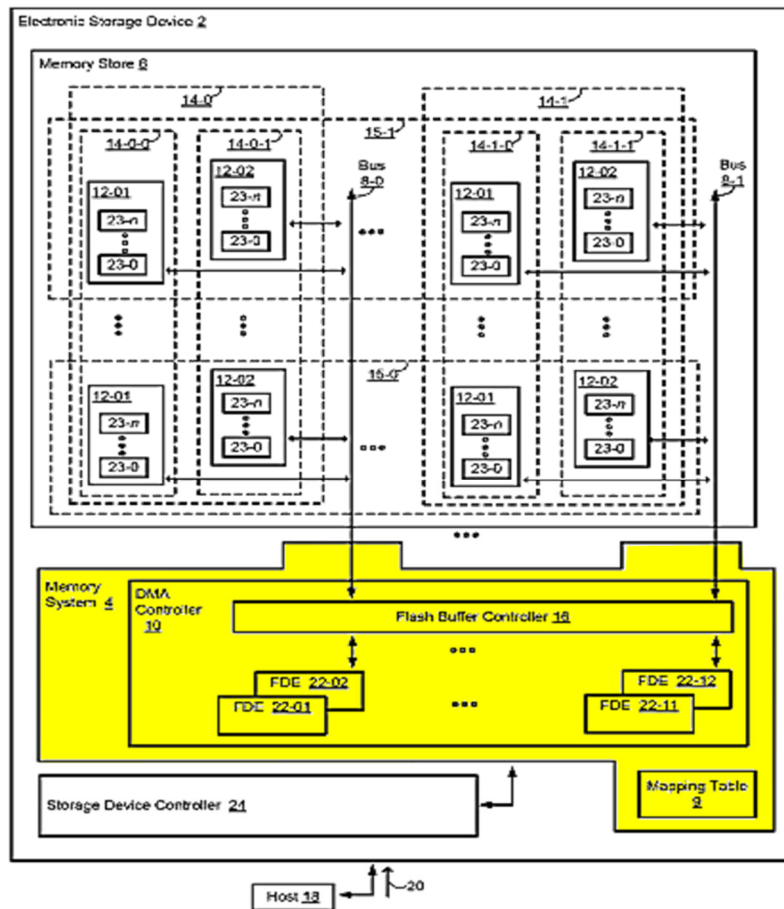


FIG. 1

(INTEL-1001, Annotated-Figure 1.)

The claim limitation here further requiring that the memory system “*uses a memory table ... during the processing of an I/O transaction request ..., said I/O transaction request received by said storage device from a requesting host*” is

substantially similar to claim 1's preamble which requires "[a] *mapping table for ... memory operations performed by an electronic storage device in response to receiving an I/O transaction request initiated by a host.*" Thus, Bruce-006 combined with Bruce-251 renders obvious this portion of the preamble for the same reasons provided at p.47-48. (INTEL-1003, ¶¶233-34.)

Bruce-006 combined with Bruce-251 further discloses the remaining limitation requiring that the mapping table is for "*increasing the likelihood that an operational load imposed on the storage apparatus during the processing of an I/O transaction request will be optimally distributed across storage device resources.*" (INTEL-1003, ¶34.) As explained (p.47-48), Bruce-006's re-map table is used in the Bruce-006-251 system for performing memory operations in response to an I/O transaction request initiated by a host. (INTEL-1005, 11:19, 11:20-22.) As further explained for ground 1's [1C] (p.26-27), Bruce-006's re-map table causes multiple wear-leveling memory operations to be performed on different flash devices in response to an I/O transaction request initiated by a host. (INTEL-1005, 7:66-9:37, Figures 7A-7E.) With Bruce-006's teaching of "**[i]nterleaving ... [such that] multiple access can occur in parallel** to the flash devices," POSITAs would understand that memory operations on different flash devices can be interleaved and the use of Bruce-006's re-map table would therefore cause the operational load imposed on the expandable flash-memory system to be optimally distributed across

