IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re patent of Gonzalez *et al.*:	Petition for *Inter Partes* Review
U.S. Patent No. 8,050,095	Attorney Docket No.: 337722-000080.095
Issued: November 1, 2011	Customer No.: 26379
Title: Flash Memory Data Correction and Scrub Techniques	Petitioner: Apple Inc. Real Party-in-Interest: Apple Inc.

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 8,050,095

Mail Stop Patent Board
Patent Trial and Appeal Board
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Pursuant to 35 U.S.C. §§ 311-319, Apple Inc. ("Petitioner") hereby petitions the Patent Trial and Appeal Board to institute an *inter partes* review of claims 1-9, 11-14 of United States Patent No. 8,050,095 (the "'095 patent") (Ex. 1001).

TABLE OF CONTENTS

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

TABLE OF CONTENTS
(continued)

Page

Exhibit Number	**Description**
1001	U.S. Patent No. 8,050,095 to Gonzalez
1002	Prosecution File History For 8,050,095 to Gonzalez
1003	Declaration of Dr. R. Jacob Baker in Support of Petition
1004	U.S. Patent No. 5,341,339 to Wells
1005	U.S. Patent No. 6,151,246 to So
1006	U.S. Patent No. 6,396,744 to Wong
1007	U.S. Patent No. 5,838,614 to Estakhri
1008	Excerpts from *Designing with FLASH MEMORY: The Definitive Guide to Designing Flash Memory Hardware and Software for Components and PCMCIA Cards* by Brian Dipert & Markus Levy Annabooks (1993, 1994)
1009	U.S. Patent No. 5,485,595 to Assar
1010	PC Card Standard, Volume 7, Media Storage Formats Specification
1011	*A Floating Gate and Its Application to Memory Devices*, D. Kahng and S. M. Sze

1012	*New Ultra High Density EPROM and Flash EEPROM with NAND Structure Cell*, Fujio Masuoka *et. al.*
1013	Toshiba Web at flash25.toshiba.com
1014	U.S. Patent No. 5,418,752 to Harari
1015	U.S. Patent No. 6,362,049 to Cagnina
1016	Intel FDI User Manual
1017	*Cleaning Policies in Mobile Computers Using Flash Memory*, M.-L. Chiang & R.-C. Chang
1018	*Managing Flash Memory in Personal Communication Devices*, Mei-Ling Chiang *et. al.*
1019	U.S. Patent No. 5,740,395 to Wells
1020	U.S. Patent No. 5,940,861 Brown
1021	U.S. Patent No. 7,012,835 to Gonzalez
1022	U.S. Publication No. 2003/0046487 to Swaminathan
1023	Intel Series 2 Flash Memory Cards Data Sheet
1024	*Curriculum Vitae* of Dr. R. Jacob Baker

I. MANDATORY NOTICES

A. Real Party-in-Interest

Pursuant to 37 C.F.R. § 42.8(b)(1), the real party-in-interest is Apple Inc.

B. Related Matters

Pursuant to 37 C.F.R. § 42.8(b)(2), Petitioner states that Longitude Flash Memory Systems S.A.R.L. ("Patent Owner") is asserting U.S. Patent 8,050,095 (the "'095 patent") against the Real Party-In-Interest in a suit filed September 23, 2014, styled *Longitude Licensing Ltd., and Longitude Flash Memory Systems S.A.R.L. v. Apple Inc.*, Case No. 3:14-cv-4275, pending in the United States District Court for the Northern District of California (the "Related Litigation"). Petitioner has filed, or soon will file, petitions for *inter partes* review of U.S. Patent Nos. 6,510,488; 6,763,424; 6,831,865; 6,968,421; 7,012,835; 7,224,607; 7,120,729; 7,181,611; 7,657,702; 7,818,490; 7,970,987;and 8,316,177.

As of the filing of this petition, no other judicial or administrative matters are known to Petitioner that would affect, or be affected by, a decision in an *inter partes* review of the '095 patent.

C. Lead and Back-up Counsel

Lead counsel for this matter is Brent Yamashita (USPTO Reg. No. 53808), and back-up counsel for this matter is Edward Sikorski (USPTO Reg. No. 39478)

and Katherine Cheung (USPTO Reg. No. 70525) all at the e-mail address: Apple-Longitude-IPR@dlapiper.com. The postal and hand delivery address for both is DLA Piper LLP (US), 2000 University Avenue, East Palo Alto, California, 94303, and the telephone and fax numbers are (650) 833-2348 (for phone) and (650) 687-1206 (for fax).

D. Service Information

Pursuant to 37 C.F.R. § 42.8(b)(4), papers concerning this matter should be served on the following email address: Apple-Longitude-IPR@dlapiper.com.

II. GROUNDS FOR STANDING

Pursuant to 37 CFR § 42.104(a), Petitioner certifies that the ’095 Patent is available for *inter partes* review, and Petitioner is not estopped or barred from requesting *inter partes* review challenging the ’095 Patent on the grounds identified in this petition.

III. RELIEF REQUESTED

Petitioner asks that the Board review the accompanying prior art and analysis, institute a trial for *inter partes* review of claims 1-9 and 11-14 of the ’095 Patent, and cancel claims 1-9 and 11-14 as invalid for the reasons set forth below.

IV. THE REASONS FOR THE REQUESTED RELIEF

The full statement of the reasons for relief requested is as follows:

A. Summary of Reasons

- **Challenge #1:** Claims 1-9, 11-14 of the ’095 are anticipated by U.S. Patent No. 5,341,339 (“Wells”) (Ex. 1004).
- **Challenge #2:** If Challenge #1 to claims 11-14 is rejected, claims 11-14 of the ’095 are rendered obvious by Wells in view of the knowledge of a person of ordinary skill in the art, or in view of Intel Series 2 Flash Memory Cards Data Sheet (“Series 2”) (Ex. 1023).
- **Challenge #3:** Claims 1-3 and 5-9 of the ’095 are anticipated by U.S. Publication No. 2003/0046487 (“Swaminathan”) (Ex. 1022).
- **Challenge #4:** Claims 4 and 11-14 of the ’095 are rendered obvious by Swaminathan in view of the knowledge of a person of ordinary skill in the art, or in view of Series 2.

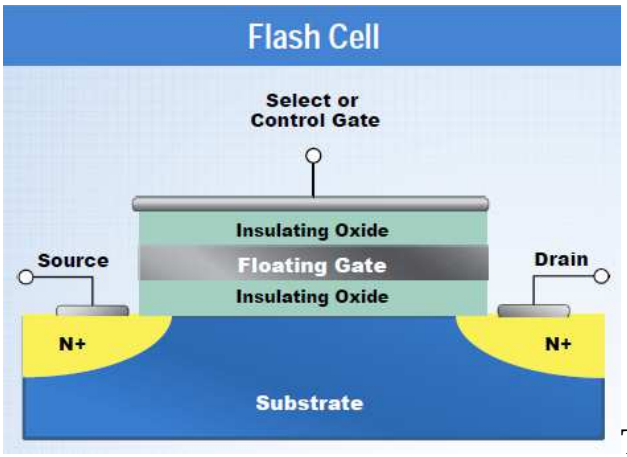
B. Relevant Background Technology

1. Overview of Flash Memory

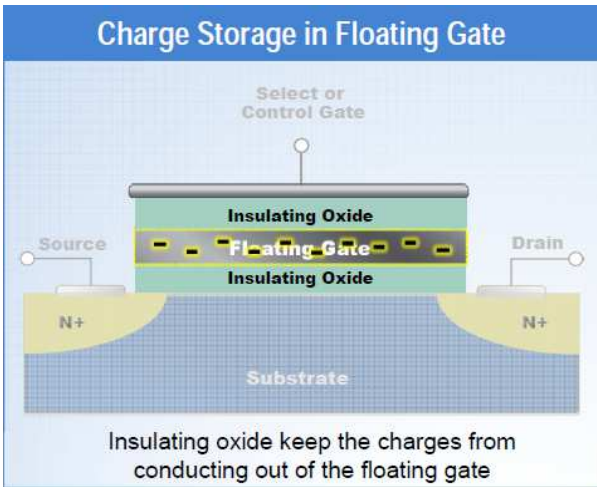
Flash memory is a type of solid state semiconductor non-volatile memory. These devices are now ubiquitous in consumer electronic devices as data storage devices, and are increasingly used as a replacement for magnetic disk drives even in desktop computers. Ex. 1003 at ¶ 15, Declaration of Dr. Jacob Baker (“Baker Decl.”).

Flash memory typically comprises an array of flash memory cells organized in rows and columns, as in conventional memory systems (such as DRAM or SRAM). Ex. 1003 at ¶¶ 24, 32, 33. Each flash memory cell utilizes a floating gate within a field effect transistor ("FET") to store electrical charge. Ex. 1003 at ¶ 19.

Shown below is an illustration of a cross-sectional view of a typical flash memory cell with a floating gate added to a standard FET structure.



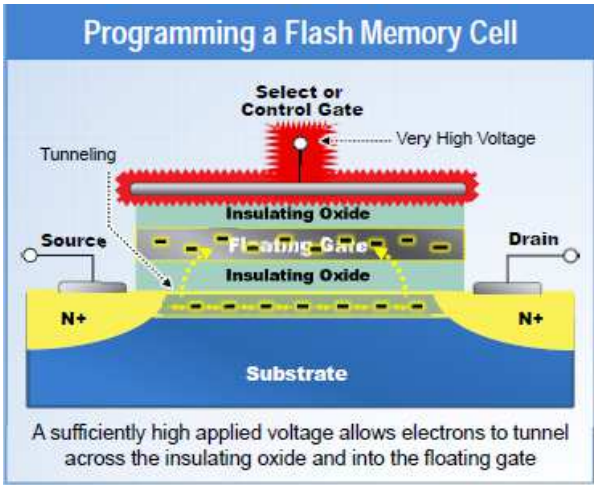
Taken from Ex. 1003 at ¶ 20.

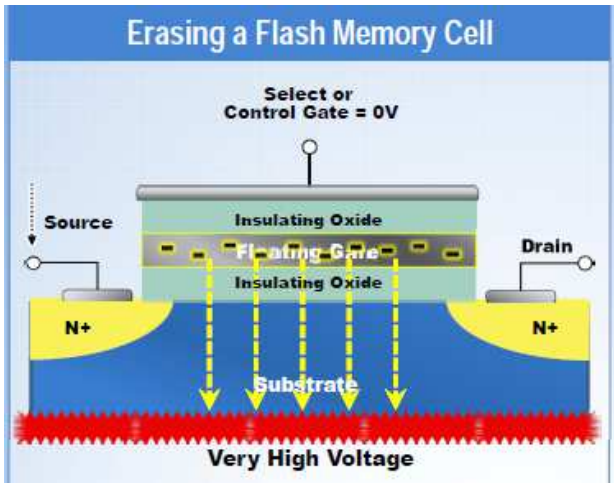


Taken from Ex. 1003 at ¶ 21.

The amount of electrical charge stored in the floating gate can be used to represent data bits ("1" or "0"). Ex. 1003 at ¶¶ 21, 22. Since the "floating gate" is electrically insulated from the terminals of the FET, charge cannot readily conduct into or out of the floating gate, which allows long-term storage of the charge, even when power is removed from the device. *Id*. at ¶ 19.

