

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

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

v.

RADIAN MEMORY SYSTEMS, LLC,
Patent Owner.

Case IPR2025-01377

U.S. Patent No. 11,347,656

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

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	STANDING	2
III.	TECHNICAL BACKGROUND	2
IV.	OVERVIEW OF THE '656 PATENT	3
A.	Priority	3
B.	Prosecution History	3
C.	Disclosure	4
V.	FOUNDATIONS	5
VI.	LEVEL OF ORDINARY SKILL	5
VII.	CLAIM CONSTRUCTION	6
VIII.	GROUND I: CLAIMS 1-12, 16, 20-23 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, AND SPC-4	6
A.	Reiter	6
B.	ZBC, SBC-4, and SPC-4	7
C.	Motivation to Combine	12
D.	Claim-by-Claim Analysis.....	14
1.	Claim 1	14
(a)	1[pre].....	14
(b)	1[a]	15
(c)	1[b]	16
(d)	1[b][i]	18
(e)	1[b][ii]	23
(f)	1[b][iii]	23
(g)	1[b][iv]	24
(h)	1[c]	27
(i)	1[d]	30
(j)	1[e]	33
2.	Claim 2	35

	(a)	2[pre].....	35
	(b)	2[a].....	35
	(c)	2[b].....	36
3.		Claim 3.....	36
	(a)	3[pre].....	36
	(b)	3[a].....	36
	(c)	3[b].....	37
	(d)	3[c]-[d].....	38
4.		Claim 4.....	40
	(a)	4[pre].....	40
	(b)	4[a].....	40
	(c)	4[b].....	42
5.		Claim 5.....	42
	(a)	5[pre].....	42
	(b)	5[a].....	43
	(c)	5[b].....	43
	(d)	5[b][i]-[ii].....	43
6.		Claim 6.....	46
	(a)	6[pre].....	46
	(b)	6[a].....	46
	(c)	6[b].....	47
7.		Claim 7.....	48
	(a)	7[pre].....	48
	(b)	7[a].....	48
	(c)	7[b].....	48
	(d)	7[c].....	49
	(e)	7[d].....	49
8.		Claim 8.....	49
	(a)	8[pre].....	49

	(b) 8[a].....	49
	(c) 8[b].....	52
9.	Claim 9.....	52
	(a) 9[pre].....	52
	(b) 9[a].....	52
10.	Claim 10.....	53
	(a) 10[pre].....	53
	(b) 10[a].....	53
11.	Claim 11.....	55
	(a) 11[pre].....	55
	(b) 11[a].....	55
12.	Claim 12.....	56
	(a) 12[pre].....	56
	(b) 12[a].....	56
13.	Claim 16.....	57
	(a) 16[pre].....	57
	(b) 16[a].....	57
	(c) 16[b].....	57
	(d) 16[b][i].....	57
	(e) 16[b][ii].....	58
14.	Claim 20.....	59
	(a) 20[pre].....	59
	(b) 20[a].....	59
15.	Claim 21-23.....	59

IX.	GROUND II: CLAIMS 1, 13-14, AND 19 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, SPC-4, AND SINCLAIR-367.....	60
A.	Sinclair-367	60
B.	Motivation to Combine	62
C.	Claim-by-Claim Analysis.....	64

1.	Claim 1	64
	(a) 1[pre]-[d].....	64
	(b) 1[e]	64
2.	Claim 13	65
	(a) 13[pre].....	65
	(b) 13[a]	65
	(c) 13[b].....	65
	(d) 13[b][i]	65
	(e) 13[b][ii].....	66
	(f) 13[b][iii].....	66
	(g) 13[b][iv]	67
3.	Claim 14	67
	(a) 14[pre].....	67
	(b) 14[a]	67
	(c) 14[b].....	68
	(d) 14[b][i]	68
	(e) 14[b][ii]-[iv].....	68
4.	Claim 19	69
	(a) 19[pre].....	69
	(b) 19[a]	69
	(c) 19[b].....	70
X.	GROUND III: CLAIMS 15, 17-18 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SINCLAIR-367, YAMADA	72
A.	Yamada.....	72
B.	Motivation to Combine	73
C.	Claim-by-Claim Analysis.....	74
1.	Claim 15	74
	(a) 15[pre].....	74
	(b) 15[a]	74

	(c)	15[b].....	74
	(d)	15[b][i].....	75
	(e)	15[b][ii].....	75
	(f)	15[b][iii].....	75
2.		Claim 17.....	76
	(a)	17[pre].....	76
	(b)	17[a].....	76
	(c)	17[b].....	76
	(d)	17[b][i].....	76
	(e)	17[b][ii].....	76
3.		Claim 18.....	77
	(a)	18[pre].....	77
	(b)	18[a].....	77
	(c)	18[b].....	77
	(d)	18[b][i].....	77
	(e)	18[b][ii].....	77
	(f)	18[b][iii].....	77

XI.	GROUND IV: CLAIMS 1-14, 16, 19-23 ARE RENDERED OBVIOUS BY BAN IN LIGHT OF ZBC, SBC-4, SPC-4, SINCLAIR-367, AND YAMADA.....		78
A.	Ban.....		78
B.	Motivation to Combine		80
C.	Claim-by-Claim Analysis.....		83
	1.	Claim 1	83
		(a) 1[pre].....	83
		(b) 1[a].....	84
		(c) 1[b].....	85
		(d) 1[b][i].....	86
		(e) 1[b][ii].....	88
		(f) 1[b][iii].....	88

	(g)	1[b][iv].....	89
	(h)	1[c].....	90
	(i)	1[d].....	91
	(j)	1[e].....	92
2.	Claim 2		92
	(a)	2[pre].....	92
	(b)	2[a]-[b].....	92
3.	Claim 3		93
	(a)	3[pre].....	93
	(b)	3[a].....	93
	(c)	3[b].....	93
	(d)	3[c]-[d].....	93
4.	Claim 4		94
	(a)	4[pre].....	94
	(b)	4[a].....	94
	(c)	4[b].....	94
5.	Claim 5		95
	(a)	5[pre].....	95
	(b)	5[a].....	95
	(c)	5[b].....	95
	(d)	5[b][i]-[ii].....	95
6.	Claim 6		96
	(a)	6[pre].....	96
	(b)	6[a].....	97
	(c)	6[b].....	97
7.	Claim 7		98
	(a)	7[pre].....	98
	(b)	7[a].....	98
	(c)	7[b].....	98

	(d) 7[c].....	98
	(e) 7[d].....	98
8.	Claim 8.....	98
	(a) 8[pre].....	98
	(b) 8[a].....	98
	(c) 8[b].....	99
9.	Claim 9.....	99
	(a) 9[pre].....	99
	(b) 9[a].....	99
10.	Claim 10.....	99
	(a) 10[pre].....	99
	(b) 10[a].....	99
11.	Claim 11.....	100
	(a) 11[pre].....	100
	(b) 11[a].....	100
12.	Claim 12.....	100
	(a) 12[pre].....	100
	(b) 12[a].....	100
13.	Claim 13.....	101
	(a) 13[pre].....	101
	(b) 13[a].....	101
	(c) 13[b].....	101
	(d) 13[b][i].....	101
	(e) 13[b][ii].....	101
	(f) 13[b][iii].....	101
	(g) 13[b][iv].....	102
14.	Claim 14.....	102
	(a) 14[pre].....	102
	(b) 14[a].....	102

	(c)	14[b].....	102
	(d)	14[c].....	102
	(e)	14[d]-[f]	102
15.	Claim 16.....		102
	(a)	16[pre].....	102
	(b)	16[a].....	102
	(c)	16[b].....	102
	(d)	16[b][i]	102
	(e)	16[b][ii].....	103
16.	Claim 19.....		103
	(a)	19[pre].....	103
	(b)	19[a].....	103
	(c)	19[b].....	103
17.	Claim 20.....		103
	(a)	20[pre].....	103
	(b)	20[a].....	103
18.	Claim 21-23.....		103
XII.	GROUND V: CLAIMS 15, 17-18 ARE RENDERED OBVIOUS BY BAN IN LIGHT OF ZBC, SINCLAIR-367, YAMADA.....		104
A.	Motivation to Combine		104
B.	Claim-by-Claim Analysis.....		105
1.	Claim 15		105
	(a)	15[pre].....	105
	(b)	15[a].....	105
	(c)	15[b].....	105
	(d)	15[b][i]	105
	(e)	15[b][ii].....	105
	(f)	15[b][iii].....	105
2.	Claim 17.....		105

(a)	17[pre].....	105
(b)	17[a].....	105
(c)	17[b].....	105
(d)	17[b][i].....	106
(e)	17[b][ii].....	106
3.	Claim 18.....	106
(a)	18[pre].....	106
(b)	18[a].....	106
(c)	18[b].....	106
(d)	18[b][i].....	106
(e)	18[b][ii].....	106
(f)	18[b][iii].....	106
XIII.	MANDATORY NOTICES AND FEES	106
A.	Real Party-In-Interest	106
B.	Related Matters.....	107
C.	Counsel and Service Information.....	107
D.	Payment of Fees	108
XIV.	CONCLUSION.....	108
	CERTIFICATE OF WORD COUNT	109
	CERTIFICATE OF SERVICE	1

EXHIBIT LIST

Ex.	Description
1001	U.S. Patent No. 11,347,656 (“’656 Patent”)
1002	Prosecution History of U.S. Patent No. 11,347,656
1003	Expert Declaration of Dr. Heiner Litz.
1004	U.S. Patent No. 8,301,861 (“Reiter”)
1005	U.S. Patent No. 5,404,485 (“Ban”)
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 ’656 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 Apr. 16, 2015), https://web.archive.org/web/20150416184431/http://www.t10.org/cgi-bin/ac.pl?t=f&f=sbc4r02.pdf (last visited Jul. 17, 2025)
1106	“File Access Monitor”, T10 (archived May 19, 2014), 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)

Ex.	Description
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	Working Draft Revision 1 of the Small Computer System Interface Zoned Block Commands
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
1211	Intentionally Omitted
1212	U.S. Pat. Pub. 2008/0189485A1

CLAIM LISTING

Claim 1	
1[pre]	A storage device comprising:
1[a]	flash memory having physical erase units; and
1[b]	circuitry to receive incoming data access requests and for each of the incoming data access requests, accompanying address information, wherein said circuitry is to:
1[b][i]	derive, from the accompanying address information, a first address portion, a second address portion and a third address portion,
1[b][ii]	identify an addressed block device from the first address portion,
1[b][iii]	identify an addressed segment within the addressed block device from the second address portion, and
1[b][iv]	identify an addressed storage location within the addressed segment from the third address portion,
1[c]	wherein the circuitry is to use a division operation to identify a logical erase unit within the addressed segment,
1[d]	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit;
1[e]	wherein the storage device is 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 of the incoming data access requests.
Claim 2	
2[pre]	The storage device of claim 1 wherein:
2[a]	said circuitry comprises at least one processor; and

2[b]	the storage device further comprises instructions stored on at least one physical storage medium, said instructions when executed to control the at least one processor.
Claim 3	
3[pre]	The storage device of claim 1 wherein:
3[a]	the incoming data access requests are read requests received from a host;
3[b]	the storage device can receive a write request from the host, wherein the circuitry is to, for the received write request, derive a first address portion, a second address portion and a third address portion, identify an addressed block device from the first address portion, identify an addressed segment within the addressed block device from the second address portion, identify a corresponding logical erase unit from the third address portion and identify a physical erase unit which is mapped to the corresponding logical erase unit;
3[c]	the storage drive is to, in association with fulfillment of the received write request, identify a physical storage location in which associated write data has been stored and which is associated with the corresponding logical erase unit, and to transmit information representing an address value to the host corresponding to the identified physical storage location; and
3[d]	for one of the read requests which is directed to data which has been written into the flash memory pursuant to the received write request, at least one of the first address portion, the second address portion or the third address portion associated with the one of the read requests is dependent on the address value which has been transmitted to the host in association with the data which has been written into the flash memory pursuant to the received write request.
Claim 4	
4[pre]	The storage device of claim 1 wherein:

4[a]	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
4[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.
Claim 5	
5[pre]	The storage device of claim 1 wherein:
5[a]	the flash memory is to have block devices, including the addressed block device; and
5[b]	in connection with a given one of the incoming data access requests, the circuitry is to:
5[c]	identify and retrieve a look-up table specific to the addressed block device which was identified from the first address portion derived from the given one of the incoming data access requests, and
5[d]	identify a physical storage location in the flash memory dependent on the look-up table.
Claim 6	
6[pre]	The storage device of claim 1 wherein:
6[a]	the storage device further comprises circuitry to track metadata for the addressed segment, where said 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
6[b]	the storage device further comprises circuitry to transmit information to a host dependent on said tracked metadata.
Claim 7	
7[pre]	The storage device of claim 1 wherein:

7[a]	the incoming data access requests are read requests received from a host;
7[b]	the storage device can receive write requests from the host, and the storage device comprises circuitry to issue commands to the flash memory in fulfillment of the write requests;
7[c]	the storage device is to assign an address value in association with fulfillment of each of the write requests, in order to identify a physical storage location in which associated write data has been stored, and comprises circuitry to transmit information representing each assigned address value to the host; and
7[d]	for one of the incoming data access requests which is directed to data which has been written into the flash memory pursuant to one of the write requests, the addressed storage location identified from the third portion is dependent on the assigned address value which was transmitted to the host in association with the one of the write requests.
Claim 8	
8[pre]	The storage device of claim 1 wherein:
8[a]	the storage device can receive a request from a host to copy host-selected data from a first storage location in flash memory to a second storage location in the flash memory, and the storage device comprises circuitry to issue commands to the flash memory, in response to the request from the host to copy, to read the host-selected data from the first storage location in the flash memory and write the host-selected data to the second storage location in the flash memory without sending the data being copied to the host; and
8[b]	the request from the host to copy is to designate destination address information corresponding to the second storage location, wherein the storage device is to identify from the destination address information a destination block device and an addressed segment within the destination block device, and wherein the second storage location corresponds to the addressed segment within the destination block device.

Claim 9	
9[pre]	The storage device of claim 8 wherein
9[a]	the storage device is to identify from the destination address information a destination logical erase unit and is to select the second storage location in dependence on the identified destination logical erase unit.
Claim 10	
10[pre]	The storage device of claim 1 wherein
10[a]	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 for which a quantum of flash memory selected by the commands which are issued in response to the at least one request is dependent on the information transmitted to the host identifying the number of structures.
Claim 11	
11[pre]	The storage device of claim 10 wherein
11[a]	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 12	
12[pre]	The storage device of claim 1 wherein
12[a]	the addressed segment corresponds to an integer number of physical erase units for which the 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 13	
13[pre]	The storage device of claim 1 wherein:
13[a]	the flash memory is to have block devices, including the addressed block device; and
13[b]	the storage device further comprises circuitry to:
13[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, the metadata representing a time since one or more physical storage locations associated with the one of the block devices has been a subject of a write operation,
13[d]	compare the metadata to at least one threshold,
13[e]	upon detection that specific metadata has satisfied the one or more thresholds, copy data stored at one of the one or more physical storage locations associated with the specific metadata from a first one of the physical erase units for which the failure condition has not been detected by said storage device to a second one of the physical erase units for which the failure condition has not been detected by said storage device, and
13[f]	remap the logical erase unit corresponding to the first one of the physical erase units for which the failure condition has not been detected by said storage device to the second one of the physical erase units for which the failure condition has not been detected by said storage device.
Claim 14	
14[pre]	The storage device of claim 1 wherein:
14[a]	the flash memory is to have block devices, including the addressed block device; and
14[b]	the storage device further comprises circuitry to:

14[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, wherein the stored metadata is dependent on an erase count of associated physical memory space,
14[d]	compare the metadata to at least one threshold,
14[e]	upon detection that specific metadata has satisfied the one or more thresholds, copy data stored at one of the one or more physical storage locations associated with the specific metadata from a first one of the physical erase units for which the failure condition has not been detected by said storage device to a second one of the physical erase units for which the failure condition has not been detected by said storage device, and
14[f]	remap the logical erase unit corresponding to the first physical erase unit for which the failure condition has not been detected by said storage device to the second physical erase unit for which the failure condition has not been detected by said storage device.
Claim 15	
15[pre]	The storage device of claim 1 wherein:
15[a]	the flash memory is to have block devices, including the addressed block device; and
15[b]	the storage device further comprises circuitry to:
15[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, the metadata representing a time since one or more physical storage locations associated with the one of the block devices has been a subject of a write operation,
15[d]	compare the metadata to at least one threshold, and
15[e]	responsive to detection that specific metadata has satisfied the one or more thresholds, transmit information to a host device wherein such transmitted information conveys address information for data stored at one of the one or more physical storage locations associated with the specific metadata.

Claim 16	
16[pre]	The storage device of claim 1 wherein:
16[a]	the flash memory is to have block devices, including the addressed block device; and
16[b]	the storage device further comprises circuitry to:
16[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, and
16[d]	receive a query, issued by a host device, and responsive to the query, compare the metadata to at least one threshold, to identify specific metadata, and transmit information to the host device wherein such transmitted information conveys address information for data stored at one of the one or more physical storage locations associated with the specific metadata.
Claim 17	
17[pre]	The storage device of claim 1 wherein:
17[a]	the flash memory is to have block devices, including the addressed block device; and
17[b]	the storage device further comprises circuitry to:
17[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, wherein the stored metadata represents an erase count of associated physical memory space, and
17[d]	receive a query, issued by a host device, wherein said query is specific to one of the block devices, and responsive to the query, identify specific metadata, associated with the specified one of the block devices, and transmit information to the host device that conveys the specific metadata.
Claim 18	
18[pre]	The storage device of claim 1 wherein:

18[a]	the flash memory is to have block devices, including the addressed block device; and
18[b]	the storage device further comprises circuitry to:
18[c]	store metadata on an independent basis for flash memory associated with each one of the block devices, wherein the stored metadata represents an erase count of associated physical memory space, and
18[d]	compare the metadata to at least one threshold, and
18[e]	responsive to detection that specific metadata has satisfied the one or more thresholds, transmit information to a host device wherein such transmitted information conveys address information for data stored at one of the one or more physical storage locations associated with the specific metadata.
Claim 19	
19[pre]	The storage device of claim 1 wherein:
19[a]	the flash memory comprises multiple planes; and
19[b]	the storage device is to, for each one of a set of the incoming data access requests having sequential addressing, identify a commonly addressed page and write data accompanying the set of incoming data access requests concurrently into respective ones of the multiple planes.
Claim 20	
20[pre]	The storage device of claim 1 wherein
20[a]	the division operation is applied to subdivide the third address portion into respective first and second bit fields.
Claim 21	
21[pre]	A storage device comprising:
21[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
21[b]	circuitry to receive incoming data access requests and for each of the incoming data access requests, accompanying address information, wherein said circuitry is to:
21[b][i]	derive, from the accompanying address information, a first address portion, a second address portion and a third address portion,
21[b][ii]	identify an addressed block device from the first address portion,
21[b][iii]	identify an addressed segment within the addressed block device from the second address portion, and
21[b][iv]	identify an addressed storage location within the addressed segment from the third address portion,
21[c]	wherein the circuitry is to use a division operation to identify a logical erase unit within the addressed segment,
21[d]	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit;
21[e]	wherein the storage device is 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 of the incoming data access requests.
Claim 22	
22[pre]	A storage device comprising:
22[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;
22[b]	circuitry to receive incoming data read requests and incoming data write requests and, for each of the incoming data read requests and incoming data write requests, accompanying address information, wherein said circuitry is to, for each of the incoming data read requests:
22[b][i]	derive, from the accompanying address information, a first address portion, a second address portion and a third address portion,
22[b][ii]	identify an addressed block device from the first address portion,
22[b][iii]	identify an addressed segment within the addressed block device from the second address portion, and
22[b][iv]	identify an addressed storage location within the addressed segment from the third address portion,
22[c]	wherein the circuitry is to use a division operation to identify a logical erase unit within the addressed segment,
22[d]	wherein the circuitry is to identify one of the physical erase units that corresponds to the identified logical erase unit;
22[e]	wherein the storage device is 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 of the incoming data access requests; and
22[f]	wherein said circuitry is to, for each of the incoming data write requests:

22[f][i]	derive, from the accompanying address information, an associated first address portion, an associated second address portion and an associated third address portion,
22[f][ii]	identify an addressed block device from the associated first address portion,
22[f][iii]	identify an addressed segment within the addressed block device from the associated second address portion,
22[f][iv]	identify a write destination logical erase unit from the associated third address portion, and identify a physical erase unit which is mapped to the write destination logical erase unit,
22[f][v]	identify a physical storage location in which associated write data is to be stored and which is associated with the write destination logical erase unit, and an address value associated with the physical storage location in which the associated write data is to be stored; and
22[f][vi]	transmit information to a host representing the address value;
22[g]	wherein for one of the read requests which is directed to data which has been written into the flash memory pursuant to one of the write requests, at least one of the first address portion, the second address portion or the third address portion associated with the one of the read requests is dependent on the address value which has been transmitted to the host in association with the data which has been written into the flash memory pursuant to one of the write requests.
Claim 23	
23[pre]	A method of operation in a storage device having flash memory, the flash memory having physical erase units, and circuitry, the method comprising:
23[a]	receiving incoming data access requests with the circuitry, wherein each of the incoming data access requests is accompanied by address information;
23[b]	with the circuitry, processing the incoming address information to, for each of the incoming data access requests:

23[b][i]	derive a first address portion, a second address portion and a third address portion, identify an addressed block device from the first address portion,
23[b][ii]	identify an addressed segment within the addressed block device from the second address portion,
23[b][iii]	identify an addressed storage location within the addressed segment from the third address portion,
23[b][iv]	perform a division operation to identify a logical erase unit within the addressed segment, and
23[b][v]	identify one of the physical erase units that corresponds to the identified logical erase unit;
23[c]	wherein the method further comprises causing the storage device to detect a failure condition of the physical erase units, 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 to utilize the remapped correspondence to service one or more of the incoming data access requests.