different storage resources in the combined Bruce-006-251 system. (INTEL-1006, 11:20-22; INTEL-1003, ¶234.) As further explained above in ground 1's [1C] at (p.24-26), Bruce-006 teaches wear-leveling operations that are associated with sequential LBAs. (INTEL-1005, 11:39-40.) With its teaching of performance improvement from interleaving "when sequential blocks are accessed," POSITAs would understand the use of Bruce-006's re-map table to teach distributing the memory operations associated with sequential LBAs across different storage resources for the performance improvement. (INTEL-1003, ¶104.) Both match the '740 patent's disclosure for optimally distributing operational load imposed on the storage device across different storage resources by interleaving or parallel memory operations. (INTEL-1001, 2:14-19.) Thus, Bruce-006 combined with Bruce-251 discloses the remaining limitation. (INTEL-1003, ¶234.)

2. "First and Second LBA Sets Mapped to First and Second PBAs" [32A]

As explained above for ground 1's [1A] and [1B] (p.14-22), Bruce-006 discloses and renders obvious a mapping table "disposed to associate said first and second LBA sets with said first and second PBAs" where "said first and second LBA sets each representing a set of consecutive LBAs." Bruce-006 combined with Bruce-251 thus renders obvious "*a mapping table which maps a first LBA set to a*

first PBA, and a second LBA set to a second PBA, said first and second LBA sets including respective sets of consecutive LBAs.” (INTEL-1003, ¶235.)

3. “First and Second Physical Memory Locations Addressable by First and Second PBAs” [32B]

As explained above for [15A] (p.59), Bruce-251 discloses an expandable flash-memory system having “[b]anks of flash-memory chips 20.” (INTEL-1006, 5:51-55.) Thus, Bruce-251 discloses “*a memory store that includes a set of flash devices.*” (INTEL-1003, ¶235-36.) And as explained above for ground 1’s [1B] (p.18-22), Bruce-006 discloses and renders obvious “*a set of PBA fields ... said set of PBA fields respectively disposed for representing a set of PBAs, ... said PBAs each associated with a physical memory location in a memory store.*” (INTEL-1005, 6:60-65.) Bruce-006 combined with Bruce-251 thus renders obvious “*a memory store that includes a set of flash devices, said set of flash devices includes a first physical memory location and a second physical memory location that are addressable by said first and second PBAs respectively.*” (INTEL-1003, ¶236.)

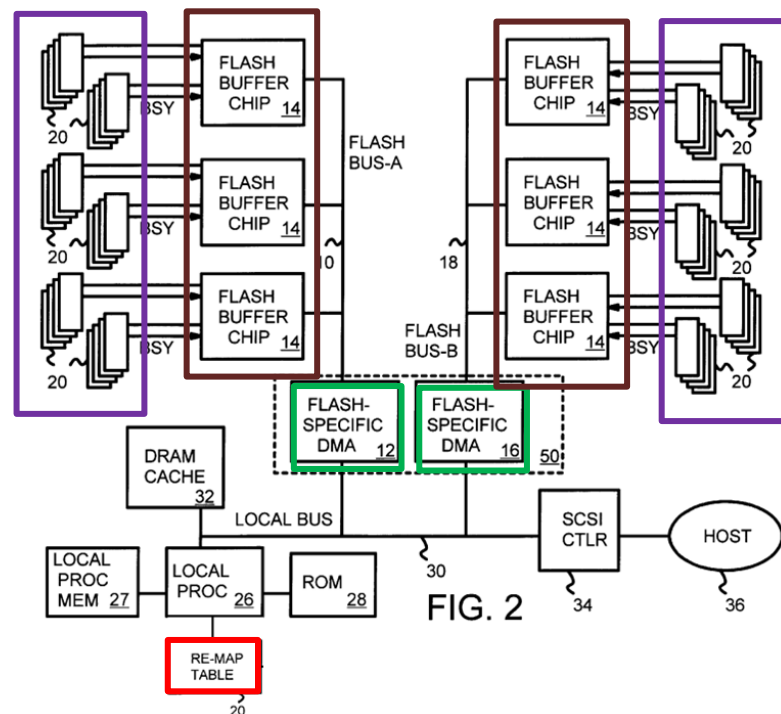
4. “Memory System” [32C]

Bruce-006 combined with Bruce-251 renders obvious “*a memory system coupled to said memory store and disposed to use said mapping table during a memory operation.*” (INTEL-1003, ¶¶237-38.)

