

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SANDISK TECHNOLOGIES, INC. AND WESTERN DIGITAL
TECHNOLOGIES, INC.,
Petitioner

v.

POLARIS POWERLED TECHNOLOGIES, LLC,
Patent Owner.

IPR2025-00515
U.S. Patent No. 9,183,085

**PETITION FOR *INTER PARTES* REVIEW
UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104**

TABLE OF CONTENTS

PETITIONER’S EXHIBIT LIST	5
I. INTRODUCTION	7
II. GROUNDS FOR STANDING.....	7
III. NOTE.....	7
IV. BACKGROUND	8
V. THE ’085 PATENT.....	8
VI. PROSECUTION HISTORY OF THE ’085 PATENT.....	8
VII. LEVEL OF ORDINARY SKILL IN THE ART.....	9
VIII. CLAIM CONSTRUCTION	10
A. “gear” (claim 1).....	10
B. Means-Plus-Function Terms	10
IX. RELIEF REQUESTED AND THE REASONS FOR RELIEF	11
X. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE....	11
A. Challenged Claims	11
B. Prior Art Patents and Printed Publications.....	11
C. Statutory Grounds for Challenges	12
D. Prior Art.....	13
1. Li	13
2. Li combined with Cideciyan.....	14
3. Li further combined with Frost.....	16

4.	Li further combined with Leung.....	16
E.	Ground 1: Claims 1-2, 4-5, 14-17, 22-25, and 29-33 are obvious in view of Li and Cideciyan.	16
1.	Claim 1	16
2.	Claim 2.....	52
3.	Claim 4.....	53
4.	Claim 5.....	56
5.	Claim 14.....	58
6.	Claim 15.....	60
7.	Claim 16.....	64
8.	Claim 17.....	65
9.	Claims 22 and 29.....	66
10.	Claim 23	67
11.	Claim 24.....	67
12.	Claim 25	70
13.	Claim 30.....	71
14.	Claims 31, 32, and 33	72
F.	Ground 2: Claims 1-2, 4-5, 14-17, 22-25, and 29-33 are obvious in view of Li, Cideciyan, and Frost.....	76
1.	Claim Analysis.....	76
G.	Grounds 3 and 4: Claims 6-7 and 18 are rendered obvious by each of Combination of Li, Cideciyan, and Leung (Ground 3) and Combination of Li, Cideciyan, Frost, and Leung (Ground 4).	80

1.	Reasons to Combine Leung with Each of Grounds 1 and 2.....	80
2.	Claim 6.....	81
3.	Claim 7.....	82
4.	Claim 18.....	83
XI.	DISCRETIONARY DENIAL IS NOT WARRANTED.....	85
A.	The Fintiv factors favor institution.	85
1.	No evidence regarding a stay	85
2.	Parallel proceeding trial date	85
3.	Investment in the parallel proceeding.....	86
4.	Overlapping issues with the parallel proceeding	86
5.	Petitioner is a defendant.....	86
6.	Other circumstances.....	86
B.	Advanced Bionics Test Favors Institution.	87
XII.	CONCLUSION.....	88
XIII.	MANDATORY NOTICES	88
A.	Real Party-in-Interest	88
B.	Related Matters.....	88
C.	Lead and Back-up Counsel and Service Information	89
	CERTIFICATE OF WORD COUNT	90
	CERTIFICATE OF SERVICE	91

PETITIONER’S EXHIBIT LIST

Ex.1001	U.S. Patent No. 9,183,085 (“the ’085 patent”)
Ex.1002	Prosecution History of U.S. App. No. 13/477,600 (issued as ’085 patent) (“’600 App”)
Ex.1003	Declaration of Dr. Lara Dolecek (“Dolecek Decl.”)
Ex.1004	CV of Dr. Lara Dolecek
Ex.1005	U.S. Pub. 2014/0136927 to Li, et al.
Ex.1006	U.S. 9,037,951 to Cideciyan, et al.
Ex.1007	U.S. Patent No. 8,176,360 to Frost et al.
Ex.1008	U.S. 2008/0222490 to Leung et al.
Ex.1009	Solid-State Drive (SSD) Requirements and Endurance Test Method, JESD218, JEDEC STANDARD, September 2010
Ex.1010	U.S. 7,823,043 to Lasser
Ex.1011	U.S. 8,042,029 to Alrod et al.
Ex.1012	Performance Impact and Interplay of SSD Parallelism through Advanced Commands, Allocation Strategy and Data Granularity, Hu et al., ICS ’11: Proceedings of the international conference on Supercomputing, pp. 96 – 107 (https://doi.org/10.1145/1995896.1995912), May 31, 2011
Ex.1013	FTL Design Exploration in Reconfigurable High-Performance SSD for Server Applications, ICS '09: Proceedings of the 23rd international conference on Supercomputing, pp. 338 – 349 (https://doi.org/10.1145/1542275.1542324), June 8, 2009
Ex.1014	Flash Management – Why and How?, A Detailed Overview of Flash Management Techniques, SMART Modular Technologies, 2009
Ex.1015	Complaint, <i>Polaris PowerLED Technologies, LLC v. Western Digital Corporation et al.</i> 2:24-cv-02864 (C.D. CA)

Ex.1016	District Court Trial Statistics
Ex.1017	Director Vidal Memorandum re: Interim Procedure for Discretionary Denials in AIA Post-Grant Proceedings with Parallel District Court Litigation, June 22, 2022
Ex.1018	Solid State Drives 101, Cactus Technologies, available at https://www.cactus-tech.com/wp-content/uploads/2019/04/Solid-State-Drives-101-EBook.pdf
Ex.1019	Product Code Schemes for Error Correction in MLC NAND Flash Memories, Yang et al., December 29, 2011, available at: https://ieeexplore.ieee.org/document/6118315 .
Ex.1020	Error Control Coding Fundamentals and Applications, Lin et. al, 1983
Ex.1021	Scheduling and Case Management Order, <i>Polaris PowerLED Technologies, LLC v. Western Digital Corporation et al.</i> 2:24-cv-02864 (C.D. CA)

I. INTRODUCTION

U.S. Patent No. 9,183,085 (“the ’085 patent”) is directed to memory systems such as flash memory systems, and more specifically to adaptively selecting an error correcting coding (ECC) scheme based on an observed bit error rate, to address memory degradation over its lifetime. Ex.1001, Abstract. As shown herein, a single prior art reference – Li (Ex.1005) – teaches or suggests nearly all the features of the challenged claims. Other references are used merely to provide obvious example implementations or to supplement teachings that are suggested in Li.

Accordingly, pursuant to 35 U.S.C. §§ 311, 314(a), and 37 C.F.R. § 42.100, Petitioner respectfully requests that the Board review and cancel as unpatentable under (pre-AIA) 35 U.S.C. § 103(a) claims 1-2, 4-7, 14-18, 22-25, and 29-33 (the “Challenged Claims”).

II. GROUNDS FOR STANDING

Petitioner certifies that the ’085 patent is eligible for IPR and that Petitioner is not barred or estopped from requesting IPR challenging the patent claims. 37 C.F.R. § 42.104(a).

III. NOTE

Petitioner cites to exhibits’ original page numbers. Emphasis in quoted material has been added.

IV. BACKGROUND

Error Correction Coding (ECC) is a well-known technique by which a memory controller can detect and correct data errors in a memory system. There are many different types of ECC schemes, some being more robust than others in certain situations. One type of memory system that uses ECC is a flash memory system, e.g., a flash based Solid-State Drive (SSD). Dr. Dolecek’s declaration provides a brief background section describing flash memory systems, error correction code techniques including product code, and SSDs. Ex.1003, 25-46.

V. THE ’085 PATENT

The ’085 patent is directed toward adaptively selecting an ECC scheme from a plurality of such schemes based on data error rate. Ex.1001, Title. The ’085 patent criticizes the conventional approaches of “specifying a particular strength of ECC code” through the total service life. Ex.1001, 1:63-2:1; 2:23-27; Ex.1003, ¶¶ 47-48.

One purported novelty of the ’085 patent is “*Adaptive ECC*” (Ex.1001, 48:47), which selects from ECC schemes of different ECC strengths (corresponding to “gears”) based on an observed bit error rate. *See also* Ex.1001, Title, 51:1-2; Ex.1003, ¶49.

VI. PROSECUTION HISTORY OF THE ’085 PATENT

The ’085 patent application was filed on May 22, 2012. In response to the

rejections, Patentee amended the claims to recite “determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof, wherein the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region.” (*see* limitations [1.1.1] and [1.1.2]). Ex.1002, 614-630. The amended claims were again rejected based on prior art. Ex.1002, 642-679.

In response, patentee amended the claims to recite “wherein the two or more flash memory pages of a region can be read simultaneously,” (*see* limitation [1.1.1]), which resulted in allowance. Ex.1002, 693-705, 715-719.

VII. LEVEL OF ORDINARY SKILL IN THE ART

A Person of Ordinary Skill in The Art (“POSITA”) as of May 22, 2012 would have had a working knowledge of the flash memory system art, including flash memories, memory controllers, and ECC art as applied to flash memory systems that is pertinent to the ’085 patent. A POSITA would have had a bachelor’s degree in electrical engineering, or an equivalent, and approximately two years of professional experience relating to the field of flash memory systems, including ECC technologies as applied to flash memory systems. Lack of professional experience can be remedied by additional education, and vice versa. Ex.1003, ¶¶14-18.

VIII. CLAIM CONSTRUCTION

In light of the prior art, no claim term other than the terms identified below requires construction. EX.1003, ¶¶55-59.

A. “*gear*” (claim 1)

A POSITA would have understood that “*gear*” is defined by the specification as follows:

In the context of this disclosure, a “*gear*” is a set of parameter(s) specifying an error correction coding scheme, including one or more of the payload capacity, the type and parameters of each component code, the interleaving scheme, and parameters used for determining whether the gear is appropriate for a particular situation. For example, the “Gear 1” sections of Tables III, IV, and V, plus the Threshold G1, collectively define Gear 1.

Ex.1001, 20:18-25; Ex.1003, ¶¶57-58.

B. Means-Plus-Function Terms

Claims 24-25 include means-plus-function claim terms. Construction of means-plus-function claim terms that invoke 35 U.S.C. §112, ¶6, requires identifying the claim terms and identifying corresponding structure disclosed in the specification that performs the identified functions and equivalents thereof. These means-plus-function claim terms are identified below at [24.1.1], [24.2.1], [24.3],

[25.0], and [25.1], together with the corresponding structure disclosed in the specification.

IX. RELIEF REQUESTED AND THE REASONS FOR RELIEF

Petitioner asks that the Board institute a trial for *inter partes* review and cancel the Challenged Claims in view of the analysis below.

X. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

A. Challenged Claims

Petitioner challenges claims 1-2, 4-7, 14-18, 22-25, and 29-33 of the '085 patent.¹

B. Prior Art Patents and Printed Publications

1. U.S. Pub. 2014/0136927 to Li, et al. ("Li") (Ex.1005), PCT filed October 26, 2011, published May 15, 2014; is prior art under §§102(a) and (e).
2. U.S. 9,037,951 to Cideciyan, et al. ("Cideciyan") (Ex.1006), PCT filed December 12, 2010, PCT published June 23, 2011, issued May 19, 2015; is prior art under §§102(a) and (e).
3. US 8,176,360 to Frost, et al. ("Frost") (Ex.1007), filed September 5, 2009, published February 17, 2011, issued May 8, 2012; is prior art under §§102(a) and (b).

¹ The pre-AIA statutory framework applies to the '085 patent.

4. US 2008/0222490 to Leung et al. (“Leung”) (Ex.1008), filed March 8, 2007, published September 11, 2008; is prior art under §§102(a) and (b).

Li (Ex.1005) and patents related to Frost (Ex.1007, specifically US 8,255,620; 8,560,881; and 8,713,245) were cited during the original prosecution, but were never relied upon by the Office. Ex.1002, 633, 730.

Cideciyan (Ex.1006) and Leung (Ex.1008) were not cited during the original prosecution. As explained in detail below, Cideciyan addresses the claim 1 limitations: “*wherein the two or more flash memory pages of a region can be read simultaneously*” and “*wherein the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region*” which served as a basis for allowance during the original prosecution.

C. Statutory Grounds for Challenges

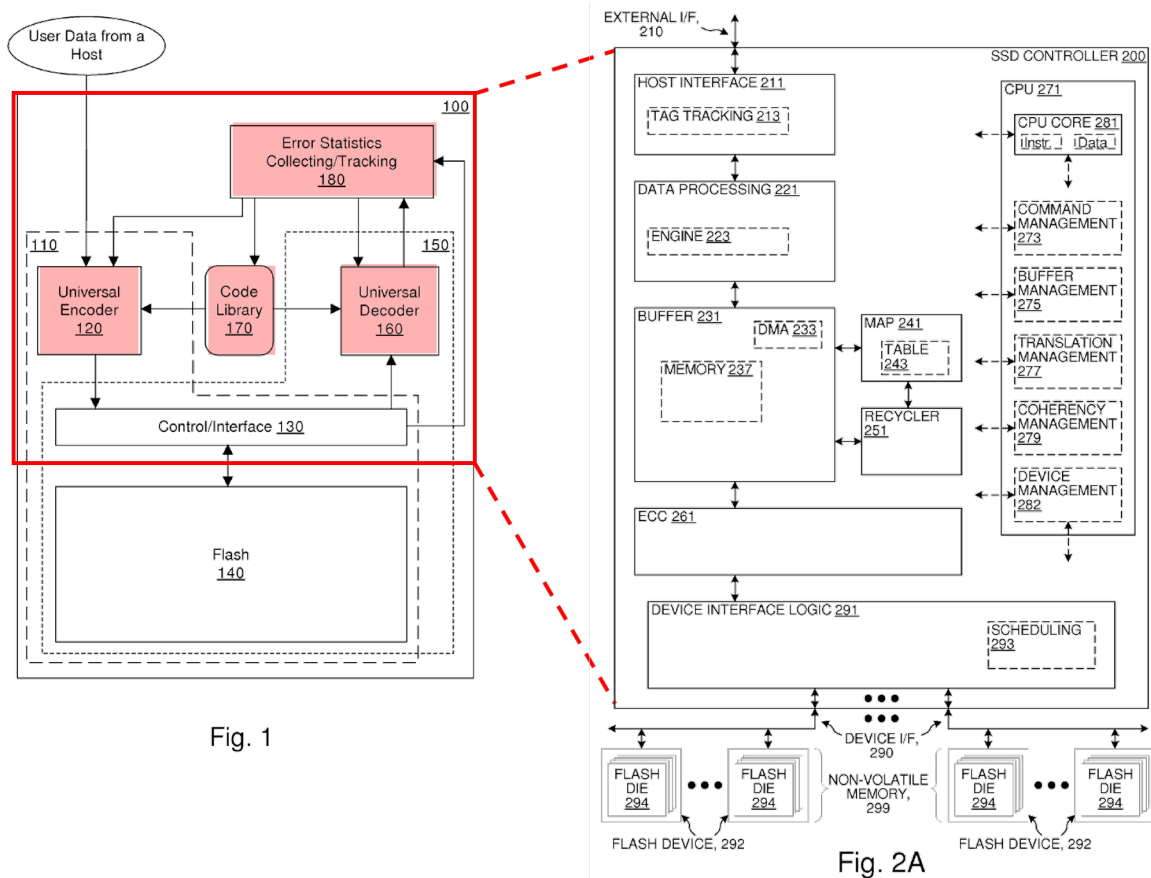
Grounds	Claims	Basis
#1	1-2, 4-5, 14-17, 22-25, 29-33	Obvious under §103 over Li and Cideciyan
#2	1-2, 4-5, 14-17, 22-25, 29-33	Obvious under §103 over Li, Cideciyan, and Frost
#3	6-7, and 18	Obvious under §103 over Li, Cideciyan, and Leung
#4	6-7, and 18	Obvious under §103 over Li, Cideciyan, Frost, and Leung

D. Prior Art

Li is the primary prior art reference for each ground listed above. Cideciyan, Frost, and Leung are secondary references provided to show various teachings or example implementation details that would be obvious for Li to use. Ex.1003, ¶60.

1. Li

As shown in FIGS. 1 and 2A below, Li, like the '085 patent, relates to “adaptive ECC techniques for flash memory based data storage.” Ex.1005, Title, [0126]; Ex.1003, ¶62.



Ex.1005, FIGS. 1 and 2A, annotated.

Li states that “**adaptive ECC** techniques for use with flash memory enable improvements in flash memory lifetime, reliability, performance, and/or storage capacity.” Ex.1005, [0019]. “The techniques include **a set of ECC schemes with various code types, code rates, and/or various code lengths** (providing **different error correcting capabilities**), and **error statistic collecting/tracking**).” *Id.* Li also describes conditions (e.g., thresholds) used to determining whether a particular ECC scheme is appropriate for a particular situation. Ex.1005, [0105]. Further discussion of these terms, including common examples, is provided in the expert declaration. Ex.1003, ¶¶63-64.

Li’s adaptive ECC scheme selection is performed on the basis of a “portion” (*a region*), which includes monitoring BER of the “portion,” selecting the ECC scheme for the “portion,” and performing encoding/decoding with the selected ECC scheme for that “portion.” *See, e.g.*, Ex.1005, [0022], [0050], claim 15.

2. Li combined with Cideciyan

While Li describes a set of ECC schemes with **various code types** (Ex.1005, [0019]), Cideciyan provides an example ECC scheme using a well-known “product code” code type, described in detail in textbooks. *See, e.g.*, Ex.1020, 274-278; Ex.1006, FIG. 4 (illustrating a “C1/C2 product code” with component codes C1 and C2), 10:44-49. Cideciyan’s product-code-type ECC scheme is applied on a “group” basis, where the resulting array of encoded symbols (including data

symbols and corresponding parity symbols including C1 and C2 parity symbols) are stored in a single group of pages (*a region*). Ex.1003, ¶66.

