

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SAMSUNG ELECTRONICS CO., LTD. AND SAMSUNG ELECTRONICS
AMERICA, INC.,
Petitioner,

v.

RADIAN MEMORY SYSTEMS LLC,
PO.

U.S. Patent No. 11,681,614

“Storage Device With Subdivisions, Subdivision Query, And Write Operations”

IPR2025-01289

PETITION FOR *INTER PARTES* REVIEW

I.	INTRODUCTION	1
II.	REQUIREMENTS FOR INTER PARTES REVIEW (37 C.F.R. §§42.101 AND 42.104)	1
A.	Grounds for Standing (37 C.F.R. §§42.104(a) and 42.101(a)-(c))	1
B.	Statement and Relief Requested (37 C.F.R. §42.104(b)).....	1
III.	TECHNOLOGY BACKGROUND.....	2
IV.	'614 PATENT.....	2
A.	Priority.....	2
B.	Prosecution History	2
C.	Specification.....	4
V.	LEVEL OF SKILL IN THE ART	4
VI.	CLAIM CONSTRUCTION	5
VII.	GROUND 1: CLAIMS 1-2, 4-8 ARE OBVIOUS IN VIEW OF ELLIS (EX1004) AND SON (EX1006).....	5
A.	Prior Art.....	5
1.	Ellis (EX1004)	5
2.	Son (EX1006)	7
B.	Claim Analysis	8
1.	Claim 1	8
2.	Claim 2	28
3.	Claim 4.....	28
4.	Claim 5	29
5.	Claim 6.....	30

6.	Claim 7	30
7.	Claim 8	30
VIII.	GROUND 2: CLAIMS 3, 10-14, 16-19, 21-29 ARE OBVIOUS IN VIEW OF ELLIS (EX1004), SON (EX1006), AND OLBRICH (EX1010)	32
A.	Prior Art.....	32
1.	Olbrich (EX1010)	32
2.	Claim 3	33
3.	Claim 10	34
4.	Claim 11	36
5.	Claim 12	36
6.	Claim 13	36
7.	Claim 14	37
8.	Claim 16.....	38
9.	Claim 17	38
10.	Claim 18	38
11.	Claim 19	38
12.	Claim 21	39
13.	Claim 22	40
14.	Claim 23	40
15.	Claim 24	40
16.	Claim 25	40
17.	Claim 26	40
18.	Claim 27	41

19.	Claim 28	41
20.	Claim 29	41
IX.	GROUND 3: CLAIMS 9, 15, 20, 30, ARE OBVIOUS IN VIEW OF ELLIS (EX1004), SON (EX1006), OLBRICH (EX1010), AND YAMADA (EX1015)	42
A.	Prior Art.....	42
1.	Yamada (EX1015)	42
B.	Claim Analysis	43
1.	Claim 9	43
2.	Claim 15	48
3.	Claim 20	50
4.	Claim 30	50
X.	GROUND 4: CLAIMS 1-2, 4-8 ARE OBVIOUS IN VIEW OF TAKEO (EX1012), SON (EX1006), AND SINCLAIR (EX1005)	51
A.	Prior Art.....	51
1.	Takeo (EX1012).....	51
2.	Sinclair (EX1005)	52
B.	Claim Analysis	55
1.	Claim 1	55
2.	Claim 2	68
3.	Claim 4	69
4.	Claim 5	69
5.	Claim 6	70
6.	Claim 7	71

7.	Claim 8.....	72
XI.	GROUND 5: CLAIMS 3, 10-13, 16-19, 21-29 ARE OBVIOUS IN VIEW OF TAKEO, SON (EX1006), SINCLAIR (EX1005), AND OLBRICH (EX1010).....	72
A.	Claim Analysis	72
1.	Claim 3.....	72
2.	Claim 10.....	73
3.	Claim 11	75
4.	Claim 12.....	75
5.	Claim 13.....	75
6.	Claim 14.....	76
7.	Claim 16.....	77
8.	Claim 17.....	78
9.	Claim 18.....	78
10.	Claim 19.....	78
11.	Claim 21	78
12.	Claim 22.....	79
13.	Claim 23	79
14.	Claim 24.....	79
15.	Claim 25.....	80
16.	Claim 26.....	80
17.	Claim 27	80
18.	Claim 28.....	80
19.	Claim 29.....	81

XII. GROUND 6: CLAIMS 9, 15, 20, AND 30 ARE OBVIOUS IN VIEW OF TAKEO, SON (EC), SINCLAIR (EX1005), OLBRICH (EX1010), AND YAMADA (EX1015)	82
A. Claim Analysis	82
1. Claim 9	82
2. Claim 15	84
XIII. MANDATORY NOTICES	85
A. Real Party-in-Interest (37 C.F.R. §42.8(b)(1)).....	85
B. Related Matters (37 C.F.R. §42.8(b)(2)).....	85
1. Related District Court Proceedings.....	85
2. Related Patent Office Proceedings.....	86
C. Lead and Back-up Counsel and Service Information	86
D. Payment of Fees (37 C.F.R. §42.15(a))	87
XIV. CONCLUSION.....	87
CERTIFICATE OF WORD COUNT	88
CERTIFICATE OF SERVICE	89

PETITIONER'S EXHIBIT LIST

EX1001	U.S. Patent No. 11,681,614 to Andrey V. Kuzmin et al. (“’614 Patent”)
EX1002	Prosecution history of the ’614 Patent (“’614 Prosecution History”)
EX1003	Expert Declaration of Dr. Heiner Litz.
EX1004	U.S. Patent 9,239,781 (“Ellis”)
EX1005	International Publication No. WO 2008/082996 (“Sinclair”)
EX1006	U.S. Pub. No. 2012/0096217 (“Son”)
EX1007	U.S. Pub. No. 2008/0189485 (“Jung”)
EX1008	Yang Hu et al., Performance Impact and Interplay of SSD Parallelism through Advanced Commands, Allocation Strategy and Data Granularity (2011)
EX1009	U.S. Pub. No. 2012/0303931 (“Asnaashari”)
EX1010	U.S. Patent 8,621,137 (“Olbrich”)
EX1011	Radian ’614 Infringement Contentions
EX1012	Certified Translation of JP2008176606A (“Takeo”)
EX1013	JP2008176606A
EX1014	Hu et al., Exploring and Exploiting Multilevel Parallelism Inside SSDs for Improved Performance and Endurance (2013)
EX1015	U.S. Pat. Pub. No. 2011/0271032A1 (“Yamada”)
EX1016	U.S. Pat. Pub. 2012/0284587 (“Yu”)
EX1017	U.S. Patent No. 9,423,970 (“Darragh”)

I. INTRODUCTION

Samsung Electronics Co., Ltd. and Samsung Electronics America, Inc., (collectively, “Petitioner”) request IPR review of claims 1-30 of USP 11,681,614 (“’614 Patent”). A reasonable likelihood exists that Petitioner will prevail for the reasons discussed below.

II. REQUIREMENTS FOR INTER PARTES REVIEW (37 C.F.R. §§42.101 AND 42.104)

A. Grounds for Standing (37 C.F.R. §§42.104(a) and 42.101(a)-(c))

Petitioner certifies that the ‘614 Patent is available for IPR and that Petitioner is not estopped. Petitioner further certifies that the prohibitions of 35 U.S.C. §315 (a)-(b) are inapplicable.

B. Statement and Relief Requested (37 C.F.R. §42.104(b))

Petitioner requests review under 35 U.S.C. §311 and cancelation of claims 1-30 in view of the following references and grounds:

Ground	Basis	Reference(s)	Claims
1	§103	Ellis, Son	1-2, 4-8
2	§103	Ellis, Son, Olbrich	3, 10-14, 16-19, 21-29
3	§103	Ellis, Son, Olbrich, Yamada	9, 15, 20, 30
4	§103	Takeo, Son, Sinclair	1-2, 4-8
5	§103	Takeo, Son, Sinclair, Olbrich	3, 10-14, 16-19, 21-29
6	§103	Takeo, Son, Sinclair, Olbrich, Yamada	9, 15, 20, 30

III. TECHNOLOGY BACKGROUND

Flash memory is organized into erase units or blocks. EX1003, ¶¶30-31. A block may only be erased a number of times prior to inoperability. EX1003, ¶37.

To increase the efficiency, SSDs may subdivide the flash memory into groupings of blocks, *e.g.*, “superblocks,” EX1004, 6:43-65, “metablocks,” EX1006, 3:43-46, or “zones,” EX1012, ¶¶41-12. Blocks of data in a grouping may be written or erased together to increase parallelism. EX1014, §2.2, Fig. 4.

For longevity, SSDs perform management operations where data may be copied, moved, or erased. EX1003, ¶¶33-37. To identify data for maintenance, metadata is tracked. *Id.*, ¶37.

Memory management functions may be carried out by the controller (internal to the SSD) or via shared responsibility with the host (external to the SSD). EX1003, ¶¶38-40. This may be referred to as “cooperative memory management.” EX1007, ¶¶11-12.

IV. '614 PATENT

A. Priority

The '614 Patent claims the benefit of Provisional Application 61/757,464. The prior art in this Petition predates this application.

B. Prosecution History

The application that issued as the '614 Patent was filed on September 12, 2022. The examiner issued one non-final office action. EX1002, 522. The only

rejection was based on double-patenting. *Id.* PO filed a terminal disclaimer. *Id.*, 365.

In a Notice of Allowance, the Examiner identified Limitations 1[pre]-1[a.ii] as the reason for allowance. EX1002, 196-197. The Examiner stated: “none of the prior art teaches or renders obvious the features as recited in the claims, taken in light of the disclosure, specifically as detailed on Par. 0183-0184.” EX1002, 198.

The paragraphs cited by the Examiner concede that multi-plane writes were known, but claim that the patent allows the host to select physical addresses to provide a “true” multi-plane write:

The ability to choose physical addresses permits the host to effectively direct multi-plane writes and reads.... **Host-selection of physical addressing thus permits the host to direct a “true” multi-plane write or read operation** without being affected by legacy multi-plane device addressing restrictions....

EX1002, 998-999.

The challenged claims, however, do not use the term “host-selection of physical addressing.” Nor has PO interpreted the claims to require host-selection of physical addressing in its infringement contentions. EX1011. PO interprets the phrase “host accessible pages,” which appears in Limitations 1[pre]-1[a.ii], to encompass SSDs that include pages that may store data originating from a host. EX1011, 8-9. As shown in this Petition, the claims are invalid based on PO’s interpretation.

C. Specification

The '614 Patent describes a memory system where responsibility for memory management is shared between a controller in the SSD and a host. EX1001, 4:12-20. The flash translation layer can be “eliminate[d]” or “a conventional translation layer” can be used. *Id.* at 61:52-67, 53:19-40.

The memory is organized by “sub-divisions,” which “is an EU [erase unit] or page of NAND flash memory.” *Id.*, 13:36-49. Information about the subdivisions may be provided by the controller to use “asynchronously,” *e.g.*, upon “reach[ing] a predetermined threshold,” or based on a host-initiated “query.” EX1001, 14:48-67. This information is used for maintenance operations, *e.g.*, “garbage collection,” *id.*, Figs. 11A-11B, or wear leveling, *id.*, Figs. 12A-12B.

The '614 Patent purports to improve upon the prior art by permitting the host to “choose physical addresses,” which the patents states allows the “host to direct a ‘true’ multi-plane write or read operation.” EX1001, 54:7-65, 53:20-40. As discussed above, the Examiner pointed to this aspect of the patent’s disclosures as the reason for allowance. **However**, PO has interpreted the claims to encompass systems that do not include direct physical addressing. *See* Section VI.B.

V. LEVEL OF SKILL IN THE ART

A person of ordinary skill in the art (“POSITA”) at the time of the '614 Patent would have had at least a Bachelor of Science degree in electrical engineering, or an

equivalent field, and at least three years of experience with SSDs or other flash memory devices. EX1003, ¶50. More education can supplement practical experience and vice versa. *Id.*

VI. CLAIM CONSTRUCTION

In the co-pending district court action, the parties have not submitted claim construction proposals. Petitioner does not believe any terms require express construction because the prior art relied upon in this petition reads on each claim term under any construction consistent with *Phillips*, and should thus be given their plain and ordinary meaning.

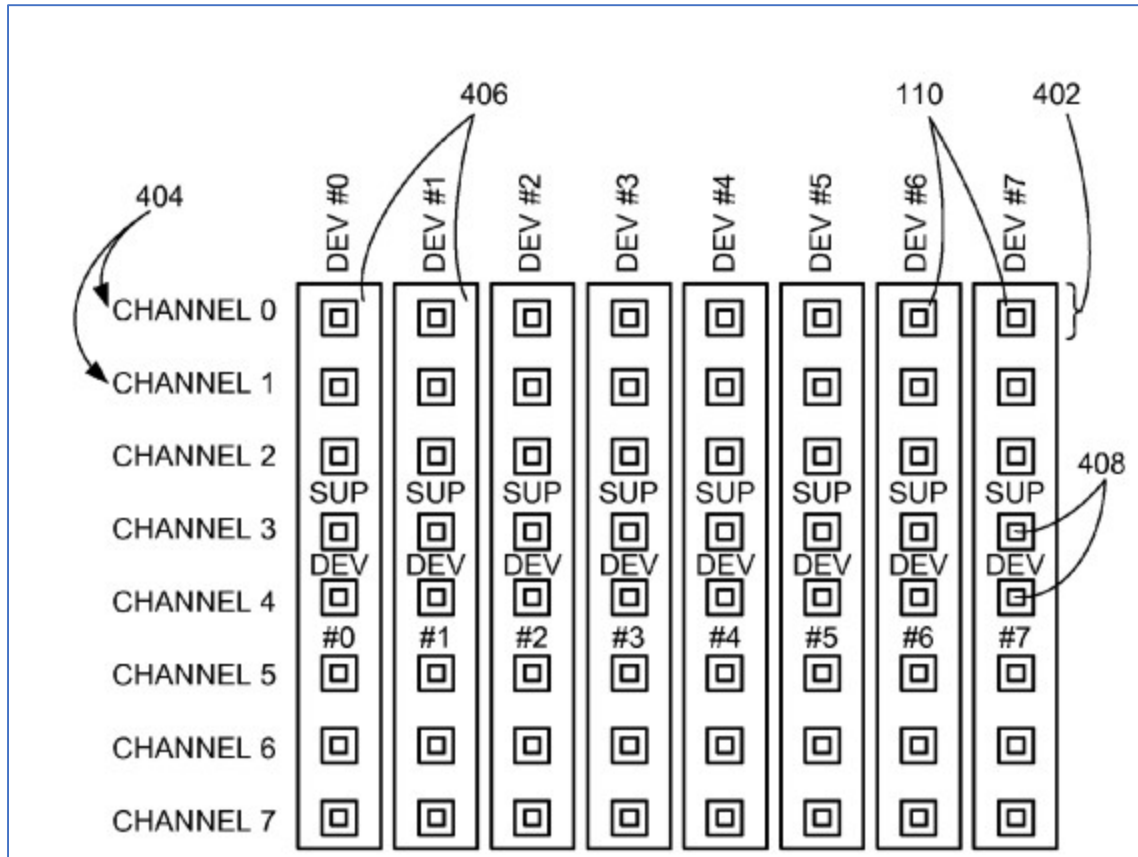
VII. GROUND 1: CLAIMS 1-2, 4-8 ARE OBVIOUS IN VIEW OF ELLIS (EX1004) AND SON (EX1006)

A. Prior Art

1. Ellis (EX1004)

Ellis is prior art under AIA 35 U.S.C. §102(a).

Ellis teaches a storage system with erase blocks 408 spanning memory channels 404:



EX1004, Fig. 4; 6:42-65.

The memory is sub-divided into superblocks, which are a group of erase blocks. EX1004, 6:43-65. Figure 5 illustrates three superblocks:

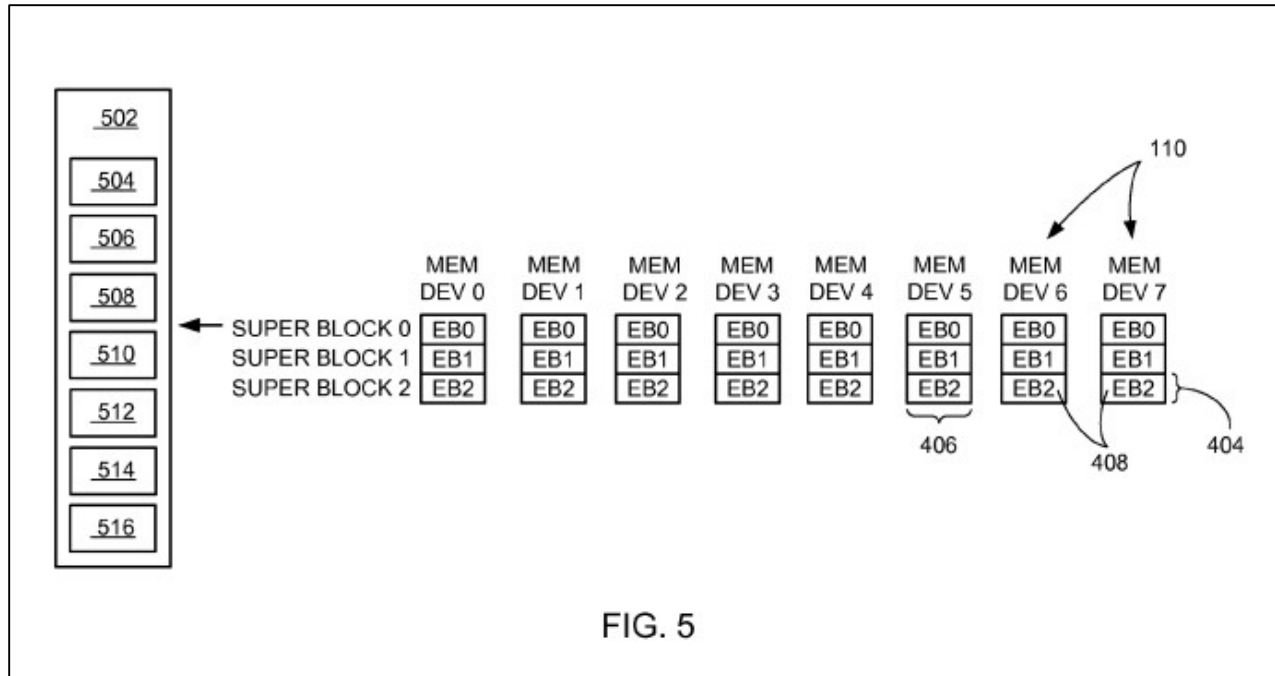


FIG. 5

Id., Fig. 5; 6:43-56.

