

Filed on behalf of Petitioners

Filed: August 5, 2026

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

META PLATFORMS, INC. AND SAMSUNG ELECTRONICS CO., LTD.,
Petitioners,

v.

GAEA, LLC,
Patent Owner.

Case No. IPR2026-00427
U.S. Patent No. 12,265,715

**Petition for *Inter Partes* Review of
U.S. Patent No. 12,265,715**

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
II.	MANDATORY NOTICES.....	2
	A. Real Parties-in-Interest	2
	B. Related Matters.....	3
	C. Notice of Counsel and Service Information	3
	D. Fee for <i>Inter Partes</i> Review	5
	E. Grounds for Standing	5
III.	IDENTIFICATION OF CHALLENGES	5
IV.	BACKGROUND	6
	A. Technology Overview	6
	B. '715 Patent.....	7
	C. The Challenged Claims	10
	D. Prosecution History	11
	1. Prosecution of the ancestor '023 Patent	11
	2. Prosecution of the '715 Patent.....	13
V.	THE PRIOR ART.....	13
	A. Meir (EX1010)	13
	B. Salessi (EX1011).....	16
	A. Chou (EX1012)	17

C. Lovinger (EX1015)	17
VI. LEVEL OF ORDINARY SKILL IN THE ART	18
VII. CLAIM CONSTRUCTION	18
VIII. GROUND 1: MEIR IN VIEW OF SALESSI RENDERS OBVIOUS CLAIMS 1-2, 5-8, 10-11, 14-18, AND 20.....	19
A. Motivation to Combine	19
B. Reasonable Expectation of Success.....	21
C. Independent Claim 1	22
1. 1[pre]: “A solid-state drive comprising:”	22
2. 1[a]: “a solid-state memory;”	23
3. 1[b]: “a device memory;”	24
4. 1[c]: “a programmable processor operatively coupled to the device memory;”	25
5. 1[d]: “a device controller operatively coupled with the solid-state memory and the programmable processor; and”	26
6. 1[e]: “an application stored on the device memory and comprising instructions that, when executed by the programmable processor, direct the programmable processor to:”	27
7. 1[e.i]: “process content for storage based on a storage policy;”	28
8. 1[e.ii]: “transfer the content to the device controller; and”	31
9. 1[f]: “the device controller to:”	33

10.	1[f.i]: “receive the content from the programmable processor; and”	34
11.	1[f.ii]: “store the content on the solid-state memory.”	34
D.	Dependent Claim 2.....	35
1.	2[pre]: “The solid-state drive of claim 1 wherein:”.....	35
2.	2[a]: “the device controller is to:”	35
3.	2[a.i]: “read requested content from the solid-state memory;”	35
4.	2[a.ii]: “transfer the requested content to the programmable processor; and”.....	36
5.	2[b]: “the instructions further direct the programmable processor to:”	37
6.	2[b.i]: “receive the requested content from the device controller;”	37
7.	2[b.ii]: “process the requested content for transfer based on the storage policy; and”	38
8.	2[b.iii]: “transfer the requested content.”	40
E.	Dependent Claim 5.....	40
1.	5[pre]: “The solid-state drive of claim 1 wherein: -”	40
2.	5[a]: “the instructions further direct the programmable processor to:”	40
3.	5[a.i]: “compress the content based on the storage policy;”	40
4.	5[a.ii]: “transfer the compressed content to the device controller; and”	42

5.	5[b]: “the device controller is to:”	42
6.	5[b.i]: “receive the compressed content from the programmable processor; and”	43
7.	5[b.ii]: “store the compressed content on the solid-state memory.”	43
F.	Dependent Claim 6	43
1.	6[pre]: “The solid-state drive of claim 5 wherein:”	43
2.	6[a]: “The device controller is to:”	43
3.	6[a.i]: “read the compressed content from the solid-state memory; and”	44
4.	6[a.ii]: “transfer the compressed content to the programmable processor; and”	44
5.	6[b]: “the instructions further direct the programmable processor to:”	44
6.	6[b.i]: “receive the compressed content from the device controller;”	45
7.	6[b.ii]: “decompress the content based on the storage policy; and”	45
8.	6[b.iii]: “transfer the decompressed content.”	46
G.	Dependent Claim 7	47
1.	7[pre]: “The solid-state drive of claim 1 wherein: -”	47
2.	7[a]: “the instructions further direct the programmable processor to:”	47
3.	7[a.i]: “encrypt the content based on the storage policy; and”	47

4.	7[a.ii]: “transfer the encrypted content to the device controller; and”	48
5.	7[b]: “the device controller is to:”	49
6.	7[b.i]: “receive the encrypted content from the programmable processor; and”	49
7.	7[b.ii]: “store the encrypted content on the solid-state memory.”	49
H.	Dependent Claim 8	49
1.	8[pre]: “The solid-state drive of claim 5 wherein:”	49
2.	8[a]: “The device controller is to:”	50
3.	8[a.i]: “read the encrypted content from the solid-state memory; and”	50
4.	8[a.ii]: “transfer the encrypted content to the programmable processor; and”	50
5.	8[b]: “the instructions further direct the programmable processor to:”	50
6.	8[b.i]: “receive the encrypted content from the device controller;”	51
7.	8[b.ii]: “decrypt the content; and”	51
8.	8[b.iii]: “transfer the decrypted content.”	52
I.	Dependent Claim 10: “The solid-state drive of claim 1 wherein the content comprises one or more of video data, image data, on-line temporal video data, cached data, financial data, online transaction data, credit card data, or medical data.”	53
J.	Independent Claim 11	53

1.	11[pre]: “A method of operating a solid-state drive, the method comprising:”	53
2.	11[a]: “in response to a storage request comprising content, processing the content based on a storage policy;”	55
3.	11[b]: “storing the processed content on the solid-state memory;”	57
4.	11[c]: “in response to a read request, reading requested content from the solid-state memory;”	58
5.	11[d]: “processing the requested content for transfer based on the storage policy; and”	59
6.	11[e]: “transferring the requested content.”	61
K.	Dependent Claim 14	61
1.	14[pre]: “The method of claim 11 wherein:”	61
2.	14[a]: “processing the content based on the storage policy comprises compressing the content based on the storage policy;”	61
3.	14[b]: “storing the processed content on the solid-state memory comprises storing the compressed content on the solid-state memory;”	62
4.	14[c]: “reading requested content from the solid-state memory comprises reading the compressed content from the solid-state memory;”	62
5.	14[d]: “processing the requested content for transfer based on the storage policy comprises decompressing the compressed content based on the storage policy; and”	62
6.	14[e]: “transferring the requested content comprises transferring the decompressed content.”	63

L.	Dependent Claim 15	63
1.	15[pre]: “The method of claim 11 wherein:”	63
2.	15[a]: “processing the content based on the storage policy comprises encrypting the content based on the storage policy;”	63
3.	15[b]: “storing the processed content on the solid-state memory comprises storing the encrypted content on the solid-state memory;”	64
4.	15[c]: “reading requested content from the solid-state memory comprises reading the encrypted content from the solid-state memory;”	64
5.	15[d]: “processing the requested content for transfer based on the storage policy comprises decrypting the encrypted content based on the storage policy; and”	64
6.	15[e]: “transferring the requested content comprises transferring the decrypted content.”	65
M.	Dependent Claim 16: “The method of claim 11 wherein the storage policy indicates how to process content for storage on the solid-state drive and how to process the requested content for transfer.”	65
N.	Dependent Claim 17: “The method of claim 16 wherein the storage policy comprises one or more of an encryption policy, a decryption policy, a compression policy, or a decompression policy.”	66
O.	Dependent Claim 18: “The method of claim 17 further comprising retrieving the storage policy from memory.”	67
P.	Independent Claim 20	67
1.	20[pre]: “A solid-state drive comprising:”	67
2.	20[a]: “a solid-state memory;”	68

3.	20[b]: “a device memory;”	69
4.	20[c]: “a programmable processor operatively coupled to the device memory;”	70
5.	20[d]: “a device controller operatively coupled with the solid-state memory and the programmable processor; and”	71
6.	20[e]: “instructions stored on the device memory that, when executed by the programmable processor, direct the programmable processor to:”	72
7.	20[e.i]: “process content for storage on the solid-state memory based on a storage policy;”	72
8.	20[e.ii]: “transfer the content to the device controller to store on the solid-state memory;”	73
9.	20[e.iii]: “in response a content request, receive requested content read from the solid-state memory by the device controller;”	73
10.	20[e.iv]: “process the requested content for transfer based on the storage policy; and”	74
11.	20[e.v]: “transfer the requested content.”	74

IX. GROUND 2: MEIR IN VIEW OF SALESSI AND CHOU RENDERS OBVIOUS CLAIM 9

A.	Motivation to Combine	75
B.	Reasonable Expectation of Success.....	76
C.	Dependent Claim 9.....	76
1.	9[pre]: “The solid-state drive of claim 1 further comprising:”	76

2. 9[a]: “a communication interface to communicatively couple the solid-state drive to a host device.”76
3. 9[b]: “a data link to communicatively couple the programmable processor and the device controller;”77
4. 9[c]: “a printed circuit board to mount the solid-state memory, the device memory, the device controller, the programmable processor, and the data link; and”78
5. 9[d]: “a case to mount the printed circuit board and the communication interface.”79

X. GROUND 3: MEIR IN VIEW OF SALESSI AND LOVINGER RENDERS OBVIOUS CLAIM 1981

- A. Motivation to Combine81
- B. Reasonable Expectation of Success.....82
- C. Claims.....84
 1. Claim 19: The method of claim 17 wherein the storage request comprises the storage policy.....84

XI. SECONDARY CONSIDERATIONS.....85

XII. CONCLUSION85

TABLE OF AUTHORITIES

	Page(s)
Cases	
<i>Activision Blizzard, Inc. v. Milestone Entertainment, LLC</i> , IPR2025-00708, Paper 13, 2 (Aug. 14, 2025)	2
<i>Caption Health, Inc. v. The Univ. of British Columbia</i> , IPR2025-01422, Paper 15, 3 (Director Dec. 18, 2025).....	18
<i>Gaea, LLC v. Meta Platforms, Inc.</i> , 4:26-cv-00348 (N.D. Tex) (filed on March 23, 2026)	3
<i>Gaea, LLC v. Samsung Electronics Co., Ltd.</i> , 2:26-cv-00264 (E.D. Tex.) (filed on March 31, 2026).....	3
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co. Ltd.</i> , 868 F.3d 1013 (Fed. Cir. 2017)	18
<i>Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.</i> , 200 F.3d 795 (Fed. Cir. 1999)	19
Statutes	
35 U.S.C. §§ 102(a)(1), 102(a)(2), and 102(b)(1).....	5
35 U.S.C. § 103.....	5, 6, 7
Other Authorities	
37 C.F.R. § 42.10(b)	4
37 C.F.R. § 42.15(a)	5
37 C.F.R. §42.24(a)	9
37 C.F.R. § 42.24(d)	9
37 C.F.R. §42.100(b)	18
37 C.F.R. § 42.104(a)	5

TABLE OF EXHIBITS

Exhibit No.	Description
1001	U.S. Patent No. 12,265,715 to Johnson, et al. (“the ’715 Patent”)
1002	File History of U.S. Patent No. 12,265,715
1003	Declaration of Heiner Litz (“Litz Decl.”)
1004	Curriculum Vitae of Heiner Litz
1005	U.S. Patent Application Publication No. 2015/0052167 to Olster et al. (“Olster”)
1006	U.S. Patent Publication No. 2016/0313943 to Hashimoto et al. (“Hashimoto”)
1007	U.S. Patent No. 9,075,851 to Kilian et al. (“Kilian”)
1008	U.S. Patent No. 8,874,835 to Davis et al. (“Davis”)
1009	U.S. Patent Application Publication No. 2012/0166749 to Eleftheriou et al. (“Eleftheriou”)
1010	U.S. Patent Application Publication No. 2012/0246435 to Meir et al. (“Meir”)
1011	U.S. Patent Application Publication No. 2015/0032938 to Salessi (“Salessi”)
1012	U.S. Patent Application Publication No. 2015/0146364 to Chou (“Chou”)
1013	RESERVED

Exhibit No.	Description
1014	RESERVED
1015	U.S. Patent Application Publication No. 2014/0089278 to Lovinger et al. (“Lovinger”)
1016	File History of U.S. Patent No. 10,776,023
1017	Chip Walter, ‘Kryder's Law' (2005)’ Scientific American, Aug. 2005.
1018	Timothy Feldman and Garth A. Gibson, ‘Shingled Magnetic Recording’ ACM Queue / CMU, 2013.
1019	Albert Chen and Jim Malina, ‘Addressing Shingled Magnetic Recording Drives with Linear Tape File System’ SNIA Storage Developer Conference, 2013
1020	Jorge Campello, ‘SMR: The Next Generation of Storage Technology’ SNIA Storage Developer Conference, 2015.
1021	Samarth Bhargava, ‘Lowering HAMR Near-Field Transducer Temperature via Inverse Electromagnetic Design’ IEEE Transactions on Magnetics, Apr. 2015.
1022	Yu Cai et al., ‘Read Disturb Errors in MLC Nand Flash Memory: Characterization, Mitigation, and Recovery’ DSN 2015.
1023	Nitin Agrawal et al., ‘Design Tradeoffs for SSD Performance’ USENIX Annual Technical Conference, 2008
1024	Feng Chen, David A. Koufaty, and Xiaodong Zhang; ‘Understanding Intrinsic Characteristics and System Implications of Flash Memory Based Solid State Drive’ SIGMETRICS 2009

Exhibit No.	Description
1025	Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, 'The Google File System' SOSP 2003
1026	Kirk McKusick and Sean Quinlan, 'GFS: Evolution on Fast-Forward,' ACM Queue, Aug. 2009 / Communications of the ACM, Mar. 2010
1027	Konstantin Shvachko et al., 'The Hadoop Distributed File System,' IEEE MSST 2010
1028	Sage A. Weil et al., 'Ceph: A Scalable, High-Performance Distributed File System' OSDI 2006
1029	Sage A. Weil et al., 'CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data' SC 2006
1030	Leslie Ashton et al., 'Two Decades of Policy-Based Storage Management for the IBM Mainframe Computer' IBM Systems Journal, 2003
1031	IBM, 'DFSMSHsm Primer'; IBM Redbooks, Sept. 2015
1032	IBM, 'GPFS: Advanced Administration Guide' including policy-based data management for GPFS, pre-2016 IBM documentation
1033	IBM, 'IBM Spectrum Scale (formerly GPFS)' IBM Redbooks, May 2015
1034	NVM Express, 'NVM Express Revision 1.0' Mar. 1, 2011
1035	NVM Express, 'NVM Express Revision 1.2' Nov. 3, 2014
1036	Open-Channel Solid State Drives, 'NVMe Specification Revision 1.2' Apr. 2016

Exhibit No.	Description
1037	Trusted Computing Group, ‘Storage Security Subsystem Class: Opal Specification Version 2.00’ Feb. 24, 2012
1038	T10, “SCSI Block Commands - 3 (SBC-3),” Working Draft Revision 25, Oct. 27, 2010
1039	Fred Knight and David L. Black, ‘SBC-3 Thin Provisioning Commands’ T10 proposal 08-356r5, Jan. 14, 2009
1040	T10, ‘SCSI Block Commands - 4 (SBC-4)’ draft index, including pre-priority revisions in 2014-2016
1041	David A. Patterson, Garth Gibson, and Randy H. Katz, ‘A Case for Redundant Arrays of Inexpensive Disks (RAID)’ SIGMOD 1988
1042	Peter M. Chen et al., ‘RAID: High-Performance, Reliable Secondary Storage’ ACM Computing Surveys, 1994
1043	John Colgrove et al., 'Purity: Building Fast, Highly-Available Enterprise Flash from Commodity Components,' SIGMOD 2015
1044	EMC Corporation, 'XtremIO Data Protection (XDP): Flash-Specific Data Protection,' White Paper, July 2014
1045	U.S. Patent Application Publication No. 2004/0230795 to Armitano (“Armitano”)
1046	U.S. Patent Application Publication Nos. 2007/0244939 and 2007/0244940 to Devarakonda (“Devarakonda”)
1047	U.S. Patent Application Publication No. 2011/0022642 to Vijayakumar (“Vijayakumar”)

Exhibit No.	Description
1048	U.S. Patent Application Publication No. 2010/0145917 to Bone (“Bone”)
1049	U.S. Patent Application Publication No. 2010/0332401 to Prahlad (“Prahlad”)
1050	U.S. Patent Application Publication No. 2014/0297695 to Dejana (“Dejana”)
1051	IBM, ‘Understanding Hard Drive Media Defects White Paper – Servers’ IBM Product Documentation, Dec. 1997

I. INTRODUCTION

Meta Platforms, Inc. (“Meta”) and Samsung Electronics Co., Ltd. (“Samsung”) (together, “Petitioners”) respectfully request *inter partes* review of claims 1-2, 5-11, and 14-20 of U.S. Patent No. 12,265,715 (EX1001, “’715 Patent”).

For decades, storage devices have used storage policies to determine how to store data. In the 1970s, for example, IBM implemented centralized, policy-based rules for its storage systems, which freed administrators from manually managing files. As storage systems advanced, so did storage policies, eventually allowing hosts (such as administrators, users, and client devices) to set storage policies by sending the policy to the storage device as opposed to relying on a policy set at manufacture. By 2016, storage policies, including user-configurable policies, were common and well-known.

During prosecution of the ancestor patent that originated the family of the ’715 Patent, the Examiner repeatedly rejected the pending claims as anticipated and obvious. Critically, the Examiner found an apparatus nearly identical to those claimed in the ’715 Patent—a storage device with a programmable processor, device controller, and solid-state memory that stores content based on a policy—anticipated by the prior art. The Examiner only allowed the claims after the Applicant amended the claims to recite receiving a storage policy “from a host” that “indicates a recording method for writing data in a storage media.” But this limitation is found

nowhere in the claims of the '715 Patent. Thus, the '715 Patent incorrectly issued—erroneously omitting the allowable subject matter of its ancestor.¹ *See Activision Blizzard, Inc. v. Milestone Entertainment, LLC*, IPR2025-00708, Paper 13, 2 (Aug. 14, 2025) (referring ultimately-instituted petition where “the challenged claims omit limitations added during prosecution of the parent patent that appear to have been the patent examiner’s reason for allowing the parent patent’s claims.”).

As such, Petitioners respectfully request that the Director institute *inter partes* review, and the Board find all the challenged claims of the '715 Patent unpatentable, as the Examiner found during the prosecution of the ancestor patent.

II. MANDATORY NOTICES

A. Real Parties-in-Interest

The real parties-in-interest are Meta Platforms, Inc., Samsung Electronics Co., Ltd., Samsung Electronics America, Inc., and Samsung Semiconductor, Inc.

¹ Though the challenged claims would still be unpatentable even if they did include this limitation, as shown in the Petitioners’ challenge against the ancestor patent. *See* IPR2026-00425, Paper 1 (challenging certain claims in U.S. Patent No. 10,776,023).

