

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

WESTERN DIGITAL TECHNOLOGIES, INC.,
Petitioner

v.

VIASAT, INC.,
Patent Owner

Case No. IPR2022-01126
Patent No. 8,615,700

**DECLARATION OF LARA DOLECEK, Ph.D.
IN SUPPORT OF PETITION FOR *INTER PARTES*
REVIEW OF U.S. PATENT NO. 8,615,700**

I. INTRODUCTION

1. My name is Lara Dolecek, Ph.D.

2. I make this declaration in support of Petitioner Western Digital Technologies, Inc.'s ("Petitioner") petition for *inter partes* review of U.S. Patent No. 8,615,700 (Exhibit 1001), which I refer to in my declaration as "the '700 Patent." I have been asked to focus on the following claims of the '700 Patent: claims 1, 2, 4, 8, 15, 17, and 19.

3. I am over 21 years of age and otherwise competent to make this declaration.

4. Although I am being compensated for my time at my normal rate in preparing this declaration, the opinions herein are my own, and I have no stake in the outcome of the *inter partes* review proceeding.

5. I am not an employee of Petitioner or any affiliate or subsidiary thereof.

6. This declaration summarizes the opinions I have formed to date. I reserve the right to modify my opinions, if necessary, based on further review and analysis of information that I receive subsequent to the filing of this report, including in response to positions that parties to the *inter partes* review proceeding, or their experts, may take that I have not yet seen.

II. MY EXPERIENCE AND QUALIFICATIONS

7. I am a full professor of Electrical and Computer Engineering at the University of California, Los Angeles (UCLA). At UCLA, I lead a research group called Laboratory for Robust Information Systems.

8. I earned a B.S., M.S., and Ph.D. in Electrical Engineering and Computer Sciences (EECS) as well as an M.A. degree in Statistics from the University of California, Berkeley (UC Berkeley). For my doctoral research on error correcting codes, I was awarded the 2007 David J. Sakrison Memorial Prize from the EECS Department at UC Berkeley, given for the most outstanding doctoral dissertation.

9. I have written over 160 peer-reviewed articles in leading venues on the topics related to error correction, statistical algorithms, and data processing and storage. I have also written a book titled “Channel Coding Methods for Non-Volatile Memories,” published in 2016.

10. I am currently a Distinguished Lecturer of the IEEE Information Theory Society and an elected member of the Board of Governors of the IEEE Information Theory Society.

11. I have received several career awards for my work, including the CAREER Award from the National Science Foundation, Intel Early Career Award, and IBM Faculty Award. With my research group and collaborators, I have

received over dozen best paper awards and recognitions for the work on error correction coding and related topics, including Best Paper Awards from the IEEE Data Storage Society, Memorable Paper Awards from the Non-Volatile Memories Workshop, and Best-of-SELSE Awards from the IEEE Workshop on Silicon Errors in Logic – System Effects (SELSE).

12. I currently serve as an Associate Editor for the Coding Theory for IEEE Transactions on Information Theory, and I am a Technical Program Committee Co-Chair for the 2022 IEEE Information Theory Workshop, a flagship conference of the IEEE Information Theory Society.

13. A current copy of my curriculum vitae is attached to this declaration as Appendix A.

14. I have used my education and experience researching, publishing and teaching, and my understanding of the knowledge, creativity, and experience of a person having ordinary skill in the forward error correction (“FEC”), error detection, and error correction coding (“ECC”) arts, including as applied to flash memory, to which the ’700 Patent relates, in forming the opinions expressed in this declaration, as well as any other materials discussed herein.

III. MATERIALS CONSIDERED

15. In forming my opinions, I read and considered the ’700 Patent and its prosecution history, the exhibits listed below (which I understand have been

included as part of the petition for *inter partes* review), as well as any other material referenced herein. The exhibits I have analyzed include:¹

No.	Document
1001	U.S. Patent No. 8,615,700
1002	File History for U.S. Patent No. 8,615,700
1004	United States Patent Application Publication No. 2011/0126079 to Wu
1005	United States Patent Application Publication No. 2009/0063934 to Jo
1006	United States Patent Application Publication No. 2010/0070681 to Wan
1007	U.S. Provisional Application No. 61/234,911
1014	Mielke et al., Bit Error Rate in NAND Flash Memories, IEEE CFP08RPS-CDR 46th Annual International Reliability, Physics Symposium, Phoenix, 2008
1015	Michelsoni et al., A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput, ISSCC 2006 / SESSION 7 / NON-VOLATILE MEMORY / 7.6
1016	Chen et al., Error Correction for Multi-Level NAND Flash Memory Using Reed-Solomon Codes, SiPS 2008
1017	Duann, SLC & MLC Hybrid, Flash Memory Summit, Santa Clara Aug. 2008
1018	Duann, Error Correcting Techniques for Future NAND Flash Memory in SSD Applications, Flash Memory Summit, Santa Clara Aug. 2009

16. For any future testimony I may give in this matter, I may use some or all of the documents and information cited to, referred to, and identified in this declaration, as well as any additional materials that are entered into evidence in this matter.

¹ Exhibit 1003 is my declaration.

IV. LEGAL PRINCIPLES OF ANTICIPATION AND OBVIOUSNESS

17. I have been informed and I understand that a patent claim is anticipated when a single piece of prior art describes every element of the claimed invention, arranged in the same way as in the claim, either expressly or inherently. I have been told to consider elements to be inherently described only if they are necessarily present in the piece of prior art.

18. I have been informed and I understand that a patent claim is unpatentable and invalid if the subject matter of the claim as a whole would have been obvious to a person of ordinary skill in the art in the field of the patent as of the time of the invention at issue. I have been informed and understand that the following factors must be evaluated to determine whether the claimed subject matter is obvious: (i) the scope and content of the prior art; (ii) the difference or differences, if any, between each claim of the patent and the prior art; (iii) the level of ordinary skill in the art at the time the patent was filed; and (iv) any objective indicia of non-obviousness.

19. I have been informed and I understand that the objective indicia of non-obviousness (or “secondary considerations”) that should be considered include, for example, the following: (i) commercial success; (ii) long-felt but unresolved needs; (iii) copying of the invention by others in the field; (iv) initial expressions of disbelief by experts in the field; (v) failure of others to solve the

problem that the inventor solved; and (vi) unexpected results. I have been informed and understand that evidence of these objective indicia must be commensurate in scope with the claimed subject matter. I am not aware of any objective indicia of non-obviousness relevant to the claims of the '700 Patent.

20. I have been informed and I understand that independently made, simultaneous inventions, made within a comparatively short space of time, are evidence that the claimed apparatus was the product only of ordinary mechanical or engineering skill.

21. In determining whether the subject matter as a whole would have been obvious at the time that the invention was made to a person having ordinary skill in the art, I have been informed of and understand certain principles regarding the combination of elements of the prior art. A combination of familiar elements according to known methods is likely to be obvious when it yields predictable results. Also, if a person of ordinary skill in the art can implement a predictable variation in a prior art device, and would see the benefit from doing so, such a variation would be obvious. In particular, when there is pressure to solve a problem and there are a finite number of identifiable, predictable solutions, it would be reasonable for a person of ordinary skill to pursue those options that fall within his or her technical grasp. If such a process leads to the claimed invention,

then the latter is not a patentable invention, but the result of ordinary skill and common sense.

22. I have also been informed and understand that a teaching, suggestion or motivation is a useful guide in establishing a rationale for combining elements of the prior art. The test poses the question as to whether there is an explicit teaching, suggestion, or motivation in the prior art to combine prior art elements in a way that realizes the claimed invention. Though useful to the obviousness inquiry, I understand that this test should not be treated as a rigid rule. It is not necessary to seek out precise teachings; it is permissible to consider the inferences and creative steps that a person of ordinary skill in the art (who is considered to have an ordinary level of creativity and is not an automaton) would employ.

V. LEVEL OF ORDINARY SKILL IN THE ART

23. I have been informed and understand that the disclosure of patents and prior art references are to be viewed from the perspective of a person having ordinary skill in the art at the time of the alleged invention. I have been told I may use the shorthand “POSITA” (or “POSA”) to describe this person. I have provided my opinions from this perspective. I understand that the relevant time period is around August 18, 2010, the filing date of the parent non-provisional application

leading to the '700 Patent.² My opinions would not change of the '700 Patent were entitled to the priority date of the provisional application, filed August 18, 2009.

24. I have been informed and understand that prior art references can provide evidence of the level of ordinary skill in the art, and that factors that may be considered in determining this level of skill can include the educational level of the inventors and active workers in the field, the types of problems encountered in the art, the prior art solutions to those problems, the rapidity with which innovations are made, and the sophistication of the technology.

25. Based on my education and experience, as described above and in my CV, as well as consistent with the '700 Patent and the prior art described below, it is my opinion that those of ordinary skill in the art during the relevant period would have had at least a B.S. in electrical engineering or computer science or a related field, and four years' experience designing, developing, or researching error correction coding ("ECC") systems. However, I note that this description is approximate, and a higher level of education or skill might make up for less experience, and additional experience could make up for a lower education level;

² As I discuss below and as shown on the Cover Page of the patent, the '700 Patent was filed on August 18, 2010 as U.S. Patent Application No. 12/858,510 (the "'510 Application") and claims priority to U.S. Patent Provisional Application No. 61/234,911 (the "'911 Provisional") (filed August 18, 2009). Ex. 1001 at Cover Page.

for example, an M.S. in the above field or related field, and two years' experience would qualify as a person of ordinary skill, in my opinion.

VI. SUMMARY OF THE '700 PATENT

26. I have reviewed the '700 Patent (Exhibit 1001) and provide the following summary.

27. According to the '700 Patent, as discussed in the "Background" section, the '700 Patent "relates to forward error correction (FEC) in general and, in particular, to FEC for flash memory." Ex. 1001 at 1:16-17.

28. The "Background" section of the '700 Patent continues, that "The advancements of flash memory technology in recent years has dramatically increased storage capacity and decreased the cost of non-volatile semiconductor memory. This has made Solid State Drives (SSDs) (typically a flash memory-based non-volatile memory system) an emerging substitute for magnetic Hard Disk Drive (HDD)." Ex. 1001 at 1:18-23.

29. The '700 Patent, in its "Detailed Description," describes that, in addition to implementing FEC in flash memory, the patent relates to such implementation for other types of solid-state memory:

Although much of the following discussion relates specifically to flash memory, aspects of the inventions are applicable to other forms of solid state memory. Generally, there are two predominant types of flash memory architectures: NOR and NAND. For the purposes of this disclosure, much discussion is

directed to NAND-type flash, while again noting that principles set forth herein have wider applicability.

Ex. 1001 at 2:63-3:2.

30. In the Detailed Description, “[s]ystems, devices, methods, and software are described for FEC for flash memory.” Ex. 1001 at 3:23-24.

31. According to the patent:

Because of this increase in density and the related issues in manufacturing, one of the challenges for flash memory is to maintain high reliability and longevity. The deterioration of the oxide over time, and the disruptions from neighboring memory pages, can lead to data retention and corruption issues translating into bit errors. While the chances of any given data bit being corrupted is quite small, the vast number of data bits in a storage system makes the likelihood of data corruption a very real possibility. FEC may be one of the tools used to address these bit errors to improve the reliability and lifespan of flash memory.

Ex. 1001 at 3:12-22.

32. Figure 1, reproduced below, “is a block diagram of a flash memory decoder system according to various embodiments of the invention.” Ex. 1001 at 2:4-5.

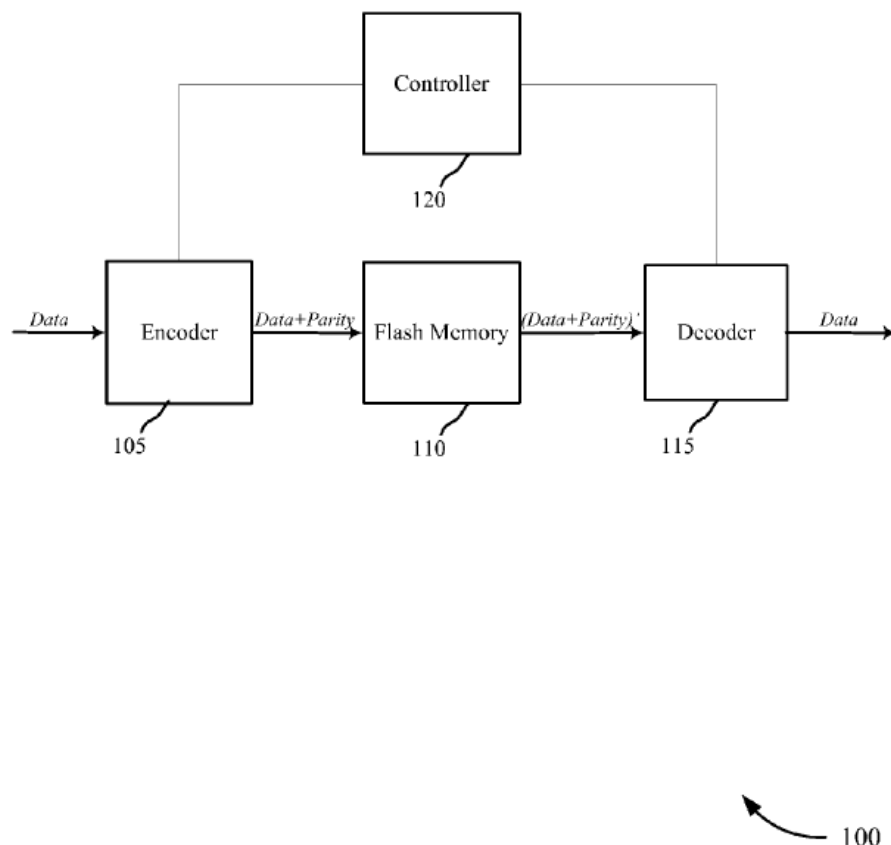


FIG. 1

33. Per the patent, Figure 1 shows “a flash memory system 100” which “includes an encoder 105, flash memory 110, a decoder 115, and a controller 120.” Ex. 1001 at 3:24-26. The patent notes that, “This flash memory system 100 may be a stand-alone device, or may be integrated (in whole or in part) with a computer, server, phone or mobile device, tablet, television, or any other computing device.” Ex. 1001 at 3:27-30.

34. The '700 Patent also discloses that:

Each of these system 100 components may be in communication with each other. The encoder 105, decoder 115, and controller 120 may, individually or collectively, be implemented with one or more Application Specific Integrated Circuits (ASICs) adapted to perform some or all of the applicable functions in hardware. Alternatively, the functions may be performed by one or more other processing units (or cores), on one or more integrated circuits. In other embodiments, other types of integrated circuits may be used (e.g., Structured/Platform ASICs, Field Programmable Gate Arrays (FPGAs), and other Semi-Custom ICs), which may be programmed in any manner known in the art. The functions of each unit may also be implemented, in whole or in part, with instructions embodied in a memory, formatted to be executed by one or more general or application-specific processors.

Ex. 1001 at 3:31-45.

35. The patent continues that, noting that “The encoder 105 may receive data to be stored, and encode the data (e.g., adding parity information) for storage in the flash memory 110 as encoded data.” Ex. 1001 at 3:46-48.

36. In Figure 2, reproduced below, the '700 Patent discloses “a block diagram of a flash memory decoder according to various embodiments of the invention.” Ex. 1001 at 2:7-8.

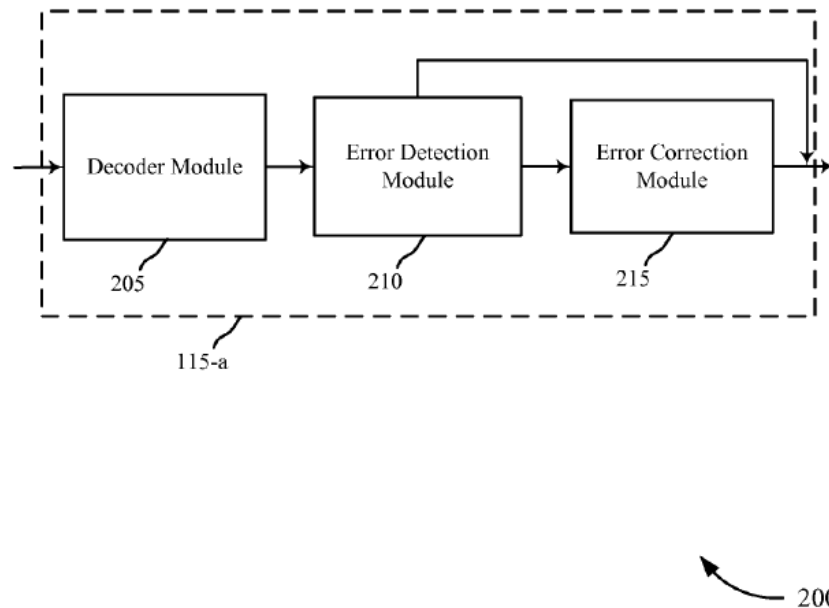


FIG. 2

37. More particularly, the '700 Patent describes its Figure 2 block diagram 200 as follows:

FIG. 2 is a block diagram 200 of a flash memory decoder 115-*a* according to various embodiments of the invention. The flash memory decoder 115-*a* may, for example, be the decoder 115 of FIG. 1, although flash memory decoder 115-*a* may be implemented in a number of other systems and devices, as well. The flash memory

decoder 115-*a* includes a decoder module 205, an error detection module 210, and an error correction module 215, and each of may be in communication with each other. In one example, the error detection module 210 includes a number of sub-modules configured to detect errors on decoded streams of data. The error correction module 215 may be physically separate from each error detection sub-module, and may also include a number of sub-modules (each of which may be time-shared by a number of error detection sub-modules).

Ex. 1001 at 5:47-61.

38. The '700 Patent continues its description of Figure 2, noting it may process received encoded data to generate a number of data streams and describes “one embodiment” of doing so:

The decoder module 205 may receive encoded data from flash memory, and process the received encoded data to generate a number of data streams. In one embodiment, the decoder module processor may buffer, de-interleave, or perform certain aspects of the decoding process, and thereby generate decoded (or partially decoded) data streams. As noted, the error detection module 210 may include a number of error detection sub-modules operating in parallel. Each (or

a subset) of the error detection sub-modules may receive a different one of the data streams, and evaluate the respective stream to determine whether a portion of the stream contains an error. Each error detection sub-module may forward the portion of the respective received stream containing an error to the error correction module 215 (bypassing the error correction module 215 with the error-free portions). The error correction module 215 corrects the received portions of the respective received streams containing an error. As noted above, the error correction module 215 may include a number of sub-modules (each of which may be time-shared by a number of error detection sub-modules).

Ex. 1001 at 6:9-28.

39. The '700 Patent describes Figure 3, reproduced below with coloring added, as “a block diagram of a flash memory decoder system illustrating error detection and correction sub-modules according to various embodiments of the invention.” Ex. 1001 at 2:9-11. More particularly, Figure 3 “is a block diagram illustrating an example of an architecture for a flash memory decoder system 300 more particularly illustrating error detection and correction sub-modules” where “The flash memory decoder system 300 may, for example, be the system 100 of FIG. 1.” Ex. 1001 at 6:29-33. As discussed below, various

components, modules and sub-modules are disclosed, including sectors 305 (such as sector 305-a, in yellow) of flash memory 110-a, decoder sub-module 310 (such as 310-a, in green), and error detection sub-module 315 (such as 315-a, in purple).

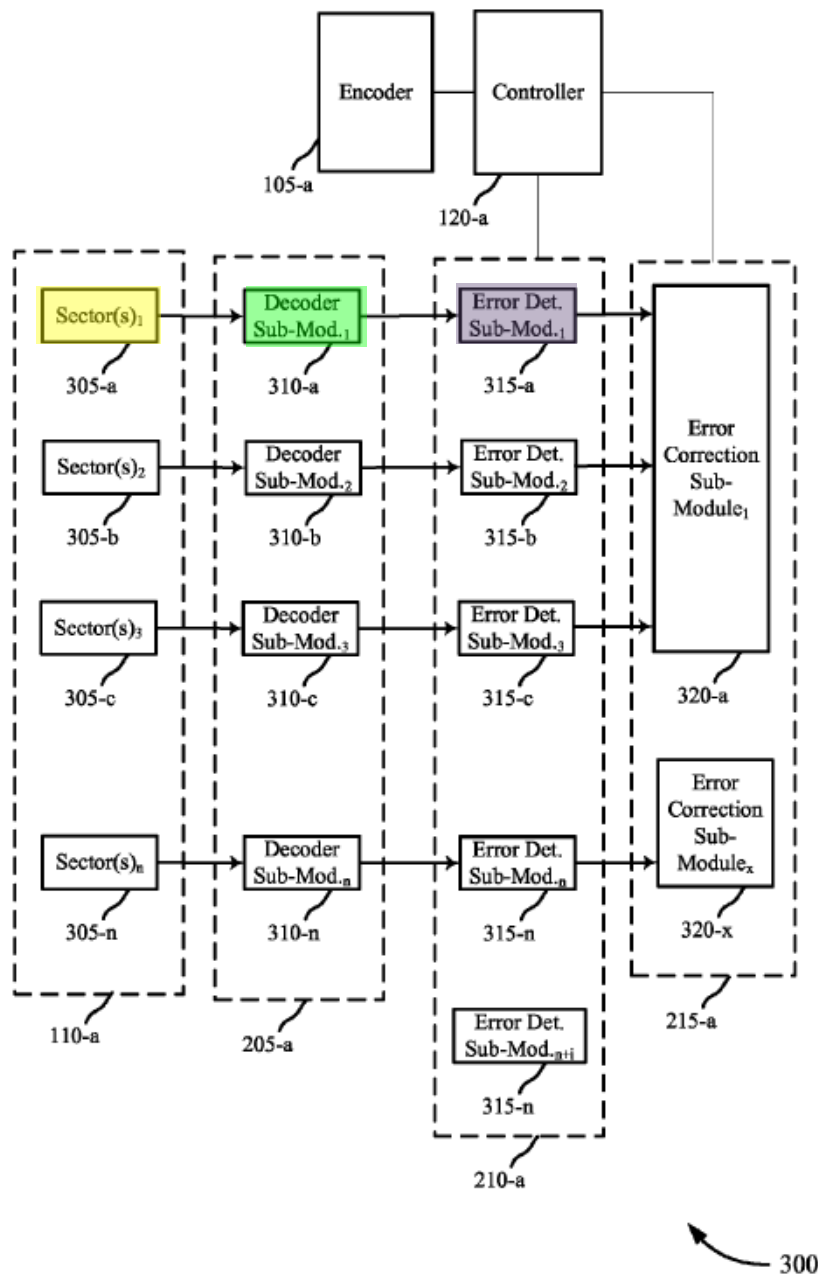


FIG. 3

40. The patent describes, in connection with Figure 3, that:

The flash memory 110-*a* includes a number of different sets of one or more active sectors 305. The decoder module 205-*a* includes a number of decoder sub-modules 310, and each decoder sub-module 310 may be associated with a different set of sectors 305. The error detection module 210-*a* includes a number of error detection sub-modules 315, and each decoder sub-module 310 may be associated with a different set of sectors 305. Thus, encoded data may be read from a set of one or more of sectors 305 (e.g., sector(s) 305-*a*) by a decoder sub-module 310 (e.g., decoder sub-module 310-*a*), and processed (e.g., buffered, de-interleaved, decoded to some extent, etc.) to generate a data stream. The data stream may be forwarded to an error detection sub-module 315 (error detection sub-module 315-*a*). Thus, encoded data from a set of one or more of sectors 305-*a* may be sent to an error detection sub-module 315-*a*. Different sets of sectors 305 may each send their respective data via different decoder sub-modules 310, as illustrated. In one embodiment, each error detection sub-module 315 is distinct from each other. In other embodiments, the decoder sub-module 310 and error detection sub-module 315 may be integrated to varying degrees.

Each error detection sub-module 315 may evaluate the respective stream to determine whether a portion of the stream contains an error. Each error detection sub-module 315 may forward the portion of the respective received stream containing an error to the error correction module 215-*a* (bypassing the error correction module 215 with the error-free portions). The error correction module 215-*a* includes a number of error correction sub-modules 320, each error correction sub-module 320 responsible for handling the errors from a number of error detection sub-modules 315 (e.g., error correction sub-module 320 may correct errors from error detection sub-modules 315-*a*, -*b*, and -*c*). Error correction sub-modules 320 may correct the received portions of the respective received streams containing an error. Each error correction sub-module 320 may receive portions of streams with errors, and these portion may have been detected in parallel.

Controller 120-*a* may perform a number of functions in the decoder process. Controller 120-*a* may dynamically modify the error detection sub-modules 315 and the error correction sub-modules 320 assigned to sectors 305 of flash memory 110-*a*. The controller 120-*a* may selectively power-up one or more of the error correction sub-modules 320 responsive to a monitored age or error rate associated with the flash memory 110-*a*. The controller 120-*a* may perform this monitoring on a per-sector basis, and the

error correction sub-modules 320 may be powered-up to serve the sectors 305 where the error rates exceed a threshold. The controller 120-*a* may adapt the coding rate to be used by the encoder responsive to a monitored age or error rate associated with the flash memory 110-*a*. The controller 120-*a* may perform this monitoring on a per-sector basis, and the controller 120-*a* may adapt the coding rate on a per sector basis.

Ex. 1001 at 6:39-7:26.

VII. SUMMARY OF PROSECUTION HISTORY

41. I have reviewed the prosecution history of the '700 Patent (Exhibit 1002) and provide the following summary.

42. The '700 Patent was filed on August 18, 2010 as Patent Application No. 12/858,510 (the “510 Application”) (filed August 18, 2010), and claim priority to U.S. Patent Provisional Application No. 61/234,911 (the “911 Provisional”) (filed August 18, 2009). Ex. 1002 at 1 (Application at page 1), 32 (Acknowledgement Receipt).

43. This information is also shown in the Cover Page of the '700 Patent:

(21) Appl. No.: **12/858,510**

(22) Filed: **Aug. 18, 2010**

(65) **Prior Publication Data**

US 2011/0047442 A1 Feb. 24, 2011

Related U.S. Application Data

(60) Provisional application No. 61/234,911, filed on Aug. 18, 2009.

44. The application was originally filed with 25 claims. Ex. 1002 at 15-19 (Application at pages 15-19) (Claims)).

45. In an office action dated January 31, 2013, the examiner rejected application claims 1, 2, 13-15, and 17 as being anticipated by U.S. Patent Application Pub. No. 2005/0138521 to Suzuki (“Suzuki”). Ex. 1002 at 143-150. In the same office action, the examiner objected to claims 3-12, 16 and 18-25 “as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims. Ex. 1002 at 148 (Office Action at page 5).

46. In response, on April 29, 2013, the applicant argued that Suzuki did not anticipate the pending claims. Ex. 1002 at 162-174. As to independent claims 1, 15 and 17, the applicant asserted:

35 U.S.C. § 102 Rejection of Claims 1, 2, 13-15 and 17

Claims 1-2, 13-15, and 17 are rejected under 35 U.S.C. §102(b) as allegedly being anticipated by U.S. Patent Publication No. 2005/0138521 to Suzuki et al. (hereinafter “Suzuki”).

Claims 1, 15, and 17

Claim 1 recites a flash memory decoder comprising a decoding module, an error detection module, and an error correction module. The decoding module is configured to decode received encoded data to generate a plurality of partially decoded data streams. The error detection module comprises a plurality of error detection sub-modules operating in parallel, where each error detection sub-module is configured to receive a different one of the partially decoded data streams, and to detect whether a portion of the respective received stream contains an error. The error correction module, which is physically separate from the error detection

module, is configured to correct received portions of the respective received streams containing an error. Applicants submit that at least some of these elements are not disclosed in Suzuki.

Suzuki is directed to a forward error correction (FEC) decoder that extracts FEC parameters encoded into a signal stream to adaptively change operational parameters. See Abstract. Suzuki's receiver includes an FEC chain decoding functional block and a parameter extraction functional block. See para. [0016]. The FEC chain decoding functional block decodes FEC blocks and detects if an FEC block is a Control Packet (CP) that includes FEC parameters. See Abstract, para. [0016]. The FEC chain decoding functional block is configured for the CP format, and all FEC blocks are decoded as being CPs, such that an error will be detected and flagged, by setting an error flag high, if the FEC block is not a CP. See para. [0140]-[0142]; FIG. 18B. If the error flag is set low (indicating an actual CP is detected/decoded), the parameter extraction functional block then processes the CP to extract the FEC parameters from the CP. See para. [0022] and [0140]-[0142]; FIG. 18B. The extracted FEC parameters are later used by the FEC chain decoding functional block for decoding subsequent FEC blocks. See para. [0017].

The Office Action relies on Suzuki at paragraphs [0122]-[0123] for the feature of the decoding module being configured to decode received encoded data to generate a plurality of partially decoded data streams. Paragraphs [0122]-[0123] of Suzuki refer to FIG. 13, which illustrates the functionality of the FEC decoding chain (i.e., the FEC chain decoding functional block). The FEC decoding chain of FIG. 13 includes, in series, a turbo decoder, a de-interleaver, a Reed Solomon (RS) decoder, a descrambler, and an MPEG packetizer. Paragraphs [0122]-[0123] describe a demodulator receiving an input signal and generating a Queue Block (QB) stream including Turbo Blocks (TBs). However, the cited paragraphs of Suzuki only discuss demodulating a single stream of data – the QB stream. This single QB stream is processed serially by the functional blocks of the FEC decoding chain of FIG. 13. In contrast, the claimed decoding module is configured to decode received encoded data to generate a plurality of partially decoded data streams. Thus, not only does the cited portion disclose demodulating versus the claimed decoding, only a single stream is disclosed versus the claimed plurality of streams. Accordingly, at least this feature of claim 1 is not shown in Suzuki.

Moreover, the claimed error detection module is also not disclosed in Suzuki. The claimed error detection module comprises a plurality of error detection sub-modules

operating in parallel, where each error detection sub-module is configured to receive a different one of the partially decoded data streams, and to detect whether a portion of the respective received stream contains an error. Thus, the multiple error detection sub-modules perform the same function on different streams. For the claimed error detection module and its functions, the Office Action relies on paragraphs [0123] and [0022] of Suzuki. As discussed above, paragraph [0123] describes the demodulator demodulating the single QB stream. Paragraph [0022] discusses the FEC decoding chain using an error flag when decoding/detecting a CP (Control Packet), and in response to the error flag setting, the CP processor extracting FEC parameters from the CP. However, the cited portions of Suzuki fail to disclose the claimed error detection module. The claimed error detection module comprises a plurality of error detection sub-modules performing the same functions and operating in parallel, whereas in Suzuki, the various functional blocks (e.g., the demodulator, blocks of the FEC decoding chain, the CP processor) perform different functions and operate serially. Accordingly, the features of the error detection module are also not shown in Suzuki.

Furthermore, the claimed error correction module is also not disclosed in Suzuki. The claimed error correction module, which is physically separate from the error detection module, is configured to correct received portions of the respective received streams containing an error. For the claimed error correction module, the Office Action relies on paragraph [0142] of Suzuki. As discussed above, paragraph [0142], which refers to FIG. 18B, describes the FEC chain decoding functional block decoding all FEC blocks as being CP, such that if an FEC block is not a CP, an error will be detected and flagged by setting the error flag high. If the FEC block is an actual CP, the error flag will be set low. A low error flag directs the CP processor (i.e., the parameter extraction functional block) to extract the FEC parameters from the CP. However, the cited portions of Suzuki fail to disclose the claimed error correction module. The error correction module is configured to correct received portions of the respective received streams containing an error, whereas in Suzuki, an error (indicated by the error flag set high) is not actually corrected by the CP processor, but simply indicates that the CP processor should not attempt to extract FEC parameters from the non-CP FEC block. Moreover, the claimed error correction module is physically separate from the claimed error detection module, but the cited portions of Suzuki teach conventional turbo decoding and RS decoding, wherein error detection and error correction for the respective FEC code is not performed by modules which are

physically separate. Accordingly, these features of the error correction module are also not shown in Suzuki.