The claim limitation requiring “a memory system ... disposed to use said mapping table during a memory operation” is substantially similar to the limitation in [32P] which requires “a memory system that uses a memory table ... during the processing of an I/O transaction request.” Thus, Bruce-006 combined with Bruce-251 renders obvious this portion of the claim limitation for the same reasons described (p.64-66) for claim limitation [32P]. (INTEL-1003, ¶¶237-38.)

Furthermore, as shown below for the combined Bruce-006-251 system, the memory system (i.e., components highlighted **green**, **red**, and **brown**) is coupled to the memory store (i.e., the banks of flash-memory chips highlighted **purple**).

(INTEL-1006, 5:51-55; see also, INTEL-1006, 5:7-9, Figure 1.)



(INTEL-1006, supplemented-Figure 2.) Bruce-251 combined with Bruce-006 therefore discloses that the memory system is *coupled to said memory store*.

(INTEL-1003, ¶¶237-38.)

**5. “One Difference Between Said First and Second PBAs”
[32D]**

As explained (p.18-22) for claim limitation [1B], Bruce-006 discloses and renders obvious “*a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters.*” And as explained (p.49-57) for claims 9 and 10, Bruce-006 combined with Bruce-251 discloses “*said first and second PBA include different bus identifiers, different FDE identifiers and different group identifiers.*” Bruce-006 combined with Bruce-251 thus discloses “*wherein said first PBA includes a first set of access parameters and said second PBA includes a second set of access parameters, and at least one difference between said first and second PBAs.*”

(INTEL-1003, ¶240.)

6. “Optimally Distributed” [32E]

Bruce-006 combined with Bruce-251 renders obvious “*said mapping table increases the likelihood that that [sic] the operational load imposed on the storage apparatus during the processing of the I/O transaction request will be optimally*

distributed across storage device resources” for the same reasons provided above for [32P]. (INTEL-1003, ¶241.)

J. Dependent Claim 34

Bruce-006 combined with Bruce-251 renders obvious “[t]he electronic storage device of claim 32, wherein the processing of the I/O transaction request includes a first memory operation performed using said first PBA and a second memory operation performed using said second PBA; said first and second PBAs respectively associated with a first set of access parameters and a second set of access parameters, and said set of first and second set of access parameters differing by at least one access parameter.”

Bruce-006 discloses and renders obvious “the processing of the I/O transaction request includes a first memory operation performed using said first PBA and a second memory operation performed using said second PBA” for the same reasons provided for limitation [1C] (p.22-27). (INTEL-1003, ¶242.)

As additionally explained (p.18-22) for claim limitation [1B], Bruce-006 discloses and renders obvious “a first PBA disposed for representing a first set of access parameters and a second PBA disposed for representing a second set of access parameters.” Bruce-006 thus additionally discloses “said first and second PBAs respectively associated with a first set of access parameters and a second set of access parameters.” (INTEL-1003, ¶243.)

And as further explained (p.49-57) for claims 9 and 10, Bruce-006 combined with Bruce-251 discloses “*wherein said first and second PBA include different bus identifiers, different FDE identifiers and different group identifiers.*” Bruce-006 combined with Bruce-251 therefore further discloses “*said set of first and second set of access parameters differing by at least one access parameter.*” (INTEL-1003, ¶244.)

VIII. GROUND 4: Claims 1, 9-15, 32, and 34 Are Obvious Over Bruce-006, Bruce-251 and SBC Combined

A. Motivation to Combine Bruce-006, Bruce-251, and SBC

POSITAs would have been motivated to use SBC’s SCSI block commands (such as the READ(10) command, highlighted **blue**) for host requests (highlighted **purple**) received by Bruce-251’s flash-memory system, which incorporates Bruce-006’s unified re-map table (highlighted **red**), as shown below. (INTEL-1003, ¶¶245-247.)