Superblocks can be operated together “*as a single unit.*” *Id.*, 8:4-7. Ellis stores metadata for each of the superblocks. *Id.*, 11:58-12:2; 12:37-41, Fig. 7.

Memory operations may be shared: either “the memory controller [or] the host system ... can operate any of the modules and functions of the memory controller.” *Id.*, EX1004, 4:58-65.

2. Son (EX1006)

Son is prior art under AIA 35 U.S.C. §102(a).

Son describes the “shortcomings of conventional systems” where management operations are accomplished primarily using a flash translation layer on the controller, and describes the benefits of shared management operations. EX1006, ¶14.

Like Ellis, Son's system is subdivided into superblocks. EX1006, ¶29. Son teaches storing superblock metadata in management tables:

Superblock Table ³⁵⁰										
	335a	335b	335c	335d	335e	335f	335g	335h	335j	335k
	Super Block ID	Wear Count	# Valid Pages	# Invalid Pages	State	Next Superpage	Next Page	Page Detail	Last Mod. Time	Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1006, Fig. 3C, ¶¶80, 29, 38. The metadata can be returned to the host via a host query. EX1006, ¶47.

B. Claim Analysis

1. Claim 1

(a) [1pre]¹

To the extent the preamble is limiting, Ellis discloses a storage device. EX1003, ¶62. Figure 1 shows a host 108 in communication with memory system 102 [storage device]:

¹ The corresponding text for each numbered limitation is identified in the Listing of Claims. See Appendix A.

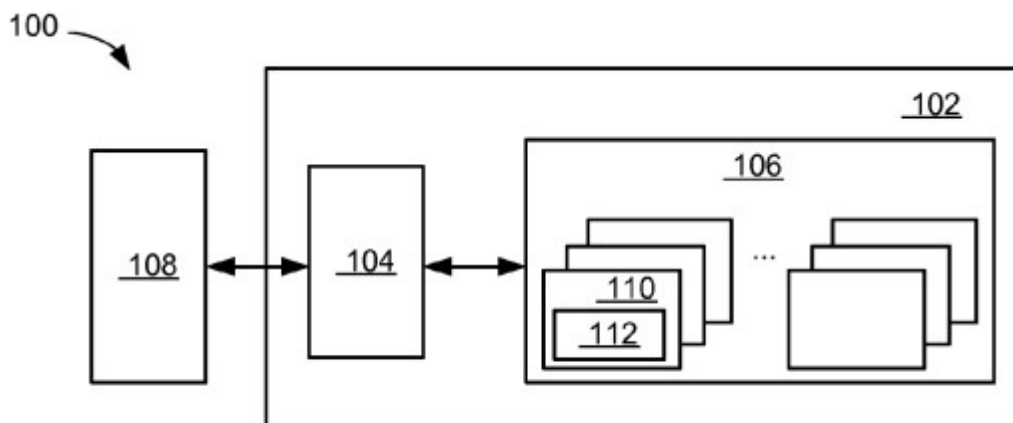


FIG. 1

EX1004, Fig. 1; *id.* 3:10-21.

(b) [1a]

Ellis discloses that the flash memory is divided into superblocks [zones], where each superblock is comprised of erase blocks [each zone of the zones being a subdivision comprising a set of erase units], and each erase block of the set of erase blocks in the superblock comprises “pages” that are accessible by the host [host accessible pages]. EX1003, ¶¶63-66.

Ellis discloses that the flash memory is divided into “superblocks” [zones], that are each “defined as logical groupings of the erase blocks” [each zone of the zones being a subdivision comprising a set of erase units]. EX1004, 7:32-39, 6:43-65, 3:10-45. Figure 5 illustrates this:

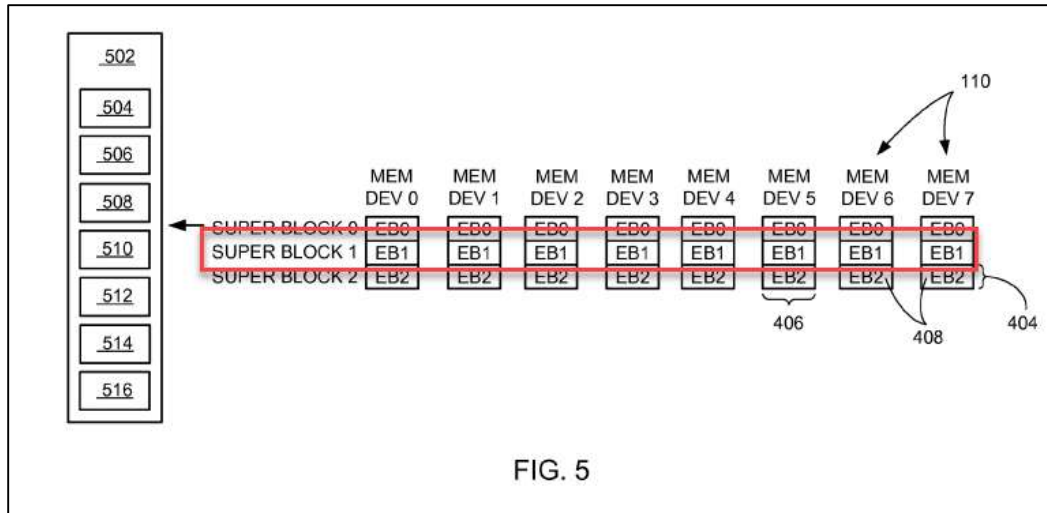


FIG. 5

EX1004, 18:5-10, 7:7-12, Fig. 5. Superblocks are “zones” and “subdivisions” because they are a logical grouping of several physical blocks. EX1003, ¶65. An erase block is an erase unit because it represents a unit of memory that can be erased at one time. EX1003, ¶66. The ’614 patent uses the terms “erase blocks” and “erase units” interchangeably. EX1001, 13:22-30.

Ellis also discloses that “each of the erase blocks” includes “256 pages” [**host accessible pages**]. EX1004, 7:7-12, 18:5-10; EX1003, ¶66.

(c) [1a.i]

Ellis discloses that the superblocks [**zones**] are associated with “logical block address (LBA) information” [**a logical address range**]. EX1003, ¶¶67-68.

Ellis discloses that the host can read and write data “from or to a logical address space.” EX1004, 3:25-28. Ellis also discloses that the memory system stores “logical block address (LBA) information” along with “superblock identification (ID) values.” EX1004, 10:28-32. A POSITA would understand this

disclosure to teach that the blocks and pages of a superblock are associated with a logical address so that the host may read and write data from and to the superblock utilizing the logical addresses. EX1003, ¶68.

(d) [1a.ii]

Ellis discloses a superblocks size [**zone size**] for which the memory controller may issue wear leveling and garbage collection commands [**management command**] that result in the memory system 102 [**storage device**] performing erasure of one or more erase blocks that form the superblock [i.e., **erasure of one or more erase units of the set of erase units**]. EX1003, ¶¶69-71. To the extent Ellis does not expressly disclose that the memory system is **operable to receive a management command from a host**, it was obvious in view of Son. EX1003, ¶¶72-79.

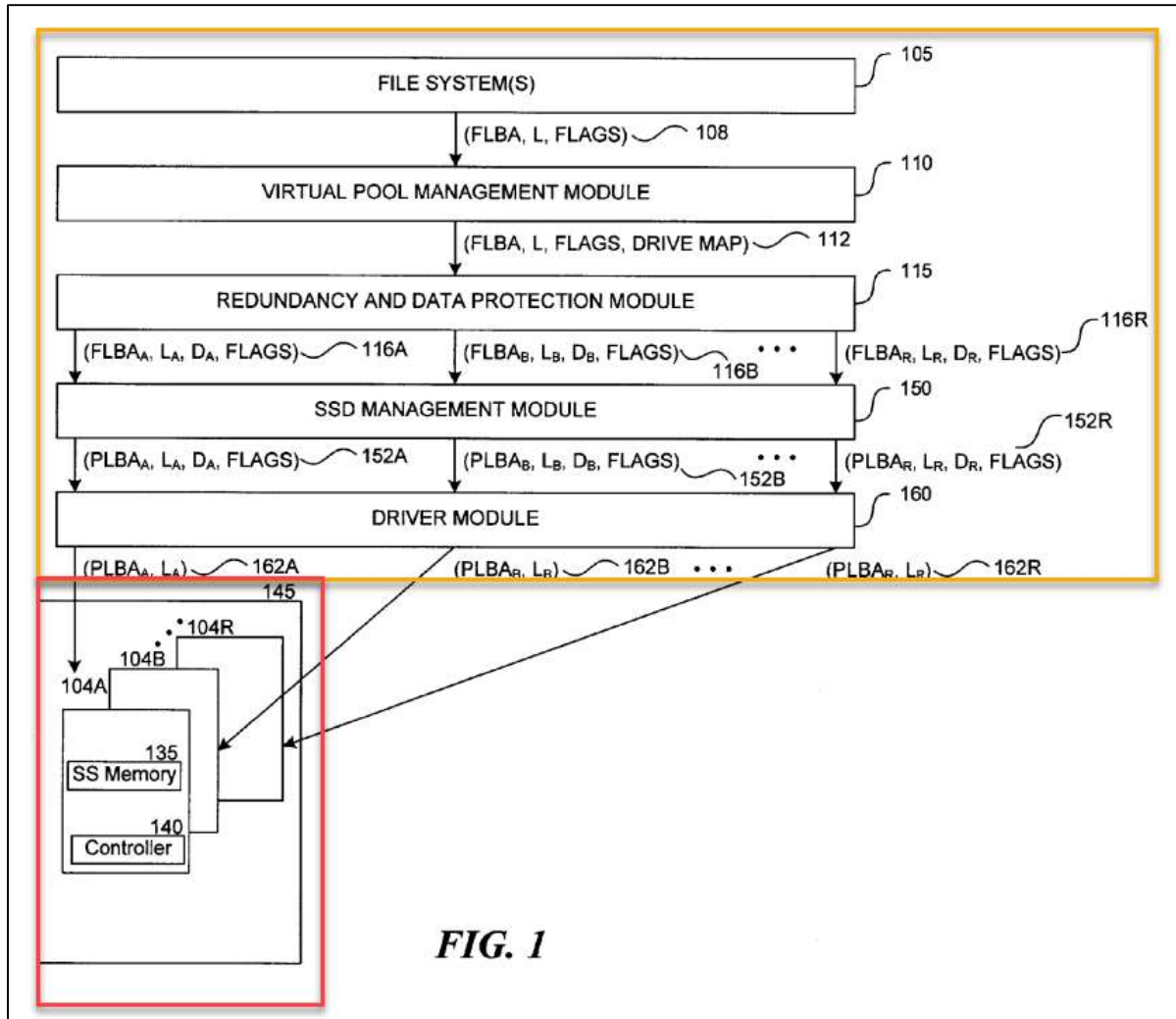
Ellis discloses that the memory controller 104 can include a “flash translation layer (FTL).” EX1004, 4:66-5:3, 16:7-13. The FTL performs maintenance operations on “the superblocks,” such as “wear leveling.” *Id.*, 16:7-13. A command for wear leveling is a management command that results in erasure of the erase blocks that form the superblock. EX1003, ¶70.

Ellis states that either “the memory controller [or] the host system... can operate any of the modules and functions of the memory controller.” *Id.*, EX1004, 4:58-65. This disclosure teaches, or at least suggests, to a POSITA that commands

for wear leveling and garbage collection [**management commands**] may be issued by the host, such that the memory system 102 [**storage device**] is operable to receive a management “**from a host.**” EX1003, ¶71.

To the extent Ellis does not expressly disclose that the memory system is **operable to receive a management from a host**, it would have been obvious to modify Ellis’s storage system in this manner, as taught in Son. EX1003, ¶¶72-79.

Son discloses an SSD Management Module of a host device (orange box) that is in communication with the flash memory (red):



EX1006, Fig. 1; EX1003, ¶73. Like Ellis, Son discloses that the flash memory is subdivided into superblocks. EX1006, ¶¶28-29. The SSD Management Module of the host initiates “maintenance operations including garbage collection and wear leveling” [**management commands**]. EX1006, ¶¶16, 63-64.

Son’s host-issued management commands result in the storage device erasing a superblock. EX1003, ¶74. Son discloses that with garbage collection the “SSD [management] module [within the host] instruct[s] the SSD controller to copy data...

to the selected target superblock” and then “*erase* the source superblock” that is no longer valid. EX1006, ¶¶68-70, Fig. 6. Son discloses that when the host initiates wear leveling it “instructs the controller to... erase[] the worn superblock.” EX1006, ¶¶78-83, Fig. 7.

Motivation to Combine: A POSITA would have been motivated to combine these references because Ellis cites to and references Son. EX1003, ¶¶75-79; EX1004, pg. 3. Both references concern optimization of NAND Flash SSD memory management. EX1004, 7:32-45; EX1006, ¶2, 14, EX1010, 1:56-2:26.

The prior art also includes express motivations to combine the teachings of Ellis and Son such that the storage device is operable to receive a management command from a host. EX1003, ¶¶76-78.

Son teaches that Ellis can be improved by configuring an “SDD module” of a host to perform management operations. EX1006, ¶¶17, 21. Son explains “the inventors have discovered that the failure of conventional SSDs to incorporate file system information” (from the host) “into their operational logic often results in inefficiencies.” EX1006, ¶13. Moving some or all of the management operations to the host helps “relieve the SSD controller and firmware of these responsibilities.” EX1006, ¶63.

Other references also teach that Ellis may be improved by sharing the management functions between the host and controller. EX1007 (Jung), ¶¶3-5, 7,

41; EX1015 (Yamada), ¶¶10, 11, 30. Son, Jung, and Yamada all provide express motivation for modifying Ellis’s device so that one or more management commands for the superblocks are issued by the host.

A POSITA would also have had a reasonable expectation of success in combining Ellis and Son as it represents a straightforward implementation of shared management and metadata management to similar flash devices organized into superblocks. EX1003, ¶79.

(e) [1a.iii]

Ellis discloses that each superblock [**zone**] may be in a closed state [**full state**] when there are no host accessible pages available for writing or an open state [**empty state**] when all host accessible pages are available for writing. EX1003, ¶¶80-84.

Ellis discloses that “when the **state** of the superblocks 502 is set to ‘closed,’ no more writing is allowed to take place.” EX1004, 15:27-38. A superblock that is completely logically written is transitioned to a “closed” state because there are no host accessible pages for writing. *Id.*

Ellis also discloses that a superblock may be in an “open” state when it has “unwritten pages” and that a “valid-page count” defines the “number of pages that are valid” in the open superblocks. EX1004, 15:27-38. An open superblock with a valid-page count of zero is in an empty state. EX1003, ¶82.

To the extent Ellis does not disclose tracking whether superblocks is in a full or empty state, it would have been obvious for Ellis's storage device to do so in view of Son which discloses tracking whether superblocks [**zones**] are in a "CLEAN" [**empty state**] or "SEALED" state [**full state**]. EX1006, ¶40, Fig 3C; *see also* Ground 1, Limitation 1[b.i]; EX1003, ¶¶83-86.

Motivation to Combine: A POSITA would have been motivated to combine Son's teaching to Ellis's storage device, such that the combined system would maintain a "CLEAN" and "SEALED" state for each superblock. EX1003, ¶85.

Both references are directed at tracking superblock level metadata to optimize memory management, writing, and erasing. EX1004, 12:26-29; EX1006, ¶¶78-80; EX1010, 47:62-49:19.

Moreover, Son provides express motivations for maintaining the state of each superblock—*e.g.*, that this information should be evaluated during management operations when selecting a superblock to write to. EX1006, ¶47. A POSITA would have had a reasonable expectation of success in combining these references because maintaining a binary state (full/empty) would have been a straightforward modification that would improve a storage controller by simplifying decision-making logic. EX1003, ¶86.

(f) [1a.iv]

Ellis discloses, or at least suggests, an empty state. *See* Limitation 1[a.iii]. This limitation is also obvious in view of Son which discloses a “CLEAN” state [empty state]. *Id.*; EX1003, ¶¶87-88.

(g) [1b]

Ellis discloses that the memory system 102 includes a memory controller with hardware and software [logic]. EX1004, 3:38-45, 3:1-5; EX1003, ¶89.

