

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

PHISON ELECTRONICS CORPORATION,
Petitioner,

v.

VERVAIN, LLC,
Patent Owner.

IPR2025-00212
U.S. Patent No. 8,891,298

**DECLARATION OF SUNIL P. KHATRI
IN SUPPORT OF PATENT OWNER'S PRELIMINARY RESPONSE**

TABLE OF CONTENTS

EXHIBIT LIST iv

I. INTRODUCTION1

II. BACKGROUND AND QUALIFICATIONS1

III. SCOPE OF ASSIGNMENT AND MATERIALS CONSIDERED14

IV. PERSON OF ORDINARY SKILL IN THE ART15

V. GENERAL BACKGROUND OF THE RELEVANT TECHNOLOGY17

 A. SLC and MLC Flash20

 B. Address Table21

 C. Data Integrity Tests22

 D. Hot and Cold Data23

VI. THE 298 PATENT23

VII. CLAIM CONSTRUCTION26

 A. “blocks”27

VIII. Petitioner’s invalidity arguments33

 A. OVERVIEW OF THE PRIOR ART33

 1. Gavens (Ex. 1045)33

 2. Moshayedi (Ex. 1043)42

 3. Sutardja (Ex. 1042)45

 B. GROUND 1: GAVENS DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 148

 1. [1b]: “SLC non-volatile memory module comprising a plurality of *individually addressable blocks*”48

2.	[1c]: “a controller coupled to” an “MLC non-volatile memory module” and an “SLC non-volatile memory module”	48
3.	[1f]: “c) determine which of the blocks of the plurality of blocks...are accessed most frequently by <u>maintaining a count</u> of the number of times each one of the blocks is accessed”	52
4.	[1f]: “ <i>controller</i> adapted to” ([1c]) ... “maintain[] a count of the number of times each of the blocks is accessed”	53
5.	[1g]: “allocate those blocks that receive the most frequent writes by <u>transferring the respective contents of those blocks</u> to the at least one SLC non-volatile memory module”	56
C.	GROUND 2: MOSHAYEDI DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 1.....	58
1.	[1b]: “SLC non-volatile memory module comprising a plurality of <i>individually addressable blocks</i> ”	58
2.	[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 <i>minimum quanta of addresses,...</i> ”	58
3.	[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 <u>MLC non-volatile memory module</u> , <i>fails a data integrity test</i> , and, in the event of such a failure, the controller <u>remaps the entry to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module”</u>	59
4.	[1g]: “d) allocate those blocks that receive the most frequent writes by <i>transferring the respective contents of those blocks</i> to the at least one SLC non-volatile memory module”	61

D.	GROUND 3: SUTARDJA DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 1.....	65
1.	[1b]: “SLC non-volatile memory module comprising a plurality of <i>individually addressable blocks</i> ”	66
2.	[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 <i>minimum quanta of addresses,...</i> ”	66
3.	[1e]: “b) determine if <i>a range of addresses listed by an entry</i> 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 <i>remaps the entry</i> to the next available equivalent range of physical addresses within the at least one SLC non-volatile memory module”	67
4.	[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”.....	69
5.	[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”	71
E.	Dependent Claims 2-11.....	75
IX.	CONCLUSION.....	76

EXHIBIT LIST

Exhibit No.	Description
2001	Declaration of Dr. Sunil Khatri
2002	Chen et al., <i>Ultra MLC Technology Introduction</i> , Advantech Technical White Paper (Oct. 5, 2012) (“Chen”)
2003	Excerpts from Micheloni et al., <i>Inside NAND Flash Memories</i> (1 st ed. 2010) (“Micheloni”)
2004	[RESERVED]
2005	[RESERVED]
2006	<i>Vervain v. Western Digital Corp. et al.</i> , No. 6:21-cv-488-ADA, Dkt. 41 (W.D. Tex. Jan. 24, 2022) (Claim Construction Order)
2007	[RESERVED]
2008	[RESERVED]
2009	[RESERVED]
2010	[RESERVED]
2011	[RESERVED]
2012	[RESERVED]
2013	[RESERVED]
2014	[RESERVED]
2015	[RESERVED]
2016	[RESERVED]
2017	[RESERVED]

Exhibit No.	Description
2018	<i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-CV-259-ADA, Dkt. 16 (W.D. Tex. Jul. 15, 2024) (agreed scheduling order)
2019	[RESERVED]
2020	[RESERVED]
2021	“Judge Albright Patent FAQ,” https://www.txwd.uscourts.gov/for-attorneys/judge-albright-courtroom-faq/ . Accessed December 18, 2024)
2022	[RESERVED]
2023	[RESERVED]
2024	[RESERVED]
2025	[RESERVED]
2026	[RESERVED]
2027	[RESERVED]
2028	[RESERVED]
2029	<i>Phison Electronics Corp. v. Vervain, LLC</i> , PGR2024-00047, Paper 2 (P.T.A.B. Aug. 27, 2024) (petition for post grant review of U.S. Patent No. 11,830,546)
2030	[RESERVED]
2031	Preliminary Claim Constructions for <i>Vervain, LLC v. Phison Electronics Corporation</i> , No.1:24-cv-259-ADA and <i>Vervain, LLC v. Kingston Technology Company., et al.</i> , No. 1:24-cv-254-ADA – email from Mr. Brown received on February 5, 2025.
2032	Markman Hearing Transcript dated February 6, 2025, for <i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-cv-259-ADA and

Exhibit No.	Description
	<i>Vervain, LLC v. Kingston Technology Company., et al.</i> No. 1:24-cv-254-ADA.
2033	<i>Vervain v. Western Digital Corp. et al.</i> , No. 6:21-cv-00488-ADA, Dkt. 180 (W.D. Tex. Jul. 26, 2023) (Redacted Copy of Order on the Pending Motions and Motions <i>In Limine</i>).
2034	Phison’s Exhibit A1 Claim Chart, served with its Final Invalidity Contentions on February 28, 2025 in <i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-CV-259-ADA (W.D. Tex.) (charting U.S. Patent No. 8,891,298 as obvious under 35 U.S.C. § 103 based on U.S. Patent No. 8,634,240 (“Gavens”))
2035	Phison’s Exhibit A3 Claim Chart, served with its Final Invalidity Contentions on February 28, 2025 in <i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-CV-259-ADA (W.D. Tex.) (charting U.S. Patent No. 8,891,298 as obvious under 35 U.S.C. § 103 based on U.S. Patent Appl. Pub. No. 2009/0327591 (“Moshayed”))
2036	Phison’s Exhibit A2 Claim Chart, served with its Final Invalidity Contentions on February 28, 2025 in <i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-CV-259-ADA (W.D. Tex.) (charting U.S. Patent No. 8,891,298 as obvious under 35 U.S.C. § 103 based on U.S. Patent Appl. Pub. No. 2008/0140918 (“Sutardja”))
2037	<i>Vervain, LLC v. Phison Electronics Corp.</i> , No. 1:24-cv-00259, Dkt. 53 (W.D. Tex. Jan. 29, 2025) (Joint Notice Extending Deadlines)
2038	Docket Sheet for <i>Vervain, LLC v. Phison Electronics Corporation</i> , No. 1:24-CV-259-ADA (W.D. Tex.). Accessed March 11, 2025.
2039	<i>Phison Electronics Corp. v. Vervain, LLC</i> , PGR2024-00047, Paper 9 (P.T.A.B. Mar. 17, 2024) (Decision Denying Institution of Post-Grant Review of U.S. Patent No. 11,830,546)
2040	<i>Vervain, LLC v. Western Digital Corp. et al.</i> , No. 6:21-cv-00488-ADA, Dkt. No. 183, Pretrial Hearing Redacted Transcript (W.D. Tex. Aug. 7, 2023)

DECLARATION OF SUNIL P. KHATRI, PH. D

I, Sunil P. Khatri, do hereby declare as follows:

I. INTRODUCTION

1. I have been retained on behalf of Vervain, LLC (“Vervain”), and its counsel, McKool Smith, P.C., as an expert in this proceeding. I am personally knowledgeable about the matters stated herein and am competent to make this declaration.

2. I understand that Vervain will submit this Declaration in connection with the Patent Owner’s Preliminary Response in IPR2025-00212, which I have been informed is a *inter partes* review (IPR) proceeding challenging the patentability of the claims of U.S. Patent No. 8,891,298 (“the 298 Patent”).

3. I receive compensation at an hourly rate of \$800 per hour for my time working on this matter, plus expenses. I have no financial interest in Vervain or in the patents involved in this litigation, and my compensation is not dependent on the outcome of this litigation. The conclusions I present are due to my own judgment.

II. BACKGROUND AND QUALIFICATIONS

4. I have over thirty-eight years of experience with electronics, electrical engineering, and computer engineering. A copy of my latest curriculum vitae (CV) is attached hereto as Appendix A and provides further details regarding my

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

background and qualifications. During my career, I have acquired extensive knowledge and experience with VLSI circuits, computer architecture, testing, computer-aided design (CAD) algorithms and algorithm acceleration, logic synthesis, semiconductor memory, redundancy, synchronous and asynchronous circuits, and related software and hardware topics. Most relevant to the Asserted Patents, my technical expertise includes extensive work with semiconductor memory devices such as DRAM, SRAM and flash. My work with semiconductor memory devices has included work on 3D integration and novel ring-based memory architectures, power and speed tradeoffs using selective body bias, architectures and circuit approaches for processing-in-memory, radiation hardening analysis for memories, the use of flash transistors for designing logic circuits (such as ternary Content-addressable Memories (CAMs), Field Programmable Gate Arrays (FPGAs), and traditional binary-valued as well as ternary-valued digital logic, secure logic circuits, analog circuits like Finite Impulse Response (FIR) filters, Discrete Fourier Transform (DFT) circuits, and in-memory mixed-signal computing architectures like Convolutional Neural Networks (CNNs)), and clocking and source-synchronous design. I recently was awarded a research grant by the Air Force Research Laboratory (AFRL) in Rome, NY, to conduct research in secure digital circuits using flash-based digital design approaches. Another recent research grant

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

from the National Science Foundation (NSF) funds research on the use of flash transistors for general purpose mixed-signal arithmetic. Additionally, I have lead-authored a book chapter on the use of flash transistors in novel Very Large Scale Integrated (VLSI) design applications. My MS thesis involved designing a memory interface for a multi-threaded Reduced Instruction Set Computing (RISC) microprocessor.

5. The following describes some of my relevant experience. I earned my Bachelor of Science in Electrical Engineering in 1987 from the Indian Institute of Technology, Kanpur, India. After graduating with my B.S. degree, I was a candidate for a Master of Science degree in Electrical and Computer Engineering at the University of Texas from 1987–89. At the University of Texas, I held the Microelectronics and Computer Development (MCD) Fellowship from 1987–89. I also conducted my M.S. research and wrote my thesis on the design of the METRIC memory interface and memory system. METRIC was one of the first super-scalar processors that was developed in the world. I earned an M.S. degree in 1989 from the University of Texas, Austin.

6. After leaving the University of Texas, I worked at Motorola Inc. from 1989–93 as a design engineer for the MC88110 reduced instruction set computing (RISC) microprocessor team. My duties included the design of digital and analog

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

circuitry, test logic and circuits, JTAG boundary scan design, input/output driver design, and clock phase-locked loop (PLL) logic. During my time at Motorola, I was independently responsible for the design of the factory test controller of the MC88110 microprocessor. I performed all attendant tasks in a “vertical” VLSI design methodology, which included high-level modeling, circuit and layout design and verification, as well as global and detailed routing. I also helped in the design of the Translation Lookaside Buffer (TLB) unit, which included a static random-access memory (SRAM) block.

7. In 1999, I earned a Doctor of Philosophy degree in Electrical Engineering and Computer Sciences from the University of California, Berkeley. While at Berkeley, I held the California Microelectronics (MICRO) Fellowship in 1993.

8. I joined the faculty at the University of Colorado, Boulder, in 2000 as an Assistant Professor of Electrical and Computer Engineering. At the University of Colorado my research focused on VLSI logic design automation, VLSI layout design automation, and VLSI design methodologies to address Deep Submicron (DSM) issues such as crosstalk and power.

9. I joined the faculty at Texas A&M University in 2004 as an Assistant Professor in Electrical and Computer Engineering. In 2010 I was promoted to

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

Associate Professor in Electrical and Computer Engineering. In 2015, I was promoted to full Professor in Electrical and Computer Engineering. My research focuses on three primary areas: the first is computer systems, including computer architecture from the circuits up, and algorithm acceleration using GPUs, FPGAs and custom ICs. The second is logic and its applications, while the third area consists of interdisciplinary extensions of the first two. Some specific recent research topics include circuit design using floating gate (flash) devices, wireless power delivery and battery-less electronic systems, machine learning architectures, secure computing approached from both a hardware and software perspective, a formal framework for designing and verifying cryptographic hash functions, and extreme high-speed serial interfaces for Artificial Intelligence / Machine Learning (AI/ML) systems such as those used in Large Language Models (LLMs) like ChatGPT. One of my new focus areas is intelligent and secure computing viewed from the hardware (circuit) as well as the software levels. I have conducted research on tamper-proof memory techniques, as well as multi-row read architectures for SRAM, DRAM and flash memory arrays.

