

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

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

v.

RADIAN MEMORY SYSTEMS, LLC,
Patent Owner.

Case IPR2025-01376

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

**PETITION FOR *INTER PARTES* REVIEW
UNDER 35 U.S.C. §§ 311-319 AND 37 C.F.R. §§ 42.100 *et seq.***

TABLE OF CONTENTS

	PAGE
I. INTRODUCTION	1
II. STANDING	1
III. TECHNICAL BACKGROUND	1
IV. OVERVIEW OF THE '995 PATENT	2
A. Priority	2
B. Prosecution History	2
C. Disclosure	3
V. GROUNDS'	4
VI. LEVEL OF ORDINARY SKILL	4
VII. CLAIM CONSTRUCTION	5
VIII. GROUND I: CLAIMS 1-5, 7-11, 13-15, 18-20, 26-30 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, SPC-4, AND SINCLAIR-367	5
A. Overview of the Prior Art.....	5
1. Reiter (EX1004).....	5
2. ZBC (EX1006), SBC-4 (EX1007), and SPC-4 (EX1008)	6
3. Sinclair-367 (EX1009).....	10
B. Motivation to Combine	13
C. Claim-by-Claim Analysis.....	17
1. Claim 1	17
(a) 1[pre]: <i>A storage device comprising:</i>	17
(b) 1[a]: <i>flash memory having physical erase units;</i> <i>and</i>	18
(c) 1[b]: <i>circuitry to, for each of incoming data write</i> <i>requests:</i>	19
(d) 1[b][i]: <i>derive, from incoming address</i> <i>information, a first address portion, a second</i> <i>address portion and a third address portion,</i>	21

(e)	1[b][ii]: <i>identify an addressed block device from the first address portion,</i>	27
(f)	1[b][iii]: <i>identify an addressed segment within the addressed block device from the second address portion, and.....</i>	27
(g)	1[b][iv]: <i>identify an addressed storage location within the addressed segment from the third address portion,</i>	28
(h)	1[c]: <i>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</i>	31
(i)	1[d]: <i>wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and</i>	35
(j)	1[e]: <i>program associated data into the identified one of the physical erase units;</i>	38
(k)	1[f]: <i>wherein the storage device also comprises circuitry to:</i>	38
(l)	1[f][i]: <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,</i>	38
(m)	1[f][ii]: <i>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</i>	39
(n)	1[f][iii]: <i>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.</i>	40
2.	Claim 2	41
(a)	2[a]: <i>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</i>	

	<i>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</i>	41
(b)	<i>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.</i>	42
3.	<i>Claim 3</i>	43
(a)	<i>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.</i>	43
4.	<i>Claim 4</i>	44
(a)	<i>4[a]: The storage device of claim 3 wherein: the flash memory comprises multiple planes; and</i>	44
(b)	<i>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.</i>	45
5.	<i>Claim 5</i>	46
(a)	<i>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</i>	46
(b)	<i>5[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.</i>	46

6.	Claim 7	47
	(a) <i>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.....</i>	47
7.	Claim 8	50
	(a) 8[a]: <i>The storage device of claim 1 wherein: each said circuitry comprises at least one processor; and</i>	50
	(b) 8[b]: <i>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.....</i>	50
8.	Claim 9	51
	(a) <i>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.....</i>	51
9.	Claim 10	51
	(a) 10[a]: <i>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.....</i>	51
	(b) 10[b]: <i>the storage device comprises circuitry to erase, in association with execution of the maintenance request, one or more of the physical</i>	

	<i>erase units which are mapped to the segment addressed by the maintenance request.</i>	52
10.	Claim 11	54
	(a) <i>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.</i>	54
11.	Claim 13	54
	(a) <i>13[a]: The storage device of claim 11 wherein: the flash memory comprises multiple planes; and</i>	54
	(b) <i>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.</i>	54
12.	Claim 14	54
	(a) <i>14[a]: The storage device of claim 1 wherein: the flash memory is to have block devices, including the addressed block device; and</i>	54
	(b) <i>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.</i>	55
13.	Claim 15	57
	(a) <i>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</i>	

	<i>segment can continue to receive additional write data without being erased; and</i>	<i>57</i>
	(b) 15[b]: <i>the storage device further comprises circuitry to transmit information to a host dependent on said tracked additional metadata.....</i>	<i>59</i>
14.	Claim 18	<i>61</i>
	(a) <i>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.</i>	<i>61</i>
15.	Claim 19	<i>63</i>
	(a) <i>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.</i>	<i>63</i>
16.	Claim 20	<i>65</i>
	(a) <i>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.....</i>	<i>65</i>
17.	Claim 26	<i>66</i>
	(a) 26[a]: <i>The storage device of claim 1 wherein: the flash memory comprises multiple planes; and</i>	<i>66</i>

	(b)	<i>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.</i>	<i>66</i>
18.	Claim 27		<i>68</i>
	(a)	<i>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.</i>	<i>68</i>
19.	Claim 28		<i>69</i>
	(a)	<i>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;</i>	<i>69</i>
	(b)	<i>28[b]: the storage device further comprises circuitry to transmit the address value to a host;</i>	<i>70</i>
	(c)	<i>28[c]: the storage device is also to receive incoming data read requests, each accompanied by an associated read address; and</i>	<i>70</i>
	(d)	<i>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.</i>	<i>71</i>
20.	Claim 29		<i>72</i>
	(a)	<i>29[pre]: A storage device comprising:</i>	<i>72</i>
	(b)	<i>29[a]: flash memory having physical erase units; and</i>	<i>72</i>
	(c)	<i>29[b]: circuitry to, for address information provided in association with each of incoming data write requests:</i>	<i>72</i>

- (d) 29[b][i]: *derive from the incoming address information a first address portion, a second address portion and a third address portion,72*
- (e) 29[b][ii]: *identify an addressed block device from the first address portion,73*
- (f) 29[b][iii]: *identify an addressed segment within the addressed block device from the second address portion, and73*
- (g) 29[b][iv]: *identify an addressed storage location within the addressed segment from the third address portion,73*
- (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, and73*
- (i) 29[d]: *wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and73*
- (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;73*
- (k) 29[f]: *wherein the storage device also comprises circuitry to:73*
- (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,74*
- (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*

	<i>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</i>	<i>74</i>
(n)	<i>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.</i>	<i>75</i>
21.	<i>Claim 30</i>	<i>75</i>
(a)	<i>30[pre]: A storage device comprising:</i>	<i>75</i>
(b)	<i>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</i>	<i>75</i>
(c)	<i>30[b]: circuitry to, for each of incoming data write requests:</i>	<i>75</i>
(d)	<i>30[b][i]: derive, from incoming address information, a first address portion, a second address portion and a third address portion,</i>	<i>75</i>
(e)	<i>30[b][ii]: identify an addressed one of the one or more block devices from the first address portion,</i>	<i>76</i>
(f)	<i>30[b][iii]: identify an addressed one of the segments within the addressed block device from the second address portion, and</i>	<i>76</i>
(g)	<i>30[b][iv]: identify an addressed storage location within the addressed one of the segments from the third address portion,</i>	<i>76</i>
(h)	<i>30[c]: wherein the circuitry is to subdivide the address information into bit fields, is to identify</i>	

	<i>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</i>	<i>76</i>
(i)	<i>30[d]: wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and</i>	<i>76</i>
(j)	<i>30[e]: program associated data into the identified one of the physical erase units at a location selected dependent on the addressed storage location;</i>	<i>76</i>
(k)	<i>30[f]: wherein the storage device also comprises circuitry to:</i>	<i>76</i>
(l)	<i>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,</i>	<i>77</i>
(m)	<i>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</i>	<i>77</i>
(n)	<i>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</i>	<i>77</i>
(o)	<i>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.....</i>	<i>77</i>
IX.	GROUND II: CLAIMS 6, 21 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SINCLAIR-367 AND YAMADA	77

A.	Overview of the Prior Art.....	77
1.	Reiter (EX1004).....	77
2.	ZBC (EX1006), SBC-4 (EX1007), and SPC-4 (EX1008)	77
3.	Sinclair-367 (EX1009).....	78
4.	Yamada (EX1010)	78
B.	Motivation to Combine	79
C.	Claim-by-Claim Analysis.....	80
1.	Claim 6	80
(a)	6[a]: <i>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</i>	80
(b)	6[b]: <i>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.</i>	83
2.	Claim 21	83
(a)	<i>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.</i>	83
X.	MANDATORY NOTICES AND FEES	84
A.	Real Party-In-Interest	84
B.	Related Matters.....	84

C.	Counsel and Service Information.....	84
D.	Payment of Fees	85
XI.	CONCLUSION.....	85
	CERTIFICATE OF WORD COUNT.....	86
	CERTIFICATE OF SERVICE	87

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)

Ex.	Description
1106	“File Access Monitor”, T10 (archived May 19, 2014), 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” (2009)
1202	White et al., “Performance Comparison of IDE and SCSI” (2001)
1203	L. Columbus, “Exploring the World of SCSI” (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
1208	Patterson et al., “A Case for Redundant Arrays of Inexpensive Disks (RAID)” (1998)
1209	Chen et al., “RAID: High-Performance, Reliable Secondary Storage” (Jun. 1994)
1210	U.S. Publication No. 2009/0271562

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 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.

I. INTRODUCTION

Samsung Electronics Co., Ltd. and Samsung Electronics America, Inc., (collectively, “Samsung” or “Petitioner”) respectfully request *inter partes* review of claims 1-11, 13-15, 18-21, 26-30 of U.S. Patent No. 11,307,995 (the “’995 Patent”). A reasonable likelihood exists that Petitioner will prevail on at least one challenged claim.

II. STANDING

Petitioner certifies under 37 C.F.R. §42.104(a) that the ’995 Patent is available for *inter partes* review and that Petitioner is not barred or estopped from requesting an *inter partes* review on the grounds set forth herein.

III. TECHNICAL BACKGROUND

Flash memory is organized into discrete units called erase units or blocks. EX1001, 4:10-35. Blocks are comprised of pages. EX1001, 4:10-35. Pages represent the smallest unit that may be written. EX1001, 4:10-35. To write the pages of a block with new data, the block must first be erased. EX1001, 4:10-35.

To increase the efficiency of flash memory, SSDs may employ a larger unit of memory. For example, SSDs may be organized into groupings of blocks spanning multiple planes operated as a single unit, termed, *e.g.*, “metablocks,” EX1009, ¶[0012].

To increase the longevity of flash memory, SSDs perform management operations including wear-leveling and other maintenance where data may be

copied, moved, or erased to prolong data storage life. EX1004, 4:4-37. To identify data for maintenance, metadata is tracked describing the history of the units of memory. EX1004, 4:4-37.

IV. OVERVIEW OF THE '995 PATENT

A. Priority

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

The prior art references relied upon in this Petition predate September 9, 2014, the earliest possible critical date.

B. Prosecution History

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

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. After the Applicant amended the claims on February 23, 2022, the Examiner issued a final rejection on March 2, 2022 that allowed Claims 1-7 and 9-30, but maintained 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. Disclosure

The '995 Patent 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.

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 actual physical sub-addresses 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.* The '995 Patent refers to this scheme as “hierarchical address virtualization.” EX1001, Abstract.

“FIG. 8 illustrates a conceptual implementation of an address generation module that may be used to extract (or recover) sub-address values from an inbound LBA.” EX1001, 18:23-29, Fig. 8. The address generation module in Figure 8 uses division operations to obtain the sub-address values. *Id.*, 19:29-41. The virtual sub-addresses are supplied to lookup/translation modules to generate physical sub-

address counterparts. *Id.* Those physical sub-addresses detail the actual location in flash memory.