For the above reasons, at least these elements of claim 1 are not disclosed by the relied-upon portions of Suzuki. Therefore, claim 1 is patentable over Suzuki. Applicants respectfully request withdrawal of the rejection and allowance of claim 1.

Claims 15 and 17 recite features similar to those discussed above with reference to claim 1. The Office Action rejected claims 15 and 17 under similar rationales as applied against claim 1. For at least similar reasons as those stated with respect to claim 1, claims 15 and 17 are patentable over Suzuki. Applicants respectfully request withdrawal of the rejections and allowance of claims 15 and 17.

Ex. 1002 at 170-173 (Response at pages 9-12).

47. The examiner then allowed the pending claims. Ex. 1002 at 178-192 (Notice of Allowance).

VIII. PRIORITY DATE FOR CHALLENGED CLAIMS

48. I have been informed and understand that, in order to be entitled to priority of an earlier filed application (for example, priority to a provisional application), the patentee bears the burden to show that every claim limitation of a claim is described in an earlier filed application.

49. I have seen no evidence that the Patent Owner has shown every claim limitation in any of the claims I analyzed is described in the provisional application to which the '900 Patent claims priority on its face, the '911 Provisional (Ex. 1007).

50. I note that a review of the '911 Provisional reveals that it does not disclose at least the claimed “decoding module,” the “error detection module,” the “error detection sub-module,” or the “error correction module.”

51. I also note that no figure or corresponding description found in the '700 Patent that may relate to at least the above limitations are found in the '911 Provisional. *See generally* Ex. 1007.

52. In fact, it is noteworthy that none of the figures or corresponding descriptions in the '700 Patent are included in the '911 Provisional (*e.g.*, the “block diagrams” of Figures 1, 2, and 3 of the '700 Patent, as shown and described above in Section VI, as well as the “block diagram” of Figure 4 of the '700 Patent). It is also noteworthy that the “flowcharts” as shown in Figures 5-7 of the '700 Patent (along with their corresponding descriptions) are not included in the '911 Provisional. Instead, the '911 Provisional includes four Figures (numbered 1 through 4) depicting various “graphs”. Ex. 1007 at ¶¶14-28 (describing the “graphs” of Figures 1-4).

IX. CLAIM CONSTRUCTION

53. I have been informed and understand that the United States Patent and Trademark Office interprets claim terms in an *inter partes* review proceeding under the same claim construction standard that is used in a United States federal court. I understand that under this standard, the meaning of claim terms is

considered from the viewpoint of one of ordinary skill in the art at the time of the alleged invention.

54. I have been informed and understand that claim terms are generally given their plain and ordinary meaning as understood by one of ordinary skill in the art in light of the specification and the prosecution history pertaining to the patent. I have been informed and understand, however, that claim terms are generally not limited by the embodiments described in the specification.

55. I have been informed and understand that in addition to the claims, specification, and prosecution history, other evidence may be considered to ascertain the meaning of claim terms, including textbooks, encyclopedias, articles, and dictionaries. I have been informed and understand that this other evidence is often less significant and less reliable than the claims, specification, and prosecution history.

56. In comparing the claims of the '700 Patent to the prior art, I have carefully considered the '700 Patent and its file history based upon my experience and knowledge in the relevant field (discussed in Section II above and in my CV, attached to my declaration). As relevant to determining the validity of the claims of the '700 Patent in this proceeding, the proper interpretation of the claim terms of the '700 Patent is generally consistent with the terms' plain and ordinary meaning,

as a POSITA would have understood them, as of the time of the invention of the '700 Patent.

X. OVERVIEW OF STATE OF THE ART

57. As evidenced by the prior art and state of the art discussed in this Section X and below in Section XI, it was well known and commonplace to a POSITA prior to the '700 Patent that: Flash memory is a noisy storage medium that requires error correction coding (ECC), and that ECC methods considered at that time included at least Hamming, BCH, Reed-Solomon, and Low Density Parity Check codes.

58. ECC refers to mathematically adding a controlled amount of redundancy to original data before being stored, such that this redundancy can be used to recover from errors in the device. The simplest form of adding such a redundancy is via *repetition coding*: in this approach, a single bit '0' is repeated say three times to yield the codeword '000'. Likewise a bit '1' is repeated three times to yield the codeword '111'. This scheme is referred to as 3x repetition coding. When ECC is used in a storage system (including Flash memories), codewords are stored instead of the original raw data. Suppose by way of an example that a bit is flipped with probability 0.01 (1%). Then, in the absence of any ECC, on average 1% of the stored data is retrieved erroneously; '0' as '1' or '1' as '0'. This is of course not acceptable for practical storage systems. Now

suppose that instead, 3x repetition coding is used. Suppose that *majority decoding* is used to decode the retrieved 3-bit sequence: if there are more 0's than 1's in the retrieved sequence, declare that a bit '0' was the original data; if there are more 1's than 0's in the retrieved sequence, declare that a bit '1' was the original data – since we are using 3x repetition coding, majority is always uniquely specified. For example, retrieved sequences '000', '001', '010', '100' would all be decoded as '0', and retrieved sequences '111', '110', '101', '011' would all be decoded as '1.'

59. We observe that any single bit flip (out of the three bits) can be recovered from using majority decoding. We also observe that a decoding error occurs if more than one bit is flipped. The probability of a decoding error is $3*(0.01)*(0.01)*(0.99)+(0.01)*(0.01)*(0.01) = 0.000298$. This is over a 33-fold *decrease* (0.01 divided by 0.000298) in the probability of erroneous retrieval of the original data compared to the uncoded case.

60. While there is a clear benefit in introducing ECC to reduce this probability of error, the price paid is added storage: instead of storing one bit of the original data, we now store three bits. The *code rate* represents the ratio of the number of bits in the original data to the total number of coded bits. For the 3x repetition coding, the code rate is 1/3.

61. The rule of mapping bits to codewords, here: $0 \rightarrow 000$ and $1 \rightarrow 111$, can be succinctly described by $m * G = c$ where m is either bit '0' or '1' and G is 1

x 3 matrix $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$. Here c is the resultant codeword: '000' for $m='0'$ and '111' for $m='1'$. Matrix G is referred to as the *generator matrix*.

62. The mathematical discipline of ECC is concerned with constructing codes that will simultaneously have good code rate properties and good error correction capabilities. This task then amounts to identifying appropriate generator matrices of size $k \times n$ that will encode a string of k bits into a codeword of length n bits such that the resultant set of codewords (the set is referred to as the *code*) will possess desirable error correction properties.

63. Given the fundamental nature of ECC in practical communication and storage systems, much is known about these methods. For example, Hamming codes were invented in 1950 by Richard Hamming and have the property that they can correct one bit of error, where the code parameters are k (the length of the string of the original data; this original data is also called a *message*) $= 2^r - r - 1$ and n (the length of the codeword) $= 2^r - 1$, for some positive integer r . Specification of a matrix G for a Hamming code is done via a parity check matrix H such that $G \cdot H^T = 0$. Here 'T' denotes the transpose operation which exchanges rows and columns in a matrix. Matrix H contains as columns all distinct non-zero r -tuples of bits. Once such a matrix H is constructed, matrix G follows using operations on matrices. Hamming codes are a typical choice for ECC when a single bit error correction is required.

64. Since the early invention of Hamming codes that can correct one error per codeword, many other codes have been proposed that can correct more than one error. Popular and well-known codes include Reed-Solomon codes and Bose-Chaudhuri-Hocquenghem (BCH) codes. For BCH codes, invented in 1959, the code construction amounts to representing each message as a polynomial $m(x)$ and its corresponding codeword as a polynomial $c(x)$ that is a multiple of $m(x)$: $c(x) = m(x) * g(x)$. Here $g(x)$ is what is referred to as the *generator polynomial*, and the rows of the generator matrix G are simply cyclic shifts of the coefficients of the generator polynomial $g(x)$. One convenient property of BCH codes is that they offer a precise control over the number of errors (including more than one error) the code can correct, and this number dictates the size and content of matrices G and H . Reed-Solomon codes are conceptually closely related to BCH codes in terms of the polynomial representation, but possess some further restrictions and properties (i.e., Reed-Solomon codes make certain restrictions in terms of how data as a group of bits is organized into symbols, and are a preferred choice for the correction of a burst of errors.)

65. Since Hamming, BCH and Reed-Solomon codes had already been known for decades, and ECC had already been routinely deployed in hard disk drives, it was natural that they be considered for Flash memories as well.

References dating back to the period prior to August 2009 on ECC for Flash

memories include but are not limited to single bit ECC [Ex. 1014], BCH codes [Ex. 1017; Ex. 1015], Reed-Solomon codes [Ex. 1016], and LDPC codes [Ex. 1018] for different types of Flash memories, including both SLC and MLC memories.

66. From my own personal knowledge and the representative references I have reviewed and cited herein, ECC (which includes forward error correction or “FEC”) was commonly applied to improve the performance and longevity of Flash memories and considerable research and product development had already taken place by August 2010 and August 2009 on how to improve ECC for Flash.

67. All of the claimed limitations were known within the state of the art by a POSITA.

XI. ANALYSIS OF THE PATENTABILITY OF ’700 PATENT CLAIMS

A. WU ANTICIPATES CLAIMS 1, 4, 8, 15, 17, AND 19.

68. In my opinion, Wu discloses the limitations of claims 1, 4, 8, 15, 17 and 19, arranged as in the claims. Thus, as I explain, it is my opinion that Wu anticipates these claims.

1. Claim 1

1[preamble] “A flash memory decoder comprising”

69. Regardless of whether the preamble is found to limit claim 1, as I discussed below in connection with the limitations of claim 1, Wu discloses the claimed “flash memory decoder.”

70. Wu is directed to a multi-channel memory apparatus utilizing an efficient ECC encoding and decoding scheme (Ex. 1004 at ¶0002) and discloses, in connection with Figure 8, reproduced below, “a multi-channel apparatus 800” including, for example, (i) host interface 101, (ii) data buffer 821, (iii) a plurality of ECC engines 822-0, 822-1 to 822-M, (iv) scrambler 103, (v) multiple memory access module 104, and (vi) memory devices 105-0 to 105-N. Ex. 1004 at ¶0033. These components are depicted in Figure 8 of Wu:

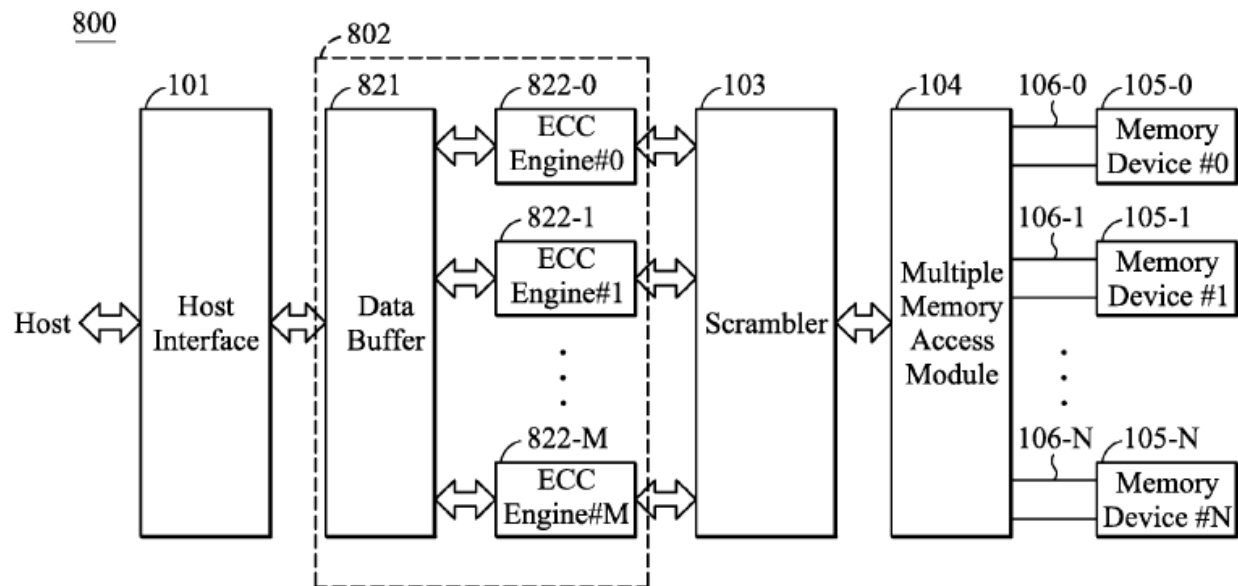


FIG. 8

71. Wu also discloses that the disclosed “memory devices” may be, among other things, flash memory; for example, according to Wu, “the memory device 105-0 to 105-N may be a NAND or NOR type flash, a solid state disk, a

Static Random Access Memory (SRAM), a Dynamic Random Access Memory (DRAM) . . . etc.” Ex. 1004 at ¶0024 (ellipses are found in Wu’s original disclosure).

72. Wu further discloses that “the error correcting module 802 may comprises an ECC engine 822-0 and a plurality of sub-ECC engines 822-1 to 822-M coupled in parallel with the ECC engine 822-0” and “The operations of the sub-ECC engines 822-1 to 822-M for performing error correction code encoding and decoding on the data to be stored in the memory devices are similar to the operations of the ECC engine 822-0.” Ex. 1004 at ¶0033.

73. For the “detailed description” of the components of Figure 8 (*e.g.*, “the host interface 101, the data buffer 821, the ECC engine/sub-ECC engines 822-0 to 822-M, the scramblers 103, the multiple memory access module 104, and the memory devices 105-0 to 105-N as shown in FIG. 8”), Wu references back to Figures 1 and 2 and the corresponding descriptions, and thus notes that “further descriptions are omitted here [in the Figure 8 discussion] for brevity.” Ex. 1004 at ¶0033; *see also* ¶¶0022-26 (discussing Figures 1 and 2); Figs. 1, 2.

74. For example, Wu discloses, in Figure 1, reproduced below, “a multi-channel memory apparatus 100 may be located within and communicate with a host device. The host device may be a computing or media device, as an example, a notebook, a cellular phone, a gaming device, a multimedia player, a Global

Positioning System (GPS), or others. The multi-channel memory apparatus 100 comprises a host interface 101, an error correcting module 102, a scrambler 103, a multiple memory access module 104, a plurality of storage channels and a plurality of memory devices 105-0, 105-1 to 105-N. Each storage channel 106-0 to 106-N is coupled to a memory device (such as memory device 105-0, 105-1 to 105-N) for storing the data. The host interface 101 is arranged to communicate with the host device. As an example, the host interface 101 may interpret host commands and receive data from or transmit data to the host device depending on the data transferring directions (e.g. write or read directions). The error correcting module 102 is shared by the storage channels and is arranged to perform error correction code encoding on the data to be stored into the memory devices and perform error correction code decoding on the data read out from the memory devices, depending on the data transferring directions (e.g.

write or read directions).” Ex. 1004 at ¶0022.

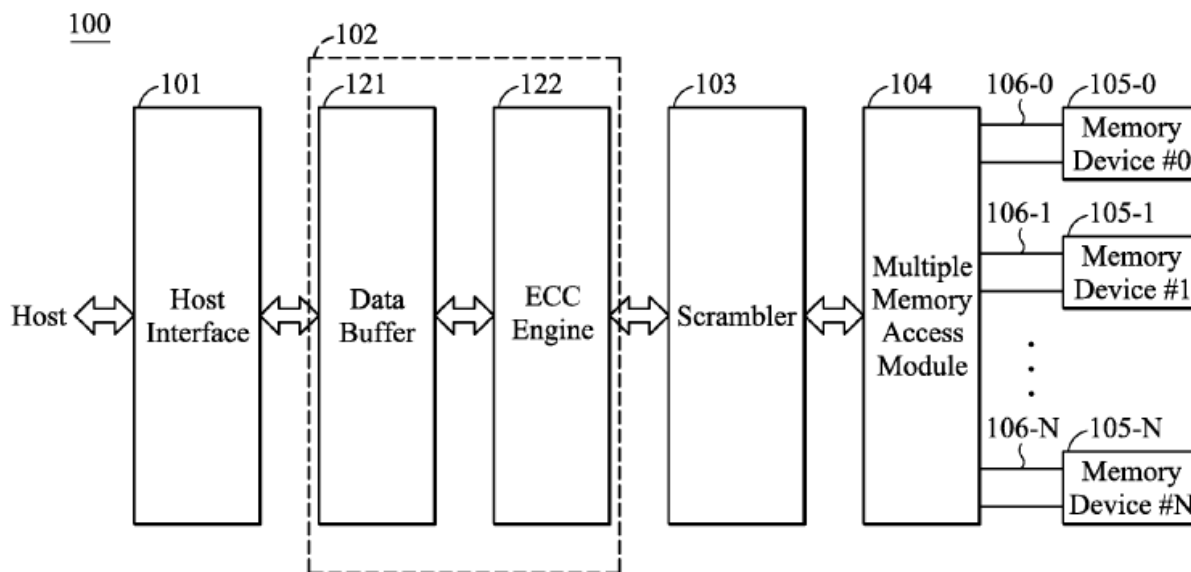


FIG. 1

75. Thus, as discussed above, Wu discloses that the plurality of ECC engines 822-0, 822-1 to 822-M of ECC correcting module 802 perform error correction code encoding and decoding on the data stored in the memory devices depending on the data transferring directions (e.g., write or read directions). Ex. 1004 at ¶¶0022, 0033.

76. Wu also describes its “scrambler 103” that “may be implemented in a scrambler module (not shown) and is arranged to perform intra-channel disarrangement for the data to be stored into the memory devices and rearrange the scrambled data read out from the memory devices in a reverse manner depending

on the data transferring directions (e.g. write or read directions).” Ex. 1004 at ¶0023; *see also* ¶0033.

77. A POSITA would therefore have understood that Wu discloses the claimed “flash memory decoder.” For example, as disclosed in Wu, a POSITA would have understood that Wu discloses a “flash memory” (yellow below) “decoder” of the flash memory (green below).

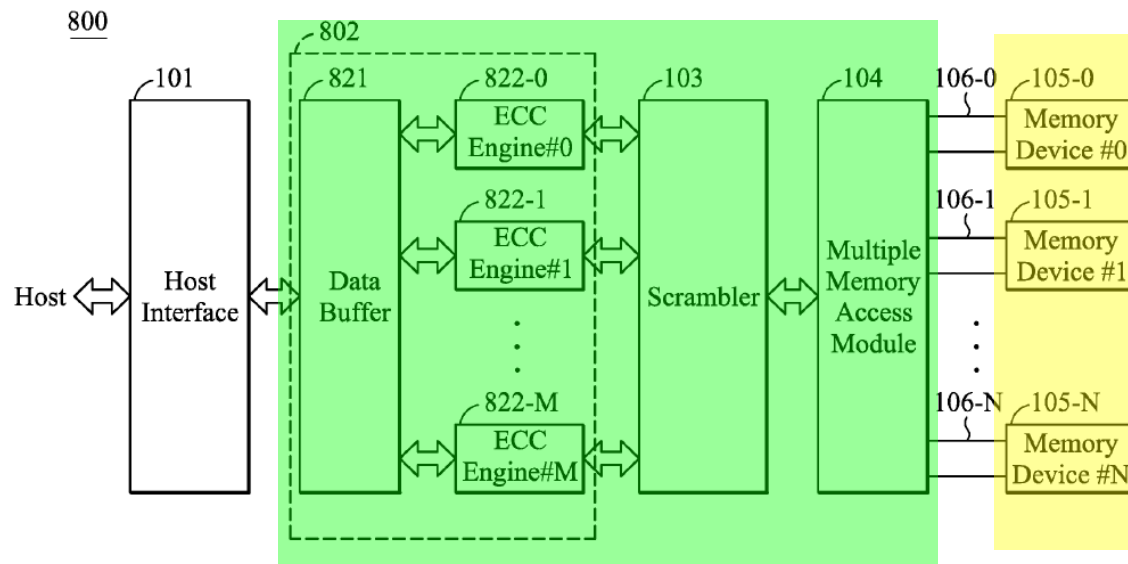


FIG. 8

1[a] “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams;”

78. Wu discloses the claimed “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams.”

79. As discussed in Figure 8, reproduced and discussed above with regard to 1[preamble], Wu's "flash memory decoder" includes a "decoding module," that is the combination of the multiple memory access module 104 and scrambler 103, which according to Wu, "may be implemented in a scrambler module." Ex. 1004 at ¶¶0023-26, 0033, Figs. 1, 2, 8.

80. Wu discloses that, "The multiple memory access module 104 is coupled between the storage channels and the error correcting module 102 and arranged to perform multiple access control of the storage channels for the error correcting module 102" and provides "As an example, [that] the multiple memory access module 104 multiplexes the data received simultaneously from different memory devices 105-0 to 105-N to the error correcting module 102." Ex. 1004 at ¶0024. That is, for example, as Wu explains, when reading the memory devices, "the multiple memory access module 104 multiplexes the data received simultaneously from different memory devices 105-0 to 105-N...." Ex. 1004 at ¶0024; *see also* ¶0027 and Fig. 3 (reproduced below), ¶0029 ("The multiple memory access module first schedules the flow queue of the read operation and reads data from the memory device (Step S501)."), Fig. 5.

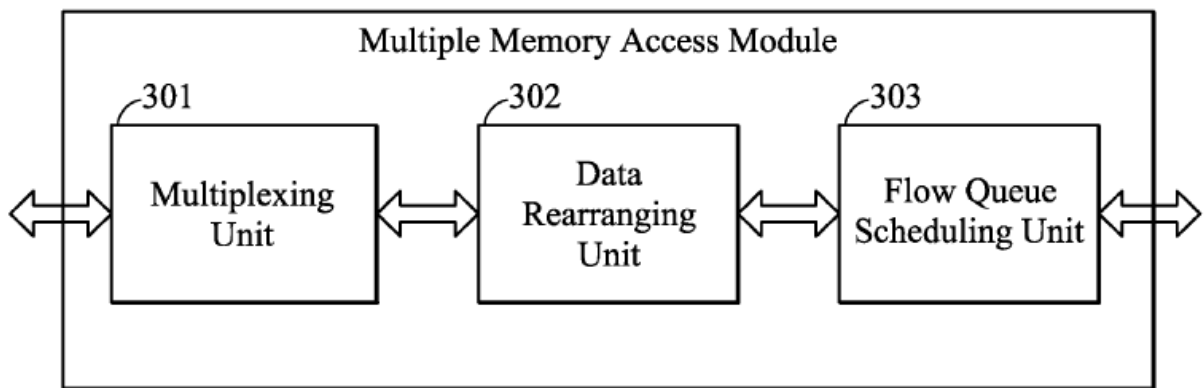


FIG. 3

81. Wu further discloses that the data that is retrieved from the memory devices (105-0 to 105-N) over storage channels (106-0 to 106-N) is (i) error correction coded by one or more ECC engines (Ex. 1004 at ¶¶0022, 0033); (ii) intra-channel disarranged by a scrambler (such as 103) (Ex. 1004 at ¶0023); and (iii) inter-channel disarranged by multiple memory access module (such as 104) (Ex. 1004 at ¶0027). For example, Wu discloses that:

- “Each storage channel 106-0 to 106-N is coupled to a memory device (such as memory device 105-0, 105-1 to 105-N) for storing the data. The host interface 101 is arranged to communicate with the host device. As an example, the host interface 101 may interpret host commands and receive data from or transmit data to the host device depending on the data transferring directions (e.g. write or read directions). The error correcting module 102 is shared by the storage channels and is arranged to perform error correction code encoding on the data to be stored into the memory devices and perform error

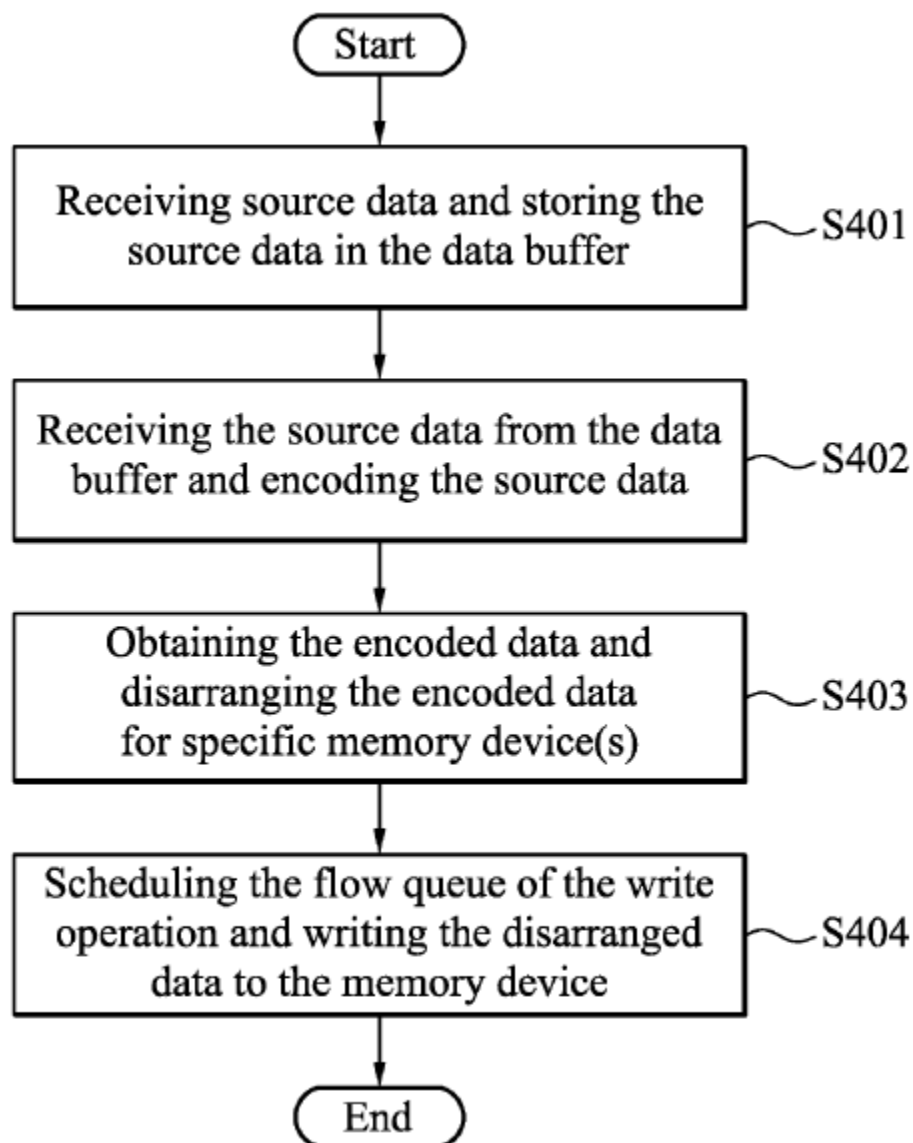
correction code decoding on the data read out from the memory devices, depending on the data transferring directions (e.g. write or read directions).” Ex. 1004 at ¶0022.

- “FIG. 8 shows a multi-channel memory apparatus 800 according to another embodiment of the invention. In the embodiment, the error correcting module 802 may comprises an ECC engine 822-0 and a plurality of sub-ECC engines 822-1 to 822-M coupled in parallel with the ECC engine 822-0. The operations of the sub-ECC engines 822-1 to 822-M for performing error correction code encoding and decoding on the data to be stored in the memory devices are similar to the operations of the ECC engine 822-0. Therefore, the multi-channel memory apparatus 800 may be regarded as comprising a plurality of ECC engines 822-0, 822-1 to 822-M.” Ex. 1004 at ¶0033.
- “The scrambler 103 may be implemented in a scrambler module (not shown) and is arranged to perform intra-channel disarrangement for the data to be stored into the memory devices and rearrange the scrambled data read out from the memory devices in a reverse manner depending on the data transferring directions (e.g. write or read directions). The intra-channel disarrangement is a disarrangement operation performed within a channel so as to scramble the data of the same codeword of the ECC engine. The scrambler 103 scrambles the data so as to randomly distribute the data ‘0’ and ‘1’ in adjacent locations. Thus, interference introduced by the adjacent data of continuous ‘0’ or ‘1’ within the memory device 105-0 to 105-N is reduced. Note that because the scrambling process is a linear operation, the scrambler (scrambler module) may be located anywhere in the data transferring paths (e.g. write or read paths). In addition, according to another embodiment of the invention, the scrambler module may also comprise a plurality of scramblers to deal with the scrambling of the multiple storage channels (which will be discussed in detail in the following paragraphs).” Ex. 1004 at ¶0023.

- “The multiple memory access module may comprise a multiplexing unit 301, a data rearranging unit 302 and a flow queue scheduling unit 303. The multiplexing unit 301 is coupled to the error correcting module 102 or 202 and arranged to multiplex the data to be transmitted to the error correcting module, or to de-multiplex the data received from the error correcting module. According to an embodiment of the invention, the multiplexing unit 301 may multiplex the data in an order of the index of the storage channels, as an example, from the storage channel 106-0 to 106-N, or from the storage channel 106-N to 106-0. The data rearranging unit 302 is arranged to perform inter-channel disarrangement for the data received from the multiplexing unit and to be stored into the memory devices, and rearrange the data read out from memory devices through the storage channels in a reverse manner depending on the data transferring directions (e.g. write or read directions). The inter-channel disarrangement is to further enhance the disorderliness of the data by exchanging the data of different storage channels so that the same type of data may not all be stored in the same memory device. The purpose of the disarrangement operation is to randomly place the data between channels so as to further reduce inter-channel interference. The flow queue scheduling unit 303 is arranged to schedule a plurality of flow queues of the data to be stored into the memory devices and the data read out from the memory devices. As an example, the flow queue scheduling unit 303 may queue the data to be stored into the corresponding memory devices until the amount of arrived data reaches a predetermined data size, thus improving the power and time efficiency of the multi-channel memory apparatus.” Ex. 1004 at ¶0027.

82. A POSITA would have understood that these disclosures in Wu demonstrate that the data stored on the flash memory devices (105-0 to 105-N) is encoded data. Ex. 1004 at ¶0028, Fig. 4. This is further confirmed in Wu, in connection with the Figure 4 flow chart, reproduced below, which “shows a flow

chart of a write operation according to an embodiment of the invention,” describes the steps as follows: “The host interface first receives source data from the host device and stores the source data in the data buffer (Step S401). Next, the ECC engine receives the source data from the data buffer and encodes the source data (Step S402). Next, the multiple memory access module obtains the encoded data from the ECC engine and disarranges the encoded data for specific memory device(s) (Step S403). Finally, the multiple memory access module schedules the flow queue of the write operation and writes the disarranged data to the memory device (Step S404).” Ex. 1004 at ¶0028.



83. Therefore, as explained above, a POSITA would have understood that, when reading from the memory devices, the “decoding module” is “configured to: receive encoded data from the flash memory,” *i.e.*, the encoded

data received over storage channels 106-0 to 106-N from memory devices 105-0 to 105-N.

84. Wu also discloses that its “decoding module” is “configured to: ... decode the received encoded data to generate a plurality of partially decoded data streams,” as in claim 1 of the ’700 Patent. For example, Wu discloses that the multiple memory access module 104 may include a “data rearranging unit 302” which “is arranged to perform inter-channel disarrangement for the data received from the multiplexing unit and to be stored into the memory devices, and rearrange the data read out from memory devices through the storage channels in a reverse manner depending on the data transferring directions (e.g. write or read directions).” Ex. 1004 at ¶0027. Wu further discloses that “The inter-channel disarrangement is to further enhance the disorderliness of the data by exchanging the data of different storage channels so that the same type of data may not all be stored in the same memory device” and “The purpose of the disarrangement operation is to randomly place the data between channels so as to further reduce inter-channel interference.” Ex. 1004 at ¶0027.