I. INTRODUCTION

Samsung Electronics Co., Ltd. and Samsung Electronics America, Inc., (“Samsung”) requests IPR of claims 1-23 of U.S. Patent No. 11,347,656 (“’656 Patent”). A reasonable likelihood exists that Petitioner will prevail on at least one challenged claim.

The ’656 Patent relates to systems and methods for a flash memory controller with a logical-to-physical address translation. EX1001, FIG. 1. The host system (*e.g.*, PC) provides the controller a logical address. *Id.* The controller translates the logical address to a physical address that details the actual location in flash memory. *Id.* The ’656 Patent purports to disclose controller for a flash memory storage device with “negligible logical-to-physical address translation time,” where “address trans[lation] may be implemented entirely with logical gates and look-up tables of a memory controller integrated circuit, without requiring processor cycles.” EX1001, Abstract. The ’656 Patent allegedly achieves this improvement with “hierarchal address virtualization,” which means the logical address is translated to sub-addresses associated with virtual flash memory structures (*e.g.*, virtual erase unit and virtual page), and each virtual sub-address is translated to the physical address for the corresponding memory structure (*e.g.*, erase unit and page). *Id.*

The ’656 Patent represents common methods and techniques applied to address translation for flash memory. The claimed novelty, virtual hierarchal

address virtualization, was known in the art prior to the effective filing date of the '656 Patent. As an example, Reiter teaches a flash memory controller that translates a logical block address (hereinafter "LBA") from the host to a logical block number (*i.e.*, erase unit number) and logical page number, which are used to determine the physical erase unit number and physical page number. EX1004, 8:54-9:25, FIG. 4A.

The '656 Patent lacks any inventive concept and for the reasons herein is rendered obvious by Grounds I-V.

II. STANDING

Petitioner certifies under 37 C.F.R. §42.104(a) that the '656 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. EX1001, 4:10-35. Erase units are comprised of pages. *Id.* Pages represent the smallest unit that may be written. *Id.* To write the pages of a block with new data, the block must first be erased. *Id.*

To increase the efficiency of flash memory, SSDs may employ a larger unit of memory. For example, SSDs may be organized into groupings of erase units 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. *Id.*

IV. OVERVIEW OF THE '656 PATENT

A. Priority

The '656 Patent issued on May 31, 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 '656 Patent was filed on September 20, 2021.

The examiner issued a non-final rejection on January 13, 2022, rejecting claims 2-3, 8-9, 13-14 under 35 U.S.C. §112(b) and for non-statutory double patenting, and rejecting claim 2 under 35 U.S.C. §101. EX1002. On March 30, 2022, Applicant amended the claims to traverse the rejections. EX1002. On April 21, 2022, the examiner issued a Notice of Allowance, stating that the prior art of record did not disclose the claims second and third address portions and did not

disclose using division to obtain an identifier for a logical erase block. EX1002. The '656 Patent issued May 31, 2022.

C. Disclosure

The '656 Patent describes a logical-to-physical address translation scheme for a flash memory controller translating LBAs accompanying commands from a host. EX1001, Abstract, 2:66-3:20.

The host system sends a command, such as a read command, with address information to the controller. *Id.* The controller translates the address information to sub-addresses associated with virtual structures of the flash memory. EX1001, 2:66-3:21. And each virtual sub-address is translated to a physical sub-address in the underlying flash memory structure. *Id.* For instance, a logical address from the host contains several bit fields (or “sub-addresses”), including bit fields associated with virtual channel, die, erase unit, and page numbers. EX1001, 4:23-35, 15:55-65. The logical address is broken up into those bit fields. *Id.* Each virtual sub-address is translated by look-up tables to a physical sub-address of the same flash memory structure (*i.e.*, channel, die, erase unit, and page), thereby identifying the location in memory. EX1001, 15:37-65, 16:49-65. The '656 Patent refers to this scheme as “hierarchical address virtualization.” EX1001, Abstract.

V. GROUNDS¹²

Ground	Basis	Reference(s)	Claims
I	§103	Reiter, ZBC	1-12, 16, 20-23
II	§103	Reiter, ZBC, Sinclair-367	1, 13-14, 19
III	§103	Reiter, ZBC, Sinclair-367, Yamada	15, 17-18
IV	§103	Ban, ZBC, Sinclair-367	1-14, 16, 19-23
V	§103	Ban, ZBC, Sinclair-367, Yamada	15, 17-18

VI. LEVEL OF ORDINARY SKILL

A person of ordinary skill in the art (“POSITA”) at the time of the ’656 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, ¶¶13-16, 27-29. More education can supplement practical experience and vice versa. *Id.* Petitioner’s expert exceeded this skill level as of the priority date. *Id.*

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 ’656 Patent. In the co-pending litigation, Radian has objected to identifying secondary considerations for specific patents and believes it need 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

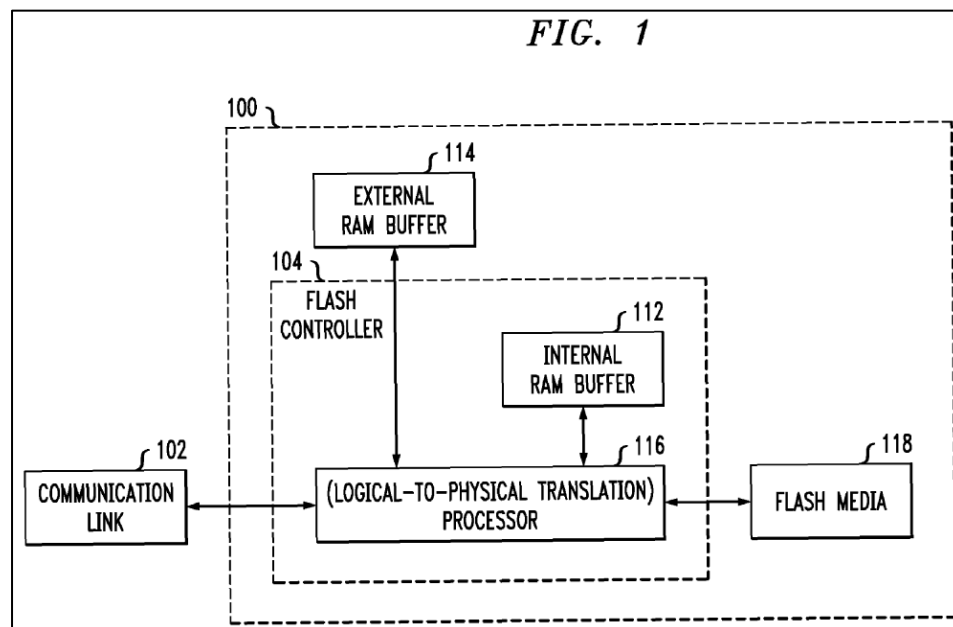
For the purposes of this proceeding, Petitioner contends that claim terms be given their plain and ordinary meaning under *Phillips*.

VIII. GROUND I: CLAIMS 1-12, 16, 20-23 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, AND SPC-4

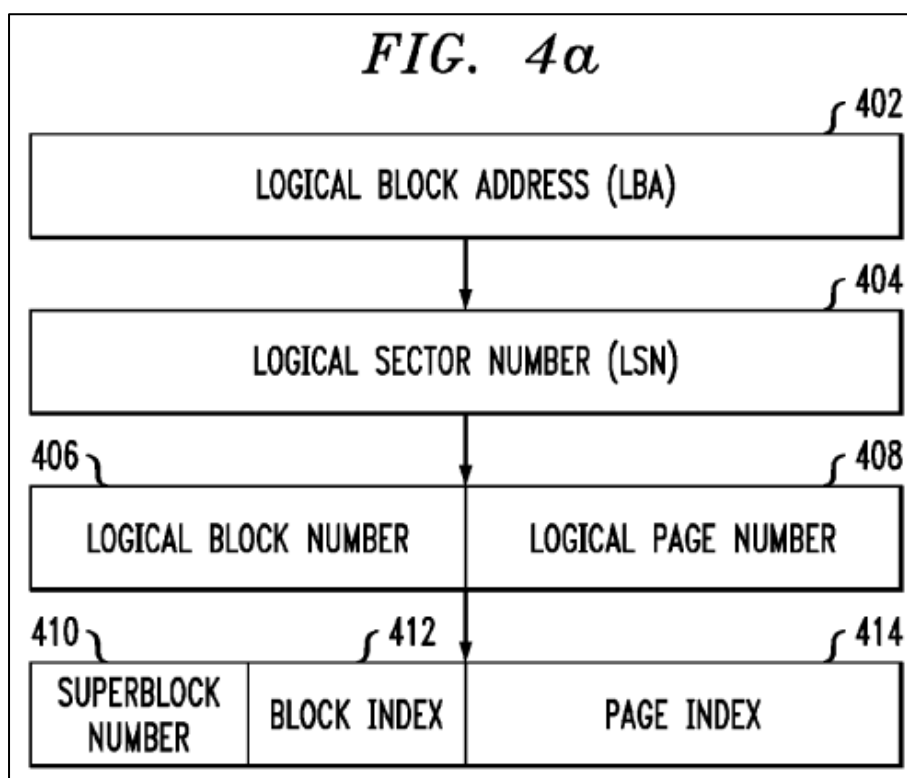
A. Reiter

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

Reiter discloses a flash system. EX1004, 6:63-66. The flash system uses a flash controller to receive and execute a host’s read/write commands sent over a communication link. EX1003, ¶¶52-54; EX1004, 8:1-6, 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 (1) an LBA, (2) logical sector number, (3) logical block number and logical page number, and (4) superblock number, block index, and page index. EX1004, 3:41-60, FIG. 4a:



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

ZBC, SBC-4, and SPC-4 (collectively, “T10 Standards”) are printed publications under 35 U.S.C. § 102. EX1003, ¶¶64-69. 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. 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; *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, ¶¶64-65. Dr. Litz was aware of T10 and the SCSI standards published by T10, and that T10 Standards would have been readily available upon publication on T10’s website. EX1003, ¶¶64-65. T10 “was well known to the community interested in the subject matter of” T10 Standards, and a POSITA “interested in” storage standards “would have been independently aware of [T10] as a prominent forum for discussing such technologies.” *Voter Verified, Inc. v. Premier Election Sols., Inc.*, 698 F.3d 1374, 1380 (Fed. Cir. 2012); EX1003, ¶¶64-65.

Moreover, a POSITA could have reasonably located T10 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. 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)." 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 T10 Standards on T10's website.

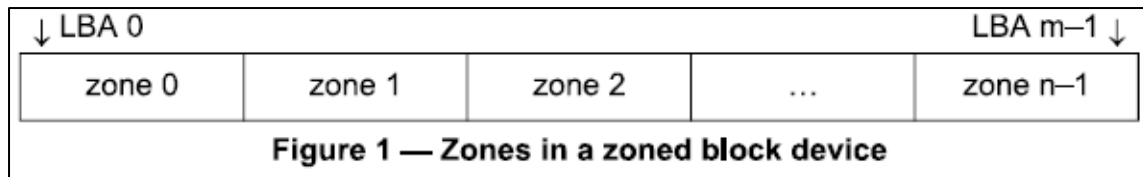
Even if T10 Standards' availability was limited to T10 members, that is no bar to publicly availability. The Board regularly finds documents published by standards bodies, including working drafts like T10 Standards, are publicly available. *See ZTE Corp v. IPR licensing, Inc.* (IPR2014-00525) (crediting 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 publicly accessible prior art). In *Ericsson Inc. v. Intellectual Ventures I LLC*,

IPR2014-00527, Paper 1 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, T10 Standards bear T10 copyright dates and/or otherwise indicate “publi[cation].” EX1006, iv; EX1007, iv; EX1008, iv.

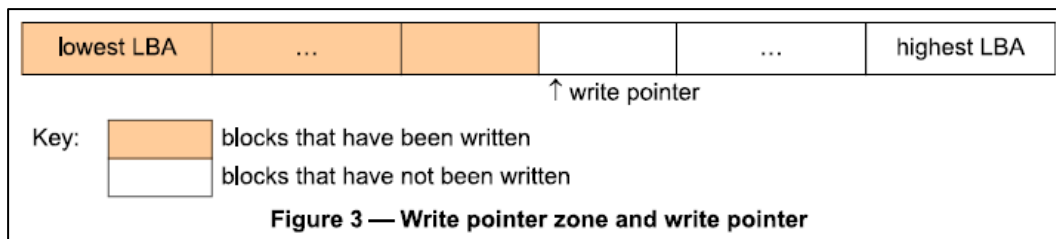
T10 Standards are authentic. EX1100, ¶¶1-17. Mr. Deglin personally received notice of publications and contemporaneously downloaded T10 Standards, and confirmed that the metadata matches the metadata listed on the T10 website as of the publication date. EX1100, ¶¶12-16. T10 Standards used in this Petition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

SCSI is a commonly used standard for storage, such as hard disk drives. EX1003, ¶65. ZBC is a 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, 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.

C. Motivation to Combine

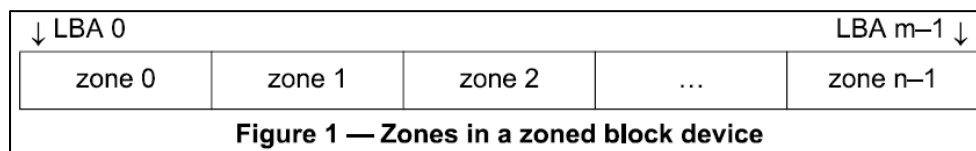
A POSITA would have been motivated to use ZBC, alongside SBC-4 and SPC-4, as the command set governing requests sent to Reiter's flash system. EX1003, ¶¶66-71.

As an initial matter, 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, ¶¶66-71. 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. *E.g.*, EX1006, 3-4, 12, 22 ("[C]ommands for host aware zoned block devices are defined in SBC-4.").

A POSITA would further be motivated to apply ZBC's teachings (including SBC-4 and SPC-4) to Reiter's flash system. EX1003, ¶¶70-76. 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.

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, ¶¶72-75.

In addition, commands based on ZBC’s zoned block device model (EX1006, 1) would enable more effective utilization of Reiter’s memory organization scheme. 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. 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. SCSI specifications were intended to be used across a wide range of storage devices, including flash memory. EX1003, ¶65. A POSITA would have also understood that Reiter’s superblock-based memory organization is suitable for zone-based commands like ZBC. EX1003, ¶¶72-76. 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. EX1003, ¶75.

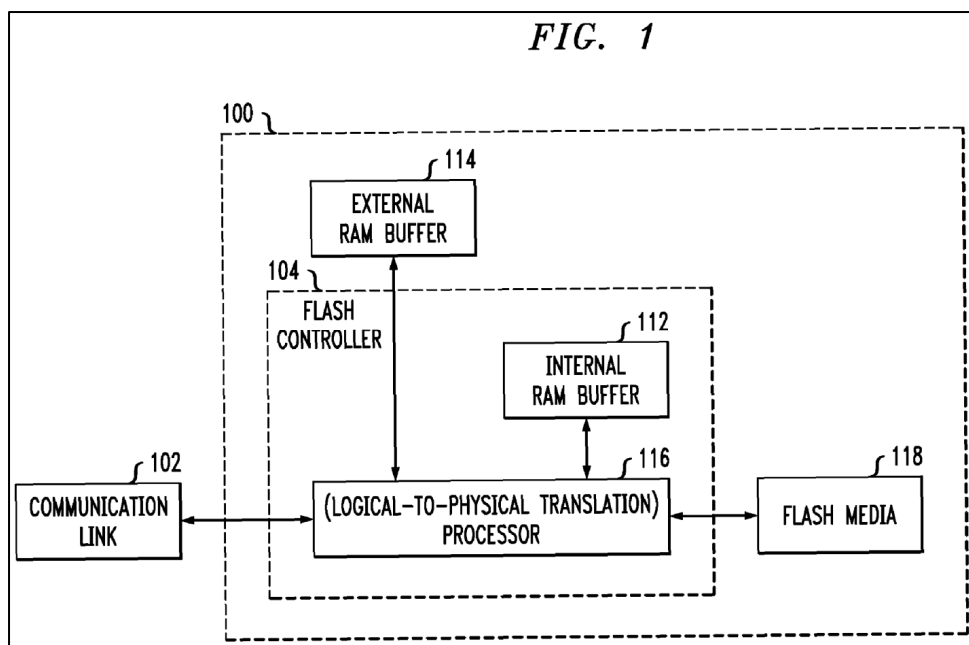
Reiter and ZBC, SBC-4, and SPC-4 are analogous art—relating to storage devices, including accessing blocks of memory. EX1003, ¶¶75-76.

D. Claim-by-Claim Analysis

1. Claim 1

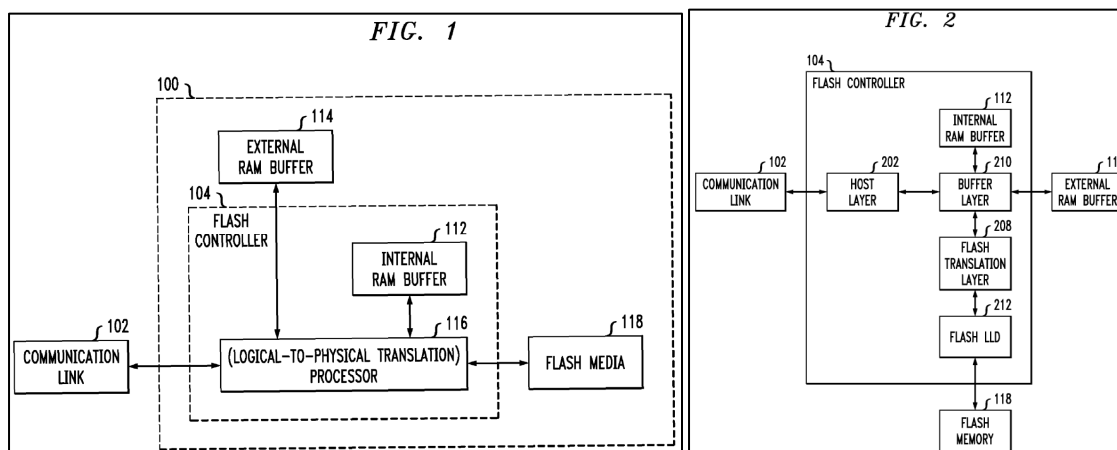
(a) 1[pre]

Reiter discloses a “flash... system 100” (*storage device*) “implementing a logical-to-physical translation” scheme. EX1004, 6:63-66, FIG. 1:



(b) 1[a]

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:

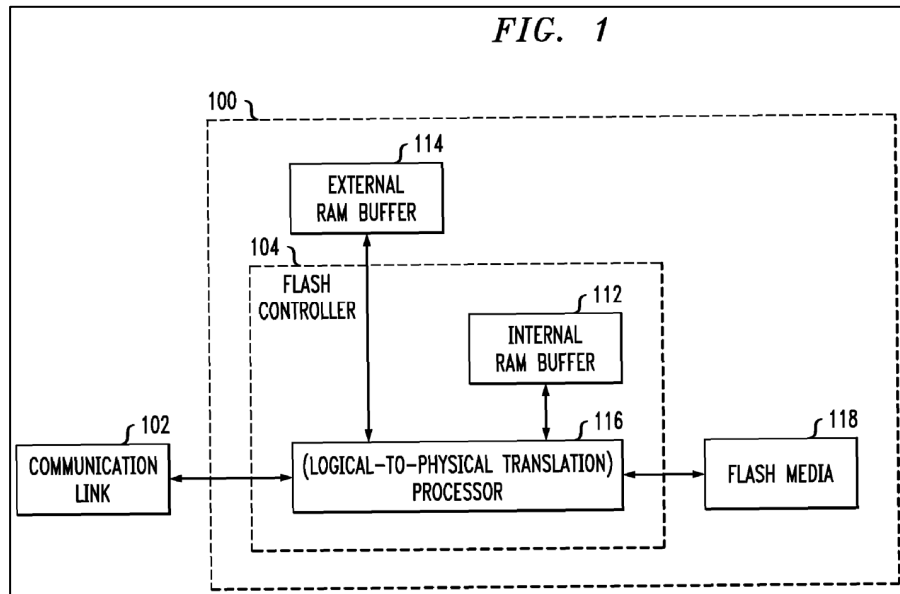


Reiter discloses that a "commonly employed type of flash memory technology is NAND flash memory." EX1004, 1:40-45. "There are three basic operations for NAND devices: read, write, and erase." EX1004, 1:51-52. Read and write

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. Thus, Reiter’s flash memory is made up of independently erasable blocks (*flash memory having physical erase units*). EX1003, ¶¶77-81.