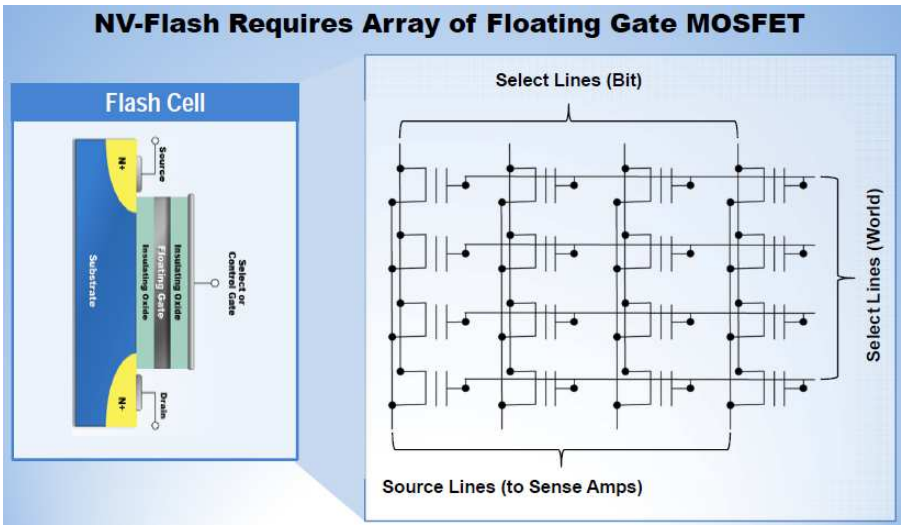
In order to utilize such floating gate FET's as memory cells, there must be a way to controllably add or remove electrical charge from the floating gate. This can be accomplished by applying high voltage differences across the terminals of the memory cell (including across the insulating oxide allowing the floating gate to float). *See e.g.*, Ex. 1008 at 27, 28, 33, 34, 36. Adding charge to the floating gate is termed "programming" (changing the cell from "1" state to "0" state) and removing charge is termed "erasing" (changing from "0" to "1"). Ex. 1003 at ¶¶ 23-24. This is shown in the following illustrations:





When these cells are used in real-world devices, electrons are moved back and forth across the floating gate's oxide region as the memory undergo numerous program and erase cycles. This movement of the electrons through the material creates stress on the insulating oxide layer and eventually causes the oxide to break down. When the oxide breaks down, the cell short circuits and becomes unusable. Ex. 1008 at 40. Therefore, it is important to equalize as much as possible the programming and erasing activity across all memory cells in the flash array so that certain memory cells do not fail prematurely from excessive use. This is a process known as "wear leveling," and was a well-known known and common practice by 2003. *Id*. at 263-265. Ex. 1003 at ¶¶ 44, 45, 46. In general, wear leveling involves copying data from one block (the "source" block) to another block (the "destination" block) to balance the amount of stress to be incurred by the source block and the destination block. *Id*.

In real-world products, it is advantageous to integrate as many memory cells into as small an area as possible to maximize the storage density. *Id*. at ¶¶ 24, 31-35, 47. Therefore, these cells are electrically interconnected together in an array with rows and columns of cells.



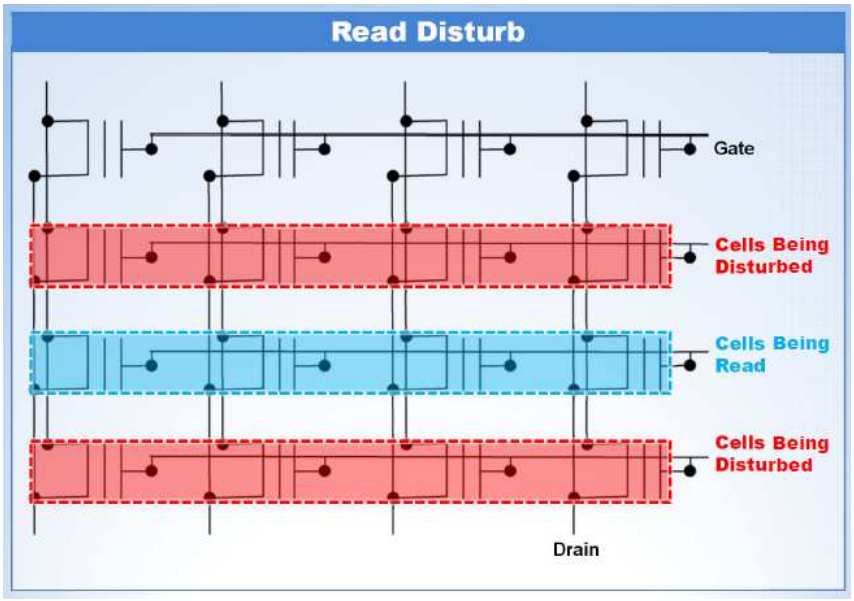
Ex. 1003 at ¶24.

In order to make the storage device more compact, the array of memory cells are interconnected in a way that a large group of such cells (typically a few kilobytes or more) must be erased simultaneously together. *Id.* at ¶ 24. This means that it is not possible to change even a single bit that has already been programmed (*i.e.* changed from "1" to a "0") back to a "1" state without erasing the entire block.

With such a constraint, overwriting previously stored data is clearly not as straightforward as it would be as with traditional magnetic storage media. *Id.* at ¶ 25. Therefore, most real-world flash memory systems incorporate a form of flash translation mechanism, and implement a "non-in-place" update scheme. A host request to overwrite stored data (*i.e.* data with the same associated logical address) is actually programmed into a new physical location in memory, and thus avoids having to erase the entire block where the original data is kept. *Id.* at ¶ 26. The flash translation mechanism is used in flash memory systems to keep track of this

translation from the host logical address to the actual physical memory address where the data can be located. This is often implemented as a look-up table that correlates the logical address (referenced by the host) for a piece of data and the corresponding physical address (referenced by the flash controller) where the current copy of the data is actually stored. *Id.* at ¶¶ 24-29.

In addition, this highly integrated arrangement of memory cells also causes greater electrical coupling between neighboring cells in a memory array, since they now share common word lines or bit lines. As was well-known at the time of the '095 patent, reading and/or writing to one part of the memory array inevitably exposes neighboring cells (that are not being accessed) to high voltages that can disturb the charges in the floating gates in those cells.



Ex. 1003 at ¶48.

When enough charge has inadvertently leaked out of (or into) the floating gate of a cell, this will appear as a bit error (i.e., a "1" will turn into a "0," or vice-versa) during a read operation. This is known to those of ordinary skill in the art as a read/write disturb error. To minimize the likelihood of read/write disturb errors, it was also known to "refresh" or "scrub" the stored data periodically. This involves reading the stored data and correcting any errors before the errors become uncorrectable. The corrected data is thereafter programmed back into the flash memory device. *Id*. at ¶¶ 49, 50, 51, 52.

In general, these operations (*i.e.* wear leveling, data scrub/refresh) are executed by the flash memory controller in the background and are implemented in a way that does not interfere with the customary memory access demands of the host system (i.e., normal read and write operations). *Id*. at ¶¶ 56-63.

C. Overview of the '095 Patent

The '095 patent, titled "Flash Memory Data Correction and Scrub Techniques," was filed on November 12, 2010, claims priority to an application No. 10/678,345 filed on October 3, 2003, and issued on November 11, 2011 to Carlos J. Gonzalez and Kevin M. Conley.

The '095 patent relates to wear-leveling and minimizing read/write disturb errors. The '095 patent "relates generally to operation of non-volatile flash memory systems, and more specifically, to techniques of refreshing and correcting

data stored therein." Ex. 1001 at 1:19-21. The claims of the '095 are directed to "housekeeping operations" on a non-volatile memory. The housekeeping operations claimed in the patent include data scrubbing and wear leveling operations. *See id*. at claim 7. The '095 patent discloses that "it may be desirable in order to maintain performance of the memory system to defer a scrub read [operation] even after the decision to perform a scrub read has been made… In such a case, the scrub operation parameters that have been decided upon are stored and processed at a later time when it is most convenient to the host." *Id.* at 15:36-51. *See also id.* at 19:41-48.

The scope of the independent claims 1, 5 and 11 are similar. Claim 1 covers a method of operation of a re-programmable, non-volatile memory system that monitors the activity of the host, and a simple binary decision is made dependent on the monitored activity. As claimed, when the memory system identifies a first pattern of activity, it enables a housekeeping operation. However, if it identifies a second pattern of activity, it does not enable a housekeeping operation. The claim further requires that this housekeeping operation be of a type that is not required for execution of a command from the host.

Claims 5 and 11 also cover a method of operating a re-programmable non-volatile memory system with simple binary decision logic. For these claims, when a housekeeping operation has been asserted, the system will monitor a parameter of

host activity, and if the parameter meets a predefined condition, execution of the housekeeping operation is enabled. If not, execution of the housekeeping operation is not enabled. For example, the system can check whether the host is accessing the memory, and if it is not accessing the memory, then a housekeeping operation can be performed. One alleged benefit of this system is to increase efficiency of the system by performing housekeeping operations during periods when host activity is relatively low. The housekeeping operation as claimed is likewise one that is not required for execution of a command from the host.

D. Level of Ordinary Skill in the Art

A person of ordinary skill in the art ("POSITA") is a hypothetical person who is presumed to have known the relevant art at the time of the alleged invention. *Custom Accessories, Inc. v. Jeffrey-Allan Indus., Inc.*, 807 F.2d 955, 962 (Fed. Cir. 1986). Petitioner submits that a person of ordinary skill in the art at the time of the '095 patent would have a minimum of a Bachelor of Science degree in electrical engineering, computer science, computer engineering, or a related field, and at least two years of experience working in the field of semiconductor memory design, or equivalent. Ex. 1003 at ¶ 78. Such a person would have been capable of understanding the '095 patent and applying the prior art references as explained in this Petition.

E. Claim Construction

Pursuant to 37 C.F.R. §§ 42.100(b) and 42.204(b)(2), this petition presents claim analysis that is consistent with the broadest reasonable construction in light of the specification. Proposed claim constructions contained below are presented using the broadest reasonable interpretation standard, which is applied solely for the purposes of *inter partes* review. Because the standards of claim interpretation used by the Courts in patent litigation are different from the claim interpretation standards used by the Office in *inter partes* review proceedings, Petitioner reserves the right to advocate a different claim interpretation in any other forum in accordance with the claim construction standards applied in such form. Petitioner further notes that in the Related Litigation, Patent Owner takes the position that none of the claim terms in the ’095 patent require construction and each should be given its plain and ordinary meaning.

1. “housekeeping operation”

The term “housekeeping operation” is not a term of art. Ex. 1003 at ¶ 83. The’095 patent does not define what the term means, and moreover, the term only appears once in the written description. *See* Ex. 1001 at 13:29-32. Examining the surrounding language of independent claims 1, 5, and 11, the claimed “housekeeping operation” must not be an operation that is “required for execution of one of the commands received from the host.” Therefore, even under the

broadest reasonable interpretation, the claimed housekeeping operation must not be one that is executed via a command from the host system.

In addition, the surrounding claim language in dependent claim 7 expressly recites that a wear leveling operation may be within the scope of the claimed "housekeeping operation." Similarly, another operation within the scope of "housekeeping operation" would be a "scrub housekeeping operation." Accordingly, for the purpose of these proceedings, Petitioner submits that the term "housekeeping operation" at least includes an operation, such as wear leveling or data scrubbing that is implemented by the memory system, but not in response to a host command. Ex. 1003 at ¶¶ 83-87.

2. "host"

The '095 patent states that "for the purposes used herein, a 'host system' is a system that generally has a functionality other than data storage, but which also either connects to the memory system, or has a memory system embedded in it. There can be host systems whose sole purpose is data storage." Ex. 1001 at 7:50-54. The '095 Patent further provides examples various types of host systems that include "personal computers, notebook computers, personal digital assistants (PDAs), various data communication devices, digital cameras, cellular telephones, portable audio players, automobile sound systems, and similar types of equipment."

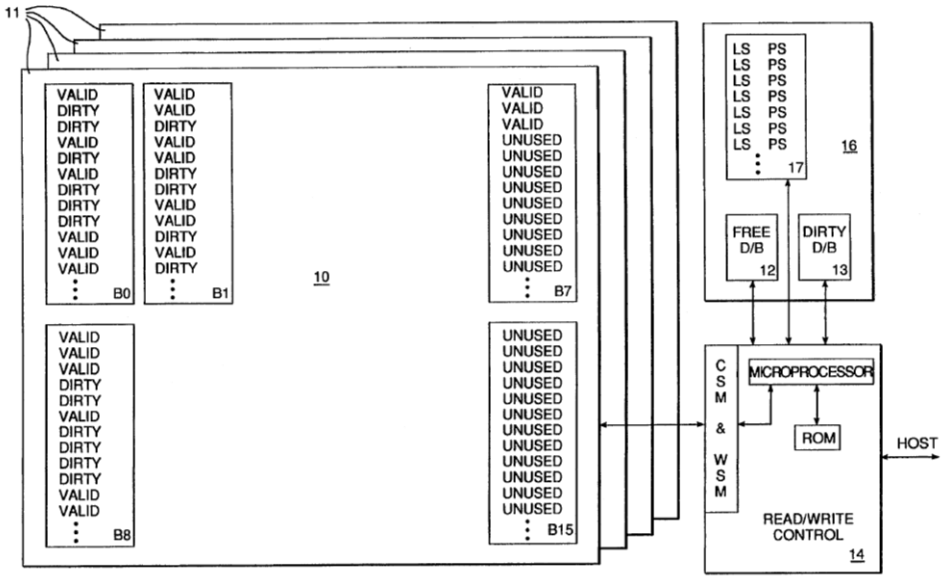
In the express definition provided, the host system generally has a functionality other than data storage, but can include systems whose sole purpose is data storage, so that portion of the definition is not limiting. Accordingly, "host" should be construed as "a system that connects to, or has embedded within it, the memory system." Ex. 1003 at ¶¶ 88-91.