10. At Texas A&M I teach classes that cover memories extensively, featuring a thorough discussion of sense amplifiers, row and column decoders, and different types of memory circuits. For example, in Electrical and Computer

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

Engineering (ECEN) 752 “Advances in VLSI Circuit Design,” a graduate level course, I cover all aspects of VLSI design, including memory design for various types of memories. In this course, (and also in ECEN 449/749 “Microprocessor System Design,”), I cover memories, including flash memories, the design of flash memory cells, the organization of memories into multiple banks, and the division of data across multiple banks of memory, as well as other design techniques that can be used to optimize and manage memories. The ECEN 449/749 course is attended by both undergraduate (ECEN 449) and graduate (ECEN 749) students. In ECEN 454 “Digital Circuit Design,” a senior undergraduate course, I cover circuit design techniques for memory in detail. The Ph.D. thesis dissertation topic for 3 of my current Ph.D. students covers the use of flash memory transistors for analog and digital applications. The Ph.D. thesis of one of my recent doctoral students dealt with the use of flash transistors to design logic circuits. The research of a recent M.S. student entailed a new ring-based source synchronous architecture for 3D DRAM technologies, which has been published at a conference and in a journal and is being submitted for dissemination as a research monograph. In the past, I have conducted research into new topologies for efficient memory redundancy as part of a course project for my graduate course.

11. Since 2000, I have earned 24 research contracts from funders including Intel, the National Science Foundation, the National Security Agency, Altera Corporation, the National Center for Atmospheric Research, National Semiconductor Corporation, and several private sources. The total amount for these research grants is \$17.53 million, of which my portion is \$2.85 million.

12. I have a total of over 289 peer-reviewed publications. Among these papers, five received a best paper award, while seven others received best paper nominations (including one journal best paper nomination). An additional six journal papers and three conference paper are currently undergoing peer review. I have co-authored nine research monographs, one edited research monograph, and four book chapters. Additionally, I have seven awarded U.S. Patents (two of which were filed during my tenure at Texas A&M), two filed provisional U.S. Patents, and another U.S. Patent which is currently under review and was also submitted during my tenure at Texas A&M. I have co-authored one invited journal paper and 13 invited conference or workshop papers (including one from Design Automation Conference (DAC) and one from Allerton). Moreover, I was invited to serve as a panelist at a conference seven times and have presented four conference tutorials. I received the “Outstanding Professor Award” in the ECE Department at Texas A&M University in 2007 and also in 2020. In 2009 and in 2019, I received a university-level and a

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

college-level teaching award respectively, at Texas A&M. My H-index is 36 (per Google Scholar).

13. Since 2003, I have published numerous research monographs, journal papers, and conference papers on flash transistors and memory systems, as detailed in my CV attached in Appendix A. A few papers on relevant subject areas authored or co-authored by me include:

- “A Novel Mixed-Signal Flash-based Finite Impulse Response (FFIR)”, Lee, Khatri, Ghayeb. Proc. of the 30th ACM Asia and South Pacific Design Automation Conference (ASPDAC) 2025. Tokyo, Japan, January 20-23, 2025. pp 1216-1222.
- *"Flash - A "Forgotten" Technology in VLSI Design"*. Khatri, Vrudhula, Abusultan, Bharathi, Chu, Lee, Scott, Singh, Wagle. Appears "Frontiers of Quality Electronic Design (QED)". Springer publishers, Sept 2022. Pages 67-136. Ali Iranmanesh, Ed. ISBN: 978-3-031-16344-9;
- *"Flash – An Overlooked Technology in VLSI Design"*, Khatri, Vrudhula. Monday Tutorial to be presented at the ACM/CEDA Design Automation Conference (DAC), San Francisco, CA, June 2024;
- *"A Mixed-Signal Quantized Neural Network Accelerator Using Flash Transistors"*, Scott, Lee, Khatri, Vrudhula. IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 71, no. 3, pp. 1025-1038, March 2024;

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

- *"An ASIC Accelerator for QNN With Variable Precision and Tunable Energy-Efficiency"*, Wagle, Singh, Khatri Vrudhula. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Jan 2024;
- *"A Digital Low Dropout (LDO) Voltage Regulator Using Pseudoflash Transistors"*, Lee, Khatri. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 31, no. 12, pp. 1960-1969, Dec. 2023;
- *"A Novel ASIC Design Flow using Weight-tunable Binary Neurons as Standard Cells"*, Wagle, Singh, Khatri, Vrudhula. IEEE Transactions on Circuits and Systems I (Regular Papers), 69(7): 2968-2981 (Sept 2022);
- *"CIDAN-XE: Computing in DRAM with Artificial Neurons"*, Singh, Wagle, Khatri, Vrudhula. Frontiers in Electronics, section Integrated Circuits and VLSI. vol. 3, Feb 2022;
- *"An Extremely Low-Voltage Floating Gate Artificial Neuron"*, Scott, Khatri. 2023 IEEE International Conference on Circuits and Systems (ISCAS), May 21-25, 2023. Monterey, CA;
- *"A Novel Pseudo-Flash Based Digital Low Dropout (LDO) Voltage Regulator"*, Lee, Khatri, Vrudhula. 2023 International Symposium on Quality Electronic Design (ISQED), April 5-7 2023. San Francisco, CA;
- *"A Flash-based Digital to Analog Converter for Low Power Applications"*, Scott, Khatri. IEEE 40th International Conference on Computer Design (ICCD), October 2022. Lake Tahoe, CA. pp 1-8;

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

- *"A Flash-based Current-mode IC to Realize Quantized Neural Networks"*, Scott, Lee, Khatri, Vrudhula. IEEE Design Automation and Test in Europe Conference, Mar 2022, Antwerp, Belgium. pp 1029-1034;
- *"CIDAN: Computing in DRAM with Artificial Neurons"*, Singh, Wagle, Vrudhula, Khatri. 39th IEEE International Conference on Computer Design (ICCD) 2021, pp 349-356, Oct 2021, virtual conference;
- *"A Statistical Methodology for Post-Fabrication Weight Tuning in a Binary Perceptron"*, Azari, Wagle, Khatri, Vrudhula. IEEE 21st International Symposium on Quality Electronic Design (ISQED) 2020. Mar 25-26, 2020, Santa Clara, CA. pp 141-148;
- *"Threshold Logic in a Flash"*, Wagle, Singh, Yang, Khatri, Vrudhula. 37th IEEE International Conference on Computer Design (ICCD), 2019. Abu Dhabi, UAE, pp 550-558. Nov 17-20, 2019;
- Fast, Ring-based Design of 3D Stacked DRAM, IEEE Transactions on Very Large Scale Integrated Circuits (TVLSI), IEEE Transactions on Very Large Scale Integration (VLSI) Systems. Vol 27 number 8, Aug 2019. pp 1731-1741.;
- Fast, Ring-Based Design of 3D Stacked DRAM, IEEE International Conference on Computer Design 2017: pp 665-672;
- Selective Forward Body Bias for High Speed and Low Power SRAMs, Journal of Low Power Electronics, Vol. 5, No. 2, Aug. 2009, pp. 185-95;

Declaration of Sunil P. Khatri, Ph. D.

IPR2025-00212

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

- Low Power and High Performance SRAM Design using Bank-based Selective Forward Body Bias, IEEE/ACM Great Lakes Symposium on VLSI, May 10-12, 2009, Boston, MA, pp. 441-44;
- Modeling Dynamic Stability of SRAMs in the Presence of Single Event Upsets (SEUs), IEEE International Symposium on Circuits and Systems, May 18-21, 2008, Seattle, WA, pp. 1788-91;
- "Design of a Flash-based Circuit for Multi-valued Logic", Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI) 2017, pp 41-46, May 10-12, 2017. Banff, Canada.
- "SAT-Based Optimization for Flash-Based Digital Designs", IEEE/ACM Design Automation Conference (DAC), Jun 18-22 2017, Austin, TX.
- "A Flash-based Digital Circuit Design Flow", IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2016, Austin, TX, Nov 2016.
- "Implementing low power digital circuits using flash devices", 2016 IEEE 34th International Conference on Computer Design (ICCD), pp 109-116, Oct 3-5, 2016, Phoenix, AZ.
- "Exploring Flash Devices to Implement Digital Circuits", IEEE/ACM Design Automation Conference (DAC), June 2016, Austin, TX.
- FTCAM: An Area-efficient Flash-based Ternary CAM Design, IEEE Transactions on Computers, Vol. 65, No. 8, Aug. 2016, pp. 2652-58;
- An Area-efficient Ternary CAM Design Using Floating Gate Transistors, IEEE International Conference on Computer Design, Oct. 19-22, 2014, Seoul, S. Kor., pp. 55-60; and

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

- A Fast Ternary CAM Design for IP Networking Applications, International Conference on Computer Communications and Networks, October 22, 2003, Dallas, TX, pp. 434-39 (awarded best paper).
- U.S. Patent 12057831: “Threshold logic gates using flash transistors”, Vrudhula, Khatri. Issued August 2024

14. In addition to my work on the papers listed above, I have also served as an editor for IEEE Transactions on Computers, ACM Transactions on Design Automation of Electronic Systems, and MDPI Journal of Electronics.

15. I have served as EDA Track Co-Chair for ICECS 2014, Panel Chair for TexasWISE 2014, Track Co-Chair (VLSI Systems, Applications and Computer Aided Design track) for ICECS 2013, Poster Session Chair for TexasWISE 2013, Advisory Committee for HotPI 2013, Panel Session Chair for SLiP 2013, Track Chair (Logic track) for ICCAD 2009-10, 2015-17, Track Chair (logic track) for DAC 2016-17, General Chair for IWLS 2009, Technical Program Chair for IWLS 2008, Track Co-Chair, Computer Aided Network DEsign (CANDE) Track, for ISCAS 2008-10, Track Co-Chair, Test and Methodologies Track, for ICCD 2007, Panel Chair for ITSW 2009, Publicity Co-Chair for GLS-VLSI 2009, and as a member of the TPC for several conferences.

16. I am generally familiar with the analysis of patents. I am a named inventor on the following U.S. Patents:

- Data Processing System Having Serial Self Address Decoding and Method of Operation, United States Patent No. 5,347,523, issued September 13, 1994;
- Circuit Identifier for Use with Focused Ion Beam Equipment, United States Patent No. 5,408,131, issued April 18, 1995 (“the ’131 patent”);
- Driver Circuit with Self-Adjusting Impedance Matching, United States Patent No. 5,448,182, issued September 5, 1995 (“the ’182 patent”);
- Circuit Identifier for Use with Focused Ion Beam Equipment, United States Patent No. 6,156,579, issued December 5, 2000 (“the ’579 patent”);
- Datapath Design Methodology and Routing Apparatus, United States Patent No. 6,598,215, issued July 22, 2003;
- Low Power Reconfigurable Circuits with Delay Compensation, United States Patent No. 7,880,505, issued February 1, 2011.
- U.S. Patent 12057831: “*Threshold logic gates using flash transistors*”, Vrudhula, Khatri. Issued August 2024.

17. The ’831 patent discloses a method to design logic gates which implement a class of logic functions called threshold functions, using flash transistors.

18. The ’131 and ’579 patents are directed to an identification means for redundant circuits that distinguishes said circuits by respective function. This allows for identification by focused ion beam equipment, which can then repair, replace, or

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

supplement circuits as necessary. These patents disclose a scheme for replacing defective cells or circuits within a larger circuit.

19. The '182 patent relates to a driver circuit capable of switching from one driving impedance to a second in response to the output signal of a first driver portion reaching a predetermined voltage. It discloses a driver circuit capable of adjusting circuit configurations depending upon the current output state.

III. SCOPE OF ASSIGNMENT AND MATERIALS CONSIDERED

20. I have been retained by Vervain (“Patent Owner”) to provide an explanation to the Board regarding the invention described and claimed in the 298 Patent. I understand Petitioner Phison Electronics Corporation (“Phison” or “Petitioner”), has filed a petition for inter partes review, No. IPR2025-00212, against the 298 Patent. I have been retained to provide my opinions regarding various technical issues relating to the validity of the challenged patent, including with regard to the validity of the challenged patent over the prior art references identified by Phison’s petition.

21. In preparing this Declaration, I am relying on my own knowledge and expertise as well as the following documents:

- U.S. Patent No. 8,891,298 (Ex. 1001) to G.R. Mohan Rao (“298 Patent”);

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

- Phison's petition for inter partes review (IPR2025-00212, Paper No. 2), as well as the declaration of Dr. Carl Sechen (Ex. 1002) submitted in support thereof;
- The exhibits cited in Phison's petition for inter partes review (IPR2025-00212, Paper No. 2);
- The alleged prior art references relied upon in Phison's petition;
- Any other documents discussed within this Declaration.