(c) 1[b]

Reiter’s flash system 100 includes a “flash controller 104” (*circuitry*). EX1003, ¶¶82-86; EX1004, 6:63-7:2, FIG. 1:



“[F]lash controller 104 *receives*... requests[,] such as read or write operations, [(*incoming data access requests*)] from... external devices via communication link 102.” EX1004, 8:1-4. The requests “generally include a[]... LBA[] where data

should be read or written” (*incoming data access requests, accompanying address information*). EX1004, 8:4-6.

For example, Reiter discloses a “flash media read operation 530” that is “*executed* by FTL 208” in flash controller 104. EX1004, 11:55-57, FIG. 2. The “flash media read operation” is executed “in response to a flash media read request received from an external device,” which “includes a[n]... LBA[.]” EX1004, 11:55-57.

A POSITA would be motivated to implement Reiter’s flash controller 104 to accept and execute ZBC commands. Section VIII.C. ZBC includes a read command—a request disclosed in Reiter (EX1004, 8:1-6)—(*incoming data access requests*). EX1007, Table 63. A ZBC read command includes an LBA (*for each of the incoming data access requests, accompanying address information*)—also as disclosed with Reiter’s requests (EX1004, 8:1-6, 11:55-63, 13:3-46). EX1007, Table 63:

Table 63 — READ (32) command									
Byte	Bit	7	6	5	4	3	2	1	0
0		OPERATION CODE (7Fh)							
1		CONTROL							
2		Reserved							
...									
5									
6		Reserved		GROUP NUMBER					
7		ADDITIONAL CDB LENGTH (18h)							
8	(MSB)	SERVICE ACTION (0009h)							
9									(LSB)
10		RDPROTECT			DPO	FUA	RARC	Obsolete	Reserved
11		Reserved							
12	(MSB)	LOGICAL BLOCK ADDRESS							
...									
19									(LSB)
20	(MSB)	EXPECTED INITIAL LOGICAL BLOCK REFERENCE TAG							
...									(LSB)
23		EXPECTED LOGICAL BLOCK APPLICATION TAG							
24	(MSB)								(LSB)
25		LOGICAL BLOCK APPLICATION TAG MASK							
26	(MSB)								(LSB)
27		TRANSFER LENGTH							
28	(MSB)								
...									
31									(LSB)

(d) 1[b][i]

From a ZBC read command (*accompanying incoming address information*), flash controller 104 *derives* (1) a logical unit number (*first address portion*), (2) superblock number 410 (*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 (*third address portion*). EX1003, ¶¶87-96.

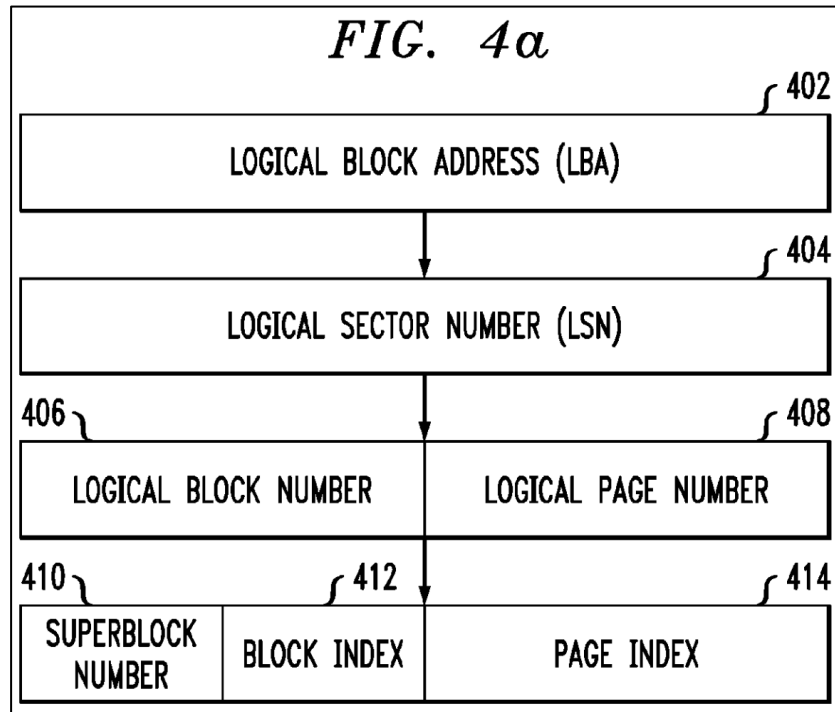
First Address Portion: A POSITA would be motivated to incorporate a logical unit number in ZBC commands, and motivated to modify flash controller 104 to decode (*derive*) that number (*first address portion*) from the command (*from*

the accompanying address information). EX1003, ¶¶88. ZBC read commands target a “logical unit”. EX1008, 14, 61; 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. EX1008, 14, 462. 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, ¶¶88-89. A POSITA would incorporate something else in commands to distinguish the same LBA to different logical units. *Id.*

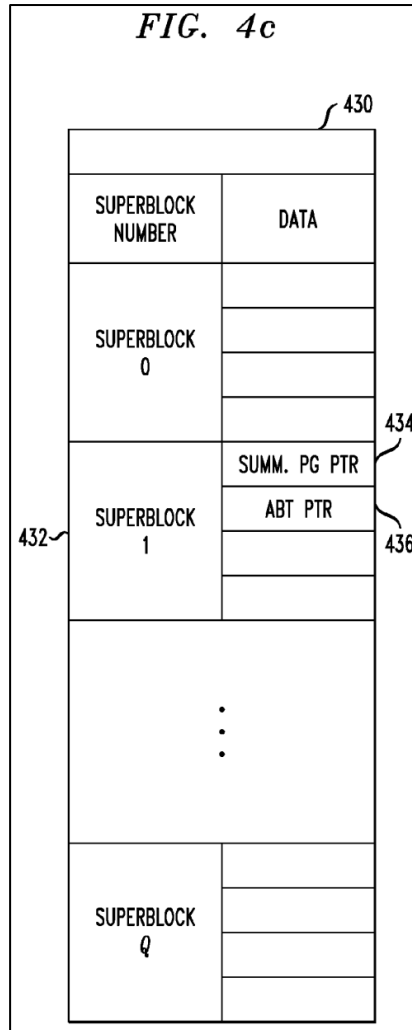
A POSITA would be motivated to incorporate the separate address found in SPC-4 into ZBC commands. Section VIII.B.1. SPC-4 recites that a “logical unit” is “[a]n **externally addressable** entity within a SCSI target device.” EX1008, 14. A logical unit is addressed by a “logical unit number”—“[a]n encoded 64 bit identifier for a logical unit.” EX1008, 14. ZBC already mentions logical unit number. EX1006, 4. A POSITA would be motivated to incorporate that SPC-4 address in ZBC commands. EX1003, ¶¶88-89.

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 read commands’ LBA (*from the accompanying address information*). ZBC

read commands include an LBA. EX1007, Table 63. 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.” EX1004, 8:56-59, FIG. 4a:



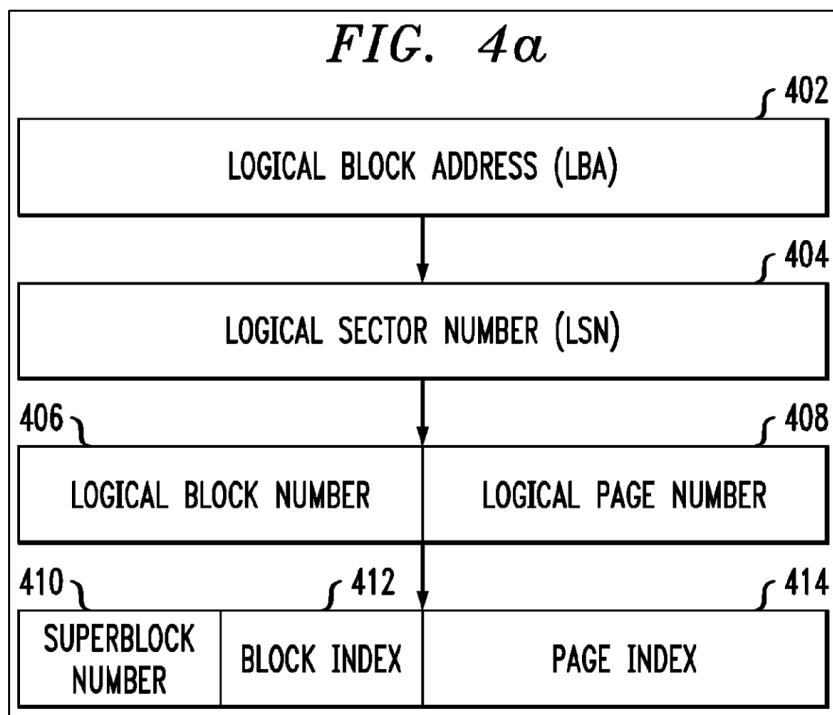
Translating an LBA results in a bitstring of superblock number 410, block index 412, and page index 414. EX1004, 8:66-9:25, FIG. 4a. Superblock number 410 is identified in the page global directory (PGD) 430 to obtain a “pointer 434... to the summary page associated with [the corresponding] Superblock.” 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 read commands’ LBA (*from the accompanying address information*). A ZBC read command includes an LBA. EX1007, Table 63. Reiter discloses that read and write “request[s]... include[] a... LBA... .” EX1004, 8:4-6, 8:62-67. “FIG. 4a shows an

exemplary relation of a[n]... LBA... to a physical address.” EX1004, 8:56-59, FIG.

4a:



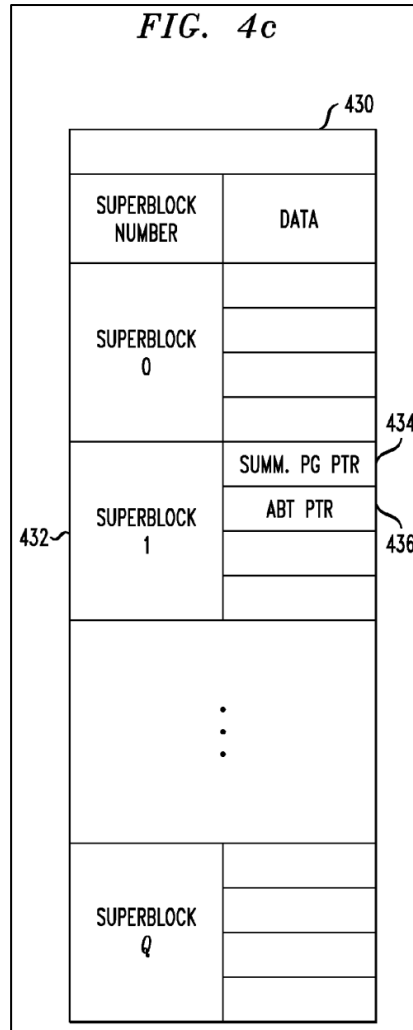
“LBA 402 corresponds to a[]... logical sector number (LSN) 404.” EX1004, 8:66-67. Logical sector number 404 is translated to a bitstring with logical block number 406 and logical page number 408 bitfields. EX1004, FIG. 4a; EX1003, ¶¶92-95. Logical block number 406 is obtained by “dividing LSN 404 by 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]

The logical unit number (*from the first address portion*) (Section VIII.D.1.d) *identif[ies]* the targeted logical unit (*addressed block device*). EX1003, ¶¶97-98. A logical unit has an independent range of LBAs starting at zero. EX1008, 61; EX1001, 7:54-63, FIG. 5.

(f) 1[b][iii]

Superblock number 410 (*from the second address portion*) *identif[ies]* the superblock, which a POSITA would be motivated implement as a ZBC zone (*addressed segment*). Section VIII.C; *see also* EX1011, 22. Superblock number 410 identifies from page global directory (PGD) 430 the “summary page associated with [the corresponding] Superblock.” EX1004, 10:13-19, FIG. 4c:

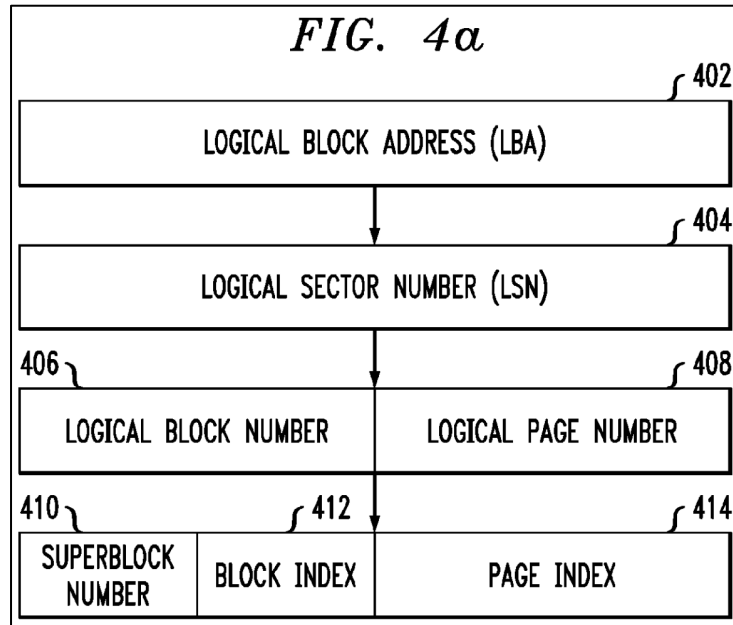


Superblock number 410 is derived from LBA 402, (EX1004, FIG. 4a) which is from a ZBC command's LBA (Section VIII.D.1.d). Because a ZBC LBA is within the LBA range specified by the targeted logical unit, (*see* Sections VIII.D.1.d-e; EX1006, x, 1, 4; EX1008, 61) superblock number 410 identifies a superblock ***within*** the targeted logical unit (***addressed block device***).

(g) 1[b][iv]

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:



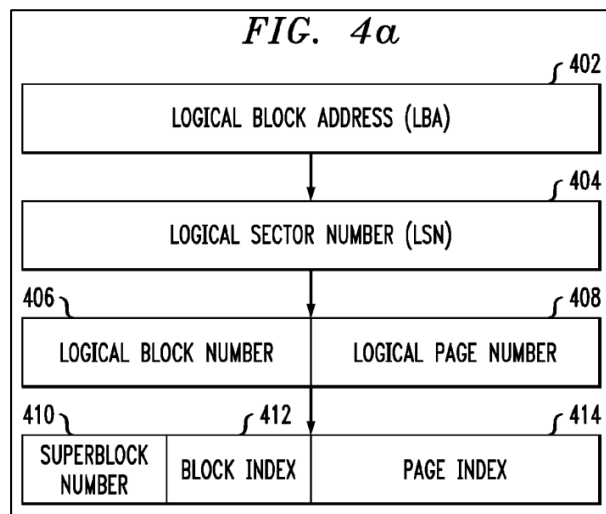
These indexes *identify* a logical block and/or logical page (*addressed storage location*) *within* the superblock (*i.e.*, zone (*addressed segment*) (Section VIII.D.1.f)) addressed by superblock number 410. EX1004, 9:52-63.

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:

(h) 1[c]

Reiter discloses flash controller 104 (*circuitry*) performing integer division on logical sector number 404 to *identify* logical block number 406 (*logical erase unit*). EX1003, ¶¶110-12,118-20. 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. EX1003, ¶¶110,113-15,118-20. 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. EX1003, ¶¶110,116-20.

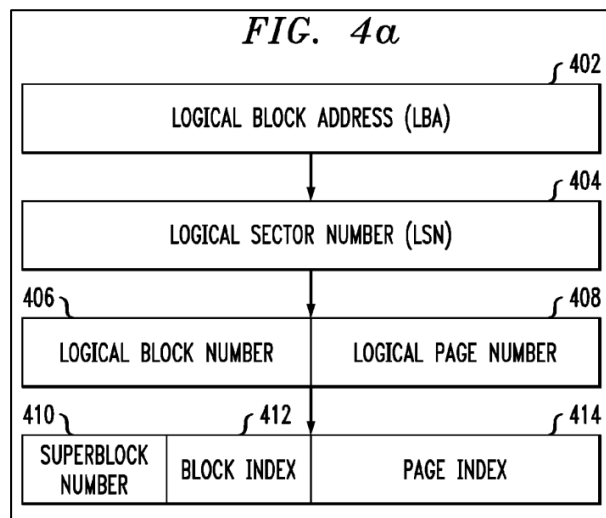
Integer Division On Logical Sector Number 404: Reiter discloses that “[e]ach LBA can be divided into a logical block number and logical page number.” EX1004, 3:48-50, 9:5-20. LBA 402 is first translated to logical sector number 404. EX1004, 8:66-67, FIG. 4a:



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.

Alternative, Integer Division To Subdivide Out Logical Block Number

406 Bitfield: Alternatively, after the second step, a new bit string is formed including logical block number 406 and logical page number 408. EX1004, FIG. 4a:

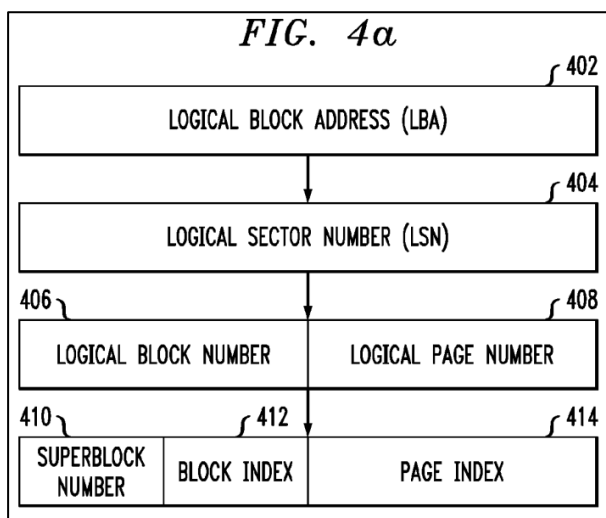


And a POSITA would be motivated to divide by a power-of-two, splitting the bitstring into bitfields. EX1003, ¶¶114-15. Integer division by a power-of-two extracts bitfields out of bitstrings. *Id.* The quotient is the most significant bitfields (*i.e.*, bits on the lefthand portion of a bitstring). *Id.* Integer division is fast, because it can be implemented with a bit shift operation. *Id.* Logical block number 406 is the higher order or most significant bitfield of the bitstring. EX1004, FIG. 4a; *Id.* Integer division causes logical block number 406 to be extracted through the

quotient. *Id.* A POSITA would be motivated to implement integer division via a bit shift operation to perform the extraction quickly. *Id.*

Alternative, Integer Division To Subdivide Out Block Index 412 Bitfield:

Alternatively, after translating logical block number 406 and logical page number 408, a new bit string is formed including superblock number 410, block index 412, and page index 414. EX1004, FIG. 4a:



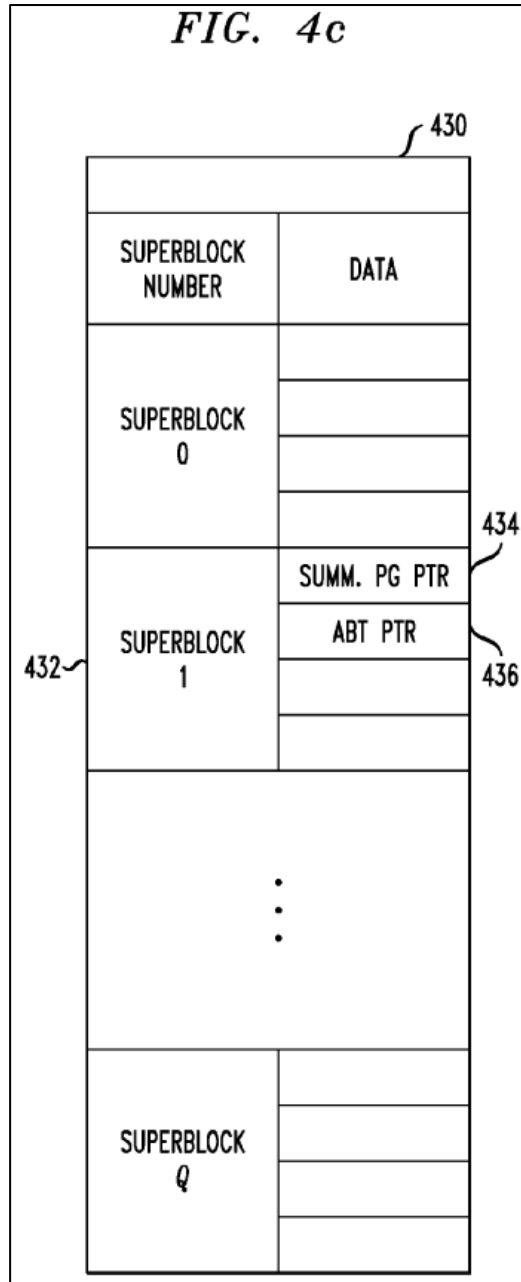
A POSITA would be motivated for the sake of speed to divide by a power-of-two to split the bitstring into the corresponding bitfields. EX1003, ¶¶114-17.

Logical Erase Unit: The resulting logical block number 406, or alternatively block index 412, is a *logical erase unit*. The mapping is explicit in the name—“**logical block** number” and “**block index**.” **Logical** block number 406 and block **index** 412 are logical units. 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). Both map to a physical block. “LBN 406... corresponds to Superblock number 410 and block index 412.” EX1004, 9:5-6. “[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.

