UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

PHISON ELECTRONICS CORPORATION,

Petitioner

v.

VERVAIN, LLC,

Patent Owner

U.S. Patent No. 8,891,298

Inter Partes Review Case No. IPR2025-00212

DECLARATION OF CARL SECHEN, PH.D.

TABLE OF CONTENTS

I. OVERVIEW: 1

II. BACKGROUND AND QUALIFICATIONS: 4

III. MY UNDERSTANDING OF THE APPLICABLE LAW 7

IV. DOCUMENTS CONSIDERED IN FORMULATING MY OPINIONS: 11

V. RELEVANT STATE OF THE ART AS OF JULY 19, 2011 15

A. NAND FLASH MEMORY AND OPERATIONS 18

B. MLC AND SLC 41

C. HYBRID USE OF MLC AND SLC MEMORY 52

D. USE OF MLC MEMORY IN "SLC MODE" ("PSEUDO-SLC") 57

E. "DATA INTEGRITY CHECKING" IN NAND FLASH MEMORY SYSTEMS 61

F. STRATEGIES FOR NAND FLASH MEMORY REMAPPING TO MANAGE SYSTEM RELIABILITY AND EFFICIENCY 73

VI. PERSON OF ORDINARY SKILL IN THE ART (POSITA) 76

VII. DISCLOSURE OF THE '298 PATENT SPECIFICATION 77

A. '298 SPECIFICATION BACKGROUND OF THE DISCLOSURE 78

B. PROFFERED EMBODIMENTS IN THE DETAILED DESCRIPTION OF THE ILLUSTRATED EMBODIMENT 84

VIII. THE '298 PATENT CLAIMS 102

A. THE ORIGINAL CLAIMS OF THE APRIL 25, 2012, APPLICATION 102

B. EXAMINATION OF THE APPLICATION: "TRANSFER" 104

C. THE ISSUED CLAIMS 107

D. ISSUED CLAIM DEPARTURES FROM THE DETAILED DESCRIPTION 109

E. CLAIM CONSTRUCTION.....112

IX. PATENTABILITY OF THE '298 CLAIMS RELATIVE TO THE ART 117

X. GROUND 1: THE CHALLENGED CLAIMS ARE OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 118

A. CLAIM 1 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA. 118

1. [1Pre] A system for storing data comprising: 120
2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;.....121
3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;.....121
4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to; 122
5. [1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;.....123
6. [1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;. 125
7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently

by maintaining a count of the number of times each one of the blocks is accessed; and 127

8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module. 129

B. CLAIM 2 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA. 134

1. [2Pre] The system of claim 1, ...134

2. [2] wherein the minimum quanta of address is equal to one block. 134

C. CLAIM 3 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA. 134

1. [3Pre] The system of claim 1, ...134

2. [3] wherein the minimum quanta of address is equal to one page. 135

D. CLAIM 4 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA. 135

1. [4Pre] The system of claim 1, ...135

2. [4] wherein the MLC non-volatile memory module is NAND flash memory….....135

E. CLAIM 5 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA. 136

1. [5Pre] The system of claim 1, ...136

2. [5] wherein the SLC non-volatile memory module is NAND flash memory….....136

F. CLAIM 6 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA.....137

1. [6Pre] The system of claim 1, ...137

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM). 137

G. CLAIM 7 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 138

1. [7Pre] The system of claim 1, ... 138

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM). 138

H. CLAIM 8 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 138

1. [8Pre] The system of claim 1, ... 138

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM). 138

I. CLAIM 9 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 139

1. [9Pre] The system of claim 1, ... 139

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM). 139

J. CLAIM 10 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 139

1. [10Pre] The system of claim 1, ... 139

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM). 139

K. CLAIM 11 IS OBVIOUS OVER GAVENS, INCLUDING INCORPORATED REFERENCES, IN VIEW OF KNOWLEDGE OF THE POSITA 140

1. [11Pre] The system of claim 1, ... 140

2. [11] wherein the controller causes the transfer of content on a periodic basis… 140

XI. GROUND 2: THE CHALLENGED CLAIMS ARE OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA 141

A. CLAIM 1 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 141

1. [1Pre] A system for storing data comprising: 142
2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;.....142
3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;.....143
4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to; 143
5. [1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;.....144
6. [1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;. 145
7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and 147
8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module. 148

B. CLAIM 2 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 150

1. [2Pre] The system of claim 1, ...150

2. [2] wherein the minimum quanta of address is equal to one block. 150

C. CLAIM 3 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 151

1. [3Pre] The system of claim 1, ...151

2. [3] wherein the minimum quanta of address is equal to one page. 151

D. CLAIM 4 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 151

1. [4Pre] The system of claim 1, ...151

2. [4] wherein the MLC non-volatile memory module is NAND flash memory….....151

E. CLAIM 5 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 152

1. [5Pre] The system of claim 1, ...152

2. [5] wherein the SLC non-volatile memory module is NAND flash memory….....152

F. CLAIM 6 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 152

1. [6Pre] The system of claim 1, ...152

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM). 152

G. CLAIM 7 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 153

1. [7Pre] The system of claim 1, ...153

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM). 153

H. CLAIM 8 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 153

1. [8Pre] The system of claim 1, ... 153

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM). 154

I. CLAIM 9 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 154

1. [9Pre] The system of claim 1, ... 154

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM). 154

J. CLAIM 10 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 155

1. [10Pre] The system of claim 1, ... 155

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM). 155

K. CLAIM 11 IS OBVIOUS OVER MOSHAYEDI IN VIEW OF KNOWLEDGE OF THE POSITA. 156

1. [11Pre] The system of claim 1, ... 156

2. [11] wherein the controller causes the transfer of content on a periodic basis… 156

XII. GROUND 3: THE CHALLENGED CLAIMS ARE OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA 157

A. CLAIM 1 IS OBVIOUS OVER SUTARDJA 157

1. [1Pre] A system for storing data comprising: 157

2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;.....157

3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;.....158

4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to; 159

5. [1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;.....159

6. [1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;.....160

7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and 162

8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module. 164

B. CLAIM 2 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 167

1. [2Pre] The system of claim 1, ...167

2. [2] wherein the minimum quanta of address is equal to one block.....167

C. CLAIM 3 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 168

1. [3Pre] The system of claim 1, ...168

2. [3] wherein the minimum quanta of address is equal to one page. 168

D. CLAIM 4 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 168

1. [4Pre] The system of claim 1, ...168

2. [4] wherein the MLC non-volatile memory module is NAND flash memory….....168

E. CLAIM 5 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 169

1. [5Pre] The system of claim 1, ...169

2. [5] wherein the SLC non-volatile memory module is NAND flash memory….....169

F. CLAIM 6 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 169

1. [6Pre] The system of claim 1, ...169

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM). 169

G. CLAIM 7 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 170

1. [7Pre] The system of claim 1, ...170

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM). 170

H. CLAIM 8 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 170

1. [8Pre] The system of claim 1, ...170

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM). 171

I. CLAIM 9 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 171

1. [9Pre] The system of claim 1, ... 171

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM). 171

J. CLAIM 10 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 172

1. [10Pre] The system of claim 1, ... 172

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM). 172

K. CLAIM 11 IS OBVIOUS OVER SUTARDJA IN VIEW OF KNOWLEDGE OF THE POSITA. 173

1. [11Pre] The system of claim 1, ... 173

2. [11] wherein the controller causes the transfer of content on a periodic basis… 173

XIII. CONCLUSION 174

I, Carl Sechen, Ph.D., declare as follows.

I. Overview:

1. I am over the age of eighteen and otherwise competent to make this Declaration. I have been retained as an expert witness on behalf of petitioner Phison Electronics Corporation ("Petitioner") for the above-captioned *inter partes* review. I am being compensated for my time in connection with this review at my standard consulting rate, which is $450.00 per hour. I have no personal or financial interest in the outcome of this proceeding. I make this Declaration on personal knowledge and on my expertise; where I state understandings with respect to patent law or procedure, the basis generally is discussion with counsel for Petitioner.

2. I have been asked to provide my expert opinions on a petition for *inter partes* review ("IPR") of U.S. Patent No. 8,891,298, entitled "LIFETIME MIXED LEVEL NON-VOLATILE MEMORY SYSTEM," to Dr. G.R. Mohan Rao (the "'298 patent") (Ex. 1001). The '298 patent, which issued on Nov. 18, 2014, from U.S. Appl. No. 13/455,267 filed Apr. 25, 2012, incorporated by reference Dr. Rao's U.S. Patent No. 7,855,916,[1] and claimed priority from U.S. Provisional App. No.

---

[1] ('298 1:12-16, "Rao '916," Ex. 1004.) Rao '916, entitled "NONVOLATILE MEMORY SYSTEMS WITH EMBEDDED FAST READ AND WRITE

61/509,257,[2] filed on July 19, 2011 ("Provisional," Ex. 1003). I understand that, according to USPTO records, the '298 patent is currently assigned to Vervain, LLC ("Patent Owner").

3. I understand that, based on the Provisional filing date, the prior art references and state of the art which I should consider in evaluating the '298 patent's patentable inventiveness (to the extent disclosed in the Provisional), should be July 18, 2011, or earlier.

4. In preparing this Declaration, I have reviewed the '298 patent and have considered each of the documents cited herein upon which I specifically rely. In formulating my opinions, I have relied upon my experience, education, and

---

MEMORIES," incorporated along with a then-pending application by Dr. Rao of the same title and specification ('298 1:16-21). Seven patents, U.S. Patent Nos. 9,196,385, 9,997,240, 10,950,300, 11,830,546, 11,854,612, 11,967,369 and 11,967,370, plus the '298 patent, all based on the same specification, are currently asserted in *Vervain, LLC v. Phison Electronics Corp.*, No. 1-24-cv-259 (W.D. Tex.). My declarations were submitted in support of pending petitions in *Phison Electronics Corp. v. Vervain, LLC*, PGR2024-00047 ('546) and -00048 ('612).

[2] ('298 1:7-12 [also incorporated by reference].) It is notable that the'298 application's title omitted "improved" from the Provisional title.

knowledge in the relevant art, including other background documents. I have also considered the viewpoint of a person of ordinary skill in the art ("POSITA") of computer memories (as described below) at the time of the alleged invention of the '298 patent, that is, close to, but earlier than July 19, 2011.

5. This Declaration is limited to my opinions on invalidity with respect to claims 1-11 of the '298 patent (the "Challenged Claims"). Those opinions, set forth in detail below, including factual bases and assumptions, are summarized:

A. The Challenged Claims are obvious over U.S. Patent No. 8,634,240 ("Gavens," Ex. 1045), including incorporated references, in view of the knowledge of a POSITA of the extensive background technology.

B. The Challenged Claims are obvious over U.S. Patent Appl. Pub. No. 2009/0327591 ("Moshayedi," Ex. 1043) in view of the knowledge of a POSITA of the extensive background technology.

C. The Challenged Claims are obvious over U.S. Patent Appl. Pub. No. 2008/0140918 ("Sutardja," Ex. 1042), in view of the knowledge of a POSITA of the extensive background technology.

6. While Gavens, Moshayedi, Sutardja, incorporated references, and others cited below as background technology form the primary bases for my opinion, I note that there are many other relevant references which further support a finding of invalidity as discussed herein. I reserve the possibility of amending or

supplementing this Declaration should additional information or arguments develop or be presented which affect my opinions.

II. Background and Qualifications:

7. My qualifications and credentials are set forth herein as further evidenced by my curriculum vitae (addendum hereto).

8. I earned a B.E.E. in Electrical Engineering from the University of Minnesota in 1975, followed by an M.S. in Electrical Engineering from the Massachusetts Institute of Technology in 1977. I earned a Ph.D. in Electrical Engineering from the University of California, Berkeley in 1986.

9. I have been a Professor of Electrical Engineering for 38 years. Since August 15, 2005, I have been a Professor of Electrical and Computer Engineering at the University of Texas at Dallas. From July 1992 to August 14, 2005, I served as a Professor of Electrical Engineering at the University of Washington. From July 1986 through June 1992, I served as an Assistant Professor and then Associate Professor of Electrical Engineering at Yale University.

10. Over these years, my research has focused on the design and computer-aided design of digital integrated circuits, including the design of dynamic random-access memory ("DRAM") and static random-access memory ("SRAM") modules. I have also designed several chips that included various types of embedded DRAM (eDRAM). In this research work, I have designed multiple sense amplifier circuits

for the various DRAMs, embedded DRAMs, and SRAMs. I have authored or coauthored over 200 papers and one book, the majority of which concern digital integrated circuit design and memory design.

11. As a professor, I have developed and taught numerous courses, including several courses that teach digital integrated circuit design and memory design, including extensive coverage of flash memory design. I have taught these courses continuously for the past 29 years. I have taught digital integrated circuit design and memory design, including flash memory design, to undergraduate and graduate students at the University of Washington and at the University of Texas at Dallas. DRAM and embedded DRAM design is covered in these courses, as is the design of sense amplifier circuits for various types of memory. NAND-Flash memory design is taught in detail, including both SLC (single-level cell) and MLC (multi-level cell) variants.

12. I was elected a Fellow of the Institute of Electrical and Electronics Engineers ("IEEE") in 2002 for contributions to placement and routing of ASICs. IEEE is the leading professional association for electrical engineers. The Board of Directors of the IEEE awards the rank of "Fellow" to individuals with an extraordinary record of accomplishments in any of the IEEE fields of interest. The total number of IEEE members who can be named Fellows in any one year cannot exceed one-tenth of one percent of the total voting IEEE membership.

13. I received several research and teaching awards during my career. I received the Semiconductor Research Corporation's Inventor's Recognition Award in 1988 and in 2001. I also received the Technical Excellence Award from the Semiconductor Research Corporation in 1994. While serving as Professor at the University of Washington, I received the Outstanding Research Advisor award from the Department of Electrical Engineering in 2002. In 2008, I received the Distinguished Teacher of the Year Award from the Department of Electrical and Computer Engineering (ECE) at the University of Texas at Dallas. I also received the Distinguished Teaching Award for the Erik Johnson School of Engineering and Computer Science in 2014. In addition, I received the *Graduate Teaching Excellence Award* from the ECE Department at UT Dallas, May 2024.

14. Over the years, I have also received funding to conduct research in computer circuits and memory designs, including area-efficient and reliable embedded DRAM and SRAM design. Together with my graduate students, I have designed and fabricated various types of computational VLSI chips.

15. I am a co-inventor on four issued patents directed to transistor and computational logic technologies. Three of the patents concern transistor-level programmable logic.

16. In summary, I am an expert in the field of computer architecture, including memory design and management, generally and as applied within the

context of the '298 patent, including NAND flash and other non-volatile memories ("NVM"). As set forth above, my experience includes practical product development, including performance considerations and design choices involving the memory technology state of the art before the earliest, July 19, 2011, priority date applicable to the challenged claims.

17. As an expert, I am qualified to evaluate, interpret and rely on reports, patents, scholarly papers and professional opinions of others, within my areas of expertise, as to authenticity and accuracy. Moreover, I am qualified to provide an opinion as to what a person of ordinary skill in the art (POSITA) would have understood, known and concluded before July 19, 2011, including the context in which a POSITA would understand and construe teachings and terminology in written publications.

III. My Understanding of the Applicable Law

18. I am not an attorney. However, as outlined in my curriculum vitae addended hereto, I have formed and presented opinions according to my technology expertise and hypotheticals presented to me in other cases involving validity of United States patents, including for the defendant in *Vervain, LLC v. Western Digital Corp.,* No. 6:21-cv-00488 (W.D. Tex. filed May 10, 2021, settled Aug. 8, 2023), which asserted the four earliest Rao patents, identified in paragraph 2 above, based on the same specification as the '298 patent. In those cases, as here, I was

informed or reminded of the legal standards which are relevant to my analysis and opinions. I summarize here the understandings relevant for this declaration, noting that greater detail may be provided upon any cross-examination, and others may be provided for any amendment or supplementation reserved in paragraph 6 above.

19. **Written Description and Claiming Requirements.** I understand that, as part of the bargain with the public of providing information to advance "science and the useful arts" in return for patent rights claimed and granted with notice to the public, 35 U.S.C. § 112 requires, in subsection (a), an adequate written description that both enables a POSITA to practice, without undue experimentation, the invention as claimed and reasonably conveys to the POSITA that "the inventor had possession of the claimed subject matter as of the filing date" and, in subsection (b), that a patent's claims be "concise" and, viewed in light of the specification and prosecution history, inform the POSITA about "the scope of the invention with reasonable certainty." I understand that an *inter partes* review does invalidate patent claims for such defects. In this declaration I will address such ambiguities that remain after applying claim construction principles in the alternative and will propose that any understood or inherent bases for claim limitations be similarly applied to prior art.

20. **What Constitutes Prior Art that May Invalidate a Patent.** I understand that former 35 U.S.C. § 102 listed conditions, including what is

considered "prior art," that may preclude (or invalidate) a patent of the '298 patent vintage, most pertinent here being:

> (1) under subsection (a), if the invention was known or used by others in this country, or was patented or described in a printed publication in this or a foreign country, before the date of the invention of the asserted patent; or
>
> (2) under subsection (b), if the invention was patented or described in a printed publication in this or a foreign country, or in public use or on sale in this country, more than one year before the effective filing date of the asserted patent; or
>
> (3) under subsection (e), if the invention was described in (a) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (b) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent; or

I understand that one or more of these conditions apply to the art relied upon here for my invalidity analysis.

21. **How the Prior Art Is Applied in the Invalidity Analysis:** ***Anticipation:*** I understand that a patent claim may be found to be invalid as "anticipated" under a subsection of 35 U.S.C. § 102 if a single prior art document (reference) (including any other prior art incorporated by reference in the prior art document) under that subsection describes every element of the claim explicitly or

inherently (necessarily present) as understood by a POSITA to allow making and using the claimed invention.

22. **How the Prior Art Is Applied in the Invalidity Analysis:** ***Obviousness:*** I understand that a patent claim may be invalid as "obvious" under 35 U.S.C. § 103 if the differences between the claimed invention and the prior art (including any other prior art incorporated by reference in the prior art document) are such that the claimed invention, as a whole, would have been obvious to a POSITA before the effective filing date of the claimed invention. The analysis calls for consideration of (a) level of ordinary skill in the pertinent art; (b) the scope and content of the prior art; (c) the differences between the prior art as a whole and the claim at issue; and (d) as appropriate, objective indicia of non-obviousness. The objective indicia, appropriate if there is a nexus with the claimed invention, include: (1) commercial success attributable to the invention; (2) long-felt but unsolved need for the invention; (3) failure of others; (4) skepticism by experts; (5) praise by others; (6) teaching away by others; (7) recognition of a problem which the claimed invention addresses; (8) copying by others; and (9) unexpected results.

23. I understand that obviousness may be found on a single reference (and any of its incorporated references) considered in view of the knowledge of the POSITA, and that a patent that incorporates another document by reference includes the disclosure of the incorporated document as if it was explicitly contained in the

patent. Where multiple references are proffered, I understand that while there is no requirement of an explicit teaching, suggestion or motivation to combine, generally there needs to be put forth some reason why a POSITA (if only by common sense or ordinary creativity) would have made the particular combination and that there is a reasonable expectation of success of the combination.

IV. Documents Considered in Formulating My Opinions:

24. In formulating my opinions, I have considered all documents cited in this Declaration, including, in particular, the documents in the chart below:

Exhibit	**Description**
1001	U.S. Patent No. 8,891,298, titled "LIFETIME MIXED LEVEL NON-VOLATILE MEMORY SYSTEM," to Rao
1003	U.S. Provisional Application No. 61/509,257, titled "IMPROVED LIFETIME MIXED LEVEL NAND FLASH SYSTEM" by Rao
1004	U.S. Patent No. 7,855,916, titled "NONVOLATILE MEMORY SYSTEMS WITH EMBEDDED FAST READ AND WRITE MEMORIES," to Rao
1006	Excerpts from prosecution history of '298 patent
1025	Excerpts from Rino Micheloni, et al., INSIDE NAND FLASH MEMORIES (Springer 2010)
1026	Excerpts from NONVOLATILE MEMORY TECHNOLOGIES WITH EMPHASIS ON FLASH: A COMPREHENSIVE GUIDE TO UNDERSTANDING AND USING NVM DEVICES (Joe E. Brewer & Manzur Gill eds. Wiley-IEEE Press 2008)

1027	Excerpts from MICROSOFT COMPUTER DICTIONARY (5th ed. 2002)
1030	Yu Cai et al., *Program Interference in MLC NAND Flash Memory: Characterization, Modeling and Mitigation* (IEEE 2013)
1031	Kang-Deog Suh et al., *A 3.3 V 32 Mb NAND Flash Memory with Incremental Step Pulse Programming Scheme*, 30 IEEE J. Solid State Circuits 1149 (Nov. 1995)
1032	Eitan Yaakobi et al., *Error Characterization and Coding Schemes for Flash Memories* (2010 IEEE Globecom Workshops, Miami, FL at 1856-60)
1033	Evans, Chris, *Enterprise MLC: How flash vendors are boosting MLC write endurance* (June 3, 2011)
1034	Nelson Duann, *SLC & MLC Hybrid* (Silicon Motion Flash Memory Summit, Santa Clara, CA, Aug. 12, 2008)
1035	Ashish Jagmohan et al., *Adaptive Endurance Coding for NAND Flash* (2010 IEEE Globecom Workshops, Miami, FL at 1841-45)
1036	Sungjin Lee et al., *FlexFS: A Flexible Flash File System for MLC NAND Flash Memory* (2009 USENIX Annual Technical Conference, San Diego, CA, June 14-19, 2009)
1037	Minyoung Sung & Kanghee Kim, *Asymmetric Flash Volume Management*, 58 *IEEE Transactions on Consumer Electronics* 455 (May 2012)
1038	Greg Atwood et al., Intel StrataFlash™ *Memory Technology Overview*, Intel Technology J. (Q4 1997)
1039	Taehee Cho et al., "A 3.3 V 1 Gb multi-level NAND flash memory with non-uniform threshold voltage distribution," *2001 IEEE International Solid-State Circuits Conference. Digest of Technical Papers,* ISSCC (Cat. No.01CH37177) (2001)

1040	U.S. Patent No. 6,456,528, titled for "SELECTIVE OPERATION OF A MULTI-STATE NON-VOLATILE MEMORY SYSTEM IN A BINARY MODE" to Chen
1041	U.S. Patent No. 8,078,794, titled "HYBRID SSD USING A COMBINATION OF SLC AND MLC FLASH MEMORY ARRAYS," to Lee et al.
1042	U.S. Patent Appl. Pub. No. 2008/0140918, titled "HYBRID NON-VOLATILE SOLD STATE MEMORY SYSTEM" by Sutardja
1043	U.S. Patent Appl. Pub. No. 2009/0327591, titled "SLC-MLC COMBINATION FLASH STORAGE DEVICE," by Moshayedi
1044	U.S. Patent Appl. Pub. No. 2010/0172179, titled "SPARE BLOCK MANAGEMENT OF NON-VOLATILE MEMORIES," by Gorobets et al.
1045	U.S. Patent No. 8,634,240, titled "NON-VOLATILE MEMORY AND METHOD WITH ACCELERATED POST-WRITE READ TO MANAGE ERRORS," to Gavens et al.
1046	U.S. Patent No. 8,806,301, titled "DATA WRITING METHOD FOR A FLASH MEMORY, AND CONTROLLER AND STORAGE SYSTEM USING THE SAME," to Yu et al.
1047	U.S. Patent Appl. Pub. No. 2010/0115192, titled "WEAR LEVELING METHOD FOR NON-VOLATILE MEMORY DEVICE HAVNG SINGLE AND MULTI LEVEL MEMORY CELL BLOCKS," by Lee
1048	U.S. Patent Appl. Pub. No. 2011/0271043, titled "SYSTEM AND METHOD FOR ALLOCATING AND USING SPARE BLOCKS IN A FLASH MEMORY," to Segal et al.
1049	U.S. Patent Appl. Pub. No. 2011/0153912, titled "MAINTAINING UPDATES OF MULTI-LEVEL NON-VOLATILE MEMORY IN BINARY NON-VOLATILE MEMORY," by Gorobets et al.

1050	U.S. Patent No. 5,930,167, titled "MULTI-STATE NON-VOLATILE FLASH MEMORY CAPABLE OF BEING ITS OWN TWO STATE WRITE CACHE," to Lee et al.
1051	U.S. Patent Appl. Pub. No. 2010/0172180, titled "NON-VOLATILE MEMORY AND METHOD WITH WRITE CACHE PARTITIONING," by Paley et al.
1053	U.S. Patent Appl. Pub. No. 2009/0268513, titled "MEMORY DEVICE WITH DIFFERENT TYPES OF PHASE CHANGE MEMORY," by De Ambroggi et al.
1054	U.S. Patent Appl. Pub. No. 2009/0307418, titled "MULTI-CHANNEL HYBRID DENSITY MEMORY STORAGE DEVICE AND CONTROL METHOD THEREFOR," by Chen et al.
1055	U.S. Patent Appl. Pub. No. 2010/0058018, "MEMORY SCHEDULER FOR MANAGING INTERNAL MEMORY OPERATIONS," by Kund et al.
1061	Duane H. Oto et al., *High-Voltage Regulation and Process Considerations for High-Density 5 V-Only E^2PROM's*, SC-18 IEEE J. Solid State Circuits 532 (Oct. 1983)
1062	Gheorghe Samachisa et al., *A 128K Flash EEPROM Using Double-Polysilicon Technology*, SC-22 IEEE J. Solid State Circuits 676 (Oct. 1987)
1063	Masaki Momodomi et al., *An Experimental 4-Mbit CMOS EEPROM with a NAND-Structured Cell*, 24 IEEE J. Solid State Circuits 1238 (Oct. 1989)
1064	Tae-Sun Chung et al., *A survey of Flash Translation Layer*, 55 J. Systems Architecture 332-43 (2009)
1065	Yoshiba Iwata et al., *A High-Density NAND EEPROM with Block-Page Programming for Microcomputer Applications*, 25 IEEE J. Solid-State Circuits 417 (Apr. 1990)

1066	Simona Boboila and Peter Desmoyers, *Write Endurance in Flash Drives: Measurements and Analysis*, Usenix Conference (Feb. 2010)
1067	SiliconSystems, *Increasing Flash SSD Reliability*, StorageSearch.com (April 2005)
1068	Taehee Cho et al., *A Dual-Mode NAND Flash Memory: 1-Gb Multilevel and High-Performance 512-Mb Single Level Modes*, 36 IEEE J. Solid State Circuits 1700 (Nov. 2001)
1069	Ken Takeuchi, *A Multipage Cell Architecture for High-Speed Programming Multilevel NAND Flash Memories*, 33 IEEE J. Solid-State Circuits 1228 (Aug. 1998)

V. Relevant State of the Art as of July 19, 2011

25. I describe here the relevant state of the art prior to July 19, 2011, of NAND flash memory technology, which is the only non-volatile memory ("NVM") explained in any detail in the '298 patent background and four "embodiments of a NAND flash storage system that provides long lifetime (endurance) storage at low cost" (3:43-45). As I analyze at Section VI below, the '298 specification disclosed a NAND flash memory system composed of three components: (1) a bank of multiple physical modules of "MLC" NAND flash memory, (2) a bank of multiple physical modules of "SLC" NAND flash memory, and (3) a "controller" that "controls access" to the memories (5:9-15) including by "maintain[ing] a translation table/address map" (5:16-17) acknowledged to be the established way to use NAND flash memory (2:28-3:13 [logical-to-physical, "L2P," mapping, flash translation layer "FTL"]; 5:18-34). No part of the disclosed and claimed memory system is

even suggested to be new – only two optional methods of operation. One is "a method for utilizing a NAND flash memory system" shown in Figs. 3a and 3b (5:55-6:23), where a "data integrity verification check" compares data to be written (but "retained") to "stored" data read after writing and, upon failure to match, results in remapping to "SLC NAND flash memory." This is a variant among many known error management options for NAND flash memory. The second optional method is unillustrated and only functionally stated in two sentences (6:24-35) as (1) "allocat[ing] 'hot' blocks, . . . that receive frequent writes, into the SLC NAND flash module" and "'cold' blocks, . . . that receive infrequent writes, into the MLC NAND flash module" and (2) "on a periodic basis, . . . transfer[ring] the contents of those blocks [written to most frequently] into the SLC NAND flash module." The first sentence, if read as "*logical* blocks," describes then-implemented operation of hybrid SLC-MLC systems; the second sentence seems to describe a variant of long-applied static "wear-leveling" among "*physical* blocks." The two (possibly three) disclosed operations purportedly "increase the reliability and lifetime" of the system or module (1:25-30; 3:58-4:24; 4:51-58), disclosed and claimed as distinct MLC and SLC NAND flash memory modules,[3] by redirecting, upon the function conditions,

---

[3] Both use the same type of memory cells, which require erase before write (program) in contrast to the proposed substitute technologies, namely, RRAM, PCM

data otherwise directed to MLC memory to SLC memory. The latter is well known to be reliable after an order of magnitude larger number of program (write)/erase cycles than the former. At a level not much higher than that of the patent disclosure, directing data to known longer lifetime memory is an obvious approach to "increase" lifetime of a mixed memory system. If there were no other constraints, such "increase" may be expected to be greatest by directing *all* the data to the more reliable memory – and, indeed, to the alternative technologies (note 3 above). Although I have some question whether the functions disclosed and claimed in the '298 patent actually increase system lifetime, I am, and a POSITA would be, familiar with a large body of prior techniques, both proposed and implemented, to extend the reliable usefulness of NAND flash memory systems, whether purely SLC memory (NAND flash before it was so distinguished after the late 1990's introduction of physically different MLC) or later in combination with MLC. I review here details of that technology that will demonstrate the obviousness of the '298 patent claims.[4]

---

and MAGRAM, which "allow data to be written over existing data (<u>without prior erase</u> of existing data)" (2:2-5 [emphasis added]).