Thus, Li combined with Cideciyan teaches: *such that the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region* – one of the two claim elements added during prosecution of the '085 patent. Ex.1003, ¶67.

Cideciyan also addresses the other claim element added during prosecution. That is, Cideciyan teaches organizing the pages in groups, where each group includes a fixed number of multiple pages across different channels, **accessible in parallel**. Ex.1006, FIG. 2, 8:52-64, 11:44-46. The array of encoded symbols stored in a single group thus span across the multiple channels. Ex.1006, FIG. 5. A POSITA would understand that pages being parallelly accessible via multiple channels means that the pages can be read simultaneously. Ex.1003, ¶68, citing Ex.1012, 2; Ex.1018, 18; Ex.1007, Abstract, FIGS. 7A-7B, 11:20-64.

Thus, Li combined with Cideciyan teaches: *wherein the two or more flash memory pages of a region can be read simultaneously* – one of the two claim elements added during prosecution of the '085 patent. Ex.1003, ¶69.

Multiple reasons to combine Li and Cideciyan are described in more detail in the analysis below. Ex.1003, ¶70.

3. Li further combined with Frost

Frost is also directed at a “flash-based storage system,” and describes in greater detail how data in a particular region accessible in parallel can be read, “**at the same time.**” Ex.1007, Title, 6:46-67. In view of Frost, it would be obvious for Li’s parallel access capabilities to include *wherein the two or more flash memory pages of a region can be read simultaneously.* Ex.1003, ¶71. The reasons to combine Frost with Li and Cideciyan are described in more detail in the analysis below.

4. Li further combined with Leung

Leung is like Li in that it teaches selecting a gear in response to a change in a bit error rate. Leung is used to address some of the implementation details of the dependent claims directed to gear selection. Ex.1003, ¶72. The reasons to combine Frost with Li and Leung are described in more detail in the analysis below.

E. Ground 1: Claims 1-2, 4-5, 14-17, 22-25, and 29-33 are obvious in view of Li and Cideciyan.

1. Claim 1

[1.0] A method of selecting an error correction coding (ECC) scheme, the method comprising:

To the extent the preamble is limiting, Li renders obvious [1.0]. Ex.1003, ¶¶74-75.

Specifically, Li describes “encoding/decoding **in accordance with one or**

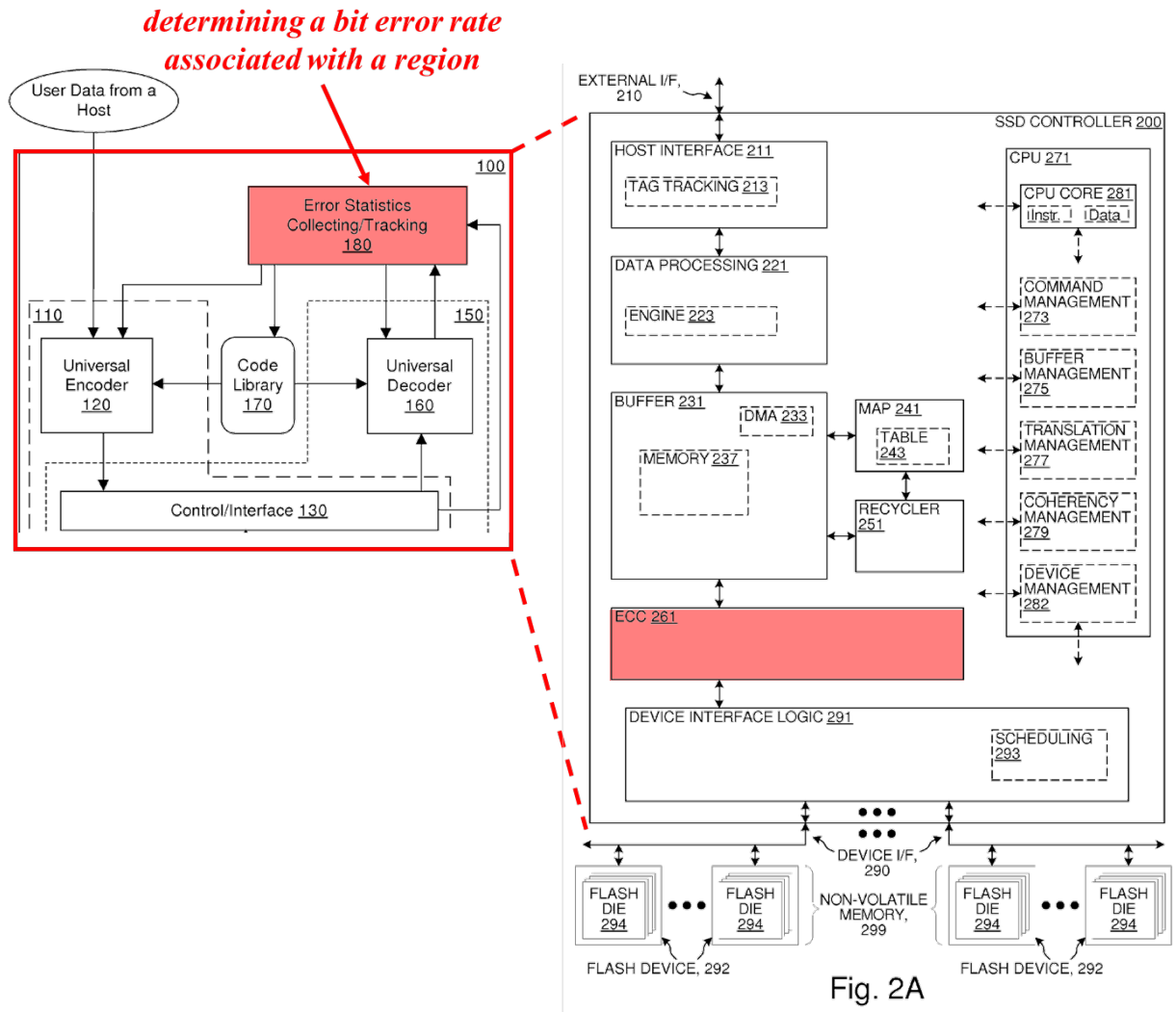
more of the ECC schemes, and dynamically switching encoding/decoding amongst one or more of the ECC schemes based at least in part on information from the error statistic collecting/tracking.” Ex.1005, Abstract; Ex.1003, ¶75.

[1.1.1] *determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof, wherein the two or more flash memory pages of a region can be read simultaneously,*

Li combined with Cideciyan renders obvious [1.1.1]. Ex.1003, ¶¶76-99.

a. Li teaches “*determining a bit error rate associated with a region.*”

As shown in FIGS. 1 and 2A below, Li teaches using an error statistics collecting and tracking hardware logic block to *determine* a raw Bit Error Rate (BER) associated with a “portion” (*a region*). Ex.1005, [0022], [0097], [0099], [0038], [0039]; Ex.1003, ¶77; ¶26.



Ex.1005, FIGS. 1 (partial) and 2A, annotated.

In more detail, Li describes adaptively selecting an ECC scheme with a correction strength for each “portion” of a plurality “portions” of the memory. In other words, Li’s adaptive ECC scheme selection is performed on the basis of a particular “portion” (*a region*), which includes monitoring BER of the “portion,” selecting the ECC scheme for the “portion,” and performing encoding/decoding with the selected ECC scheme for that “portion” (*the region*). See, e.g., Ex.1005,

[0022] (“determine a raw Bit Error Rate (BER) of accesses to **a portion** of a flash memory”); [0047]-[0048], [0050]-[0051]; claim 15; *see also id.* [0042]-[0043]; Ex.1003, ¶78.

b. Li’s “portion” associated with the bit error rate determination comprises a fixed number of multiple pages.

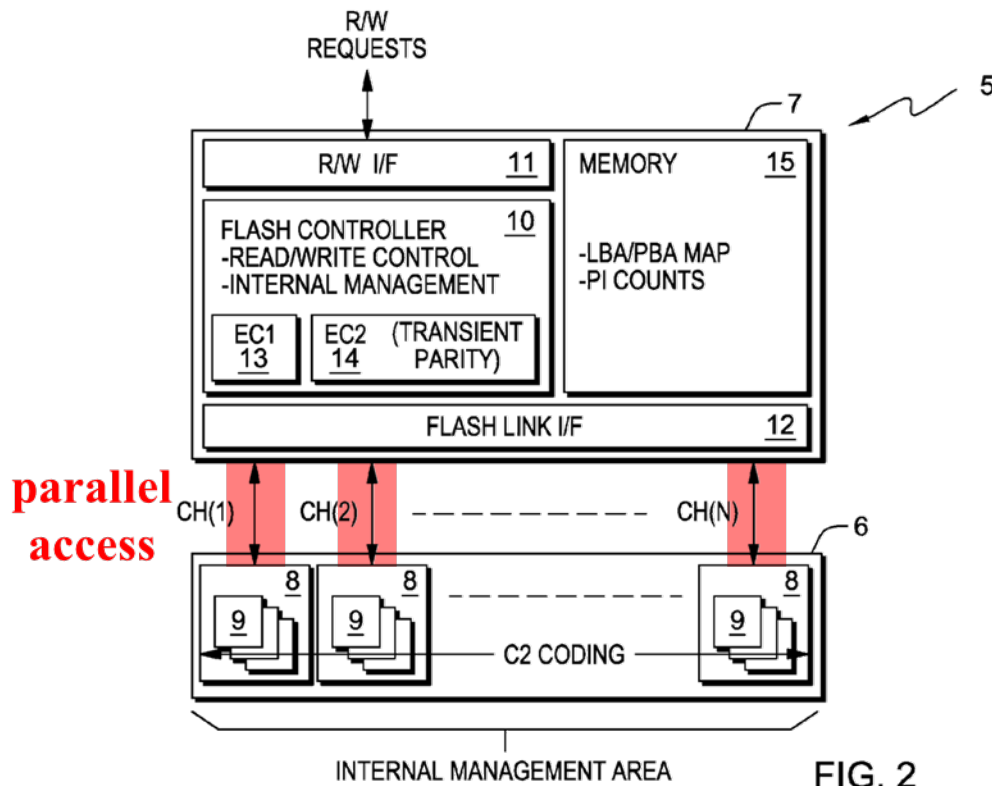
Li describes a “portion” of the flash memory that includes multiple pages, and “each of the portions is enabled to store a **pre-determined** amount of information” (*comprising a fixed number of two or more flash memory pages*). Ex.1005, [0099] (“a flash memory is operated as portions (such as **pages[]** or **multiples** thereof), [0038] (“the portion is one or **more** pages of the flash memory”). Li’s “portion” including a predetermined fixed number of pages, which is fixed by the “predetermined amount of information” of the portion and the predetermined fixed size for each page. *Id.*, [0097] (“each of the portions is enabled to store a **pre-determined** amount of information”). Ex.1003, ¶¶79-80.

c. It would be obvious for Li’s portion including a fixed number of multiple pages to span multiple channels as taught by Cideciyan, thereby supporting simultaneous read for the pages.

First, Li explains that its plurality of flash dies in the flash memory are “**accessible in parallel.**” Ex.1005, Li, [0111], [0112]. Li also describes that its adaptive ECC scheme selection is performed on a “portion” basis, where the portion can be a group of multiple pages. Ex.1005, Li, [0022], [0097], [0099],

[0038], [0039]. Ex.1003, ¶64. As such, in light of the well-known benefits that parallel access improves performance (e.g., reduced latency, increased speed, distributed wear for extending the memory device life span) by allowing simultaneous read/write operations across the busses, Li's flash dies accessible in parallel (Ex.1005, [111]-[112]) at least suggest organizing its portion including a fixed number of multiple pages to span multiple channels for simultaneous read and write and improved performance. Ex.1003, ¶64-81.

Second, as shown in FIG. 2 of Cideciyan below, Cideciyan describes **N channels** (CH(1) through CH(N)) in the SSD, providing **parallel access** to pages of the flash memory chips. Ex.1003, ¶81.



Ex.1006, FIG. 2, annotated

In further detail, Cideciyan describes “N logical subdivisions corresponding to respective storage banks 8 on the set of N channels,” which are accessible in parallel. Ex.1006, 10:3-11; 11: 44-46 (pages allocated on different channels “can be written **in parallel** to the N channels”). Ex.1003, ¶81.

Cideciyan teaches “storing input data in **groups**” (Ex.1006, 19:43), and performing ECC encoding, storing the encoded symbols, and ECC decoding in “groups,” each group including pages distributed across the channels. Ex.1006, FIGS. 4 and 5, 4:30-45 (“storing input data in successive **groups**,” “encoding the

input data to be **stored in each group**"). As will be discussed in detail at [1.1.2] below, the array of encoded symbols (including data symbols and parity symbols as illustrated in Ex.1006, FIG. 4) generated using Cideciyan's product-code-type ECC scheme will be written into a single group of pages distributed across the channels. Similarly, Cideciyan's decoding process is performed on data read from each group. Ex.1003, ¶82.

Cideciyan teaches "the **pages of each group** are **distributed across all [N]** **logical subdivisions.**" Ex.1006, Abstract; 19:43; 10:9-11. Cideciyan uses the term "stride" to refer to such a group of pages distributed across the channels. *Id.*, 10:11-12. As shown in FIG. 5 below, the example group of pages (a stride) includes "one page in each of the N channels," which is what is more commonly referred to as a "page stripe." Ex.1006, 11:9-18; Ex.1003, ¶¶83, 46. The group of pages ("page stripe") of FIG. 5 is *a region comprising a **fixed** number (N) of multiple pages as claimed.* As shown in FIG. 5, there are at least 4 channels, which teaches *a fixed number of **two or more** flash memory pages.* Ex.1003, ¶83.

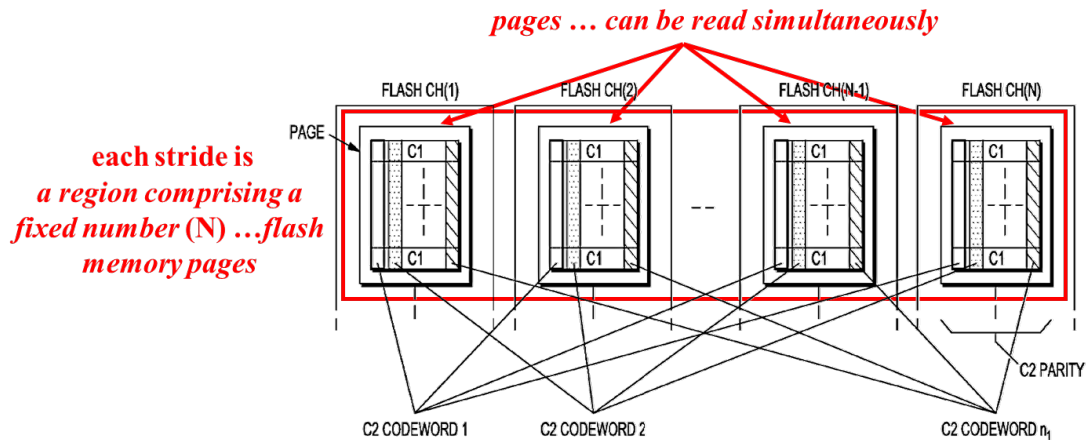


FIG. 5

Ex.1006, FIG. 5, annotated.

Thus, it would be obvious for Li's portion including a fixed number of multiple pages to span multiple channels as taught by Cideciyan, which achieves various benefits of parallel access, and additional benefits of implementing the product-code-type ECC scheme on the group basis as taught by Cideciyan (e.g., "efficient, reliable and robust solid state storage systems with exceptional overall performance" with orthogonal primary and secondary codewords, with "improved error protection offered by the coding process both within storage subdivisions and across the storage as a whole." Ex.1006, 5:2-13), which will be discussed in detail in [1.1.2]. Ex.1003, ¶84.

d. It would be obvious for multiple pages spanning multiple channels to be read simultaneously.

In the group of pages in FIG. 5 of Cideciyan above, each of the N pages can

be accessed in parallel through the N channels, and a POSITA would understand that accessing the pages in parallel includes performing *read* operations *simultaneously* using the N channels. Ex.1012, FIG. 4, 98 (“a **request is served by channel 0 and channel 1 simultaneously**, it exploits the channel-level parallelism.”), FIG. 11, 102; Ex.1018, 18 (“Channels refer to the number of flash chips the controller can talk to **simultaneously**.”). Ex.1003, ¶85.

To the extent that it is argued that parallel access is not necessarily simultaneous, that argument is addressed at Ground 2 below. Ex.1003, ¶86.

e. Reasons to Combine Li and Cideciyan

It would have been obvious to a POSITA to combine Li and Cideciyan to provide the obvious, predictable, and beneficial results obtained from Cideciyan’s memory arrangement and accessibility. Ex.1003, ¶87.

First, the implementation with Cideciyan’s technique of organizing data in “groups,” each group include pages distributed across channels, has the advantage that the fixed number of multiple pages in a group are distributed across the channels of an SSD, utilizing the channel-level parallelism in SSD to gain better parallel performance, which is important for overall SSD performance. Ex.1012, 105 (explaining the well-known channel-level parallelism in SSDs which is important for SSD performance); Ex.1013, 1 (describing various well-known technologies of using flash channels in an SSD for achieving desirable parallel

read and write access). Ex.1003, ¶88. Benefits of parallelism include lower latency, better data protection by spreading data across multiple dies if one die has higher failures. *See, e.g.*, Ex.1006, 5:2-6 (“protection against failures of individual storage subdivisions in the system” by error correcting coding process across the channels); Ex.1007, Frost, 11:57-60 (parallel access through channels allowing “relatively quick read and write operations and allows data to be stored to and retrieved from the memory array 14 in an efficient and effective manner.”). Ex.1003, ¶88.

Second, Cideciyan provides implementation details of the “portions” and the parallel accessibility across a plurality of flash dies described in Li. Therefore, it would be obvious to implement Li’s teaching of organizing the flash memory (including a plurality of flash dies assessable in parallel) in portions, with the organization of the flash memory in groups of pages across channels as taught by Cideciyan. Ex.1003, ¶89.