B. Related Matters

U.S. Patent Office records indicate that the '715 Patent is assigned to Gaea, LLC ("Patent Owner"), which is currently asserting the '715 Patent in the following concurrent litigations: *Gaea, LLC v. Meta Platforms, Inc.*, 4:26-cv-00348 (N.D. Tex) (filed on March 23, 2026) and *Gaea, LLC v. Samsung Electronics Co., Ltd.*, 2:26-cv-00264 (E.D. Tex.) (filed on March 31, 2026).

There are three patents asserted in the concurrent litigation. Petitioners have filed petitions for *inter partes* review against the other asserted patents at substantially the same time: U.S. Patent No. 10,776,023 (IPR2026-00425), and U.S. Patent No. 11,907,553 (IPR2026-00426).

C. Notice of Counsel and Service Information

Petitioners provide the following designation of counsel:

Lead Counsel
Lisa K. Nguyen (Reg. No. 58,018) Paul Hastings LLP 1117 S. California Avenue Palo Alto, CA 94303 Telephone: 650.320.1800 Fax: 650.320.1900 PH-META-Gaea@paulhastings.com
Back-Up Counsel

U.S. Patent No. 12,265,715
Petition for *Inter Partes* Review

David Tennant (Reg No. 48,362) Paul Hastings LLP 2050 M. Street NW Washington, D.C. 20036 Telephone: 202.551.1997 PH-META-Gaea@paulhastings.com	Joshua Nightingale (Reg. No. 67,865) Jones Day 500 Grant Street, Suite 4500 Pittsburgh, PA 15219 Telephone: 412.394.7950 Fax: 412.394.7959 jrnightingale@jonesday.com
Grace I. Wang (Reg. 69,892) Paul Hastings LLP 200 Park Avenue New York, NY 10166 Telephone: 212.318.6000 Fax: 212.319.4090 PH-META-Gaea@paulhastings.com	Stephanie Mishaga (Reg. No. 75,378) Jones Day 4655 Executive Drive, Suite 1500 San Diego, CA 92121 Telephone: 858.703.3140 Fax: 844.345.3178 smishaga@jonesday.com
Sasha Vujcic (Reg. 76,485) Paul Hastings LLP 4655 Executive Drive Suite 350 San Diego, CA 92121 Telephone: 858.458.3000 Fax: 858.458.3005 PH-META-Gaea@paulhastings.com	Robert Breetz (Reg. No. 75,550) Jones Day 901 Lakeside Ave. Cleveland, OH 44114 Telephone: 216-586-7213 Fax: 216.579.0212 rbreetz@jonesday.com
Allison Greubel (Reg. No. 82,758) Paul Hastings LLP 1117 S. California Avenue Palo Alto, CA 94303 Telephone: 650.320.1800 Fax: 650.320.1900 PH-META-Gaea@paulhastings.com	

Pursuant to 37 C.F.R. § 42.10(b), a Power of Attorney accompanies this petition. Petitioners consent to electronic service by e-mail.

D. Fee for *Inter Partes* Review

The Director is authorized to charge the fee specified by 37 C.F.R. § 42.15(a) to Deposit Account 50-2613.

E. Grounds for Standing

Petitioners certify pursuant to 37 C.F.R. § 42.104(a) that the '715 Patent is available for *inter partes* review and that Petitioners are not barred or estopped from requesting an *inter partes* review challenging the patent claims on the grounds identified in this Petition.

III. IDENTIFICATION OF CHALLENGES

Ground	Claims	Statute	Prior Art
1	1-2, 5-8, 10-11, 14-18, 20	35 U.S.C. § 103	Meir ² in view of Salessi ³
2	9	35 U.S.C. § 103	Meir in view of Salessi and Chou ⁴
3	19	35 U.S.C. § 103	Meir in view of Salessi and Lovinger ⁵

² Meir is prior art under 35 U.S.C. §§ 102(a)(1), 102(a)(2), and 102(b)(1).

³ Salessi is prior art under 35 U.S.C. §§ 102(a)(1), 102(a)(2), and 102(b)(1).

⁴ Chou is prior art under 35 U.S.C. §§ 102(a)(1), 102(a)(2), and 102(b)(1).

⁵ Lovinger is prior art under 35 U.S.C. §§ 102(a)(1), 102(a)(2), and 102(b)(1).

IV. BACKGROUND

A. Technology Overview

For decades, storage devices have used storage policies to determine how to store data. EX1003, ¶¶25-27. In the 1980s, user-configurable storage policies garnered popularity with the advent of hierarchical storage management. *Id.*, ¶¶35-38. These systems allowed users or administrators to define rules for automatically migrating data. *Id.* For example, IT administrators could direct storage devices to store data in different tiers, storing hot data (frequently used data) in the more expensive disk drives in one tier, while cold data (less frequently used or LFU data) in cheaper, long-term tape archives in another tier. *Id.*

In the late 1980s and 1990s, the invention of NAND memory (the chip inside SSDs) revolutionized data storage, allowing data to be written in large blocks rather than individually, leading to cheaper, higher-capacity storage. *Id.*, ¶¶25-27. With cheaper storage, more storage policies included rules and instructions related to redundancy schemes like RAID (Redundant Arrays of Independent Disks), which defined how content is stored, protected, and accessed across multiple drives. *Id.*, ¶¶39-40. Specifically, RAID can process content for storage by mirroring content (storing an exact copy of data in another drive), striping data (dividing data into blocks for storage), and calculating parity information (information that can be used to rebuild the data if a drive fails). *Id.* The advent of SSDs allowed RAID to be

implemented more cost-effectively, particularly when used in conjunction with compression (which reduced the size of the data) and encryption (which allowed failed drives to be discarded without risk that a third party could recover the data). *Id.*

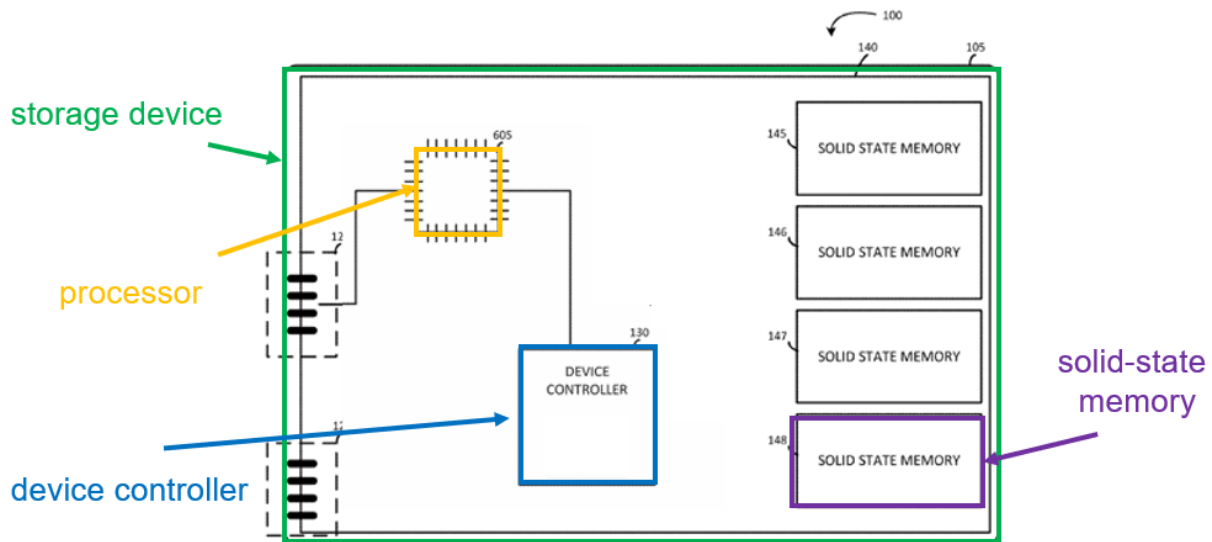
But a primary drawback of NAND memory is its limited lifespan—each memory cell can only endure a finite number of write and erase cycles before it wears out. *Id.*, ¶¶32-34. By the late 2000s when SSDs became mainstream, wear-leveling (evenly distributing write and erase cycles across all NAND memory cells) and bad block (defect) management became an important recording method for any NAND memory storage policy. *Id.*

Thus, by 2016 (the priority date of the '715 Patent), storage policies (including those defining RAID, compression, and encryption) were well-known, routine, and conventional.

B. '715 Patent

The '715 Patent describes well-known “data storage devices with configurable policy-based read and write behavior.” EX1001, 1:26-28. The '715 Patent complains that in prior art storage devices, storage decisions were “largely [] made by the manufacturer at the time of design and/or manufacture” providing users “very little to no influence” on how data is stored. *Id.*, 1:45-49. The '715 Patent purportedly addresses that problem using a routine storage device policy that allows

“a user to customize the operation of the storage device. *Id.*, 3:16-17.



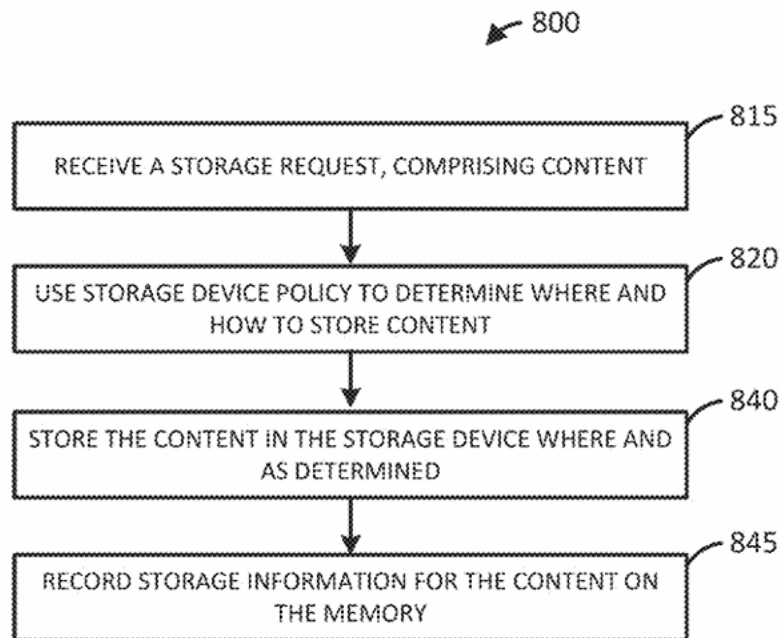
EX1001, Figs. 1B and 6 (modified and annotated)

The claimed **solid-state drive** includes a **processor**, **device controller**, and **solid-state memory**. *Id.*, FIGS. 1B, 6, 1:53-55. The '715 Patent discloses that the solid-state drive “uses a storage device policy...to determine where to store the content.” *Id.*, 14:41-43. “The determination can relate partially to the content being stored, and partially to the storage media.” *Id.*, 14:43-45.

The **processor**, such as programmable logic device 605, “can be programmed by user of the drive 600 for any operation need that is desired.” *Id.*, 11:21-23. Accordingly, the programmable processor processes the data according to the storage policy, but can also implement other processing operations such as “custom encryption through hardware” or “compression.” *Id.*, 11:23-41. The claims recite an application (comprising code or program instructions) stored on the **device**

memory, coupled to the programmable processor, that directs the programmable processor on how to operate.

The processor transfers and receives data to and from the solid-state memory through the device controller. The “programmable logic device 605 is shown connected between interface connector 125 and device controller 130.” *Id.*, 11:42-44.



EX1001, Fig. 8A

In Figure 8A, the '715 Patent describes one implementation of writing content according to a storage device policy. *Id.*, FIG. 8A, 14:31-15:5. “A user sends a request to store content in step 815.” *Id.*, 14:33-34. The '715 Patent explains that “this request comprises content to be stored.” *Id.*, 14:37-38. At step 820, the

programmable processor “uses a storage device policy...to determine where to store the content.” *Id.*, 14:41-42. The programmable processor will also process the content based on the storage policy. *Id.*, 11:21-36. At step 840, “the device controller can write the content to the storage media 110 as determined in step 820.” *Id.*, 14:59-60.

The solid-state drive can further read content according to a storage device policy, with the device controller reading the requested content from the solid-state memory, transferring the content to the programmable processor, which then processes the content according to the storage policy.

The solid-state drive can further include “case 105, which physically supports” the hardware components of the drive (*id.*, 3:33-36).

C. The Challenged Claims

The challenged claims are entitled to an effective date of no earlier than November 7, 2016.⁶ The ’715 Patent has 20 claims, including three independent claims (claim 1 and 20 directed to a solid-state drive, and claim 11 directed to a

⁶ Petitioners do not concede that any challenged claim is entitled to this priority date. For the purpose of this Petition, it is unnecessary to break the priority chain to a later date.

method of operating a solid-state drive) and 17 dependent claims. This Petition challenges all three independent claims and 12 dependent claims.

Independent claim 1 recites a solid-state drive with four components: a solid-state memory, a device memory, a programmable processor and a device controller, together with an application stored on the device memory. The application's instructions direct the programmable processor to perform a write operation: to process content for storage based on a storage policy and transfer the content to the device controller, which receives the content and stores it on the solid-state memory.

Independent claim 20 recites a solid-state drive with the same four components as claim 1. In addition to a write operation, the instructions further direct the programmable processor to perform a read operation: in response a content request, receive requested content read from the solid-state memory by the device controller; process the requested content for transfer based on the storage policy; and transfer the requested content.

Independent claim 11 recites the same general write and read operations as claim 20, but as method steps.

D. Prosecution History

1. Prosecution of the ancestor '023 Patent

The '715 Patent is a continuation from a line of continuations originating from the '023 Patent. During the prosecution of the '023 Patent, the Applicant struggled

to identify allowable subject matter. Specifically, the pending claims of the '023 Patent were repeatedly rejected on both §§ 102 and 103 grounds over U.S. Patent App. Pub. No. 2015/0052167 to Olster et al. ("Olster"). Notably, the Examiner found Olster anticipated pending claim 26, which substantially mirrors claim 1 of the '715 Patent:

26. An apparatus comprising:

a storage device;

a device controller operatively coupled with the storage device, comprising a memory, the device controller configured to receive content to be stored on the storage device;

a programmable processor, communicatively coupled with the device controller such that the content received by the device controller to be stored on the storage media must pass through the programmable processor; and

an application stored on the memory and comprising instructions that, when executed, direct the device controller to at least store the content on the storage media after the content has been processed by the programmable processor.

EX1016, 12. After some back and forth, the Examiner eventually allowed the claims after the Applicant amended them to require that the device controller receives "from a host" a storage policy that "indicates at least a recording method for writing content to the storage media." *Id.*, 224. The '023 Patent subsequently issued on September 15, 2020.

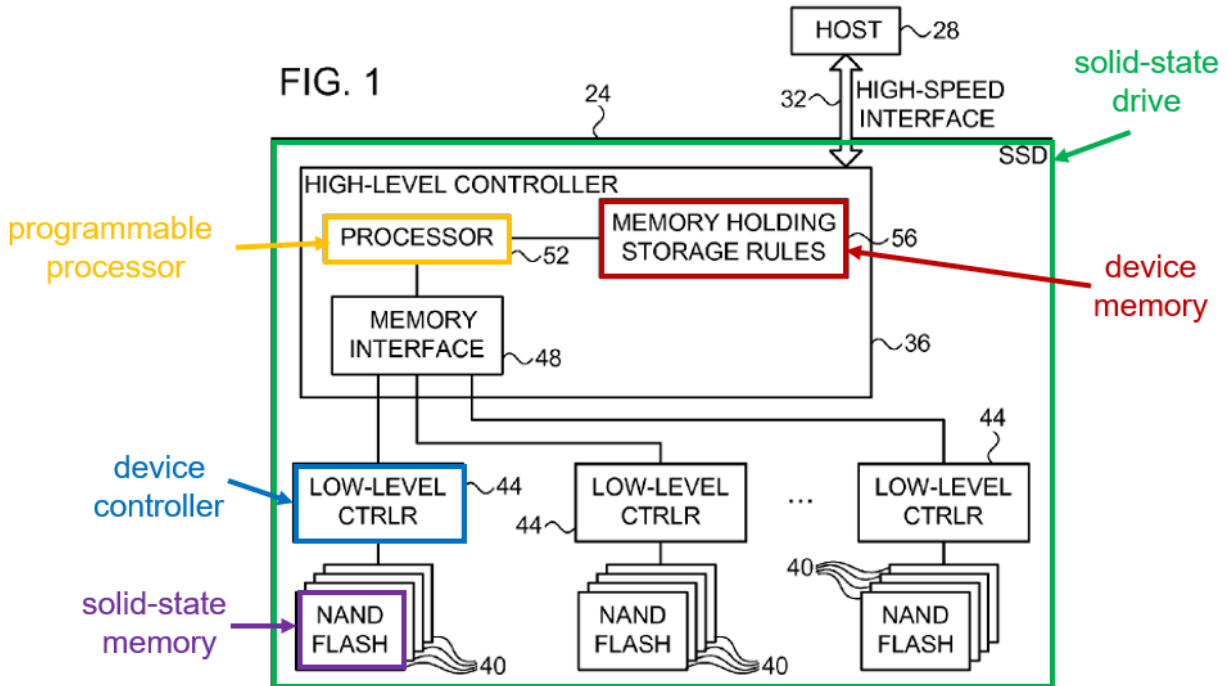
2. Prosecution of the '715 Patent

Although the Examiner only allowed the claims of the '023 Patent after the Applicant amended the claims to recite that the device controller receives “from a host” a storage policy that “indicates at least a recording method for writing content to the storage media,” none of the continuations following the '023 Patent included this limitation. Rather during the prosecution of the '715 Patent (and the other children of the '023 Patent), the Examiner allowed the claims after the Applicant terminally disclaimed other family members. Accordingly, the challenged claims of the '715 Patent issued with little (if any) substantive consideration.

V. THE PRIOR ART

A. Meir (EX1010)

Meir discloses each limitation of the independent claims of the '715 Patent except that it does not explicitly disclose that the application directs the programmable processor to *process content for storage* using *compression* and *decompression* or *encryption* and *decryption*. Like the '715 Patent, Meir discloses data storage devices with configurable policy-based read and write behavior. EX1010, Abstract (“A data storage method includes...holding a definition of a policy to be applied by the memory controller in the execution of the memory access commands in the memory unit”), [0055] (“one or more of the rules in the memory controller policy may be configurable”).



EX1010, Fig. 1 (annotated)

Meir discloses a **solid-state disk** (“SSD”) 24 which comprises a high-level controller 36 which includes **processor 52**, and **low-level controller 44** which is coupled with **memory units 40**. EX1010, FIG. 1, [0030] (“SSD 24 comprises a high-level controller 36, which stores data in one or more non-volatile memory units 40”), [0033] (“[h]igh-level controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example via low-level controllers 44), and a processor 52 that carries out the various functions of the high-level controller. High-level controller 36 comprises a memory 56...memory 56 holds a definition of a policy that the memory controller applies in storing data in memory units 40”).

