UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

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

v.

RADIAN MEMORY SYSTEMS, LLC,
Patent Owner.

Case IPR2025-01376

U.S. Patent No. 11,307,995

DECLARATION OF DR. HEINER LITZ IN SUPPORT OF PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO. 11,307,995

Samsung Ex. 1003

TABLE OF CONTENTS

Page

I. SUMMARY OF OPINIONS 1

II. QUALIFICATIONS 1

III. LEGAL STANDARD 4

A. Person of Ordinary Skill in the Art 4

B. Claim Construction 5

C. Anticipation 5

D. Obviousness 6

IV. TECHNOLOGY BACKGROUND AND STATE OF THE ART 9

V. OVERVIEW OF THE '995 PATENT 13

A. Priority 13

B. Prosecution History 13

C. Specification 13

D. The Challenged Claims 14

VI. LEVEL OF ORDINARY SKILL IN THE ART 15

VII. CLAIM CONSTRUCTION 15

VIII. OVERVIEW OF PRIOR ART REFERENCES 16

A. Reiter 16

B. Sinclair-367 17

C. Yamada 20

D. ZBC, SBC-4, and SPC-4 22

IX. INVALIDITY GROUNDS 24

A. GROUND I: Claims 1-5, 7-11, 13-15, 18-20, 26-30 ARE RENDERED OBVIOUS BY Reiter In Light Of ZBC, SBC-4, and SPC-4 24

1. To Combine 24

2. Claim 1 29

3. Claim 2 53
4. Claim 3 55
5. Claim 4 56
6. Claim 5 58
7. Claim 7 59
8. Claim 8 62
9. Claim 9 63
10. Claim 10 63
11. Claim 11 66
12. Claim 13 66
13. Claim 14 67
14. Claim 15 70
15. Claim 18 73
16. Claim 19 76
17. Claim 20 77
18. Claim 26 79
19. Claim 27 81
20. Claim 28 81
21. Claim 29 85
22. Claim 30 88

B. GROUND II: CLAIMS 6, 21 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC AND SINCLAIR-367 92
1. Motivation To Combine 92
2. Claim 6 93
3. Claim 21 96

X. CONCLUSION 96

EXHIBIT LIST

Ex.	Description
1001	U.S. Patent No. 11,307,995 ("'995 Patent")
1002	Prosecution History of U.S. Patent No. 11,307,995
1003	Expert Declaration of Dr. Heiner Litz
1004	U.S. Patent No. 8,301,861 ("Reiter")
1005	Intentionally Omitted
1006	Working Draft Revision 1a of the Small Computer System Interface Zoned Block Commands ("ZBC")
1007	Working Draft Revision 2 of SCSI Block Commands – 4 ("SBC-4")
1008	Working Draft Revision 37 of SCSI Primary Commands – 4 ("SPC-4")
1009	U.S. Publication No. 2005/0144367 ("Sinclair-367")
1010	U.S. Publication No. 2011/0271032 ("Yamada")
1011	Patent Owner's District Court Infringement Chart for '995 Patent
1100	Declaration of Rich Deglin
1101	"T10 Working Drafts", T10, http://www.t10.org/drafts.htm (last visited Jul. 17, 2025)
1102	"File Access Monitor", T10, https://www.t10.org/cgi-bin/ac.pl?t=f&f=sbc5r08.pdf (last visited Jul. 17, 2025)
1103	"T10 Working Drafts", T10 (archived Jul. 15, 2014), https://web.archive.org/web/20140715124727/http:/www.t10.org/drafts.htm (last visited Jul. 17, 2025)
1104	"File Access Monitor", T10 (archived May 19, 2014), https://web.archive.org/web/20140519064154/http://www.t10.org/cgi-bin/ac.pl?t=f&f=spc4r37.pdf (last visited Jul. 17, 2025)
1105	"File Access Monitor", T10 (archived April 16, 2015), https://web.archive.org/web/20150416184431/http://www.t10.org/cgi-bin/ac.pl?t=f&f=sbc4r02.pdf (last visited Jul. 17, 2025)
1106	"File Access Monitor", T10 (archived May 19, 2014),

Ex.	Description
	https://web.archive.org/web/20140519063435/http://www.t10.org/cgi-bin/ac.pl?t=f&f=zbc-r01a.pdf (last visited Jul. 17, 2025)
1107	May 17, 2014 Email From T10
1108	May 24, 2014 Email From T10
1109	"Introduction to T10", T10, https://www.t10.org/intro.htm (last visited Jul. 20, 2025)
1110	U.S. Pub. No. 2012/0096217 ("Son")
1111	International Publication No. WO 2008/082996 ("Sinclair-996")
1200	"Google Patents Search", Google, https://patents.google.com/?q=SCSI&q=%22Small+Computer+System+Interface%22&before=publication:20140901&num=100 (last visited Jul 7, 24, 2025) (searching for "SCSI" or "Small Computer System Interface" patent publications prior to Sep. 1, 2014)
1201	Somayaji & Murty, "Securing Internet Protocol (IP) Storage: A Case Study", 1 (2009)
1202	White et al., "Performance Comparison of IDE and SCSI" (2001)
1203	L. Columbus, "Exploring the World of SCSI", 2, 8, 10 (2000)
1204	U.S. Publication No. 2014/0215147
1205	U.S. Publication No. 2009/0271659
1206	James S. Plank, "The RAID-6 Liberation Codes" (2008)
1207	U.S. Patent No. 5,857,208, 1:22-50
1208	Patterson et al., "A Case for Redundant Arrays of Inexpensive Disks (RAID)" 110-15 (1998)
1209	Chen et al., "RAID: High-Performance, Reliable Secondary Storage" (Jun. 1994)
1210	U.S. Publication No. 2009/0271562

DECLARATION OF DR. HEINER LITZ, Ph.D.

I, HEINER LITZ, Ph.D. hereby declare as follows:

1. My name is Heiner Litz. I have been retained to testify as an expert in this Inter Partes Petition on behalf of Samsung Electronics Co., Ltd. ("SEC") and Samsung Electronics America, Inc. ("SEA") (collectively, "Samsung" or "Petitioner").

2. I have prepared this declaration in connection with Samsung's related petition for inter partes review ("IPR") of U.S. Patent No. 11,307,995 (the "'995 Patent," EX1001), which I am informed is being filed concurrently with this declaration. I have been asked to provide certain opinions relating to the patentability of the '995 Patent. Specifically, I have been asked to provide my opinion regarding whether claims 1-11, 13-15, 18-21, 26-30 of the '995 patent ("the Challenged Claims") are invalid as obvious in view of the prior art.

I. SUMMARY OF OPINIONS

3. It is my opinion that the Challenged Claims are rendered obvious pursuant to Grounds I-V and the reasons discussed herein.

II. QUALIFICATIONS

4. My education, experience, and qualifications are detailed in my résumé, included as Exhibit 1 to this declaration. While I incorporate Exhibit 1 by reference, below is a brief summary of my background, including my background and experience relevant to this case.

5. I earned a degree in Computer Engineering from the University of Mannheim, Germany, in 2006. In 2012, I completed a Ph.D. in Computer Engineering from the same institution, graduating Summa Cum Laude. Between 2006 and 2012, my graduate research focused on computer architecture, hardware chip design, scalable storage systems, and high-performance computing networks.

6. In 2013, I began my academic career as a Postdoctoral Researcher at Stanford University, working in the research groups of Professors David Cheriton and Christos Kozyrakis. From 2012 to 2016, my research centered on programmable "Open-Channel" solid-state drives (SSDs). I introduced techniques to reduce access latency by shifting performance-critical tasks such as data placement from the SSD to the host processor. This work led to "RAIL: Predictable, Low Tail Latency for NVMe Flash", which achieved a 7× reduction in read access latency through host-orchestrated SSD data placement.

7. In 2017, I served as Visiting Faculty at Google, working on programmable storage devices within the Platforms Infrastructure Group.

8. In 2018, I joined the Computer Science Department at the University of California, Santa Cruz (UCSC), as an Assistant Professor. I was promoted to Associate Professor with tenure in 2023. In 2024, I was named the Kumar Malavalli Endowed Professor for Storage Systems Research, a position supported by a $1 million gift from Kumar Malavalli.

9. Since 2023, I have served as Director of the Center for Research in Storage Systems (CRSS) at UCSC, a center comprised of 5 faculty and 15 Ph.D. students. CRSS advances research in storage systems and is supported by the National Science Foundation (NSF) and industry sponsors. Our work spans a range of topics, including programmable storage and emerging technologies such as Open-Channel, Zoned Namespace (ZNS), and Flexible Data Placement (FDP) SSDs. In the context of ZNS, my group proposed the group append feature to optimize ZNS SSD integration with file systems (*e.g.*, "Append is Near: Log-based Data Management on ZNS SSDs").

10. In 2024, I was a Visiting Professor at the Massachusetts Institute of Technology (MIT) in Cambridge.

11. Over the course of my academic career, I have published more than 80 peer-reviewed research papers, journal articles, and patents in the areas of storage systems and computer hardware. I have advised over 15 Ph.D. students and secured more than $9 million in research funding. My honors include the NSF CAREER Award, Intel Outstanding Researcher Award, and multiple Faculty Awards from Google and Facebook. I received the Best Paper Award at MICRO 2023 and several IEEE MICRO Top Picks. I have also served as Program Chair for HotChips 2023 and NVMW 2024, and as Associate Editor for IEEE Computer Architecture Letters (CAL).

III. LEGAL STANDARD

12. Counsel for Samsung has informed me of the following legal standards. I am not an attorney, and I am relying only on instructions from Samsung's attorneys for these legal standards.

A. Person of Ordinary Skill in the Art

13. I understand that issues of claim construction and invalidity are analyzed from the perspective of a person of ordinary skill in the art ("POSITA").

14. I understand that a POSITA is a hypothetical person who is used to analyzing the prior art without the benefit of hindsight. A POSITA is presumed to be one who thinks along the lines of conventional wisdom in the art.

15. I understand that the hypothetical POSITA is presumed to have knowledge of all references that are sufficiently related to one another and to the pertinent art, and to have knowledge of all arts reasonably pertinent to the particular problem that the claimed invention addresses.

16. I understand that factors considered in determining the level of ordinary skill in the art at the time of an invention include (1) the type of problems encountered in the art; (2) the prior art solutions to those problems; (3) the educational level of active workers in the field; (4) the rapidity with which innovations are made; and (5) the sophistication of the technology. I further understand that the inventors' specific levels of skill are not relevant to this inquiry.

B. Claim Construction

17. I have been informed by counsel that the Patent Trial and Appeal Board ("PTAB") applies the same claim construction standard used in district courts, where the claims are given their ordinary meaning as understood by one skilled in the art at the time of the invention, informed by the claim language itself, the specification, and the prosecution history. I also understand that "extrinsic evidence"—*i.e.*, evidence other than the patent and prosecution history—can be relevant in determining how a POSITA would understand terms of art used in the claims. I have been informed, however, that extrinsic evidence may not be used to contradict the meaning of the claims as described in the intrinsic evidence—*i.e.*, evidence in the claim language itself, the specification, and the prosecution history.

C. Anticipation

18. I understand that a patent claim is invalid based on anticipation if a single prior art reference discloses all of the features of that claim, and does so in a way that enables one of ordinary skill in the art to make and use the invention.

19. I understand that a prior art reference describes the claimed invention if it either expressly or inherently describes each and every feature set forth in the claim; *i.e.*, in determining whether a single item of prior art anticipates a patent claim, one should take into consideration not only what is expressly disclosed in that item, but also what is inherently present as a natural result of the practice of the system or method disclosed in that item.

D. Obviousness

20. I understand that even if a patent claim is not anticipated, it is still invalid if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious at the time the invention was made to a person of ordinary skill in the pertinent art.

21. I understand that a person of ordinary skill in the art provides a reference point from which the prior art and claimed invention should be viewed. This reference point prevents one from using his or her own insight or hindsight in deciding whether a claim is obvious.

22. I also understand that an obviousness determination includes the consideration of various factors such as (1) the scope and content of the prior art, (2) the differences between the prior art and the asserted claims, (3) the level of ordinary skill in the pertinent art, and (4) the existence of secondary considerations or objective indicia of obviousness or non-obviousness.

23. I understand that an obviousness determination can be based on a single prior art reference or a combination of multiple prior art references or the knowledge of one of ordinary skill in the art. I understand that the prior art itself may provide a suggestion, motivation, or reason to combine or modify the teachings of the prior art, or that such a reason may come from other sources, such as the knowledge of a person having ordinary skill in the art, common sense, and market forces. I

understand that the following rationales may support a finding of obviousness:

- combining prior art elements according to known methods to yield predictable results;
- simple substitution of one known element for another to obtain predictable results;
- use of known technique to improve similar devices (methods, or products) in the same way;
- applying a known technique to a known device (method or product) ready for improvement to yield predictable results;
- "obvious to try" - choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success;
- known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations are predictable to one of ordinary skill in the art;
- some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

24. I understand that an obviousness determination requires a reasonable expectation of success in achieving the claimed invention.

25. I understand that secondary considerations or objective indicia of obviousness or non-obviousness may include (1) a long felt but unmet need in the prior art that was satisfied by the invention of the patent; (2) commercial success or lack of commercial success of processes covered by the patent; (3) unexpected results achieved by the invention; (4) praise of the invention by others skilled in the art; (5) the taking of licenses under the patent by others; (6) deliberate copying of

the invention; (7) teaching away; and (8) the simultaneous invention of the claimed subject matter. I also understand that there must be a relationship between any such secondary considerations and the invention. I further understand that contemporaneous and independent invention by others is a secondary consideration supporting an obviousness determination.

26. I understand that any secondary considerations or objective indicia must bear a nexus to the claimed invention. Where the offered secondary consideration actually results from something other than what is both claimed and novel in the claim, there is no nexus to the merits of the claimed invention. I further understand that the patentee bears the burden of demonstrating that the relevant commercial success is attributable to the claimed invention, as opposed to other economic and commercial factors unrelated to the technical quality of the patented subject matter.

27. I understand that whether a prior art reference "teaches away" from the claimed invention is a fact to be considered in determining obviousness. I understand that it is improper to combine references where the references teach away from their combination. However, I understand that the nature of the teaching is highly relevant and must be weighed in substance – a known or obvious composition does not become patentable simply because it has been described as somewhat inferior to some other product for the same use. Furthermore, I understand that the

prior art's mere disclosure of more than one alternative does not constitute a teaching away from any of these alternatives because such disclosure does not criticize, discredit, or otherwise discourage the solution claimed.

IV. TECHNOLOGY BACKGROUND AND STATE OF THE ART

28. Solid-state drives (SSDs) are semiconductor-based data storage devices that provide large storage capacities, non-volatility, and high read and write performance. SSDs are a mature and widely adopted storage technology, present in nearly all modern smartphones, laptops, and servers.

29. SSDs store data on NAND flash memory, which is composed of numerous floating-gate transistors or Charge Trap Flash (CTF) memory cells. To achieve high storage density, SSDs group many of these transistors into units of variable sizes. Data can only be read, written, and erased at the granularity of these units. Notably, the read unit size—often 4KB—can differ from the write unit size, which is generally larger (*e.g.*, 64KB or more), and the erase unit size, which may have a size of *e.g.*, 256KB.

30. A key characteristic of SSDs is that data cannot be modified in place without first performing an explicit erase operation. This constraint means each block of data can only be written once, unless it is being explicitly erased. To enable data modification, SSDs employ an *out-of-place update* strategy: when data is updated, the new version is written to a different location instead of overwriting the

old one. This results in two versions of the same data element coexisting—the outdated version and the newly written version.

31. To ensure that read operations retrieve the most recent data, SSDs utilize a *flash translation layer* (FTL), which manages a *logical-to-physical (L2P) mapping table*. Applications issue I/O operations using logical block addresses (LBAs), while the SSD stores data at physical block addresses (PBAs). The L2P table ensures that each LBA always maps to the PBA containing the most recently written data.

32. For example, assume the SSD has a read granularity of 4KB and the application writes a 4KB block to LBA=2. The SSD stores it at PBA=12 and updates the L2P so that LBA=2 maps to PBA=12. Upon a subsequent overwrite of LBA=2, the SSD writes the new block to PBA=13, leaving PBA=12 unchanged. The L2P is updated to map LBA=2 to PBA=13. If the write unit size is larger (*e.g.*, 64KB), a write buffer is used to coalesce multiple 4KB writes (*e.g.*, 16 blocks) into a single 64KB write operation.

33. Since each write consumes a previously unwritten block, the SSD will eventually run out of such blocks. To reclaim space, SSDs support an *erase* operation, which may be directed to an erase block or erase unit. SSDs may also operate at another level of granularity—commonly 256KB—where erases, for example, are conducted for 64 consecutive 4KB blocks. This unit is also referred to

as a *reclaim unit*, *subdivision*, or *zone*, all of which are functionally equivalent.

34. To safely erase a unit, it must not contain any *valid* blocks. A valid block is defined as a block that is still referenced by the L2P and thus represents the latest version of a data element. If no such erase unit is entirely free of valid blocks, the SSD performs *garbage collection*. During this process, the SSD itself rewrites valid blocks to new locations, updates the L2P accordingly, and invalidates the old PBAs. Once all valid blocks have been relocated, the erase unit can be safely erased. Garbage collection involves additional reads and writes, and therefore, should be minimized to improve performance and endurance, as NAND flash has a limited number of program/erase (P/E) cycles.

35. To optimize garbage collection, the SSD maintains metadata to track the number of valid blocks per erase unit. Selecting an erase unit with the fewest valid blocks minimizes the number of costly copy operations. Furthermore, it is beneficial to segregate frequently updated (*hot*) data from rarely updated (*cold*) data by placing them into separate erase units. This increases the likelihood of finding hot erase units with few valid blocks. One implementation approach is to maintain multiple open units to which writes can be applied: one for cold LBAs and one for hot LBAs. More than two units may be maintained, with write frequency tracked using finer-grained *temperature* metrics.

36. Due to the limited endurance of NAND flash cells (*e.g.*, a floating-gate

transistor may only withstand ~10,000 P/E cycles), *wear-leveling* is essential. Wear-leveling moves cold data from infrequently used units, enabling those blocks to be erased and reused, thereby balancing wear across all units in the SSD.

37. In summary, the FTL performs several key tasks:

- L2P Management: Mapping LBAs to the most recent PBAs for servicing reads.
- Data Placement: Determining whether hot or cold data should be written to a given unit.
- Garbage Collection: Reclaiming space by relocating valid data and erasing blocks.
- Wear-Leveling: Ensuring uniform wear across erase units by migrating cold data.
- Erase Management: Erasing only those units that no longer contain valid data.

38. In traditional SSDs, the FTL is implemented entirely within the SSD controller. This architecture presents a simple hardware interface to software, exposing only read and write operations at fixed granularities (*e.g.*, 4KB).

39. An alternative approach is to implement the FTL—fully or partially—on the *host processor*. Host-based FTLs can leverage application-specific knowledge (*e.g.*, data temperature) to make more informed decisions about data placement, garbage collection, and wear-leveling. For instance, garbage collection can be deferred to periods when the application is idle, thereby improving performance and efficiency.

V. OVERVIEW OF THE ’995 PATENT

A. Priority

40. The ’995 Patent issued on April 19, 2022, and claims priority to Provisional Application No. 61/048,162, filed on September 9, 2014.

41. The prior art that I discuss in this Declaration predates September 9, 2014, and I express no view on whether or not the ’995 Patent is entitled to a priority date before its filing.