Li describes organizing the flash memory (including a plurality of dies accessible in parallel), in a plurality of “**portions**” such that each “portion” includes a fixed number of multiple pages. Li describes applying ECC schemes and monitoring the bit error rate of “portions” generally. Ex.1005, Li, [0037]-[0038], [0099]. A POSITA would have found it obvious to look to known techniques for how to implement those portions. Ex.1003, ¶90.

Cideciyan is such a reference. The specific implementation in Cideciyan's FIG. 5 includes storing data in groups, each group including N pages across N channels, with the advantage that each page is accessible in parallel through the N channels, thereby achieving channel-level parallelism among these pages in the group. It was well-known in the art that channel-level parallelism is important for achieving peak performance in SSD. Ex.1012, 2. Ex.1003, ¶91.

Additionally, Cideciyan is a patent application by International Business Machines (IBM) Corporation, which was a top storage provider during the timeframe of the earliest priority date of the '085 patent. A POSITA with Li in hand would have looked to patents and publications like Cideciyan from IBM to improve Li's flash-based SSD. Ex.1003, ¶¶91-95.

Third, as shown in FIG. 2A of Li and FIG. 2 of Cideciyan below, Li and Cideciyan are similar devices with similar objectives. Ex.1005, FIG. 2A (illustrating a flash memory based storage system with SSD controller and flash devices accessible in parallel channels), [0019] (aiming for “improvements in flash memory lifetime, reliability, performance, and/or storage capacity”); Ex.1006, FIG. 2 (illustrating a flash memory based storage system with a flash controller and flash devices accessible in parallel channels), 5:11-13 (to “provide efficient, reliable and robust solid state storage systems with exceptional overall performance”). Ex.1003, ¶96.

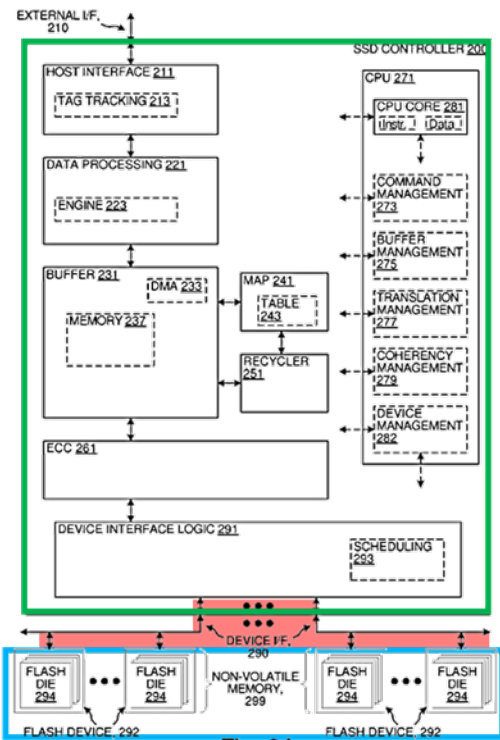


Fig. 2A

Ex.1005, Li, FIG. 2A, annotated

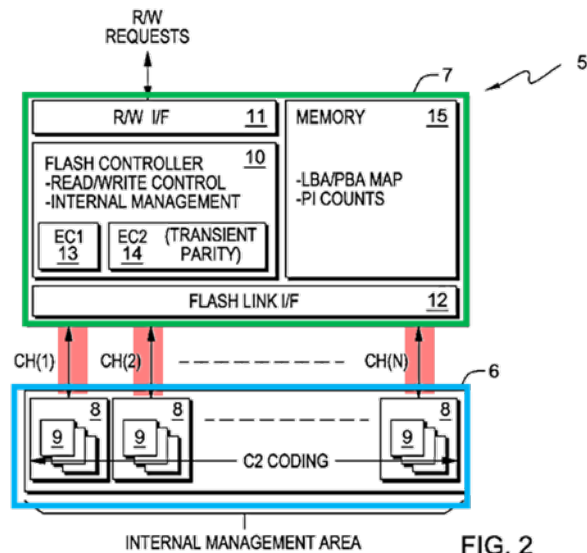


FIG. 2

Ex.1006, Cideciyan, FIG. 2,
annotated

A POSITA would recognize that a known technique (using strides, where each stride including N pages distributed across N channels) has been used to improve one device (the flash-based storage system of Cideciyan), it would improve similar devices (Li's flash memory based storage system) in the same way (with the benefits as taught by Cideciyan), using the prior art elements (using group of pages, where each group including N pages distributed across N channels) according to their established functions. Ex.1003, ¶97; *Intel Corp.*, 61 F.4th at 1380–81.

Fourth, and as will be explained in detail below at [1.1.2], it would have

been obvious to implement Li's ECC technique using the product code type ECC scheme applied on groups of pages as taught by Cideciyan. Using that technique, the "portion" for ECC and the BER monitoring of Li associated with a portion would span multiple channels that are accessible simultaneously and in parallel. Ex.1003, ¶87.

Additionally, the results of the proposed combination would have been predictable and a POSITA would have had a reasonable expectation of success, because the combination of Li and Cideciyan merely applies known technique to a known device (Li's flash memory based storage system implementing adaptive ECC schemes) to yield the predictable results of "efficient, reliable and robust solid state storage systems with exceptional overall performance." Ex.1006, 5:11-13; Ex.1003, ¶98.

As shown in Example 1.1 of Li combined with Cideciyan (based on based on FIGS. 1, 2A of Li and FIG. 5 of Cideciyan) below, a POSITA would have reasonably expected to succeed in combining Li with Cideciyan, because Cideciyan's group-based ECC encoding/decoding is described as applicable to flash memory devices such as those taught by Li. *See, e.g.*, Ex.1006, FIG. 2. Additionally, the implementation of Cideciyan's teaching would have required only routine knowledge well within the skill of a POSITA prior to the earliest effective filing date of the '085 patent. *See, e.g.* Ex.1006, 11:55-57

(“Corresponding decoder implementations will be readily apparent to those skilled in the art from the encoder description”); 17:13-17 (describing that considerations related to grouping pages “are similar to those encountered in RAID applications as will be apparent to those skilled in the art.”); Ex.1003, ¶¶99.

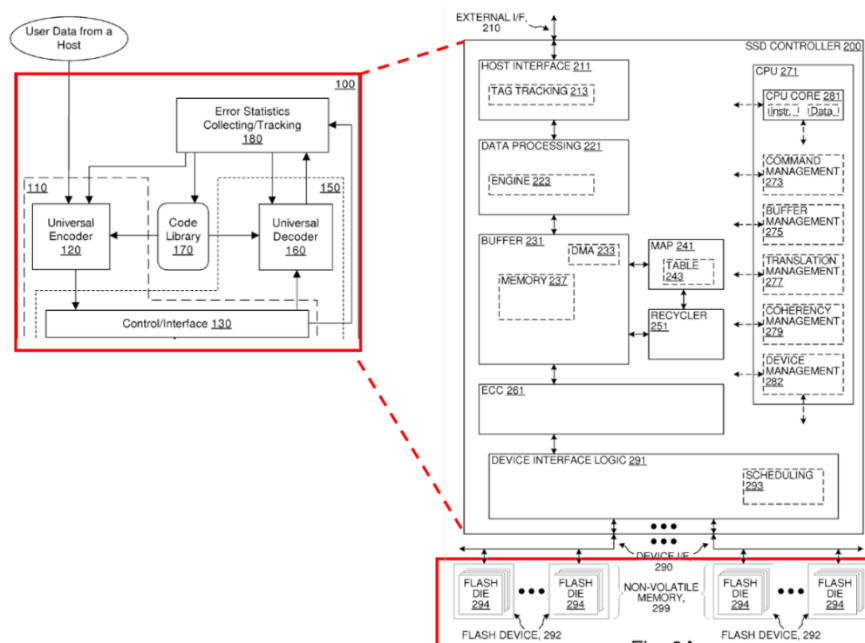
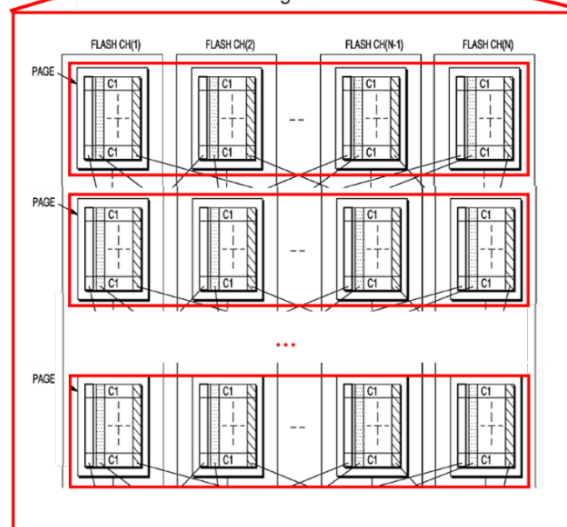


Fig. 2A

region

region

region



Example 1.1 of Combination of Li and Cideciyan.

[1.1.2] wherein the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region;

Li combined with Cideciyan renders obvious [1.1.2]. Ex.1003, ¶¶100-118.

Li teaches that the “portion” of a flash memory (“*the region*”) as discussed at [1.1.1] stores user data and a number of error correcting bits (“*data payload and ECC parity symbols corresponding to the data payload of the region*”). Ex.1003, ¶101.

For example, claims 1-3 of Li recite:

1. A system, comprising:

means for error statistics collecting and tracking enabled to dynamically determine a raw Bit Error Rate (BER) of accesses to a portion of a flash memory; and

means for adaptive encoding enabled to encode according to a dynamically selected one of a plurality of error correcting codes, and further enabled to dynamically determine the dynamically selected error correcting code based at least in part on the raw BER.

2. The system of claim 1, wherein encoding according to a first one of the error correcting codes results in a number of error correcting bits to store in the portion that is less than when encoding according to a second one of the error correcting codes.

3. The system of claim 2, wherein when encoding according to the first error correcting code, a number of bits of the portion used as user data is increased by up to a difference between the number of error correcting bits used by the second error correcting code minus the number of error correcting bits used by the first error correcting code.

See also Ex.1005, [0021]-[0025], claim 15; [0099] (“each portion is organized as a **data sub-portion** and a respective corresponding **error correcting sub-portion**”); [0060] (“the portion comprises a plurality of sub-portions, and the adaptive encoder is further enabled to encode such that error correcting information is storable to one or more of the sub-portions and data information is storable to only one of the sub-portions”); Ex.1003, ¶101.

It would have been obvious to implement Li’s ECC technique using one of the known, available ECC techniques such as Cideciyan’s product code type ECC scheme. As discussed below, Cideciyan describes implementing a product code ECC scheme on a group basis, where each group spans across the channels, which provides implementation detail of an ECC scheme and associated “portions” that would be obvious to use in Li. Ex.1003, ¶102.

Generation of the primary and secondary ECC parity symbols:

Cideciyan describes **generating** the primary codewords (including primary parity symbols) and secondary codewords (including secondary ECC parity

symbols) using a product code type ECC scheme as shown in FIG. 4. To be clear, FIG. 4 is merely a logical representation of the well-known product code scheme (see, e.g., Ex.1020, FIG. 9.6; Ex.1019, FIG. 5), which illustrates the encoded symbols using the product code (with component codes C1 and C2) **to be stored in a single physical region**. FIG. 4 does not display physical pages of a flash memory. However, FIG. 5 which is discussed in detail below, is a **physical** representation of **a single physical region** (a group of N pages from across N channels (“stride”)) storing the array of encoded symbols of FIG. 4; Ex.1003, ¶103.

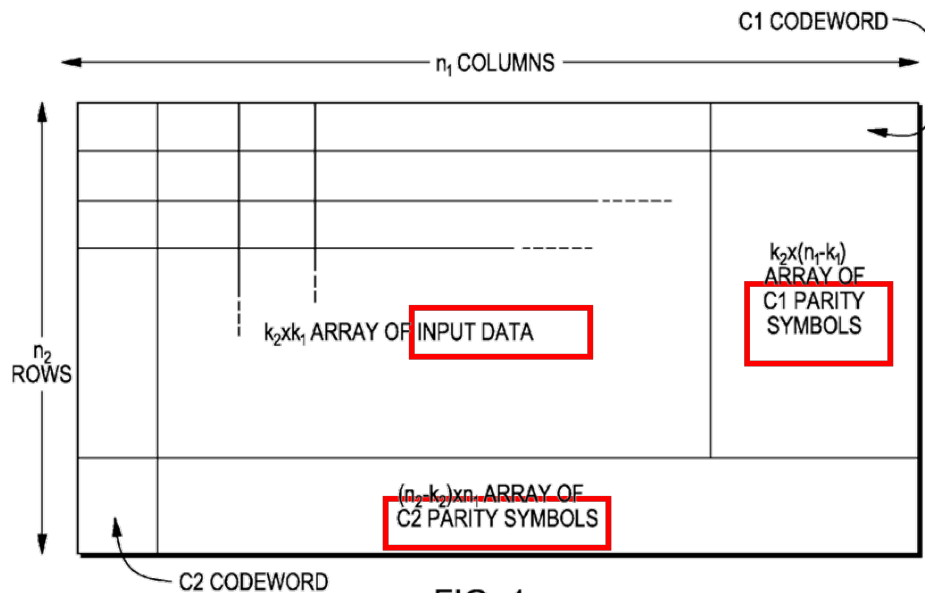


FIG. 4

Ex.1006, FIG. 4, annotated.

Referring to the logical representation of the product code scheme shown in FIG. 4, the encoded symbols of the product code include input data symbols (*data*

payload symbols), C1 parity symbols (*primary ECC parity symbols*), and C1 parity symbols (*secondary ECC parity symbols* generated during the C1 and C2 encoding processes. Ex.1006, 10:53-11:8. “The array of encoded symbols shown in the FIG. 4 is written **in a single stride**”. *Id.*, 11:9-11; Ex.1003, ¶¶35-40, 104.

In further detail, Cideciyan describes “systematic encoding” is used, “whereby each codeword is produced by addition of parity symbols to the unencoded data and the data itself remains unchanged by the encoding process.” Ex.1006, 10:54-56. Ex.1003, ¶105. Cideciyan describes in detail the C1 and C2 encoding process:

The input data to be encoded is partitioned by controller 10 into successive logical blocks each comprising a logical array of k_2 rows and k_1 columns of data symbols.... The EC1 encoder 13 systematically encodes each row in accordance with the C1 code This **C1 parity code** is added to the **k_1 input data symbols** to produce an **n_1 -symbol C1 codeword**. ... the resulting set of k_2 C1 codewords forms a logical array of k_2 by n_1 C1-encoded symbols.... The EC2 encoder 14 systematically encodes each column in accordance with the C2 code to generate **(n_2-k_2) parity symbols** for that column. This **C2 parity code** is added to the **k_2 column symbols** to produce an **n_2 -symbol C2 codeword**.

Ex.1006, 10:53-11:8; Ex.1003, ¶105.

Storage of the primary and secondary ECC parity symbols:

Referring to FIG. 5 below, Cideciyan further describes storing the array of encoded symbols of FIG. 4 in the single group of pages (“stride”). Ex.1006, 11:9-11. Specifically, Cideciyan describes that *the region* (a group of pages including N pages across N channels) storing a long product code codeword (with a codelength n_1n_2), which includes C1 and C2 codewords for component codes C1 and C2. It is noted that in FIG. 5, the C2 parity symbols are stored in the N^{th} page (corresponding to CH(N)) in this group (“stride”) as an example only, and “**any channel could be used for the parity.**” Ex.1006, 11:39-42); Ex.1003, ¶106.

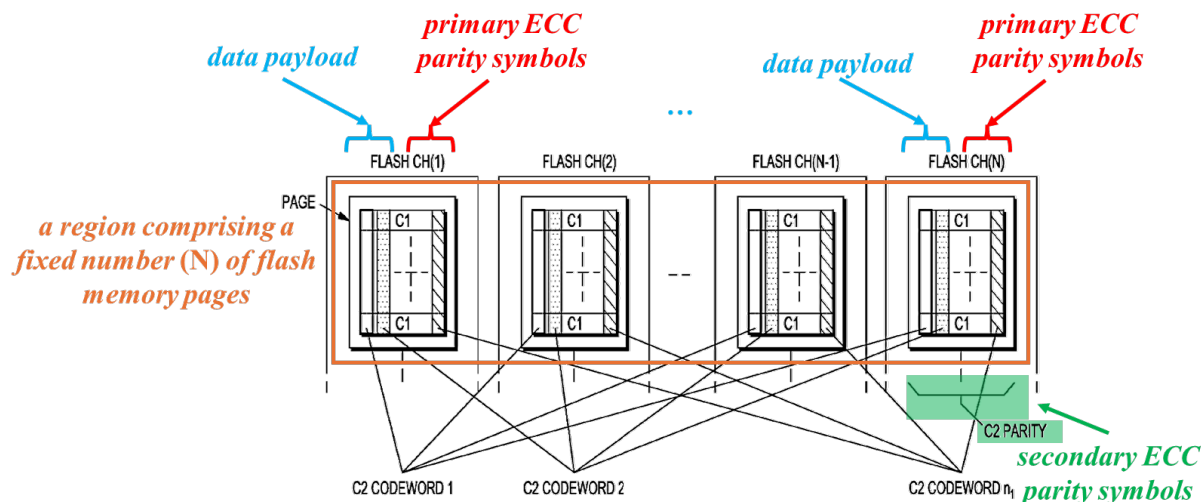


FIG. 5

Ex.1006, FIG. 5, annotated.