V. GROUNDS^{1,2}

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

A person of ordinary skill in the art (“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. EX1003, ¶48. More education can supplement practical experience and vice versa. *Id.* Petitioner’s expert exceeded this skill level as of the priority date. *Id.* ¶¶ 1-11, 48.

¹ None of these references relied upon by this Petition were before the Examiner, much less included in a rejection, cited during prosecution.

² Petitioners are not presently aware of any secondary considerations of non-obviousness applicable to the ’995 Patent. In the co-pending litigation, Radian has objected to identifying secondary considerations tethered to specific patents and has agreed to only “generally address secondary considerations” without any particularity or showing of nexus. Radian has also marked its response as confidential such that Petitioner cannot reproduce its response here.

VII. CLAIM CONSTRUCTION

For the purposes of this proceeding, Petitioner contends that claim terms be given their plain and ordinary meaning under *Philips*.

VIII. GROUND I: CLAIMS 1-5, 7-11, 13-15, 18-20, 26-30 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, SPC-4, AND SINCLAIR-367

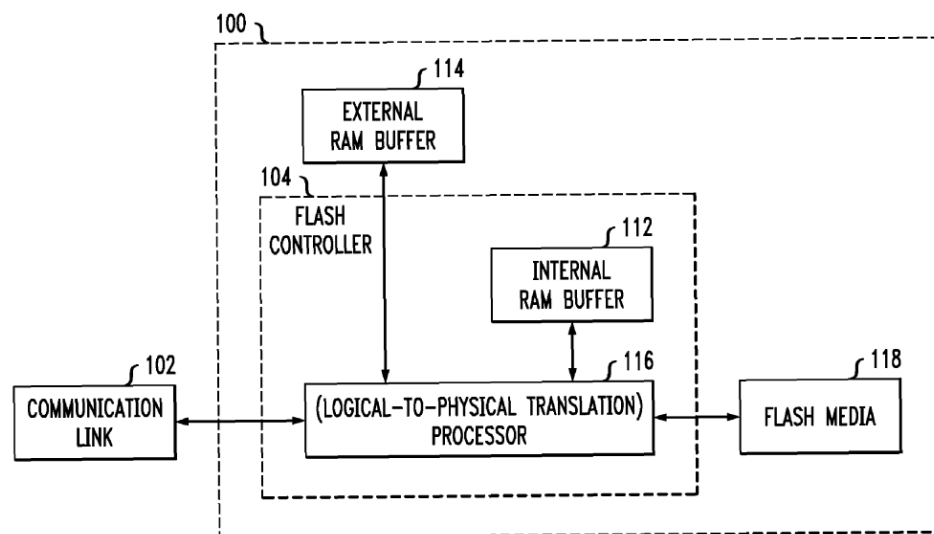
A. Overview of the Prior Art

1. Reiter (EX1004)

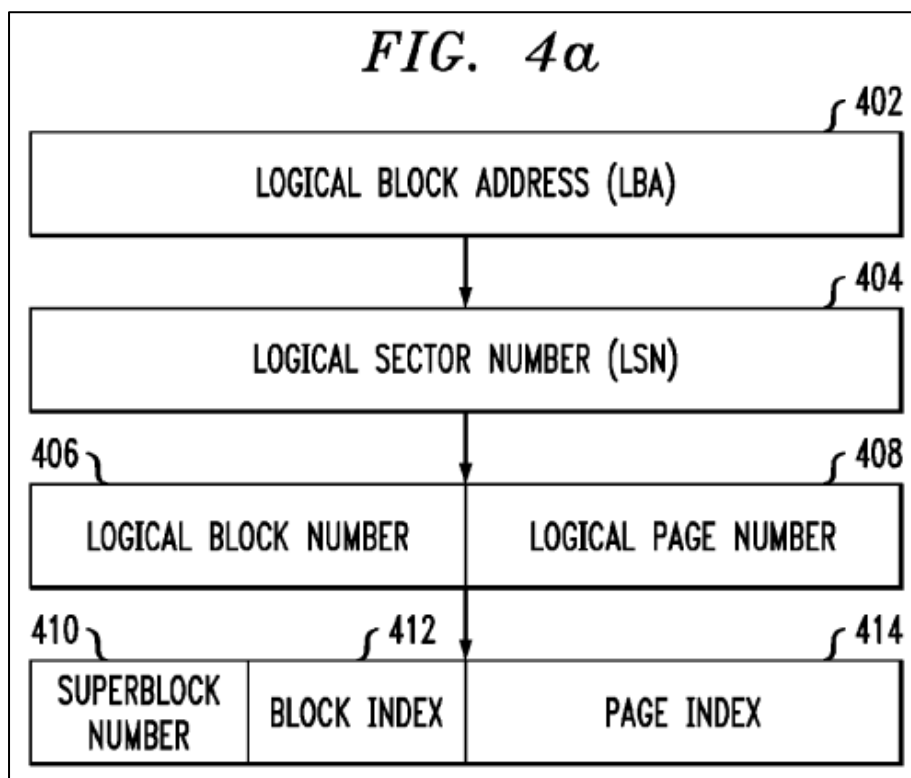
U.S. Patent 8,301,861 (“Reiter”) was filed on April 29, 2010, was published on March 24, 2011, and issued on October 30, 2012. It is prior art under 35 U.S.C. §102(a)(1).

Reiter discloses a flash system. EX1004, 6:63-66. The storage system uses a flash controller to receive and execute a host’s read/write commands sent over a communication link. EX1004, 8:1-6, FIG. 1:

FIG. 1



The controller performs logical-to-physical translation on the accompanying logical address and executes those commands on flash media at the resulting physical address. EX1004, 7:31-35, 8:1-10. Reiter’s addressing scheme includes a logical block address, a logical sector number, a logical block number, a logical page number, and a superblock number, block index, and page index. EX1004, 3:55-60, FIG. 4a:



2. ZBC (EX1006), SBC-4 (EX1007), and SPC-4 (EX1008)

ZBC, SBC-4, and SPC-4 (collectively, “T10 Standards”) are printed publications. As set forth in Rich Deglin’s declaration, T10 Standards qualify because of publication on the website of T10, an INCITS Technical Committee responsible for SCSI architecture standards, as of May 15, 2014, May 20, 2014, and

May 17, 2014, respectively. EX1100, ¶¶2-10; EX1103; *see also* EX1109 (depicting T10’s website). T10 Standards are working drafts that were made available to both T10 members and guests (*i.e.*, the public at large) for download on T10’s website, as shown in the archived versions. EX1100, ¶¶11; *see also Valve Corporation v. Ironburg Inventions Ltd.*, 8 F.4th 1364, 1374 (Fed. Cir. Aug. 17, 2021) (establishing public availability with archive.org).

The website’s well-known nature is confirmed by Dr. Litz. EX1003, ¶65. Dr. Litz was aware of T10 and the SCSI standards published by T10, and that ZBC Standards would have been readily available upon publication on T10’s website. *Id.* T10 “was well known to the community interested in the subject matter of” ZBC Standards, and a POSITA “interested in” storage standards “would have been independently aware of [T10] as a prominent forum for discussing such technologies.” *See id.*; *Voter Verified, Inc. v. Premier Election Sols., Inc.*, 698 F.3d 1374, 1380 (Fed. Cir. 2012).

Moreover, a POSITA could have reasonably located ZBC Standards on T10’s website. Via a “Drafts” hyperlink at the top of the website (*see, e.g.*, EX1109), the website has a “T10 Working Drafts” webpage linking a limited number of standards and working drafts all related to accessing/managing storage devices. EX1103; *Roku, Inc. v. Universal Electronics, Inc.*, 2025 WL 1692663, at *2 (Fed. Cir. Jun. 17, 2025) (holding that a POSITA could reasonably locate a user manual on a

webpage amongst several hundred manuals because a POSITA could reasonably find the webpage in the website, and because the webpage's manuals were the "same subject matter as the invention"). The webpage also categorizes those documents into sections, and provides a table of contents with descriptive section names and hyperlinks to each section, including "Block Commands (ZBC)," "Block Commands (SBC)," and "Primary Commands (SPC)" leading to ZBC, SPC-4, and SPC-4, respectively. EX1103; *see Roku*, 2025 WL 1692663 at *2 (finding public availability for a user manual amongst other manuals on a website because the website's manuals "were ... sorted by subject matter"). Thus, an interested POSITA would have been aware of the availability of, and could easily locate the ZBC Standards on T10's website.

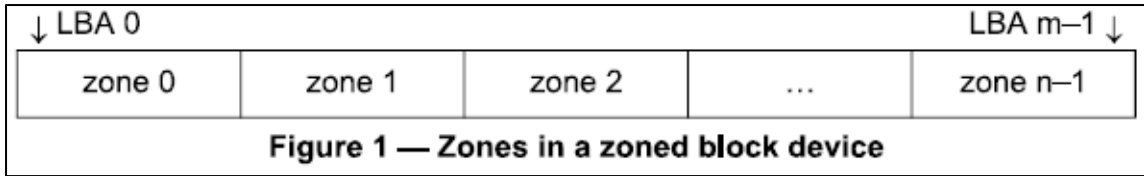
Even if ZBC Standards' availability was limited to T10 members, that is no bar to public availability. The Board regularly finds documents published by standards bodies, including working drafts like ZBC Standards, are publicly available. *See, e.g., ZTE Corp v. IPR licensing, Inc.* (IPR2014-00525) (crediting expert that GPRS Standards "were available to an interested member of the public as of the date listed on the top of the specification."); *M&K Holdings, Inc. v. Samsung Electronics Co., Ltd.*, 985 F.3d 1376, 1380–83 (Fed. Cir. 2021) (finding that three papers associated with an industry standard task force (JCT-VC), one being a working draft document, which were all posted on the organization's public

website, were publicly accessible as prior art). In *Ericsson Inc. v. Intellectual Ventures I LLC*, IPR2014-00527, Paper No. 1, Petition at 7 (PTAB March 21, 2014), the Board found a paywalled IEEE article as publicly available, and that an IEEE copyright line on the first page of the reference sufficiently evidenced publication date and public accessibility. Here, ZBC Standards bear T10 copyright dates and/or otherwise indicate “public[ation].” EX1006, iv; EX1007, iv; EX1008, iv.

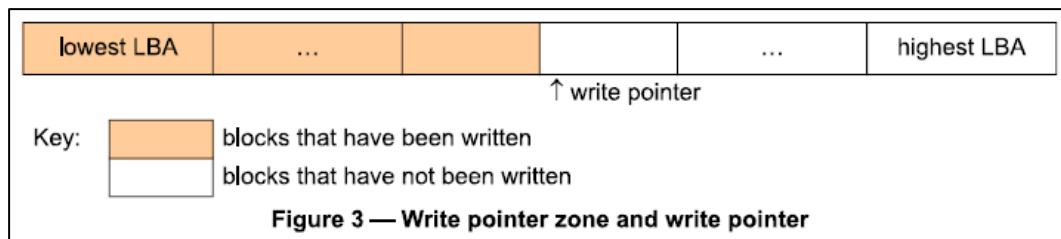
Moreover, ZBC Standards are authentic. EX1100, ¶¶1-17. Mr. Deglin personally received notice of publications and contemporaneously downloaded ZBC Standards, and confirmed that the metadata matches the metadata listed on the T10 website as of the publication date. EX1100, ¶¶12-16. ZBC Standards used in this Petition are the same as those downloaded by Mr. Deglin.

SCSI is a commonly used standard for storage, such as hard disk drives. EX1003, ¶65. ZBC is an SCSI command set for a zoned block device model. EX1006, 1. The zoned block device model also accepts commands from core SCSI specifications such as SBC-4 and SPC-4. EX1006, x, 12-13, 22, Figure 0.