(h) [1b.i]

Ellis teaches that control and status information about the superblocks [information about the zones] is stored as “metadata.” EX1004, 12:35-41, 10:27-32; 12:26-29, 7:45-52; EX1003, ¶90. To the extent Ellis does not expressly disclose returning the information recited in this limitation in response to a query from the host, this limitation is at least obvious in view of Son, which identifies additional superblock metadata that should be returned to the host in response to a query so that it may be evaluated when issuing maintenance commands. EX1003, ¶¶91-105.

return information about the zones in response to a query command from the host: Son discloses that management tables containing information about the superblocks [information about the zones] may be stored in “persistent storage” of the “SSD.” EX1006, ¶¶34, 65, 80, Fig. 3A-3D. An example “superblock table” is reproduced below:

Superblock Table ³⁵⁰

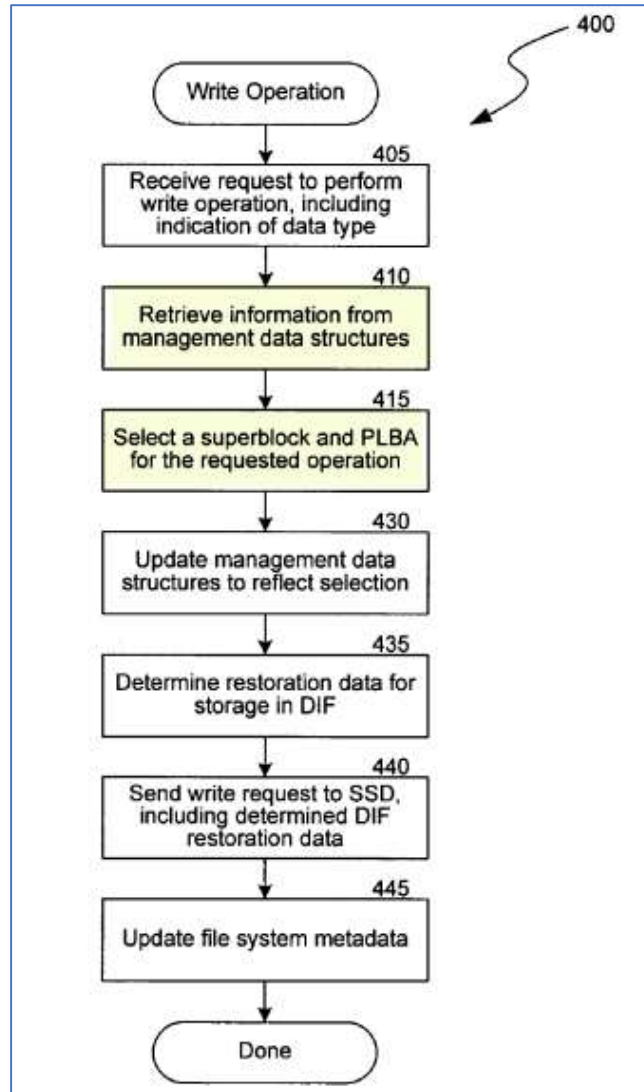
	335a	335b	335c	335d	335e	335f	335g	335h	335j	335k
	Super Block ID	Wear Count	# Valid Pages	# Invalid Pages	State	Next Superpage	Next Page	Page Detail	Last Mod. Time	Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1006, Fig. 3C; *see also id.*, Figs. 3A-3B, 3D.

Son also discloses that the SSD module of the host may retrieve the information about the superblocks [**information about the zones**] by “querying the SSD controller” [*e.g.* **query command from the host**]. EX1006, ¶47.

Figure 4 illustrates a write operation. EX1006, Fig. 4. At step 410, the SSD module “retrieve[s] information from the management data structures” (EX1006, ¶46) by “querying the controller” (EX1006, ¶47) and then uses that information to “select a superblock” at step 415:



EX1006, Fig. 4.

Similarly, Figure 6 shows a garbage collection process that involves “identifying source superblocks and target superblocks” at step 603. EX1006, Fig. 6. Son discloses that in the identify step the SSD module evaluates “information obtained from the controller,” such as “information shown in the superblock table, other management data structures, or information obtained from the controller.”

EX1006, ¶65. A POSITA would understand that information may be obtained using a host query. EX1006, ¶47; EX1003, ¶94.

As another example, Son discloses a wear leveling process in Figure 7 that selects superblocks at step 710. EX1006, Fig. 7. Son confirms that the SSD module may evaluate “any” of the superblock information in the management tables and that the superblock information is obtained from the controller. EX1006, ¶80.

Additionally, Son discloses that “the SSD module may manipulate the values in a data structure by reading a portion of the data structure into system memory.” EX1006, ¶34. A POSITA would understand that the information may be read using a host query. EX1003, ¶96. A POSITA would have also found it obvious to retrieve the entirety of the management tables from persistent memory because storing them in system memory permits faster retrieval and lower latency than having to retrieve them from the flash memory each time. *Id.*

the information about the zones comprising a list of the zones accessible by the host: Son discloses that the information about the zone may include a list of the superblocks [**list of the zones**] accessible by the host.

For example, Figure 3C discloses an example management table that includes a list of superblocks each of which is identified by a superblock ID:

Superblock Table ³⁵⁰										
	335a	335b	335c	335d	335e	335f	335g	335h	335j	335k
	Super Block ID	Wear Count	# Valid Pages	# Invalid Pages	State	Next Superpage	Next Page	Page Detail	Last Mod. Time	Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1003, ¶97 (EX1006, Fig. 3C).

Additionally, Son teaches that as part of the write operation shown in Figure 4, the host will “access” the “superblock table” at step 410 to “retrieve” information about the list of superblocks and their state, so that it can “identify which superblocks in the SSD are... available as candidate superblocks to store the source data.” EX1006, ¶¶46-47.

Son further teaches that the SSD module may obtain a list of superblocks “from the controller” for garbage collection and wear leveling. EX1006, ¶¶65 (SSD module may obtain the “management data structures”), 80 (same).

a zone state: Son teaches that the superblock table includes three states: “CLEAN,” “IN_USE,” or “SEALED.” EX1006, ¶40, Fig 3C. The “state” column is shown below:

Superblock Table ³⁵⁰

	335a	335b	335c	335d	335e	335f	335g	335h	335j	335k
	Super Block ID	Wear Count	# Valid Pages	# Invalid Pages	State	Next Superpage	Next Page	Page Detail	Last Mod. Time	Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1006, Fig. 3C.

As part of the write operation the SSD module of the host identifies a list of “candidate superblocks” by, among other things, “querying” the “state of each candidate superblock” and determining if it is in an “in use or clean” state. EX1006, ¶¶46, 47 (“When selecting the superblock, the SSD module may evaluate any combination of several factors including... **the state of each candidate superblock**... The SSD module may determine... these various factors... **by querying the... controller**.”). Similarly, the garbage collection and wear leveling operations also retrieve and evaluate the superblock state. EX1006, ¶¶65, 80.

the zone size: Son discloses that in response to a query from the SSD module in the host, the memory controller may return the “number of free, valid and/or invalid pages in each candidate superblock” [zone size]. EX1006, ¶¶47 (“When selecting the superblock, the SSD module may evaluate... the number of free, valid, and/or invalid pages in each candidate superblock.”), 65, 80. A POSITA would

understand that because Son returns the number of pages in the superblock, it discloses returning the size of the superblock. EX1003, ¶99; EX1006, ¶28 (disclosing pages are “8KB of data” in size).

logical addresses associated with respective zones: Son discloses that the management information may include a “mapping table” (*e.g.*, as shown in Figure 3B) that associates each FLBA, file system logical block address (column 320*a*), with a data type (column 320*b*), superblock address (column 320*c*), a superpage address (column 320*d*), a page address (column 320*e*), and a sector address (column 320*f*).”). EX1006, ¶38, Fig. 3B. A POSITA would understand that a FLBA is a **logical block address** and associating superblocks with logical block addresses discloses logical addresses associated with respective zones. EX1003, ¶100.

Son also discloses a “page detail array table 375” (Figure 3D) that may include a row for “or all pages within a superblock.” EX1006., ¶43. Son “associates each page with an FLBA.” *Id.* Since the superblocks are comprised of pages, Son’s disclosure of logical addresses associated with each page in a superblock also discloses logical address for the respective zones [**FLBAs associated with respective superblocks**]. EX1003, ¶101.

Further, the SSD module may retrieve the mapping table and page detail array table in connection with step 410 of the write operation shown in Figure 4. EX1006, ¶¶46 (“At block 410, the SSD module retrieves information from management data

structures, such as those illustrated by Figs. 3A-3D.”), 65 (same), 80 (same). Son teaches that the SSD module may retrieve the mapping table to identify whether an FLBA that the host seeks to write data to “is already associated with a previously-stored PLBA.” EX1006, ¶45.

information representing an extent to which the respective zones include host accessible pages available for writing: As discussed above, Son discloses that in response to a query from the SSD module in the host, the memory controller may return the “number of free, valid and/or invalid pages in each candidate superblock” (EX1006, ¶¶47, 65, 80) and may consider whether a superblock has “an insufficient number of pages available” (*id.*, ¶48). At least the free page count is information representing the extent to which the specified zone includes host accessible pages available for writing. EX1003, ¶103.

Motivation to Combine: A POSITA would be motivated to combine Son’s teachings of a host query for returning information about superblocks [**zones**] with Ellis’s storage device. EX1003, ¶¶104-105.

Ellis already teaches that “metadata can include... [any] NAND statistics that [are] collected as a consequence of normal operatives,” and “any additional information... that can aid in retrieval, organization, health of, and current condition or state of the memory devices.” EX1004, 10:21-32, 12:26-29. And, as discussed above (*see* Limitation [1.a.ii]), Son discloses the benefits of returning the

information about zones to the host so that the host may utilize the information to issue management commands. EX1006, ¶¶13-14.

A POSITA would have had a reasonable expectation of success in combining Son's teachings and Ellis's storage device as it merely represents a known technique of returning metadata to the host via a host query. EX1003, ¶105.

(i) [1b.ii]

Ellis discloses writing data to the set of erase blocks [**sets of erase units**] of a specified superblock [**specified zone**] using a multi-plane write command such that the data is written concurrently to two or more erase blocks of the set of erase blocks. EX1003, ¶¶106-109.

Ellis teaches that a superblock is comprised of a set of erase blocks that are operated together as a "single unit" (EX1004, 8:4-8) and that a superblock can be formed from erase blocks on different dies. EX1004, 7:25-31, 6:66-7:5. A POSITA would understand that erase blocks that are on different dies are also on different planes. EX1003, ¶107. Thus, Ellis discloses that a superblock may comprise erase blocks from multiple planes.

Ellis also teaches that superblocks can provide "improved performance" by allowing "multiple pages within each of the superblocks 502" to be "written **concurrently**," wherein the pages are "not just within one of the erase blocks." EX1004, 10:49-55, 13:57-61, 14:39-44 ("virtual erase block [superblock] can be

ordered such that multiple of the data pages can be written at the same time”). A POSITA would thus understand Ellis to disclose a multi-plane write command that concurrently writes to erase blocks on different planes. EX1003, ¶108.

To the extent Ellis does not explicitly disclose the use of a multi-plane command, it was at least obvious in view of the general knowledge of a POSITA. EX1003, ¶109. Multi-plane commands were well-known in the art. EX1014, §1.3 (“SSD manufacturers have provided advanced commands, such as... multi-plane...”), §2.2 (“Multi-plane command activates multiple read, program or erase operations in all planes of the same die”), Fig. 4. The prior art includes express teachings, suggestions and motivations to utilize a multi-plane command with Ellis’s storage device by disclosing that the commands permit “parallel” operations and that “parallelism has been regarded as a key to achieving the peak performance.” EX1004, 5:32-52; EX1010, 2:41-43; EX1014, §1.4.

(j) [1b.iii]

Ellis discloses that after data is written to the flash memory the storage device updates the information reflecting the extent to which a superblock [*e.g.*, **zone**] includes unwritten pages in any of the erase blocks [**host accessible pages available for writing**]. EX1003, ¶¶110-112. Ellis teaches that the controller may perform “a write operation... for host write data,” EX1004, 14:25-34, where the “host interface” is the “communication mechanism” for the “host system,” *id.*, 4:46-50. Following

the write execution, Ellis teaches tracking and “updating metadata” such as “valid-page count in... the superblocks.” *Id.*, 20:11-20; Claim 1 (“...performing an operation on the superblock; and updating metadata corresponding to the superblock...”). Ellis also discloses that the controller tracks the number of “stale or invalid pages” and that the controller tracks “unwritten pages in any of the erase blocks” of the superblock, to determine whether it is in an “open” (empty) stage or a “closed” state (full). *Id.* at 19:11-15, 15:27-38.

To the extent Ellis does not expressly teach this limitation, it was obvious in view of Son. EX1003, ¶111. Son discloses additional metadata that should be tracked and evaluated in connection with management operations, such as the “free page count,” “valid page count,” and “invalid page count” of a superblock. EX1006, ¶¶65, 80; *see also id.*, ¶47. The free page count is **information representing the extent to which the specified zone includes host accessible pages available for writing**. EX1003, ¶111. Son further teaches that following the write execution, the superblock “valid page count, invalid page count, [and] next page,” are “update[d].” EX1006, ¶¶50-52. A POSITA would understand Son to disclose that the storage device is operable to update the information about the superblocks that is stored in the SSD module. EX1003, ¶111.

A POSITA would have also been motivated to apply Son’s teaching to Ellis’s storage device to update the number of free pages in a super block after a write

operation. EX1003, ¶112. As discussed above, Son discloses that it is useful to evaluate this information in connection with write and management operations. *See* Ground 1, Limitations [1.a.ii]. A POSITA would have recognized that the information must be updated so that it remains accurate. EX1003, ¶112.

(k) [1c]

Ellis discloses a memory controller that comprises software instructions that control operation of hardware [**logic**]. *See* Ground 1, Limitation 1[b]; EX1003, ¶113.

2. Claim 2

Ellis discloses that the storage device is operable to update a state of a superblock [**zone state**] to the closed state [**full state**] when the specified zone has no host accessible pages available for writing. *See* Ground 1, Limitation 1[a.iii]. Ellis also discloses that the superblock is transitioned to a full state in response to a “multi-plane” write command, as discussed above. *See* Limitation 1[b.ii]; EX1003, ¶¶114-115.

3. Claim 4

As discussed above, in view of Son it would be obvious to modify Ellis’s storage device so that the host issues wear leveling and garbage collection commands [**management commands**] to the storage device. *See* Ground 1, Limitation 1[a.ii].

These management commands are associated with the memory size of at least a superblock. Ex. 1003, ¶¶116-117. Figure 7 discloses management commands for wear leveling that selects, a worn superblock and target superblock (step 710), and copies the contents of the worn superblock to a target superblock and erases the worn superblock (step 715). EX1006, Fig. 7, ¶80-83. Thus, the management command specifies at least a superblock and is associated with the memory size of that superblock [**a memory size of a zone**]. EX1003, ¶117.

4. Claim 5

As discussed above, the prior art includes express teachings, suggestions, and motivations to modify Ellis's memory system so that the memory controller is operable to receive a management command from the host, as taught in Son. *See* Ground 1, Limitation 1[a.ii].

Son also discloses that the management commands may be commands for wear leveling and garbage collection that result in the memory controller [**logic**] issuing erase commands to the set of erase blocks [**erase units**] of the specified superblock [**specified zone**]. *See* Ground 1, Limitation 1[a.ii].

Further, the combined Ellis-Son system would continue to organize superblocks as a set of erase blocks that are on different planes, as taught in Ellis. *See* Ground 1, Limitation 1[b.ii]. *See also* EX1003, ¶¶118-120.

5. Claim 6

Son discloses that the information about the zones is stored in “persistent storage” [**non-volatile memory**] of the “SSD.” *See* Ground 1, Limitation 1[b.i]; *see also* EX1006, ¶34; EX1003, ¶121.

6. Claim 7

Figure 2 of Ellis illustrates memory controller 104 with a storage unit 204 for the “control firmware” [**the logic**]. EX1004, 4:4-12, Fig. 2. A POSITA would understand the memory controller and its storage unit is located on a die. EX1003, ¶¶122-123.

Ellis also discloses that the memory controller 104 includes a host interface that “allows the host system 108 of FIG. 1 to interface” and exchange information “with the memory controller 104.” *Id.* 4:46-50, Figs. 1-2. As discussed above, in the combined Ellis-Son system, the information transmitted to the host would include information about the superblocks [**information about the zones**]. Ground 1, Limitation 1[b.i]; *see also* EX1006, ¶34. A POSITA would understand that the transmission would involve the memory controller retrieving the information from persistent storage and buffering it before transmission. EX1003, ¶123; EX1012, ¶32, Fig. 1.

7. Claim 8

Ellis discloses that the memory controller tracks the program erase-count [**number of erase operations**] performed on a particular superblock [*e.g.*, a

particular zone]. EX1004, 12:60-12 (“metadata pages need to be managed by the block control module” and “can include a program-erase count”), 13:15-24, 14:15-20; EX1003, ¶124.

Son similarly discloses that the memory controller tracks the wear count [number of erase operations]. EX1006, Fig. ¶39, Fig. 3C. The wear count column is illustrated in the superblock table of Figure 3C, and “indicates how many time a particular superblock has been erased”:

Superblock Table ³⁵⁰

	^{335a} Super Block ID	^{335b} Wear Count	^{335c} # Valid Pages	^{335d} # Invalid Pages	^{335e} State	^{335f} Next Superpage	^{335g} Next Page	^{335h} Page Detail	^{335j} Last Mod. Time	^{335k} Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1006, Fig. 3C. Son teaches that the superblock table (including the wear count) may be returned in response to the host query. *See* Limitation 1[b.i]; *see also* EX1006, ¶¶47, 65, 80.

As discussed above, in the combined Ellis-Son system, metadata the superblock table is returned to the host in response to a host-query so that its information may be evaluated by the host in connection with maintenance commands. *See* Ground 1, Limitation 1[a.ii]; EX1003, ¶¶125-126.

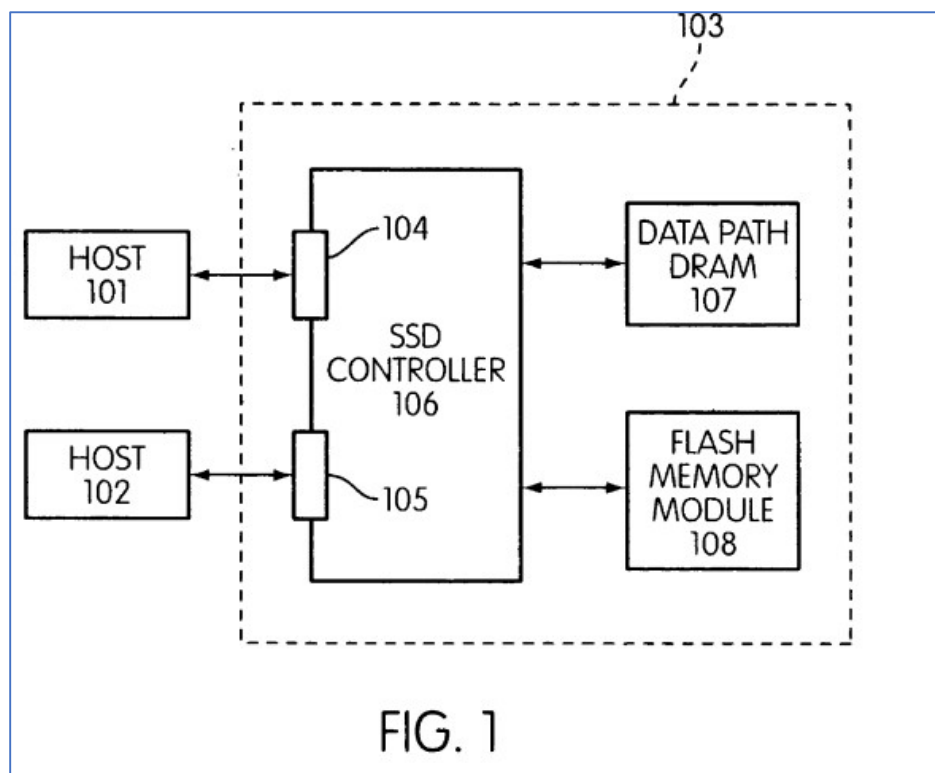
VIII. GROUND 2: CLAIMS 3, 10-14, 16-19, 21-29 ARE OBVIOUS IN VIEW OF ELLIS (EX1004), SON (EX1006), AND OLBRICH (EX1010)

A. Prior Art

1. Olbrich (EX1010)

U.S. Patent 8,621,137 (“Olbrich”) was filed on April 8, 2008 and was published on July 2, 2009. It is prior art under at least AIA 35 U.S.C. §102(a).