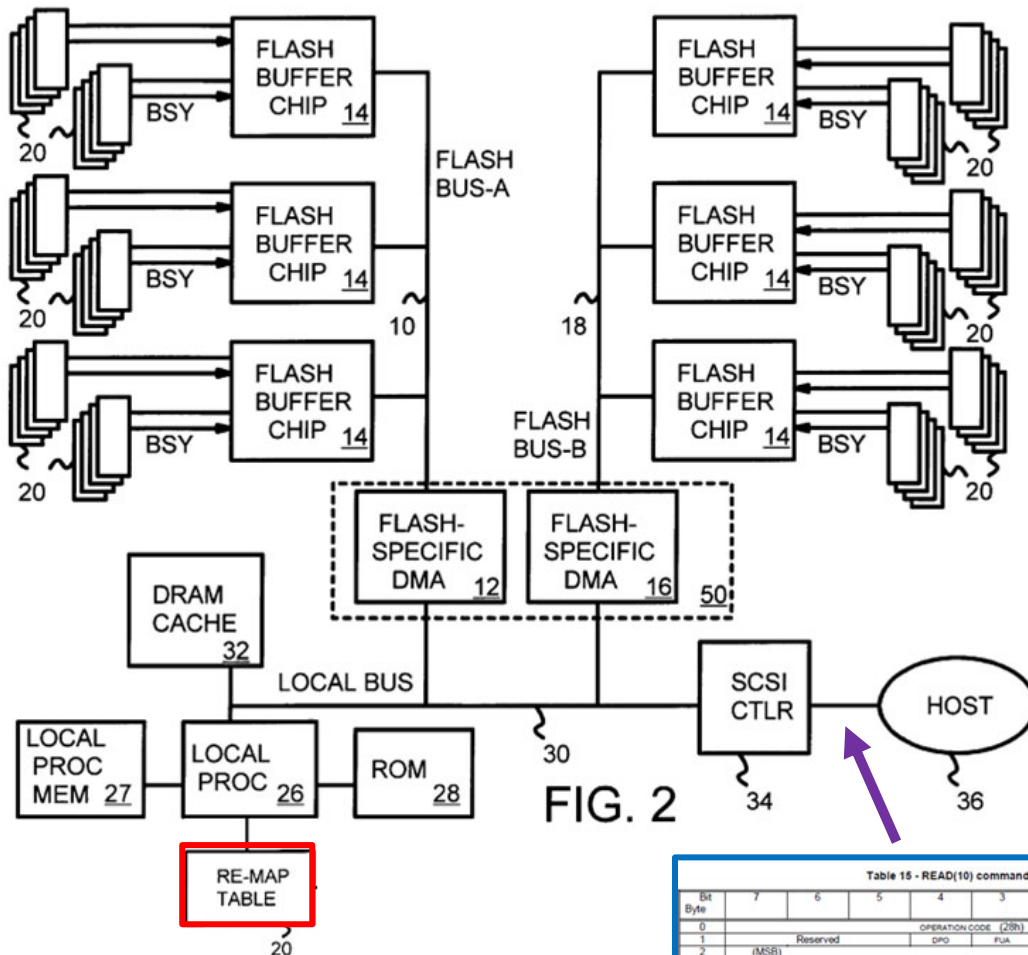


Table 15 - READ(10) command									
Bit	7	6	5	4	3	2	1	0	
Byte									
0	OPERATION CODE (20h)								
1	Reserved			DPO	FUA	Reserved			RELADR
2	(MSB)								
3									
4									
5	LOGICAL BLOCK ADDRESS								(LSB)
6	Reserved								
7	(MSB)								
8	TRANSFER LENGTH								(LSB)
9									

(INTEL-1006, supplemented-Figure 2.)