IV. PERSON OF ORDINARY SKILL IN THE ART

22. When interpreting a patent, I understand that it is important to view the disclosure and claims of that patent from the level of ordinary skill in the relevant art at the time of the invention, which I have been asked to initially consider as the 2011 time frame, including and up to July 19, 2011, which is the filing date of Provisional Application No. 61/509,257, which I am informed is a provisional application to which the 298 Patent claims priority. I am informed that the 298 Patent was filed April 25, 2012, and my opinions and analysis are still applicable if the April 25, 2012 date is considered as the time of the invention. My opinion of the level of ordinary skill in the art with regard to the challenged patent is based on my personal experience working and teaching in the fields of electrical engineering and computer science, including work with memory technologies, my knowledge of the background and education of colleagues and others working in that general field as

of and for several years prior to the 2011-2012 time frame, my study of the challenged patent, and its file history, and my knowledge of:

- The level of education and experience of persons actively working in the field at the time the subject matter at issue was developed;
- The types of problems encountered in the art at the time the subject matter was developed;
- The prior art patents and publications;
- The activities of others working in that same technical field;
- Prior art solutions to the problems addressed by the relevant art; and
- The sophistication of the technology at issue in this case.

23. In determining the level of ordinary skill in the art, I also considered the following factors: (1) the sophistication of the relevant technology; (2) the rapidity with which innovations are made in that field; and (3) the educational level of active workers in that field. It is my further understanding that these factors are not exhaustive and are merely a useful guide to determining the level of ordinary skill in the art.

24. Taking the above factors into account, in my opinion a person of ordinary skill in the art (POSA) in the technology field of the 298 Patent would be a person with at least a Bachelor of Science degree in electrical engineering, computer engineering, or a closely related field, with at least 3-5 years of experience in the design of non-volatile memory devices. An individual with an advanced degree in

a relevant field would require less experience in the design of non-volatile memory devices.

25. I understand that Phison's expert, Dr. Carl Sechen, provided a different definition for the POSA. I understand that Dr. Sechen's definition is as follows:

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.

Ex. 1002, ¶64.

26. For the limited purpose of this declaration, I rely on the definition of the POSA provided by Dr. Sechen. My opinions in this declaration would remain unchanged if I had adopted my definition of the POSA. At the time of the invention, I qualified as a POSA under either definition.

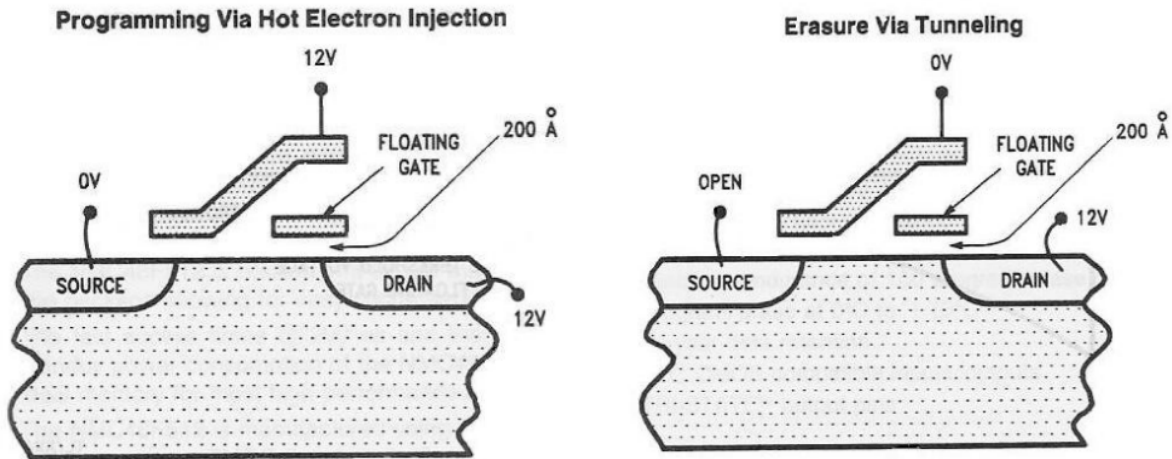
V. GENERAL BACKGROUND OF THE RELEVANT TECHNOLOGY

27. Volatile memory technologies, such as static random access memory ("SRAM") and dynamic random access memory ("DRAM"), lose their contents when power is turned off. However, persistent (non-volatile) memory is needed for many applications, such as storage for photos in a digital camera, bootable code or

settings in desktops, laptops or cellular phones, and a wide range of other applications. Non-volatile memories (e.g., thumb drives, hard drives, and compact discs) can store information even after the system is powered off. Traditionally, media such as hard disks, floppy disks, compact discs, or magnetic tapes was used for non-volatile storage in computing systems, but these media are large, bulky, and slow, and use a large amount of power. Flash memory is a specific type of non-volatile memory, where data is stored in “blocks” of “pages.” Flash memory chips have come to be used for persistent data storage in a wide range of applications. Data stored in flash memory persists across power on/off cycles and has a small size, high performance, and low power consumption.

28. Flash memory uses a special type of transistor called a floating gate transistor. Such a transistor, in addition to a “control gate”, also has a “floating gate”. In a floating gate transistor, charge is stored on the floating gate, which is an isolated conductor. The floating gate cannot discharge, since it is in a high-impedance state. The charge on the floating gate controls the current flowing between the source and drain of the floating gate transistor, by altering the threshold voltage of the floating gate transistor. This current allows the user to determine the value stored in the cell. The charge stored in the floating gate is a proxy for the logical value stored in the flash memory cell that is implemented using the floating gate transistor.

29. To erase the cell, a reverse voltage is applied between the drain and the control gate. Charge is then dissipated through Fowler-Nordheim tunneling.



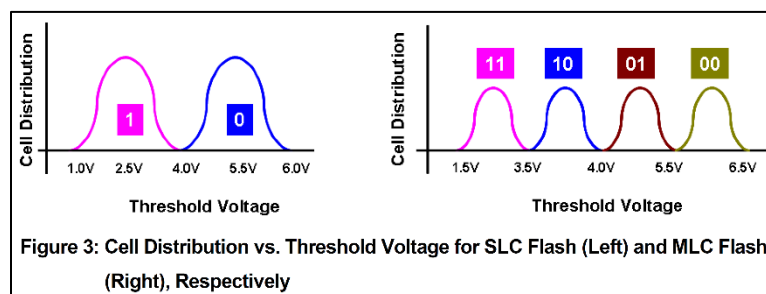
30. Floating gate transistors individually have limited endurance. Depending on the type of transistor and how it is configured, an individual transistor may not work reliably after it has been programmed and erased too many times. For example, the endurance for some flash transistors is on the order of 10,000 to 100,000 program-erase cycles.

31. Because of the limited endurance of floating gate transistors, and because some locations in a memory may be frequently rewritten, some floating gate transistors in the memory may wear out earlier than others. Therefore, flash memory typically includes a controller that uses a flash translation layer to map logical addresses presented to the host to physical addresses used to address the physical flash memory. This flash translation layer allows for wear to be leveled across all

the transistors on a device, and for bad blocks to be managed and avoided. This is done by changing the logical-to-physical address mapping.

A. SLC and MLC Flash

32. Early flash memory stored only a “0” or a “1” in each transistor. This is today known as “SLC,” or “single level cell,” flash. Later, to increase the density of storage in flash memory, a technique known as “MLC”, or “multiple level cell,” was introduced, where multiple threshold voltage levels were used in each transistor, to store multiple bits of data. For example, the following diagram shows how in SLC flash, there are large threshold voltage intervals between the “1” and “0” values, whereas in MLC flash, there are smaller voltage intervals between the multiple-bit logic values such as “11,” “10,” “01,” and “00.” The MLC flash design below stores 2 memory bits.



Ex. 2002 (Chen et al., *Ultra MLC Technology Introduction*, Advantech Technical White Paper (Oct. 5, 2012)), 3.

33. The primary difference between SLC and MLC is what data each threshold voltage represents. With SLC flash, the transistor stores only a 1 or 0, so

a wide range of threshold voltages can be allotted to a single bit. This allows for faster and more reliable memory access. On the other hand, MLC flash must be slowly and carefully programmed to a narrower, more precise range of threshold voltages, with each threshold voltage range representing a specific pair of bits (see the figure above, which shows four pairs of bits—11, 10, 01, and 00—corresponding to different ranges of threshold voltages). As a result, the above MLC cell requires two successive read operations to determine which pair of bits were stored in it.

34. Flash memories sometimes use dual-mode flash, where the cells can be configured as SLC or MLC. When the cell is in SLC mode, it uses only two threshold voltages to store a single bit. Ex. 2003 (Micheloni), 511 (where a single NAND memory “has two sub sections, the first of them to be used as SLC, the other one as MLC”, in which “[t]he partition is dynamic” and “the “SLC partition” has “a single distribution in the positive side of the threshold voltage values”).

B. Address Table

35. To provide wear leveling, garbage collection, and bad block management, a translation layer is used to map logical addresses to actual physical locations. As part of this translation layer, “tables are widely used in order to map sectors and pages from logical to physical (Flash Translation Layer or FTL).” Ex. 2003 (Micheloni), 40. These tables map logical blocks to physical blocks. *Id.*, 40-

42. Using a “block” or similar granularity is important, since flash memory is arranged so that when erasing and rewriting data, all the memory in a block is “erased together.” *Id.*, 22. Therefore, Dr. Rao explained that “[t]he address ranges within the translation table will assume some minimum quantum, such as, for example, one block, although a smaller size, such as one page could be used, if the NAND flash has the capability of erasing the smaller size quantum.” Ex. 1001 (298 Patent), 5:27-31. Dr. Rao further explained that memory is written and mapped on the granularity of a “quantum,” such as a block or page. *Id.*, 5:27-31; Figs. 3A-B.

C. Data Integrity Tests

36. When data is stored in MLC memory, it is more prone to errors, and some data is more prone to errors than other data. One reason for this is that the threshold voltage intervals for MLC memory are smaller than the corresponding intervals for SLC memory. Errors can occur when writing or reading the data. Errors can also be caused by the data stored in neighboring cells. A data integrity test is a test that checks the integrity of the data (*i.e.*, whether errors have occurred). This test can be run immediately after data is written, or at a later time. If the test reveals a problem such as corrupt data, the data can be remapped to SLC (which is less error-prone) or less-used MLC, and the address table is modified accordingly. Ex. 1001, 4:4-10.

D. Hot and Cold Data

37. One of the key features of Dr. Rao's invention is how it distinguishes between "hot" blocks (which receive more frequent writes), and "cold" blocks (which receive less frequent writes). Ex. 1001, 6:24-29. Because SLC flash has the endurance to handle frequent writes, "hot" blocks can be allocated to SLC flash to increase the lifetime of the system. "Cold" blocks, on the other hand, can be allocated to MLC flash to take advantage of its higher density storage.

VI. THE 298 PATENT

38. U.S. Patent No. 8,891,298 (the "'298 patent"), entitled "Lifetime Mixed Level Non-Volatile Memory System," issued on November 18, 2014 and relates to systems for storing data using SLC and MLC flash memory (*see supra* Section V.A for my discussion regarding SLC and MLC flash memory). I understand the challenged claims from the '298 patent are claims 1-11.

39. G.R. Mohan Rao is the sole named inventor of the 298 patent.

40. In the challenged claims, data is stored in non-volatile memory using single level cell (SLC) memory that stores 1 bit per cell, and multiple level cell (MLC) memory that stores more than 1 bit per cell. There are pros and cons to SLC and MLC flash. SLC is faster and less prone to errors, but requires more area to store a given amount of data, and has better endurance (i.e. the number of write-erase cycles before failure) characteristics. The opposite is true of MLC. MLC flash

is slower and more prone to errors, but stores data more densely, and has worse endurance.

41. The challenged claims are directed to specific techniques for efficiently using SLC and MLC flash to improve the overall performance of the memory. For example, if certain data is more prone to errors, then it is transferred to SLC (which has better endurance) or less-used MLC. By doing so, the number of errors is reduced, and the overall endurance of the memory is increased. Similarly, if certain data is used more frequently, then it is transferred to SLC (which is faster). By doing so, the overall speed of the memory is increased.

42. Claim 1 of the Challenged Patent requires:

Claim 1	
[1Pre]	A system for storing data comprising:
[1a]	at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks ;
[1b]	at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks ; and
[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:

[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;
[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;
[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
[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.

VII. CLAIM CONSTRUCTION

43. I understand that claim terms are typically given their ordinary and customary meanings, as would have been understood by a person of ordinary skill in the art (POSA) at the time of the invention. In considering the meaning of the claims, however, I understand that one must consider the language of the claims, the specification, and the prosecution history of record.

44. Petitioner proposes construction of five terms: “blocks,” “MLC non-volatile memory module,” “SLC non-volatile memory module,” “controller,” and “data integrity test.” Petition, 17-19.

45. I understand that Petitioner proposes that “blocks” is “indefinite as to whether it refers to logical or physical blocks.” Petition, 18.

46. I understand that Petitioner proposes that “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.” Petition, 17.

47. I understand that Petitioner proposes that “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.” Petition, 17.

48. I understand that Petitioner contends “controller” “is indefinite for, among other things, failing to provide necessary structure or algorithms, but will show that the recited functions are met by art.” Petition, 19.

49. I understand that Petitioner proposes that “data integrity test” means “a test that compares stored data to retained data.” Petition, 19.