B. Prosecution History

42. The nonprovisional application that became the ’995 Patent was filed on October 20, 2021.

43. I understand that the examiner issued a non-final rejection on December 3, 2021. EX1002. The examiner rejected the claims under §112. EX1002. Additionally, claim 8 was rejected under §101. EX1002. On February 23, 2022, the Applicant amended the claims. EX1002. Thereafter, the examiner issued a final rejection on March 2, 2022, allowing Claims 1-7 and 9-30, but maintaining the §101 rejection against Claim 8. After a subsequent Examiner Interview, a Notice of Allowance issued on March 14, 2022. The Examiner’s statement of reasons for allowance reproduces the entirety of Claim 1.

C. Specification

44. I have also reviewed the ’995 Patent, which describes a logical-to-physical address translation scheme for a memory controller taking commands from

a host to manage flash memory (*i.e.*, SSD). EX1001, 2:66-3:11.

45. In one embodiment, the host system sends a command, such as a write command, with accompanying address information to the controller, and the controller translates the address information to sub-addresses associated with virtual structures of the flash memory. EX1001, 2:65-3:21. And each virtual sub-address is translated to an actual physical sub-address of the underlying flash memory structure. *Id*. For instance, in one embodiment, a logical address from the host received by the controller is made up of several bit fields, including bit fields associated with a virtual channel number, virtual die number, virtual erase unit number, and virtual page number. EX1001, 4:23-35. The logical address is broken up into those bit fields. *Id*. Each virtual sub-address is translated by look-up tables to a physical sub-address corresponding to the same flash memory structure (*i.e.*, channel, die, erase unit, and page) and identifying the location in physical flash memory. EX1001, 15:37-65, 16:49-65. The '995 Patent refers to this scheme as "hierarchical address virtualization." EX1001, Abstract.

D. The Challenged Claims

46. Appendix A to my Declaration includes a table reciting the Challenged Claims, including claims 1-11, 13-15, 18-21, 26-30.

47. In my opinion, the challenged claims are invalid based on the following combinations of prior art:

Ground	Basis	Reference(s)	Claims
I	§ 103	Reiter, ZBC, Sinclair-367	1-5, 7-11, 13-15, 18-20, 26-30
II	§ 103	Reiter, ZBC, Sinclair-367, Yamada	6, 21

VI. LEVEL OF ORDINARY SKILL IN THE ART

48. In my opinion, a POSITA at the time of the '995 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. More education can supplement practical experience and vice versa.

VII. CLAIM CONSTRUCTION

49. I understand that in the co-pending district court action, the parties have not yet submitted claim construction positions.

50. I have been informed that, for the purposes of my analysis in this IPR proceeding, the terms appearing in the claims of the '995 Patent should be interpreted according to how a POSITA would have understood the claim terms at the time of the earliest priority date, in light of the patent specification, the prosecution history, and any other relevant intrinsic and extrinsic evidence. I have used the earliest possible priority date of the '995 Patent (September 9, 2014) as the point in time for claim interpretation purposes. However, the interpretations that I provide below would have also been correct if the date of alleged invention was earlier or later.

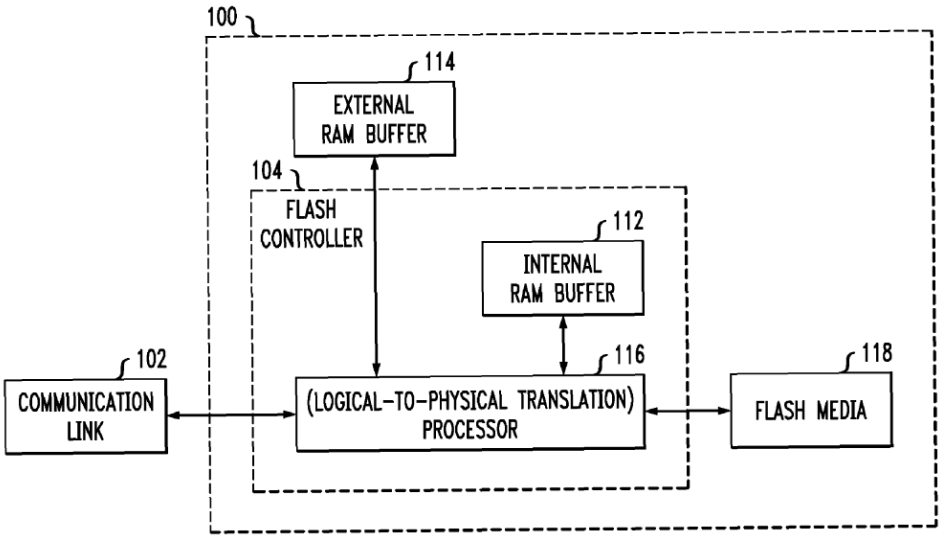
VIII. OVERVIEW OF PRIOR ART REFERENCES

A. Reiter

51. U.S. Patent 8,301,861 ("Reiter") was filed on April 29, 2010, was published on March 24, 2011, and issued on October 30, 2012. I understand that it is prior art because its filing date pre-dates the earliest possible priority date of the '995 Patent.

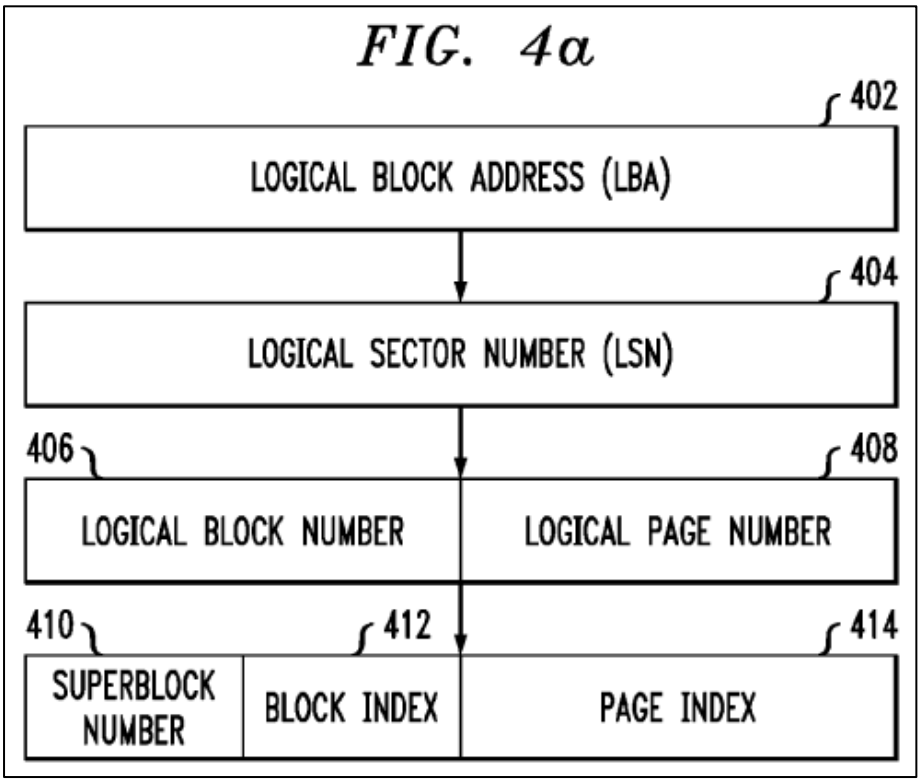
52. Reiter discloses a flash system 100, comprising flash controller 104 for receiving and executing a external computer systems' (*i.e.*, hosts) read and write commands sent over communication link 102. EX1004, 6:63-66, 8:1-6, FIG. 1:

FIG. 1



53. The read and write commands are accompanied by a logical address, which corresponds to a physical address that details the location in flash media 118. And to execute the command at that location, flash controller 104 performs a logical-to-physical translation method to translate the logical address to a physical address.

EX1004, 7:31-35, 8:1-10. Reiter discloses a four-level logical-to-physical translation method including (1) the incoming logical block address 402, (2) logical sector number 404, (3) a bitstring comprising logical block number 406 and logical page number 408, and (4) a bitstring comprising superblock number 410, block index 412, and page index 414. EX1004, 8:54-56, 8:66-9:2, 9:5-20, FIG. 4a:



B. Sinclair-367

54. U.S. Patent Publication No. 2005/0144367 ("Sinclair-367") was filed on December 16, 2004 and published on June 30, 2005. I understand that it is prior art because its filing date pre-dates the earliest possible priority date of the '995 Patent.

55. Sinclair-367 discloses a non-volatile memory array 11 and 13 (*i.e.*, flash

memory) managed by memory controller 19 with various hardware and software features to improve memory management, including concurrent execution of operations and data retention. EX1009, Abstract, FIG. 1A:

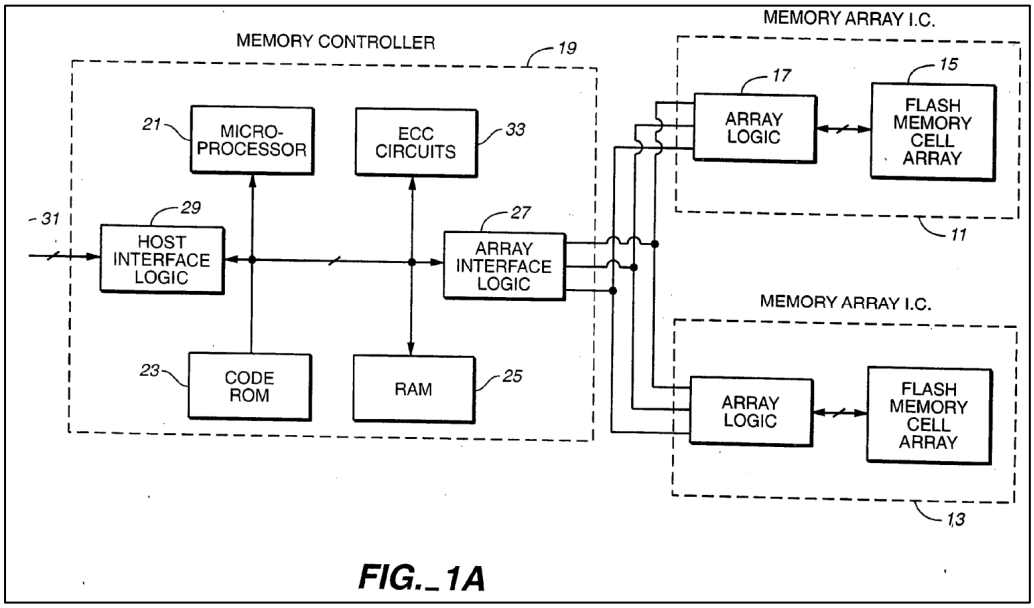


FIG._1A

56. For concurrent execution of operations, Sinclair-367 discloses that "[i]n order to increase the degree of parallelism during programming user data into the memory array and read user data from it, the array is typically divided into … planes, which … allow parallel operation … such that sectors of data may be programmed to or read from each of … the planes simultaneously." EX1009, ¶¶[0010]-[0011]. And "[t]o further efficiently manage the memory, erase blocks may be linked together to form virtual blocks or metablocks," which allow simultaneous reading, writing, and erasing of the blocks in the metablock. EX1009, ¶¶ [0012], [0114], FIG. 6:

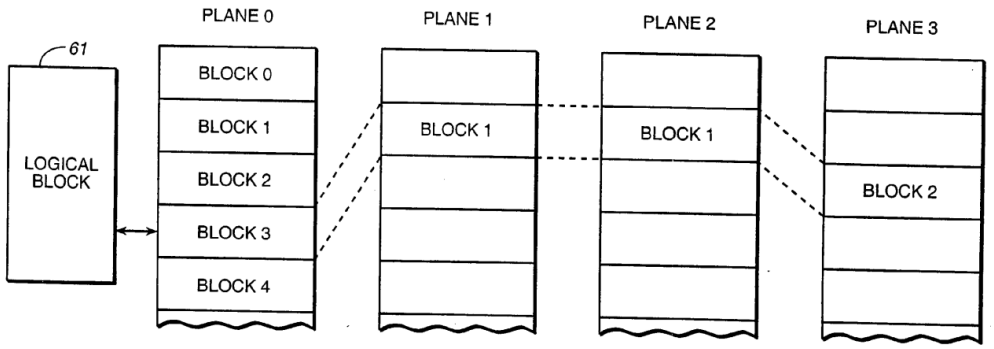


FIG._6

57. For bad block management, Sinclair-367 discloses "identification of any defective cells within the erase block" (detect a failure condition of the physical erase units) and remapping the "logical address" (logical erase unit which corresponds to one of the physical erase units for which the failure condition is detected) with "the address of any substitute erase block in case the primary erase block is defective" (instead correspond to one of the physical erase units for which the failure condition has not been detected) for later use (utilize the remapped correspondence to service one or more of the incoming data access requests). EX1009, ¶¶[0107]-[0108].

58. For data retention, Sinclair-367 discloses maintaining "overhead data" (*i.e.,* metadata) in each erase block or each page. EX1009, ¶¶[0107], [0110], FIG. 3 (below). The overhead data includes various metadata that can improve data retention, such as error correcting codes 53, "number of program/erase cycles experienced by the erase block." EX1009, ¶¶[0107]-[0111]. Error correcting codes can detect and repair damaged data. EX1009, ¶¶[0007], [0107]. Program/erase

cycles "are used in … wear leveling algorithm[s] … in order to even out the usage (wear) of all the erase blocks." EX1009, ¶[0108].

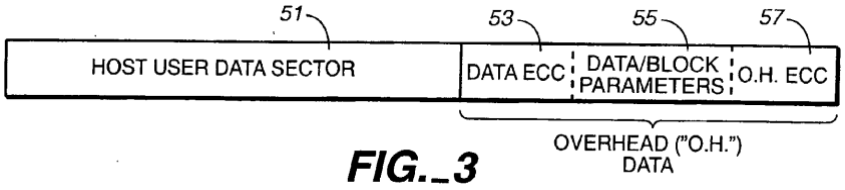


FIG._3

59. In case of wear causing memory cells to become defective, Sinclair-367 discloses "parameters 55 includ[ing] identification of any defective cells within the erase block …." EX1009, ¶[0109].

C. Yamada

60. U.S. Publication No. 2011/0271032 ("Yamada") was filed on July 27, 2010, and was published on November 3, 2011. I understand that it is prior art because its filing date pre-dates the earliest possible priority date of the '995 Patent.

61. Yamada discloses a maintenance scheme for managing nonvolatile memory module 100 (*i.e.*, flash memory). Specifically, Yamada describes memory controller 110 including a "maintenance request unit," and access module 130 (*i.e.*, host) including a maintenance determination unit 152 with command generation unit 154. EX1010, ¶¶[0012], [0017], [0031].

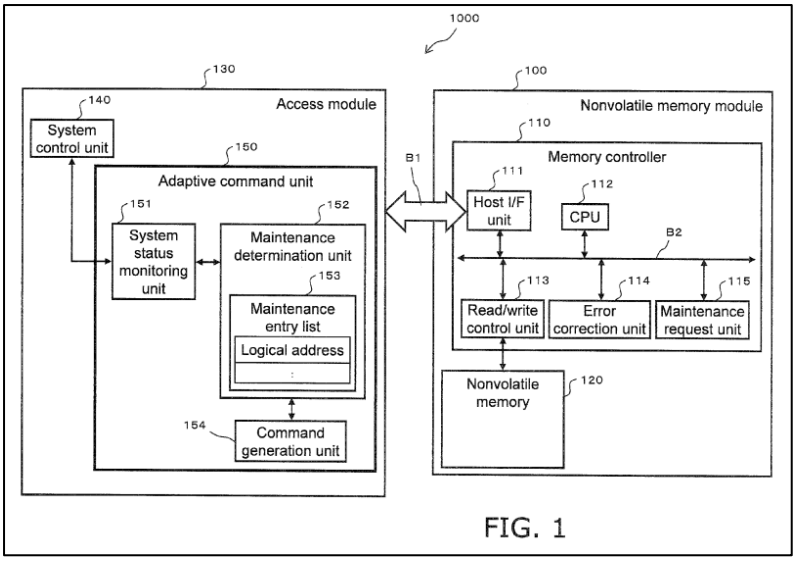


FIG. 1

62. Yamada describes that the host analyzes the metadata of blocks or groups of blocks to determine if certain thresholds have been met. EX1010, ¶¶[0013], [0032], [0080]-[0081]. If so, the controller transmits the metadata and address information to the host in "maintenance request" from the "maintenance request unit." EX1010, ¶¶[0014], [0109]-[0111]. In response, the host may then issue a "maintenance command" from the "maintenance command unit," commanding the controller to copy data from the block or blocks that need maintenance to new block(s). EX1010, ¶¶[0014], [0109]-[0111].

63. Moreover, Yamada discloses performing maintenance operations during the lulls between sets of normal storage operations. EX1010, Abstract. The maintenance determination unit, by being on the access module, monitors the access module's status. EX1010, ¶¶[0018]-[0019], [0022]. When the maintenance determination unit finds downtime in normal processing, the maintenance

determination unit performs requested maintenance operations. EX1010, ¶¶[0018]-[0019], [0022].

D. ZBC, SBC-4, and SPC-4

64. ZBC, SBC-4, and SPC-4 published on May 15, 2014, May 20, 2014, and May 17, 2014, respectively. EX1100, ¶¶8-10; EX1103; *see also* EX1109 (depicting T10's website). I understand that all three were published by T10 InterNational Committee for Information Technology Standards ("INCITS"), and publicly available through the T10 website. EX1101; EX1103; *see also* EX1109. Thus, I understand that all three were publicly accessible before the earliest possible priority date of the '995 Patent and are prior art. EX1100.

65. I am aware of INCITS, and T10 and T10's website in particular. T10 publishes SCSI standards and makes publications readily available on T10's website. *See, e.g.*, EX1101; EX1103. And SCSI protocols and command sets are popular interfaces for storage devices. *See, e.g.*, EX1200; EX1201; EX1202; EX1203.

66. ZBC is another SCSI command set detailing an interface for a zoned block device model. EX1006, 1, Figure 0. Moreover, ZBC is an extension command set, thereby also accepting commands from core SCSI specifications such as SBC-4 and SPC-4. EX1006, x, 12-13, 22, Figure 0.