Like the '715 Patent, Meir describes a solid-state drive with configurable policy-based read and write behavior. EX1010, [0014] (“a data storage method including receiving from a memory controller a communication...reporting a definition of a policy to be applied by the memory controller in accessing one or more memory units”). As Meir explains, “one or more of the rules in the memory controller policy may be configurable.” *Id.*, [0055].

Like the '715 Patent, the processor 52 is “programmed in software to carry out the functions described.” *Id.*, [0034]. Accordingly, the programmable processor processes the data for storage according to the storage policy including rules applied to the data such as “applying a redundant storage scheme such as Redundant Array of Inexpensive Disks.” *Id.*, [0025]. “[R]edundancy rules may specify whether such a scheme is applied or not, as well as various parameters of the redundant storage scheme.” *Id.*, [0048]. For example, the parameters may include “the formulas or rules used by the memory controller in calculating parity information over the data.” *Id.* Software (comprising program instructions) can be stored along with the storage policy in memory 56, which is coupled to processor 52 and can direct processor 52 how to operate. *Id.*, [0034] (“The software may be downloaded to the processors...and/or stored on non-transitory tangible media, such as magnetic, optical, or electronic memory”).

B. Salessi (EX1011)

Salessi discloses *process[ing] the content*, which includes compressing or encrypting the content for storage and decompressing or decrypting the content upon retrieval.

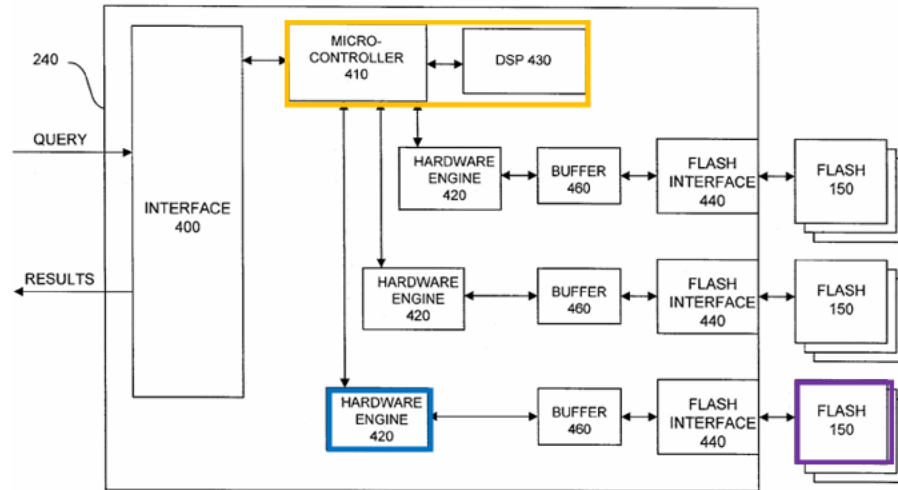


FIG. 7

EX1011, Fig. 7

Like the '715 Patent and Meir, Salessi's SSD that "includes flash memory and an SSD processing unit." EX1011, FIG. 7, [0006]. Salessi's processing unit "process[es] the data...being stored or retrieved from the flash memory." *Id.*, [0006]. "Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression." *Id.* Salessi further teaches that the processing unit "us[es] a processing algorithm[] to generate processed data." *Id.*, [0023].

C. Chou (EX1012)

Chou discloses a printed circuit board mounting components within a case, as required in certain dependent claims. Like the '715 Patent and Meir, Chou discloses a solid-state drive (SSD). Chou's SSD includes a printed circuit board assembly ("PCBA") and a housing that supports the PCBA. EX1012, [0006] ("A SSD generally includes a printed circuit board assembly (PCBA) within a metallic housing."), [0034] ("To assemble the SSD 100, the PCBA 112 is placed inside the upper and lower covers 114 a and 114 b"). Chou also describes "[o]ne or more memory modules and other electronic components 122," which can be the solid-state memory, device memory, programmable processor, device controller and other components, are mounted "on the PCBA." *Id.*, [0033].

D. Lovinger (EX1015)

Lovinger discloses a storage request that includes a policy, as required by claim 19. Like the '715 Patent and Meir, Lovinger discloses a storage device for storing data. EX1015, Abstract. Lovinger discloses a method "for storing content based on one or more attributes associated with the content." EX1015, [0003]. The client "may transmit content...with information corresponding to the retention policy." *Id.*, [0040]. "[W]hen the content and retention policy have been received, the server 120 writes the content 112 to one or more storage devices associated with the server 120 according to the retention policy." *Id.* This enables a client "to

store content according to the retention policy without reprogramming device firmware.” *Id.*, [0041].

VI. LEVEL OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art (“POSITA”) in 2016 would have had at least a bachelor’s degree in electrical engineering, computer science, computer engineering, or a similar technical field, with at least two years of experience in the field of data storage systems. Additional experience could substitute for less education, and additional education could likewise substitute for less experience. EX1003, ¶¶55-57.

This Petition does not turn on this precise definition, and the challenged claim would be unpatentable from the perspective of any reasonable person of ordinary skill in the art at the relevant time. *Id.*

VII. CLAIM CONSTRUCTION

The Board construes the claims “using the same claim construction standard that would be used” in district courts. 37 C.F.R. §42.100(b). This Petition establishes the prior art meets each of the claim limitations under its plain and ordinary meaning.⁷ Thus, the claim terms require no express construction. *Nidec*

⁷ Petitioners reserve the right to provide an express construction consistent with that applied in this proceeding and the Parallel Litigation. *See Caption Health, Inc. v.*

Motor Corp. v. Zhongshan Broad Ocean Motor Co. Ltd., 868 F.3d 1013, 1017 (Fed. Cir. 2017); *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999).

VIII. GROUND 1: MEIR IN VIEW OF SALESSI RENDERS OBVIOUS CLAIMS 1-2, 5-8, 10-11, 14-18, AND 20

The data storage system resulting from the combination of the teachings of Meir-Salessi is a SSD including a processor that executes access commands from a host according to a policy that includes algorithms and rules for compression, decompression, encryption, and decryption.

A. Motivation to Combine

A POSITA would have been motivated to modify the software (*application*) and policy (*policy*) stored in memory 56 (*device memory*) of Meir to direct the processor 52 to compress and encrypt the data to be stored according to algorithms and rules in the policy as taught by Salessi. EX1003, ¶¶71-82.

First, Salessi expressly teaches that compression and encryption rules may be used in the implementation of redundancy schemes like the one taught in Meir. EX1011, [0057] (compression and encryption “may be used in a redundant array of independent disks (RAID) to achieve similar advantages in optimizing performance

The Univ. of British Columbia, IPR2025-01422, Paper 15, 3 (Director Dec. 18, 2025).

and resource utilization.”); EX1010, [0025]. A POSITA implementing Meir’s system would have naturally looked at teachings improving the performance of solid-state drives, particularly, those implementing RAID schemes.

Second, it would have been well-understood by a POSITA that modifying policies to include compression and encryption would improve the performance and security of the data storage system, respectively. EX1010, [0036]; EX1011, [0050]; EX1003, ¶¶75-77.

Third, modifying Meir’s policy to include Salessi’s compression and encryption algorithm and rules would also result in cost and resource savings. EX1003, ¶78. Because compression allows the data storage system to store more data, an administrator could use compression to expand the storage capacity of the system without expending resources on additional drives. *Id.* Further, encryption conserves resources that may be expended in ensuring that sensitive data is protected when a drive fails. *Id.*

Fourth, storing less data through compression reduces wear on SSDs. EX1003, ¶80. A major drawback of NAND memory (the chip inside SSDs) is its limited lifespan—each memory cell can only endure a finite number of write and erase cycles before it wears out. *Id.* Reducing wear will increase the life of the drives and reduce drive failure.

Fifth, modifying the policy to include encryption would have been driven by regulatory compliance requirements. EX1003, ¶¶81-82. At the relevant time, privacy mandates such as GDPR had just been enacted. *Id.* Encryption would have provided an efficient approach to render certain personal data requiring protection to be unreadable. *Id.*

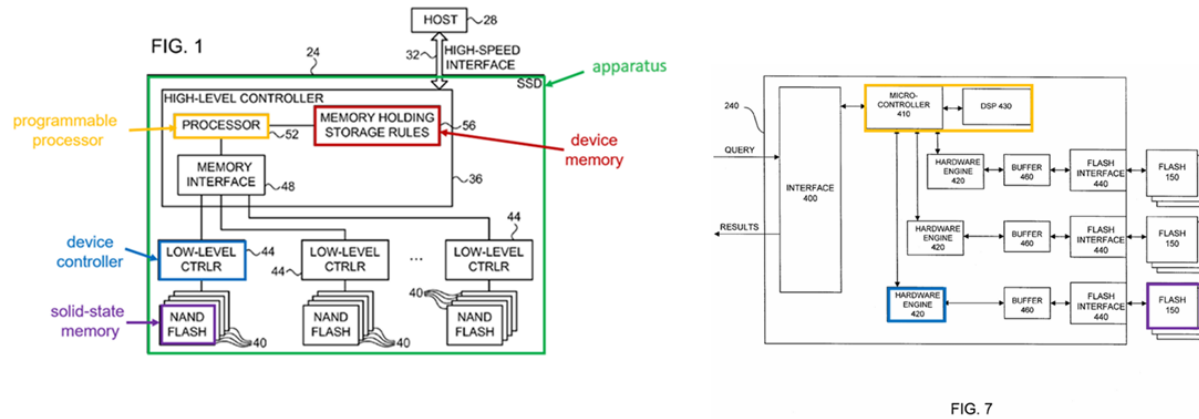
B. Reasonable Expectation of Success

A POSITA would have been able to modify Meir's processor 52 to perform compression, decompression, encryption, and decryption according to the storage policy with a reasonable expectation of success. EX1003, ¶¶83-89.

First, the storage drives of Meir and Salessi have similar structures and components. EX1010, FIG.1; EX1011, FIGS. 2, 4, 7; EX1003, ¶84. Meir and Salessi's SSDs both include memory controller or processing unit that interfaces with the solid-state memory. EX1010, [0031], [0033]; EX1011 [0047], [0055]; EX1003, ¶84.

Second, Meir and Salessi both teach optimization of performance through parallelism and pipeline rules utilizing multiple processors. EX1010, [0042]; EX1011, [0055]. Specifically, Meir teaches a high-controller 36 including processor 52 (*programmable processor*) that interfaces with low-level controllers 44 (*device controller*) and Salessi teaches a microcontroller 410 (*programmable processor*) that interfaces with hardware engines 420 (*device controller*), where the one processor

executes various operations including processing data and the other interfaces with memory. EX1010, [0030], [0033]; EX1011, [0042], [0047], [0055]; ; EX1003, ¶85.



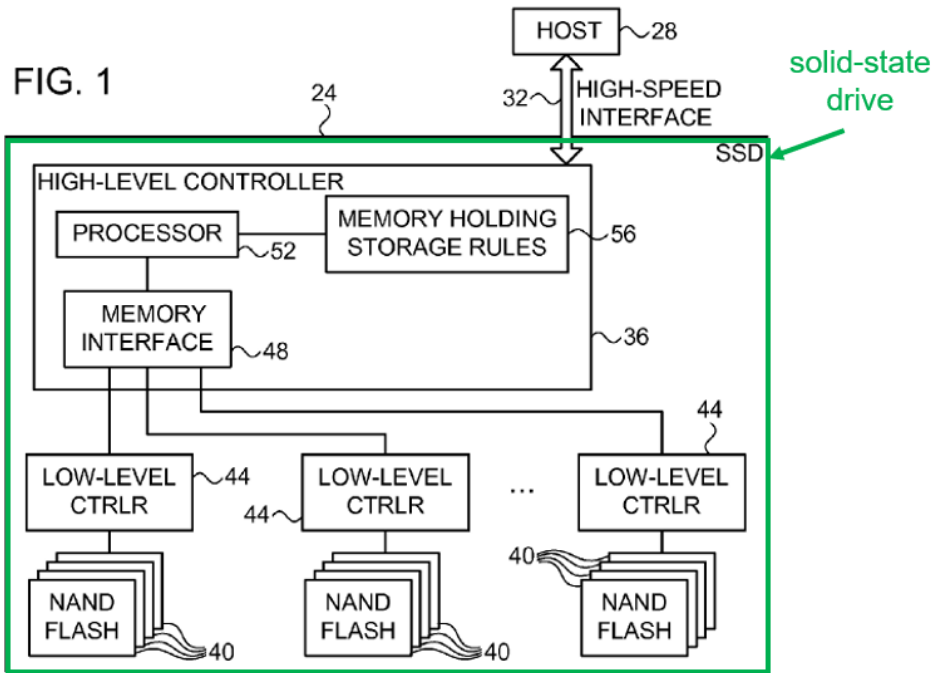
EX1010, Fig. 1 (annotated); EX1011, Fig. 7 (annotated)

Third, the modification would be a simple change of programming. Indeed, Meir teaches that “a given rule in the policy is configurable.” EX1010, [0011]. Further, Salessi teaches that compression, decompression, encryption, and decryption are implemented through “processing algorithm[s].” EX1011, [0007]-[0011]. Updating the policy of Meir to include the processing algorithms of Salessi would have been well within the skill set of a POSITA at the relevant time. EX1003, ¶¶84-89.

C. Independent Claim 1

1. 1[pre]: “A solid-state drive comprising:”

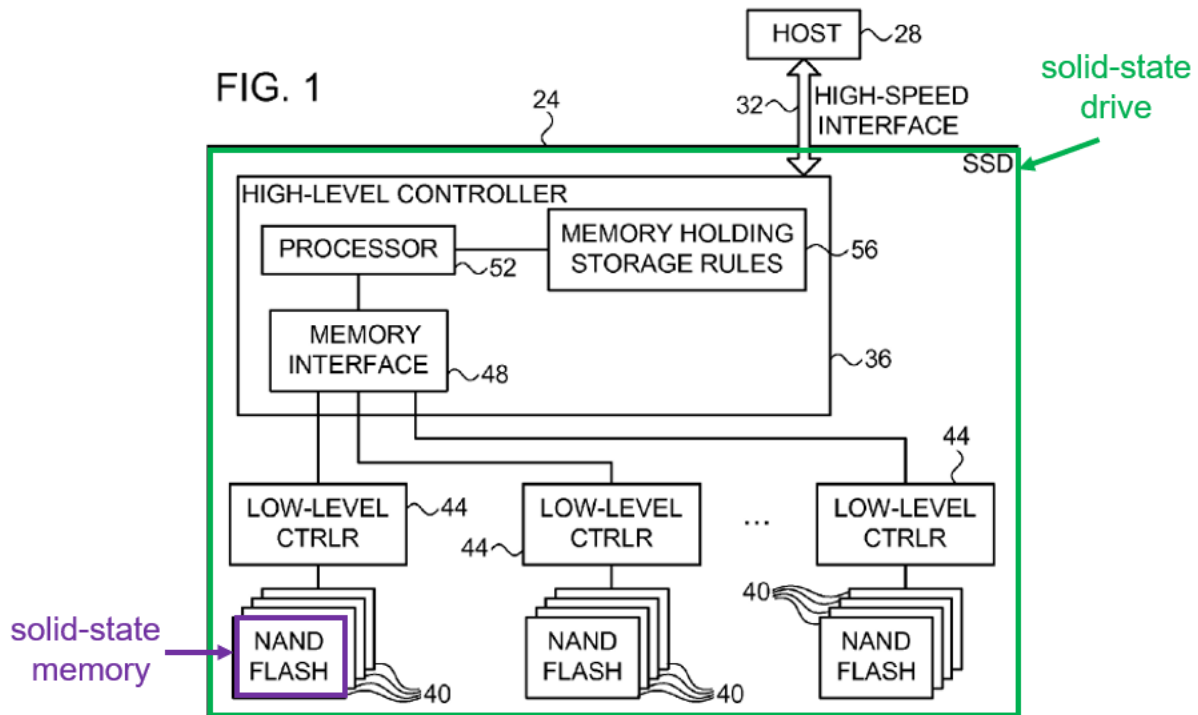
To the extent limiting, Meir discloses the preamble. EX1003, ¶90. Meir discloses a solid-state drive (“SSD”) 24 (*solid-state drive*). EX1010, FIG. 1, [0029].



EX1010, Fig. 1 (annotated)

2. 1[a]: “a solid-state memory;”

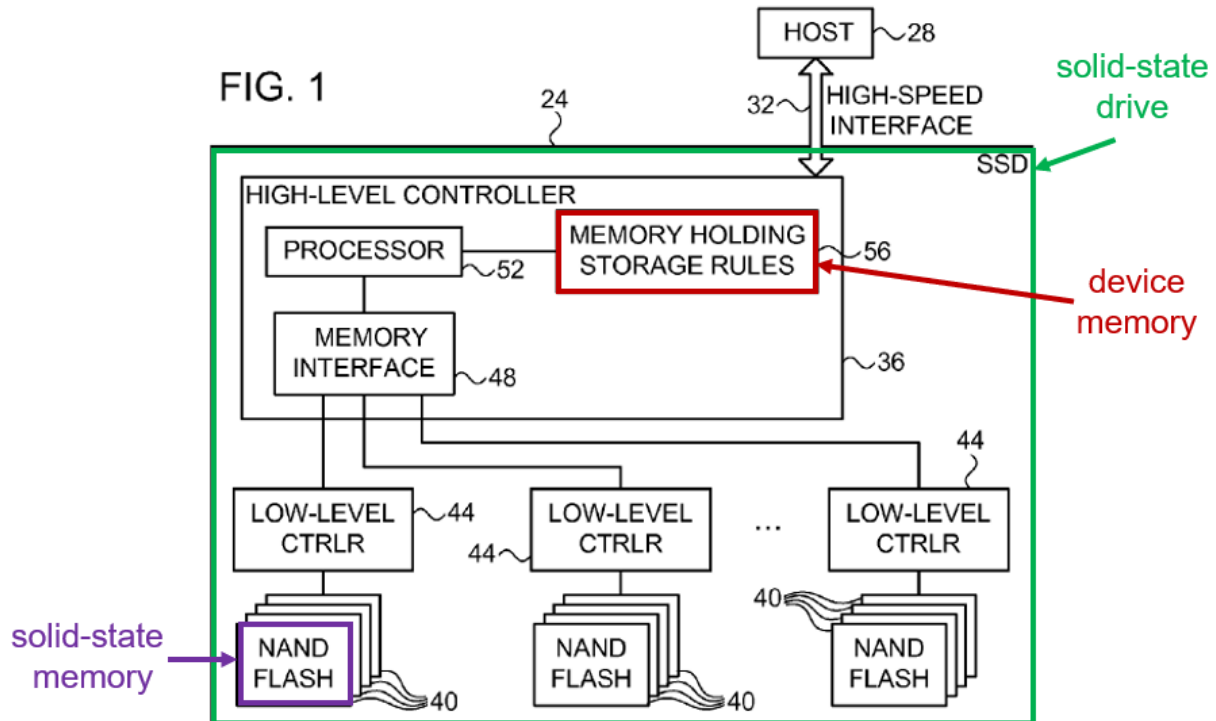
Meir discloses this limitation. EX1003, ¶91. Meir’s SSD 24 comprises memory units 40 (*solid-state memory*). EX1010, FIG. 1, [0030]. “[M]emory units 40 comprise NAND Flash devices or dies” or “any other suitable type of non-volatile memory units.” *Id.* NAND Flash and other non-volatile memories are types of solid-state memories. EX1003, ¶91.