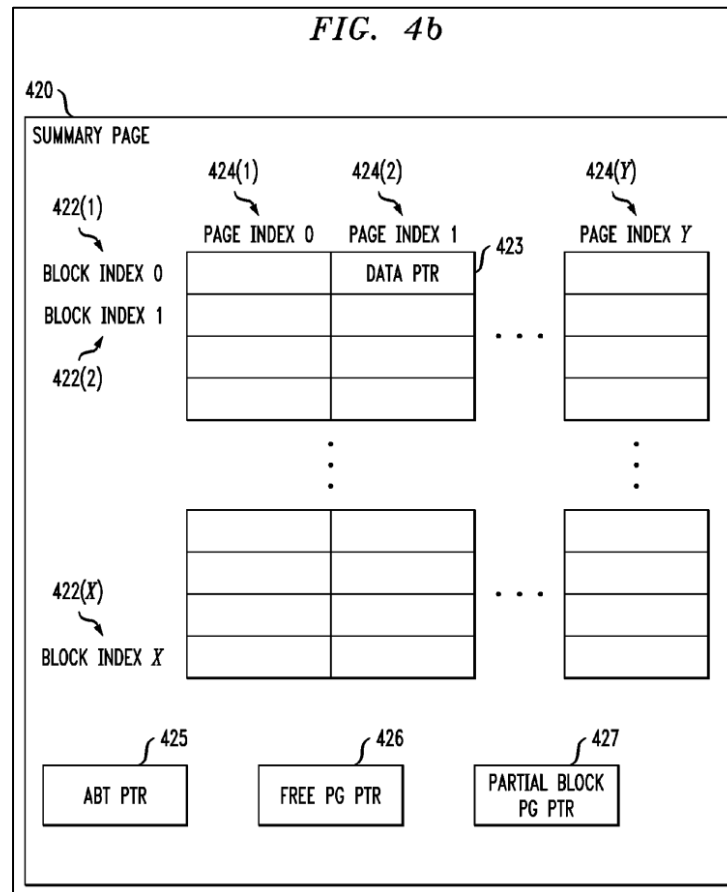
(i) 1[d]

Reiter discloses flash controller 104 (*circuitry*) translating logical block number 406, and alternatively block index 412, (*identified logical erase unit*) (Section VIII.D.1.h) to a physical block number (*identify one of the physical erase units that corresponds to the identified logical erase unit*). EX1003, ¶121. “LBN 406... corresponds to Superblock number 410 and block index 412.” EX1004, 9:5-6, FIG. 4a. Superblock number 410 selects the summary page “contain[ing]... mapping data for the associated Superblock.” EX1004, 9:52-54, FIG. 4c:



“[A] Superblock generally is a logical collection of blocks,” (EX1004, 9:11-17.) where the “summary page 420 contains... block indices 422(1)-422(X) and page indices 424(1)-424(Y) for all X blocks and all Y pages in [the s]uperblock.” EX1004, 9:54-58, FIG. 4b. Each pair of “block indices 422(1)-424(Y) and page

indices 424(1)-424(Y) are stored as data pointer 423, which points to the physical address.” EX1004, 9:58-63, FIG. 4b:



Block index 412 with page index 414 selects one of those data pointers 423. EX1004, 9:11-17, 9:52-63, FIG. 4b. Data pointer 423 points to the underlying, selected physical block, as mentioned previously. EX1004, 9:52-63, 10:1-3, FIG. 4b; *see also* EX1004, 2:59-62, 3:41-55 (listing various translation schemes translating an LBA to physical block number and page number, such as a “superblock mapping scheme,” which includes translating an “LBA” to a “logical block number and... page number,” turning logical block number into a “superblock

number and a PGD index offset,” turning logical page number into “a PMD index offset and PT index offset,” and translating superblock number, PGD index offset, PMD index offset, and PT index offset into a “physical block number and physical page number”). Because the underlying physical block corresponds to superblock index 412 and block index 414, which were derived from logical block number 406, flash controller’s 104 translation *identifies one of the physical erase units that corresponds to* logical block number 406, or alternatively block index 414 (*identified logical erase unit*).

(j) 1[e]

Reiter’s flash storage system 100 (*storage device*) *detects* bad blocks (*failure condition of the physical erase units*). Upon detecting a bad block, a POSITA would be motivated to *remap* logical block number 406, and block index 412, (*logical erase unit*) from the bad block (*one of the physical erase units for which the failure condition is detected*) to *instead* a working block (*one of the physical erase units for which the failure condition has not been detected*). EX1003, ¶¶123-29.

Detect A Failure Condition Of The Physical Erase Unit: Reiter discloses “maintain[ing] a record of bad blocks” by scanning for bad blocks after powering on flash system 100. EX1004, 4:27-37, 17:13-15, 17:47-63. For example, “[o]n startup of flash... system 100, mapping data... requires reconstruction.” EX1004, 17:13-

15. This includes “scan[ning] and group[ing]... blocks” into various “categories,” including “bad blocks.” EX1004, 17:47-52. Bad blocks are detected if (1) FTL 208 is “unsuccessful” “read[in]g metadata that is stored in the... first page of the block” and (2) the following “erase operation of the [same] block was unsuccessful, the block cannot be erased and has failed.” EX1004, 17:59-67. “[T]he block address is [then] added to the bad block list.” EX1004, 18:4-8.

Remap Each Logical Erase Unit Which Corresponds To One Of The Physical Erase Units For Which The Failure Condition Is Detected: 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. Data from a bad block cannot be trusted, especially given the unsuccessful read during the startup scan and subsequent failed erasure of the block. 4:27-37, 17:13-15, 17:47-63. Keeping a bad block around risks reading erroneous data. EX1003, ¶126. As a first remapping step, Reiter discloses, upon detecting a bad block, adding the “block address... to the bad block list.” EX1004, 18:4-8.

A POSITA would be motivated to reconstruct the lost data through Reiter’s RAID functionalities, and change data pointer 423 from the old bad block to a new working block with the reconstructed data. EX1003, ¶¶127-28. Reiter discloses configuring flash system 100 “analogously to hard drives in a redundant array of

independent disks (RAID).” EX1004, 8:46-50. RAID allows data lost on one disk to be reconstructed with data from the other redundant disks. EX1003, ¶127.

Because the previous block is an unreliable bad block, a POSITA would understand that a new block is needed for the reconstructed data. *Id.* 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. *Id.* This allows the LBA addressing, including logical block number 406 and block index 412, to be unchanged despite the data loss. With the new working block, data at logical block number 406 and block index 412 is trustworthy and usable for later reads (*utilize the remapped correspondence to service one or more of the incoming data access requests*). *Id.*; see EX1004, 17:13-21.

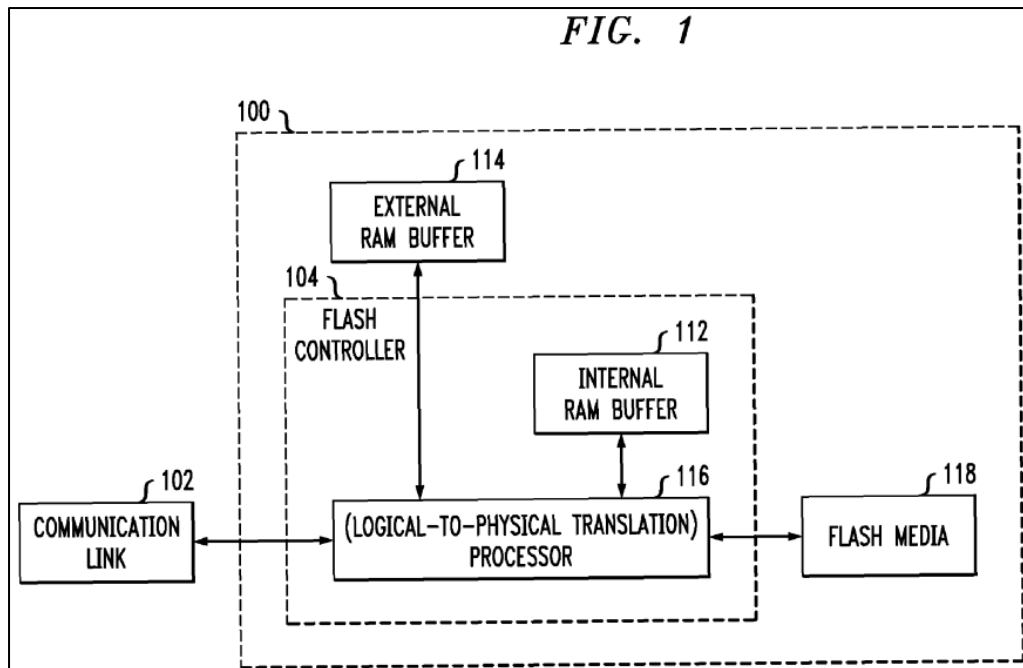
2. Claim 2

(a) 2[pre]

Sections VIII.D.1, IX.C.1.

(b) 2[a]

Reiter’s flash controller 104 (*said circuitry*) *comprises* “processor 116” (*processor*). EX1004, 7:21-22, 7:31-39, FIG. 1:



(c) 2[b]

Reiter's flash system 100 (*storage device*) includes flash controller 104, which includes processor 116, and (EX1004, 7:19-39, FIG. 1) and “[p]rocessor 116 includes software and/or firmware as needed for operation” (*said instructions when executed control the at least one processor*), which a POSITA would understand as including *instructions stored on at least one physical storage medium*. EX1004, 7:31-34; EX1003, ¶¶133-34.

3. Claim 3

(a) 3[pre]

Sections VIII.D.1, IX.C.1.

(b) 3[a]

Section VIII.D.1.c.

(c) 3[b]

A POSITA would be motivated to implement Reiter's flash system 100 (*storage device*) to *receive* and execute ZBC commands (*from the host*). Section VIII.D.1.c. ZBC includes a write command. 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		(LSB)							
26	(MSB)	LOGICAL BLOCK APPLICATION TAG MASK							
27									
28	(MSB)	TRANSFER LENGTH							
...									
31		(LSB)							

Flash system's 100 translation of ZBC read command's LBA applies to a ZBC write command, thereby disclosing this limitation. EX1003, ¶¶135-39; EX1004, 8:54-56; Sections VIII.D.1.d-g.

(d) 3[c]-[d]

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 (*in association with fulfillment of the received write request*), except the resulting pointer is called the "active page address" (*identify a physical storage location in which associated write data has been stored and which is associated with the corresponding logical erase unit*). EX1003, ¶140; EX1004, 8:54-56, 13:55-60, FIG. 6b; Sections VIII.D.1.h-i.

ZBC write commands also cause flash system 100 (*storage drive, in association with fulfillment of the received write request*) to transmit information back to the host the upon command completion. "The SCSI command set assumes an underlying request-response protocol." EX1008, xl. The request is "submitted by an application client, conveying a SCSI command." EX1008, 11. The response is "returned... on completion of a SCSI command." EX1008, 11.

A POSITA would be motivated to modify the transmission to include information identifying the completed command (*information representing an address value to the host corresponding to the identified physical storage location*). EX1003, ¶¶141-45. Reiter discloses processing multiple commands in parallel. EX1004, 8:50-53, 15:55-64, 16:62-65, FIG. 3. SPC-4 also discloses processing multiple commands in parallel. EX1008, xl, 431, 438, 472, 821, 867, Table A.1;

Section VIII.D.3.d. Because concurrent commands can complete out of order, information is needed in the response to identify what command completed. EX1003, ¶142. 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 to the host corresponding to the identified storage location.*** *Id.*

Moreover, ZBC discloses incorporating such information in some circumstances. If a ZBC write command writes across two or more zones, the zoned block device “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.

A POSITA would be motivated to restrict subsequent read commands (***for one of the read requests***) targeting LBAs subject to pending write operations (***for one of the read requests which is directed to data which has been written into the flash memory pursuant to the received write request***) until after receiving a completion response of the pending write operations (***dependent on the address value which has been transmitted to the host in association with the data which has been written into the flash memory pursuant to the received write request***). EX1003, ¶144. Incorporating command-identifying information into the SCSI

response ensures command completion. *Id.* 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 (*address value which has been transmitted to the host in association with the data which has been written into the flash memory pursuant to the received write request*). The underlying *third address portion* is translated from the LBA, (Section VIII.D.1.d). and is therefore also *dependent on* the completion response.

4. Claim 4

(a) 4[pre]

Sections VIII.D.1, IX.C.1.

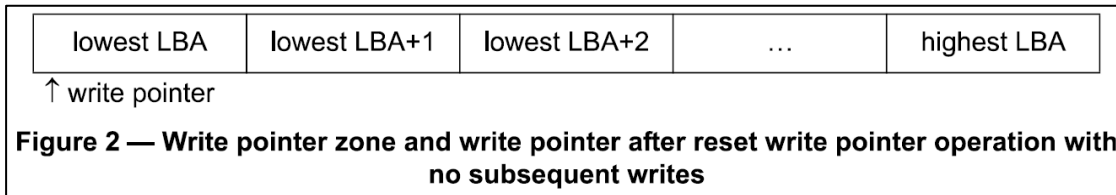
(b) 4[a]

A POSITA would be motivated to implement Reiter's flash system 100 (*storage device*) to *receive* and execute ZBC commands. Section VIII.D.1.c. ZBC includes a reset write pointer command (*maintenance request from a host*). 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							

The reset write pointer command resets the write pointer to the zone’s starting LBA.

EX1006, 14, Figure 2:



Like a ZBC read command, a POSITA would be motivated to implement the reset write pointer command to *address* a logical unit (*block device*). Section VIII.D.1.c; EX1008, 14, 61; EX1006, x, 1, 4, 29. Also like a ZBC read command, the reset write pointer command *addresses* a zone (*segment*) via a “ZONE ID”—parameter “specif[ying] the zone start LBA”—(EX1006, 29, Table 16) which is *within* the logical unit (*within the block device addressed*). Section VIII.D.1.f; EX1003, ¶¶146-50.

(c) 4[b]

A POSITA would be motivated to implement the reset write pointer command (*in association with execution of the maintenance request*) to cause Reiter's flash system 100 (*storage device*) to erase the physical blocks (*physical erase units*) associated with the targeted zone (*mapped to the segment addressed by the maintenance request*). EX1003, ¶¶151-53. After applying a reset write pointer command, the zone's data is no longer accessible. EX1006, 16-17. "A read operation on an LBA that has not been written since the last reset write pointer operation... return[s]... the last initialization pattern... set by... the manufacturer[]." EX1006, 16-17.

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 with less risk of error. EX1003, ¶152. Deferred execution risks an inappropriate order of operations, and requires complex management to avoid an inappropriate ordering. *Id.* Alternatively, immediate queueing for erasure provides flexibility for erasing when read/write operations are not being processed. *Id.* Reiter expressly complains about erasure delaying normal operations. EX1004, 4:19-26.

5. Claim 5

(a) 5[pre]

Sections VIII.D.1, IX.C.1.

(b) 5[a]

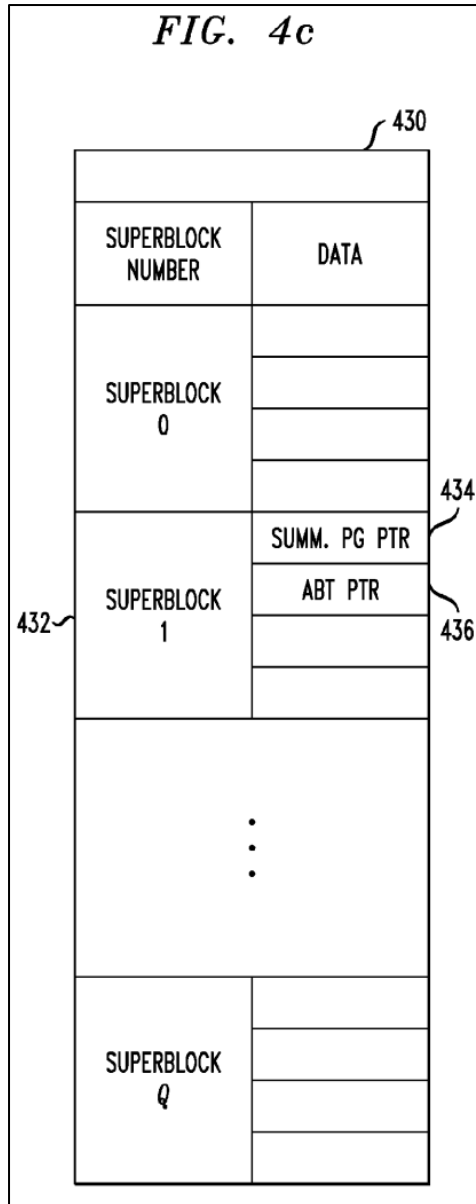
A POSITA would understand that a ZBC read commands targets one of the logical units (*addressed block device*) on flash media 118 (*flash memory is to have block devices*). Section VIII.D.1.d.

(c) 5[b]

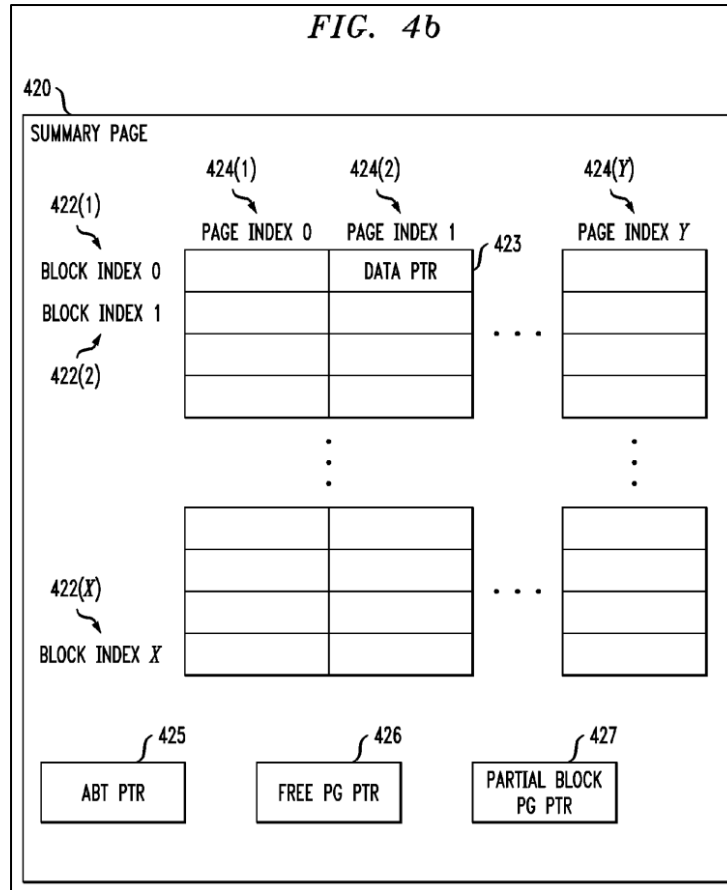
Sections VIII.D.5.d.

(d) 5[b][i]-[ii]

Reiter discloses page global directory 430 including a summary page pointer 434 for each superblock, where the summary page pointer is selected by superblock index 410. EX1004, 10:13-19, FIG. 4c. The “summary page associated with [the] Superblock” is identified and retrieved with summary page pointer 434 (*identify and retrieve a look-up table*). EX1004, 10:13-19, FIG. 4c:



The “summary page contains mapping data for the... Superblock” (*look-up table*). EX1004, 9:52-54. Block index 412 and page index 414 are used to select from summary page’s “block indices 422(1)-422(X) and page indices 424(1)-424(Y)” (*dependent on the look-up table*) “data pointer 423, which points to the [corresponding] physical address” (*identify a physical storage location in the flash memory*). EX1004, 9:60-63, FIG. 4b:



Summary pages are *specific* to the targeted logical unit identified by the logical unit number of the ZBC read command (*addressed block device which was identified from the first address portion derived from the given one of the incoming access requests*). EX1003, ¶¶154-61. Each logical unit has its own zones. EX1006, 1, 13. Because a POSITA would be motivated to use Reiter’s superblock as a ZBC zone, (Sections VIII.C, VIII.D.1.f) each logical unit has its own superblocks. And “[e]ach Superblock has a summary page.” EX1004, 9:52-53.

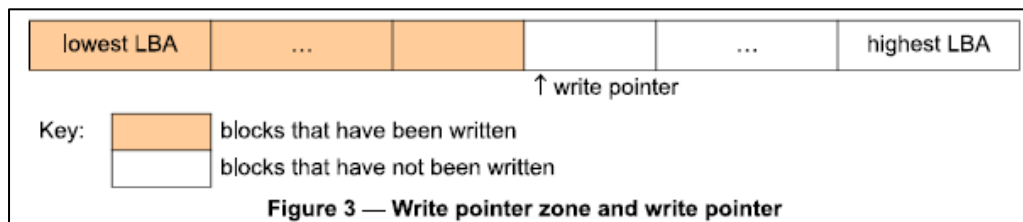
6. Claim 6

(a) 6[pre]

Sections VIII.D.1, IX.C.1.