SCSI creates logical block devices, including logical units, volumes, and/or partitions (collectively, “logical units”), on physical storage devices. EX1008, 14-16, 61. Each logical unit has an LBA range from zero to its highest LBA. EX1008, 61. When a logical unit is set as a zoned block device, the LBA range is divided into zones. EX1006, 1, 13, Figure 1:



ZBC allows for different types of zoned block devices including “host aware zoned block device[s]” and “host managed zoned block device.” EX1006, 4, 13. ZBC also supports different types of zones including a “sequential write preferred zone” and a “sequential write required zone.” EX1006, 14. These zones have write pointers, which generally track the next available LBA in the zone. EX1006, 14, 16-17, Figure 3:



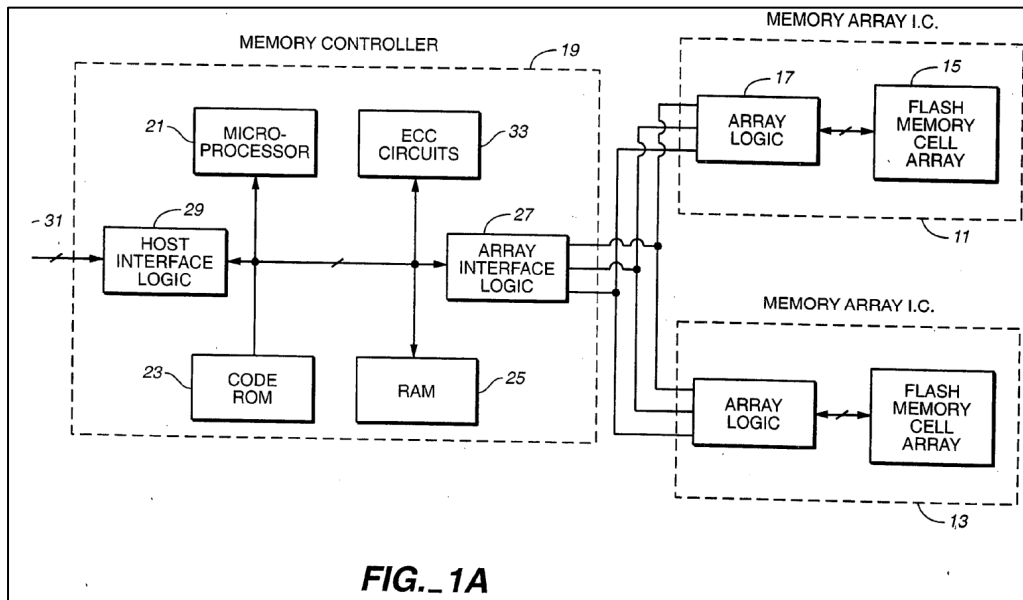
The write pointer increments with each write command until the zone is full or a reset write pointer command is applied. EX1006, Table 15. A reset write pointer command resets the write pointer back to the lowest LBA in the zone, and resets the data in the zone to the manufacturer’s default initialization pattern. EX1006, 3, 15-18.

3. Sinclair-367 (EX1009)

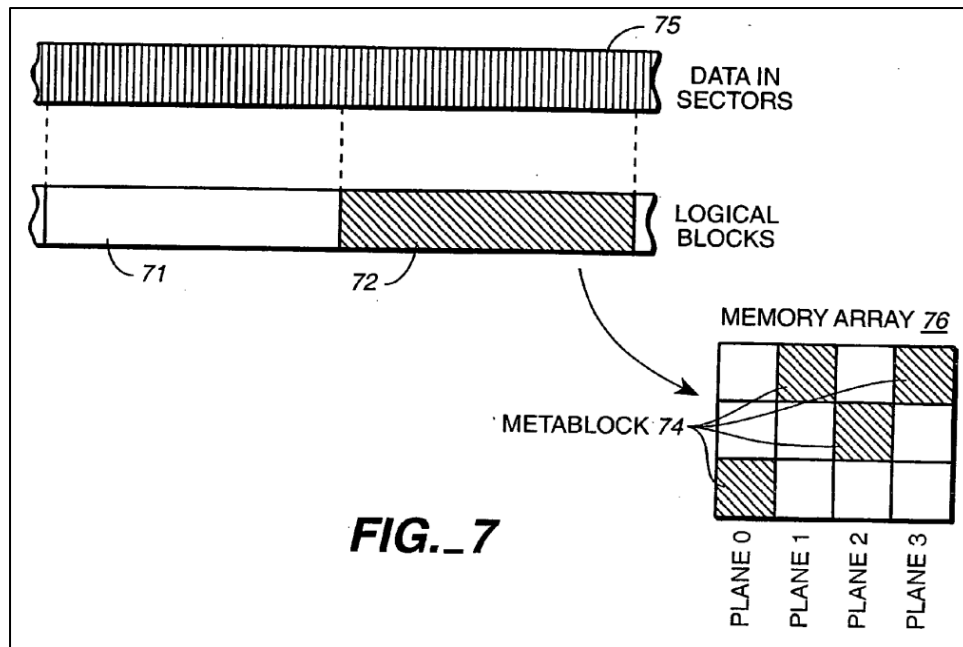
U.S. Patent Publication No. 2005/0144367 (“Sinclair-367”) was filed on December 16, 2004 and published on June 30, 2005. It is prior art under AIA 35

U.S.C. § 102.

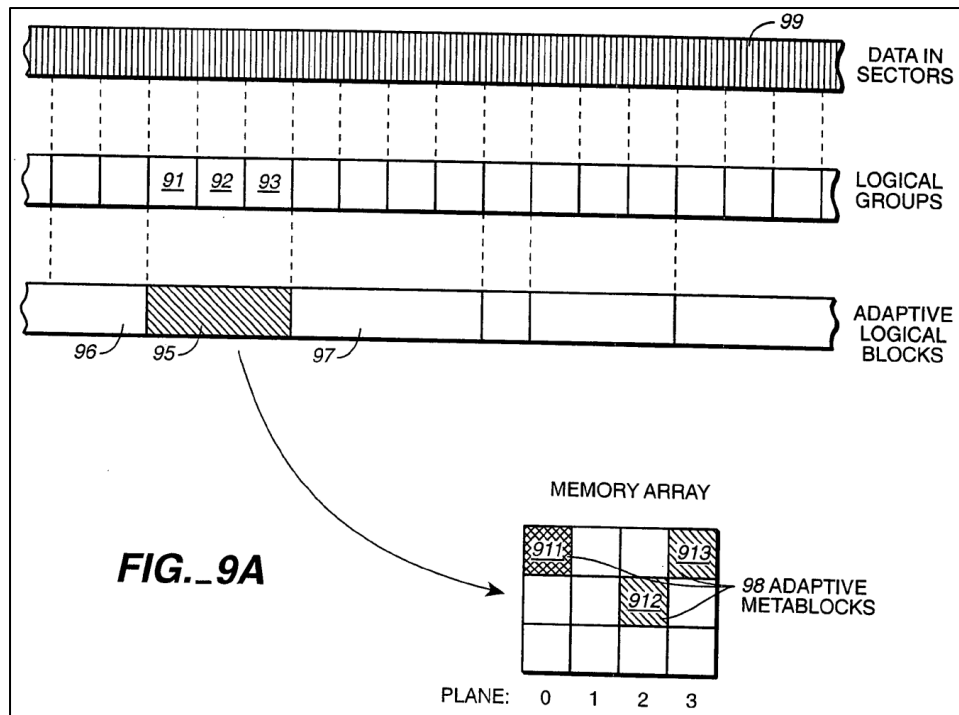
Sinclair-367 discloses a non-volatile memory array (*i.e.*, flash memory) via adaptive metablocks. EX1009, Abstract, FIG. 1A:



Metablocks are a logical grouping of erase units across planes, where reading, writing, and erasing occurs simultaneously across the metablock. EX1009, ¶[0117], FIG. 7:



Adaptive metablocks are also a logical grouping of erase units, but do not have a fixed size. EX1009, ¶[0117]. Adaptive metablocks might not have one erase unit in each plane, sometimes doubling up erase units in one plane, thereby limiting the extent of simultaneous operation across the blocks. EX1009, ¶¶[0117], [0119]-[0120], FIG. 9A:



B. Motivation to Combine

A POSITA would have been motivated to use ZBC, alongside SBC-4 and SPC-4, as the command set governing requests sent to Reiter's flash system. EX1003, ¶¶70-75. 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. EX1003, ¶¶70, 76-78.

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. EX1003, ¶71. ZBC is a "command set extension" of SCSI that builds on the core command sets defined in SBC-4 and SPC-4. EX1006, x, 1, 12-13, 22, Figure 0. ZBC expressly

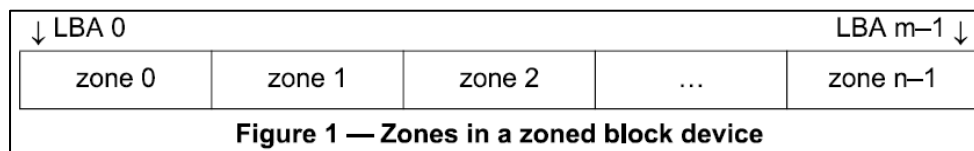
discloses being “implemented in conjunction with ... SPC-4,” and SPC-4 is listed as “shared for all device types.” EX1006, 1, Figure 0. ZBC also repeatedly relies on SBC-4. EX1006, 3-4, 12, 22 (“[C]ommands for host aware zoned block devices are defined in SBC-4.”).

Second, a POSITA would further be motivated to apply ZBC’s teachings (including SBC-4 and SPC-4) to Reiter’s flash system. EX1003, ¶72. Reiter does not require a specific command set, but expressly discloses support for the widely used SCSI protocols. EX1004, 7:9-18. Reiter also discloses “receiv[ing] ... requests ... such as read or write operations from ... external devices,” which “generally include a[] ... [LBA] where data should be read or written.” EX1004, 8:1-6. A POSITA would thus be motivated to include support for commands such as those in ZBC to enable specific read and write operations based on Reiter’s memory structure. EX1003, ¶72.

A POSITA would particularly be motivated to implement ZBC commands in Reiter’s system because SCSI protocols and commands such as those disclosed in ZBC are widely used, thereby improving interoperability. EX1003, ¶73.

In addition, commands based on ZBC’s zoned block device model (EX1006, 1) would enable more effective utilization of Reiter’s memory organization scheme. EX1003, ¶73. ZBC is designed for zoned block devices, where address space

comprising LBAs is “divided into ranges called zones.” EX1006, 13, 29, Table 16, Figure 1:



ZBC zones are analogous to Reiter’s superblocks, which also comprise “a logical collection of blocks representing a fixed range of LBAs.” EX1004, 9:11-13; EX1003, ¶74. Zones and superblocks both include metadata that indicate the next available memory location within the zone/superblock. 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 the next available memory location). And like ZBC, which uses LBAs to access different zones, Reiter’s superblocks are addressed through the received LBA. EX1004, 8:54-9:25, FIG. 4a; EX1006, 13.

A POSITA would have a reasonable expectation of success implementing Reiter’s flash system to support ZBC. EX1003, ¶75. SCSI commands and protocols were widely known and would have been familiar to a POSITA. *Id.* SCSI specifications were intended to be used across a wide range of storage devices, including flash memory. *Id.* A POSITA would have also understood that Reiter’s superblock-based memory organization is suitable for zone-based commands like

ZBC. *Id.* 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. *Id.*

Third, a POSITA would have also 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. EX1003, ¶¶76-78.

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]; EX1003, ¶77. 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. *Id.*

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. EX1003, ¶78. 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].