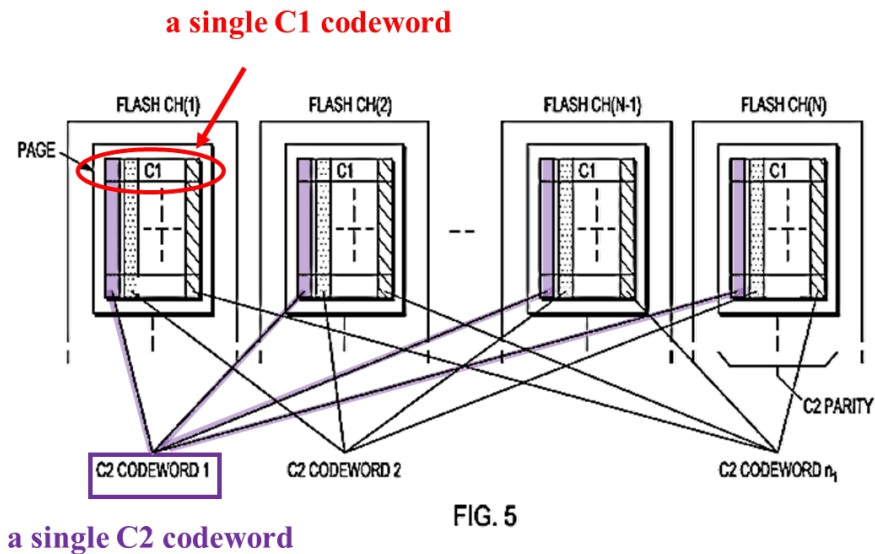
In further detail, referring to FIG. 5 above, Cideciyan describes:

The array of encoded symbols shown in the FIG. 4 is written in a **single**

stride....illustrated by the schematic diagram of FIG. 5....a stride contains one page in each of the N channels and each page stores a plurality of C1 codewords. The key features of the codeword arrangement here are that: (1) **each storage subdivision spanned by the stride contains a plurality of C1 codewords;** and (2) **each of these storage subdivisions contains a portion of each of the second codewords** in the stride. In the stride shown in FIG. 5, different C2 codewords are indicated by differently-shaded vertical bars. The first channel contains the first portion of each of the n_1 C2 codewords. The second channel contains the second portion of each of the C2 codewords, and so on. The last channel, channel N, contains the C2 parity codes for each C2 codeword **in this example**.

Ex.1006, 11:9-28; Ex.1003, ¶107.

In annotated FIG. 5 below, a single C1 (primary) codeword is illustrated in **red**, and a single C2 (secondary) codeword is annotated in **purple**.



Ex.1006, FIG. 5, annotated.

Like the '085 patent, Cideciyan's C1 (primary) codewords each are stored within a page and provides protection within a page, and C2 (secondary) codewords each are stored across the group of N pages across the N channels, providing protection within a page stripe, because the example stride of FIG. 5 is a page stripe. Ex.1003, ¶108; Ex.1001, Table 1 (describing "Primary" and "Secondary" and corresponding data protection).

TABLE I

Type of Error Correction	Data Protection
Primary	within a page or integer fraction of a page
Secondary	within a page stripe
Tertiary	within a page grid

Ex.1001, Table 1.

Additionally, like the '085 patent, Cideciyan's C1 (primary) and C2 (secondary) codewords disclosure provides **orthogonal** primary and secondary ECC codewords. Ex.1001, 3:20-25; Ex.1006, FIG. 4, 11:29-31 (“[t]he ‘**orthogonal**’ arrangement of C1 and C2 codewords...**gives exceptional error performance.**”); Ex.1003, ¶109.

Further like the '085 patent, Cideciyan also describes that various code types may be used for C1 code and C2 code. Ex.1006, 18:31-34; Ex.1003, ¶110.

Reasons to Combine Li and Cideciyan

It would have been obvious to a POSITA to combine Li and Cideciyan to provide the obvious, predictable, and beneficial results from implementing the product-code-type ECC scheme as taught by Cideciyan. Ex.1003, ¶111.

First, a POSITA would have been motivated to combine Li with Cideciyan's teaching of using a product code type ECC scheme that provides improved error recovery in Li, because Li describes adaptive ECC selection using “a set of ECC schemes with **various code types.**” Ex.1005, [0019]. Cideciyan teaches an obvious way of implementing Li's ECC schemes, by providing product-code-type-ECC schemes, which use the well-known “product code” code type described in textbooks. Ex.1020, 274-278. “Compared to plain long length codes,” a POSITA would have found that such implementation of applying a product-code-type ECC scheme to the region provides the benefits of “high performance from

cross parity check” with orthogonal arrangement of C1 and C2 codewords, provides “low circuit overhead” by using two shorter length component codes, “improved error recovery,” (Ex.1006, 3:54-4:56), and is a suitable option for Li’s implementation of ECC schemes with various code types. Ex.1018, Yang, 4; Ex.1006, FIG. 4, 11:29-31 (“the ‘orthogonal’ arrangement of C1 and C2 codewords ...gives exceptional error performance”); Ex.1003, ¶¶35-40, ¶112; *Intel Corp. v. Qualcomm Inc.*, 21 F.4th 784, 800 (Fed. Cir. 2021).

Second, implementing Li’s “portion” of multiple pages to span different channels for parallel, simultaneous access provides the additional benefits of “efficient, reliable and robust solid state storage systems with exceptional overall performance,” and “improved error protection offered by the coding process **both within storage subdivisions** and **across the storage as a whole**.” Ex.1006, 5:2-13; *see also* 7:30-41(describing its invention provides “improved error protection... achieved **without the performance penalties**, in terms of increased write amplification and reduced endurance”). Ex.1003, ¶113.

In other words, in the example of Li combined with Cideciyan, improved protection can be achieved across multiple flash memory subdivisions, e.g., using information on other dies to reconstruct information on a failed die, as taught by Cideciyan, which Li alone does not describe. Ex.1003, ¶114.

Third, as shown in FIG. 2A of Li and FIG. 2 of Cideciyan below, Li and

Cideciyan are similar flash-based SSD devices (each including a SSD controller and flash devices accessible in parallel) with similar objectives. Ex.1005, FIG. 2A, [0019], Ex.1006, FIG. 2, 5:12-13; Ex.1003, ¶115.

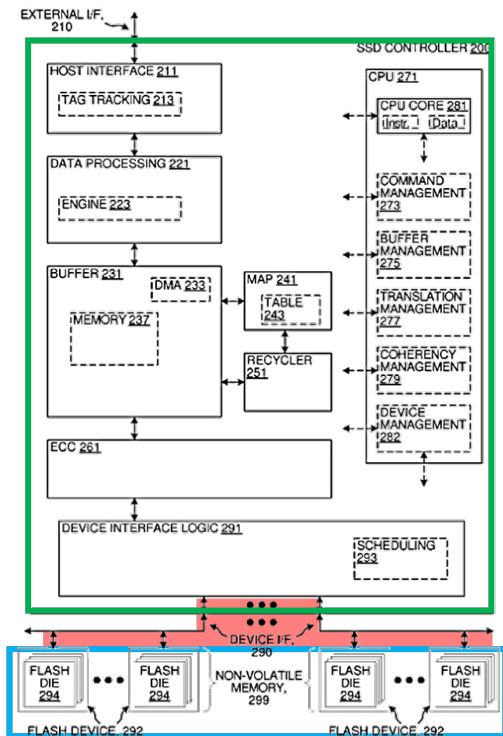


Fig. 2A

Ex.1005, Li, FIG. 2A, annotated

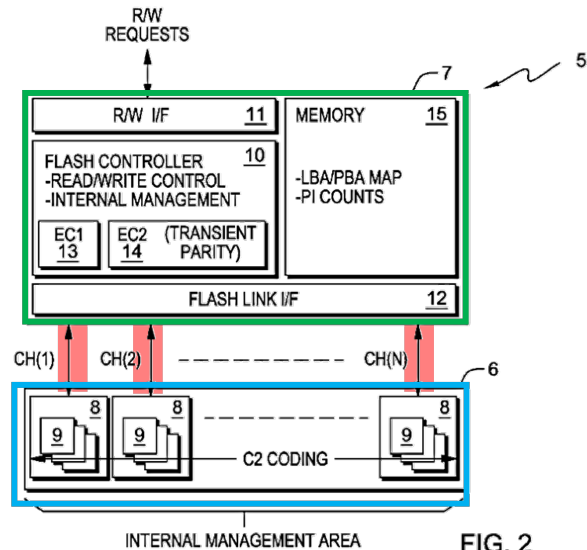


FIG. 2

Ex.1006, Cideciyan, FIG. 2,
annotated

A POSITA would have found it obvious for a known technique (applying a product-code-type ECC scheme to a group of pages across the channels) improves one device (the flash-based storage system of Cideciyan) to improve similar devices (Li's flash-based storage system) in the same way (with the benefits as taught by Cideciyan), using the prior art elements according to their established functions. Ex.1003, ¶116; *Intel Corp.*, 61 F.4th at 1380–81.

Fourth, the combination of Li and Cideciyan allows the pages of the memory to be operated in multi-level cell (MLC) mode for a longer period of time than Li alone, such that the switching from MLC to SLC is delayed or avoided. Li describes that “when a raw BER exceeds a threshold, a portion (such as a page) of flash memory previously operated as an MLC page is thereafter operated as an SLC [single level cell].” Ex.1005, [0103]. Switching from MLC to SLC is at expense of a big reduction of data payload – for example, a reduction around 50% by switching from a 2-bit-level MLC mode to an SLC mode. This switching should be avoided or delayed as much as possible. Ex.1003, ¶117.

By applying Cideciyan’s product code ECC scheme with “exceptional error performance,” (Ex.1006, FIG. 4, 11:29-31), the switch from MLC mode to SLC mode is delayed. *See also* Ex.1006, 3:54-4:4 (“MLC is inherently less reliable than single-level memory cells (SLC)... **Additional coding is therefore required to extend MLC NAND lifetime**”); Ex.1003, ¶117.

Additionally, the results of the proposed combination would have been predictable and a POSITA would have had a reasonable expectation of success, because the combination of Li and Cideciyan merely applies known technique to a known device (Li’s flash memory based storage system implementing adaptive ECC schemes) to yield the predictable results of “efficient, reliable and robust solid state storage systems with exceptional overall performance.” Ex.1006, 5:11-

13; Ex.1003, ¶118.

[1.2.1] comparing the determined bit error rate to one or more predetermined thresholds corresponding to a set of predefined gears comprising at least a first gear and a second gear, wherein the predefined gears correspond to different predefined ECC schemes,

Li, as combined above with Cideciyan, renders obvious [1.2.1]. Ex.1003,

¶¶119-135. A “gear” is:

a set of parameter(s) specifying an error correction coding [ECC] scheme, including one or more of the payload capacity, the type and parameters of each component code, the interleaving scheme, **and parameters** used for determining whether the gear is appropriate for a particular situation.

Ex.1001, 20:18-25

a. Li teaches “a set of predefined gears... the predefined gears correspond to different predefined ECC schemes”

First, Li teaches “a set of predefined gears,” and each predefined gear corresponds to a predefined ECC scheme. Specifically, Li describes multiple sets of parameters (*predefined gears*):

The techniques include a set of ECC schemes with **various code types, code rates, and/or various code lengths (providing different error correcting capabilities)**, and error statistic collecting/tracking (such as via a dedicated hardware logic block).

Ex.1005, [0019]; Ex.1003, ¶121. Each set of parameters in Li (including but not

limited to “code type,” “code rate,” “code length”) is a “*gear*” specifying an ECC scheme providing a corresponding error correcting capability (ECC capability/strength). *See* similarly 085 patent equating a “*gear*” with “ECC strength.” Ex.1001, 51:1-2; *see also* Li’s teaching of BER thresholds at [1.2.1].b below. Li’s above multiple sets of parameters are *predefined*, and specify *predefined ECC schemes* with corresponding ECC strengths. Ex.1003, ¶122, Ex.1005, [0091].

In Li combined with Cideciyan, as discussed at [1.1.2], it would be obvious for a predefined set of parameters specifying an ECC scheme to include a product-code-type ECC scheme as taught by Cideciyan. Each product-code-type ECC scheme is specified with a set of parameters specifying a code type of “product code,” component code C1 (with corresponding parameters), component code C2 (with corresponding parameters), a code rate, and a code length. Ex.1003, ¶¶35-40; *see also* Ex.1020, xv, 274-278; Ex.1019, 4. The set of parameters specifying a product-code-type ECC scheme, together with parameters (including the corresponding BER thresholds) used for determining whether this set of parameters (with the specified ECC scheme) is appropriate for a particular situation (as discussed below at [1.2.1.b]), is a “*gear*” corresponding to that ECC scheme. Ex.1003, ¶124.

Dr. Dolecek has provided a table including examples “*gears*” in “*a set of*

predefined gears” in Li combined with Cideciyan. Ex.1003, ¶125; Ex.1005, Abstract, [0019], [0091], [0035], [0036]; Ex.1006, Cideciyan, 18:31-34, FIG. 11, 16:34-60, FIG. 12, 16:61-17:12; Ex.1009, 5; Ex.1003, ¶¶32-34 and 35-40.

<i>parameters specifying an ECC scheme</i>		<i>ECC scheme</i>	<i>parameters used for determining whether the gear is appropriate</i>
Set 1 (Example Gear 1)	code type: Product Code with c1:BCH, c2: Hamming code; code-rate-1; code-length-1; component-code-c1: c1-code-rate-1, c1-code-length-1; component-code-c2: c2- code-rate-1; c2-code-length-1; etc.	ECC scheme 1	BER-threshold- x1 < Determined BER < BER- threshold-y1
Set 2 (Example Gear 2)	code type: Product Code with c1:BCH, c2:RS; code-rate-2; code-length-2; component-code-c1; c1-code-rate-2; c1-code-length-2; component-code-c2: c2-code-rate-2; c2-code-length-2; etc.	ECC scheme 2	BER-threshold- x2 < Determined BER < BER- threshold-y2

Ex.1003, Table 1.

As shown in Table 1, in Li combined with Cideciyan, *set of predefined gears comprising at least a first gear and a second gear* includes at least “*a first gear*” and “*a second gear*”, each corresponding to a predefined set of parameters specifying an ECC scheme and parameters for determining whether the gear is

appropriate for a particular situation. Ex.1003, ¶¶126-128; *see also* Ex.1005, Claims 1-2.

b. “*comparing the determined bit error rate to one or more predetermined thresholds corresponding to a set of predefined gears*”

Li teaches the “*comparing*” as claimed, because Li teaches comparing the determined raw BER to “*one or more predetermined thresholds*” (including “a first threshold,” “a second threshold,” and “a third threshold”), where the comparison results are used to select a gear having a corresponding ECC scheme. Ex.1005, [0105]. The one or more thresholds are parameters used for determining whether the gear is appropriate for a particular situation, and are included in the “*gear*” for that corresponding ECC scheme, as construed. Ex.1003, ¶129.

Namely, as an example, Li teaches

while a raw BER of a page of a flash memory is less than **a first threshold**, the page is operated as an MLC page and encoded with **a first short code length ECC**. If/when the raw BER **exceeds the first threshold (but remains less than a second threshold)**, then the page is encoded with **a first longer code length ECC** (while still operated as an MLC page).

If/when the raw BER **exceeds the second threshold (but remains less than a third threshold)**, then the page is encoded with **an even longer code length ECC**.

Ex.1005, [0105]; Ex.1003, ¶130.

A POSITA would have understood that “a first short code length ECC,” “a first longer code length ECC,” “an even longer code length ECC” corresponding to different ECC schemes, each ECC scheme is specified by a set of parameters of a “*gear*,” including parameters of code type and code rate, in addition to the parameter of the code length. Ex.1003, ¶131.

Further, a POSITA would have understood that Li’s teaching of comparing and then selecting a gear corresponding an ECC scheme based on the comparison as described in [0105] applies to gears for various ECC schemes having various code types, code rates, and code lengths, and is not limited to only “a first short code length ECC,” “a first longer code length ECC,” “an even longer code length ECC” as described in [0105] of Li. Ex.1003, ¶132.

c. Li’s thresholds relate to “gears”.

Li’s thresholds relate to the “gears” corresponding to ECC schemes. The thresholds used for determining whether the particular ECC scheme with corresponding correction capability is appropriate for a particular situation, together with the parameters (including e.g., code type, code rate, code length) for specifying the particular ECC scheme, collectively define a “*gear*”. Ex.1003, ¶133.

In other words, Li’s set of ECC schemes (“*different predefined ECC schemes*”) providing different error correcting capabilities correspond to “*a set of*

predefined gears” (each gear includes a plurality of parameters specifying an ECC scheme and parameters including the one or more thresholds to determine whether the gear is appropriate for the situation). Ex.1005, [0019]. Ex.1003, ¶134.

According, as discussed at [1.2.1].a-c, Li as combined with Cideciyan renders obvious [1.2.1]. Ex.1003, ¶135.

[1.2.2] wherein the first gear has a different data payload size and correction capability than the second gear, wherein the amount of memory space allocated for the storage of data payload within the region varies between the first gear and the second gear to accommodate a varying number of parity symbols between the first gear and the second gear, and

Li, as combined above with Cideciyan, renders obvious [1.2.2]. Ex.1003, ¶¶136-144.

First, as discussed at [1.2.1].a and shown in Table 1 provided by Dr. Dolecek below, Li as combined with Cideciyan renders obvious “**first gear**” and “**second gear**,” which are selected based on different raw BERs, and as such, have different correction capabilities corresponding to the different raw BERs. Ex.1003, ¶¶125, 137.

<i>parameters specifying an ECC scheme</i>		<i>ECC scheme</i>	<i>parameters used for determining whether the gear is appropriate</i>
Set 1 (Example Gear 1)	code type: Product Code with c1:BCH, c2: Hamming code; code-rate-1; code-length-1;	ECC scheme 1	BER-threshold- x1 < Determined BER < BER- threshold-y1

	component-code-c1: c1-code-rate-1, c1-code-length-1; component-code-c2: c2-code-rate-1; c2-code-length-1; etc.		
Set 2 (Example Gear 2)	code type: Product Code with c1:BCH, c2:RS; code-rate-2; code-length-2; component-code-c1; c1-code-rate-2; c1-code-length-2; component-code-c2: c2-code-rate-2; c2-code-length-2; etc.	ECC scheme 2	BER-threshold- x2 < Determined BER < BER- threshold-y2

Ex.1003, Table 1

Second, Li describes different gears having different data payload sizes and correction capabilities, and different number of parity symbols. Ex.1003, ¶138. As an example, Li describes:

2. The system of claim 1, wherein encoding according to a **first** one of the error correcting codes results in a number of **error correcting bits** to **store in the portion** that is **less than** when encoding according to a **second** one of the error correcting codes.