As explained (p.41-46), POSITAs would have been motivated to combine Bruce-006's re-map table with Bruce-251's flash-memory system. Bruce-251's teachings would have further motivated POSITAs to incorporate SBC's SCSI block commands into the Bruce-006-251 flash-memory system to arrive at the claimed invention. *KSR*, 550 U.S. at 415-421. Bruce-251 teaches a flash-memory system with a "small-computer system interface (SCSI) controller 34." (*See, e.g.,*

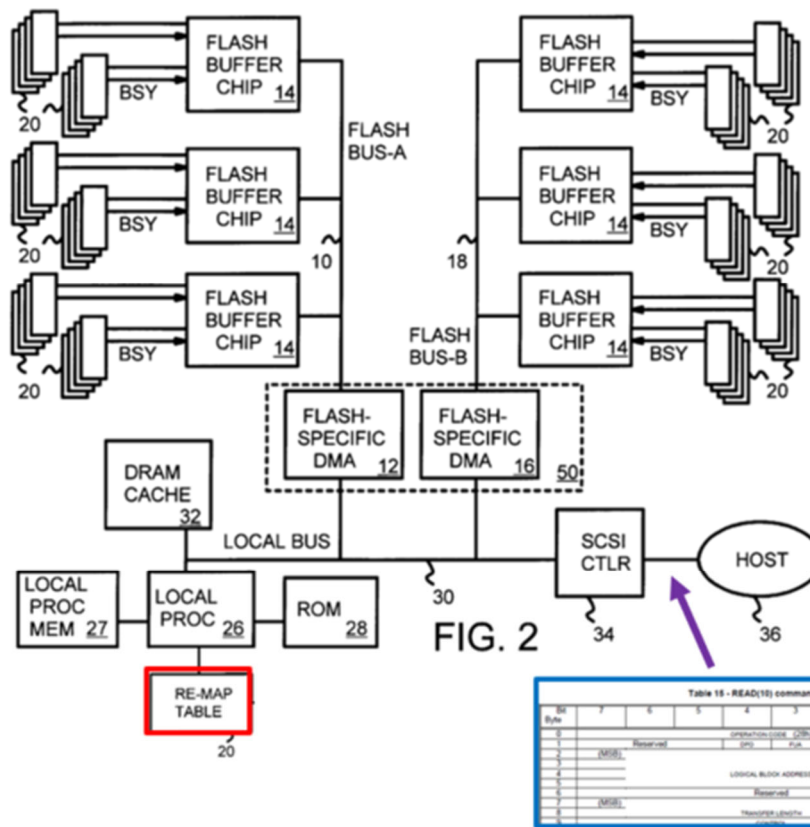
INTEL-1006, 5:51-67 and Fig. 2.) “Requests from host 36 are received by SCSI controller 34.” (INTEL-1006, 6:1-5.) Bruce-251’s express teachings of a SCSI controller for receiving host requests provides an express motivation for POSITAs to implement Bruce-251’s host requests as SCSI block commands disclosed in SBC. (*See* INTEL-1007, 1, 15-16, 18-29.) POSITAs would have looked to references disclosing SCSI commands, such as SBC, in order to carry out Bruce-251’s expandable flash-memory system. (INTEL-1003, ¶246.) POSITAs therefore would have incorporated the SCSI command set extensions disclosed by SBC into Bruce-006-251 flash-memory system. (INTEL-1007, 1, 4, 6-7, 15-16; INTEL-1003, ¶246.)

As discussed (p.45-46), POSITAs would reasonably have expected success incorporating Bruce-006’s unified re-map table into Bruce-251’s flash-memory system. And since Bruce-251 uses a SCSI interface for host requests, POSITAs would also reasonably have expected success combining SBC’s SCSI block commands with Bruce-251. (INTEL-1003, ¶247.)

B. Independent Claim 1

1. Preamble [1P]

If limiting, the preamble is obvious in view of Bruce-251, Bruce-006, and SBC. As explained in detail above, POSITAs would have been motivated to combine them as shown below. (INTEL-1003, ¶¶245-247.)



(INTEL-1006, supplemented-Figure 2.) The entire Bruce-006-251-SBC system corresponds to the claimed “*electronic storage device*” for the same reasons provided in ground 3 (p.47). (INTEL-1003, ¶251.) Bruce-251, Bruce-006, and SBC combined also renders obvious “*an I/O transaction request initiated by a host*” for the same reasons provided in ground 3 (p.47-48). (INTEL-1003, ¶252.)