(b) 6[a]

Reiter's flash system's 100 (*storage device*) flash controller 104 (*circuitry to*) maintains a free page pointer, or alternatively partial block pointer, (*track metadata*) *for* each superblock (*addressed segment*). EX1003, ¶¶162-65; EX1004, 9:52-10:1, FIG. 4b. The next free page pointer points to the “next free page... within the Superblock, and 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, 9:62-10:1. Both pointers are akin to a ZBC write pointer generally tracking the next free LBA. Section VIII.1.C; EX1006, 14, 17, Figure 3:



Patent Owner's infringement contentions asserts the same, alleging that a zone's write pointer discloses the limitation. EX1011, 63.

(c) 6[b]

A POSITA would be motivated to implement Reiter’s flash system 100 (*storage device*) to execute ZBC commands. Section VIII.D.1.c. ZBC includes a report zones command. EX1006, Table 12. The report zones command “requests that the device server return the zone structure of the zoned block device”—called “parameter data”—(*transmit information to a host*). EX1006, 24. This 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								
4	Reserved							SAME
5	Reserved							
...								
63								
Zone descriptors list								
64	Zone descriptor [first]							
...								
127								
	⋮							
n–63	Zone descriptor [last]							
...								
n								

The zone descriptors include the zone’s write pointer (*dependent on said tracked metadata*). EX1006, Table 13.

Table 13 — Zone descriptor format								
Bit Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2	Reserved							
...								
7								
8	(MSB)	ZONE LENGTH						
...								
15		(LSB)						
16	(MSB)	ZONE START LBA						
...								
23		(LSB)						
24	(MSB)	WRITE POINTER LBA						
...								
31		(LSB)						
32	Reserved							
...								
63								

Thus, the reported data is *dependent on* the write pointer (*tracked metadata*).

Patent Owner’s infringement contentions asserts the same mapping. EX1011, 65. The NVMe Zoned Namespace Specification is “able to transmit a Report Zones data structure that includes a Zone Descriptor to the host, which includes the write pointer (metadata).” EX1011, 65.

7. Claim 7

(a) 7[pre]

Sections VIII.D.1, IX.C.1.

(b) 7[a]

Section VIII.D.3.b.

(c) 7[b]

Section VIII.D.3.c-d.

(d) 7[c]

Section VIII.D.3.d.

(e) 7[d]

Section VIII.D.3.d.

8. Claim 8

(a) 8[pre]

Sections VIII.D.1, IX.C.1.

(b) 8[a]

A POSITA would be motivated to implement Reiter's flash system 100 (*storage device*) to *receive* and execute ZBC commands. Section VIII.D.1.c. ZBC includes an extended copy command (*request from a host to copy*). EX1006, 1, 22; EX1007, Table 33; EX1008, 287-89, Table 128. The extended copy command "cop[ies] data from one set of copy sources (e.g.,... logical units) to a set of copy destinations (e.g.,... logical units)" on flash system 100 (*copy host-selected data from a first storage location in flash memory to a second storage location in flash memory*). EX1008, 287-89. Specifically, the extended copy command includes a parameter list, which has segment descriptors. EX1008, Table 128:

Table 128 — EXTENDED COPY(LID4) parameter list (part 1 of 2)								
Bit Byte	7	6	5	4	3	2	1	0
0	PARAMETER LIST FORMAT (01h)							
1	Reserved		STR	LIST ID USAGE		PRIORITY		
2	(MSB) _____							
3	HEADER CSCD DESCRIPTOR LIST LENGTH (0020h)							(LSB)
4	_____							
...	Reserved							_____
14	_____							
15	Reserved					G_SENSE		IMMED
16	HEADER CSCD DESCRIPTOR TYPE CODE (FFh)							
17	_____							
18	Reserved							_____
19	_____							
20	(MSB) _____							
...	LIST IDENTIFIER							_____
23	_____							(LSB)
24	_____							
...	Reserved							_____
41	_____							
42	(MSB) _____							
43	CSCD DESCRIPTOR LIST LENGTH (n-47)							(LSB)
44	(MSB) _____							
45	SEGMENT DESCRIPTOR LIST LENGTH (m-n)							(LSB)
46	(MSB) _____							
47	INLINE DATA LENGTH (k-m)							(LSB)
CSCD descriptor list								
48	_____							
...	CSCD descriptor [ID 1] (see 6.4.5)							_____
79	_____							
⋮								
n-31	_____							
...	CSCD descriptor [ID x] (see 6.4.5)							_____
n	_____							

Table 128 — EXTENDED COPY(LID4) parameter list (part 2 of 2)								
Bit Byte	7	6	5	4	3	2	1	0
	Segment descriptor list							
n+1	Segment descriptor (see 6.4.6) [first]							
...								
	⋮							
...	Segment descriptor (see 6.4.6) [last]							
m								
m+1	Inline data							
...								
k								

The segment descriptors detail the source LBA (*first storage location*) and destination LBA (*second storage location*). EX1008, Table 153:

Table 153 — Block device to block device segment descriptor													
Bit Byte	7	6	5	4	3	2	1	0					
0	DESCRIPTOR TYPE CODE (02h or 0Dh)												
1	Reserved						DC	CAT					
2	(MSB)	DESCRIPTOR LENGTH (0018h)						(LSB)					
3													
4	(MSB)	SOURCE CSCD DESCRIPTOR ID						(LSB)					
5													
6	(MSB)	DESTINATION CSCD DESCRIPTOR ID						(LSB)					
7													
8	Reserved												
9	Reserved												
10	(MSB)	BLOCK DEVICE NUMBER OF BLOCKS						(LSB)					
11													
12	(MSB)	SOURCE BLOCK DEVICE LOGICAL BLOCK ADDRESS						(LSB)					
...													
19		DESTINATION BLOCK DEVICE LOGICAL BLOCK ADDRESS						(LSB)					
20	(MSB)												
...								(LSB)					
27													

A POSITA would understand that executing the extended copy command *(in response to the request from the host to copy)* includes flash controller 104 *(circuitry) to issue commands reading* data from the source LBA *(host-selected data from the first storage location in the flash memory)* and *writing* the data to the destination LBA *(host-selected data to the second storage location in the flash memory)*. EX1003, ¶¶177-82; EX1008, 245, 270.

SPC-4 discloses executing the extended copy command only via a “copy manager,” which “may be contained in... a standalone SCSI target device”—Reiter’s flash system 100 *(without sending the data being copied to the host)*. EX1003, ¶181; EX1008, 245-46.

(c) 8[b]

SPC-4 discloses that the extended copy command from the host (*the request from the host to copy*) includes a destination LBA (*designate destination address information corresponding to the second storage location*). Section VIII.D.8.b; EX1003, ¶¶183-84. A POSITA would be motivated to perform the same address translation as provided to ZBC write commands, thereby disclosing this limitation. Section VIII.D.3.c.

9. Claim 9

(a) 9[pre]

Section VIII.D.8.

(b) 9[a]

A POSITA would be motivated to implement Reiter's flash system 100 (*storage device*) to perform the same address translation on the LBA of a ZBC extended copy command (*from the destination address information*) to *identify* logical block number 406 and block index 412 (*destination logical erase unit*). Section VIII.D.8.b; *see* EX1003, ¶¶185-87. Further, logical block number 406 and block index 412 (*in dependence on the identified destination logical erase unit*) *selects* data write pointer 423 or active page address (*the second storage location*). EX1004, FIGs. 4a-b.

10. Claim 10

(a) 10[pre]

Sections VIII.D.1, IX.C.1.

(b) 10[a]

ZBC's report zones command optionally "return[s]" (*transmit*) a "[l]ist of zones in the zoned block device" back *to [the] host* via "parameter data" that includes the length of the zone list and zone descriptors indicating the starting LBA and length of each zone (*information identifying the number of structures to a host*). EX1006, 25, Table 11, Tables 12, 13:

Table 12 — REPORT ZONES parameter data								
Bit Byte	7	6	5	4	3	2	1	0
1	(MSB)							
...	ZONE LIST LENGTH (n-63)							
3	(LSB)							
4	Reserved							SAME
5	Reserved							
...								
63								
Zone descriptors list								
64	Zone descriptor [first]							
...								
127								
	⋮							
n-63	Zone descriptor [last]							
...								
n								

Table 13 — Zone descriptor format								
Bit Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2	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								

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.

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.

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" (*quantum of flash memory selected by the commands which are issued in response to the at least one request is dependent on the information transmitted to the host identifying the number of structures*). EX1003, ¶188-92; EX1006, 3.

11. Claim 11

(a) 11[pre]

Section VIII.D.10.

(b) 11[a]

The list of zones reported by the report zones command (*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*). EX1003, ¶¶193-96 Specifically, the returned zone descriptors indicate the length of each zone. EX1006, 25, Tables 11-13:

Table 13 — Zone descriptor format								
Bit Byte	7	6	5	4	3	2	1	0
0	Reserved				ZONE TYPE			
1	ZONE CONDITION				Reserved			RESET
2	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								

A POSITA would be motivated to implement the reset write pointer command to cause erasure of the targeted zone. Section VIII.D.4.c. 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*.

12. Claim 12

(a) 12[pre]

Sections VIII.D.1, IX.C.1.

(b) 12[a]

A POSITA would be motivated to use Reiter’s superblocks as ZBC zones (*addressed segment*). Section VIII.D.1.f. “[A] Superblock... is a logical collection of blocks” (*integer number of physical erase units*). EX1004, 9:11-13.

Further, logical sector number 404, or alternatively the bit string of logical block number 406 and logical page number 408, or alternatively logical page number 408 or page index 414, (*third address portion*) correspond to superblock's sequentially addressed blocks and pages (*specifies a location within a sequential range of locations that correspond to the integer number of physical erase units*).

EX1003, ¶¶197-99; EX1004, 9:5-25, 9:52-63, 12:33-35, 12:67-13:3, FIGs. 4a-c.

13. Claim 16

(a) 16[pre]

Sections VIII.D.1, IX.C.1.

(b) 16[a]

Section VIII.D.5.b.

(c) 16[b]

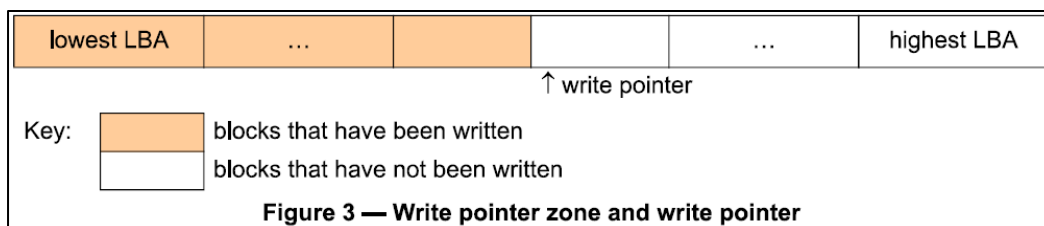
Section VIII.D.13.d-e.

(d) 16[b][i]

Reiter's flash controller 104 (*circuitry*) stores a RESET bit (*metadata*) for each zone of a logical unit (*on an independent basis for flash memory associated with each one of the block devices*). EX1003, ¶¶200-04. A RESET bit is stored for each zone because a report zones command is returns the “[l]ist [of] ones for which the RESET bit is set to one.” EX1006, 25. And a POSITA would understand that returning the list of zones with RESET bits set to one requires the flash controller to store a RESET bit for each zone. EX1003, ¶203.

(e) 16[b][ii]

Reiter's flash controller 104 (*circuitry*) *receives* a ZBC report zones command (*query*), that is sent by an external device (*issued by a host device*). EX1004, 7:5-9, Section VIII.D.6.c. ZBC discloses a report zones command that returns the “[l]ist [of] ones for which the RESET bit is set to one.” EX1006, 25. After receiving this command (*responsive to the query*), flash controller 104 reads each zone's RESET bit (*identify specific metadata*), checks whether each zone's RESET bit is equal to one (*compare metadata to at least one threshold*), and returns the list of zones where the RESET bit is equal to one (*transmit information to the host device*). EX1003, ¶¶206; EX1006, 25, Tables 11-13. The list of zones is sent back with a list of zone descriptors. EX1006, 25, Tables 11-13. The zone descriptors include the zone's write pointer LBA, (EX1006, Table 13) which details written-to LBAs in the zone (*such transmitted information conveys address information for data stored at one of the one or more physical storage locations associated with the specific metadata*). See EX1006, Figure 3:



14. Claim 20

(a) 20[pre]

Sections VIII.D.1, IX.C.1.

(b) 20[a]

EX1003, ¶¶206-08; Section VIII.D.1.h.

15. Claim 21-23

Claim 21 is substantively identical to claim 1. Claim 22 is substantively identical to claim 21 incorporated with dependent claim 3. EX1003, ¶¶209-10. Claim 23 is substantively identical to claim 1. While claim 1 is directed to “[a] storage device,” claim 23 is directed to “[a] method of operation in a storage device.” This distinction is immaterial since Reiter discloses systems and steps in operating those systems. *E.g.*, EX1004, FIGs. 4a-c, 5a-b, 6a-b.

Claim 21	Claim 22	Claim 23	Corresponding Claim Limitation(s)
21[pre]	22[pre]	23[pre]	1[pre]
21[a]	22[a]	23[pre]	1[a] (1[b][iii]-1[d] for claims 21-22)
21[b]	22[b]	23[a], 23[b]	1[b] (3[b] for claim 22)
21[b][i]	22[b][i]	23[b][i]	1[b][i]
21[b][ii]	22[b][ii]	23[b][i]	1[b][ii]
21[b][iii]	22[b][iii]	23[b][ii]	1[b][iii]
21[b][iv]	22[b][iv]	23[b][iii]	1[b][iv]
21[c]	22[c]	23[b][iv]	1[c]
21[d]	22[d]	23[b][v]	1[d]
21[e]	22[e]	23[c]	1[e]
	22[f], 22[f][i], 22[f][ii], 22[f][iii],		3[b]

Claim 21	Claim 22	Claim 23	Corresponding Claim Limitation(s)
	22[f][iv]		
	22[f][v], 22[f][vi]		3[c]
	22[g]		3[d]

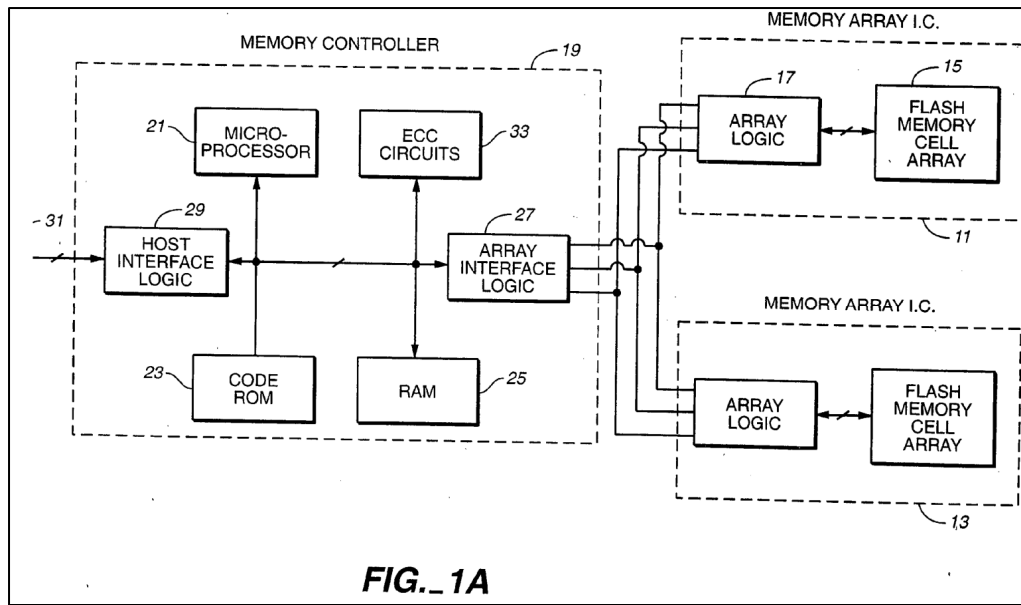
Limitations 21[a] and 22[a] adds that “block devices,” “segments,” and “storage locations” are at “hierarchically-superior,” “-intermediate,” and “-inferior level[s],” respectively, and that block devices and segments contain mutually exclusive sets of segments and logical erase units, respectively. But Ground I discloses logical units, each with its own LBA range, thereby its own zones/superblocks, where each zone/superblock contains its own logical blocks and pages. Sections VIII.D.1.d-f. Therefore, limitations 21[a] and 22[a] are rendered obvious for the same reasons as claim 1.

IX. GROUND II: CLAIMS 1, 13-14, AND 19 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SBC-4, SPC-4, AND SINCLAIR-367

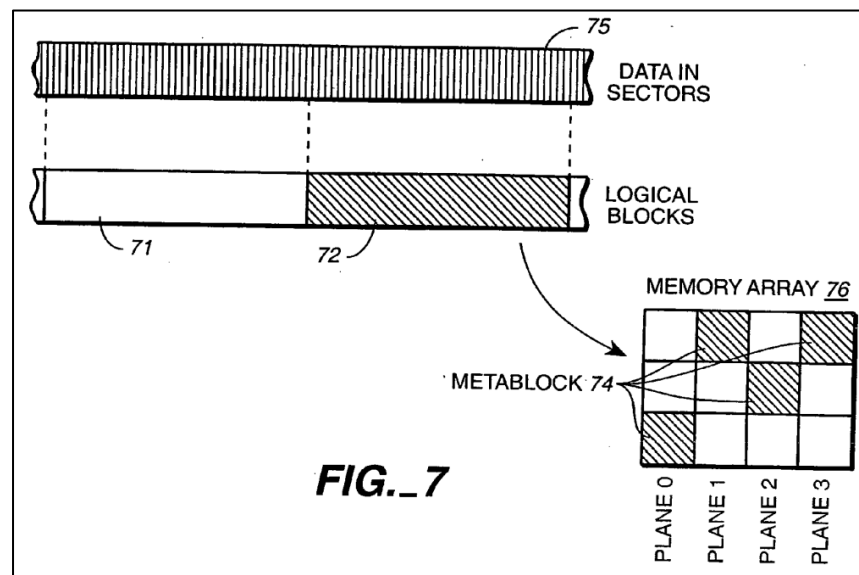
A. Sinclair-367

U.S. Patent Publication No. 2005/0144367 (“Sinclair-367”) was filed on December 16, 2004, published on June 30, 2005, and is prior art under 35 U.S.C. § 102.

Sinclair-367 discloses a non-volatile memory array (*i.e.*, flash memory) via adaptive metablocks. EX1003, ¶¶55-58; EX1009, Abstract, FIG. 1A:

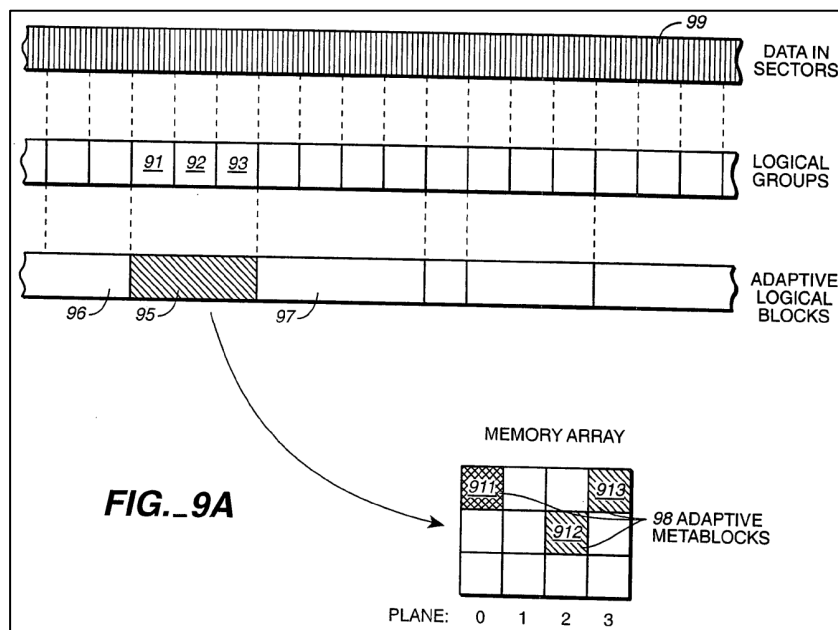


Metablocks are a logical grouping of erase units with each erase unit in a different plane so that reading, writing, and erasing across the metablock occurs simultaneously. EX1009, ¶¶[0114]-[0115], FIG. 7:



Adaptive metablocks are also a logical grouping of erase units, but do not have a fixed size. EX1009, ¶¶[0114]-[0115]. Adaptive metablocks might not have one

erase unit in each plane (e.g., two erase units in one plane), limiting the extent of simultaneous operation. EX1009, ¶¶[0114]-[0115], [0119]-[0120], FIG. 9A:



B. Motivation to Combine

A POSITA would have been motivated to combine Reiter’s flash system executing ZBC commands (Section VIII.C) with Sinclair-367’s bad block management, metadata storage, and planes and metablocks. EX1003, ¶¶211-14.

Sinclair-367 manages bad blocks—identifying “defective cells within the erase block,” and remapping the “logical address” with “the address of any substitute” block, where the “primary erase block is defective.” EX1009, ¶¶[0107]-[0109], [0110]. This improves system resiliency. Because Reiter contemplates bad block management, (*see* Section VIII.D.1.j) a POSITA would have a reasonable

expectation of success implementing bad block management in Reiter's flash device. EX1003, ¶¶212-14.