Ex.1005, claim 2; Ex.1003, ¶138. A POSITA would have understood that Li's teaching of gears as described in claims 1-4 applies to various gears corresponding to various ECC schemes having various code types, code rates, and code lengths, including Example Gear 1 and Example Gear 2 in Ex.1003, Table 1.

Li as combined with Cideciyan teaches “*a varying number of parity symbols between the first gear and the second gear,*” because Li describes a number of error correcting bits (“*number of parity symbols*”) according to a first gear corresponding to a first ECC scheme (e.g., specified by a set of parameters including a first ECC code of Example Gear 1 and other parameters) is less than (“*varies*”) when encoding according to a second gear corresponding to a second ECC scheme (e.g., specified by a set of parameters including a second ECC code of Example Gear 2 and other parameters). Ex.1003, ¶139.

Third, Li as combined with Cideciyan renders obvious that “the amount of memory space allocated for the storage of data payload within the region varies between the first gear and the second gear to accommodate a varying number of parity symbols” as claimed. Specifically, Li provides,

3. The system of claim 2, wherein when encoding according to the first error correcting code, a number of bits of the portion used as user data is increased by up to a difference between the number of error correcting bits used by the second error correcting code minus the number of error correcting bits used by the first error correcting code.

4. The system of claim 2, wherein when encoding according to the second error correcting code, a number of bits of the portion used as user data is decreased by up to a difference between the number of

error correcting bits used by the second error correcting code minus the number of error correcting bits used by the first error correcting code.

Ex.1005, claims 3-4; Ex.1003, ¶140.

As described in claims 3 and 4 of Li, in the combination of Li and Cideciyan, “*the first gear has a different data payload size,*” because a number of bits of the portion used as user data (“*data payload size*”) of the first ECC scheme (corresponding to “*first gear*”, e.g., Example Gear 1) is greater than number of bits of the portion used as user data of the second ECC scheme (“*second gear*”, e.g., Example Gear 2). Ex.1003, ¶141.

Further, as described in claims 3 and 4 of Li, Li teaches that in the “portion” of the flash memory, the number of bits of the portion used as user data between the first ECC scheme and the second ECC scheme changes (“increased by up to a difference” or “decreased by up to a difference”) based on the difference of error correcting bits in the region. As such, Li as combined with Cideciyan renders obvious “*the amount of memory space allocated for the storage of data payload within the region varies between the first gear and the second gear to accommodate a varying number of parity symbols.*” Ex.1003, ¶142.

In other words, Li as combined with Cideciyan applies adaptive ECC scheme selection to a portion (a group of pages across the channel) of the flash memory, and the portion has a fixed size. See discussion at Ground 1:[1.1.1]. In

this portion with a fixed size, when dynamically switching between gears (corresponding to different ECC schemes with respective ECC strength), the number of parity symbols (error correcting bits) varies (e.g., increases/decreases) between the gears, and the storage for the data payload changes (decreases/increases according to the difference of number of parity symbols) to accommodate the varying number parity symbols. Ex.1003, ¶143.

This is also consistent with the well-known knowledge in the art. *See e.g.*, Ex.1008, Table 1 (illustrating that gear 2 reduces data payload size to accommodate increased parity size). Ex.1003, ¶144.

[1.3] *selecting a gear from the set for the region based at least partly on the comparisons to the one or more predetermined thresholds;*

Li, as combined above with Cideciyan, renders obvious [1.3]. Ex.1003, ¶¶145-148, VIII.A.

First, as discussed at [1.2.1].b, Li as combined with Cideciyan describes *comparing the determined bit error rate to one or more predetermined thresholds* for adaptive ECC scheme selection. Ex.1003, ¶146.

Second, as discussed at [1.2.1].b, Li as combined with Cideciyan describes that SSD controller 200 (e.g., implementing an adaptive encoder hardware block) is used for *selecting a gear from the set* (selecting from the set of predefined gears as discussed at [1.2.1]) *for the region* (the group including N pages distributed

across N channels) *based at least partly on the comparisons to the one or more predetermined thresholds*. Ex.1005, [0105], [0023] (“**an adaptive encoder hardware block** ... enabled to dynamically **determine** the selected error correcting code **based at least in part on the raw BER.**”); [0105]; *see also* Ex.1005, Claim 1 (“**dynamically determine** the dynamically selected error correcting code based at least in part on the raw BER”); Ex.1003, ¶147.

It is noted that a POSITA would have understood that Li’s teaching of comparing and then selecting an ECC scheme based on the comparison as described in [0105] applies to various ECC schemes having various code types, code rates, and code lengths, and is not limited to only “a first short code length ECC,” “a first longer code length ECC,” “an even longer code length ECC” as described in [0105] of Li. Ex.1003, ¶148.

[1.4] *wherein determining, comparing, and selecting are performed by an integrated circuit,*

Li, as combined above with Cideciyan, renders obvious [1.4]. Ex.1003, ¶¶149-151.

As discussed at [1.1.1], [1.2.1], and [1.3], Li as combined with Cideciyan describes an SSD controller for performing *determining, comparing, and selecting*. Li further describes that the SSD controller is implemented on an integrated circuit. Ex.1005, [0125], [0014] (“all [] of an SSD controller (or a computing-host flash

memory controller) are implemented **on a single IC [Integrated Circuit]**, a single die of a multi-die IC.” *See also* Ex.1007, Frost, 5:4-7 (“The FLASH controller 10 may take many forms. In the example of FIG. 1, the FLASH controller 10 is **a field programmable gate array (FPGA)** that, during start-up of the system is programmed and configured by the CPU 15.”). It was well-known that an FPGA is an integrated circuit; Ex.1003, ¶150 (explaining that it was well-known that an FPGA is an integrated circuit).

This is consistent with the description of the ’085 patent. Ex.1001, 56:56-59 (“the SSD controller 1300 is implemented in an integrated circuit, such as by an application specific integrated circuit (ASIC), a field programmable gate array (FPGA).”); Ex.1003, ¶151.

2. Claim 2

[2.0] *The method of claim 1, further comprising repeatedly determining the bit error rate and comparing the determined bit error rate to one or more predetermined thresholds such that the bit error rate is actively monitored.*

Li combined with Cideciyan renders obvious [2.0]. Ex.1003, ¶¶152-153.

Specifically, Li teaches repeatedly determining (periodically determining) the bit error rate and comparing it to the thresholds based on the number of program/erase cycles, *such that the bit error rate is actively monitored* (for adaptively selecting the suitable ECC scheme throughout the lifetime of the flash memory). Ex.1005, [0092] (“the Error Statistics Collecting/Tracking hardware

block is enabled to **periodically** determine (such as **once every 100 PE cycles**) a current raw (measured) BER of one or more portions of flash storage.”), [0048] (“wherein the measured raw BER is determined periodically”), [0019] (“**dynamically switching** encoding/decoding of all or any portions of the flash memory amongst a respective one or more of the ECC schemes based at least in part on information from the error statistic collecting/tracking”); Ex.1003, ¶153.

3. Claim 4

[4.0] *The method of claim 1, wherein determining, comparing, and selecting are performed in connection with at least one of flash initialization, read, or write operations for an end user.*

Li combined with Cideciyan renders obvious [4.0]. Ex.1003, ¶¶154-158.

As illustrated in FIGS. 1 and 2A below, Li describes determining, comparing, and *selecting* (as discussed at [1.1.1], [1.2.1], and [1.3]) *are performed in connection with* (based on parameters related to) *at least one of flash initialization, read* (a number of errors corrected, a number of errors detected, a number of read cycles, a read time), *or write operations* (a number of program/erase cycles, a program time, an erase time) *for an end user* (including read and write operations for an end user received through host interface 211). Ex.1005, [0114] (“**Host interface 211 sends and receives commands and/or data via external interface 210....** the commands include a **read command** specifying an address (such as an LBA) and an amount of data (such as a number

of LBA quanta, e.g. sectors) to read; **in response the SSD provides read status and/or read data**. For another example, the commands include a write command specifying an address (such as an LBA) and an amount of data (such as a number of LBA quanta, e.g. sectors) to write; in response the SSD provides write status and/or requests write data and optionally subsequently provides write status.”). Ex.1003, ¶155.

As further detail, Li describes a write-storage-data path and a read-storage-data path. Ex.1005, [0088] (“A **write-storage-data path 110** includes various hardware blocks:....A **read-storage-data path 150** includes various hardware blocks:... An **Error Statistics Collecting/Tracking 180** hardware block is coupled to the Universal Encoder, the Code Library, the Universal Decoder, and the Control/Interface hardware blocks”). Ex.1003, ¶156.

A POSITA would have understood that in Li, the parameters “a number of errors corrected, a number of errors detected,...a number of read cycles..., a read time” are related to (*in connection with*) *read operations for an end user*, and parameters “a number of program/erase cycles,...a program time, an erase time” are related to (*in connection*) *with write operations for an end user*. Ex.1005, Claim 16. Ex.1003, ¶157.

determining, comparing, and selecting are performed in connection with at least one of flash initialization, read, or write operations for an end user

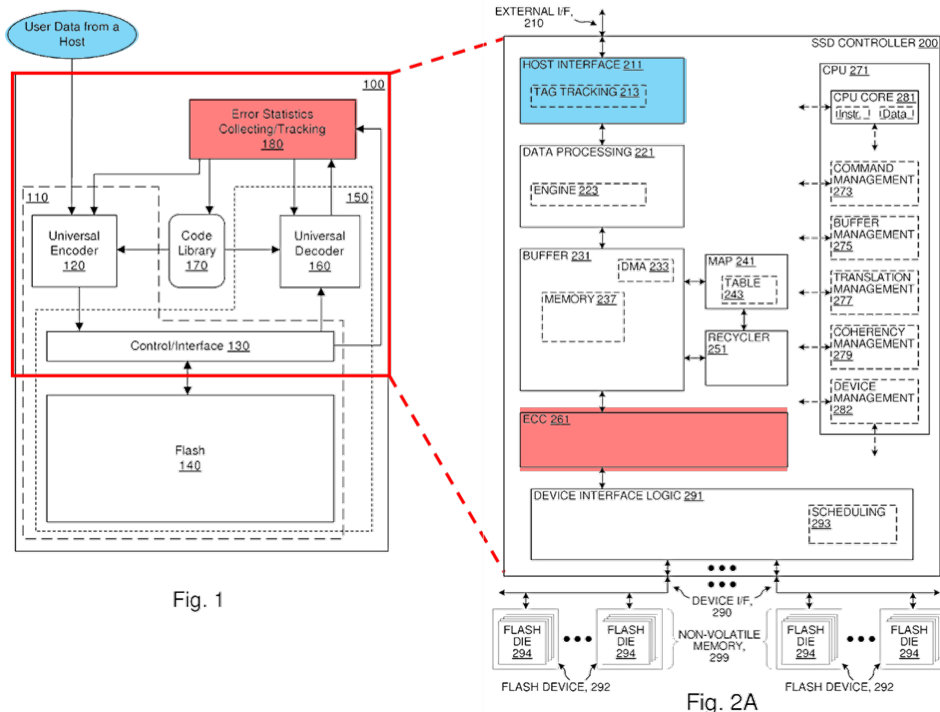


Fig. 1

Fig. 2A

Ex.1005, FIGS. 1 and 2A, annotated.

Li provides additional description on how the determining, comparing, and selecting are performed in connection with at least one of flash initialization, read, or write operations **for an end user**, because they are performed based on parameters related to read/write operations commanded by the host for an end user. Ex.1005, FIG. 1 (illustrating “**user data** from a host,” which is from an end user), [0089] (describing write operation from an end user, where ““User Data **from a Host**’ to write as storage data is received by the Universal Encoder and encoded according to an error correcting code”); [0090] (describing read operations from an

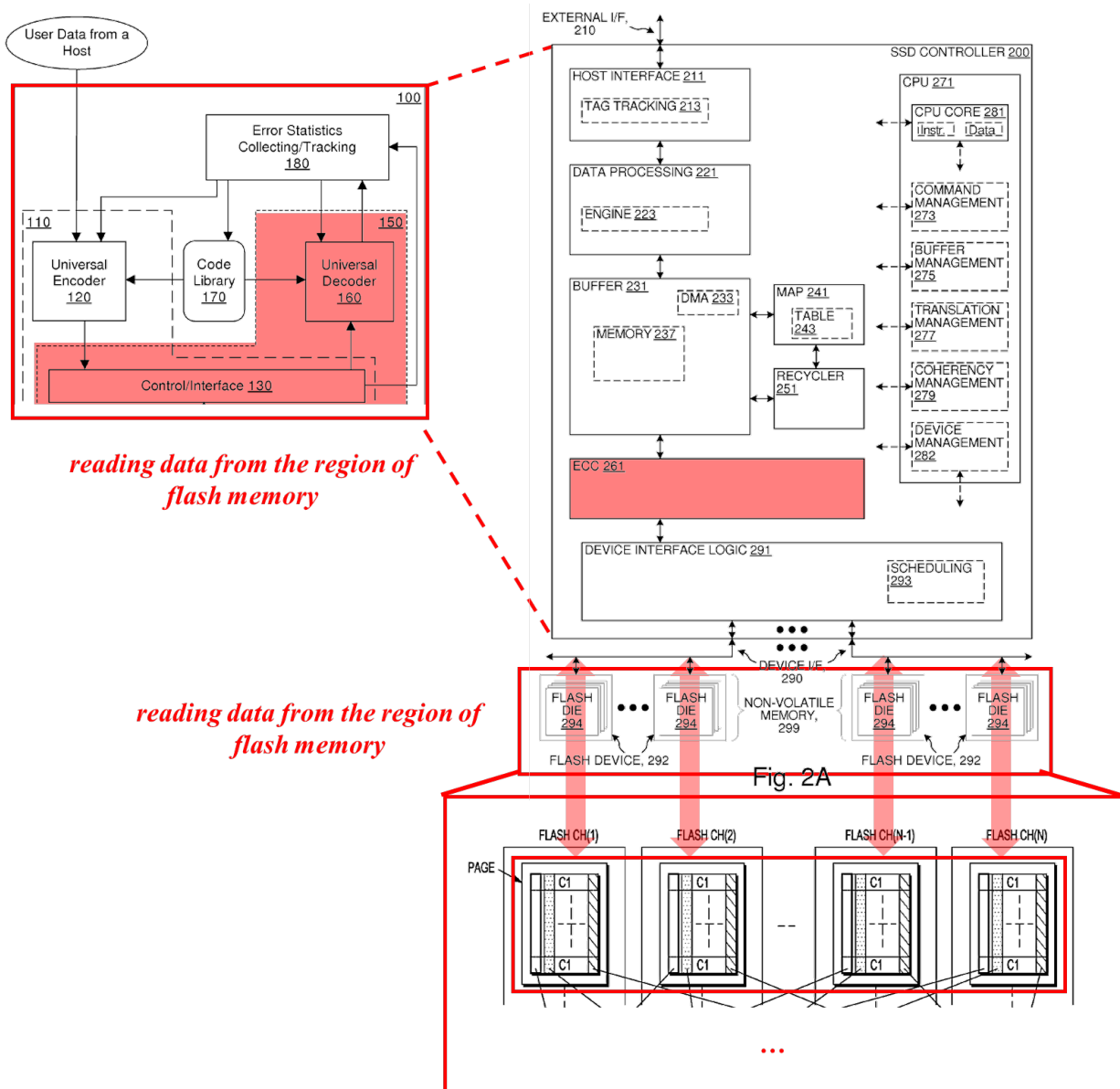
end user, where “[r]eading storage data begins by Control/Interface reading raw information,” and the decoded “data information is then **passed to the Host.**”); Ex.1005, [0023] (“**an adaptive encoder hardware block** enabled to encode according to a selected one of a plurality of error correcting codes, and further enabled to dynamically **determine the selected error correcting code** based at least in part on the raw BER.”). Ex.1003, ¶158.

4. Claim 5

[5.0] *The method of claim 1, further comprising reading data from the region of flash memory,*

Li combined with Cideciyan renders obvious [5.0]. Ex.1003, ¶¶159-160.

As shown in Example 1.1 of Li combined with Cideciyan below, Li as combined with Cideciyan describes “*reading data*” (e.g., using SSD controller 200 including elements implementing the read-storage-data path) “*from the region of flash memory*” (from a portion/a group of pages across channels of the flash-based SSD). Ex.1005, [0050] (“observing a BER associated with at least some **reads of the portion**”); [0088]. Ex.1003, ¶160.



Example 1.1 of Li Combined with Cideciyan.

[5.1] wherein determining the bit error rate further comprises estimating the bit error rate based at least partly on a number of errors corrected by ECC in connection with reading the data from the region.

Li combined with Cideciyan renders obvious [5.1]. Ex.1003, ¶¶161-162.

Li as combined with Cideciyan describes determining the bit error rate

*further comprises includes estimating (using SSD controller 200 including elements implementing Error Statistics Collecting/Tracking 180) the bit error rate based at least partly on a number of errors corrected by ECC (by comparing raw read data from the flash memory with an error-corrected version of the raw read data) in connection with reading the data from the region (associated with the reads of the portion). Ex.1005, [0050] (“the measured raw BER is determined at least in part by observing a BER associated with at least some **reads of the portion**”); [0051] (“measured raw BER is determined at least in part by comparing raw read data from the flash memory with an error-corrected version of the raw read data.”); see also Ex.1005, [0079] (describing that an ECC scheme and/or its parameter base on “a number of errors corrected.”). Ex.1003, ¶162.*

5. Claim 14

[14.0] *The method of claim 1, wherein the gear includes a lower-latency ECC decode option and one or more higher-latency ECC decode options,*

[14.1] *wherein the gear is selected such that performing ECC decoding using the lower-latency ECC decode option alone is sufficient to provide corrected data at least 90% of the time.*

Li combined with Cideciyan renders obvious [14.0]-[14.1]. Ex.1003, ¶¶163-170.