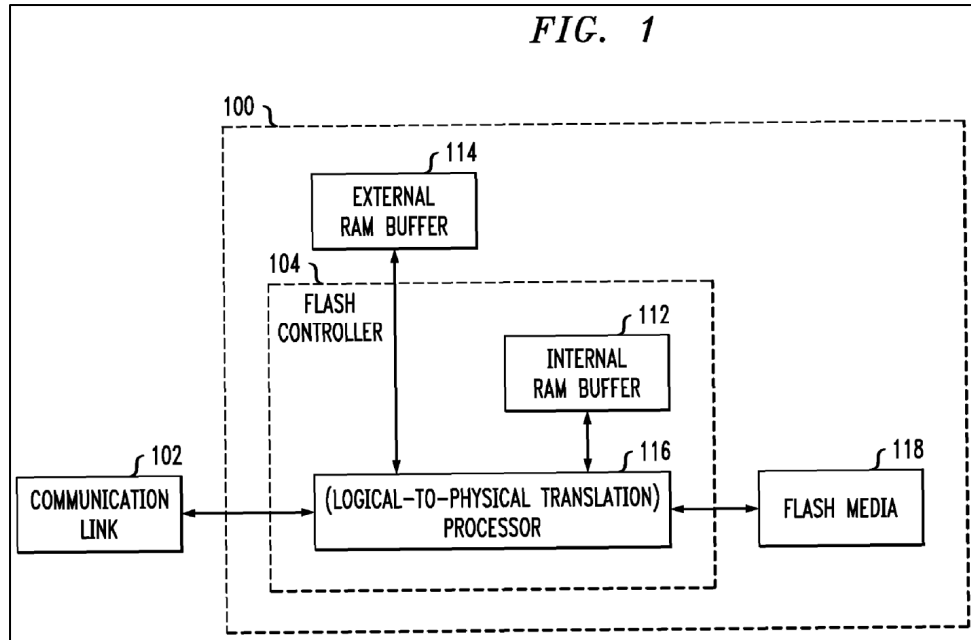
Reiter, ZBC, and Sinclair-367 are analogous art—relating to storage devices, including accessing blocks of memory. EX1003, ¶79.

C. Claim-by-Claim Analysis

1. Claim 1

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

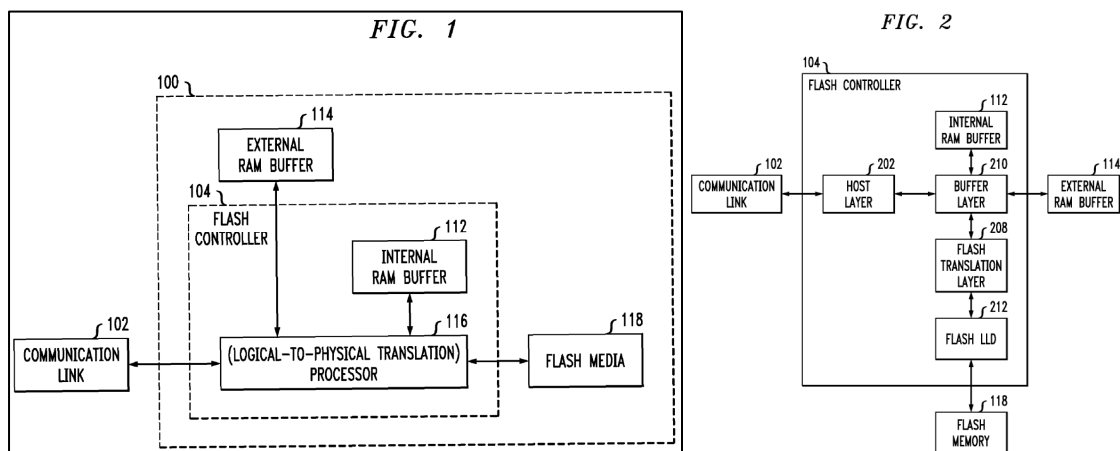
Reiter discloses a “flash ... system 100” (*storage device*) “implementing a logical-to-physical translation” scheme:



EX1004, 6:63-66, FIG. 1; EX1003, ¶80.

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

Reiter’s flash system 100 includes “flash media 118” or “flash memory 118” (*flash memory*). EX1004, 6:63-7:5, 7:56-61, FIGS. 1, 2:

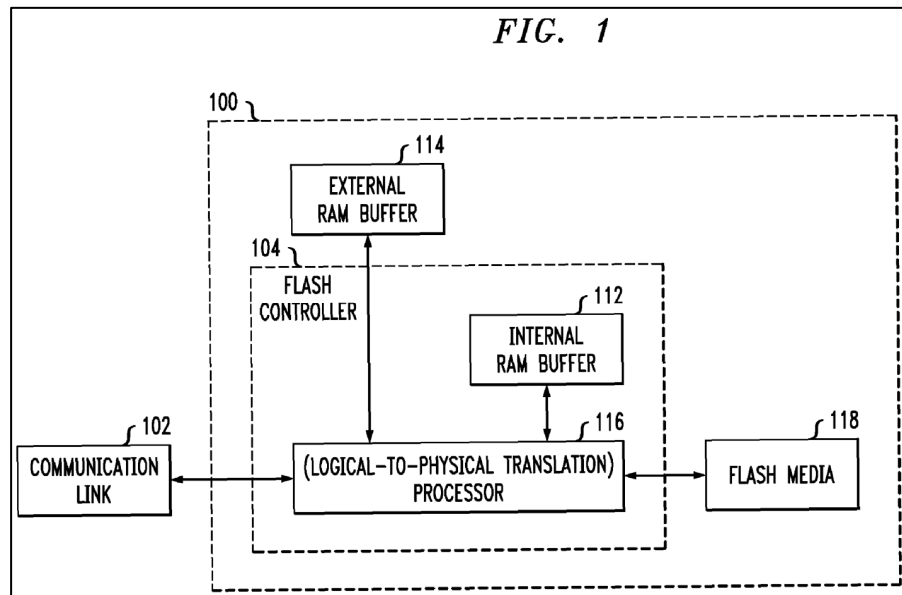


Reiter discloses that a “commonly employed type of flash memory technology is NAND flash memory.” EX1004, 1:40-45. “There are three basic operations for NAND devices: read, write, and erase.” EX1004, 1:51-52. Read and write

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*). EX1003, ¶¶81-82.

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

Reiter’s flash memory storage system 100 includes a “flash controller 104” (*circuitry*). EX1004, 6:63-7:2, FIG. 1 (below); EX1003, ¶¶83-84.



“[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. For example, Reiter discloses a “flash media write operation 630” that is “***executed*** by FTL 208” in flash controller 104. EX1004, 13:38-46, FIG. 2; EX1003, ¶¶84-85.

A POSITA would also be motivated to implement Reiter’s flash controller 104 to accept and execute ZBC commands. Section VIII.B. 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.”); EX1003, ¶86. A ZBC write command is shown below:

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)	SERVICE ACTION (000Bh)							
9		(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)	EXPECTED LOGICAL BLOCK APPLICATION TAG							
25		(LSB)							
26	(MSB)	LOGICAL BLOCK APPLICATION TAG MASK							
27									
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,*

ZBC write command includes an LBA (*incoming address information*)—also as disclosed with Reiter’s requests (EX1004, 8:1-6, 11:55-63, 13:3-46).

EX1007, Table 112:

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)	SERVICE ACTION (000Bh)							
9									
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)	EXPECTED LOGICAL BLOCK APPLICATION TAG							
25									
26	(MSB)	LOGICAL BLOCK APPLICATION TAG MASK							
27									
28	(MSB)	TRANSFER LENGTH							
...									
31		(LSB)							

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*). EX1003, ¶87.

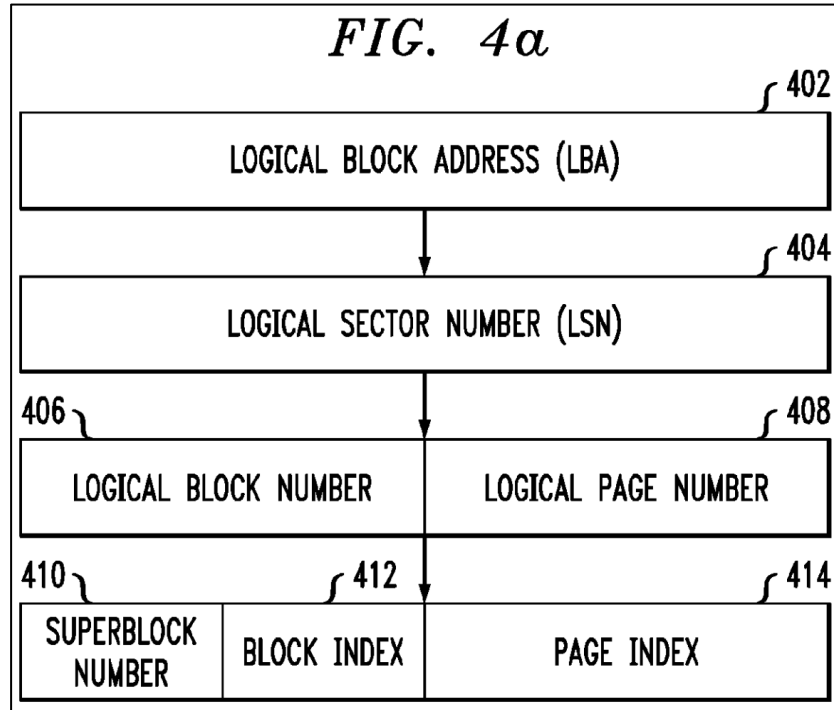
First Address Portion: A POSITA would be motivated to incorporate a logical unit number in ZBC commands, and motivated to modify flash controller 104 to decode (*derive*) that number (*first address portion*) from the command (*from the incoming address information*). EX1003, ¶88. ZBC write commands target a

“logical unit,” “volume,” or “partition.” EX1008, 14, 61; *see also* EX1006, x, 1, 4. An “objective of ... [ZBC]” is to allow “communicat[ion] with a logical unit that declares itself to be a zoned block device.” EX1006, 1. But logical units are logical structures, where multiple can exist on the same physical storage device. EX1007, 15, 327; EX1003, ¶88. Each logical unit “shall begin with block zero and be contiguous up to the last logical block of that logical unit.” EX1008, 61. With multiple logical units, and overlapping LBA spaces, an LBA by itself is not enough to identify a location on a physical storage device. EX1003, ¶88. A POSITA would look to incorporate something else into its commands to distinguish the same LBA to two different logical units. *Id.*

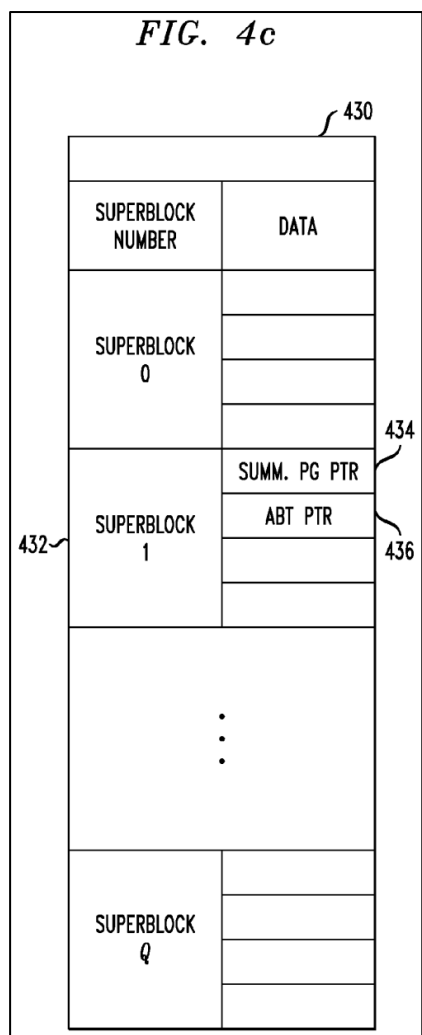
A POSITA would be motivated to incorporate the separate address found in SPC-4 into ZBC commands. Section VIII.B.1. SPC-4 recites that a “logical unit” is “[a]n **externally addressable** entity within a SCSI target device.” EX1008, 7, 14. A logical unit is addressed by a “logical unit number”—“[a]n encoded 64 bit identifier for a logical unit.” EX1008, 14. ZBC already mentions logical unit number. EX1006, 4. A POSITA would be motivated to more fully incorporate that SPC-4 address in ZBC commands. EX1003, ¶89.