Olbrich discloses a storage system comprising an SSD Controller in communication with one or more hosts and a flash memory module:



EX1010, Fig. 1. The storage device is subdivided into superblocks that are comprised of superpages, and the system maintains information about the superblocks in a “superblock metadata table.” EX1010, 47:55-62, Fig. 35. Olbrich

also discloses that superpages of a superblock consist of pages spread across multiple dies (or planes) and that the host can issue a multi-plane write to these pages using the “same address location.” EX1010, 15:58-67.

2. Claim 3

Ellis discloses, or at least renders obvious, a multi-plane write command that is operable to write the data to the two or more erase units of the set of erase units of the specified zone. *See* Limitation 1[b.ii].

A POSITA would have understood that multi-plane commands utilize a common page address. EX1003, ¶132. As Hu explains, SSDs at the time had “multi-plane commands” and these commands use the “same... page addresses” across planes [**common page address**]. EX1014, §2.2.

To the extent Ellis alone does not expressly disclose that the multi-plane command uses a common page address, it was at least obvious that the multi-plane command would use a common page address in view of Olbrich. EX1003, ¶¶133-136. Olbrich discloses a multi-plane command that uses the “same address location” [**common page address**] for writing to pages that are on different dies. EX1010, 15:60-67, Fig. 12; EX1003, ¶133. A POSITA would understand that pages on different dies are on multiple planes, such that the combined system would write concurrently to pages within erase blocks on different planes. EX1003, ¶133.

A POSITA would have been motivated to combine Olbrich’s teachings such that Ellis’s storage device performs a multi-plane write command using the “same address location” across planes. EX1003, ¶134. As explained in the prior art, multi-plane commands in which the “blocks executing a multi-plane” command have the “same... page addresses” (EX1014, §§1.3, 2.2) “improve the performance of SSDs” (EX1014, §1.3).

Further, the combination reflects the application of a known technique (multi-plane commands which use a common page address) to a known device (Ellis-Son’s flash memory system) ready for improvement (improving parallelism of Ellis-Son’s base device) to yield predictable results (a device that supports multi-plane commands). EX1003, ¶135.

A POSITA would also have had a reasonable expectation of success in combining the Ellis-Son storage device with Olbrich’s teachings because it represents a straightforward implementation of multi-plane write commands to similar flash devices. EX1003, ¶136.

3. Claim 10

(a) [10pre]

See Ground 1, Limitation 1[pre].

(b) [10a]

See Ground 1, Limitation 1[a].

(c) [10a.i]

See Ground 1, Limitation 1[a.i].

(d) [10a.ii]

See Ground 1, Limitation 1[a.ii].

(e) [10a.iii]

See Ground 1, Limitation 1[a.iii].

(f) [10a.iv]

Ellis alone, or in combination with Son, discloses a superblock [zone] may be in an empty state. *See* Ground 1, Limitation 1[a.iv].

Further, in the combined Ellis-Son system the storage device would receive a management command that specifies a superblock for erasure from the host. *See* Ground 1, Limitation 1[a.ii]. A POSITA would understand that the combined Ellis-Son system would transition the superblock to the empty state after the erase operation so that the state of the superblocks remains accurate. EX1003, ¶143.

(g) [10b]

See Ground 1, Limitation 1[b].

(h) [10b.i]

See Ground 1, Limitation 1[b.i].

(i) [10b.ii]

See Ground 1, Limitation 1[b.i].

(j) [10b.iii]

See Ground 1, Limitation 1[b.i].

(k) [10b.iv]

See Ground 1, Limitation 1[b.i].

(l) [10b.v]

See Ground 1, Limitation 1[b.i].

(m) [10b.vi]

See Ground 1, Limitation 1[b.i].

(n) [10b.vii]

See Ground 1, Claim 3.

(o) [10b.viii]

See Ground 1, Limitation 1[b.iii].

(p) [10b.ix]

See Ground 1, Claim 2.

(q) [10c]

See Ground 1, Limitation 1[c].

4. Claim 11

See Ground 1, Limitation 1[b.ii].

5. Claim 12

See Ground 1, Limitation 1[a.ii], 10[a.iv].

6. Claim 13

In the Ellis-Son combination the host instructs the memory controller [**logic**] to issue an erase command associated with erase blocks [**erase units**] of the specified superblock [*e.g.*, **zone**]. See Ground 1, Limitation 1[a.ii]. Ellis also discloses that the erase blocks of the specified superblock are on two or more different planes. See Ground 1, Limitation 1[b.ii]; EX1003, ¶¶157-158.

7. Claim 14

Ellis discloses that bad blocks are detected “to be bad during normal operation of the storage device.” EX1004, 11:41-45. A POSITA would understand this disclosure to teach or suggest that a block that is determined to be bad is marked as such so that it is not used for storage of new write data. EX1003, ¶159. At minimum, it was obvious to mark bad blocks in this manner because the memory system must know not to write data to blocks that cannot reliably store data. *Id.*

A POSITA would also understand Ellis’s disclosure of detecting bad blocks “during normal operation” to teach that the bad blocks may be determined as a result of the host issuing the erasure command. EX1003, ¶160. Erasure is part of the normal operation of the device and it was known to detect bad blocks during the erasure process. EX1016, ¶¶241-243 (disclosing “a process for detection and handling of bad pages and bad blocks” where “[a]fter the block is erased, the status of each page in that block is checked” to determine if it is bad.). In fact, a POSITA would have been motivated to detect bad blocks during erasure because erasure

performance can help identify bad blocks. *See, e.g.*, EX1017, 6:21-7:7. If a “block needs more than predetermined number of erase cycles applied to fully erase the block,” the block can be marked as bad. *Id.*; EX1003, ¶160.

8. Claim 16

The Ellis-Son system includes a host interface that transmits information about the zones to the host in response to a query from the host. *See* Ground 1, Limitation 1[b.i]. A POSITA would have understood that the information about the zones may need to be buffered and queued at the host interface prior to transmission. EX1003, ¶161.

To the extent the Ellis-Son system does not render this limitation obvious, it was obvious in view of Olbrich which discloses that the “host interface” comprises a “transmit stage buffer” that is used to queue outgoing data for transmission. EX1010, 20:31-54, Fig. 17. A POSITA would have been motivated to combine Olbrich’s teaching with the Ellis-Son storage device because using a buffer to queue information for transmission from flash memory to a host is a common design that optimizes performance and error handling. EX1003, ¶¶162-164.

9. Claim 17

See Ground 1, Claim 6.

10. Claim 18

See Ground 1, Claim 7.

11. Claim 19

See Ground 1, Claim 8.

12. Claim 21

(a) [21pre]

See Ground 1, Limitation 1[pre].

(b) [21a]

See Ground 1, Limitation 1[a.i].

(c) [21b]

Ellis discloses a **memory controller 104** in communication with the **flash memory 106**. See Ground 1, Limitation 1[pre].

(d) [21b.i]

See Ground 1, Limitation 1[b.i].

(e) [21b.ii]

In the Ellis-Son combination the memory controller logic is operable to cause the storage device to output the information about the superblocks [**zones**] in response to a query command from a host. See Ground 1, Limitation 1[b.i].

Olbrich further renders obvious writing to the set of erase blocks [**erase units**] of a superblock using a common page address. See Ground 2, Claim 3.

The Ellis-Son-Olbrich combination further discloses updating the information representing the extent to which the zone comprises host accessible pages available for writing in response to writing to the set of erase units of the zone. See Ground 1, Limitation 1[b.iii]; EX1003, ¶174.

13. Claim 22

See Ground 1, Limitation 1[b.i].

14. Claim 23

See Ground 1, Limitation 1[a.iii].

15. Claim 24

The Ellis-Son combination discloses tracking whether a superblock is in a full state. *See* Ground 1, Limitation 1[a.iii]. In view of Olbrich, it was obvious to write to the set of erase block [**erase units**] of the superblock [**zone**] using a common page address. *See* Ground 1, Claim 3. A POSITA would have understood that when a superblock has no host accessible pages available for writing the state of the superblock should be updated so that it remains accurate. EX1003, ¶179.

16. Claim 25

See Ground 1, Limitation 1[a.iv].

17. Claim 26

The combined Ellis-Son system returns information about the superblocks [**zones**] in response to a host query. *See* Limitation 1[b.i]; *see also* EX1006, ¶¶47, 65, 80. The information about the superblocks may include a last modification time [**information indicative of an amount of time since data was written**] for the

superblock, as disclosed in Son:

Superblock Table ³⁵⁰										
	335a	335b	335c	335d	335e	335f	335g	335h	335j	335k
	Super Block ID	Wear Count	# Valid Pages	# Invalid Pages	State	Next Superpage	Next Page	Page Detail	Last Mod. Time	Data Type
340a	0	115	0	0	CLEAN	0	0	Pagelist ₁ []	5/5/11 09:11:01	Metadata
340b	1	462	856	741	IN_USE	68	5	Pagelist ₂ []	1/1/09 22:08:11	User Data
	...	...	...	...	...	...	...	...	...	...

FIG. 3C

EX1006, ¶39, Fig. 3C; EX1003, ¶181.

18. Claim 27

See Ground 1, Limitation 1[b.i].

19. Claim 28

See Ground 1, Claim 4.

20. Claim 29

Ellis discloses that the memory controller is operable to cause the memory system to concurrently erase the superblock. EX1004, 8:4-8, 14:35-36, 17:31-36 (“all the erase blocks 408 in each of the superblocks 502 can be erased at the same time”); EX1003, ¶184.

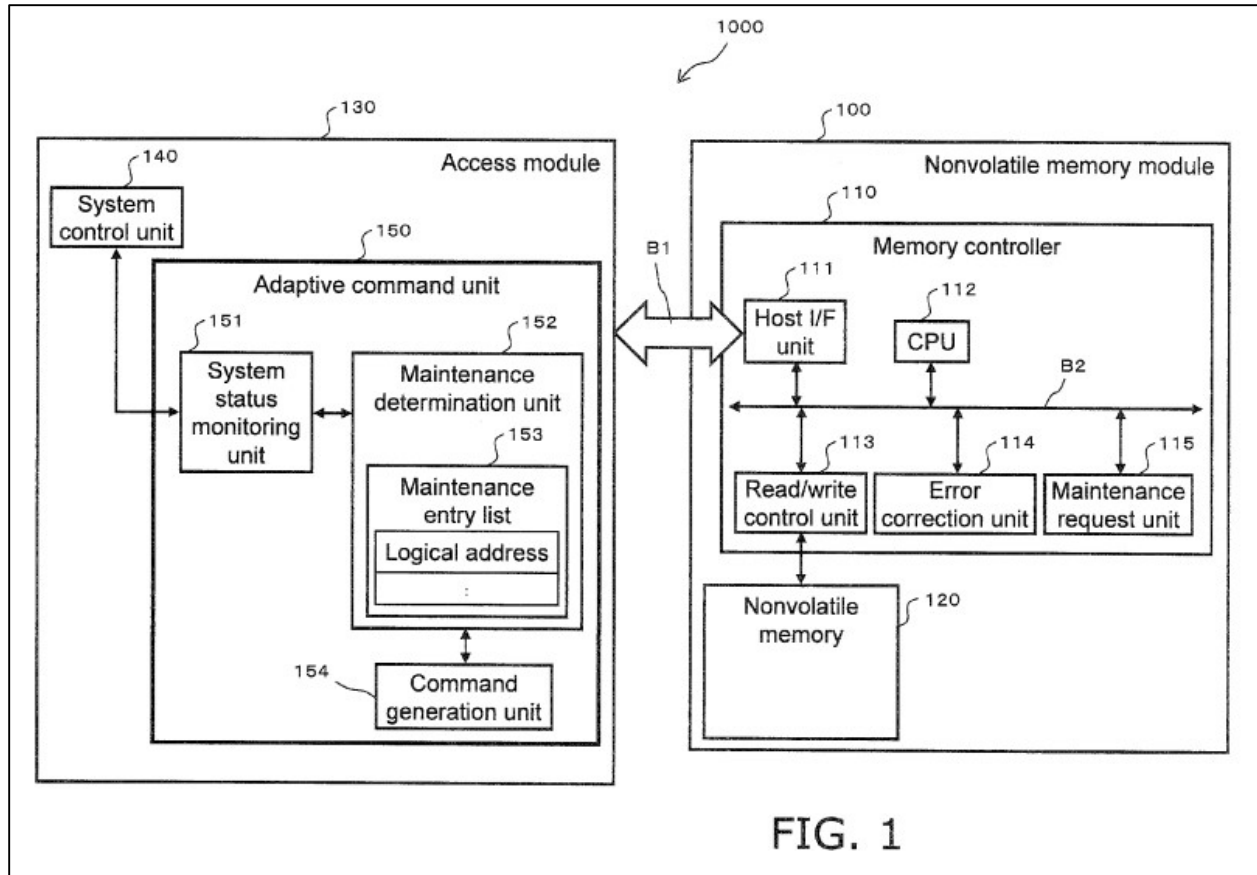
IX. GROUND 3: CLAIMS 9, 15, 20, 30, ARE OBVIOUS IN VIEW OF ELLIS (EX1004), SON (EX1006), OLBRICH (EX1010), AND YAMADA (EX1015)

A. Prior Art

1. Yamada (EX1015)

U.S. Pat. Pub. No. 2011/0271032A1 (“Yamada”) was filed on July 27, 2010 and was published November 3, 2011. It is prior art under at least AIA 35 U.S.C. §102(a).

Yamada teaches an SSD system where memory management is shared between the host and controller. Figure 1 illustrates a memory module that includes a “memory controller” comprising a “maintenance request unit,” and an access module (host) that includes a “maintenance command unit.” EX1015, ¶12.



The controller identifies a LBA (or group of LBAs) that need maintenance based on comparison of metadata against thresholds. *Id.*, ¶¶13, 32, 80-81. The controller then transmits the LBA (or groups of LBAs) to the host via a maintenance request, and the host may then issue a maintenance command. *Id.*, ¶14, 109-111.

B. Claim Analysis

1. Claim 9

(a) 9[a]

Ellis teaches a “recycle operation.” EX1003, ¶188. To identify a superblock to copy, Ellis teaches that the controller tracks time since data was programmed and identifies “superblocks 502 that are older or written a longer time in the past”

[**metadata indicating an amount of time since the zone was written to**]. EX1004, 19:4-15.

To the extent Ellis does not expressly disclose this limitation, a POSITA would have found it obvious to modify Ellis’s storage device to generate metadata that indicates the “last modification time” for the superblock in view of Son and/or Yamada. EX1003, ¶189; EX1006, ¶39, Fig. 3C; EX1015, ¶164. A POSITA would understand that charge in the memory cells will leak over time such that data may become corrupted if it is not written for a long time. EX1003, ¶189; EX1015, ¶9. Thus, a POSITA would have been motivated to track metadata regarding last write times and issue a maintenance command to “refresh” (rewrite) data if the last write time exceeds a threshold amount of time to avoid this type of data loss. *Id.*

(b) 9[b]

As explained above, a POSITA would have found it obvious to modify Ellis’s storage device so that metadata regarding last write times is maintained and a maintenance command is issued to “refresh” (rewrite) data if the last write time exceeds a threshold amount of time to avoid data loss. *See* Ground 3, Limitation 9[a]; EX1003, ¶190. And as explained above, in the combined Ellis-Son system the management commands would be issued by the host. *See* Ground 1, Limitation 1[a.ii].

To the extent the combined Ellis-Son system does not render this limitation obvious, the limitation is obvious in further view of Yamada. EX1003, ¶191. Yamada discloses a memory module [storage device] that includes a “maintenance request unit 115” and a access module [host] that includes a “maintenance determination unit 152”:

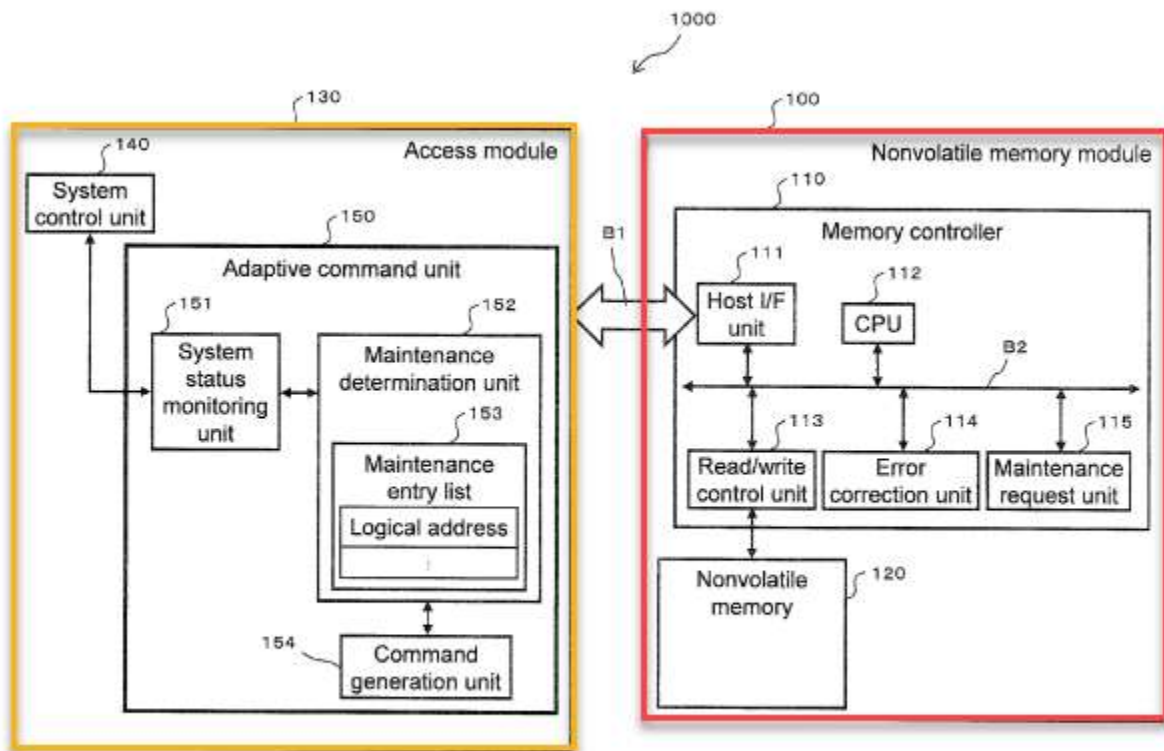


FIG. 1

EX1015, Fig. 1. The maintenance request compares metadata to a “criterion value” [threshold] and sends “a maintenance request indicating that the nonvolatile memory needs maintenance to the access device” when the criterion value is exceeded [recommends that the host send a second management command].