Further, 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], [0103]. Program/erase cycles determine the wear and tear on a physical block. EX1009, ¶[0108]. 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]. 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. EX1003, ¶213.

Further, Sinclair-367 discloses dividing flash memory into planes, and forming metablocks by linking erase units in each plane. EX1009, ¶¶[0009]-[0012], [0112]-[0114]. This improve speed and throughput. Each plane operates independently—concurrently executing reading/writing. EX1009, ¶[0112]. And “to increase... parallelism[,] erase blocks... are logically linked to form metablocks.” EX1009, ¶[0114]. Because Sinclair-367 and Reiter disclose analogous flash systems, and planes and metablocks are common in the flash memory art, a POSITA

would have a reasonable likelihood of success implementing them in Reiter's flash system. EX1003, ¶214.

Reiter, ZBC, and Sinclair-367 are analogous art—relating to storage devices, including accessing blocks of memory. EX1003, ¶214.

C. Claim-by-Claim Analysis

1. Claim 1

(a) 1[pre]-[d]

Sections VIII.D.1.a-i.

(b) 1[e]

A POSITA would be motivated to modify Reiter's flash system 100 (*storage device*) to implement Sinclair-367's bad block management. Section IX.B. Sinclair-367 discloses "identification of any defective cells within the erase block," (*detect a failure condition of the physical erase units*) and *remapping* the "logical address," such as updating the Reiter's summary page 420 which "contains the ... [superblock] mapping data" for translating logical block number 406 or alternatively block index 412 (*logical erase unit which corresponds to one of the physical erase units for which the failure condition is detected*) (EX1004, 9:52-63, FIGs. 4a-b), with the "address of any substitute erase block in case the primary erase block is defective" (*instead correspond to one of the physical erase units for which the failure condition has not been detected*) for later use (*utilize the remapped correspondence*

to service one or more of the incoming data access requests). EX1003, ¶¶215-16; EX1009, ¶¶[0107]-[0109].

2. Claim 13

(a) 13[pre]

Section VIII.D.1.

(b) 13[a]

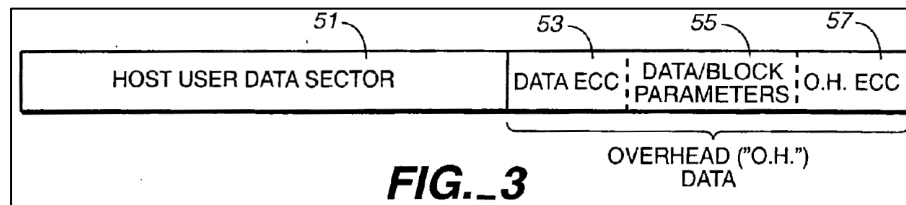
Section VIII.D.5.b.

(c) 13[b]

Sections IX.C.2.d-g.

(d) 13[b][i]

Using Reiter’s flash controller 104 (*circuitry*), Sinclair-367’s erase blocks have a “single sector of data” to *store* overhead data, including ECC (*metadata*). EX1009, ¶¶[0106]-[0107], [0110], FIG. 3:



Therefore, every block in a logical unit includes these sectors of overhead data (*independent basis for flash memory associated with each one of the block devices*). EX1009, ¶¶[0107], [0110]. Sinclair-367 also includes the “time of programming... as a field in [the] sector or page overhead data” (*metadata representing a time since one or more physical storage locations associated with*

the one or more block devices has been a subject of a write operation). EX1003, ¶¶217-22; EX1009, ¶[0111].

(e) 13[b][ii]

A POSITA would be motivated to use Reiter’s flash controller 104 (*circuitry*) to *compare* the time of programming data in a physical block (*metadata*) against a retention period (*at least one threshold*), where a retention period is the period of time where flash memory reliably stores a value. EX1003, ¶¶223-25.

Sinclair-367 recognizes that flash memory has a retention period problem. EX1009, ¶¶[0007]-[0008]. 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 a slim ranges associated with certain storage states is difficult. EX1009, ¶¶[0007]-[0008]. “[S]torage levels do shift” 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, ¶224. This can alert flash controller 104 of potential data loss, and provide an opportunity to save the data.

(f) 13[b][iii]

Upon nearing the average retention period (*upon detection that specific metadata has satisfied one or more thresholds*), a POSITA would be motivated to

copy data out of the unreliable block nearing its retention period (*copy data stored at one of the one or more physical storage locations associated with the specific metadata*) and into a working physical block (*from a first one of the physical erase units for which the failure condition has not been detected by said storage device to a second one of the physical erase units for which the failure condition has not been detected by said storage device*). EX1003, ¶¶226-27; EX1009, ¶¶[0108]-[0109], [0114], [0132], [0177].

(g) 13[b][iv]

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 the logical erase unit corresponding to the first one of the physical erase units for which the failure condition has not been detected by said storage device to the second one of the physical erase units for which the failure condition has not been detected by said storage device*). EX1003, ¶¶228-29; EX1009, ¶¶[0108]-[0109]; Section IX.C.1.b.

3. Claim 14

(a) 14[pre]

Sections VIII.D.1, IX.C.1.

(b) 14[a]

Section VIII.D.5.b.

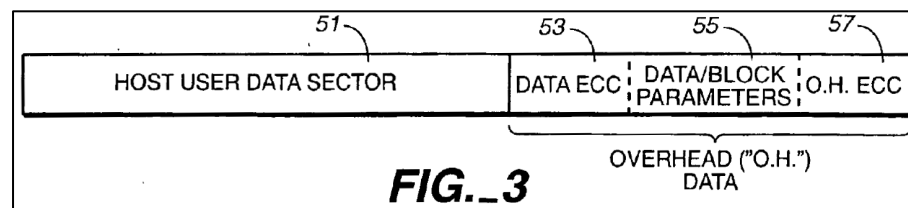
(c) 14[b]

Sections IX.C.3.d-e.

(d) 14[b][i]

Using Reiter's flash controller 104 (*circuitry*), Sinclair-367's erase blocks use a "single sector of data" to *store* overhead data, including ECC 53 (*metadata*).

EX1009, ¶¶[0107], [0110], FIG. 3:



Therefore, every block in a logical unit includes these sectors of overhead data (*independent basis for flash memory associated with each one of the block devices*). EX1009, ¶¶[0107], [0110]. Sinclair-367 also discloses that "parameter 55 may include a quantity related to the number of program/erase cycles experiences by the erase block" (*wherein the stored metadata is dependent on an erase count of associated physical memory space*). EX1003, ¶¶230-32; EX1009, ¶[0108].

(e) 14[b][ii]-[iv]

Sinclair-367 discloses that the "number of program/erase cycles" is "used in a wear leveling algorithm," where "LBAs are regularly re-mapped to a different physical block addresses in order to even out the usage." EX1009, ¶[0108].

A POSITA would understand, or be motivated to modify so, that the remapping hinges on the number of program/erase cycles (*metadata*) being high

enough to indicate a heavily used erase block (*at least one threshold*). EX1003, ¶¶236-39; EX1009, ¶¶[0108]-[0109]. Thus, *upon detection that* an erase block is heavily used (*specific metadata has satisfied the one or more thresholds*), LBAs are remapped. EX1009, ¶¶[0108]-[0109]. A POSITA would understand that remapping an LBA to a different physical block means the data of the old block is copied to the new block to prevent data loss (*copy data stored at one of the one or more physical storage locations associated with the specific metadata from a first one of the physical erase units for which the failure condition has not been detected by said storage device to a second one of the physical erase units for which the failure condition has not been detected by said storage device*). EX1003, ¶¶236-39; EX1009, ¶¶[0108]-[0109], [0114], [0132], [0177].

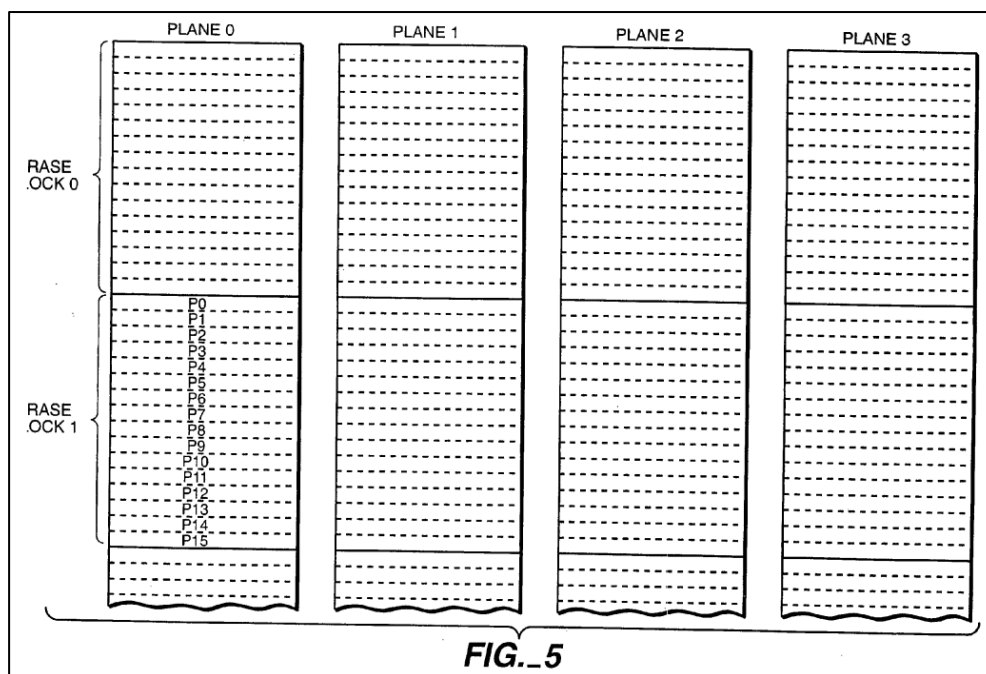
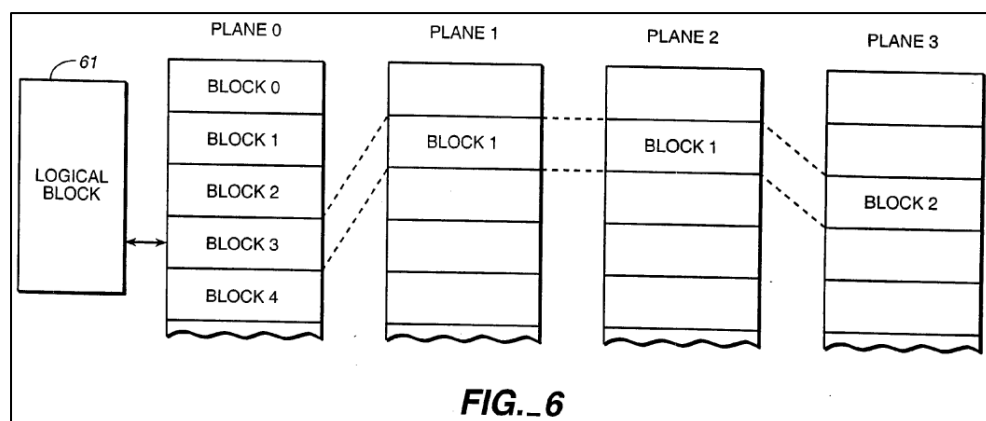
4. Claim 19

(a) 19[pre]

Sections VIII.D.1, IX.C.1.

(b) 19[a]

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, ¶[0112], FIGs. 5-6 (below); Section IX.B.



(c) 19[b]

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, ¶244.

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, ¶¶[0112]-[0114]; Sections VII.B.4.a, VII.B.1. “All blocks of the metablock are erased together, and pages of each block are generally programmed and read

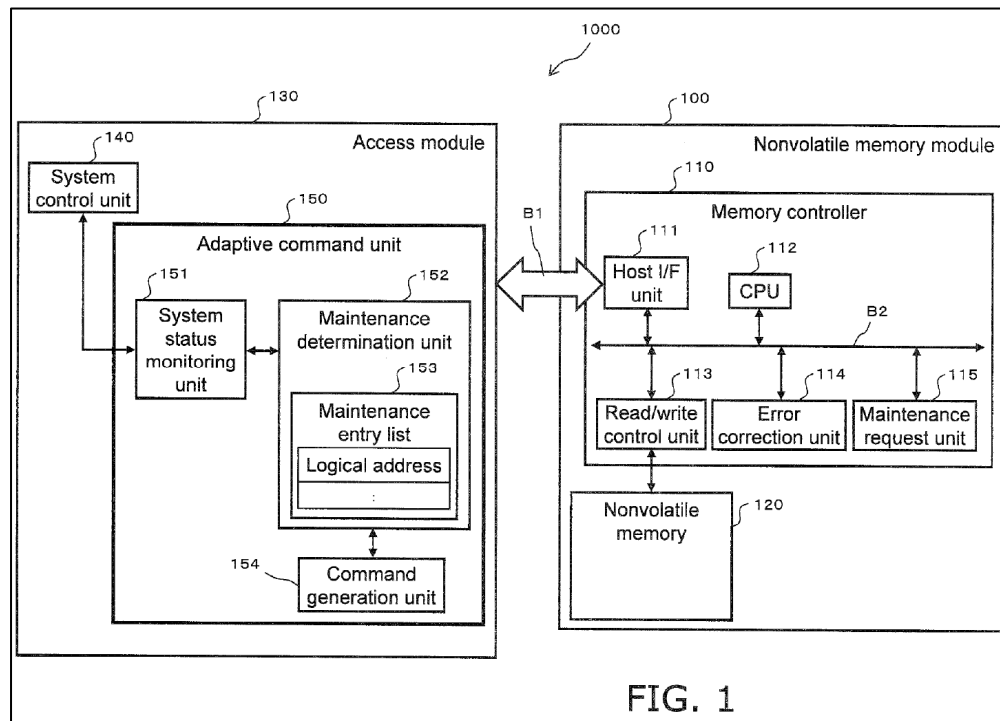
simultaneously” (*identify a commonly addressed page and write data accompanying the set of incoming data access requests concurrently into respective ones of the multiple planes*). EX1003, ¶¶240-46; EX1009, ¶[0114].

X. GROUND III: CLAIMS 15, 17-18 ARE RENDERED OBVIOUS BY REITER IN LIGHT OF ZBC, SINCLAIR-367, YAMADA

A. Yamada

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

Yamada discloses a nonvolatile memory module with flash memory and memory controller. EX1003, ¶¶60-63; EX1010, FIG.1. 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 tries to perform 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 implement Reiter's flash system to execute ZBC commands (Section VIII.B.2) and with Yamada's maintenance request and determination units in Reiter's flash system and host processor, respectively. Reiter complains that "[g]arbage collection routines impact performance of the flash memory system by utilizing processor resources and potentially delaying write operations." EX1004, 4:23-26. In Yamada, "maintenance is performed without

being... performed together with normal system processing,” but “with sufficiently long processing time.” EX1010, Abstract. The maintenance request unit on the flash device determines and requests maintenance operations from the maintenance determination unit. EX1010, Abstract, ¶¶[0081]-[0082], [0089]. The maintenance determination unit monitors status of the host, “determin[es] that the access device is not engaged in normal processing,” and performs maintenance. EX1010, Abstract, ¶¶[0018]-[0019], [0022]. Because maintenance scheme is implemented on a nonvolatile system analogous like that in Reiter, a POSITA would have a reasonable expectation of success implementing Reiter’s flash system with this scheme. EX1003, ¶247.

Reiter, ZBC, Sinclair-367, and Yamada are analogous art—relating to storage devices. EX1003, ¶248.

C. Claim-by-Claim Analysis

1. Claim 15

(a) 15[pre]

Sections VIII.D.1, IX.C.1.

(b) 15[a]

Section VIII.D.5.b.

(c) 15[b]

Section X.C.1.d-f.

(d) 15[b][i]

Section IX.C.2.d.

(e) 15[b][ii]

Section IX.C.2.e.

(f) 15[b][iii]

Yamada's maintenance determination unit 152 (on Reiter's external device (*host*) (Section X.B)) receives a "logical address included in [a] maintenance request transmitted from... maintenance request unit 115" (on Reiter's flash system 100 (Section X.B)) (*transmit information to a host device wherein such transmitted information conveys address information*). EX1003, ¶¶249-56; EX1010, ¶[0089].

Further, a POSITA would be motivated to trigger Yamada's maintenance request upon detecting a time since programming nearing the average retention period (*responsive to detection that specific metadata has satisfied the one or more thresholds*) recorded in a physical block per Sinclair-367's teachings. EX1009, ¶¶[0106]-[0107], [0110]-[0111]; Sections IX.C.2.e, X.B. Yamada discloses triggering maintenance requests for garbage collection, including wear-level management for improving data retention. EX1010, Abstract, ¶¶[0009], [0011], [0033], [0090]. Sinclair-367 recognizes that flash memory has a retention period problem, as maintaining the voltage level within one of the slim ranges associated with certain storage states is difficult. EX1009, ¶¶[0007]-[0008]. "[S]torage levels do shift" shift over usage of the memory cell and surrounding memory cells.

EX1009, ¶[0007]. The problem disclosed by Sinclair-367 is recognized and resolved with Yamada’s maintenance requests and commands. EX1010, Abstract, ¶¶[0009], [0011], [0033], [0090].

2. Claim 17

(a) 17[pre]

Sections VIII.D.1, IX.C.1.

(b) 17[a]

Section VIII.D.5.b.

(c) 17[b]

Sections X.C.2.d-e.

(d) 17[b][i]

Section IX.C.3.d.

(e) 17[b][ii]

Yamada discloses that “non-volatile memory 1020 stores management information including the information about the importance level of data to enable the management information to be shared between different access modules.” EX1010, ¶[0190]. Because management information is information governing whether maintenance is requested to improve data retention, (EX1010, Abstract, ¶¶[0009], [0011], [0033], [0090]) a POSITA would be motivated to include Sinclair-367’s program/erase-cycle-count metadata in Yamada’s management information. EX1009, ¶¶[0107]-[0108], [0110]; *see* Section IX.C.3.d-e.

Moreover, a POSITA would understand, or be motivated to modify so, that Yamada's sharing of such metadata means the access module *querying (issued by a host device)*, where the query causes (*responsive to the query*) the storage device *identify* metadata and *transmit* metadata to the access module. EX1003, ¶¶257-263.

3. Claim 18

(a) 18[pre]

Sections VIII.D.1, IX.C.1.

(b) 18[a]

Section VIII.D.5.b.

(c) 18[b]

Sections X.C.3.d-f.

(d) 18[b][i]

Section IX.C.3.d.

(e) 18[b][ii]

Section IX.C.3.e.

(f) 18[b][iii]

Yamada discloses *transmit information to a host device wherein such transmitted information conveys address information*. Section X.C.1.f.

Further, a POSITA would be motivated to trigger a maintenance request upon detecting a high number of program/erase cycles (*responsive to detection that specific metadata has satisfied the one or more thresholds*) recorded in a physical

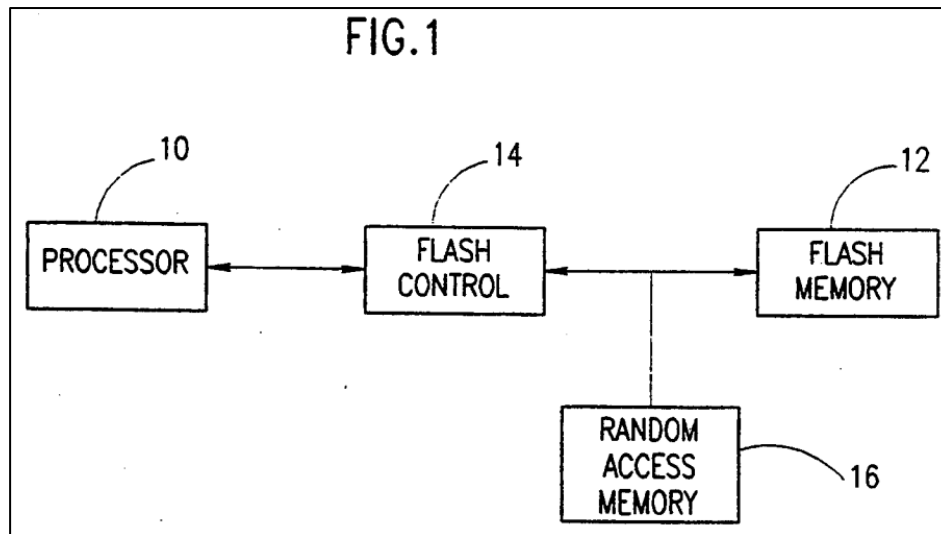
block per Sinclair-367's teachings. EX1003, ¶¶264-71; EX1009, ¶¶[0106]-[0107], [0110]-[0111]. Yamada discloses triggering maintenance requests for garbage collection, including wear-level management for improving data retention. EX1010, Abstract, ¶¶[0009], [0011]. Sinclair-367 discloses that program/erase counts for physical blocks are “used in a wear leveling algorithm... in order to even out the usage (wear) of all erase blocks”—avoiding a high number of program/erase cycles in any one block. EX1009, ¶[0108].