F. Challenge #1: Claims 1-9 and 11-14 of the '095 are anticipated by U.S. Patent No. 5,341,339 ("Wells").

Wells anticipates claims 1-9 and 11-14 of the '095 Patent. Wells was filed on November 1, 1993 and claims a priority date of October 30, 1992, based on Ser. No. 969,467, and issued on August 23, 1994. Therefore, it is prior art against the '095 Patent under pre-AIA 35 U.S.C. §§ 102(a), 102(b), and 102(e).

Wells discloses the alleged point of novelty of the '095 patent and seeks to solve the same challenges described in the '095 patent. Wells describes a method of cleaning up a flash memory in the background as a "housekeeping operation" that can be interrupted by a host. Figure 1 of Wells depicts the memory 10 and the host interface, which includes a read/write controller 14.

FIG_1



Ex. 1004 at Figure 1.

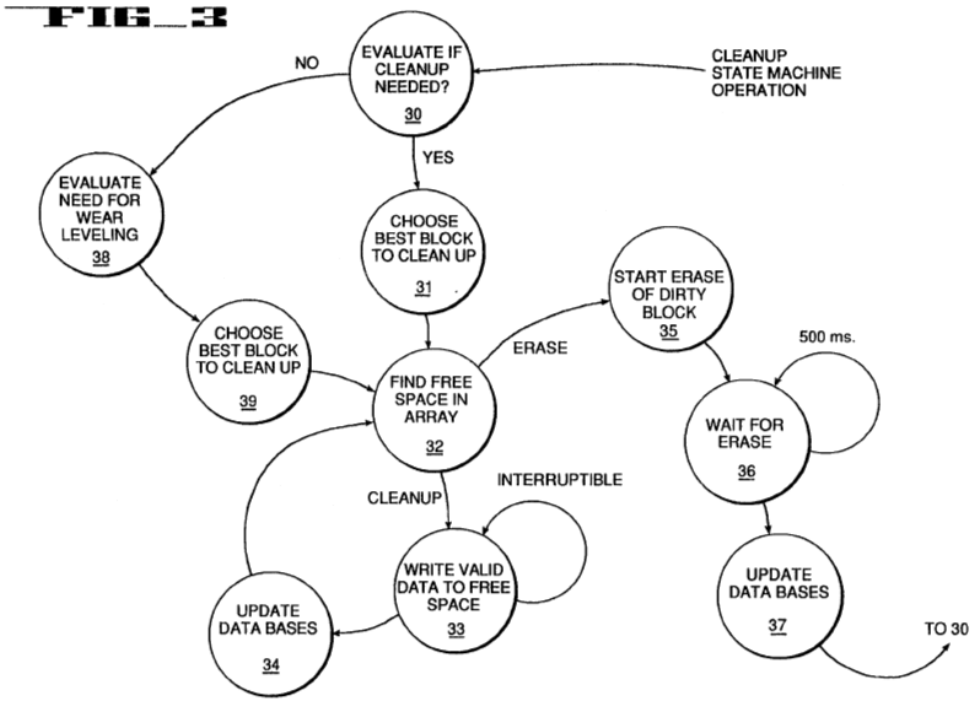
It is known that flash memory must be erased before it can be reprogrammed. Ex. 1003 at ¶ 25. Wells acknowledges the well-known problem that when a host system needs to modify pre-existing data that has already been stored in the flash memory (*i.e.* overwrite), it is not feasible to attempt to erase all of the data in a block, and then replace the modified information along with the unmodified information that had been erased. Ex. 1004 at 2:65-3:2.

Wells discloses that to overcome this problem, it was known to use a scheme where data that needs to be replaced is not immediately "erased" from the physical flash memory. By contrast, the new data is written to an "empty" sector located on an erased block (*i.e.* a different physical location), and the old sector (with the

original data still stored therein) is marked invalid, or "dirty." *Id.*at 3:3-22; 6:28-54.

Wells further discloses that under such an overwrite scheme, memory blocks would eventually still need to be erased in order to free up storage space to accept new data. *Id.* at 7:1-18. (This would occur, for example, when the host repeatedly overwrites previously stored data, leading to many invalid sectors. Ex. 1003 at ¶ 104.) Wells refers to this process as "cleaning up a block," whereby the memory controller (element 14 of Figure 1, apart from the host) moves any remaining valid data to a fresh block, and thereafter the old block is erased. Overall, because the erase operation need not be performed immediately before the overwrite operation, the advantage of this scheme is that erasure of blocks, which is a particularly slow operation, can occur in the background. Ex. 1004 at 3:22-40. Notably, Wells teaches that this cleanup operation is recognized as part of a wear leveling operation. *Id*. at 12:15-18. Moreover, Wells specifically teaches that the cleanup operation performed by the memory controller can be interrupted by the host. *Id.* at 11:26-41. This allows the entire cleanup operation to occur as a background task, and therefore, higher priority operations from the host can still be serviced by the memory system.

Figure 3 of Wells is a flow chart depicting the cleanup state machine implemented by the memory controller.



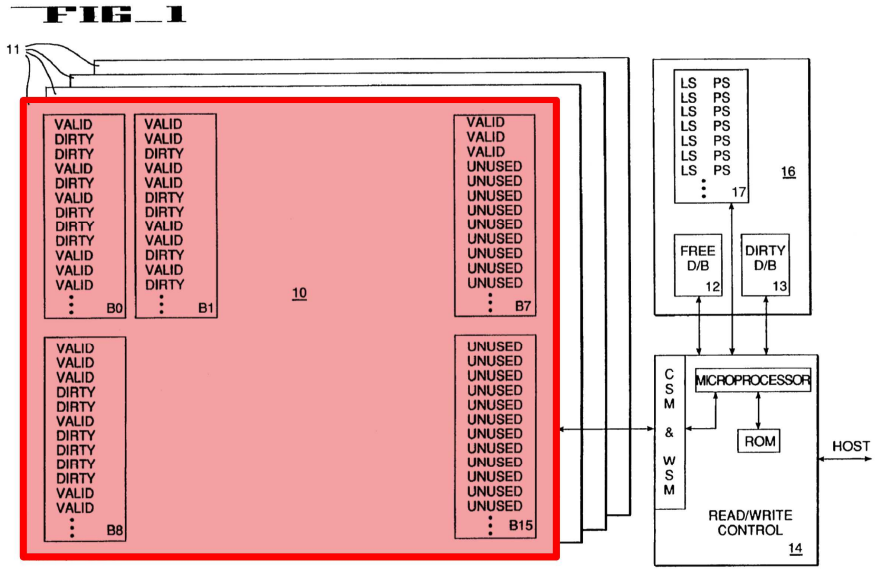
Id. at Figure 3.

Since the cleanup operation disclosed by Wells is a wear leveling operation, and further is not executed responsive to a host command, the Wells cleanup operation lies within the scope of the claimed housekeeping operation of the ’095 patent.

1. Claim 1:

a. A method of operating a re-programmable non-volatile memory system, comprising:

Wells discloses the features recited in this preamble. Wells discloses a system that uses a non-volatile flash electrically erasable programmable read only memory (EEPROM) array. Ex. 1004 at 1:10-14; 1:47-55; 1:65-2:30; 4:60-62; 5:43-56. The flash EEPROM array 10 is depicted in Figure 1:



Id. at Figure 1. Ex. 1003 at Table 1, Claim 1.[pre].

b. receiving commands from a host and executing the received commands,

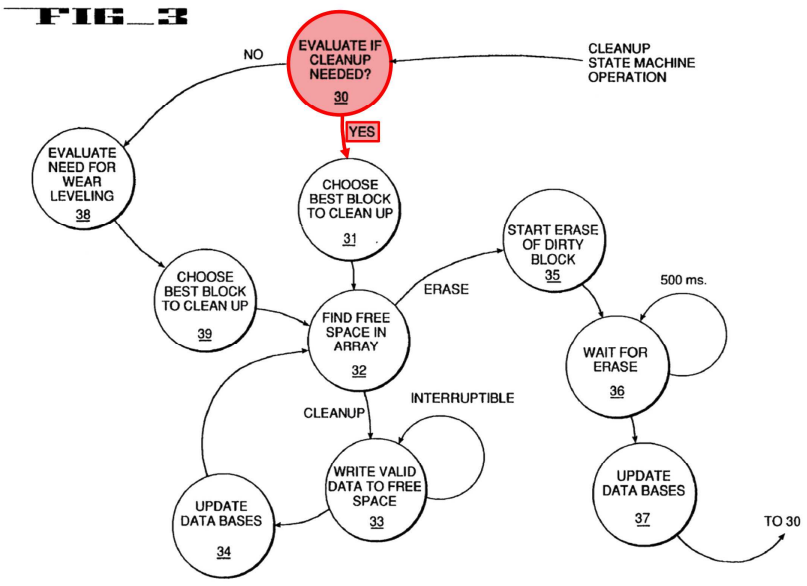
Wells discloses receiving commands from the host to write data to and read data from the array. Ex. 1004 at 6:18-27; 9:64-10:4. Wells discloses that "[w]hen a host begins writing data to be stored in the array (such as an application program to some block of the array which has been completely erased, the data to be stored is written sequentially, sector by sector, to that block until that block has been filled with data…At any point after writing is completed, the information may be read back from the array 10 by interrogating the block and sector at which the data is stored." *Id*. at 6:10-27. The read/write control circuit 14 of Figure 1 forms an interface between the host and the array for executing the host read and write commands. *Id*. at 9:64-10:4. Ex. 1003 at Table 1, Claim 1.[a].

c. monitoring patterns of activity of the host, at least in connection with the received commands, and

Wells discloses monitoring patterns of activity of the host by tracking the sectors that are marked dirty after a write command from the host is executed. As data is continually written and overwritten to the same logical address, more and more sectors will become dirty throughout the array. Ex. 1003 at ¶ 104. Eventually, this will require a housekeeping operation to be performed in order to make this space available for use again. Ex. 1004 at 7:1-8. The amount of free memory space and dirty memory space depends on the pattern of write/overwrite activity of the host, and is tracked in a "small data base 12 of the amount of free space and a second small data base 13 of the amount of dirty space in the static RAM 16 associated with the array…Whenever a sector is marked dirty, the size of the sector is added to the total of dirty space." *Id*. at 10:24-33. By monitoring the host activity, these databases can be maintained routinely so that "a ratio of dirty to dirty-plus-free space is always available as a valid measure of the percentage of the remaining array space which is dirty." *Id*. at 10:43-47. Ex. 1003 at Table 1, Claim 1.[b].

d. upon identifying a first pattern of host activity, a housekeeping operation is enabled to be executed, the housekeeping operation being of a type not required for execution of one of the commands received from the host, or

Wells discloses that an "evaluation takes place to decide whether a cleanup operation is necessary or not." Ex. 1004 at 10:15-16. Since the Wells cleanup operation is recognized as a wear leveling operation, it is within the scope of the claimed housekeeping operation. Ex. 1003 at ¶ 107. If the controller identifies a first pattern of host activity where 80% or more of the total memory space is dirty, then insufficient space is available and it is necessary to perform a cleaning operation. Ex. 1004 at 7:8-11; 10:15-24. This is depicted in Figure 3 at state 30, where the decision arrow "YES" will enable a cleanup operation to be executed.