EX1015, ¶¶32-33, 81-82, Figs. 1-2. The maintenance determination unit of the access device then schedules a maintenance request that “includes refresh and/or wear leveling of the nonvolatile memory.” EX1015, ¶30. Yamada discloses that the maintenance request “may include a plurality of logical addresses of a plurality of blocks,” such that the “area that needs maintenance consists of... a plurality of blocks.” EX1015, ¶¶116-118. Thus, in view of Yamada, it was obvious for the host to send a second management command directed at Ellis’s superblocks [**directed to the specified zone**]. EX1003, ¶191.

Although Yamada discloses that the maintenance request is generated when the number of errors occurring in the data exceeds a threshold, a POSITA would understand that the criterion used to determine maintenance may be any metadata that indicates a need for maintenance, such as the last modification time of a superblock exceeding a threshold value (thereby indicating the need for the data to be refreshed). EX1003, ¶192.

Motivation to Combine. A POSITA would have been motivated to incorporate Yamada’s teaching regarding notifying a host to schedule maintenance commands when a metadata threshold is exceeded. EX1003, ¶193.

First, Ellis, Son, Olbrich and Yamada involve similar techniques to solve the same problem, *i.e.*, collecting, storing and using metadata for maintenance activities

to improve the efficiency, performance and “integrity of data” in NAND flash memory systems. EX1015, ¶¶5-7, 11; EX1003, ¶194.

Second, the prior art includes express teachings, suggestions and motivations for combining the references in this manner. EX1003, ¶195. Ellis, Son, and Yamada each disclose the need to maintain metadata that indicates the amount of time since a superblock was last written and the references also teach the need to perform maintenance when the last written time exceeds a threshold. EX1004, 19:4-15; EX1006, ¶81; EX1015, ¶164; EX1003, ¶195. Yamada further teaches that the memory controller can be responsible for comparing metadata to a threshold and then notifying the host that maintenance is needed. EX1015, ¶¶32-33, 81-82. Yamada explains that this type of shared management in which the memory controller performs the comparison and the host sends the management command is beneficial for a number of reasons, including that it avoid overloading the memory controller, improves data retention policies, and enables the system to evaluate devices conditions when scheduling the maintenance. EX1003, ¶196. In light of the teaching in Ellis that memory management responsibility may be shared between the controller and the host, EX1004, 4:58-65; 20:40-44, Son’s teaching of the benefits of sharing responsibility between the host and controller, *see, e.g.*, EX1006, ¶¶13-14, and the benefits described in Yamada, a POSITA would be motivated to

incorporate Yamada’s teaching regarding host-initiated maintenance commands which are based on metadata generated by the controller. EX1015, ¶148.

Third, a POSITA would have been motivated to implement this type of shared memory management because it avoid the storage device having to expose its performance characteristics to the host. EX1003, ¶197. The optimal threshold (*e.g.*, the last modified time or erase count) for triggering a maintenance command may differ across storage devices. EX1003, ¶197. By having the memory controller determine whether a threshold has been exceeded internally and then simply notify the host of the need for a management operation, the vendor of the storage device can avoid exposing this information to the host. EX1003, ¶197.

A POSITA would have also had a reasonable expectation of success moving the management command responsibility from the controller firmware to the host-side software. EX1003, ¶198.

2. Claim 15

(a) [15a]

Ellis teaches that the controller is configured to store a “program-erase count” for each superblock, which is “defined as a numerical value indicating a number of times or cycles the memory devices 110 associated with the corresponding instances of the superblocks 502 are programmed or erased.” EX1004, 11:63-67. *See also id.*, 13:14-24, claim 1. Thus, Ellis teaches a controller with logic to generate “program-

erase count,” *i.e.*, “**metadata indicating a count of erasures of the erase units.**”

EX1003, ¶199.

Son also teaches the maintenance of “wear count” for each superblock. EX1006, ¶80. The superblock table includes a “*wear count column indicat[ing] how many times a particular superblock has been erased.*” EX1006, ¶39. Given that Ellis teaches that its memory controller includes a block control module 306 that can perform wear leveling (EX1004, 4:66-5:15, 16:7-13, Fig. 3) a POSITA would have been motivated to combined Son’s teaching of generating wear count information with Ellis’s storage device. EX1003, ¶200.

- (b) [15b] in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.

As mentioned above, Ellis teaches that its memory controller includes a block control module 306 that can perform wear leveling. EX1004, 4:66-5:15, 16:7-13, Fig. 3. The block control module can use the “program-erase count” [**the metadata**] to “determine if wear begins to deteriorate the integrity of the memory devices.” EX1004, 13:15-25. A POSITA would understand this disclosure to teach that the block control module determines deterioration by comparing whether the program-erase count has exceeded a threshold. EX1003, ¶201. At minimum, it would have been obvious for Ellis’s storage device to perform this threshold comparison in view of Son, which discloses that wear leveling involves comparing a wear count to a

threshold value. EX1006, Fig. 7, ¶¶64, 79; EX1003, ¶201. And, as discussed above, in the combined Ellis-Son system the host would be responsible for sending the wear leveling command when the threshold is exceeded. *See* Ground 1, Limitation 1[a.ii].

To the extent Ellis-Son do not render obvious this limitation, it is obvious in view of Yamada. EX1003, ¶202. Claim 15 is similar to Claim 9. The difference between these claims is that in Claim 15 the metadata at issue is “metadata indicating a count of erasures of the erase units,” while in Claim 9 the metadata is “metadata indicating an amount of time since the specified zone was written to.” As discussed in connection with Claim 9, Yamada discloses a maintenance request unit within a memory controller that sends a “maintenance request” to the host when a criterion value is exceeded. EX1015, ¶32; *see also id.* ¶33. In the preferred embodiment of Yamada the criterion value used is the number of errors, but POSITA would understand that the criterion value could include other information, such as wear count, that indicates the need for maintenance. EX1003, ¶202.

3. Claim 20

(a) [20a]

See Ground 3, Claim 9[a].

(b) [20b]

See Ground 3, Claim 9[b].

4. Claim 30

(a) [30a]

See Ground 3, Claim 9[a].

(b) [30b]

See Ground 3, Claim 9[b].

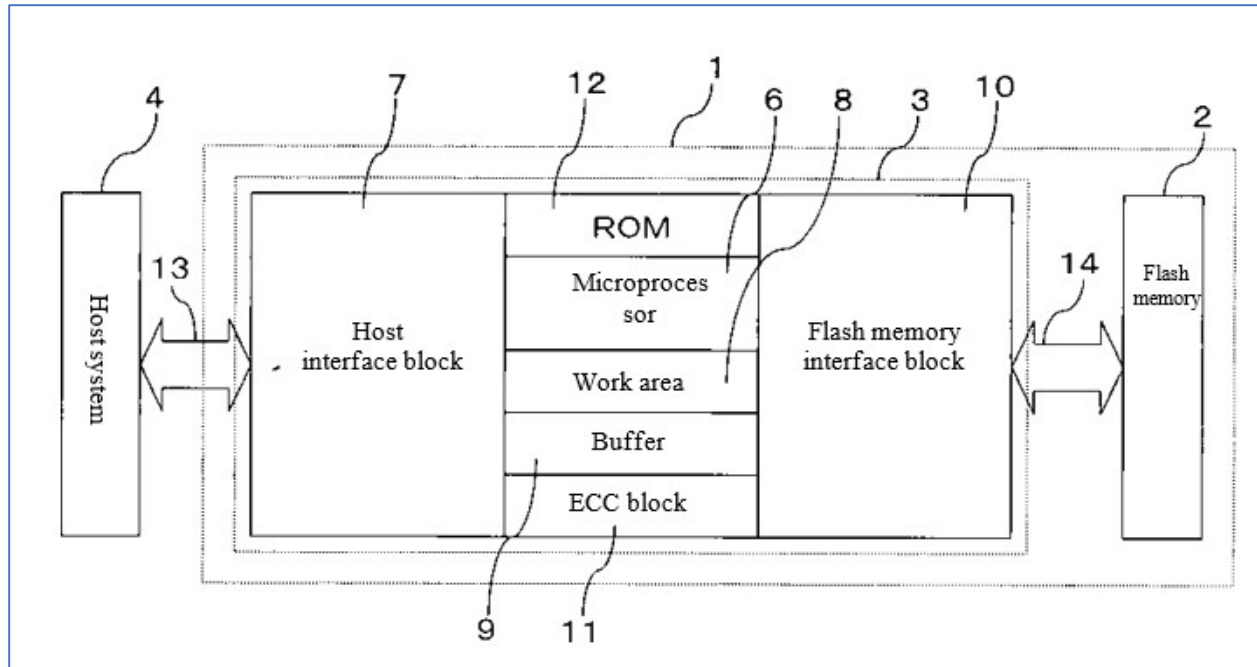
**X. GROUND 4: CLAIMS 1-2, 4-8 ARE OBVIOUS IN VIEW OF
TAKEO (EX1012), SON (EX1006), AND SINCLAIR (EX1005)**

A. Prior Art

1. Takeo (EX1012)

Japanese Patent Publication No. JP2008-176606 (“Takeo”) was filed on January 19, 2007 and was published on July 31, 2008. It is prior art under at least AIA 35 U.S.C. §102(a).

Takeo discloses a storage system designed to “efficiently transfer internal information within a flash memory system... to the host system side in response to a request from the host system side.” EX1012, Abstract.



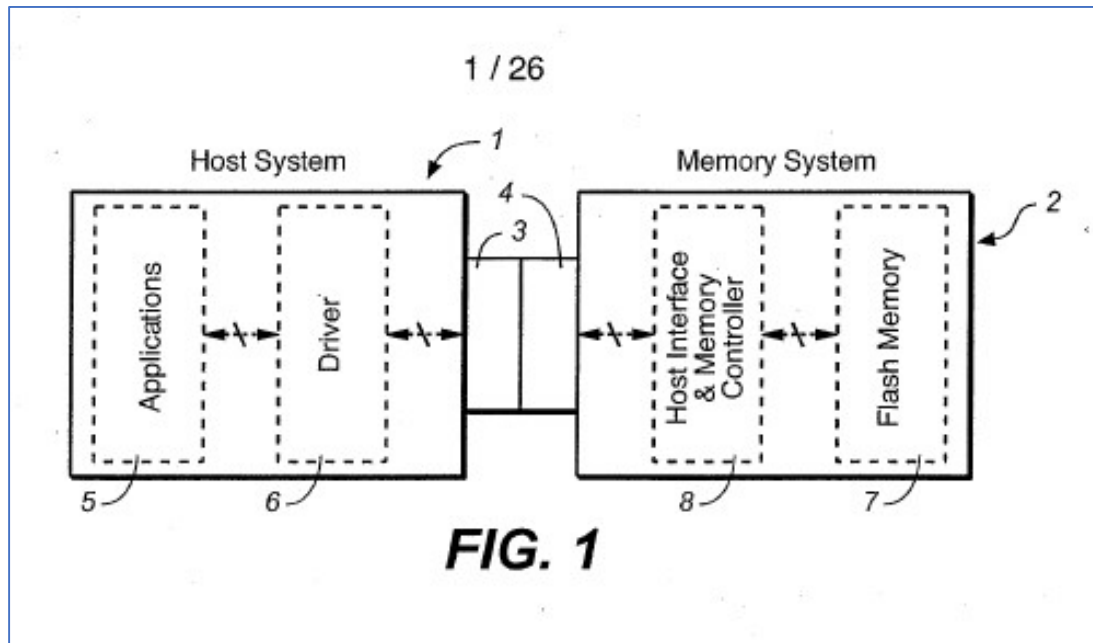
EX1012, Fig. 1, ¶¶41-43. The memory is subdivided into “zones,” which Takeo explains are comprised of blocks that are in turn comprised of pages. *Id.*

In response to receiving a host query, referred to as “a vendor command,” the memory controller will write internal information regarding the flash memory to buffer 9 and then return that information to the host system 4 via the host interface 7. EX1012, Abstract; *see also id.*, ¶¶44-50.

2. Sinclair (EX1005)

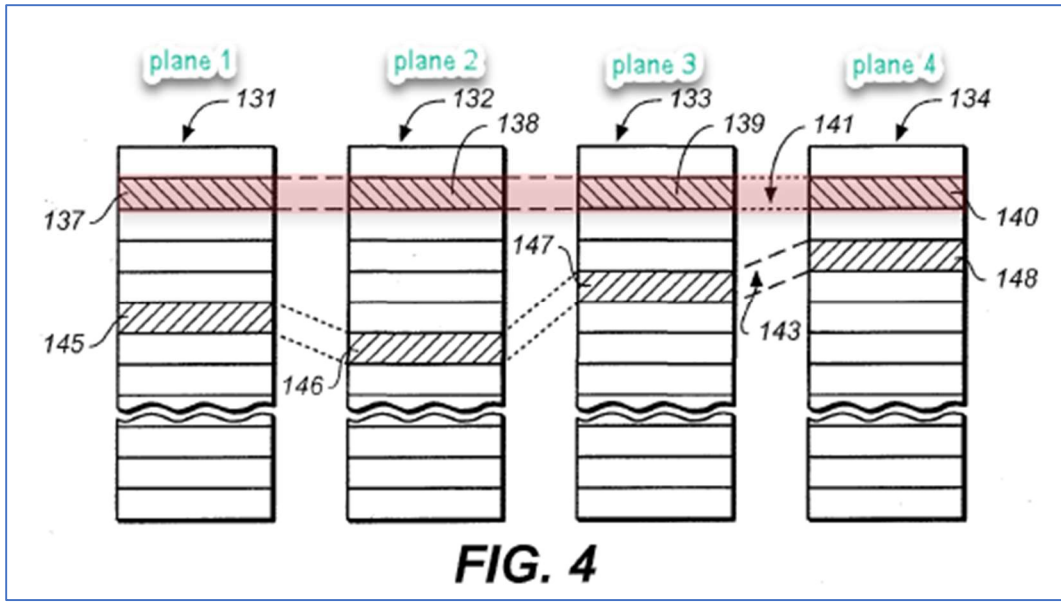
International Publication No. WO 2008/082996 (“Sinclair”) was filed on December 19, 2007 and was published on July 2008. It is prior art under at least AIA 35 U.S.C. §102(a).

Sinclair discloses a host system that communicates with a memory system:



EX1005, Fig. 1.

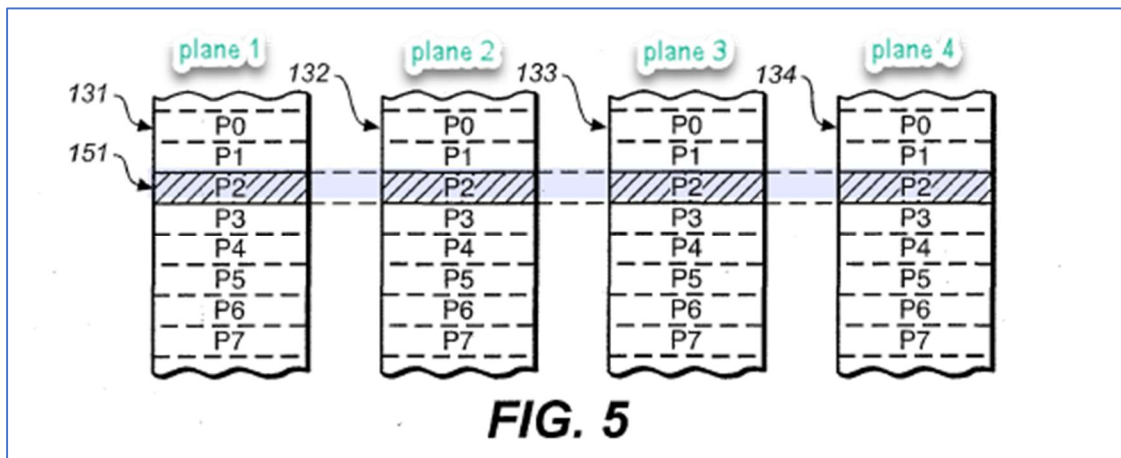
The memory system includes a flash memory that may be organized into “metablocks,” which comprise a group of blocks across multiple planes that may be operated together (*e.g.*, written and erased together). EX1005, Fig. 4, ¶62. Figure 4 illustrates a metablock comprising four individual blocks, 137-140:



EX1005, Fig. 4, ¶62.

The individual blocks are in turn divided into pages of memory cells.