XI. GROUND IV: CLAIMS 1-14, 16, 19-23 ARE RENDERED OBVIOUS BY BAN IN LIGHT OF ZBC, SBC-4, SPC-4, SINCLAIR-367, AND YAMADA

A. Ban

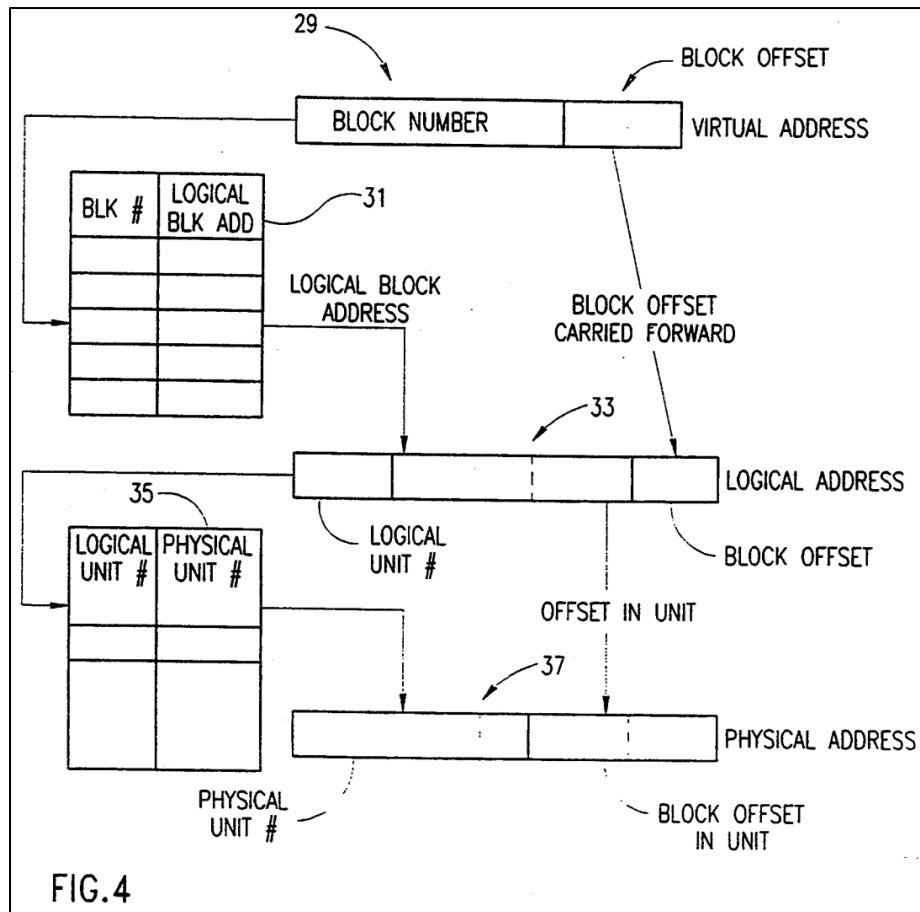
U.S. Patent No. 5,404,485 (“Ban”) was filed on March 8, 1993, and was issued on April 4, 1995. It is prior art under 35 U.S.C. § 102.

Ban discloses a flash memory system, where the flash memory is managed as to emulate random access memories—read/write operations to any location. EX1003, ¶¶46-51; EX1005, 3:52-4:5, FIG. 1:



Emulating random access memories, like standard hard disk drives, means interoperability with existing operating systems. EX1005, 1:29-68.

For emulation, Ban uses a two-step address translation, where the address accompanying the host's read/write command is translated to a logical address, which is translated to a physical address. EX1005, FIG. 4:



This abstracts the flash memory's addressing scheme. EX1005, 3:64-4:5. Ban's flash memory is made up of units, unlike disk drives, and addressing requires a unit number. EX1005, 4:11-24, FIG. 4. The host operating system, treating flash memory as a standard hard disk drive, does not provide a unit number. EX1005, 3:52-4:10. Ban rectifies this by translating the host's address to a unit number. EX1005, 4:51-5:35, FIG. 4.

B. Motivation to Combine

A POSITA would have been motivated to use ZBC, alongside SBC-4 and SPC-4, (collectively, "ZBC") as the command set that governs requests sent to Ban's

flash device, and incorporate the flash device with Sinclair-367's bad block management, metadata storage, and planes and metablocks. EX1003, ¶¶272-81.

First, A POSITA would be motivated to apply the teachings of ZBC, SBC-4, and SPC-4. Section VIII.C.

Second, a POSITA would be motivated to apply ZBC's teachings to Ban's flash system. EX1003, ¶274. Ban discloses that a host "writes data to, and reads data from, a flash memory 12 in blocks at specific address locations," (EX1005, 3:59-62), but does not require a specific command set. There is a need to formally detail such read/write commands with a command set, such as ZBC. EX1003.

A POSITA would particularly be motivated to implement Ban's flash device to accept and execute read/write commands based on ZBC. EX1003, ¶275. Ban expressly aims to be compatible with "typical existing operating systems." EX1005, 1:29-39, 1:52-57. SCSI protocols and commands such as those disclosed in ZBC are widely used, and would thus improve the compatibility and interoperability of Ban's flash system.

Commands for ZBC's zoned block device model (EX1006, 1) are particularly suitable for Ban's flash device. EX1003, ¶276. ZBC allows for random-access across an LBA range (*see* EX1006, 14, 16, Figures 5-6)—the "object[ive] of [Ban's] invention"—while also enabling effective utilization of flash memory. EX1005, 1:52-62, 3:52-4:10. ZBC includes a reset write pointer command, which resets a

zone's write pointer to the zone's lowest LBA, and makes the zone's data no longer accessible. EX1006, 3, 17-18, Table 16. This lines up well for flash memory's erase operation, which is a unique feature of flash memory. EX1005, 1:16-27, 2:38-45.

A POSITA would have a reasonable expectation of success in adding support for ZBC in Ban's flash system. EX1003, ¶277. SCSI commands and protocols such as ZBC were widely known and would have been familiar to a POSITA. EX1003, ¶277. SCSI specifications were intended to be used across a wide range of storage devices, including flash memory. EX1003, ¶277. 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. EX1003, ¶277.

Third, Sinclair-367 manages bad blocks, which improves system resiliency. Section IX.B. A POSITA would have a reasonable expectation of success implementing this bad block management in Ban's flash device because Ban's translation scheme is intended for swapping out blocks. EX1005, 2:50-3:2, FIGs. 4, 7.

Further, Sinclair-367 discloses storing in each erase unit and/or page a sector of metadata, which is used to increase the flash device's lifespan. Section IX.B. A POSITA would have a reasonable expectation of success implementing this metadata sector in a flash device because a sector is a small overhead for each block and/or page. Section IX.B.

Further, Sinclair-367 discloses dividing flash memory into planes, and forming metablocks by linking erase units in each plane, which improves speed and throughput. Section IX.B. A POSITA would have a reasonable expectation of success implementing these teachings in Ban’s flash device because Sinclair-367 and Ban disclose analogous flash systems, and planes and metablocks are common in the flash memory art. EX1003, ¶280.

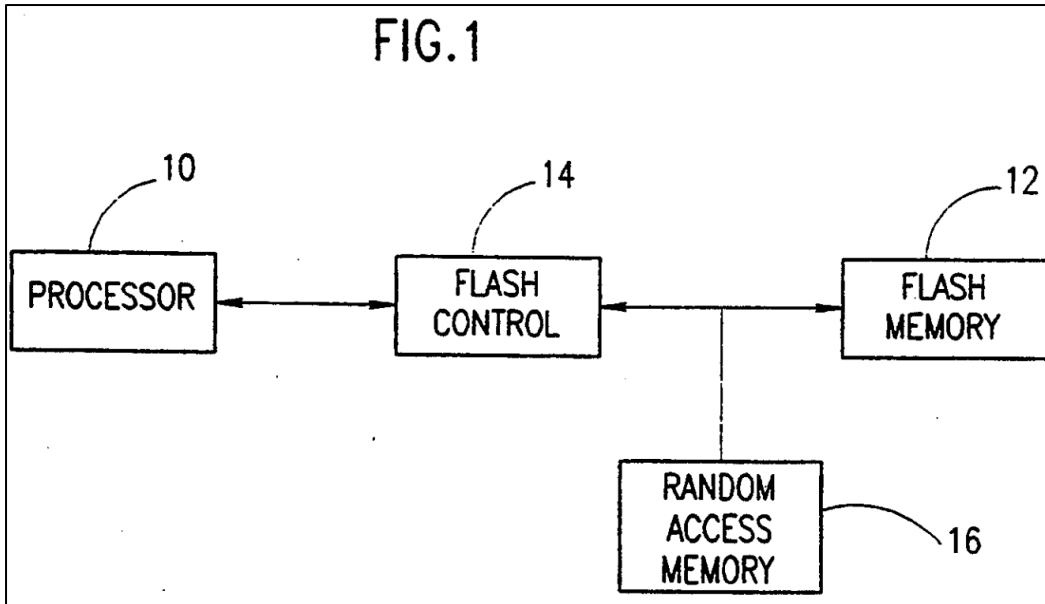
Ban, ZBC, and Sinclair-367 are analogous art—relating to storage devices, including techniques for accessing blocks of memory. EX1003, ¶281.

C. Claim-by-Claim Analysis

1. Claim 1

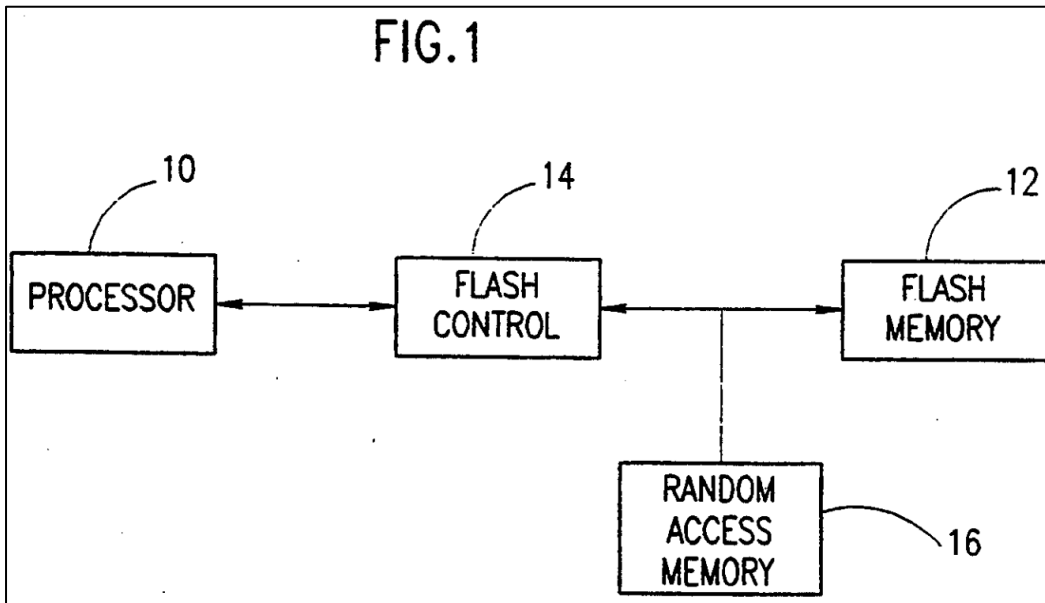
(a) 1[pre]

Ban discloses a flash device (*storage device*) with “flash memory controller 14,” “flash memory 12,” and “random access memory 16.” EX1005, 3:59-4:7, FIG. 1:



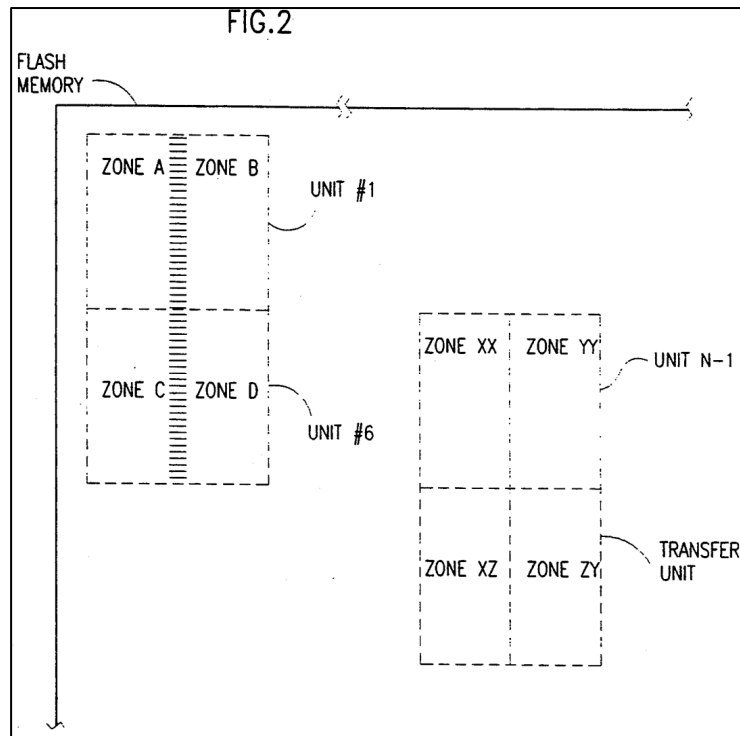
(b) 1[a]

Ban's flash device includes *flash memory* 12. EX1005, 3:59-4:7, FIG. 1:



Flash memory 12 broken up into “zones” or “unit[s]” (*physical erase units*).

EX1005, 1:15-20, 4:11-14, FIG. 2:

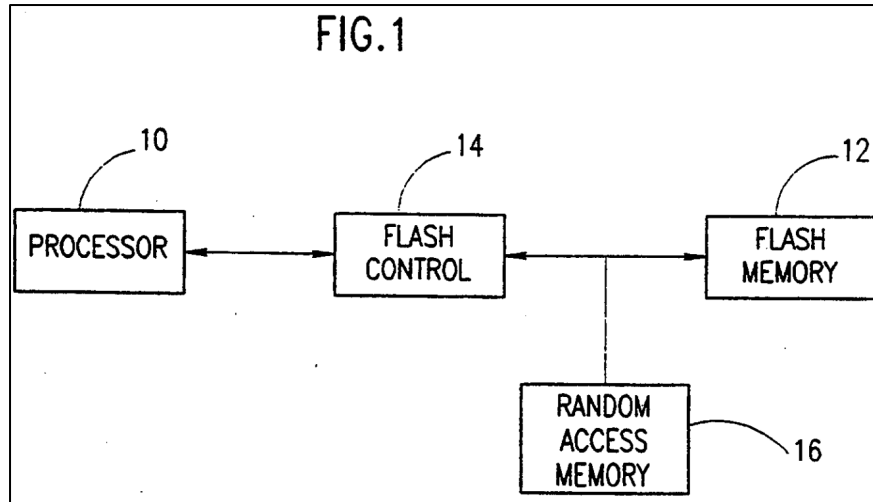


Zones and units are interchangeable as “[e]ach unit is comprised of at least one zone.” EX1005, 4:20-24. “Each unit contains blocks,” and “byte[s]... form a block.” EX1005, 2:11-19. Reading/writing is applied to bytes. EX1005, 2:11-15, 2:24-30. Erasing is performed with a “zone erase operation,” erasing every block in the zone. EX1005, 2:15-18, 2:67-3:2, 4:38-43.

(c) 1[b]

Ban’s flash device includes flash controller 14 (*circuitry*). EX1005, 3:59-4:7,

FIG. 1:



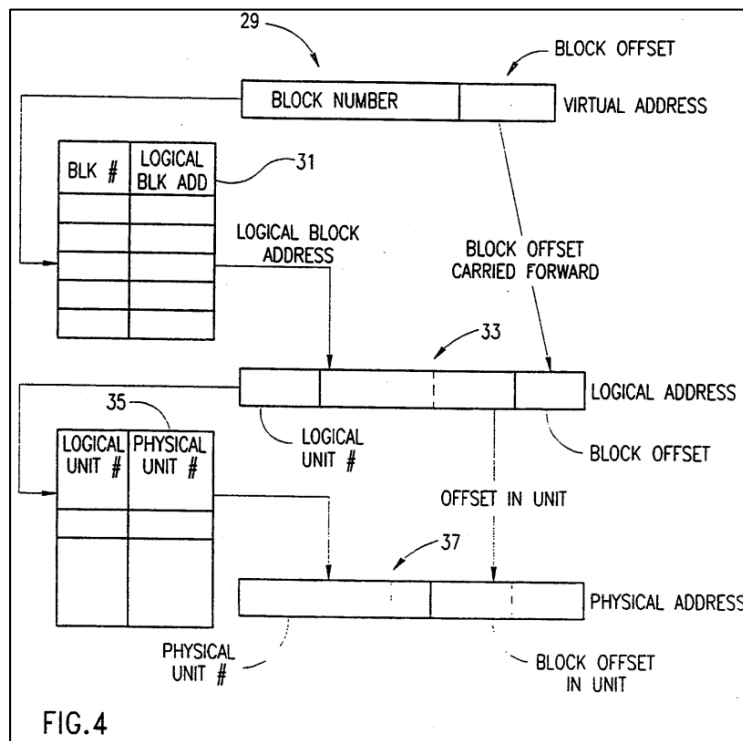
“[P]rocessor 10... issues... read and write commands to... specific address locations in a random access memory” (*incoming data access requests accompanying address information*). EX1005, 3:52-56. “[F]lash memory controller 14 provides a... virtual address space” to “emulate[]... random access memory” (*receive*). EX1005, 3:64-4:5, FIG. 2.

(d) 1[b][i]

From a ZBC read command (*accompanying incoming address information*), flash controller 14 *derives* (1) a ZBC logical unit number (*first address portion*), (2) virtual address 29 (*second address portion*), and (3) logical address 33 (*third address portion*). EX1003, ¶¶282-93.

First Address Portion: A POSITA would be motivated to incorporate a ZBC logical unit number in ZBC commands, and motivated to modify flash controller 14 to decode (*derive*) that number (*first address portion*) from the command (*from the accompanying address information*). See Section VIII.D.1.d.

Second Address Portion and Third Address Portion: A POSITA would be motivated to use Ban’s flash controller 14 to *derive* virtual address 29 (*second address portion*), which is further translated to logical address 33 (*third address portion*), from the ZBC read command’s LBA (*from the accompanying address information*). EX1005, FIG. 4. A ZBC read command includes an LBA. EX1007, Table 63. Flash controller 14 receives read commands targeting specific addresses, and a “computer generated address is decoded as a virtual block address and a byte location within the block” (*i.e.*, “virtual address 29”). EX1005, 2:24-26, 2:31-33, 4:54-56, FIG 4 (below). Virtual address 29 is translated by flash controller 14 into logical address 33. EX1005, 5:2-12, FIG. 4:

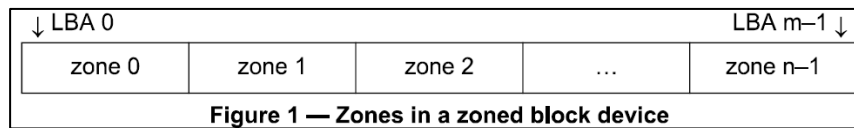


(e) 1[b][ii]

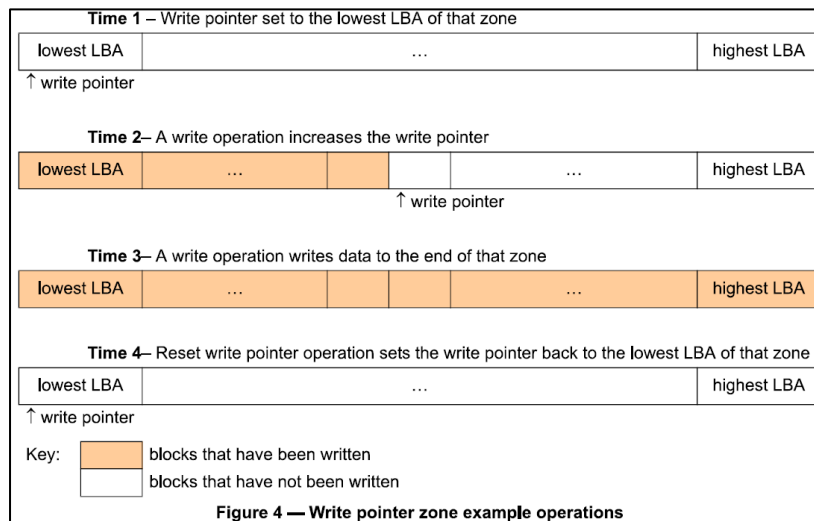
The ZBC logical unit number (*from the first address portion*) *identif[ies]* a ZBC logical unit (*addressed block device*). A ZBC logical unit has an independent range of LBAs starting at zero. EX1003, ¶¶294-95; EX1008, 61.