85. Wu also discloses that a scrambler module 103 may “perform intra-channel disarrangement for the data to be stored into the memory devices and rearrange the scrambled data read out from the memory devices in a reverse manner depending on the data transferring directions (e.g. write or read

directions).” Ex. 1004 at ¶0023. Wu further discloses that the intra-channel disarrangement/arrangement operation is performed to scramble/unscramble the data of the ECC engine codewords, the error correction coded data, so as to randomly distribute the data ‘0’ and ‘1’ in adjacent locations thereby further reducing interference. Ex. 1004 at ¶0023 (“The intra-channel disarrangement is a disarrangement operation performed within a channel so as to scramble the data of the same codeword of the ECC engine. The scrambler 103 scrambles the data so as to randomly distribute the data ‘0’ and ‘1’ in adjacent locations. Thus, interference introduced by the adjacent data of continuous ‘0’ or ‘1’ within the memory device 105-0 to 105-N is reduced. Note that because the scrambling process is a linear operation, the scrambler (scrambler module) may be located anywhere in the data transferring paths (e.g. write or read paths). In addition, according to another embodiment of the invention, the scrambler module may also comprise a plurality of scramblers to deal with the scrambling of the multiple storage channels (which will be discussed in detail in the following paragraphs”).

86. As shown in Wu’s Figure 8, reproduced below, a POSITA would have understood that the data retrieved from memory devices 105-0 to 105-N and provided to ECC Engines 822-0 to 822-M corresponds to the claimed “plurality of partially decoded data streams” (yellow in Figure 8). A POSITA would also have understood that the partial decoding is disclosed by the intra-channel

rearrangement performed by scrambler 103. It is understood, for example, that scrambling corresponds to encoding and descrambling corresponds to decoding. Ex. 1006 at ¶0052 (Wan disclosing that “scrambling is considered to be a form of encoding and descrambling is considered to be a form of decoding”), ¶0062 (“By analogy to the write process discussed previously, portions of the scrambled or otherwise modified user data are read from respective locations in the memory array and descrambled using non-user data which is read from corresponding respective locations in the memory array. For example, a portion of the user data on an *i*th page in M2 may be descrambled or otherwise decoded by non-user data on an *i*th page in M1. Each word line may have one or more pages. Further, a page may be made up of sub-pages, such that a portion of the user data on a *j*th sub-page in M2 may be descrambled or otherwise decoded by non-user data on a *j*th sub-page in M1”).

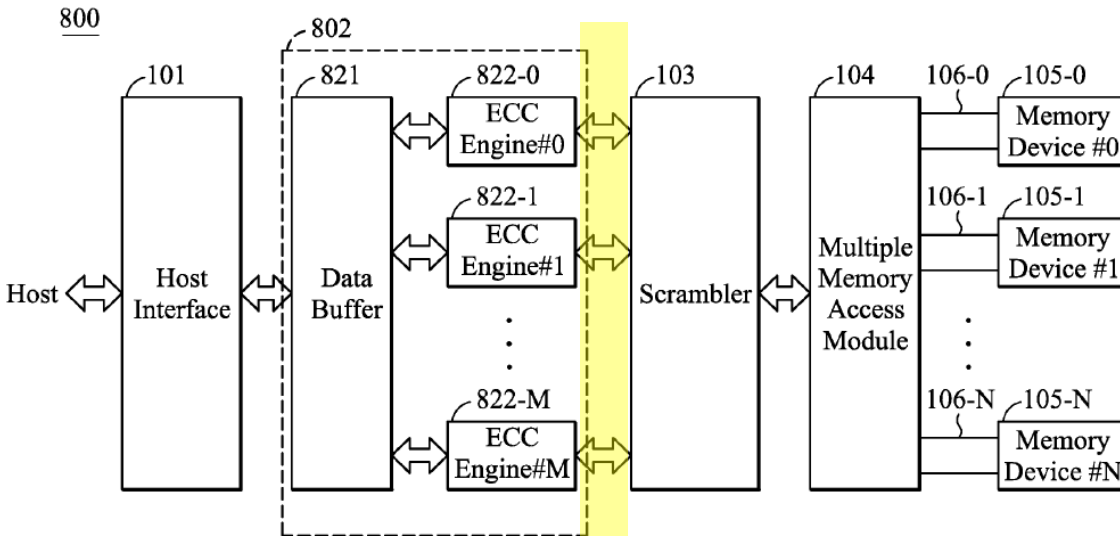


FIG. 8

87. I understand from counsel for Petitioner that Patent Owner, in the district court case filed against Petitioner, has asserted that the recited “partially decoded data streams” covers de-interleaved data streams.

88. Under such an alternative interpretation, Wu also discloses the claim limitation. In particular, such partial decoding is disclosed by the inter-channel rearrangement performed by multiple memory access module 104. Ex. 1004 at ¶0027 (quoted above and describing an inter-channel rearrangement de-interleaving process), Figs. 3 and 8. I note that although only (M+1) data streams are visible in Figure 8, reproduced above (one for each ECC engine 822-0 to 822-M), as I explain in paragraphs 97 to 104 below in connection with limitation 1[c], each of these data streams is further subdivided into (K+1) sub-streams, for a total of $(K+1) \times (M+1)$ “partially decoded data streams.” Ex. 1004 Fig. 8.

89. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams.”

1[b] “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel,”

90. Wu discloses the claimed “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel.”

91. As I have discussed, Wu discloses, in connection with Figure 8, “a plurality of ECC engines 822-0, 822-1 to 822-M.” Ex. 1004 at ¶0033. Wu also discloses that each of these ECC engines “may comprise one or more error detectors 902-0 to 902-K and an error corrector 901,” as shown in Figure 9, reproduced below, which “shows a block diagram of an ECC engine/sub-ECC engine.” Ex. 1004 at ¶0034.

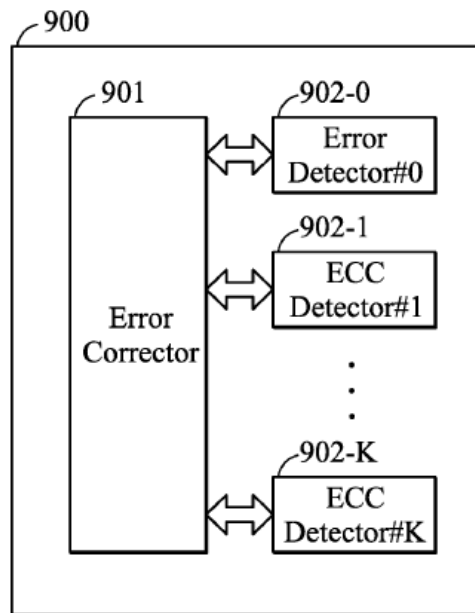


FIG. 9

92. Of note, this disclosure in Wu regarding Figure 9’s ECC engine/sub-engine also applies to Figure 8’s disclosure (as well as others). That is, per Wu, “[t]he structure of the ECC engine/sub-ECC engine 900 shown in FIG. 9 may also be applicable to the ECC engine 122 as shown in FIG. 1, the ECC engine 222 as shown in FIG. 2, the ECC engines as shown in FIG. 6 and FIG. 7 and the ECC engine/sub-ECC engines 822-0 to 822-M as shown in FIG. 8.” Ex. 1004 at ¶0034. In other words, a POSITA would have understood that the disclosure in connection with Figure 9 applies to Figure 8 (and others) such that Figure 8 and 9 are treated as the same embodiment for purposes of understanding Wu’s teachings. Accordingly, a POSITA would have understood that each ECC Engine 822-0, 822-

1 to 822-M (as shown in reference to Figure 8) comprises multiple ECC Detectors 902-0 to 902-K (as shown in reference to Figure 9 and which meet the error detection sub-module limitation).

93. Wu further discloses that “the ECC engine (such as the ECC engine 122, 222, 822-0 to 822-M and 900 . . . etc.) is arranged to perform suitable ECC algorithm, such as Reed-Solomon or Bose, Ray-Chaudhuri, Hocquenghem (BCH) encoding and decoding,” and that “The error corrector 901 is shared by the error detectors 902-0 to 902-K and arranged to correct the error(s) of the data detected by the error detectors.” Ex. 1004 at ¶0034 (ellipses original in Wu).

94. A POSITA would have understood that the error detectors 902-0 to 902-K (which per Wu, reside within each ECC engine 822-0 to 822-M) collectively constitute disclosure of the claimed “error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel.” They are shown below in yellow in Figure 8, which has been modified to with the additional of the ECC engine structure in Figure 9.

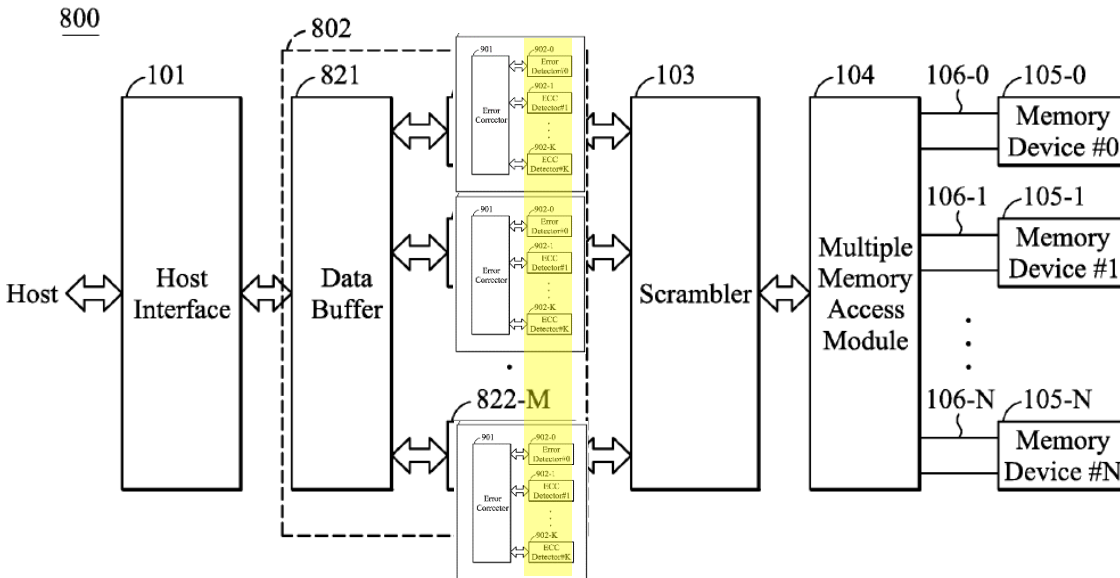


FIG. 8

95. Based on the above disclosures, a POSITA would have understood that Wu's error detectors 902 operate in parallel. This is because the modules are created and arranged in parallel. A POSITA would not understand these modules as alternatively working one-at-a-time, as this would detrimentally impact performance (*e.g.*, by delaying memory access times) and would render the inclusion of multiple error detectors, as taught by Wu, an unnecessary cost in terms of circuitry, complexity, and space, because only one detector would be operating at any given time and, concomitantly, only one detector would be needed if the detectors do not operate in parallel.

96. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed "an error detection module

communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel.”

1[c] “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module; and”

97. Wu discloses the claimed “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module.”

98. Wu discloses in its flash memory decoder (in which the ECC engines 822-0 to 822-M (depicted in Figure 8) are implemented using the ECC engine 900 structure (depicted in Figure 9)) that the error detectors 902-0 to 902-K within each ECC engine are arranged in parallel (depicted and described with respect to Figure 8, above in limitation 1[b]). Each of the (K+1) error detectors within each ECC engine, (*i.e.*, each “error detection sub-module”) receives (on a memory read operation) a portion of one of the (M+1) partially decoded data streams provided at the output of the “decoder module,” *i.e.*, at the output of scrambler 103. A POSITA would have understood that these portions of partially decoded data

streams are themselves the claimed $(K+1) \times (M+1)$ “plurality of partially decoded data streams.”

99. Wu also discloses that the error detectors 902-0 to 902-K are arranged to “detect error(s) in the data read out from the memory devices.” Ex. 1004 at ¶0034 (“The error corrector 901 is shared by the error detectors 902-0 to 902-K and arranged to correct the error(s) of the data detected by the error detectors.”).

100. Accordingly, a POSITA would have understood that Wu discloses the claimed “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; [and] detect whether a portion of the respective received stream contains an error,” as required by the first part of limitation 1[c] of claim 1 of the ’700 Patent.

101. Wu also discloses that the error correctors 901 are “shared by the error detectors 902-0 to 902-K and arranged to correct the error(s) of the data detected by the error detectors.” Ex. 1004 at ¶0034. A POSITA would have understood that data containing errors is sent to these error correctors 901, which perform a variety of known decoding, such as, “Reed-Solomon or Bose, Ray-Chaudhuri, Hocquenghem (BCH) ... decoding ... to correct the error(s) of the data detected by the error detectors.” Ex. 1004 at ¶0034, Fig. 9.

102. Accordingly, a POSITA would have understood that Wu further discloses the “each error detection sub-module configured to: ... forward the

portion of the respective received stream containing an error to an error correction module,” as required by the second part of limitation 1[c].

103. I further note that limitation 1[d] (discussed below) does not preclude error-free data from also being forwarded to the error corrector 901. Instead, the forwarding operation is only limited in dependent claim 13, which depends from claim 1 and specifies “wherein each error detection sub-module is configured to direct error-free portions of the respective received streams to bypass the error correction module.” This further confirms that Wu discloses limitation 1[c].

104. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module.”

1[d] “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

105. Wu discloses the claimed “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

106. As I have discussed above in the earlier limitations of claim 1, Wu discloses that each of the ECC engines 822-0, 822-1 to 822-M “may comprise one or more error detectors 902-0 to 902-K and an error corrector 901,” as shown in the “block diagram” of Figure 9, reproduced above and again below. Ex. 1004 at ¶0034. Again, as I have also discussed above, Wu discloses that Figure 9 is applicable to other Figures, including Figure 8, for example, disclosing that “the structure of the ECC engine/sub-ECC engine 900 shown in FIG. 9 may also be applicable to ... the ECC engine/sub-ECC engines 822-0 to 822-M as shown in FIG. 8.” Ex. 1004 at ¶0034; *see also* Ex. 1004 at claim 10 (“The multi-channel memory apparatus as claimed in claim 1, wherein the ECC engine comprises ... a plurality of error detectors ... and an error corrector, shared by the error detectors).

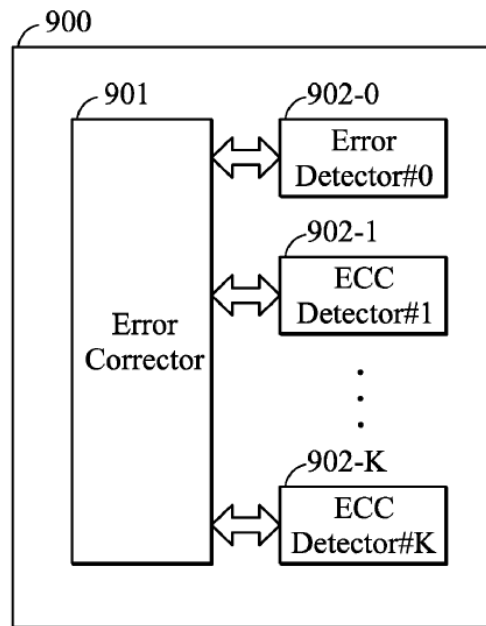


FIG. 9

107. Also as I have discussed, Wu discloses that “the ECC engine (such as the ECC engine ... 822-0 to 822-M and 900 ... etc.) is arranged to perform suitable ECC algorithm, such as Reed-Solomon or Bose, Ray-Chaudhuri, Hocquenghem (BCH) encoding and decoding,” (algorithms that commonly employ physically separate error detection, *e.g.*, syndrome computation blocks, and error correction modules) and that “The error corrector 901 is shared by the error detectors 902-0 to 902-K and arranged to correct the error(s) of the data detected by the error detectors.” Ex. 1004 at ¶0034.

108. Wu’s error correctors 901 (which reside within each ECC engine 822-0 to 822-M shown in the “multi-channel memory apparatus” of Figure 8 (Ex. 1004 ¶0019)) collectively meet the claimed “the error correction module,” that are “communicatively coupled with ... the error detection module, and configured to correct the received portions of the respective received streams containing an error.” Wu further discloses that the error correctors 901 shared by the error detectors 902-0 to 902-K and incorporated in the multi-channel memory apparatus of Figure 8 are “physically separate from” the error detectors 902-0 to 902-K. The claimed “error correction module,” which includes error correctors 901 is shown below in yellow in Figure 8, which has been modified to include the additional ECC engine structural blocks in Figure 9.

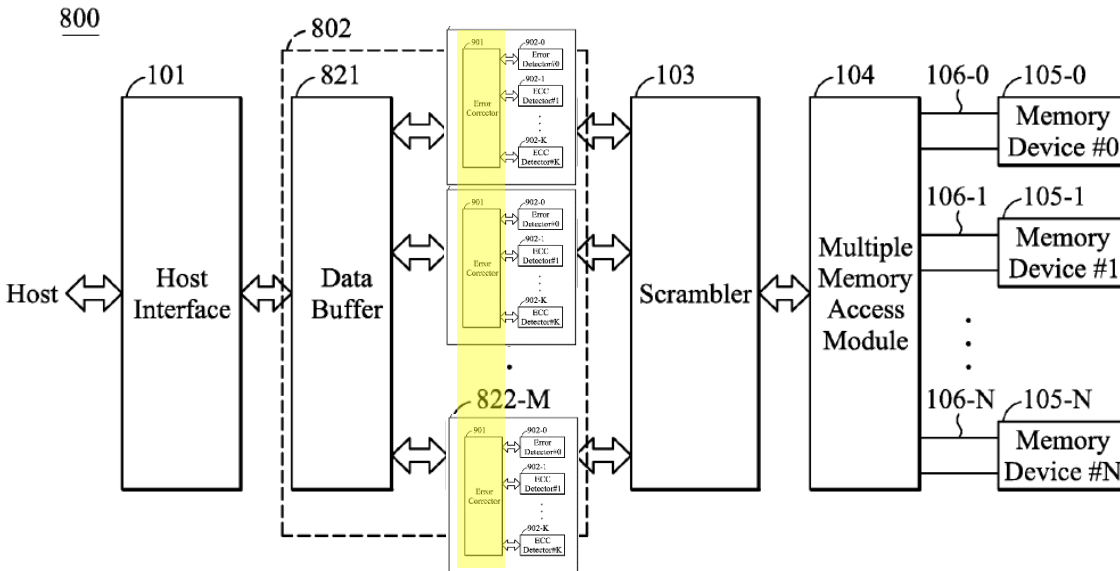


FIG. 8

109. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

110. Accordingly, because Wu discloses the claimed limitations of claim 1 of the '700 Patent, arranged as in the claim, a POSITA would have understood that Wu anticipates claim 1.

2. Claim 4: “The flash memory decoder of claim 1, wherein the error correction module comprises a plurality of error

correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.”

111. Wu discloses that “the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.”

112. As I discussed above, Wu discloses in its Figure 8, reproduced below with the structure of ECC engines 822-0 to 822-M conforming to Wu’s Figure 9, an embodiment taught by Wu, discloses the “flash memory decoder of claim 1, wherein the error correction module,” (yellow) which “comprises a plurality of error correction sub-modules,” that is error correctors 901 indicated (red boxes), “operating in parallel and fewer in number than the error detection sub-modules,” that is, detectors 902 (green). Ex. 1004 at ¶0034. As can be seen, the error correctors 901 operating in parallel are fewer in number than detectors 902.

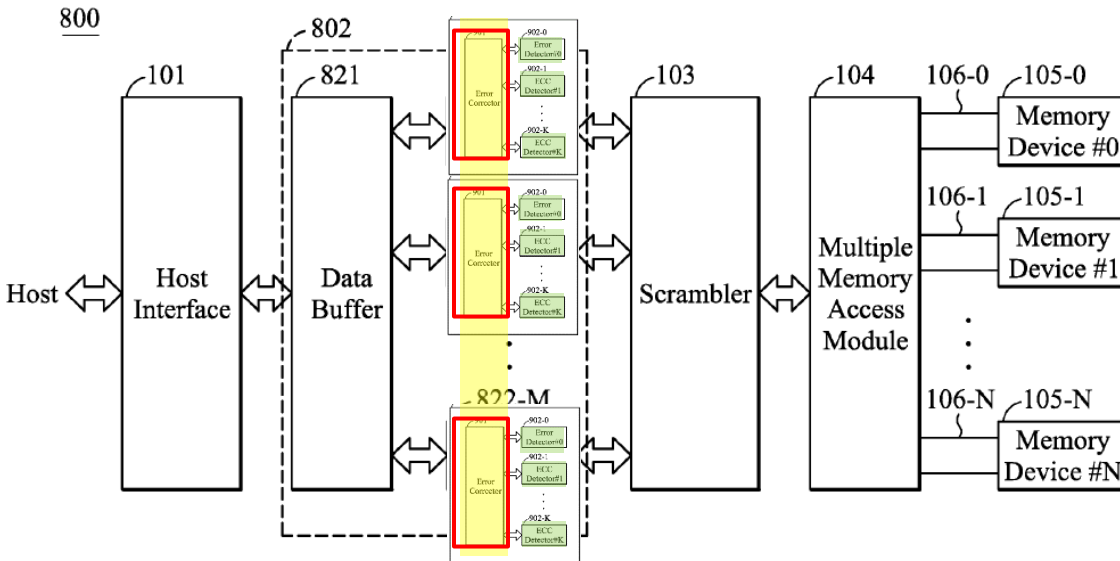


FIG. 8

113. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses that “the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.”

114. Accordingly, because Wu discloses the additional claimed limitation in claim 4 of the '700 Patent, arranged as in the claim, a POSITA would have understood that Wu anticipates claim 4.

3. Claim 8

115. I discuss below claim 8 in two parts (limitations 8[a] and 8[b]).

8[a] “The flash memory decoder of claim 4, wherein, each error correction sub-module is assigned a set of two or more error

detection sub-modules different from error detection sub-modules associated with other of the error correction sub-modules; and”

116. Wu discloses that “each error correction sub-module is assigned a set of two or more error detection sub-modules different from error detection sub-modules associated with other of the error correction sub-modules.”

117. Wu’s Figure 8, as described and annotated above in connection with claim 4 to include the exemplary ECC Engine 822 structure shown in Wu’s Figure 9, discloses “each error correction sub-module,” that is, error corrector modules 901 within ECC Engines 822-0- to 822-M, “is assigned a set of two or more error detection sub-modules,” that is, $(k+1)$ error detector modules 902 within each ECC Engines 822-0- to 822-M “different from error detection sub-modules associated with other of the error correction sub-modules.”

118. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “each error correction sub-module is assigned a set of two or more error detection sub-modules different from error detection sub-modules associated with other of the error correction sub-modules.”

8[b] “each error correction sub-module is configured to receive portions of the respective received streams containing an error detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-module.”

119. Wu also discloses that “each error correction sub-module is configured to receive portions of the respective received streams containing an error detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-module.”

120. Wu’s Figure 8 flash memory decoder embodiment includes ECC engines 822-0 to 822-M “coupled in parallel.” Ex. 1004 at ¶0033 (“the error correcting module 802 may comprises an ECC engine 822-0 and a plurality of sub-ECC engines 822-1 to 822-M coupled in parallel with the ECC engine 822-0.”). Wu further discloses that each ECC engine 822-0 to 822-M can be implemented using the structure shown in Figure 9, which includes an ECC corrector connected to error detectors 902-0 to 902-K further arranged in parallel. Ex. 1004 at ¶0034; Fig. 9 (showing parallel arrangement).

121. Wu further discloses that “The error corrector 901 is shared by the error detectors 902-0 to 902-K and arranged to correct the error(s) of the data detected by the error detectors.” Ex. 1004 at ¶0034.

122. A POSITA would have understood that these disclosures in Wu teach that “each error correction sub-module,” that error corrector modules 901 within ECC Engines 822-0- to 822-M, “is configured to receive portions of the respective received streams containing an error detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-

module,” that is, $(k+1)$ error detector modules 902 within each ECC Engines 822-0- to 822-M. I further refer back to my discussion of limitation 1[b], explaining Wu’s disclosure that error detector modules 902 operate so as to detect errors in parallel.

123. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “each error correction sub-module is configured to receive portions of the respective received streams containing an error detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-module.”

124. Accordingly, because Wu discloses the additional claimed limitations in claim 8 of the ’700 Patent, arranged as in the claim, a POSITA would have understood that Wu anticipates claim 8.

4. Claims 15 and 17

125. Wu discloses the claimed limitations in claims 15 and 17 for effectively the same reasons as claim 1 (discussed above).

126. I note that independent claims 15 and 17 are very similar to apparatus claim 1, except that claim 15 is a “system” claim and claim 17 is a “method” claim. Therefore, claims 15 and 17 are disclosed and anticipated by Wu for the effectively the same reasons that Wu disclose and anticipates claim 1.

127. I have included in the table below a comparison of the limitations of claims 15 and 17 to the limitations of claim 1. As seen in the table, only limitations 15[a] and 15[b] are not equivalently recited in claim 1. I explain below how Wu discloses these limitations as well.

Claim 15	Claim 17	Claim 1
15[preamble] A system comprising: 15[a] an encoder configured to encode data for storage on a flash memory;		
15[b] the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data; and		
15[c] a decoder, communicatively coupled with the flash memory, and configured to: 15[d] retrieve the encoded data from flash memory to generate a plurality of data streams; and <i>[I note that the above limitations, unlike 1[preamble] and 1[a], do not restrict the data streams to be of a “partially decoded” type. Thus, the disclosure of the “partially decoded” element that I identify with respect to claim 1 is not</i>	17[pre] A method comprising: 17[a] receiving encoded data from a flash memory comprising a plurality of data streams; <i>[I note that the above limitations, unlike 1[preamble] and 1[a], do not restrict the data streams to be of a “partially decoded” type. Thus, the disclosure of the “partially decoded” element that I identify with respect to claim 1 is not required to meet these claim 17 limitations.]</i>	1[preamble] A flash memory decoder comprising: 1[a] a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams;

<i>required to meet these claim 15 limitations.]</i>		
15[e] process, at each of a plurality of error detection sub-modules of the decoder operating in parallel, a different one of the plurality of data streams,	17[b] processing, at each of a plurality of error detection sub-modules operating in parallel, a different one of the plurality of data streams,	1[b] an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel, 1[c] each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams;...
15[f] wherein each error detection sub-module is configured to: detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module of the decoder; and	17[c] wherein each error detection sub-module is configured to: detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module; and	1[c] each error detection sub-module configured to:... detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module; and
15[g] correct, with the error correction module physically separate from the error detection module, the forwarded portions of the respective received streams containing an error.	17[d] correcting, with the error correction module physically separate from an error detection module, the forwarded portions of the respective received streams containing an error.	1[d] the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.

15[a] “an encoder configured to encode data for storage on a flash memory,”

15[b] “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data; and”

128. Wu discloses both the claimed “an encoder configured to encode data for storage on a flash memory” and “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data.”

129. As I have discussed above (including in connection with claim 1), Wu discloses a multi-channel memory apparatus with an error correcting module 802 comprising “a plurality of ECC engines 822-0, 822-1 to 822-M” (in Figure 8, reproduced below) that “perform[] error correction code encoding and decoding on the data to be stored in the memory devices.” Ex. 1004 at ¶0033, Fig. 8.

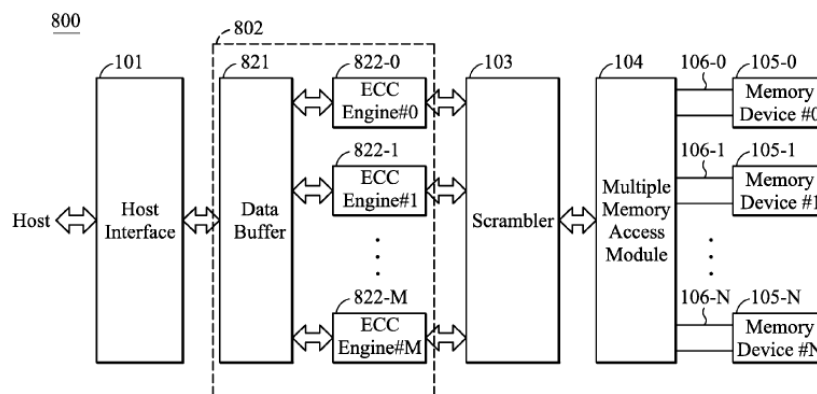


FIG. 8

130. Wu discloses—in connection with Figure 8—that “operations of the data buffer 821 and the ECC engine/sub-ECC engines 822-0 to 822-M are similar

to the data buffer 121 and the ECC engine 122 as shown in FIG. 1” and, as I discussed above with respect to claim 1, refers back to Figures 1 and 2 and the corresponding description “For detailed description of the host interface 101, the data buffer 821, the ECC engine/sub-ECC engines 822-0 to 822-M, the scramblers 103, the multiple memory access module 104, and the memory devices 105-0 to 105-N as shown in FIG. 8.” Ex. 1004 at ¶0033.

131. Thus, a POSITA would have understood that Wu discloses that ECC engine/sub-ECC engines 822-0 to 822-M collectively perform the function of ECC engine 122 shown in Figure 1. Wu further discloses, with respect to Figure 1, that “The error correcting module 102 [comprising ECC engine 122 and data buffer 121] is arranged to perform error correction code encoding on the data to be stored into the memory devices [105-0 to 105-N may be a NAND or NOR flash memory] and perform error correction code decoding on the data read out from the memory devices, depending on the data transferring directions (e.g. write or read directions).” Ex. 1004 at ¶0022; *see also* ¶¶0024-25, 0034 (“According to the embodiments of the invention, the ECC engine (such as the ECC engine 122, 222, 822-0 to 822-M and 900 . . . etc.) is arranged to perform suitable ECC algorithm, such as Reed-Solomon or Bose, Ray-Chaudhuri, Hocquenghem (BCH) encoding and decoding”) (ellipses in Wu).

132. Wu therefore discloses “an encoder,” that is, ECC engine/sub-ECC engines 822-0 to 822-M (which are also “decoding module” components) (yellow) “configured to encode data for storage on a flash memory,” comprised of flash memory devices 105-0 to 105-N (green), and “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data.”

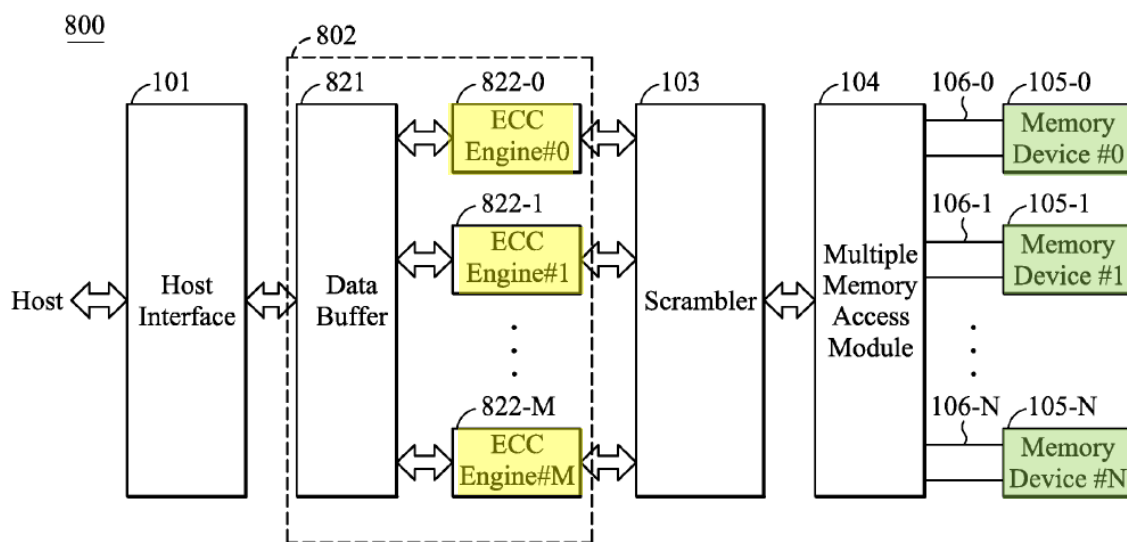


FIG. 8

133. Therefore, based on these disclosures, a POSITA would have understood that Wu discloses the claimed “an encoder configured to encode data for storage on a flash memory” and “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data.”