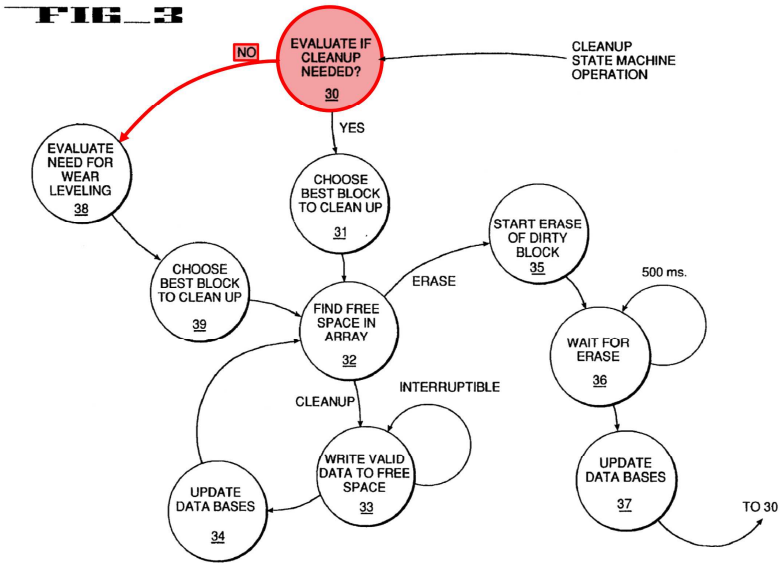


Id. at Figure 3. Once a cleanup is found to be necessary, the steps of the cleanup

operation (*e.g.* steps 31-37) are enabled. *Id*. at 10:50-12:11. Ex. 1003 at Table 1, Claim 1.[c].

e. upon identifying a second pattern of host activity different from the first pattern, execution of the housekeeping operation is not enabled.

By contrast, if the controller identifies a second pattern of host activity different from the first pattern, for example, where less than 80% of the total memory space is dirty, then execution of the cleanup operation is not enabled. Ex. 1004 at 7:8-11; 10:15-24. This is depicted in Figure 3 at state 30, where the decision arrow "NO" will not enable a cleanup operation to be executed.



Id. at Figure 3. Ex. 1003 at Table 1, Claim 1.[d].

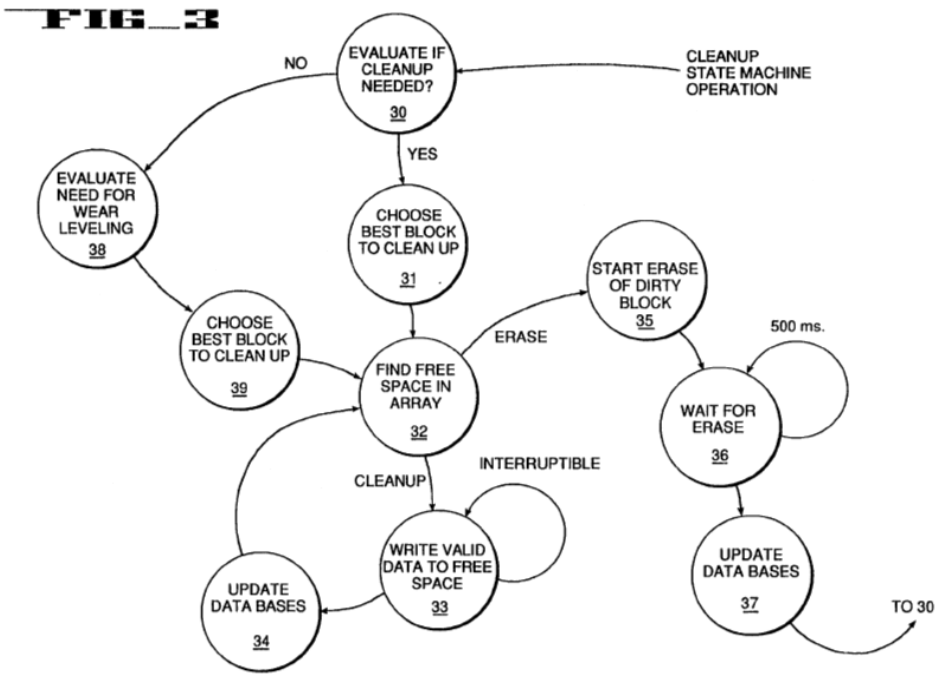
2. Claim 2: The method of claim 1, additionally comprising, in response the first pattern of host activity being identified, executing at least one portion of the enabled housekeeping operation.

Once the cleanup operation is enabled to be executed as described above, at least one portion of the cleanup operation steps 31-37 are executed. Ex. 1004 at Fig. 3; 10:50-12:11. ("If cleanup is found to be necessary, the program moves to a step 31 in which the block best suited to cleanup is selected") *Id*. at 10:50-52; ("Once the appropriate block to clean up has been chosen, the process moves to a step 32 at which available free sector space in other blocks is located. Once the space to store a valid sector has been located, the process moves to step 33 to write the valid data from the sector of the block being cleaned up to the available space.") *Id*. at 11:20-26. Ex. 1003 at Table 1, Claim 2.

3. Claim 3: The method of claim 2, wherein executing the enabled housekeeping operation includes reading a block of data from one location of the memory system and thereafter writing the read data into another location of the memory system.

In Figure 3, executing the cleanup operation steps includes step 31 where the program selects the block best suited to clean up. Ex. 1004 at 10:50-11:19. In step 32, free sector space in other blocks is located and in step 33, the valid data from the block being cleaned up is written to the available free space that was located in step 32. *Id*. at 11:20-26. In order to write the valid data from the cleanup block to the free space, the block must first be read from one location, the cleanup block,

and then written into another location, the free space. Ex. 1003 at Table 1, Claim 3.



Ex. 1004 at Figure 3.

4. Claim 4:

a. The method of claim 1, wherein receiving commands from a host and executing the received commands includes receiving and executing

As shown above in element 1(b), Wells discloses receiving commands from a host and executing the received commands.

b. (1) a write command to write data received from the host with the command into logical addresses of the memory specified by the write command, or

Wells discloses the host specifying a "logical sector number rather than a physical sector number" to specify a write data command. Ex. 1004 at 6:58-61. A

lookup table 17 stored in random access memory 16 within the array is used to determine the physical position in the array where the logical sector specified by the host command is located. *Id*. at 6:50-7:1. See also *id*. at 7:45-66; 8:37-53. The physical location of the logical address is retrieved from the lookup table and the data is written to the physical location specified. *Id*. Ex. 1003 at Table 1, Claim 4.[a].

c. (2) a read command to read data from logical addresses of the memory specified by the read command and send the read data to the host.

As with the write command element described above, the host specifies a logical address for the read data command. Then, the "data stored in any sector of the block 20 may be retrieved by determining the physical address of the sector number from the lookup table 17, using that address to go to the physical position on the block 20 where the sector number is stored, and retrieving the pointer to the beginning position of the data and the pointer to the beginning position of the sector whose number is stored immediately above the sector number being retrieved." Ex. 1004 at 8:43-51. Ex. 1003 at Table 1, Claim 4.[b].

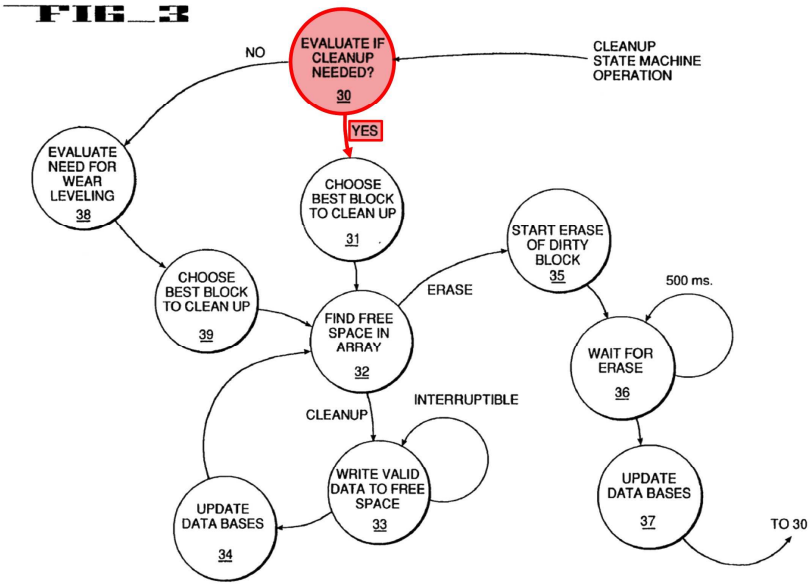
5. Claims 5:

a. Claim 5[pre]: A method of operating a re-programmable non-volatile memory system, comprising:

As shown in element 1.a above, Wells discloses a method of operating a re-programmable non-volatile memory system.

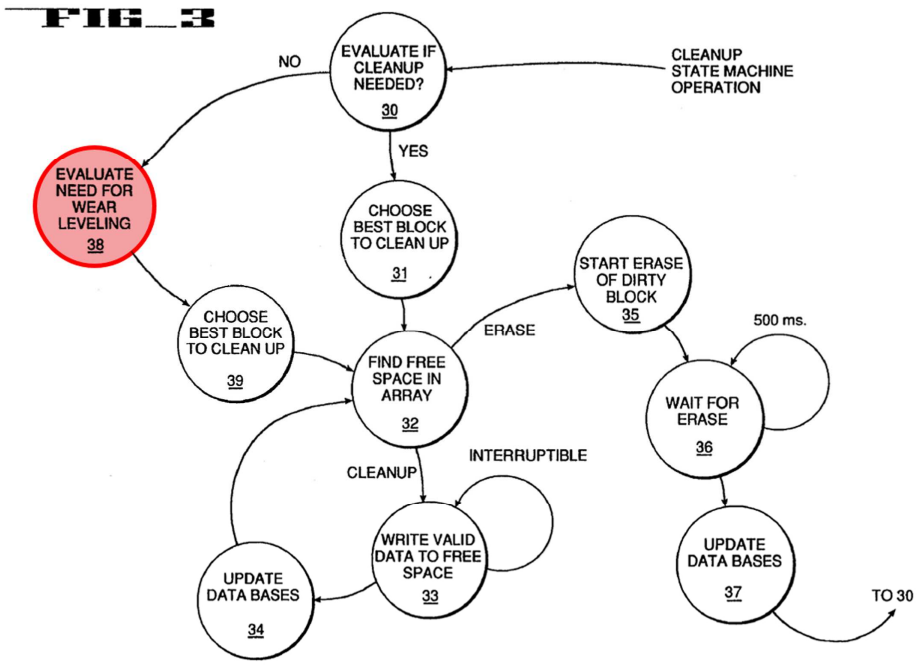
b. Claim 5.[a]: note when a housekeeping operation not required for execution of a command received from a host has been asserted,

Wells discloses noting when a housekeeping operation not required for execution of a command received from a host has been asserted. As explained above, the cleanup state machine will evaluate when a cleanup housekeeping operation is necessary by looking at the ratio of dirty memory space to total memory space and assert a cleanup operation if there is insufficient free space available for the memory to perform its operations. Ex. 1004 at 7:8-11; 10:15-24. This is shown in Figure 3, step 30, where the evaluation is performed. If a cleanup is necessary, then the decision arrow "YES" indicates that the cleanup operation will be asserted.



Id. at Figure 3. Ex. 1003 at Table 1, Claim 5.[a].

In addition, Wells also discloses a separate process in "step 38 in which a determination is made whether switching operation equalization (termed "**wear leveling**") is necessary." *Id*. at 12:12-15, emphasis added.



This additional wear leveling process begins with an evaluation as to whether any block has been cleaned up five hundred times less than the most cycled block of the array. *Id*. at 12:23-29. If so, the wear leveling operation is asserted.