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 ZBC write commands (*from the accompanying address information*). EX1003,

¶90. 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:

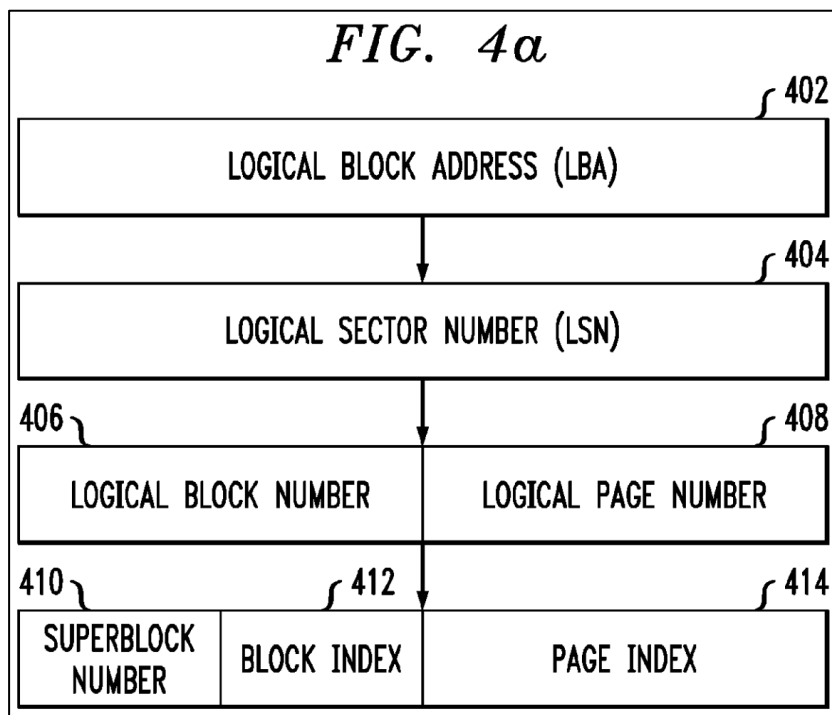


Translating an LBA results in a bitstring of superblock number 410, block index 412, and page index 414. EX1004, 8:66-9:25, FIG. 4a; EX1003, ¶¶90-91. Superblock number 410 is identified in the page global directory (PGD) 430 to obtain a “pointer 434 ... to the summary page associated with [the corresponding] Superblock.” EX1004, 10:13-19, FIG. 4c:



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*) from the ZBC write commands (*from the incoming address information*). EX1003, ¶92. A ZBC write command includes an LBA. EX1007, Table 112. Reiter 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:



“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; EX1003, ¶93. 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,*

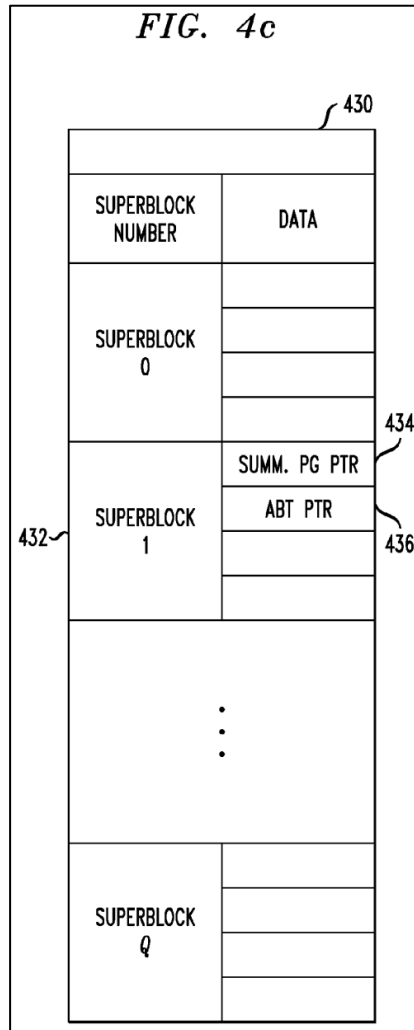
The number associated with a logical unit (*from the first address portion*) *identifies* the targeted logical unit, volume, or partition (*addressed block device*). Whether a logical unit number, volume, or partition is targeted, each structure includes an independent range of logical block addresses “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; EX1001, 7:54-63, FIG. 5; EX1003, ¶94.

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

Superblock number 410 (*from the second address portion*) *identif[ies]* the superblock, which a POSITA would be motivated to implement as a ZBC zone (**address segment**). Section VIII.C; EX1011, 20 (Patent Owner identifying an “addressed zone” as an “**address segment**” for purposes of infringement); EX1003, ¶¶95-96.

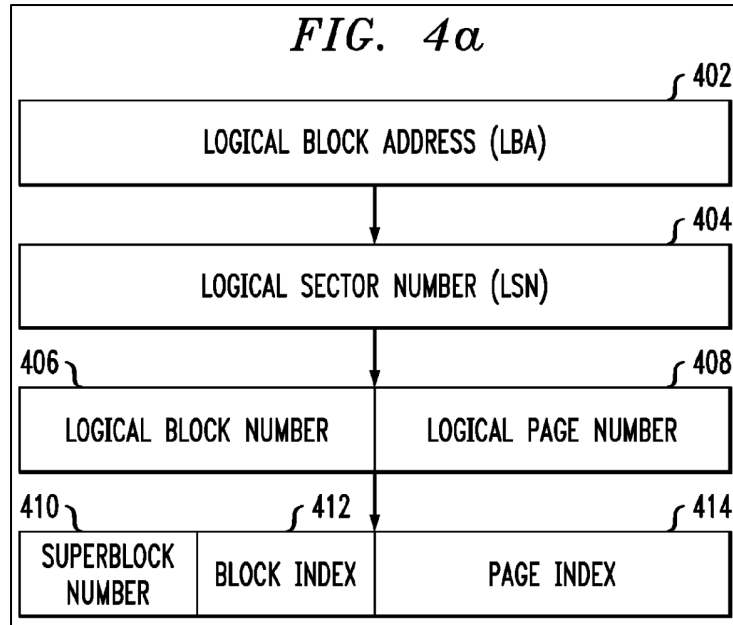
Reiter discloses that superblock number 410 indexes page global directory (PGD) 430 to obtain the “summary page associated with [the corresponding] Superblock.” EX1004, 10:13-19, FIG. 4c (reproduced below). Superblock number 410 is derived from LBA 402 (EX1004, FIG. 4a), which is from a ZBC command’s LBA. Because a ZBC LBA is within the LBA range specified by the targeted logical

unit (*see* Sections VIII.C.1.d-e; 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,*

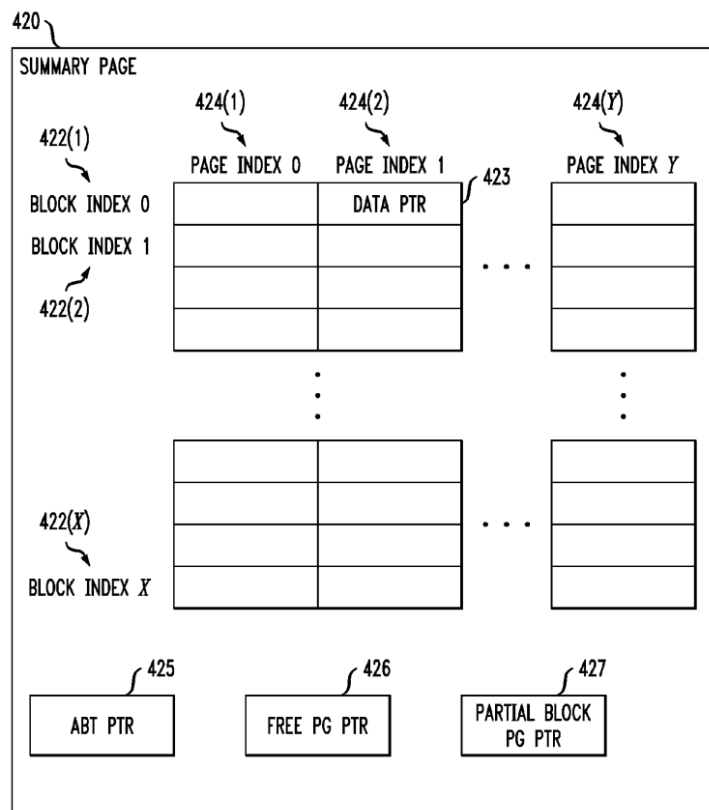
Flash controller 104 translates logical sector number 404, and alternatively the bitstring of logical block number 406 and logical page number 408, and alternatively logical page number 408 or page index 414 (*from the third address portion*) to block index 412 and/or page index 414. EX1004, 9:5-25, FIG. 4a:



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; EX1003, ¶¶97-98.

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



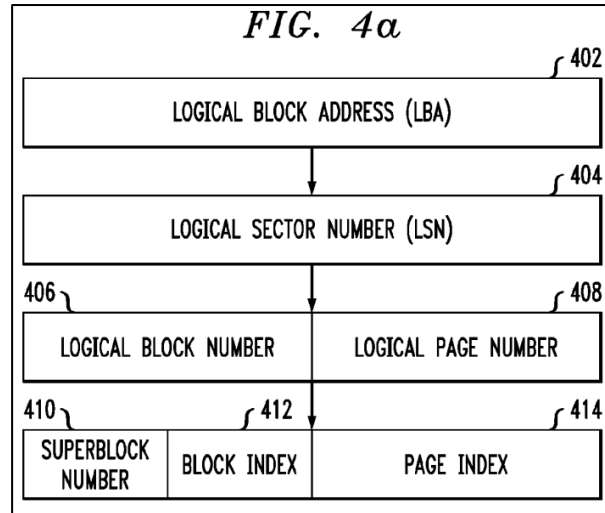
Because logical sector number 404, and alternatively bitstring of logical block number 404 and logical page number 406, and alternatively logical page number 406 and page index 414, are translated into block index 412 and/or page index 414, Reiter discloses *identifying an addressed storage location ... from the third address portion*. EX1003, ¶¶99-100.