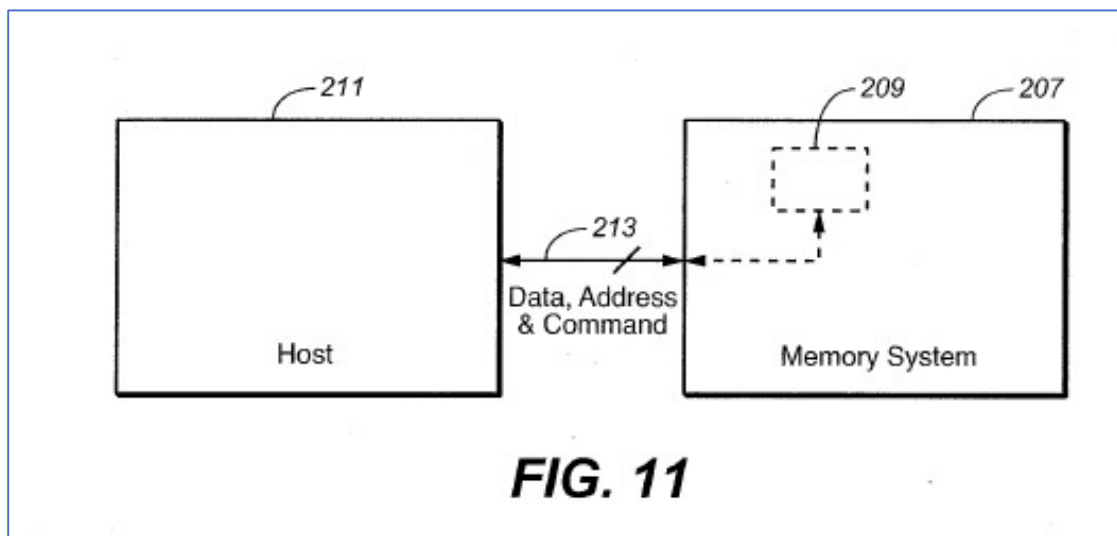
EX1005, Fig. 4, ¶63. The pages within two block may be logically linked to form a “metapage”:



EX1005, Fig. 5, ¶63.

Sinclair further discloses that when a host write data files to the memory system they may become “scattered among a larger number of metablocks when file

boundaries are not known.” EX1005, ¶¶86. Thus, Sinclair teaches that it is desirable for the host to “know the physical characteristics of the memory with which it is operating” and teaches that the host may query the memory for parameters relating to the characteristics of the memory. EX1005, ¶¶99-100. “Figure 11 illustrates a memory device 207 containing these parameter data in non-volatile storage space 209 that is accessible by the host.” EX1005, ¶101.

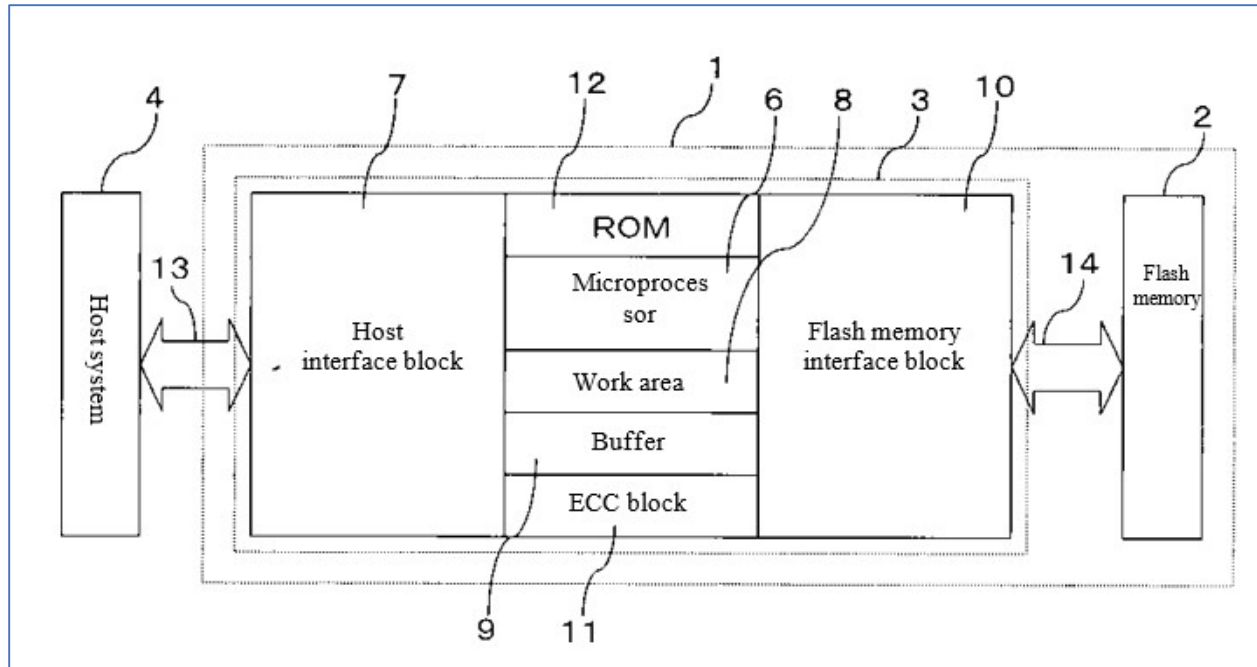


B. Claim Analysis

1. Claim 1

(a) [1pre]

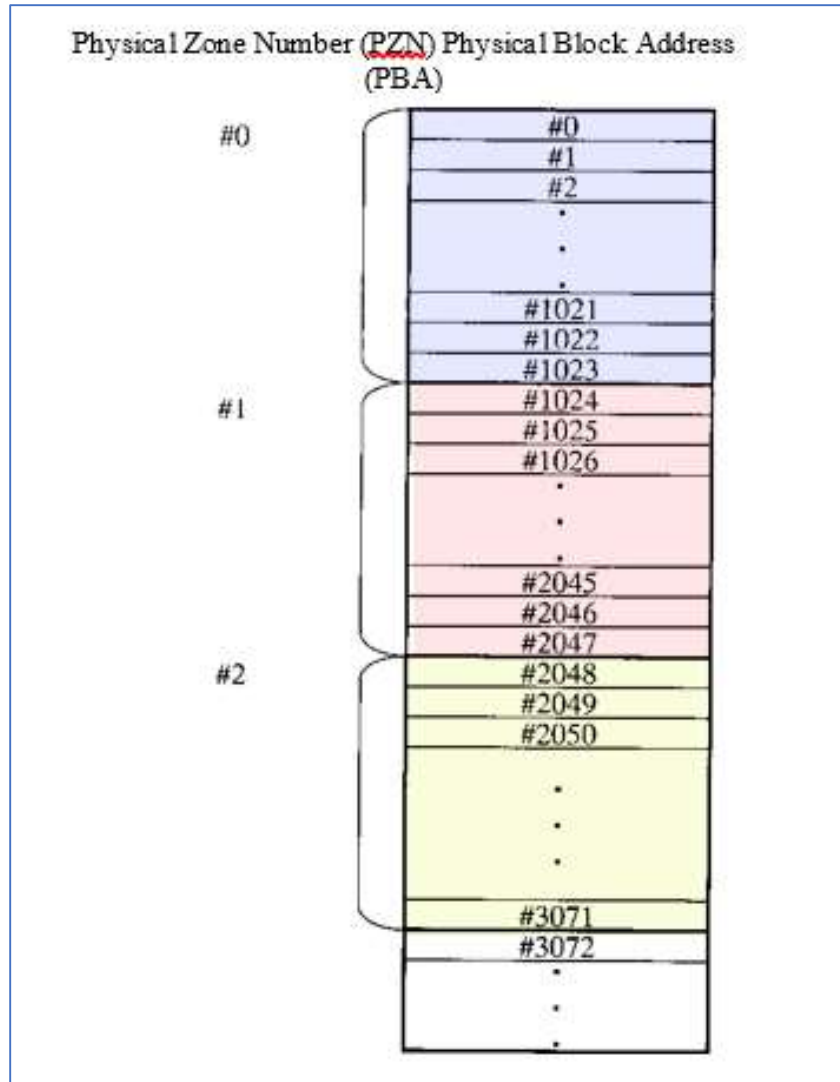
To the extent the preamble is limiting, Takeo discloses a flash memory system
1 [storage system] in communication with a host system 4:



EX1012, Fig. 1, ¶17.

(b) [1a]

Takeo discloses that the flash memory is divided into “**zones**.” EX1012, ¶41. Each zone is comprised of a set of blocks. *Id.* Takeo refers to the blocks as “**erase units**” (EX1012, ¶3) and Figure 3 illustrates an example where each zone (*e.g.*, PZN #0-2) has “1024 physical blocks”:



EX1012, Fig. 3, ¶41. Each physical zone is assigned to a logical zone, wherein “the number of logical blocks in the logical zone will be set to be less than the number of physical blocks in the physical zone” to account for “bad blocks.” EX1012, ¶42. Each physical block is comprised of “a plurality of **pages**.” EX1012, ¶¶3, 22-24.

(c) [1a.i]

Takeo discloses that the zones are associated with “logical address information” [a **logical address range**]. EX1003, ¶219. Takeo discloses that “the

address space on the host” is divided into “multiple zones (logical zones).” EX1012, ¶42. Takeo also discloses “logical address information” for each “logical block” that is “contained in each logical zone.” EX1012, ¶43. A POSITA would understand that the “logical address information” for each “logical block” of a “logical zone” forms a “logical address range” for the zone. EX1003, ¶219.

(d) [1a.ii]

Takeo discloses a zone size. Takeo teaches that a physical zone may comprise 1024 physical blocks and that the blocks in a logical zone may be set to a number less than this. EX1012, ¶¶41-42 (“logical zone containing 1000 logical blocks will be assigned to a physical zone containing 1024 physical blocks.”). Each block has a size in bytes. *Id.*, ¶24; EX1003, ¶220.

Takeo does not expressly discuss management commands. However, Son discloses the claimed management command. *See* Ground 1, Limitation 1[a.ii]. Applying Son’s teaching to Takeo, the flash memory system of Takeo would receive a management command from the host system that results in the erasure of erase units in a zone. EX1003, ¶221.

A POSITA would understand that a flash memory system should perform management operations such as wear leveling and garbage collection. EX1003, ¶222. Because Takeo is silent regarding these management operations, a POSITA would look to references like Son that describe these operations. *See Wavetronix*

LLC v. Smart Microwave Sensors Gmbh, 746 Fed. Appx 984, 987, 991 (Fed. Cir. 2018) (affirming motivation finding where base reference “taught a transmitter that generates a signal” but “failed to disclose its details,” and secondary reference “discloses the details.”); EX1003, ¶222. And as discussed above (*see* Ground 1, Limitation 1[a.ii]), the prior art includes express teachings, suggestions and motivations for combining Son’s disclosure of host-issued management operations to Takeo’s storage device, including that it would allow the system to incorporate device conditions in its management logic, improve execution speed, and avoid the controller becoming overloaded. EX1003, ¶222.

(e) [1a.iii]

Takeo discloses that a zone is in a full state when there are no host accessible pages available for writing [**the zone has zero empty blocks**] and an empty state when all host accessible pages are available for writing [*e.g.*, **all the blocks in the zone are empty**]. EX1003, ¶223.

Takeo discloses that the memory controller tracks “the number of... empty blocks per physical zone as shown in Figure 4.” EX1012, ¶44. Figure 4 shows eight physical zones (PZN#0-PZN#7) and the “number of empty blocks” in each zone. EX1012, Fig. 4. When the number of empty blocks is zero, the zone is in a “full state” and “there are no host accessible pages available for writing.” EX1003, ¶224.

PZN	Number of bad blocks	Number of empty blocks
#0	3	10
#1	2	17
#2	4	12
#3	1	11
#4	3	15
#5	2	16
#6	3	14
#7	2	22

EX1012, Fig. 4.

Takeo also discloses that the host is aware of the number of blocks in “logical zones,” (EX1012, ¶42) such that it discloses to a POSITA that a zone is in the empty state when the number of empty blocks in the table for a given zone is equal to the number of logical blocks in that zone. EX1003, ¶225.

To the extent this limitation requires that the memory controller contain an explicit state value (full/empty), the limitation is obvious in view of Son and/or Sinclair. EX1003, ¶226.

Son discloses maintaining whether superblocks [zones] are in a “CLEAN” state [empty state] or “SEALED” state [full state]. EX1006, ¶40, Fig 3C; *see also* Ground 1, Limitation 1[a.iii].

Similarly, Sinclair teaches that a “metablock” may be “in one of three states.” EX1005, ¶¶105-106. In one state the metablock is “storing valid file data without reclaimable capacity,” such as “reclaimable capacity from unprogrammed (erased) pages and/or obsolete (invalid data).” EX1005, ¶106. A POSITA would understand Sinclair’s disclosure of a metablock with no reclaimable capacity to disclose a **full state** because the metablock [zone] is filled with valid data and has no host accessible pages for writing. EX1003, ¶228.

Sinclair also discloses an empty state. Sinclair discloses that “a logical block is said to be ‘erased’ when no data are allocated to it. An ‘erased’ logical block is one that... is fully available to have addresses of data assigned to it.” EX1005, ¶104. Because in the erased state the metablock [zone] is “fully available to have addresses of data assigned to it,” the erased state is an **empty state** where all host accessible pages are available for writing. EX1003, ¶229.

A POSITA would have been motivated to combine Son and Sinclair’s teachings to maintain an explicit state value (empty/full) because it allows the system to quickly determine where data can and cannot be written. EX1003, ¶230. Further, an explicit state value helps a system decide which blocks to target for garbage collection and to avoid wasting time attempting to reclaim space from zones that are already empty. EX1003, ¶230. Indeed, Sinclair teaches that it is advantageous for a “reclaim operation” to maintain information about the “amount of valid data in

each block and the amount of erased space in each block” (EX1005, ¶169) and Takeo already discloses that its system tracks the number of empty blocks per zone (EX1012, Fig. 4). Thus, a POSITA would understand that maintaining a binary state value (full/empty) would have been a straightforward modification that would improve Takeo by simplifying decision-making logic. EX1003, ¶230.

(f) [1a.iv]

Takeo discloses an empty state. *See* Ground 4 Limitation 1[a.iii]. At minimum, this limitation is obvious. *Id.*; EX1003, ¶231.

(g) [1b]

Figure 1 illustrate memory controller 3. EX1012, Fig. 1, ¶¶17, 44, 46. The memory controller includes hardware and software [**logic operable to**]. EX1012, ¶33; EX1003, ¶232.

(h) [1b.i]

Son renders obvious returning the claimed information about the zones in response to a query command from the host for the reasons discussed in connection with Ground 1, Limitation 1[b.i].

Alternatively, the limitation is rendered obvious in view of Takeo, Son and Sinclair. EX1003, ¶233.

return information about the zones in response to a query command from the host: Takeo discloses that “[h]ost interface block 7 exchanges data, address information, status information, and external commands, etc., to and from host

system 4.” EX1012, ¶34. Takeo discloses a “vendor internal information acquisition command” [**query command from the host**] for returning information about the zones. EX1012, ¶¶45-50; EX1003, ¶¶234.

the information about the zones comprising a list of the zones accessible by the host: Figure 4 of Takeo discloses that the vendor internal information acquisition command [**query command from the host**] returns a list of physical zone numbers (PZN#0-7) [**list of the zones accessible by the host**]. EX1012, ¶¶44-51, Fig. 4.

PZN	Number of bad blocks	Number of empty blocks
#0	3	10
#1	2	17
#2	4	12
#3	1	11
#4	3	15
#5	2	16
#6	3	14
#7	2	22

a zone state: Takeo discloses the information returned to the host includes the number of empty blocks for each zone. EX1012, ¶¶45-51, Fig. 4. A POSITA would

understand this information discloses returning a zone state. when a zone contains zero empty blocks it is in a “full state.” EX1003, ¶236; *see also* Ground 4, Limitation 1[a.iii]-[iv].

To the extent this limitation requires that the memory controller include an explicit state value (full/empty), this limitation obvious. EX1003, ¶237. Son and Sinclair disclose maintaining an empty and full state. *See* Ground 4, Limitation 1[a.iii], 1[a.iv]. Because Takeo already discloses that the device tracks the number of empty blocks in each zone, it would have been obvious for Takeo to include an express state value (empty/full), as taught in Son and Sinclair. The combination reflects the use of a known technique (returning state information to the host) to improve similar devices (Takeo’s storage device would be improved by explicitly stating whether the number of blocks reflects an empty or full state) in the same way (Takeo already sends the number of empty blocks such that the improvement would not alter Takeo’s principles of operation). EX1003, ¶237.

information representing an extent to which the respective zones include host accessible pages available for writing: Takeo discloses the information returned to the host includes the number of empty blocks in a zone [**information representing the extent to which the specified zone includes host accessible pages available for writing**]. EX1012, ¶¶45-51, Fig. 4.

PZN	Number of bad blocks	Number of empty blocks
#0	3	10
#1	2	17
#2	4	12
#3	1	11
#4	3	15
#5	2	16
#6	3	14
#7	2	22

the zone size and logical addresses associated with respective zones:

Takeo does not expressly disclose that the vendor internal information acquisition command returns a zone size and logical addresses associated with respective zones. However, a POSITA would have found it obvious to return this information in view of Sinclair. EX1003, ¶239.

Sinclair discloses that the host may query the memory for information about the physical characteristics of the memory system using a “vendor specific command that is issued by the host 211 to the memory device 207.” EX1005, ¶¶99-100, Fig. 11. The data may include for each metablock [a zone] the “number of pages per metablock” and the “size of a physical page” [zone size] and the “the lowest logical

address that is mapped to a first page of a physical metablock [**logical addresses associated with respective zones**].” EX1005, ¶99.

A POSITA would have been motivated to modify Takeo to return this information as part of Takeo’s vendor internal information acquisition command. EX1003, ¶241. Sinclair discloses that when writing data files they may become “scattered among a larger number of metablocks when file boundaries are not known.” EX1005, ¶86. Sinclair teaches that if the host “know[s] the physical characteristics of the memory with which it is operating” (*e.g.*, the zone size and logical addressing information discussed above) it can avoid having files scattered across metablocks. EX1005, ¶¶99-100; EX1003, ¶241.

A POSITA would have had a reasonable expectation of success in making the combination because the combination represents a straightforward implementation of a host query for zone information using existing command types. EX1003, ¶242; EX1012, ¶46 (“vendor command”); EX1005, ¶100 (“vendor specific command”).