67. According to SPC-4, which is "shared across all device types"

(EX1006, Figure 0), SCSI logical block devices (*e.g.*, logical units, volumes, and/or partitions) are created for physical storage devices. EX1008, 14, 61. Each logical block device has an independent address range that "shall begin with block zero and be contiguous up to the last block of that logical unit or within that volume or partition." EX1008, 14. According to ZBC, a "logical unit," volume, or partition can "be a zoned block device," (EX1006, 1) and with zoned block devices, the underlying address range is "divided into ranges called zones … The entire capacity of a zoned block device is organized into a set of contiguous, non-overlapping zones." EX1006, 13, Figure 1:

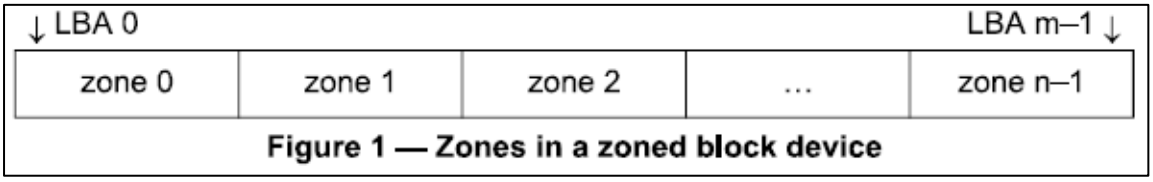


Figure 1 — Zones in a zoned block device

68. ZBC includes at least two different kinds of zoned block devices—(1) "host aware zoned block device[s]" and (2) "host managed zoned block device"—and at least two different kinds of zones—(1) "sequential write preferred zone" and (2) "sequential write required zone." EX1006, 4, 13-14, Table 14. Sequential write preferred zones and sequential write required zones include metadata such as the state of the zone and write pointers. The state of a zone is the zone condition, and includes "EMPTY," "OPEN," "READ ONLY," "FULL," and "OFFLINE." EX1006, Table 15. Write pointers generally track the next available logical block address zones. EX1006, 14, 16-17, Figure 3:

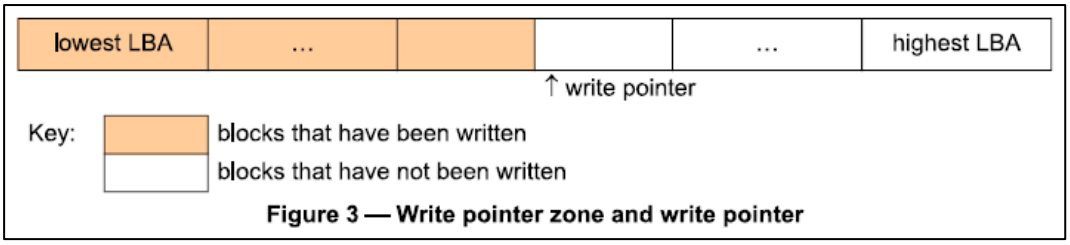


Figure 3 — Write pointer zone and write pointer

69. With each write command, the write pointer increments its logical block address until the end of the zone is reached or a reset write pointer command is applied. EX1006, Table 15. A reset write pointer command resets the write pointer back to the lowest logical block address in the zone and also resets the data in the Motivation zone to the manufacturer's default initialization pattern. EX1006, 3, 15-18.

IX. INVALIDITY GROUNDS

A. GROUND I: Claims 1-5, 7-11, 13-15, 18-20, 26-30 ARE RENDERED OBVIOUS BY Reiter In Light Of ZBC, SBC-4, and SPC-4

1. To Combine

70. A POSITA would be motivated to combine the teachings of ZBC, SBC-4, and SPC-4 (collectively, "ZBC") with flash system 100 disclosed in Reiter. Specifically, a POSITA would be motivated to use ZBC, SBC-4, and SPC-4 as the standard governing requests sent to Reiter's flash system 100. A POSITA would have further been motivated to combine Reiter's flash system executing ZBC commands with Sinclair-367's teachings of incorporating metadata in each physical block and concurrent read/write operations via planes and metablocks.

71. **First**, a POSITA would be motivated to apply the teachings of ZBC, SBC-4, and SPC-4. All three are SCSI specifications designed to be used together. This combination makes sense. ZBC, SBC-4, and SPC-4 are SCSI specifications intended to be used with one another. *See* EX1006, x, 1, 12-13, 22, Figure 0:

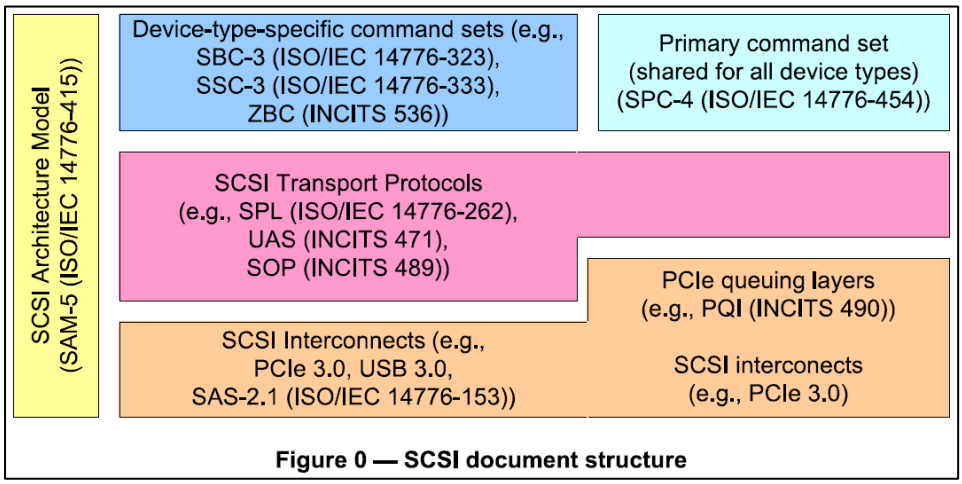


Figure 0 — SCSI document structure

The "purpose of [ZBC] is to define the model and command set extensions to be used in conjunction with … SPC-4 …." EX1006, x, 1. ZBC expressly discloses that it is "implemented in conjunction with … SPC-4," and SPC-4 is listed as "shared for all device types." EX1006, 1, Figure 0. ZBC expressly relies not only on SPC-4, but also SBC-4, throughout the specification. EX1006, x, 1, 12-13, 22, Figure 0.

72. **Second**, with a combined ZBC, SBC-4, and SPC-4, a POSITA would be motivated to apply that standard to govern Reiter's flash system 100. Reiter does not disclose using, or otherwise requiring, a particular command set, but generally discloses support for the widely used SCSI protocols. EX1004, 7:9-18. Reiter states that "[c]ommunication link 102 might be a custom-designed communication link, or

might conform to a standard communication protocol such as, for example … SCSI … protocol bus, … Serial Attached SCSI … protocol bus, … or any other similar interface link for connecting a peripheral device to a computer." EX1004, 7:9-18. Reiter also discloses "receiv[ing] … requests … such as read or write operations from … external devices," where those "requests … generally include a[] … logical block address where data should be read or written." EX1004, 8:1-6. Thus, given the ZBC being an SCSI extension and being able to fill Reiter's lack of a detailed interface, a POSITA would be motivated to support commands such as those disclosed in ZBC to enable specific read and write operations based on Reiter's memory structure.

73. Moreover, a POSITA would particularly be motivated to implement Reiter to run on ZBC commands because ZBC is a good fit for Reiter's flash system. SCSI protocols and commands such as those disclosed in ZBC are widely used, and would thus improve the compatibility and interoperability of Reiter's flash system. And ZBC's zoned block device model is well suited for Reiter's flash system because it would enable effective utilization of Reiter's superblock memory organization. *See* EX1004, FIG. 4c. ZBC manages an interface for a "zoned block device." EX1006, 1. Zoned block devices have an address space comprising logical block addresses ("LBAs") "divided into ranges called zones." EX1006, 13, 29, Table 16, Figure 1:

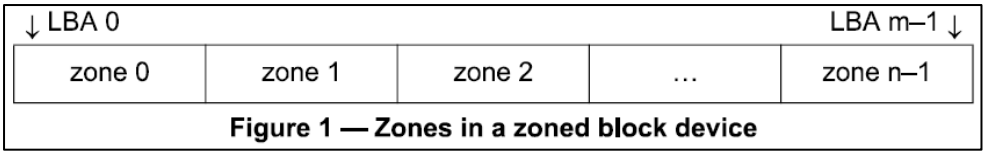


Figure 1 — Zones in a zoned block device

74. Zoned block devices' zones, as governed by ZBC, are analogous to Reiter's superblocks. Like zones, superblocks comprise "a logical collection of blocks representing a fixed range of LBAs." EX1004, 9:11-13. ZBC's zones and Reiter's superblocks include metadata indicating the next available memory location within the zone and superblock, respectively. EX1006, 14, 16-17, Figure 3 (illustrating a zone with a write pointer indicating next available LBA); EX1004, 9:52-10:1, FIG. 4b (illustrating superblock summary page with "free page pointer" indicating next available memory location). And like ZBC, which use LBAs to access different zones, Reiter's superblocks are addressed through the received LBA. EX1004, 8:54-9:25, FIG. 4a (deriving superblock number 410 from logical block address 402); EX1006, 13.

75. A POSITA would have a reasonable expectation of success in adding support for ZBC in Reiter's flash system 100. SCSI commands and protocols such as ZBC were widely known and would have been familiar to a POSITA. SCSI specifications were intended to be used across a wide range of storage devices, including flash memory. A POSITA would have also understood that Reiter's superblock-based memory organization is suitable for zone-based commands like ZBC. Further, a POSITA would have understood that ZBC's zoned block device

model is a logical construct that can be layered on top of any storage device.

76. **Third**, a POSITA would have been motivated to combine the Reiter's flash system executing ZBC commands with Sinclair-367's teachings of incorporating metadata in each physical block and concurrent read/write operations via planes and metablocks.

77. Sinclair-367 discloses storing in each erase unit a sector of metadata (*e.g.*, error correcting codes ("ECC") and number of program/erase cycles). EX1009, ¶¶[0106]-[0107], FIG. 3. ECCs recognize and repair damaged data. EX1009, ¶¶[0007], [0102]. Program/erase cycles determine the wear and tear on a physical block. EX1009, ¶[0107]. Cycle count "is used in a wear-leveling algorithm … even[ing] out the usage (wear) of all the erase blocks," increasing the resiliency and lifespan of the flash system. EX1009, ¶[0108]. Further, because a single sector is a small overhead for a block, yet reduces data loss, a POSITA would have a reasonable expectation of success storing a metadata sector in physical blocks.

78. Additionally, Sinclair-367 discloses dividing flash memory into planes, and logically linking erase units in each plane to form metablocks. A POSITA would be motivated to incorporate planes and metablock operations into Reiter's flash system. These features greatly improve speed and throughput. Each plane in flash memory operates independently from one another. Each plane executes read/write operations separately and concurrently with one another. Each read/write operation

does not need to wait for the next operation to finish as required for sequential processing. Sinclair-367 furthers this with metablocks, stating that "to increase the degree of parallelism of operation, erase blocks within different planes are logically linked to form metablocks." EX1009, ¶[0114]. "All blocks of the metablock are erased together, and pages of each block are generally programmed and read simultaneously." EX1009, ¶[0114].

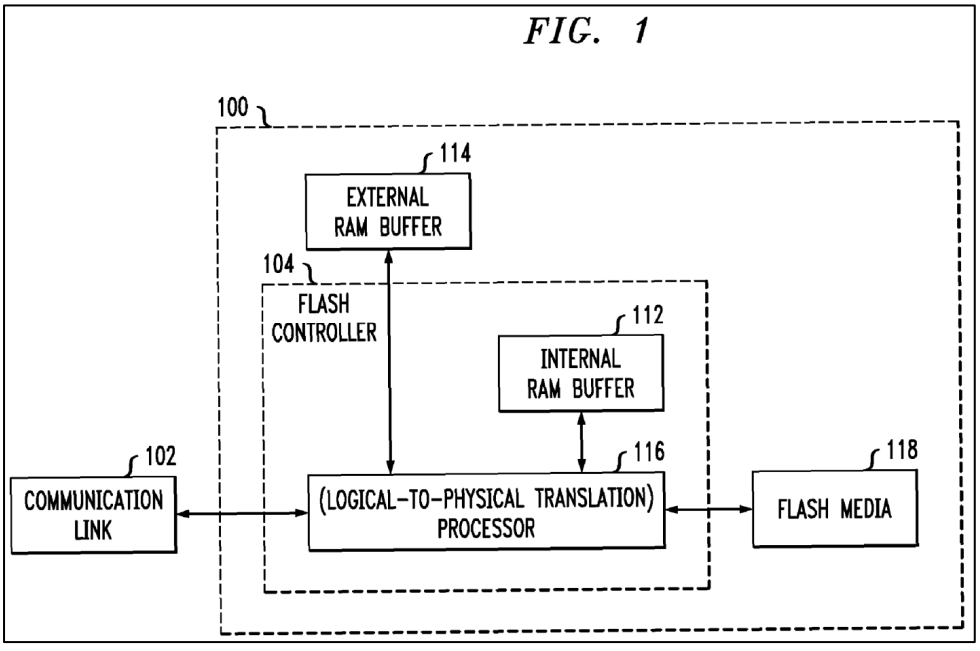
79. Moreover, Reiter, ZBC, SBC-4, SPC-4, and Sinclair-367 disclose analogous art. All relate to storage devices, including accessing blocks of memory.

2. Claim 1

(a) 1[pre]: *A storage device comprising:*

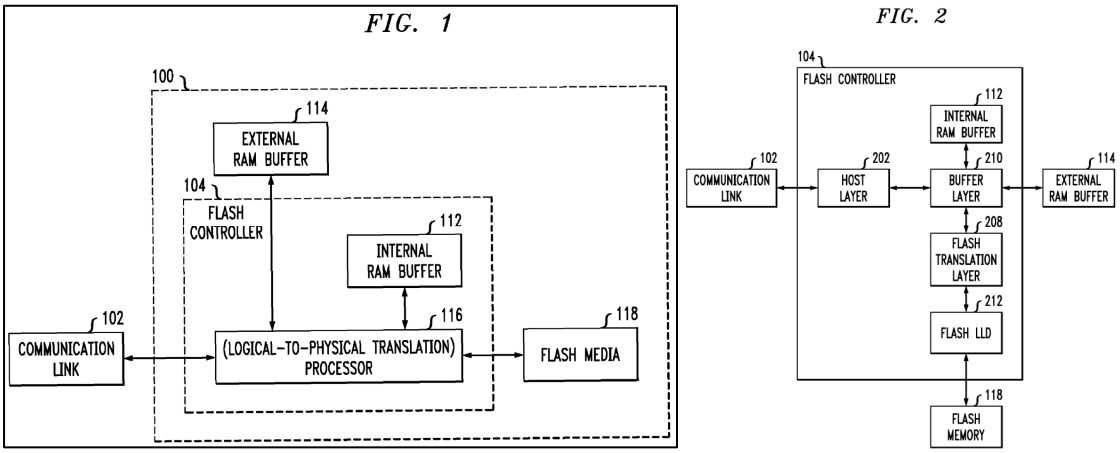
80. To the extent the preamble is limiting, it is my opinion that Reiter discloses it. For instance, Reiter discloses a "flash … system 100" (***storage device***) "implementing a logical-to-physical translation" scheme. EX1004, 6:63-66, FIG. 1:

FIG. 1



(b) 1[a]: *flash memory having physical erase units; and*

81. Reiter's flash memory storage system 100 includes "flash media 118" or "flash memory 118" (flash memory). EX1004, 6:63-7:5, 7:56-61, FIGS. 1, 2:

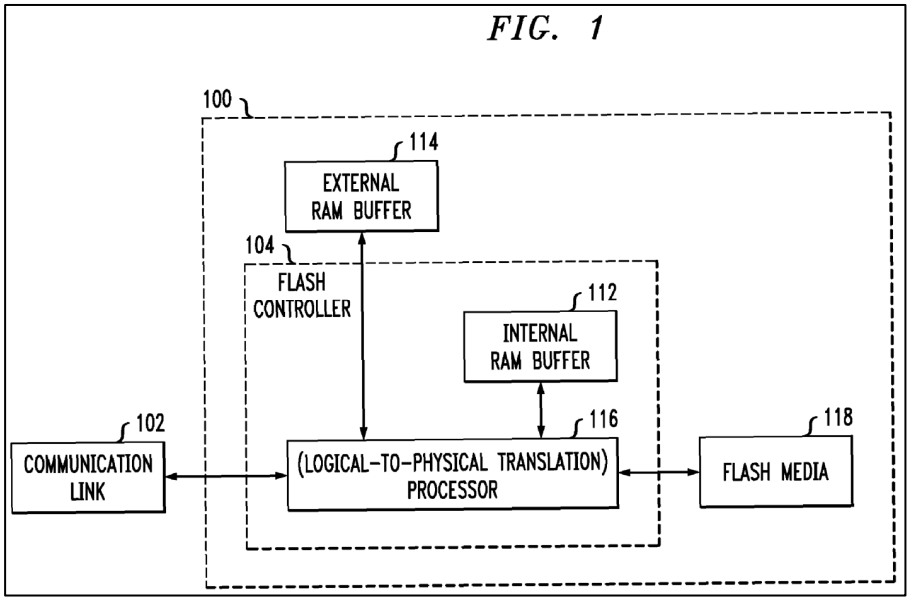


82. Reiter discloses that a "commonly employed type of flash memory technology is NAND flash memory." EX1004, 1:40-45. "There are three basic operations for NAND devices: read, write, and erase." EX1004, 1:51-52. Read and write operations are performed on "[p]ages," which "are typically arranged in blocks." EX1004, 1:52-57. "[A]n erase operation is performed on a block by block basis." EX1004, 1:56-57. Reiter also expressly discloses that its flash media comprises "blocks" that may be erased. EX1004, 6:60-62, 17:61, cls. 2, 9. Reiter thus discloses that its flash memory is made up of independently erasable blocks (flash memory having physical erase units).

(c) 1[b]: *circuitry to, for each of incoming data write requests:*

83. Reiter's flash memory storage system 100 includes a "flash controller

104" (circuitry). EX1004, 6:63-7:2, FIG. 1 (below).



84. Reiter's "[F]lash controller 104 ***receives*** one or more requests for flash media access, such as read or write operations [(***incoming data write requests***)] from one or more external devices via communication link 102. EX1004, 8:1-4. The "requests for access to flash media 118 generally include at least one logical block address (LBA) where data should be read or written." EX1004, 8:4-6.

85. For example, Reiter discloses a flash media "write operation" that is executed by "flash memory storage system 100." EX1004, 13:4-37, Fig. 6A. A data transfer request is divided into chunks based on a "starting LBA," and "FTL 208 writes one or more pages of the chunk data to flash media at step 118." *Id*.

86. A POSITA would be motivated to implement Reiter's flash controller 104 to accept and execute ZBC commands. Section IX.A.1. The ZBC command set includes a write command—a request disclosed in Reiter (EX1004, 8:1-6)—

(incoming data write requests). EX1007, Table 112; *see also* EX1006, 22 ("The commands for host aware zoned block devices are defined in SBC-4."). A ZBC write command is shown below and includes a logical block address:

Table 112 — WRITE (32) command

Byte \ Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	Reserved							
•••								
5								
6	Reserved			GROUP NUMBER				
7	ADDITIONAL CDB LENGTH (18h)							
8	(MSB)							
9	SERVICE ACTION (000Bh)							(LSB)
10	WRPROTECT			DPO	FUA	Reserved	Obsolete	Reserved
11	Reserved							
12	(MSB)							
•••	LOGICAL BLOCK ADDRESS							
19								(LSB)
20	(MSB)							
•••	EXPECTED INITIAL LOGICAL BLOCK REFERENCE TAG							
23								(LSB)
24	(MSB)							
25	EXPECTED LOGICAL BLOCK APPLICATION TAG							(LSB)
26	(MSB)							
27	LOGICAL BLOCK APPLICATION TAG MASK							(LSB)
28	(MSB)							
•••	TRANSFER LENGTH							
31								(LSB)

EX1007, Table 112.