134. Accordingly, for the reasons discussed in this section, as well as in Section XI.1A.1 above (explaining why Wu anticipates claim 1) because Wu discloses the claimed limitations of claims 15 and 17 of the '700 Patent, arranged

as in the claim, a POSITA would have understood that Wu anticipates claims 15 and 17.

5. Claim 19

“The method of claim 17, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.”

135. Claim 19 depends from claim 17 and contains the same limitations as claim 4, which depends from claim 1:

Claim 19	Claim 4
<u>The method of claim 17, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.</u>	<u>The flash memory decoder of claim 1, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.</u>

136. The limitations of claim 19 are therefore disclosed (and anticipated) by Wu for the reasons as I discussed above for claims 1, 4, and 17.

B. JO IN VIEW OF WU RENDERS OBVIOUS CLAIMS 1, 2, 4, 8, 15, 17, AND 19.

137. In my opinion, the combination of Jo and Wu discloses and renders obvious claims 1, 2, 4, 8, 15, 17, and 19. Jo discloses all limitations of these claims, except it does not expressly disclose generation of “partially decoded” data streams. However, as I explain herein, a POSITA would have been motivated to add, and had a reasonable expectation of success in adding, descrambler modules

to Jo, for example, as disclosed by Wu and further evidenced by the state of the art so as to render the challenged claims obvious.

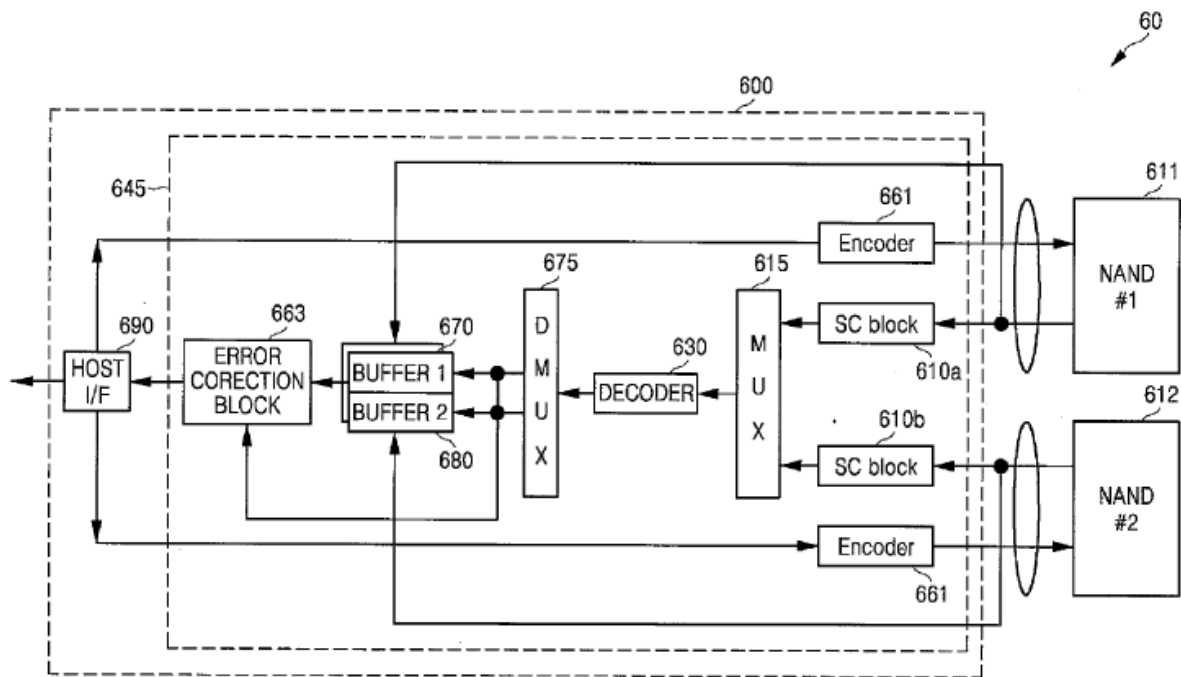
1. Claim 1

1[preamble] “A flash memory decoder comprising”

138. Regardless of whether the preamble is limiting, it is disclosed by Jo, as discussed regarding 1[preamble] and further regarding limitations 1[a] to 1[d].

139. For example, Figure 6 of Jo, reproduced below, “illustrates a two channel flash memory system 60 that can operate in a pipelined mode with a buffer memory structure. Flash memory system 60 includes a flash memory controller 600 and NAND memory devices 611 and 612. Flash memory controller 600 communicates data with memory devices 611 and 612 over two communication channels operating at the same time as each other in parallel.” Ex. 1005 at ¶0036.

FIG. 6



140. As shown in Figure 6, “Flash memory controller 600 includes ECC block 645 and host interface 690. ECC block 645 includes two data encoders 661, one each for the first and second communication channels. ECC block 645 also includes two syndrome computation blocks 610 *a* and 610 *b*, again one for each of the first and second communication channels. ECC block 645 further includes a multiplexer (or data switch) 615, a decoder block 630, a demultiplexer 675, first and second memory buffers 670 and 680, third and fourth memory buffers (not labeled), and an error correction block 663.” Ex. 1005 at ¶10037.

141. Per Jo, “In operation, ECC block 645 communicates data with memory devices 611 and 612 over two communication channels operating at the same time as each other in parallel.” Ex. 1005 at ¶0038. Jo further discloses that “When data from a host device (e.g., a processor) is to be written to the memory devices ... the data is first encoded (e.g. with a BCH code) by a corresponding encoder 661.” Ex. 1005 at ¶0038; *see also* ¶0023. Reference to BCH is to the well-known Bose, Ray-Chaudhuri, Hocquenghem code. *See also* Ex. 1001 (’700 Patent) at 4:10-15 (discussing same). Jo then discloses that “When the encoded data is to be read ... it is decoded before being sent to a host device (e.g., a processor).” Ex. 1005 at ¶0039.

142. Jo further discloses that—as part of the decoding process in connection with Figure 6—“syndrome computation block 610a computes the syndrome of a first set of data (e.g., data from an Nth sector of memory) received [from memory device 611, *e.g.*, as reflected by the arrow in Figure 6 indicating data flow] at the same time that syndrome computation block 610b computes the syndrome of a second set of data (e.g., data from an Mth sector of memory) [received from memory device 612, *e.g.*, as reflected by the arrow in Figure 6 indicating data flow].” Ex. 1005 at ¶0039.

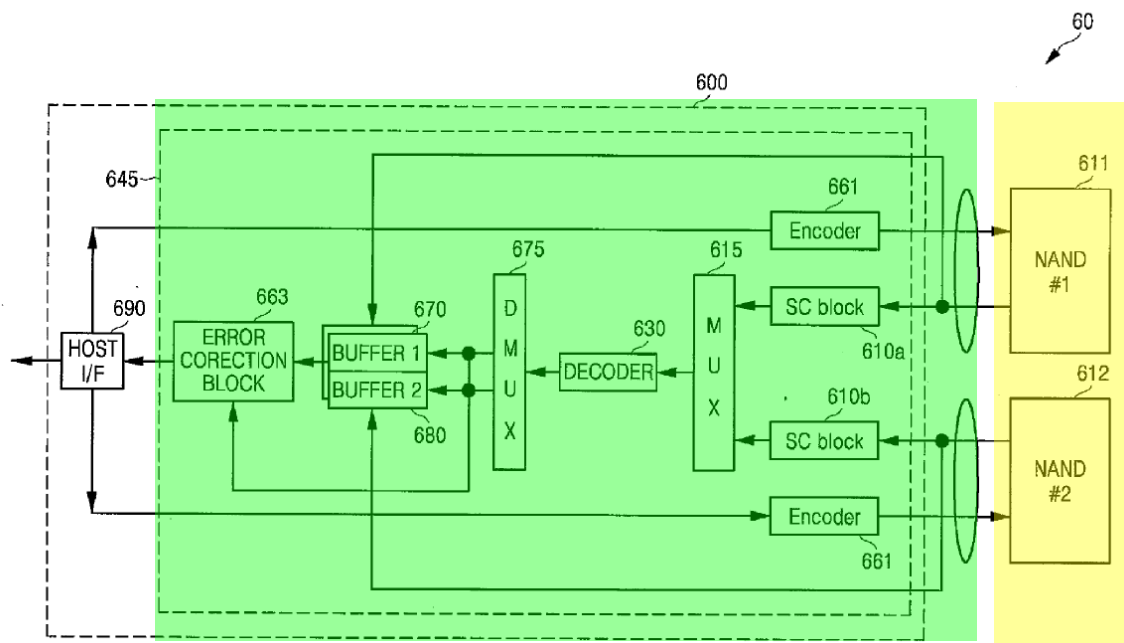
143. Jo further discloses that errors are detected by the syndrome computation blocks, disclosing that “If the syndrome values for a set of data are

zero, then this indicates that the data set does not include errors. Otherwise, then the data set includes one or more errors and therefore, the errors should be located and corrected. That is, the syndrome computation blocks 610a and 610b detect whether or not a data set includes errors.” Ex. 1005 at ¶0040. Syndrome computation involves taking the product, *i.e.*, multiplying, a parity check matrix and a vector of retrieved data to generate a set of “syndrome values.”

144. According to Jo, “Beneficially, ECC block 645 includes multiplexer 615 so that the two communication channels can share one key equation solver (error decoder block 630.... Once the errors are identified in a data set, then the errors are corrected in error correction block 663 (e.g., an exclusive-or (XOR) circuit) and the decoded data set may then be sent to a host device (e.g., a processor).” Ex. 1005 at ¶0041.

145. The two-channel memory system depicted in Jo’s Figure 6 (and in Jo’s Figures 3 and 4 in alternate embodiments) discloses a “flash memory” (yellow) “decoder” of the flash memory (green), as shown in Figure 8 below.

FIG. 6



1[a] “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams;”

146. The combination of Jo and Wu discloses the claimed “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams.”

147. Jo discloses that, during a write operation to NAND flash memory devices 611 and 612, data from a host device (e.g., a processor) is “first encoded (e.g. with a BCH code) by a corresponding encoder 661 for each communication channel. Then, the encoded data is transmitted in parallel across the two communication channels to memory devices 611 and 612.” Ex. 1005 at ¶0038 (“In operation, ECC block 645 communicates data with memory

devices 611 and 612 over two communication channels operating at the same time as each other in parallel. When data from a host device (e.g., a processor) is to be written to the memory devices via the two communication channels, the data is first encoded (e.g. with a BCH code) by a corresponding encoder 661 for each communication channel. Then, the encoded data is transmitted in parallel across the two communication channels to memory devices 611 and 612.”).

148. Jo further discloses that, “When the encoded data is to be read from memory devices 611 and 612 via the two communication channels, it is decoded before being sent to a host device (e.g., a processor).” Ex. 1005 at ¶0039.

149. Jo does not expressly disclose that the data received over the two communications channels at the syndrome computation blocks 610a and 610b that are part of an “error detection module” (discussed in limitation 1[b] below) has been partially decoded by a “decoding module.” However, Jo in combination with Wu, prior art that is closely analogous Jo, teaches this limitation. (See, for example, my discussion of Wu in Section IX.A and my discussion of Jo in this section). Among other things, both Jo and Wu are both directed to similar multi-channel reduced area controllers for accessing and performing ECC on data stored and read on multiple flash memory devices. *See, e.g.*, Ex. 1005, Figs. 1-3 and 6; Ex. 1004, Figs. 1, 8, 9, ¶0034.

150. Wu discloses the use of a scrambler module having, e.g., a plurality of scramblers, such as scramblers I discussed above in Section XI.A, as well as the scramblers 703-1 to 703-N that are “each coupled to a storage channel and arranged to perform intra-channel disarrangement for the data to be stored into the corresponding memory device and rearrange the data read out from the corresponding memory device.” Ex. 1004 at ¶0032.

151. As shown in Figure 7, reproduced below, scramblers 703-0 to 703-N of Wu are associated with corresponding memory devices 105-0 to 105-N (which, according to Wu, “may be a NAND or NOR type flash” and collectively constitute a “decoding module”). Ex. 1004 at ¶¶0024, 0032. As with Figure 8 (discussed above for Wu’s anticipation of claim 1), Wu similarly discloses that “ For detailed description of the host interface 101, the error correcting module 102, the scramblers 603 and 703-1 to 703-N, the multiple memory access module 104, and the memory devices 105-0 to 105-N as shown in FIG. 6 and FIG. 7, reference may be made to FIG. 1 and FIG. 2 and the corresponding descriptions, thus further descriptions are omitted here for brevity” (*i.e.*, it refers a POSITA back to Figures 1 and 2 for these components). Ex. 1004 at ¶0032.

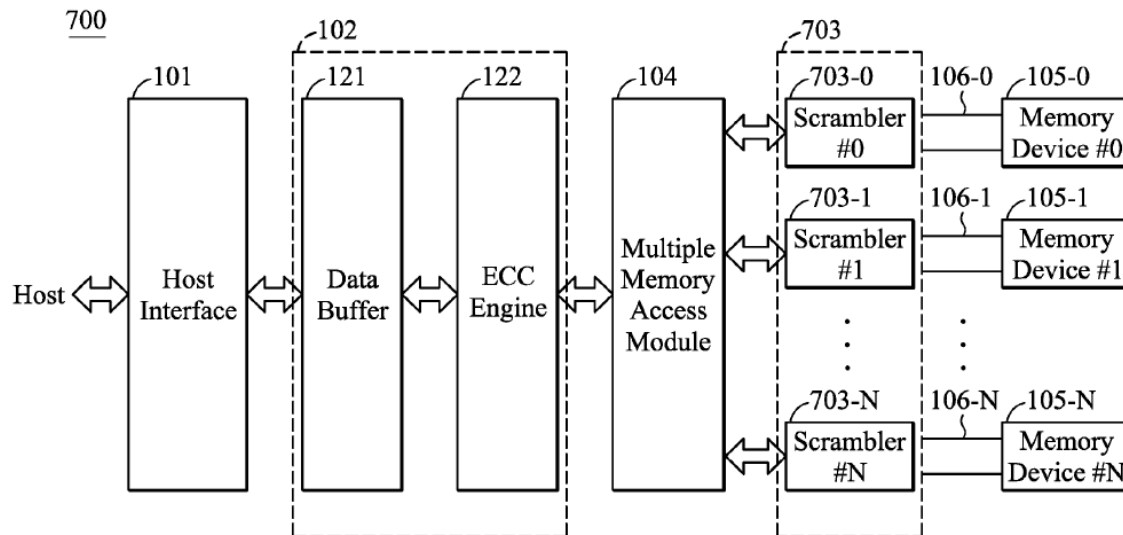


FIG. 7

152. Wu further discloses that the scramblers 703-0 to 703-N are like the scrambler 103 as shown in Wu's Figure 1, but operate only on the data read and written to a single corresponding memory device 105-0 to 105-N (as depicted in Figure 7 above). Ex. 1004 at ¶0032 ("FIG. 7 shows yet another multi-channel memory apparatus 700 according to another embodiment of the invention. As shown in FIG. 7, the scrambler module 703 may comprise a plurality of scramblers, such as the scramblers 703-1 to 703-N each coupled to a storage channel and arranged to perform intra-channel disarrangement for the data to be stored into the corresponding memory device and rearrange the data read out from the corresponding memory device. Note that the operations of the scramblers 603 and 703-1 to 703-N are similar to the scrambler 103 as shown in FIG. 1 and FIG. 2. For detailed description of the host interface 101, the error

correcting module 102, the scramblers 603 and 703-1 to 703-N, the multiple memory access module 104, and the memory devices 105-0 to 105-N as shown in FIG. 6 and FIG. 7, reference may be made to FIG. 1 and FIG. 2 and the corresponding descriptions, thus further descriptions are omitted here for brevity.”).

153. Thus, as disclosed in Wu, each scrambler 703-0 to 703-N “perform intra-channel disarrangement for the data to be stored into the memory devices and rearrange the scrambled data read out from the memory devices in a reverse manner depending on the data transferring directions (e.g. write or read directions).” Ex. 1004 at ¶¶0023, 0032. Wu further discloses that the intra-channel disarrangement/arrangement operation is performed to scramble/unscramble the data of the ECC engine codewords, the error correction coded data, so as to randomly distribute the data ‘0’ and ‘1’ in adjacent locations thereby further reducing interference. Ex. 1004 at ¶0023 (“The intra-channel disarrangement is a disarrangement operation performed within a channel so as to scramble the data of the same codeword of the ECC engine. The scrambler 103 scrambles the data so as to randomly distribute the data ‘0’ and ‘1’ in adjacent locations. Thus, interference introduced by the adjacent data of continuous ‘0’ or ‘1’ within the memory device 105-0 to 105-N is reduced. Note that because the scrambling process is a linear operation, the scrambler (scrambler module) may be located anywhere in the

data transferring paths (e.g. write or read paths). In addition, according to another embodiment of the invention, the scrambler module may also comprise a plurality of scramblers to deal with the scrambling of the multiple storage channels (which will be discussed in detail in the following paragraphs”).

154. A POSITA would have understood that Jo’s flash memory system for interfacing a host with NAND flash memory devices 611 and 612 (as shown in Jo’s Figure 6) would likewise benefit from the interference reduction provided by the use of Wu’s scramblers 703 at each of Jo’s memory channels. This is further evidenced by the fact that it was known at the time that data scrambling had become of increased importance in addressing NAND flash memory reliability issues associated with worst-case data patterns, for example, as described in Wan. Ex. 1006 at ¶0049 (“Data scrambling for non-volatile memory such as NAND memories has become increasingly important in addressing system reliability issues that are inherently susceptible by design to worst-case data patterns which trigger memory program failures. The data patterns which induce memory failures are referred to as worst-case patterns. Such failures are catastrophic due to uncorrectable error correction code (ECC) decoding as a result of program disturb. Moreover, program disturb issues depend on memory bit-line cell states in adjacent word lines and in series of word lines. Worst-case patterns include fixed repetitive data such as may be found in control data and file access table (FAT) areas. FAT

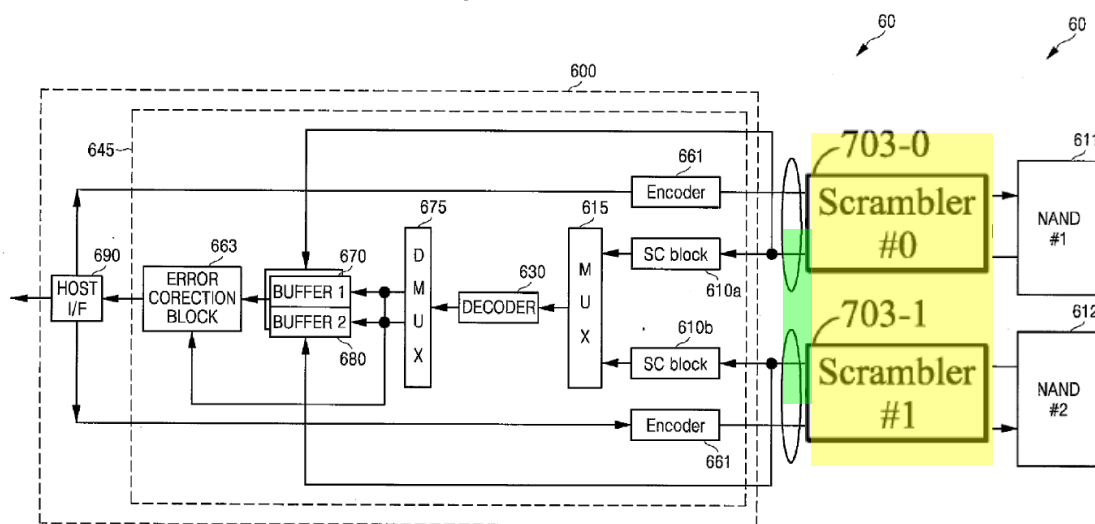
data relates to file system data, and how file names, folders and the like are organized. Worst-case patterns may be caused by other factors as well. For example, a word line with more low state storage elements may be disturbed by a word line with more high state cells. Data scrambling randomizes the distribution of data states in the storage elements so that worst-case data patterns are avoided. Data scrambling and descrambling should be transparent to the user.”).

155. A POSITA would also have a reasonable expectation of success in making the combination; that the combination would perform as intended; and that it could be implemented with minimal, if any experimentation. A POSITA would have understood, for example, that Wu’s scramblers 703-0 to 703-N could be implemented by adding circuitry to Jo’s memory controller 600, to Jo’s memory devices 611 and 612, or by providing a standalone scrambling circuitry interfacing with the memory controller 600 and the memory devices 611 and 612.

156. There were additional advantages known to a POSITA of locating Wu’s scramblers as shown in Jo’s Figure 6 (annotated below). For example, as evidenced by Wan (Exhibit 1006), it has the additional advantage of helping avoid header and ECC type data pattern failures, and, if located on Jo’s memory devices, may promote portability of the memory die so that they can be used with different controllers. Ex. 1006 at ¶0070 (“In this approach, a scrambler/descrambler 812 is on the memory die 808 rather than in a controller 802 of the memory device 800.

This approach provides portability of the memory die so that it can be used with different controllers, for instance, which are made by different manufacturers.”).

FIG. 6



157. Accordingly, Jo’s Figure 6 embodiment, as modified by Wu to further include scramblers for disarranging, *i.e.*, partially decoding, data stored in each memory device 611 and 612 upon a read operation, discloses and renders obvious the claimed “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams.” Again, as discussed above, Wan explains that “scrambling is considered to be a form of encoding and descrambling is considered to be a form of decoding.” Ex. 1006 at ¶¶0052, 0062. The combination of Jo and Wu is shown

in the Figure above, with the “decoding module” in yellow and the “plurality of partially decoded data streams” in green.

1[b] “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel,”

158. Jo discloses the claimed “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel.”

159. Jo discloses that, when the encoded data stored in memory devices 611 and 612 is read “via the two communication channels ... syndrome computation block 610a computes the syndrome of a first set of data (e.g., data from an Nth sector of memory) received via the first communication channel at the same time that syndrome computation block 610b computes the syndrome of a second set of data (e.g., data from an Mth sector of memory) via the second communication channel. If the syndrome values for a set of data are zero, then this indicates that the data set does not include errors. Otherwise, then the data set includes one or more errors and therefore, the errors should be located and corrected. That is, the syndrome computation blocks 610a and 610b detect whether or not a data set includes errors.” Ex. 1005 at ¶¶0039-40. This is also evidenced by claim 7 of Jo, which states that “the first error detector calculates a first syndrome value of a first data set received via the first communication channel,

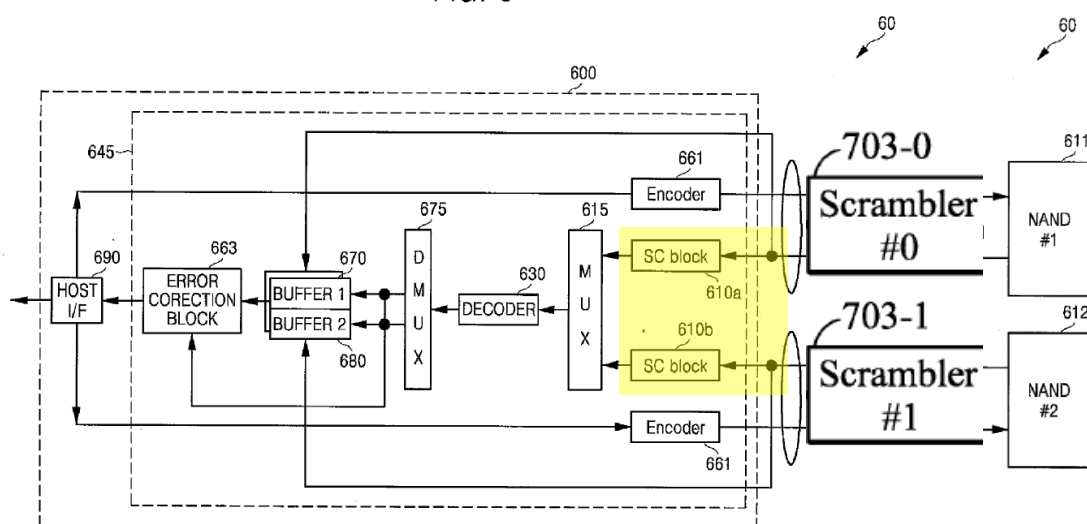
and wherein the second error detector calculates a second syndrome value of a second data set received via the second communication channel.” Ex 1005 at claim 7.

160. Thus, a POSITA would have understood that Jo’s syndrome computation blocks 610a and 610b (yellow in the figure in the following paragraph) determine whether there are errors in the data, that is, whether the syndrome values are non-zero or zero. A POSITA would have also thus understood that the syndrome computation blocks collectively disclose “an error detection module communicatively coupled with the decoding module,” as shown in limitation 1[b], and “comprising a plurality of error detection sub-modules,” including blocks 610a and 610b, “operating in parallel” (*see, e.g.*, Ex. 1005 at ¶39 (“Accordingly, syndrome computation block 610a computes the syndrome of a first set of data (e.g., data from an Nth sector of memory) received via the first communication channel at the same time that syndrome computation block 610b computes the syndrome of a second set of data (e.g., data from an Mth sector of memory) via the second communication channel.”), cl. 16 (“wherein the first syndrome computation block calculates the first syndrome value for the data received via the first channel while the second syndrome computation block calculates the second syndrome value for the data received via the second

channel”)). I again note that claim 1 of the ’700 Patent covers flash memory decoders that perform BCH decoding, such as disclosed by Jo. Ex. 1005 at ¶0038.

161. Figure 6 of Jo is shown below, which has been modified to include scramblers disclosed in Wu’s Figure 7, with Jo’s disclosed “error detection module,” comprising syndrome computation blocks 610a and 610b, in yellow.

FIG. 6



162. Based on the above disclosures, a POSITA would have understood that Jo, as modified by Wu, discloses “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel.”

1[c] “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the

respective received stream containing an error to an error correction module; and”

163. Jo, as modified by Wu, discloses the claimed “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module.”

164. For example, as shown in Jo’s Figure 6 annotated above in limitation 1[b], Jo, as modified by Wu, discloses “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams.” That is, annotated Figure 6 shows that each of blocks 610a and 610b (the error detection sub-modules) receives a different (partially decoded) data stream from the scramblers of Wu (the decoding module).

165. Jo also discloses that its “error detection sub-modules” 610a and 610b compute the syndromes of a first set of data from memory 611 and a second set of data from memory 612, respectively. Ex. 1005 at ¶0039 (“When the encoded data is to be read from memory devices 611 and 612 via the two communication channels, it is decoded before being sent to a host device (e.g., a processor). Accordingly, syndrome computation block 610 *a* computes the syndrome of a first set of data (e.g., data from an Nth sector of memory) received via the first communication channel at the same time that syndrome computation

block 610 *b* computes the syndrome of a second set of data (e.g., data from an *M*th sector of memory) via the second communication channel.”).

166. Jo further discloses that “If the syndrome values for a set of data are zero, then this indicates that the data set does not include errors. Otherwise, then the data set includes one or more errors and therefore, the errors should be located and corrected. That is, the syndrome computation blocks 610a and 610b detect whether or not a data set includes errors.” Ex. 1005 at ¶0040.

167. A POSITA would thus have understood that, based on these disclosures, Jo discloses “each error detection sub-module configured to: ... detect whether a portion of the respective received stream contains an error.”

168. Further, as discussed above, Jo discloses error location and correction—when “the data set includes one or more errors ... the errors should be located and corrected.” Ex. 1004 at ¶0040. Jo discloses this error location and correction process as follows (referring to Figure 6): “When data sets are read into memory controller 600 from the two memory chips 611 and 612, errors in the data sets received from the two communication channels are detected at the same time. At this time, the data sets are stored in buffer memories 670 and 680, waiting for correction and transmission to the host.” Ex. 1005 at ¶0042.

169. According to Jo, after errors are detected “multiplexer 615 multiplexes between the first and second communication channels for locating

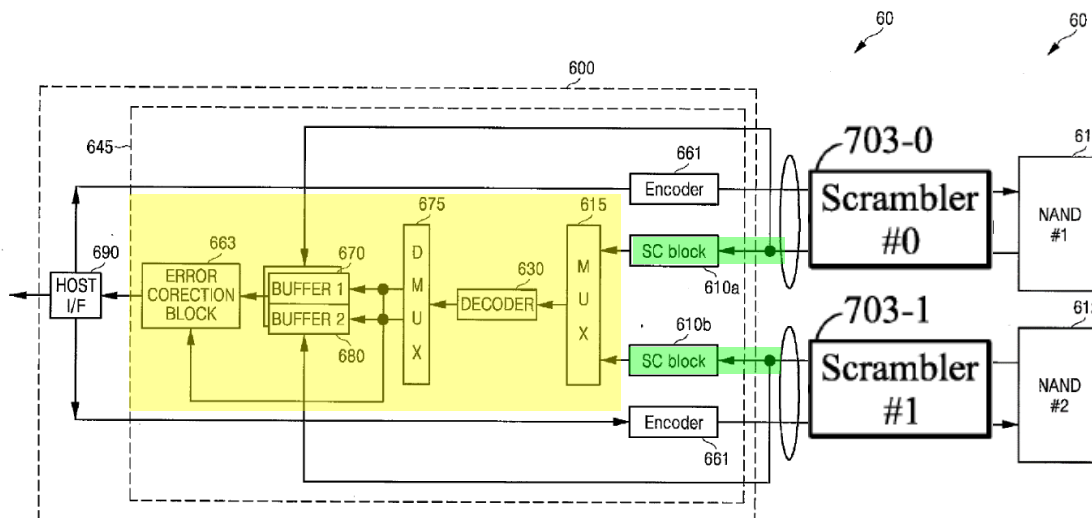
[the] errors that occur in data sets received from the memory devices via the communication channels.” Ex. 1005 at ¶0041.

170. Jo also discloses that (i) “Decoder block 630 calculates the error locations for a first data set read data [*sic*] from memory device 611, and then the first set of data stored in buffer memory 670 is corrected....” (Ex. 1005 at ¶0043); and (ii) “the errors are corrected in error correction block 663 (e.g., an exclusive-or (XOR) circuit) and the decoded data set may then be sent to a host device (e.g., a processor).” Ex. 1005 at ¶0041. Per Jo, “While the first data set stored in buffer memory 670 is being transmitted to the host, decoder block 630 calculates error locations for a second data set read data from memory device 612, and then the second set of data stored in buffer memory 680 is corrected and transmitted to the host....” Ex. 1005 at ¶0043.

171. Thus, based on these disclosures, a POSITA would have understood that Jo discloses “each error detection sub-module,” shaded green below, “configured to: ... forward the portion of the respective received stream containing an error to an error correction module.”

172. In annotated Figure 6 of Jo (annotated to include scramblers disclosed in Wu’s Figure 7), Jo’s “error correction module” includes the blocks in yellow, and the forwarded portion of the stream is received at buffers 670 and 680.