Further, 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

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*). EX1003, ¶102. 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. *Id.* 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. *Id.*

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.” EX1004, 3:48-50, 9:5-20. LBA 402 is first translated to logical sector number 404:

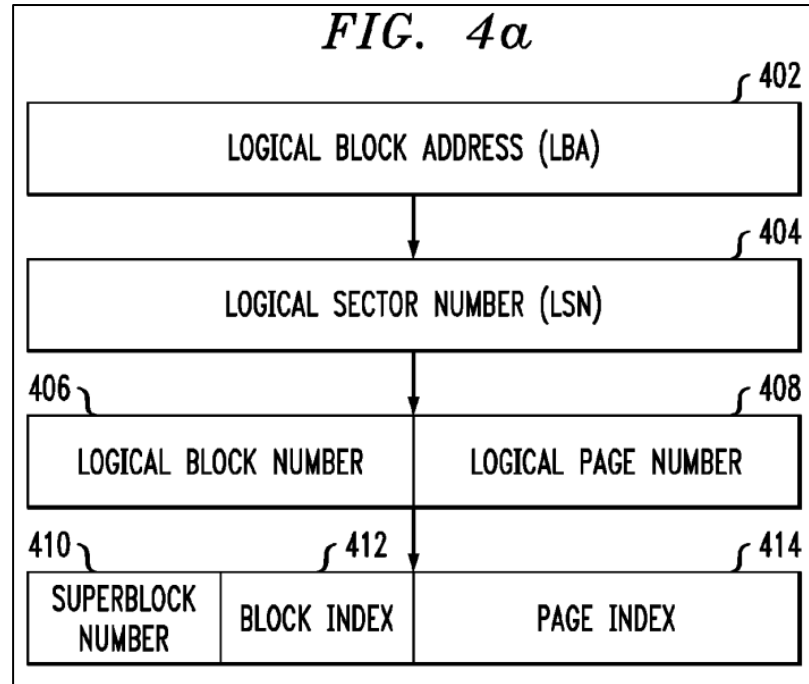


EX1004, 8:66-67, FIG. 4a; EX1003, ¶103.

In the second step, “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*. EX1003, ¶104.

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. EX1003, ¶105; EX1004, FIG. 4a:



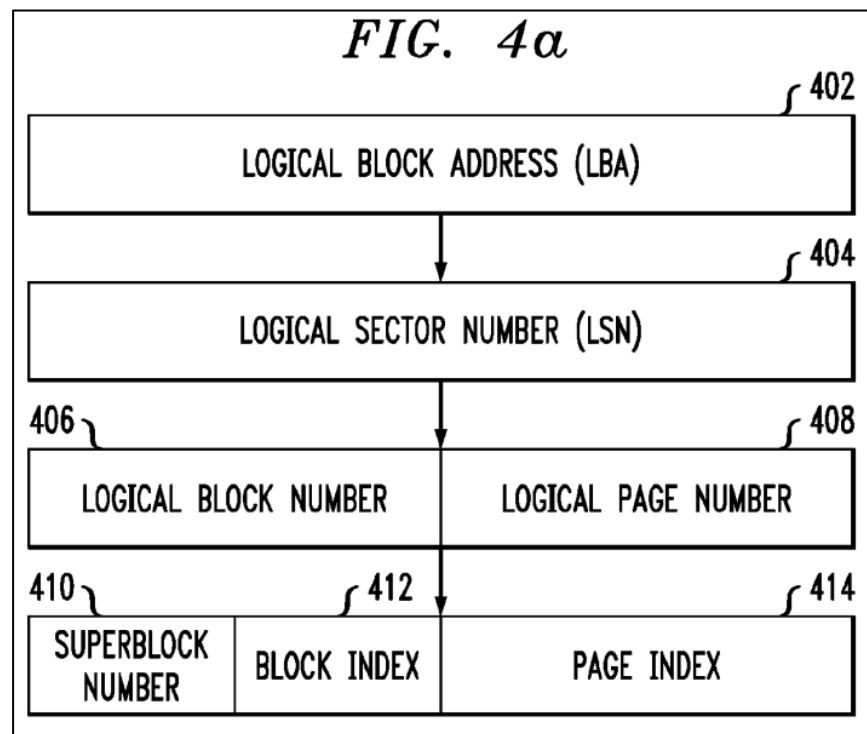
A POSITA would be motivated to divide by a certain power-of-two to split the bitstring into a first and second bitfields. Integer division by a certain power-of-two extracts bitfields out of bitstrings. EX1003, ¶106. The quotient is the extracted most significant bitfields (*i.e.*, bits on the lefthand portion of a bitstring). *Id.* Integer division is fast because it can be implemented with a simple bit shift operation. *Id.* Logical block number 406 is the higher order or most significant bitfield of the bitstring. EX1004, FIG. 4a; EX1003, ¶106. Applying integer division on the bitstring causes superblock number 410 to be extracted through the quotient. EX1003, ¶106. A POSITA would be motivated to implement integer division via a bit shift operation to perform the extraction quickly. *Id.*

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. EX1003, ¶107. 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*).

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. EX1003, ¶108; EX1004, FIG. 4a:



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. EX1003, ¶109.

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.

More substantively, logical block number 406 and block index 412 are abstractions of a physical block (*i.e.*, erase unit). EX1003, ¶110. 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.” EX1004, 9:13-17, 9:52-63, FIG. 4b; EX1003, ¶111.

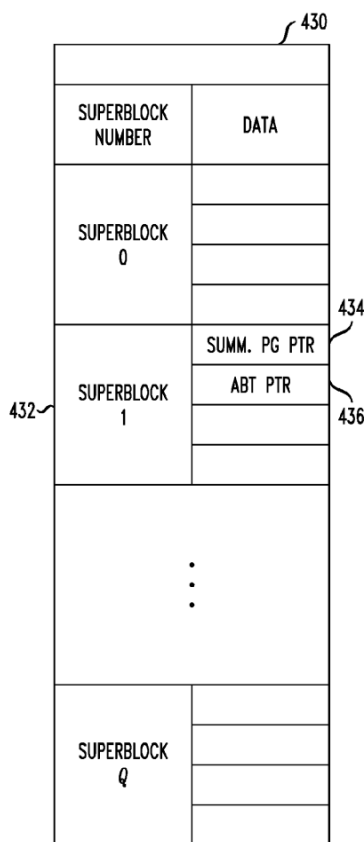
(i) 1[d]: *wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit, and*

Reiter discloses flash controller 104 (*circuitry*) translates the logical block number 406, and alternatively block index 412 (*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; EX1003, ¶112.

Superblock number 410 selects the summary page “contain[ing] ... mapping data for the associated Superblock”:

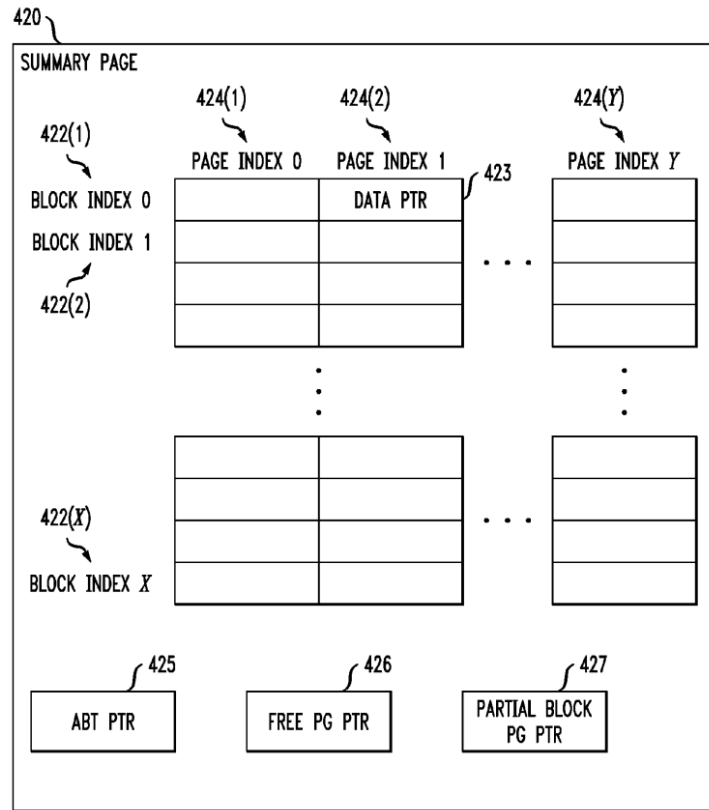
FIG. 4c



EX1004, 9:52-54, Fig. 4c; EX1003, ¶113.

“[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”:

FIG. 4b



EX1004, 9:11-17, 9:52-63, FIG. 4b; EX1003, ¶114.

Data pointer 423 points to the underlying, selected physical block. EX1004, 9:60-63, FIGS. 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*). EX1003, ¶115.

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

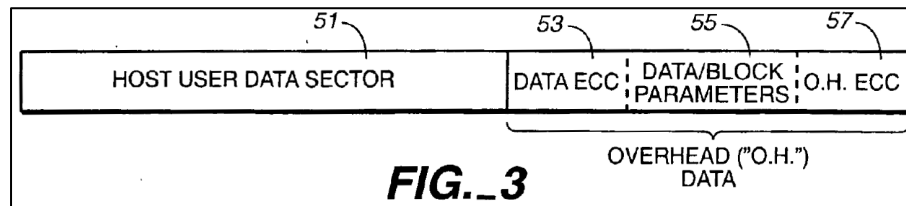
Reiter discloses flash controller 104's "determin[ing] the Superblock number, Block index, and Page index of the first page to be written" (*identified one of the physical erase units*), and the "requested data is written at the active page address at step 644" (*program associated data*). EX1004, 13:38-60, FIG. 6b; EX1003, ¶116.

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

Reiter's flash storage system 100 (*storage device*) includes a "flash controller 104" (*circuitry*). EX1004, 6:63-7:2, FIG. 1; EX1003, ¶117.

(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,*

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:



Therefore, every block in a logical unit includes this single sector of overhead data (*metadata for each physical erase unit which contains data*). EX1003, ¶¶118-119. 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*

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. EX1003, ¶120; EX1010, ¶[0009] (“Data loss can occur in an area that has not been refreshed for a sufficiently long period of time.”). 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. EX1003, ¶120.

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. EX1003, ¶121. This can alert flash controller 104 of potential data loss, and provide an opportunity to save the data. *Id.*; EX1010, ¶¶[0009], [0169].

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 away from a bad block to a working block is a common, simple solution that is faster than reading, erasing, and rewriting the same block and prevents data loss. EX1009, ¶¶[0007], [0015], [0136]; EX1003, ¶122.

(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.*

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]; EX1003, ¶123.

2. 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*

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. EX1003, ¶124. A POSITA would understand that “valid data” refers to the latest version of a data element. *Id.* 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. EX1003, ¶124; 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.*

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

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. EX1003, ¶¶125-126. 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 may change 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***). EX1003, ¶126. A POSITA would have also found it obvious to generate this metadata after a copy event so that the metadata is updated and remains accurate. EX1003, ¶126.

3. 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.*

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. 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 of 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. EX1003, ¶127.

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; EX1003, ¶128. If the data is no longer accessible, it should be erased immediately or at least immediately queued

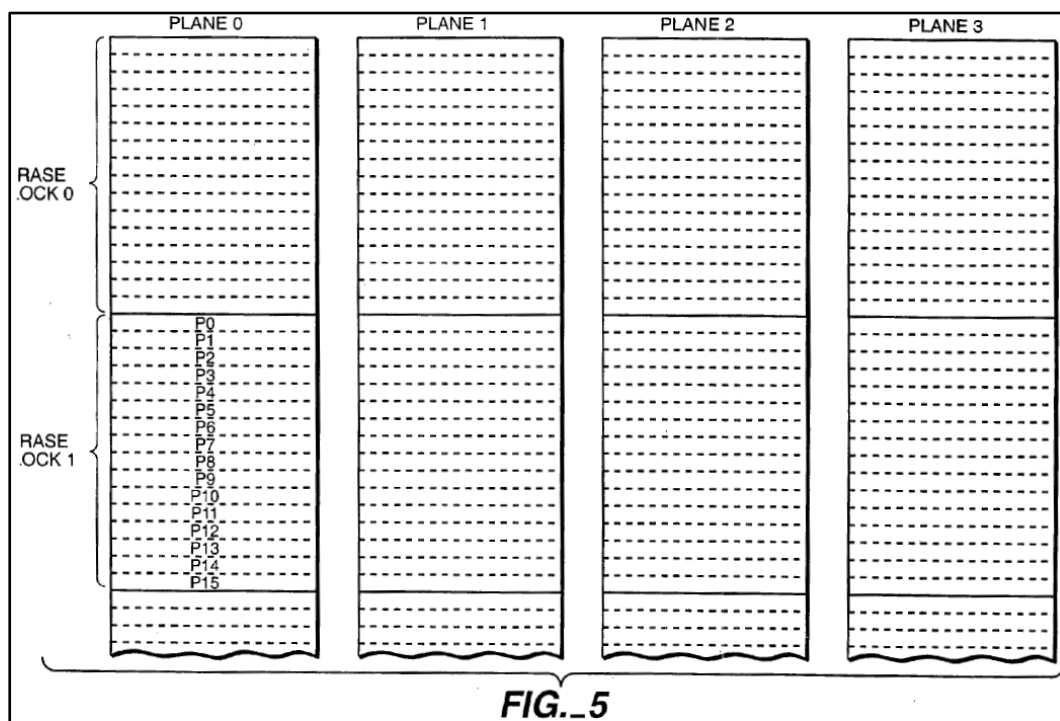
for erasure. Immediate erasure is simpler to implement and with less risk of error.