(d) 1[b][i]: ***derive, from incoming address information, a first address portion, a second address portion and a third address portion,***

87. A ZBC write command includes a logical block address (incoming

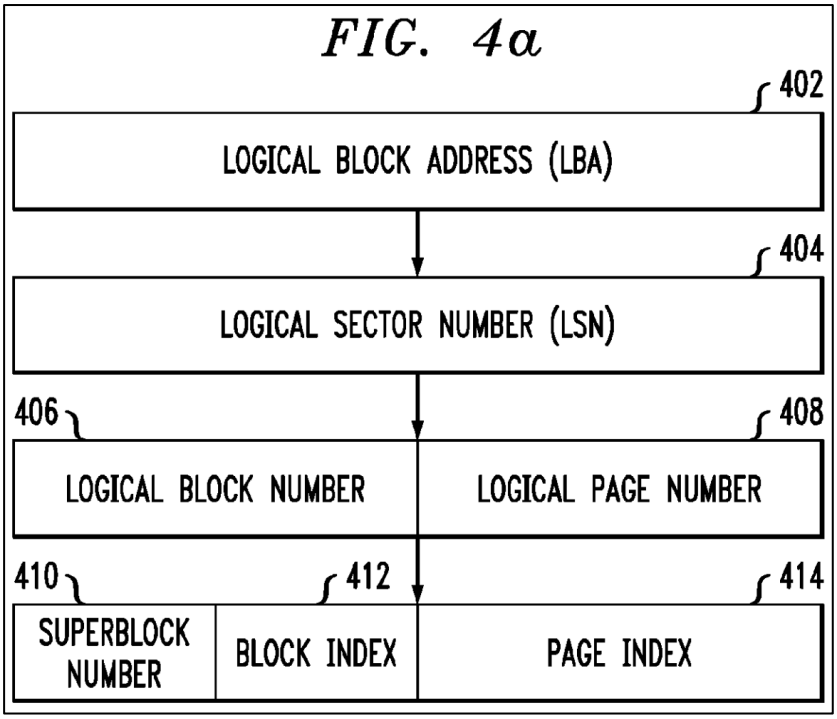
address information)—also as disclosed with Reiter's requests (EX1004, 8:1-6, 11:55-63, 13:3-46). From the write command address information, flash controller 104 derives (1) a number for a logical unit number (**a first address portion**), (2) superblock number 410 (a ***second address portion***), and (3) a logical sector number 404, or alternatively a bitstring of logical block number 406 and logical page number 408, or alternatively logical page number 408 or page index 414 (a ***third address portion***).

88. ***First Address Portion***: In my opinion, a POSITA would be motivated to incorporate a number associated with a logical unit in ZBC commands, and motivated to modify flash controller 104 to decode (derive) that number (first address portion) from the command (from the incoming address information). ZBC write commands target a "logical unit," "volume," or "partition." *See* EX1008, 14, 61; *see also* EX1006, x, 1, 4. An "objective of … [ZBC]" is "to permit an application client to communicate with a logical unit that declares itself to be a zoned block device …." EX1006, 1. But multiple logical units, volumes, and/or partitions are logical structures, each of which can exist on the same physical storage device. *See* EX1008, 14 (describing a logical unit as an "externally addressable entity within a SCSI target device that implements a SCSI device model"). And each logical unit, volume, and partition "shall begin with block zero and be contiguous up to the last logical block of that logical unit or within that volume or partition." EX1008, 61.

With multiple logical units, volumes, and/or partitions and overlapping LBA spaces, an LBA by itself is not enough to identify a location on a physical storage device. Thus, a POSITA would look to incorporate something else into its commands to distinguish the same LBA to two different logical units.

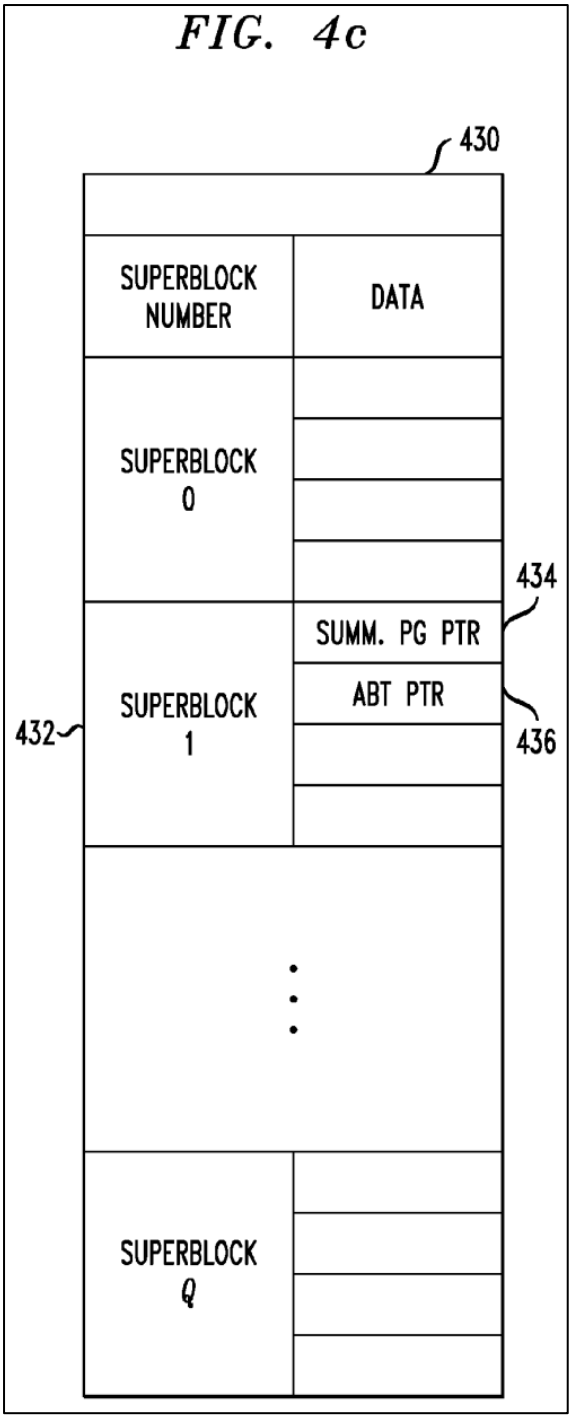
89. A POSITA would be motivated to incorporate a separate address found in SPC-4 into ZBC commands, as I explained above. SPC-4 recites that a "logical unit" is "[a]n externally addressable entity within a SCSI target device …." EX1008, 7, 14. Addressing a logical unit is done with a "logical unit number"—"[a]n encoded 64 bit identifier for a logical unit." EX1008, 14; *see also* EX1006, 4. ZBC already mentions logical unit number. EX1006, 4. A POSITA would be motivated to more fully incorporate that address in its commands via the teachings in SPC-4.

90. ***Second Address Portion***: A POSITA would be motivated to use Reiter's flash controller 104 to derive superblock number 410 (second address portion) from the LBA in the ZBC write commands (from the accompanying address information). ZBC write commands include LBAs. EX1007, Table 112. Reiter discloses that read and write "request[s] … include[] an … LBA …." EX1004, 8:4-6, 8:62-67. "FIG. 4a shows an exemplary relation of a[n] … LBA … to a physical address … as managed by FTL …." EX1004, 8:56-59, FIG. 4a:



Translating an LBA results in a bitstring of superblock number 410, block index 412, and page index 414. EX1004, 8:66-9:25, FIG. 4a. Superblock number 410 is identified in the page global directory (PGD) 430 to obtain a "pointer 434 … to the summary page associated with [the corresponding] Superblock." *See* EX1004, 10:13-19, FIG. 4c:

FIG. 4c

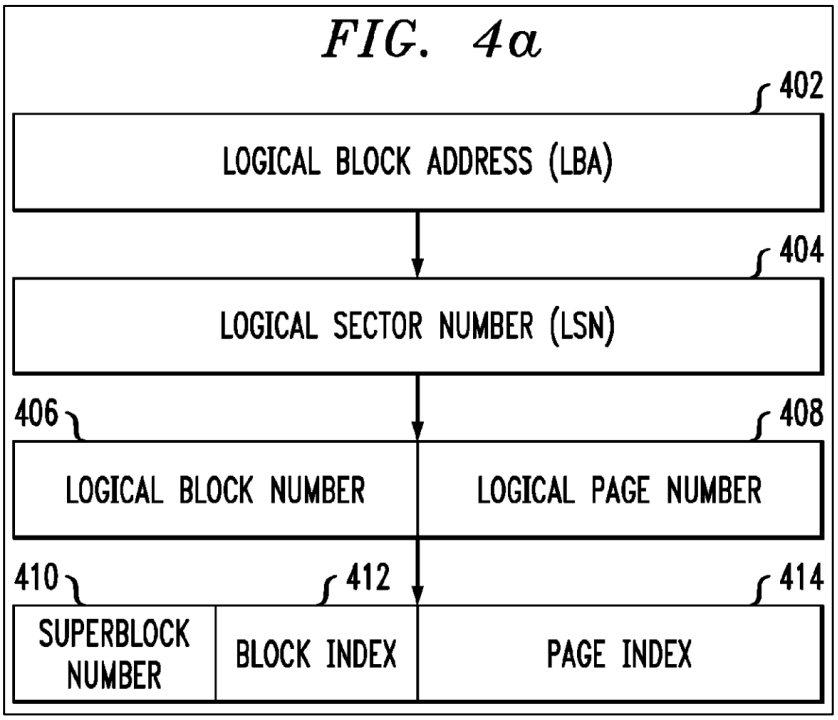


91. A POSITA would understand the (1) LBA 402, (2) LSN 404, (3) logical block number 406 and logical page number 408, and (4) superblock number 410, block index 412, and page index 414 in FIG. 4a to be bitstrings. Each is depicted as

one unit. EX1004, FIG. 4a (above). This matches the single unit depiction of logical block address 402 and logical sector number 404 bitstrings in the prior translation steps. EX1004, FIG. 4a (above). Other prior art references depicting bits as one block with separate portions of the block associated with different values (EX1004, FIG. 4a (above for superblock number 410, block index 412, and page index 414)) describe that block as a bitstring, and the values as bitfields within the bitstring. *See* EX1005, 5:7-10 ("The logical unit number is the high order binary digits of the logical address and may be derived from the logical address by a bit shift operation."), FIG. 4; EX1210, ¶[0157], FIG. 33. A POSITA would also be motivated to store or keep the bitfields of each translation level together in bitstrings (as depicted in FIG. 4a). This creates a simple system where one translation step is only started after completing the prior completion step (*see* EX1004, FIG. 4a), because the resulting bitfields need to be produced and then combined into one bitstring prior to the next translation step. Ensuring lock-step translation of addresses simplifies management where (1) bitfields are stored all over the place and gathered from different parts of memory and (2) some bitfields are further along translation than other bitfields associated with the same address. This simplicity becomes increasingly important with concurrent and/or out-of-order execution of read/write/erase commands.

92. ***Third Address Portion***: A POSITA would be motivated to use Reiter's

flash controller 104 to ***derive*** logical sector number 404, or alternatively bitstring of logical block number 406 and logical page number 408, or alternatively logical page number 408 or page index 414 (***third address portion***) in the ZBC write commands (***from the incoming address information***). A ZBC write command includes an LBA. EX1007, Table 112. Reiter also discloses that read and write "request[s] … include[] a[n] … LBA … ." EX1004, 8:4-6, 8:62-67. "FIG. 4a shows an exemplary relation of a[n] … LBA … to a physical address … as managed by FTL …." EX1004, 8:56-59, FIG. 4a:



93. "LBA 402 corresponds to at least one logical sector number (LSN) 404." EX1004, 8:66-67. Logical sector number 404 is translated to a bitstring with logical block number 406 and logical page number 408 bitfields. EX1004, FIG. 4a. Logical block number 406 is obtained by "dividing LSN 404 by a number of sectors

per block," and logical page number 408 is obtained by "dividing LSN 404 by a number of sectors per page …." EX1004, 9:5-9. Logical page number 408 is "divid[ed] … by a number of pages per block" to obtain page index 414. EX1004, 9:17-20.

(e) 1[b][ii]: *identify an addressed block device from the first address portion,*

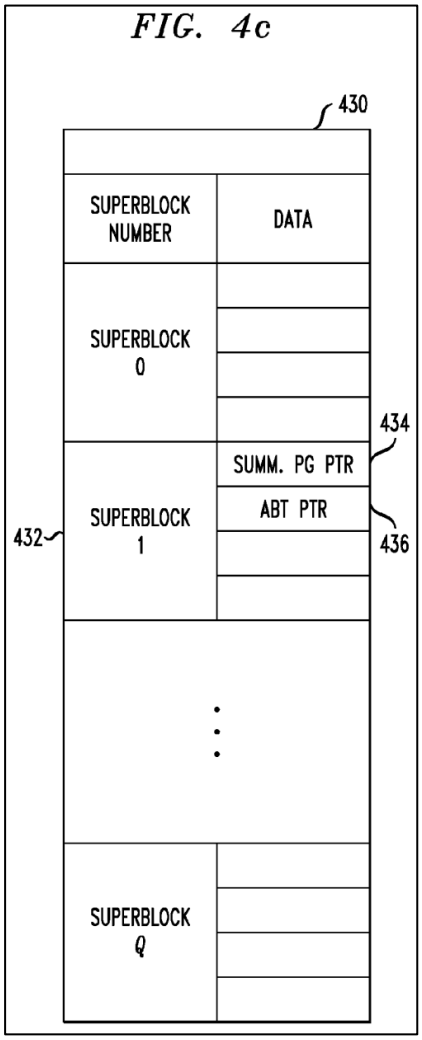
94. In my opinion, a POSITA would be motivated to modify Reiter's flash controller 104 to derive a number associated with a logical unit (**first address portion**) for the reasons I discussed above. *See* Limitation 1[b][i]. This number (**from the first address portion**) identifies the targeted logical unit, volume, or partition (**addressed block device**). Whether a logical unit number, volume, or partition is targeted, each structure includes an independent range of logical block addresses "begin[ning] with block zero and be contiguous up to the last logical block of that logical unit or within that volume or partition." EX1008, 61.

(f) 1[b][iii]: *identify an addressed segment within the addressed block device from the second address portion, and*

95. In my opinion, a POSITA would be motivated to modify Reiter's flash controller 104 to derive superblock number 410 (**second address portion**) from ZBC write commands. Limitation, 1[b][i]. Superblock number 410 (**from the second address portion**) identifies the superblock, which a POSITA would be motivated implement as a ZBC zone (**address segment)**. For instance, superblock

number 410 indexes page global directory (PGD) 430 to obtain the "summary page associated with [the corresponding] Superblock." EX1004, 10:13-19, FIG. 4c. Superblock number 410 is derived from LBA 402 (EX1004, FIG. 4a), which is from a ZBC command's LBA.

FIG. 4c

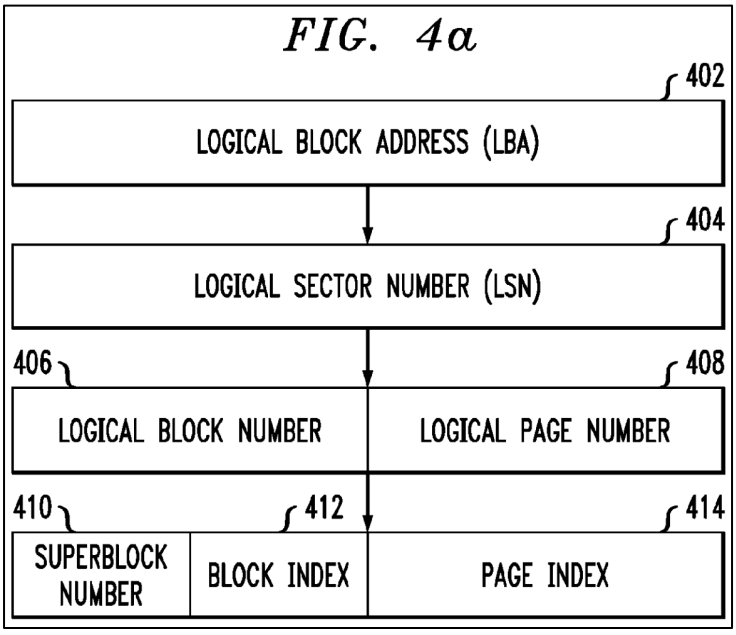


96. Indeed, I understand that Patent Owner has also identified an "addressed zone" as an "**address segment**" for purposes of infringement. And because a ZBC LBA is within the LBA range specified by the targeted logical unit (*see* EX1006, x, 1, 4; EX1008, 61), superblock number 410 identifies a superblock

within the targeted logical unit (***addressed block device***).

(g) 1[b][iv]: ***identify an addressed storage location within the addressed segment from the third address portion,***

97. In my opinion, a POSITA would be motivated to modify Reiter's flash controller 104 to derive logical sector number 404, or alternatively bitstring of logical block number 406 and logical page number 408, (***third address portion***) from ZBC write commands. Limitation, 1[b][i]. Flash controller 104 translates logical sector number 404, and alternatively the bitstring of logical block number 406 and logical page number 408, and alternatively logical page number 408 or page index 414 (***from the third address portion***), to block index 412 and/or page index 414. EX1004, 9:5-25, FIG. 4a:

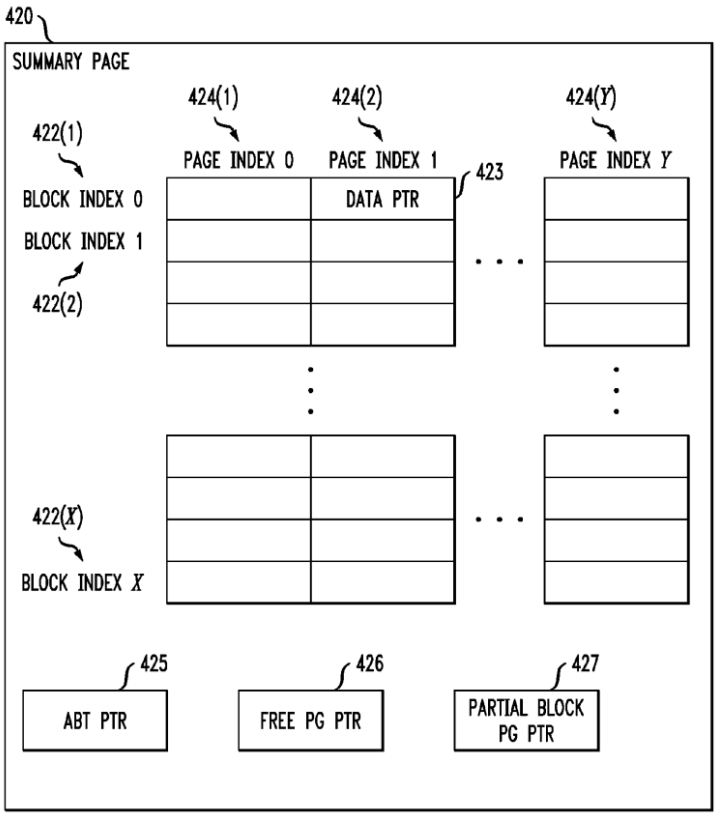


98. In my opinion, these indexes ***identify*** a logical block and/or logical page (***addressed storage location***) ***within*** the superblock (*i.e.*, zone (***addressed segment***)

(Section VIII.C.1.f)) addressed by superblock number 410. EX1004, 9:52-63.