Li describes “decoding in accordance with one or more of the ECC schemes.” Ex.1005, Abstract. It would be obvious to implement in Li, well-known

examples of providing decode options with different latencies for a single ECC scheme, as taught by Cideciyan. Ex.1003, ¶164.

First, Cideciyan provides *a lower-latency ECC decode option* (C1 decode option) *and one or more higher-latency ECC decode option* (the decode option of using both C1 and C2 decoding), where the decode option of using both C1 and C2 decoding inherently has a higher-latency because the system first incurs the latency of C1 decoding. Ex.1006, 14:29-33 (the failed “**C1 erasures can then be corrected** by the C2 decoder”); 14:41-50 (“Whenever a strip contains page or block errors that **cannot be corrected by the C1 error correction process**,” it is reconstructed “via the **C2 decoding** algorithm.”). Ex.1003, ¶165.

Second and alternatively, Cideciyan describes C1 decode option and C2 decode option having different latencies. Ex.1006, 18:21-25, 14:29-33, 14:41-50. Ex.1003, ¶¶166-167 (Dr. Dolecek explaining that it was well-known to provide different coding options having different latencies, citing Ex.1010, 14:1- 5; Ex.1011, 11:5-8); Ex.1003, ¶168.

Third, [14.1] is prima facie obvious, because the specification of the ’085 patent does not show any unexpected result or other unexpectedly improved properties. *See* Ex.1001, 23: 40-46; Ex.1003, ¶169. A long-held rule is that a percentage amount is prima facie obvious without a showing of some unexpected result or other unexpectedly improved properties. *See In re Dillon*, 919 F.2d 688,

692 (Fed. Cir. 1990) (en banc); *see also*, *In re Aller*, 220 F.2d 454, 456 (CCPA 1955); *In re Scherl*, 156 F.2d 72, 74-75 (CCPA 1946).

Regardless, a POSITA would have found it obvious to select a gear based on the determined BER, such that *performing ECC decoding using the lower-latency ECC decode option alone is sufficient to provide corrected data at least 90% of the time*. Ex.1006, FIG. 11, 4:1-4; 16:25-40; Ex.1009, JESD218, 11; Ex.1003, ¶170.

6. Claim 15

[15.0] *An apparatus comprising:*

To the extent that the preamble is limiting, Li as combined with Cideciyan renders obvious [15.0]. Specifically, Li describes that its invention “may be implemented in ...an **apparatus**.” Ex.1005, [0007]. Further, as discussed below at [15.1.1] and [15.3], Li describes that the apparatus includes the claimed first circuit and second circuit. Ex.1003, ¶¶171-172.

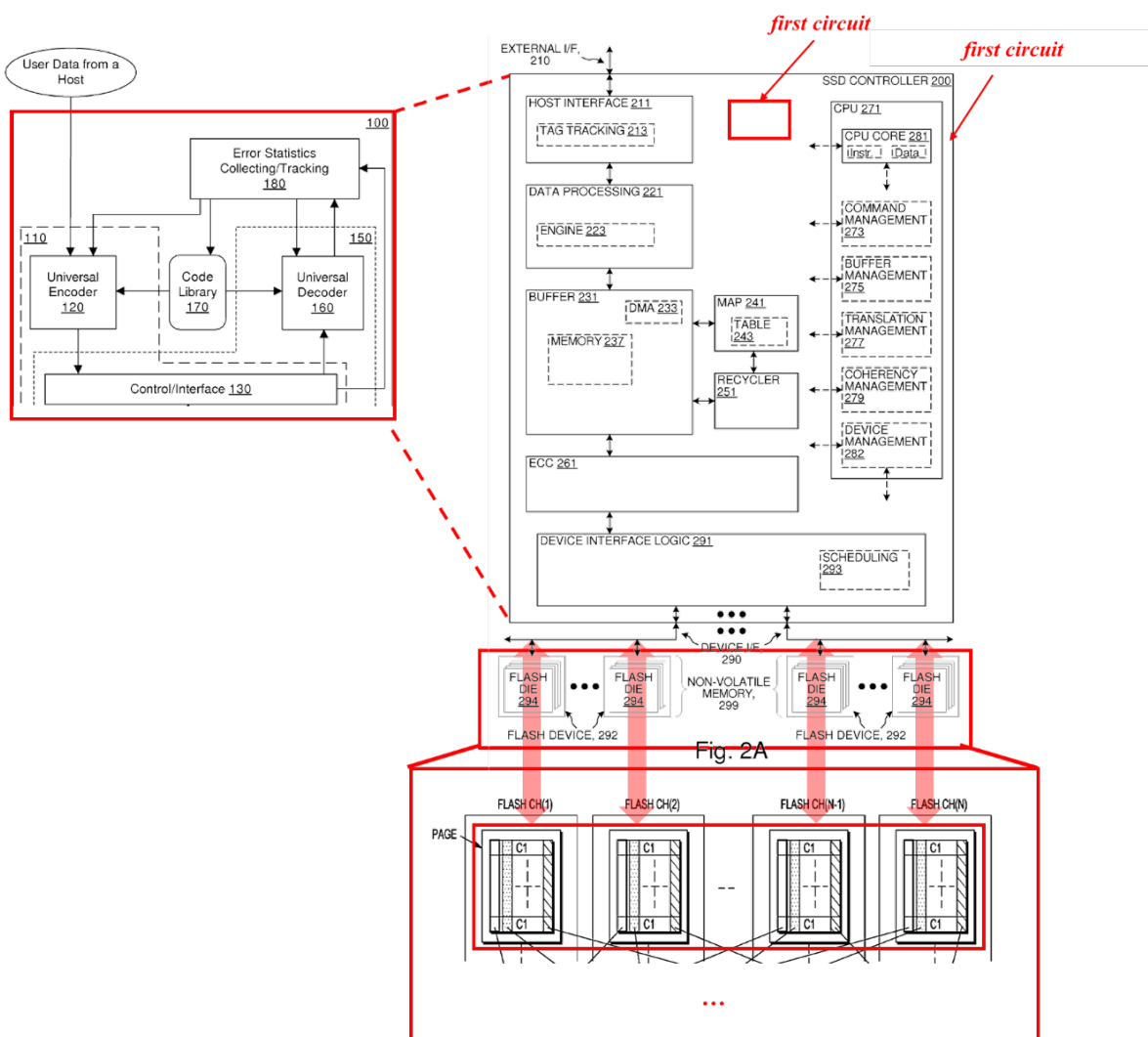
[15.1.1] *a first circuit configured to determine a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof, wherein the two or more flash memory pages of a region can be read simultaneously,*

Li combined with Cideciyan renders obvious [15.1.1]. Ex.1003, Ex.1003, ¶¶173-174.

As discussed at [1.2.1] and shown in Example 1.1, the combination of Li and Cideciyan renders obvious *a first circuit* (e.g., using SSD controller 200 and/or

elements thereof implementing the read-storage-data path, including the Code Library, Error Statistics Collecting/Tracking 180, Universal Decoder 160) configured to determine a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof.

Ex.1005, [0126]; Ex.1003, ¶174.



Example 1.1 of Combination of Li and Cideciyan

[15.1.2] wherein the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region and

Li combined with Cideciyan renders obvious [15.1.2] for reasons discussed at [1.1.2]. Ex.1003, ¶175.

[15.2.1] [the first circuit] configured to compare the determined bit error rate to one or more predetermined thresholds corresponding to a set of predefined gears comprising at least a first gear and a second gear, wherein the predefined gears correspond to different error correction coding (ECC) schemes,

Li combined with Cideciyan renders obvious [15.2.1], because *the first circuit* (SSD controller 200 including elements implementing Error Statistics Collecting/Tracking 180, e.g., using ECC 261 or any other suitable elements of SSD controller 200 as well-known in the art) performs the comparing for reasons discussed at [1.2.1] and [1.2.2]. Ex.1003, ¶176.

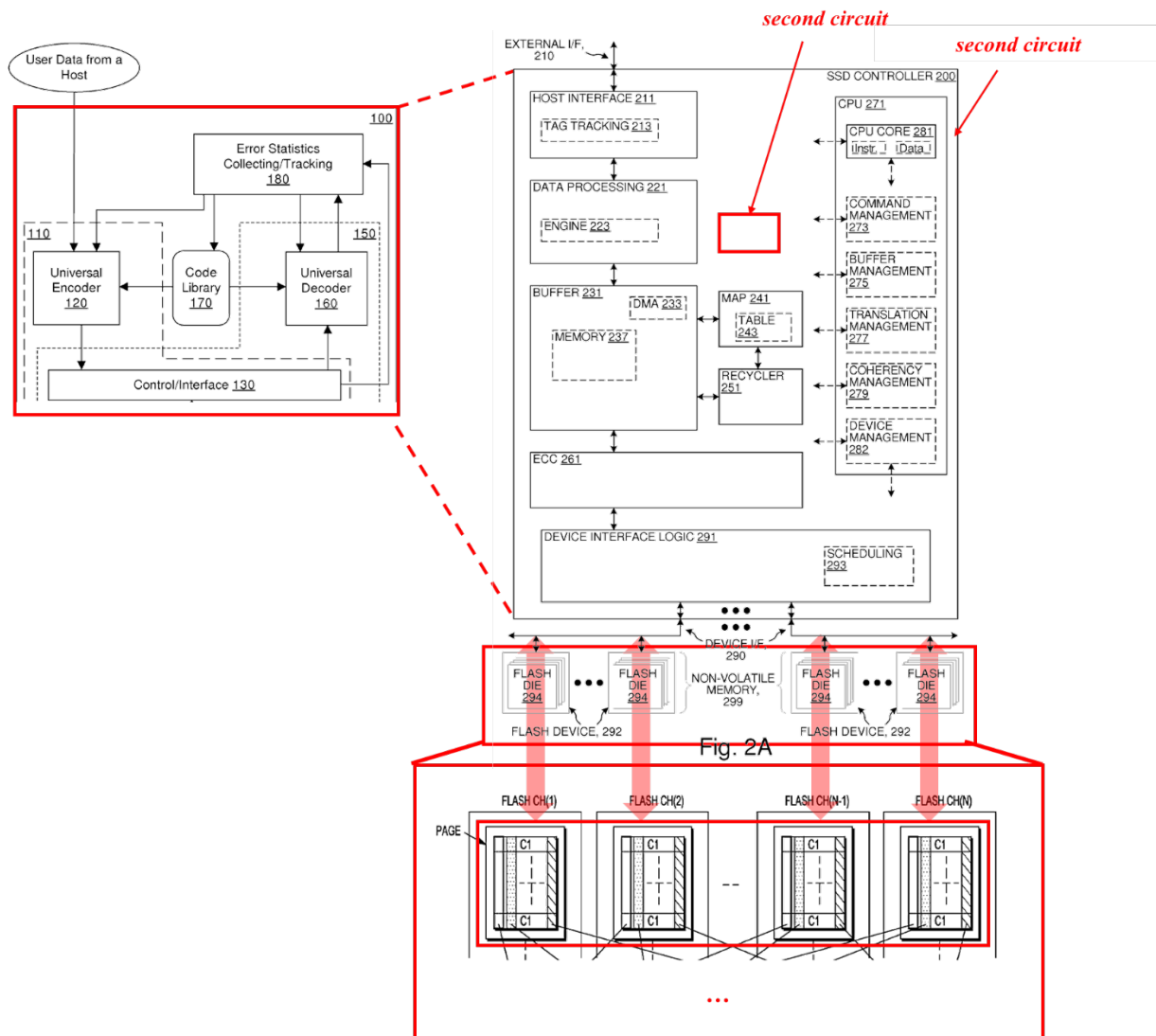
[15.2.2] wherein the first gear has a different data payload size and correction capability than the second gear, wherein the amount of memory space allocated for the storage of data payload within the region varies between the first gear and the second gear to accommodate a varying number of parity symbols between the first gear and the second gear; and

Li combined with Cideciyan renders obvious [15.2.2] for the reasons discussed at [1.2.2]. Ex.1003, ¶177.

[15.3] a second circuit configured to select a gear from the set for the region based at least partly on the comparisons to the one or more predetermined thresholds.

Li combined with Cideciyan renders obvious [15.3]. Ex.1003, ¶178.

As discussed at [1.3] and shown in Example 1.1 of the combination below, Li combined with Cideciyan renders obvious *a second circuit* (SSD controller 200 or elements thereof implementing adaptive encoder hardware block) *configured to select a gear from the set for the region based at least partly on the comparisons to the one or more predetermined thresholds*. Ex.1005, [0023] (“**an adaptive encoder hardware block** enabled to encode according to a selected one of a plurality of error correcting codes, and further enabled to dynamically **determine the selected error correcting code** based at least in part on the raw BER.”); [0126]; Ex.1003, ¶¶180. The “selected error correcting code” corresponds to an ECC scheme.



Example 1.1 of Combination Li and Cideciyan.

7. Claim 16

[16.0] *The apparatus of claim 15, wherein the ECC encoder is configured to determine the bit error rate and*

Li combined with Cideciyan renders obvious [16.0]. Ex.1003, ¶¶181-182.

Specifically, Li describes that *the ECC encoder (ECC 261) is configured to determine the bit error rate*. Ex.1005, [0126] (“ECC 261 implements one or more

functions performed by the **Error Statistics Collecting/Tracking, Universal Encoder**); Ex.1003, ¶182.

[16.1] *the processor is configured to compare and to select in connection with at least one of flash initialization, read, or write operations for an end user.*

Li combined with Cideciyan renders obvious [16.1]. Ex.1003, ¶¶183-185.

As discussed at [4.0], Li combined with Cideciyan renders obvious performing comparing and selecting *in connection with at least one of flash initialization, read, or write operations for an end user.* Ex.1005, [0114], Claim 16, FIGS. 1, 2A; Ex.1003, ¶184.

Li combined with Cideciyan renders that the “**processor**” is configured to perform selected functions including the comparing and selecting as a routine implementation. Ex.1005, [0137] (“It is also understood that **many functional aspects** of embodiments **are realizable selectively** [using] **processor**”); [0131]; *see also* Ex.1006, 9:29-31; Ex.1003, ¶185.

8. Claim 17

[17.0] *The apparatus of claim 15, wherein the first circuit is further configured to read data from the region of flash memory,*

Li combined with Cideciyan renders obvious [17.0]. Ex.1003, ¶¶186-187.

Specifically, in Li combined with Cideciyan, the first circuit (e.g., using SSD controller 200 including elements implementing the read-storage-data path) *is further configured to read data from the region of flash memory* for the reasons

discussed at [5.0]. Ex.1003, ¶187.

[17.1] wherein the ECC decoder is further configured to determine the bit error rate based at least partly on a number of errors corrected by ECC in connection with a read of data from the region.

Li combined with Cideciyan renders obvious [17.1]. Ex.1003, ¶¶188-189.

As discussed at [5.1], in Li combined with Cideciyan, *the ECC decoder* (e.g., using SSD controller 200 including elements implementing the read-storage-data path) is further configured to determine the raw BER “*based at least partly on a number of errors corrected by ECC*” (by comparison between the raw BER and the error-corrected version of the raw read data) *in connection with a read of data from the region* (associated with the reads of the portion). Ex.1005, [0050] (“the measured raw BER is determined at least in part by observing a BER associated with at least some **reads of the portion**”); [0051] (“measured raw BER is determined at least in part by comparing raw read data from the flash memory with an error-corrected version of the raw read data”); *see also* Ex.1005, [0079] (describing that an ECC scheme selected based on “a number of errors corrected.”). Ex.1003, ¶189.

9. Claims 22 and 29

[22.0/29.0] The apparatus of claim 15/24, wherein the gear includes a lower-latency ECC decode option and one or more higher-latency ECC decode options, wherein the gear is selected such that performance of ECC decoding with the lower-latency ECC decode option alone is sufficient to provide corrected data at least 90% of the time.

Li combined with Cideciyan renders obvious [22.0/29.0] for the reasons discussed at [14.0]. Ex.1003, ¶190.

10. Claim 23

[23.0] *The apparatus of claim 15, further comprising a plurality of flash memory devices, wherein the apparatus comprises a flash drive.*

Li combined with Cideciyan renders obvious [23.0]. Ex.1003, ¶191.

As shown in FIG. 2A, Li describes an apparatus comprising *a plurality of flash memory devices* (flash devices 292). Ex.1005, FIG. 2, [0110] (“one or more storage devices, such as **flash devices 292**.”). Li further describes the apparatus *comprises a flash drive* (Solid State Disk/Drive (SSD)). Ex.1005, [0014], [0016] (“Some SSDs use flash memory to provide non-volatile storage.”); Ex.1003, ¶192.

11. Claim 24

[24.0] *An apparatus for selecting an error correction coding (ECC) scheme, the apparatus comprising:*

To the extent that the preamble is limiting, Li combined with Cideciyan renders obvious [24.0] for reasons discussed at [15.0]. Ex.1003, ¶193.

[24.1.1] *a means for determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof, wherein the two or more flash memory pages of a region can be read simultaneously,*

Li combined with Cideciyan renders obvious [24.1.1] for the reasons discussed at [1.1.1] and [15.1.1]. Ex.1003, ¶¶194-199; VII.B.1.

A POSITA would have understood that [24.1.1] to have:

Function: determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions. Ex.1001, 4:38-40. Ex.1003, ¶197.

Structure: a circuit. Ex.1001, 4:38-40 (“**a first circuit** configured to determine a bit error rate associated with a region...”); Ex.1003, ¶198.