(i) [1b.ii]

Takeo in view of Sinclair renders obvious Limitation 1[b.ii]. EX1003, ¶243.

Sinclair discloses that erase blocks that span multiple planes may be grouped together into a metablock [**zone**] and that data may be written to the set of erase blocks [**sets of erase units**] of a specified metablock [**specified zone**] using a multi-

plane write command such that the data is written concurrently to two or more erase blocks of the set of erase blocks. *See* Ground 1, Limitation 1[b.ii].

A POSITA would have been motivated to combine Sinclair’s teaching with Takeo such that Takeo would write data to the blocks of a specified zone concurrently using a multi-plane write command because the prior art teaches that the combination would “increase[] parallelism.” EX1005, ¶62. In flash memory it is “preferabl[e]” to read and write data with the “maximum degree of parallelism” across a grouping of blocks [**zone**]. *Id.*, ¶76. *See also* EX1014, § 1 (“parallelism” is a “key factor” for “SSD performance”); EX1003, ¶246.

The combination of Takeo and Sinclair also reflects the application of a known technique, namely multi-plane writes, to Takeo’s known flash memory. EX1014, §2.4 (“[m]ost flash manufacturers provide advance commands, such as ... multiplane... to further improve the performance of SSD.”). The combination improves Takeo’s flash memory by increasing its parallelism and yields predictable results, the ability to write to blocks of the zone in parallel. EX 1003, ¶247. A POSITA would have had a reasonable expectation of success because the combination represents a straightforward implementation of multi-plane write commands to similar flash devices. EX1003, ¶248.

(j) [1b.iii]

Takeo discloses update the information representing the extent to which the specified zone includes host accessible pages available for writing [**updating the number of empty blocks in a zone**]. EX1003, ¶249. Takeo explains that “the number of empty blocks... will change over time, so it will be necessary to update the internal information with each change.” EX1012, ¶58.

(k) [1c]

Takeo discloses a controller that include instructions for controlling operation of the hardware [**logic**]. EX1012, ¶33; *see also* Limitation 1[b]; EX1003, ¶250.

2. Claim 2

Takeo in view of Sinclair renders obvious Claim 2. EX1003, ¶251.

As shown above, Takeo discloses that when the specified zone has no host accessible pages available for writing after writing data to the set of erase units of the specified zone, the storage device is operable to update a state of the specified zone to the full state. *See* Limitation 1[a.iii], 1[b.iii]. Takeo teaches that “if user data has been written to an empty block,” the “number of empty blocks in the corresponding physical zone will be decremented (decreased by one).” EX1012, ¶58. When the number of empty blocks is decremented to zero, the zone is transitioned into the full state. EX1003, ¶252.

Takeo does not, however, expressly disclose a multi-plane write command. However, Sinclair renders obvious that data may be written to a zone using a multi-plane write command. *See* Limitation 1[b.ii].

3. Claim 4

As discussed above, in view of Son a POSITA would have been motivated to modify Takeo so that the storage device receives a management command from a host. *See* Limitation 1[a.ii]. The management commands in the combined Takeo-Son system are associated with the memory size of at least a zone (grouping of blocks). EX1003, ¶254. Figure 7 of Son discloses a wear leveling command that selects a worn superblock [**zone**] and target superblock [**zone**] (step 710), copies the contents of the worn superblock to a target superblock, and then erases the worn superblock (step 715). EX1006, Fig. 7, ¶¶80-83.

4. Claim 5

In view of Son, a POSITA would have been motivated to modify Takeo so that the storage device receives a management command from a host. *See* Limitation 1[a.ii]. Son's management command from the host results in the logic issuing erase commands to the set of erase units of the specified zone. *See* Ground 1, Claim 5.

While Takeo and Son do not expressly disclose that the erase units of the zone are on "different planes," this limitation is obvious in view of Sinclair which discloses that the groups of blocks [**erase units**] that comprise a metablock [**zone**]

may be on different planes and “erased together.” EX1005, ¶¶62, 105. As a discussed above, a POSITA would have been motivated to implement the multi-plane commands described in Sinclair to increase parallelism. See Ground 4, Limitation 1[b.ii]; EX1003, ¶256.

5. Claim 6

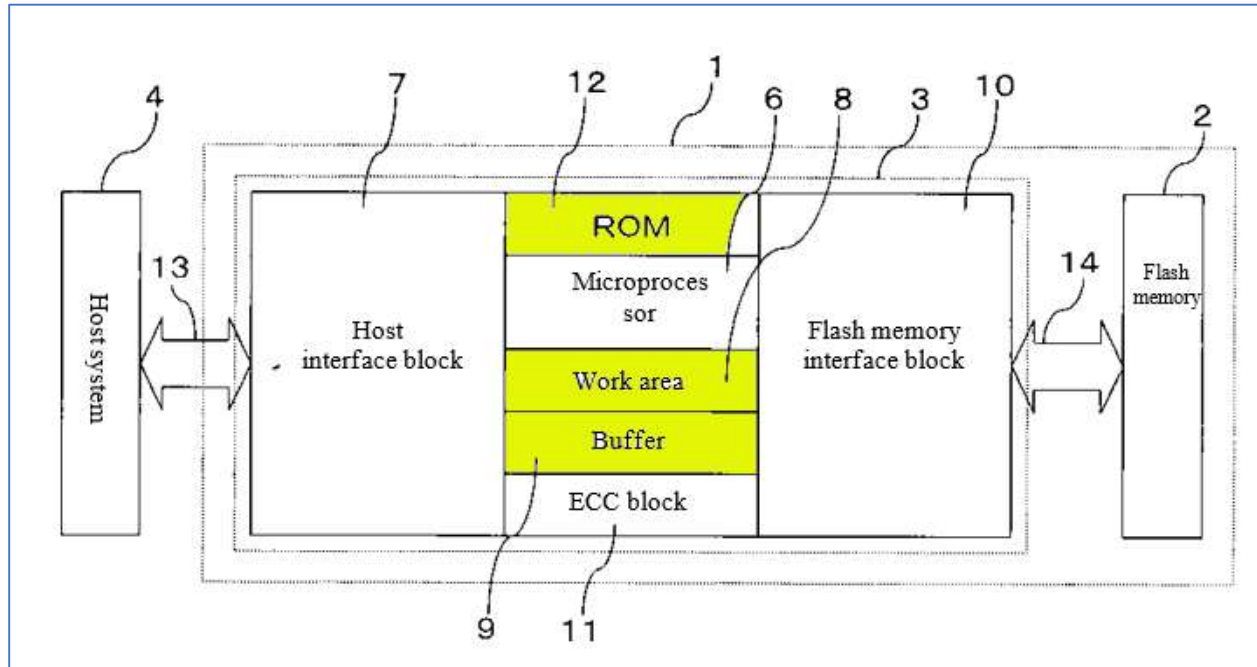
Takeo does not expressly disclose that the information about the zones is stored in non-volatile memory. However, this limitation is obvious in view of Sinclair, which teaches that the information about the metablocks [**zones**] may be “stored in the memory device itself, in a manner that they may be read by a host.” EX1006, ¶¶100, 111. Sinclair discloses that the information may be stored in a “**non-volatile storage** space 209 that is accessible by a host.” EX1006, ¶101, Fig. 11.

A POSITA would have found it obvious for apply Sinclair’s teaching about storing the information about the zones in non-volatile memory to Takeo’s storage device. EX1003, ¶258. A POSITA would have understood that storing the information in non-volatile (*i.e.*, persistent) memory avoids losing the data when power is lost or reset. EX1003, ¶258.

6. Claim 7

- (a) The storage device of claim 1, wherein the information about the zones is stored on a die that includes the logic.

Takeo discloses that memory controller 3 includes, in part, a ROM, a work area, and a buffer that are “integrated on a single semiconductor chip” [**die**]:



EX1012, Fig. 1, ¶¶17, 32. Takeo further discloses that the “internal information” about the zones [*e.g.*, **information about the zones**] is temporarily “held in work area 8” along with the software logic [] and outputted to buffer 9 for transmission to the host system. EX1012, ¶¶34, 36, 49, 54. ROM 12 also stores the program logic [**logic**]. EX1012, ¶33. Thus, the information about the zones is stored on a single semiconductor chip [**die**] that includes the logic. EX1003, ¶259.

7. Claim 8

Son discloses that the information about the superblocks [**zones**] comprises a “wear count,” which “indicates how many times a particular superblock has been erased” [**number of erase operations**]. EX1006, ¶39, Fig. 7.

A POSITA would have been motivated to combine the teachings of Son to Takeo’s storage device for the reasons discussed above. *See* Ground 1, Claim 8. A POSITA would be motivated to implement wear leveling to extend the lifespan of the flash memory by evenly distributing write and erase operations across the memory. EX1003, ¶261. A POSITA would understand that wear leveling involves tracking wear count, as disclosed by Son. EX1003, ¶261.

XI. GROUND 5: CLAIMS 3, 10-13, 16-19, 21-29 ARE OBVIOUS IN VIEW OF TAKEO, SON (EX1006), SINCLAIR (EX1005), AND OLBRICH (EX1010)

A. Claim Analysis

1. Claim 3

Sinclair renders obvious the logic is operable to write the data to the two or more erase units of the set of erase units of the specified zone. *See* Ground 4, Limitation 1[b.ii].

To the extent Sinclair does not also render obvious writing data to the two or more erase units using a “common page address” this limitation is at least obvious in further view of Olbrich. EX1003, ¶263-64. As discussed above, Olbrich discloses a multi-plane write command that uses a common page address to

concurrently write to pages of a superblock that are on different planes. *See* Ground 2, Claim 3. Olbrich's superblocks and super pages are similar to Sinclair's metablocks and metapages. EX1003, ¶263.

The prior art includes express teachings, suggestions and motivations to combine the teachings of Olbrich to Takeo's storage device, such that the storage device performs a multi-plane write command using a common page address. *See* Ground 2, Claim 3; EX1003, ¶264. The combination is obvious because it reflects the application of a known technique (multi-plane commands with a common page address) to a known device (Takeo flash memory system) ready for improvement (improvement by allowing parallel writes using a common page address) to yield predictable results (a device with a multi-plane write command). EX1003, ¶265.

A POSITA would also have had a reasonable expectation of success in combining Olbrich because it represents a straightforward implementation of multi-plane write commands to similar flash devices. EX1003, ¶266. Indeed, as Hu explains, SSD manufacturers at the time provided multi-plane command with a common page address with their SSDs. EX1014, §1.3.

2. Claim 10

(a) [10pre]

See Ground 4, Limitation 1[pre].

(b) [10a]

See Ground 4, Limitation 1[a].

(c) [10a.i]

See Ground 4, Limitation 1[a.i].

(d) [10a.ii]

See Ground 4, Limitation 1[a.ii].

(e) [10a.iii]

See Ground 4, Limitation 1[a.iii].

(f) [10a.iv]

See Ground 4, Limitation 1[a.iv].

(g) [10b]

See Ground 4, Limitation 1[b].

(h) [10b.i]

See Ground 4, Limitation 1[b.i].

(i) [10b.ii]

See Ground 4, Limitation 1[b.i].

(j) [10b.iii]

See Ground 4, Limitation 1[b.i].

(k) [10b.iv]

See Ground 4, Limitation 1[b.i].

(l) [10b.v]

See Ground 4, Limitation 1[b.i].

(m) [10b.vi]

See Ground 4, Limitation 1[b.i].

(n) [10b.vii]

See Ground 4, Claim 3.

(o) [10b.viii]

See Ground 4, Limitation 1[b.iii].

(p) [10b.ix]

See Ground 4, Claim 2.

(q) [10c]

See Ground 4, Limitation 1[c].

3. Claim 11

See Ground 4, Limitation 1[b.ii].

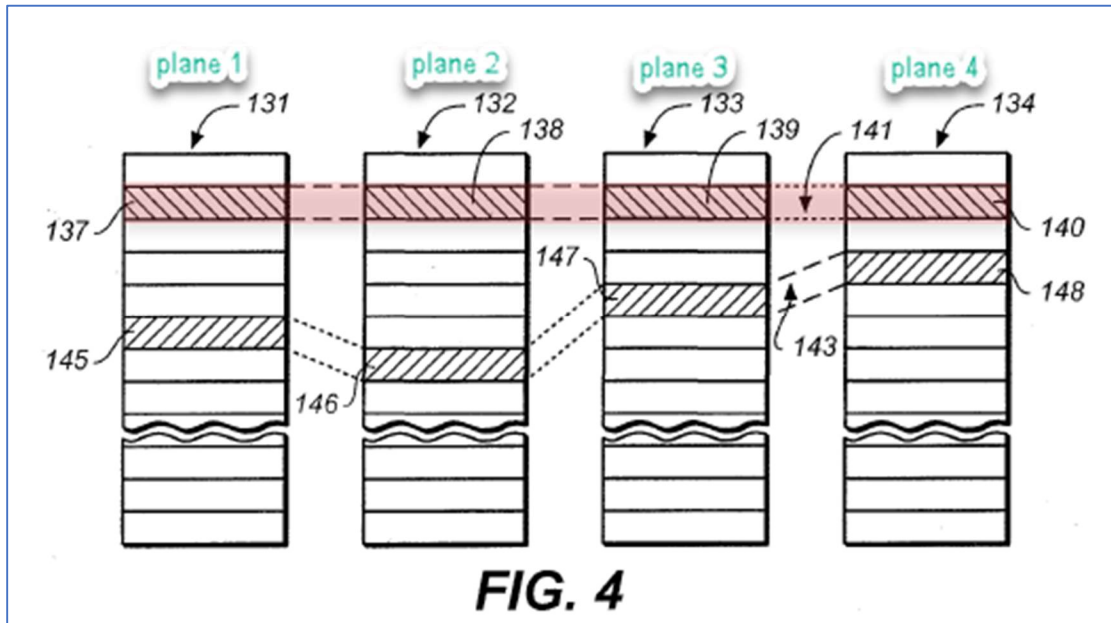
4. Claim 12

As discussed above, the combination of Takeo and Son renders obvious a management command from the host [*e.g.*, **wear leveling or garbage collection command**] that results in the logic issuing an erase command to erase one or more erase units of the specified zone [*e.g.*, **erase blocks of the specified superblock**]. See Ground 4, Limitation 1[a.ii]; EX1003, ¶286; EX1006, ¶¶68-70, ¶¶78-83, Figs. 6-7.

5. Claim 13

As discussed above, in the combined Takeo-Son system the management command from the host results in the logic issuing an erase command associated with erase units of the specified zone. See Ground 4, Limitation 1[a.ii].

Sinclair further discloses that the erase units of the specified metablock [specified zone] are on two or more different planes and erased together:



EX1005, Fig. 4, ¶62 (“all of the cells within a metablock are typically erased together.”). As discussed above, a POSITA would have been motivated to apply Sinclair’s teachings regarding a multi-plane architecture to the Takeo-Son system to improve parallelism of the SSD. See Ground 4, Limitation 1[b.ii]; EX1003, ¶289.

6. Claim 14

This limitation is obvious in view of Sinclair. EX1003, ¶290. Sinclair teaches that if a bad block [bad erase unit] is detected in a metablock a different block is responsively substituted:

One or more of these extra blocks may be provided as redundant blocks for *substitution for other blocks that may become defective during the lifetime of the memory.*

EX1005, ¶77. A POSITA would thus understand Sinclair to teach detecting a bad erase unit and marking it as unusable for storage of new write data. EX1003, ¶290. A POSITA would also understand that detection of a defective block “during the lifetime of the memory” discloses, or at least suggests, that the detection may be “*as a result of issuing the erase command.*” EX1003, ¶290.

A POSITA would have been motivated to apply Sinclair’s teachings to Takeo in this manner and would have had a reasonable expectation of success. EX1003, ¶291. Takeo already recognizes the importance of detecting blocks and that the number of bad blocks “will change over time.” EX1012, ¶¶30 (“The block status is information that indicates whether the physical block is a bad block”), 58 (“if a new bad block has occurred...”). And detecting bad blocks during erasure was well-known by the time of the ’614 patent. EX1016, ¶¶241-243 (“disclosing a process for detection and handling of bad pages and bad blocks” where “*[a]fter the block is erased, the status of each page in that block is checked to determine if it is bad.*”). A POSITA would have also understood that erasure performance can help identify bad blocks, such that integrating bad block detection with erasure improves efficiency and accuracy. *See, e.g.*, EX1017, 6:21-7:7. Where a “block needs more than predetermined number of erase cycles applied to fully erase the block,” it is an indication that the block has failed and should be marked bad. *Id.*; EX1003, ¶291.

7. Claim 16

Takeo discloses that the **host interface** comprises a work area 8 and **buffer** 9 that are used to temporarily store the information about the zones and queue the information about the zones for transmission to the host. EX1012, ¶¶35-36, 47-50.

Sinclair also discloses that the host interface comprises “random-access memory (RAM) 31 used primarily to buffer data being transferred between the memory and a host.” EX1005, ¶53, Fig. 1 (illustrating host interface & memory controller 8 communicating with host), Fig. 2 (illustrating host interface 35 coupled to RAM 31). A POSITA would have been motivated to combine Sinclair’s teaching with Takeo because using a buffer to queue information for transmission from flash memory to a host is a common design that optimizes performance and error handling. EX1003, ¶293.

8. Claim 17

See Ground 4, Claim 6.

9. Claim 18

See Ground 4, Claim 7.

10. Claim 19

See Ground 4, Claim 8.

11. Claim 21

(a) [21pre]

See Ground 4, Limitation 1[pre].

(b) [21a]

See Ground 4, Limitation 1[a] and 1[a.i].

(c) [21b]

Takeo discloses **memory controller 3** in communication with the **flash memory 2**. EX1012, Fig. 1, ¶¶17, 44, 46; EX1003, ¶299.

(d) [21b.i]

See Ground 4, Limitation 1[b.i].

(e) [21b.ii]

Takeo in combination with Son and/or Sinclair discloses logic operable to cause the storage device to output the information about the zones in response to a query command from a host. *See* Ground 4, Limitation 1[b.i].

Olbrich renders obvious write to the set of erase units of the zone of the zones using a common page address. *See* Ground 5, Claim 3.