[4] This fills in material gaps in the '298 specification which otherwise assumes the understanding of a POSITA as to the structure and operation of NAND flash and only provides examples of a NAND flash memory and controller in the incorporated

I will follow in Section VII with a review of the '298 patent disclosure and prosecution history. In Sections VIII through X, I cover examples of specific prior art over which the memory systems of claims 1-11 are obvious.

A. NAND Flash Memory and Operations

26. **The Base NAND Flash Memory Cell**: Unlike other nonvolatile memory (NVM) systems such as rotating magnetic disk storage, in which information could be written into a location and then written over ("direct write" or "write in place"), NAND flash evolved from Read-Only Memory ("ROM") which were only written once (perhaps on manufacture) to Erasable Programmable Read-Only Memory ("EPROM") which required erasing (originally by ultraviolet radiation) of constituent memory cells before they could be programmed. Electrically Erasable Programmable Read-Only Memory (EEPROM), such as

---

'916 patent. I review in this declaration eleven (Exs. 1034, 1038, and 1061-1069) of the forty "information disclosure statement" ("IDS") documents filed by Dr. Rao with his April 25, 2012, application (cited here by "IDS#_" in reference to their listing on the three IDSs filed that day [Ex. 1006 at 26-36]). I understand these documents to provide both admitted prior art and, importantly, the use of terms in the art as Dr. Rao used in the '298 specification and claims.

NAND flash memory, use floating-gate field-effect transistors that may be electrically erased and written:

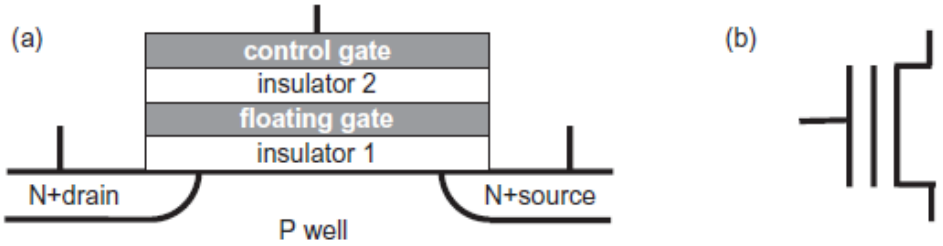


Figure 1.1. Floating-gate transistor: (*a*) elements of the transistor structure and (*b*) circuit symbol.

(NONVOLATILE MEMORY TECHNOLOGIES WITH EMPHASIS ON FLASH: A COMPREHENSIVE GUIDE TO UNDERSTANDING AND USING NVM DEVICES (Joe E. Brewer & Manzur Gill eds. Wiley-IEEE Press 2008) ["Brewer," Ex. 1026] at 3.) At this cell level, data is represented by a charge stored on the floating gate to accomplish nonvolatile data storage; the charge determines whether the transistor will conduct when a fixed set of "read" bias conditions is applied. *Id.* The threshold voltage (related to the charge Q) is shown in the following relationship to the conduction of current:

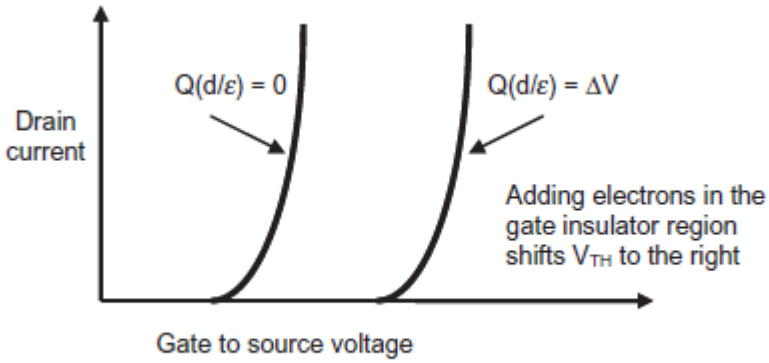


Figure 1.3. Shift of current–voltage characteristics because of inserted charge.

(Brewer at 4.) Thus, a threshold voltage can be programmed into the cell by injection of electrons (charge) along an analog continuum, which charge is held in the floating gate. From the early decades of NAND flash, the two threshold voltage levels or states shown were used to represent the two possible states – one binary "bit" – of information as "erased" or "programmed":

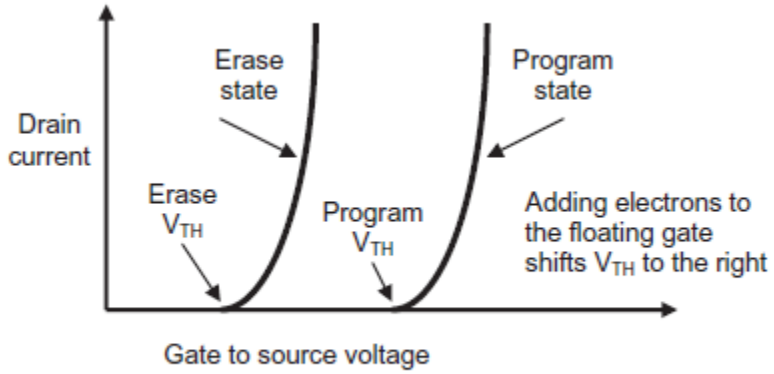


Figure 1.10. Erase state and program state current–voltage characteristics.

(Brewer at 11.) NAND flash uses Fowler-Nordheim ("FN") tunneling to inject electrons to shift the threshold voltage curve from the "erase" (customarily representing a binary "1") to the right to a "program" state (customarily representing a binary "0") (*e.g.*, Brewer at 64; Rino Micheloni, et al., INSIDE NAND FLASH MEMORIES (2010) ("Micheloni," Ex. 1025) at 24); in this write process, the charge can only be increased. One cannot by the same process move the curve to the left; it must be erased in a different, slower and more destructive process before programming again (thus "erasable programmable read-only" versus "write in place"). The tunneling operation, along with other processes (including "disrupts"

by operations on nearby cells organized in groups), even sensing (reading), progressively degrades the functionality of the EEPROM transistor much faster than the storage components of other NVM are degraded by their processes.[5] The NAND erase process also applies FN tunneling (Brewer at 64), typically applied involving additional areas of the semiconductor substrate (Micheloni at 25-26). Although the NAND cells need to be erased before being programmed (two tunneling stresses), these operations (each a few milliseconds) typically are not performed seriatim on a particular cell, but erasing is typically done in the background (slow, as much as a second), and already erased cells are written to (thus not "write in place"). Before the priority date of the challenged patent, basic strategies had been settled upon and

---

[5] (*See* Duane H. Oto et al., *High-Voltage Regulation and Process Considerations for High-Density 5 V-Only E^2PROM's*, SC-18 IEEE J. Solid State Circuits 532, 533 Fig. 2 (Oct. 1983) ["Oto," IDS#13, Ex. 1061] [showing the difference between threshold voltages for charged and discharged states of a FLOTOX {floating gate tunneling oxide} cell at 10-100K "clear/write cycles" shrinking to about the threshold voltage for the discharged state]; Gheorghe Samachisa et al., *A 128K Flash EEPROM Using Double-Polysilicon Technology*, SC-22 IEEE J. Solid State Circuits 676, 680 Fig. 10, Table I (Oct. 1987) ["Samachisa," IDS#14, Ex. 1062] [voltage margin v. program/erase cycles].)

many others proposed to optimize programming throughput and balance degradation to put off failure of enough groups of cells that would compromise the intended use of a NAND memory system. These solutions involve both hardware structure as well as different ways to operate the hardware. A NAND memory cell by itself is not "memory" as it does not even provide for retrieval of stored information, essential to its usefulness.[6] "While the heart of a semiconductor memory is the cell, the surrounding circuitry is the mechanism that makes it usable." Brewer at 5.

27. **NAND memory cell arrays.** EEPROM memory cells are organized in a matrix to optimize the use of silicon area. (*E.g.*, Micheloni at 20.) In the NAND configuration, which became the prevalent configuration by the time of the 2011 priority date of the patent at issue, multiple EEPROM storage cells are arranged in strings with selection transistors at either end[7] along a *bitline*. (*See* Masaki Momodomi et al., *An Experimental 4-Mbit CMOS EEPROM with a NAND-Structured Cell*, 24 IEEE J. Solid State Circuits 1238, 1238-39 Fig. 2 (Oct. 1989)

---

[6] (*See* MICROSOFT COMPUTER DICTIONARY 333 (5th ed. 2002) [Ex. 1027]: "**memory** *n.* A device where information can be stored and retrieved.")

[7] This saves circuitry on a chip – such as a transistor – associated in other arrangements for reading and writing of *each* storage cell. (*See* Oto at 534 Fig. 6; Samachisa at 676 [separate select transistor per cell].)

["Momodomi," IDS#8, Ex. 1063]). In the standard array, each of the storage cells are connected across *wordlines* that define physical "rows," sometimes called "pages," of typically 4KB or 32K cells:

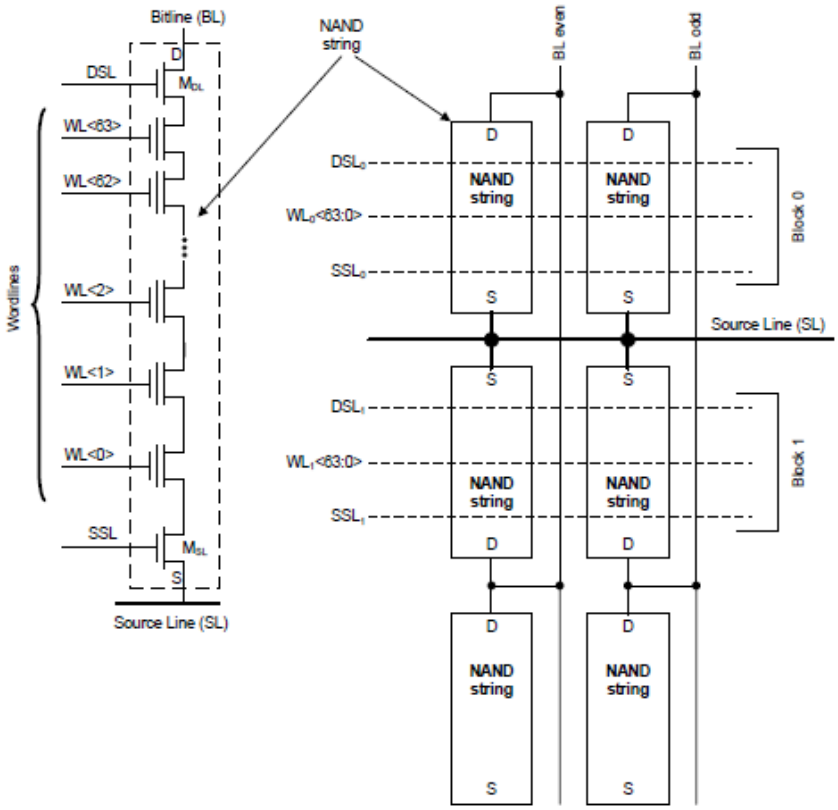


Fig. 2.2. NAND string (left) and NAND array (right)

(Micheloni at 21.)

> The wordlines are used to select the transistor of interest in the string. To access data from a string, the reading process requires that all nonselected transistors be turned on while only the selected transistor is allowed to influence [by its stored charge/threshold voltage] the current flow through the string. . . . For a single write operation the word present in an input/output register is used to determine the data inserted into the cells for that word [also called a page].

(Brewer at 6.) In long standard NAND operation, the smallest programmable unit of NAND flash is a row of cells constituting a "page" (traditionally a binary string that represents approximately a page of type), where the parallel programming of the cells in a page is such that a cell may be left in the logical 1 state from erasure or a cell may be programed to logical 0. In the latter case, trapped charge is added to the floating gate, and in this situation, the detection circuitry in the module will find largely an absence of current conducting from drain to source for the nominally applied control gate voltage.

28. **NAND *flash* memory – *erasable blocks*.** In advancing EEPROM technology, Toshiba in 1985 coined the term "Flash memory" for its new device erased in a "flash." (Brewer at xvii.) "Flash memory is an EEPROM where the entire chip or a subarray within the chip may be erased at one time." (Brewer at 2 [note omitted]; Samachisa at 676; Momodomi at 1239.) A NAND flash "block," as shown in Micheloni Fig. 2.2 (para. 27 above), is an organization of multiple (typically 64 or 128) "pages" or rows of (typically 32K) data storage cells controlled by respective wordlines. (Spare cells may be included for such functions as error correction code, "ECC".) A depiction of a typical NAND memory array is:

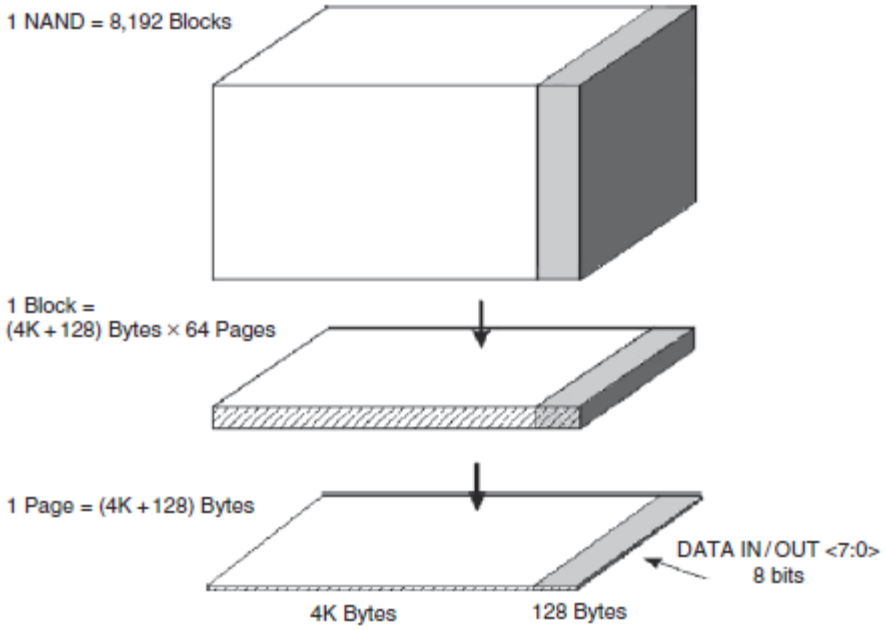


Fig. 2.7. NAND memory logic organization

The throughput (sometimes called "performance") strategy here is to write 4KB pages (32K+ cells) in parallel (taking on the order of a few milliseconds), but erase a whole block of 2,048K+ cells (taking on the order of several second, and typically stressing the cells in the block more than writing) in the background in between reading and writing operations.

29. A memory cell array or a block organized within the array to store multiple pages of information as in a NAND flash memory, is not "memory." To make a memory cell array a "memory," one needs the circuitry to program the information into the cells, strings and blocks. (*E.g.*, Momodomi at 1241 [data latch circuitry and charged pumping circuitry, on-chip high-voltage generator]; Brewer at 236-37.) The memory also needs the basic ability to sense and retrieve what is

stored, requiring a sense amplifier (SA) and one or more data latches per cell (one per bit stored in a cell) of a page (*e.g.,* Micheloni at 225, Brewer at 237-38), that may be shared with the programming operation in a page buffer (Brewer at 239, Fig. 6.20). There are also required interfaces for commands and addresses for which cells are to be programmed, read or erased and input/output of the data programmed or read. Thus, a **NAND flash memory** is shown as:

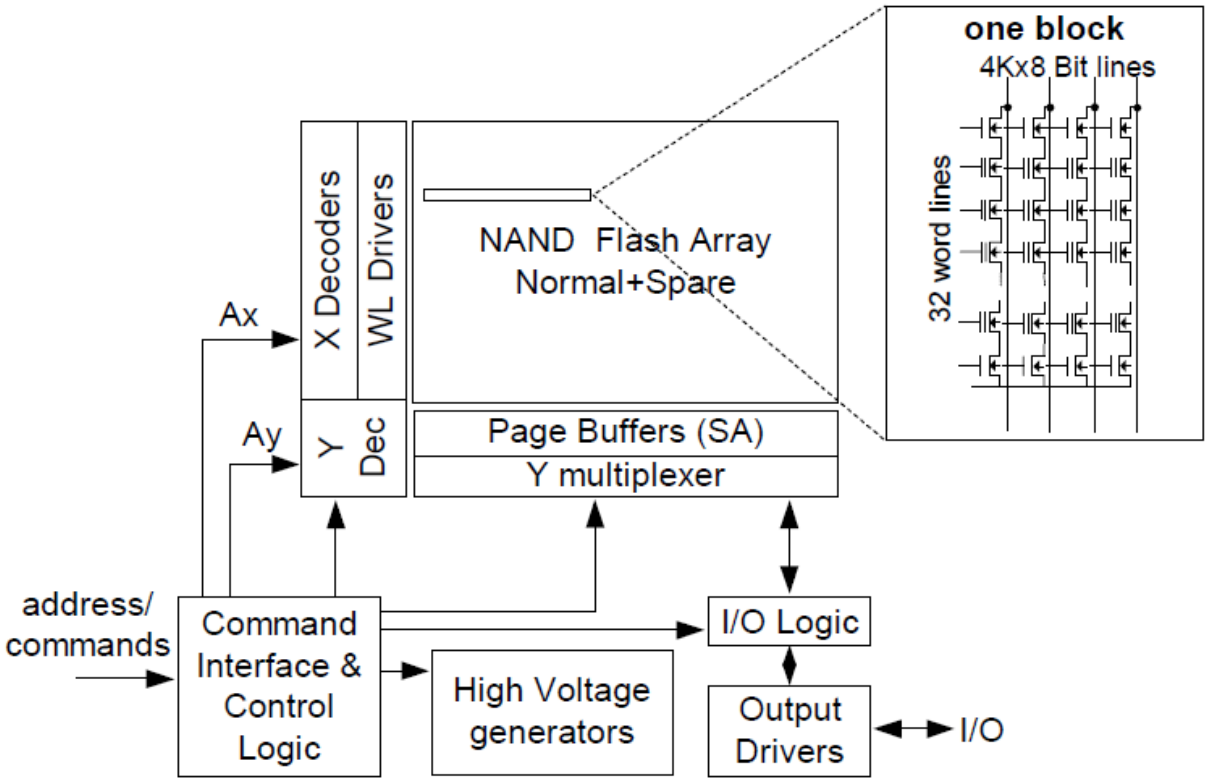


Fig. 15.1. NAND Flash memory block diagram

(Micheloni at 424.) A POSITA would understand this to be a flash memory that includes memory cells, strings and blocks in an array none of which themselves constitute a "memory."

30. **NAND flash memory *modules.*** While this arrangement may constitute a memory embedded with other circuitry, typically it will be understood as its own chip or module[8] that is self-contained in its function and can be interchanged with other NAND flash memory modules such as a NAND flash memory card or SSD (solid state drive) including an array or banks of flash memories (sometimes simply called "flash"):

---

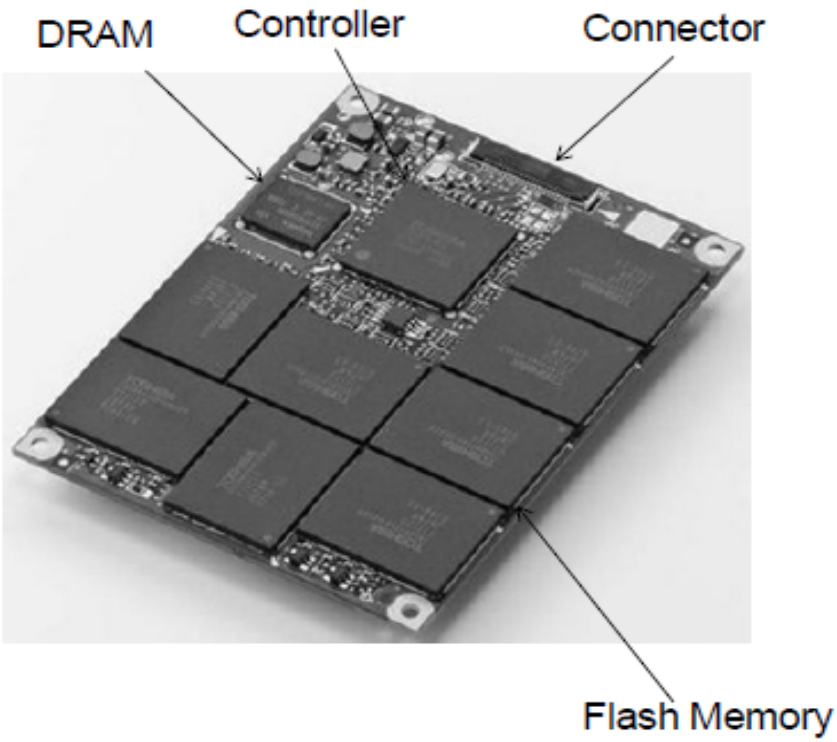
[8] The MICROSOFT COMPUTER DICTIONARY 346 (Ex. 1027) provides the following definition for "module":

> In hardware, a self-contained component that can provide a complete function to a system and can be interchanged with other modules that provide similar functions. *See also* memory card, SIMM.

Similarly, from the Merriam-Webster online dictionary, https//www.merriam-webster.com/dictionary/module:

> **3b:** a usually packaged functional assembly of electronic components for use with other such assemblies

A "flash memory module," as used for embedded applications, "can readily be integrated into the host logic device" and includes a bus interface unit distinguished from the flash memory core, which includes a memory control as well as a flash memory array. (Brewer [Ex. 1026] at 395-96, Fig. 9.13.)



(Micheloni at 14, Fig. 1.16.)

31. The memory **device controller** typically has a microprocessor with some internal volatile memory (typically RAM) and often external volatile memory (as the DRAM shown in Micheloni Fig. 1.16 above) to support its functions of interfacing with a host computer to read and write pages of data with optimal performance (transfer speed) and reliability (data integrity and retention), including appropriate data correction. A functional block diagram follows including the host and SSD with a memory controller interfacing with banks of NAND flash modules:

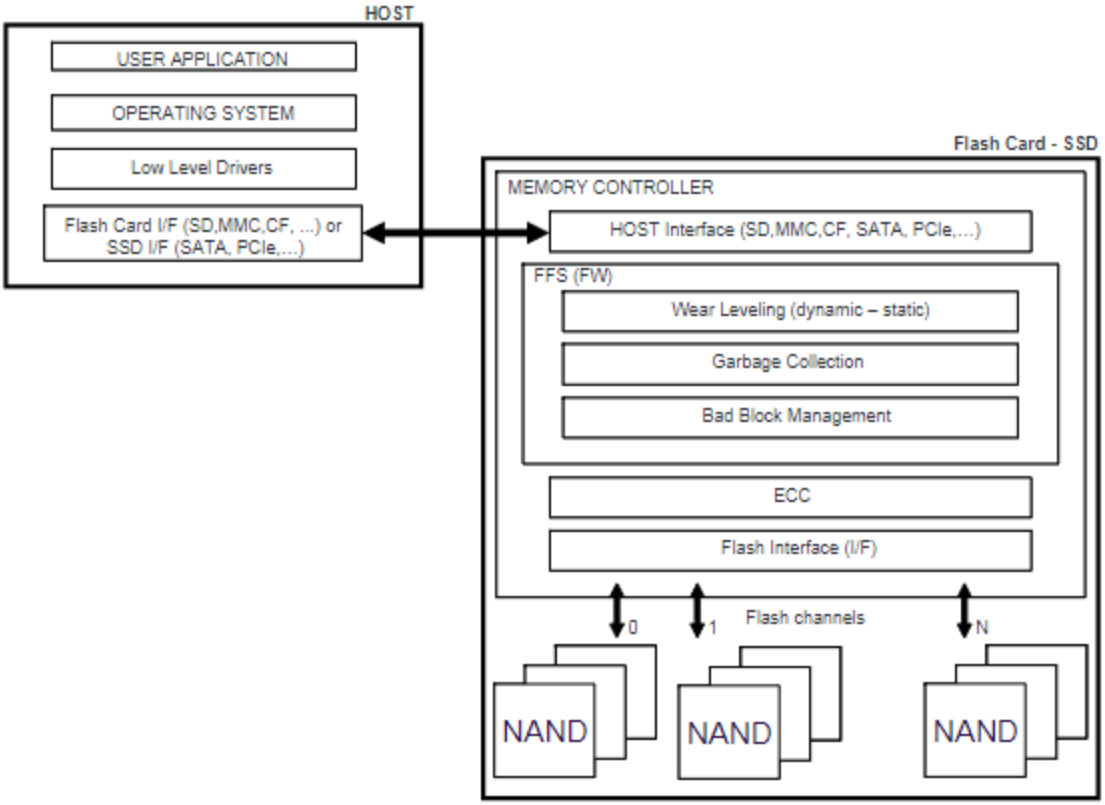


Fig. 2.33. Functional representation of a Flash card (or SSD)

(Micheloni at 39.)

32. In addition to the interfaces with the host (providing commands and "logical addresses" as described below for data to be written or read) and with the banks of NAND flash modules (with "physical addresses" as described below), the controller may include an ECC processor that detects and corrects data errors (up to a performance trade-off discussed below).

33. Crucial to the operation of flash memory is a **flash file system** (FFS) that performs the matching (or translation) to physical memory cells of host commands and addresses visible to the host such as file names associated with a

"logical address." Because, as described in paragraph 26 above, NAND flash programming can only increase the threshold voltage of a cell representing data and must be returned to the erased state before programming; that is, updating (or changing) of a logical file (or page) needs to be made to physical cells in an erased state. Flash memory operation (shown in functional sublayers in Micheloni Fig. 2.33 at ¶ 31 above) also includes garbage collection (to consolidate valid data and to erase blocks with invalid data to free them up for new writes), wear leveling (to balance wear on blocks to avoid premature failure of overused blocks) and bad block management (to retire blocks not considered reliable for continued use). (Micheloni at 40.)