EX1010, Fig. 1 (annotated)

3. 1[b]: “a device memory;”

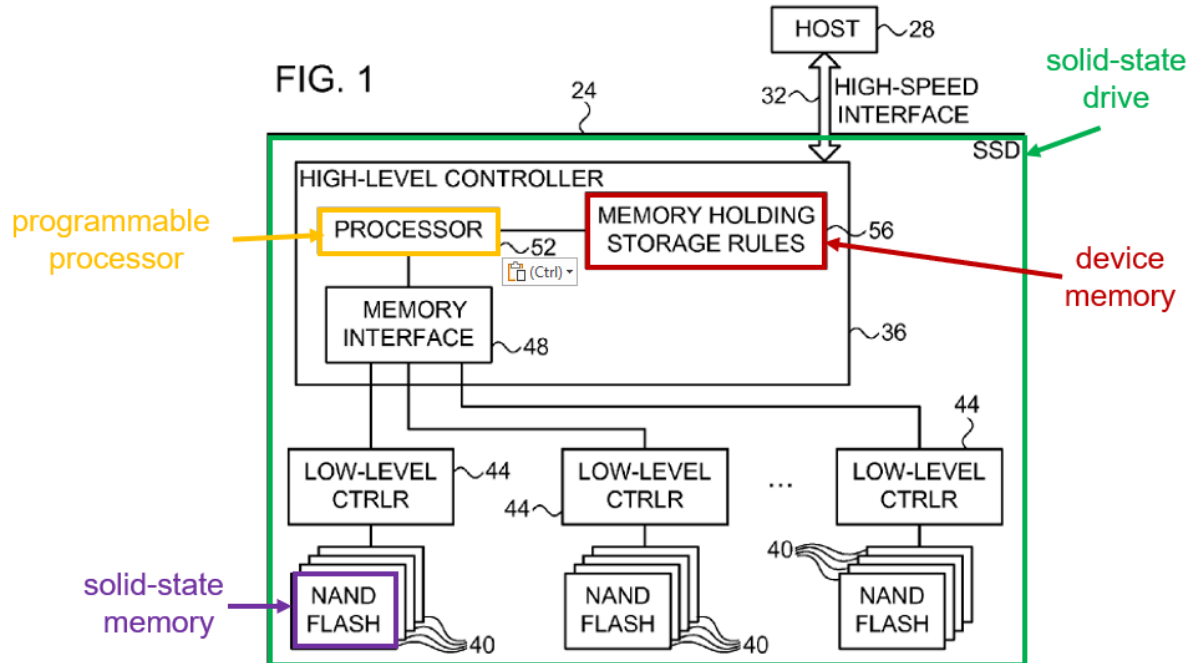
Meir discloses this limitation. EX1003, ¶92. Meir discloses memory 56 (*device memory*). EX1010, FIG. 1, [0033] (“memory 56 holds a definition of a policy that the memory controller applies in storing data in memory units 40”).



EX1010, Fig. 1 (annotated)

4. 1[c]: “a programmable processor operatively coupled to the device memory;”

Meir discloses this limitation. EX1003, ¶93. Meir discloses processor 52 (*programmable processor*). EX1010, FIG. 1, [0033] (“a processor 52 that carries out the various functions of the high-level controller”). Processor 52 (*programmable processor*) is *operatively coupled to* memory 56 (*device memory*) as shown by the connection between these two components in Figure 1. *Id.*; EX1003, ¶93.

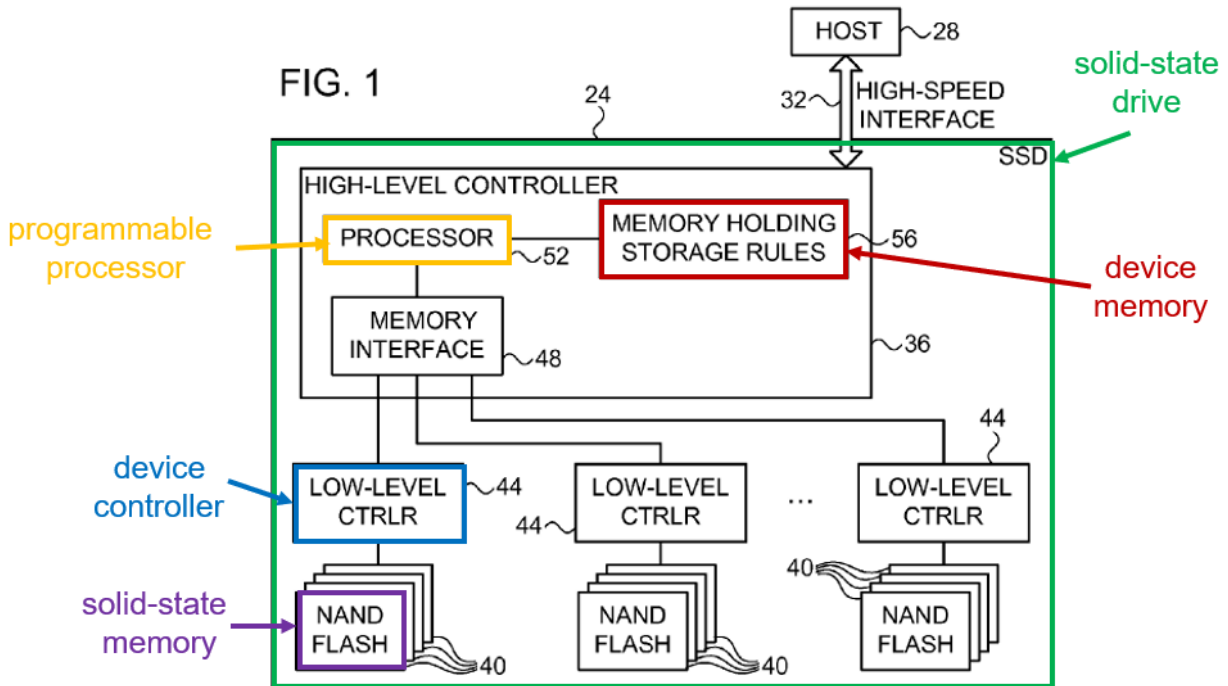


EX1010, Fig. 1 (annotated)

5. 1[d]: “a device controller operatively coupled with the solid-state memory and the programmable processor; and”

Meir discloses this limitation. EX1003, ¶¶94-95. Meir discloses a low-level controller 44 (*device controller*). *Id.*, FIG. 1, [0031]. Low-level controller 44 (*device controller*) is *operatively coupled with* memory unit 40 (*solid-state memory*) as shown by the connection between these two components in Figure 1. *Id.*; EX1003, ¶95. Low-level controller 44 (*device controller*) is also *operatively coupled to* processor 52 (*programmable processor*) through memory interface 48 which allows processor 52 to communicate memory units 40 via the low-level controller 44. *Id.*; EX1003, ¶95; *see also* EX1010, [0033] (“High-level

controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example via low-level controllers 44), and a processor 52 that carries out the various functions of the high-level controller.”).



EX1010, Fig. 1 (annotated)

6. 1[e]: “an application stored on the device memory and comprising instructions that, when executed by the programmable processor, direct the programmable processor to:”

Meir in view of Salessi discloses this limitation. EX1003, ¶¶96-99. Meir discloses “the memory controller may...run suitable software” (*an application...when executed by the programmable processor, direct the programmable processor*). EX1010, [0034]; *see also* [0034] (“programmed in software to carry out the functions described”), [0032]. Meir further discloses that

“[t]he software may be downloaded to the processors in electronic form, over a network, for example, or it may, alternatively or additionally, be provided and/or stored on non-transitory tangible media, such as magnetic, optical, or electronic memory,” confirming that it can be stored in the memory 56 (*device memory*) for processor 52 to execute. *Id.*; EX1003, ¶¶96-97. Software comprises program instructions (*comprising instructions*) that perform processes when executed. EX1003, ¶97.

As discussed further in 1[e.i], a POSITA would have been motivated to modify the software (*application*) to direct the processor 52 to perform compression/decompression and encryption/decryption in accordance with algorithms and rules defined by the policy. EX1010, [0027]; EX1011, [0057] (“these embodiments can be used to make RAID controllers and subsystems more efficient”); *see* Section VIII.C.7; EX1003, ¶99.

7. 1[e.i]: “process content for storage based on a storage policy;”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶100-109. Meir discloses that the software (*application*) directs the processor 52 to *process* the data for storage (*content for storage*) according to the policy (*based on the storage device policy*), such as by applying a redundancy scheme.

Meir teaches that “SSD 24 comprises a high-level controller 36, which stores data in one or more non-volatile memory units 40.” EX1010, [0030]. This data is

content for storage. EX1003, ¶100. Meir explains that “[h]ost 28 typically sends memory access commands (e.g., read and write commands) to SSD 24 over interface 32” and “[t]he memory system comprises a memory controller which receives from the host memory access commands (e.g., read and write commands) and executes the commands in the memory units.” EX1010, [0030], [0020].

Meir discloses that the memory controller will *process* commands *based on a storage policy* because “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021], [0033]; EX1003, ¶¶101-102. “The policy comprises a set of rules, for example parallelization rules, pipeline rules, prioritization rules, redundancy rules, power consumption rules and/or memory unit characteristics rules.” *Id.*, [0037].

When applying the redundancy rules from the policy, the software (*application*) can direct the processor 52 to *process* the data (*content for storage*) by making a copy of the data to store a redundant copy in other memory units, striping the data dividing it into blocks, or calculating parity information on the data. *Id.*, [0025] (“[t]he memory controller policy involves storing the data while applying a redundant storage scheme such as Redundant Array of Inexpensive Disks (RAID) among the memory units”); EX1003, ¶102.

To the extent that Meir does not disclose that the software executed by processor 52 (*instructions*) *process* the data (*content for storage*) included with the commands *based on a storage policy*, this is rendered obvious by the system resulting from the combination of the teachings of Meir and Salessi.

Like Meir's SSD (which includes memory units 40 and a memory controller), Salessi's SSD "includes flash memory and an SSD processing unit." EX1011, FIG. 7, [0006]; *see also* EX1010, FIG. 1. Salessi's processing unit can *process content for storage* because it is "capable of processing the data...being stored or retrieved from the flash memory." EX1011, [0006]. Salessi's data includes *content*, such as social media posts and pictures and video content, among other types of content. EX1011, [0002]. "Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression." *Id.* Salessi further teaches that the processing unit "us[es] a processing algorithm[] to generate processed data." *Id.*, [0023].

In the combination, Meir's policy stored in memory 56 includes Salessi's compression, decompression, decryption, and encryption algorithms and instructions. Specifically, Meir teaches that the "one or more of the rules in the memory controller policy may be configurable" and "the host may issue to the memory controller a command that configures or modifies the memory controller, so as to improve performance." EX1010, [0055]. A POSITA would therefore

modify the memory controller policy of Meir to incorporate the processing algorithms and instructions of Salessi, so that Meir's policy would define which data to apply Salessi's processing algorithms to. EX1011, [0006]; EX1003, ¶¶107.

As discussed, a POSITA would have been motivated to modify the software (*application*) and *policy* store in memory 56 (*policy*) of Meir to direct the processor 52 to compress and encrypt the data to be stored according to algorithms and rules in the policy because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶108. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components, and Salessi expressly teaches that the use of its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, 0033; EX1011, FIG. 7, [0004]; EX1003, ¶109.

**8. 1[e.ii]: “transfer the content to the device controller;
and”**

Meir discloses this limitation. EX1003, ¶¶110-115.

In Meir, the processor 52's software (*application*) sends the data (*transfer the content*) to the low-level controller 44 (*device controller*) to store. EX1003, ¶111. In certain embodiments, Meir teaches that the storage policy defines parallelism rules. EX1010, [0037]. “Parallelism rules define how the memory controller distributes the memory access commands among the memory units.” *Id.*

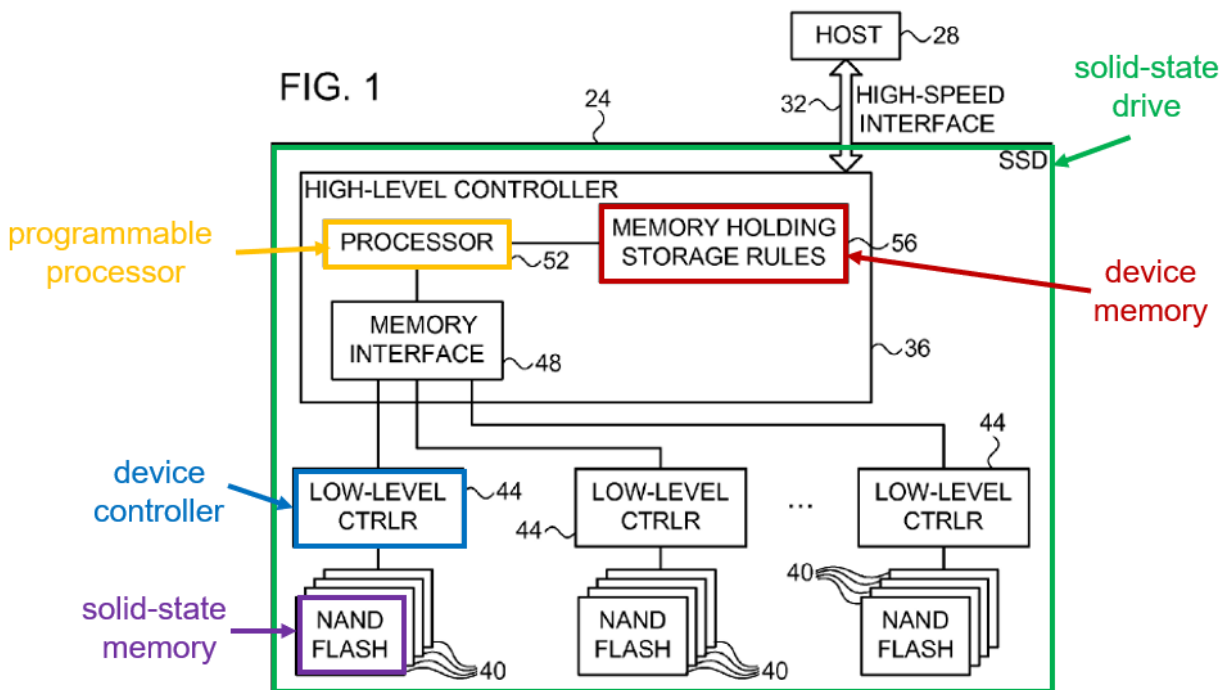
When a write command is received, the processor 52 will process the command according to the policy and transfer the data to the assigned low-level controller 44. *Id.*, [0005], [0041]. Specifically, the parallelism rules defined by the policy will “specify distribution of the memory access commands among the memory units.” *Id.*, [0005] “[A] mapping, defined in the memory controller, between logical addresses associated with the memory access commands and respective physical storage location in the memory units” will be used by processor 52 via its software to direct the access command to the appropriate low-level controller 44 (the low-level controller assigned to those memory addresses). *Id.*, [0007], [0031] (“each low-level controller assigned to a group of one or more memory units 40”); *see also* [0009], [0023], [0031] (“logical-to-physical address translation is performed by the high-level controller”). Specifically, because “a parallelism rule may define...ranges of LBAs that are mapped to respective low-level controllers,” processor 52 will use the parallelism rule that is part of the policy to identify the assigned low-level controller 44 to send the data to be stored.” *Id.*, [0041]; *see also id.* (“a certain subset of bits in the logical address may indicate the memory unit or low-level controller in which this address is assigned”); EX1003, ¶¶112-113.

Because the data to be stored is transferred from the host to high-level controller 36, the software (*application*) of processor 51 necessarily sends the data

(*transfers the content*) to the low-level controller 44 (*device controller*). *Id.*, FIG. 1, [0033]. As Meir explains, “[h]igh-level controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example *via low-level controllers 44*).” *Id.*, [0033] (emphasis added); EX1003, ¶¶114-115.

9. 1[f]: “the device controller to:”

Meir discloses this limitation. EX1003, ¶116. As explained for 1[d], the solid-state drive 24 includes low-level controller 44. *See* Section VIII.A.5; EX1010, FIG. 1; EX1003, ¶¶94-95.



EX1010, Fig. 1 (annotated)

10. 1[f.i]: “receive the content from the programmable processor; and”

Meir discloses this limitation. EX1003, ¶117. As explained for 1[e.ii], the software (*application*) of processor 52 (*programmable processor*) necessarily sends the data to the assigned low-level controller 44 (*device controller*). See Section VIII.A.8; EX1010, FIG. 1; EX1003, ¶¶110-115. In doing so, the low-level controller 44 (*device controller*) necessarily receives the data for storage (*receive the content*) from processor 52 (*programmable processor*). EX1010, FIG. 1; EX1003, ¶¶110-115, 117.

11. 1[f.ii]: “store the content on the solid-state memory.”

Meir discloses this limitation. EX1003, ¶¶118-120. Meir explains that the “memory controller...executes the commands in the memory units.” EX1010, [0020]. In the embodiment where the memory controller comprises high-level controller 36 and low-level controller 44 (*device controller*), the high-level controller 36 communicates with memory units 40 (*solid-state memory*) via low-level controller 44 (*device controller*). *Id.*, [0033] (“High-level controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example via low-level controllers 44), and a processor 52 that carries out the various functions of the high-level controller.”). Accordingly, when the memory controller executes a write command, the low-level controller 44 (*device controller*) writes the data (*stores the content*) to the memory

unit 40 (*on the solid-state memory*) according to the logical-to-physical address mapping defined by the parallelism rule that is part of the policy. *Id.*, [0040] (“According to the policy, the memory controller translates the logical addresses to respective physical storage locations in memory units 40, and executes the commands in the memory units according to the logical-to-physical address mapping.”).

A POSITA would have been motivated to use the implementation of Meir from Figure 1, because Meir explains that advantages of partitioning the execution of a command between the high-level controller 36 (as executed by processor 52 or *programmable processor*) and low-level controller 44. EX1003, ¶¶119-120.

D. Dependent Claim 2

1. 2[pre]: “The solid-state drive of claim 1 wherein:”

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1. *See* Section VIII.C.1; EX1003, ¶121.

2. 2[a]: “the device controller is to:”

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*). *See* Section VIII.C.5; EX1003, ¶¶94-95, 122.