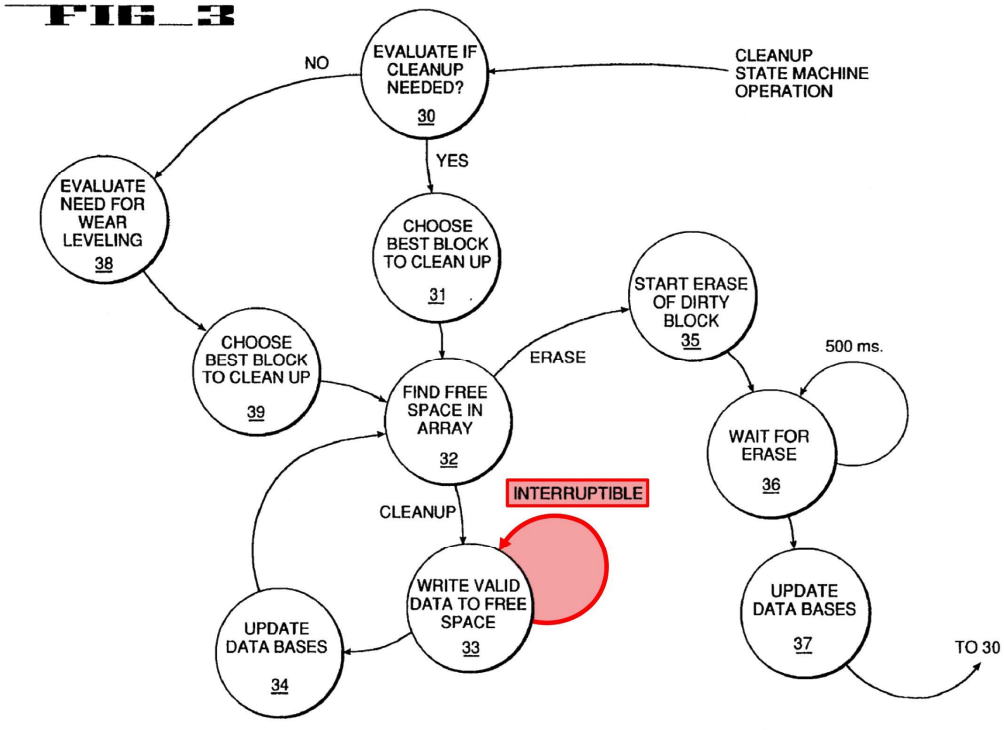
If the wear leveling operation is asserted, the program selects the best block to clean up in step 39 by considering the "dirty sectors and number of switching operations that have taken place." Id. at 12:36-37. However, the selection logic weighs the number of switching operations more heavily in order to focus on leveling the erase counts. *Id*. at 12:37-39; Ex. 1003 at Table 1, Claim 5.[a]. Therefore, this operation that can be asserted in step 38 is expressly a wear leveling operation, and accordingly meets the limitation of the claimed housekeeping operation. Once a block is selected in step 39, the remainder of the process is identical to the cleanup operation initiated at step 31 as described above, specifically steps 32-37 in Wells. The discussion of those steps below are therefore applicable to both the Wells cleanup operation as well as the Wells wear leveling operation.

c. Claim 5.[b]: determine at least one parameter of activity of the host, and

Wells discloses determining a parameter of activity of the host where the controller can determine if there is an interrupt request by the host so that

"operations desired by the host may be accomplished." Ex. 1004 at 11:31-32. *See also id.* at 11:26-41, 62-66, For example, if the host is idle, or perhaps only requires data stored in the volatile system memory, then the "inactivity" of the host with respect to the flash memory will give rise to a parameter that does not generate an interrupt. Ex. 1003 at Table 1, Claim 5.[b]. Alternatively, if the host system activity requires some high priority read/write operation to/from the flash memory, the activity of the host will give rise to a parameter that does generate an interrupt. *Id.* In this aspect, the interrupt level itself can be a parameter of activity of the host. *Id.* This is reflected in Figure 3 where step 33 is performed subject to the host's interruptions.

FIG_3



Ex. 1004 at Fig. 3.

d. Claim 5.[c]: if the determined at least one parameter meets at least one predefined condition, execution of the housekeeping operation is not enabled, but

As shown in step 33 of Figure 3 above, when the host interrupts the cleanup state machine with a host operation, the cleanup operation is halted to allow the operations requested by the host to be completed. Ex. 1004 at 11:31-32. Thus, when the parameter of activity meets the predefined condition (*i.e.* host interrupt being asserted), the housekeeping operation is not enabled so that the host operation can be completed. *Id*. at 11:26-41. Ex. 1003 at Table 1, Claim 5.[c].

e. Claim 5.[d]: if the determined at least one parameter does not meet the predefined condition, the housekeeping operation is enabled for execution.

Accordingly, when the parameter of activity does not meet the predefined condition (host interrupt not asserted), the housekeeping operation is enabled so that the cleanup state machine can continue being executed. Ex. 1004 at 11:35-41. Ex. 1003 at Table 1, Claim 5.[d].

6. Claim 6: The method of claim 5, wherein the housekeeping operation includes rewriting data from one location in the memory system to another location in the memory system.

Wells discloses performing a cleanup operation of a dirty block by writing the valid data from the dirty block to free sector space in another block. Referring to Figure 3 again, a dirty block is chosen for cleanup in step 31. Ex. 1004 at 10:50-51. “Once the appropriate block to clean up has been chosen, the process moves to

a step 32 at which available free sector space in other blocks is located. Once the space to store a valid sector has been located, the process moves to step 33 to write the valid data from the sector of the block being cleaned up to the available space." *Id*. at 11:20-26. Ex. 1003 at Table 1, Claim 6.

Similarly, the Wells wear leveling operation includes rewriting data from one location in the memory system to another location in the memory system. As discussed above, the wear leveling operation has in common with the cleanup operation steps 32-37, once a suitable block has been chosen in step 39. "Once the appropriate block has been chosen to implement the wear leveling cleanup operation, the program moves to the step 32 at which the cleanup begins. It then follows the same steps for cleanup as were described above." Ex. 1004 at 12:55-59. Therefore, the wear leveling operation will, in step 33, rewrite data from the block (selected in step 39) into the free block (selected in step 32). *Id*. at 11:20-26. Ex. 1003 at Table 1, Claim 6.

7. Claim 7: The method of claim 6, wherein the housekeeping operation data rewriting is performed as part of either a wear leveling or scrub housekeeping operation.

Wells discloses performing the data rewriting as part of a wear leveling operation. During the cleanup operation, "some substantial portion of this wear leveling has already been accomplished in the step at which the best block to cleanup was chosen in the main portion of the process." Ex. 1004 at 12:15-18.

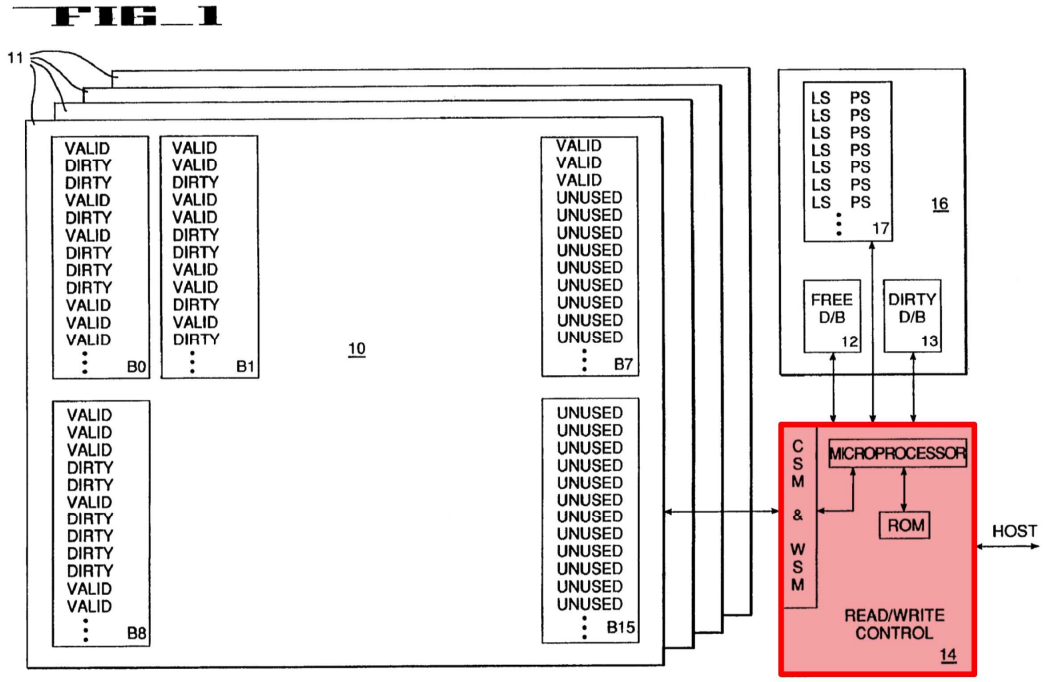
Therefore, the rewriting step of the cleanup operation is part of a wear leveling operation. Furthermore, Wells discloses that the selection logic in the additional wear leveling cleanup operation weighs the number of switching operations more heavily in order to focus on leveling the erase counts. *Id*. at 12:37-39; Ex. 1003 at Table 1, Claim 7. Accordingly, the rewriting step of this alternative operation is also part of a wear leveling operation.

8. Claim 8: The method of claim 5, wherein determining at least one parameter of activity of the host includes monitoring said at least one parameter during execution by the memory system of one of the commands received from the host.

As explained above, Wells discloses monitoring a parameter of activity of the host during step 33 in Figure 3 by monitoring whether the host is asserting an interrupt signal in order to perform a host operation. Ex. 1004 at 11:31-32 Ex. 1003 at Table 1, Claim 8. The program monitors the host's interrupt requests during execution by the memory system of the command received from the host while the "operations desired by the host" are accomplished. Ex. 1004 at 11:31-32. Once the operations requested by the host have been completed, the cleanup operation resumes. *Id*. at 11:32-41. Ex. 1003 at Table 1, Claim 8.

9. Claim 9: The method of claim 5, wherein the current received command is one of a group of commands that individually include data read and data write.

Wells discloses the memory receiving data read and data write commands from the host, which are controlled by read/write control circuit 14 in Figure 1.



Ex. 1004 at Figure 1. Wells describes the write operation to the memory array blocks, sector by sector, and "[a]t any point after writing is completed, the information may be read back from the array 10 by interrogating the block and sector at which the data is stored. *Id*. at 6:18-27. See also, *id*. at 6:28-57. The read and write operations are controlled by software in the read/write/erase control circuit 14, which Wells refers to also as simply the read/write control circuit. *Id*. at 9:64-66. The read/write control circuit includes "a command state machine and a

write state machine which form an interface with the chips of the array for reading, writing, and erasing." *Id*. at 10:1-4. Ex. 1003 at Table 1, Claim 9.

10. Claim 11:

a. Claim 11.[pre]: A memory system adapted to be removably connected with a host system, comprising:

Under the broadest reasonable interpretation standard, this preamble should not be found to be limiting. "In general, a preamble limits the invention if it recites essential structure or steps, or if it is 'necessary to give life, meaning, and vitality' to the claim. *Catalina Mktg. Intl. v. Coolsavings.com, Inc.*, 289 F.3d 801, 808 (Fed. Cir. 2002) ("*Catalina I*") (citing *Pitney Bowes, Inc. v. Hewlett-Packard Company*, 182 F.3d 1298, 1305 (Fed. Cir. 1999)). "A preamble is not limiting 'where a patentee defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention.'" *Id*. (citing *Rowe v. Dror*, 112 F.3d 473, 478 (Fed.Cir.1997)).

Here, this preamble "is intended to summarize the invention and its purpose and not to give any information that is indispensable to understanding the invention." *Catalina Mktg. Intl. v. Coolsavings.com, Inc.*, 115 Fed. Appx. 84, 90 (Fed. Cir. 2004) (*Catalina II*) (holding that a preamble was not limiting). Furthermore, the preamble here provides no antecedent basis to the rest of the claim elements. *In re Wertheim*, 541 F.2d 257, 270 (CCPA 1976) (holding that the preamble was not limiting because "[i]n no claim is the preamble relied on to

provide an antecedent basis for terms in the body."). The words "adapted to be removably connected" is found nowhere else in the claims, and does not provide to one of skill in the art "any information that is indispensable to understanding the invention." Ex. 1003 at Table 1, Claim 11.[pre].

