PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

EX PARTE Idan Alrod et al.

Application for Patent

Filed November 23, 2008

Application No. 12/276,344

Examiner: Ramon A. Mercado

FOR:

METHODS FOR TAG-GROUPING OF BLOCKS IN STORAGE DEVICES

APPEAL BRIEF

CERTIFICATE OF EFILING

I hereby certify that this correspondence is being transmitted to the United States Patent and Trademark Office via electronic filing with EFS-Web on May 9, 2012.

Signed: _____ /Jose M. Nunez /_____
Jose M. Nunez

MARTINE PENILLA GROUP, LLP
Attorneys for Appellants

TABLE OF CONTENTS

Page No

I. REAL PARTY IN INTEREST 3

II. RELATED APPEALS AND INTERFERENCES 3

III. SUMMARY OF CLAIMED SUBJECT MATTER 3

IV. ARGUMENT 11

A. Rejection of claims 1-30 under 35 U.S.C. 103(a) over Sinclair (US Publication No. 2005/0144357) and Gorobets et al. (U.S. Publication No. 2006/0039196) 11

1. Claims 1, 7, 12, 19, and 23-30 11
2. Claims 4 and 16 18
3. Claims 5, 10, and 17 19
4. Claims 2, 8, 13 and 20 20
5. Claims 3, 9, 14, and 21 20
6. Claims 6, 11, and 18 21
7. Claims 15 and 22 21

B. Conclusion 22

V. CLAIMS APPENDIX 23

I. REAL PARTY IN INTEREST

The real party in interest is SanDisk Corporation, the assignee of the present application.

II. RELATED APPEALS AND INTERFERENCES

The Appellants are not aware of any related appeals or interferences.

III. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1 defines a method for storing data **{page 10, line 17; Figures 2-3}**, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of blocks **{page 10, line 18; Figure 2, 18; page 18, lines 12-13}**, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block **{page 10, lines 18-20; Figure 2}**, said block metadata value being dependent upon a writing time of said each block **{page 8, lines 21 – 25}**;

(b) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks **{page 10, lines 20-22; Figure 3, 32; page 27, lines 5-6}**;

(c) associating a group metadata value with each block group **{page 10, lines 22-23; Figure 3, 34; page 27, line 6}**; and

(d) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value **{page 10, lines 23-25; page 24, lines 10-14}**, and each block

utilizes the group metadata value instead of the block metadata value **{page 6, lines 14-17; page 8, line 25 – page 9, line 4}**.

Claim 7 recites a method for storing data **{page 11, line 16; Figures 2-3}**, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of blocks **{page 10, line 18; Figure 2, 18; page 18, lines 12-13}**, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block **{page 10, lines 18-20; Figure 2}**, said block metadata value being dependent upon a writing time of said each block;

(b) grouping said plurality of blocks into block groups, wherein at least one said block group contains at least two blocks **{page 10, lines 20-22; Figure 3, 32; page 27, lines 5-6}**;

(c) associating a sequential tag with each block group **{page 11, lines 7-9 and 21-24; Figure 3, 34}**, wherein blocks in a first block group having a previous sequential tag are written before blocks in a second block group having a subsequent sequential tag **{page 11, lines 21-23}** ; and

(d) associating a group metadata value with each respective sequential tag, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value **{page 11, lines 23-24; page 24, lines 10-14}**, and each block utilizes the group metadata value instead of the block metadata value **{page 6, lines 14-17; page 8, line 25 – page 9, line 4}**.

Claim 12 recites method for storing data **{page 13, line 15; Figures 2-3}**, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 13, lines 16-19; page 25, line 17}**;

(b) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 13, lines 19-21}**;

(c) associating a group metadata value with each word-line group **{page 13, lines 21-24}**; and

(d) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value **{page 13, lines 24-25}**.

Claim 19 recites a method for storing data **{page 13, line 15; Figures 2-3}**, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 13, lines 16-19; page 25, line 17}**;

(b) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 13, lines 19-21}**;

(c) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag **{page 11, lines 7-9 and 21-24; Figure 3, 34}**; and

(d) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value **{page 11, lines 23-24; page 24, lines 10-14}**, and each word-line utilizes the group metadata value instead of the line metadata value **{page 6, lines 14-17; page 8, line 25 – page 9, line 4}**.

Claim 23 recites a memory device comprising **{page 14, line 10}**:

(a) a controller for storing data in a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block **{page 14, lines 10-13}**, said controller configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks **{page 14, lines 13-15}**;

(ii) associating a group metadata value with each block group **{page 14, line 15}**; and

(iii) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value **{page 14, lines 16-18}**.

Claim 24 recites a memory device comprising **{page 14, line 20}**:

(a) a controller for storing data in a memory that includes a plurality of blocks **{page 10, line 18; Figure 2, 18; page 18, lines 12-13}**, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block

{page 10, lines 18-20; Figure 2}, said block metadata value being dependent upon a writing time of said each block **{page 8, lines 21 – 25}**, said controller configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two locks **{page 10, lines 20-22; Figure 3, 32; page 27, lines 5-6}**;

(ii) associating a group metadata value with each block group **{page 10, lines 22-23; Figure 3, 34; page 27, line 6}**; and

(iii) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value **{page 10, lines 23-25; page 24, lines 10-14}**, and each block utilizes the group metadata value instead of the block metadata value **{page 6, lines 14-17; page 8, line 25 – page 9, line 4}**.

Claim 25 recites a memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of word-lines, wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 13, lines 16-19; page 25, line 17}**, said controller configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 13, lines 19-21}**;

(ii) associating a group metadata value with each word-line group **{page 13, lines 21-24}**; and

(iii) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value **{page 13, lines 24-25}**.