FIG. 6



1[d] “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

173. Jo discloses the claimed “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

174. With respect to Figure 6, Jo discloses the claimed “error correction module, communicatively coupled with ... the error detection module,” at the SC block 610a/b outputs and multiplexer 615 inputs. I have discussed the error correction and detection modules in communication with each other in limitations 1[b] and 1[c] in connection with Figure 6.

175. In the same Figure 6, Jo also discloses the claimed “error correction module ... physically separate from the error detection module.” I have discussed

in connection with limitations 1[b] and 1[c] that the “error correction module” comprises circuit components including “multiplexer (or data switch) 615,” “key equation solv[ing] error decoder block 630,” demultiplexer 675, “first and second memory buffers 670 and 680, third and fourth memory buffers (not labeled),” and “error correction block 663 (e.g., an exclusive-or (XOR) circuit)” and that the “error detection module” comprises circuit components including syndrome computation blocks 610a and 610b which are “physically separate” from the components of the “error correction module.” *See also* Ex. 1005 at ¶¶ 0036-37, 0041, Fig. 6. Jo also discloses that this structure promotes efficient use of circuitry within the memory controller. Ex. 1005 at ¶0010 (“It would be desirable to provide a memory system that can provide robust error detection and correction with a more efficient utilization of area and circuitry in a memory controller. It would also be desirable to provide a method of processing data in a memory system that supports a more efficient utilization of area and circuitry in a memory controller.”), Title (“Multi-channel memory system including error correction decoder architecture with efficient area utilization”).

176. Jo also discloses that its “error correction module [is] configured to correct the received portions of the respective received streams containing an error,” as claimed. For example, Jo describes an operation of the “error correction module” with reference to Figure 6. Although errors in the data read from

memory devices 611 and 612 are being detected by the error detection sub-module 610a and 610b, the data is stored in buffer memories 670 and 680, waiting for correction and transmission to the host. Ex. 1005 at ¶0042.

177. According to Jo, after errors are detected “multiplexer 615 multiplexes between the first and second communication channels” so the “channels can share one key equation solver (error decoder block 630[]).” Ex. 1005 at ¶0041. As discussed above, Jo also discloses that (i) “Decoder block 630 calculates the error locations for a first data set read data [*sic*] from memory device 611, and then the first set of data stored in buffer memory 670 is corrected....” (Ex. 1005 at ¶0043); and (ii) “the errors are corrected in error correction block 663 (e.g., an exclusive-or (XOR) circuit) and the decoded data set may then be sent to a host device (e.g., a processor).” Ex. 1005 at ¶0041.

178. Based on the above disclosures, a POSITA would have understood that Jo discloses “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

179. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wu disclose all of the limitations of claim 1 of the ’700 Patent, and that such combination renders claim 1 obvious.

2. Claim 2: “The flash memory decoder of claim 1, wherein each of the error detection sub-modules is configured to read data from a set of one or more sectors of flash memory different from other of the error detection sub-modules.”

180. Jo discloses that “each of the error detection sub-modules is configured to read data from a set of one or more sectors of flash memory different from other of the error detection sub-modules.”

181. When discussing Figure 6 of Jo, Jo discloses that “syndrome computation block 610a computes the syndrome of a first set of data (e.g., data from an Nth sector of memory) received via the first communication channel at the same time that syndrome computation block 610b computes the syndrome of a second set of data (e.g., data from an Mth sector of memory) via the second communication channel.” Ex. 1005 at ¶0036.

182. A POSITA would have understood that Jo discloses that “each of the error detection sub-modules is configured to read data from a set of one or more sectors of flash memory different from other of the error detection sub-modules,” as claimed. That is, syndrome computation block 610a, *i.e.*, a first “error detection sub-module,” is configured to read data from a set of sectors in NAND memory 611, and error detection sub-module 610b, *i.e.*, a second “error detection sub-module,” is configured to read data from a set of sectors in NAND memory 612.

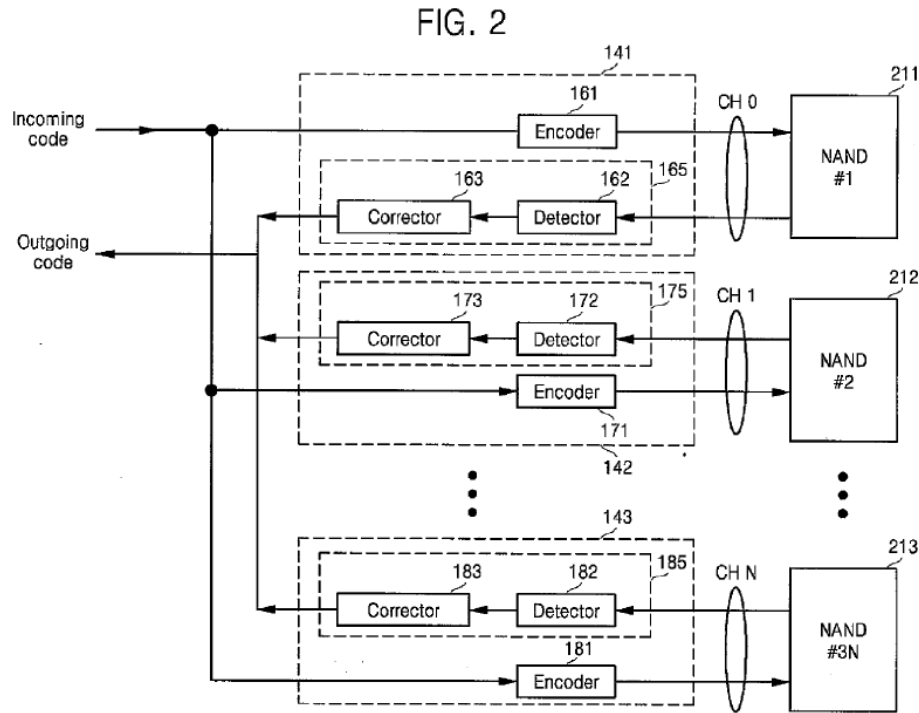
183. Thus, for the reasons I've explained above, a POSITA would have understood that the combination of Jo and Wu disclose all of the limitations of claim 2 of the '700 Patent, and that such combination renders claim 2 obvious for the same reasons as claim 1.

3. Claim 4: "The flash memory decoder of claim 1, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules."

184. Jo discloses that "the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules."

185. Jo discloses that "There is a trend for flash memory systems to have more and more memory devices" and "to employ multi-level cell (MLC) NAND memory devices for increased storage capacity." Ex. 1005 at ¶0009. Jo discloses that, as a result of this increased capacity, flash memory systems require more and more ECC modules, and that this enlarges the size of the integrated circuit, and increases the number of ECC IP core gates, for the flash memory controller, and that "This increases the complexity and cost of the flash memory system." Ex. 1005 at ¶0009. Jo discloses a memory system "that can provide robust error detection and correction with a more efficient utilization of area and circuitry in a memory controller." Ex. 1005 at ¶0010. For example, in Figure 2 of Jo,

reproduced below, Jo discloses an exemplary prior art flash memory system with $N > 2$ memory devices. Ex. 1005 at ¶0007; *see also* ¶¶0005-6 (discussing prior art flash memory systems).



186. Against this backdrop, to reduce circuitry area, Jo then discloses an exemplary “error corrector adapted to correct errors detected by each of the at least first and second error detectors” corresponding to first and second memory devices. Ex. 1005 at ¶0012, Figs. 3, 4, and 6.

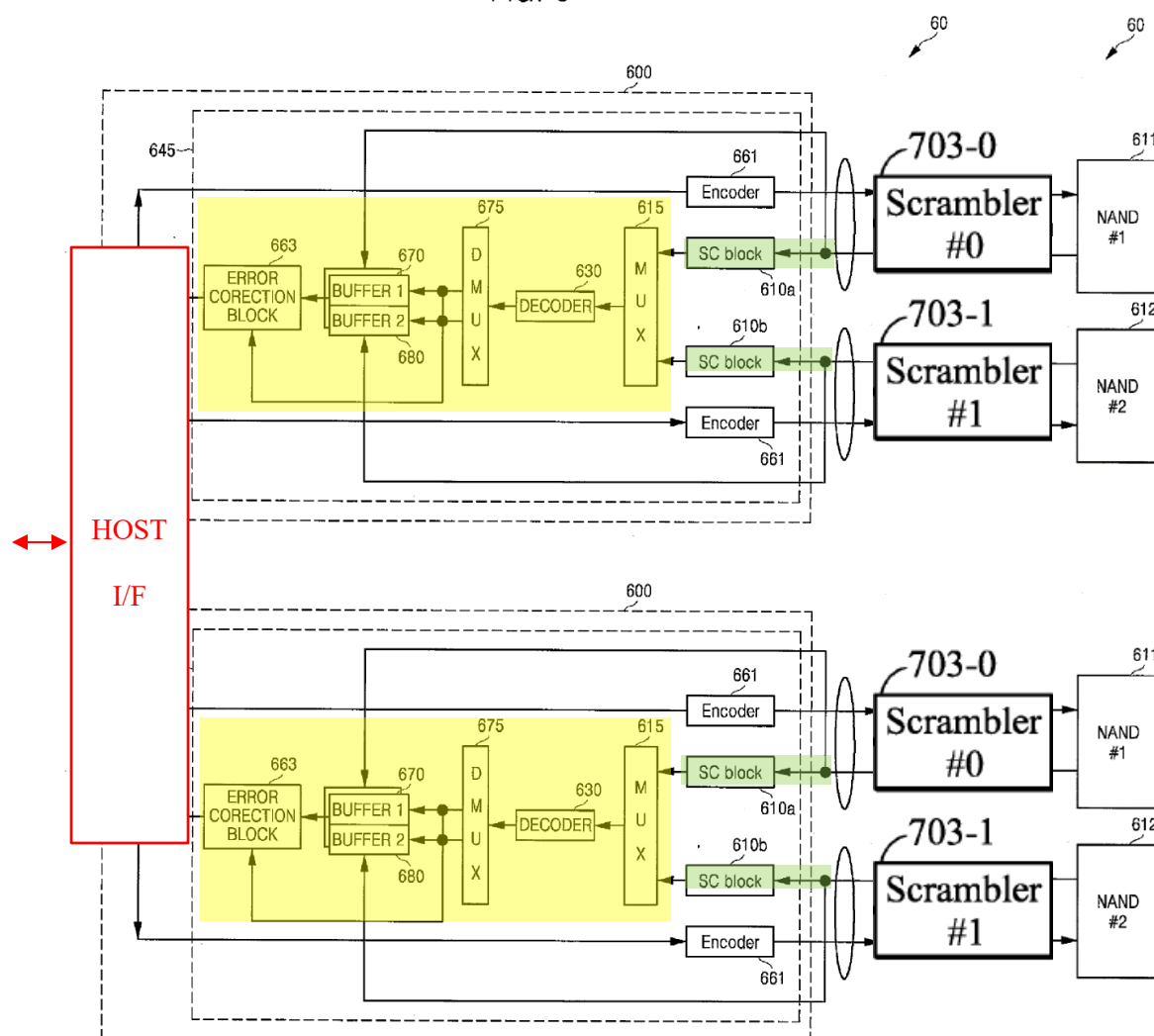
187. Although the exemplary embodiments shown in Figures 3, 4 and 6 of Jo depict two memory devices and two channels, Jo expressly discloses that its inventive concept is not limited to systems with just two memory devices and two channels. In particular, Jo discloses that “In one aspect of the inventive concept, a

memory system comprises: at least two memory devices; and a memory controller having at least first and second communication channels.” Ex. 1005 ¶0012. A POSITA would have understood this disclosure to include two or more (e.g., three, four, five, etc.). That is, a POSITA would have understood that Jo discloses, at a minimum, that its 2-channel embodiments can be scaled to, at a minimum, 2N channel embodiments, *e.g.*, by combining 2-channel embodiments.

188. Regardless, to the extent that Jo’s express disclosure is limited to only two (rather than “at least” two, as actually disclosed), such replication would have been obvious to try, as it would allow Jo’s 2-channel embodiments to be scaled to memory systems having larger capacity, *i.e.*, more flash memory devices.

189. By way of example, a disclosed and obvious exemplary 4-channel system is illustrated below “wherein the error correction module comprises a plurality of error correction sub-modules,” (yellow), “operating in parallel and fewer in number than the error detection sub-modules,” (green). In the figure below, Figure 6 of Jo has been annotated to include scramblers disclosed in Wu’s Figure 7; reproduced twice to depict a 4-channel memory device; and with a modified host interface (in red) expanded to interface with the four NAND memory devices via the four encoders 661 and two error correction blocks 663.

FIG. 6



191. This decoder configuration allows the area saving technique disclosed by Jo with respect to its exemplary two memory device embodiments of Figures 3, 4, and 6 to be applied to systems having larger memory capacity (i.e., more memory devices, as shown in Figures 1 and 2), as demanded by market trends expressly noted by Jo. Ex. 1005 at ¶0009. Modifications, including to the host interface, to accommodate more memory devices are implicitly disclosed by Jo's teaching that its representative embodiments are not limited to 2-channel devices,

and in any case would have required no undue experimentation and would have been trivial to implement for a POSITA. This is further confirmed by the fact that Jo discloses a prior art host interface 190 shown in Figure 1 that interfaces with more than two NAND memory devices. Ex. 1005 ¶0006.

192. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wu disclose all of the limitations of claim 4 of the ’700 Patent, and that such combination renders claim 4 obvious for the same reasons as claim 1.

4. Claim 8

193. I discuss below claim 8 in two parts (limitations 8[a] and 8[b]).

8[a] “The flash memory decoder of claim 4, wherein, each error correction sub-module is assigned a set of two or more error detection sub-modules different from error detection sub-modules associated with other of the error correction sub-modules; and”

194. As shown in Jo’s annotated Figure 6 (discussed in connection with Figure 4 above) Jo discloses “each error correction sub-module,” yellow, “is assigned a set of two or more error detection sub-modules,” shaded green “different from error detection sub-modules associated with other of the error correction sub-modules.”

8[b] “each error correction sub-module is configured to receive portions of the respective received streams containing an error

detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-module.”

195. As shown in Jo’s annotated Figure 6 (discussed in connection with claim 4 above), Jo includes two error correction sub-modules. Each of these sub-modules, and their connected error detection sub-modules, independently meet limitations 1[b], [c], and [d] based on the disclosures identified above in paragraphs 158 to 179.

196. These same disclosures establish that Jo discloses limitation 8[b].

197. For example, as explained with reference to limitation 1[c] and [d], Jo discloses an “error correction module” (and sub-module in the context of claims 4 and 8) that “is configured to receive portions of the respective received streams containing an error detected in parallel by the set of two or more error detection sub-modules assigned to the respective error correction sub-module.”

198. Further for example, Jo describes an operation of the error correction module (or sub-module) with reference to Figure 6. While errors in the data read from memory devices 611 and 612 are being detected in parallel by assigned error detection sub-modules comprising blocks 610a and 610b, the data is received and stored in buffer memories 670 and 680 of the error correction module (or sub-module). *See* discussion of limitations 1[b], [c], and [d]; Ex. 1005 at ¶¶0036-37, 0039-43. Data containing errors is corrected by the error correction module (or

sub-module) and transmitted to a host. *See* limitation 1[d]; Ex. 1005 at ¶¶0036-37, 0041-43.

199. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wu disclose all of the limitations of claim 8 of the ’700 Patent, and that such combination renders claim 8 obvious for the same reasons as claim 1 and 4.

5. Claims 15 and 17

200. As I explained above with respect to my discussion of Wu anticipating claims 15 and 17, I note that independent claims 15 and 17 are very similar to apparatus claim 1, except that claim 15 is a “system” claim and claim 17 is a “method” claim and the limitations 15[a] and 15[b]. Therefore, claims 15 and 17 are obvious in view of Jo and Wu for the effectively the same reasons as claim 1.

15[a] “an encoder configured to encode data for storage on a flash memory,”

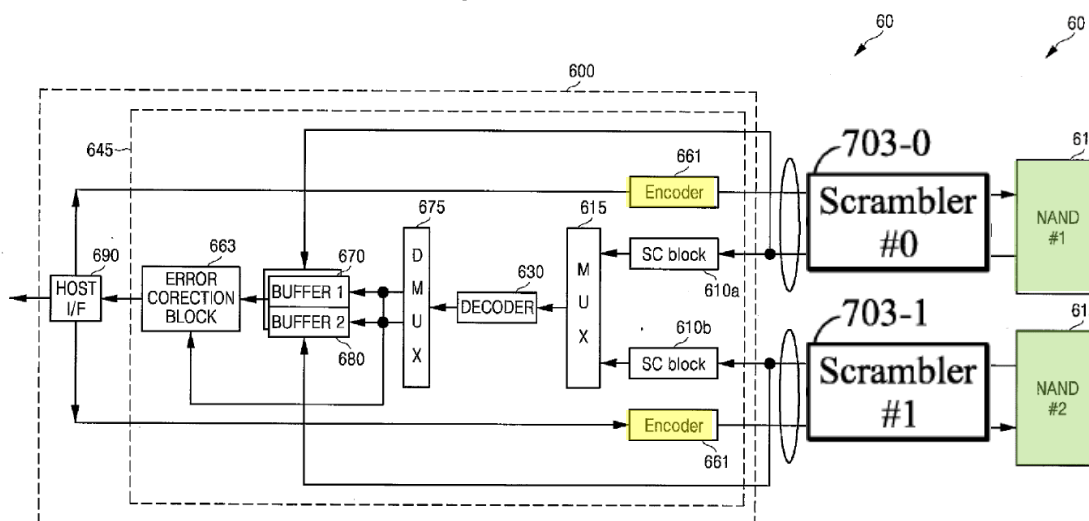
15[b] “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data; and”

201. Figure 6 of Jo, reproduced below, depicts a “Flash memory controller 600” that “includes two data encoders 661, one each for the first and second communication channels.” Ex. 1005 at ¶0037. Jo discloses that “In operation, ECC block 645 communicates data with memory devices 611 and 612 over two

communication channels operating at the same time as each other in parallel. When data from a host device (e.g., a processor) is to be written to the memory devices via the two communication channels, the data is first encoded (e.g. with a BCH code) by a corresponding encoder 661 for each communication channel. Then, the encoded data is transmitted in parallel across the two communication channels to memory devices 611 and 612.” Ex. 1005 at ¶¶0038; *see also* ¶¶0036 (disclosing that memory devices 611 and 612 are NAND flash memory).

202. A POSITA would have thus understood that Jo discloses “an encoder,” encoders 661 in yellow below and collectively forming the recited “encoding module,” “configured to encode data for storage on a flash memory,” comprised of memory devices 611 and 612 in green, and “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data.” The below discloses Figure 6 of Jo annotated to include scramblers disclosed in Wu’s Figure 7. *See also* Ex. 1004 at ¶¶0023, 0032 (Wu disclosing that scramblers 703 “perform intra-channel disarrangement for the data to be stored into the memory devices”).

FIG. 6



203. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wu disclose all of the limitations of claims 15 and 17 of the ’700 Patent, and that such combination renders these claims obvious for the same reasons as claim 1.

6. Claim 19: “The method of claim 17, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.”

204. Claim 19 depends from claim 17 and contains the same limitations as claim 4, which depends from claim 1:

Claim 19	Claim 4
The method of claim 17, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.	The flash memory decoder of claim 1, wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.

205. The limitations of claim 19 are therefore disclosed (and obvious over) the combination of Jo and Wu for the reasons as I discussed above for claims 1, 4, and 17.

C. JO IN VIEW OF WAN RENDERS OBVIOUS 1, 2, 4, 8, 15, 17, AND 19.

206. In my opinion, the combination of Jo and Wan discloses and renders obvious claims 1, 2, 4, 8, 15, 17, and 19. Jo discloses all limitations of these claims, except it does not expressly disclose generation of “partially decoded” data streams. However, as I explain herein, a POSITA would have been motivated to add, and had a reasonable expectation of success in adding, descrambler circuitry to Jo, for example, as disclosed by Wan and further evidenced by the state of the art so as to render the challenged claims obvious.

207. As a threshold matter, for the combination of Jo and Wan, the disclosures in Jo are substantially the same as those I discussed above for the combination of Jo and Wu. Thus, I incorporate by reference my discussion of Jo in the preceding section discussing the obviousness of the claims in view of the combination of Jo and Wu.

208. As I discussed above and below, Wu and Wan each disclose a “decoding module” as required by limitation 1[a] of claim 1 of the ’700 Patent. Further, I note that although both Wu and Wan disclose such a “decoding module”

comprising of scrambling/descrambling modules, the implementation of these modules differs in the references (*e.g.*, Wan additionally discloses that the scrambler/descrambler may be implemented on a memory die).

1. Claim 1

1[preamble] “A flash memory decoder comprising”

209. Regardless of whether the preamble is limiting, it is disclosed by Jo, as discussed regarding 1[preamble] in the combination of Jo and Wu above.

1[a] “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams;”

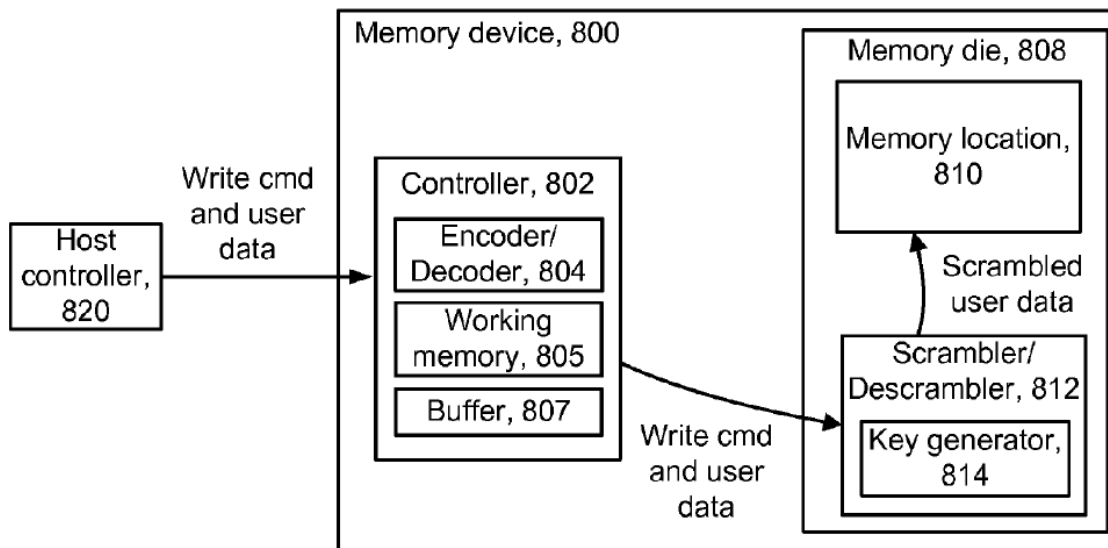
210. The combination of Jo and Wan discloses the claimed “a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams.”

211. I incorporate my discussion of Jo in limitation 1[a] for the combination of Jo and Wu above. As I discussed above, Jo discloses that, during a write operation to NAND flash memory devices 611 and 612, data from a host device (*e.g.*, a processor) is “first encoded (*e.g.* with a BCH code) by a corresponding encoder 661 for each communication channel. Then, the encoded data is transmitted in parallel across the two communication channels to memory devices 611 and 612,” which may be memory chips. Ex. 1005 at ¶¶0038, 0042.

212. Also as I discussed above regarding Jo and Wu, Jo also discloses that, “When the encoded data is to be read from memory devices 611 and 612 via the two communication channels, it is decoded before being sent to a host device (e.g., a processor).” Ex. 1005 at ¶0039.

213. Again, Jo does not expressly disclose that the data received over the two communications channels at the syndrome computation blocks 610a and 610b that are part of an “error detection module” (discussed limitation 1[b] below) has been partially decoded by a “decoding module.” However, Jo in combination with Wan teaches this limitation, which like Wu, is closely analogous art with Jo. (See, for example, my discussion of Wan in this section and my discussion of Jo in this section and in Section XI.B). Among other things, both Jo and Wan are directed to similar multi-channel controllers for accessing and performing ECC on data stored and read on multiple flash memory devices. *See, e.g.*, Ex. 1006, Figs. 5a, 5b, 8a, 8b, 12, ¶¶0042, 0089 (“Memory device 1196 may include one or more memory die 1198.”); Ex. 1004, Figs. 1, 8, 9, ¶0034. As shown in Figure 8a of Wan, reproduced below, and as I discuss below, Wan discloses a flash memory system embodiment.

Fig. 8a

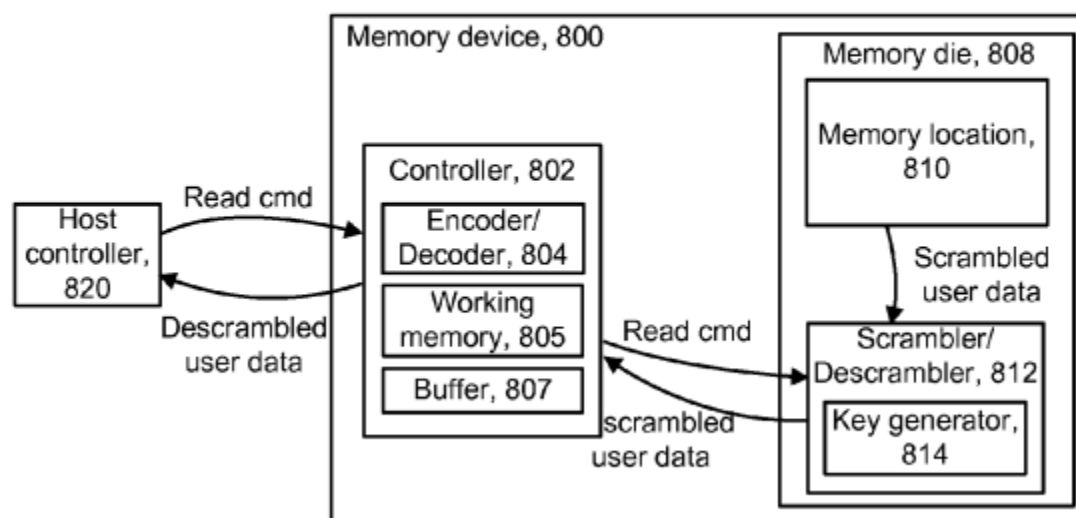


214. Wan discloses that the memory die (numbered 808 in Figure 8a above) includes a memory array of NAND flash storage elements capable of storing one or multiple bits of information. Ex. 1006 at ¶0042 (“One example of a memory system suitable for implementing the present invention uses the NAND flash memory structure...”), ¶0045 (“Each storage element can store ... one bit of digital data.... A storage element can also store multiple levels of information, for example, multiple bits of digital data.”), ¶0050 (“A chip refers to a memory die on which a memory array is formed.”), ¶0054 (“A memory array 509 having a number of blocks of storage elements may be formed on the memory die 508.”), ¶0089 (“Memory die 1198 includes a two-dimensional array of storage elements 1000, control circuitry 1110, and read/write circuits 1165.”), claim 20 of Wan. A POSITA would have understood this disclosure to apply to the various figures,

including, *e.g.*, based on Wan's general description that "One example of a memory system suitable for implementing the present invention uses the NAND flash memory structure" in Paragraph 0042, which indicates to a POSITA that each of the figures and embodiments rely upon such a NAND flash memory structure.

215. As disclosed in Wan's Figure 8a (above) and 8b (below), and corresponding discussion in the specification of Wan, data scrambling and descrambling is performed on the data to be stored to, and read from memory. Ex. 1006 at ¶¶0070-71. These operations are performed by scrambler/descrambler circuitry 812 residing on memory die, or chips, 808. Ex. 1006 at ¶¶0070-71.

Fig. 8b



216. For example, Wan discloses, in Figure 8a, that:

FIG. 8 *a* depicts scrambling performed by on-chip circuitry using a key generator in a storage system. In this approach, a scrambler/descrambler 812 is on the memory die 808 rather than in a controller 802 of the memory device 800. This approach provides portability of the memory die so that it can be used with different controllers, for instance, which are made by different manufacturers. The scrambler/descrambler 812 may be a state machine, for instance. In one possible approach, the host controller 820 provides a write command and associated user to the controller 802, which in turn communicates one or more corresponding write commands and the user data to the scrambler/descrambler 812 on the memory die 808. The controller 802 may process the user data such as by performing ECC coding and adding overhead data at an encoder/decoder 804. With this approach, the overhead data is also scrambled with the user data. This helps avoid header and ECC type data pattern failures. The scrambler/descrambler 812 may use a key generator 814 to generate different random or pseudo random keys, where each key is used to scramble a different portion of the user data. The scrambled user data is then written to a memory location 810. The controller 802 may have a working memory 805 and buffer 807. As with the approach

of FIG. 5 *a*, the user data may be scrambled and written in successive portions which are smaller than the amount of data provided by the host, or in a single unit which is the same as provided by the host.

Ex. 1006 at ¶0070; *see also* Ex. 1006 at Figs. 5a and 5b and ¶0050-64 (describing another embodiment in which the descrambling is performed by the off-chip controller 502).

217. As discussed above, Wan discloses that “This approach provides portability of the memory die so that it can be used with different controllers, for instance, which are made by different manufacturers.” Ex. 1005 at ¶0070. A POSITA would have understood and been motivated to apply Wan’s broad applicability to such other controllers.

218. Also as discussed above, to write user data to memory array 810, Wan discloses that “the host controller 820 provides a write command and associated user [data] to the controller 802, which in turn communicates one or more corresponding write commands and the user data to the scrambler/descrambler 812 on the memory die 808.” Ex. 1006 at ¶0070. According to Wan, a key generator 814 within scrambler/descrambler 812 may be used to generate different random or pseudo random keys used to scramble the user data. Ex. 1006 at ¶0070; *see also* ¶0072.

219. Wan further discloses that “the key which is used to scramble a portion of user data can be based on an address of a memory location at which the portion of user data is to be stored” and that “[w]hen the user data is subsequently read back, the same key can be generated for descrambling or other decoding.” Ex. 1006 at ¶0074.

220. As discussed above, Wan also discloses that, prior to sending to the scrambler/descrambler 812, “controller 802 may process the user data such as by performing ECC coding and adding overhead data [*e.g.*, coding redundancy/parity bits] at an encoder/decoder 804. With this approach, the overhead data is also scrambled with the user data. This helps avoid header and ECC type data pattern failures.” Ex. 1006 at ¶0070.

221. Regarding reading user data from the memory array, Wan discloses that when the controller 802 receives a read command from the host controller 820 “The controller 802 provides one or more corresponding read commands to the scrambler/descrambler 812. The scrambler/descrambler 812 may use the key generator 814 to generate the different random or pseudo random keys which were used for scrambling.... The descrambled user data is then provided to the controller 802 and stored in the buffer 807 for read out by the host controller 820.” Ex. 1006 at ¶0071.