50. While I disagree with all of Petitioner’s proposed constructions, I am not presenting opinions for any of these terms but “blocks” in this declaration. I have been informed that claim construction is only performed in IPR proceedings to the extent that it affects the outcome of the IPR proceeding. Here, I do not believe that any of Petitioner’s proposed constructions (including “blocks”) that are identified above affect whether this IPR should be instituted because, as I explain below, Petitioner is wrong about numerous points unrelated to these proposed constructions. Still, when the construction I have proposed for “blocks” is applied, Petitioner’s grounds are wrong for additional reasons.

A. “blocks”

51. Consistent with my declaration testimony in other proceedings, and with the construction the Court adopted in the prior litigation against Western Digital and the currently active litigation against Phison, it is my opinion that regarding the claimed “blocks” of the ’298 Patents, a “block” is “in a non-volatile memory, a

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

physical group of memory cells.” *See* Ex. 2031, 3 (Preliminary Claim Construction for “blocks” in *Vervain, LLC v. Phison Electronics Corporation*, No.1:24-cv-259-ADA); 2032, 22 (Court’s consideration of “blocks,” adopting its preliminary construction during *Markman* hearing in *Vervain, LLC v. Phison Electronics Corporation*, No. 1:24-cv-259-ADA); Ex. 2033, 1 (construing “blocks” in *Vervain, LLC v. Western Digital Corp. et al.*, No. 6:21-cv-00488-ADA, Dkt. No. 180, Order on the Pending Motions and Motions In Limine (W.D. Tex. Jul. 26, 2023)); Ex. 2040 (*Vervain, LLC v. Western Digital Corp. et al.*, No. 6:21-cv-00488-ADA, Dkt. No. 183, Pretrial Hearing Redacted Transcript (W.D. Tex. Aug. 7, 2023)). This construction is supported by the claim language and specification of the patents, as I explain below.

52. Claim 1 of the ’298 Patent recites “at least one MLC non-volatile memory module comprising a plurality of individually erasable blocks” and “at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks.” Ex. 1001, 7:9-12. From such context regarding “erasable blocks” and “MLC [multi-level cell] non-volatile memory module” and “SLC [single-level cell] non-volatile memory module” it is my opinion that a POSA would have understood that the “blocks” are physical groups of memory cells in a non-volatile memory.

53. These references to “individually erasable blocks” of SLC and MLC memory modules form the antecedent basis of further references to “blocks” in subsequent claims of the ’298 Patent. MLC (multiple level cell) and SLC (single level cell) refer to physical memory cells in flash memory, e.g., implemented with transistors—and thus the blocks in MLC or SLC memory are physical (as opposed to logical) groups of memory cells. Thus, erasable “blocks” in the context of claim 1 are erasable physical groups of MLC or SLC memory cells that the controller (as opposed to host) erases.

54. The ’298 patent states that the controller “determines which physical block to use each time data is programmed”. Ex. 1001, 3:2-3.

55. The ’298 patent also states that “any “write” or “program” to a block ... requires an “erase” ... before “write.”” Ex. 1001, 3:27-30. From this, a POSA would understand that the patent teaches that a write and a program are identical.

56. Therefore, by combining the two statements supra, the POSA would understand that the controller writes to physical blocks.

57. Now the ’298 patent also says that the controller “writes to the available erased block”, i.e., blocks must be erased before writing. Ex. 1001, 3:8.

58. Since the POSA understands that the controller writes to physical blocks, and also, the controller writes to erased blocks, the POSA would readily

understand that the '298 patent incontrovertibly and intrinsically teaches that the erased blocks are physical blocks.

59. Additionally, because the claimed “blocks” are “erasable,” a POSA would have understood that the “blocks” are physical groups of memory cells, since erasability from the perspective of the controller requires that the blocks be physical groups of memory cells. A POSA would have recognized that flash memory systems were known to include logical blocks and logical addresses that map to physical blocks/addresses. Ex. 2042 (U.S. Patent App. Pub. No. 2008/0140918 (“Sutardja”)), [0109]. Claim 1 of the '298 recites functionality that the controller performs regarding the claimed “blocks,” and the specification explains the controller deals with physical groups of memory cells, as described supra. As explained by the '298 patent, a host processor deals with logical blocks, i.e., is aware of the blocks at a higher level of abstraction than the controller is aware. Ex. 1001, FIG. 1, 2:65-3:13, 3:27-30.

60. A POSA would have further recognized that the property of “blocks” being “erasable” requires that the “blocks” be physical as opposed to logical blocks. That is because logical blocks/addresses can be re-mapped to different physical blocks/addresses, but only physical (and not logical) blocks can be erased in a SLC or MLC non-volatile memory. The '298 patent teaches that programming (writing)

is performed on physical blocks, and erasing is also done to physical blocks, as explained supra. Relevant lines from the '298 patent are quoted below for context.

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).

Ex. 1001, 2:65-3:13 (emphasis added).

61. Based on such disclosure of determining which physical block to use in the context of erasable blocks, a POSA would have understood that the “blocks” of claim 1 of the '298 Patent, which are “erasable” as discussed above, are physical (and not logical) groups of memory cells. This is further confirmed by claim 1 of the '298 Patent requiring that the claimed functionality being performed by a controller, which a POSA would have known performs erases on *physical* groups of memory cells. For example, as I noted above, the '298 Patent discloses that the controller “determines which *physical* block to use each time data is programmed,” making clear the physical aspect. Ex. 1001, 2:65-3:13, emphasis mine.

62. Additionally, the specification of the '298 Patent explains that “[b]locks can only be erased in their entirety, and when erased, are usually written to '1' bits.” Ex. 1001, 2:43-45. Such disclosure confirms that the “blocks” are physical groups of memory cells, because a POSA would have understood that erasing or writing bits in memory cells is functionality applicable to physical groups of memory cells, whereas logical blocks are mapped to physical blocks. Further, the statement that erasing results in the writing of a '1' bit is based on an encoding of the state of trapped electrons in the floating gate, to the value of the corresponding memory cell. A POSA would readily recognize that the encoding mentioned above agrees with the well-known electrical behavior of a floating gate transistor (which is a physical circuit that is used to implement blocks in a SLC or MLC flash memory). Additionally, based on such disclosure, a POSA would have understood that “blocks” as claimed in the '298 Patent (unless explicitly referred to as “logical blocks”) refer to physical groups of memory cells that must be erased all at once. *See* Ex. 2042 (U.S. Patent App. Pub. No. 2008/0140918 (“Sutardja”)), [0157].

VIII. PETITIONER'S INVALIDITY ARGUMENTS

A. OVERVIEW OF THE PRIOR ART

63. I understand that Petitioner asserts that the challenged claims are obvious based on three different grounds, each presented with regard to a single reference: **Ground 1**: Petitioner alleges that Claims 1-11 are obvious over Gavens (U.S. Patent No. 8,634,240) (Ex. 1045) in view of the knowledge of a POSA (Petition, 19-36); **Ground 2**: Petitioner alleges that Claims 1-11 are obvious over Moshayedi (U.S. Patent Appl. Pub. No. 2009/032759) (Ex. 1043) in view of the knowledge of a POSA (Petition, 36-48); and **Ground 3**: Petitioner alleges that Claims 1-11 are obvious over Sutardja (U.S. Patent Appl. Pub. No. 2008/0140918) (Ex. 1042) in view of the knowledge of a POSA (Petition, 49-63).

1. Gavens (Ex. 1045)

64. Gavens discusses a problem that occurs with flash memory (a type of non-volatile memory)—as it ages, its error rate increases. Ex. 1045, 3:56-4:5. To ensure data integrity in such situations, the memory uses a resource-intensive error correction code (ECC) to correct errors “at the end of life of the memory device.” *Id.* Gavens explains that this poses a problem because, “for most of the life time of the device [(before the device ages and the error rate increases)], the ECC is only marginally utilized, resulting in its large over-heads being wasted[.]” *Id.*, 3:67-4:2.

65. To perform ECC, Gavens describes a flash memory device including **controller 102** (highlighted purple) and **memory chip 100** (highlighted teal).

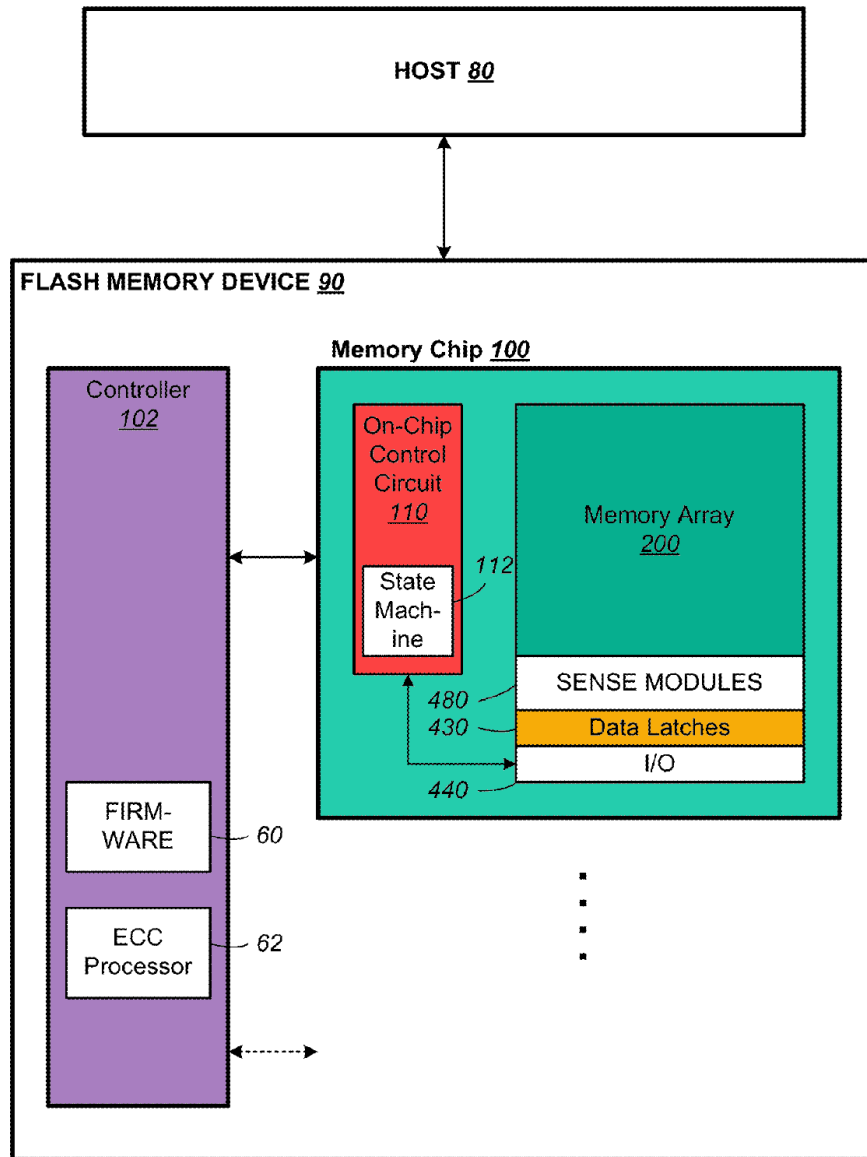


FIG. 1

Ex. 1045, Fig. 1, 8:13-39. The **memory chip 100** also includes an **on-chip control circuit 110** with state machine 112. The **controller 102** “controls and manages

higher level memory operations” while the **on-chip control circuit 110** controls “low-level memory operations of each chip.” Ex. 1045, 8:23-39. The memory chip also includes input/output (I/O) circuits, **data latches 430**, sense modules, and a non-volatile flash **memory array 200**. *Id.*, 4:13-16, 8:13-39.

66. Before writing a page of data to the memory array, “ECC is computed for the data page” by the ECC Processor 62. *Id.*, 12:51-63. The page of write-data then consists of two portions: the “user data” and the computed ECC. *Id.*

67. When a page is read from the memory array, it is “shifted out” (or “toggled”) to **controller 102**. There, “the data page’s existing ECC is compared to a second version of the ECC computed on the read data [(the “user data”)]”. The ECC first runs an error detection code (EDC) which “indicates the existence of any errors in the read data page[,]” and if there are errors, “the ECC is invoked to correct erroneous bits in the read data page” *Id.*, 12:64-13:8.

68. Gavens teaches that using ECC in this way—where the ECC must be programmed to correct the “worst-case” number of bits—is burdensome in terms of processing time as well as chip area on the controller:

Using ECC to correct a worst-case number of error bits will consume a great amount of processing time. The more bits it has to correct, the more computational time is required. The memory performance will be degraded. Additional dedicated hardware may be implemented to

perform the ECC in a reasonable amount of time. *Such dedicated hardware can take up a considerable amount of space on the controller ASIC chip.* Moreover, for most of the life time of the device, *the ECC is only marginally utilized*, resulting in its large over-heads being wasted and realizing no real benefits.

Ex. 1045, 3:60-4:2; *see also id.*, 5:15-30, 13:9-16.