34. To perform these functions, tables, implemented as a software **Flash Translation Layer** or FTL, are widely used in order to map sectors (traditional organization for hard disk drives) and pages from logical (host-visible) locations (addresses) to physical locations (addresses) in the flash memory.[9] (Micheloni at

---

[9] There is a mismatch between the traditional HDD "sector" of 512 bytes of an HDD and the NAND flash "block" of multiple (*e.g.*, 64) "pages," typically of multiple (*e.g.*, 8) "sectors." Various strategies were considered for mapping of bits of a data file (its name visible to the host/user, associated with a logical address, "page" or "blocks") to the NAND flash physical location (groups of addressable cells). (*E.g.*,

40.) This is illustrated as follows, with the upper row being the host-logical view of the memory and the lower row being the physical flash memory:

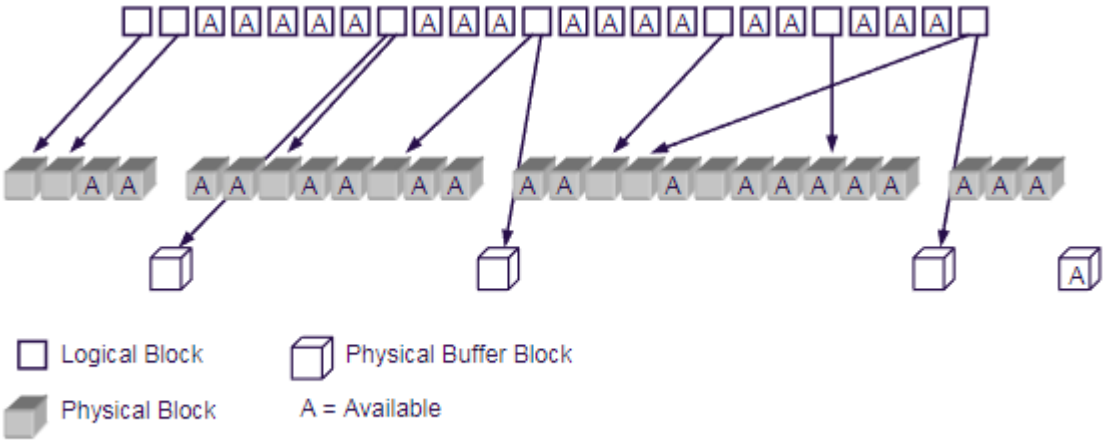


Fig. 2.34. Logical to physical block management

(Micheloni at 41.) Flash memories lack the "update in place" capability of other NVMs, such as magnetic disk: programming of a flash memory page must start from its erased state, the entire block must be erased to erase a page that is written, and erasing a block would substantially delay writing to that block. Thus, the updated data needs to be written to a different location from the outdated data, generally to another block. To make this transparent to the host, which only "sees" a logical address target, the FTL allows emulation of independent modification of NAND pages through "logical to physical remapping." (*Id.* at 492; also called "L2P").

Tae-Sun Chung et al., *A survey of Flash Translation Layer*, 55 J. Systems Architecture 332-43 (2009) ["Chung," IDS#39, Ex. 1064] [page, block or hybrid mapping].)

When the host sends a command to write or update a certain logical page, the FTL writes the associated data (received from the host) to a different available physical page than currently mapped from the logical address and updates the table storing the logical to physical mapping. (*Id.*) The physical page containing the old data is marked as invalid. (*Id.*) Thus, in NAND flash operation, every update or write to a previously mapped physical location that has content must be remapped to another empty or erased location. If the update is to modify only part of the data in the targeted (currently mapped) page, the remaining data in that page, not to be modified (thus, considered valid), is read and combined with the update and written to the remapped location. In both cases, the previously mapped page is marked invalid. The garbage collection process works to coagulate the invalid pages into entire blocks of invalid pages which are then erased in background operation when the memory device is not occupied with reading or writing.

35. **Wear leveling**. It is considered important to keep flash memory aging (the number of program/erase cycles) of each block in a NAND flash memory system to a minimum and as uniform as possible. (Micheloni at 41.) As reviewed at paragraph 26 above, the NAND flash cell degrades (ages) to some extent after each (program/erase) cycle, comprising an initial erase and a subsequent number of writes (programs). "*Endurance* is the term used to describe the ability of a device to withstand [stresses of normal processes of writing and/or erasing cells that will

eventually degrade the properties of the memory or disturb the contents of the memory] and is quantified as a minimum number of erase-write cycles or write-read cycles that the chip can be expected to survive." (Brewer at 13.) This is a rating guarantee of the manufacturer; most often, "the device will continue operating well after the max cycling spec has been exceeded." (*Id.* at 30.)[10]

36. A wear leveling software module makes sure all NAND blocks [of the same endurance expectation] are cycled an identical number of times, avoiding some blocks ending their cycle life [and compromising the useful life of the system] while

---

[10] The determination of endurance is not exact. Various simulations have been performed with assumptions that each cycle comprises one simultaneous block erase and eight successive page program cycles. (Yoshiba Iwata et al., *A High-Density NAND EEPROM with Block-Page Programming for Microcomputer Applications*, 25 IEEE J. Solid-State Circuits 417, 419 (Apr. 1990) [IDS#5, Ex. 1065].) In an investigation of system ("SLC" or "MLC" devices) endurance, an "end-of-life signature" was discerned in increased latency of operations. (Simona Boboila and Peter Desmoyers, *Write Endurance in Flash Drives: Measurements and Analysis* p. 4, Table 1, p. 8, Fig. 13, Usenix Conference (Feb. 2010) ["Boboila," IDS#16, Ex. 1066].) In that investigation, "nominal" endurances provided by manufacturers were exceeded by nearly a factor of 100. (*Id.* p. 4.)

others being cycled only a few times. (Micheloni at 494.) Some wear leveling is accomplished by ordinary NAND logical to physical programming: "that is, each time the host application requires updates to the same (logical) sector, the memory controller dynamically maps the sector onto a different (physical) sector, keeping track of the mapping either in a specific table or with pointers. The out-of-date copy of the sector is tagged as both invalid and eligible for erase. In this way, all the physical sectors are evenly used, thus keeping the aging under a reasonable value. (*Id.* at 41.)

> There are two levels of wear leveling. In the *first level wear leveling* [also called dynamic] new data [directed by the host to a logical block] is programmed into a free [physical] block with the fewest write/erase cycles. Two techniques to achieve this are the chain method and the array method. (*Id*. at 494.)
>
> In the *chain method,* the pool of available blocks is organized in a chain . . . . A simple round robin algorithm selects one block after another. When a block is erased, it is added to the chain. (*Id*.)
>
> In the *array method* an age vector stores the number of each block's program/erase cycles. This information can be used by the wear leveling algorithm to ensure a uniform exploitation of all the blocks. (*Id*.)
>
> In the *second level wear leveling* [also called static wear leveling] from time to time blocks containing long-lived [static] data are moved (copied) into other blocks even though they never received an update

> command. The original block can then be used to store new incoming data. The second level wear leveling is triggered when the difference between the maximum and the minimum number of cycles per block reaches a predetermined threshold. (*Id*.)

Both dynamic (at the time of writing) and static (in the background between write and read operations) wear leveling may be used in a system.[11] "Flash cards that use dynamic wear leveling algorithm only write across dynamic or free data areas. By

---

[11] It is important to understand information flow in NAND flash operation relative to high-level descriptions such as the ambiguous use of the term "block" above. Dynamic wear leveling involves writing of new data (updated data) to a location currently pointed to by the logical block address (LBA). The original data at the physical block address (PBA) pointed to by the LBA in the FTL is placed in a page buffer, and the page buffer is updated with the new data. Finally, the updated page buffer is written to a new, favored location. Static wear leveling "moves" all of the pages from an original (physical) block with a low erase count to a new (physical) block that has a high erase count. This achieves wear leveling since the data in the original block is presumed to have been updated relatively rarely (the erase count lagging other blocks), and so when placed in (by remapping to) the new block, it will typically result in the erase count of the new block stagnating at its current value. Meanwhile, the original block is erased and placed on the free block list.

far the best endurance is provided by static wear leveling, where the data is written equally to all blocks of the storage device." (SiliconSystems, *Increasing Flash SSD Reliability*, StorageSearch.com at p. 4 (April 2005) ["SiliconSystems," IDS#18, Ex. 1067]; *also* Boboila p. 9 [devices employing static wear leveling took longer to wear out than those only employing dynamic wear leveling].)

37. **Garbage collection**. Because in NAND flash a logical page may be written only to a physical page that is in an erased state, and because erasure is time-consuming, an update to a logical page (mapped to and represented) in a physical page of a physical block results in the logical page being remapped to another physical block whose corresponding page is in an erased state. This leaves the data in the previously mapped physical location (page) obsolete or "invalid." An increasing portion of the memory is thus filled with pages comprising invalid data, which will need to be erased to make room for new data. This is resource-intensive because, before an entire block can be erased, valid data in the block needs to be copied to a new physical location. The "garbage collection" function of FTL is the consolidation of valid data in a new physical block while the prior physical block(s) are erased to make available blocks for new data.

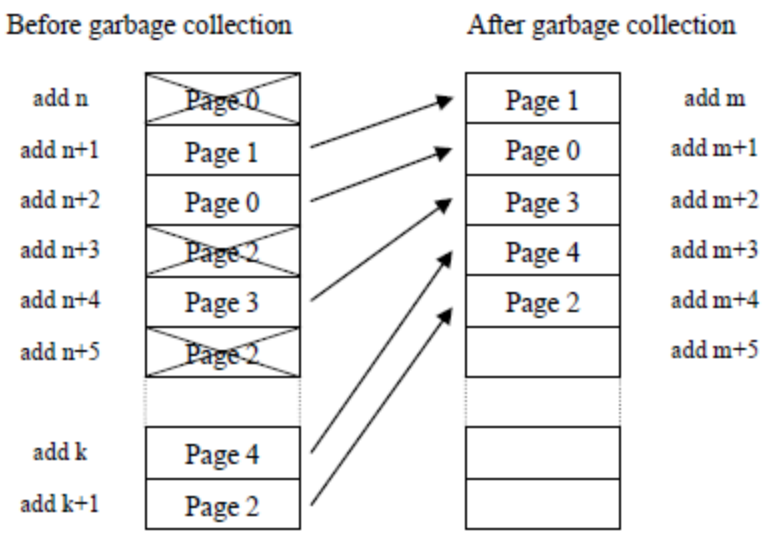


Fig. 17.7. Garbage collection

(Micheloni at 493.) Higher garbage collection efficiency, measured by the ratio of the number of invalid pages to the total number of pages in a physical block, is desirable in order to reduce the number of rewrites of valid data while erasing and opening the block to new data. (*Id.*)

> *Hot data* are the most frequently updated data. *Cold data* (or non-hot data) are seldom updated data. In a given block, it is advantageous to gather (cluster) all hot data or all cold data, because blocks with hot data will soon become garbage. Cleaning such blocks is beneficial because they have high garbage collection efficiency.

(*Id.* at 494.)

38. **Bad block management and ECC.** "Bad blocks" are "blocks which contain one or more locations whose reliability is not guaranteed." (Micheloni at 42.) Thus, a "*Bad Block Management* (BBM) module creates and maintains a map

of bad blocks." (*Id.*) "[T]his map is created during factory initialization of the memory card, thus containing the list of bad blocks already present during the factory testing of the NAND Flash memory modules." (*Id.*) The map is updated as the memory device is used when a block is determined to be bad (*id.*) and thereby retired. It is obvious that that determination may be made in various ways, depending, among other things, on what measure of unreliability is used. As described in paragraph 51 below, there are various "data integrity checks" that may be applied, with different actions taken in response to the results of those checks. One would expect there will be some less than optimally performing cells in any given block of 64 pages each of 32K cells. Rather than take out of service an entire block of millions of cells for a few unreliable cells, with ECC, the suspect cells can be kept in service and the data represented therein corrected upon retrieval (*see* ¶ 52 below). As explained in the prior art discussed below, there is a trade-off as to how many erroneous bits can be corrected through the selected levels of ECC code and processing.[12]

---

[12] Some manufacturers rate their flash product endurance including consideration of the number of bits of error correction implemented. (*E.g.*, SiliconSystems, pp. 3-4 [including chart comparing SLC NAND and MLC NAND, the number of bits corrected by ECC and the application of no, dynamic, or static wear leveling].)

39. **System Life**: The '298 patent specification presents "embodiments of *a NAND flash storage system that provides long lifetime* (endurance) storage at low cost" (3:43-45 [emphasis added]) or relating "to a system and method of increasing the reliability and lifetime of a NAND flash storage system . . . through the use of a combination of multi-level cell (MLC) and single level cell (SLC) NAND flash storage" (6:53-57). The BACKGROUND OF THE DISCLOSURE provides the patent's only hint of how the disclosed (or claimed) embodiments provide long or increased storage lifetime:

> *MLC NAND flash* SSDs are slowly replacing and/or *coexisting with SLC NAND flash in newer SSD systems*. . . . Generally, MLC NAND flash enjoys greater density than SLC NAND flash, at the cost of a decrease in access speed and lifetime (endurance). . . MLC NAND flash has a much lower lifetime (endurance) [3-10K writes] than SLC NAND flash [50-100K writes].

(3:14-27 [emphasis added].) It is a straightforward, obvious proposition to favor the use of a memory with higher (expected) endurance over a memory with lower endurance in order to "increase" the (average) endurance of a system that includes both, relative to greater use of the lower endurance memory. Endurance tests for the cell level are not exact (note 6 above), and the tests for devices have resulted in as much as an order of magnitude increase in actual use relative to the manufacturer-stated endurance (note 10 above). Any write operation (particularly if preceded by

an erase) in a NAND flash system will marginally decrease the lifetime of the written part of the system, i.e., a page of cells. However, the wear leveling outlined in paragraph 36 above "increases" useful lifetime of the flash memory system by minimizing unnecessarily early failure caused by greater than average wear on parts of the system.[13] The clustering of "hot" or "cold" data for rewriting in garbage collection as described in paragraph 37 above is another strategy to avoid unnecessary writes and erases. The use of ECC described in paragraph 38 above clearly avoids unnecessary writes and erases. These NAND flash system life-enhancing strategies were well-known before the July 19, 2011, priority date of the '298 patent. Those strategies had already been applied to hybrid systems including both "SLC" and "MLC" memories.

---

[13] "A device with no wear leveling wears out faster because data is written to the same physical block." (SiliconSystems, p. 4.) The Boboila investigation concluded that "wear-leveling techniques lead to a significant increase in the endurance of the whole device, compared to the endurance of the memory chip itself, with static wear-leveling providing much higher endurance than dynamic wear leveling." (Boboila, p. 9 [both SLC and MLC devices investigated].)

B. MLC and SLC

40. "SLC" as an abbreviation for "single level cell" is somewhat of a misnomer, as it actually describes the use in "original" (previous section) NAND flash memory of not one, but *two*, floating gate charge or threshold voltage levels in the cell (¶ 26, Fig. 1.10 above). In my investigation (reviewed at ¶ 42 below), the earliest use I found of "SLC" was in 2001, and it referred to the older "single level *program* cell NAND flash memory" as distinguished from the then new "MLC" or "multi-level *program* cell NAND flash memory; the latter, as described in paragraphs 41-52 below, allowed *programming* of multiple, "lower" and "upper", logical pages of data into a single row (a single physical page) of NAND flash cells. In fact, the storage "cells" of both NAND flash memories (SLC and MLC) are the same: the floating gate transistors shown in paragraph 26 above, which may be programmed to different charge ranges. As explained by the team that developed what they called the "Multi-Level-Cell (M.L.C.) concept":

> The charge storage ability of the flash memory cell is a key to the storage of multiple bits in a single cell. <u>The flash cell is an analog storage device not a digital storage device</u>. It stores charge (quantized at a single electron) not bits. By using a controlled programming technique, it is possible to place a precise amount of charge on the floating gate. If charge can be accurately placed to one of four charge states (or ranges), then the cell can be said to store two bits. Each of the four charge states is associated with a two-bit data pattern. Figure 7

illustrates the threshold voltage distributions for a l/2Mc block for two bit per cell storage."

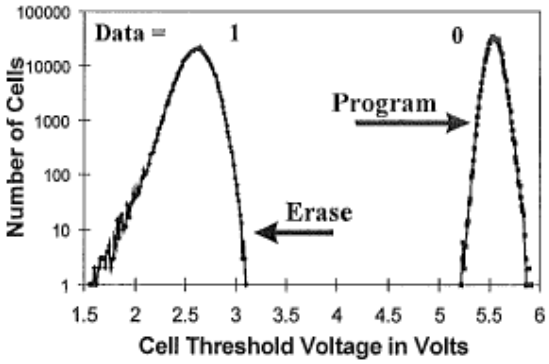


Figure 6: Single bit/cell array threshold voltage histogram

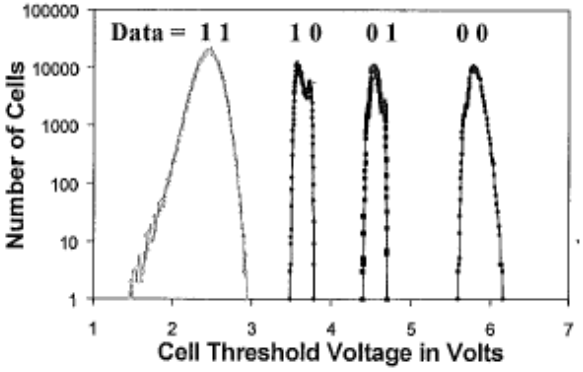


Figure 7: Two bit/cell array threshold voltage histogram

(Greg Atwood et al., Intel StrataFlash™ *Memory Technology Overview* at 4, Intel Technology J. (Q4 1997) ["Atwood," IDS#29, Ex. 1038] [emphasis added].) "*M*-level cells" allow storage of the representation of multiple (*m*) bits of information in a cell by allowing charging of the floating gate to 2^m possible threshold voltage levels, each level representing a logical *m*-bit (-place) binary number. (The representation assignment often is made by Gray code, where adjacent threshold voltage levels represent binary numbers differing by only one bit in order to minimize the number of erroneous bits if a level is read as its neighboring level. Thus, rather than the descending binary order shown in Figure 7 above, the bit-pair assignments to the threshold levels would instead be, for example, in left to right order: 11 10 00 01.)

41. The full, continuous (analog) range (window) of threshold voltages/charges possible to be programmed into (and stored in) a floating gate cell need not be divided into binary powers of two levels to represent a binary number. Following is diagram of non-binary seven memory states (seven voltage threshold levels) from U.S. Patent Appl. Pub. No. 2010/0172179 to Gorobets [Fig. 3, [0061] ("Gorobets '179," Ex. 1044):

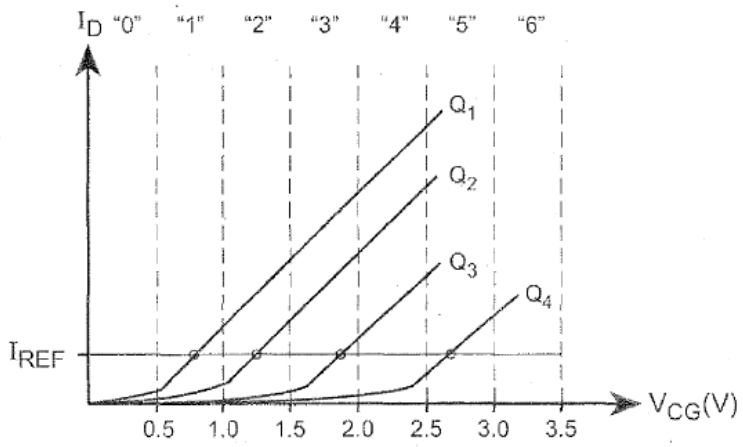


FIG. 3

The diagram also shows another important aspect of memory: how to sense the information representation using a reference current I_{REF} to sense the charge Q associated with the logical state. Memory requires not only the ability to store, but the ability to retrieve information. (See ¶ 26 above, note 6.) While multi-level storage implies a capability to represent multiple bits, that is not enough to make a "memory." In addition to storage and retrieval, NAND flash memory operation

requires the ability to erase previous read-only states to program new states in a different block. (*See* ¶¶ 27-29 above.)

42. The first reference to "SLC" I have found is in the article, Taehee Cho et al., "A 3.3 V 1 Gb multi-level NAND flash memory with non-uniform threshold voltage distribution," *2001 IEEE International Solid-State Circuits Conference. Digest of Technical Papers*, p. 1, ISSCC (Cat. No.01CH37177) (2001) ("Cho," Ex. 1039), where a Samsung Electronics team noted the difficulty of "multi-level program cell NAND flash memory (MLC)" to compete in the market with "single-level program cell NAND flash memory" (emphasis added). One of the disadvantages of MLC they noted was "[t]he 2b [MLC] NAND flash memory having a three-step programming sequence requires 3 times longer program time than that of a 1b [SLC] NAND flash memory."[14] (*Id.*) From this early reference and my

[14] To improve performance, the team experimented with a sense-and-latch page buffer that allowed a "single-bit-per-cell option transistor." (*Id.* pp. 1-2, Fig. 2.1.2.) A paper by the same team later that year – submitted by Dr. Rao with his application resulting in the '298 patent – also compared "multilevel program cell (MLC) technique" with "single-level program cell (SLC) technique"; they noted again the market disadvantage of "MLC NAND flash memories" relative to "SLC NAND flash memories" but reported advances in a "dual-mode NAND flash memory

knowledge of the history of NAND flash, I believe the term "SLC" was introduced to distinguish prior NAND flash memory from the "MLC" concept of programming. Although I have heard "MLC" applied to NOR flash memory, I don't recall the term "SLC" applied to anything other than NAND flash memory.

43. **Program level**. The 2001 reference of the Cho team to "multi-level program cell NAND flash memory (MLC)" and "single-level program cell NAND flash memory (SLC)" may have reflected the by-then accepted use of MLC NAND cells to represent multiple bits of binary data, where the bits are sourced from distinct strings (logical pages), rather than the same string. The approach abandoned an early approach taking two bits of one logical page and "encoding" (or compressing) them to be represented by one of the four threshold voltage levels of an MLC NAND flash memory cell. Instead, one bit of one logical page is programmed into the cell as one of two threshold voltages much as in traditional (now called "SLC") memories, but a second bit of a different logical page is later programmed into the cell by moving the threshold voltage previously programmed to a higher, final voltage threshold

---

having 1-Gb MLC and 512-Mb SLC modes." (Taehee Cho et al., *A Dual-Mode NAND Flash Memory: 1-Gb Multilevel and High-Performance 512-Mb Single Level Modes*, 36 IEEE J. Solid State Circuits 1700, 1700 (Nov. 2001) ["Cho Dual-Mode," IDS#11, Ex. 1068] (emphasis added).]

representing a two-bit number where one bit comes from one page and the other bit from the other page. The approach was proposed in Ken Takeuchi, *A Multipage Cell Architecture for High-Speed Programming Multilevel NAND Flash Memories*, 33 IEEE J. Solid-State Circuits 1228, 1230 Figs. 5-7 (Aug. 1998) ("Takeuchi," IDS#8, Ex. 1069) ("conventional" four-level cell[15] programs in the same operation two bits of data belonging to the same page, where "page" is "the group of memory cell that are programmed simultaneously):

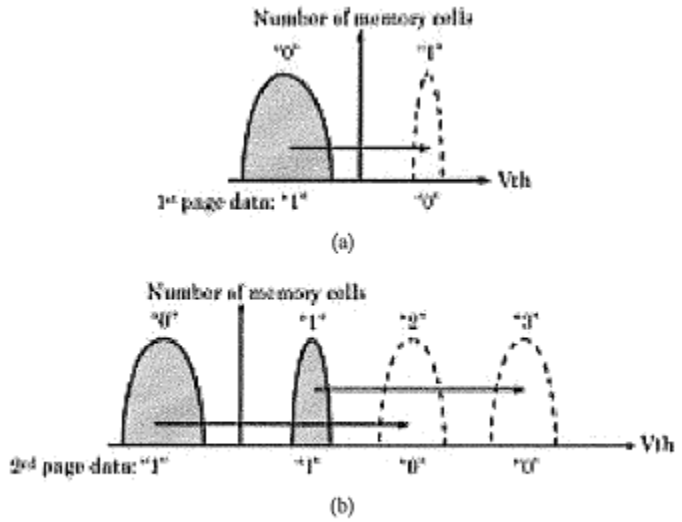


Fig. 6. Program operation of the proposed architecture. (a) First page program. (b) Second page program.

[15] Note that what was called "single-level program cell" by Cho in 2001 was at this time (1998) described as "two-level NAND cell." (*E.g.,* Takeuchi at 1229, Fig. 2, 1231, Table 1.)

The two-pass approach has the advantage of automatically encoding the bit value of the first page with relatively little modification to traditional, 2-(threshold) level NAND flash programming (such as making room for the final, 2-bit threshold level) on the first pass and the bit value of the second page on the second pass. The threshold level sensed on reading needs to be decoded. For both one- and two-pass approaches, two latches are required for such decoding (*id*. at 1230, 1232, Fig. 12) as well as for the reading necessary prior to second pass programming (*id.* at 1233, Fig. 13). The Takeuchi approach was refined by the time of the 2001 Cho articles, reflected in the Sept. 17, 2001, application resulting in U.S. Patent No. 6,456,528 to Chen for "Selective Operation of a Multi-State Non-Volatile Memory System in a Binary Mode" ("Chen '528," Ex. 1040): "Figs. 10A and 10B illustrate a specific existing technique . . . In a first programming pass, the cell's threshold is set[16] according to the bit from the <u>lower logical page</u> [the least significant bit of the two-bit representation]." (8:6-10 [emphasis added]).

---

[16] Pulse programming is used, where between the stepped up voltage pulses "verify (read) operations are carried out," where "the programmed level of each cell being programmed in parallel is read" to determine if it has reached or exceeded the verify level, and if so, it is taken off the pulse voltage line, while other cells are continued until they reach their verify levels. (7:52-8:5.)

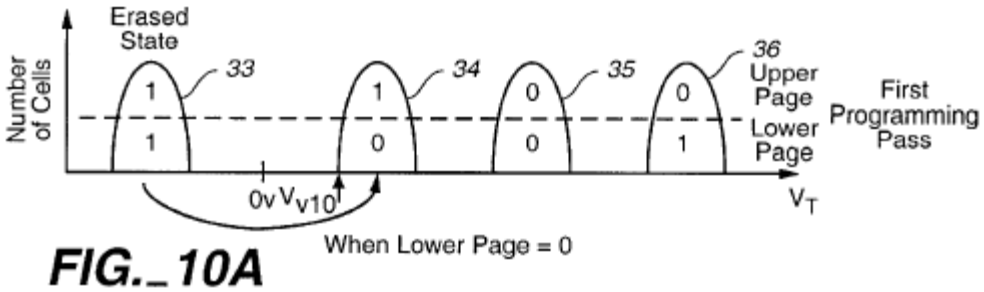


FIG._10A

"In a second programming pass, the cell's threshold level is set according to the bit being stored in the cell from the upper logical page [the most significant bit of the two-bit representation]." (Chen '528 8:15-18 [emphasis added].)

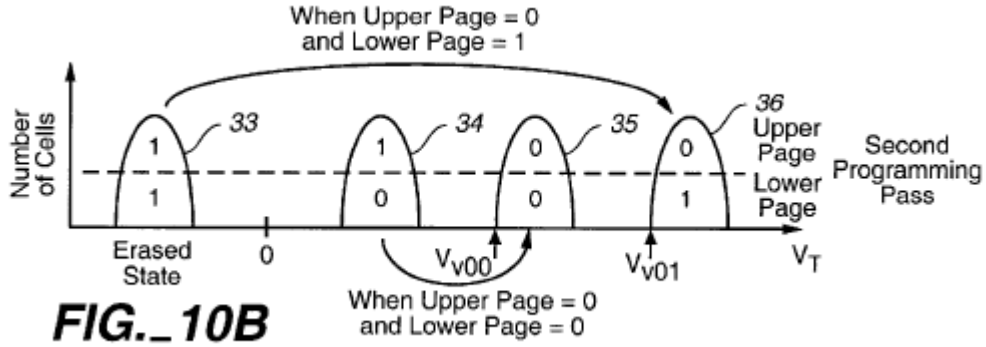


FIG._10B

The two passes result in the possible threshold levels shown in Fig. 11:[17]

---

[17] Also shown in Fig. 11 are "the voltages used to read each cell in order to determine which of the four threshold states the cell lies," including verification of the states "between repetitive programming pulses" (8:51-57) as well as reading from the memory array (8:60-65). Two registers are required for read and verification operations (6:38-7:8, Fig. 7) to decode the threshold values into 2-bit logical values.