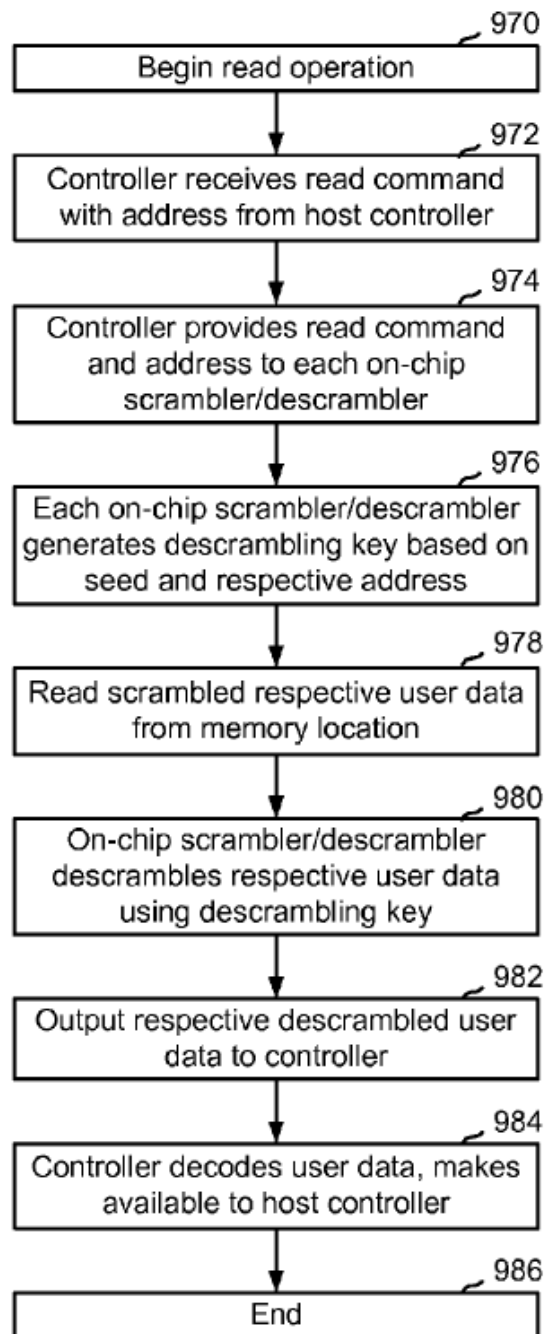
222. Wan depicts the read procedure in detail in Figure 10b, reproduced below and described in Wan as follows:

FIG. 10*b* depicts a descrambling process performed by on-chip circuitry using a key generator. The process corresponds generally to the system of FIG. 8*b* and provides a reverse process to that of FIG. 10*a*. A read operation begins at step 970. At step 972, the controller receives a read command with an address from a host controller. At step 974, the controller provides a read command and address to each on-chip scrambler/descrambler. At step 976, each on-chip scrambler/descrambler generates a scrambling key based on a seed and the address, and optionally, and additional shift, as discussed. At step 978, scrambled user data is read from the memory array, stored in a bit line latch, and accessed by each scrambler/descrambler. At step 980, each on-chip scrambler/descrambler descrambles its respective user data using its respective scrambling key. At step 982, each scrambler/descrambler outputs the descrambled user data to the controller. At step 984, the controller processes the user data such as by performing ECC decoding, and makes the resulting decoded data available to the host controller, such as by storing it in a buffer and setting a READY/BUSY flag to READY. The process ends at

step 986. The descrambler thus is an intermediary which provides additional processing of the user data when it is read from the memory location.

Ex. 1006 at ¶0081.

Fig. 10b



223. Wan also discloses a removable flash storage card embodiment in Figure 12, reproduced below, in which “an off-chip controller 1150 (analogous to

... controller 802 in FIG. 8a and 8b) is included in the same memory device ... as ... one or more memory die.” Ex. 1006 at ¶0089. In this embodiment of Figure 12, “Commands and Data are transferred ... between the controller and the one or more memory die 1198 via lines 1118.” Ex. 1006 at ¶0089. According to Wan, each memory die 1198 contains on-chip scrambler/descrambler circuitry within the read/write circuits 1165 “which provides the functionality described in connection, e.g., with FIGS. 8a to 10b.” Ex. 1006 at ¶0101; Figs. 12-13.

224. A POSITA would have understood that Jo’s flash memory system for interfacing a host with NAND flash memory devices 611 and 612 (as shown in Jo’s Figure 6 above) would benefit from the reliability improvement provided by incorporating Wan’s on die scrambler/descrambler circuitry into Jo’s NAND flash memory die 611 and 612. The benefits and motivation for doing so would, for example include those described in this section and in Section XI.B.1 above.

225. Wan also discloses that “[d]ata scrambling for non-volatile memory such as NAND memories has become increasingly important in addressing system reliability issues that are inherently susceptible by design to worst-case data patterns [including fixed repetitive data,] which trigger memory program failures” and that “[d]ata scrambling randomizes the distribution of data states in the storage elements so that worst-case data patterns are avoided.” Ex. 1006 at ¶0049. This provides a POSITA further motivation to incorporate Wan’s disclosure into Jo. In

other words, Wan itself teaches that modifying Jo's memory devices 611 and 612 to include the scrambler/descrambler circuitry disclosed by Wan and summarized above would improve memory device reliability.

226. Wan further discloses that adding scrambler/descrambler to the memory die "provides portability of the memory die so that [they] can be used with different controllers, for instance, which are made by different manufacturers," for example, flash memory controller 600 disclosed by Jo. Ex. 1006 at ¶0070. As noted above, this would have led a POSITA to try Wan's disclosure in other controllers, such as specifically disclosed by Jo.

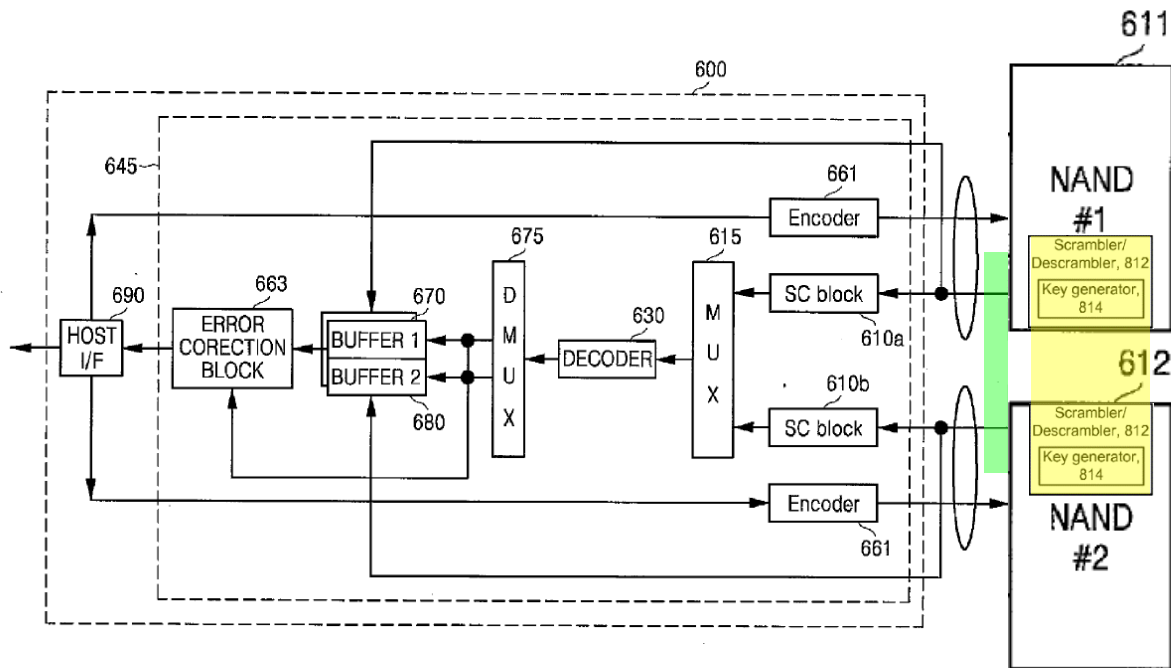
227. A POSITA would likewise have an expectation that this combination of Jo and Wan, closely analogous art as reflected in this Section and Section XI.B.1 above, would be successful; would perform as intended; and could be implemented with minimal, if any experimentation.

228. Wan also explains that "scrambling is considered to be a form of encoding and descrambling is considered to be a form of decoding," and that Wan uses the terms scrambling and descrambling to describe examples "in which random or pseudorandom data is used to encode or decode user data." Ex. 1006 at ¶0052; *see also* ¶0069 ("the controller decodes, e.g., descrambles, the user data"). In other words, a POSITA would have understood that Wan's

scrambler/descrambler circuitry partially encodes and decodes data which is further encoded and decoded by Jo's ECC block 645.

229. Accordingly, Jo's Figure 6 embodiment, as modified by Wan to incorporate on-die scrambler/descrambler circuitry into Jo's NAND flash memory chips 611 and 612 that scrambles ECC encoded data to be stored upon write operations and descrambles, *i.e.*, partially decodes, data retrieved from memory chips 611 and 612 upon read operations prior to ECC decoding, discloses and renders obvious the claimed "a decoding module configured to: receive encoded data from the flash memory; and decode the received encoded data to generate a plurality of partially decoded data streams." The combination is shown in the Figure below, with the "decoding module" in yellow and the "plurality of partially decoded data streams" in green, where Jo's Figure 6 has been annotated to show scrambler/descramblers of Wan's Figures 8a and 8b within Jo's memory chips 611 and 612.

FIG. 6



1[b] “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel,”

230. Jo, as modified by Wan, discloses the claimed “an error detection module communicatively coupled with the decoding module, and comprising a plurality of error detection sub-modules operating in parallel” for the reasons discussed in connection with the combination of Jo and Wu above.

1[c] “each error detection sub-module configured to: receive a different one of the plurality of partially decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module; and”

231. Jo, as modified by Wan, discloses the claimed “each error detection sub-module configured to: receive a different one of the plurality of partially

decoded data streams; detect whether a portion of the respective received stream contains an error; and forward the portion of the respective received stream containing an error to an error correction module” for the reasons discussed in connection with the combination of Jo and Wu above.

1[d] “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error.”

232. Jo discloses the claimed “the error correction module, communicatively coupled with and physically separate from the error detection module, and configured to correct the received portions of the respective received streams containing an error” for the reasons discussed in connection with the combination of Jo and Wu above.

233. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wan disclose all of the limitations of claim 1 of the ’700 Patent, and that such combination renders claim 1 obvious.

2. Claim 2

234. Jo discloses the limitations of claim 2, as discussed above in connection with Jo and Wu, *i.e.*, syndrome computation block 610a, *i.e.*, a first “error detection sub-module,” is configured to read data from a set of sectors in NAND memory 611, and error detection sub-module 610b, *i.e.*, a second “error

detection sub-module,” is configured to read data from a set of sectors in NAND memory 612.

235. Claim 2 is rendered obvious by the combination of Jo and Wan for the same reasons provided in Section XI.B.2 above.

236. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wan disclose all of the limitations of claim 2 of the ’700 Patent, and that such combination renders claim 2 obvious for the same reasons as claim 1.

3. Claim 4

237. Jo discloses the limitations of claim 4, as discussed above in connection with Jo and Wu.

238. Claim 4 is rendered obvious by the combination of Jo and Wan for the same reasons provided in Section XI.B.3 above.

239. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wan disclose all of the limitations of claim 4 of the ’700 Patent, and that such combination renders claim 4 obvious for the same reasons as claim 1.

4. Claim 8

240. Jo discloses the limitations of claim 8, as discussed above in connection with Jo and Wu.

241. Claim 8 is rendered obvious by the combination of Jo and Wan for the same reasons provided in Section XI.B.4 above.

242. Thus, for the reasons I've explained above, a POSITA would have understood that the combination of Jo and Wan disclose all of the limitations of claim 8 of the '700 Patent, and that such combination renders claim 8 obvious for the same reasons as claims 1 and 4.

5. Claims 15 and 17

243. As I explained above with respect to my discussion of Wu anticipating claims 15 and 17, I note that independent claims 15 and 17 are very similar to apparatus claim 1, except that claim 15 is a “system” claim and claim 17 is a “method” claim and the limitations 15[a] and 15[b]. Therefore, claims 15 and 17 are obvious in view of Jo and Wan for the effectively same reasons as claim 1.

15[a] “an encoder configured to encode data for storage on a flash memory,”

15[b] “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data; and”

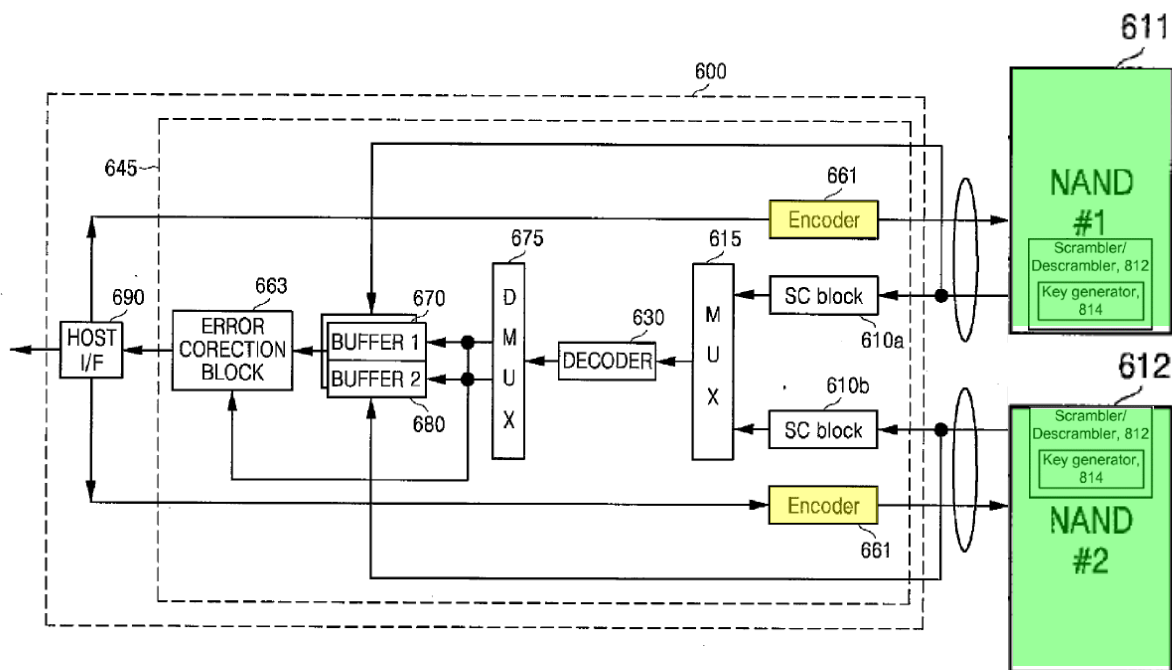
244. As I explained above, Figure 6 of Jo, reproduced below, depicts a “Flash memory controller 600” that “includes two data encoders 661, one each for the first and second communication channels.” Ex. 1005 at ¶0037. Jo discloses that “In operation, ECC block 645 communicates data with memory devices 611 and 612 over two communication channels operating at the same time as each

other in parallel. When data from a host device (e.g., a processor) is to be written to the memory devices via the two communication channels, the data is first encoded (e.g. with a BCH code) by a corresponding encoder 661 for each communication channel. Then, the encoded data is transmitted in parallel across the two communication channels to memory devices 611 and 612.” Ex. 1005 at ¶0038; *see also* ¶0036 (disclosing that memory devices 611 and 612 are NAND flash memory).

245. A POSITA would have thus understood that Jo discloses “an encoder,” encoders 661 in yellow below and collectively forming the recited “encoding module,” “configured to encode data for storage on a flash memory,” comprised of memory devices 611 and 612 in green, and “the flash memory, communicatively coupled with the encoding module, and configured to store the encoded data.” The below discloses Figure 6 of Jo annotated to include scrambler/descramblers of Wan’s Figures 8a and 8b within Jo’s memory chips 611 and 612. *See also* Ex. 1006 at ¶0070 (Wan disclosing a data storage operation in which Wan’s “host controller 820 provides a write command and associated user [data] to the [Wan’s] controller 802, which in turn communicates one or more corresponding write commands and the user data to the scrambler/descrambler 812 on the memory die 808. The controller 802 may process the user data such as by

performing ECC coding and adding overhead data at an encoder/decoder 804 [prior to sending the data to the scrambler/descrambler 812].”), Fig. 8a.

FIG. 6



246. Thus, for the reasons I’ve explained above, a POSITA would have understood that the combination of Jo and Wan disclose all of the limitations of claims 15 and 17 of the ’700 Patent, and that such combination renders these claims obvious for the same reasons as claim 1.

6. Claim 19

247. Claim 19 depends from claim 17 and contains the same limitations as claim 4, which depends from claim 1:

Claim 19	Claim 4
The method of claim 17, <u>wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.</u>	The flash memory decoder of claim 1, <u>wherein the error correction module comprises a plurality of error correction sub-modules operating in parallel and fewer in number than the error detection sub-modules.</u>

248. The limitations of claim 19 are therefore disclosed (and obvious over) the combination of Jo and Wan for the reasons as I discussed above for claims 1, 4, and 17.

XII. CONCLUDING STATEMENTS

249. In signing this declaration, I understand that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I acknowledge that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross-examination within the United States during the time allotted for cross-examination.

250. I declare that all statements made herein of my knowledge are true and correct and that all statements made on information and belief are believed to be true. I also declare that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 35 U.S.C. § 1001.

251. I declare under the penalty or perjury that the foregoing is true and correct.

Executed on 6/10/22.

A handwritten signature in black ink, appearing to read 'plecek', written over a horizontal line.

Lara Dolecek, Ph.D.

APPENDIX A

Lara Dolecek, PhD

CONTACT INFORMATION

56-147b, Engineering IV
Department of Electrical and Computer Engineering
University of California, Los Angeles
Los Angeles, CA 90095 USA

Voice: (310) 825-2108
Fax: (310) 206-8495
E-mail: dolecek@ee.ucla.edu
Web: <https://loris.seas.ucla.edu/>

ACADEMIC EMPLOYMENT

University of California, Los Angeles, Los Angeles, California, USA

Full Professor, Department of Electrical and Computer Engineering, Department of Mathematics (by courtesy), 2018 - present.

Associate Professor, Department of Electrical and Computer Engineering, 2015 - 2018.

Assistant Professor, Department of Electrical Engineering, 2009 - 2015.

- Research, graduate, and undergraduate teaching and mentoring in coding theory, with applications to modern data storage, computing, and information processing systems.
- Founder and Director of Laboratory for Robust Information Systems (LORIS).

Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

Postdoctoral Research Associate, Department of Electrical Engineering and Computer Sciences, 2007 - 2009.

- Statistical inference and fast sampling methods for complex networks; Advisor: Devavrat Shah

EDUCATION

University of California, Berkeley, Berkeley, California, USA

PhD, Electrical Engineering and Computer Sciences, Dec. 2007

- **2007 David J. Sakrison Memorial Prize for the most outstanding doctoral research in the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley.**
- Dissertation Title: “Rethinking the Minimum Distance: Channels With Varying Sampling Rate and Iterative Decoding of LDPC Codes”
- Major: Coding and Communications Theory; Advisor: Venkat Anantharam

University of California, Berkeley, Berkeley, California, USA

M.A., Statistics, May 2007

- Thesis title: “Applications of importance sampling methods to evaluation of the low frame error rate performance of LDPC codes”; Advisor: Martin J. Wainwright

University of California, Berkeley, Berkeley, California, USA

M.S., Electrical Engineering and Computer Sciences, May 2004

- Thesis title: “Runlength properties of a Reed-Muller (1,m) code with applications in channels with synchronization errors”; Advisor: Venkat Anantharam

University of California, Berkeley, Berkeley, California, USA

B.S. with honors, Electrical Engineering and Computer Sciences, Dec. 1999

HONORS AND AWARDS

- Board of Governors, IEEE Information Theory Society, Elected Member 2022-2024.
- Distinguished Lecturer, IEEE Information Theory Society, 2021-2023.
- Memorable Paper Award at the 2021 Non-Volatile Memories Workshop, UC San Diego.
- IEEE Data Storage Society 2018-2019 Best Student Paper Award.
- Runner up for the Memorable Paper Award at the 2019 Non-Volatile Memories Workshop, UC San Diego.
- Memorable Paper Award at the 2018 Non-Volatile Memories Workshop, UC San Diego.
- Runner Up for the Memorable Paper Award at the 2018 Non-Volatile Memories Workshop, UC San Diego.
- Best of SELSE Award, March 2018.
- Best Paper Award at the Embedded Systems Week – International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES), Nov. 2017.
- IEEE Data Storage Society 2016 Best Student Paper Award.
- Best of SELSE Award, March 2016 – received two awards, for two separate papers.
- Best Paper Award, IEEE Globecom Conference, 2015.
- IBM Faculty Award, 2014.
- Northrop Grumman Excellence in Teaching Award, 2013, UCLA School of Engineering.
- Intel Early Career Faculty Award, 2013.
- Faculty Development Award, University of California, 2013.
- Okawa Foundation Research Grant, 2013, one of six nation-wide recipients for innovative research in Information Technologies.
- NSF CAREER Award, 2012.
- Hellman Fellow, a campus-wide program established to support promising assistant professors, UCLA 2011.
- David J. Sakrison Memorial Prize for the most outstanding dissertation in Electrical Engineering and Computer Sciences, UC Berkeley, 2007.
- Dissertation Year Fellowship, UC Berkeley, 2006-2007.
- GAANN Graduate Fellowship, UC Berkeley, 2004-2005.
- Mentored Research Award, UC Berkeley, 2003-2004.
- Eugene Cota-Robles Fellowship, UC Berkeley, 2000-2003.
- Marian Wojciech Para Memorial Prize, UC Berkeley, 1999.
- Member of HKN, TBP and Golden Key Honor societies.
- Senior member of *IEEE* since 2012.

PROFESSIONAL
ACTIVITIES

- Distinguished Lecturer of the *IEEE Information Theory Society*, 2021-2023.
- Secretary of the *IEEE Information Theory Society*, 2019 – 2021.
- Editorship
 - Associate Editor for Coding Theory for *IEEE Transactions on Information Theory*, 2019 – present.
 - Guest Editor for the Special Issue dedicated to V. I. Levenshtein, *IEEE Transactions on Information Theory*, 2020.
 - Associate Editor for Coding Theory for *IEEE Transactions on Communications*, 2013 – 2019.
 - Associate Editor for Coding Theory and Sequences for *IEEE Communications Letters*, 2012 – 2015.
 - Lead guest editor for *IEEE JSAC Special Issue on Communication Methodologies for the Next-Generation Storage Systems*, 2013 – 2014.
- Conference Chairing
 - Technical Program Committee Chair, IEEE Information Theory Workshop, 2022.
 - Program Chair, Non-Volatile Memories Workshop (NVMW), 2021.
 - Chair of Data Storage Symposium at *IEEE Globecom*, Austin, TX, Dec. 2014.
 - Co-chair of Data Storage and Applications Symposium at *IEEE ICNC*, San Diego, CA, Jan. 2013.
- Technical Program Committee Membership
 - *IEEE International Symposium on Information Theory (ISIT)*, 2022; *IEEE International Conference on Communications (ICC)*, 2022; *IEEE Information Theory Workshop (ITW)*, 2021; *IEEE International Symposium on Information Theory (ISIT)*, 2021; *IEEE International Conference on Communications (ICC)*, 2021; *Non-Volatile Memories Workshop (NVMW)*, 2020; *IEEE International Symposium on Information Theory (ISIT)*, 2020; *Third Conference on Systems and Machine Learning*, 2020; *IEEE Information Theory Workshop (ITW)*, 2019; *IEEE Globecom*, 2019; *IEEE International Symposium on Information Theory (ISIT)*, 2019; *IEEE Globecom*, 2018; *IEEE International Conference on Communications (ICC)*, 2018; *IEEE International Symposium on Turbo Coding (ISTC)*, 2018; *IEEE Globecom*, 2017; *IEEE International Conference on Communications (ICC)*, 2017; *IEEE International Symposium on Information Theory (ISIT)*, 2016; *IEEE International Symposium on Turbo Coding (ISTC)*, 2016; *IEEE International Conference on Communications (ICC)*, 2016; *IEEE International Symposium on Information Theory (ISIT)*, 2015; *IEEE Information Theory Workshop (ITW)*, 2015; *IEEE Globecom*, 2015; *IEEE International Conference on Communications (ICC)*, 2015; *IEEE International Symposium on Information Theory (ISIT)*, 2014; *IEEE International Symposium on Turbo Coding (ISTC)*, 2014; *IEEE Information Theory Workshop (ITW)*, 2014; *IEEE Globecom*, 2014; *IEEE International Conference on Communications (ICC)*, 2014; *IEEE Information Theory Workshop (ITW)*, 2013; *IEEE Globecom*, 2013; *IEEE International Conference on Communications (ICC)*, 2013; *Non-volatile Memories Workshop (NVMW)*, 2013; *IEEE International Symposium on Turbo Coding (ISTC)*, 2012; *IEEE International Conference on Communications (ICC)*, 2012; *IEEE Globecom*, 2012; *IEEE International Conference on Communications and Networking (ICNC)*, 2012; *Non-volatile Memories Workshop (NVMW)*, 2012; *IEEE Globecom*, 2011; *IEEE International Conference on Communications (ICC)*, 2011; *IEEE International Symposium on Computers and Communications (ISCC)*, 2011.
- Member of *IEEE Information Theory Society Committee on New Directions*

- Member of *IEEE Standards Working Group* for developing standards that specify the advanced error correction coding for non-volatile memories
- Selected Tutorial Presentations
 - “Modern Coding Theory And Connections to Error-Efficient Computing,” presented at the *SNF Workshop on Theory and Practice of Error-Efficient Computing Systems*, Mar. 2017, EPFL, Switzerland.
 - “Making Error Correcting Codes Work for Flash Memory,” a 3-hr tutorial at *Flash Memory Summit*, Santa Clara, CA, Aug. 2014, co-presented with Dr. Steve Hetzler and Prof. Anxiao Jiang.
 - “Channel Coding Methods for Emerging Data Storage and Memory Systems: Opportunities to Innovate Beyond the Hamming Metric,” a 3-hr tutorial, *IEEE International Symposium on Information Theory (ISIT)*, Jun. 2014, co-presented with Prof. Anxiao Jiang.
 - “Coding Methodologies for Emerging Data Storage Systems: Opportunities and Challenges,” a 3-hr tutorial presented at the *27th Annual Symposium on Communications*, Kingston, Canada, Jun. 2014.
 - “Making Error Correcting Codes Work for Flash Memory,” a 3-hr tutorial at *Flash Memory Summit*, Santa Clara, CA, Aug. 2013, co-presented with Mr. Charles Camp and Prof. Anxiao Jiang.
 - “Coding Methods for Emerging Storage Systems,” a 3-hr tutorial at *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2012, co-presented with Prof. Anxiao Jiang.
 - “High-Performance Graph-Based Codes,” a 3-hr tutorial at *IEEE AUSCTW 2011*, Melbourne, Australia, Jan. 2011.
- Session Organizer
 - *IEEE Information Theory Workshop*, 2018, session title: “Coding for Memories”; *IEEE Information Theory Workshop*, 2017, session title: “Coding for Memories”; *IEEE Asilomar 2017*, session title: “DNA Storage”; *IEEE Asilomar 2015*, session title: “Coding and Signal Processing for Modern Memories”; *IEEE Information Theory and Applications Workshop*, 2014, session title: “Variability-Aware Information Processing”; *IEEE Information Theory and Applications Workshop 2013*; session title “Fault-Tolerant Information Processing”; *IEEE Allerton 2013*, session title “Coding for Deletion Channels”; *IEEE Asilomar 2012*, session title: “Coding-Theoretic Methods for the Next-Generation Storage Systems”; *IEEE Allerton 2011*, session title “Information Theory for Circuits”.
- University Coordinator and Student Outreach Organizer, *Flash Memory Summit*, 2013, 2014, 2016, and 2017.
- Reviewer for *IEEE Transactions on Information Theory*, *IEEE Transactions on Communications*, *IEEE Transactions on Circuits and Systems*, *Eurasip Journal on Signal Processing*, *IEEE Journal on Special Topics in Communications*, *IEEE Journal on Special Topics in Signal Processing*, *Codes, Designs and Cryptography* and numerous *IEEE* conferences.
- Panelist for National Science Foundation (NSF), US-Israel Binational Research Foundation, Qatar National Research Foundation (QNRF)
- Area Director, Signals and Systems, ECE Department, UCLA, 2020-present.
- Member of the Recruitment Committee, ECE Department, UCLA, 2019-present.
- Hellman Fellows Selection Committee, UCLA, 2019.
- ECE Department Representative to UCLA School of Engineering Faculty Executive Committee, 2017-2020.

UNIVERSITY
SERVICE

- Co-organizer of Undergraduate Visit Day, EE Department, UCLA, 2017.
- Member of an Ad-Hoc Recruitment Evaluation Committee, ECE Department, UCLA, 2017-present.
- Member of an Ad-Hoc Promotion Committee, ECE Department, UCLA, 2019-present.
- Member and then Chair of the Non-Tenure Evaluation Committee, ECE Department, UCLA, 2016-present.
- Hellman Fellows Selection Committee, UCLA, 2016.
- EE Department Representative to UCLA Legislative Assembly, 2014-2015.
- Chair of the EE Department Annual Research Review, 2012.
- Co-Chair of the EE Department Annual Research Review, 2011.
- Organizer of the EE Department Seminar Series, Fall 2010.
- Member of the Graduate Admissions Committee, EE Department, UCLA, 2010-2011.
- Speaker at various workshops and townhalls for ECE students (Tips on Getting into Graduate Schools, Post-College Career Choices, etc.), EE Department, UCLA, 2011-present.

CURRICULUM DEVELOPMENT

- Introduction and complete development of “ECE M148: Introduction to Data Science,” ECE Department/Computer Engineering Track, UCLA, 2022-present.
- Introduction and complete development of “ECE M146: Introduction to Machine Learning,” ECE Department/Computer Engineering Track, UCLA, 2017-present.
- Introduction and complete development of “ECE 235A: Mathematical Foundations of Data Storage Systems,” EE Department, UCLA, 2014-present.
- Introduction and complete development of “EE 239A: Special topics: Statistical Inference and Monte Carlo Methods,” EE Department, UCLA, 2011-present.
- Introduction and complete development of “ENG 116: Statistics for Management Decisions,” (co-taught with Prof. C. Barz/Prof. H. Mamani) Engineering Masters Online (MSOL) Program, UCLA, 2016.
- Other classes taught at UCLA: “EE103: Applied Numerical Computing”, “EE131A: Probability”, “EE132A: Introduction to Communication Systems” (all high enrollment sophomore/junior classes); “EE205A: Matrix Analysis for Engineers,” “EE231A: Information and Coding Theory”.

SELECTED INVITED TALKS AND SEMINARS (TUTORIALS LISTED EARLIER)

- L. Dolecek, “Overcoming Data Availability Attacks in Blockchain Systems: A Graph-Coding Perspective,” **Plenary Talk**, *National Conference on Communications (NCC)*, India, May 2022.
- L. Dolecek. “How to Correct Errors ?” *Math Prize for Girls*, **Plenary Speaker**, November 2021.
- L. Dolecek, “Overcoming Data Availability Attacks in Blockchain Systems: A Graph-Coding Perspective,” *ECE Department Seminar, Rutgers University*, October 2021.
- L. Dolecek, “Context-Aware Coding for Computing Memories,” *Approximate Systems Dagstuhl Seminar* (by invitation only), July 2021.
- L. Dolecek, “Overcoming Data Availability Attacks in Blockchain Systems: A Graph-Coding Perspective,” *ECE Department Seminar, University of Toronto, Canada*, June 2021.
- L. Dolecek, “Spatially-Coupled Codes: Recent Trends and Applications in Data Storage and Memories,” *IEEE Rourkela Subsection*, India, June 2021.
- L. Dolecek, “Overcoming Data Availability Attacks in Blockchain Systems: A Graph-Coding Perspective,” *London Symposium on Information Theory (LSIT)*, May 2021.