99. Specifically, block index 412 and page index 414 select a block within the superblock and page within that block, respectively, from a superblock's summary page. EX1004, 9:52-63, FIG. 4b. The "summary page … contains mapping data for the … Superblock." EX1004, 9:52-54. "[S]ummary page 420 contains the block index and page index … for all X blocks and Y pages in [the] Superblock." EX1004, 9:54-58, FIG. 4b:

FIG. 4b



100. Because logical sector number 404, and alternatively bitstring of logical block number 404 and logical page number 406, and alternatively logical page number 406 and page index 414, are translated into block index 412 and/or page

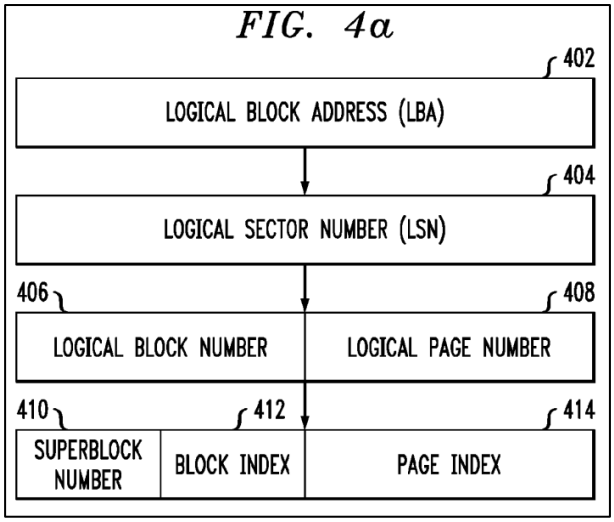
index 414, in my opinion Reiter discloses ***identifying an addressed storage location … from the third address portion***.

101. Further, I understand that Patent Owner agrees with this mapping as its infringement contentions alleges that "identify[ing] an addressed storage location[] … includ[es] a logical block, page, and/or erase unit." EX1011, 25.

(h) 1[c]: *wherein the circuitry is to use a division operation on at least one of the incoming address information or information derived therefrom, to identify a logical erase unit within the addressed segment, and*

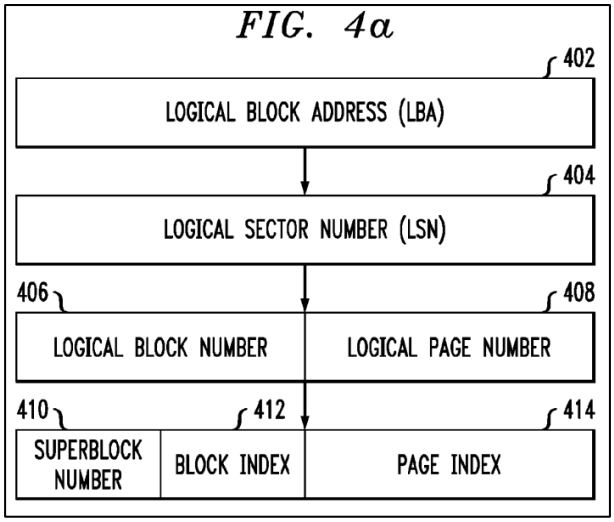
102. In my opinion, Reiter discloses flash controller 104 (***circuitry***) performing integer division on logical sector number 404 (***incoming address information or information derived therefrom***) to ***identify*** logical block number 406 (***logical erase unit***). Alternatively, a POSITA would be motivated to apply integer division to subdivide out (***identify***) logical block number 406 bitfield (***logical erase unit***) from the bitstring of logical block number 406 and logical page number 408 bitfields. Alternatively, a POSITA would be motivated to apply integer division to subdivide out (***identify***) block index 412 bitfield (***logical erase unit***) from the bitstring of superblock number 410, block index 412, and page index 414.

103. **Integer Division On Logical Sector Number 404**: Reiter discloses that "[e]ach LBA can be divided into a logical block number and logical page number …." The first translation step is translation of LBA 402 to logical sector number 404. EX1004, 8:66-67, FIG. 4a:



104. In the second step, "LSN 404 corresponds to … logical block number (LBN) 406 and … logical page number (LPN) 408." EX1004, 9:5-6, FIG. 4a (above). "FTL 208 derives LBN 406 by **dividing** LSN 404 by a number of sectors per block of flash media 118." EX1004, 9:6-8. This integer division constitutes the ***division operation to identify a logical erase unit***.

105. **Alternative, Integer Division To Subdivide Out Logical Block Number 406 Bitfield**: Alternatively, after the second step, a new bit string is formed including logical block number 406 and logical page number 408. EX1004, FIG. 4a:

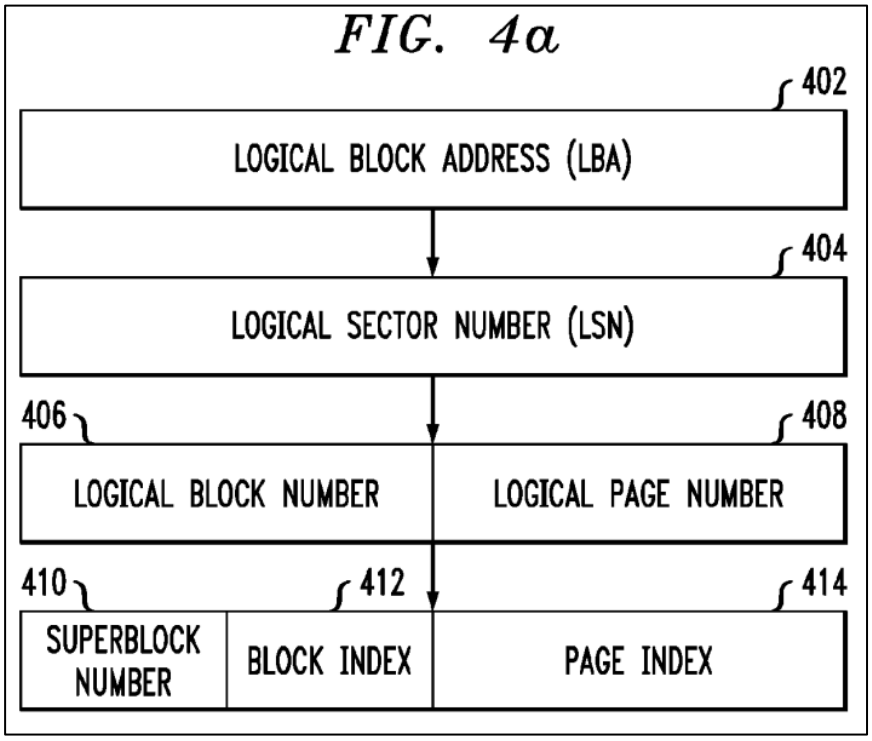


106. A POSITA would be motivated to divide by a certain power-of-two to split the bitstring into a first and second bitfields. Integer division by a certain power-of-two extracts bitfields out of bitstrings. The quotient is the extracted most significant bitfields (*i.e.*, bits on the lefthand portion of a bitstring). Integer division is preferable because it is fast, as division by a power-of-two on a bitstring is implemented with a simple bit shift operation. Logical block number 406 is the higher order or most significant bitfield of the bitstring. EX1004, FIG. 4a. Applying integer division on the bitstring causes superblock number 410 to be extracted through the quotient. Because superblock number 410 is a bitfield, a POSITA would be motivated to implement integer division via a bit shift operation to perform the extraction quickly.

107. In my opinion, a POSITA would also be motivated to extract out logical block number 406. Reiter discloses later translation steps that are only performed on logical block number 406 and logical page number 408. For instance, "FTL 208

derives Superblock number 410 and block index 412 from LBN 406 by dividing LBN 406 by a number of blocks per Superblock ….” EX1004, 9:13-17. Similarly, “Page index 414 is derived from LPN 408 by dividing LPN 408 by a number of pages per block ….” EX1004, 9:17-20. Dividing to subdivide the bit string into bitfields alternatively constitutes the division operation to identify logical block number 406 (**logical erase unit**).

108. **Alternative, Integer Division To Subdivide Out Block Index 412 Bitfield**: Alternatively, after translating logical block number 406 and logical page number 408, a new bit string is formed including superblock number 410, block index 412, and page index 414. EX1004, FIG. 4a:



109. For similar reasons as I discussed above, a POSITA would be motivated

for the sake of speed to divide by a power-of-two to split the bitstring into the corresponding bitfields.

110. Logical Erase Unit: The resulting logical block number 406, or alternatively block index 412, is a ***logical erase unit***. For the logical block number 406, the mapping is explicit in the name—"**logical block** number." Logical block number 406 is a logical unit. Block refers to an erase unit, as "an erase operation is performed on a block by block basis." EX1004, 1:56-57.

111. Additionally, logical block number 406 and block index 412 are an abstraction of a physical block— erase unit—because both map to a physical block. "LBN 406 … corresponds to Superblock number 410 and block index 412." EX1004, 9:5-10. "[A] Superblock generally is a logical collection of blocks …." EX1004, 9:11-14. And the block index 412 indexes the superblock's list, identifying a logical block within the collection and, with page index 414, selecting "data pointer 423, which points to the physical address." *See* EX1004, 9:13-17, 9:52-63, FIG. 4b.

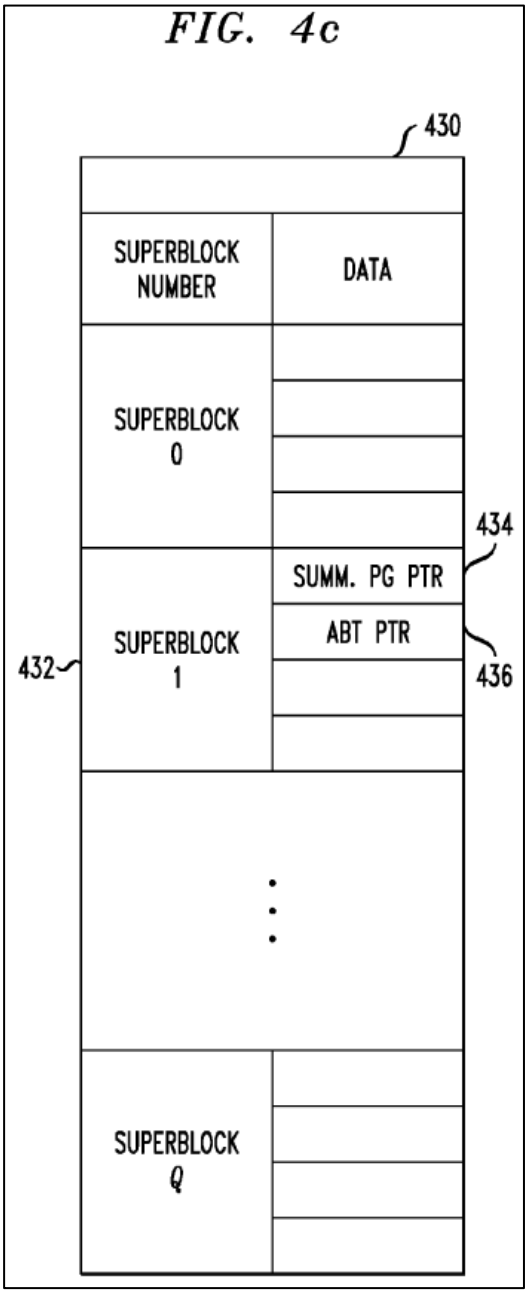
(i) 1[d]: *wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and*

112. In my opinion, logical block number 406 constitutes an ***erase unit***. Limitation 1[c]. Reiter discloses flash controller 104 (*i.e.*, ***circuitry***) translates the logical block number 406, and alternatively block index 412 (*i.e.*, ***identified logical erase unit***) to a physical block number (***identify one of the physical erase units that***

corresponds to the identified logical erase unit). "LBN 406 … corresponds to Superblock number 410 and block index 412." EX1004, 9:5-10, FIG. 4a.

113. Superblock number 410 selects the summary page "contain[ing] … mapping data for the associated Superblock." EX1004, 9:52-54, Fig. 4c:

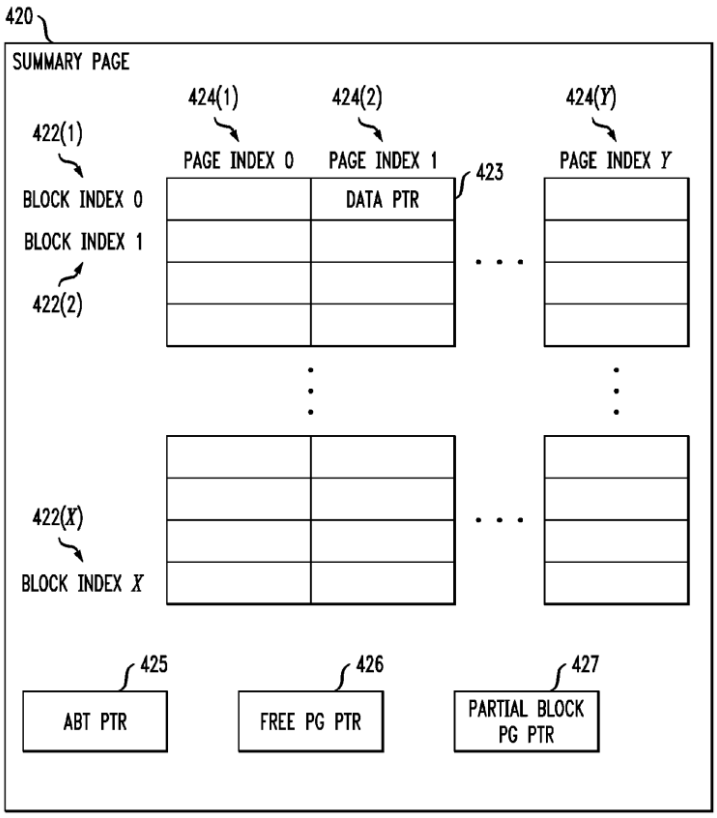
FIG. 4c



114. "[A] Superblock generally is a logical collection of blocks," where

block index 412 selects one of those blocks and, with page index 414, selects "data pointer 423, which points to the physical address." *See* EX1004, 9:11-17, 9:52-63, FIG. 4b:

FIG. 4b



115. Data pointer 423 points to the underlying, selected physical block. EX1004, 9:60-63, FIG. 4b. Because the underlying physical block corresponds to superblock index 412 and block index 414, which were derived from logical block number 406, flash controller's 104 translation ***identifies one of the physical erase units that corresponds to*** logical block number 406, or alternatively block index 414 (***identified logical erase unit***).

(j) 1[e]: *program associated data into the identified one of the physical erase units;*

116. In my opinion, Reiter discloses flash controller 104's "determin[ing] the Superblock number, Block index, and Page index of the first page to be written," (***identified one of the physical erase units***) and the "requested data is written at the active page address at step 644" (***program associated data***). EX1004, 13:38-60, FIG. 6b.

(k) 1[f]: *wherein the storage device also comprises circuitry to:*

117. In my opinion, Reiter discloses this limitation because Reiter's flash storage system 100 (***storage device***) includes a "flash controller 104" (***circuitry***). EX1004, 6:63-7:2, FIG. 1.

(l) 1[f][i]: *maintain metadata for each physical erase unit which contains data, the metadata representing a time since data was programmed in the respective erase unit,*

118. With the writing capabilities of Reiter's flash controller 104 (***circuitry***), Sinclair-367's erase blocks use a "single sector of data" to ***store*** overhead data, including error correction codes (ECC) 53 for data and data/block parameters 55 (***maintain metadata***). EX1009, FIG. 3:

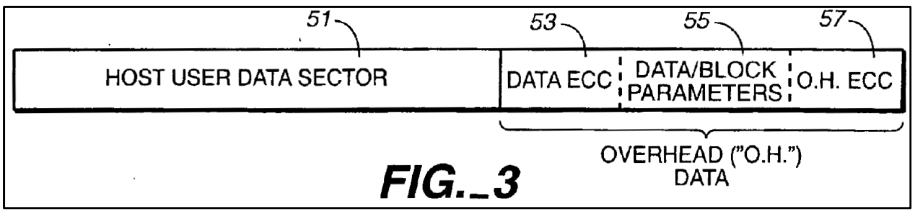


FIG._3

119. Therefore, a POSITA would understand that every block in a logical

unit, volume, and partition includes this single sector of overhead data (***metadata for each physical erase unit which contains data***). Sinclair-367 also includes the "time of programming that is recorded as a field in [the] sector or page overhead data" (***the metadata representing a time since data was programmed in the respective erase unit***). EX1009, ¶[0111].

(m) 1[f][ii]: ***copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and***

120. A POSITA would be motivated to implement flash controller 104 to compare the time of programming data in a physical block (***the maintained metadata***) against a retention period (***at least one criterion***). A retention period is the period of time where flash memory reliably stores a value, and flash memory does not reliably store the value after that period of time. This is a problem well known in the art. *See* EX1010, ¶[0009] ("Data loss can occur in an area that has not been refreshed for a sufficiently long period of time."). It was known to address this problem by measuring a time-since-programming to identify blocks or pages that need to be rewritten. EX1010, ¶[0169] ("maintenance of data that has been left unread for a long period of time after the data has been written … reduces the risk of data loss occurring in an area of the nonvolatile memory"). Thus, in my opinion, a POSITA would be motivated to measure a time-since-programming to identify blocks or pages potentially becoming unreliable, and use the flash controller 104 to

reduce data loss by copying data from an unreliable physical erase unit to a reliable one.

121. Sinclair-367 recognizes that flash memory has a retention period problem. For instance, Sinclair-367 discloses how a memory cell relies on "threshold voltage window … divided into a number of ranges" to create storage states. EX1009, ¶[0007]. But maintaining the voltage level within one of the slim ranges associated with certain storage states is difficult. "[S]torage levels do shift" over usage of the memory cell and surrounding memory cells. EX1009, ¶[0007]. One way to predict and stop an intolerable shift in voltage levels is to check if the time-since-programming is nearing the average retention period of a memory cell. This can alert flash controller 104 of potential data loss, and provide an opportunity to save the data. EX1010, ¶¶[0009], [0169].

122. Upon nearing or reaching the average retention period (***maintained metadata meets at least one criterion***), a POSITA would be motivated to copy data out of the unreliable block past its retention period (***copy data from a first one of the physical erase units***) and into a working physical block (***to a second one of the physical erase units***). EX1009, ¶[0136] ("Copying of data from a partially full metablock to a smaller metablock may be triggered by an elapse of time from the receipt of the stream of sectors of data 1305."). Copying data into a new block, such as through garbage collection, is faster than reading, erasing, and rewriting the same

block. *See also* EX1009, ¶¶[0007], [0015], [0136]. It also prevents data loss. EX1010, ¶[0169] ("maintenance of data that has been left unread for a long period of time after the data has been written … reduces the risk of data loss occurring in an area of the nonvolatile memory").

(n) 1[f][iii]: ***remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units.***

123. Sinclair-367 discloses that copying data away from a failing block to a new block also includes remapping the logical address from the old physical address to the new physical address (***remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units***). For example, the "logical address of the data … mapped into th[e] … address of any substitute erase block" if "the primary erase block is defective." EX1009, ¶[0109].

3. Claim 2

(a) 2[a]: ***The storage device of claim 1 wherein: in connection with the copy of data from the first one of the physical erase units to the second one of the physical erase units, the storage device is to copy all valid data from the first one of the physical erase units to the second one of the physical erase units; and***

124. Upon nearing or reaching the average retention period, a POSITA would be motivated to copy data out of the unreliable block past its retention period (***copy of data from the first one of the physical erase units to the second one of the***

physical erase units) and into a working physical block. A POSITA would understand that "valid data" refers to the latest version of a data element. Thus, a POSITA would have understood or found it obvious to **copy all valid data** from the unreliable block into the working physical block to avoid data loss and allow the obsolete block to be erased and reused. EX1009, ¶[0240] ("metablock may be considered a fully obsolete original metablock when all valid sectors within it have been copied. A fully obsolete original metablock may be erased.").