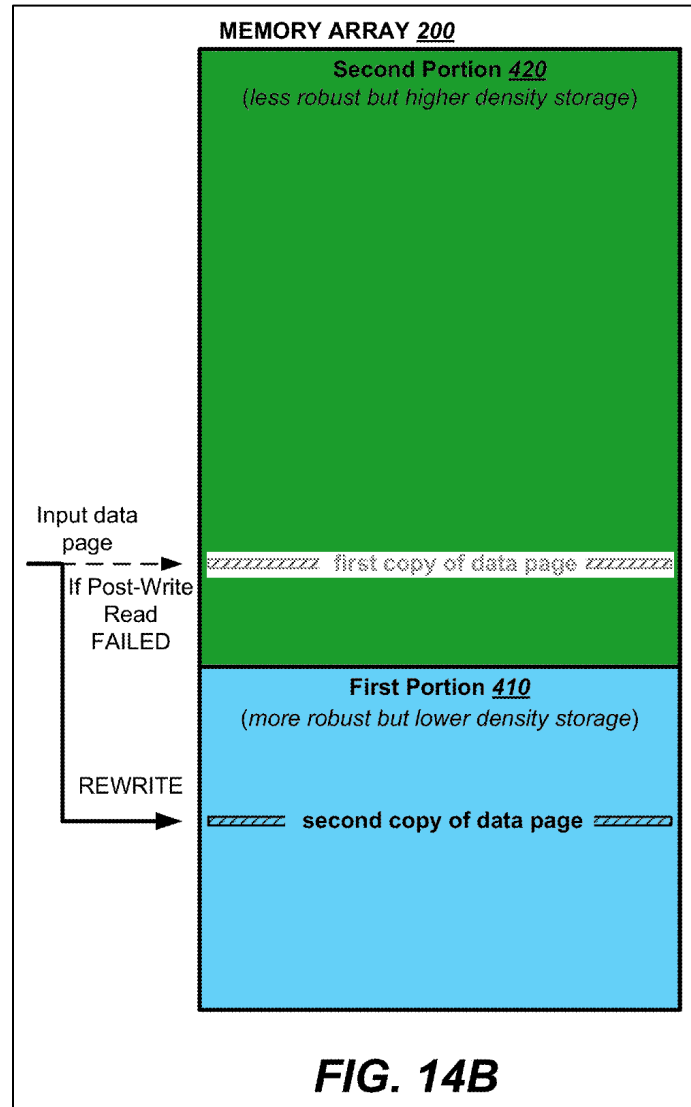
69. To address this problem, Gavens discloses the embodiments of Figures 14A-14B and 16A-16C, which describe the **memory array 200** of Figure 1 in greater detail. *See id.*, 8:13-15 (“Fig. 1 illustrates...a memory device in which the features of the present invention are embodied.”).

70. These embodiments rely on a memory with a first SLC portion (which is less error prone) and a second MLC portion (which is more error prone). Gavens explains that optimizing where data is stored (in the SLC or MLC portion) based on the expected number of error bits “allows a smaller and more efficient...ECC...to be designed for correcting a smaller number of error bits, thereby improving the performance and reducing the cost of the memory.” *Id.*, 4:28-31.

a) First Embodiment (Fig. 14A-14B)

71. Gavens describes a first embodiment with respect to Figures 14A-14B. *See id.*, 17:24-26 (referring to “the embodiment described in Fig. 14A and Fig. 14B”). *See also id.*, 16:14-16. For this embodiment, as shown in Figure 14B below,

Gavens describes a memory array that relies on a “**lower density**” first portion 410 (*i.e.*, SLC) and a “**higher density**” second portion 420 (*i.e.*, MLC). *Id.*, 4:13-19, 16:14-28. The MLC portion is more error prone than the SLC portion but can be used “for efficient storage.” *Id.*, 4:16-19, 16:41-45. Gavens explains that data can first be written to the MLC portion, then “read back in a post-write read operation to check for excessive error bits.” *Id.*, 4:19-21; *see also id.*, Fig. 14A, 16:41-56. This is performed within the memory chip 200.



72. Gavens explains that the post-write error detection process can check for errors in one of two ways: “either by comparison with the original copy which may be cached or by checking the EDC portion of the ECC.” *Id.*, 16:46-49. While Gavens at 16:46-49 does not specify how the “original copy” is cached, Gavens later explains that “[i]n the preferred embodiment” (where the antecedent basis of “the” refers to the immediately preceding “preferred embodiment” of Figures 14A-14B)

the “data is cached in [a] first section of the first portion” of non-volatile memory (*i.e.*, a section of the non-volatile SLC memory). *Id.*, 18:1-8 (“In the preferred embodiment, the first portion is further provided with **a first section** and a second section. The incoming data is cached in **the first section** of the first portion and a first copy of the data is written to the second portion.”); *see also id.*, 16:14-16.

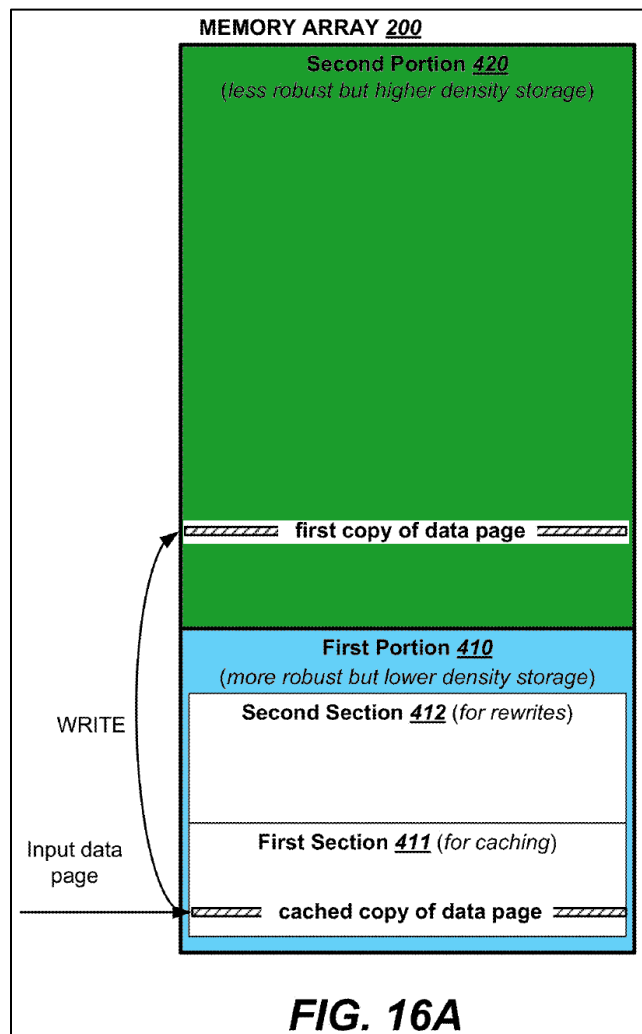
73. Once the number of error bits is determined, the device determines whether to keep the data in MLC or instead move it to SLC. If the stored data has an acceptable number of error bits, then the stored copy remains in the MLC and becomes the “valid” copy. *Id.*, 16:41-56. “If the error bits exceeded a predetermined amount, the data is” instead stored in “the less error-prone” SLC portion. *Id.*, 4:21-22; *see also id.*, Fig., 14B, 17:7-20, 18:1-8. To rewrite the data, Gavens teaches that the memory may rely on the “cached” copy of the original data or obtain the data by “correcting the errors bits [from the copy stored in the MLC] with the ECC.” *Id.*, 17:12-14; *see also id.*, 18:1-8.

b) Second Embodiment (Fig. 16A-16C)

74. Figures 16A-16C depict a second embodiment. *See id.*, 7:26-28 (referring to “the embodiment described in Fig. 16A to Fig. 16C”). For this embodiment, Gavens, as is explained further below, teaches splitting the first portion (*i.e.* the SLC portion) into a first cache section and a second section for rewriting

data. *Id.*, 18:9-23. Gavens emphasizes that this approach avoids making the comparison at the controller 102, which is a separate chip in the flash memory device 90. *Id.*, Fig 1, 18:36-39.

75. As shown in Figure 16A below, Gavens explains that the lower density First Portion 410 (*i.e.*, SLC) includes two sections—a “first section” 411 for caching write data, and a “second section” 412 for rewriting data from the higher density Second Portion 420 (*i.e.*, MLC). *Id.*, 18:9-23.



76. Incoming data is written both to the MLC and as a “cached copy” to the first section of the SLC. *Id.*, 18:21-29. Detecting the number of error bits is then “accomplished by comparison [of the data written to MLC] with the cached copy.” *Id.*, 18:40-44.

77. Gavens specifies that this comparison may be performed at the **data latches 430** (highlighted orange in Figure 1 above) on the **flash memory chip 100**. *Id.*, 18:30-39. Gavens strongly recommends using **data latches 430** because it eliminates the need to “toggle” the data out to the **controller chip 102**, as would be required if one were to use ECC Processor 62 for EDC-based error detection. *Id.*, 18:36-39. Toggling or transferring the data to the controller chip 102 (a separate chip than the memory chip 100 on the flash memory device 90) would require a large delay, power and energy consumption. Gavens explains that “much time can be saved” by eliminating the need to toggle data to the controller chip, thus increasing the performance of the device. *Id.*

78. If there is an acceptable number of error bits, the MLC portion becomes the “valid” copy and the cached copy in the first section of the first portion is obsoleted. *Id.*, 18:45-52, Figure 16B. If there is an unacceptable number of error bits, the “cached copy” in the “first section” of the SLC is rewritten to the “second section” of the SLC. *Id.*, 18:53-67, Figure 16C.

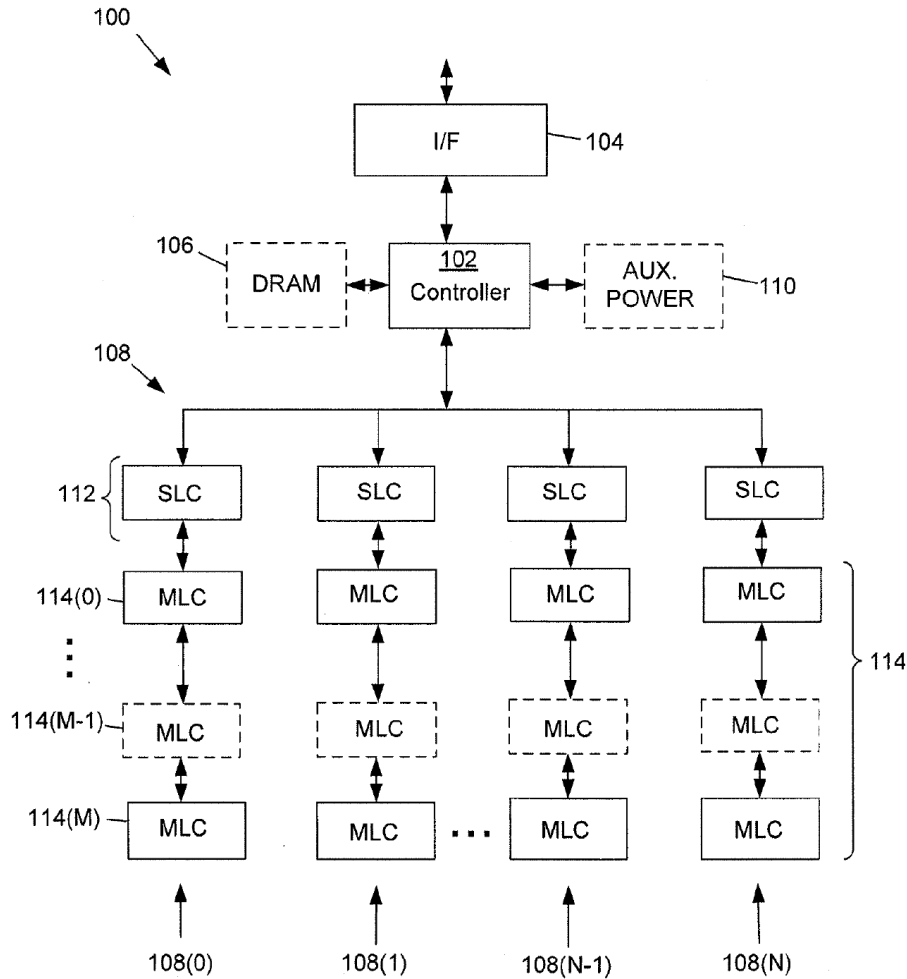
79. By caching the data in the non-volatile memory array and using the latches to compare the data, Gavens lowers overhead and reduces the reliance on the controller chip. *Id.*, 18:40-67.

c) Third Embodiment (Fig. 19): “Hot Count”

80. Gavens presents a third embodiment with Figure 19, where “the age of the memory device is determined by a hot count maintained with each erase block of memory cells.” Ex. 1045, 19:37-20:39, 19:47-49; *see also id.*, 7:30-32. The “hot count” tracks “the number of times each erase block has been cycled through erase and program operations.” *Id.*, 19:47-54. Once the “hot count” for an erase block reaches a threshold, “enhanced post-write-read error management” is enabled. *Id.* Gavens explains, for example, that, when enhanced error management is enabled for an MLC block, the error rate of that block is determined and, if the error rate is excessive, the data from that block may be moved to a different MLC block. *See id.*, 20:30-50.

2. Moshayedi (Ex. 1043)

81. As shown in Figure 1 below, Moshayedi describes a flash memory having both SLC and MLC memory, with the described goals of “accommodat[ing] application memory needs at desirable prices, in addition to increasing read/write performance.” Ex. 1043, Abstract, ¶8.