(f) 1[b][iii]

A POSITA would be motivated to modify Ban’s flash controller 14 to *identify* a zone number (*addressed segment*) *from* virtual address 29 (*second address portion*). EX1003, ¶¶296-300. A zone number is the zone that contains the ZBC command’s LBA—virtual address 29. EX1006, 13, Figure 1:



Identifying a ZBC zone number is helpful for pulling and/or updating necessary ZBC zone metadata. For instance, a write command changes a zone’s write pointer. EX1006, Figures 4 (below), 5-6:

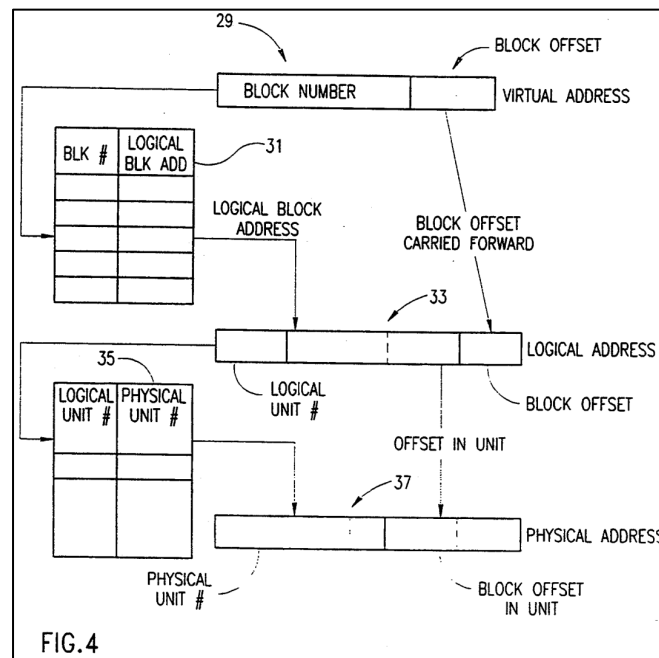


For a read command, ZBC zone metadata determines whether an LBA has been written since the last reset write pointer command, where, if not, the “manufacturer’s default initialization pattern” is returned. EX1006, 16-17.

A ZBC zone constitutes *an addressed segment* and is *within* the targeted logical unit (*addressed block device*). See Section VIII.D.1.f.

(g) 1[b][iv]

Logical address 33 includes (*from the third address portion*) “a logical unit [(i.e., zone)] number along with the offset of the block within the [zone]” (*identify an addressed storage location*), where the offset of the block is passed down to also be the “offset of the addressed 37 block” in the physical address. EX1003, ¶¶301-03; EX1005, 5:10-12, 5:18-24, FIG. 4:



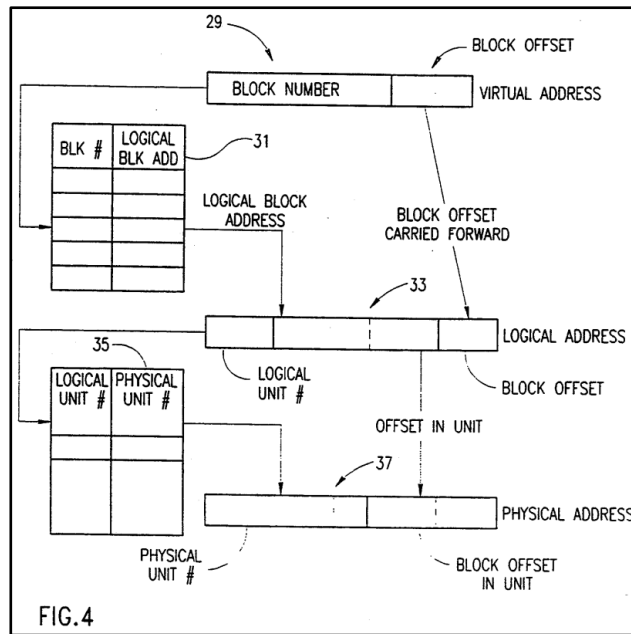
The block offset in logical address 33 is *within* the ZBC zone (*addressed segment*) because that block offset is the same, or otherwise derived from, virtual address 29—ZBC command’s LBA—which falls within a range of LBAs that make up the identified ZBC zone. Section XI.C.f; EX1005, FIG. 4; EX1006, Figure 1.

Further, Patent Owner makes the same mapping in its infringement contentions. Patent Owner alleges that “identify[ing] an addressed storage location[]... includ[es] a logical block, page, and/or erase unit.” EX1011, 27.

(h) 1[c]

Flash controller 14 (*circuitry*) *uses* bit shifting (*division operation*), *to identify* the logical zone number (*logical erase unit*). EX1003, ¶¶304-08; EX1005, 5:7-10. And the logical zone number is derived from virtual address 29—ZBC command’s LBA—and therefore *within* the ZBC zone (*addressed segment*). Section XI.C.1.f; EX1005, FIG. 4.

Ban discloses that the “logical unit [(i.e., zone)] number is the high order binary digits of the logical address and may be derived from the logical address by a bit shift operation.” EX1005, 4:67-5:12, FIG. 4:

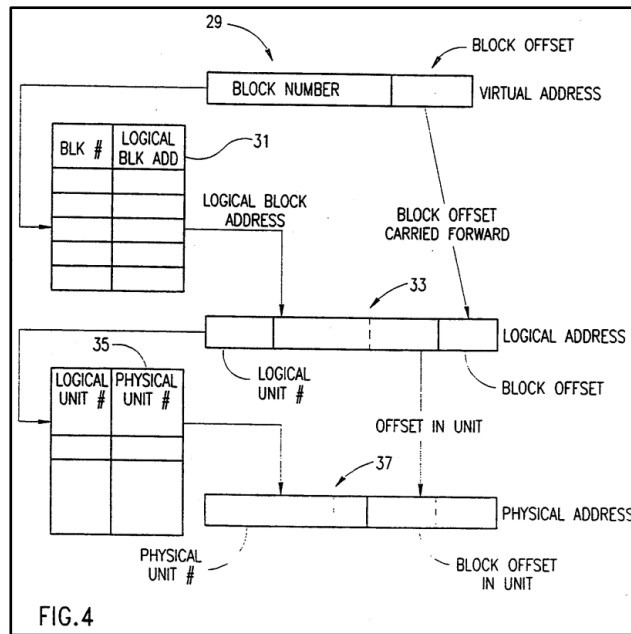


The logical zone number is a *logical erase unit*. This is explicit in the name—**“logical unit number.”** Logical zone number is a logical unit. A zone refers to an erase unit. Section XI.C.1.b.

A logical zone number is an abstraction of a physical unit. It maps to the physical zone number. EX1005, 5:13-17, FIG. 4.

(i) 1[d]

Flash controller 14 (*circuitry is to*) translates the logical zone number (*identified logical erase unit*) to a physical zone number (*identify one of the physical erase units that corresponds to the identified logical erase unit*). EX1003, ¶309-10; EX1005, 5:13-17, FIG. 4:



(j) 1[e]

Sections IX.C.1.b, XI.B.

2. Claim 2

(a) 2[pre]

Section XI.C.1.

(b) 2[a]-[b]

A POSITA would understand that Ban’s flash controller 14 (*said circuitry*) *comprises at least one processor* and *instructions stored on at least one physical storage medium, said instructions when executed to control the at least one processor*. EX1003, ¶¶313-15; EX1005, 3:52-4:10 (“[C]ontroller 14 functions may be carried out in software, firmware[,] or hardware.”).

3. Claim 3

(a) 3[pre]

Section XI.C.1.

(b) 3[a]

Section XI.C.1.c.

(c) 3[b]

Sections VIII.D.3.c, XI.B, XI.C.1.d-g.

(d) 3[c]-[d]

Ban discloses translating a read request's LBA to a physical address. EX1005, FIGs. 4-5. The same translation applies to write operations (*in association with fulfillment of the received write request, identify a physical storage location in which associated write data has been stored and which is associated with the corresponding logical erase unit*). EX1003, ¶¶316-21; EX1005, 5:36-6:2, FIG. 6; Sections XI.C.1.d-i.

ZBC write commands also cause flash system 100 (*storage drive, in association with fulfillment of the received write request*) to transmit information back to the host the upon command completion. EX1003, ¶¶322-25; Section VIII.D.3.d.

A POSITA would be motivated to modify the transmission to include information identifying the completed command (*information representing an address value to the host corresponding to the identified physical storage location*)

because SPC-4 discloses processing multiple commands in parallel. EX1008, xl, 431, 438, 472, 867, Table A.1; Section VIII.D.3.d.

A POSITA would be motivated to restrict subsequent read commands (*for one of the read requests*) targeting LBAs subject to pending write operations (*for one of the read requests which is directed to data which has been written into the flash memory pursuant to the received write request*) until after receiving a completion response of the pending write operations (*dependent on the address value which has been transmitted to the host in association with the data which has been written into the flash memory pursuant to the received write request*). EX1003, ¶324; Section VIII.D.3.d. The underlying *third address portion* is translated from the LBA, (Section XI.C.1.d) and is therefore also *dependent on* the completion response.

4. Claim 4

(a) 4[pre]

Section XI.C.1.

(b) 4[a]

A POSITA would be motivated to implement Ban's flash device (*storage device*) to *receive* and execute ZBC commands. Section XI.C.1.c. ZBC includes a reset write pointer command, which teaches this limitation. Section VIII.D.4.b.

(c) 4[b]

EX1003, ¶¶236-29; Sections VIII.C.4.c, XI.B.

5. Claim 5

(a) 5[pre]

Section XI.C.1.

(b) 5[a]

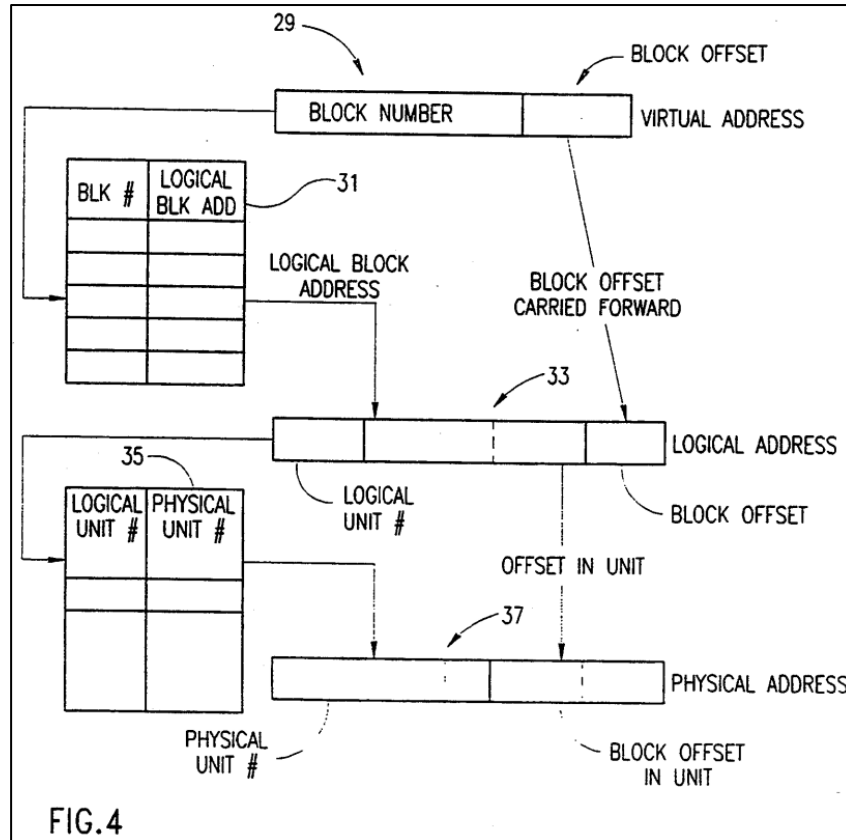
A POSITA would understand that a ZBC read command targets one of the ZBC logical units (*addressed block device*) on Ban's flash device's flash memory 12 (*flash memory is to have block devices*). Section XI.C.1.d.

(c) 5[b]

Section XI.C.5.d.

(d) 5[b][i]-[ii]

Ban discloses translating a read command's virtual address 29 (*in connection with a given one of the incoming data access requests*) into logical address 33, and logical address 33 into a physical address with look-up tables 31 and 35, respectively (*identify and retrieve a look-up table*) (*identify a physical storage location in the flash memory dependent on the look-up table*). EX1005, FIG. 4:



A POSITA would understand that look-up tables 31 and 35 are *specific* to the targeted logical unit derived from ZBC read command (*addressed block device which was identified from the first address portion derived from the given one of the incoming access requests*). EX1003, ¶¶330-36. Logical units have overlapping logical address ranges, which means separate address translation structures should be used.

6. Claim 6

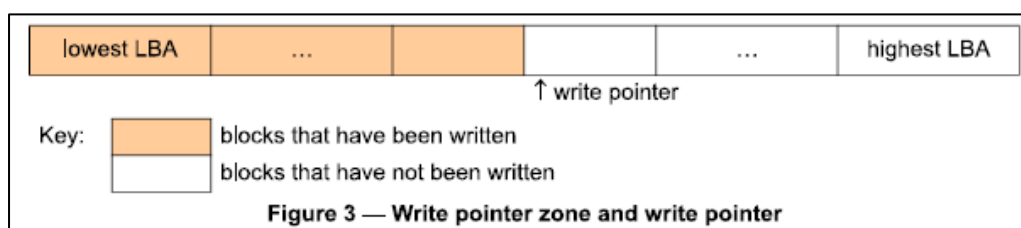
(a) 6[pre]

Section XI.C.1.

(b) 6[a]

Ban’s flash device (*storage device*) flash controller 14 (*circuitry to*) maintains a “usage map of all the blocks within [a] unit” (*track metadata for the addressed segment*). EX1005, 2:57-60. The usage map includes “special characters to denote free blocks and to denote unusable blocks.” EX1005, 2:57-60. Because a unit is made up of blocks, showing the free and written blocks *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*.

Alternatively, Patent Owner’s infringement contentions assert that a zone’s write pointer discloses the limitation. EX1011, 63. ZBC also relies on write pointers for ZBC zones. EX1006, 14, 17, Figure 3:



(c) 6[b]

A POSITA would be motivated to implement Ban’s flash device (*storage device*) to execute ZBC commands. Section XI.C.1.c. ZBC includes a report zones command, (EX1006, Table 12) which teaches this limitation. EX1003, ¶¶337-41; Section VIII.D.6.c.

7. Claim 7

(a) 7[pre]

Section XI.C.1.

(b) 7[a]

Section XI.C.3.b.

(c) 7[b]

Sections XI.C.3.c-d.

(d) 7[c]

Section XI.C.3.d.

(e) 7[d]

Section XI.C.3.d.

8. Claim 8

(a) 8[pre]

Section XI.C.1.

(b) 8[a]

A POSITA would be motivated to implement Reiter's flash system 100 (*storage device*) to *receive* and execute ZBC commands. Section XI.C.1.c. ZBC includes an extended copy command (*request from a host to copy*), (EX1006, 1, 22; EX1007, Table 33; EX1008, 287-89, Table 128) which teaches his limitation. EX1003, ¶¶342-46; Sections VIII.D.8.b, XI.B.

(c) 8[b]

EX1003, ¶351; Section VIII.D.8.c, XI.B.

9. Claim 9

(a) 9[pre]

Section XI.C.1.

(b) 9[a]

A POSITA would be motivated to implement Ban's flash device (*storage device*) to perform its address translation on a ZBC extended copy command's LBA (*from the destination address information*) to *identify* a logical zone number (*destination logical erase unit*). Section XI.C.8.b. Further translation, logical zone number (*in dependence on the identified destination logical erase unit*) *selects* physical unit number of the physical address (*the second storage location*). EX1003, ¶¶352-54; EX1005, FIG. 4.

10. Claim 10

(a) 10[pre]

Section XI.C.1.

(b) 10[a]

ZBC's report zones command *transmits the information identifying the number of structures to a host*. Section VIII.D.10.b. A POSITA would understand *storing*, or be motivated to *store*, that metadata rather than wasting time re-

calculating such information for every needing or requesting the metadata. EX1003, ¶¶355-57.

Further, Ban's flash device executing a valid ZBC's reset write pointer command discloses the rest of the limitation. Sections VIII.D.10.c, X.B.

11. Claim 11

(a) 11[pre]

Section XI.C.1.

(b) 11[a]

Section VIII.D.11.b.

12. Claim 12

(a) 12[pre]

Section XI.C.1.

(b) 12[a]

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*) because a POSITA would also be motivated to implement the reset write pointer command to erase a zone. EX1003, ¶¶360-63; Section XI.C.4.c. If a zone contained a non-whole number of physical units, zone erasure erases units only partially within the zone. Because unit erasure erases everything in the unit, a reset write pointer command erases data outside the zone, risking data loss. EX1005,

2:38-43. This, or complexity to avoid data loss, is avoided by requiring whole units in each zone.

Logical address 33 (*third address portion*) *specifies a location within a sequential range of locations*. Logical address 33 includes a block offset—position within a logical unit’s sequential list of blocks. EX1005, 5:36-38. And because logical address 33 is derived from virtual address 29, which determines the zone, logical address 33 *specifies a location corresponding to the ZBC zone’s whole number of units (integer number of physical erase units)*. EX1005, FIG. 5.

13. Claim 13

(a) 13[pre]

Section XI.C.1.

(b) 13[a]

Section XI.C.5.b.

(c) 13[b]

Sections XI.C.13.c-g.

(d) 13[b][i]

Sections IX.C.1.d, XI.B.

(e) 13[b][ii]

Sections IX.C.1.e, XI.B.

(f) 13[b][iii]

Sections IX.C.1.f, XI.B.

(g) 13[b][iv]

EX1003, ¶¶364-70; Sections IX.C.1.g, XI.B.

14. Claim 14

(a) 14[pre]

Section XI.C.1.

(b) 14[a]

Section XI.C.5.b.

(c) 14[b]

Sections XI.C.14.d-e.

(d) 14[c]

Sections IX.C.2.d, XI.B.

(e) 14[d]-[f]

EX1003, ¶¶371-75; Sections IX.C.2.e, XI.B.

15. Claim 16

(a) 16[pre]

Section XI.C.1.

(b) 16[a]

Section XI.C.5.b.

(c) 16[b]

Sections XI.C.15.d-e.

(d) 16[b][i]

Section VIII.D.13.d, XI.B.

(e) 16[b][ii]

EX1003, ¶¶376-80; Section VIII.D.13.d, XI.B.

16. Claim 19

(a) 19[pre]

Section XI.C.1.

(b) 19[a]

Sections IX.C.3.b, XI.B.

(c) 19[b]

EX1003, ¶¶381-87; Sections IX.C.3.c, XI.B.

17. Claim 20

(a) 20[pre]

Section XI.C.1.

(b) 20[a]

EX1003, ¶¶388-91; Section XI.C.1.h

18. Claim 21-23

Claims 21 and 23 is substantively identical to claim 1, and Claim 22 is substantively identical to claim 21 incorporated with dependent claim 3. EX1003, ¶¶390-91; Section VIII.D.15. The additional elements of claim 21 are disclosed in Ground IV—logical units containing zones, which contain logical blocks and pages. Sections VIII.D.15, XI.C.1.d-f. Claim 23 is directed to “[a] method of operation in a storage device while claim 1 is directed to “[a] storage device.” This distinction is

immaterial since Ban discloses systems and steps in operating those systems. *E.g.*, EX1005, FIGs. 4-5.

XII. GROUND V: CLAIMS 15, 17-18 ARE RENDERED OBVIOUS BY BAN IN LIGHT OF ZBC, SINCLAIR-367, YAMADA

A. Motivation to Combine

A POSITA would have been motivated to implement Ban’s flash device with Yamada’s maintenance request and determination units in Ban’s flash device and host processor, respectively. EX1003, ¶¶392-93. Ban mentions the necessity, but with little description, of a maintenance system, stating that “enabl[ing] reading and writing operations to continue without limitation, physical memory space is reclaimed periodically.” EX1005, 6:3-5. Yamada discloses a maintenance system, where “maintenance is performed without being... performed together with normal system processing,” but “with sufficiently long processing time.” EX1010, Abstract, ¶¶[0018]-[0019], [0022]; Section IX.B. Because Yamada and Ban implement analogous nonvolatile systems, a POSITA would have a reasonable expectation of success implementing Ban’s flash device with this scheme. EX1003, ¶392.

Ban, ZBC, Sinclair-367, and Yamada are analogous art—relating to storage devices. EX1003, ¶393.

B. Claim-by-Claim Analysis

1. Claim 15

(a) 15[pre]

Section XI.C.1.

(b) 15[a]

Section XI.C.5.b.

(c) 15[b]

Sections XII.B.1.d-f.

(d) 15[b][i]

Section XI.C.1.d.

(e) 15[b][ii]

Section XI.C.1.e.

(f) 15[b][iii]

EX1003, ¶¶394-99; Sections X.C.1.f, XII.A.

2. Claim 17

(a) 17[pre]

Section XI.C.1.

(b) 17[a]

Section XI.C.5.b.

(c) 17[b]

Sections XII.B.2.d-e.

(d) 17[b][i]

Section XI.C.2.d.

(e) 17[b][ii]

EX1003, ¶¶400-04; Sections X.C.2.e, XII.A.

3. Claim 18

(a) 18[pre]

Section XI.C.1.

(b) 18[a]

Section XI.C.5.b.

(c) 18[b]

Sections XII.B.3.d-f.

(d) 18[b][i]

Section XI.C.2.d.

(e) 18[b][ii]

Section XI.C.2.e.

(f) 18[b][iii]

EX1003, ¶¶405-10; Sections X.C.3.f, XII.A.

XIII. 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.

XIV. CONCLUSION

For these reasons, Petitioner requests IPR of the '656 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 13,999. 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 '656 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