Id.

4. Claim 4

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

Sinclair-367 teaches that the flash memory may be organized into groups of blocks that span multiple planes:



EX1009, FIG. 5, ¶[0111]. A POSITA would be motivated to implement Reiter's *flash* media 118 as "divided into two or more planes" (*multiple planes*) per Sinclair-367's teachings. EX1009, ¶111, FIG. 5 (above); Section VIII.B. For instance, by the time of the '995 Patent, multi-plane architectures were well-known and 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.
EX1003, ¶129.

(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.*

As discussed above, 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* Ground I, Claim 3. 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 discussed above, to improve parallelism. See Section VIII.B; EX1003, ¶¶130-131.

5. 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*

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.* A POSITA would have also 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. EX1003, ¶¶132-133. Wear-leveling algorithms typically determine the need for wear-leveling by evaluating erase count metadata. *Id.*

(b) 5[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.*

Sinclair-367 discloses that the program/erase cycle's 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]. A POSITA would have also found it obvious to update this metadata after an erasure so that the metadata remains up-to-date. EX1003, ¶134.

6. 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.*

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; EX1003, ¶135.

When a block is added to the bad block list, a POSITA would be motivated to replace the bad block with a working block—remapping logical block number 406, and block index 412, to a new block. EX1003, ¶136. Data from a bad block cannot be trusted, especially given the unsuccessful read during the startup scan and subsequent failed erasure of the block. EX1004, 4:27-37, 17:13-15, 17:47-63. Keeping a bad block around risks reading erroneous data. EX1003, ¶136. Reiter discloses, upon detecting a bad block, adding the “block address ... to the bad block list.” EX1004, 18:4-8.

Further, a POSITA would understand that the bad block and data would be replaced with a working block written with the lost data. EX1003, ¶137. Reiter discloses configuring flash system 100 “analogously to hard drives in a redundant array of independent disks (RAID)” EX1004, 8:46-50. With RAID, data lost on one disc can be reconstructed with the data on the other redundant disks. Because the previous block is an unreliable bad block, a POSITA would understand that a new block is needed for the reconstructed data. EX1003, ¶137. And a POSITA would be motivated to change data pointers 423 in summary page 420 pointing to the old bad block to the new block with the reconstructed data (*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*). EX1003, ¶¶137-138.

This allows the LBA addressing, including logical block number 406 and block index 412, to be unchanged despite the data loss. *Id.* With the new working block, data at logical block number 406 and block index 412 is trustworthy and usable for later reads and writes (*utilize the remapped correspondence to service one or more of the incoming data access requests*). *Id.*

A POSITA would also be motivated to implement this failure detection and block replacement scheme. EX1003, ¶138. 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. 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. EX1003, ¶138. 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. EX1003, ¶138.

7. **Claim 8**

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

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; EX1003, ¶139.

(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.*

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*). EX1003, ¶140.

8. 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.*

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; EX1003, ¶141.

9. 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*

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.

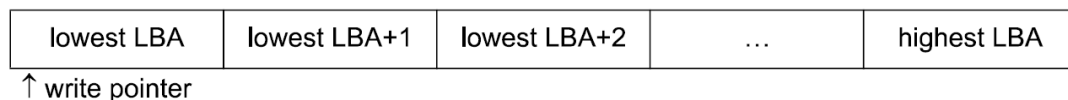


Figure 2 — Write pointer zone and write pointer after reset write pointer operation with no subsequent writes

EX1006, 14, Figure 2; EX1003, ¶¶142-144.

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							

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. EX1003, ¶144. Similarly, the zone is also within the targeted logical unit, volume, or partition (*segment within the block device addressed by the maintenance request*). *Id.*

(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.*

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). EX1003, ¶145. 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.

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. EX1003, ¶146. 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. *Id.* 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. *Id.* Delay of read and write operations for erasure operations is a complaint expressed by Reiter. EX1004, 4:19-26 (“[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”).

10. 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.*

See Ground I, Claim 5.

11. Claim 13

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

See Ground I, Limitation 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.*

See Ground I, Limitation 4[b].

12. Claim 14

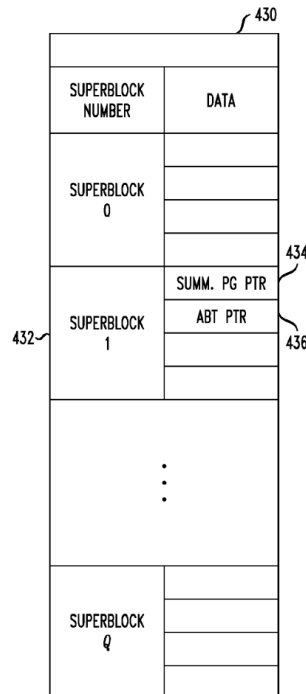
(a) *14[a]: The storage device of claim 1 wherein: the flash memory is to have block devices, including the addressed block device; and*

As 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, which is the addressed block device. EX1003, ¶150.

(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.*

As 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*).

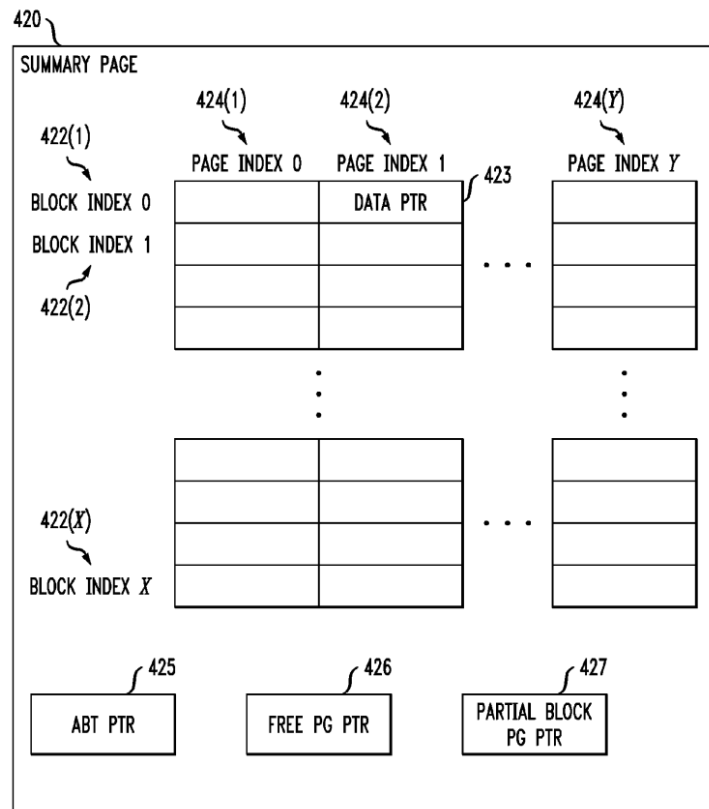
FIG. 4c



EX1004, 10:13-19, FIG. 4c; EX1003, ¶151.

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*).

FIG. 4b



EX1004, 9:60-63, FIG. 4b; EX1003, ¶152.

A POSITA would understand that the summary page is *specific* to the targeted logical unit, volume, or partition identified by the logical unit number of the ZBC read command (*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. The overlapping logical address ranges require separate address translation structures, including the summary pages, because Reiter’s disclosed address translation only takes the incoming logical block address as input. EX1003, ¶153. 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. *Id.* In order to resolve ambiguity of overlapping logical block addresses, each logical unit, volume, and partition need its own address translation structures. *Id.*

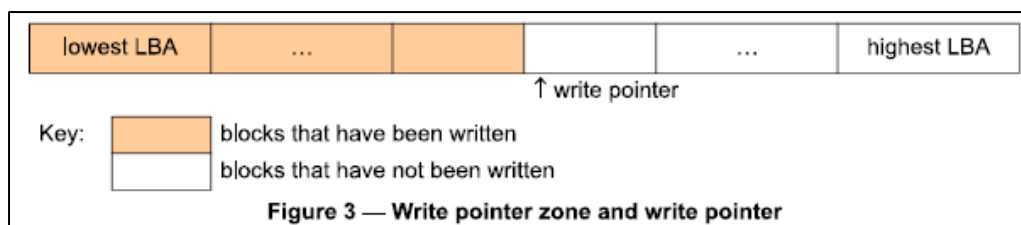
13. Claim 15

(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*

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; EX1003, ¶154.

Alternatively, Patent Owner’s infringement contentions assert that a zone’s write pointer discloses the limitation. EX1011, 170-171; EX1003, ¶155. 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:



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

Reiter's flash memory storage system's 100 (*storage device*) flash controller 104 (*circuitry*) performs 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*). EX1003, ¶156.

Alternatively, Patent Owner's infringement contentions assert 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:

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								

EX1006, Table 12; EX1003, ¶157.

The returned zone descriptors include the zone's write pointer.

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								

EX1006, Table 13; EX1003, ¶158.

14. 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.*

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*):

Table 12 — REPORT ZONES parameter data

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

EX1006, 25, Tables 12-13; EX1003, ¶159.

Because a POSITA would be motivated to map zones onto superblocks, (Section VIII.C), 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; EX1003, ¶160.

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; EX1003, ¶161.

Reiter's flash system 100 executing a reset write pointer command (Section VIII.D.4.b) (*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" (*a quantum of flash memory which is dependent on the information transmitted to the host identifying the number of structures*). EX1006, 3; EX1003, ¶162.

15. 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.*

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:

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								

EX1006, 25, Table 13; *see also id.* Tables 11-12; EX1003, ¶163.

A POSITA would be motivated to implement the reset write pointer command to cause erasure of the targeted zone. Ground I, Claim 10[b]; EX1003, ¶164. Because the reset write pointer command erases an entire zone, the zone length *identifies an amount of flash memory that is to be erased as a unit*.

16. 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.*

As discussed above, a ZBC zone constitutes *an addressed segment*. See Ground I, Limitation 1[b][iii].

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*). EX1003, ¶166. 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. *Id.* 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. *Id.* 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. *Id.* To avoid data loss, while maintaining capacity and speed, a POSITA would require zones to contain a whole number of physical blocks. *Id.* Further, as discussed with Claim 7, Reiter discloses detecting blocks that have failed. EX1004, 4:27-53. 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. *Id.*

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 I, 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*. EX1003, ¶167.

17. Claim 26

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

As discussed above, the Reiter-ZBC-Sinclair-367 system comprises multiple planes, as disclosed in Sinclair-367. See Ground I, Claim 4[a]; EX1009, FIG. 5, ¶[0111]; EX1003, ¶168.

(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.

A ZBC write command includes an LBA and “transfer length.” EX1007, Table 112.