As discussed at [1.1.1] and [15.1.1], Li combined with Cideciyan renders obvious [24.1.1], because it renders obvious a circuit (ECC 261 implementing Error Statistics Collecting/Tracking 180) for performing the function of determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages. Ex.1003, ¶199.

[24.1.2] wherein the region stores at least data payload and primary and secondary ECC parity symbols corresponding to the data payload of the region;

Li combined with Cideciyan renders obvious [24.1.2] for the reasons discussed at [1.1.2]. Ex.1003, ¶200.

[24.2.1] a means for comparing the determined bit error rate to one or more predetermined thresholds corresponding to a set of predefined gears comprising at least a first gear and a second gear, wherein the predefined gears correspond to different predefined ECC schemes,

Li combined with Cideciyan renders obvious [24.2.1] for reasons discussed at [1.2.1] and [15.2.1]. Ex.1003, ¶¶201-206.

A POSITA would have understood that [24.2.1] to have:

Function: comparing the determined bit error rate to one or more

predetermined thresholds corresponding to a set of predefined gears comprising at least a first gear and a second gear. Ex.1001, 4:38-40; Ex.1003, ¶204.

Structure: a circuit. Ex.1001, 4:38-47 (“**a first circuit** configured...to compare the determined bit error rate...”); Ex.1003, ¶205.

As discussed at [1.2.1] and [15.2.1], Li combined with Cideciyan renders obvious [24.2.1], because it renders obvious a circuit (Error Statistics Collecting/Tracking 180) for performing the function of comparing the determined bit error rate to one or more predetermined thresholds corresponding to a set of predefined gears, wherein the predefined gears correspond to different predefined ECC schemes. Ex.1003, ¶206.

[24.2.2] wherein the first gear has a different data payload size and correction capability than the second gear, wherein the amount of memory space allocated for the storage of data payload within the region varies between the first gear and the second gear to accommodate a varying number of parity symbols between the first gear and the second gear;

Li combined with Cideciyan renders obvious [24.2.2] for reasons discussed at [1.2.2]. Ex.1003, ¶207.

[24.3] a means for selecting a gear from the set for the region based at least partly on the comparisons to the one or more predetermined thresholds.

Li combined with Cideciyan renders obvious [24.3] for reasons discussed at [1.3] and [15.3]. Ex.1003, ¶¶208-213.

A POSITA would have understood that [24.3] to have:

Function: selecting a gear from the set for the region based at least partly on the comparisons to the one or more predetermined thresholds. Ex.1003, ¶¶211.

Structure: a circuit. Ex.1001, 4:47-49 (“**a second circuit** configured to select a gear...”); Ex.1003, ¶212.

As discussed at [1.3] and [15.3], Li combined with Cideciyan renders obvious renders obvious a circuit (SSD controller 200 including elements implementing adaptive encoder hardware block) for performing the function of selecting a gear as recited. Ex.1005, [0023], [0126]; Ex.1003, ¶213.

12. Claim 25

[25.0] *The apparatus of claim 24, further comprising means for reading data from the region of flash memory,*

Li combined with Cideciyan renders obvious [25.0] for the reasons discussed at [5.0] and [17.0]. Ex.1003, ¶¶214-218.

First, a POSITA would have understood that [25.0] to have:

Function: reading data from the region of flash memory. Ex.1003, ¶216.

Structure: a circuit including an ECC encoder/decoder. Ex.1001, 53:1-7; Ex.1003, ¶217.

As discussed at [5.0] and [17.0], Li combined with Cideciyan renders obvious a circuit (e.g., using SSD controller 200 including elements implementing the read-storage-data path) that performs the function of *reading data from the*

region of flash memory. Ex.1003, ¶218.

[25.1] wherein the determining means further comprises means for estimating the bit error rate based at least partly on a number of errors corrected by ECC in connection with reading the data from the region.

Li combined with Cideciyan renders obvious [25.1] for the reasons discussed at [5.1] and [17.1]. Ex.1003, ¶¶219-223.

A POSITA would have understood that [25.1] to have:

Function: estimating the bit error rate based at least partly on a number of errors corrected by ECC in connection with reading the data from the region. Ex.1003, ¶221.

Structure: a circuit including an ECC encoder/decoder. Ex.1001, 4:38-40 (“**a first circuit** configured to determine a bit error rate associated with a region associated with a plurality of flash memory pages or integer fractions thereof”); 53:1-7; Ex.1003, ¶222.

As discussed at [5.1] and [17.1], Li combined with Cideciyan renders obvious a circuit (ECC 261 implementing Error Statistics Collecting/Tracking 180, Universal Encoder 120 and Universal Decoder 160) for determining (*see* [24.1.1]) further performs the estimating function above. Ex.1003, ¶223.

13. Claim 30

[30.0] The apparatus of claim 24, further comprising a plurality of flash memory devices, wherein the apparatus comprises a flash drive.

Li combined with Cideciyan renders obvious [30.0] for reasons discussed at [23.0]. Ex.1003, ¶225.

14. Claims 31, 32, and 33

[31.0/32.0/33.0] *The method of claim 1/apparatus of claim 15/apparatus of claim 24, wherein the data payload and corresponding primary ECC parity symbols comprise primary codewords,*

Li combined with Cideciyan renders obvious [31.0/32.0/33.0]. Ex.1003, ¶¶226-227.

As shown in FIG. 4 below, Cideciyan as applied in the combination provides that *data payload* (input data) *and corresponding primary ECC parity symbols* (C1 parity symbols) *comprise primary codewords* (C1 codeword). Ex.1006, 10:60-64; 11:9-14; Ex.1003, ¶227.

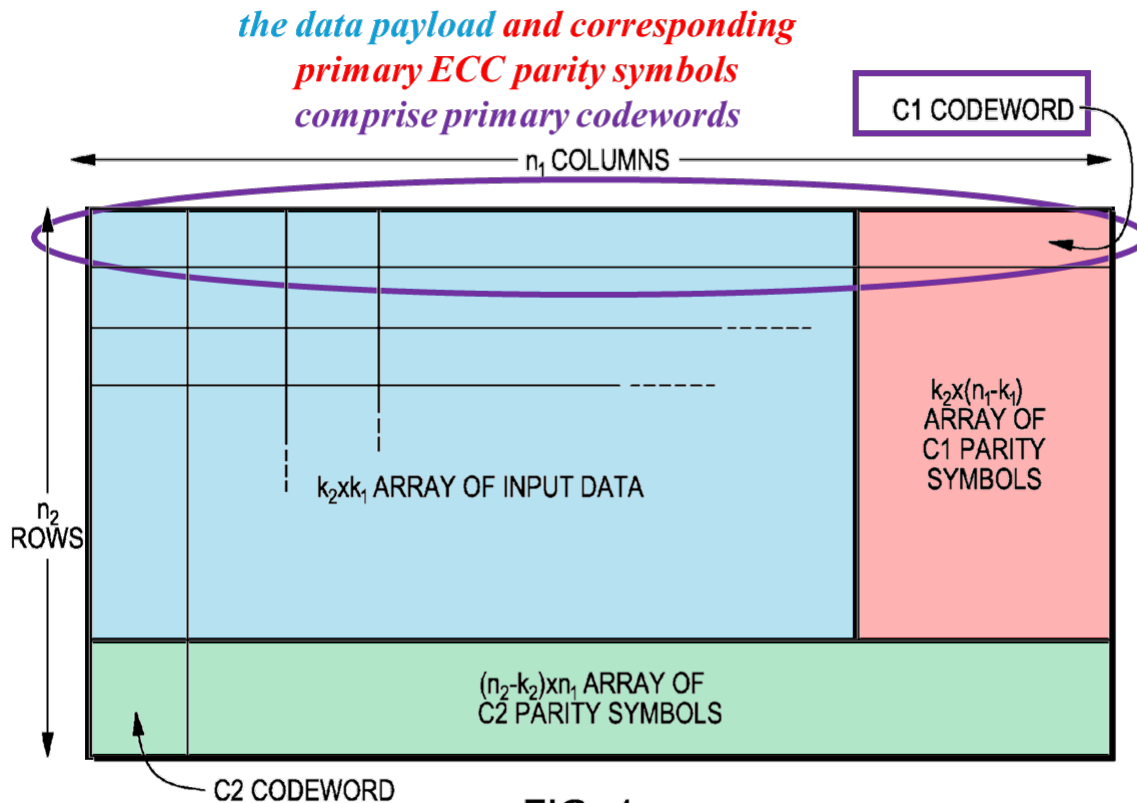


FIG. 4

Ex.1006, FIG. 4, annotated.

[31.1/32.1/33.1] wherein the data payload and corresponding secondary ECC parity symbols comprise secondary codewords,

Li combined with Cideciyan renders obvious [31.1/32.1/33.1]. Ex.1003, ¶¶228-229.

As shown in FIG. 4 below, Cideciyan as applied in the combination provides that data payload (input data) and corresponding secondary ECC parity symbols (C2 parity symbols) comprise secondary codewords (C2 codeword). Ex.1006, 11:1-5; 11:9-14; Ex.1003, ¶229.

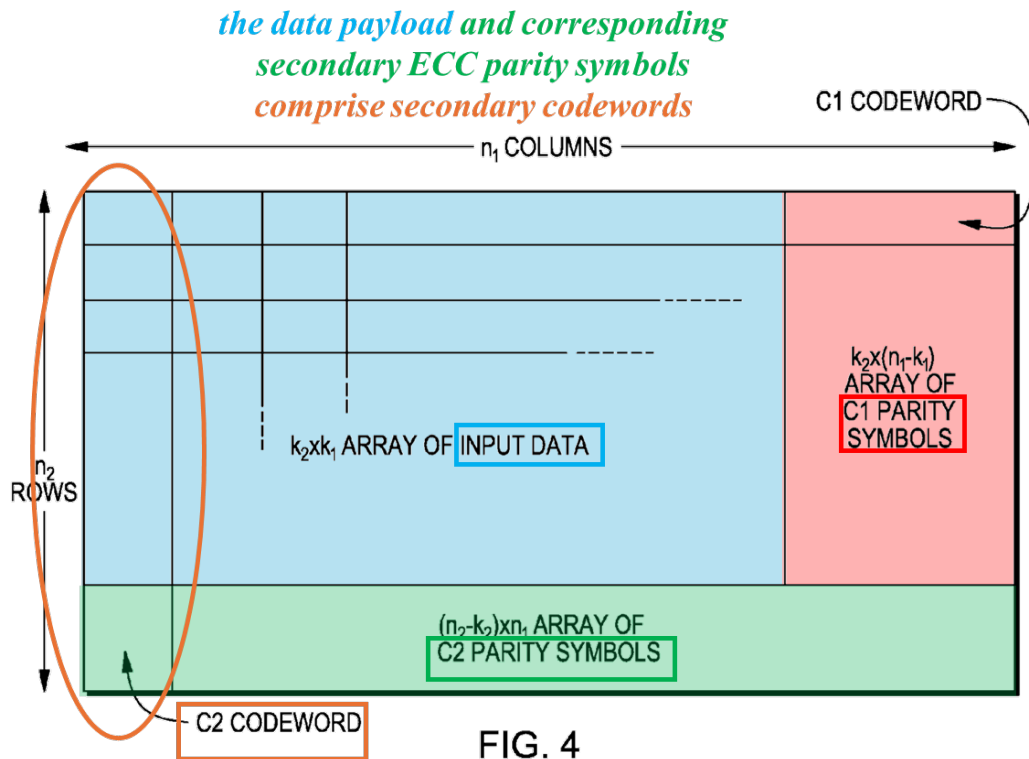


FIG. 4

[31.2/32.2/33.2] wherein the secondary ECC parity symbols of each secondary codeword are programmed across the region,

Li combined with Cideciyan renders obvious [31.2/32.2/33.2]. Ex.1003, ¶¶230-232.

Cideciyan describes that “while the C2 parity codes are stored in the N^{th} channel here, **any channel could be used for the parity.**” Ex.1006, 11:39-42; Ex.1003, ¶231 (citing Ex.1006, 11:9-14; 17:50-55). As shown in annotated FIG. 5 below, a POSITA would have understood that one implementation that “**any channel**” could be used is to program C2 parity symbols *across the region* (across the N channels), for the benefits of balancing the workload of both reading and writing operations, reducing the burden on any single channel or chip. Ex.1003,

¶232.

*the secondary ECC parity symbols
of each secondary codeword are
programmed across the region*

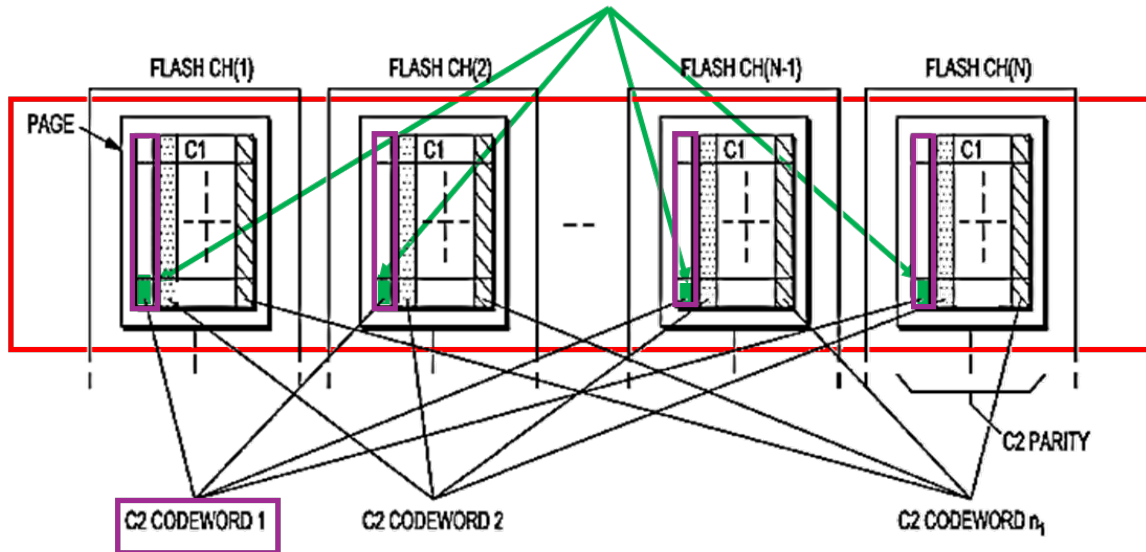


FIG. 5

Ex.1006, FIG. 5, annotated.

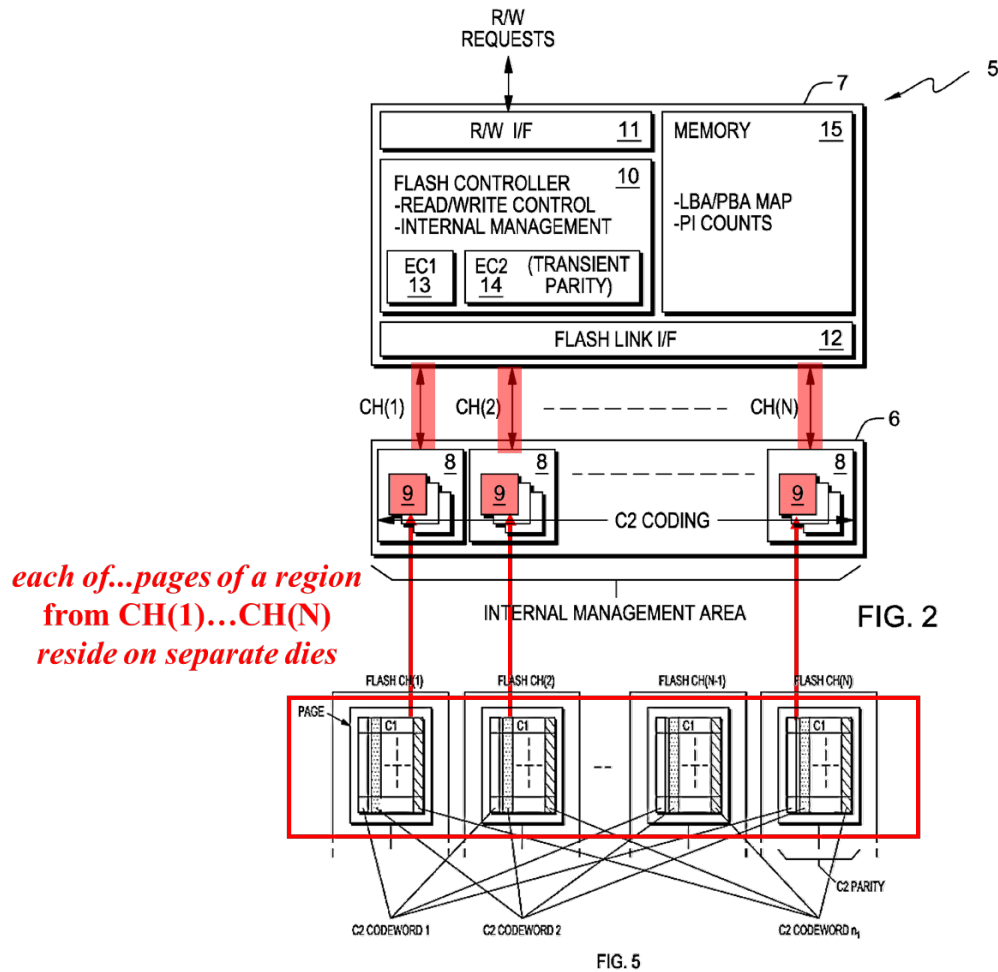
[31.3/32.3/33.3] wherein each of the two or more flash memory pages of a region reside on separate dies.

Li combined with Cideciyan renders obvious [31.3/32.3/33.3]. Ex.1003,

¶233-234.

Cideciyan as applied in the combination provides that the fixed number of multiple pages of a region reside on separate dies, corresponding to N channels.

Ex.1006, 8:53-59; Ex.1003, ¶234.



Ex.1006, FIGS. 2 and 5, annotated.