Ex. 1043, Fig. 1.

82. To accomplish these goals, the memory distinguishes between “static” (not frequently written) and dynamic (“frequently written”) data, and changes where that data is stored—in MLC or SLC—accordingly. *Id.*, Abstract. Moshayedi explains that all data may initially be written to SLC. *See id.*, ¶¶9 (“The threshold may be set at 0 initially, resulting in all data being written to SLC flash...”), 24, 54, 73. But later, the memory may rely on both MLC and SLC, and determines where

to store data based on whether it is “static” and “dynamic.” *Id.*, ¶¶9, 24, 31-33. Moshayedi describes several, alternative embodiments for doing this, including embodiments that consider (1) write count, (2) erase count, and (3) error count. *Id.*

83. **Write Count:** Moshayedi explains that the memory may distinguish between static and dynamic data by determining the **write count** of a logical block address (LBA). *See* Ex. 1043, Abstract, ¶¶9, 24. The memory “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.” Ex. 1043, ¶¶9, 24. Unlike the Challenged Claims—where the respective contents of blocks are transferred from MLC to SLC—here, only “newly received data” is *redirected* from MLC to SLC.

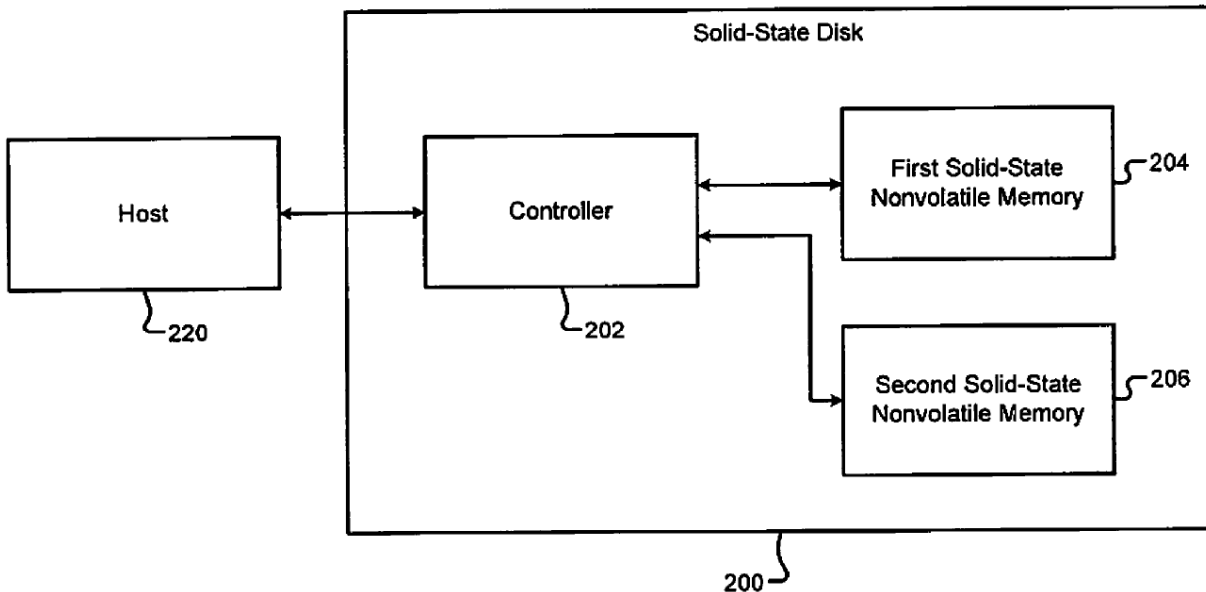
84. **Erase Count:** Alternatively, static and dynamic data can be distinguished based on the “erase count” of physical blocks. Ex. 1043, ¶¶30, 32, 47-50. “The flash drive may use the erase count to move data between MLC flash and SLC flash.” *Id.*, ¶32. Moshayedi explains that, “once a block in MLC flash reaches a threshold erase count...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.” *Id.* Thus, unlike the Challenged Claims and like the “write count” embodiment

described above, only new data—for the “next write operation”—is *redirected* from MLC to SLC.

85. **Error Count:** Moshayedi describes another alternative embodiment, which relies on the error count of physical blocks to determine whether data is static/dynamic. Ex. 1043, ¶¶31, 33. This embodiment describes relocating data blocks with a threshold “number of read errors” to “*data blocks with less wear.*” Ex. 1043, ¶33. Unlike the write count and erase count embodiments, this embodiment does not disclose redirecting data from MLC to SLC; instead, data is redirected to “blocks with less wear.” *Id.*

3. Sutardja (Ex. 1042)

86. Sutardja describes a system with a “First...Memory” 204 and “Second...Memory” 206, as depicted in Figure 2 below. Sutardja describes mapping logical addresses to the first and second memories, based on characteristics of each memory related to their projected lifetimes. Ex. 1042, Abstract.



Ex. 1042, Fig. 2.

87. Particularly, Sutardja describes performing “degradation testing” on physical addresses to determine the storage capability of the memory. Ex. 1042, ¶¶134-139, 151. Sutardja explains that, during “periods of inactivity,” *test data* can be written to a physical address. *Id.*, ¶135 (“degradation testing module...*may provide addresses and data*”). This provided data is therefore test data. The test data is read back to check for errors, and based on that test a “degradation value for the address” is determined. *Id.*, ¶¶135-138. The memory “may adapt its mapping based on the degradation value.” *Id.*, ¶138. Further, if one of the first or second memories (204 and 206 in Figure 2 above) fails the degradation test, the memory “may assign *all new writes* to the other one of the memories 204 and 206” that did not fail the

test. *Id.*, ¶139. Accordingly, unlike the Challenged Claims, and just like Moshayedi (explained above), Sutardja does not disclose transferring the respective contents of blocks based on the degradation testing—only to make “assignment decisions” for performing “assign[ing]”/redirecting of “*new writes.*” *Id.*

88. Sutardja describes additional embodiments where metrics about the memory are determined and, based on those metrics, the memory adjusts its mapping. *First*, with Figure 7A and ¶¶146-147, Sutardja considers the “*write frequencies* for *logical addresses* where data is *to be written*” and adjusts its mapping to the first/second memory accordingly. *Second*, with Figure 7C and ¶149, Sutardja considers “the *number of write operations* to [a] first [*physical*] *block*...during a predetermined time” and adjusts its mapping accordingly. *Third*, with Figure 7E and ¶153, Sutardja describes mapping data based on whether “the *wear level* of the first...memory is greater than a predetermined threshold”—if so, the controller “maps *all* the logical blocks to physical blocks of the second...memory.”

89. As shown in Figure 7B of Sutardja, the steps of Figures 7C, 7D and 7E are performed whenever it is deemed to be the “time to perform” certain analyses, and after the writing is done in Step 510 of Figure 7A. As a result of the analyses of

Figures 7C, 7D and 7E as described above, the mapping is adjusted for subsequent writes in the future.

B. GROUND 1: GAVENS DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 1

90. In my opinion, the Petition’s analysis of why Gavens in view of the knowledge of a POSA discloses or suggests claim 1 is deficient. I address some of those deficiencies in Petitioner’s analysis below.

1. [1b]: “SLC non-volatile memory module comprising a plurality of *individually addressable blocks*”

91. Limitation [1b] recites: “at least one SLC non-volatile memory module comprising a plurality of **individually erasable blocks**.” In my opinion, the Petition fails to identify anything from Gavens corresponding to the “**individually erasable blocks**.” Petition, 22. The Petition never addresses this portion of limitation [1b] at all.

2. [1c]: “a controller coupled to” an “MLC non-volatile memory module” and an “SLC non-volatile memory module”

92. Limitation [1c] recites: “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:” In my opinion, Petitioner fails to show that Gavens discloses this limitation as claimed.

93. *First*, for this limitation, I understand that Petitioner relies only on its own construction for “SLC non-volatile memory”: “non-volatile memory cells arranged with circuitry incapable of storing multiple logical pages in a single physical page of cells.” Petition, 17. Applying this construction, Petitioner suggests that limitation [1c] requires a “‘*native*’ SLC flash chip.” Petition, 23. While I disagree with construction, even if it is applied (I note that Petitioner presents its ground based solely on its own construction), Petitioner’s ground fails.

94. Specifically, Petitioner fails to explain why it would have been obvious to replace one of Gavens’s MLC memory chips, which are each coupled Gavens’s “controller 102,” with a “‘*native*’ SLC flash chip.” Petition, 22-23. Instead, Petitioner merely asserts that this modification would have been obvious, and cites Gavens at 8:17-18 and 8:25-29. *See* Petition, 23 (“It would be obvious to have one or more of those NAND flash chips be ‘*native*’ SLC flash chips.... Gavens...disclose the controller coupled to an MLC module and obviously substituted SLC module....”). I have reviewed these citations, and they do not support Petitioner’s conclusion. The cited portions of Gavens say nothing about SLC, let alone “*native*” SLC, as Petitioner contends. These citations describe only Gavens’s “memory device” and its controllers. Ex. 1045, 8:17-18 (“The memory device 90 includes one or more memory chip 100 managed by a controller.”), 8:25-

29 (“The control circuitry 110 is an on-chip controller.... The control circuitry 110 typically....”). In my opinion, the paragraph of Gavens that Petitioner cites instead directly contradicts its argument, stating that “memory chip 100 includes a memory array ... of memory cells with *each cell* capable of being configured as a *multi-level cell (‘MLC’) for storing multiple bits of data.*” *Id.*, 8:18-21 (emphasis added). Thus, in my opinion, Petitioner fails to support its assertion that a POSA would have “substituted” one of Gavens’s MLC chips with a “‘native’ SLC flash chip[.]” Actually, this change would have resulted in an additional design burden on the controller, which would motivate a POSA not to make the change proposed by Petitioner.

95. ***Second***, contrary to Petitioner’s conclusory assertion, in my opinion, a POSA would not have been motivated to replace one of Gavens’s memory chips with a “‘native’ SLC flash chip.” Gavens explains that the memory array within its memory chips are already capable of operating with one portion as MLC, and another portion as SLC (i.e., not “native” SLC). *See* Ex. 1045, Abstract, 4:13-31, 6:17-25, 15:62-16:12, 16:17-25. That is the essence of Gavens’s invention—taking advantage of the efficiencies of MLC storage, by relying on a “less error-prone” SLC portion when an MLC portion has too many errors. *See id.*, 4:13-31, 4:18-22 (“Data is written to the second portion for efficient storage. Afterwards, the data is readback

in a post-write read operation to check for excessive error bits. If the error bits exceeded a predetermined amount, the data is rewritten or kept at the less error-prone first portion.”), 15:66-16:13, 16:41-45.

96. Additionally, Gavens explains that using the SLC portion to cache data, as described in its First and Second Embodiments (*see supra* Section VIII.A.1), contributes to avoiding the need to “toggle[]” data out to controller 102 when comparing data for error checking. *See* Ex. 1045, 18:21-39, 18:36-39. This is because “data latches” within the memory chip (which includes both the MLC and SLC portions) enable data comparison without relying on external controller 102. *See id.*, 18:30-39, Fig. 1 (showing “Data Latches” 430 as part of the Memory Chip, separate from the Controller 102). In my opinion, replacing the SLC portion of the *same* memory chip with a *separate* “native” SLC memory chip connected to the controller 102 would negate this benefit, and in fact excessively complicate the design of the memory, since the data cached in a separate “native” SLC chip would necessarily need to be toggled out for comparison with data stored in the MLC chip. In my opinion, a POSA would therefore not be motivated to make the choice proposed by the Petitioner.

97. Therefore, in my opinion, Gavens teaches away from Petitioner’s proposed modification, and a POSA would not have been motivated to replace one

of Gavens’s MLC memory chips with a “‘native’ SLC flash chip,” as this modification would undermine Gavens’s intended advantage—efficiently utilizing higher-density MLC memory.

3. [1f]: “c) determine which of the blocks of the plurality of blocks...are accessed most frequently by maintaining a count of the number of times each one of the blocks is accessed”

98. Limitation [1f] recites: “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.” In my opinion, Petitioner fails to clearly explain which prior art disclosure it is mapping to this limitation, particularly with respect to “maintaining a count.” Petition, 26-27. Petitioner ambiguously references multiple unrelated disclosures that mention a “count,” without explaining how they are mapped—individually or in combination—to the claims. Petitioner first cites Gavens’s “hot count” but then also relies on Lee (Ex. 1050) and Gorobets (Ex. 1049), citing their unrelated teachings on maintaining “counts” for memories. *See* Petition, 27. However, Petitioner provides no explanation of how Lee and/or Gorobets—which do not describe the same embodiments as Gavens—apply to or combine with Gavens’s disclosure, or what a POSA’s motivation would be to use one or more of these alleged “count” citations.