(b) 2[b]: ***the storage device is to generate the metadata for the second one of the physical erase units so as to correspond to the copy of the valid data from the first one of the physical erase units to the second one of the physical erase units.***

125. As I discussed above, Sinclair-367 discloses recording overhead data for each erase block, such as error correcting codes, number of program/erase cycles, and time of programming. *See* Limitation 1[f][i].

126. In my opinion, a POSITA would understand that this metadata needs to be moved or otherwise updated following a copy operation (generate the metadata for the second one of the physical erase units so as to correspond to the copy of the valid data from the first one of the physical erase units to the second one of the physical erase units) so that it remains accurate. For instance, Sinclair-367 discloses that metadata should be "updated after each cycle." EX1009, ¶[0108]. Thus, following the copy operation, the program/erase cycles and the time of programming

are changed such that a POSITA would understand the SSD will generate this metadata for the working physical block to which the data was copied (second one of the physical erase units). A POSITA would have also found it obvious to generate this metadata after a copy event so that the metadata remains accurate.

4. Claim 3

(a) ***The storage device of claim 2 wherein the storage device is further to automatically erase the first one of the physical erase units in connection with the copy of all valid data from the first one of the physical erase units to the second one of the physical erase units.***

127. Sinclair-367 discloses that a block may be considered "fully obsolete" when "all valid sectors within it have been copied," and further teaches that "[a] fully obsolete" block "may be erased." EX1009, ¶[0240]. Similarly, Reiter discloses that SSD must perform garbage collection to "erase" blocks with "'stale' or out-of-date" data. EX1004, 4:4-26. In my opinion, a POSITA would understand that the data in the unreliable block (***first one of the physical erase units***) is out-of-date because all valid data was copied into the working physical block (***the second one of the physical erase units***). Thus, in view of the disclosures or Sinclair-367 and/or Reiter, a POSITA would understand that in connection with copying the valid data from the ***first one of the physical erase units to the second one of the physical erase units***, the data should be automatically erased.

128. At minimum, it would have been obvious to automatically erase the unreliable block after the copy operation because, as explained in Reiter, "it would

be apparent to one of skill in the art" that if the data is not erased, the SSD would "fill up with data that is mostly out-of-date." EX1004, 4:4-26. If the data is no longer accessible, it should be erased immediately or at least immediately queued for erasure. Immediate erasure is simpler and comes with less risk of error.

5. Claim 4

(a) 4[a]: *The storage device of claim 3 wherein: the flash memory comprises multiple planes; and*

129. To the extent Reiter does not expressly disclose multiple planes, in my opinion it was obvious to modify Reiter to utilize a multiple plane architecture in view of Sinclair-367 which teaches that the flash memory may be organized into groups of blocks that span multiple planes:

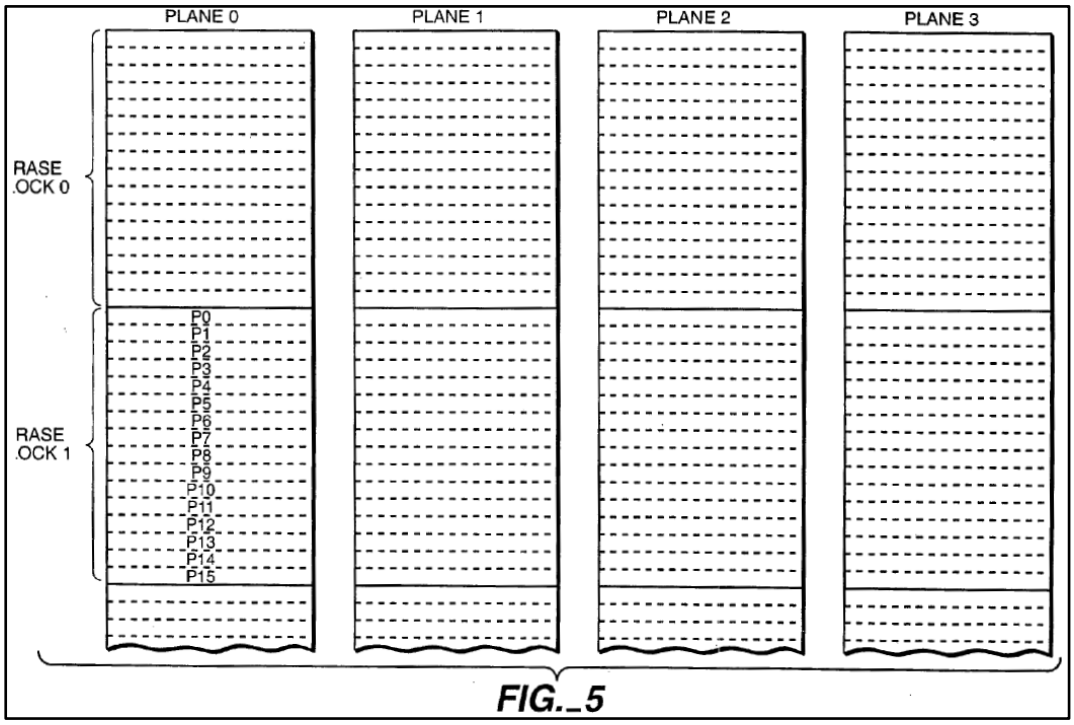


FIG._5

EX1009, FIG. 5; EX1009, ¶[0111]. Multi-plane architectures were well-known by the time of the '995 Patent and, as I discussed above (*see* Section IX.A.1), a POSITA would be motivated to incorporate multi-plane operations into Reiter's flash system to increase the degree of parallelism of operation and improve speed. Thus, it would have been obvious for the flash memory to comprise multiple planes.

(b) 4[b]: *the storage device is to, in connection with the automatic erase, automatically and concurrently erase a physical erase unit in each of the multiple planes.*

130. As I discussed above, in my opinion, the storage device of Reiter alone, or in combination with Sinclair-367, discloses and renders obvious automatically erasing the first one of the physical erase units in connection with the copy of all valid data from the first one of the physical erase units to the second one of the physical erase units. *See* Claim 3.

131. Sinclair-367 further discloses that to improve "efficiency" and "parallelism" the "erase blocks may be linked together" to form "virtual blocks" that may be programmed and "erased together." EX1009, ¶¶[0012], [0114] ("all blocks of the metablock [virtual block] are erased together"); *see also id*., ¶[0124] ("maximum parallelism is desirable"). The erased blocks that are linked together are "within different planes," such that the automatic erase will ***automatically and concurrently erase a physical erase unit in each of the multiple planes***. *Id*., ¶[0114] ("In order to increase the degree of parallelism of operation, erase blocks

within different planes are logically linked to form metablocks [virtual blocks]."), Fig. 6 (illustrating virtual block that links together erase blocks from each plane). A POSITA would have been motivated to implement this multi-plane erase command in Reiter's system for the reason I discussed above, including to improve parallelism. *See* Section IX.A.1.

6. Claim 5

(a) 5[a]: *The storage device of claim 4 wherein: for at least one of the multiple planes, the storage device is further to maintain metadata for each given physical erase unit which represents a number of times the given physical erase unit has been erased; and*

132. Sinclair-367 discloses that recorded overhead data for an erase block may "include a quantity related to the number of program/erase cycles experienced by the erase block." EX1009, ¶[0108]. Sinclair-367 also discloses that this metadata may be used "in a wear leveling algorithm." *Id*.

133. In my opinion, a POSITA would have understood Sinclair-367's teaching of wear-leveling algorithms to disclose the use of metadata indicating the number of times the given physical erase unit has been erased. Wear-leveling algorithms typically determine the need for wear-leveling by evaluating erase count metadata. For instance, consistent with the understanding of a POSITA, U.S. Pub. No. 2012/0096217 ("Son") discloses an exemplary wear-leveling algorithm that is initiated when a "wear count" that "indicates how many times a particular superblock has been erased" exceeds a threshold. EX1110, ¶¶39, 79, Fig. 7 (step

705).

(b) **[b]:** ***in connection with the erase of the first one of the physical erase units, the storage device is to update the metadata for the first one of the physical erase units to reflect an additional instance that the first one of the physical erase units has been erased.***

134. Sinclair-367 discloses that the program/erase cycles metadata (the metadata for the first one of the physical erase units) may be "updated after each cycle" (in connection with the erase of the first one of the physical erase units, the storage device is to update the metadata … to reflect an additional instance that the first one of the physical erase units has been erased). EX1009, ¶[0108]. In my opinion, a POSITA would have also found it obvious to update this metadata after an erasure. Doing so ensures that the metadata remains up-to-date and accurate.

7. Claim 7

(a) ***The storage device of claim 1 wherein the storage device is also to detect a failure condition of the physical erase units and is to remap each logical erase unit which corresponds to one of the physical erase units for which the failure condition is detected to instead correspond to one of the physical erase units for which the failure condition has not been detected and is to utilize the remapped correspondence to service one or more incoming data access requests.***

135. Reiter's flash storage system 100 (***storage device***) "maintain[s] a record of bad blocks", which includes ***detecting a failure condition of the physical erase units***. EX1004, 4:27-37. For example, "[o]n startup of flash memory storage system 100, mapping data … requires reconstruction." EX1004, 17:13-18. This includes

"scan[ning] and group[ing] … blocks" into various "categories," including "bad blocks." EX1004, 17:47-53. Bad blocks are detected if FTL 208 is "unsuccessful" in "read[ing] metadata that is stored in the … first page of the block" and if the following "erase [operation] of the [same] block was unsuccessful, the block cannot be erased and has failed." EX1004, 17:60-67. "[T]he block address is [then] added to the bad block list." EX1004, 18:1-13.

136. When a block is added to the bad block list, a POSITA would be motivated to remove the block from the corresponding superblock index 412 and block index 414 mapping in the summary page 420. Data read from the block cannot be trusted, given the unsuccessful read of the metadata in the first page of the block and subsequent erase operation on the block. Keeping the block in the summary page mapping risks erroneous data to be read and treated as correct.

137. Further, a POSITA would understand that, under Reiter, the lost block and data would be replaced with a working block written with the lost data. Reiter discloses how the flash memory storage system "might be configured such that data is striped across two or more [flash] dies analogously to hard drives in a redundant array of independent disks (RAID) …." EX1004, 8:46-50. RAID allows data lost on one disk to be reconstructed with data from the other redundant disks, thereby reconstructing lost data. *See, e.g.*, EX1204, ¶¶[0007]-[0009]; EX1205, ¶¶[0003]-[0006]; EX1206, 97-99; EX1207, 1:22-50; EX1208, 110-15; EX1209, 164-65, 168,

170. Because the previous block is an unreliable bad block, a POSITA would understand that a new block is needed for the reconstructed data. After losing data due to a failed block, a new block can be re-written with the reconstructed data, and the summary page mapping the logical block number 406 (***logical erase unit***) mapped to the failed block (***logical erase unit which corresponds to one of the physical erase units for which the failure condition is detected***) is ***remapped*** to this new block (***instead correspond to one of the physical erase units for which the failure condition has not been detected***). In this case, the data in the corresponding logical block number 406 can be trusted and used for later reads (***utilize the remapped correspondence to service one or more incoming data access requests***).

138. A POSITA would also be motivated to implement this failure detection and block replacement scheme. Reiter expressly acknowledges that "NAND devices can [only] be erased relatively few times before device failure …. Therefore, over the operational life of an SSD, blocks of flash memory will fail and become unusable." EX1004, 4:27-37. Moreover, the reduced life expectancy of SSDs due to bad blocks, as well as management and remapping of bad blocks, is well known in the prior art. For instance, consistent with the knowledge of a POSITA, Sinclair-996 discloses that a flash memory system may remap blocks that have become defective during memory operation. EX1111, ¶[0077] ("In a typical memory system … extra blocks may be provided as redundant blocks for ***substitution for other***

blocks that may become defective during the lifetime of the memory.”). Thus, to improve resiliency, extend lifespan, and reduce data loss of flash memory storage system 100, a POSITA would implement de-mapping bad blocks from the logical block numbers, reconstructing data lost to bad blocks via RAID, and remapping logical block numbers to working blocks with the reconstructed data.

8. Claim 8

(a) 8[a]: *The storage device of claim 1 wherein: each said circuitry comprises at least one processor; and*

139. In my opinion, Reiter discloses this limitation. For instance, Reiter’s flash memory storage system (***storage device***) includes flash controller 104. EX1004, FIG. 1. And flash controller 104 includes **processor** 116. EX1004, 7:19-39, FIG.1.

(b) 8[b]: *the storage device further comprises instructions stored on at least one physical non-transitory storage medium, said instructions when executed to control the at least one processor so as to operate as special purpose circuitry.*

140. In my opinion, Reiter’s flash memory storage system is a **storage device** comprising **processing** 116. EX1004, 7:19-39, FIG.1. “Processor 116 includes software and/or firmware as needed for operation.” EX1004, 7:31-34. A POSITA would understand that “[p]rocessor 116 includ[ing] software and/or firmware … for operation” requires ***instructions stored on at least one physical storage medium***. EX1004, 7:31-34. A POSITA would also understand that the

software and firmware instructions are **executed** to control the processor so that it operates to perform its programmed functions (***operate as special purpose circuitry***).

9. Claim 9

(a) ***The storage device of claim 8 wherein the storage device is to store the metadata in nonvolatile memory which is resident on said storage device, said nonvolatile memory being one of said flash memory or another memory.***

141. Reiter's flash memory storage system 100 (*storage device*) reserves a portion of each page of the flash memory for metadata (stores the metadata in nonvolatile memory which is resident on said storage device, said nonvolatile memory being one of said flash memory). EX1004, 9:26-51. "Flash memory is a type of non-volatile memory." EX1004, 1:34-40.

10. Claim 10

(a) **10[a]:** ***The storage device of claim 1 wherein: the storage device is to receive a maintenance request from a host which addresses a block device and a segment within the block device addressed by the maintenance request; and***

142. Reiter's flash memory storage system 100 (***storage device***) as a ZBC zoned block device can ***receive*** a reset write pointer command (***a maintenance request from a host***). The reset write pointer command resets the write pointer for a zone to the zone's starting logical block address. EX1006, 14, Figure 2:

lowest LBA	lowest LBA+1	lowest LBA+2	…	highest LBA

↑ write pointer

Figure 2 — Write pointer zone and write pointer after reset write pointer operation with no subsequent writes

143. A reset write pointer command targets a zone with a "ZONE ID" parameter (***segment***). EX1006, 29, Table 16:

Table 16 — RESET WRITE POINTER command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (9Fh)							
1	Reserved			SERVICE ACTION (14h)				
3	(MSB)							
•••	ZONE ID							
9								(LSB)
10								
•••	Reserved							
13								
14	Reserved							RESET ALL
15	CONTROL							

144. A POSITA would understand that a reset write pointer command must target a logical unit, volume, or partition (***addresses a block device***), for the same reasons as a ZBC write command. Similarly, the zone is also within the targeted logical unit, volume, or partition (***segment within the block device addressed by the maintenance request***).

(b) 10[b]: ***the storage device comprises circuitry to erase, in association with execution of the maintenance request, one or more of the physical erase units which are***

mapped to the segment addressed by the maintenance request.

145. A POSITA would be motivated to implement the reset write pointer command (***in association with execution of the maintenance request***) to cause flash memory storage system 100 (***storage device***) ***to erase*** the physical blocks (***one or more physical erase units***) associated with the targeted zone (***mapped to the segment addressed by the maintenance request***). After applying a reset write pointer command, the data in the targeted zone is no longer accessible. For instance, "[a] read operation on an LBA that has not been written since the last reset write pointer operation … shall return user data set to the last initialization pattern … set by … the manufacturer's default initialization pattern," not user data. EX1006, 16-17.

146. Given that user data is already no longer accessible, the physical blocks associated with the zone should be erased immediately or at least immediately queued for erasure. Immediate erasure is a simpler approach to implement, with less risk of error, as the effects of the reset write pointer command is fulfilled immediately and not potentially tangled up with later commands. Alternatively, immediate queueing for erasure gives flash memory storage system 100 flexibility on erasing those blocks when normal read and write operations are not being processed. Delay of read and write operations for erasure operations is a complaint expressed by Reiter. "[G]arbage collection routines," such as for "erasing out-of-

date blocks," "impact performance of the flash memory system by utilizing processor resources and potentially delaying write operations …." EX1004, 4:19-26.

11. Claim 11

(a) ***The storage device of claim 10 wherein the storage device is further to track for each given physical erase unit second metadata representing a number of times the given erase unit has been erased and is to update, in connection with execution of the maintenance request, the second metadata for each physical erase unit that is erased in association with execution of the maintenance request.***

147. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 5.

12. Claim 13

(a) **13[a]:** ***The storage device of claim 11 wherein: the flash memory comprises multiple planes; and***

148. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 4[a].

(b) **13[b]:** ***the storage device is to, in connection with execution of the maintenance request, automatically and concurrently erase a physical erase unit in each of the multiple planes.***

149. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 4[b].

13. Claim 14

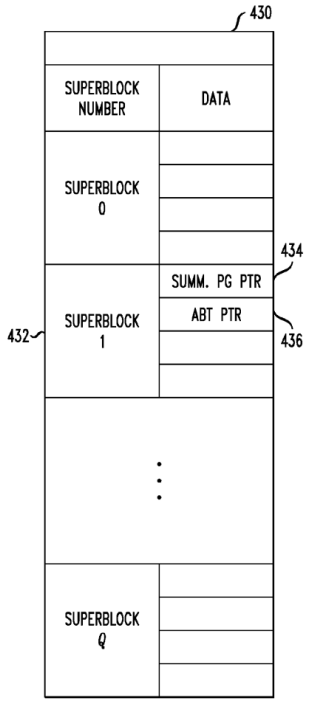
(a) 14[a]: *The storage device of claim 1 wherein: the flash memory is to have block devices, including the addressed block device; and*

150. As I explained in connection with Limitation 1[b][ii], in the Reiter-ZBC combination, the flash memory identifies addressed block devices from the first address portion. The first address portion identifies the targeted logical unit, volume, or partition, which is the addressed block device.