- L. Dolecek, "Modulation and Coding for Energy-Time Entanglement Quantum Key Distribution," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2020.
- L. Dolecek, "Context-Aware Coding for Computing Memories," *Center for Cyber Physical Systems and the Internet of Things Seminar Series*, University of Southern California, Los Angeles, CA, Dec. 2019.
- L. Dolecek, "A Finite-Length Construction of Irregular Spatially-Coupled Codes," *IEEE Information Theory Workshop*, Visby, Sweden, Aug. 2019.
- L. Dolecek, "Spatially-Coupled Codes: Recent Trends and Applications in Data Storage and Memories," **Plenary Talk**, *The 10th Annual Non-Volatile Memories Workshop*, UC San Diego, La Jolla, CA, Mar. 2019.
- L. Dolecek, "Error Correction and Detection for Computing Memories Using System Side Information," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2019.
- L. Dolecek, "Recent Advances in Spatially Coupled codes and Applications in Data Storage Systems," *Joint Mathematics Meeting*, American Mathematical Society, Baltimore, MD, Jan. 2019.
- L. Dolecek, "Spatially Coupled Codes for Modern Memories," **Plenary Talk**, *IEEE International Symposium on Turbo Coding (ISTC)*, Hong Kong, Hong Kong, Dec. 2018.
- L. Dolecek, "Error Correction and Detection for Computing Memories Using System Side Information," *IEEE Information Theory Workshop*, Guangzhou, China, Nov. 2018.
- L. Dolecek, "Robust Computing Under Hardware Noise: A Coding Theoretic Perspective," **Keynote Talk**, *The 7th Annual Henry Taub TCE Conference*, Technion, Haifa, Israel, Jun. 2017.
- L. Dolecek, "New Trends in Coding Theory & Opportunities in Modern Memories and Computing," *Electrical Engineering Systems Seminar, Caltech*, Pasadena, CA, May 2017.
- L. Dolecek, "Coding Theoretic Techniques for Modern Memories," **Plenary Talk**, *American Mathematical Society Annual Meeting*, Atlanta, GA, Dec. 2016.
- L. Dolecek, "Coding Theoretic Methods for Robust and Approximate Computing," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2017.
- L. Dolecek, "Flash Memories in High Radiation Environments: LDPC Decoder Study," *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2016.
- L. Dolecek, "Coding Theory for Robust Computing: Models, Tools, Applications," **Plenary Talk**, *IEEE International Symposium on Turbo Coding (ISTC)*, Brest, France, Sep. 2016.
- L. Dolecek, "Coding Theory for Modern Computing," *Communications Seminar, Texas A&M University*, College Station, TX, Apr. 2016.
- L. Dolecek, "ECC-based Iterative Computing Under Uncertainty: Theory and Applications," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2016.
- L. Dolecek, "New Problems in Coding Theory with Applications to Modern Data Storage and Memories," *Simons Institute*, UC Berkeley, Berkeley, CA, Feb. 2015.
- L. Dolecek, "Synchronizing Files Under a Large Number of Edits: Theory and Practice," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2015.
- L. Dolecek, "Mathematical Foundations of the Next-Generation Robust and Resource-Efficient Information Systems: Algorithms and Code Designs," *ISL Seminar, UIUC*, Urbana, IL, Nov. 2014.
- L. Dolecek, "Mathematical Foundations of the Next-Generation Robust and Resource-Efficient Information Systems: Algorithms and Code Designs," *Communications Theory and Systems Seminar, UC San Diego*, La Jolla, CA, Nov. 2014.

- L. Dolecek, N. Bitouze, F. Sala, and S. M. S. Tabatabaei Yazdi, "A Practical Framework for Efficient File Synchronization," *Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2013.
- L. Dolecek, "Channel Coding Methods for Green Data Storage: Innovations Beyond the Hamming Metric," *ISL Colloquium, Stanford University*, Palo Alto, CA, Jun. 2013.
- L. Dolecek, "Channel Coding Methods for Green Data Storage: Innovations Beyond the Hamming Metric," *Rutgers University*, New Brunswick, NJ, Jun. 2013.
- L. Dolecek, "Channel Coding Methods for Green Data Storage: Innovations Beyond the Hamming Metric," *Communications, Networks and Systems Seminar, University of Southern California*, Los Angeles, CA, May 2013.
- L. Dolecek, "Exploiting Variability in Flash Memories Through Non-binary Error Correction Coding," *DATE Workshop on Software Approaches to Resilient System Design*, Grenoble, France, Mar. 2013.
- L. Dolecek, C. Huang, and S. M. S. Tabatabaei Yazdi, "Analysis of iterative decoders under processing errors," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2013.
- S. Tabatabaei Yazdi and L. Dolecek, "Synchronization from Deletions Through Interactive Communication," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2013.
- L. Dolecek, C. Huang, and S. M. S. Tabatabaei Yazdi, "Analysis of Iterative Decoders Under Processing Errors," *Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2012.
- L. Dolecek, "Coding-Theoretic Strategies for the Emerging Storage Systems," *Princeton University*, Princeton, NJ, Feb. 2012.
- S. M. S. Tabatabaei Yazdi and L. Dolecek, "LDPC Decoders on Unreliable Hardware: Fundamental Limits and Practical Implications," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2012.
- L. Dolecek, R. Gabrys, E. Yaakobi, L. Grupp, and S. Swanson, "Flexible Coding Strategies for Next-Generation Storage Systems," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2012.
- L. Dolecek, "Flexible Coding Schemes With Applications to Emerging Memory Technologies," *American Mathematical Society (AMS) Annual Meeting*, Boston, MA, Jan. 2012.
- L. Dolecek, R. Gabrys, E. Yaakobi, L. Grupp, and S. Swanson, "Tackling Temporal Variability in Multilevel Flash: New Error-Control Code Design and Architectural Validation," *Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2011.
- L. Dolecek, "Flexible Coding Schemes With Applications to Emerging Memory Technologies," *Aspects of Coding Theory Workshop, EPFL*, Lausanne, Switzerland, Jun. 2011.
- L. Dolecek, "Coding Methods for Emerging Non-Volatile Memories," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2011.
- L. Dolecek, "Using Structural Properties of Rare Events for Better System Design: Examples From Codes and Circuits," *JPL-NASA*, Pasadena, CA May 2010.
- L. Dolecek, "Using Structural Properties of Rare Events for Better System Design: Examples From Codes and Circuits," *Cornell University*, NY, NY, Apr. 2010.
- L. Dolecek, "Using Structural Properties of Rare Events for Better System Design: Examples From Codes and Circuits," *Communications, Networks and Systems Seminar, University of Southern California*, Los Angeles, CA, Apr. 2010.
- L. Dolecek, "On Absorbing Sets of Structured Sparse Graph codes," *Information Theory and Applications Workshop*, UC San Diego, La Jolla, CA, Feb. 2010.

INDUSTRIAL
EXPERIENCE

Consultant for law firms, 2015-present

- Provided expert advice on various memory and communications technologies.

Consultant for a distributed data storage start-up, 2016-2018

- Provided expert advice on channel codes for new storage applications.

Consultant for an SBIR start-up, 2015-2016

- Provided expert advice on LDPC codes for new memories.

Consultant for Cadence, 2012

- Provided expert advice on LDPC codes for new memories.

Consultant for a Bay Area Startup, Santa Clara, CA, 2009

- Provided expert advice and tutorial sessions on the implementation of Reed-Solomon and LDPC decoders for high-throughput optical links.

Research and Development Engineer, Agilent Technologies, Santa Rosa, CA, 2000

- Developed platform independent detection and decoding algorithms and implemented them in C++ and Matlab for various testing instruments used in communications systems.

FUNDING SUPPORT

- National Science Foundations (NSF) Grant title: “QuIC-TAQS: A high-dimensional multi-access scalable testbed for the interconnected quantum network,” 2021-2025. PI: C. W. Wong, co-PIs: L. Bouchard, L. Dolecek, G. Smith, and A. Faraon.
- National Science Foundation (NSF), CCF-CIF program. Grant title: “FET: Small: Towards full photon utilization by adaptive modulation and coding on quantum links,” 2020-2022. PI: L. Dolecek, Co-PI: C. W. Wong.
- SanDisk Corp., Grant title: “Binary Spatially Coupled Codes for Flash Memories,” 2017-present. PI: L. Dolecek.
- National Science Foundation (NSF), BSF-CCF-CIF program. Grant title: “Coding for Fast Storage Access and In-Memory Computing,” 2017-2020. PI: L. Dolecek.
- Qualcomm Inc. through Qualcomm Innovation Fellowship for my PhD student Clayton Schoeny, 2016-2017.
- UC Mexus Collaborative Research Program. Grant title: “Channel Coding and Signal Processing for Resilient, Ultra-Dense Memories,” 2016-2017. PI: L. Dolecek.
- National Science Foundation (NSF), CCF-CIF program. Grant title: “Synchronization and Deduplication of Distributed Coded Data: Fundamental Limits and Algorithms,” 2015-2018. UCLA PI: L. Dolecek.
- Advanced Storage Technologies Consortium (ASTC), research grant for research on advanced coding methods for magnetic recording applications, 2015-present. PI: L. Dolecek.
- JPL-NASA: Research and Technology Development Fund. Grant title: “Breaking the Limitations of Radiation-Hardened Devices,” 2014-2017. UCLA PI: L. Dolecek.
- Grant from Micron Technologies to support research in CODESS Center, 2014-2018. PIs: L. Dolecek and R. Wesel.

- Intel Early Career Faculty Honor Program, funding for research on modern data storage and data processing system design, 2013-present. PI: L. Dolecek.
- Okawa Research Grant, funding for research on advanced coding methods for modern data storage systems, 2013-2014. PI: L. Dolecek.
- National Science Foundation (NSF): CAREER program. Grant title: “Channel Coding Paradigms for Next Generation Storage Systems,” 2012-2018. PI: L. Dolecek.
- The Intelligence Advanced Research Project Activity (IARPA). Grant title: “Reliable Inference with Missing, Masked, Malfunctioning, or Malicious Sensors (RIM4S),” 2012-2013. PI: L. Dolecek, co-PIs: D. Cabric, M. Srivastava, and G. Pottie.
- National Science Foundation (NSF): CCF-CIF program. Grant title: “Collaborative Research: Spatially Coupled Sparse Codes on Graphs - Theory, Practice, and Extensions,” 2012-2016. L. Dolecek is UCLA PI, other PIs: D. Costello, T. Fuja, J. Kliever, and R. Smarandache.
- National Science Foundation (NSF): CCF-CIF program. Grant title: “Collaborative Research: Code Design and Analysis to Approach Capacity with Short Blocklengths Using Feedback,” 2012-2016. PI: R. Wesel, co-PIs: B. Daneshrad and D. Divsalar, and L. Dolecek.
- National Science Foundation (NSF): Expeditions in Computing program. Grant title: “Collaborative Research: Variability-Aware Software for Efficient Computing with Nanoscale Devices,” 2011-2018. PI: M. Srivastava, co-PIs: L. Dolecek and P. Gupta.
- IBM, gift funding for research on reliable data storage, 2014-present. PI: L. Dolecek.
- Intel Corporation, gift funding for research on reliable data storage, 2012-present. PI: L. Dolecek.
- Advanced Storage Technologies Consortium (ASTC), gift funding and research grant for research on non-binary LDPC codes for data storage systems, 2011-2014. PI: L. Dolecek, co-PI: R. Wesel.
- UC Faculty Development Award, funding for research on advanced coding methods for modern data storage systems, 2013-2014. PI: L. Dolecek.
- UC Discovery Program. Grant title: “Codes with provable guarantees for storage applications,” 2011-2013. PI: L. Dolecek.
- Western Digital Corporation. Grant title: “Codes with provable guarantees for storage applications,” 2011-2013. PI: L. Dolecek.
- Hellman Fellows Program, funding for research on coding methods for reliable data storage, 2011-2012. PI: L. Dolecek.
- UC Mexus Collaborative Research Program. Grant title: “Channel Codes for Emerging Highly Reliable Systems: Efficient Algorithms and Performance Guarantees,” 2012-2014. PI: L. Dolecek.

MENTORSHIP

Current Graduate Students

- Debnab Mitra (PhD), ECE Department, UCLA, 2020-present.
- Lev Tauz (PhD), ECE Department, UCLA, 2020-present. Winner of the 2019 Preliminary Exam Award in Signals and Systems, ECE Department, UCLA.

Current Undergraduate Students

- Justin Shao, ECE Department, UCLA, 2020-present.

Past Post-Doctoral Researchers

- Nicolas Bitouze, (PhD, ENST, France, 2013), EE Department, UCLA, 2012-2014.
- Yao Li (PhD, Rutgers University, 2012), EE Department, UCLA, 2012-2014, now Senior Engineer at Akamai.

- Seyed M. Sadegh Tabatabaei Yazdi (PhD, Texas AM, 2011), EE Department, UCLA, 2011-2012, next Senior Engineer at Marvell.

Graduated PhD Students

- Siyi Yang, ECE Department, UCLA, PhD, 2021. Dissertation title: “Application-Driven Coding Techniques: From Cloud Storage to Quantum Communications.”
 - Qualcomm Innovation Fellowship Finalist.
 - Next: Postdoctoral Researcher, Duke University
- Zehui Chen, ECE Department, UCLA, PhD, 2021. Dissertation title: “Coding Assisted Methods for Crossbar Resistive Memory.”
 - Qualcomm Innovation Fellowship Finalist.
 - Next: Senior Engineer, Huawei.
- Homa Esfahanizadeh, ECE Department, UCLA, PhD, 2019. Dissertation title: “Spatially-Coupled Codes for Modern Data Storage Systems.”
 - Next: Post-doctoral researcher at the Massachusetts Institute of Technology (MIT).
- Ahmed Hareedy, ECE Department, UCLA, PhD, 2018. Dissertation title: “Graph-Based Error Correcting Codes for Modern Dense Storage Devices.”
 - Winner of 2016-2017 ECE Department Excellence in Teaching Award. Winner of 2017-2018 UCLA Dissertation Year Fellowship. Winner of 2016-2017 Outstanding Dissertation Award in Signals and Systems, ECE Department, UCLA.
 - Next: post-doctoral researcher at Duke University, then Assistant Professor at the Middle Eastern Technical University (METU), Ankara, Turkey.
- Clayton Schoeny, ECE Department, PhD, 2018. Dissertation title: “Coding for Future Large-Scale Data Systems.”
 - Winner of 2016-2017 ECE Department Excellence in Teaching Award. Winner of 2017-2018 UCLA Dissertation Year Fellowship. Winner of 2016-2017 Qualcomm Innovation Fellowship. Winner of 2016-2017 Outstanding Dissertation Award in Signals and Systems, ECE Department, UCLA.
 - Next: joined Fair as a Senior Data Scientist.
- Frederic Sala, ECE Department, UCLA, PhD, 2016. Dissertation title: “Algorithms and Coding Techniques for Reliable Data Management and Storage.”
 - Winner of 2015-2016 UCLA Dissertation Year Fellowship. Winner of 2016-2017 Outstanding Dissertation Award in Signals and Systems, ECE Department, UCLA.
 - Next: post-doctoral researcher at Stanford University, then Assistant Professor at the University of Wisconsin.
- Behzad Amiri, EE Department, UCLA, PhD, 2015. Dissertation title: “Theory and Practice of Non-Binary Graph Based Codes: A Combinatorial View.”
 - Next: Senior Engineer at Pure Storage.
- Chu-Hsiang Huang, EE Department, UCLA, PhD, 2015. Dissertation title: “Iterative Information Processing on Unreliable Hardware: An Information Theoretic Approach.”
 - Next: Senior Engineer at Qualcomm.
- Ryan Gabrys, EE Department, UCLA, PhD, 2014. Dissertation title: “Channel Coding Strategies for the Emerging Data Storage Systems.”

- Now: Senior Scientist at Army-Spawar and Research Associate Professor at UC San Diego.
- Jiadong Wang (co-advised with R. Wesel), EE Department, UCLA, PhD, 2012. Dissertation title: “Absorbing Set Analysis of LDPC Codes and Read-Channel Quantization in Flash Memory.”
- Now: Senior Engineer at Qualcomm.

Graduated MS Students (thesis option)

- Debarnab Mitra, ECE Department, UCLA, M.S., 2020. Best Masters Thesis in Signals and Systems, ECE Department, UCLA.
- Lev Tauz, ECE Department, UCLA, M.S., 2020.
- Siyi Yang, ECE Department, UCLA, M.S., 2018.
- Zehui Chen, ECE Department, UCLA, M.S., 2018.
- Shahroze Kabir, ECE Department, UCLA, M.S., 2017.
- Kayvon Mazooji, EE Department, UCLA, M.S., 2016.
- Amirhossein Reisizadeh, EE Department, UCLA, M.S., 2016.
- Chimmayi Lanka, EE Department, UCLA, M.S., 2016.
- Clayton, Schoeny, EE Department, UCLA, M.S., 2014.
- Eric Goss, EE Department, UCLA, M.S., 2014.
- Frederic Sala, EE Department, UCLA, M.S., 2013. Best Masters Thesis in Signals and Systems, EE Department. Outstanding Masters Thesis Award, UCLA School of Engineering.
- Yizeng Sun, EE Department, UCLA, M.S., 2013.
- Behzad Amiri, EE Department, UCLA, M.S., 2012.
- Zhujun Lin, EE Department, UCLA, M.S., 2012.
- Ben-Yue Chang, EE Department, UCLA, M.S., 2011.
- Chi-Wei Lin, EE Department, UCLA, M.S., 2011.
- Yifan Sun, EE Department, UCLA, M.S., 2011.

Past Academic Visitors

- Laura Conde-Cinencia (professor, University of South Brittany, France), EE Department, 2016.
- Jose Carmena (graduate student at INAOE, Mexico), EE Department, UCLA, 2017.
- Arturo Flores (graduate student at INAOE, Mexico), EE Department, UCLA, 2014.
- Wilson Barbosa (graduate student at INAOE, Mexico), EE Department, UCLA, 2013.
- Shancheng Zhao (graduate student at Sun-yat Sen University, China), EE Department, UCLA, 2013-2014.

Past Undergraduate Students

- Shyam Venkatasubramanian, ECE Department, UCLA, 2019-2021. Next: Duke University graduate program.
- Andrew Tan, EE Department, UCLA, 2017-2018.
- Jesse Wang, EE Department, UCLA, 2017-2018.
- Nian Guo, EE Department, UCLA, 2016-2017. Next: Caltech graduate program.
- Chi-Yo Tsai, EE Department, UCLA, 2015-2016.
- Manie Tadayon, EE Department, UCLA, 2014-2015. Next: UCLA graduate program.

- Jimmy Zhu, EE Department, UCLA, 2012-2013. Next: UCLA graduate program.
- Aaditya Ramesh, EE Department, UCLA, 2011-2012. Next: Cornell University graduate program.
- Jiajun (Dennis) Zhang, EE Department, UCLA, 2011-2012. Next: Northwestern University graduate program. Now: Associate Professor, Olin Business School, Washington University, Saint Louis, MO.
- Yun-Feng (Dennis) Lo, EE Department, UCLA, 2011.
- Babak Khouienia, EE Department, UCLA, 2011. Next: UCLA graduate program.
- Pamela Lee, EE Department, UC Berkeley, 2008. Next: UC Berkeley graduate program.

OUTREACH ACTIVITY

Los Angeles Computing Circle (LACC), UCLA, 2011 – present.

- Co-founder (with Profs. P. Gupta and M. Srivastava, UCLA) of LACC, a brand-new outreach program aimed at promoting careers in electrical and computer engineering among talented and underrepresented high-schoolers.

PUBLICATIONS

Monographs

1. L. Dolecek and F. Sala, “Channel Coding Methods for Non-Volatile Memories,” *Foundations and Trends in Communications and Information Theory*, vol. 13 (1), pp. 1- 136, Feb. 2016.

Book Chapters

1. I. Alam, L. Dolecek, and P. Gupta, “Lightweight Software-Defined Error Correction for Memories,” in *Dependable Embedded Systems*, Eds. J. Henkel and N. Dutt, Springer, 2021.
2. F. Sala, C. Schoeny, and L. Dolecek, “Advanced Algebraic and Graph-Based ECC Schemes for Flash Memories,” *3-D Flash Memories*, Ed. Rino Micheloni, Springer, Aug. 2016.
3. L. Dolecek, “Towards practical graph-based, iteratively decoded channel codes: insights through absorbing sets,” in *Tractability*, eds. L. Bordeaux, Y. Hamadi, and P. Kohli, Cambridge University Press, Mar. 2014.

Guest Editorials

1. A. Barg, L. Dolecek, R. Gabrys, Gyula O. H. Katona, J. Körner, A. McGregor, O. Milenkovic, S. Mesnager, G. Zémor, “Guest Editorial Special Issue: From Deletion-Correction to Graph Reconstruction: In Memory of Vladimir I. Levenshtein”, *IEEE Transactions on Information Theory Theory* vol. 67(6), pp. 3187-3189, June 2021.
2. L. Dolecek, M. Blaum, S. Bruck, A. Jiang, K. Ramchadran, and B. Vasic, “Guest Editorial: Communication Methodologies for the Next Generation Data Storage Systems,” *IEEE Journal on Selected Areas in Communications*, vol. 32 (5), pp. 825 – 830, May 2014.

Refereed Journal Publications

1. S. Yang, A. Hareedy, A. R. Calderbank, and L. Dolecek, “Hierarchical Coding for Cloud Storage: Topology-Adaptivity, Scalability, and Flexibility,” *IEEE Transactions on Communications*, vol. 68 (6), pp. 3657 – 3680, June 2022.
2. A. Hareedy, R. Wu, and L. Dolecek, “A Channel-Aware Combinatorial Approach to Design High Performance Spatially-Coupled Codes for Magnetic Recording Systems,” *IEEE Transactions on Information Theory*, vol. 66 (8), pp. 4834 – 4852, Aug. 2020.
3. P. Stanley-Marbell, A. Alaghi, M. Carbin, E. Darulova, L. Dolecek, A. Gerstlauer, G. Gillani, D. Jevdjic, T. Moreau, M. Cacciotti, A. Daglis, N. Enright Jerger, B. Falsafi, S. Misailovic, A. Sampson, D. Zufferey, “Exploiting Errors for Efficiency: A Survey from Circuits to Applications,” *ACM Computing Surveys*, vol. 53 (3), no. 3, pp. 1– 39, March 2020.

4. H. Esfahanizadeh, L. Tauz, and L. Dolecek, "Multi-Dimensional Spatially-Coupled Code Design: Enhancing the Cycle Properties," *IEEE Transactions on Communications*, vol. 68 (5), pp. 2653 – 2666, May 2020.
5. Z. Chen, C. Schoeny, L. Dolecek, "Pilot Assisted Adaptive Thresholding for Sneak-Path Mitigation in Resistive Memories With Failed Selection Devices," *IEEE Transactions on Communications*, vol. 68(1), pp. 66 – 81, Jan. 2020.
6. C. Schoeny, F. Sala, M. Gottscho, I. Alam, P. Gupta, and L. Dolecek, "Context-Aware Resiliency: Unequal Message Protection for Random-Access Memories," *IEEE Transactions on Information Theory*, vol. 65 (10), pp. 6146 – 6159, Oct. 2019.
7. S. Yang, C. Schoeny, and L. Dolecek, "Theoretical Bounds and Constructions of Codes in the Generalized Cayley Metric," *IEEE Transactions on Information Theory*, vol. 65 (8), pp. 4746 – 4763, Aug. 2019.
8. A. Hareedy, C. Lanka, N. Guo, and L. Dolecek, "A Combinatorial Methodology for Optimizing Non-Binary Graph-Based Codes: Theoretical Analysis and Applications in Data Storage," *IEEE Transactions on Information Theory*, vol. 65 (4), pp. 2128 – 2154, Apr. 2019. **2018-2019 Best Student Paper Award IEEE Data Storage Society.**
9. H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Finite-Length Construction of High Performance Spatially-Coupled Codes via Optimized Partitioning and Lifting," *IEEE Transactions on Communications*, vol. 67(1), pp. 3-16, Jan. 2019.
10. Z. Chen, C. Schoeny, and L. Dolecek, "Hamming Distance Computation in Unreliable Resistive Memory," *IEEE Transactions on Communications*, vol. 66 (11), pp. 5013 – 5027, Nov. 2018.
11. S. S. Garani, L. Dolecek, J. Barry, F. Sala, and B. Vasic, "Signal Processing and Coding Techniques for 2-D Magnetic Recording: An Overview," *Proceedings of the IEEE*, vol. 106 (9), pp. 286-318, Feb. 2018.
12. M. Gottscho, I. Alam, C. Schoeny, L. Dolecek, P. Gupta, "Low-Cost Memory Fault Tolerance for IoT Devices," *ACM Transactions on Embedded Computing Systems*, vol. 16 (5), pp. 1-25, Nov. 2017. **Best Paper Award at the ACM/IEEE Int. Conference on Compilers, Architecture, and System Synthesis (CASES).**
13. L. Dolecek and Y. Cassuto, "Channel Coding for Nonvolatile Memory Technologies: Theoretical Advances and Practical Considerations," *Proceedings of the IEEE*, vol. 105 (9), pp. 1705-1724, Sept. 2017.
14. F. Sala, R. Gabrys, C. Schoeny, and L. Dolecek, "Exact Reconstruction from Insertions in Synchronization Codes," *IEEE Transactions on Information Theory*, vol. 63 (4), pp. 2428 – 2445, Apr. 2017.
15. F. Sala, C. Schoeny, S. Kabir, D. Divsalar, and L. Dolecek, "On Nonuniform Noisy Decoding for LDPC Codes with Application to Radiation-Induced Errors," *IEEE Transactions on Communications*, vol. 65 (4), pp. 1438 – 1450, Apr. 2017.
16. H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Spatially-Coupled Codes Optimized for Magnetic Recording Applications," *IEEE Transactions on Magnetics*, vol. 53 (2), pp. 1-11, Feb. 2017.
17. A. Hareedy, C. Lanka, and L. Dolecek, "A General Non-Binary LDPC Code Optimization Framework Suitable for Dense Flash Memory and Magnetic Storage," *IEEE Journal On Selected Areas of Communications (JSAC) N Special Issue on Data Storage*, vol. 34 (9), pp. 2402 – 2415, Sep. 2016.
18. B. Amiri, A. Reisizadeh, H. Esfahanizadeh, J. Kliever, and L. Dolecek, "Optimized Design of Finite-Length Separable Circulant-Based Spatially-Coupled Codes: An Absorbing Set-Based Analysis," *IEEE Transactions on Communications*, vol. 64 (10) pp. 4029 – 4043, Oct. 2016.

19. A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Non-Binary LDPC Codes for Magnetic Recording Channels: Error Floor Analysis and Optimized Code Design," *IEEE Transactions on Communications*, vol. 64 (8), pp. 3194 – 3207, Aug. 2016.
20. F. Sala, C. Schoeny, N. Bitouze, and L. Dolecek, "Synchronizing Files from a Large Number of Insertions and Deletions," *IEEE Transactions on Communications*, vol. 64 (6), pp. 2258 – 2273, Jun. 2016.
21. R. Gabrys, E. Yaakobi, F. Farnoud, F. Sala, S. Bruck, and L. Dolecek, "Codes Correcting Erasures and Deletions for Rank Modulation," *IEEE Transactions on Information Theory*, vol. 62 (1), pp. 136 – 150, Jan. 2016.
22. C.-H. Huang, Y. Li, and L. Dolecek, "ACOCO: Adaptive Coding for Approximate Computing on Faulty Memories," *IEEE Transactions on Communications*, vol. 63 (12), pp. 4615 – 4628, Dec. 2015.
23. L. Wanner, L. Lai, A. Rahimi, M. Gottscho, P. Mercati, C.-H. Huang, F. Sala, Y. Agarwal, L. Dolecek, N. Dutt, P. Gupta, R. Gupta, R. Jhala, R. Kumar, S. Lerner, S. Mitra, A. Nicolau, T. Simunic-Rosing, M. B. Srivastava, S. Swanson, D. Sylvester, and Y. Zhou, "NSF Expedition on Variability-Aware Software: Recent Results and Contributions," *Information Technology*, vol. 57 (3), pp. 181 – 198, Jun. 2015.
24. Y. Li, Y. Chi, C.-H. Huang, and L. Dolecek, "Orthogonal Matching Pursuit on Noisy Circuits," *IEEE Transactions on Communications*, vol. 63 (4), pp. 1 – 14, Apr. 2015.
25. R. Gabrys, E. Yaakobi, and L. Dolecek, "Correcting Grain-Errors in Magnetic Media," *IEEE Transactions on Information Theory*, vol. 61 (5), pp. 2256 – 2272, May 2015. **2016 Best Student Paper Award IEEE Data Storage Society.**
26. R. Gabrys and L. Dolecek, "Constructions of Non-Binary WOM-Codes for Multilevel Flash Memories," *IEEE Transactions on Information Theory*, vol. 61 (4), pp. 1905 – 1919, Apr. 2015.
27. C.-H. Huang, Y. Li, and L. Dolecek, "Belief Propagation Algorithms on Noisy Hardware," *IEEE Transactions on Communications*, vol. 63 (1), pp. 11 – 24, Jan. 2015.
28. F. Sala, K. A. S. Immink, and L. Dolecek, "Error Control Schemes for Modern Flash Memories: Solutions for Flash Deficiencies," *IEEE Consumer Electronics*, vol. 4 (1), pp. 66 – 73, Jan. 2015.
29. R. Gabrys, F. Sala, and L. Dolecek, "Coding for Unreliable Flash Memory Cells," *IEEE Communication Letters*, vol. 18 (9), pp. 1491 – 1494, Jul. 2014.
30. L. Dolecek, D. Divsalar, Y. Sun and B. Amiri, "Non-Binary Protograph-Based LDPC Codes: Enumerators, Analysis, and Designs," *IEEE Transactions on Information Theory*, vol. 60 (7), pp. 3913 – 3941, Jul. 2014.
31. D. J. Costello, Jr., L. Dolecek, T. E. Fuja, J. Klierwer, D. G. M. Mitchell, and R. Smarandache, "Spatially Coupled Sparse Codes on Graphs - Theory and Practice," *IEEE Communications Magazine*, vol. 52 (7), pp. 168 – 176, Jun. 2014.
32. B. Amiri, J. Klierwer, and L. Dolecek, "Analysis and Enumeration of Absorbing Sets for Non-Binary Graph-Based Codes," *IEEE Transactions on Communications*, vol. 62 (2), pp. 398 – 409, Feb. 2014.
33. C.-H. Huang, Y. Li, and L. Dolecek, "Gallager B LDPC Decoder with Transient and Permanent Errors," *IEEE Transactions on Communications*, vol. 62 (1), pp. 15 – 28, Jan. 2014.
34. S. M. S. Tabatabaei and L. Dolecek, "A Deterministic, Polynomial-time Protocol for Synchronizing from Deletions," *IEEE Transactions on Information Theory*, vol. 60 (1), pp. 397 – 407, Jan. 2014.
35. M. Qazi, M. Tikekar, L. Dolecek, D. Shah, and A. Chandrakasan, "A Technique for Efficient Evaluation of SRAM Timing Failure," *IEEE Transactions on VLSI*, vol. 21 (8), pp. 1558 – 1562, Aug. 2013.