3. 2[a.i]: “read requested content from the solid-state memory;”

Meir discloses this limitation. EX1003, ¶¶123-125. When Meir’s memory controller executes a read command, the processor 52 of high-level controller 36

“translates the logical addresses to respective physical storage locations in memory units 40, and executes the commands in the memory units according to the logical-to-physical address mapping.” EX1010, [0040], [0020], [0033]. In one embodiment, “a certain subset of bits in the logical address may indicate the...low-level controller in which this address is assigned” and after translation, the high-level controller 36 will direct that assigned low-level controller 44 (*device controller*), to execute the access command, here the read command (*read requested content from the solid-state memory*). *Id.*, [0044]; EX1003, ¶¶124-125.

In certain embodiments, the SSD 24 may use error correction code (“ECC”) to encode the data (*content*) to prevent errors. EX1010, [0045]. When a read command is received, the low-level controller 44 will use the processing circuitry in the memory units 40 to fetch the ECC-encoded data (*read requested content from the solid-state memory*). *Id.* The low-level controller 44 will then decode the data. *Id.*; EX1003, ¶¶124-125.

4. 2[a.ii]: “transfer the requested content to the programmable processor; and”

Meir discloses this limitation. EX1003, ¶126. Meir discloses that after the low-level controller 44 (*device controller*) decodes the requested data (*requested content*), the low-level controller 44 will send the decoded data (*transfer the requested content*) to the processor 52 (*programmable processor*) of high-level controller 36. EX1010, [0045]; EX1003, ¶126.

5. 2[b]: “the instructions further direct the programmable processor to:”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶127. As discussed in 1[e], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to* execute the access commands according to the Meir-Salessi policy stored on the memory 56. See Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99.

6. 2[b.i]: “receive the requested content from the device controller;”

Meir discloses this limitation. EX1003, ¶¶128-129. Meir discloses that the software comprising *instructions further directs processor 52 (programmable processor) to receive the decoded data (requested content) from low-level controller 44 (device controller)*. EX1010, [0044], [0045]. Meir explains that when a “read command is broken into three pipeline stages,” “decoding the ECC” can be performed by the low-level controller 44, which then sends the decoded data to the processor 52 (*programmable processor*) of the high-level controller 36. *Id.*, [0045]; EX1003, ¶129. The step of “sending the data to the host” can then be performed by processor 52 (*programmable processor*) as directed by the software comprising *instructions*. *Id.*

7. 2[b.ii]: “process the requested content for transfer based on the storage policy; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶130-137.

Meir discloses that the memory controller will *process* commands *based on a storage policy* because “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021], [0033] (“memory 56 holds a definition of a policy that the memory controller applies in storing data in memory units 40”); EX1003, ¶¶130-131. “The policy comprises a set of rules, for example parallelization rules, pipeline rules, prioritization rules, redundancy rules, power consumption rules and/or memory unit characteristics rules.” *Id.*, [0037]; *see also id.*, [0025] (“[t]he memory controller policy involves storing the data while applying a redundant storage scheme such as Redundant Array of Inexpensive Disks (RAID) among the memory units”). A POSITA would have appreciated that implementing RAID would have also involved processing when reading data. EX1010, [0025], [0038]; EX1003, ¶¶131-132.

To the extent that Meir does not disclose this limitation, this is rendered obvious by the system resulting from the combination of the teachings of Meir-Salessi. Like Meir’s SSD (which includes memory units 40 and a memory controller), Salessi teaches an SSD that “includes flash memory and an SSD processing unit.” EX1011, FIG. 7, [0006]; *see also* EX1010, FIG. 1. Salessi’s processing unit can *process requested content* because it is “capable of processing

the data...being stored or retrieved from the flash memory.” EX1011, [0006]. “Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” *Id.* Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” *Id.*, [0023].

In the combination, Meir’s policy stored in memory 56 includes Salessi’s compression, decompression, decryption, and encryption algorithms and instructions. Specifically, Meir teaches that the “one or more of the rules in the memory controller policy may be configurable” and “the host may issue to the memory controller a command that configures or modifies the memory controller, so as to improve performance.” EX1010, [0045]. A POSITA would therefore modify the memory controller policy of Meir to incorporate the processing algorithms and instructions of Salessi, wherein Meir’s policy would define to which data to apply Salessi’s processing algorithms. EX1011, [0006]; EX1003, ¶¶133-135.

As discussed, a POSITA would have been motivated to make this modification because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶136. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components,

and Salessi expressly teaches that the use of its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, [0033]; EX1011, FIG. 7, [0004]; EX1003, ¶137.

8. 2[b.iii]: “transfer the requested content.”

Meir discloses this limitation. EX1003, ¶138. Meir discloses “sending the data” (*transfer the requested content*) “to the host (e.g., by the high-level controller).” EX1010, [0045].

E. Dependent Claim 5

1. 5[pre]: “The solid-state drive of claim 1 wherein: -”

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1. *See* Section VIII.C.1; EX1003, ¶139.

2. 5[a]: “the instructions further direct the programmable processor to:”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶140. As discussed in 1[e], Meir discloses software comprising *instructions stored on* memory 56 (*device memory*), *that when executed by* processor 52 (*programmable processor*), *direct the programmable processor to* execute the access commands according to a policy stored on the memory 56. *See* Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99, 140.

3. 5[a.i]: “compress the content based on the storage policy;”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶141-145.

Although Meir does not disclose that the software executed by processor 52 (*instructions*) *compress* the data (*requested content*) included with the commands *based on a storage policy*, this is rendered obvious by the system resulting from the combination of the teachings of Meir-Salessi.

Salessi's processing unit can *compress content for storage* because “[p]rocessing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” EX1011, [0006]; *see* Section VIII.C.7. Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” EX1011, [0023].

In the combination, Meir's policy stored in memory 56 includes Salessi's compression. Specifically, Meir teaches that the “one or more of the rules in the memory controller policy may be configurable” and “the host may issue to the memory controller a command that configures or modifies the memory controller, so as to improve performance.” EX1010, [0045]. A POSITA would therefore modify the memory controller policy of Meir to incorporate the compression algorithms and instructions of Salessi, wherein Meir's policy would define which data to apply Salessi's compression algorithms to. EX1011, [0006]; EX1003, ¶¶142-143.

As discussed, a POSITA would have been motivated to modify the software (*application*) and policy store in memory 56 (*policy*) of Meir to direct the processor

52 to compress the data to be stored according to algorithms and rules in the policy because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶¶144-145. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components, and Salessi expressly teaches that the use of its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, [0033]; EX1011, FIG. 7, [0004]; EX1003, ¶¶144-145.

4. 5[a.ii]: “transfer the compressed content to the device controller; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶146. Meir discloses *transferring the content to the device controller*. *See* Section VIII.C.8. Salessi discloses that the *content* is *compressed*. *See* Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.E.3; EX1003, ¶¶71-89, 141-145.

5. 5[b]: “the device controller is to:”

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*) “that executes the memory access commands issued by host 28.” *See* Section VIII.C.5; EX1003, ¶¶94-99, 147.

6. 5[b.i]: “receive the compressed content from the programmable processor; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶148. Meir discloses *receiving the content from the programmable processor*. See Section VIII.C.10. Salessi discloses that the *content* is *compressed*. See Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.E.3; EX1003, ¶¶71-89, 141-145.

7. 5[b.ii]: “store the compressed content on the solid-state memory.”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶149. Meir discloses *storing the content on the solid-state memory*. See Section VIII.C.11. Salessi discloses that the *content* is *compressed*. See Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.E.3; EX1003, 71-89, 141-145.

F. Dependent Claim 6

1. 6[pre]: “The solid-state drive of claim 5 wherein:”

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1. See Section VIII.C.1; EX1003, ¶150.

2. 6[a]: “The device controller is to:”

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*) “that executes the memory access commands issued by host 28.” See Section VIII.C.5; EX1003, ¶¶91, 151.

3. 6[a.i]: “read the compressed content from the solid-state memory; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶152. Meir discloses *reading the content from the solid-state memory*. See Section VIII.D.3. Salessi discloses that the *content is compressed*. See Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.E.3; EX1003, ¶¶71-89, 141-145.

4. 6[a.ii]: “transfer the compressed content to the programmable processor; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶153. Meir discloses *transferring the content to the programmable processor*. See Section VIII.D.4. Salessi discloses that the *content is compressed*. See Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.E.3; EX1003, ¶¶71-89, 141-145.

5. 6[b]: “the instructions further direct the programmable processor to:”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶154. As discussed in 1[e], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to execute the access commands according to a policy stored on the memory 56*. See Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99.

6. 6[b.i]: “receive the compressed content from the device controller;”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶155. Meir discloses *receiving the content from the device controller*. See Section VIII.D.6. Salessi discloses that the *content is compressed*. See Section VIII.E.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.E.3; EX1003, ¶¶71-89, 141-145.

7. 6[b.ii]: “decompress the content based on the storage policy; and”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶156-160.

Salessi’s processing unit can *decompress content for storage* because “[p]rocessing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” EX1011, [0042], [0023]; see Section VIII.C.7.

In the combination, the policy stored in memory 56 as taught by Meir includes decompression as taught by Salessi. Specifically, Meir teaches that the “one or more of the rules in the memory controller policy may be configurable” and “the host may issue to the memory controller a command that configures or modifies the memory controller, so as to improve performance.” EX1010, [0045]. A POSITA would therefore modify the memory controller policy of Meir to incorporate the decompression algorithms and instructions of Salessi, wherein Meir’s policy would

define which data to apply Salessi's decompression algorithms to. EX1011, [0006]; EX1003, ¶¶158-159.

As discussed, a POSITA would have been motivated to modify the software (*application*) and policy store in memory 56 (*policy*) of Meir to direct the processor 52 to decompress the data to be stored according to algorithms and rules in the policy because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶¶71-82, 159. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components, and Salessi expressly teaches using its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, [0033]; EX1011, FIG. 7, [0004]; EX1003, ¶¶83-89, 160.

8. 6[b.iii]: “transfer the decompressed content.”

Meir in view of Salessi renders obvious this limitation. EX1003, ¶161. Meir discloses *transferring the content*. *See* Section VIII.D.8. Salessi discloses that the *content* is *decompressed*. *See* Section VIII.F.7. A POSITA would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.F.7; EX1003, ¶¶71-89, 156-160.

G. Dependent Claim 7

1. 7[pre]

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1. *See* Section VIII.C.1; EX1003, ¶162.

2. 7[a]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶163. As discussed in 1[e], Meir discloses software comprising *instructions stored on* memory 56 (*device memory*), *that when executed by* processor 52 (*programmable processor*), *direct the programmable processor to* execute the access commands according to a policy stored on the memory 56. *See* Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99.

3. 7[a.i]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶164-167.

Salessi's processing unit can *encrypt content for storage* because "[p]rocessing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression." EX1011, [0042], [0023] ("Salessi further teaches that the processing unit "us[es] a processing algorithm[] to generate processed data."); *see* Section VIII.C.7.

In the combination, the policy stored in memory 56 as taught by Meir includes encryption as taught by Salessi. Specifically, Meir teaches that the "one or more of the rules in the memory controller policy may be configurable" and "the host may

issue to the memory controller a command that configures or modifies the memory controller, so as to improve performance.” EX1010, [0045]. A POSITA would therefore modify the memory controller policy of Meir to incorporate the encryption algorithms and instructions of Salessi, wherein Meir’s policy would define which data to apply Salessi’s encryption algorithms to. EX1011, [0006]; EX1003, ¶¶166-167.

As discussed, a POSITA would have been motivated to modify the software (*application*) and policy store in memory 56 (*policy*) of Meir to direct the processor 52 to compress the data to be stored according to algorithms and rules in the policy because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶¶71-82, 167. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components, and Salessi expressly teaches that the use of its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, [0033]; EX1011, FIG. 7, [0004]; EX1003, ¶¶83-89, 167.

4. 7[a.ii]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶168. Meir discloses *transferring the content to the device controller*. *See* Section VIII.C.8. Salessi discloses that the *content* is *encrypted*. *See* Section VIII.G.3. A POSITA

would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.

5. 7[b]

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*) “that executes the memory access commands issued by host 28.” *See* Section VIII.C.5; EX1003, ¶¶94-95, 169.

6. 7[b.i]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶170. Meir discloses *receiving the content from the programmable processor*. *See* Section VIII.C.10. Salessi discloses that the *content* is *encrypted*. *See* Section VIII.G.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.

7. 7[b.ii]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶171. Meir discloses *storing the content on the solid-state memory*. *See* Section VIII.C.11. Salessi discloses that the *content* is *encrypted*. *See* Section VIII.G.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.

H. Dependent Claim 8

1. 8[pre]

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1.

See Section VIII.C.1; EX1003, ¶172.

2. 8[a]

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*) “that executes the memory access commands issued by host 28.” See Section VIII.C.5; EX1003, ¶¶94-95, 173.

3. 8[a.i]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶174. Meir discloses *reading the content from the solid-state memory*. See Section VIII.D.3. Salessi discloses that the *content* is *encrypted*. See Section VIII.G.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.

4. 8[a.ii]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶175. Meir discloses *transferring the content to the programmable processor*. See Section VIII.D.4. Salessi discloses that the *content* is *encrypted*. See Section VIII.G.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.

5. 8[b]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶176. As discussed in 1[e], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor),*

direct the programmable processor to execute the access commands according to a policy stored on the memory 56. See Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99, 176.

6. 8[b.i]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶177. Meir discloses *transferring the content to the programmable processor. See Section VIII.D.8. Salessi discloses that the content is encrypted. See Section VIII.G.3. A POSITA would have been motivated to combine Meir with Salessi as discussed. See Sections VIII.A-B, VIII.G.3; EX1003, ¶¶71-89, 164-167.*

7. 8[b.ii]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶178-181.

Salessi's processing unit can *decrypt content for storage* because “[p]rocessing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” EX1011, [0042], [0023] (“Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.”); *see* Section VIII.C.7.

In the combination, the policy stored in memory 56 as taught by Meir includes decryption as taught by Salessi. Specifically, Meir teaches that the “one or more of the rules in the memory controller policy may be configurable” and “the host may issue to the memory controller a command that configures or modifies the memory

controller, so as to improve performance.” EX1010, [0045]. A POSITA would therefore modify the memory controller policy of Meir to incorporate the decryption algorithms and instructions of Salessi, wherein Meir’s policy would define which data to apply Salessi’s decryption algorithms to. EX1011, [0006]; EX1003, ¶¶180-181.

As discussed, a POSITA would have been motivated to modify the software (*application*) and policy store in memory 56 (*policy*) of Meir to direct the processor 52 to decrypt the data to be stored according to algorithms and rules in the policy because it would optimize the performance and improve the security of the system. *See* Section VIII.A; EX1010, [0036]; EX1011, [0050]; EX1003, ¶¶71-82, 181. Further, as discussed, a POSITA would have had a reasonable expectation of success since the SSDs of Meir and Salessi have similar structures and components, and Salessi expressly teaches that the use of its processing algorithms in such similar systems. *See* Section VIII.B; EX1010, FIG. 1, [0033]; EX1011, FIG. 7, [0004]; EX1003, ¶¶83-89, 181.

8. 8[b.iii]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶182. Meir discloses *transferring the content*. *See* Section VIII.D.8. Salessi discloses that the *content is decrypted*. *See* Section VIII.H.7. A POSITA would have been motivated to combine Meir with Salessi as discussed. *See* Sections VIII.A-B, VIII.H.7;

EX1003, ¶¶71-89, 178-181.

I. Dependent Claim 10

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶183-186.

Meir discloses *data*. See Section VIII.C.7.

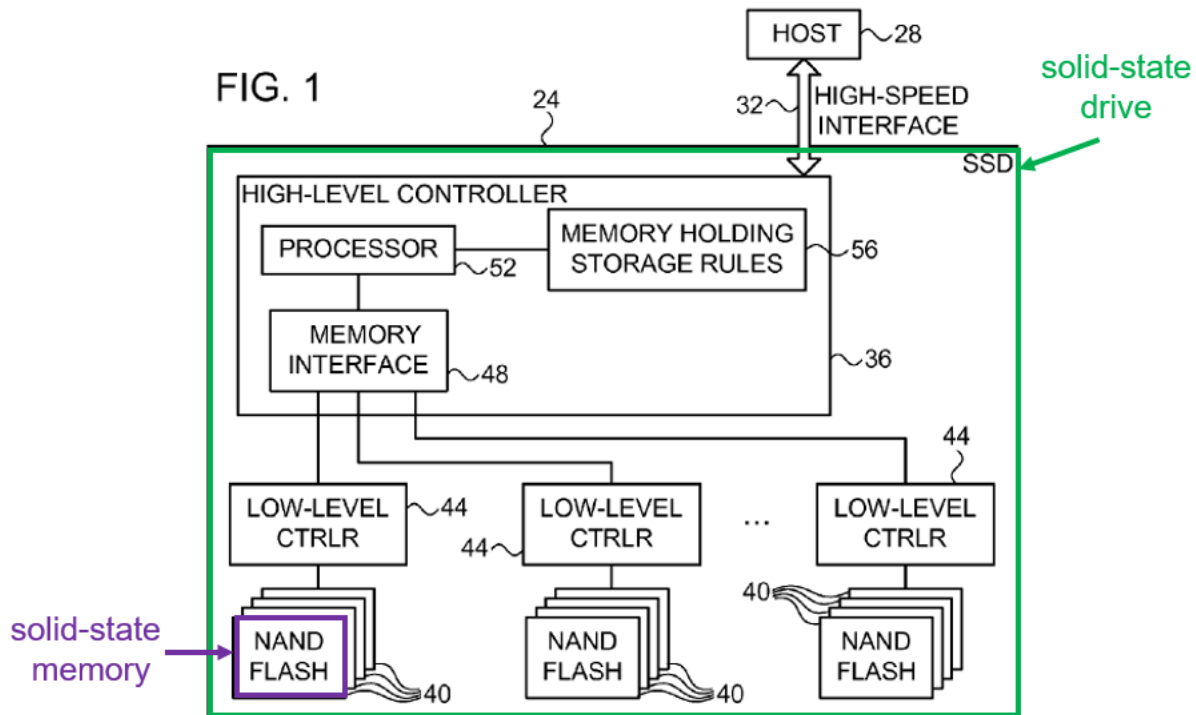
Salessi explains that its data for storage (*content*) is *video data, image data, on-line temporal video data, financial data, or online transaction data*. EX1011, [0006], [0023]. For example, Salessi’s “data comes from everywhere: posts to social media sites, digital pictures and video, purchase transaction records, bank transactions, sensors used to gather data and intelligence, like climate information, cell phone GPS signal, and many others.” EX1011, [0002].

A POSITA would have understood that the data being stored or retrieved is *cached data* in embodiments where the data is being stored in cache devices. EX1011, [0058]; ¶¶185-186. A POSITA would have been motivated to combine Meir with Salessi as discussed above. See Sections VIII.A-B; EX1003, ¶¶71-89, 185-186.