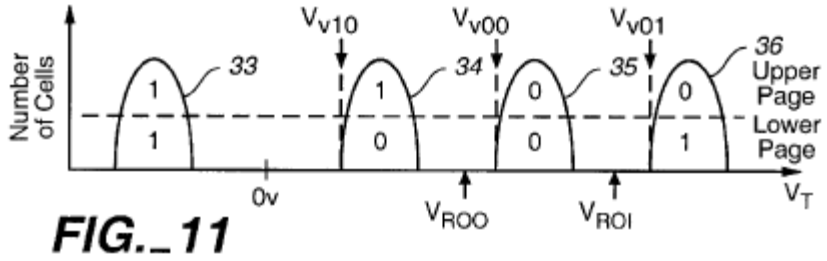


FIG._11

The lower page/bit and upper page/bit programming and threshold representation provide convenient encoding with Gray code mitigation of read errors. This characterization is consistent with the Cho reference to "multi-level program cell NAND flash memory (MLC)" and "single-level program cell NAND flash memory (SLC)," where the first allows for multiple-pass programming of multiple "levels" of different logical pages into a single physical row (page) of NAND flash cells, supported by hardware including at least a register or set of latches per level and the

---

Note the pairs of latches in Takeuchi at 1232, Fig. 12. (*Also* Micheloni at 275-77 ["Since two pieces of information must be stored, it is evident that at least two latches are needed inside the sensing circuit."], 456 ["the number of latches inside a sensing circuit is (as a minimum) equal to the number *n* of bits stored inside the cell"].) The decoding of threshold levels representing bits from different logical pages requires two read passes. (*E.g.*, U.S. Patent No. 8,634,240 to Gavens et al., published April 28, 2011 ["Gavens," Ex. 1045, Figs. 24A and 24B and associated text at 26:31-27:2].)

latter only having the hardware capable of programming only one "level" of a logical page.

44. **Endurance issues.** As explained at paragraph 26 above, through repeated stress of FN tunneling particularly in erasing but also in writing, the floating gate transistor memory cells used in NAND flash degrades at least by a narrowing of separation (window) between the threshold voltages available to represent the fully erased and fully programmed states in 2-state operation. With 2^n states needed to represent n bits of binary data from n different logical pages (in NAND flash operation since at least 2001), the separation between threshold voltages shrinks exponentially, so that various "disturbs" or loss of charge may not be tolerated and program verification and retries increase time to program until failure. (*See* ¶ 26, note 6, ¶ 35, note 10.) In one empirical study published in 2010, Eitan Yaakobi et al., *Error Characterization and Coding Schemes for Flash Memories*, p. 1 (2010 IEEE Globecom Workshops, Miami, FL) ("Yaakobi," Ex. 1032), it was remarked that "the observed lifetimes of the tested flash blocks were much longer than the lifetimes specified by the manufacturer . . . [a]n SLC block can typically tolerate 10^5 to 10^6 erasures whereas an MLC block 10^4 . . .." Raw bit error rates were presented:

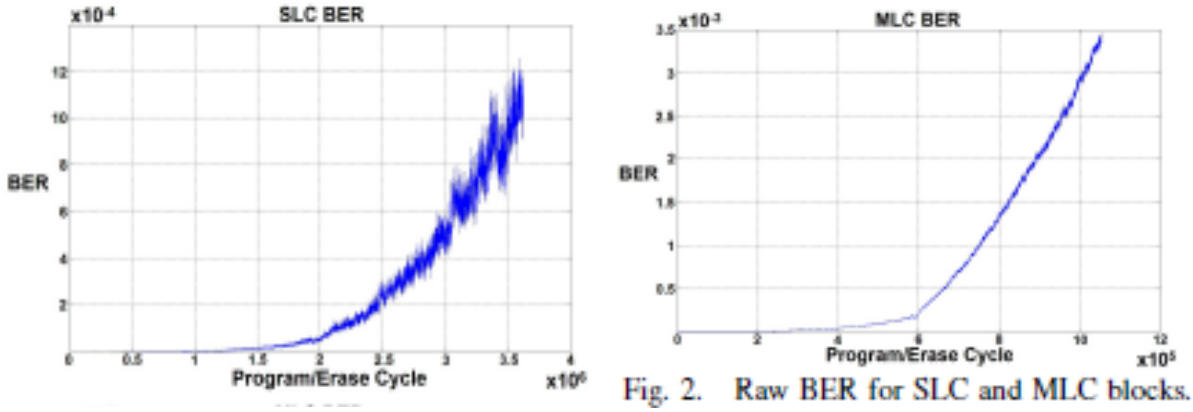


Fig. 2. Raw BER for SLC and MLC blocks.

These are factors towards the greater error probability and wear for MLCs compared to SLCs, leading to the known trade-off between shorter erase-write cycle life of MLCs but their lower cost per bit of memory.

45. By 2010, the comparative characteristics of SLCs and MLCs (often understood to be limited 2 bits/cell, compared to TLC [3 b/c] and QLC [4 b/c] were known and summarized in the following graph (Micheloni at 9):

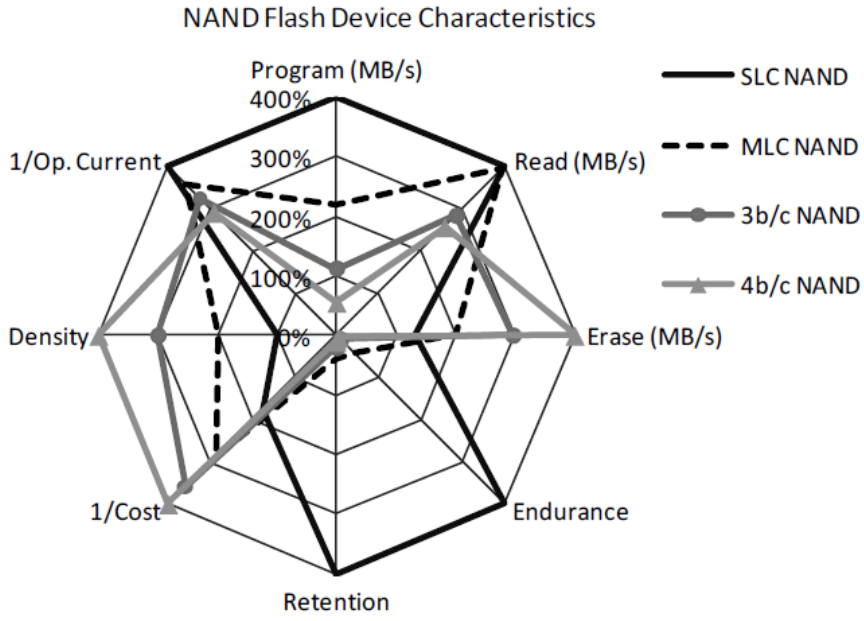


Fig. 1.10. NAND device characteristics (Source: Forward Insights)

Notably, the rated "endurance" for MLCs is an order of magnitude less than for SLCs.

C. Hybrid Use of MLC and SLC Memory

46. This reliability trade-off favored the use of MLCs for infrequent write applications such as memory cards for consumer digital cameras, an application which also tolerates bit errors. Enterprise uses demanding endurance and quality were often met with SLC NAND. (Micheloni at 169.) However, it was recognized that, for enterprise use, frequently programmed or critical data (such as chip or memory system information) might be advantageously held in an SLC memory module and less frequently programmed data (such as user data) held in an MLC system. (*See* Chris Evans, *Enterprise MLC: How flash vendors are boosting MLC write endurance*, ComputerWeekly.com (June 3, 2011) (Ex. 1033).) The hybrid SLC and MLC modules approach had been proposed by 2008. (*See generally* Nelson Duann, SLC & MLC Hybrid (Silicon Motion Flash Memory Summit, Santa Clara, CA, Aug. 12, 2008) ["**Duann**," IDS#15, Ex. 1034].) U.S. Patent No. 8,078,794 to Lee et al., published Sept. 4, 2008 ("**Lee '794**," Ex. 1041) disclosed the use of distinct "SLC flash memory array 226 [that] may include at least one flash memory chip" and "MLC flash memory array 228 [that] may include at least one MLC flash memory chip" (5:10-13, Fig. 2 [the "hybrid solid state device (SSD) 200 also includes a controller 210, a data read cache 203 and a data write cache 204, and

a logical-to-physical address mapping memory 220"], 4:11-20). Fig. 3 shows mapping of logical block address (LBA) of system files and user directories (both frequently written) to SLC physical blocks 326 and user files to MLC physical blocks 328, where the physical blocks are physically part of their respective memory arrays (chips):

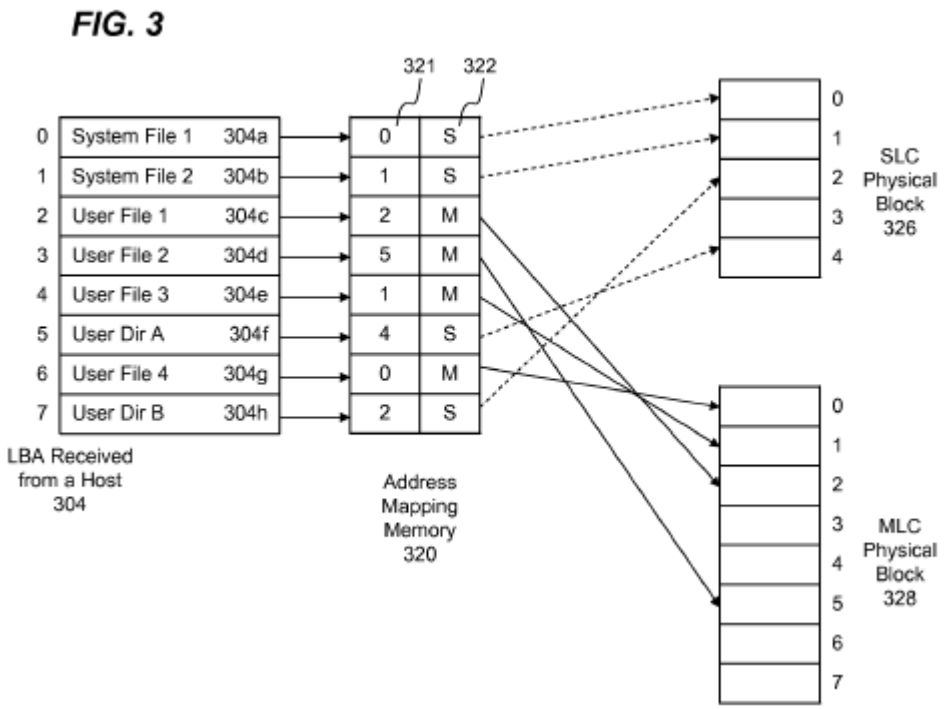


Fig. 5A shows decisions to remap system files and user directories – "hot data," expected to be updated frequently – to the SLC blocks. Fig. 5B shows an additional decision

> whether the user file has been accessed more than a threshold. The threshold may be defined as following examples: 1) the user file has been accesses [sic] more than a predefined number of times; 2) data blocks of the user file are found in the data write cache (i.e., have

> recently been accessed); 3) previously modified time of the user file indicates very recent access; or 4) an access counter of the user file shows a high number of accesses (i.e., more than a predefined threshold).

(Lee '794 6:58-66.) Thus, "hot data" determined by logical block count under thresholds 1) or 4) are sent to the SLC. This has some similarity to dynamic wear leveling by the array method favoring a physical block by its age vector (¶ 36 above, which is not as effective as static wear leveling, note 13 above) where a block in SLC likely will have much lower normalized (versus endurance) age than a block in MLC.

47. The hybrid use of SLC and MLC memory chips/modules to capture their respective advantages was also applied to other known wear-leveling techniques prior to June 19, 2011. *E.g.*, **Sutardja**, U.S. Patent Appl. Pub. No. 2008/0140918 "Hybrid Non-Volatile Sold State Memory System" ("Sutardja," Ex. 1042). The Sutardja architecture included a first solid state nonvolatile memory 204 with relatively large capacity (such as MLC with a device usable lifetime of 10K write cycles) and a second solid state nonvolatile memory 206 with lower density and greater lifetime (such as SLC with a device usable lifetime of 100K write cycles). *Id.* ¶¶ [0006], [0106]-[0108]. Multiple, detailed criteria are provided for mapping data to one or the other memory, ranging from write cycle lifetimes to ratios of use to frequencies of writes, normalizing the wear on a physical block, using a

write leveling software module and a write monitoring software module. *Id.* ¶¶ [0011]-[0083]. Detailed algorithms are disclosed for application of these criteria. *Id.* Figs. 7A-7E, 8 and explanations. The base algorithm is similar to that of Lee '794, mapping logical addresses with low write frequency (as reported by the host) to the lower endurance memory and mapping logical addresses with high write frequency to the higher endurance memory. (Fig. 7A (506) & (508).) There are optional processes (not necessarily intervening, but at an appropriate "time") that are based on actual wear to the physical blocks by (a) considering the number of writes during a predetermined time to the physical lower endurance memory and identifying the least used block in the higher endurance memory (Fig. 7C "data shift analysis"), (b) testing degradation of a physical addresses by successive writes and read backs (Fig. 7D "degradation analysis"), and (c) generating wear levels for both memories and mapping logical addresses to the other memory if the wear level of one exceeds a threshold (Fig. 7E). The data shift (¶¶ [0126]-[0127]) resembles the traditional static wear leveling of moving static data to more worn (lower residual endurance) blocks (¶ 37 above), but also optionally includes biasing mapping a logical address associated with a block in the lower endurance memory which exceeds a number of writes in a predetermined time to a block in the higher endurance memory (¶ [0128]).

48. Also, **Moshayedi** U.S. Patent Appl. No. 2009/00327591 ("Moshayedi," Ex. 1043), specifically showed four channels (banks) Fig. 1 (108) each connecting "five chips of flash (e.g., the first chip can be a K9K8G08 SLC flash, and the rest can be K9G8G08 MLC flash)" (¶ [0038]) where, similarly to Lee '794 and Sutardja in favoring sending "hot data" to SLC, the drive

> keeps track of the number of times that data for each logical block address (LBA) has been written to the flash memory, and determines whether to store newly received data associated with a particular LBA in SLC flash or in MLC flash, depending on the number of writes that have occurred for that particular LBA.

(¶ [0024].) As in Sutardja, there are other optional processes that are based on actual wear (erase count) or content of the physical blocks. In a variation of the garbage collection I reviewed at paragraph 37 above,

> . . . a set of linked lists can be employed to track the number of valid segments in each block of flash memory . . . When empty blocks are needed to write new data from the host, [physical] blocks with the least number of valid segments are selected for garbage collection, where any remaining valid segments are copied to a new block and the old block is erased.

(¶ [0029].) In a variation of the static wear leveling that I reviewed at paragraph 37 above, data presumed to be relatively static or "cold" by their presence in blocks of relatively low erase counts:

> A set of linked lists can be used for device 100 to track the relative number of times blocks have been erased based on the lowest erase count among the blocks. For example, one list may contain all blocks with erase count within 0 to 1000 of the erase count of the lowest count block, the next list may contain all blocks with erase counts that are within 1001 to 2000 of the erase count of the lowest count block, etc. When a block reaches the highest erase count list, the data contained within that block is swapped with data from a block on the lowest erase count list, which is presumed to be more static since the block containing the data has a low erase count. This wear leveling is done to keep all of the flash memory chips within a specified range of erase counts.

(¶ [0030].) The physical block erase count list may be used to move data between MLC flash and SLC flash in another variant of traditional wear leveling:

> once a block in MLC flash reaches a threshold erase count (e.g., 500), the next write operation to that block triggers a swap where the data from the MLC flash block is written to a block in SLC flash. In this manner, data for an LBA that is frequently written and causes frequent erasures is moved to SLC flash which can perform more erase cycles than MLC flash.

(¶ [0032].)

D. Use of MLC Memory in "SLC Mode" ("Pseudo-SLC")

49. Meanwhile, memory system manufacturers such as SanDisk provided memory systems that included MLC blocks that could be configured to function in

one-bit (two-state, two threshold voltage level) mode, later sometimes called "pseudo-SLC" or "pSLC," and this was disclosed as early as September 24, 2002, in the issuance of Chen '528," (Ex. 1040) for "Selective Operation of a Multi-State Non-Volatile Memory System in a Binary **Mode**" (emphasis added), where the "existing technique" of two page pass programming of MLC flash memory (¶ 43 above) where only "the first two states 33 and 36" (in Chen '528 Fig. 10B) below are programmed as shown in Fig. 12:

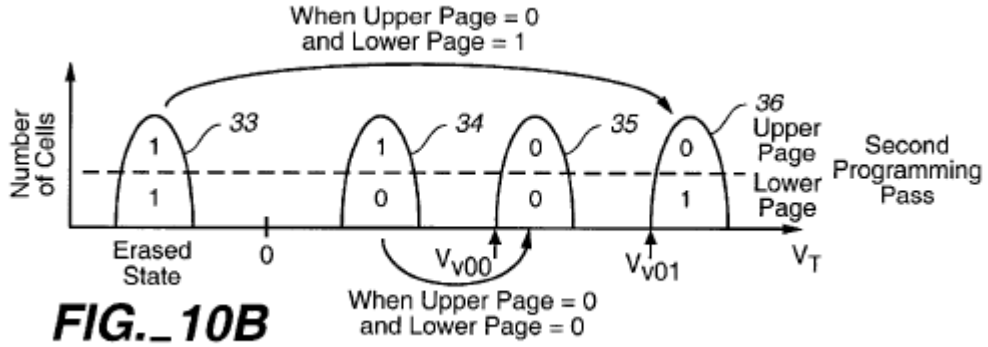


FIG._10B

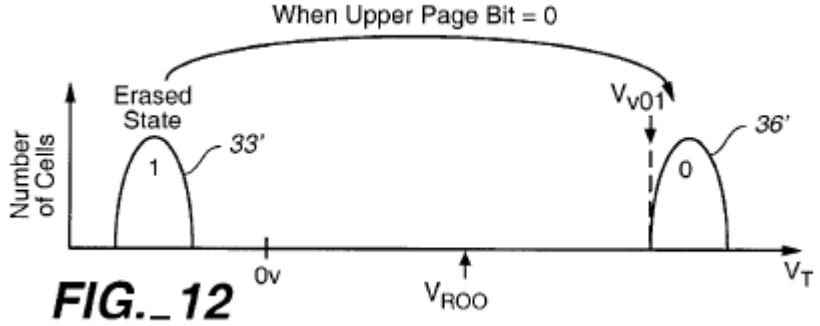


FIG._12

In essence, the lower page is assumed to be a null "1" and not programmed, and only the upper page is programmed. Nonetheless, "[t]he reference voltage V_{v01} is used to verify the programming in the same manner as the multi-state case." (9:22-24.)

Recognizing the retention advantage of the two-state mode because of the increased margin and fewer programming passes (writes, wear) (9:52-10:30), Chen suggested one use in which "memory cell blocks to which data are written most frequently are operated in two states while the remaining blocks that are rewritten less frequently are operated in multi-state" (10:41-52). Gorobets '179 (Ex. 1044) disclosed adaptive use of spare blocks in binary-operated and MLC pools and observed of binary operation (lower page/bit): "While this approach does not fully optimize the range of the threshold window as in the case of a single-level cell ('SLC') memory, it has the advantage of using the same demarcation or sensing level as in the operations of the lower bit of the MLC memory" (¶ [0083]).[18] Such configuration was disclosed in Gavens (Ex. 1045, 8:19-21 ["[a] memory chip 100 includes a memory array 200 of memory cells with each cell capable of being configured as a multi-level cell ('MLC')]") and the multiple patents it incorporated by reference. The benefits of split mode operation include similar trade-offs as between "actual" SLC and MLC modules or chips as in Lee '794, although the program cycle and processing overhead may be the same between an MLC operating in MLC mode and one

[18] Micheloni at 511 also notes that the greater distance between threshold voltage levels in an "SLC partition . . . allows speeding up program and read operations getting closer to native SLC memories."

operating in SLC mode. An early appraisal of relative (Samsung) performance is provided, where "pseudo-SLC" is "MLC_{LSB}":

Table 1: Performance comparison of different types of cell programming (us)

Operation	SLC	MLC_{LSB}	MLC_{BOTH}
Read (page)	399	409	403
Write (page)	417	431	994
Erase (block)	860	872	872

(Sungjin Lee et al., *FlexFS: A Flexible Flash File System for MLC NAND Flash Memory* (2009 USENIX Annual Technical Conference, San Diego, CA, June 14-19, 2009) [Ex. 1036].) Yaakobi, p. 5 (Ex. 1032) remarked that the 2-state MLC technique can be made "adaptive by initially using the MLC device in its native mode, and then switching to SLC mode after a certain number of iterations, when the BER [bit error rate] has become larger due to device wear." Another paper at the same conference, Ashish Jagmohan et al., *Adaptive Endurance Coding for NAND Flash* at 1841-45 (2010 IEEE Globecom Workshops, Miami, FL) (Ex. 1035), shows that 2-state operation of MLC has a BER after 20K p-e cycles an order of magnitude better than native-mode operation of MLC, but definitely not as good as SLC. This is confirmed in current commercial offerings that compare SLC as rated for 50-100K p/e cycles, pSLC for 20-30K and TLC for 3K. *E.g.*, Smart Modules Technologies, *Pseudo-SLC For Flash Storage*, https://www.smartm.com/technology/pseudo-slc-pslc (visited Nov. 12, 2024);

Cactus Technologies, *Industrial pSLC Flash Storage*, https://cactus-tech.com/products/industrial-pslc (visited Nov. 12, 2024).

50. Prior to June 19, 2011, and since, system design considerations included different deployment of SLC or pSLC versus MLC memory, such as is exemplified in Fig 3 of Segal et al. U.S. Patent Appl. Publ. No. 2011/0271043 ("Segal," Ex. 1048), for bad block replacement by adjusting MLC and SLC spare pools, including optimizing the diversion of MLC spares for SLC-mode spare usage, and similarly in Gorobets '179 (Ex. 1044) for MLC and pSLC spares.

E. "Data Integrity Checking" in NAND Flash Memory Systems

51. In addition to the routine threshold voltage programming checks mentioned above (*e.g.*, ¶ 43, notes 16 & 17) for readable writing (including for determining when to stop pulsing and for "read errors"), NAND flash systems – particularly near the end of useful life – often operated with an "integrity check" to determine whether the data written into a NAND flash page of cells (the smallest group for writing) is not only readably written, but is read (decoded) into a binary string identical to the one directed to be written. Different actions may or may not follow if bit errors (mismatches) are detected, including simply keeping the written

copy to be corrected by ECC on future reads, retrying,[19] or remapping and writing to a different location. Excessive detected errors may lead to flagging a block as a "bad block" with respect to bad block management (BBM).

52. In most commercialized NAND flash memory systems, checks of data read against the data directed to be written are performed indirectly (if at all until end-of-rated-life) as part of the processing of customized ECC schemes. (ECC schemes have long been used for automated correction of digital communications, partly because such communications may suffer intervening interruptions such as

---

[19] Ordinarily it would be unduly disruptive to erase and retry the same physical location for the same directed write. However, various actions are performed during a period between access (read/write) activities, *i.e.*, in the background. These include garbage collection, static wear leveling, "refreshes" of programmed threshold voltage levels that naturally drift over time. For MLC memories, there is the possibility of reconfiguring and using a "failed" page's block in a lower density, more tolerant mode in later writes (*e.g.,* Gorobets '179 [Ex. 1044][partition of MLC memory into blocks with different *operating modes* of higher and lower endurance; higher endurance mode "spares" configured as necessary], Segal [Ex. 1048][same, ¶ [0007]: "In MLC flash memory devices, memory blocks may be used as different type" [emphasis added].)

radio frequency interference. Such interference also occurs in NAND flash memory due to the effects of writing into adjacent cells, more vulnerable with more closely spaced threshold voltage levels from higher density use and wear.) A limited number of bad cells (those that cannot be read properly) in a page can be tolerated if ECC is used, thereby avoiding the stresses of erasing the block and writing to another, bringing both closer to rated "end of life." In a typical NAND flash memory system, an ECC is generated for the data to be written, which may be stored with that data as written (as shown) or in a separate memory associated with that data. When the written data is read, a new ECC may be generated based on the read data and compared to the stored ECC (including embedded "EDC" or error detection code) to detect and correct erroneous data bits. Following is a diagram of an on-chip ECC from Brewer (Ex. 1026) at 88 showing the writing of user data D into the memory with generation by an ECC encoder of ECC correction bits C, stored in the memory, and when the written data WD is read with the original C, a new code S is generated based on the read data WD to indicate which bits of WD are different from

D to correct those erroneous bits to output D:

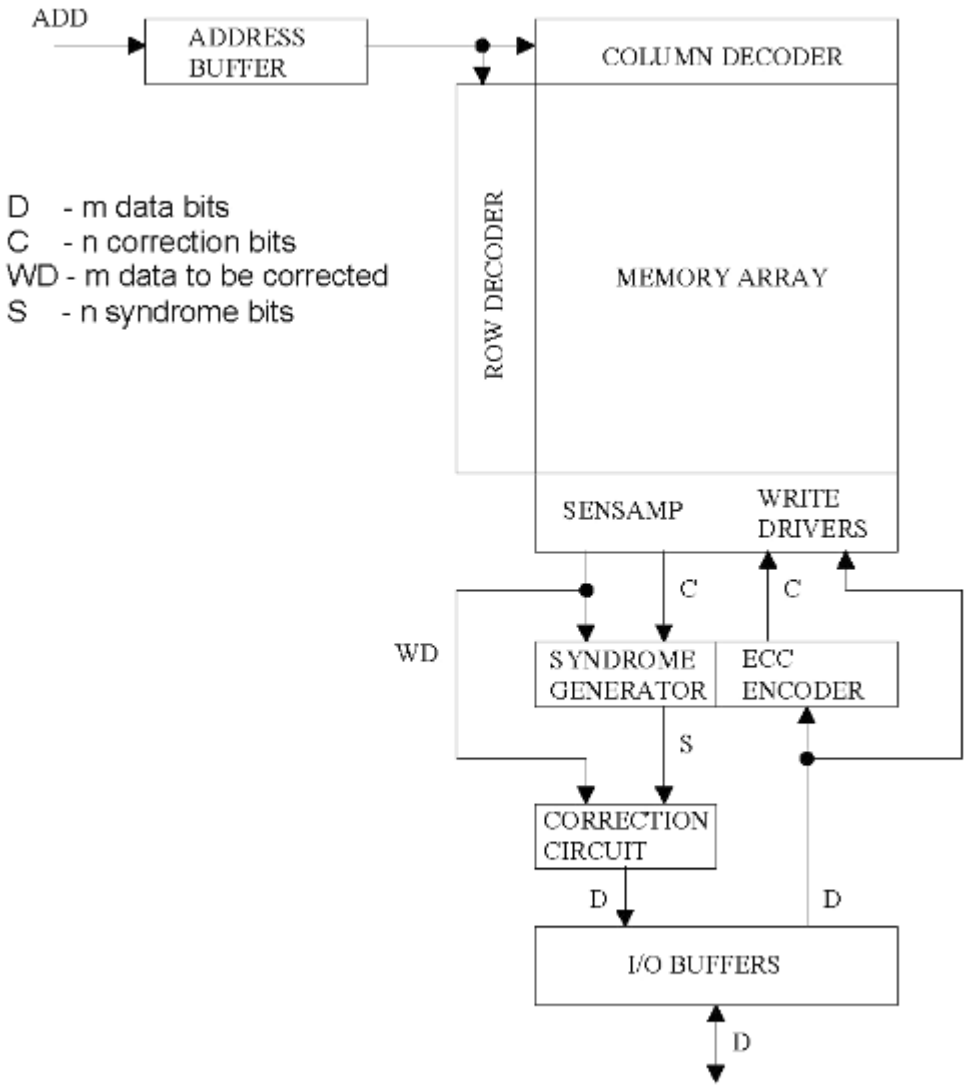


Figure 3.20. On-chip ECC for Flash memory.

The ECC processor may instead be in the device controller (¶ 31 above, showing Micheloni Fig. 2.33), and with auxiliary memory (Micheloni at 508, Fig. 17.21 [used to store code bits]). With enough processing resources and ECC space needed (which may be external to the chip) to correct multiple error bits, nothing needs to be done with the data (WD) with the error bits in the memory system, as what is read

from that data will be corrected to what was received to be written (D). However, as a flash memory ages, the number of required error correction bits for a block may exceed the storage available for error correction bits and that block will have to be retired. As more and more blocks are retired, at some point the flash memory is no longer useful.

53. Gavens (Ex. 1045), published April 28, 2011, is one of a portfolio of SanDisk patents, many incorporated by reference in the context of a NAND flash memory system including "[a] memory chip 100 [that] includes a memory array 200 of memory cells with each cell capable of being configured as a multi-level cell." (8:18-20.) Gavens discloses the use of such ECC similarly to that shown in the preceding paragraph:

> In the types of memory systems described herein . . . the integrity of the data being stored is maintained by use of an error correction technique. Most commonly, an error correction code (ECC) is calculated for each sector or other unit of data that is being stored at one time, and that ECC is stored along with the data. The ECC is most commonly stored together with a unit group of user data from which the ECC has been calculated. The unit group of user data may be a sector or a multi-sector page. When this data is read from the memory, the ECC is used to determine the integrity of the user data being read. Erroneous bits of data within the unit group of data can often be corrected by use of the ECC. (3:27-39.)