36. F. Sala, R. Gabrys and L. Dolecek, "Dynamic Threshold Schemes for Multi-Level Non-Volatile Memories," *IEEE Transactions on Communications*, vol. 61 (7), pp. 2624 – 2634, Jul. 2013.
37. S. Tabatabaei, H. Cho, S. and L. Dolecek, "Gallager B Decoder on Noisy Hardware," *IEEE Transactions on Communications*, vol. 61 (5), pp. 1660 – 1673, May 2013.
38. R. Gabrys, E. Yaakobi, and L. Dolecek, "Graded Bit-Error Correcting Codes with Applications to Flash Memory," *IEEE Transactions on Information Theory*, vol. 59 (4), pp. 2315 – 2327, Apr. 2013.
39. J. Wang, L. Dolecek, and R. Wesel, "The Cycle Consistency Matrix Approach to Absorbing Sets in Separable Circulant-Based LDPC Codes," *IEEE Transactions on Information Theory*, vol. 59 (4), pp. 2293 – 2314, Apr. 2013.
40. B. Amiri, C. W. Lin, and L. Dolecek, "Asymptotic Distribution of Absorbing Sets and Fully Absorbing Sets for Regular Sparse Code Ensembles," *IEEE Transactions on Communications*, vol. 61 (2), pp. 455 – 464, Feb. 2013.
41. P. Gupta, Y. Agarwal, L. Dolecek, N. Dutt, R. K. Gupta, R. Kumar, S. Mitra, A. Nicolau, T. Simunic Rosing, M. B. Srivastava, S. Swanson, and D. Sylvester, "Underdesigned and Opportunistic Computing in Presence of Hardware Variability," *IEEE Transactions on CAD*, vol. 32 (1), pp. 8 – 23, Jan. 2013.
42. S. M. S. Tabatabaei, S. Huang, and L. Dolecek, "Optimal Design of a Gallager B Noisy Decoder for Irregular LDPC Codes," *IEEE Communication Letters*, vol. 16 (12), pp. 2052 – 2055, Dec. 2012.
43. F. Penna, Y. Sun, L. Dolecek, and D. Cabric, "Detecting and Counteracting Statistical Attacks in Cooperative Spectrum Sensing," *IEEE Transactions on Signal Processing*, vol. 60 (4), pp. 1806 – 1822, Apr. 2012.
44. L. Dolecek and V. Anantharam, "Repetition Error Correcting Sets: Explicit Constructions and Prefixing Methods," *SIAM Journal on Discrete Mathematics*, vol. 23 (4), pp. 2120 – 2146, Jan. 2010.
45. L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright, and B. Nikolic, "Analysis of Absorbing Sets and Fully Absorbing Sets for Array-Based LDPC Codes," *IEEE Transactions on Information Theory*, vol. 56 (1), pp. 181 – 201, Jan. 2010.
46. Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam and M. Wainwright, "Design of LDPC Decoders for Low Bit Error Rate Performance: Quantization and Algorithm Choices," *IEEE Transactions on Communications*, vol. 57 (11), pp. 3258 – 3268, Nov. 2009.
47. L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic, and M. J. Wainwright, "Predicting Error Floors of LDPC Codes: Deterministic Bounds and Estimates," *IEEE Journal on Selected Areas of Communications*, vol. 27 (6) pp. 908 – 917, Aug. 2009.
48. L. Dolecek and V. Anantharam, "Using Reed Muller Codes in Channels With Synchronization and Substitution Errors," *IEEE Transactions on Information Theory*, vol. 53 (4), pp. 1430 – 1443, Apr. 2007.

Refereed and Invited Conference Publications

1. L. Tauz and L. Dolecek, "Variable Coded Batch Matrix Multiplication," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Madrid, Spain, Dec. 2021.
2. Z. Chen and L. Dolecek, "Coding Schemes for Crossbar Resistive Memory with High Line Resistance in SCM Applications," in Proc. *IEEE International Symposium on Topics in Coding*, Montreal, Canada, August 2021. (invited)
3. S. Yang, A. Hareedy, S. Venkatasubramanian, A. R. Calderbank, and L. Dolecek, "GRADE-AO: Towards Near-Optimal Spatially-Coupled Codes With High Memories," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Melbourne, Australia, July 2021.

4. D. Mitra, L. Tauz, and L. Dolecek, "Communication-Efficient LDPC Code Design for Data Availability Oracle in Side Blockchains," in Proc. *IEEE Information Theory Workshop (ITW)*, Kanazawa, Japan, October 2021.
5. A. Chen and L. Dolecek, "Write and Read Channel Models for 1S1R Crossbar Resistive Memory with High Line Resistance," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Taipei, Taiwan, Dec. 2020.
6. D. Mitra, L. Tauz, and L. Dolecek, "Concentrated Stopping Set Design for Coded Merkle Tree: Improving Security Against Data Availability Attacks in Blockchain Systems," in Proc. *IEEE Information Theory Workshop (ITW)*, Riva del Garda, Italy, Aug. 2020 (held in 2021).
7. G. Jovanovic and L. Dolecek, "Improving Aliasing Rejection by Inserting Additional Zeros into Folding Bands Using Simple Filters," 24th European Conference on Circuit Theory and Design, Sofia, Bulgaria, Sept. 2020.
8. S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Topology-Aware Cooperative Data Protection in Blockchain-Based Decentralized Storage Networks," Proc. *IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, Jun. 2020. **Runner Up for the Memorable Paper Award at NVMW 2021.**
9. H. Esfahanizadeh, E. Ram, Y. Cassuto, and L. Dolecek, "Spatially Coupled Codes with Sub-Block Locality: Joint Finite Length-Asymptotic Design Approach," Proc. *IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, Jun. 2020.
10. L. Tauz, H. Esfahanizadeh, and L. Dolecek, "Non-Uniform Window Decoding for Multi-Dimensional Spatially-Coupled LDPC Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, Jun. 2020. **Winner of the Memorable Paper Award at NVMW 2021.**
11. S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek, "Hierarchical coding to enable scalability and flexibility in heterogeneous cloud storage," *IEEE GLOBECOM*, Waikoloa, HI, USA, Dec. 2019.
12. S. Yang, M. Can Sarihan, K.-C. Chang, C.W. Wong, and L. Dolecek, "Efficient Information Reconciliation for Energy-Time Entanglement Quantum Key Distribution," *Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2019.
13. D. Mitra and L. Dolecek, "Patterned Erasure Correcting Codes for Low Storage-Overhead Blockchain Systems," *Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2019.
14. L. Tauz, L. Dolecek, "Multi-Message Gradient Coding for Utilizing Non-Persistent Stragglers," *Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2019.
15. H. Esfahanizadeh, R. Wu, and L. Dolecek, "A Finite-Length Construction of Irregular Spatially-Coupled Codes," in Proc. *IEEE Information Theory Workshop (ITW)*, Visby, Sweden, Aug. 2019. (invited)
16. M. Abroshan, R. Venkataramanan, L. Dolecek, and A. Guillen i Fabregas, "Coding for Deletion Channels with Multiple Traces," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Paris, France, Jul. 2019.
17. H. Esfahanizadeh, A. Hareedy, L. Dolecek, "Multi-Dimensional Spatially-Coupled Code Design Through Informed Relocation of Circulants," in Proc. *57th Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2019. **Runner up for Memorable Paper Award at NVMW 2019.**
18. L. Conde-Canencia, T. Condie, and L. Dolecek, "Data Deduplication with Edit Errors," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, Dec. 2018.
19. A. Hareedy, H. Esfahanizadeh, A. Tan, and L. Dolecek, "Spatially-Coupled Code Design for Partial-Response Channels: Optimal Object-Minimization Approach," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, Dec. 2018.

20. S. Jiang, C. Schoeny, and L. Dolecek, "An Interactive Algorithm for Synchronizing from Burst Deletions," in Proc. *IEEE International Symposium on Turbo Coding and Iterative Information Processing*, Hong Kong, Hong Kong, Dec. 2018.
21. Z. Chen, C. Schoeny, and L. Dolecek, "Coding Assisted Adaptive Thresholding for Sneak-Path Mitigation in Resistive Memories," in Proc. *IEEE Information Theory Workshop (ITW)*, Guangzhou, China, Nov. 2018. **Runner up for Memorable Paper Award at NVMW 2019.**
22. C. Schoeny, I. Alam, M. Gottscho, P. Gupta, and L. Dolecek, "Error Correction and Detection for Computing Memories Using System Side Information," in Proc. *IEEE Information Theory Workshop (ITW)*, Guangzhou, China, Nov. 2018. (invited)
23. L. Conde-Canencia and L. Dolecek, "Nanopore DNA Sequencing Channel Modeling," in Proc. *IEEE Workshop on Signal Processing Systems (SiPS)*, Cape Town, South Africa, Oct. 2018.
24. L. Dolecek, J. Lee, and J. Cheung, "Unveiling Intrinsic Locality Properties of Reed-Solomon Codes with Applications to Distributed Storage," in Proc. *IEEE International Conference on Communications (ICC)*, Kansas City, MS May 2018.
25. I. Alam, C. Schoeny, L. Dolecek, and P. Gupta, "Parity++: Lightweight Error Correction for Last Level Caches," Workshop on Silicon Errors in Logic System Effects (SELSE), March 2018. **Best of SELSE & also presented at DSN 2018.**
26. Z. Chen, C. Schoney, Y. Cassuto, and L. Dolecek, "Coding Scheme for Reliable In-Memory Hamming Distance Computation," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Asilomar, CA, Nov. 2017.
27. A. Hareedy, H. Esfahanizadeh, and L. Dolecek, "High performance spatially coupled codes for flash memories," in Proc. *IEEE Information Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017. **Winner of the Memorable Paper Award at NVMW 2018.**
28. S. Yang, C. Schoeny, and L. Dolecek, "Order-optimal permutation codes in the generalized Cayley metric," in Proc. *IEEE Information Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017. **Runner up for Memorable Paper Award at NVMW 2018.**
29. F. Sala, C. Schoeny, and L. Dolecek, "Faster Reconstruction Through Coding for DNA Storage," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Asilomar, CA, Nov. 2017. (invited)
30. C. Schoeny, F. Sala, M. Gottcho, I. Aram, P. Gupta, and L. Dolecek, "Context-Aware Resiliency: Unequal Message Protection for Memory Applications," in Proc. *IEEE Information Theory Workshop (ITW)*, Kaohsiung, Taiwan, Nov. 2017. (invited)
31. S. Kabir, F. Sala, G. Van Den Broeck, "Coded Machine Learning: joint informed replication and learning for linear regression," in Proc. *55th Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2017.
32. M. Gottscho, I. Alam, C. Schoeny, L. Dolecek and P. Gupta, "Low-Cost Memory Fault Tolerance for IoT Devices," in Proc. *Embedded Systems Week – CASES*, Seoul, Korea, Oct. 2017. **Best paper award at CASES.**
33. H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "A Novel Combinatorial Framework to Construct Spatially-Coupled Codes: Minimum Overlap Partitioning," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Aachen, Germany, June, 2017.
34. F. Sala, C. Schoeny, S. Kabir, D. Divsalar, and L. Dolecek, "Flash Memories in High Radiation Environments: LDPC Decoder Study," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Nov. 2016. (invited)
35. H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Spatially Coupled LDPC Codes Optimized for 1-D Magnetic Recording Channels," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Nov. 2016.

36. K. Mazooji, F. Sala, G. van Den Broeck, and L. Dolecek, "Robust Channel Coding Strategies for Machine Learning Data," in Proc. *54th Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2016. (invited)
37. L. Dolecek, "Coding Theory for Robust Computing: Models, Tools, Applications," in Proc. *IEEE International Symposium on Turbo Coding and Iterative Information Processing*, Brest, France, Sep. 2016 (invited).
38. A. Reisizadeh, C. Schoeny, C.Y. Tsai, and L. Dolecek, "Approximate File Synchronization: Upper Bounds and Interactive Algorithms," in Proc. *IEEE Information Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016.
39. L. Dolecek, H. Esfahanizadeh, A. Hareedy, "Spatially-Coupled Codes for Advanced Magnetic Recording," in Proc. *The 27th IEEE Magnetic Recording Conference (TMRC)*, Stanford Univ., Aug. 2016. (invited)
40. A. Hareedy, C. Lanka, C. Schoeny, and L. Dolecek, "The Weight Consistency Matrix Framework for General Non-Binary LDPC Code Optimization: Applications in Flash Memories," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, Jul. 2016.
41. F. Sala, R. Gabrys, C. Schoeny, K. Mazooji, and L. Dolecek, "Exact Sequence Reconstruction for Insertion-Correcting Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, Jul. 2016.
42. M. Gottscho, C. Schoeny, L. Dolecek and P. Gupta, "Software Defined Error-Correcting Codes," *Workshop on Silicon Errors in Logic System Effects (SELSE)*, Mar. 2016. **Best of SELSE & presented by invitation at DSN 2016.**
43. F. Sala, H. Duwe, L. Dolecek, and R. Kumar, "A Unified Framework for Error Correction Techniques in On-Chip Memories," *Workshop on Silicon Errors in Logic System Effects (SELSE)*, Mar. 2016, **Best of SELSE & presented by invitation at DSN 2016.**
44. P. Schlafer, C.H. Huang, C. Schoeny, C. Weiss, Y. Li, N. When, and L. Dolecek, "Error Resilience and Energy Efficiency: An LDPC Decoder Design Study," in Proc. *DATE Conference*, Dresden, Germany, Mar. 2016.
45. A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Non-binary LDPC Code Optimization for Partial Response Channels," in Proc. *IEEE Globecom*, San Diego, CA, Dec. 2015. **Best Paper Award.**
46. F. Sala, C. Schoney, D. Divsalar, and L. Dolecek, "Asymmetric ECC for Flash in High-Radiation Environments," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA Nov. 2015. (invited)
47. C. Schoeny, F. Sala, and L. Dolecek, "Analysis and Coding Schemes for the Flash Normal-Laplace Mixture Channel," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, Jun. 2015.
48. F. Sala, C. Schoeny, and L. Dolecek, "Three Novel Combinatorial Theorems for the Insertion/Deletion Channel," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, Jun. 2015.
49. F. Sala, C. Schoeny, D. Divsalar, and L. Dolecek, "Asymmetric Error-Correcting Codes for Flash Memories in High-Radiation Environments," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, Jun. 2015.
50. C.H. Huang, Y. Li, and L. Dolecek, "Adaptive Error Correction Coding Scheme for Computations in the Noisy Min-Sum Decoder," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, Jun. 2015.
51. B. Amiri, A. Reisizadeh, J. Kliewer, and L. Dolecek, "Optimized Array-Based Spatially-Coupled LDPC Codes: An Absorbing Set Approach," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, Jun. 2015.

52. F. Sala, R. Gabrys, and L. Dolecek, "Gilbert-Varshamov-like Lower Bounds for Deletion-Correcting Codes," in Proc. *IEEE Information Theory Workshop (ITW)*, Hobart, Australia, Nov. 2014.
53. B. Amiri, A. Flores, and L. Dolecek, "Design of Non-binary Quasi-Cyclic LDPC Codes by Absorbing Set Removal," in Proc. *IEEE Information Theory Workshop (ITW)*, Hobart, Australia, Nov. 2014.
54. Y. Toriyama, B. Amiri, L. Dolecek, and D. Markovic, "Logarithmic Quantization Scheme for Reduced Hardware Cost and Improved Error Floor in Non-Binary LDPC Codes," in Proc. *IEEE Globecom*, Austin, TX, Dec. 2014.
55. C. Schoeny, N. Bitouze, F. Sala, and L. Dolecek, "Efficient File Synchronization: Extensions and Simulations," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA Nov. 2014. (invited)
56. Y. Toriyama, B. Amiri, L. Dolecek, and D. Markovic, "Field-Order Based Hardware Cost Analysis of Non-Binary LDPC Decoders," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2014.
57. C.-H. Huang, Y. Li, and L. Dolecek, "Noisy Belief Propagation," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2014.
58. F. Sala, R. Gabrys, and L. Dolecek, "Deletions in Multipermutations," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, Jul. 2014.
59. D. Mitchell, L. Dolecek, and D. Costello, "Absorbing Set Characterization of Array-Based Spatially Coupled LDPC Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, Jul. 2014.
60. R. Gabrys, E. Yaakobi, F. Farnoud, F. Sala, S. Bruck, and L. Dolecek, "Single-Deletion-Correcting Codes over Permutations," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, Jul. 2014.
61. N. Bitouze, F. Sala, S. M. S. Tabatabaei, and L. Dolecek, "A practical framework for efficient file synchronization," in Proc. *IEEE 51st Allerton Conference on Communication, Control, and Computing*, Moticello, IL, Oct. 2013. (invited)
62. Y. Li and L. Dolecek, "Effects of Approximate Representation in Belief Propagation for Inference in Wireless Sensor Networks," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2013.
63. X. Wu, H. Chang, C.-H. Huang, Y. Wang, L. Dolecek, and G. Pottie, "Virtual Inertial Measurements for Motion Inference in Wireless Health," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2013.
64. M. Laghate, C.H. Huang, C.-K. Yu, L. Dolecek, and D. Cabric, "Identifying Statistical Mimicry Attacks in Distributed Spectrum Sensing," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2013.
65. F. Sala and L. Dolecek, "Constrained Rank Modulation," in Proc. *IEEE Information Theory Workshop (ITW)*, Sevilla, Spain, Sep. 2013.
66. S. Chakraborty, N. Bitouze, M. Srivastava, and L. Dolecek, "Protecting Data Against Unwanted Inferences," in Proc. *IEEE Information Theory Workshop (ITW)*, Sevilla, Spain, Sep. 2013.
67. N. Bitouze and L. Dolecek, "Synchronization from Insertions and Deletions under a Non-binary, Non-uniform Source," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, Jul. 2013.
68. F. Sala and L. Dolecek, "Counting Sequences Obtained from the Synchronization Channel," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, Jul. 2013.

69. R. Gabrys, E. Yaakobi, and L. Dolecek, "Correcting Grain Errors in Magnetic Media," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, Jul. 2013.
70. B. Amiri, J. Kliewer, and L. Dolecek, "Analysis and Enumeration of Absorbing Sets for Non-Binary Graph-Based Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, Jul. 2013.
71. C. H. Huang, Y. Li, and L. Dolecek, "Gallager B LDPC Decoder with Transient and Permanent Errors," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, Jul. 2013.
72. C. H. Huang and L. Dolecek, "Analysis of Finite-Alphabet Iterative Decoders Under Processing Errors," in Proc. *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May, 2013.
73. F. Sala, R. Gabrys, and L. Dolecek, "Dynamic Threshold Schemes for Multi-Level Nonvolatile Memories," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2012.
74. B. Amiri, S. Garani Srinivasa, and L. Dolecek, "Quantization, Absorbing Regions and Practical Message Passing Decoders," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2012.
75. B. Y. Chang, D. Divsalar, and L. Dolecek, "Non-binary Protograph-Based LDPC Codes for Short Block-Lengths," in Proc. *IEEE Information Theory Workshop (ITW)*, Lausanne, Switzerland, Sep. 2012.
76. S. Tabatabaei Yazdi and L. Dolecek, "Synchronization from Deletions Through Interactive Communication," in Proc. *IEEE International Symposium on Turbo Coding and Iterative Information Processing (ISTC)*, Gothenburg, Sweden, Aug. 2012.
77. R. Gabrys, E. Yaakobi, L. Grupp, S. Swanson, and L. Dolecek, "Tackling Intracell Variability in TLC Flash Through Tensor Product Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Boston, MA, Jul. 2012.
78. D. Divsalar and L. Dolecek, "Graph Cover Ensembles of Non-binary Protograph LDPC Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Boston, MA, Jul. 2012.
79. S. M. S. Tabatabaei Yazdi, H. Cho, Y. Sun, S. Mitra and L. Dolecek, "Probabilistic Analysis of Gallager B Faulty Decoder," in Proc. *IEEE International Conference on Communications (ICC)*, Ottawa, Canada, Jun. 2012.
80. F. Gong, S. Basir-Kazeruni, L. Dolecek and L. He, "Fast Estimation of SRAM Failure Rate Using Probability Collectives," in Proc. *ACM International Symposium on Physical Design (ISPD)*, Napa, CA, Mar. 2012.
81. R. Gabrys and L. Dolecek, "Coding for the Binary Asymmetric Channel," in Proc. *IEEE International Conference on Computing, Networking and Communications (ICNC)*, Maui, HI, Jan. 2012.
82. D. Divsalar and L. Dolecek, "On the Typical Minimum Distance of Protograph-Based Non-Binary LDPC Codes," in Proc. *IEEE Information Theory and Applications Workshop (ITA)*, La Jolla, CA, Feb. 2012. (invited)
83. R. Gabrys and L. Dolecek, "Spatially-Aware Adaptive Error Correcting Codes for Flash Memory," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2011.
84. J. Zhang, J. Wang, S. G. Srinivasa, and L. Dolecek, "Achieving Flexibility in LDPC Code Design by Absorbing Set Elimination," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2011.

85. Y. Sun and L. Dolecek, "Complexity Analysis of Interior Point Methods for LP Decoding," in Proc. *IEEE Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, Nov. 2011.
86. R. Gabrys, E. Yaakobi, L. Dolecek, P. Siegel, A. Vardy, and J. K. Wolf, "Non-binary WOM Codes for Multilevel Flash Memories," in Proc. *IEEE Information Theory Workshop (ITW)*, Paraty, Brazil, Oct. 2011.
87. D. Divsalar and L. Dolecek, "Ensemble Analysis of Pseudocodewords of Non-binary Protograph-based LDPC codes," in Proc. *IEEE Information Theory Workshop (ITW)*, Paraty, Brazil, Oct. 2011.
88. B. Y. Chang, L. Dolecek, and D. Divsalar, "EXIT Chart Analysis and Design of Non-binary Protograph Based LDPC Codes," in Proc. *IEEE Military Communications Conference (Milcom)*, Baltimore, MD, Nov. 2011.
89. F. Penna, Y. Sun, L. Dolecek, and D. Cabric, "Joint Spectrum Sensing and Detection of Malicious Nodes via Belief Propagation," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Houston, TX, Dec. 2011.
90. R. Gabrys and L. Dolecek, "Characterizing Capacity Achieving Write Once Memory Codes for Multilevel Flash Memories," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, St. Petersburg, Russia, Jul. - Aug. 2011.
91. D. Divsalar and L. Dolecek, "Enumerators for Protograph-Based Ensembles of Nonbinary LDPC Codes," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, St. Petersburg, Russia, Jul. - Aug. 2011.
92. J. Wang, D. Dolecek, Z. Zhang, and R. Wesel, "Absorbing Set Spectrum Approach for Practical Code Design," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, St. Petersburg, Russia, Jul. - Aug. 2011.
93. J. Wang, L. Dolecek, and R. Wesel, "Controlling LDPC Absorbing Sets via the Null Space of the Cycle Consistency Matrix," in Proc. *IEEE International Conference on Communications (ICC)*, Kyoto, Japan, Jun. 2011.
94. B. Y. Chang, M. Ivkovic, and L. Dolecek, "Computationally-Efficient Iterative Decoding for Reliable System Design: Min-Sum Refined," in Proc. *IEEE International Symposium on Circuits and Systems (ISCAS)*, Rio de Janeiro, Brazil, May 2011.
95. J. Wang, L. Dolecek, and R. Wesel, "LDPC Absorbing Sets, the Null Space of the Cycle Consistency Matrix, and Tanner's Constructions," in Proc. *IEEE Information Theory and Applications Workshop (ITA)*, La Jolla, CA, Feb. 2011. (invited)
96. C. W. Lin and L. Dolecek, "Asymptotic Distribution of Absorbing Sets and Fully Absorbing Sets for Regular Sparse Code Ensembles," in Proc. *IEEE Australian Communications Theory Workshop (AusCTW)*, Melbourne, Australia, Feb. 2011.
97. L. Dolecek, "Towards Longer Lifetime of Emerging Memory Technologies Using Number Theory," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Miami, FL, Dec. 2010.
98. L. Dolecek, Z. Wang, and Z. Zhang, "Towards Improved LDPC Code Designs Using Absorbing Set Spectrum Properties," in Proc. *IEEE International Symposium on Turbo Coding and Iterative Information Processing (ISTC)*, Brest, France, Sep. 2010.
99. L. Dolecek, "On Absorbing Sets of Structured Sparse Graph Codes," in Proc. *IEEE Information Theory and Applications Workshop (ITA)*, La Jolla, CA, Jan. - Feb. 2010. (invited)
100. G. Jovanovic and L. Dolecek, "Novel Multiplierless Wide-band CIC Compensator," in Proc. *IEEE International Symposium on Circuits and Systems (ISCAS)*, Paris, France, May - Jun. 2010.

101. M. Qazi, M. Tikekar, L. Dolecek, D. Shah, and A.P. Chandrakasan, "Loop Flattening & Spherical Sampling: Highly Efficient Model Reduction Techniques for SRAM Yield Analysis" in Proc. *Design, Automation, and Test in Europe Conference (DATE)*, Dresden, Germany, Mar. 2010.
102. Z. Zhang, L. Dolecek, P. Lee, V. Anantharam, M. J. Wainwright, B. Richards, and B. Nikolic, "Low error rate LDPC decoders," in Proc. *IEEE Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2009.
103. L. Dolecek and D. Shah, "Influence in a Large Society: Interplay Between Information Dynamics and Network Structure," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Seoul, Korea, Jun.-Jul. 2009.
104. Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. J. Wainwright, "Lowering LDPC Error Floors by Postprocessing," in Proc. *IEEE Global Telecommunications Conference (GLOBECOM)*, New Orleans, LA, Nov. 2008.
105. L. Dolecek, M. Qazi, D. Shah and A. Chandrakasan, "Breaking the Simulation Barrier: SRAM Evaluation through Norm Minimization," in Proc. *IEEE International Conference on Computer-Aided Design (ICCAD)*, San Jose, CA, Nov. 2008.
106. L. Dolecek and D. Shah, "Social Networks With(out) Influential Agents," in Proc. *Third Congress of the Game Theory Society (GAMES 2008)*, Evanston, IL, Jul. 2008.
107. P. Lee, L. Dolecek, Z. Zhang, V. Anantharam, B. Nikolic, and M. Wainwright, "Error Floors in LDPC Codes: Fast Simulation, Bounds and Hardware Emulation," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Toronto, Canada, Jul. 2008.
108. L. Dolecek and V. Anantharam, "Prefixing Method for Correcting Repetition Errors," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Toronto, Canada, Jul. 2008.
109. L. Dolecek, Z. Zhang, M. J. Wainwright, V. Anantharam, and B. Nikolic, "Evaluation of the Low Frame Error Rate Performance of LDPC Codes Using Importance Sampling," in Proc. *IEEE Information Theory Workshop (ITW)*, Lake Tahoe CA, Sep. 2007.
110. L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright, and B. Nikolic, "Analysis of Absorbing Sets for Array-Based LDPC Codes," in Proc. *IEEE International Conference on Communications (ICC)*, Glasgow, United Kingdom, Jun. 2007.
111. Z. Zhang, L. Dolecek, V. Anantharam, M. Wainwright, and B. Nikolic, "Quantization Effects in Low-Density Parity-Check Decoders," in Proc. *IEEE International Conference on Communications (ICC)*, Glasgow, United Kingdom, Jun. 2007.
112. L. Dolecek and V. Anantharam, "On Subsets of Binary Strings Immune to Multiple Repetitions," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Nice, France, Jun. 2007, pp. 1691-1695.
113. L. Dolecek and V. Anantharam, "On Communication over Channels with Varying Sampling Rate," in Proc. *Information Theory and Applications Workshop (ITA)*, UC San Diego, La Jolla, Jan.-Feb. 2007.
114. Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright, "Investigation of error floors of a structured low-density parity-check code by hardware simulation," in Proc. *IEEE Global Telecommunications Conference (GLOBECOM)*, San Francisco, Nov. – Dec. 2006. Best paper award finalist.
115. L. Dolecek and V. Anantharam, "A Synchronization Technique for Array-Based LDPC Codes in Channels with Varying Sampling Rate," in Proc. *IEEE International Symposium on Information Theory (ISIT)*, Seattle, WA, Jul. 2006.
116. L. Dolecek and V. Anantharam, "Run-length Properties of a Reed-Muller $RM(1,m)$ Code with Applications in Channels With at Most One Synchronization Error," in Proc. *42nd Allerton Conference on Communications, Control, and Computing*, Champaign, IL, Sept. – Oct. 2004.

Selected Refereed Workshop Articles (without proceedings)

1. D. Mitra, L. Tauz, and L. Dolecek, "Concentrated Stopping Set Design for Coded Merkle Tree: Improving Security Against Data Availability Attacks in Blockchain Systems," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, May 2022.
2. S. Yang, A. Hareedy, A. R. Calderbank, and L. Dolecek, "Cooperative Data Protection for Topology-Aware Decentralized Storage Networks," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2021.
3. L. Tauz, H. Esfahanizadeh, and L. Dolecek, "Non-Uniform Windowed Decoding For Multi-Dimensional Spatially-Coupled LDPC Codes," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2021.
4. Z. Chen and L. Dolecek, "Coding Assisted Adaptive Thresholding for Sneak-Path Mitigation in Resistive Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2019.
5. H. Esfahanizadeh, A. Hareedy, and L. Dolecek, "Multi-Dimensional Spatially-Coupled Code Design with Improved Cycle Properties," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2019.
6. S. Yang, C. Schoeny, L. Conde - Canencia, and L. Dolecek, "Multi-level Access and Information Leakage in Scalable Cloud Storage," *Multi-level Access and Information Leakage in Scalable Cloud Storage*, *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2019.
7. A. Hareedy, H. Esfahanizadeh, and L. Dolecek, "A Three-Stage Approach for Designing Non-Binary Spatially-Coupled Codes for Flash Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2018.
8. L. Tauz and L. Dolecek, "Multi-Message Gradient Coding for Utilization of Non-Persistent Stragglers," *CodML Workshop, International Conference on Machine Learning*, Long Beach, CA, Jun. 2019.
9. F. Sala, S. Kabir, G. van Den Broeck, and L. Dolecek, "Don't Fear the Bit Flips: Optimized Coding Strategies for Binary Classification", *Workshop on Robust Machine Learning in the Wild*, Melbourne, Australia, Aug. 2017.
10. F. Sala, C. Schoeny, S. Kabir, D. Divsalar, L. Dolecek, "Modeling the effects of radiation induced soft errors on LDPC decoding," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2017.
11. A. Hareedy, C. Lanka, and L. Dolecek, "Non-Binary LDPC Code Optimization for Modern Storage Systems," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2017.
12. F. Sala, C. Schoeny, and L. Dolecek, "Approximate and Noisy Computing: Connections to the Information-Theory World," *The 2015 Workshop on Approximate Computing Across the Stack*, Atlanta, GA, Apr. 2016.
13. R. Gabrys, E. Yaakobi, F. Sala, F. Farnoud, S. Bruck, and L. Dolecek, "Codes Correcting Erasures and Deletions for Rank Modulation," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2016.
14. F. Sala, C. Schoeny, D. Divsalar, and L. Dolecek, "Error-Correcting Codes for Radiation-Induced Error Patterns in Flash Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2016.
15. C.H. Huang, Y. Li, and L. Dolecek, "ACOCO: Adaptive Coding for Approximate Computing on Faulty Memories," *The 2015 Workshop on Approximate Computing Across the Stack*, Portland, OR, Jun. 2015.

16. C. Schoeny, B. Amiri, A. Hareedy, and L. Dolecek, "Quasi-Cyclic Non-Binary LDPC Codes for MLC NAND Flash Memory," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2015.
17. R. Gabrys and L. Dolecek, "Coding for Unreliable Memory Cells in TLC Flash," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2014.
18. F. Sala, R. Gabrys, and L. Dolecek, "Constrained Rank Modulation for Flash Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2014.
19. F. Sala, R. Gabrys, and L. Dolecek, "Dynamic Threshold Schemes for Multi-Level Nonvolatile Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2013.
20. R. Gabrys, E. Yaakobi, L. Grupp, S. Swanson, and L. Dolecek, "Tackling Intracell Variability in TLC Flash Through Tensor Product Codes," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2012.
21. R. Gabrys and L. Dolecek, "Coding Schemes and Information Theoretic Bounds for Generalized Write Once Memories," *Non-Volatile Memories Workshop (NVMW)*, UC San Diego, La Jolla, CA, Mar. 2012.