99. Additionally, with its citations to Gavens, Petitioner relies on different embodiments of Gavens than they did for previous limitations: Petitioner relies on Gavens's third (Figure 19) embodiment for this limitation but relied on Gavens's first (Figure 14) and second (Figure 16) embodiments for limitation [1e]. Specifically, for limitation [1f], Petitioner cites Figure 19 for its discussion of the "hot count." Petition, 26-27; *see* Ex. 1045, 19:47-49, 20:7-29 (describing Figure 19 as part of "a preferred embodiment"). But Petitioner relied on Figures 14 and 16 for limitation [1e]. *See* Petition, 24-26; *see* Ex. 1045, 16:14-15 (referring to "Fig. 14A" as part of "a preferred embodiment"), 18:9-12 (referring to "Fig. 16A" as part of another "preferred embodiment")

4. [1f]: "controller adapted to" ([1c]) ... "maintain[] a count of the number of times each of the blocks is accessed"

100. For limitation [1f], Petitioner also fails to demonstrate that Gavens's Figure 19 discloses "*controller 102*" maintaining the "hot count." Notably, limitation [1c] requires a "controller adapted to" perform limitations [1d]-[1g]. For limitation [1c], Petitioner maps the claimed controller to Gavens's "controller 102." Petition, 22-23. Therefore, for limitation [1f]—as well as limitations [1d]-[e] and [1g]—Petitioner must show that the same controller 102 performs each claimed function. In my opinion, Petitioner fails to do so.

101. Instead, for limitation [1f], Petitioner conclusorily states, “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..., *the controller clearly* is adapted to maintain the counts and determine from those counts the blocks accessed most frequently.” Petition, 27. But Petitioner provides no explanation how Gavens discloses that *controller 102* performs these functions. Petitioner cites Gavens at 20:18-27 and Figure 19, but this disclosure says nothing about controller 102, or any other controller.

102. To the contrary, a POSA would have understood that a different and separate controller—“*on-chip control circuit 110*”—maintains Gavens’s “hot count”:

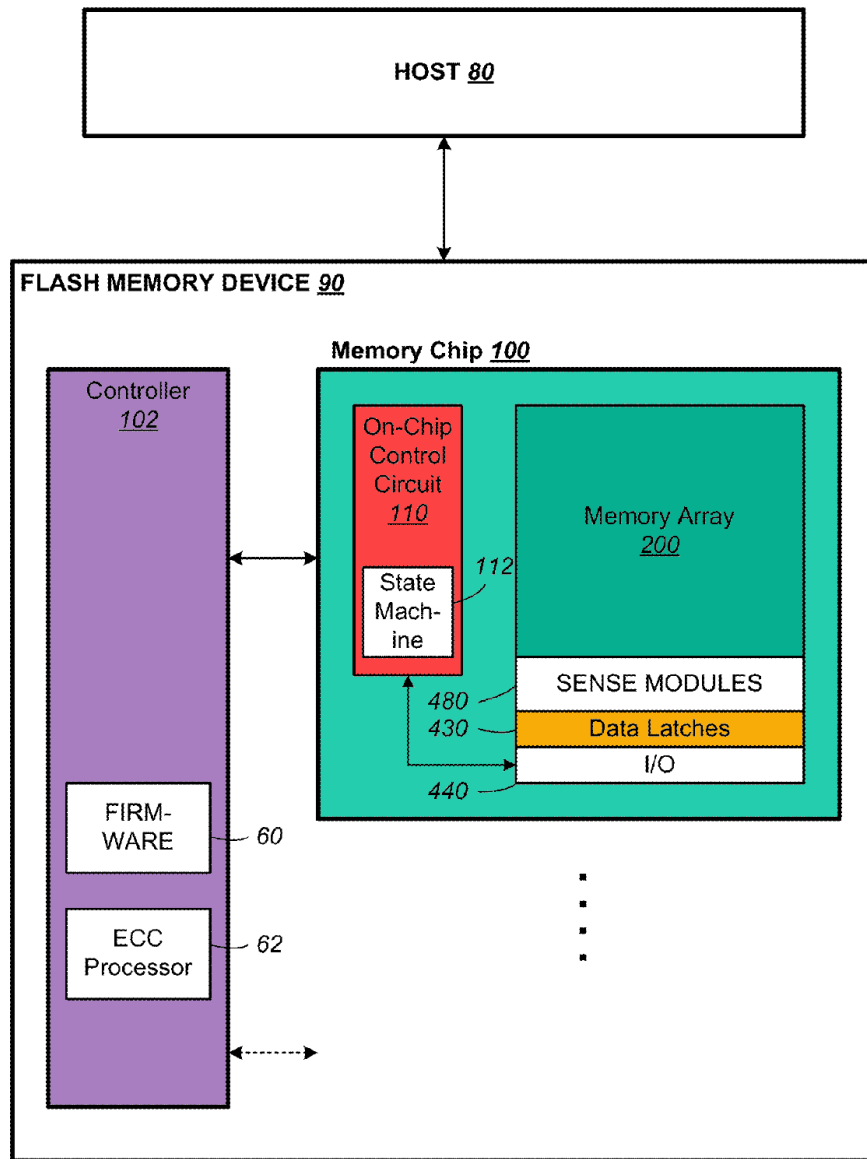


FIG. 1

Ex. 1045, Fig. 1 (annotated).

103. Gavens explains that “the hot count for each block can be store[d] in the system data area of the **data page 70**’....” Ex. 1045, 20:2-6; *see also id.*, 12:66-67 (“The **data page**...is then written to the **memory array 200**.”). Gavens does not

specify which controller maintains the “hot count,” and its storage in the **memory array 200**. In my opinion, a POSA would have understood, however, that this function is performed by **on-chip control circuit 110**—not **controller 102**, as required for Petitioner to remain consistent with its analysis for limitation [1c]. Gavens distinguishes between these components, explaining that “on-chip control circuitry 110 controls low-level memory operations of each chip,” while “controller 102...controls and manages higher level memory operations.” Ex. 1045, 8:23-25, 8:32-33. Accordingly, in my opinion, a POSA would have determined that on-chip control circuit 110 maintains the hot count, as this qualifies as a “low-level memory operation.”

5. [1g]: “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”

104. Limitation [1g] recites: “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.” Petitioner contends that Gavens discloses limitation [1g] under both of its constructions of “blocks”: logical and physical. *See* Petition, 28-30. In my opinion, both arguments fail.

105. **Physical blocks**: As explained above, in my opinion, “blocks” should be construed to refer only to “physical” blocks. Applying this construction,

Petitioner's mapping is deficient. *See* Petition, 28-29. Petitioner does not explain how Gavens teaches "transferring" physical blocks to SLC, as required by limitation [1g].

106. Petitioner asserts that Gavens discloses that "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.*" Petition, 28. However, in my opinion, Petitioner provides no supporting evidence for this assertion. While citing Gavens's Figure 19 embodiment, including Gavens at 20:21-29, Petitioner fails to show that this passage addresses data transfer in any capacity. *See* Ex. 1045, 20:21-29 (stating only that "error management for the rest of the life of the memory" is "enable[d]" when a "memory block" fails the data integrity test).

107. **Logical blocks:** Petitioner's ground also fails under its construction of "logical" blocks. For this, Petitioner relies entirely on teachings from secondary references, Sutardja (Ex. 1042) and Moshayedi (Ex. 1043). Petition, 29. However, in my opinion, Petitioner fails to show why a POSA would have relied on the teachings of these unrelated secondary references.

C. GROUND 2: MOSHAYEDI DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 1

108. In my opinion, the Petition’s analysis of why Moshayedi in view of the knowledge of a POSA discloses or suggests claim 1 is deficient. I address some of those deficiencies in Petitioner’s analysis below.

1. [1b]: “SLC non-volatile memory module comprising a plurality of *individually addressable blocks*”

109. Limitation [1b] recites: “at least one SLC non-volatile memory module comprising a plurality of individually erasable blocks.” I understand that the Petition cannot incorporate by reference arguments from an expert’s declaration. In my opinion, the Petition itself fails to identify anything from Moshayedi corresponding to the “individually erasable blocks.” Petition, 37. The Petition never addresses this portion of limitation [1b] at all.

2. [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*,...”

110. Limitation [1d] recites: “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.” In my opinion, Petitioner fails to show that Moshayedi discloses this “list of logical address ranges having a *minimum quanta of addresses*” as claimed. Petition, 38-39. The Petition never addresses this portion of limitation [1d] at all.

111. Further, Petitioner references several paragraphs from Moshayedi, but none of these citations show that Moshayedi discloses or suggests the “minimum quanta of addresses.” Petition, 38-39 (citing Ex. 1043, ¶¶6, 36-39, 44). Paragraphs 6 and 44 disclose logical-to-physical mapping but do not describe a minimum quanta of addresses. *See* Ex. 1043, ¶¶6 (“logical block address (‘LBA’)”), 44 (“assigning LBA to physical address”). Similarly, paragraphs 36-39 disclose a “controller architecture” and “virtual-to-physical mapping” but do not describe a minimum quanta of addresses. *See id.*

3. **[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*”**

112. Limitation [1e] recites: “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.” In my opinion, Petitioner fails to show that Moshayedi discloses or suggests that “the controller remaps” a range of logical addresses (“the entry”) that “fails a data integrity test” from MLC to SLC, as required by [1e]. Petition, 39-40.

113. Petitioner relies on ¶¶31 and 33 of Moshayedi, which describe relocating data blocks with a threshold “number of read errors” to “data blocks with less wear.” Ex. 1043, ¶33. Based on this, Petitioner makes two assumptions: (1) that a POSA would have understood “‘data blocks with less wear’ to mean SLC”; and (2) that the data must be relocated *from MLC*. Petition, 39. In my opinion, both assumptions are incorrect.

114. When describing other embodiments, Moshayedi explicitly refers to relocating data between MLC and SLC, and does not use the term “blocks with less wear.” For example, Moshayedi clearly describes “mov[ing] data between MLC flash and SLC flash” based on erase count—distinct from the error count referenced in ¶¶31 and 33. Ex. 1043, ¶32 (“The flash drive may use the erase count to move data between MLC flash and SLC flash.”). In ¶33, however, Moshayedi uses

different language—“blocks with less wear”—which, in my opinion, a POSA would not have understood to refer exclusively to SLC, as Petitioner contends.

115. Instead, Moshayedi explains that *data is initially written only to SLC*. See, e.g., Ex. 1043, ¶¶9 (“The threshold may be set at 0 initially, resulting in all data being written to SLC flash...”), 24 (same), 54 (“The amount of SLC that is designated to fill-up initially, before factoring write frequency and moving blocks to MLC, can be any number (%).”). If data is initially written only to SLC and no MLC blocks have been written to, then MLC blocks (and unwritten SLC blocks) would have “less wear.”

116. Accordingly, based on the disclosure of ¶¶31 and 33, in my opinion, a POSA would have understood that read error rates are initially considered only for SLC blocks, as those are the only blocks that have been written to. When an SLC block reaches the threshold “number of read errors” (Ex. 1043, ¶33), it would be moved *from SLC* to an MLC or SLC block with “less wear.” Limitation [1e], in contrast requires *the opposite—remapping data from MLC to SLC....*,

4. [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*”

117. Limitation [1g] recites: “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.” Petitioner asserts that Moshayedi discloses limitation [1g] under both of its constructions of “blocks”: logical and physical. *See* Petition, 42. In my opinion, both arguments fail.

118. **Physical blocks:** In my opinion, petitioner does not and cannot show that Moshayedi discloses “transferring the respective contents of those blocks” that “receive the most frequent writes” to SLC, as required by [1g]. For this ground, the Petition simply states, “in another embodiment, the physical blocks are counted,” followed by a block quote and a list of citations to Moshayedi. Petition, 42 (citing Ex. 1043, ¶¶30, 32, 47-50, 60; Figs. 7A, 8). Petitioner fails to explain how this “count” results in “transferring the respective contents of...blocks” to SLC.

119. In my opinion, even after searching through Petitioner’s citations, it is clear that Moshayedi does not disclose or suggest “transferring the respective contents” of frequently written MLC blocks to SLC; only *newly received data* for a next/separate write operation is written to SLC. Moshayedi clearly states in paragraph 32 (cited by Petitioner) that “once a block in MLC flash reaches a threshold erase count...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.” Ex. 1043, ¶32. Paragraph 49 (also cited Petitioner) crystalizes this point, again making clear that, “When the erase count of a block at MLC area is less than a specified threshold

(e.g., 500) the next written data can be moved to a SLC area with erase count less than, e.g., 500” and “When the erase count of the block at MLC area is 500, *next written data* ca[n] require the data to move to SLC area wh[ere] erase count is less than 500.” Ex. 1043, ¶49.

120. Accordingly, unlike [1g], in my opinion, the “respective contents” of the MLC block are never transferred to a SLC block in Moshayedi. Rather, during the *next write operation*, a single block of data that is supposed to be written to the MLC block is instead redirected to a SLC block.

121. **Logical blocks:** In my opinion, Petitioner’s ground based on its logical blocks construction fails because, as explained above, “blocks” should be construed to refer only to “physical” blocks. Even when applying Petitioner’s incorrect “logical” blocks construction, however, in my opinion, Petitioner cannot show that Moshayedi discloses “transferring the respective contents of those blocks” to SLC, as required by [1g].

122. Here, Petitioner relies on the write count embodiment of Moshayedi. Petition, 42 (citing Ex. 1043, ¶¶9, 24). Moshayedi explains that this embodiment, just like the “erase count” embodiment of ¶32 (explained above for the “physical” blocks construction), does not result in the transfer of the “contents” of MLC blocks to SLC. Instead, only “*newly received data*” is redirected to SLC. See, e.g., Ex.