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)	SERVICE ACTION (000Bh)							
9									(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)	EXPECTED LOGICAL BLOCK APPLICATION TAG							
25									(LSB)
26	(MSB)	LOGICAL BLOCK APPLICATION TAG MASK							
27									(LSB)
28	(MSB)	TRANSFER LENGTH							
...									
31									(LSB)

“TRANSFER LENGTH ... specifies the ... logical blocks ... that shall be transferred ... starting [from] ... the [LBA].” EX1007, 198. With a large transfer length, a ZBC *write* command causes several write operations across *sequential addresses*. EX1003, ¶168.

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. EX1009, ¶¶[0111]-[0114]; *see* Ground I, Claim 4[a]-4[b]. “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]; EX1003, ¶170.

18. 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.*

As 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. Ground I, Limitation 1[b][i]. As 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*). EX1003, ¶171.

19. 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;*

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; EX1003, ¶172.

ZBC write commands also cause flash system 100 to transmit information back to the host 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; EX1003, ¶173.

A POSITA would be motivated to modify the transmission to include information identifying the completed command. EX1003, ¶174. Reiter discloses processing multiple commands in parallel. 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. EX1003, ¶174. Identifying the completed command allows the host to verify command completion or repeat

commands that return errors. *Id.* 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.*** *Id.*

Moreover, ZBC discloses incorporating such information in some circumstances. EX1003, ¶175. 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;

As discussed above, the Reiter-ZBC system discloses circuitry to transmit the address value to a host. *See* Ground I, Claim 28[a]. 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.*** EX1003, ¶176.

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

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 discussed above. *See* Section IX.A.1. A ZBC read command includes an LBA (*an associated read address*):

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)	SERVICE ACTION (0009h)							
9									
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)	EXPECTED LOGICAL BLOCK APPLICATION TAG							
...									
25		(LSB)							
26	(MSB)	LOGICAL BLOCK APPLICATION TAG MASK							
...									
27		(LSB)							
28	(MSB)	TRANSFER LENGTH							
...									
31		(LSB)							

EX1007, Table 63; EX1003, ¶177.

- (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.*

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. EX1003, ¶178. Allowing read commands prior to write command completion risks reading incorrect data. *Id.* 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. *Id.*

20. Claim 29

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

See Ground I, Claim 1[pre].

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

See Ground I, Claim 1[a].

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

See Ground I, 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,*

See Ground I, Claim 1[b][i].

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

See Ground I, Claim 1[b][ii].

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

See Ground I, Claim 1[b][iii].

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

See Ground I, 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

See Ground I, 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

See Ground I, 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;

See Ground I, Claim 1[e].

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

See Ground I, 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,

See Ground I, 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

As 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. Ground I, 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.*; EX1003, ¶191.

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 change 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*). EX1003, ¶192. A POSITA would have also found it

obvious to generate this metadata after a copy event so that the metadata remains accurate. *Id.*

(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.*

See Ground I, Claim 1[f][iii].

21. Claim 30

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

See Ground I, 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*

See Ground I, Claim 1[a], 1[b][ii]-1[d].

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

See Ground I, 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,*

See Ground I, Claim 1[b][i].

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

See Ground I, 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

See Ground I, 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,

See Ground I, 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

See Ground I, Claim 1[c], 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

See Ground I, 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;

See Ground I, Claim 30[e].

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

See Ground I, 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,*

See Ground I, 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*

See Ground I, 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*

See Ground I, 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.*

See Limitation 1[f][i], Claim 2[b].

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

A. Overview of the Prior Art

1. Reiter (EX1004)

Section VIII.A.1, *supra*.

2. ZBC (EX1006), SBC-4 (EX1007), and SPC-4 (EX1008)

Section VIII.A.2, *supra*.

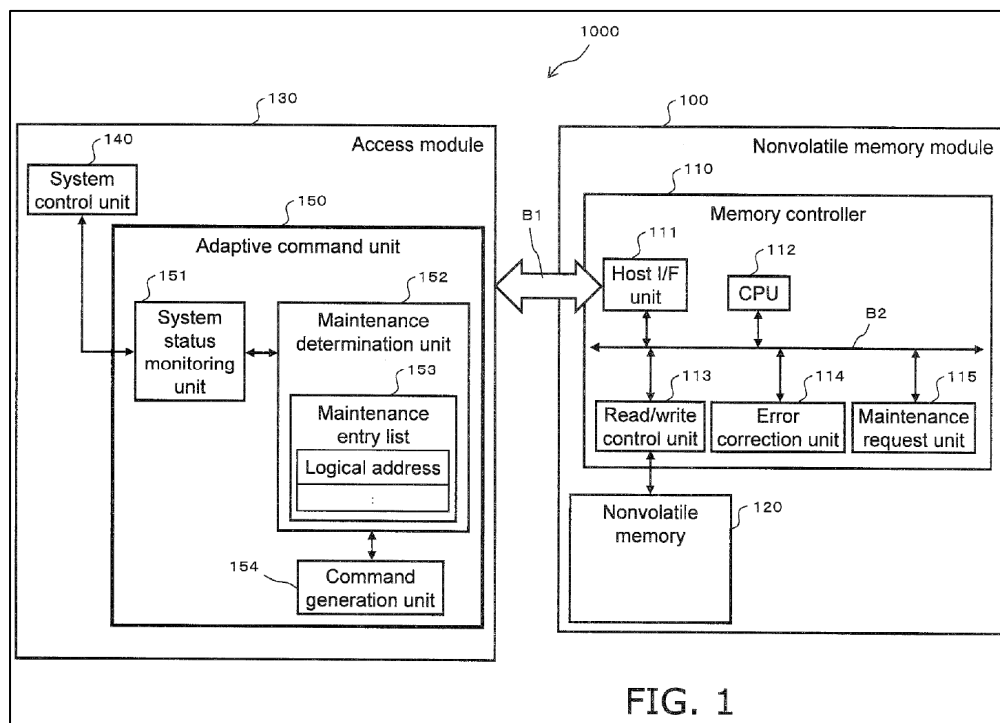
3. Sinclair-367 (EX1009)

Section VIII.A.3, *supra*.

4. Yamada (EX1010)

U.S. Publication No. 2011/0271032 (“Yamada”) was filed on July 27, 2010, and was published on November 3, 2011. It is prior art under AIA 35 U.S.C. § 102(a)(1).

Yamada discloses a nonvolatile memory module with flash memory and memory controller. Read and write commands are issued to the nonvolatile memory module by an access module. EX1010, FIG. 1:



Yamada complains about maintenance operations, such as garbage collection or bad block management, delaying normal storage operations for flash memory

devices. EX1010, Abstract. Yamada performs maintenance operations during the lulls between sets of normal storage operations. *Id.* To do so, Yamada implements a maintenance request unit on the nonvolatile memory module, and a maintenance determination unit on the access module. EX1010, ¶¶[0081]-[0082], [0089], FIG. 1. The maintenance request unit determines when certain parts of flash memory needs garbage collection or bad block management, and transmits a request for maintenance to the maintenance determination unit. EX1010, ¶¶[0081]-[0082]. The maintenance determination unit, by being on the access module, monitors the access module's status. EX1010, Abstract, ¶[0089]. When the maintenance determination unit finds downtime in normal processing, the maintenance determination unit performs requested maintenance operations. *Id.*

B. Motivation to Combine

A POSITA would have been motivated to combine the teachings of Reiter with ZBC, Sinclair-367, and Yamada. Specifically, a POSITA would have been motivated to (1) modify Reiter's flash device to accept and execute ZBC commands (Section VIII.B, *supra*), (2) incorporate Sinclair-367's metadata practices and implementations for performing concurrent read/write operations on flash memory (Section VIII.B, *supra*), and (3) incorporate Yamada's maintenance request and determination units in the flash device and processor 10 disclosed in Reiter. EX1003, ¶¶209-210.

A POSITA would be motivated to incorporate Yamada’s maintenance scheme in Reiter’s flash memory storage system. *Id.* Reiter complains 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. Yamada resolves this problem, discussing a maintenance scheme that reduces delay of normal processing caused by maintenance operations. EX1010, Abstract (“[M]aintenance is performed without being ... performed together with normal system processing,” “enabl[ing] the maintenance to be performed with sufficiently long processing time.”). To resolve Reiter’s complaint, a POSITA would be motivated to incorporate Yamada’s maintenance disclosures in Reiter’s flash device. EX1003, ¶210.

C. Claim-by-Claim Analysis

1. 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*

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. EX1003, ¶212. 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*). *Id.* 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. EX1003, ¶213. After the average retention period, a memory cell can be unreliable, risking loss of the data stored in the cell. *Id.*; 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. EX1003, ¶213. Thus, Yamada teaches or at least suggests that the maintenance request unit that resided in the memory controller may compare and transmit a maintenance request to the host when a criterion value for maintenance, such as the average retention period, is exceeded. EX1010, Fig. 1, ¶¶[0033], [0189]. A POSITA would

have 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. EX1003, ¶214.

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]; EX1003, ¶215.

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. EX1003, ¶216. 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; EX1003, ¶216.

(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.*

As 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. EX1003, ¶217.

2. 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.*

See Ground II, Limitation 6[a]-[b].

X. MANDATORY NOTICES AND FEES

A. Real Party-In-Interest

The real party-in-interest for Petitioner is Samsung Electronics Co., Ltd. and Samsung Electronics America, Inc.

B. Related Matters

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

C. Counsel and Service Information

Petitioner provides the following counsel and service information. Petitioner consents to electronic service the email addresses listed in the table below. Pursuant to 37 C.F.R. § 42.10(b), a Power of Attorney accompanies this Petition.

LEAD COUNSEL	BACK-UP COUNSEL
James M. Glass (Reg. No. 46,729) jimglass@quinnemanuel.com QUINN EMANUEL URQUHART & SULLIVAN, LLP 295 Fifth Avenue, 9th Floor New York, NY 10016 Tel: (212) 849-7000 Fax: (212) 849-7100	John T. McKee (Reg. No. 65,926) johnmckee@quinnemanuel.com QUINN EMANUEL URQUHART & SULLIVAN, LLP 295 5th Avenue, 9th Floor New York, New York 10016 Tel: (212) 849-7000 Fax: (212) 849-7100 Nima Hefazi (Reg. No. 63,658) QUINN EMANUEL URQUHART & SULLIVAN, LLP 865 S. Figueroa St., 10 th Floor Los Angeles, California 90017 nimahefazi@quinnemanuel.com Tel: (213) 443-3000 Fax: (213) 443-3100

--	--

D. Payment of Fees

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

XI. CONCLUSION

For these reasons, Petitioner requests *inter partes* review of the '995 patent.

DATED: August 5, 2025

Respectfully submitted,

By /s/ John T. McKee
 Reg. No. 65,926
 QUINN EMANUEL URQUHART &
 SULLIVAN, LLP
 295 Fifth Ave.
 New York, NY 10016
 Tel: (212) 849-7000
 Fax: (212) 849-7100

Attorney for Petitioners

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. §42.24(d), Petitioner hereby certifies, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,996. Pursuant to 37 C.F.R. §42.24(d), this word count excludes the table of contents, table of authorities, mandatory notices under §42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing.

Date: August 5, 2025

Respectfully submitted,
By: /s/ John T. McKee
Reg. No. 65,926

Attorney for Petitioners

CERTIFICATE OF SERVICE

Pursuant to 37 C.F.R. §§ 42.6(e), 42.105(a), the undersigned hereby certifies service on the Patent Owner of a copy of this Petition and its respective exhibits at the official correspondence address for the attorneys of record for the '995 patent as shown in USPTO PAIR via FedEx:

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

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

andrea@millerfairhenry.com
radian@caldwellcc.com

DATED: August 6, 2025

Respectfully submitted,
By: /s/ John T. McKee
Reg. No. 65,926

Attorney for Petitioners