J. Independent Claim 11

1. 11[pre]: “A method of operating a solid-state drive, the method comprising:”

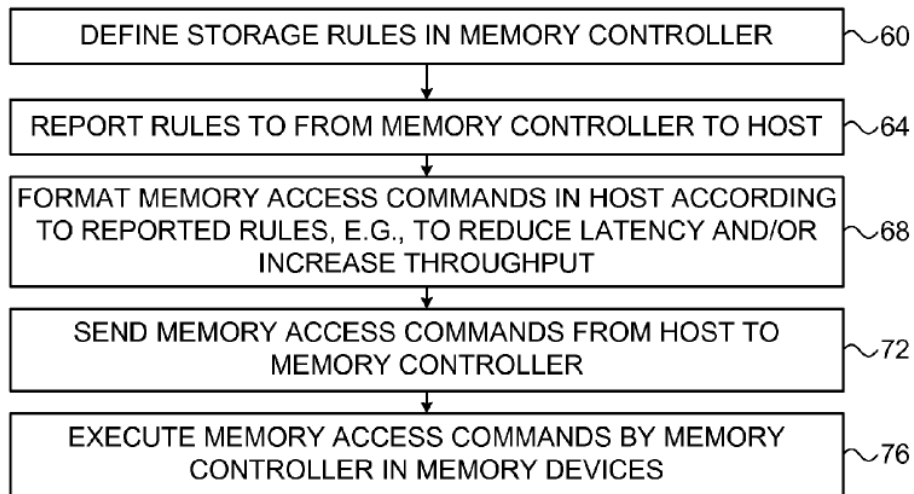
To the extent limiting, Meir discloses the preamble. EX1003, ¶¶187-188. As discussed for 1[pre], Meir discloses a solid-state drive (“SSD”) 24 (*solid-state drive*). See Section VIII.C.1; EX1010, FIG. 1, [0029]; EX1003, ¶90.



EX1010, Fig. 1 (annotated)

Meir discloses that “SSD 24 comprises a high-level controller 36, which stores data in one or more non-volatile memory units 40.” *Id.*, [0030]. Specifically, Meir teaches *a method of operating SSD 24 (solid-state drive)* because Meir discloses “[a] data storage method” using SSD 24 that “accepts memory access commands from a host for execution in one or more memory units.” *Id.*, Abstract. “FIG. 2 is a flow chart that schematically illustrates a method for data storage” in accordance with Meir’s disclosures. *Id.*, [0019]; EX1003, ¶¶187-188.

FIG. 2



EX1010, Fig. 2

2. **11[a]: “in response to a storage request comprising content, processing the content based on a storage policy; ”**

Meir in view of Salessi discloses this limitation as discussed for claim 1[e.i].

EX1003, ¶¶100-109, 189-194; *see* Section VIII.C.7.

In response to a write command including the data to be stored (*storage request comprising content*), Meir discloses that the memory controller will *process* the data (*content*) *based on a storage policy* because “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021], [0033] (“memory 56 holds a definition of a policy that the memory controller applies in storing data in memory units 40”); EX1003, ¶191. “The policy comprises a set of rules, for example parallelization rules, pipeline rules, prioritization rules, redundancy rules, power

consumption rules and/or memory unit characteristics rules.” *Id.*, [0033].

When applying the redundancy rules from the policy, the software (*application*) can direct the processor 52 to *process* the data (*content for storage*) by making a copy of the data to store a redundant copy in other memory units, striping the data dividing it into blocks, or calculating parity information on the data. *Id.*, [0025] (“[t]he memory controller policy involves storing the data while applying a redundant storage scheme such as Redundant Array of Inexpensive Disks (RAID) among the memory units”); EX1003, ¶191.

To the extent that Meir does not disclose that the software executed by processor 52 (*instructions*) *process* the data (*content for storage*) included with the commands *based on a storage policy*, this is rendered obvious by the system resulting from the combination of the teachings of Meir and Salessi.

Salessi’s processing unit can *process content for storage* because it is “capable of processing the data...being stored or retrieved from the flash memory.” EX1011, [0006]. “Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” *Id.* Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” *Id.*, [0023].

A POSITA would combine the teachings of Meir with that of Salessi as discussed above. *See* Sections VIII.A-B, VIII.C.7; EX1003, ¶¶71-89, 100-109.

3. 11[b]: “storing the processed content on the solid-state memory;”

Meir discloses this limitation as discussed for claim 1[f.ii]. EX1003, ¶¶118-120, 195-198; *see also* Section VIII.C.11.

Meir explains that the “memory controller...executes the commands in the memory units.” EX1010, [0020]. In the embodiment where the memory controller comprises high-level controller 36 and low-level controller 44, the high-level controller 36 communicates with memory units 40 (*solid-state memory*) via low-level controller 44 (*device controller*). *Id.*, [0033] (“High-level controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example via low-level controllers 44), and a processor 52 that carries out the various functions of the high-level controller.”). Accordingly, when the memory controller executes a write command, the low-level controller 44 writes the data (*storing the content*) to the memory unit 40 (*on the solid-state memory*) according to the logical-to-physical address mapping defined by the parallelism rule that is part of the policy. *Id.*, [0040] (“According to the policy, the memory controller translates the logical addresses to respective physical storage locations in memory units 40, and executes the commands in the memory units according to the logical-to-physical address mapping.”).

Meir explains that advantages of partitioning the execution of a command between the high-level controller 36 (as executed by processor 52) and low-level

controller 44. EX1003, ¶197. Specifically, the processor 52 in high-level controller 36 can distribute the execution of various commands among various memory units 40 (*solid-state memory*) via the low-level controller 44. EX1010, [0037], [0042]. For example, while one low-level controller 44 is storing data, processor 52 of the high-level controller 36 can execute another command, sending data to another low-level controller 44 to store. EX1003, ¶198. The SSD including a processor 52 executing a policy defining “LBA assignment or ordering of this sort may avoid temporary congestion in a certain memory unit or low-level controller, and thus reduce latency.” EX1010, [0042].

4. 11[c]: “in response to a read request, reading requested content from the solid-state memory;”

Meir discloses this limitation as discussed in 2[a.i]. EX1003, ¶¶123-125, 199-202; *see* Section VIII.D.3.

In the embodiment where the memory controller comprises high-level controller 36 and low-level controller 44 (*device controller*), the high-level controller 36 communicates with memory units 40 (*solid-state memory*) via low-level controller 44 (*device controller*). *Id.*, [0033]. Accordingly, when the memory controller executes a read command, the high-level controller 36 “translates the logical addresses to respective physical storage locations in memory units 40, and executes the commands in the memory units according to the logical-to-physical address mapping.” *Id.* [0040]. In one embodiment, “a certain subset of bits in the

logical address may indicate the...low-level controller in which this address is assigned.” After translation and *in response to the* read command (*read request*), the high-level controller 36 will direct that assigned low-level controller 44 (*device controller*), to execute the access command, here the read command (*read requested content from the solid-state memory*). *Id.*, [0044]; EX1003, ¶200-201.

As Meir further explains, in certain embodiments, the SSD 24 may use error correction code (“ECC”) to encode the data (*content*) to prevent errors. *Id.*, [0045]. When a read command is received, the low-level controller 44 will use the processing circuitry in the memory units 40 to fetch the ECC-encoded data (*read requested content from the solid-state memory*). *Id.* The low-level controller 44 will then decode the data. *Id.*; EX1003, ¶202.

5. 11[d]: “processing the requested content for transfer based on the storage policy; and”

Meir in view of Salessi discloses this limitation as discussed for claim 2[b.ii]. EX1003, ¶¶130-137, 203-208; *see also* Section VIII.D.7.

Meir teaches that “SSD 24 comprises a high-level controller 36, which stores data in one or more non-volatile memory units 40.” EX1010, [0030]. This data is *requested content*. EX1003, ¶204.

Meir discloses that the memory controller *processing* memory access commands *based on a storage policy* because “[t]he memory controller typically executes the memory access commands received from the host in accordance with a

certain policy.” EX1010, [0021], [0033] (“memory 56 holds a definition of a policy that the memory controller applies in storing data in memory units 40”); EX1003, ¶205. “The policy comprises a set of rules, for example one or more parallelization rules, pipeline rules and/or prioritization rules.” *Id.*, [0033]; *see also* [0025] (“[t]he memory controller policy involves storing the data while applying a redundant storage scheme such as Redundant Array of Inexpensive Disks (RAID) among the memory units”).

To the extent that Meir does not disclose that the software executed by processor 52 (*instructions*) *process* the data (*requested content*) included with the commands *based on a storage policy*, this is rendered obvious by the system resulting from the combination of the teachings of Meir and Salessi.

Salessi’s processing unit can *process requested content* because it is “capable of processing the data...being stored or retrieved from the flash memory.” EX1011, [0006]. “Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” *Id.* Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” *Id.*, [0023].

A POSITA would combine the teachings of Meir with Salessi as discussed above. *See* Section VIII.D.7; EX1003, ¶¶130-137, 206-208. The software (*application*) can direct the processor 52 to *process* the received data (*requested*

content for transfer) based on the redundant storage scheme. *Id.*, [0025]. For example, when the data has been striped, the processor 52 splits the read request and pulls different parts of the requested data from multiple drives. *Id.*, [0038]; EX1003, ¶132. Processor 52 (*programmable processor*) will reassemble the different parts to transfer to the host. EX1003, ¶132. Similarly, when a drive failure has been identified, processor 52 will use the parity data to rebuild the requested data. *Id.*

6. 11[e]: “transferring the requested content.”

Meir discloses this limitation as discussed for 2[b.iii]. EX1003, ¶¶138, 209; *see* Section VIII.D.8. Meir discloses “sending the data” (*transfer the requested content*) “to the host (e.g., by the high-level controller).” EX1010, [0045].

K. Dependent Claim 14

1. 14[pre]

As discussed, Meir discloses a ***method by*** a SSD 24 (*solid-state drive*) as recited in claim 11. *See* Section VIII.J.1; EX1003, ¶210.

2. 14[a]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶211-213. Meir-Salessi teaches *processing the content based on the storage policy*. *See* Section VIII.J.5. Meir-Salessi teaches *compressing the content based on the storage policy*. *See* Section VIII.E.3.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 141-145, 203-208.

3. 14[b]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶214-216. Meir-Salessi teaches *storing the processed content on the solid-state memory*. See Section VIII.J.3. Meir-Salessi further teaches *storing the compressed content on the solid-state memory*. See Section VIII.E.7.

A POSITA would combine the teachings of Meir with that of Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 171, 195-198.

4. 14[c]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶217-219. Meir-Salessi teaches *reading requested content from the solid-state memory*. See Section VIII.J.4. Meir-Salessi teaches *reading the compressed content from the solid-state memory*. See Section VIII.F.3.

A POSITA would combine the teachings of Meir with that of Salessi as discussed above. *Id.*; EX1003, ¶¶71-89, 152, 199-202.

5. 14[d]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶220-222. As discussed, Meir-Salessi teaches *processing the requested content for transfer based on the storage policy*. See Section VIII.J.5. Meir-Salessi further teaches *decompressing the compressed content based on the storage policy*. See Section VIII.F.7.

A POSITA would combine the teachings of Meir with that of Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 156-160, 203-208.

6. 14[e]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶223-225. As discussed, Meir-Salessi teaches *transferring the requested content*. See Section VIII.J.6. Meir-Salessi teaches *transferring the decompressed content*. See Section VIII.F.8.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 161, 209.

L. Dependent Claim 15

1. 15[pre]

As discussed, Meir discloses a method by a SSD 24 (solid-state drive) as recited in claim 11. See Section VIII.J.1; EX1003, ¶226.

2. 15[a]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶227-229. Meir-Salessi teaches *processing the content based on the storage policy*. See Section VIII.J.2. Meir-Salessi further teaches *encrypting the content based on the storage policy*. See Section VIII.G.3.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 164-167, 189-194.

3. 15[b]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶230-232. Meir-Salessi teaches *storing the processed content on the solid-state memory*. See Section VIII.J.3. Meir-Salessi further teaches *storing the encrypted content on the solid-state memory*. See Section VIII.G.7.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 171, 195-198.

4. 15[c]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶233-235. Meir-Salessi teaches *reading requested content from the solid-state memory*. See Section VIII.J.4. Meir-Salessi teaches *reading the encrypted content from the solid-state memory*. See Section VIII.H.3.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 174, 199-202.

5. 15[d]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶236-238. Meir-Salessi teaches *processing the requested content for transfer based on the storage policy*. See Section VIII.J.5. Meir-Salessi teaches *decrypting the encrypted content based on the storage policy*. See Section VIII.H.7.

A POSITA would combine the teachings of Meir with Salessi as discussed.

Id.; EX1003, ¶¶71-89, 178-181, 203-208.

6. 15[e]

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶239-241.

Meir-Salessi teaches *transferring the requested content*. See Section VIII.J.6.

Meir-Salessi further teaches *transferring the decrypted content*. See Section VIII.H.8.

A POSITA would combine the teachings of Meir with Salessi as discussed. *Id.*; EX1003, ¶¶71-89, 182, 209.

M. Dependent Claim 16

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶242-246. Meir discloses that the memory controller will process commands based on a storage policy because “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021], [0033]; EX1003, ¶243. “The policy comprises a set of rules, for example one or more parallelization rules, pipeline rules and/or prioritization rules.” *Id.*, [0033]; *see also* [0025] (“[t]he memory controller policy involves storing the data while applying a redundant storage scheme such as Redundant Array of Inexpensive Disks (RAID) among the memory units”).

To the extent that Meir does not disclose wherein the storage policy indicates how to process content for storage on the solid-state drive and how to process the

requested content for transfer, this is rendered obvious by the system resulting from the combination of the teachings of Meir and Salessi.

Salessi teaches how to process content for storage on the solid-state drive and how to process the requested content for transfer because its processing unit is “capable of processing the data...being stored or retrieved from the flash memory.” EX1011, [0006]. “Processing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and decompression.” *Id.* Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” *Id.*, [0023].

A POSITA would combine the teachings of Meir with Salessi as discussed. *See* Section VIII.C.7; EX1003, ¶¶71-89, 100-109, 244-246.

N. Dependent Claim 17

Meir in view of Salessi renders obvious this limitation. EX1003, ¶¶247-251. Meir discloses “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021], [0033]; EX1003, ¶248. “The policy comprises a set of rules.” *Id.*, [0033]; *see also* [0025].

To the extent that Meir does not disclose this limitation, it is rendered obvious by the Meir-Salessi SSD. Salessi teaches “[p]rocessing capabilities incorporated into the SSD may include encryption and decryptions [and] compression and

decompression.” EX1011, [0006]. Salessi further teaches that the processing unit “us[es] a processing algorithm[] to generate processed data.” *Id.*, [0023].

A POSITA would have combined the teachings of Meir and Salessi as discussed. *See* Section VIII.C.7; EX1003, ¶¶71-89, 100-109, 249-251.

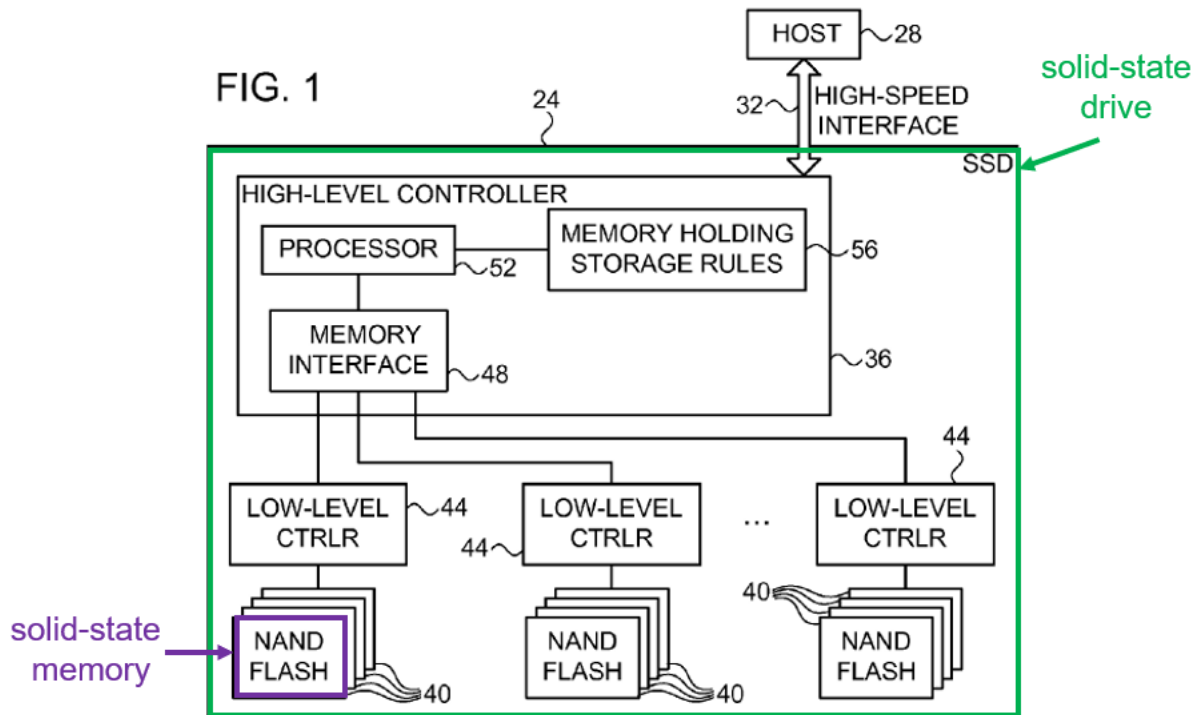
O. Dependent Claim 18

Meir discloses this limitation. EX1003, ¶¶252-253. Meir discloses that the storage policy is stored in memory 56. EX1010, FIG. 1, [0033]. Meir further discloses that “[t]he memory controller typically executes the memory access commands received from the host in accordance with a certain policy.” EX1010, [0021]. The memory controller would need to retrieve the storage policy from memory to execute memory access commands according to that policy. EX1003, ¶253.

P. Independent Claim 20

1. 20[pre]: “A solid-state drive comprising:”

To the extent limiting, Meir discloses the preamble. EX1003, ¶254. As discussed for 1[pre], Meir discloses a solid-state drive (“SSD”) 24 (*solid-state drive*). *See* Section VIII.C.1; EX1010, FIG. 1, [0029]; EX1003, ¶90.