The Bruce-006-251-SBC system differs from the SBC-Bruce-006 system described in ground 2 in that SBC commands will be received by a SCSI controller in Bruce-251 instead of the re-map table within the flash memory system as shown in Bruce-006. (INTEL-1003, ¶253.) However, this does not change how Bruce-

006's re-map table is used to perform memory operations in response to host-initiated SBC block commands as explained above for ground 2 (p.32-35).

(INTEL-1003, ¶253.) As in ground 2 (p.35-39), when the Bruce-006-251-SBC system receives a READ(10) or WRITE(10) command for reading/writing more than 16 contiguous logical blocks which would necessarily result in the flash-memory system traversing multiple sequential rows of the re-map table taught in Bruce-006 and require multiple read/write operations, use of Bruce-006's re-map table would cause the combined Bruce-006-251-SBC system to perform

“optimiz[ed] memory operations ... in response to receiving an I/O transaction request initiated by a host” for the same reasons explained above for ground 2.

(INTEL-1003, ¶253.) Thus, Bruce-006, Bruce-251, and SBC in combination render the preamble obvious, if it is limiting. (*Id.*)

2. “Set of Logical Fields” [1A] / “Set of PBA Fields” [1B]

As in ground 1, Bruce-006 alone discloses and renders these claim limitations obvious. (INTEL-1003, ¶¶254-55.)

3. “Causes Optimized Memory Operations” [1C]

As explained in [1P] (p.75-76), the use of Bruce-006's re-map table to perform memory operations in response to SBC's block commands initiated by a host in the combined Bruce-006-251-SBC system remains the same as explained above for ground 2 (p.38-39). (INTEL-1003, ¶256.) Thus, Bruce-006, Bruce-251,

SBC combined renders this claim limitation obvious for the same reasons provided above for ground 2 (p.38-39). (INTEL-1003, ¶256.)

C. Dependent Claims 9-15

Claims 9-15 all depend, directly or indirectly, on claim 1. As explained in ground 3 (p.49-63), the additional claim limitations in claims 9-15 are rendered obvious over Bruce-006 combined with Bruce-251. For the same reasons provided in ground 3, these claims are also obvious over Bruce-006, Bruce-251, and SBC combined. (INTEL-1003, ¶257.)

D. Independent Claim 32

1. Preamble [32P]

If the preamble is limiting, as explained (p.63-67), Bruce-006 combined with Bruce-251 renders it obvious. Bruce-006 combined with Bruce-251 and SBC also renders obvious the preamble portion regarding optimally distributing memory load across storage device resources. (INTEL-1003, ¶258.) As explained in [1P] (p.75-76), the use of Bruce-006's re-map table to perform memory operations in response to SBC's block commands initiated by a host in the combined Bruce-006-251-SBC system remains the same as explained above for ground 2 (p.37-38). (INTEL-1003, ¶259.) Therefore, as explained (p.37-38), when Bruce-006's re-map table is used in response to receiving a SBC READ(10) or WRITE(10) command with transfer length greater than 16 pages in the Bruce-006-251-SBC

system, Bruce-006's re-map would require multiple read or write operations directed to memory locations mapped to two or more PBAs in Bruce-006's re-map table to be performed. With Bruce-006's teachings that (i) interleaving improves performance through parallel accesses (detailed at p.13-14); (ii) sequential block accesses improve performance; and (iii) using interleave bits in the PBA to interleave "a sequence of four blocks ... among four flash devices" (INTEL-1005, 11:20-26, 11, 39-40, 15:53-59), POSITAs would understand that interleaving memory operations performed in response to a READ(10) or WRITE(10) command that spans multiple sequential LBAs in Bruce-006 among different flash devices would increase the likelihood of distributing memory operations across different resources. (INTEL-1003, ¶260.) Thus, Bruce-251, Bruce-006, and SBC combined also renders this preamble obvious. (INTEL-1003, ¶261)

2. "First and Second LBA Sets Mapped to First and Second PBAs" [32A] / "First and Second Physical Memory Locations Addressable by First and Second PBAs" [32B] / "Memory System" [32C] / "One Difference Between Said First and Second PBAs" [32D]

As explained (p.14-22 and 68-70), Bruce-006 alone and/or combined with Bruce-251 discloses and renders these claim limitations obvious. (INTEL-1003, ¶262.)