Further, even if there are arguments to be made that the preamble is limiting, they must be viewed in the context of an *inter partes* review where the standard of claim construction is the broadest reasonable interpretation. This standard would necessitate a construction that the preamble of claim 11 is not limiting, since construing it as such would be both reasonable as well as the broadest way of viewing the preamble.

b. Claim 11.[a]: an array of re-programmable non-volatile memory cells organized into blocks of memory cells wherein the memory cells of the individual blocks are simultaneously erasable,

As described above, Wells discloses that "[i]t has been found possible to reduce the amount of flash memory which must be erased at once by physically separating the flash array during chip layout into groups (blocks) of cells which may be erased together." *Id*. at 2:31-34. By organizing the flash memory into blocks, Wells enables the memory cells of the individual blocks to be simultaneously eraseable. Specifically, Wells discloses "a thirty chip flash array with sixteen individually-erasable subblocks per chip [that] holds the same amount

of data as does a thirty megabyte electromechanical hard disk." *Id.* at 2:62-65. Ex. 1003 at Table 1, Claim 11.[a].

Thus, Wells discloses this limitation, if not expressly, then inherently.

c. Claim 11.[b]: a controller including a microprocessor that operates to:

Wells discloses a controller that includes a microprocessor. Wells discloses that the cleanup process "is a software process carried out by the microprocessor of a controller 14 utilizing instructions stored in read only memory therewith." Ex. 1004 at 9:55-57. Ex. 1003 at Table 1, Claim 11.[b].

d. Claim 11.[c]: note when a housekeeping operation not required for execution of a command received from a host has been asserted,

See discussion of claim 5.[a], above.

e. Claim 11.[d]: determine at least one parameter of activity of the host, and

See discussion of claim 5.[b], above.

f. Claim 11.[e]: if the determined at least one parameter meets at least one predefined condition, execution of the housekeeping operation is not enabled, but See discussion of claim 5.[b] above.

See discussion of claim 5.[c], above.

g. Claim 11.[f]: if the determined at least one parameter does not meet the predefined condition, the housekeeping operation is enabled for execution.

See discussion of claim 5.[d], above.

11. Claim 12: The method of claim 11, wherein the housekeeping operation includes rewriting data from one location in the memory system to another location in the memory system.

See discussion of claim 6, above.

12. Claim 13: The method of claim 11, wherein determining at least one parameter of activity of the host includes monitoring said at least one parameter during execution by the memory system of one of the commands received from the host.

See discussion of claim 8, above.

13. Claim 14: The method of claim 11, wherein the current received command is one of a group of commands that individually include data read and data write.

See discussion of claim 9, above.

G. Challenge #2: Claims 11-14 are rendered obvious by Wells in view of the knowledge of a POSITA, or in view of Intel Series 2 Flash Memory Cards Data Sheet ("Series 2").

Except for the argument for the preamble of claim 11, all of the anticipation analysis for Wells above applies here for Wells as an obviousness ground. As stated above, all of the elements of claims 1-9 and 11-14 are expressly disclosed by Wells because the preamble of claim 11 should reasonably be found to be not limiting. However, to the extent that Challenge #1 to claims 11-14 fails, and the preamble is found to be limiting, Wells in combination with the knowledge of a person of skill in the art or in view of Intel Series 2 Flash Memory Cards Data Sheet ("Series 2") (Ex. 1023) renders claim 11 and its dependents obvious. Series

2 was published on December 1996 and is therefore prior art against the '095 Patent under pre-AIA 35 U.S.C. §§ 102(a) and 102(b). Series 2 is a product manual for the Intel Series 2 PCMCIA Flash Memory Cards.

1. Claim 11.[pre]: A memory system adapted to be removably connected with a host system, comprising:

If the preamble of claim 11 is found to be limiting, then the '095 patent is rendered obvious by Wells because one of ordinary skill in the art would immediately understand that the system of Wells could be adapted to be removably connected with the host system. Ex. 1003 at Table 1, Claim 11.[pre]. Removable flash media—such as PCMCIA, CompactFlash ("CF") or Secure Digital ("SD") cards and USB drives— were well known in the art at the time. *Id*. at ¶¶ 30, 67-68; Table 1, Claim 11.[Pre]. In fact, the Wells patent is assigned to Intel, and Intel manufactured and sold flash memory cards that could be removably connected with a host system. Ex. 1023 at 5. Ex. 1003 at Table 1, Claim 11.[pre].

Moreover, the prior art considered by the examiner during prosecution includes a publication describing an innovation in the field of removable memory card systems & design, *Solid-State Mass Storage Arrives, Product Feature, Memory Card Systems & Design, Jul./Aug. 1992.* Ex. 1004 at Other Publications. This is a clear indication that the state of the art was such that a removably attached memory system was well-known, and it would have been obvious to one

of skill in the art to apply the teachings of Wells into a memory system adapted to be removably connected with a host system. Ex. 1003 at Table 1, Claim 11.[pre].

To the extent this was not common knowledge to a POSITA at the time, the Intel Series 2 Flash Memory Cards Data Sheet ("Series 2") expressly discloses a memory system adapted to be removably connected with the host system. Ex. 1023 at 5. The teachings in Wells share many similarities, and are consistent with what is disclosed in Series 2. Ex. 1003 at Table 1, Claim 11.[pre]. It would have been obvious to apply all the relevant teachings in Wells and package it into a removable card form factor as expressly disclosed by Series 2. *Id.*

H. Challenge #3: Claims 1-3 and 5-9 of the '095 are anticipated by U.S. Publication No. 2003/0046487 ("Swaminathan").

Swaminathan anticipates claims 1-9 and 11-14 of the '095 Patent. Swaminathan was filed on August 30, 2001 and published on March 6, 2003, and therefore is prior art against the '095 Patent under pre-AIA 35 U.S.C. §§ 102(a) and 102(e). Ex. 1022.

Swaminathan describes a method of refreshing the data in a flash memory. Figure 1 depicts a processor circuit that utilizes the flash memory device 100.

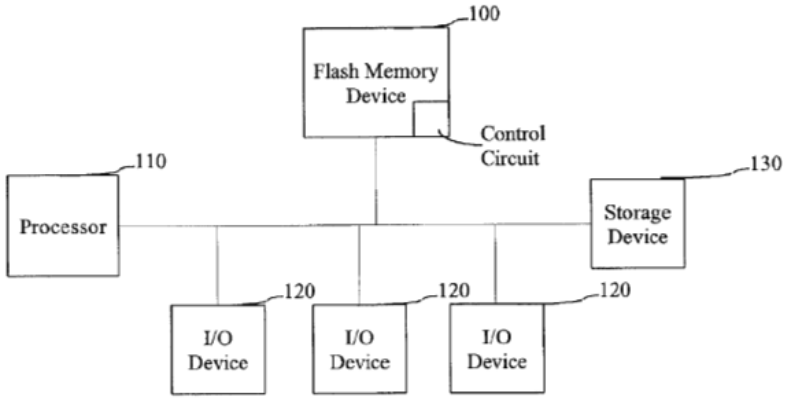


Fig. 1

Ex. 1022 at Figure 1. The flash memory device 100 has a control circuit and is connected to processor 110, I/O devices 120, and storage device 130.

The flash memory system described in Swaminathan is divided into main blocks, which are physically isolated from each other. *Id*. at ¶ [0002]. Each main block is subdivided into erase blocks, which are not physically isolated from each other, as they share a common bit line. *Id*. The high voltages used for programming disturb the data stored in the erase blocks within the same main block, causing errors in the data. *Id*. at ¶ [0003]. The data can be corrected by performing a refresh process, but as Swaminathan explains, refreshing the data "creates dead time when the flash memory device is inaccessible to the host system." *Id*. at ¶ [0018]. Swaminathan proposes two solutions that use counters to refresh the data at certain intervals to optimize the frequency at which sectors or erase blocks are

refreshed. *Id*. at ¶¶ [0008], [[0018]. Figure 2 depicts a method where a counter is maintained for each sector and the system uses this counter to determine whether to refresh a sector.

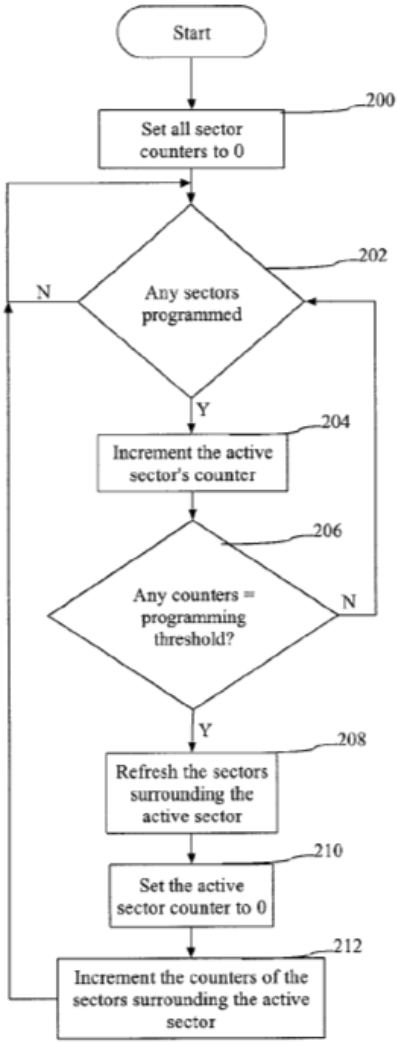


Fig. 2

Id. at Figure 2. Swaminathan explains that because most flash memories have 128 sectors per erase block, maintaining a counter for each sector can consume significant amounts of memory space. *Id*. at ¶ [0023]. Figure 3 depicts an alternative method of refresh by which a counter is maintained for each erase block, thereby "significantly reducing the space used by counters." *Id*. at ¶ [0024].

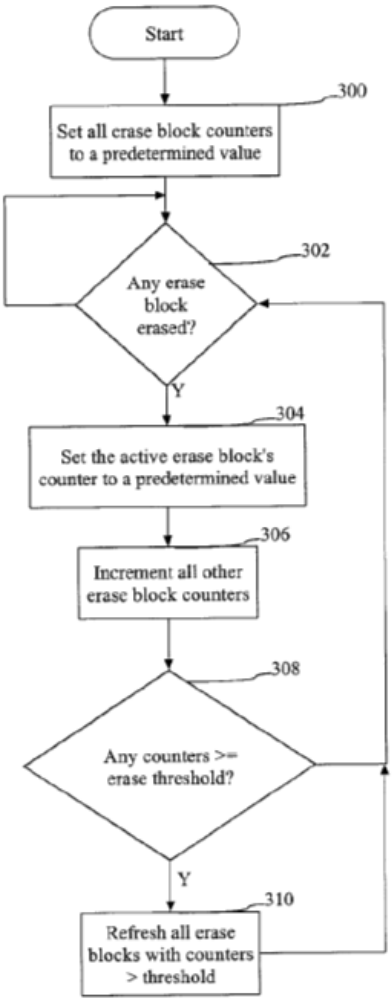


Fig. 3

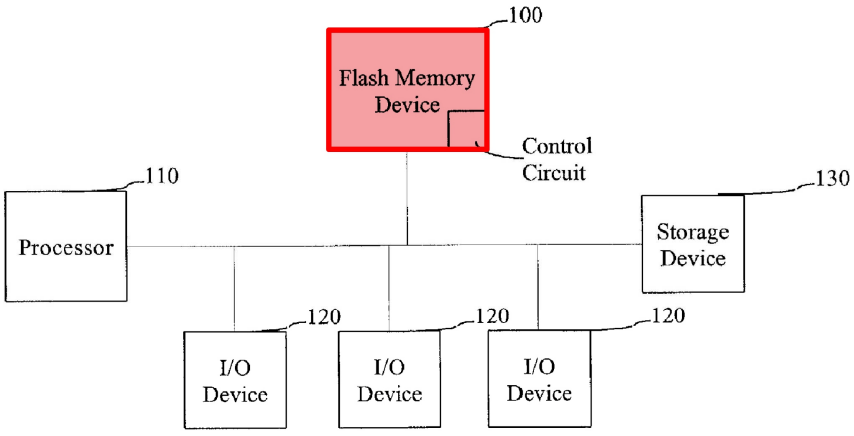
Id. at Figure 3. Both embodiments have a set predetermined threshold that will trigger a refresh operation when a counter meets or exceeds the threshold.

Importantly, Swaminathan acknowledges that "when multiple counters equal or exceed the predetermined threshold value simultaneously, flash memory device 100 may appear busy to processor 110 for an extended period of time. Since this situation is undesirable, the time spent refreshing multiple erase block can be hidden from processor 110 by allowing processor 110 to continue accessing flash memory device 100 between refresh operations." *Id*. at ¶ [0031], emphasis added.