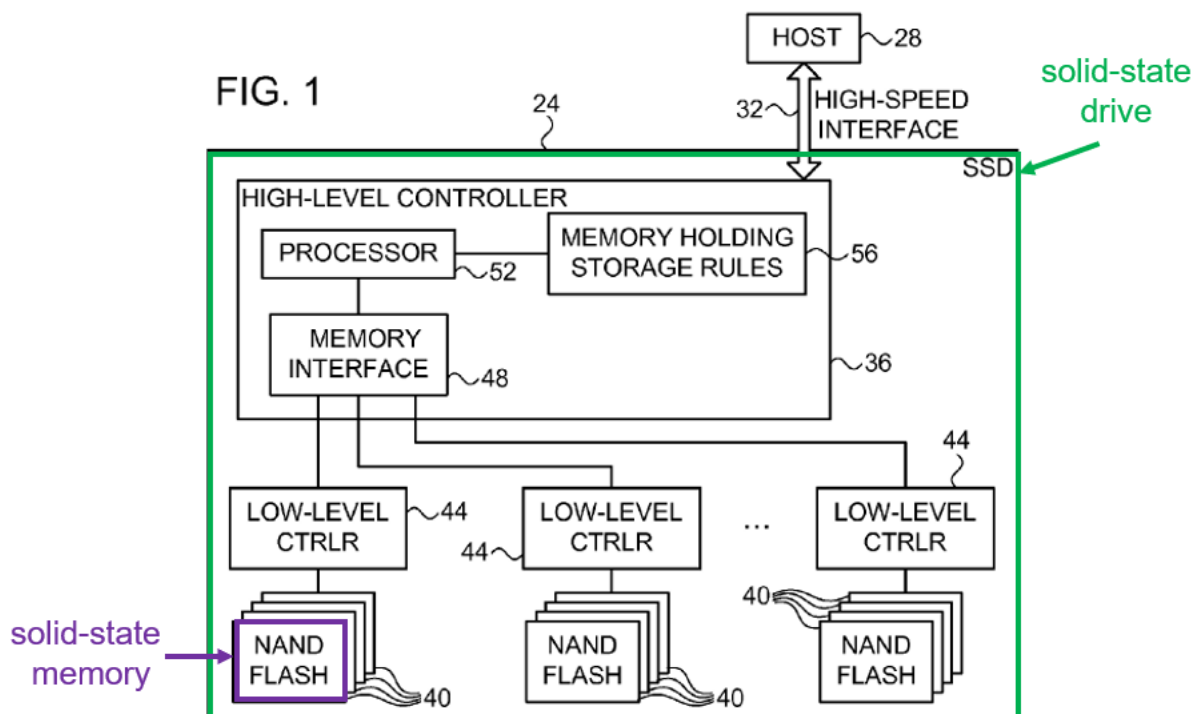


EX1010, Fig. 1 (annotated)

2. 20[a]: “a solid-state memory;”

As discussed for 1[a], Meir discloses memory unit 40 (*solid-state memory*).

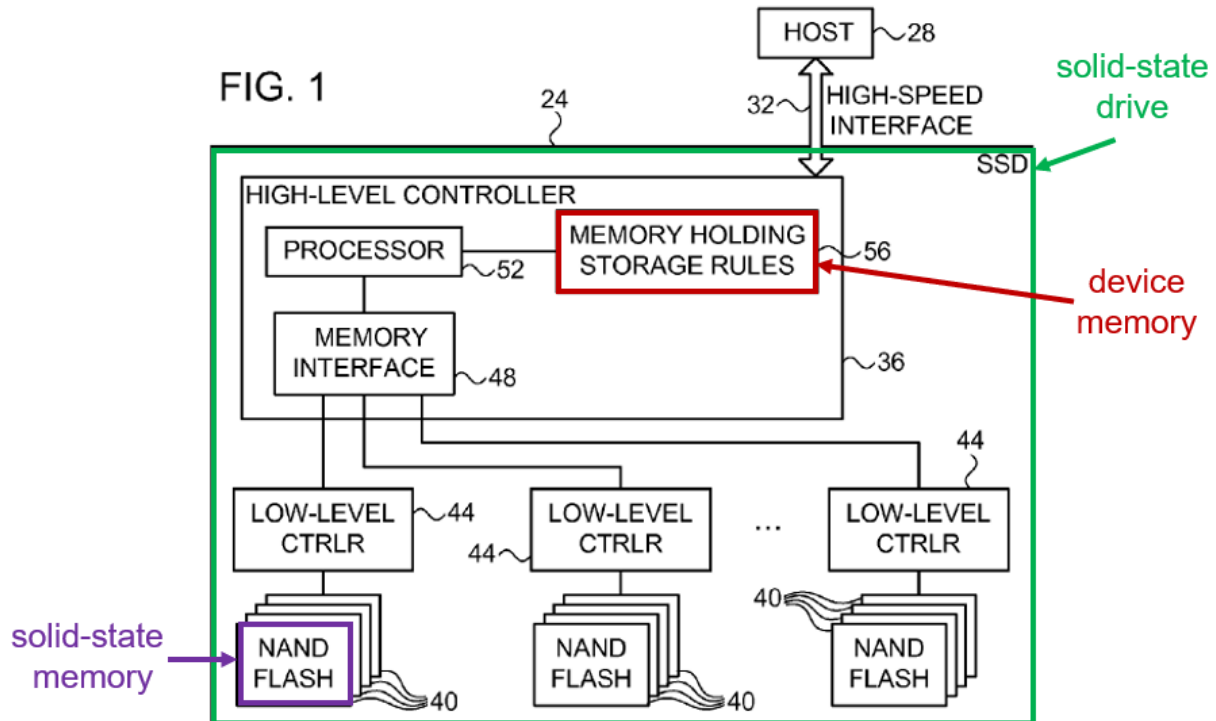
See Section VIII.C.2; EX1010, FIG. 1, [0030]; EX1003, ¶¶91, 255.



EX1010, Fig. 1 (annotated)

3. 20[b]: “a device memory;”

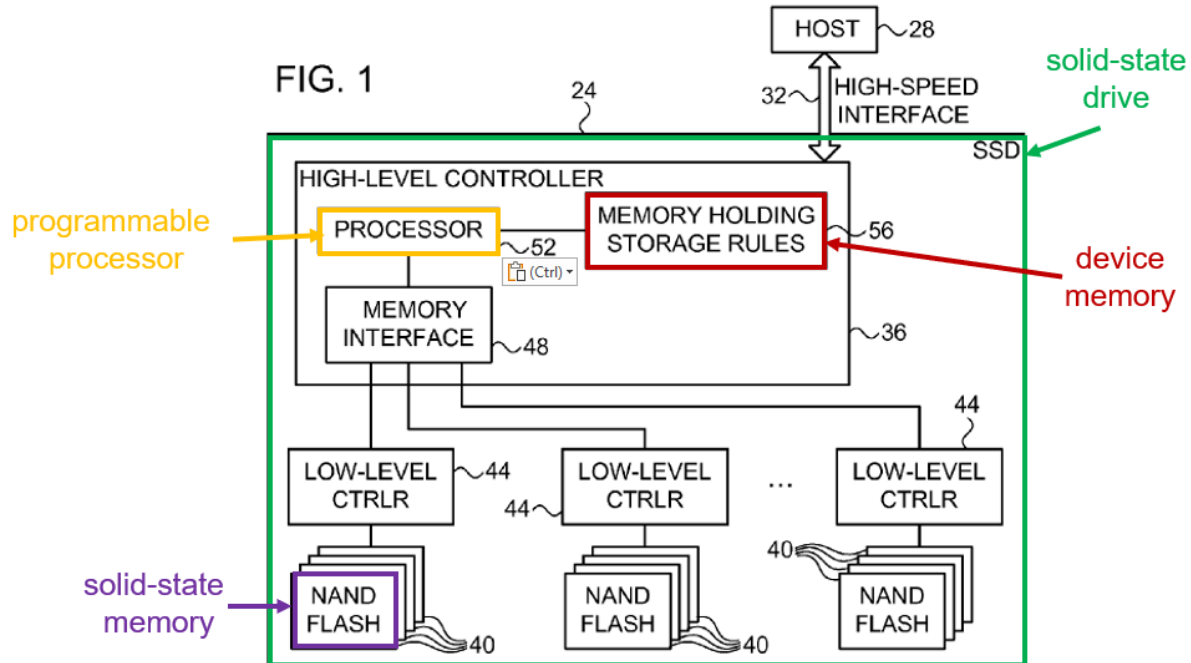
As discussed for 1[b], Meir discloses memory 56 (*device memory*). See Section VIII.C.3; EX1010, FIG. 1, [0033]; EX1003, ¶¶92, 256.



EX1010, Fig. 1 (annotated)

4. 20[c]: “a programmable processor operatively coupled to the device memory;”

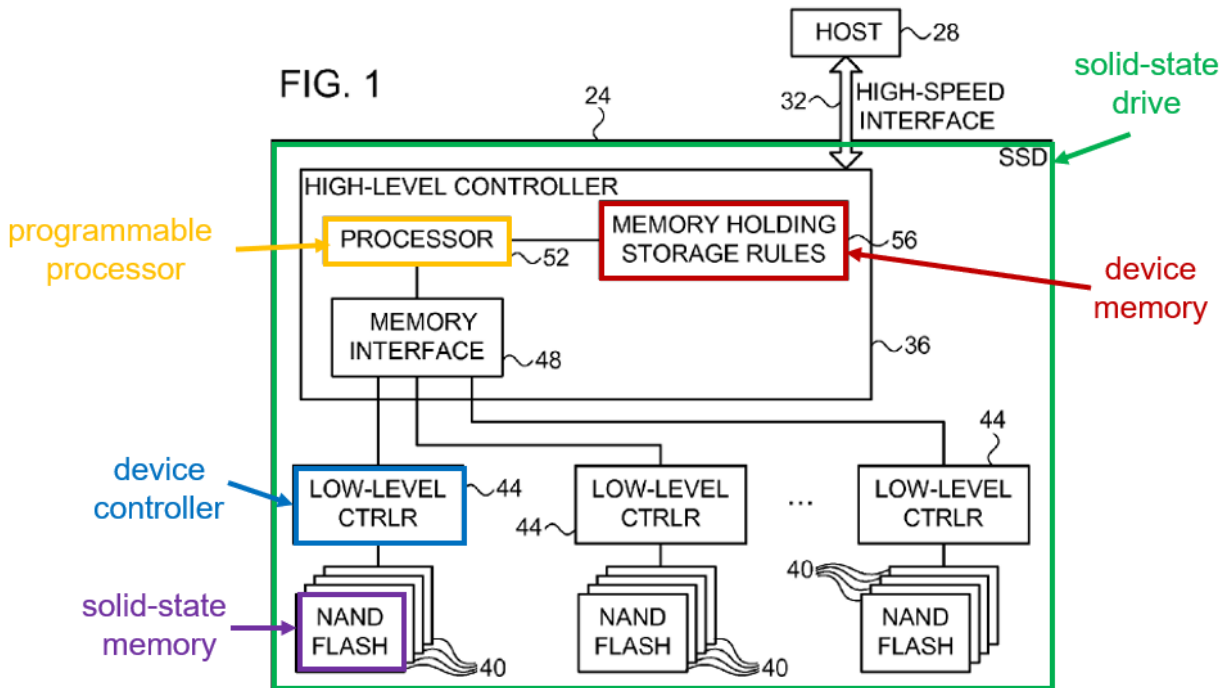
As discussed for 1[c], Meir discloses processor 52 (*programmable processor*) operatively coupled to memory 56 (*device memory*). See Section VIII.C.4; EX1010, FIG. 1, [0033]; EX1003, ¶¶93, 257.



EX1010, Fig. 1 (annotated)

5. 20[d]: “a device controller operatively coupled with the solid-state memory and the programmable processor; and”

As discussed for 1[d], Meir discloses low-level controller 44 (*device controller*) operatively coupled with the memory unit 40 (*solid-state memory*) and processor 52 (*programmable processor*). See Section VIII.C.5; EX1010, FIG. 1, [0031], [0033]; EX1003, ¶¶94-95, 258.



EX1010, Fig. 1 (annotated)

6. 20[e]: “instructions stored on the device memory that, when executed by the programmable processor, direct the programmable processor to:”

As discussed for 1[e], Meir discloses software comprising *instructions stored on memory 56 (device memory)*, that when executed by processor 52 (*programmable processor*), direct the programmable processor to execute the access commands according to a policy stored on the memory 56. See Section VIII.C.6; EX1010, FIG. 1, [0034]; EX1003, ¶¶96-99, 259-260.

7. 20[e.i]: “process content for storage on the solid-state memory based on a storage policy;”

As discussed for 1[e.i], Meir discloses software comprising *instructions stored on memory 56 (device memory)*, that when executed by processor 52

(programmable processor), direct the programmable processor to execute a write command to store data (process content for storage) on the memory unit 40 (solid-state memory) according to a policy stored on the memory 56 (based on a storage policy), including applying redundancy rules. See Section VIII.C.7; EX1010, FIG. 1, [0020]-[0021], [0025], [0030], [0033]; EX1011, [0006], [0023]; EX1003, 100-109, 261-266.

8. 20[e.ii]: “transfer the content to the device controller to store on the solid-state memory;”

As discussed for 1[e.ii], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to execute a write command by sending the data (transfer the content) to the assigned low-level controller 44 (device controller) to store on the appropriate memory unit 40 (solid-state memory).* See Sections VIII.C.8, VIII.C.11; EX1010, FIG. 1, [0005], [0007], [0009], [0023], [0031], [0037], [0041]; EX1003, ¶¶110-115, 118-120, 267-272.

9. 20[e.iii]: “in response a content request, receive requested content read from the solid-state memory by the device controller;”

As discussed for 11[d], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to execute receive data (requested content) from the memory unit 40 (solid-state memory) according to*

a policy stored on the memory 56 (*based on a storage policy*), including applying parallelism rules. *See* Section VIII.J.5; EX1010, FIG. 1, [0020]-[0021], [0025], [0030], [0033]; EX1003, ¶¶203-208, 273-274.

10. 20[e.iv]: “process the requested content for transfer based on the storage policy; and”

As discussed for 11[d], Meir in view of Salessi renders obvious software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to execute a read command to fetch stored data (process the requested comment for transfer) according to a policy stored on the memory 56 (based on a storage policy), including applying redundancy rules, compression, and encryption. See* Section VIII.J.5; EX1010, FIG. 1, [0020]-[0021], [0025], [0030], [0033], [0045]; EX1011, [0006], [0023]; EX1003, ¶¶203-208, 275-278.

11. 20[e.v]: “transfer the requested content.”

As discussed for 11[e], Meir discloses software comprising *instructions stored on memory 56 (device memory), that when executed by processor 52 (programmable processor), direct the programmable processor to execute send data to the host (transfer the requested content). See* Section VIII.J.6; EX1010, FIG. 1, [0020]-[0021], [0025], [0030], [0033], [0045]; EX1003, ¶¶209, 279.

**IX. GROUND 2: MEIR IN VIEW OF SALESSI AND CHOU RENDERS
OBVIOUS CLAIM 9**

A. Motivation to Combine

A POSITA would have been motivated to modify Meir's storage drive, including the memory unit 40 (*solid-state memory*), the processor 52 (*programmable processor*), the memory 56 (*device memory*), the low-level controller 44 (*device controller*), the high-speed interface 32 (*communication interface*), and the memory interface 48 (*data link*), to be mounted on Chou's printed circuit board that is mounted by a case. EX1003, ¶¶280-283.

First, such modification would optimize the performance of the system by reducing over-the-air interference from nearby signals, thereby reducing the likelihood of errors and retransmission. EX1010, [0042] ("maximiz[ing] a throughput of the execution [and] minimiz[ing] a latency of the execution"); EX1003, ¶281. **Second**, such modification would improve the resiliency and extend the life-time of the components. EX1003, ¶282. A case provides protection for the internal components from external environmental conditions. *Id.*

Nothing in Salessi teaches away from this combination because Salessi also taught better performance in data storage in SSDs like Meir and Chou.

B. Reasonable Expectation of Success

A POSITA would have been able to modify Meir's storage drive to incorporate Chou's printed circuit board and case with a reasonable expectation of success. EX1003, ¶284. The storage drives of Meir and Chou are both SSDs that include controllers and solid-state memory. EX1010, FIG.1; EX1012, [0004]; EX1003, ¶284. Using printed circuit boards and cases as taught in Chou was a well-known and common design option that was routinely used to provide better stability and protection for hardware. EX1003, ¶284.

Nothing in Salessi prevents this combination because Salessi's SSD has similar structure and components to Meir-Chou's SSDs. *Id.*

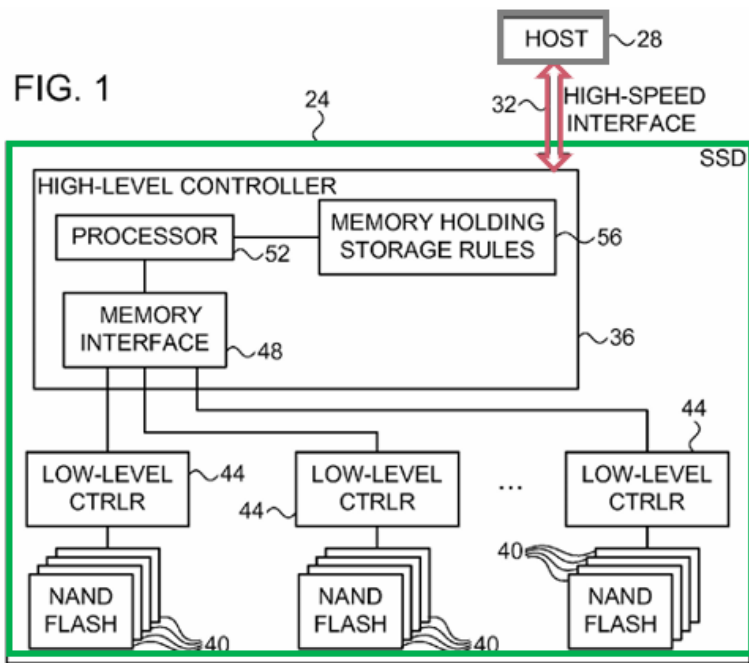
C. Dependent Claim 9

1. 9[pre]

As discussed, Meir discloses SSD 24 (*solid-state drive*) as recited in claim 1. *See* Section VIII.C.1; EX1003, ¶¶90, 285.

2. 9[a]

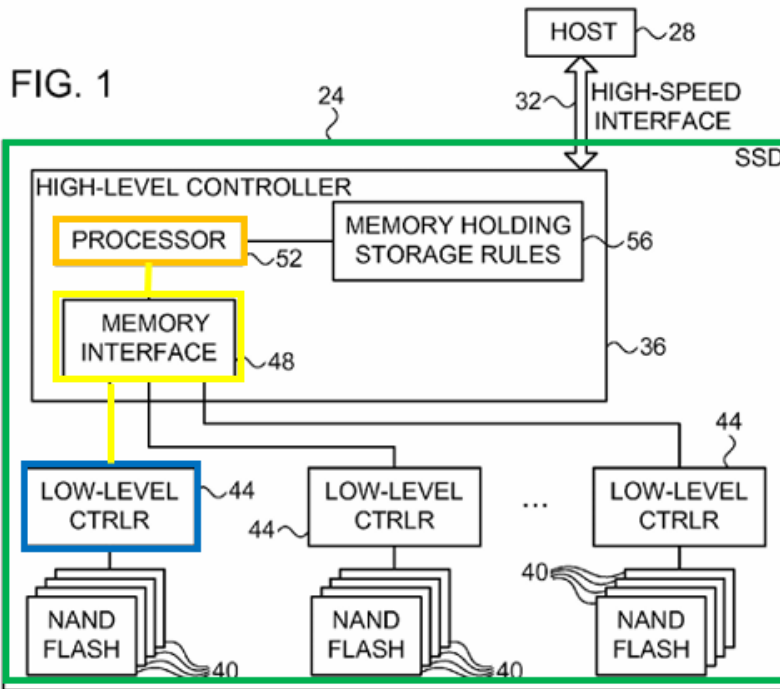
Meir discloses this limitation. EX1003, ¶¶286-287. Meir discloses a *high-speed interface 32 (communication interface)* that connects (*communicatively couples*) SSD 24 (*solid-state device*) to host 28 (*host device*) as shown by the connection between these two components in Figure 1; EX1010, FIG. 1; *see also id.*, [0029], FIG. 1.