Takeo discloses update the information representing the extent to which the zone comprises host accessible pages available for writing in response to writing to the set of erase units of the zone. *See* Ground 4, Limitation 1[b.iii].

12. Claim 22

See Ground 4, Limitation 1[b.i].

13. Claim 23

See Ground 4, Limitation 1[a.iii].

14. Claim 24

See Ground 4, Limitation 1[a.iii], Ground 5, Claim 3.

15. Claim 25

See Ground 4, Limitation 1[a.iv].

16. Claim 26

Son discloses that the information about the zones comprises information indicative of an amount of time since data was written to the set of erase units of the zone [**last modification time**]. EX1006, ¶39, Fig. 3C. Sinclair also discloses that information indicating the “oldest” block is maintained in a block list. EX1005, ¶¶182, 169, Fig. 31

A POSITA would have been motivated to apply the teachings of Son and/or Sinclair to Takeo. A POSITA would understand that flash cells leak charge over time such that data that sits too long without being rewritten risks bit errors. By tracking the elapsed time since a zone was last written to, aging data can be refreshed to avoid corruption. Moreover, as disclosed in Son (EX1006) and Yamada (EX1015), it is beneficial to evaluate last written time or frequency of data writes when performing wear leveling or garbage collection. EX1006, ¶¶80-81, EX1015, ¶¶155, 163-165. Data that has not been written for a long time can be written to blocks that have a high write count. EX1003, ¶309.

17. Claim 27

See Ground 4, Limitation 1[b.i]

18. Claim 28

See Ground 4, Claim 4.

19. Claim 29

Sinclair discloses that erase blocks of a metablock are “erased together.” EX1005, ¶¶62, 105. Thus, in the Takeo-Sinclair combination, the erase blocks of the zones are erased together. EX1003, ¶312.

A POSITA would have been motivated to apply Sinclair’s teachings to Takeo in this manner for the reason discussed above. *See* Ground 4, Limiation 1[b.ii]. The prior art includes express teachings, suggestions and motivations to apply Sinclair’s teachings regarding multi-plane architectures and commands to Takeo’s storage system to achieve increased parallelism. *Id.*

The combination of Takeo and Sinclair also reflects the application of a known technique, namely parallel erasure, to Takeo’s known flash memory. By the time of the ’614 patent, multi-plane erase commands were well-known. EX 1003, ¶314 (citing EX1014, EX1008). Applying Sinclair’s teaching of multi-plane erase to Takeo improves Takeo’s flash memory by increasing its parallelism and yields predictable results, the ability to erase blocks of the zone in parallel. EX 1003, ¶314.

A POSITA would have had a reasonable expectation of success because the combination represents a straightforward implementation of an erase command to similar flash devices. EX1003, ¶315.

XII. GROUND 6: CLAIMS 9, 15, 20, AND 30 ARE OBVIOUS IN VIEW OF TAKEO, SON (EC), SINCLAIR (EX1005), OLBRICH (EX1010), AND YAMADA (EX1015)

A. Claim Analysis

1. Claim 9

(a) 9[a]

In view of Yamada and the general knowledge of a POSITA, it would have been obvious for Takeo's flash device [**storage device**] to generate and track the elapsed time since data was last written to a zone [**generate metadata indicating an amount of time since the specified zone was written to**] and indicate to the host that maintenance is needed when a threshold value is exceeded. EX1003, ¶316.

Yamada discloses a maintenance request unit of a memory module that compares metadata to a "criterion value" and sends "a maintenance request indicating that the nonvolatile memory needs maintenance to the access device" when the criterion value is exceeded. EX1015, ¶¶32-33, 81-82, Figs. 1-2. A maintenance determination unit of the access device then schedules a maintenance request that "includes refresh and/or wear leveling of the nonvolatile memory." EX1015, ¶30.

In the preferred embodiment of Yamada the maintenance request is sent when metadata indicating the number of errors occurring in the data exceeds a threshold. However, a POSITA would understand that the criterion used to trigger a maintenance request may be any metadata that indicates a need for maintenance.

EX1003, ¶318. Memory cells will leak over time such that data may become corrupted if it is not written for a long time. EX1015, ¶9. Thus, a POSITA would have been motivated to track “the elapsed time from when the data has been written previously” and refresh data “when the elapsed time is greater than or equal to a criterion value.” EX1015, ¶164. Similarly, a POSITA would understand that metadata regarding the elapsed time may also be used to initiate other maintenance operations, such as wear leveling. EX1006, ¶¶47, 69; EX1004, 19:4-15. Thus, a POSITA would have found it obvious for Takeo’s flash device to generate metadata indicating the elapsed time since data was last written to a zone. EX1003, ¶318.

(b) 9[b]

As discussed in the prior limitation, in view of Yamada and the general knowledge of a POSITA it would have been obvious for Takeo’s flash device **[storage device]** to track the elapsed time since data was last written to a zone and then indicate that the storage device recommends that the host send a second management command directed to the specified zone when a threshold value is exceeded **[the storage device recommends that the host send a second management command directed to the specified zone]**. *See also* EX1015, ¶164; EX1003, ¶319.

A POSITA would have been motivated to compare the metadata to a threshold on the storage device and then recommend that the host issue a management

command and had a reasonable expectation of success for the reasons discussed above. *See* Ground 3, Claim 9; EX1015, ¶¶10, 36. A POSITA would have been motivated to implement this type of shared memory management because it avoid the storage device having to expose its performance characteristics to the host. EX1003, ¶320.

2. Claim 15

Claims 15 and 30 are similar to Claim 9. The difference between these claims is that in Claims 15 and 30 the metadata at issue is “metadata indicating a count of erasures of the erase units,” while in Claim 9 the metadata is “metadata indicating an amount of time since the specified zone was written to.” Claim 15 is the obvious for substantially the same reasons discussed in connection with Claim 9. EX1003, ¶321.

As discussed in connection with Claim 9, Yamada discloses transmitting a maintenance request to the host when metadata indicating the number of errors occurring in the data exceeds a threshold. EX1015, ¶¶32-33, 81-82, Figs. 1-2. However, a POSITA would have understand that the criterion used to determine maintenance may be any metadata that would indicate a need for a maintenance operation. A POSITA would have understood the benefit of implementing wear leveling in a flash memory device to distribute the wear across zones evenly and prevent premature failure of high-usage areas. EX1003, ¶322. A POSITA would

understand that wear leveling involved maintaining a wear count for a zone **[metadata indicating a count of erasures of the erase units]** and indicating the need for wear leveling when the wear count exceeds a threshold value. EX1006, Fig. 7, ¶¶64, 79; EX1004, 11:63-67. *See also id.*, 13:14-24, claim 1. And for the reasons discussed in connection with Claim 9, a POSITA would have been motivated to implement shared management in which the memory controller performs the comparison and the host sends the management command. *See* Ground 6, Claim 9. Thus, a POSITA would have found it obvious to for Takeo’s flash device **[storage device]** to track the wear count of a zone **[metadata indicating a count of erasures of the erase units]** and then indicate that the storage device recommends that the host send a second management command directed to the specified zone when a threshold value is exceeded **[recommends that the host send a second management command directed to the specified zone]**. EX1003, ¶322.

XIII. MANDATORY NOTICES

A. Real Party-in-Interest (37 C.F.R. §42.8(b)(1))

Petitioner certifies that the real parties-in-interest are Samsung Electronics Co., Ltd., And Samsung Electronics America, Inc.

B. Related Matters (37 C.F.R. §42.8(b)(2))

1. Related District Court Proceedings

Radian Memory Systems LLC (“Radian” or “PO”) filed a lawsuit in the Eastern District of Texas alleging that Petitioner infringes the ’614 Patent. *See, e.g.,*

Radian Memory Systems LLC, v. Samsung Electronics Co., Ltd., and Samsung Electronics America, Inc., Case No. 2:24-cv-1073 (E.D. Tex.).

Petitioner is currently unaware of any other lawsuits involving the ‘614 Patent.

2. Related Patent Office Proceedings

Petitioner is unaware of any co-pending patent office proceedings involving the ‘614 Patent. This is the first IPR petition challenging a claim of the ‘614 Patent.

C. Lead and Back-up Counsel and Service Information

Petitioner provides the following counsel and service information. Pursuant to 37 C.F.R. §42.10(b), a Power of Attorney accompanies this Petition.

LEAD COUNSEL	BACK-UP COUNSEL
James M. Glass (Reg. No. 46,729) jimglass@quinnemanuel.com QUINN EMANUEL URQUHART & SULLIVAN, LLP 295 Fifth Avenue, 9th Floor New York, NY 10016 Tel: (212) 849-7000 Fax: (212) 849-7100	John T. Mckee (Reg. No. 65,926) johnmckee@quinnemanuel.com QUINN EMANUEL URQUHART & SULLIVAN, LLP 295 Fifth Ave., 9th Floor New York, NY 10016 Tel: (212) 849-7000 Fax: (212) 849-7100 Nima Hefazi (Reg. No. 63,658) QUINN EMANUEL URQUHART & SULLIVAN, LLP 865 S. Figueroa St., 10 th Floor Los Angeles, California 90017 nimahefazi@quinnemanuel.com Tel: (213) 443-3000 Fax: (213) 443-3100

D. Payment of Fees (37 C.F.R. §42.15(a))

The undersigned authorizes the Office to charge the fee required for this Petition for *Inter Partes* Review to Deposit Account No. 50-5708. Any additional fees that might be due are also authorized.

XIV. CONCLUSION

For these reasons, Petitioner requests *inter partes* review of the ‘614 Patent.

Date: July 14, 2025

By: / John McKee

John McKee (Reg. No. 65,926)

johnmckee@quinnemanuel.com

QUINN EMANUEL URQUHART & SULLIVAN,
LLP

295 Fifth Ave., 9th Floor

New York, NY 10016

Tel: (212) 849-7000

Fax: (212) 849-7100

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. §42.24(d), Petitioner hereby certifies, 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 13,981. Pursuant to 37 C.F.R. §42.24(d), this word count 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.

Date: July 14, 2025

Respectfully submitted,

By: /s/ John McKee

Attorney for Petitioners

CERTIFICATE OF SERVICE

Pursuant to 37 C.F.R. §§ 42.6(e), 42.105(a), the undersigned hereby certifies service on the Patent Owner of a copy of this Petition and its respective exhibits at the official correspondence address for the attorneys of record for the '614 patent as shown in USPTO P-TACTS via FedEx:

Radian Memory Systems, LLC
6275 West Plano Parkway
Suite 500
Plano, TX 75093

Courtesy copies were also sent via electronic mail to Patent Owner's counsel of record in the related district court proceeding:

Andrea Fair

andrea@millerfairhenry.com
radian@caldwellcc.com

DATED: July 14, 2025

Respectfully submitted,

By /s/ John McKee

John McKee (Reg. No. 65,926)
johnmckee@quinnemanuel.com

QUINN EMANUEL URQUHART &
SULLIVAN, LLP
295 Fifth Ave., 9th Floor
New York, NY 10016
Tel: (212) 849-7000
Fax: (212) 849-7100

Attorney for Petitioners

APPENDIX A – CLAIM LISTING

Claim 1	
1pre	A storage device comprising:
1a	flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone:
1a.i	is associated with a logical address range,
1a.ii	has a zone size for which the storage device is operable to receive a management command from a host that results in the storage device performing an erasure of one or more erase units of the set of erase units,
1a.iii	is in a full state when there are no host accessible pages available for writing, and
1a.iv	is in an empty state when all host accessible pages are available for writing; and
1b	logic operable to:
1b.i	return information about the zones in response to a query command from the host, the information about the zones comprising a list of the zones accessible by the host, a zone state, the zone size, logical addresses associated with respective zones, and information representing an extent to which the respective zones include host accessible pages available for writing;
1b.ii	write data to the set of erase units of a specified zone using a multi-plane write command such that the data is written concurrently to two or more erase units of the set of erase units; and
1b.iii	update the information representing the extent to which the specified zone includes host accessible pages available for writing,
1c	wherein the logic comprises at least one of hardware logic or instructions stored on non-transitory machine-readable media that, when executed, control operation of hardware circuitry.
Claim 2	

2.	The storage device of claim 1, wherein when the specified zone has no host accessible pages available for writing after writing data to the set of erase units of the specified zone using the multi-plane write command, the storage device is operable to update a state of the specified zone to the full state.
Claim 3	
3	The storage device of claim 1, wherein the logic is operable to write the data to the two or more erase units of the set of erase units of the specified zone using a common page address.
Claim 4	
4	The storage device of claim 1, wherein the zone size is a minimum memory size associated with the management command.
Claim 5	
5	The storage device of claim 1, wherein the management command from the host results in the logic issuing erase commands to the set of erase units of the specified zone that are on two or more different planes.
Claim 6	
6	The storage device of claim 1, wherein the information about the zones is stored on non-volatile memory of the storage device.
Claim 7	
7	The storage device of claim 1, wherein the information about the zones is stored on a die that includes the logic.
Claim 8	
8	The storage device of claim 1, wherein the information about the zones comprises information indicative of a number of erase operations performed on a particular zone of the zones.
Claim 9	
9[a]	The storage device of claim 1, wherein the storage device is operable to: generate metadata indicating an amount of time since the specified zone was written to, and

9[b]	in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.
Claim 10	
10[pre]	A storage device comprising:
10a	flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, and each erase unit of the set of erase units comprising host accessible pages, wherein each zone:
10a.i	is associated with a logical address range,
10a.ii	has a zone size associated with a management command from a host,
10a.iii	transitions to a full state in response to no longer having host accessible pages available for writing, and
10a.iv	in response to the storage device receiving the management command associated therewith, transitions to an empty state such that all host accessible pages are available for writing; and
10b	logic operable to:
10b.i	return information about zones stored by the storage device in response to a query command from the host, the information about the zones comprising:
10b.ii	a list of the zones accessible by the host,
10b.iii	a respective logical address associated with a zone of the zones stored,
10b.iv	the zone size,
10bv	information indicating, for respective zones in which there are no host accessible pages available for writing the respective zone, that the respective zones are in the full state, and
10b.vi	information representing, for respective zones, an extent to which the respective zones include host accessible pages available for writing;
10b.vii	write data to the set of erase units of a specified zone using a common page address;
10b.viii	update the information representing the extent to which the specified zone includes host accessible pages available for writing; an

10b.ix	when the specified zone has no host accessible pages available for writing the zone after writing data to the set of erase units of the specified zone, update the information indicating that the specified zone is in the full state,
10c	wherein the logic comprises at least one of hardware logic or instructions stored on non-transitory machine-readable media that, when executed, control operation of hardware circuitry.
Claim 11	
11	The storage device of claim 10, wherein the logic is operable to write the data to the set of erase units of the specified zone using a multi-plane write command such that the data is concurrently written to two or more erase units of the set of erase units.
Claim 12	
12	The storage device of claim 10, wherein the management command from the host results in the logic issuing at least one erase command to erase one or more erase units of the set of erase units of the specified zone.
Claim 13	
13	The storage device of claim 10, wherein the management command from the host results in the logic issuing an erase command associated with erase units of the specified zone that are on two or more different planes.
Claim 14	
	The storage device of claim 13, wherein, as a result of issuing the erase command, the storage device is operable to detect a bad erase unit and mark the bad erase unit as unusable for storage of new write data.
Claim 15	
15[a]	The storage device of claim 13, wherein, the storage device is operable to: generate metadata indicating a count of erasures of the erase units, and
15[b]	in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.
Claim 16	

16	The storage device of claim 10, further comprising a host interface that comprises buffers to receive the information about the zones and queue the information about the zones for transmission to the host.
Claim 17	
17	The storage device of claim 10, wherein the information about the zones is stored on non-volatile memory of the storage device.
Claim 18	
18	The storage device of claim 10, wherein the information about the zones is stored on a die that includes the logic.
Claim 19	
19	The storage device of claim 10, wherein the information about the zones comprises information indicative of a number of erase operations performed on a particular zone of the zones.
Claim 20	
20[a]	The storage device of claim 10, wherein the storage device is operable to: generate metadata indicating an amount of time since the specified zone was written to, and
20[b]	in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a second management command directed to the specified zone.
Claim 21	
21pre	A storage device comprising:
21a	flash memory divided into zones, each zone of the zones being a subdivision comprising a set of erase units, each erase unit of the set of erase units comprising host accessible pages, and the zones are each associated with a respective logical address range; and
21b	a memory controller in communication with the flash memory, the memory controller comprising:
21b.i	storage arranged to retain information about the zones, the information about the zones comprising zone size, a logical address associated with a zone of the zones, and information representing an extent to which the zone of the zones comprises host accessible pages available for writing; and

21b.ii	logic operable to cause the storage device to output the information about the zones in response to a query command from a host, write to the set of erase units of the zone of the zones using a common page address, and update the information representing the extent to which the zone comprises host accessible pages available for writing in response to writing to the set of erase units of the zone.
Claim 22	
22	The storage device of claim 21, wherein the information about the zones comprises a state of the zone.
Claim 23	
23	The storage device of claim 22, wherein the state of the zone is a full state when the zone has no host accessible pages available for writing the zone.
Claim 24	
24	The storage device of claim 23, wherein the state of the zone is updated to the full state in response to the zone having no host accessible pages available for writing the zone after writing to the set of erase units of the zone of the zones using the common page address.
Claim 25	
25	The storage device of claim 22, wherein the state of the zone is an empty state when all host accessible pages of the zone are available for writing.
Claim 26	
26	The storage device of claim 21, wherein the information about the zones comprises information indicative of an amount of time since data was written to the set of erase units of the zone.
Claim 27	
27	The storage device of claim 21, wherein the information about the zones comprises logical addresses associated with respective ones of a plurality of zones of the zones, and information representing an extent to which each of the plurality of zones comprises host accessible pages available for writing.
Claim 28	

28	The storage device of claim 21, wherein the zone is a minimum memory space associated with a management command from the host.
Claim 29	
29	The storage device of claim 21, wherein the logic is operable to cause the storage device to concurrently erase the set of erase units of the zone.
Claim 30	
30[a]	The storage device of claim 21, wherein the storage device is operable to: generate metadata indicating an amount of time since the zone was written to, and
30[b]	in response to comparing the metadata to a threshold, provide an indication that the storage device recommends that the host send a management command directed to the zone.