1. Claim 1:

a. A method of operating a re-programmable non-volatile memory system, comprising:

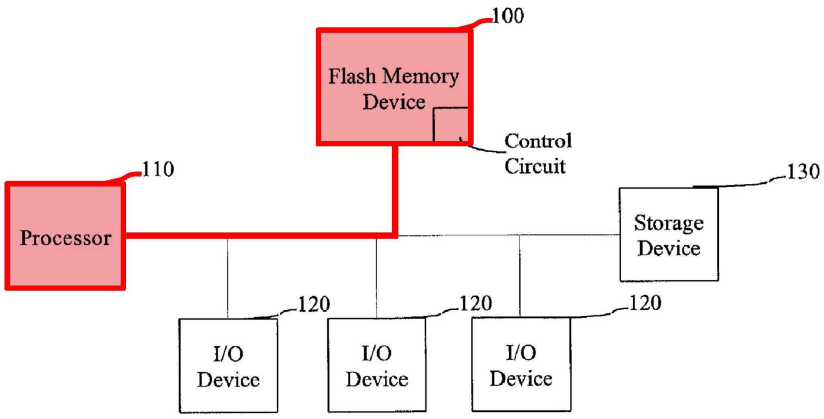
Swaminathan discloses the features recited in this preamble. Swaminathan discloses a system that uses a nonvolatile flash memory. Ex. 1022 at Abstract; ¶¶ [0002]; [0007]. This flash memory is depicted in Figure 1:



Id. at Figure 1. Ex. 1003 at Table 2, Claim 1.[pre.].

b. receiving commands from a host and executing the received commands,

Swaminathan discloses a host system that accesses the flash memory device by *e.g.* programming data to be stored into the flash memory. Ex. 1022 at ¶ ¶ [0017], [0018], [0020], [0031]. The processor 110 is part of the host system that is connected to and accesses the flash memory device and performs *e.g.* programming operations between refresh operations. *Id*. at ¶ [0031] and at Fig. 1.



These operations are initiated by the processor 110, thereby satisfying this element. Ex. 1003 at Table 2, Claim 1.[a.].

c. monitoring patterns of activity of the host, at least in connection with the received commands, and

Swaminathan discloses monitoring patterns of activity of the host by tracking the number of programming operations performed by the host using the counters for each sector. Ex. 1022 at Figure 2; ¶¶ [0019]; [0020]; [0025]; [0026];

[0027] [0028]; [0029]. Ex. 1003 at Table 2, Claim 1.[b]. Furthermore, Swaminathan also discloses this element in a second embodiment by tracking the number of erase operations performed on each erase block. Ex. 1022 at Figure 3; ¶¶ [0008]; [0025]; [0026]; [0027] [0028]; [0029]. Ex. 1003 at Table 2, Claim 1.[b].

Initially at step 200 in Figure 2, the first embodiment, the counters are set to some predetermined value, usually zero. *Id*. at Figure 2; ¶¶ [0019]. The control circuit in the flash memory 100 checks if a sector was programmed in processing segment 202. *Id*. at Figure 2; ¶ [0019]. If yes, then it moves to step 204, where the control circuit increments the counter of the programmed sector. Ex. 1003 at Table 2, Claim 1.[b].

Likewise in the second embodiment, at step 300 in Figure 3, the counters are set to zero. Ex. 1022 at Figure 3; ¶ [0026]. However, in this embodiment, if the control circuit determines in step 302 that a block was erased, the counter for that active erase block is reset to the predetermined value and in step 306, all of the counters for the other erase blocks in the same main block as the active erase block are incremented. *Id*. Ex. 1003 at Table 2, Claim 1.[b].

d. upon identifying a first pattern of host activity, a housekeeping operation is enabled to be executed, the housekeeping operation being of a type not required for execution of one of the commands received from the host, or

Swaminathan discloses that in both embodiments, there is a predetermined threshold set in the software that triggers a data refresh in the sector or the erase block. Ex. 1022 at ¶¶ [0020]; [0026].

In the embodiment of Figure 2, in step 206, if the control circuit identifies a first pattern of host activity where it determines that a sector counter equals or exceeds this predetermined threshold, then it enables the housekeeping operation to be executed and refreshes the data in the sectors surrounding the active sector in step 208. *Id*. at Figure 2, ¶ [0022].

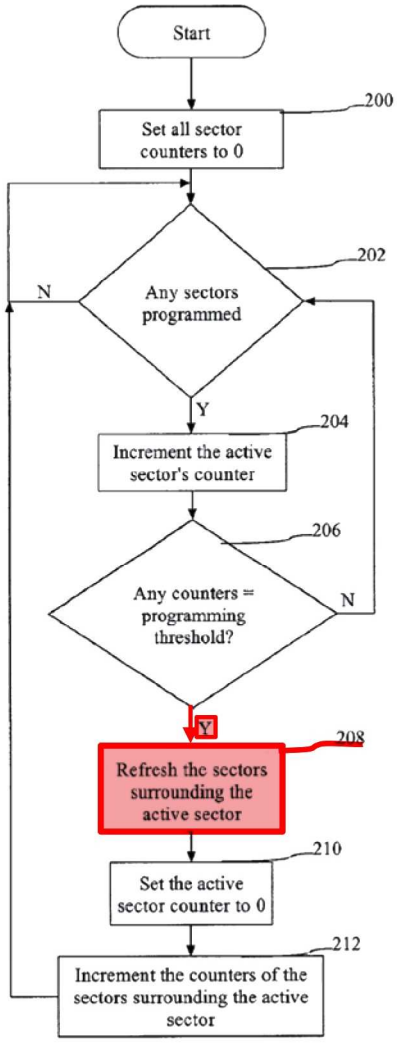


Fig. 2

Id. at Figure 2. Ex. 1003 at Table 2, Claim 1.[c].

Similarly in the embodiment of Figure 3, if the control circuit identifies a first pattern of host activity where the control circuit determines that any erase block counters equal or exceed a predetermined threshold at processing segment 308, then it enables the housekeeping operation to be executed and refreshes the

data in those erase blocks. *Id*. at Figure 3, ¶ [0026]. Ex. 1003 at Table 2, Claim 1.[c].

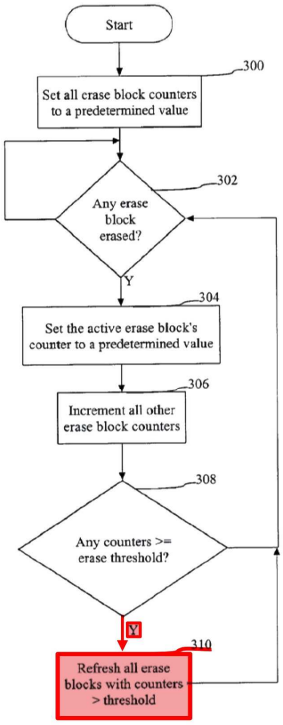


Fig. 3

e. upon identifying a second pattern of host activity different from the first pattern, execution of the housekeeping operation is not enabled.

In both embodiments, if the control circuit identifies a second pattern of host activity different from the first pattern, where the counters do not equal or exceed a predetermined threshold, the housekeeping data refresh operation is not enabled. This is depicted in the "N" arrow in step 206 in Figure 2, and step 308 in Figure 3.

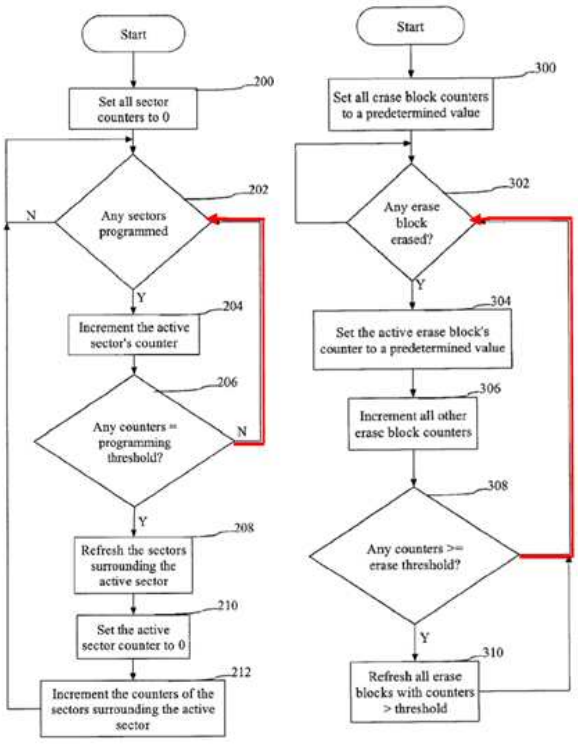


Fig. 2 **Fig. 3**

Ex. 1022 at Figures 2, 3.

2. Claim 2: The method of claim 1, additionally comprising, in response the first pattern of host activity being identified, executing at least one portion of the enabled housekeeping operation.

As shown above in Figures 2 and 3, the enabled housekeeping operation is executed in steps 208 and 310 respectively. Ex. 1022 at Figures 2, 3; ¶¶ [0022]; [0026]. Ex. 1003 at Table 2, Claim 2.

3. Claim 3: The method of claim 2, wherein executing the enabled housekeeping operation includes reading a block of data from one location of the memory system and thereafter writing the read data into another location of the memory system.

Swaminathan discloses this element when it describes the embodiment in Figure 3 where "[i]f an erase block contains user data, when that erase block is refreshed, the data is moved to a new location, that is, the data is moved and then the block is erased. By moving the data to a new location, it can be written to and read from the new location." Ex. 1022 at ¶[0030]. This disclosure indicates that the enabled housekeeping operation includes reading a block of data from one location and writing it to another location of the memory system. Ex. 1003 at Table 2, Claim 3.

4. Claims 5:

a. Claim 5[pre]: A method of operating a re-programmable non-volatile memory system, comprising:

As shown above in element 1[a] above, Swaminathan discloses a method of operating a re-programmable non-volatile memory system.

b. Claim 5.[a]: note when a housekeeping operation not required for execution of a command received from a host has been asserted,

Swaminathan discloses this element when it describes the embodiments of Figures 2 and 3 where the control circuit checks if any counters are equal to or

greater than the program or erase threshold in step 208 and step 308 respectively. Ex. 1022 at Figures 2 and 3, ¶¶ [0020], [0022], [0026]. If any counters are equal to or greater than the program or erase threshold, then a housekeeping data refresh operation is asserted in step 210 or step 310 respectively. *Id*. *See also* Ex. 1003 at Table 2, Claim 5.[a].

c. Claim 5.[b]: determine at least one parameter of activity of the host, and

Swaminathan discloses determining a parameter of activity of the host when the system determines if the processor is performing an operation. During the check described in step 308, multiple counters may equal or exceed the predetermined threshold value simultaneously. Ex. 1022 at ¶ [0031]. To avoid making the flash memory device unavailable to the processor for an extended period of time, the system allows the processor "to continue accessing flash memory device 100 between refresh operations." *Id*. Ex. 1003 at Table 2, Claim 5.[b].

d. Claim 5.[c]: if the determined at least one parameter meets at least one predefined condition, execution of the housekeeping operation is not enabled, but

If the parameter meets the predefined condition (*i.e.* host processor is performing an operation), execution of the housekeeping operation is not enabled. Ex. 1003 at Table 2, Claim 5.[c]. When there are multiple erase blocks that need to be refreshed because their counters have met or exceeded the threshold, in order to

avoid making the flash memory unavailable to the processor for an extended period of time, "[e]ach erase block that requires refreshing is refreshed after an operation initiated by processor 110". Ex. 1022 at ¶ [0031]. "As a result, instead of using 100% of flash memory 100's capacity during multiple refreshes and interfering with any other operations being performed by process 110, each operation that processor 110 performs will take a little longer to complete". *Id*. When the processor is performing an operation (parameter meets predefined condition), the housekeeping operation is not enabled and the refresh operation is not performed. Ex. 1003 at Table 2, Claim 5.[c].

e. Claim 5.[d]: if the determined at least one parameter does not meet the predefined condition, the housekeeping operation is enabled for execution.