3. “*Optimally Distributed*” [32E]

As explained at p.70-71, Bruce-006 combined with Bruce-251 renders this claim limitation obvious. As further explained at p.77-78, Bruce-006, Bruce-251, SBC combined teaches optimally distributing memory load across storage device resources. Thus, Bruce-006, Bruce-251, SBC combined also renders this claim limitation obvious. (INTEL-1003, ¶263.)

E. Dependent Claim 34

As explained in ground 2 (p.37-38), receiving SBC READ(10)/WRITE(10) command for reading/writing more than 16 contiguous logical blocks would necessarily result in traversing multiple sequential rows of Bruce-006’s re-map, which would result in using multiple PBAs in Bruce-006’s re-map table to perform multiple memory operations. (INTEL-1007, 28-29, 39-41; INTEL-1003, ¶264.) As explained in [1P] (p.75-76), the use of Bruce-006’s re-map table to perform memory operations in response to SBC’s block commands initiated by a host in the combined Bruce-006-251-SBC system remains the same as explained above for ground 2 (p.35). (INTEL-1003, ¶265.) Thus, Bruce-006, Bruce-251, SBC combined renders obvious “*wherein the processing of the I/O transaction request includes a first memory operation performed using said first PBA and a second memory operation performed using said second PBA.*” (INTEL-1003, ¶265.)

And as further explained at p.77-78, POSITAs would understand that interleaving those multiple memory operations among different flash devices would increase the likelihood of distributing memory operations across different resources. (INTEL-1003, ¶¶165-70.) As Bruce-006 teaches using portions of the PBAs for identifying “[d]evice within a bank” and, as explained above (p.21-22), those portions of the PBAs correspond to the claimed “access parameters,” POSITAs would understand the multiple memory operations described above would be performed where “*said first and second PBAs respectively associated with a first set of access parameters and a second set of access parameters, and said set of first and second set of access parameters differing by at least one access parameter.*” (INTEL-1005, 11:30; INTEL-1003, ¶266.)

IX. Discretionary Denial Is Not Appropriate

A. 35 U.S.C. §325(d)

Step one of the §325(d) framework ends the inquiry: The same or substantially the same art and arguments were not presented previously to the Patent Office. *See Advanced Bionics, LLC v. MED-EL Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6, 8 (PTAB Feb. 13, 2020) (precedential); *Caterpillar Inc. v. Wirtgen Am. Inc.*, IPR2022-01278, Paper 12, 43 (PTAB Feb. 7, 2023). Specifically, Bruce-006, Bruce-251, and SBC were not presented to the Office during prosecution of the ’740 patent.

If considered, the framework's second step also indicates that §325(d) does not apply. Although Bruce-251 was cited against the '098 patent (the purported CIP parent of the '740 patent, INTEL-1027), discretionary denial is still strongly disfavored based on factors (c), (e), and (f) from *Becton, Dickinson and Co. v. B. Braun Melsungen AG*, IPR2017-01586, Paper 8 (PTAB Dec. 15, 2017).

Factor (c) considers “the extent to which the asserted art was evaluated during examination, including whether the prior art was the basis for rejection,” and Factor (e) considers “whether Petitioner has pointed out sufficiently how the examiner erred in its evaluation of the asserted prior art.” *Id.*, 17-18. Bruce-251 was never disclosed during '740 prosecution and was never a basis for rejecting any challenged claim. The only '740 rejection was for inadequate written description. (INTEL-1002, 76-78.) Although the '098 Examiner used Bruce-251 to reject the sole independent claim, the applicant merely canceled that claim, and the remaining claim included many limitations that the challenged '740 claims lack. In addition, Bruce-251 was never combined with Bruce-006 or SBC as it is here. Consequently, the PTO never considered claims like the challenged claims or the same or similar arguments regarding their patentability.