> As data is received from a host, a page of data is staged in the controller 102 and its ECC 76' is computed by the ECC processor 62. The data page incorporating the ECC is then written to the memory array 200. Typically, when the data page is read, the data page is latched in the data latches 430 and shifted out of the I/O circuits 440 to the controller 102. At the controller 102, the data page's existing ECC is compared to a second version of the ECC computed on the read data. The ECC typically includes an error detection code ("EDC") for rapid detection of any error in the data page. If the EDC indicates the existence of any error in the read data page, the ECC is invoked to correct erroneous bits in the read data page. (12:64-13:8; *also* 29:33-62.)

This is the preferred approach for checking and correcting for errors (without the need to repeat the process of programming (and erasing before the next programming). However, although "[t]he ECC can be designed to correct any number of bits[, t]he more bits it has to correct, the more complex and computationally intensive will the ECC be."[20] (13:9-11.) Thus, in a section entitled

---

[20] "[F]or most of the lifetime of the device, the ECC is only marginally utilized, resulting in its large overheads being wasted and realizing no real benefits." (Gavens 3:67-4:2.) Gavens explores the design of ECC "based on the expected worse case cell error rate ('CER') at the end of life ('EOL') of the memory device," based on statistical likelihood of errors among a given population of cells. (13:11-35, Figs.

"Enhanced Post-Write-Read Error Management" ("EPWR," 19:37-24:33, Figs. 18-21), Gavens discloses the delay of application of error management to a device (including post-write-read[21]) until after a "hot count" of "the number of times [an] erase block has been cycled through erase and program operations" exceeds a predetermined threshold (5:31-6:2, 19:38-20:29, Fig. 18) and an embodiment in which data is first "staged" in (3 pages) of a "less error prone, low density [1-bit] storage portion of the memory (D1)" and "subsequently folded into [one physical page of high density, 3-bit] (D3)" (20:30-35). When the EPWR is enabled (Fig. 22A [820], Fig. 22B [820]),

> . . . a current filled block in D3 is read back; and if the error rate exceeds a predetermined threshold, the current D3 block is rejected and a retry takes place with data being refolded into a new D3 block. The new D3

10A and 10B.) It reviews the sources of post-write errors for a sample memory device (13:40-15:17, Fig. 11) with number of "program-erase cycles" being the major factor and number of reads (disturbs correctable by "read scrubs") a much smaller factor, and concluding that, near EOL (10K p-e cycles), "the ECC must be capable of correcting at least 21 error bits" (15:18-58, Fig. 12).

[21] This is distinguished from write verification, ordinary read (retrieval of data, which may apply ECC), periodic refresh or "read scrubs" and other background processes such as wear-leveling and garbage collection.

> block is again read back and checked for excessive error rate. If the new D3 block passes, then it has good data and the original data in D1 is made obsolete. If the new D3 block again shows excessive error rate, the new D3 block is again discarded. If the excessive error rate persists after a predetermined number of retries, no further retry is attempted and the D1 to D3 folding operation is abandoned with the original data kept at D1. At this point the memory device is deemed too old for further programming operations and is made read-only to preserve the integrity of the existing data stored in the memory device.

(20:36-50; details and variations at Figs.20A-20C and explaining text at 19:51-23:29.) Optional process 864-865 retires a D3 block if it has been retried more than a pre-determined maximum number of retries (Fig. 22C explained at 24:11-25), serving bad block management.

54. While this embodiment describes the use of ECC to indirectly determine the number of bit errors, Gavens explicitly does not so limit its embodiments. Gavens also discloses alternatives of direct comparison of the data written into memory (and then read back in "post-write read") with a cached copy of what was to be written, among various embodiments under the rubric "Adaptively Rewrite Data from a Higher Density Memory Portion to a Lower Error Rate Memory Potion to Control Error Rate" (15:59-61). In one group of adaptive rewrite embodiments (Figs. 14 and 15 and associated text at 16:13-17:42), user data is written to the higher density (*e.g.*, MLC) portion as a "first copy." (16:41-45.) Then,

in a "post write read," there is a check for errors "either by comparison with the **original copy**[22] which may be cached **or** by checking the EDC portion of the ECC" (16:46-49 [emphasis added]). If the number of error bits is less than a predetermined amount, the first copy is deemed valid and subsequent reads of the data page will be from that copy with any errors corrected by ECC. (16:50-56.) If the number is greater, then a copy is written to the lower density, lower error rate (more robust, *e.g.*, "SLC") portion; this is performed in one embodiment by copying from a cached copy ("of the original data") or in another embodiment from application of ECC to correct error bits in the first copy. (17:9-14.)

55. In a second group of adaptive rewrite embodiments (Figs. 16 and 17 and associated text at 17:43-19:24), the more robust ("SLC") portion is further divided into a first section for caching incoming data (simultaneously with a "first copy" written into the higher density "MLC" portion) and a second section for storing rewrites from the higher density MLC portion. (18:21-29.) In one of these embodiments,

---

[22] Obviously, this "original copy" is upstream of the "first copy" referenced in the preceding paragraph, likely off the memory chip in some cache from which the device controller directs data to be written to the memory chip as the "first copy" in its higher density portion.

> the memory array is provided with a set of data latches on an integrated circuit chip, the checking of the error bits in the first copy is accomplished by loading the first copy and the cached copy into the set of data latches and making a comparison at the set of data latches.

(18:30-35 [emphasis added]; 18:40-44 [details].) If the number of error bits is less than a predetermined amount, the first copy is deemed valid and subsequent reads of the data page will be from that copy with any errors corrected by ECC. (18:45-52.) If the number is greater, then a copy is rewritten from the cached copy to the rewrite section of the more robust SLC portion. (18:53-66.) This then shows that when the number of errors in a copy written in MLC (for example) exceeds a certain amount, then that data is moved (rewritten) to SLC.

56. U.S. Patent No. 8,806,301 to Yu, published April 14, 2011 ("Yu '301," Ex. 1046) uses an error checking and correcting unit 210 to determine the number of error bits in the data read after writing:[23]

---

[23] Yu '301 discloses an "exemplary embodiment" of "a multi-level cell (MLC) NAND flash memory chip" as the "flash memory chip 106," but notes that another embodiment would have that flash memory chip as the distinct "single-level cell (SLC) flash memory." (7:5-9.) For each of those distinct memories, not disclosed to be used simultaneously, if the ECC unit (which may be on-chip, Fig. 6 [620]) finds an error bit and the number of error bits measured exceeds either of two

> . . . when the memory management unit 204 receives a host write command from the host system 1000, the error checking and correcting unit 210 generates an error checking and correcting code (ECC code) corresponding to write dat[a] of the host write command, and the memory management unit 204 writes the write data and the ECC code into the flash memory chip 106. And, when the memory management unit 204 reads data from the flash memory chip 106, the memory management unit 204 simultaneously reads the ECC code corresponding the read data, and the error checking and correcting unit 210 performs the error checking and correcting process to the read data based on the ECC code corresponding to the read data. In the present exemplary embodiment, the maximum number of error bits which can be corrected by the error checking and correcting unit 210 is 26. . . .

(6:15-29.) However, "in another exemplary embodiment of the present invention, the memory management unit 204 may compare the read data and the writing data from the host system 1000 <u>bit by bit</u> to determine whether the read data has any error bit and the number of error bits thereof." (11:49-53 [emphasis added]).

---

thresholds (maximum correctable bits, Fig. 5 [S509] and Fig. 7 [S709] [on-chip ECC responding to confirm write command], and pre-determined threshold level, Fig. 5 [S517] and Fig. 7 [S717]), the physical block is marked a bad block (Fig. 5 [S521] and Fig. 7 [S721]).

57. A bit-by-bit comparison of a page of user data to be written (held in some temporary memory, or cache) with the data read from the memory cells where it is written requires more storage and processing than typical ECC schemes using a controller. The ECC schemes do not require the storage of the "page of user data to be written," and the number of bits used for ECC is a tiny fraction of the bits in a whole page of data. The ECC bits are generated and stored upon writing; then upon reading a page from the memory the ECC bits are also read and used to ascertain whether or not there are errors, and to determine which bits (if any) are erroneous. While there are the alternative bit-by-bit embodiments recited in Gavens and Yu – which necessarily (inherently) would require adequate memory (typically volatile) available to the controller to hold a to-be-written page to compare with a written-and-read page – I am unaware of any commercial use of bit-by-bit "comparison" for read-after-write "integrity checking" for flash memory management. In my view, it would be an inefficient choice. Keeping in mind that a page of data is on the order of 8kB or 64k bits, it is a daunting task to compare two 64k-bit data items bit-by-bit. Typically, a subtractor is used to do the comparison, and this takes place using an adder, since subtracting two vectors (A – B) is equivalent to adding A to the complement of B. Now the largest adders handle about 64 bits (8 bytes). That means the adder would have to be applied 1000 times! Or you would need 1000 such adders to accomplish the task in the time it takes for one 64-bit adder to execute. A

combination of the two approaches (serial and parallel) could also be attempted. It is also necessary to accumulate the total number of errors, so each 64-bit addition result must be accumulated (all 64-bit adder results must be added together) in a memory or register. Overall, this takes an enormous amount of resources and time, and a POSITA would not expect this method to be used in industry, even though it is theoretically a possible approach.

F. Strategies for NAND Flash Memory Remapping To Manage System Reliability and Efficiency

58. Prior to July 19, 2011, there were many related strategies for avoiding enough memory components of a NAND flash memory system reaching the end of their lives before other components thereby degrading whole system reliability and its effective "lifetime." An obvious general strategy as discussed at paragraph 35 and 39 above, is to avoid overstressing components with lower endurance by sending data likely to stress because of frequent updating to components with higher endurance. As reviewed at paragraphs 46 to 48 above, Lee '794 (Ex. 1041), Sutardja (Ex. 1042) and Moshayedi (Ex. 1043), each applied the strategy of sending to SLC memory data associated with logical addresses likely to be frequently updated. (*Also*

Duann [Ex. 1034] at 10-12.)[24] Another strategy is to make sure that all blocks in the system reach their end-of-life at the same time by maintaining uniform wear. With a system using a homogenous type of memory, traditional static wear leveling, moving apparently "cold data" to more worn blocks, has been found to be most effective. (Note 13 above.) Systems with memories of different endurances present complexities, but there are many solutions proposed in the prior art.

59. U.S. Patent Appl. Pub. No. 2010/0115192 to Lee ("Lee '192," Ex. 1047, Samsung assignee) disclosed wear leveling in a memory system with both SLC and MLC blocks (Fig. 2), such that if the sum of "SLC mode usage" plus "MLC mode usage" minus an average usage of all blocks is greater than a threshold set for the mix of applications for the SLC (faster and more frequent writes) and MLC blocks, then wear leveling is performed (Fig. 4), in which the mapping of the LBA is changed to a new physical location, defining a new logical/physical memory block relationship (¶ [0023]). Duann taught the use of global wear leveling, treating "all

---

[24] This strategy may be analogized to the array method of dynamic wear leveling explained at paragraph 36 above, in which each new write is sent to the least worn block of an available (erased) block list, where SLC blocks are the least worn, but only data expected to be updated frequently is allocated to SLC which is more expensive per bit of storage than multi-level memory.

NAND components as one memory unit" (Ex. 1034 at 5, 9).

60. As reviewed at paragraphs 47 and 48 above, Sutardja (Ex. 1042) and Moshayedi (Ex. 1043), disclosed embodiments for preferential writing into SLC or MLC modules that also considered actual wear on memory blocks as a predictor of future wear. As explained in an article contemporaneous with the filing of the application leading to the '298 patent, Minyoung Sung & Kanghee Kim, *Asymmetric Flash Volume Management*, 58 *IEEE Transactions on Consumer Electronics* 455 (May 2012) (Ex. 1037), the asymmetry between SLC and MLC is not just endurance, but in the organization of page and block capacity to be considered relative to user data.[25] Thus, an aspect of preferential writing is "to maximize the overall read/write performance of the logical volume by (1) hot data identification and (2) periodic data

---

[25] At the time of the '298 patent application, a marketable trade-off between cost, performance, and endurance was a hybrid use of SLC to MLC modules in the proportion of 1:4 (*E.g.*, Duann at 14 ["72 GB hybrid SSD (8GB SLC + 64GB MLC)"]; Moshayedi Fig. 1 [four channels (banks) 108 each connecting "five chips of flash (e.g., the first chip can be a K9K8G08 SLC flash, and the rest can be K9G8G08 MLC flash)" {¶ [0038]}{same as cited in Duann at 14}].) These proportions of SLC and MLC flash memory is consistent with Fig. 4 of the '298 patent and its suggested proportion (4:61-63).

migration between SLC and MLC [with the objective of providing] the highest overall read/write throughput for the next period." *Id*. In one implementation, "write operations from a process with a high storage update frequency are marked 'hot' before forwarded to the flash storage system." (*Id.* at 456; *compare* Lee '794.)

61. In addition to these strategies of preferential mapping and wear leveling, at least another group of tools was available to avoid unnecessarily early failure of a NAND flash system: error management by remapping (and writing) to binary memory when the bit-errors in writes to one multi-level block or successive writes to multi-level blocks reach predetermined thresholds. (*E.g.*, Gorobets '179 [Ex. 1044], Segal [Ex. 1048], Gavens [Ex. 1046, *see* ¶¶ 53-55 above].)

62. These strategies and knowledge of NAND flash systems and operations prior to July 19, 2011, make obvious the base claim 1 of the '298 patent.

VI. Person of Ordinary Skill in the Art (POSITA)

63. I am informed by counsel that a "person of ordinary skill in the art at the time of the inventions" ("POSITA") is a hypothetical person who is presumed to be familiar with the relevant field and its literature at the time of the inventions. This hypothetical person would also be a person of ordinary creativity, capable of understanding the scientific principles applicable to the relevant field.

64. In view of this instruction, my experience with the issues, and the '298 specification, in my opinion, a person of ordinary skill in the art at the time of the

claimed inventions would have a bachelor's degree in computer engineering, electrical engineering, computer science, or a closely related field, along with at least two years of experience in the design, development, implementation, or management of memory devices and systems. A person with an advanced degree in a relevant field, such as computer or electrical engineering, would require less experience in the development and use of memory devices and systems. As is common, one could obtain equivalent knowledge and perspective from other life experiences as well.

VII. Disclosure of the '298 Patent Specification

65. Having read many patents, the specification of '298 patent, which incorporates Rao '916 struck me as disclosing nothing that might be even considered to be new, and included only imprecise or inaccurate description of two variants of well-known, optional NAND flash operations. The purportedly inventive memory system was disclosed as well-known MLC NAND flash memory modules and well-known, distinct SLC NAND flash memory modules in the known useful proportion (note 25 above), with the well-known superior endurance of the latter over the former. A third component of the disclosed memory system, "coupled" to the other two, is a "controller" that is "adapted" to perform four functions recited in the claims, the first being L2P/FTL mapping that is known and essential for any NAND flash memory. The second function is an abbreviated recitation of a flowchart (Figs. 3a and 3b) for application of a "data integrity test," failure of which results in remapping

to the SLC module. (This is a variant, if not an instance, of well-known NAND flash error management operations reviewed above.) The third and fourth functions are restatements of two sentences in the specification that describe what may be considered dynamic *or* static wear leveling. As I review in the next section, there is no more. There is no structure disclosed other than by implicit assumption of known structures and necessary and common operations of MLC and SLC NAND flash memory modules. There is nothing to show how the controller is adapted to perform the recited functions, only a reliance on known NAND flash memory systems, admitted in the specification to have been operated using L2P/FTL mapping and forms of wear leveling.

A. '298 Specification BACKGROUND OF THE DISCLOSURE

66. The specification of the '298 patent, admits a "BACKGROUND OF THE DISCLOSURE" state of the art that includes bits of relevant NAND flash background that I reviewed in Section V above, particularly that it "is well known in the art, any 'write' or 'program' to a block in NAND flash (floating gate) requires an 'erase' of a block before 'write.'" (Ex. 1001 3:27-30). The concepts and vocabulary of the disclosure is established, starting with the history of HDD organization into 512-byte sectors having a unique "physical (memory) address" where sectors are combined to from a "logical block" having a unique "physical address." (2:10-15.)

> . . . A computer [operating system] organizes **data** as "files." Each file may be **located (stored)** in either a single **logical block** or a plurality of logical blocks, and therefore [by mapping], the location typically traverses the boundaries of individual (physical) sectors.

(2:21-25 [emphasis added].) It was recognized that generally NAND flash memory is organized in blocks of 32 to 64 pages, each with 2-4 Kbits (2:31-3). The erase-before-write limitation of NAND flash is described:

> the process of writing data to a NAND flash memory device is complicated by the fact that, during normal operation of, for example, **single-level storage (SLC)**, erased bits (usually all bits in a block with the value of `1`) **can only be changed** to the opposite state (usually `0`) **once before the entire block must be erased**. Blocks can only be erased in their entirety, and, when erased, are usually written to `1` bits. However, if an erased block is already there, and if the addresses (block, page, etc.) are allowed, data can be written immediately; if not, a block has to be erased before it can be written to

(2:38-48 [emphasis added].) L2P mapping is also described:

> **FTL** is the driver that works in conjunction with an existing operating system (or, in some embedded applications, as the operating system) to make linear flash memory appear to the system like a disk drive, i.e., it emulates a HDD. This is achieved by creating **"virtual" small blocks of data**, or sectors, **out of flash's large erase blocks** and managing data on the flash so that it appears to be "write in place" when in fact it is

> being stored in different locations in the flash. FTL further manages the flash so that there are clean/erased places to store data.
>
> Given the limited number of **writes** that individual blocks within flash devices can tolerate, **wear leveling** algorithms are used within the flash devices (as firmware commonly known as **FTL** or managed by a controller) to attempt to ensure that **"hot" blocks**, i.e., blocks that are frequently written, are **not rendered unusable much faster than other blocks**. This task is usually performed within a flash translation layer. In most cases, the controller maintains a lookup table to **translate** the **memory array physical block address (PBA)** to the **logical block address (LBA) used by the host system**. The controller's wear-leveling algorithm determines which physical block to use each time data is programmed, eliminating the relevance of the physical location of data and enabling data to be stored anywhere within the memory array and **thus prolonging the service life** of the flash memory. Depending on the wear-leveling method used, the controller typically either writes to the available **erased block** with the lowest erase count (**dynamic wear leveling**); **or** it selects an available target block with the lowest overall erase count, erases the block if necessary, writes new data to the block, and ensures that **blocks of static data** are **moved** when their block erase count is **below** a certain threshold (**static wear leveling**).

(2:38-48 [emphasis added].) These actions are not stated to be sequential or relating to the same data or block; the static wear leveling can be done in the background. The description of dynamic wear leveling is consistent with the "first level"

(dynamic) wear leveling using the "array method" (tracking "age"), which I described in paragraph 36 above. The description of static wear leveling is consistent with the "second level" (static) wear leveling described in the same paragraph as a "move" of "static data" triggered on the condition of a difference between maximum and minimum p-e cycles among blocks reaching a predetermined threshold. The specification's threshold would not make sense for any particular physical block, since they each have increasing p-e cycle counts. It is considered beneficial to move static ("cold") data in a physical block less worn than average to one more worn than average where the data, *presumed* to be mapped to a logical address (logical data or block) not often updated, and hopefully can stay in the more worn physical block without being updated. (An update would result in further wear on erasure in garbage collection.)

67. The BACKGROUND states that the prior art wear leveling of the preceding paragraph, by "determin[ing] which physical block to use each time data is programmed" to be written, and thus "eliminating the relevance of the physical location of the data and enabling data to be stored anywhere within the memory array," thereby – without further explanation – "prolong[s] the service life of the

flash memory." (3:1-6.)[26] The BACKGROUND states that "various embodiments of a NAND flash storage system that provides long life time (endurance) storage at low cost are described herein." (3:43-45.) But nowhere is such long lifetime at low cost explained, other than by implication from the background establishment that

---

[26] For general wear leveling, there is stated the objective that "hot" blocks not to be "rendered unusable much faster than other blocks" (2:62-64). However, without other explanation, the specification recites a need for "writing data optimally for improved life time (endurance) of flash memory" (3:41-43), embodiments of NAND flash "that provides long lifetime (endurance) storage" (3:44-45), the use of SLC "to boost the lifetime (endurance) of the storage system (4:58-59), and of a system and method of "increasing the reliability and lifetime of a NAND flash storage system, module, or chip" (6:54-55). I understand from my experience and the literature, and believe that a POSITA in computer memory would understand, that the improvement sought in the field and referenced in the '298 patent specification is to avoid premature failure (actual or rated, such as the "hot blocks" at 2:62-64) of enough memory components to result in degradation of a hybrid SLC-MLC NAND flash system making it no longer reliable for its intended use. This "improvement" is relative to a hybrid NAND system configured for projected uses (such as enterprise versus consumer) balanced with cost and performance.

"MLC NAND flash SSDs are slowly replacing and/or coexisting with SLC NAND flash" (3:14-15) in 2011, where the "MLC enjoys greater density than SLC NAND flash [implying MLC's lower cost per memory cell], at the cost of a decrease in access speed and lifetime (endurance)" (3:19-22). It is implied from juxtaposition of the background's stated endurance of SLC NAND flash as 50-100K writes and of MLC NAND flash as 3-5K writes (3:22-27[27]; *see* ¶¶ 44, 49 above). Juxtaposed against a summary of remapping to the SLC module after failure of the '298 patent data integrity test, it is stated, without explanation, that "[a]s the SLC NAND flash is used to boost the lifetime (endurance) of the storage system, it can be considerably lesser in amount than the MLC NAND flash" (4:58-61). But if lifetime were the only consideration, its extension (or rather, avoidance of premature ending/failure) would be boosted by a *greater* amount of SLC NAND flash. It is suggested, without explanation, that SLC memory storage space be 1:8 to 1:4 of MLC memory storage space (4:61-63), which is reflected in Fig. 4 (2+ SLC memory modules [62a and 62b] to 8+ MLC memory modules [60a and 60b]) and conforming with Duann (one

---

[27] The text also explains that writing to a floating gate NAND flash block requires prior erasure (3:27-29), without explaining that it would be disruptive to erase immediately before write, an important consideration in performance and endurance management (*e.g.* ¶¶ 36-39 above).

of the IDS documents submitted with the '298 patent application) at 14 ([“72 GB hybrid SSD (8GB SLC + 64GB MLC)”). These ratios, relevant to the purported balance of cost, performance, and endurance, were not recited in the claims to limit the system to a configuration of useful balance of cost and lifetime. Instead, what was disclosed were methods of using a system of at least one MLC module and at least one SLC module, claimed as general functional configurations of the otherwise unlimited, but still abstract controller.

B. Proffered Embodiments in the DETAILED DESCRIPTION OF THE ILLUSTRATED EMBODIMENT

68. The only embodiment described in the '298 patent specification “DETAILED DESCRIPTION OF THE ILLUSTRATED EMBODIMENT” is a system of known SLC memory modules, known MLC memory modules (their relative expected endurances given in the BACKGROUND at 2:7-9 and 3:17-27) and an abstract controller *not* limited to a known hardware or software module.[28]

---

[28] The specification states that the controller *may* be a “chipset” (4:67), but does not otherwise disclose the controller's structure or how it is adapted to perform the claimed functions. Rao '916 (Ex. 1004), incorporated by reference (Ex. 1006, filed application at ¶ [001], in a prior claim to novel operation of existing NAND flash memory (particularly volatile memory in NAND flash modules), at least provided

Instead, the specification and claims attempt to cast the claimed system (rather than its operation) new and unobvious by functionally defining the abstract controller as performing (1) admitted, ordinary, necessary L2P/FTL operations (5:16-34, 6:36-52), (2) a two-sentence variant of admitted, ordinary wear leveling (6:24-35), and (3) a facially straightforward "data integrity test" with remapping to the SLC module upon failure, shown by Figs. 3a and 3b and a two-paragraph explanation of that "method for utilizing a NAND flash memory system" (5:55-6:23). The "embodiment" of replacement of "SLC flash" (4:10-11) with alternative technology with greater endurance at least because they "allow data to be written over existing data" (2:1-9) is claimed without other support such as any modifications needed to the NAND flash memory or controller structure.

69. **SLC and MLC modules' physical and operational foundation:** The proffered innovative system is stated in the "DETAILED DESCRIPTION" as

---

(1) a functional block diagram of a "monolithic" (single chip) controller used with NVMS flash memory (Fig. 3, compare with Micheloni Fig. 2.33 reproduced at ¶ 31 above) and (2) a circuit block diagram of a commercial "NVMS Flash Memory" (Fig. 6, compare with Micheloni Fig. 15.1 reproduced at ¶ 29 above). Nothing like this was provided in the '298 specification.

the combination of POSITA-understood physical SLC and MLC NAND flash memory modules and an abstract controller:

> Generally, and in particular regard to NAND flash memory, two separate banks of NAND flash are maintained by a controller. One bank contains economical MLC NAND flash, while a second bank contains high endurance SLC NAND flash. The controller conducts a data integrity test after every write. If a particular address range fails a data integrity test, the address range is remapped from MLC NAND flash to SLC NAND flash.

(4:51-58.) The only alternative embodiment is shown in Fig. 4 and described as

> two NAND flash memory banks **56** and **58**, whereby memory bank **56** comprises a plurality of MLC NAND flash memory <u>modules</u> **60***a* and a plurality of SLC NAND flash memory <u>modules</u> **62***a*. Memory bank **58** comprises a plurality of MLC NAND flash memory <u>modules</u> **60***b* and a plurality of SLC NAND flash memory <u>modules</u> **62***b*.

(6:40-46 [emphasis added].)[29] In language reflecting the first, '298 patent claim 1 (7:8-8:9), the "SUMMARY" states,

---

[29] The term "module" is used throughout the specification, including in the context of alternative physical items: "NAND flash storage system, module <u>or</u> chip" (1:28-29, 6:55 [emphasis added], *also* Fig. 7 in Rao '916 (Ex. 1004) 1:31-32 ["IC's mounted on a memory card or module").

> there is provided a system for storing data which comprises at least one MLC nonvolatile memory module (hereinafter referred to as "MLC module") and at least one SLC non-volatile memory module (hereinafter referred to as "SLC module"), each module comprises a plurality of individually erasable blocks. The data storage system according to one embodiment of the present disclosure further comprises a controller for controlling both the at least one MLC module and the at least one SLC module. In particular, the controller maintains an address map comprising a list of individual logical address ranges each of which maps to a similar range of physical addresses within either the at least one MLC module or the at least one SLC module. After each write to (flash) memory, the controller conducts a data integrity check to ensure that the data was written correctly. When the data was not written correctly, the controller modifies the table so that the range of addresses on which the write failed is remapped to the next available range of physical addresses within the at least one SLC module.