(b) 14[b]: *the storage device comprises circuitry to, in connection with an incoming data read request, identify a block device, retrieve a look-up table specific to the block device addressed by the incoming data read request, and identify a physical storage location in the flash memory dependent on the look-up table.*

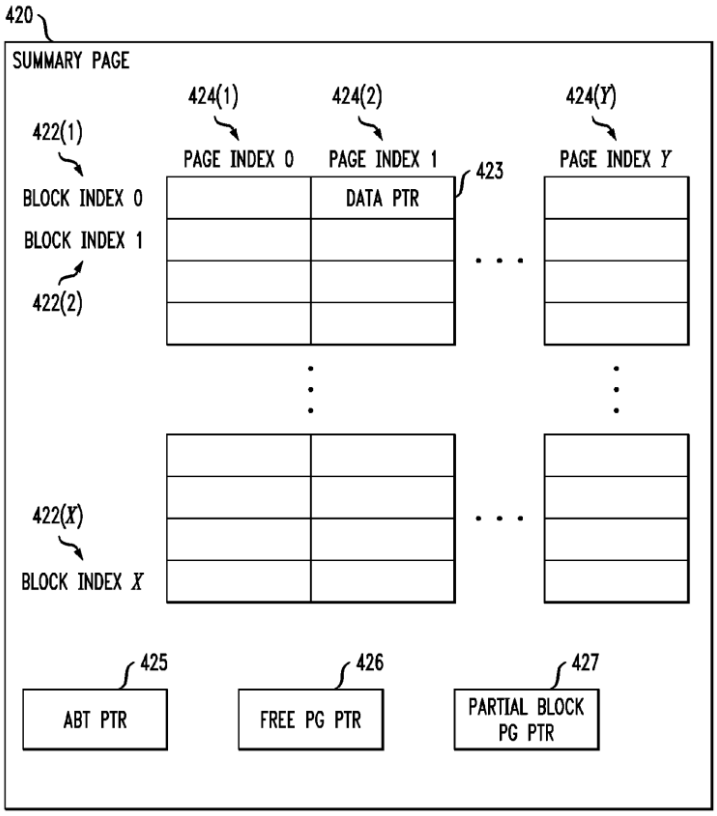
151. As I discussed with limitations 1[c]-[e], Reiter's flash controller 104 (***circuitry is to***) translates a read operation and write operation's logical block address (***in connection with an incoming data read request***) into superblock number 410, block index 412, and page index 414. Superblock index 410 indexes page global directory (PGD) 430 to obtain the "summary page associated with [the] Superblock" (***identify a block device, retrieve a look-up table***). EX1004, 10:13-19, FIG. 4c:

FIG. 4c



152. The "summary page" (***look-up table***) "contains mapping data for the … Superblock." EX1004, 9:52-54. "For example, summary page 420 contains the block index and page index … for all X blocks and Y pages in [the] Superblock …." EX1004, 9:54-58, FIG. 4b. Using the translated block index 412 and page index 414, summary page's "block indices 422(1)-422(X) and page indices 424(1)-424(Y)" are indexed (***dependent on the look-up table***) to obtain "data pointer 423, which points to the [corresponding] physical address" (***identify a physical storage location in the flash memory***). EX1004, 9:60-63, FIG. 4b:

FIG. 4b



153. A POSITA would understand that the summary page is ***specific*** to the ZBC read command's targeted logical unit, volume, or partition (***block device addressed by the incoming data read request***). Each of these logical structures "shall begin with block zero and be contiguous up to the last logical block of that logical unit or within that volume or partition." EX1008, 61. In my opinion, a POSITA would further understand that the overlapping logical address ranges require separate address translation structures, including the summary pages, because Reiter's disclosed address translation only take the incoming logical block address as input. If the translation structures were shared, the overlapping logical block addresses for each logical unit, volume, and partition would be translated to

the same physical address. In order to resolve ambiguity of overlapping logical block addresses, each logical unit, volume, and partition need its own address translation structures.

14. Claim 15

(a) 15[a]: ***The storage device of claim 1 wherein: the storage device further comprises circuitry to track additional metadata for the addressed segment, where said additional metadata indicates an extent to which memory comprising at least one of the physical erase units mapped to the addressed segment can continue to receive additional write data without being erased; and***

154. Reiter's flash memory storage system's 100 (***storage device***) flash controller 104 (***circuitry to***) maintains a partial block pointer (***track additional metadata***) for each summary page (***for the addressed segment***). The partial block pointer points to the "location of the next page of a partially written block" (***metadata indicates an extent to which memory comprising at least one of the physical erase units mapped to the addressed segment can continue to receive additional write data without being erased***). EX1004, 9:51-10:1.

155. Alternatively, based on my review of Patent Owner's infringement contentions, I understand that Patent Owner asserts that a zone's write pointer discloses the limitation. EX1011, 170-171. Specifically, Patent Owner relies on NVMe Zoned Namespace Specification's write pointer for sequential write required zones, where "the extent to which [memory] can continue to receive additional write data without being erased can be calculated by subtracting the write pointer from the

sum of the zone start LBA and the zone capacity (*i.e.*, remaining space = ZSLBA + ZCAP – write pointer)." EX1011, 170-171. ZBC has the same disclosure. ZBC discloses a write pointer for a sequential write required zone, where the remaining memory in the zone can be calculated with the same formula. EX1006, 14, 17, Figure 3:

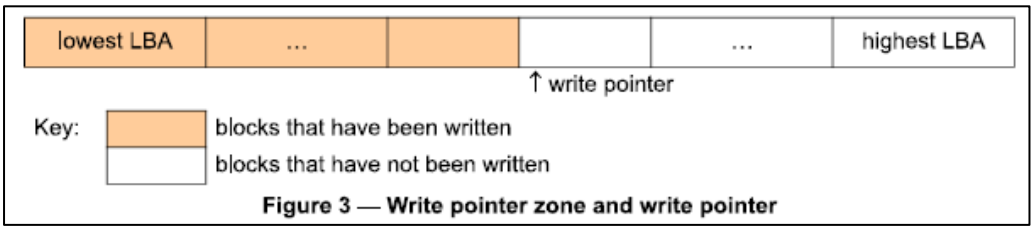


Figure 3 — Write pointer zone and write pointer

(b) 15[b]: ***the storage device further comprises circuitry to transmit information to a host dependent on said tracked additional metadata.***

156. In the combined Reiter-ZBC system, Reiter's flash memory storage system's 100 (***storage device***) flash controller 104 (***circuitry***) perform ZBC report zone command. The ZBC report zone command "requests that the device server return the zone structure of the zoned block device" (***transmit information to a host***). EX1006, 24. The ZBC report zone command includes an option where the reported information is the "[l]ist [of] empty zones." EX1006, 25, Table 11. An empty zone is a zone where the "device server has not performed a write operation to this … zone since the write pointer was set to the lowest LBA of this zone." EX1006, 28. The reported list of empty zones is ***dependent on*** the partial block values (***tracked metadata***).

157. Alternatively, based on my review of Patent Owner's infringement contentions, I understand that Patent Owner asserts that the NVMe Zoned Namespace Specification's report zones command returning the write pointer teaches this limitation. EX1011, 172-173. ZBC's report zone command does the same. The report zone command sends back "parameter data," which includes a list of zone descriptors. EX1006, Table 12:

Table 12 — REPORT ZONES parameter data

Bit / Byte	7	6	5	4	3	2	1	0
1	(MSB)							
•••				ZONE LIST LENGTH (n–63)				
3								(LSB)
4				Reserved				SAME
5								
•••				Reserved				
63								
Zone descriptors list								
64								
•••				Zone descriptor [first]				
127								
				⋮				
n–63								
•••				Zone descriptor [last]				
n								

158. The returned zone descriptors include the zone's write pointer. EX1006, Table 13.

Table 13 — Zone descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2	Reserved							
•••								
7								
8	(MSB)							
•••	ZONE LENGTH							
15								(LSB)
16	(MSB)							
•••	ZONE START LBA							
23								(LSB)
24	(MSB)							
•••	WRITE POINTER LBA							
31								(LSB)
32	Reserved							
•••								
63								

15. Claim 18

(a) ***The storage device of claim 1 wherein the storage device is to store information identifying a number of structures in the flash memory and is to transmit the information identifying the number of structures to a host, wherein the host is to issue at least one request to the storage device in which the at least one request is dependent on the information transmitted to the host identifying the number of structures, and wherein the storage device comprises circuitry to fulfill the at least one request by issuing commands to the flash memory that each operate on a quantum of flash memory which is dependent on the information transmitted to the host identifying the number of structures.***

159. ZBC's report zones command "return[s]" (transmit) a "[l]ist of zones in the zoned block device" that is stored by the storage device back to [the] host via "parameter data" that includes the length of the zone list and zone descriptors indicating the starting LBA and length of each zone (information identifying the

number of structures to a host). EX1006 25, Table 11, Tables 12, 13:

Table 12 — REPORT ZONES parameter data

Bit / Byte	7	6	5	4	3	2	1	0
1	(MSB)							
•••	ZONE LIST LENGTH (n–63)							
3								(LSB)
4	Reserved							SAME
5	Reserved							
•••								
63								
Zone descriptors list								
64	Zone descriptor [first]							
•••								
127								
	⋮							
n–63	Zone descriptor [last]							
•••								
n								

Table 13 — Zone descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2 ••• 7	Reserved							
8 ••• 15	(MSB) ZONE LENGTH (LSB)							
16 ••• 23	(MSB) ZONE START LBA (LSB)							
24 ••• 31	(MSB) WRITE POINTER LBA (LSB)							
32 ••• 63	Reserved							

160. Because a POSITA would be motivated to map zones onto superblocks as I discussed above (Section IX.A.1), a POSITA would understand storing, or be motivated to store, zone metadata in a superblock's summary page, which stores a superblock's metadata. EX1004, 9:52-10:12, FIG. 4b.

161. Executing a valid reset write pointer command (***host is to issue at least one request to the storage device***) depends on information reported by the report zones command (***information transmitted to the host identifying the number of structures***). A valid reset write pointer command requires a "ZONE ID" which "specifies the zone start LBA" and is provided by the report zones command. EX1006, 25, 29, Table 13.

162. Reiter's flash system 100 executing a reset write pointer command (***circuitry to fulfill the at least one request by issuing commands to the flash memory***) "sets a write pointer to the lowest LBA of a write pointer zone," which a POSITA would understand discloses ***a quantum of flash memory which is dependent on the information transmitted to the host identifying the number of structures***. EX1006, 3.

16. Claim 19

(a) ***The storage device of claim 18 wherein the information transmitted to the host identifying the number of structures identifies an amount of flash memory that is to be erased as a unit.***

163. The list of zones reported by the report zones command (***the information transmitted to the host identifying the number of structures***) provides the length of each zone (***identifies an amount of flash memory that is to be erased as a unit***). Specifically, the returned zone descriptors indicate the length of each zone. EX1006, 25, Tables 11-13:

Table 13 — Zone descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2 ... 7	Reserved							
8 ... 15	(MSB) ... ZONE LENGTH ... (LSB)							
16 ... 23	(MSB) ... ZONE START LBA ... (LSB)							
24 ... 31	(MSB) ... WRITE POINTER LBA ... (LSB)							
32 ... 63	Reserved							

164. In my opinion, a POSITA would be motivated to implement the reset write pointer command to cause erasure of the targeted zone as I discussed above in connection with Claim 10[b]. Because the reset write pointer command erases an entire zone, a POSITA would understand the zone length ***identifies an amount of flash memory that is to be erased as a unit***.

17. Claim 20

(a) ***The storage device of claim 1 wherein the addressed segment corresponds to an integer number of physical erase units for which a failure condition has not been detected, and the third address portion specifies a location within a sequential range of locations that correspond to the integer number of physical erase units.***

165. As I discussed above, a ZBC zone constitutes ***an addressed segment***.

See Limitation 1[b][iii].

166. In my opinion, a POSITA would be motivated to require that zones (***addressed segment***) contain a whole number of physical units (***corresponds to an integer number of physical erase units***). As discussed with Claim 10, a ZBC zoned block device can receive a reset write pointer command to cause erasure of the targeted zone. But if a zone contained a non-whole number of physical units, erasing a full zone causes erasure of units only partially within the zone. Because erasure of a unit erases everything in the unit, the portion of the units not associated with the zone also gets erased, risking data loss. Avoiding data loss by copying out data on the other portion increases complexity of the system and delays execution of the command and following commands. To avoid data loss, while maintaining capacity and speed, a POSITA would require zones to contain a whole number of physical blocks. Further, as discussed with Claim 7, Reiter discloses detecting blocks that have failed. A POSITA would be motivated to include only good physical blocks (***physical erase units for which a failure condition has not been detected***) because physical blocks that are bad cannot reliably store data.

167. The Reiter-ZBC system derives from the write command address information a logical sector number, or alternatively a bitstring of logical block number 406 and logical page number 408 (***third address portion***). *See* Ground 1, Limitation 1[b][i]. Because zones of ZBC are "contiguous, non-overlapping" ranges

of LBAs (EX1006, 13), the logical sector number, or alternatively the logical block and page number, specify ***a location within a sequential range of locations that correspond to the integer number of physical erase units***.

18. Claim 26

(a) 26[a]: *The storage device of claim 1 wherein: the flash memory comprises multiple planes; and*

168. As I discussed above in connection with Claim 4[a], the Reiter-ZBC-Sinclair-367 system comprises multiple planes, as disclosed in Sinclair-367. EX1009, FIG. 5, ¶[0111].

(b) 26[b]: *the storage device is to, for each one of a set of the incoming data access requests having sequential addressing, write data accompanying the set of incoming data access requests concurrently into respective ones of the multiple planes.*

169. A ZBC write command includes an LBA and "transfer length." EX1007.

Table 112 — WRITE (32) command

Byte \ Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	Reserved							
•••								
5								
6	Reserved			GROUP NUMBER				
7	ADDITIONAL CDB LENGTH (18h)							
8	(MSB)							
9	SERVICE ACTION (000Bh)							(LSB)
10	WRPROTECT			DPO	FUA	Reserved	Obsolete	Reserved
11	Reserved							
12	(MSB)							
•••	LOGICAL BLOCK ADDRESS							
19								(LSB)
20	(MSB)							
•••	EXPECTED INITIAL LOGICAL BLOCK REFERENCE TAG							
23								(LSB)
24	(MSB)							
25	EXPECTED LOGICAL BLOCK APPLICATION TAG							(LSB)
26	(MSB)							
27	LOGICAL BLOCK APPLICATION TAG MASK							(LSB)
28	(MSB)							
•••	TRANSFER LENGTH							
31								(LSB)

"TRANSFER LENGTH … specifies the … logical blocks … that shall be transferred … starting [from] … the [LBA]." EX1007, 198. A POSITA would understand that with a large transfer length, a ZBC ***write*** command causes several write operations across ***sequential addresses***.

170. As I explained in connection with Claim 4, to increase throughput for these large write commands, a POSITA would be motivated to implement Reiter's flash media 118 with multiple planes, and metablocks spanning the multiple planes, as disclosed in Sinclair-367. *See also* EX1009, ¶¶[0111]-[0114]. "All blocks of the metablock are erased together, and pages of each block are generally programmed

and read simultaneously" (***write data accompanying the set of incoming data access requests concurrently into respective ones of the multiple planes***). EX1009, ¶[0114].

19. Claim 27

(a) ***The storage device of claim 1 wherein the division operation is applied to subdivide the third address portion into respective first and second bit fields.***

171. As I discussed above, a POSITA would be motivated to modify Reiter's flash controller 104 to derive logical sector number 404, or alternatively bitstring of logical block number 406 and logical page number 408, or alternatively logical page number 408 or page index 414 (***third address portion***), from ZBC write commands. Limitation 1[b][i]. As I also discussed above, a POSITA would be motivated to divide by a certain power-of-two to split the bitstring (***subdivide the third address portion***) into a first and second bitfields for the logical block number 406 and logical page number 408 (***subdivide the third address portion into respective first and second bit fields***).

20. Claim 28

(a) **28[a]:** ***The storage device of claim 1 wherein: for each of the incoming data write requests, the circuitry is to write data into a specific physical storage location within the identified one of the physical erase units and is to identify an address value corresponding to the specific physical storage location***

172. Reiter discloses translating a read request's LBA to "data pointer 423,

which points to the physical address." EX1004, 9:60-63. The same translation applies to write operations (***the incoming data write requests***), except the resulting pointer is called the "active page address" (***write data into a specific physical storage location within the identified one of the physical erase units***). EX1004, 8:54-56, 13:55-60, FIG. 6b.

173. ZBC write commands also cause flash system 100 to transmit information back to the host the upon command completion. "The SCSI command set assumes an underlying request-response protocol." EX1008, xl. The response is "returned … on completion of a SCSI command." EX1008, 11.

174. In my opinion, a POSITA would be motivated to modify the transmission to include information identifying the completed command. Reiter discloses processing multiple commands in parallel. *See* EX1004, 8:50-53, 15:55-64, 16:62-65. Because concurrent commands can complete out of order, information is needed in the response to identify what command completed. Identifying the completed command allows the host to verify command completion or repeat commands that return errors. Because information identifying the command also indicates the LBA associated with the command, such information represents ***an address value corresponding to the specific physical storage location.***

175. Moreover, ZBC discloses incorporating such information in some circumstances. For instance, if a ZBC write command writes across two or more

zones, the zoned block device flags "terminates [the] command" and "return[s] the write pointer[,] … specified by the [LBA] field of that command[,] in the information field of the sense data." EX1006, 18. "[S]ense data" is part of the SCSI response. EX1008, 21.

(b) 28[b]: *the storage device further comprises circuitry to transmit the address value to a host;*

176. As discussed above in connection with Limitation 28[a], the Reiter-ZBC system discloses circuitry to transmit the address value to a host. For instance, a POSITA would be motivated to modify the transmission to include information identifying the completed write command. *Id*. Because information identifying the command also indicates the LBA associated with the command, such information ***transmits the address value to a host***.

(c) 28[c]: *the storage device is also to receive incoming data read requests, each accompanied by an associated read address; and*

177. In my opinion, a POSITA would be motivated to implement Reiter's flash controller 104 to accept and execute ZBC read commands (***incoming data read requests***) for the reasons I discussed above. *See* Section IX.A.1. A ZBC read command includes an LBA (***an associated read address***). EX1007, Table 63:

Table 63 — READ (32) command

Byte \ Bit	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	Reserved							
...								
5								
6	Reserved			GROUP NUMBER				
7	ADDITIONAL CDB LENGTH (18h)							
8	(MSB)							
9	SERVICE ACTION (0009h)							(LSB)
10	RDPROTECT			DPO	FUA	RARC	Obsolete	Reserved
11	Reserved							
12	(MSB)							
...	LOGICAL BLOCK ADDRESS							
19								(LSB)
20	(MSB)							
...	EXPECTED INITIAL LOGICAL BLOCK REFERENCE TAG							
23								(LSB)
24	(MSB)							
25	EXPECTED LOGICAL BLOCK APPLICATION TAG							(LSB)
26	(MSB)							
27	LOGICAL BLOCK APPLICATION TAG MASK							(LSB)
28	(MSB)							
...	TRANSFER LENGTH							
31								(LSB)

(d) 28[d]: ***for each of the incoming data read requests which is directed to the data storage at the specific physical storage location, the associated read address is dependent on the address value which was transmitted to the host.***

178. In my opinion, a POSITA would be motivated to restrict subsequent read commands (***each of the incoming data read requests***) targeting LBAs subject to pending write operations (***which is directed to the data storage at the specific physical storage location***) until after receiving a completion response of the pending write operations (***the associated read address is dependent on the address value***

which was transmitted to the host). Incorporating command-identifying information into the SCSI response ensures command completion. Allowing read commands prior to write command completion risks reading incorrect data. Because incoming read requests are restricted to LBAs without pending write commands, the LBAs of incoming read commands that were previously written are dependent on the response's command-identifying information.

21. Claim 29

(a) 29[pre]: *A storage device comprising:*

179. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[pre].

(b) 29[a]: *flash memory having physical erase units; and*

180. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[a].

(c) 29[b]: *circuitry to, for address information provided in association with each of incoming data write requests:*

181. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b].

(d) 29[b][i]: *derive from the incoming address information a first address portion, a second address portion and a third address portion,*

182. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][i].

(e) 29[b][ii]: ***identify an addressed block device from the first address portion,***

183. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][ii].

(f) 29[b][iii]: ***identify an addressed segment within the addressed block device from the second address portion, and***

184. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][iii].

(g) 29[b][iv]: ***identify an addressed storage location within the addressed segment from the third address portion,***

185. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][iv].

(h) 29[c]: ***wherein the circuitry is to subdivide the address information into bit fields, is to identify from a first one of the bit fields a logical erase unit within the addressed segment, and is to identify the addressed storage location dependent on the second one of the bit fields, and***

186. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[c]-1[d].

(i) 29[d]: ***wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and***

187. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[d].