F. Ground 2: Claims 1-2, 4-5, 14-17, 22-25, and 29-33 are obvious in view of Li, Cideciyan, and Frost.

1. Claim Analysis

The claim analysis for Ground 2 is similar to that of Ground 1, with the additional analysis below explaining that Li combined with Cideciyan and Frost renders [1.1.1] of Claim 1 obvious. *See supra* X.E; Ex.1003, ¶235.

[1.1.1] *determining a bit error rate associated with a region comprising a fixed number of two or more flash memory pages or integer fractions thereof, wherein*

the two or more flash memory pages of a region can be read simultaneously,

The combination of Li, Cideciyan, and Frost renders obvious [1.1.1].

Ex.1003, ¶¶236-247.

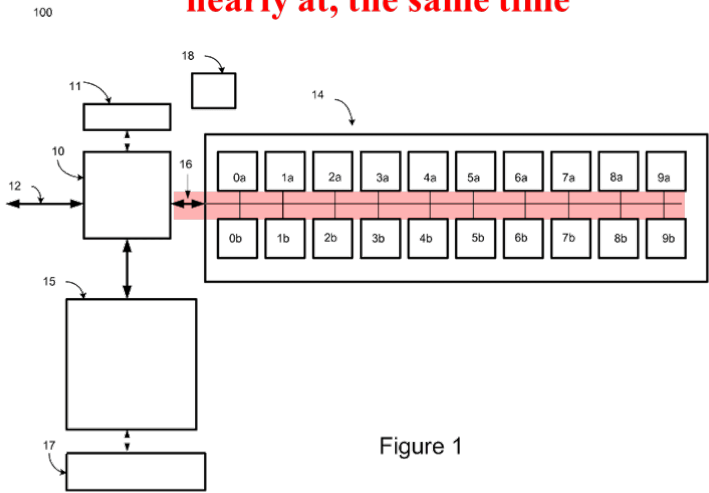
To the extent that it is argued that Li combined with Cideciyan’s teaching that a group of N pages from across N channels (which is a traditional “page stripe”) accessible in parallel (*see* Ground 1:[1.1.1]) does not explicitly teach that those pages “*can be read simultaneously*,” the further combination with Frost renders it obvious, because the feature that pages from different channels “*can be read simultaneously*” as recited in [1.1.1] is a well-known and conventional implementation detail as taught by Frost. Ex.1003, ¶238.

For further detail, referring to FIG. 1 reproduced below, Frost describes that: each group of two chips forms a **‘Lane,’ also sometimes referred to as a ‘Channel,’**....the controller 10 can issue commands and send or **receive data across the various communication buses at the same time** such that the system controller can access the memory chips corresponding to the individual Lanes **at, or very nearly at, the same time.**

Ex.1007, 6:46-67, 11:51-60 (all of the pages from separate lanes “can be written to or **read from the memory array 14 simultaneously or near-simultaneously**”);

Ex.1003, ¶239.

“the system controller can access the memory chips corresponding to the individual Lanes at, or very nearly at, the same time”



Ex.1007, FIG. 1, annotated.

As one example, referring to FIG. 7B below, Frost describes all of the pages associated with a **page stripe** can be read simultaneously across the multiple Lanes/Channels. Ex.1007, 11:51-60; Ex.1003, ¶240

“all of the Pages associated with Page Stripe 700 can be written to or read from the memory array 14 simultaneously or near-simultaneously.”

70a	70b	70c	70d	70e	70f	70g	70h	70i	70j
DPAGE0	DPAGE1	DPAGE2	DPAGE3	DPAGE4	DPAGE5	DPAGE6	DPAGE7	DPAGE8	PPAGE9
LANE0	LANE1	LANE2	LANE3	LANE4	LANE5	LANE6	LANE7	LANE8	LANE9

Figure 7B

Ex.1007, FIG. 7B, annotated.

As such, in Li combined with Cideciyan and Frost, pages in the region (a group of pages (page stripe) distributed across the channels) can be read simultaneously in Li combined with Cideciyan, which provides the benefit of “relatively quick read[] operations,” which renders obvious [1.1.1]. Ex.1007, 11:51-60; Ex.1003, ¶¶241-242.

Reasons to Combine Li, Cideciyan, and Frost

First, a POSITA would understand that parallel access (including read operations) using channels in Cideciyan (and Li) would be implemented simultaneously as shown in Frost. Ex.1003, ¶243.

Second, Li combined with Cideciyan teaches a region including pages distributed across multiple channels, while Frost provides how channels provide simultaneous access to those pages. The combination would provide the obvious, predictable, and beneficial result of simultaneous access, including “relatively quick read and write operations” and efficient and effective data storage and retrieval, as taught by Frost. Ex.1003, ¶244.

Third, as discussed at Ground 1:[1.1.1], a POSITA would have understood Cideciyan’s group of pages of FIG. 5 to be a traditional “page stripe.” Ex.1003, ¶245. Frost provides that pages in a “page stripe” are read simultaneously. *See, e.g.*, Ex.1007, FIG. 7B; Ex.1003, ¶245.

Additionally, the results of the proposed combination would have been

predictable and a POSITA would have had a reasonable expectation of success, because the combination merely applies known technique (channels and page stripes providing simultaneously read of pages) to a known device to yield the predictable results of quick read and write operations. Ex.1003, ¶246.

G. Grounds 3 and 4: Claims 6-7 and 18 are rendered obvious by each of Combination of Li, Cideciyan, and Leung (Ground 3) and Combination of Li, Cideciyan, Frost, and Leung (Ground 4).

1. Reasons to Combine Leung with Each of Grounds 1 and 2.

Li combined with Cideciyan and/or Frost, teaches that selecting a gear from the set of predetermined gears for the region, based on the determined bit error rate. Leung provides an obvious way of selection of the gear based on *a predetermined acceptable uncorrectable bit error rate (aUBER)* in addition to the bit error rate. Applying Leung's teaching in Li combined with Cideciyan and/or Frost would provide the benefit of optimal ECC scheme selection and memory array configuration to satisfy the aUBER requirements, while maximizing data payload size. Ex.1008, Table 1, [0033]; Ex.1003, ¶¶248-249.

The results of the proposed combinations would have been predictable and a POSITA would have had a reasonable expectation of success, because the combination merely applies known technique (Leung's selecting a gear based on a predetermined aUBER, in addition to the bit error rate) to a known device to yield the predictable results. Ex.1003, ¶249.

2. Claim 6

[6.0] *The method of claim 1, wherein selecting further comprises selecting the gear that has the highest data payload size and sufficient correction capability for the determined bit error rate such that memory reads from the region can be corrected with no more than a predetermined acceptable uncorrectable bit error rate (aUBER).*

Leung as combined with Grounds 1 and 2 each renders obvious [6.0].

Ex.1003, ¶¶250-255.

First, Li at least suggests [6.0] because its goal to provide more space “available to store data.” Ex.1005, [0104]; Ex.1003, ¶251.

Second, Leung describes choosing a gear (e.g., choosing bit error rate level 1 for incoming BER 3.4E-5) with the lowest error correction capability among all three gears (bit error rate levels 1, 2, and 3), where all three gears satisfy the output error rate requirement (e.g., 1E-15). Ex.1008, Table 1 reproduced below. Ex.1003, ¶¶252-253; Ex.1008, [0034] (“Initially, [operating] using **bit error rate level 1**. When the detected error rate approaches 3.4e-5, the device may...operate at **bit error level 2..**”).

TABLE 1

Bit error rate level	Information bits per codeword	Parity per codeword	Codeword size	Information bits per 524 B	Capacity	Code rate	Incoming BER to maintain 1E-15 output error rate
1	512 B	12 B	524 B	512 B	100%	97.7%	3.4E-5
2	250 B	12 B	262 B	500 B	97.7%	95.45	6.3E-5
3	119 B	12 B	131 B	476 B	93.0%	90.8%	11.6E-5

Ex.1008, FIG. 1.

As discussed at X.G.1, a POSITA would have implemented Leung's teaching in the combination of Li, Cideciyan, and Frost, for maximizing data payload size while satisfying the aUBER requirement. Ex.1003, ¶254.

3. Claim 7

[7.0] The method of claim 6, wherein the aUBER is 1E-16.

Leung as combined with Grounds 1 and 2 each renders obvious [7.0].
Ex.1003, ¶¶256-258.

First, the limitation is a percentage without showing of any unexpected result, therefore is prima facia obvious.

Second, Leung describes that the aUBER (outgoing error rate) is 1E-16.
Ex.1008, FIG. 2, [0035]; Ex.1003, ¶257.

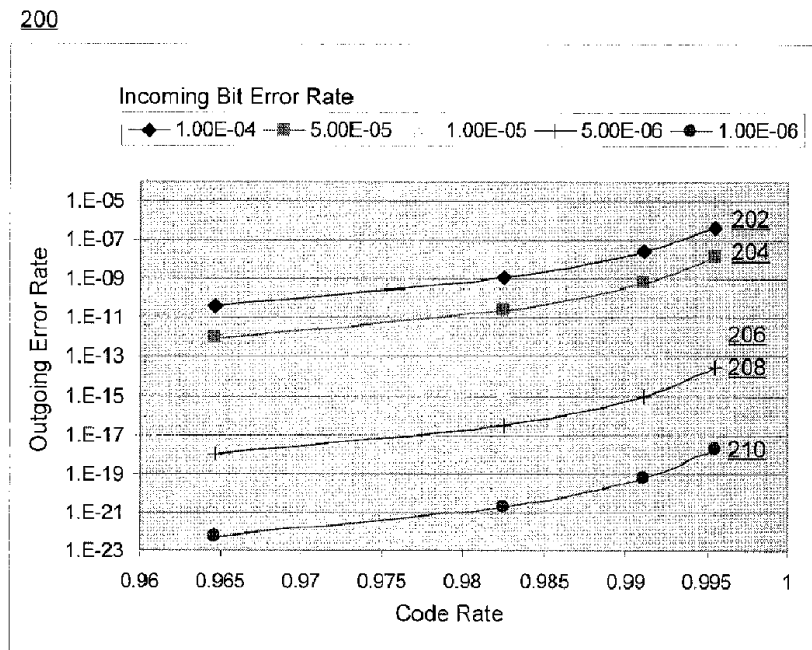


Fig. 2

Ex.1008, FIG. 2.

Thus, Leung as combined with Grounds 1 and 2 each renders obvious [7.0].

Ex.1003, ¶258.

4. Claim 18

[18.0] *The apparatus of claim 15, wherein the processor is further configured to select a gear that has the highest data payload size and sufficient correction capability for the determined bit error rate such that memory reads from the region can be corrected with no more than a predetermined acceptable uncorrectable bit error rate (aUBER).*

Leung as combined with Grounds 1 and 2 each renders obvious [18.0].

Ex.1003, ¶¶259-262.

First, Leung as combined with Grounds 1 and 2 renders obvious selecting a

gear as recited for reasons discussed at Grounds 3 and 4:[6.0]. Ex.1003, ¶260.

Second, Li combined with Cideciyan and Leung renders that the “*processor*” is configured to perform selected functions including the selecting a gear function. Ex.1005, [0137] (“many functional aspects of embodiments are realizable selectively [using] processor”); [0131]; *see also* Ex.1006, 9:29-31; Ex.1003, ¶261.

Accordingly, Li combined with Cideciyan and Leung renders obvious [18.0]. Ex.1003, ¶262.

XI. DISCRETIONARY DENIAL IS NOT WARRANTED

Petitioner respectfully submits that the Board should not exercise its discretion under 35 U.S.C. §§314(a) or 325(d) to deny this Petition.

A. The Fintiv factors favor institution.

Denial would be improper based on factors in *Apple v. Fintiv*, IPR2020-00019, Paper 11 (P.T.A.B. Mar. 20, 2020) (precedential).

1. No evidence regarding a stay

No motion to stay has been filed, so the Board should not infer the outcome of such a motion.

2. Parallel proceeding trial date

This factor weighs against discretionary denial. While the scheduled trial date is June 16, 2026, (Ex.1021, 25), the projected trial date—based on median time-to-trial statistics—is in January of 2027², which is well after the projected date for the Board to issue a Final Written Decision in July of 2026. Because the

² The current average time-to-trial for the district court hearing the co-pending litigation is 29.1 months. *See* Ex.1015 and Ex.1016. The Director recently clarified that in weighing Fintiv Factor 2 the Board may consider “evidence regarding the most recent statistics on median time-to-trial for civil actions in the district court in which the parallel litigation resides.” Ex.1017, *Fintiv* Memorandum, at 8–9.

projected trial date is approximately six months *after* the Board’s expected date for issuing a Final Written Decision, this factor weighs against discretionary denial.

3. Investment in the parallel proceeding

The investment in the co-pending litigation has been minimal. No schedule has been set. Thus, the Markman hearing is unlikely to occur before the Board issues an institution decision. This lack of investment favors institution.

4. Overlapping issues with the parallel proceeding

The co-pending litigation is in its early stages. Petitioner served its preliminary invalidity contentions on December 16, 2024. However, Petitioner has not served its final invalidity contentions, nor has the deadline for doing so been set. The extent of overlap is thus speculative at this point in time.

5. Petitioner is a defendant

Petitioner is a defendant in the co-pending litigation. That is true of most Petitioners in IPR proceedings, making this factor neutral.

6. Other circumstances

“[T]he PTAB will not deny institution of an IPR or PGR under *Fintiv* (i) when a petition presents compelling evidence of unpatentability.” Ex.1017, 9. Here, the petition—along with Dr. Dolecek expert testimony—plainly shows that the ’085 patent claims no more than known concepts related to adaptive ECC in a flash memory based storage system.

B. Advanced Bionics Test Favors Institution.

Denial under § 325(d) is not warranted because the challenges presented in this petition are neither cumulative nor redundant to the prosecution of the '085 patent. The Examiner did not consider Cideciyan, Frost, and Leung during prosecution. Li (Ex.1005) and patents (US 8,255,620; 8,560,881; and 8,713,245) related to Frost (Ex.1007) were listed on IDS, but were never relied upon by the Office. Ex.1002, 633, 730. There is no evidence that the Examiner substantively considered Li and the patents related to Frost with respect to the Challenged Claims. Moreover, the Examiner did not consider Li and the patents related to Frost in combination with Cideciyan and/or Leung as presented in Petitioner's grounds.

Thus, the first prong of the *Advanced Bionics* analysis is not met, because the same or substantially the same art or arguments were not presented during prosecution. *Advanced Bionics*, IPR2019-01469, Paper 6 at 8 (Feb. 13, 2020) (precedential). The second prong is also not met, because the Examiner erred in a material to patentability by overlooking Li and Cideciyan for teaching adaptive ECC techniques applying to a region including a fixed number of pages that can be read simultaneously, where the region stores data payload and primary and secondary ECC parity symbols. *Id.* at 8-9, n.10.

XII. CONCLUSION

Petitioner has established a reasonable likelihood that the Challenged Claims are unpatentable.

XIII. MANDATORY NOTICES

A. Real Party-in-Interest

The real parties-in-interest (“RPI”) are Sandisk Technologies, Inc. and Western Digital Technologies, Inc. (“Petitioner”).

B. Related Matters

Pursuant to 37 C.F.R. § 42.8(b)(2), to the best knowledge of the Petitioner, the ’085 patent is or was involved in the following cases:

Case Heading	Number	Court	Filed	Status
<i>Polaris PowerLED Technologies, LLC v. Western Digital Corporation et al.</i>	2-24-cv-02864	CDCA	April 9, 2024	Pending
<i>Polaris PowerLED Technologies, LLC v. Western Digital Corporation et al.</i>	1-24-cv-00101	WDTX	January 30, 2024	Pending

C. Lead and Back-up Counsel and Service Information

Lead Counsel

David O'Dell
HAYNES AND BOONE, LLP
2801 N. Harwood Street Suite 2300
Dallas, TX 75201

Phone: (972) 739-8635
Fax: (214) 200-0853
david.odell.ipr@haynesboone.com
USPTO Reg. No. 42,044

Back-up Counsel

Hong Shi
HAYNES AND BOONE, LLP
2801 N. Harwood Street Suite 2300
Dallas, TX 75201

Phone: (512) 867-8440
Fax: (214) 200-0853
hong.shi.ipr@haynesboone.com
USPTO Reg. No. 69,009

Calmann J. Clements
HAYNES AND BOONE, LLP
2801 N. Harwood Street Suite 2300
Dallas, TX 75201

Phone: (972) 739-8638
Fax: (214) 200-0853
calmann.clements.ipr@haynesboone.com
USPTO Reg. No. 66,910

Please address all correspondence in this proceeding to lead and back-up counsel. Petitioner consents to service in this proceeding by email at the addresses above.

Respectfully submitted,

Dated: January 29, 2025
HAYNES AND BOONE, LLP
2801 N. Harwood Street Suite 2300
Dallas, Texas 75201
Customer No. 27683

/David M. O'Dell/
David M. O'Dell
Lead Counsel for Petitioner
Registration No. 42,044

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), Petitioner hereby certifies, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,858. Pursuant to 37 C.F.R. § 42.24(d), this word count excludes the table of contents, table of authorities, mandatory notices under § 42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing.

Dated: January 29, 2025

/David M. O'Dell/
David M. O'Dell
Lead Counsel for Petitioner
Registration No. 42,044

CERTIFICATE OF SERVICE

The undersigned certifies that, in accordance with 37 C.F.R. § 42.6(e) and 37 C.F.R. § 42.105, service was made on Patent Owner as detailed below.

Date of service January 29, 2025

Manner of service FEDERAL EXPRESS

Documents served Petition for *Inter Partes* Review Under 35 U.S.C. § 312 and 37 C.F.R. § 42.104 of U.S. 9,183,085; Petitioner's Power of Attorney; Petitioner's Exhibit List; and All Exhibits.

Persons served Knobbe, Martens, Olson & Bear, LLP
2040 Main Street
Fourteenth Floor
Irvine, CA 62614

/David M. O'Dell/
David M. O'Dell
Lead Counsel for Petitioner
Registration No. 42,044