(3:59-4:10 [emphasis added].) A POSITA would understand "module" in this context to be a NAND flash memory chip or package such as the labeled "NAND" or "FLASH" in the block diagram (Micheloni Fig. 15.1) reproduced at paragraph 29 above and the photograph (Micheloni Fig. 1.16) reproduced at paragraph 27 above

and in Figs. 6a and 7 of the incorporated Rao '916[30] (Ex. 1004). It comprises the NAND flash memory array and input/output (including addressing, writing and sensing) circuitry. This understanding of "module" is compelled by computer electronics usage (*see* ¶ 30, note 8 above) and the inclusion of multiple individually erasable blocks, understood by a POSITA as a defining, characteristic structure of NAND flash memory. In fact, as explained at paragraph 43 above, "SLC memory" and "MLC memory" are physically distinct structures required for distinct programming and reading, even without additional compartmentalization by modularization. There is no subdivision of flash storage cells that could be usefully coupled to the controller without the input/output circuitry. A POSITA would also understand not only from their distinct recitation, but from essentially different capacity, organization and incremental writing in passes, that an SLC module is different from an MLC module. Operation of part (such as a block) of an MLC module in 2-state (although sometimes loosely called "SLC") <u>mode</u> (¶ 49 above) does not transform it into an SLC <u>module</u> both because of the distinct ordinary usage

---

[30] Figs. 6-8 show a "current" Samsung NAND flash chip K9F1G08XOA with a "1024M + 32M BIT NAND FLASH ARRAY." With "ENDURANCE: 100K PROGRAM/ERASE CYCLES" (Fig. 8), this is clearly an SLC module. (*See* ¶¶ 44, 49 above.)

of "mode" versus "module" (a block of connected cells is not by itself a memory, much less a "module") and the substantial operational, performance and endurance differences between "pseudo-SLC" and real SLC (*id.*). The only suggestion in the common specification of a beneficial hybrid SLC-MLC memory system called for "SLC NAND flash equal to 12.5% or 25% of MLC NAND flash" (4:61-62), which is consistent with Fig. 4 showing two banks each of 2+ SLC modules compared to 8+ MLC modules.[31] The '298 patent uses the term "SLC module" to describe a

---

[31] *See also* Duann and Moshayedi ratios at note 25 above. This also corroborates the POSITA's understanding that SLC modules comprise all SLCs as distinguished from MLC modules, rather than any partition for 2-state operation of MLCs: the specification for the first, '298 patent was filed on April 25, 2012, nearly ten years after issuance of Chen '528 (Ex. 1040), which disclosed the 2-state operation of MLC, along with at least the 2001 Cho Dual-Mode article (Ex. 1068), which Dr. Rao filed with his application resulting in the '298 patent. Despite this, the '298 patent, claiming controller-directed use of the distinct MLC and SLC modules, mentions nothing about such dual mode operation, even with dependent claims to specific alternative technologies. There is no suggestion of a partition of MLC modules for 2-state operation. The word "module" appears throughout, but "mode" not at all.

distinct memory device for proposed replacement by alternative technologies – including comparison to "rotating storage media (e.g., HDDs)."[32] This is consistent with the POSITA understanding of "SLC module" as a "self-contained, interchangeable" chip (note 8 above).

70. **The controller and defining, SLC remapping function:** Tying the subject SLC and MLC foundational modules together in the NAND flash memory system is a "controller." The term by itself is abstract, merely suggesting some "control" of the physical components, namely SLC and MLC NAND flash modules. The '298 patent specification might have recited a POSITA-understood NAND flash memory controller with the functions outlined at paragraph 31 above or in

---

[32] The SUMMARY states that "SLC module can be (NAND) flash, PCM, RRAM, MAGRAM or any other solid-state non-volatile memory as long as it has endurance that is superior to that of MLC flash, and it provides data access speeds that are faster than that of MLC flash or rotating storage media (e.g., HDDs)." (4:10-15 [emphasis added].) The '298 patent does not and cannot define "SLC" to include the alternative technologies, but merely posits that an "SLC module" may be a NAND "flash" or any other "solid state [physical] non-volatile memory" faster and having more endurance than MLC "flash" and hard disk drives. The clear implication is that these systems are physically complete and interchangeable with the SLC module.

incorporated Rao '916 Fig. 3; it instead maintained abstraction, stating in the SUMMARY reproduced above that in "one embodiment" there is a controller for "controlling" the SLC and MLC modules (3:65-67), particularly "maintain[ing]" a logical to physical address map (3:67-4:4). The function further defining this embodiment is "conduct[ing] a data integrity check" after "each write" "to ensure that data was written correctly," and if not, "modif[ying]" the L2P address map to remap into the SLC module (4:4-10).

71. **The Data Integrity Check operational feature:** The DETAILED DESCRIPTION states that "[t]he controller conducts a data integrity test after every write" (4:55-56), but a further description of this "read-modify-write" function at 5:32-40 qualifies "every" with "as time permits." The only structure for the function is set forth in method steps of Figs. 3a and 3b and the description at 5:55-6:23.[33] According to that description, in Fig. 3a (100) a command is received (presumably

[33] Figs. 3a and 3b of the '298 patent are virtually identical to the hand-drawn Figs. 3a and 3b in the Provisional (Ex. 1003, sheets 2 and 4) except for a renumbering. The descriptions are close (*compare* '298 5:55-6:23 with Ex. 1003 at 7-8, ¶¶ [0022] and [0023]) with some exceptions noted below.

from the host) to write a "quantum of data stored in DRAM"[34] to a particular location in NAND flash (not limited to the MLC module or the SLC module). In step 102, that data is "read from DRAM into memory within the device controller (which acts as the memory controller)"[35] (5:60-61 [emphasis added]). In step 104, "both the

---

[34] A POSITA would understand that (1) the abstract "quantum" refers to data from the host directed to a logical NAND block address (with sub-addresses) for that data (which may have been previously established in update cases), and (2) a NAND flash device controller would assign (or have assigned) a physical NAND block address and issue appropriate memory commands to be executed at the SLC NAND flash and/or MLC NAND flash module level according to their distinct physical structures and operations. None of this is disclosed for alternative technologies. The specification also does not disclose how the quantum of data gets to DRAM in the disclosed system, which might be assumed to be DRAM 20. However, in my view, the referenced DRAM should be that of (or directly connected to) processor/host 12, rather than DRAM 20 shown as directly connected to device controller 14 in Fig. 1.

[35] The "controller" is connected in Fig. 1 (described at 4:64-5:15) as device controller 14 by lines ("coupl[ing]") to processor 12 (host computer), DRAM 20, MLC flash 26, SLC flash 28 (such as "NAND flash memory modules, confirming their physical distinction) and other discrete devices such as disks 24. Other than the

logical address range and the NAND physical address range to which [the data] is to be written is read into memory of the device controller [presumably the antecedent "internal" memory]" (5:61-65). In step 106 that data (in that "controller" memory) is "combined with the contents of the NAND flash memory"[36] (5:66-67); it is

---

just quoted description of "memory within the device controller," there is no description of the structure of a controller having internal memory into which contents of "DRAM" may be "read." A POSITA would expect a NAND device controller to be a separate chip (*see* photograph at ¶ 30 above, Rao '916 [Ex. 1004] Fig. 3) with some volatile memory for logic operations using the same chip die (*e.g.*, data registers in Rao '916 Fig. 3) but not a different technology DRAM device for temporarily storing NAND-page quantities of data. Block/step 102 and block/step 104 in Fig. 3a both refer to "memory of device controller," where it makes more sense that on-controller memory holds the logical and physical addresses in step 104 than a NAND page in step 102 and possibly the combined data in step 106.

[36] This appears ambiguous as to whether the "contents" are at the target range of physical addresses that must be erased before writing. However, the Provisional stated that the data to be written "is combined with the present contents of the entire address range that the data will be written to." ¶ [0021]. Thus the target range is the

unstated where this is done and where the results are held. In step 108, that "NAND flash physical address range to be written is erased."[37] (5:67-6:1). Then, in step 110, "the combined data is written to the appropriate NAND flash physical address range." (6:1-3.) In step 112, the combined data that was written in step 110 into NAND "is read into device controller memory"[38] (6:3-4 [emphasis added]) – which, again, is not otherwise identified in the specification. Then, in step 114 "the NAND flash physical address range that was read into device controller memory is

---

currently mapped range for the logical file, and the actual destination is updated after or for the combination as the "appropriate" address range in step 110.

[37] This step is highly inefficient as a step close in time to the writing. To erase the address targeted, the entire block must be erased, and other valid data rewritten to other physical blocks – disrupting, adding latency to memory system operations. Normally the erasure would be part of an optimized garbage collection process that could be performed in the background.

[38] It is unclear whether "device controller memory" is the previously referenced "memory within the device controller," which is not explained as part of the device controller physical structure (note 34 above), or DRAM 20 which is shown as coupled to the device controller, which thus might be considered "device controller memory."

compared to the retained data representing the combination of the previous contents of the physical address range and the quantum of data to be written"[39] (6:5-9). If at branch 116 there is no "match" in the step 114 comparison,[40] and if there is available SLC memory, "the failed NAND flash physical address range is remapped[41] to the

---

[39] The text fails to explain, as in block 114 of Fig. 3b that it is the data written in that physical address range, not the address range, that is compared. It is not disclosed where the "retained data representing the combination" is held or compared or how.

[40] This "match" language strongly suggests in context that the '298 patent' read-after-write integrity check is disclosed as a bit-by-bit comparison between data. As I explained at paragraph 57 above, bit-by-bit comparison is inefficient. It is particularly inefficient if made after each write, severely contributing to latency.

[41] It doesn't make sense that the physical address range is remapped to another physical location, since mapping is between logical addresses and physical addresses. There is confusion among the "physical address range" as data (or metadata), the information represented in the "physical address range" and the memory cells of the "physical address range." Note that this two-paragraph embodiment is disclosed (5:55-6:23) in terms of address ranges in contrast to the two-sentence disclosure (6:24-35) of the "wear leveling" embodiment(s) in terms of

next available quantum of SLC NAND flash memory" and the write process from step 110 repeated. (6:20-23.) Nothing is said (for example, for bad block management) about flagging or retiring the physical address range in which data was written and found not to match the data that was mapped to be written there: if that range is not retired, there may be problems in the future resulting in wasteful checking, remapping and rewriting, adding to wear. On the other hand, if the range is retired early in device lifetime, this is also wasteful, taking memory out of service upon a single error. That would lead to earlier exhaustion of system capacity (and thus life) compared to prior art ECC schemes, statistically deployed later in device life, which still allow use (corrected reading) of data found incorrect until error rates exceed chosen thresholds of correctability leading to bad block identification and retirement (¶¶ 51-53 above). Contradicting one statement that the disclosed data integrity check is performed after every write to flash memory (5:4-5), the DETAILED DESCRIPTION qualifies that function: "as time permits" (5:35-36).

72. **Known, essential L2P/FTL as "additional embodiment":** Fig. 4 is described as "another embodiment" where "FTL" logic is "entirely resident" in a NAND flash module that includes two banks of NAND flash memory banks of SLC

"blocks," which, as explained at paragraph 73 below, leaves uncertainty by failure to specify "logical" or "physical."

and MLC NAND flash modules. (6:36-46.) There was nothing new about FTL. I explained at paragraph 34 above the prior art FTL's role in NAND flash-required L2P mapping, wear leveling and garbage collection. FTL is admitted prior art in the specification's background section, mixed with reference to L2P mapping implemented other than in a software layer (2:49-3:13). Nothing is added by the alternative FTL embodiment other than that it exists in some "logic" in a physical module that includes MLC and SLC modules. Such a module typically is a NAND flash memory card as depicted at paragraph 31 above (Micheloni Fig. 2.33) to include a flash file system "implemented as firmware operating in different functional sublayers for wear leveling, garbage collection and bad block management" such as a Flash Translation Layer.

73. **Wear leveling operational feature as an additional "embodiment":** Another "embodiment" is proposed in a two-sentence DETAILED DESCRIPTION possible restatement of the BACKGROUND "wear leveling" (3:1-13) as applied to the hybrid SLC-MLC module system. The first sentence states:

> Another application of one embodiment of the present disclosure, not depicted in any of the drawings, is to allocate "hot" blocks; i.e., those blocks that receive frequent writes, into the SLC NAND flash memory module **28**, while allocating "cold" blocks; i.e., those blocks that only receive infrequent writes, into the MLC NAND flash memory module **26**.

6:24-29.) Although this has some resemblance to the BACKGROUND dynamic wear leveling (3:8-9), it is not explained in the DETAILED DESCRIPTION whether the referenced blocks are the *logical* blocks to which a host directs data frequently or infrequently (like "hot" or "cold" *data* as explained for garbage collection[42]) or *physical* blocks that may be worn by frequent or infrequent writing of different data (since updating data generally means writing to a different physical block). On its face, this embodiment executes the prior art strategy of directing to SLC or MLC "hot" or "cold" *data*, respectively.[43] Nor is it explained what allocation "into" SLC

---

[42] As reviewed at paragraph 37 above, the Micheloni textbook concept of "hot data" relates to clustering for garbage collection efficiency often-updated data associated with logical blocks.

[43] At paragraph 46 above, Lee '794 (Ex. 1041) is shown to write frequently written ("hot") data such as system data and user directories (with associated logical blocks) to physical SLC blocks and less frequently written data such as user data to physical MLC blocks, also directing "hot data" by various logical block counts. (*Also* the base writing to SLC or MLC algorithm Sutardja and Moshayedi as explained at paragraphs 47 and 48 above.)

or MLC modules means and when and how it is performed.[44] The second sentence appears to be a sub-embodiment within the same process described in the first sentence:

> <u>This could be accomplished</u> within the device controller **14** described above, which could simply maintain a count of those blocks that are accessed (written to) most frequently, and, on a periodic basis, such as, for example, every 1000 writes, or every 10,000 writes, transfer the contents of those blocks into the SLC NAND flash memory module **28**.

(6:30-35 [emphasis added].) This would maintain the first sentence's ambiguous reference to logical or physical blocks and adds further ambiguity as to whether

---

[44] The Provisional favors the logical block or "data" interpretation, describing a "use of the invention to segregate hot blocks; i.e., those that are changed the most" (Ex. 1003 ¶ [0023]. "Change" aligns with "update" of a logical file, in contrast to the non-provisional specification of "those blocks that receive frequent writes" which suggests physical wear on a physical block, although neither specification explicitly calls this feature wear leveling. The non-provisional specification also changed "a count of those blocks that are accessed to the most" to "those blocks that are accessed (written to) most frequently." However, both Provisional and non-provisional formulations are consistent with benefiting system lifetime by "periodic transfer" of the contents (*i.e.*, of frequently updated logical files (blocks)) to SLC memory.

"maintain[ing] a count of those blocks" means counting blocks that are written *most* frequently or maintaining a count for each block. The first clause of the second sentence is consistent with selective writing of "hot data" (frequently written logical blocks) to SLC as in Lee '794, Sutardja, and Moshayedi (note 43 above). However, it is also consistent with counting writes of the physical block (which is also done in parallel operations in Sutardja [¶ 47 above] and Moshayedi [¶ 48 above). The second clause of the second sentence presents at least two additional ambiguities. First, the term, of "periodic" suggests a time regularity, which may conflict conceptually with the thresholds reached on a random basis but may mean "occasional" reaching of thresholds. Second, "transfer the contents of those blocks" may refer to the data stored in physical blocks or the data associated with logical blocks to be written. Ordinary static wear leveling as reviewed at paragraph 36 above and in the '298 patent background (3:9-13) at paragraph 66 above, involves a move of "cold data" to a relatively worn block.[45] Thus, the second sentence could describe static wear leveling. However, the rationale for moving static or "cold" data in static wear

---

[45] That move would be understood by a POSITA to involve sensing and decoding the data from its previous encoding in a previous block, holding the digital data in a buffer, and then writing into a new block. Terms like "move" and "transfer" could be understood to involve other buffers for the data (or "content").

leveling doesn't apply to moving the data in a physical block that is pushed over a "hot" threshold number of writes. If that data happens to be infrequently updated data, the embodiment would wastefully move infrequently written data to more expensive (and less numerous – such as the 12.5% recommendation, 4:61-63) SLC memory.[46] In my understanding, most physical blocks have similar erase counts because wear leveling has been effectively implemented, and if enough reach a 1,000 threshold, the proposed transfer would overwhelm the SLC module; again, much depends on particular usage patterns. The two-sentence embodiment is unclear in its operation and effect.

---

[46] One can't determine from counting erases of the physical block (PBA) to which data (LBA) is currently mapped, whether that LBA mapped data is frequently or infrequently updated, since p-e cycles of a PBA will likely reflect many different LBA mappings. One might reasonably infer that if the erase count of a PBA falls substantially behind that of other blocks, it is currently mapped to an infrequently updated LBA, and its "contents" may be beneficially moved to a more worn block (¶ 36 and note 11 above). One cannot reasonably make the converse inference that a frequently erased PBA is currently mapped to a frequently updated LBA, since a frequently updated LBA keeps getting its data moved (and mapped) to different PBAs.

VIII. The '298 Patent Claims

74. The '298 patent has one independent claim for a system for storing data comprising (1) an MLC nonvolatile memory module, (2) an SLC nonvolatile memory module and (3) a controller adapted to perform functions disclosed as embodiments as reviewed in the prior Section VI (a) to maintain the known, essential NAND flash L2P mapping (¶ 72 above), (b) to apply a data integrity test, failure of which causes remapping to the SLC module (¶ 71 above), and (c) to apply an apparent wear leveling process of allocating most frequently accessed "blocks" (unspecified as to logical or physical) to the SLC module (¶ 73 above).

A. The Original Claims of the April 25, 2012, Application

75. U.S. Patent Application No. 13/455,267 (resulting in the '298 patent) was filed Apr. 25, 2012, with the following claims:

1. A system for storing data comprising:

at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;

at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks; and

a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module, the controller maintaining an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list

of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module; and

wherein the controller is adapted to determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module.

. . . .

11. The system of claim 1, wherein the controller is further adapted to determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently and wherein the controller allocates those blocks that receive the most frequent writes to the at least one SLC non-volatile memory module.

12. The system of claim 11, wherein the controller determines which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of said blocks is accessed.

13. The system of claim 12, wherein the controller allocates those blocks that receive the most frequent writes by transferring the

> respective contents of those blocks to the at least one SLC non-volatile memory module.

(April 25, 2012, Application at 15, 17 [Ex. 1006 at 17, 19].)

B. Examination of the Application: "Transfer"

76. The Examiner rejected original claim 1 (including the MLC and SLC modules and the controller with the L2P and data integrity test functions) and dependent claims 2-5 (minimal quanta of addresses, use of NAND flash) as anticipated by Gorobets '179 (Ex. 1044) based on the standard L2P mapping of NAND flash and the following disclosure of the data integrity test function:

> . . . wherein the controller is adapted to determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses (Fig. 20 block 2018) within the at least one MLC non-volatile memory module (Main Portion (MLC) No. 2003), fails a data integrity test (e.g. failure programming only the upper page, but still usable as binary, see ¶ [0165]), and, in the event of such a failure, the controller remaps (No. 2017) the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module (remapped to the spare binary block, see ¶ [0165]).

(June 17, 2014, Office Action at 4 [Ex. 1006 at 42].)

77. The Examiner rejected original claims 11 and 12 (including the "wear leveling" allocation function by maintaining a count of access) as obvious over Gorobets '179 (Ex. 1044) in view of Segal (Ex. 1048) based on the following:

> Regarding Claim 11, Gorobets et al. teach, that for wear leveling the system may exchange hot (heavily used) blocks with cold (little used) blocks or spare blocks using a hot count. See ¶ [0172]. Gorobets et al. fail to teach that the exchange would be for SLC portions. Albeit, this would likely happen over the course of normal operation as the MLC with the highest counts are most likely to have programming failure and be allocated to SLC. See ¶ [0165]. Additionally Gorobets et al. teach that areas such as a cache area which will have a higher hot count should be in the binary (SLC) area. See ¶ [0164].
>
> Segal et al. teach that MLC spare blocks can be reallocated to SLC spare blocks in response to a determination that the MLC blocks have exceeded the P/E cycle rating. See ¶¶ [0040 & 0049]. This would ensure that the blocks that are accessed most frequently (the blocks with the highest P/E cycle counts) would be allocated as SLC.
>
> . . . .
>
> Regarding claim 12, both Gorobets et al. and Segal et al. teach that the controller determines which of the blocks of the plurality of the blocks in the MLC and SLC nonvolatile memory modules are accessed most frequently by maintaining a count of the number of times each one of said blocks is accessed (See Gorobets et al. hot counts ¶¶ [0164 and 0172-17 4] and Segal et al. P/E cycle count ¶¶ [0040 & 0049].

(June 17, 2014, Office Action at 7-8 [Ex. 1006 at 45-46].)

78. The Examiner also found obvious the original dependent claims 6-10 to substitution of alternative technologies for the SLC and MLC memories over

Gorobets '179 (Ex. 1044) in view of U.S. Patent Appl. Publ. No. 2009/0268513 ("De Amborri," Ex. 1053) (RRAM, PCM), U.S. Patent Appl. Publ. No. 2009/0307418 ("Chen '418," Ex. 1054) (MRAM), and U.S. Patent Appl. Publ. No. 2010/0058018 ("Kund," Ex. 1055) (alternative technologies having advantages over flash memory). (June 17, 2014, Office Action at 5-7 [Ex. 1006 at 43-45].)

79. The Examiner found that original dependent claim 13 (**allocating by transferring** contents of "those blocks that receive the most frequent writes" to the SLC module) and claim 14 (depending from claim 13, where transfer is on "a periodic basis") "would be allowable if rewritten in independent form including all the limitations of the base claim and any intervening claims." (*Id.* at 8 [Ex. 1006 at 46].)

80. The Applicant did not argue against the rejections but accepted the Examiner's offer to rewrite dependent claim 13. (Aug. 19, 2014, Amendment at 2-3, 5 [Ex. 1006 at 67-68, 70].)

81. The Examiner stated as reasons for allowance: "The prior art of record fails to teach that the controller allocates those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory." (Sept. 2, 2014, Notice of Allowability at 2 [Ex. 1006 at 80].)

82. In my view, Gorobets '179 did not fully anticipate the original claim 1 because its disclosure, as that of Segal, was in the context of "spare blocks" in

"sections" of MLC memory operated in binary ("SLC") *mode*. However, there are many prior art publications that disclose the '298 patent's claimed data integrity test and allocation of a memory system comprising actual MLC and SLC modules where an actual SLC module in appropriate cases can obviously be substituted for binary-mode MLC blocks with clear expectation of success. While "transfer" was viewed by the Examiner as the "point of invention" or distinction from the art that he found, the '298 patent's skeletal description of broad, common functions with overlapping or ambiguous terms (*e.g.*, the specification of "addresses" but not "blocks" as "logical" or "physical") leaves terms like "move" or "transfer" also ambiguous and met by a broad variety of circumstances.

C. The Issued Claims.

83. The '298 patent issued on Nov. 18, 2014 (Ex. 1001 cover) with one independent claim that corresponds to the L2P, data integrity check and wear leveling embodiments reviewed in Section VII.B:

1.A system for storing data comprising:

at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;

at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks; and

a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to:

a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;

b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;

c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and

d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module.

(7:6-8:9). The first two elements of physical SLC and MLC modules are "coupled" to a third element of a controller defined by four functions that reflect with widely varying faithfulness the DETAILED DESCRIPTION disclosures reviewed at paragraphs 71, 72 and 73 above. Controller function a) corresponds to the FTL/L2P functions admitted in the BACKGROUND (2:30-3:13). Controller function b) is recited as a severe abbreviation or abstraction of the DETAILED DESCRIPTION of the Figs. 3a and 3b data integrity test (5:55-6:23) reviewed at paragraph 71 above. Controller functions c) and d) are recited in segregation of terms and concepts of the two-sentence DETAILED DESCRIPTION of apparent wear leveling (6:24-35) reviewed at paragraph 73 above. The Examiner found controller function d), "allocation" of "those blocks that receive the most frequent writes" by "transfer" to the SLC module, to be the sole point of invention.

84. **Dependent Claims:** Claims 2 and 3 limit the system of claim 1 to a minimum quanta of addresses to one block and one page respectively, without specifying logical or physical. Claims 4-10 limit the system of claim 1 to MLC and SLC NVM modules of NAND flash and alternative technologies. Claim 11 limits the system of claim 1 to transfer of content on a periodic basis.

D. Issued Claim Departures from the Detailed Description

85. Both the original filed claims as well as the rolled-up issued claim**s** departed from the DETAILED DESCRIPTION embodiments in significant respects.

86. **Address Map**: Controller function a) (7:16-25) recites basic NAND flash L2P/FTL functionality admitted in the BACKGROUND, but only requires mapping of address in either the MLC or SLC modules (7:15-16, 7:22-25). If only one were mapped, the controller functions b) and c), which explicitly require operation of both modules (7:28-33, 8:1-3), could not be performed. The structure of the claim reinforces the distinct nature and roles of the MLC and SLC *modules*.

87. **Data Integrity Test**: Controller function b) (7:26-33) so abbreviates Figs. 3a and 3b and their explanation (5:55-6:24) that the claim language is not understandable without reading the specification explanation into the claim to replace ambiguous language. It is unclear from the claim language what "data integrity test" is recited and what it means to "fail." But what is to be determined to "fail" (the subject of the test) is the "*range of addresses* listed by an entry and mapped [to physical addresses in the MLC module]," that is, the logical block address of the data *to be* written. That makes no sense. The claimed system has no way of determining whether the data to be written (even if limited by the controller function b) limitation to that data mapped to the MLC module) is "correct." Furthermore, because the test has no temporal or other qualification, it is *not* restricted literally to the "post-write-read" and comparison to other data explained in the DETAILED DESCRIPTION of Figs. 3a and 3b.

88. **Wear Leveling**: Issued claim 1's recitations of controller functions c) and d) reorganize terms and concepts from the two-sentence DETAILED DESCRIPTION "wear leveling" embodiment (6:24-35). Each limitation draws from the two sentences but separates the claimed functions in a way suggesting that they are as independent of one another as they are of function b). Controller function c) is to "*determine*" which blocks are "*accessed*" most frequently (8:1-5), while controller function d) is to "*allocate those* blocks that receive the most frequent *writes by transferring* the respective contents of those blocks" (8:6-9). Claim 1 recites the separate functions without any express temporal or conditional relationship between them. Although the second sentence of the DETAILED DESCRIPTION of the wear leveling embodiment explains that "[*t*]*his* [referring to the first sentence] could be accomplished within the controller . . ." (6:30), there is no such connection between recited controller functions c) and d). Moreover, the recitation of controller function c) doesn't include the terms "allocate" and "write" of the first sentence, which appear instead in the recitation of controller function d). These distinctions support the possibility that the blocks determined to be "accessed most frequently" by controller function c) may not be "those blocks that receive the most frequent writes" which are allocated and whose contents are transferred by controller function d). Whether "those blocks" are physical or logical blocks – and

thus their "contents" and how they are "transferred" – remains ambiguous as I explained in paragraph 73.

E. Claim Construction.

89. As reviewed in Sections VII(A)-(D), the '298 patent specification and claims leave substantial ambiguities. No new hardware structure or software algorithm or structure is disclosed or claimed, only well-known (before July 19, 2011) MLC and SLC NAND flash memory modules (that include known differences in structure, endurance and operation including mitigating wear on their common floating gate cells programmed and read with different circuitry and algorithms) and a controller to perform ordinary and necessary NAND flash memory operations including L2P/FTL addressing and mapping. The '298 patent proposes and claims that known system with the controller characterized at a high level as performing ordinary and necessary L2P/FTL mapping and optional "data integrity test" and "allocation" functions. Those optional functions are recited in broad, abstract terms, some of which are ambiguous in the context of the claims and the known structure and operation of the MLC and SLC NAND flash memory modules.

90. **MLC and SLC memory modules**: Dr. Rao proposed in the incorporated Rao '916 a new use of RAM in the Samsung SLC NAND memory module considered there (Ex. 1004 Fig. 6); his IDS submissions with the '298 patent application distinguished between MLC memory, SLC memory and portions of

MLC memory used in 1-bit mode (*e.g.*, Cho Dual-Mode [Ex. 1068], Takeuchi [Ex. 1069]). He deliberately chose to claim distinct MLC and SLC memory *modules* as the basis of the '298 patent claims – not operating a dual-mode MLC in a 1-bit *mode.* The difference is important as a notice to the public at the 2014 issuance of the '298 patent of what might be manufactured or aggregated that may infringe;[47] it is also important for understanding the limitations or operation of other terms in a validity analysis (*e.g.*, considerations of the different read and write operations). As I explained in paragraphs 40-45 above, even though "original" EEPROM (called SLC after 2001) and MLC share the same kind of floating gate cell, that cell or group of cells, even a flash "block," do not constitute a "memory," which requires at least circuity allowing retrieval/decoding of information stored. That retrieval and decoding of the two levels of logical pages programmed into an MLC memory row of cells (a physical page) requires different circuitry for processing than for the single level of pages programmed into an SLC memory row of the same type of cells. (*Id.*) MLC can be operated to store and retrieve only one logical page per physical page, but SLC cannot be operated to store and retrieve more than one logical page in one

[47] Even ten years later, when Patent Owner sued Phison, SLC and MLC, and dedicated pSLC (MLC all in binary mode) are marketed with distinct physical characteristics, performance and endurance ratings. (¶ 49 above.)

physical page.[48] (*Id*.) With the claim terms' recitation of "memory" and "module," terms that have meaning, a POSITA's ordinary and customary understanding of the specifically recited "SLC memory" and "MLC memory" as distinct types of NAND flash memories and "SLC memory modules" and "MLC memory modules" are self-contained, interchangeable manifestation of those memories. Reflecting my opinion of the POSITA's understanding, compact, but accurate constructions (proposed by Phison in the court litigation) are:

- "**MLC memory modules**" means "modules comprising MLC non-volatile memory" *where*
  "**MLC non-volatile memory**" means "non-volatile memory cells arranged with circuitry capable of storing multiple logical pages in a single physical page of cells" and

[48] The '298 specification explains that in "single-level storage (SLC), erased bits . . . can only be changed [programmed] to the opposite state [programmed] once before the entire block must be erased." (2:40-43.) Thus, a row (page) of SLC cells cannot be programmed for a second, upper logical page as can a page of MLC cells. This is at least because SLC memory lacks a second page buffer, the associated circuitry needed to operate the two buffers, as well as the circuitry to enable the encoding of two bits of information in a single cell.