Factor (f) considers “the extent to which additional evidence and facts presented in the Petition warrant reconsideration of the prior art or arguments.” *Id.* The Examiner never had the opportunity to consider Bruce-251 combined with

Bruce-006 and/or SBC, which were never submitted during examination of either the '098 or '740 patents. In addition, the applicant failed to disclose Bruce-251 during the '740 examination even though the reference was the sole basis for rejection during the '098 examination and the applicant disclosed another reference—Zilberman—that was made of record but not relied upon during the '098 prosecution. Moreover, the Sechen Declaration and supporting extrinsic evidence were not available to the Examiner.

B. 35 U.S.C. §314(a)

The Board should reach the merits in this proceeding because the Petition presents a compelling unpatentability challenge. (*See* June 21, 2022 Interim Procedure for Discretionary Denials (“Interim Procedure”), 4-5 (“[W]here the PTAB determines that the information presented at the institution stage presents a compelling unpatentability challenge, that determination alone demonstrates that the PTAB should not discretionarily deny institution under *Fintiv*.”).) As set forth herein, the evidence presented in this Petition, “if unrebutted in trial, would plainly lead to a conclusion that one or more claims are unpatentable by a preponderance of the evidence.” (Interim Procedure, 4.)

The challenged claims of the '740 patent are unpatentable over the combination of Bruce-006 and Bruce-251, with or without SBC. Bruce-006 discloses and renders obvious all aspects of the claimed mapping table (e.g.,

logical fields, LBA sets, PBA fields, access parameters, and optimized memory operations) and Bruce-251 discloses an expandable flash-memory system with claimed bus, FDE, and group identifiers. Even if the Board were to find that Bruce-006 does not expressly disclose an I/O transaction mapped to multiple memory operations, that was disclosed by a well-known industry standard—SBC. As further described above (p.30-32, 41-46, 72-74), POSITAs would have been motivated to combine the re-map table taught in Bruce-006 with the expandable flash-memory system taught in Bruce-251, with or without further combining the teaching of the SCSI Block Command taught in SBC, for multiple independent reasons. The case here is strong.

In addition to the Petition’s compelling merits, the Fintiv factors weigh heavily against discretionary denial. The ’740 patent was asserted against Petitioner 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-1008, 66.) Assuming that timing from the original filing date, the NDCA trial date is December 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.

The General Plastic factors also weigh heavily against denial. Regarding Factor 1, although Kioxia recently filed an IPR challenging the '740 patent ("Kioxia IPR"), this is Intel's first challenge to the patent. Further, Kioxia and Intel are accused of infringing based on different products and have no significant business relationship. *Cf. Valve Corp. v. Electronic Scripting Products, Inc.*, IPR2019-00062, Paper 11, 2 (PTAB Apr. 2, 2019) (precedential). And Intel was not involved in preparing or filing the Kioxia IPR. Factors 2, 4, and 5 do not apply because Intel did not file the Kioxia IPR. For Factor 3, road mapping is not a concern here. This IPR relies on different grounds than the Kioxia IPR. Factors 6 and 7 also weigh against denial. *See Western Digital Techs., Inc. v. Viasat, Inc.*, IPR2022-01126, Paper 13, 13-15 (PTAB Nov. 29, 2022).

X. 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 '740 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.*, 5-23-cv-00625 (NDCA)

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 2005-3960, phone number 202-654-6200, as its lead counsel; and **Tyler R. Bowen** (Reg. No. 60,461) at the address of Perkins Coie LLP, 2901 N. Central Avenue, Suite 2000, Phoenix, Arizona 85012, phone number 602-351-8000, as its backup counsel.

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

XI. Conclusion

Inter partes review of the challenged claims is respectfully requested.

Date: April 1, 2023

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

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. 8,010,740, 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 13,978. 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, appendix of exhibits, and any claim listing. This word count was prepared using Microsoft Word.

Date: April 1, 2023

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
Attorney for
Petitioner

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of the Petition for *Inter Partes* Review of U.S. Patent No. 8,010,740 and supporting materials (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.*, 5-23-cv-00625 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

Date: April 1, 2023

Respectfully submitted,

/LORI A. GORDON/

Lori A. Gordon
Attorney for Petitioner