As described above, during a refresh of multiple erase blocks, the refreshing operations will occur between operations of the processor so that the refresh operation does not tie up 100% of the memory's capacity during refreshing. Ex. 1022 at ¶[0031]. Therefore, when the host processor is not performing an operation (parameter does not meet predefined condition), the housekeeping operation is enabled for execution and the refresh operation is performed. Ex. 1003 at Table 2, Claim 5.[d].

5. Claim 6: The method of claim 5, wherein the housekeeping operation includes rewriting data from one location in the memory system to another location in the memory system.

Swaminathan discloses performing a data refresh of an erase block by rewriting data from the erase block to be refreshed to a new location. "[T]he data is moved and then the block is erased. By moving the data to a new location, it can be written to and read from the new location." Ex. 1022 at ¶ [0030]. Ex. 1003 at Table 2, Claim 6.

6. Claim 7: The method of claim 6, wherein the housekeeping operation data rewriting is performed as part of either a wear leveling or scrub housekeeping operation.

Swaminathan discloses the housekeeping operation is a data refresh operation, which the '095 patent acknowledges is also alternatively known as a scrub operation. Ex. 1001 at 4:10-15. Ex. 1003 at Table 2, Claim 7.

7. Claim 8: The method of claim 5, wherein determining at least one parameter of activity of the host includes monitoring said at least one parameter during execution by the memory system of one of the commands received from the host.

Swaminathan discloses monitoring a parameter of activity of the host where the parameter is whether the host processor is performing an operation, such as a write operation. Ex. 1022 at ¶ [0031]. By default, this monitoring must occur during execution by the memory system of a command received from the host. For example, the activity of the processor will be monitored while the processor is

performing write operations, so that the memory can perform its operations in between. Therefore, Swaminathan discloses the limitations of claims 8 and 13. Ex. 1003 at Table 2, Claim 8.

8. Claim 9: The method of claim 5, wherein the current received command is one of a group of commands that individually include data read and data write.

Swaminathan discloses performing operations initiated by processor. Ex. 1022 at ¶¶ [0030-31]. This claim is satisfied by the disclosure that these operations include "for example, write operations." *Id*. It would also be clear to one of skill in the art reading the disclosure of Swaminathan that the host processor would inherently also issue read commands to the memory in order to retrieve the data stored by the write commands. Ex. 1003 at Table 2, Claim 4.[b] and 9.

I. Challenge #4: Claims 4, 11-14 are rendered obvious by Swaminathan in view of the knowledge of a POSITA or in view of Series 2.

As shown below, one of skill in the art would read Swaminathan and understand that the claims of the '095 are disclosed by Swaminathan. Therefore, Swaminathan in combination with the knowledge of one of skill in the art would render claims 4 and 11-14 of the '095 patent obvious.

1. Claim 4:

a. The method of claim 1, wherein receiving commands from a host and executing the received commands includes receiving and executing

As shown in element 1(b) in Challenge #3, Swaminathan discloses receiving commands from a host and executing the received commands.

b. (1) a write command to write data received from the host with the command into logical addresses of the memory specified by the write command,

Swaminathan discloses a write command to write data received from the host, describing a refresh operation that happens "after an operation initiated by processor 110, such as for example, write operations." Ex. 1022 at ¶ [0031]. As detailed above in Claim 3 in Challenge #3, Swaminathan also discloses that during the housekeeping data refresh operation, data is read from one location and written to another one. One of skill in the art, well versed in flash technology, would read this disclosure and understand that such an operation would necessitate the usage of logical to physical address translation in order to track the migration of the data to a new physical address. Ex. 1003 at ¶¶ 24-29; *Id.* at Table 2, Claim 4.[a].The data refresh operation is opaque to the processor, and one of skill in the art would understand that the processor would specify a logical address to access and then the memory would access the physical address corresponding to that logical address per a logical/physical address translation table. Ex. 1003 at ¶¶ 27-29; *Id.* at Table 2, Claim 4.[a].

To the extent Longitude were to argue that this logical to physical address translation was not already well known to a POSITA, Series 2 expressly discloses the use of a flash translation layer that would allow a "user to interact with the flash memory card in precisely the same way as a magnetic disk." Ex. 1023 at 5. It is known that conventional host computing systems interact with magnetic disks using logical addresses. It would have been obvious to a POSITA to modify the teachings in Swaminathan and implement such a flash translation layer in order to allow a user to interact with the flash memory system in the same way as a conventional magnetic disk, thereby allowing the Swaminathan to be used with conventional host computing systems that issue read commands into logical addresses. Ex. 1003 at Table 2, Claim 4.[a]. *Id.* at ¶¶ 24-29.

c. (2) or a read command to read data from logical addresses of the memory specified by the read command and send the read data to the host.

Similarly to the write command above, one of skill in the art would understand that the operations initiated by the processor 110 would include a read command specifying a logical address. Ex. 1003 at Table 2, Claim 4.[b]. *Id.* at ¶¶ 26-29. The purpose of any memory system is for data storage. When a host writes data to a memory as the processor 110 does, it is obvious to a lay user, much less to one of skill in the art, that the intended purpose is for the processor to issue read commands to read back the data that was previously written. *Id*.

To the extent Longitude were to argue that in issuing reading from a logical address, this logical to physical address translation was not already well known to a POSITA, Series 2 expressly discloses the use of a flash translation layer that would allow a "user to interact with the flash memory card in precisely the same way as a magnetic disk." Ex. 1023 at 5. It is known that conventional host computing systems interact with magnetic disks using logical addresses. It would have been obvious to a POSITA to modify the teachings in Swaminathan and implement such a flash translation layer in order to allow a user to interact with the flash memory system in the same way as a conventional magnetic disk, thereby allowing Swaminathan to be used with conventional host computing systems that issue write commands into logical addresses. Ex. 1003 at Table 2, Claim 4.[b]. *Id.* at ¶¶ 24-29.

2. Claim 11:

a. Claim 11.[pre]: A memory system adapted to be removably connected with a host system, comprising:

As explained above in the discussion of the claim 11 preamble in Challenge #1, under the broadest reasonable construction standard, this preamble should not be limiting. However, to the extent that the Board construes this preamble to be limiting, as with the disclosure in Wells, one of skill in the art reading the disclosure of Swaminathan would have found it obvious that the disclosed system could be adapted to be removably connected with the host system. Ex. 1003 at

Table 2, Claim 11.[pre]. The filing date of Swaminathan is eight years after that of Wells, when removable flash memory cards were already known in the art, as discussed in claim 11 preamble in Challenge #2. *Id*.

To the extent that it was not common knowledge to a POSITA to generally take the express teachings related to flash memory management operations, and apply them into a removable flash card form factor, Series 2 expressly discloses a memory system adapted to be removably connected with the host system. Ex. 1023 at 5. Ex. 1003 at Table 2, Claim 11.[pre]. *Id*. at ¶¶ 30, 67-68. It would have been obvious to apply all the relevant teachings in Swaminathan regarding data refresh into a removable card form factor as expressly disclosed by Series 2, since the read/write disturb problem addressed by Swaminathan also affects flash memory arrays that are in removable card form. Ex. 1003 at Table 2, Claim 11.[pre]. *Id.* at ¶¶ 47-48.

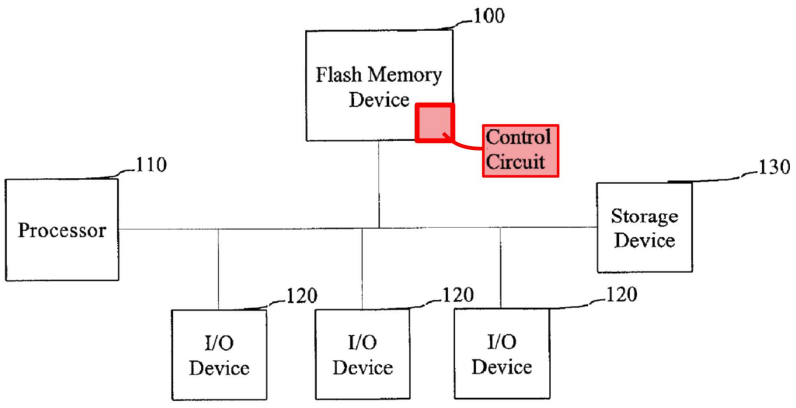
b. Claim 11.[a]: an array of re-programmable non-volatile memory cells organized into blocks of memory cells wherein the memory cells of the individual blocks are simultaneously erasable,

Swaminathan discloses a flash memory architecture where memory cells are divided into a plurality of main blocks, which are further subdivided into erase blocks. Ex. 1022 at ¶[0002]. Each "erase block may contain, for example, 128 sectors each comprising 512 bytes." *Id.* A counter is maintained for each erase

block. *Id*. ¶ [0008]. When an erase block is erased, the counter for that erase block is set to a predetermined value." *Id.* This disclosure indicates that the memory cells of each individual erase block are simultaneously eraseable. Furthermore, claims 17 and 18 disclose erasing a memory block. *Id*. at Claims 17, 18. Ex. 1003 at Table 2, Claim 11.[a].

c. Claim 11.[b]: a controller including a microprocessor that operates to:

Swaminathan discloses a control circuit in flash memory 100 that performs the algorithms disclosed in the embodiments of Figures 2 and 3. It was well known to a POSITA that this control circuit can be implemented using a microprocessor with suitable programming. Indeed, a POSITA would be particularly motivated to implement the controller using a microprocessor, because it would have been more advantageous to allow more flexibility for upgrading with improved algorithms to achieve better performance. Ex. 1003 at Table 2, Claim 11.[b]. *See also* Ex. 1022 at Fig. 1.



d. Claim 11.[c]: note when a housekeeping operation not required for execution of a command received from a host has been asserted,

See discussion of claim 5.[a] in Challenge #3 above.

e. Claim 11.[d]: determine at least one parameter of activity of the host, and

See discussion of claim 5.[b] in Challenge #3 above.

f. Claim 11.[e]: if the determined at least one parameter meets at least one predefined condition, execution of the housekeeping operation is not enabled, but

See discussion of claim 5.[c] in Challenge #3 above.

g. Claim 11.[f]: if the determined at least one parameter does not meet the predefined condition, the housekeeping operation is enabled for execution.

See discussion of claim 5.[d] in Challenge #3 above.

3. Claim 12: The method of claim 11, wherein the housekeeping operation includes rewriting data from one location in the memory system to another location in the memory system.

See discussion of claim 6 in Challenge #3 above.

4. Claim 13: The method of claim 11, wherein determining at least one parameter of activity of the host includes monitoring said at least one parameter during execution by the memory system of one of the commands received from the host.

See discussion of claim 8 in Challenge #3 above.

5. Claim 14: The method of claim 11, wherein the current received command is one of a group of commands that individually include data read and data write.

See discussion of claim 9 in Challenge #3 above.

V. CONCLUSION

Claims 1-9, and 11-14 of the '095 patent are anticipated or rendered obvious by prior art not yet considered by the Patent Office. There is a reasonable likelihood that Petitioner will prevail as to each of the claims. Petitioner respectfully requests that the Patent Office initiate an *inter partes* review of claims 1-9 and 11-14, that it find those claims invalid in light of the prior art, and that it cancel those claims.

Respectfully submitted,

Dated: September 14, 2015

/s/ Brent Yamashita

Brent Yamashita
Registration No. 53,808

DLA PIPER LLP (US)
2000 University Avenue
East Palo Alto, CA 94303-2214
Telephone: 650.833.2348
Facsimile: 650.833.2001

CERTIFICATE OF SERVICE

The undersigned certifies service pursuant to 37 C.F.R. 37 C.F.R. §§ 42.6(e) and 42.105(b) on September 14, 2015 by UPS Overnight Delivery of a copy of this Petition for *Inter Partes* Review and supporting material at the following correspondence address of record for the '095 Patent:

DAVIS WRIGHT TREMAINE LLP - SANDISK CORPORATION
IP Docketing Dept.
Davis Wright Tremaine LLP
1201 Third Avenue, Suite 2200
Seattle, WA 98101

Dated: September 14, 2015

/s/ Brent K. Yamashita
Brent K. Yamashita
Registration No. 53,808

DLA PIPER LLP (US)
2000 University Avenue
East Palo Alto, CA 94303-2215
Telephone: 650.833.2348
Facsimile: 650.687.1206