- "**SLC memory module**" means "module comprising SLC non-volatile memory" *where*
  "**SLC non-volatile memory**" means "non-volatile memory cells arranged with circuitry incapable of storing multiple logical pages in a single physical page of cells"

91. "**Blocks**": As I reviewed in paragraph 88, the ambiguity as to whether the "blocks" in the two-sentence description of the "wear leveling" embodiment (6:24-35) are physical (memory) or logical (data) blocks, and in addition, it is ambiguous as to what the contents are and how they are "transferred" (¶ 73). This ambiguity persists at least for the limitation of controller function d), for the transfer of "those blocks that receive the most frequent writes." The "blocks" could be logical blocks that receive the most frequent writes, and the contents would be the data associated with the logical block address (LBA) at the time of writing, which could be transferred to the SLC module, as in the common approach explained at paragraphs 46-48 above and strategy explained at paragraphs 58 and 60 above. Alternatively, if the "blocks" were physical blocks, and one happened to reach a threshold count with infrequently updated data in the block, this would result in a wasteful transfer to SLC. (*See* ¶ 73 above.) I understand that Phison has taken the position in the litigation that the ambiguous uses of the term "blocks" in claim 1 makes the claim indefinite. Because such a finding isn't an option here, I will explain that the art discloses wear leveling operations based on both interpretations.

92. **"Controller"**: Here too, I understand that Phison has taken the position in the litigation that the term is indefinite for failure of the patent specification to disclose structure in the way of specific algorithms for its recited functions. Again, because such a finding is precluded here, I will analyze the patentability of the Challenged Claims as the claim language and disclosure permit.

93. **"Data Integrity Test"**: I understand that Phison has taken the position in the court litigation that this term (across multiple patents) means "a test that compares stored data to retained data." As reviewed at paragraph 87 above, the recited controller function b), which includes the data integrity test, tests "a range of addresses listed by an entry and is mapped," which, absurdly, is the logical address of the data to be written. Even if the Board reads the object of the data integrity test to be the *data* associated with the logical address mapped to the physical address rather than the *logical address itself*, the data integrity test recited as controller function b) is not literally limited to the comparison of the DETAILED DESCRIPTION of Figs. 3a and 3b.

94. **"Transfer," "Allocation," and "Content"**: Because of the remaining ambiguity of whether these terms are used relative to physical or logical blocks, these terms may apply to different aspects of information movement in the claimed or prior art NAND flash memory systems, including the necessity of coding, decoding, buffering and alternative uses of pointers (*e.g.*, note 45 above). The terms,

as others, should be given their ordinary and customary meaning in the context of their use.

IX. Patentability of the '298 Claims Relative to the Art

95. The independent claim 1 of the '298 patent claims a method of operating a known hybrid SLC-MLC memory system with known, necessary L2P mapping operations, a mis-stated "data integrity test" feature, a *determination* of most frequently *accessed* blocks feature, and a potentially distinct feature of *allocation* by "*transfer*" of the most frequently *written* blocks to the SLC module. Multiple alternative "data integrity checks" and approaches to avoid premature unreliability of NAND flash systems, including SLC-MLC hybrids, have been detailed in multiple prior art textbooks, articles and patents, as reviewed in Section V(C). Although, as reviewed at Section VIII(B) above, the Examiner did not find a reference considered to disclose the limitation of controller function d), "transferring the respective content," it is my opinion that a POSITA would find that each recited limitation of the Challenged Claims, including that one, is disclosed by the multiple NAND flash error management and wear leveling operations disclosed in each of Gavens (Ground 1), Moshayedi (Ground 2), and Sutardja (Ground 3), in view of the knowledge of a POSITA. Starting with a Gavens NAND flash system with multiple alternative error management operations, a POSITA would understand how to use that system with wear leveling operations together meeting the

Challenged Claim limitations directed to avoiding premature system failure. Alternatively, starting with the Moshayedi or Sutardja systems, each with multiple wear leveling operations, a POSITA would understand how to use that system with error management operations meeting those Challenged Claim limitations.

X. Ground 1: The Challenged Claims Are Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA

A. Claim 1 Is Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA.

96. Gavens (Ex. 1045), among the SanDisk portfolio and incorporating by reference others of that portfolio,[49] is a published reference that discloses a complete

[49] *See, e.g.*, U.S. Patent No. 6,456,528 to Chen ("Chen '528"), Ex. 1040, incorporated by reference at Gavens 16:29-40; U.S. Patent Appl. Pub. No. 2011/0153912 ("Gorobets '912"), Ex. 1049, incorporated by reference at Gavens 20:53-59; U.S. Patent No. 5,930,167 to Lee et al. ("Lee '167"), Ex. 1050, incorporated by reference at Gavens 17:57-67; U.S. Patent Appl. Pub. No. 2010/0172180, to Paley et al. ("Paley '180), Ex. 1051, incorporated by reference at Gavens 8:41-42. Hereinafter, it should be understood that reference to a specific portion of any of the above references is understood to be disclosed by Gavens due

multi-modal NAND flash system with the ordinary L2P/FTL function of its controller, a variety of post-write read error detection and management schemes, and the capability of and compatibility for employing other known tools (disclosed in incorporated documents and exemplified in paragraphs 58-61 above) used to avoid premature NAND system failure.

97. Since the 2001 (*e.g.*, Cho, Ex. 1039) emergence of the terms "multi-level program cell" (MLC) memory and (thereby distinguished) "single level program cell" (SLC) memory, there have been implementations of hybrid MLC memory modules and SLC memory modules availing of the endurance advantages of SLC. (*E.g.*, Lee '794 [Ex. 1041], Sutardja [Ex. 1042], and Moshayedi [Ex. 1043].) Many NVM implementations disclosed in the SanDisk patent portfolio involve the partitioning and configuration of some MLC (or TLC) blocks for one-level (two states) or pseudo-SLC (pSLC) operation. As reviewed at paragraph 49 above and in Gavens itself (*e.g.*, 12:6-10) the pSLC mode of operation enjoys superior endurance compared to MLC in multi-level mode – though not as superior or as fast as "real" or "native" SLC memory which uses different circuitry. It would be obvious to a POSITA to have one MLC module operating entirely in pSLC mode

---

to incorporation by reference, as if the incorporated document was explicitly contained in Gavens.

(and thus be considered a pSLC memory module, which, as mentioned in paragraph 49, are marketed as such) with an expectation of success for the relative superiority to MLC in multi-level mode[50] or use a "real" or "native" SLC module as understood by a POSITA from the embodiments actually disclosed in the specification. Like a memory cell, a block of memory cells is not memory – it does not have the necessary circuitry to program and read the threshold voltages that represent the logical data.

98. I show here that Gavens and its incorporated references, in view of the knowledge of the POSITA, disclose the limitations recited in claim 1 as set forth in the groupings as printed in the '298 patent:

1. [1Pre] A system for storing data comprising:

99. I understand the question of whether a preamble is limiting is a question of claim construction. I do not provide an opinion as to whether the preamble of claim 1 is limiting.

100. To the extent the preamble is limiting, Gavens discloses "a system for storing data," namely, the "memory device 90" that includes memory chip 100 with

[50] This would meet the '298 patent specification suggestion that "SLC module" may be "any other solid-state non-volatile memory as long as it has endurance that is superior to that of MLC flash" and faster ('298 4:10-14).

a memory array 200 and other circuitry, to which the host 80 "sends data to be stored" or from which it "retrieves data by reading" (Gavens Fig. 1 (90), 8:13-17).

2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;

101. Gavens' memory chip 100 "includes a memory array 200 of memory cells with each cell capable of being configured as a multi-level cell ("MLC") (8:19-21). The memory array may be organized in erasable blocks erased together in a "flash" operation (Fig. 6, 11:4-20). Thus, Gavens discloses in memory chip 100 meeting limitation **[1a]**. The base memory chip of Gavens meets the characteristics of the POSITA-understood MLC flash memory module of cells, circuitry and programming explained in paragraphs 26-30, 42 and 43 above. It particularly meets Phison's multiple level page programming requirement (distinguishing from SLC) set forth in paragraph 90 above, as programming of those levels of logical pages is disclosed in Fig. 23 (25:18-26:30). Thus, Gavens (Fig. 1 (100), 8:13-21) discloses claim element **[1a]**.

3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;

102. Gavens does not disclose the use of a "native" SLC flash memory module or chip as described at paragraphs 26-30 and 90 above because the Gavens memory module or chip 100 is capable, with its circuitry, of programming more than one logical page into a physical page, as explained in paragraph 99 above. However,

Gavens discloses partitioning, configuration and use of portions of its memory array 200 in 1-bit mode, the "pseudo-SLC" mode explained at paragraph 49 above. Specifically, Gavens discloses, as a base system for multiple operational embodiments, "[a] first portion has each memory cell storing one bit of data and [a] second portion has each memory cell storing more than one bit of data" (4:32-34, 12:6-8 [calling memory operating in 1-bit mode "binary" or "SLC" memory]). Regarding one aspect of the difference, it is explained that the first portion, being "configured as lower density storage[,] . . . operates with a wider margin of error than that of the second portion" (18:16-20, also 12:6-10, 21:6-9). As explained in paragraph 96 above, it would be obvious to a POSITA to substitute a "native" SLC flash memory module for the Gavens first portion, with a similar, if not wider margin of error than binary mode operation of MLC (*see* ¶ 49 above) and achieve similar, if not better results with the same NAND flash controller Fig. 1(102) with great expectation of success. Thus, Gavens (4:32-34, 12:6-8), in view of the knowledge of the POSITA's knowledge of NAND flash, discloses the obvious use in the claimed system of claim element **[1b]**.

4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to;

103. Gavens discloses controller 102 which "cooperates with the memory chip and controls and manages high level operations" (Fig. 1(102), 8:32-33). It is

noted that "one or more memory chip[s] 100 [are] managed by a controller 102" (8:17-18), multiple (NAND flash) chips shown in Fig. 1 by ellipsis with bi-directional coupling to the controller. It would be obvious to have one or more of those NAND flash chips be "native" SLC flash chips (modules), with the controller directing host commands and data to the SLC and MLC NAND chips with their particular circuitry "to perform memory operations on the memory array 200" (8:25-29.) Thus, Gavens (8:17-18), in view of the knowledge of the POSITA's knowledge of NAND flash, discloses the controller coupled to an MLC module and obviously substituted SLC module, meeting claim element **[1c]**.

5. **[1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;**

104. Gavens discloses this controller function a) as the ordinary NAND controller function: "A memory block management system implemented in the controller stages the sectors and maps and stores them to the physical structure of the memory array" (8:36-39.) For various operational embodiments, Gavens discloses updating of "a directory in a block management system embedded in the firmware of the controller (see Fig. 1)" to direct accesses to one or the other portions

of the memory (17:17-19, 18:47-49, 18:64-66). The known NAND flash organization and mapping operation (*see* ¶¶ 27-29, 33, and 34 above) is explained, with "the page being a minimum unit of programming and reading" (11:9-29).

105. Incorporated reference Paley (Ex. 2051) provides further detail:

> A memory-side memory manager 300 is implemented in the controller 100 of the memory system 90 to manage the storage and retrieval of the data of host logical sectors among metablocks of the flash memory 200. The memory manager comprises a front-end system 310 and a back-end system 320. The front-end system 310 includes a host interface 312. The back-end system 320 includes a number of software modules for managing erase, read and write operations of the metablocks. The memory manager also maintains system control data and directory data associated with its operations among the flash memory 200 and the controller RAM 130.

(¶ [0155].) Paley explains the use of "the currently prevalent LBA interface" (¶ [0020]).

106. Thus, Gavens (8:36-39. 11:9-29, 17:17-19, 18:47-49, 18:64-66) and its incorporated Paley reference (¶¶ [0020], [0155]) disclose the ordinary NAND flash controller function a) of claim element **[1d]**.

6. **[1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;**

107. As explained in paragraph 88 above, the literal reading of controller function b), determining whether a range of (logical) addresses listed by an entry in an L2P map "fails a data integrity test," is absurd. As the Board may not invalidate on this basis, it should look to the "data integrity test" disclosed in the specification (Figs. 3a and 3b) to understand this recited "data integrity test" function to be a comparison of data to be written to a range of physical addresses (NAND cells), written there and read back and compared to a retained copy of the data to be written, failure of which results in remapping to the SLC module. As explained at paragraphs 51 and 52 above, there are various types of data integrity tests, including by ECC that compares the original data and written data only indirectly and with "failure" not typically remapping at all, much less to an SLC module. As reviewed at paragraphs 53-55 above, Gavens discloses multiple operations for error management, most relying on ECC, and deployment near the end-of-life of a memory system. However, two embodiments describe the controller function b) comparing read data to original data and remapping to SLC if there is no match.

108. In one variant of the Gavens post-write-read error management embodiment of Figs. 14 and 15 (explaining text at 16:13-17:42), user data is written to the higher density (*e.g.*, MLC) portion as a "first copy." (16:41-45.) Then, in a "post write read," there may be a check for errors "by comparison with the **original copy** which may be cached" (16:46-49 [emphasis added]). If the number of error bits is less than a predetermined amount, the first copy is deemed valid and subsequent reads of the data page will be from that copy with any errors corrected by ECC. (16:50-56.) If the number is greater, then a copy is written to the lower density, lower error rate (more robust, *e.g.*, "SLC") portion; this is performed in one embodiment by copying from a cached copy ("of the original data") or in another embodiment from application of ECC to the first copy. (17:9-14.) In incorporated Paley (Ex. 2051), it was noted that "[u]sing RAM in a write cache operating with flash memory has been disclosed" (¶ [0027]); the '298 specification shows in Fig. 1 DRAM 20 which is where data to be written in the data integrity check apparently is first held (5:57-60). Thus, this variant discloses that written (stored) data that fails a data integrity test against retained (cached) original data to be written causes remapping of the original data to the SLC portion.

109. In another group of Gavens alternative post-write-read error management embodiments of Figs. 16 and 17 (explaining text at 17:43-19:24), the more robust portion is further divided into a first section for caching incoming data

(simultaneously with a "first copy" written into the higher density portion) and a second section for storing rewrites from the higher density portion. (18:21-29.) In a variant of that embodiment, the first copy (in higher density) is compared with the cached copy in the first section of the more robust portion in the on-chip data latches (18:30-44), and if the number of error bits is greater than a threshold, then a copy is rewritten from the cached copy to the rewrite section of the more robust portion. (18:53-66.).

110. It is understood that these actions are directed by the controller. The use of a separate SLC module instead of a binary-mode portion of one MLC module does not significantly increase latency, since any "copying" involves decoding, temporarily storing in a buffer, and coding (see notes 11 and 45 above) and comparison may be more efficiently done off the memory chip.

111. Thus, Gavens, in both the variants recited (16:13-17:42, 17:43-19:24) discloses the data integrity test controller function b) of element **[1e]**.

7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and

112. As explained in paragraph 88 above, this recited controller function c) is not tied to controller function d) explicitly, temporally or even by use of the same terms. All that is required by claim 1 is that the controller is "adapted" to

"determine" which of the blocks in the NAND flash system are "accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed." This limitation is met on its face by Gavens's "[t]racking the age of each block by maintaining a hot count that records the number of erase/program cycling each block has undergone" (Fig. 19(720), 20:18-21). Although (and because) the controller uses that information to determine according to a threshold count whether a particular post-write-read scheme should be implemented for the block (Fig. 19(730), 20:22-27), the controller clearly is adapted to maintain the counts and determine from those counts the blocks accessed most frequently.

113. Incorporated by reference Lee '167 (Ex. 1050) (2:54-3:7) recites "[t]he usual desire to evenly wear the memory" in the SanDisk portfolio that is met by its wear leveling approach that maintains "separate counts of the number of times that each sector has been erased and programmed." Incorporated by reference Gorobets '912 (Ex. 1049) ¶ [0126] also discloses wear leveling erase/rewrite counts among all blocks.

114. Incorporated by reference Chen '528 (Ex. 1040), was discussed at paragraph 49 as one tool for avoiding premature system failure by taking advantage of the greater wear tolerance of two-state operation by operating memory cell blocks to which data are written most frequently in two-state mode. To accomplish this, the Chen system (controller) maintains counts of the number of erase/programming

cycles of the individual blocks (11:7-10). Incorporated by reference Paley '180 (Ex. 1051) ¶ [0542] discloses a similar use of two-state mode for physical blocks with a high "hot count" that is tracked by the system. These disclosed embodiments clearly use the controller to determine which blocks are accessed most frequently. They are not inconsistent with the operation of the Gavens post-write-read remapping to binary MLC memory, meeting controller function b) (¶¶ 108-110 above).

115. Thus, Gavens and incorporated references disclose the determining number of accesses controller function c) of claim element **[1f]**.

8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module.

116. As explained in paragraph 88 above, this recited controller function d) is not tied to controller function c) explicitly, temporally or even by use of the same terms. A possibility exists, however that "those blocks that receive the most writes" of controller function d) finds antecedent in the controller c) function of determining which of the blocks "in" the MLC and SLC modules are "accessed most frequently" so that both referenced blocks are the NAND flash physical blocks in the MLC and SLC modules. As explained in paragraph 88, such a reading would make sense for the common direction of logical data frequently updated to more robust SLC memory; however, it is contrary to traditional wear leveling in which data that is stored in *less* frequently written physical blocks (presumed to not have been updated

for some time[51]) are moved to more worn blocks. I consider here construction of the blocks allocated alternatively as physical or logical.

117. The Gavens embodiments of post-write-read mapping to 2-state memory that meet controller function b) (element [1e], ¶¶ 108 and 109 above), as deployed after the physical block reaches a threshold erase/program count (Fig. 19 (730, 740), 20:21-29) literally meet limitation [1g], which has no temporal or completeness qualification: when those blocks exceeding the threshold (receiving the most frequent writes) are written with data that fails a data integrity test, they are rewritten (transferred) to SLC.

118. Claim 1 does not have an explicit, rather than implicit, wear leveling qualification; however, the various tools are disclosed by Gavens and its incorporated references disclose transfer of data to avoid premature failure of the system because of excessive wear on a component, track wear at a physical block level. For example, paragraph 113 above recited such disclosures in the incorporated

---

[51] A converse presumption of frequently updated data is less warranted for a threshold of writes to a physical location, since that location may happen to have a write of infrequently updated data pushing the location over the threshold. (*See* note 46 above.)

by reference Lee '167 (Ex. 1050) (2:54-3:7) and Gorobets '912 (Ex. 1049) (¶ [0126]).

119. As reviewed in paragraph 114 above, incorporated by reference Chen '528 (Ex. 1040) (10:41-60) and Paley '180 (Ex. 1051) (¶ [0542]) disclose an additional tool for avoiding premature failure due to block wear by operating high-erase count (most frequently written) blocks in 2-state mode. These, and the operation of the Gavens post-write-read embodiments result in writing to 2-state mode after the target physical blocks exceed a threshold disclose controller function d) if not literally, by teaching such writing or transfer.

120. That objective and implementation at least at the functional level was admitted in the Provisional (Ex. 1003) "Description of the Prior Art" section) paragraph [007]: "[W]ear leveling algorithms within the FLASH devices . . . to attempt to ensure that hot blocks; i.e., those that are frequently written, are not rendered unusable much faster than other blocks." At least two disclosures for wear leveling with NAND flash memories of different endurance provide for transfer to the higher endurance (SLC) memory of data "in" logical blocks determined to be frequently updated ("hot data"), but with additional embodiments for "swapping" of data stored in frequently written physical (memory) blocks.

121. As explained at paragraph 47 above, Sutardja, (Ex. 1042) discloses a controller directing a base algorithm similar to that of Lee '794 (Ex. 1041), mapping

logical addresses with low write frequency (as reported by the host) to the lower endurance memory and logical write addresses with high write frequency to the higher endurance memory. (Fig. 7A (506) & (508).) However, Sutardja also discloses a wear leveling process to allocate those blocks (understood to be NAND flash) that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module (disclosed as higher-cost, faster memory distinct from another lower-cost, slower memory) (Fig. 7E, ¶ [0141]).

122. Moshayedi (Ex. 1043) discloses a controller that is adapted to allocate those blocks (again understood to be NAND flash) that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module (explicitly disclosed as distinct from an MLC non-volatile memory module). ([Abstract] disclosing a system that keeps frequently written data, which results in frequently erased blocks, in the SLC flash module, and relatively static data in MLC flash module; *also* Section XI(A)(8) below.) Disclosed is a process similar to that of Lee '794: the drive (controller)

> keeps track of the number of times that data for each logical block address (LBA) has been written to the flash memory, and determines whether to store newly received data associated with a particular LBA in SLC flash or in MLC flash, depending on the number of writes that have occurred for that particular LBA

(¶ [0024].) However, also disclosed is a variant of traditional wear leveling that resembles the second sentence of the DETAILED DESCRIPTION embodiment that roughly corresponds to controller function d), element [1g]:

> once a block in MLC flash reaches a threshold erase count (e.g., 500), the next write operation to that block triggers a swap where the data from the MLC flash block is written to a block in SLC flash. In this manner, data for an LBA that is frequently written and causes frequent erasures is moved to SLC flash which can perform more erase cycles than MLC flash.

(¶ [0032].)

123. Thus, Gavens and incorporated references (with the POSITA's knowledge of obvious substitution of a "pure" SLC module for its partitioned and 2-state (pseudo-SLC) configured MLC module), in view of background knowledge of wear leveling exemplified by Sutardja, Moshayedi, and Lee '794 (with "pure SLC" modules, and as background processes compatible with Gavens' error management techniques), disclose the allocation by transfer controller function d) claim element **[1g]**.

* * * * * * * * * * *

124. In summary, Gavens, with its incorporated references, meets each limitation of '298 claim 1 with known hybrid SLC-MLC NAND flash memory module systems and mixed SLC-mode and MLC-mode operation of MLC memory

meeting elements [1a]-[1d], with the POSITA's understanding that Gavens's extensive technology-justified teachings include or render obvious, with the knowledge of the POSITA of many NAND flash operations used to avoid premature failure, the features of elements [1e]-[1g].

B. Claim 2 Is Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA.

1. [2Pre] The system of claim 1,

125. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [2] wherein the minimum quanta of address is equal to one block.

126. Gavens discloses or renders obvious limitation [2]. (Fig. 20C, 21:22 ["virtual block"].)

127. Incorporated Paley '180 (Ex. 1051) explains block-level addressing (¶ [0016] ("logical blocks"), ¶ [0163] ("logical address LBA").

128. Thus, Gavens and incorporated references, in view of knowledge of the POSITA, discloses claim 2.

C. Claim 3 Is Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA.

1. [3Pre] The system of claim 1,

129. I incorporate here my opinions set forth in Section IX(A) for claim 1.

2. [3] wherein the minimum quanta of address is equal to one page.

130. Gavens discloses a minimum quanta of addresses equal to one page. (Fig. 4 (70), 10:19-20, 41-47, 20:2-4 [a page 70 is a group of memory cells enabled to be sensed or programmed in parallel], 3:21-22[a page of memory elements are read or programmed together],11:20-29 [A page is a minimum unit of programming and reading], 9:36-40 [the memory array 200 is arranged in rows and columns of memory cells, addressable by word lines and bit lines].)

131. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 3.

D. Claim 4 Is Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA.

1. [4Pre] The system of claim 1,

132. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [4] wherein the MLC non-volatile memory module is NAND flash memory.

133. Gavens, which would be understood by a POSITA to describe a NAND flash system with multiple MLC NAND flash modules, expressly describes its MLC non-volatile memory modules as including MLC NAND flash memory elements, which by definition are flash memory. (Fig. 4, 9:49-50 [page of memory cells in the memory array 200 is organized in the NAND configuration],1:18-19, 4:14-18 [flash memory having a first portion and a second portion storing higher density compared

to the first portion], 2:65-66, 8:65-67 [flash memory devices with NAND string structures; Fig. 1 (90) [flash memory device].)

134. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 4.

E. Claim 5 Is Obvious Over Gavens, Including Incorporated References, in View of Knowledge of the POSITA.

1. [5Pre] The system of claim 1,

135. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [5] wherein the SLC non-volatile memory module is NAND flash memory.

136. I incorporate here my opinions set forth in paragraph 133 above (Gavens's use of MLC).

137. As explained at paragraphs 97 and 102 above, a POSITA would readily and successfully substitute for Gavens' partitions of MLC memory elements configured for 2-state pseudo-SLC operation an SLC non-volatile memory module which is understood to include SLC NAND flash memory elements. As to Gavens's MLC NAND flash memory operated in pseudo-SLC mode, they are also described as flash memory. (Fig. 4, 9:49-50 [a page of memory cells in the memory array 200 is organized in the NAND configuration], 1:18-19, 4:14-18 [flash memory having a first portion and a second portion storing higher density compared to the first

portion], 2:65-66, 8:65-67 [flash memory devices with NAND string structures], Fig. 1 [flash memory device 90].)

138. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 5.

F. Claim 6 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [6Pre] The system of claim 1,

139. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM).

140. Although I do not see how RRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

141. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex. 1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

142. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 6.

G. Claim 7 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [7Pre] The system of claim 1,

143. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM).

144. I incorporate here my opinions set forth in paragraphs 140 and 141 above.

145. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 7.

H. Claim 8 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [8Pre] The system of claim 1,

146. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM).

147. Although I do not see how PCM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

148. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex.

1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

149. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 8.

I. Claim 9 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [9Pre] The system of claim 1,

150. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM).

151. I incorporate here my opinions set forth in paragraphs 147 and 148 above.

152. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 9.

J. Claim 10 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [10Pre] The system of claim 1,

153. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM).

154. Although I do not see how MAGRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations

required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

155. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of Chen '418 (Ex. 1054) (MRAM), and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

156. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 10.

K. Claim 11 Is Obvious Over Gavens, Including Incorporated references, in View of Knowledge of the POSITA.

1. [11Pre] The system of claim 1,

157. I incorporate here my opinions set forth in Section X(A) for claim 1.

2. [11] wherein the controller causes the transfer of content on a periodic basis.

158. In my view, as explained at paragraph 73 above, the use of the term "periodic" in the DETAILED DESCRIPTION (6:30-35) is in tension with the p-e count thresholds recited in the same sentence. To the extent that "periodic" means "occasional," the actions disclosed in Gavens and incorporated references and background as reviewed at paragraphs 117-121 above are "occasional" or responsive to conditions such as reaching a threshold.

159. Also, Gavens discloses that the controller performs operations in the background on a periodic basis. (15:39-41 [data retention errors can be alleviated by periodically refreshing the threshold levels of the cells in a "read scrub" operation. Incorporated by reference Lee '167 (Ex. 1050) (2:7-10, 30-37 [new data is initially stored in SLC. Later, preferably in the background without slowing down other operations, the data is read out of SLC and reprogrammed into MLC].).

160. Sutardja discloses that its system "determines whether [the] time to perform data shift analysis has arrived in step 514." (¶ [0148].)

161. Thus, Gavens and incorporated references, in view of the knowledge of the POSITA, discloses claim 11.

XI. Ground 2: The Challenged Claims Are Obvious Over Moshayedi in View of Knowledge of the POSITA

A. Claim 1 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

162. Moshayedi (Ex. 1043) is a published reference that discloses a complete hybrid NAND flash system both MLC and SLC modules, with the ordinary L2P/FTL function of its controller, a variety of wear leveling schemes, and the capability of and compatibility for employing other known tools (exemplified in paragraphs 58-61 above) used to avoid premature NAND system failure.