(j) 29[e]: ***program associated data into the identified one of the physical erase units at a location selected dependent on the addressed storage location;***

188. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[e].

(k) 29[f]: ***wherein the storage device also comprises circuitry to:***

189. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f].

(l) 29[f][i]: ***maintain metadata for each physical erase unit which contains data, the metadata representing a time since data was programmed in the respective erase unit,***

190. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f][i].

(m) 29[f][ii]: ***copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and, in association with said copy of the data, generate the metadata for the second one of the physical erase units so as to correspond to the copy of the data from the first one of the physical erase units to the second one of the physical erase units; and***

191. As I discussed above, Sinclair-367 renders obvious copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units. *See* Limitation 1[f][ii], Claim 2. For instance, upon nearing or reaching the average retention period (***maintained metadata meets at least one criterion***), a POSITA would be motivated

to copy data out of the unreliable block past its retention period (***copy data from a first one of the physical erase units***) and into a working physical block (***to a second one of the physical erase units***). *Id*.

192. Sinclair-367 also discloses that metadata should be "updated after each cycle." EX1009, ¶[0108]. For instance, following the copy operation the program/erase cycles and the time of programming are changed such that the SSD must generate the metadata for the working physical block to which the data was copied (***second one of the physical erase units***). A POSITA would have also found it obvious to generate this metadata after a copy event so that the metadata remains accurate.

(n) 29[f][iii]: ***remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units.***

193. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f][iii].

22. Claim 30

(a) 30[pre]: ***A storage device comprising:***

194. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[pre].

(b) 30[a]: ***flash memory, the flash memory to have one or more block devices at a hierarchically-superior level, segments within each of the block devices at a hierarchically-intermediate level, and addressable***

storage locations within each of the segments at a hierarchically-inferior level, each segment including a mutually exclusive set of one or more logical erase units that each map to a physical erase unit, each segment being associated with only one of the block devices, each logical erase unit being associated with only one of the segments; and

195. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[a], 1[b][ii]-1[d].

(c) 30[b]: ***circuitry to, for each of incoming data write requests:***

196. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b].

(d) 30[b][i]: ***derive, from incoming address information, a first address portion, a second address portion and a third address portion,***

197. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][i].

(e) 30[b][ii]: ***identify an addressed one of the one or more block devices from the first address portion,***

198. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][ii].

(f) 30[b][iii]: ***identify an addressed one of the segments within the addressed block device from the second address portion, and***

199. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][iii].

(g) 30[b][iv]: ***identify an addressed storage location within the addressed one of the segments from the third address portion,***

200. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[b][iv].

(h) 30[c]: ***wherein the circuitry is to subdivide the address information into bit fields, is to identify from a first one of the bit fields a logical erase unit within the addressed segment, and is to identify the addressed storage location dependent on the second one of the bit fields, and***

201. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[c] and Claim 29[c].

(i) 30[d]: ***wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and***

202. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 30[d].

(j) 30[e]: ***program associated data into the identified one of the physical erase units at a location selected dependent on the addressed storage location;***

203. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 30[e].

(k) 30[f]: ***wherein the storage device also comprises circuitry to:***

204. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 30[f].

(l) **30[f][i]:** ***maintain metadata for each physical erase unit which contains valid data, the metadata representing a time since data was programmed in the respective erase unit,***

205. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f][i].

(m) **30[f][ii]:** ***copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and, in association with said copy of the data, generate the metadata for the second one of the physical erase units so as to correspond to the copy of the data; and***

206. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f][ii], 29[f][ii].

(n) ***30[f][iii]: remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units; and***

207. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 1[f][iii].

(o) **30[g]:** ***wherein the storage device is to generate the metadata for the second one of the physical erase units so as to correspond to a time that the data was copied from the first one of the physical erase units to the second one of the physical erase units.***

208. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 2[b].

B. GROUND II: CLAIMS 6, 21 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC AND SINCLAIR-367

1. Motivation To Combine

209. As I discussed above, a POSITA would have been motivated to combine the teachings of Reiter with ZBC and Sinclair-367, as I discussed above. *See* Section IX.A.1.

210. A POSITA would have further been motivated to incorporate Yamada's maintenance request and determination units in Reiter's flash system and host processor, respectively. Reiter complains that "[g]arbage collection routines impact performance of the flash memory system by utilizing processor resources and potentially delaying write operations …." EX1004, 4:23-26. In Yamada, "maintenance is performed without being … performed together with normal system processing," but "with sufficiently long processing time." EX1010, Abstract. The maintenance request unit on the flash device determines and requests maintenance operations from the maintenance determination unit. The maintenance determination unit monitors status of the host, "determin[es] that the access device is not engaged in normal processing," and performs maintenance. EX1010, Abstract, ¶¶[0018]-[0019], [0022]. Because maintenance scheme is implemented on a nonvolatile system analogous like that in Reiter, a POSITA would have a reasonable expectation of success implementing Reiter's flash system with this scheme.

211. Moreover, Reiter, ZBC, SBC-4, SPC-4, Sinclair-367, and Yamada disclose analogous art. All relate to storage devices, including the management and control of storage devices.

2. Claim 6

(a) 6[a]: ***The storage device of claim 1 wherein: the storage device further comprises circuitry to compare the metadata to at least one threshold and, responsive to detection that specific metadata has satisfied the at least one threshold, to transmit information to a host wherein such transmitted information conveys address information for data stored at a physical storage location associated with the specific metadata; and***

212. Yamada discloses a maintenance request unit within a memory controller that can generate a "maintenance request" to the host when a criterion value is exceeded. EX1010, ¶[0032]; *see also id*. ¶¶[0033] ("The maintenance request unit dynamically changes the criterion value used to determine whether to generate a maintenance request in accordance with a status of data stored in the nonvolatile memory."). While in the preferred embodiment of Yamada the criterion value used is the number of errors, a POSITA would understand Yamada to disclose, or at least suggest, that the criterion value could include other information, such as wear count. For instance, a POSITA would be motivated to trigger a maintenance request upon detecting a physical block time since programming nears or reaches the average retention period (***compare the metadata to at least one threshold and, responsive to detection that specific metadata has satisfied the at least one***

threshold).

213. As indicated by the name of Yamada's units, maintenance requests should be triggered for garbage collections, including operations to reduce the risk of lost data. After the average retention period, a memory cell can be unreliable, risking loss of the data stored in the cell. EX1010, ¶[0009] ("Data loss can occur in an area that has not been refreshed for a sufficiently long period of time."). Saving that data from loss is done through garbage collection or maintenance operations, such as those issued from a maintenance request.

214. Yamada thus teaches or at least suggests that the maintenance request unit that resided in the memory controller (EX1010, Fig. 1) may compare and transmit a maintenance request to the host when a threshold criterion value for maintenance, such as the average retention period, is exceeded (EX1010, ¶¶[0033], [0189]). A POSITA would have also been motivated to implement this type of shared memory management because it avoids the storage device having to expose its performance characteristics to the host. By having the memory controller determine whether a threshold has been exceeded internally and then notifying the host of the need for a management operation, the vendor of the storage device can avoid exposing this information to the host. This type of shared management in which the memory controller performs the comparison and the host sends the management command is also beneficial for other reasons, including that it avoids

overloading the memory controller, improves data retention policies, and enables the system to evaluate devices conditions when scheduling the maintenance.

215. Yamada further discloses maintenance determination unit 152 in the host that receives a "logical address included in [a] maintenance request transmitted from … maintenance request unit 115" on the flash storage device (***transmit information to a host wherein such transmitted information conveys address information for data stored at a physical storage location associated with the specific metadata***). EX1010, ¶[0089].

216. In my opinion, a POSITA would be motivated to incorporate Yamada's maintenance determination unit and maintenance request unit scheme because Yamada's implementation reduces the delay caused by garbage collection procedures. Reiter makes this complaint expressly, stating that "[g]arbage collection routines impact performance of the flash memory system by utilizing processor resources and potentially delaying write operations to the flash media." EX1004, 4:23-26. The maintenance determination unit, by being on the host, can determine the "system status" of the host, and is able to issue "data maintenance … without being required to be performed together with normal system processing." EX1010, Abstract.

(b) 6[b]: *the copy of data by the storage device from the first one of the physical erase units to the second one of the*

physical erase units is dependent on the transmitted information.

217. As I mentioned above, Yamada discloses a maintenance determination unit 152 in the host that receives a "logical address included in [a] maintenance request transmitted from … maintenance request unit 115" on the flash storage device. EX1010, ¶[0089]. The logical address information (***transmitted information***) is used to initiate a maintenance operation, such as a refresh operation or other maintenance operations such as wear leveling or garbage collection, on the specified block. EX1010, ¶¶[0030] ("maintenance includes refresh and/or wear leveling of the nonvolatile memory"), [0033]. A POSITA would also understand these maintenance operations copy data to the new erase block.

3. Claim 21

(a) ***The storage device of claim 1 wherein the storage device further comprises circuitry to compare the metadata to at least one threshold and, responsive to detection that specific metadata has satisfied the at least one threshold, to transmit information to a host wherein such transmitted information conveys address information for data stored in a physical storage location associated with the specific metadata prior to the copy of the data to the second one of the physical erase units.***

218. In my opinion, this limitation is satisfied for the reasons I discussed in connection with Claim 6.

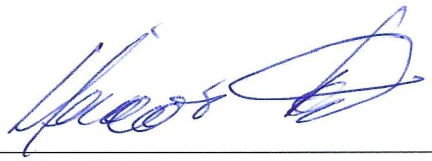
X. CONCLUSION

219. I understand that this Declaration will be filed as evidence in a

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

220. I declare that all statements made in this Declaration of my own knowledge are true to the best of my knowledge and that all statements made on information and belief are believed to be true. These statements were made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: August 5, 2025



Heiner Litz, Ph.D.

APPENDIX A – CLAIM LISTING

Claim 1	
1pre	1. A storage device comprising:
1a	flash memory having physical erase units; and
1b	circuitry to, for each of incoming data write requests:
1bi	derive, from incoming address information, a first address portion, a second address portion and a third address portion,
1bii	identify an addressed block device from the first address portion,
1biii	identify an addressed segment within the addressed block device from the second address portion, and
1biv	identify an addressed storage location within the addressed segment from the third address portion,
1c	wherein the circuitry is to use a division operation on at least one of the incoming address information or information derived therefrom, to identify a logical erase unit within the addressed segment, and
1d	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and
1e	program associated data into the identified one of the physical erase units;
1f	wherein the storage device also comprises circuitry to:
1fi	maintain metadata for each physical erase unit which contains data, the metadata representing a time since data was programmed in the respective erase unit,
1fii	copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and
1fiii	remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units.
Claim 2	

2a	The storage device of claim 1 wherein: in connection with the copy of data from the first one of the physical erase units to the second one of the physical erase units, the storage device is to copy all valid data from the first one of the physical erase units to the second one of the physical erase units; and
2b	the storage device is to generate the metadata for the second one of the physical erase units so as to correspond to the copy of the valid data from the first one of the physical erase units to the second one of the physical erase units.
Claim 3	
3	The storage device of claim 2 wherein the storage device is further to automatically erase the first one of the physical erase units in connection with the copy of all valid data from the first one of the physical erase units to the second one of the physical erase units.
Claim 4	
4a	The storage device of claim 3 wherein: the flash memory comprises multiple planes; and
4b	the storage device is to, in connection with the automatic erase, automatically and concurrently erase a physical erase unit in each of the multiple planes.
Claim 5	
5a	The storage device of claim 4 wherein: for at least one of the multiple planes, the storage device is further to maintain metadata for each given physical erase unit which represents a number of times the given physical erase unit has been erased; and
5b	in connection with the erase of the first one of the physical erase units, the storage device is to update the metadata for the first one of the physical erase units to reflect an additional instance that the first one of the physical erase units has been erased.
Claim 6	

6a	The storage device of claim 1 wherein: the storage device further comprises circuitry to compare the metadata to at least one threshold and, responsive to detection that specific metadata has satisfied the at least one threshold, to transmit information to a host wherein such transmitted information conveys address information for data stored at a physical storage location associated with the specific metadata; and
6b	the copy of data by the storage device from the first one of the physical erase units to the second one of the physical erase units is dependent on the transmitted information.
Claim 7	
7	The storage device of claim 1 wherein the storage device is also to detect a failure condition of the physical erase units and is to remap each logical erase unit which corresponds to one of the physical erase units for which the failure condition is detected to instead correspond to one of the physical erase units for which the failure condition has not been detected and is to utilize the remapped correspondence to service one or more incoming data access requests.
Claim 8	
8a	The storage device of claim 1 wherein: each said circuitry comprises at least one processor; and
8b	the storage device further comprises instructions stored on at least one physical non-transitory storage medium, said instructions when executed to control the at least one processor so as to operate as special purpose circuitry.
Claim 9	
9	The storage device of claim 8 wherein the storage device is to store the metadata in nonvolatile memory which is resident on said storage device, said nonvolatile memory being one of said flash memory or another memory.
Claim 10	
10a	The storage device of claim 1 wherein: the storage device is to receive a maintenance request from a host which addresses a block device and a segment within the block device addressed by the maintenance request; and

10b	the storage device comprises circuitry to erase, in association with execution of the maintenance request, one or more of the physical erase units which are mapped to the segment addressed by the maintenance request.
Claim 11	
11	The storage device of claim 10 wherein the storage device is further to track for each given physical erase unit second metadata representing a number of times the given erase unit has been erased and is to update, in connection with execution of the maintenance request, the second metadata for each physical erase unit that is erased in association with execution of the maintenance request.
Claim 12	
12	The storage device of claim 11 wherein the storage device further comprises circuitry to selectively transmit to the host information which is dependent upon the second metadata.
Claim 13	
13a	The storage device of claim 11 wherein: the flash memory comprises multiple planes; and
13b	the storage device is to, in connection with execution of the maintenance request, automatically and concurrently erase a physical erase unit in each of the multiple planes.
Claim 14	
14a	The storage device of claim 1 wherein: the flash memory is to have block devices, including the addressed block device; and
14b	the storage device comprises circuitry to, in connection with an incoming data read request, identify a block device, retrieve a look-up table specific to the block device addressed by the incoming data read request, and identify a physical storage location in the flash memory dependent on the look-up table.
Claim 15	

15a	The storage device of claim 1 wherein: the storage device further comprises circuitry to track additional metadata for the addressed segment, where said additional metadata indicates an extent to which memory comprising at least one of the physical erase units mapped to the addressed segment can continue to receive additional write data without being erased; and
15b	the storage device further comprises circuitry to transmit information to a host dependent on said tracked additional metadata.
Claim 18	
18	The storage device of claim 1 wherein the storage device is to store information identifying a number of structures in the flash memory and is to transmit the information identifying the number of structures to a host, wherein the host is to issue at least one request to the storage device in which the at least one request is dependent on the information transmitted to the host identifying the number of structures, and wherein the storage device comprises circuitry to fulfill the at least one request by issuing commands to the flash memory that each operate on a quantum of flash memory which is dependent on the information transmitted to the host identifying the number of structures.
Claim 19	
19	The storage device of claim 18 wherein the information transmitted to the host identifying the number of structures identifies an amount of flash memory that is to be erased as a unit.
Claim 20	
20	The storage device of claim 1 wherein the addressed segment corresponds to an integer number of physical erase units for which a failure condition has not been detected, and the third address portion specifies a location within a sequential range of locations that correspond to the integer number of physical erase units.
Claim 21	

21	The storage device of claim 1 wherein the storage device further comprises circuitry to compare the metadata to at least one threshold and, responsive to detection that specific metadata has satisfied the at least one threshold, to transmit information to a host wherein such transmitted information conveys address information for data stored in a physical storage location associated with the specific metadata prior to the copy of the data to the second one of the physical erase units.
Claim 26	
26a	The storage device of claim 1 wherein: the flash memory comprises multiple planes; and
26b	the storage device is to, for each one of a set of the incoming data access requests having sequential addressing, write data accompanying the set of incoming data access requests concurrently into respective ones of the multiple planes.
Claim 27	
27	The storage device of claim 1 wherein the division operation is applied to subdivide the third address portion into respective first and second bit fields.
Claim 28	
28a	The storage device of claim 1 wherein: for each of the incoming data write requests, the circuitry is to write data into a specific physical storage location within the identified one of the physical erase units and is to identify an address value corresponding to the specific physical storage location;
28b	the storage device further comprises circuitry to transmit the address value to a host;
28c	the storage device is also to receive incoming data read requests, each accompanied by an associated read address; and
28d	for each of the incoming data read requests which is directed to the data storage at the specific physical storage location, the associated read address is dependent on the address value which was transmitted to the host.
Claim 29	
29pre	A storage device comprising:
29a	flash memory having physical erase units; and

29b	circuitry to, for address information provided in association with each of incoming data write requests:
29b[i]	derive from the incoming address information a first address portion, a second address portion and a third address portion,
29b[ii]	identify an addressed block device from the first address portion,
29b[iii]	identify an addressed segment within the addressed block device from the second address portion, and
29b[iv]	identify an addressed storage location within the addressed segment from the third address portion,
29c	wherein the circuitry is to subdivide the address information into bit fields, is to identify from a first one of the bit fields a logical erase unit within the addressed segment, and is to identify the addressed storage location dependent on the second one of the bit fields, and
29d	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and
29e	program associated data into the identified one of the physical erase units at a location selected dependent on the addressed storage location;
29f	wherein the storage device also comprises circuitry to:
29f[i]	maintain metadata for each physical erase unit which contains valid data, the metadata representing a time since data was programmed in the respective erase unit,
29f[ii]	copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and, in association with said copy of the data, generate the metadata for the second one of the physical erase units so as to correspond to the copy of the data from the first one of the physical erase units to the second one of the physical erase units; and
29f[iii]	remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units.
Claim 30	
30pre	A storage device comprising:

30a	flash memory, the flash memory to have one or more block devices at a hierarchically-superior level, segments within each of the block devices at a hierarchically-intermediate level, and addressable storage locations within each of the segments at a hierarchically-inferior level, each segment including a mutually exclusive set of one or more logical erase units that each map to a physical erase unit, each segment being associated with only one of the block devices, each logical erase unit being associated with only one of the segments; and
30b	circuitry to, for each of incoming data write requests:
30b[i]	derive, from incoming address information, a first address portion, a second address portion and a third address portion,
30b[ii]	identify an addressed one of the one or more block devices from the first address portion,
30b[iii]	identify an addressed one of the segments within the addressed block device from the second address portion, and
30b[iv]	identify an addressed storage location within the addressed one of the segments from the third address portion,
30c	wherein the circuitry is to use a division operation on at least one of the incoming address information or information derived from the incoming address information, to identify a logical erase unit within the addressed one of the segments, and
30d	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and
30e	program associated data into the identified one of the physical erase units;
30f	wherein the storage device also comprises circuitry to:
30f[i]	maintain metadata for each physical erase unit which contains data, the metadata representing a time since data was programmed in the respective erase unit,
30f[ii]	copy data from a first one of the physical erase units for which the maintained metadata meets at least one criterion to a second one of the physical erase units, and, in association with said copy of the data, generate the metadata for the second one of the physical erase units so as to correspond to the copy of the data; and

30f[iii]	remap a correspondence of a logical erase unit which corresponds to the first one of the physical erase units to instead correspond to the second one of the physical erase units; and
30[g]	wherein the storage device is to generate the metadata for the second one of the physical erase units so as to correspond to a time that the data was copied from the first one of the physical erase units to the second one of the physical erase units.