1043, ¶9 (“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”), 24 (same).

123. Moreover, Moshayedi explains that the “contents” of MLC blocks are never transferred to SLC blocks, because only the *opposite* transfer — from SLC to MLC — is necessary:

As the level of user blocks within the SLC goes above the configured limit, the logical blocks with the lowest write count are moved from SLC to MLC. *No converse operation is necessary* (copy from MLC to SLC) as logical blocks are *directed towards SLC when they are written* by the host if the write count is at a sufficient level to warrant this.

Ex. 1043, ¶73. Moshayedi explains that, while data can be “moved” from SLC to MLC, data is not moved from MLC to SLC. *Id.* Instead, data is only “*directed towards SLC when they are written.*” *Id.* (emphasis mine).

124. **Physical or Logical Blocks:** Under either construction, Petitioner’s ground also fails because Petitioner cites between different embodiments of Moshayedi without explaining why it would have been obvious to a POSA combine them. For limitation [1e], Petitioner relies on the “read error” embodiment. Petition, 39-40 (citing Ex. 1043, ¶¶31/33). Yet for limitation [1g], Petitioner shifts to the

erase count embodiment (*i.e.*, Ex. 1043, ¶32) for the “physical” construction, and the write count embodiment (*i.e.*, Ex. 1043, ¶¶9/24) for the “logical” construction. Petition, 42.

125. In my opinion, a POSA would have understood these embodiments as *alternative, competing* measures for determining whether data is “static” or “dynamic,” with overlapping functionality in wear leveling blocks. *Compare, e.g.*, Ex. 1043, ¶9 (“... to keep frequently written data, which results in frequently erased blocks, in SLC flash, and relatively static data in MLC flash”) *with id.*, ¶30 (“data...is presumed to be more static since the block containing the data has a low erase count”) *with id.*, ¶33 (“Information regarding the number of data read errors associated with a given data block may be used to determine whether the data stored therein is dynamic or static.”).

D. GROUND 3: SUTARDJA DOES NOT DISCLOSE OR SUGGEST ALL OF THE FEATURES OF CLAIM 1

126. In my opinion, the Petition’s analysis of why Sutardja in view of the knowledge of a POSA discloses or suggests claim 1 is deficient. I address some of those deficiencies in Petitioner’s analysis below.

1. [1b]: “SLC non-volatile memory module comprising a plurality of *individually addressable blocks*”

127. Limitation [1b] recites: “at least one SLC non-volatile memory module comprising a plurality of **individually erasable blocks**.” I understand that the Petition cannot incorporate by reference arguments from an expert’s declaration. In my opinion, the Petition itself fails to identify anything from Sutardja corresponding to the “**individually erasable blocks**.” Petition, 50. The Petition never addresses this portion of limitation [1b] at all.

128. Furthermore, Petitioner cites “Paragraph [106]” of Sutardja, but this paragraph discloses only “nonvolatile memory” and says nothing about “individually erasable blocks.” Ex. 1042, ¶106.

2. [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*,...”

129. Limitation [1d] recites: “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.” In my opinion, Petitioner fails to show that Sutardja discloses this “list of logical address ranges having a *minimum quanta of addresses*” as claimed. Petition, 51-52. The Petition never addresses this portion of limitation [1d] at all.

130. In any event, Petitioner’s cited portions of Sutardja do not disclose the “minimum quanta of addresses.” Petitioner points to the Abstract and ¶107 of Sutardja, but, in my opinion, neither discloses or suggests a “minimum quanta.” Petition, 51. Instead, these portions merely reference mapping “logical addresses” in general terms. Ex. 2042, abstract, ¶107.

3. [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”

131. Limitation [1e] recites: “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.” In my opinion, Petitioner fails to show that Sutardja discloses or suggests

“remap[ping]” an “entry” that “fails a data integrity test...[from MLC] to...SLC.”
Petition, 52-53.

132. For limitation [1e], Petitioner relies on ¶¶134-139 and ¶151 of Sutardja, which describe “degradation testing” of a nonvolatile memory during “periods of inactivity.” Ex. 1042, ¶135. A “degradation testing module” “provide[s] addresses and data”—that is, test data—to the memory. *Id.* The degradation module then performs a data integrity test *on that test data*; specifically, the module “may determine a degradation value for [a tested] physical address” and a “wear leveling module...may adapt its mapping based on the degradation value measured[.]” *Id.*, ¶¶135-138. Sutardja additionally explains that, based on this degradation value, “the wear leveling module...may assign *all new writes* to [a different] memor[y].” *Id.*, ¶139; *see also id.*, ¶151.

133. The Petition acknowledges that Sutardja’s degradation testing, which Petitioner relies on for limitation [1e], uses only *test* data and only maps *new* writes based on test results. Petition, 52 (“Fig. 7D and paragraph [151] explain the *test*...resulting in a ‘degradation value’ for that physical address.”), 52 (“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”), 53 (“Paragraph [0135] discloses periodically *testing* the...NVM”).

134. Accordingly, unlike the Challenged Claims, Sutardja does not disclose transferring the respective contents of blocks based on the degradation testing—rather, it makes “assignment decisions” and to perform “assign[ing]”/redirecting of “*new writes*.” Ex. 1042, ¶139.

135. In my opinion, however, limitation [1e]—unlike Petitioner’s cited embodiment in Sutardja—requires *remapping an entry of addresses that fails a data integrity test* to SLC. Sutardja, by contrast, performs a data integrity test on *test data*, and based on the results, remaps a *different entry*—not the same “entry” as required by [1e]. Sutardja’s degradation testing involves only “test” data, which is not itself “remap[ped]” upon failure of the data integrity test (since there is no need to do so, given the data is test data). Accordingly, in my opinion, the Petition fails to demonstrate that Sutardja discloses or suggests limitation [1e].

- 4. [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”**

136. Limitation [1f] recites: “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.” In my opinion, the Petition fails to clearly map the language of limitation [1f] to the disclosure of Sutardja. Petition, 54. Instead of providing a

clear mapping to the prior art, Petitioner ambiguously and conclusorily cites multiple embodiments of Sutardja. *Id.* Specifically, Petitioner cites a first embodiment, including ¶¶110-111 and 121 of Sutardja, allegedly disclosing a “wear leveling module 260 may track and store the number of write and/or erase operations performed on the blocks of the ... memories”. Petition, 54. Petitioner then cites another embodiment, including Figure 7A and ¶¶146-147 of Sutardja, allegedly disclosing “receiving write frequencies for logical addresses.” *Id.* (emphasis in original).

137. The Petition, however, does not make clear how these different embodiments map to limitation [1f], either individually or in tandem. For example, in my opinion, Petitioner does not explain whether it is proposing to combine these embodiments, or whether it is presenting these embodiments as *alternative* reads for this Ground.

138. Notwithstanding, Petitioner’s Ground also fails when “blocks” is properly construed to require “physical” blocks. Petitioner cites Sutardja’s embodiment in Fig. 7A and ¶¶146-147, and admits that, with this embodiment, Sutardja’s controller determines the “write frequencies for *logical* addresses.” Petition, 54. Limitation [1f], however, requires that the controller determine which *physical* blocks “are accessed most frequently.”

5. [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”

139. Limitation [1g] recites: “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.” In my opinion, for this limitation, Petitioner vaguely references multiple embodiments from Sutardja without clarifying how these embodiments map to limitation [1g], individually or as a group. Petition, 55 (citing Fig. 7A and ¶146, describing the “transfer to SLC...of the most frequently written LBAs”), 55-56 (citing Fig. 7C and ¶149, describing mapping based on “the number of write operations to [a] first physical block...during a predetermined time”), 56 (citing Fig. 7E and ¶153, describing mapping based on whether “the wear level of...MLC...is greater than a predetermined threshold”).

140. Notwithstanding, in my opinion, even when Petitioner’s cited embodiments are considered individually, each embodiment does not disclose the requirements of limitation [1g].

141. *First*, regarding Petitioner’s reliance on the embodiment of Fig. 7A and ¶146, this Ground fails for two reasons. To begin with, Petitioner’s argument relies on an incorrect construction of “blocks” that improperly includes “logical” blocks. *See* Petition, 55 (“Paragraph [0146] discloses the controller mapping the *logical*

addresses having write frequencies...”). However, “blocks” should be construed to mean only “physical” blocks. *See* Section VII.A.

142. Additionally, Sutardja’s embodiment of Fig. 7A and ¶146 does not disclose “*transferring the respective contents of those blocks* to the at least one SLC non-volatile memory module,” as required by [1g]. Rather, this embodiment describes “*map[ping]*” “logical addresses where data *is to be written*” (i.e., data that has not been written yet) to a first or second memory based on the “write frequencies” for those addresses. Ex. 1042, ¶146. Thus, in my opinion, ¶146 relates to *directing a subsequent (new) write to a certain memory* (a “first NVS memory” or “second NVS memory”), as opposed to transferring contents of a block that is already in MLC/SLC to SLC, as required by [1g].

143. *Second*, regarding Petitioner’s reliance on the embodiment of Fig. 7C and ¶149, Petitioner again fails to explain how this embodiment results in the “*transferring the respective contents of those blocks* to the at least one SLC non-volatile memory module,” as required by [1g]. Petition, 55-56. Sutardja’s ¶149 describes a “data shift analysis” where, “if a number of write operations to a first block of the first NVS memory...is greater than...a predetermined threshold[,]” then the “control *maps* the logical addresses that correspond to the first block to a second block of the second NVS memory....” Ex. 1042, ¶149. In my opinion, Paragraph

149 does not state that the existing contents of the block is “transferred” as a result of this “map[ping] [of] the logical addresses that correspond to the first block to a second block of the second NVS memory,” as required by [1g]. *Id.* Instead, in my opinion, a POSA would have understood that, just like Sutardja’s embodiment of ¶146 discussed above, this “map[ping] [of] the logical addresses that correspond to the first block to a second block of the second NVS memory” would result in ***directing a subsequent (new) write to a certain memory*** (a “first NVS memory” or “second NVS memory”). *Id.*; *see also, e.g.*, Ex. 1042, ¶143 (“The mapping module...may map the logical addresses that are ***to be written*** more frequently...to the physical addresses of second solid-state nonvolatile memory.”); *contra* Ex. 1042, ¶131 (describing data that is “**remapped and moved**”).

144. In the same paragraph where Petitioner cites the embodiment of Fig. 7C and ¶149, it also ambiguously cites Sutardja’s unrelated ¶¶126 and 131-132. Petition, 56. The Petition does not make clear how it is mapping these additional paragraphs to the claim language, or how their unrelated disclosures apply to that of ¶149. *Id.* Petitioner merely suggests that ¶¶126 and 131-132 disclose where data is “moved” as part of “wear leveling” and “remap[ping].” Petition, 56. But, in my opinion, this disclosure is entirely unrelated to that of ¶149. For example, ¶126 relates to “mov[ing]” data that “is ***unchanged*** over a period of time,” whereas ¶149

relates to “map[ping]” data that is frequently written (i.e., frequently changed). Ex. 1042, ¶¶126, 149. And ¶¶131-132 instead disclose “remap[ing] and [mov[ing]]” data from a memory “*to create space*” when a memory has become “full.”

145. *Third*, regarding Petitioner’s reliance on the embodiment of Fig. 7E and ¶153, this embodiment fails for similar reasons as the embodiment of ¶149 and Fig. 7C. Petition, 56. In my opinion, Sutardja’s ¶153 only states that the “control *maps* all the logical blocks to physical blocks of the second NVS memory,” and says nothing about “*transferring the respective contents of...blocks*,” as required by [1g]. Petitioner also ambiguously cites ¶¶128, 164, and 168 of Sutardja, but again fails to make clear how these paragraphs relate to the disclosure of ¶153. Petition, 56 (including an “*Also*” cite to ¶¶128, 164, and 168). Even so, the Petition does not allege that ¶¶128, 164, and 168 disclose the missing limitation—“*transferring the respective contents of...blocks*.”

146. Furthermore, Petitioner’s citation to the embodiment of Fig. 7E and ¶153 also fails because this embodiment describes mapping “*all* the logical blocks” from a first memory to a second memory once the “wear level of the first NVS memory is greater than a predetermined threshold.” Ex. 1042, ¶153. Sutardja ¶153 describes mapping the first memory to the second memory when the memory *as a whole* reaches a certain “wear level.” *Id.* In my opinion, in contrast, limitation [1g]

requires determining the “blocks that receive the most frequent writes,” and transferring the respective contents of those blocks.

E. Dependent Claims 2-11

147. For dependent claims 2-11, the Petition’s Grounds also state that these claims are disclosed or suggested by Gavens, Moshayedi, and Sutardja, respectively. The Petition’s analysis of claims 2-11 is flawed at least due to the above-discussed deficiencies of the Petition’s analysis of independent claim 1.

Declaration of Sunil P. Khatri, Ph. D.
IPR2025-00212
U.S. Patent No. 8,891,298

IX. CONCLUSION

148. I declare that all statements made herein on 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 or Title 18 of the United States Code.

Respectfully submitted,

Date: March 23, 2025

By: 
Sunil P. Khatri, Ph.D.