163. I show here that Moshayedi, in view of the knowledge of the POSITA, discloses the limitations recited in claim 1 as set forth in the groupings as printed in the '298 patent:

1. [1Pre] A system for storing data comprising:

164. I understand the question of whether a preamble is limiting is a question of claim construction. I do not provide an opinion as to whether the preamble of claim 1 is limiting.

165. To the extent the preamble is limiting, Moshayedi's title is "SLC-MLC COMBINATION FLASH STORAGE DEVICE" and claim 13 claims "A flash storage device.

2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;

166. Fig. 1 (114) shows four channels of "chips of flash (e.g., the first chip can be a K9K8G08 SLC flash, and the rest can be K9G8G08 MLC flash)" (¶ [0038]).

167. Paragraph [0003] explains that flash memory is non-volatile; paragraph [0005] explains that flash memory is organized into blocks and that it writes individual segments and pages but can only erase entire blocks.

168. Thus, Moshayedi discloses the first physical component of the claimed system, MLC module claim element **[1a]**.

3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;

169. See paragraphs 166 and 167 above, which recite the use of SLC flash chips shown as Fig. 1(112), thus disclosing the second physical component of the claimed system, SLC module claim element **[1b]**.

4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to;

170. Fig. 1 (102) shows a controller 102 coupled to the four channels of SLC and MLC chips.

171. Paragraph [0025] explains that controller 102 "controls operations of the flash storage device 10." Paragraph [0026] explains how the controller 102 is coupled to the SLC and MLC flash memory chips by an I/O bus. Paragraph [0036] explains that the controller controls the writing and erasing of data to the channels of flash memory in the memory array.

172. Thus, Moshayedi discloses the third physical component of the claimed system, controller claim element **[1c]**.

5. **[1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;**

173. Paragraph [0006] explains that a logical block address (LBA) is mapped to a location within a physical block in the flash memory system. Paragraphs [0036] and [0038] explain "virtual-to-physical mapping (V2P) that is standard L2P mapping for NAND flash. Paragraph [0044] explains that the system firmware assigns LBAs to the physical address. Paragraph [0037] explains that V2P RAM 214 in controller 200 (architecture of controller 102) includes one or more tables that serve to record the physical block address of a specific virtual block. Fig. 3 and paragraph [0039] show V2P entries.

174. Thus, Moshayedi discloses the ordinary NAND flash controller function a) of claim element **[1d]**.

6. **[1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;**

175. Paragraph [0031] discloses that "data structures or linked lists may store information about the number of data errors that have occurred in read operations corresponding to each of the data blocks" and that the information "may allow controller to select a data block from which to move dynamic data in favor of static data." Additional information about read error relative to their frequency or accumulation is explained. Paragraph [0033] explains that the information regarding the number of data read errors associated with a given data block may be used to determine whether the data is dynamic or static, where dynamic data may be "relocated to data blocks with less wear." As SLC is indicated in paragraphs [0007] and [0022] to have 100 times the endurance of MLC, a POSITA would understand "blocks with less wear" to mean SLC. Moshayedi says as much at paragraph [0009] which explains that "static" and dynamic ("frequently written") data are to be kept in MLC and SLC, respectively. This discloses that failures of some data integrity test on a physical block (which is "read") resulting in "read errors" may be relocated (moved by decoding and encoding as explained in note 45 above) to a less worn block. It would be understood by a POSITA that an SLC block would have less

wear (relative to its maximum number of p-e cycles) than an MLC block, or likely have more life remaining when MLC blocks near the end-of-life. This would align with the apparent rationale for the '298 patent's Figs. 3a and 3b direction to SLC of test-failed data stored in a location presumed to be worn, with wear being the focus of the patent and of Gavens late-in-life error management.

176. Limitation [1e] (controller function b)) not only fails to identify what is to be tested with any apparent reason but does not qualify the test as one triggered by a NAND flash operation such as a write or as a comparison to something else. (*See* ¶ 88 above.) If the Board does not read a comparison between stored and retained data into claim element [1e] as Phison has requested in the court proceeding (¶ 93 above), Moshayedi meets its literal limitation if the recited test is applied to the data stored in the physical range of addresses.

177. Even if Moshayedi is not deemed to disclose limitation [1e], a POSITA seeking operations to help avoid premature failure of a NAND flash memory system would look not only to wear leveling but to known error management techniques such as those disclosed in Gavens. As shown in paragraphs 108 and 109 above, at least two Gavens post-write-read error management embodiments meet limitation [1e] even if the Board reads into it the comparison test of Figs. 13a and 13b.

178. Thus, Moshayedi, by its read error management of paragraphs [0031] and [0033], or through knowledge of the POSITA, discloses the data integrity test controller function b) of claim element **[1f]**.

7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and

179. Moshayedi discloses that the controller determines which physical blocks in the MLC flash and the SLC flash are accessed most frequently (¶ [0009] explains that write (access) frequency is reflected in erase frequency) by maintaining an erase count of the number of times each physical block is erased. Additionally, Moshayedi discloses that the controller determines which logical blocks in the MLC flash and the SLC flash are written most frequently by maintaining a write count of the number of times each logical block is written.

180. Fig. 7A (710-712) and paragraph [0047] disclose checking the erase count of the block. If the erase count is over a specified number (e.g., 1000), then swap MLC flash data to SLC flash.

181. Fig. 8 (802) and paragraphs [0049] and [0050] disclose that if an MLC block has an erase count that reaches 500, then move the data in the MLC block to a free block of SLC.

182. Paragraphs [0030] and [0051] disclose tracking the number of times blocks have been erased. (*Also* ¶¶ [0051] and [0071] [keep the number of times that each logical block has been written to NAND] ¶ [0052] [write counts].)

183. The Abstract and paragraphs [0009] and [0024] state the general operation of the Moshayedi main embodiment: Keep track of the number of times that data for each logical block address (LBA) has been written to the flash memory. If the write count of an LBA is above the threshold, the logical block is written to SLC flash.

184. Thus, Moshayedi discloses the counting number of accesses controller function c) claim element **[1f]**.

8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module.

185. As explained in paragraph 122 above, Moshayedi discloses the common transfer of the "hot data" of logical blocks that receive the most frequent writes to the SLC module (¶ [0024]) plus, in another embodiment, discloses that the physical blocks that receive the most frequent writes (reaches a threshold) have their data "swapped" or moved to SLC (¶ [0032]).

186. The Abstract and paragraphs [0009] and [0024] state the operation of keeping frequently written data in SLC flash. If the write count of an LBA is above the threshold, the logical block data is written to SLC flash. Specifically, "the host

compares the write count of the associated LBA against a threshold. If the write count is above the threshold, the logical block is written to SLC flash" (¶ [0024].)

187. Claim 4 recites transferring the logical block from an MLC chip to an SLC chip. Fig. 3 states "This block is moved from MLC."

188. Alternatively, paragraph [0030] explains that the data contained within a block with the highest erase count is swapped with data from a block with the lowest erase count.

189. Paragraph [0032], quoted at paragraph 122 above, states that when the erase count of the physical MLC block reaches a threshold, a swap is triggered where the data from the MLC flock is written to an SLC block

190. Fig. 7A (712) and paragraphs [0047] and [0060] explain that if the erase count of an MLC block is over a specified number (e.g., 1000), then process MLC flash data swap to SLC flash. The purpose is to keep frequently updated data in SLC.

191. Fig. 8 (802), (812) and paragraphs [0049] and [0050] explain that if an MLC block has an erase count that reaches 500, then move the data in the MLC block to a free block of SLC. Perform wear leveling for MLC based on erase count.

192. Thus, Moshayedi, under either logical or physical block construction, discloses the allocating by transfer controller function d) of claim element **[1g]**.

* * * * * * * * * * *

193. In summary, Moshayedi meets each limitation of '298 claim 1 with known hybrid SLC-MLC NAND flash memory module systems meeting elements [1a]-[1d], with its "read error" meeting the "data integrity test" claim element [1e] and its wear leveling embodiments meeting, under either logical or physical block construction, controller functions c) and d) of claim elements [1f] and [1g].

B. Claim 2 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [2Pre] The system of claim 1,

194. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [2] wherein the minimum quanta of address is equal to one block.

195. Paragraph [0005] explains that flash memory is organized into blocks, which are each divided into pages. Flash memory writes individual segments. Paragraph [0006] explains the use of L2P addressing using blocks. The controller maps "physical block address [to] a specific virtual block." (¶ [0037].)

196. Although a physical flash block may not be a minimum quanta of address to be written in standard NAND flash operation, it is the minimum quanta to be erased (*see* ¶ 28 above). A POSITA would understand that physical and logical blocks may not correspond in size. (*See* note 9 above.)

197. Thus, Moshayedi, in view of knowledge of the POSITA, discloses claim 2.

C. Claim 3 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [3Pre] The system of claim 1,

198. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [3] wherein the minimum quanta of address is equal to one page.

199. Paragraph [0005] explains that flash memory is organized into blocks, which are each divided into pages. Flash memory writes individual segments, which are normally pages.

200. It would be obvious to set the page as a minimum quanta of address to be written, as it is a design choice that has been adopted by the industry. (*See* ¶ 27 above.)

201. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 3.

D. Claim 4 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [4Pre] The system of claim 1,

202. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [4] wherein the MLC non-volatile memory module is NAND flash memory.

203. *See* paragraphs 166 and 167 above, which recite the use of MLC flash chips shown as Fig. 1(114). The chips identified at ¶ [0038] are NAND flash.

204. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 4.

E. Claim 5 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [5Pre] The system of claim 1,

205. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [5] wherein the SLC non-volatile memory module is NAND flash memory.

206. See paragraphs 166, 167 and 169 above, which recite the use of SLC flash chips shown as Fig. 1(112). The chips identified at ¶ [0038] are NAND flash.

207. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 5.

F. Claim 6 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [6Pre] The system of claim 1,

208. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM).

209. Although I do not see how RRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

210. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex. 1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

211. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 6.

G. Claim 7 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [7Pre] The system of claim 1,

212. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM).

213. I incorporate here my opinions set forth in paragraphs 209 and 210 above.

214. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 7.

H. Claim 8 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [8Pre] The system of claim 1,

215. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM).

216. Although I do not see how PCM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

217. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex. 1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

218. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 8.

I. Claim 9 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [9Pre] The system of claim 1,

219. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM).

220. I incorporate here my opinions set forth in paragraphs 216 and 217 above.

221. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 9.

J. Claim 10 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [10Pre] The system of claim 1,

222. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM).

223. Although I do not see how MAGRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

224. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of Chen '418 (Ex. 1054) (MRAM), and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

225. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 10.

K. Claim 11 Is Obvious Over Moshayedi in View of Knowledge of the POSITA.

1. [11Pre] The system of claim 1,

226. I incorporate here my opinions set forth in Section XI(A) for claim 1.

2. [11] wherein the controller causes the transfer of content on a periodic basis.

227. In my view, as explained at paragraph 73 above, the use of the term "periodic" in the DETAILED DESCRIPTION (6:30-35) is in tension with the p-e count thresholds recited in the same sentence. To the extent that "periodic" means "occasional," the actions disclosed in Moshayedi as reviewed at paragraphs 186-191 above are "occasional" or responsive to conditions such as reaching a threshold. (*E.g*. ¶ [0032] [reaching threshold erase count triggers a swap].)

228. Sutardja (Ex. 1042), another prior art reference that discloses various techniques of hybrid SLC-MLC memory systems wear leveling, describes that its system "determines whether [the] time to perform data shift analysis has arrived in step 514." (¶ [0148].) It was common knowledge of the POSITA that wear leveling functions may be performed at the time of a write (dynamic) or in the background when the device is not performing host-initiated read and write operations. (¶ 36 above, admitted art at 3:6-13.)

229. Thus, Moshayedi, in view of the knowledge of the POSITA, discloses claim 11.

XII. Ground 3: The Challenged Claims Are Obvious Over Sutardja in View of Knowledge of the POSITA

A. Claim 1 Is Obvious Over Sutardja

230. Sutardja (Ex. 1042) is a published reference that discloses a complete hybrid NAND flash system using MLC and SLC modules, with the ordinary L2P/FTL function of its controller, a variety of wear leveling schemes, and the capability of and compatibility for employing other known tools (exemplified in paragraphs 58-61 above) used to avoid premature NAND system failure.

231. I show here that Sutardja, in view of the knowledge of the POSITA, discloses the limitations recited in claim 1 as set forth in the groupings as printed in the '298 patent:

1. [1Pre] A system for storing data comprising:

232. I understand the question of whether a preamble is limiting is a question of claim construction. I do not provide an opinion as to whether the preamble of claim 1 is limiting.

233. To the extent the preamble is limiting, Sutardja's title is "HYBRID NON-VOLATILE SOLID STATE MEMORY SYSTEM."

2. [1a] at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks;

234. Fig. 2 shows a First Solid-State Nonvolatile Memory ("NVM") 204 and a Second Solid-State NVM 206. Paragraph [0108] explains the memories 204 and

206 each "may include single-level cell (SLC) flash memory or multi-level cell (MLC) flash memory". Claim 37 recites the system wherein the second memory includes single-level cell (SLC) flash memory, and the first memory includes multi-level cell (MLC) flash memory. As well-known to the POSITA, a flash memory includes many blocks, and the individually erasable block is the defining portion of NAND "flash" (*see* ¶ 28 above, Sutardja ¶ [0157] [block is a group of memory cells erased together].)

235. Paragraph [0106] describes an embodiment in which the first NVM 204 may include "inexpensive nonvolatile memory arrays and have a large capacity." This would be understood by a POSITA that the first NVM may include a plurality of MLC flash memory modules.

236. Thus, Sutardja discloses the first physical component of the claimed system, MLC module claim element **[1a]**.

3. [1b] at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks;

237. *See* paragraph 234 above, which recites the use of SLC flash memory, which may be included in the second memory. Paragraph [106] describes an embodiment in which the second NVM 206 "may have a greater write cycle lifetime while being more expensive and having a smaller capacity" than the first NVM 204.

This would be understood by a POSITA that the second NVM may be an SLC module

238. Thus, Sutardja discloses the second physical component of the claimed system, SLC module claim element **[1b]**.

4. [1c] a controller coupled to the at least one MLC non-volatile memory module and the at least one SLC non-volatile memory module wherein the controller is adapted to;

239. Fig. 3 shows a controller 252 that is coupled to both the first NVM 204 and the second NVM 206. Paragraph [118] explains this coupling as communication.

240. Paragraph [0009] explains that, in the generic (prior art) SSD system that controller 102 "reads or writes data to the flash memory 104."

241. Thus, Sutardja discloses the third component of the claimed system, controller claim element **[1c]**.

5. [1d] a) maintain an address map of at least one of the MLC and SLC non-volatile memory modules, the address map comprising a list of logical address ranges accessible by a computer system, the list of logical address ranges having a minimum quanta of addresses, wherein each entry in the list of logical address ranges maps to a similar range of physical addresses within either the at least one SLC non-volatile memory module or within the at least one MLC non-volatile memory module;

242. Sutardja uses NAND flash standard L2P mapping (see ¶¶ 32-34 above) as corroborated by Sutardja's specific use of that mapping in its wear leveling and other features. (*E.g.*, Abs., ¶ [0107] [map logical addresses corresponding to data

that will change relatively frequently to physical addresses in the second NVM and the logical addresses corresponding to data that will change relatively infrequently to physical addresses in the first NVM]. A POSITA would understand that Sutardja's various modules and sub-modules, such as the wear leveling module 260 (Figs. 3, 4A, 4B, 5, and 6), the write mapping module 356 (Fig. 4B), and the mapping module 465 (Fig. 6), which collect and respond to different defined conditions, use the basic NAND flash system of L2P mapping.

243. Thus, Sutardja in view of the knowledge of the POSITA discloses the ordinary NAND flash controller function a) of claim element **[1d]**.

6. [1e] b) determine if a range of addresses listed by an entry and mapped to a similar range of physical addresses within the at least one MLC non-volatile memory module, fails a data integrity test, and, in the event of such a failure, the controller remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module;

244. Figs. 5 (406) and 7D, explained at paragraphs [0134]-[0139] and [0151], disclose a "degradation" test of a physical location (range of physical addresses). Fig. 7D and paragraph [0151] explain the test as writing data to that location, reading back the data, writing the data to that location a second time (after a predetermined time), reading back the data a second time, and then comparing the data read back the first time and the data read back the second time, resulting in a "degradation value" for that physical address. One use for values that show

degradation (in a sense, failing that data integrity test), is for estimating the maximum life cycle for a block based on its degradation and for the wear leveling process to "normalize" wear. A POSITA understands that such normalization is wear leveling to avoid failure of one portion of the memory ahead of others. Paragraph [0139] discloses that using the number of write cycles remaining (from the estimated maximum), the wear leveling process may assign all new writes to another one of the memories rather than the one which is approaching the end of its useful life. Paragraph [0135] discloses periodically testing the first, lesser write cycle lifetime (MLC) NVM, so that paragraph [0139] would call for remapping to the second, greater write cycle lifetime (SLC) NVM if the degradation test on the MLC location has "failed," indicated by degradation values that have increased to a level showing suggesting near end-of-life for the location. Although this is not the test and immediate rewrite disclosed in the '298 patent Figs. 3a and 3b, a POSITA would understand that the Sutardja degradation test could result in remapping to SLC through the Sutardja processes directed to avoiding failure of some physical memory locations ahead of others.

245. Starting from that objective of a Sutardja system, it would also be obvious to a POSITA to apply the knowledge of other tools to avoid premature failure, for example, those discussed at paragraphs 58 to 61 above. These tools would include other end-of-life mitigations such as the known error management

techniques disclosed in Gavens. As shown in paragraphs 108 and 109 above, at least two Gavens post-write-read error management embodiments meet limitation [1e]. Those processes are compatible with Sutardja's processes shown at Figs. 7A-8; for example, Gavens post-write-read direction to SLC can be fit after Sutardja Fig. 7A (510).

246. Thus, Sutardja's degradation test of paragraphs [0134]-[0139] and [0151], applying knowledge of the POSITA, discloses the data integrity test controller function b) of element **[1f]**.

7. [1f] c) determine which of the blocks of the plurality of the blocks in the MLC and SLC non-volatile memory modules are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed; and

247. Sutardja discloses that the controller includes a wear leveling module that determines which physical blocks in the first solid-state nonvolatile memory and the second solid-state nonvolatile memory are written to and/or erased most frequently by maintaining a count of the number of times each physical block is written to and/or erased. Additionally, Sutardja discloses that the controller includes a write mapping module and a write monitoring module that determines which logical blocks in the first solid-state nonvolatile memory and the second solid-state nonvolatile memory are written most frequently by maintaining a count of the number of times each logical block is written.

248. Fig. 3 (260) and paragraphs [0110]-[0111] and [0121] disclose that wear leveling module 260 tracks the number times the write and/or erase operations are performed on each block in the first and second solid-state nonvolatile memories 204 and 206.

249. Paragraphs [0157] and [0159] disclose that wear leveling module tracks the number of erases experienced by each block of the first and second nonvolatile semiconductor memories.

250. Fig. 7C (520) and paragraph [0149] disclose, in the data shift analysis, determining if a number of write operations to a first block of the first NVM during a predetermined time is greater than a predetermined threshold.

251. Fig. 7E (548) and paragraph [0153] disclose, in the wear leveling analysis, determining if the wear level of the first NVM is greater than a predetermined threshold.

252. Paragraph [0128] discloses monitoring when the number of write operations performed on a block of the first solid-state nonvolatile memory 204 exceeds a predetermined threshold.

253. Fig. 4A (306) and paragraph [0129] discloses that the write monitoring module 306 tracks the logical addresses to which data is frequently written by measuring frequencies at which data is written to the logical addresses

254. Fig. 4B (356) and paragraph [0113] discloses that the write mapping module determines how frequently data is actually written to the logical addresses.

255. Fig. 7A (504) and (512) and paragraphs [0146] and [0147] disclose receiving write frequencies for logical addresses from the host and measuring actual write frequencies at which data is in fact written to the logical addresses.

256. Fig. 8 (604) and (612) and paragraphs [0155] and [0156] disclose receiving data related to write frequency for writing data to logical addresses from the host and measuring actual write frequencies at which data is in fact written to the logical addresses.

257. Thus, Sutardja discloses counting the number of accesses controller function c) of claim element **[1f]**.

8. [1g] d) allocate those blocks that receive the most frequent writes by transferring the respective contents of those blocks to the at least one SLC non-volatile memory module.

258. Sutardja discloses operation of a NAND flash system in a "flow" diagram (Figs. 7A-7E) that has several branches (Figs. 7B-7E) determining whether it is "[t]ime to perform" certain analyses (Fig. 7B [514]["data shift analysis"], [516]["degradation analysis"] and [518]["wear analysis"]). The base algorithm is shown in Fig. 7A: step 504 ("Receive Write Frequencies For Logical Addresses From Host"), step 506 ("Map Logical Addresses With Low Write Frequencies To First Memory"], step 508 ("Map Logical Addresses With High Write Frequencies

To Second Memory"), and step 510 ("Write Data According To Mapping"). This is the ordinary mapping and transfer to SLC memory of the data of the most frequently written LBAs ("hot data" of logical blocks) reviewed at paragraphs 46-48, 58 and 60 above. Paragraph [0146] discloses the controller mapping the logical addresses having write frequencies greater than a predetermined threshold to the second NVM (SLC).

259. In the "data shift" branch, Fig. 7C (522) and paragraph [0149] discloses that if the number of write operations to the first block of the first NVM (MLC) during a predetermined time is greater than the predetermined threshold, then map the logical addresses that correspond to the first block of the first NVM (MLC) to a second block of the second NVM (SLC).

260. In the "wear leveling" branch, Fig. 7E (550) and paragraph [0153] disclose that if the wear level of the first NVM (MLC) is greater than a predetermined threshold, then map all logical blocks to physical blocks of the second NVM (SLC).

261. Paragraph [0128] discloses that when the number of write operations performed on a first block of the first NVM (MLC) exceeds a predetermined threshold, the wear leveling module 260 of the controller may bias mapping of logical addresses that were originally mapped to the first block of the first NVM (MLC) to a second block of the second NVM (SLC).

262. Paragraph [0168] explains that the wear leveling module maintains a list of blocks that have surpassed their write cycle lifetime. The data that was previously stored in those blocks is written to other blocks.

263. Paragraph [0132] discloses that data can be remapped and moved from the second solid-state nonvolatile memory 206 to the first solid-state nonvolatile memory 204, and vice versa.

264. Fig. 8 (604) and (612) and paragraphs [0155] and [0156] disclose mapping the logical addresses having high write frequencies, greater than a predetermined threshold, to the second NVM (SLC) and measuring actual write frequencies at which data is in fact written to the logical addresses, and updates the mapping.

265. Paragraph [0164] discloses: When a write request for a logical address arrives, the wear leveling module determines if the logical address is already mapped to a physical address and if the write requires an erase of the block. If so, the wear leveling module determine if there are any unused blocks with lower wear levels. If so, the wear leveling module directs the write to the unused block having the lowest wear level.

266. Thus, Sutardja under either the logical or physical block construction, discloses the allocate by transfer controller function d) claim element **[1g]**.

* * * * * * * * * * *

267. In summary, Sutardja meets each limitation of '298 claim 1 with known hybrid SLC-MLC NAND flash memory module systems meeting elements [1a]-[1d], with its "degradation test" meeting the "data integrity test" claim element [1e] and its wear leveling embodiments meeting controller functions c) and d) claim elements [1f] and [1g].

B. Claim 2 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [2Pre] The system of claim 1,

268. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [2] wherein the minimum quanta of address is equal to one block.

269. As well-known to the POSITA, a flash memory includes many blocks, and the individually erasable block is the defining portion of NAND "flash" (*see* ¶ 28 above, Sutardja ¶ [0157] [block is a group of memory cells erased together].)

270. Although a physical flash block may not be a minimum quanta of address to be written in standard NAND flash operation, it is the minimum quanta to be erased (*see* ¶ 28 above). A POSITA would understand that physical and logical blocks may not correspond in size. (*See* note 9 above.)

271. Thus, Sutardja, in view of knowledge of the POSITA, discloses claim 2.

C. Claim 3 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [3Pre] The system of claim 1,

272. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [3] wherein the minimum quanta of address is equal to one page.

273. It would be obvious to set the page as a minimum quanta of address to be written, as it is a design choice that has been adopted by the industry. (*See* ¶ 27 above.)

274. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 3.

D. Claim 4 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [4Pre] The system of claim 1,

275. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [4] wherein the MLC non-volatile memory module is NAND flash memory.

276. *See* paragraphs 234 and 235 above, which recite the use of MLC flash memory (¶ [0108].) This would be understood to be NAND flash as at least disclosed at paragraphs [0103] and [0172].

277. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 4.

E. Claim 5 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [5Pre] The system of claim 1,

278. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [5] wherein the SLC non-volatile memory module is NAND flash memory.

279. *See* paragraphs 234 and 235 above, which recite the use of SLC flash memory (¶ [0108]). This would be understood to be NAND flash as at least disclosed at paragraphs [0103] and [0172].

280. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 5.

F. Claim 6 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [6Pre] The system of claim 1,

281. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [6] wherein the MLC non-volatile memory module is resistive random-access memory (RRAM).

282. Although I do not see how RRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

283. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex. 1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory).

284. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 6.

G. Claim 7 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [7Pre] The system of claim 1,

285. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [7] wherein the SLC non-volatile memory module is resistive random-access memory (RRAM).

286. I incorporate here my opinions set forth in paragraphs 282 and 283 above.

287. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 7.

H. Claim 8 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [8Pre] The system of claim 1,

288. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [8] wherein the MLC non-volatile memory module is phase change memory (PCM).

289. Although I do not see how PCM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

290. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of De Ambroggi (Ex. 1053) (RRAM, PCM) and Kund (Ex. 1055) (alternative technologies having advantages over flash memory). Sutardja paragraphs [0103], [0104], and [0172] also mention PCM chips.

291. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 8.

I. Claim 9 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [9Pre] The system of claim 1,

292. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [9] wherein the SLC non-volatile memory module is phase change memory (PCM).

293. I incorporate here my opinions set forth in paragraphs 289 and 290 above.

294. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 9.

J. Claim 10 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [10Pre] The system of claim 1,

295. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [10] wherein the SLC non-volatile memory module is magnetic random-access memory (MAGRAM).

296. Although I do not see how MAGRAM memory can simply be substituted for NAND flash memory, particularly for the "hot block" operations required by claim 1, to the extent that this claim is enabled by the disclosures in the specification, it is obvious in view of the assumed background knowledge.

297. This background knowledge is corroborated by the Examiner's rejection (¶ 78 above, citing Gorobets '179 (Ex. 1044) in view of Chen '418 (Ex. 1054) (MRAM), and Kund (Ex. 1055) (alternative technologies having advantages over flash memory). Sutardja paragraph [0172] also mentions "magnetic RAM."

298. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 10.

K. Claim 11 Is Obvious Over Sutardja in View of Knowledge of the POSITA.

1. [11Pre] The system of claim 1,

299. I incorporate here my opinions set forth in Section XII(A) for claim 1.

2. [11] wherein the controller causes the transfer of content on a periodic basis.

300. In my view, as explained at paragraph 73 above, the use of the term "periodic" in the DETAILED DESCRIPTION (6:30-35) is in tension with the p-e count thresholds recited in the same sentence. To the extent that "periodic" means "occasional," the actions disclosed in Sutardja as reviewed at paragraphs 259-265 above are "occasional" or responsive to conditions such as reaching a threshold.

301. As reviewed at paragraphs 36, 227 and 228 above, it was common knowledge of the POSITA that wear leveling functions may be performed at the time of write (dynamic) or in the background when the device is not performing host-initiated read and write operations. (Admitted art at 3:6-13.)

302. Sutardja specifically discloses that its system "determines whether [the] time to perform data shift analysis has arrived in step 514." (¶ [0148].) (*Also* ¶¶ [0135][periodic degradation testing] and [0167]["At various times, such as periodically, the wear leveling module analyzes the wear levels of the blocks, and remaps relatively frequently rewritten logical addresses to blocks with low wear levels"].)

303. Thus, Sutardja, in view of the knowledge of the POSITA, discloses claim 11.

XIII. Conclusion

304. For the reasons given above, claims 1-11 of the '298 patent are invalid.

305. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further 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 Section 1001 of Title 18 of the United States Code.

Dated: November 25, 2024

Carl Sechen

Carl Sechen, Ph.D.