EX1010, Fig. 1 (annotated)

3. 9[b]

Meir discloses this limitation. EX1003, ¶¶288-289. Meir discloses a memory interface 48 (*data link*) to *communicatively couple* processor 52 (*the programmable processor*) and low-level controller 44 (*the device controller*) as shown by the connection between these components in Figure 1; EX1010, FIG. 1; *see also id.*, [0033].



EX1010, Fig. 1 (annotated)

4. 9[c]

Meir in view of Salesi and Chou renders obvious this limitation. EX1003, ¶¶290-294. To the extent that Meir does not explicitly disclose a *printed circuit board to mount* the recited components, this would have been obvious in view of Chou. EX1003, ¶¶292-294.

Chou discloses a printed circuit board assembly (PCBA) (*printed circuit board*). EX1012, [0006]. Chou also discloses that “[o]ne or more memory modules and other electronic components 122 are on the PCBA 112.” *Id.*, [0033]. A POSITA would then understand that the *solid-state memories, device memories, device*

controllers, and *programmable processors* are all either “memory modules” or “other electronic components” that would be on the PCBA. EX1003, ¶293.

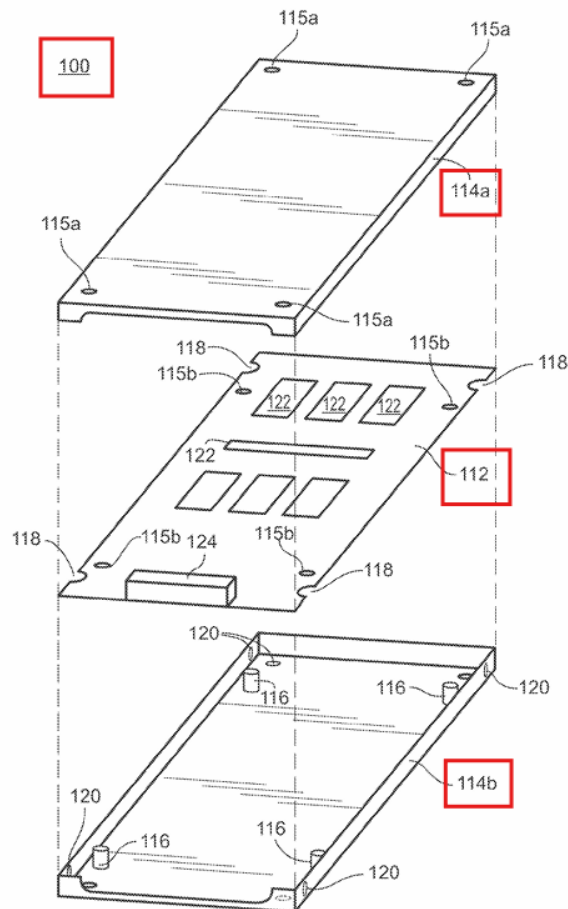
A POSITA would have combined the Meir-Salessi storage drive with Chou’s PCBA because this would have furthered Meir’s goal of “maximiz[ing] a throughput of the execution [and] minimiz[ing] a latency of the execution” because having the different components on a single PCB would reduce the amount of time and distance that data has to travel from one component to another. EX1010, [0042]; *see* Sections IX.A-B. To have a data storage system (like Meir-Salessi) that includes a solid-state memory, a device memory, a device controller, and a programmable processor, a POSITA would have understood that a physical means was required to connect these electronic components, and a printed circuit board would have been the most conventional and routine means used for mounting the components. EX1003, ¶294.

5. 9[d]

Meir in view of Salessi and Chou renders obvious this limitation. EX1003, ¶¶295-298.

Meir discloses a *communication interface* (*see* Section IX.C.2). To the extent that neither Meir nor Salessi discloses *a case to mount the printed circuit board and the communication interface*, this would have been obvious in view of Chou. EX1003, ¶296.

Like Meir, Chou discloses a solid-state drive (SSD) that includes a printed circuit board assembly (“PCBA”) (*printed circuit board*) and a “housing” (*case*). EX1007, [0025]. The “housing” (*case*) supports the “PCBA” (*printed circuit board*) because “[t]o assemble the SSD 100, the PCBA 112 is placed inside the upper and lower covers 114a and 114b.” *Id.*, [0034], FIG. 2.



EX1012, Fig. 2

A POSITA would have combined the Meir/Salessi storage drive with Chou’s housing because this would have furthered Meir’s goal of “maximiz[ing] a

throughput of the execution [and] minimiz[ing] a latency of the execution” because having a case mounting the PCB would reduce over-the-air interference from nearby signals, thereby reducing the likelihood of errors and retransmission. EX1010, [0042]; *see* Sections IX.A-B.

**X. GROUND 3: MEIR IN VIEW OF SALESSI AND LOVINGER
RENDERS OBVIOUS CLAIM 19**

A. Motivation to Combine

A POSITA would have been motivated to modify Meir-Salessi’s SSD to receive a write command that includes the policy as taught by Lovinger. EX1003, ¶¶299-303.

First, such a modification would enhance the goals of Meir. One of Meir’s goals is to make policies configurable. EX1010, [0011]. Lovinger teaches transmitting a storage request that include information associated with the storage policy. EX1015, [0040]. A POSITA would understand that configuring Meir and Salessi’s storage system to include a storage policy in the storage request would increase the configurability of the policy because the host can configure the policy before sending it as part of the request. EX1003, ¶300.

Second, Lovinger’s approach is a natural extension of Meir’s configurable policy. Lovinger teaches that including the policy with the storage request enables a client “to store content according to the retention policy without reprogramming device firmware.” *Id.*, [0041]. A POSITA improving on the system of Meir would

have looked at ways to configure the policy without reprogramming device firmware. EX1003, ¶301.

Third, such a modification would provide more flexibility, particularly as to the compression or encryption algorithm used. Indeed, Salessi teaches that “[t]he algorithm used by the application layer may be communicated to the DSP by the host CPU.” EX1011, [0055]. A POSITA would have looked at known ways to communicate the Meir-Salessi compression and encryption policy from the host to the storage system as taught in Lovinger. EX1003, ¶302.

Fourth, including the policy with the storage request as taught by Lovinger was a well-known design option. By 2016, this approach was a common option among multiple specifications. EX1003, ¶303. For example, the Open-Channel SSD Interface Rev. 1.2 specifies a write command that includes the content to be stored as well as a per-write policy word that instructs the device on the recording method that should be performed. *Id.*; EX1036.

B. Reasonable Expectation of Success

A POSITA would have been able to modify Meir-Salessi’s system to incorporate the policy in a storage request as taught by Lovinger with a reasonable expectation of success. EX1003, ¶¶304-306.

The storage devices of Meir, Salessi, and Lovinger have similar structures and components. EX1010, FIG.1; EX1011, FIGS. 2, 4, 7, EX1015, [0029], [0037]-

[0040], [0069], FIGS. 1, 6; EX1003, ¶305. Meir, Salessi, and Lovinger’s storage devices each include memory controller or processing unit that interfaces with the solid-state memory. EX1010, [0031] (“SSD 24 comprises multiple low-level controllers 44, each low-level controller assigned to a group of one or more memory units 40.”), [0033] (“[h]igh-level controller 36 comprises a memory interface 48 for communicating with memory units 40 (in the present example via low-level controllers 44)”; EX1011 [0047] (“[t]he SSD processing unit 240 includes...a hardware engine 420 [and] there may be a separate hardware engine 420 for each flash channel”), [0055] (“the microcontroller 410 may be connected to a plurality of hardware engines 420.”); EX1015, [0069] (“the computing device 600 may include at least one processing unit 602 [and] data storage devices”).

Given the similar structures, the modification would simply be a change of the storage request flow. Indeed, Meir includes “memory access commands (e.g., read and write commands)” and storage policies including “redundancy rules.” EX1010, [0030], [0033], [0038]. Further, Lovinger teaches sending the policy as part of the memory access command. EX1015, [0040]. Updating the storage request flow would have been well within the skill set of a POSITA at the relevant time. EX1003, ¶¶305-306.

C. Claims

1. Claim 19

Meir in view of Salessi and Lovinger renders obvious this limitation. EX1003, ¶¶307-311.

Although Meir-Salessi does not explicitly disclose a *storage request* that *comprises a storage policy*, this would have been obvious in view of Lovinger. EX1003, ¶¶100-109.

Lovinger discloses that “the client 110 may store content 112 on a server 120.” EX1015, [0029]. A POSITA would have understood that storing content 112 requires the client to send a *storage request* that the storage device can execute. EX1003, ¶¶308-309. Lovinger further teaches a retention policy which is a *storage policy* because “the content is written to the one or more storage devices based on the retention policy.” EX1015, [0003]; *see also id.* [0029].

Lovinger discloses *storage request* that *comprises a storage policy*: “the client 110 may transmit the content 112 to the server 120 with information corresponding to the retention policy 130.” *Id.*, [0040]; *see also id.*, FIG. 1. Specifically, “when the content and retention policy have been received, the server 120 writes the content 112 to one or more storage devices associated with the server 120 according to the retention policy 130.” *Id.*, [0040].

As discussed, a POSITA would have been motivated to modify the Meir-

Salessi system to include the storage policy in the storage request as taught by Lovinger with a reasonable expectation of success. *See* Section X.A-B.

XI. SECONDARY CONSIDERATIONS

There are no secondary considerations known to Petitioners that affect, let alone overcome—this strong case of obviousness. Should PO proffer any relevant evidence of secondary considerations, Petitioners reserve its rights to address.

XII. CONCLUSION

For these reasons, Petitioners respectfully request institution.

U.S. Patent No. 12,265,715
Petition for *Inter Partes* Review

Dated: August 5, 2026

Respectfully submitted,

By: /s/Lisa K. Nguyen
Lisa K. Nguyen (Reg. No. 58,018)
PAUL HASTINGS LLP
1117 S. California Avenue
Palo Alto, CA 94304
Telephone: 650-320-1800
lisanguyen@paulhastings.com

Counsel for Petitioner
META PLATFORMS, INC.

CLAIM LISTING APPENDIX

Claim 1

- 1 [pre]** A solid-state drive comprising:
- 1 [a]** a solid-state memory;
- 1 [b]** a device memory;
- 1 [c]** a programmable processor operatively coupled to the device memory;
- 1 [d]** a device controller operatively coupled with the solid-state memory and the programmable processor; and
- 1 [e]** an application stored on the device memory comprising instructions that, when executed by the programmable processor, direct the programmable processor to:
- 1 [e.i]** process content for storage based on a storage policy;
- 1 [e.ii]** transfer the content to the device controller; and
- 1 [f]** the device controller to:
- 1 [f.i]** receive the content from the programmable processor; and
- 1 [f.ii]** store the content on the solid-state memory.

Claim 2

- 2[pre]** The solid-state drive of claim 1 wherein:
- 2[a]** the device controller is to:
- 2[a.i]** read requested content from the solid-state memory;

- 2[a.ii] transfer the requested content to the programmable processor; and
- 2[b] the instructions further direct the programmable processor to:
 - 2[b.i] receive the requested content from the programmable processor;
 - 2[b.ii] process the requested content for transfer based on the storage policy; and
 - 2[b.iii] transfer the requested content.

Claim 5

- 5[pre] The solid-state drive of claim 1 wherein:
 - 5[a] the instructions further direct the programmable processor to:
 - 5[a.i] compress the content based on the storage policy;
 - 5[a.ii] transfer the compressed content to the device controller; and
 - 5[b] the device controller is to:
 - 5[b.i] receive the compressed content from the programmable processor; and
 - 5[b.ii] store the compressed content on the solid-state memory.

Claim 6

- 6 The solid-state drive of claim 5 wherein:
 - 6[a] the device controller is to:
 - 6[a.i] read the compressed content from the solid-state memory; and:

- 6[a.ii] transfer the compressed content to the programmable processor;
and
- 6[b] the instructions further direct the programmable processor to:
 - 6[b.i] receive the compressed content from the device controller;
 - 6[b.ii] decompress the content based on the storage policy; and
 - 6[b.iii] transfer the decompressed content.

Claim 7

- 7 [pre] The solid-state drive of claim 1 wherein:
 - 7 [a] the instructions further direct the programmable processor to:
 - 7 [a.i] encrypt the content based on the storage policy; and
 - 7 [a.ii] transfer the encrypted content to the device controller; and
 - 7 [b] the device controller is to:
 - 7 [b.i] receive the encrypted content from the programmable processor;
and
 - 7 [b.ii] store the encrypted content on the solid-state memory.

Claim 8

- 8 [pre] The solid-state drive of claim 5 wherein:
 - 8 [a] The device controller is to:

- 8 [a.i] read the encrypted content from the solid-state memory; and
- 8 [a.ii] transfer the encrypted content to the programmable processor; and
- 8 [b] the instructions further direct the programmable processor to:
- 8 [b.i] receive the encrypted content from the device controller;
- 8 [b.ii] decrypt the content; and
- 8 [b.iii] transfer the decrypted content.

Claim 9

- 9[pre] The solid-state drive of claim 1 further comprising:
- 9[a] a communication interface to communicatively couple the solid-state drive to a host device;
- 9[b] a data link to communicatively couple the programmable processor and the device controller;
- 9[c] a printed circuit board to mount the solid-state memory, the device memory, the device controller, the programmable processor, and the data link; and
- 9[d] a case to mount the printed circuit board and the communication interface.

Claim 10

- 10 The solid-state drive of claim 1 wherein the content comprises one or more of video data, image data, on-line temporal video data,

cached data, financial data, online transaction data, credit card data, or medical data.

Claim 11

- 11 [pre]** A method of operating a solid-state drive, the method comprising:
- 11 [a]** in response to a storage request comprising content, processing the content based on a storage policy;
- 11 [b]** storing the processed content on the solid-state memory;
- 11 [c]** in response to a read request, reading requested content from the solid-state memory;
- 11 [d]** processing the requested content for transfer based on the storage policy; and
- 11[e]** transferring the requested content.

Claim 14

- 14** The method of claim 11 wherein:
- 14[a]** processing the content based on the storage policy comprises compressing the content based on the storage policy;
- 14[b]** storing the processed content on the solid-state memory comprises storing the compressed content on the solid-state memory;
- 14[c]** reading requested content from the solid-state memory comprises reading the compressed content from the solid-state memory;
- 14[d]** processing the requested content for transfer based on the storage policy comprises decompressing the compressed content based on the storage policy; and

- 14[e] transferring the requested content comprises transferring the decompressed content.

Claim 15

- 15 [pre] The method of claim 11 wherein:
- 15 [a] processing the content based on the storage policy comprises encrypting the content based on the storage policy;
- 15 [b] storing the processed content on the solid-state memory comprises storing the encrypted content on the solid-state memory;
- 15 [c] reading requested content from the solid-state memory comprises reading the encrypted content from the solid-state memory;
- 15 [d] processing the requested content for transfer based on the storage policy comprises decrypting the encrypted content based on the storage policy; and
- 15 [e] transferring the requested content comprises transferring the decrypted content.

Claim 16

- 16 The method of claim 11 wherein the storage policy indicates how to process content for storage on the solid-state drive and how to process the requested content for transfer.

Claim 17

- 17 The method of claim 16 wherein the storage policy comprises one or more of an encryption policy, a decryption policy, a compression policy, or a decompression policy.

Claim 18

- 18** The method of claim 17 further comprising retrieving the storage policy from memory.

Claim 19

- 19** The method of claim 17 wherein the storage request comprises the storage policy.

Claim 20

- 20[pre]** A solid-state drive comprising:
- 20[a]** a solid-state memory;
- 20[b]** a device memory;
- 20[c]** a programmable processor operatively coupled to the device memory;
- 20[d]** a device controller operatively coupled with the solid-state memory and the programmable processor; and
- 20[e]** instructions stored on the device memory that, when executed by the programmable processor, direct the programmable processor to:
- 20[e.i]** process content for storage on the solid-state memory based on a storage policy;
- 20[e.ii]** transfer the content to the device controller to store on the solid-state memory;

- 21 [e.iii] in response a content request, receive requested content read from the solid-state memory by the device controller;
- 21 [e.iv] process the requested content for transfer based on the storage policy; and
- 21 [e.v] storing lookup information comprising at least a content identifier for the content and an information pointer for the storage information;
- 21 [e.vi] transfer the requested content.

CERTIFICATE OF COMPLIANCE

Pursuant to 37 C.F.R. § 42.24(d), the undersigned certifies that foregoing
**PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO.
12,265,715** exclusive of the parts exempted as provided in 37 C.F.R. §42.24(a),
contains 13,921 words and therefore complies with the type-volume limitations of
37 C.F.R. §42.24(a).

Dated: August 5, 2026

Respectfully submitted,

By: /s/ Lisa K. Nguyen
Lisa K. Nguyen (Reg. No. 58,018)
PAUL HASTINGS LLP
1117 S. California Avenue
Palo Alto, CA 94304
Telephone: 650-320-1800
lisanguyen@paulhastings.com

Counsel for Petitioner
META PLATFORMS, INC.

CERTIFICATE OF SERVICE

I hereby certify that a true and correct copy of the foregoing **Petition for *Inter Partes* Review of U.S. Patent No. 12,265,715 and Exhibits** are being served on August 5, 2026 via Federal Express overnight mail on counsel of record for U.S. Patent No. 12,265,715 as addressed below:

**POLSINELLI PC
PO Box 140310
KANSAS CITY, MO 64114-0310**

A courtesy copy is also being served on counsel for the patent holder in the pending litigation:

Rajeev Gupta, raj.gupta@finnegan.com
Karthik Kumar, karthik.kumar@finnegan.com
Daniel F. Roland, daniel.roland@finnegan.com
Daniel F. Klodowski, daniel.klodowski@finnegan.com
C. Collette Corser, collette.corser@finnegan.com
Victor M. Palace, victor.palace@finnegan.com
Jordan M. Cowger, jordan.cowger@finnegan.com
FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER, LLP
901 New York Avenue, N.W., Washington, D.C. 20001-4413

Dated: August 5, 2026

Respectfully submitted,

By: /s/ Lisa K. Nguyen
Lisa K. Nguyen (Reg. No. 58,018)
PAUL HASTINGS LLP
1117 S. California Avenue
Palo Alto, CA 94304
Telephone: 650-320-1800
lisanguyen@paulhastings.com

Counsel for Petitioner
META PLATFORMS, INC.