Claim 26 recites a memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of word-lines, wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 13, lines 16-19; page 25, line 17}**, said controller configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 13, lines 19-21}**;

(ii) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag **{page 11, lines 7-9 and 21-24; Figure 3, 34}**; and

(iii) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value **{page 11, lines 23-24; page 24, lines 10-14}**, and each word-line utilizes the group metadata value instead of the line value **{page 6, lines 14-17; page 8, line 25 – page 9, line 4}**.

Claim 27 recites a computer-readable non-transitory storage medium having computer-readable code embodied therein **{page 16, line 2}**, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block **{page 16, lines 3-6}**, said driver code configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks **{page 16, lines 6-8}**;

(ii) associating a group metadata value with each block group **{page 16, line 8}**; and

(iii) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value **{page 16, lines 9-11 to}**.

Claim 28 recites a computer-readable non-transitory storage medium having computer-readable code embodied therein **{page 16, line 13}**, the computer-readable code comprising:

(a) driver code for a memory device having a that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block **{page 16, lines 13-17}**, said driver code configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks **{page 16, lines 17-19}**;

(ii) associating a sequential tag with each block group, wherein blocks in a first block group having a previous sequential tag are written before blocks in a second block group having a subsequent sequential tag **{page 16, lines 19-21}**; and

(iii) associating a group metadata value with each respective sequential tag, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value **{page 16, lines 21-22}**.

Claim 29 recites a computer-readable non-transitory storage medium having computer-readable code embodied therein **{page 16, lines 24}**, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 16, lines 25-page 17, line 3}**, said driver code configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 17, lines 3-5}**;

(ii) associating a group metadata value with each word-line group **{page 17, lines 5-6}**; and

(iii) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value **{page 17, lines 6-9}**.

Claim 30 recites a computer-readable non-transitory storage medium having computer-readable code embodied therein **{page 17, lines 11-12}**, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line **{page 17, lines 12-15}**, said driver code configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines **{page 17, lines 15-17}**;

(ii) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag **{page 17, lines 17-20}**; and

(iii) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value, and each word-line utilizes the group metadata value instead of the line metadata value **{page 17, lines 20-21}**.

It should be appreciated that the above description represents only a summary of the present embodiments. A more in-depth discussion is provided in the Detailed Description section of the application.

IV. ARGUMENT

Appellants present the following arguments with respect to the rejected claims:

A. Rejection of claims 1-30 under 35 U.S.C. 103(a) over Sinclair (US Publication No. 2005/0144357) and Gorobets et al. (U.S. Publication No. 2006/0039196)

1. Claims 1, 7, 12, 19, and 23-30

i. The prior art does not teach that said block metadata value [is] dependent upon a writing time of said each block

Claim 1 recites that "said block metadata value [is] dependent upon a writing time of said each block."

Appellants note that the Examiner has not considered the claim as a whole. Claim 1 recites a memory with a plurality of **blocks**, but in the Office's rejection of claim 1, the claimed **blocks** appear to be taught by "adaptive logical blocks" (see page 4, third line from the bottom of the Office Action dated November 28, 2011), "adaptive logical

metablocks" (page 4, third line from the bottom), "sector data" (page 5, line 1), "erase blocks" (page 5, line 4), and "memory cells" (page 2, last paragraph). The Examiner has failed to analyze Appellants' claims as a whole.

In addition, the Examiner is inconsistent regarding how the prior art teaches the claimed elements. For example, the Examiner appears to indicate that an "adaptive logical block" teaches both a block and a block group. The Examiner has asserted that "receiving data to be stored in a memory that includes a plurality of blocks/word-lines [adaptive logical blocks/adaptive logical megablocks, paragraph 0016; and word lines, paragraph 004" (page 4, last paragraph, emphasis added). Therefore, the Examiner seems to indicate that an "adaptive logical block" teaches the claimed "block."

In addition, the Examiner has asserted that "the blocks/word-lines that form these adaptive logical blocks and made a blocks have overhead data" (page 5, last paragraph, emphasis added). Therefore, it appears that blocks form adaptive logical blocks, which means that the "adaptive logical blocks" teach the claimed "block group."

Consequently, the Examiner believes that "adaptive logical blocks" teaches both a "block" and a "block group". The Examiner's rejection lacks rational underpinnings because the rejection lacks consistency.

Furthermore, the Examiner has asserted that Sinclair teaches that "the blocks/word-lines to form these adaptive logical blocks and metablocks have overhead data, including parameters such as the voltage levels used for programming and/or erasing these blocks [paragraph 0058]" (page 5, last paragraph). This assertion by the Examiner further creates confusion because it is unclear whether a block group is taught by an adaptive logical block or by a metablock, or maybe by both.

The Examiner has asserted that "receiving data to be stored in a memory that includes a plurality of blocks" is taught by Sinclair because Sinclair teaches "adaptive logical blocks/adaptive logical megablocks, paragraph 0016" (page 4, last paragraph). Therefore, and "adaptive logical block" appears to teach the claimed "block." Further, the Examiner has asserted that "said block metadata value being dependent upon a writing time of said each block" is taught by Sinclair because Sinclair teaches "indications of voltage levels to be used for programming erasing the erase block, paragraph 0058" (page 5, first paragraph, emphasis added). If Sinclair would teach this feature, Sinclair would have to teach a metadata value related to adaptive logical blocks. However, an erase block is different from an adaptive logical block, and the Examiner's rejection lacks rational underpinnings because it refers to different elements regarding the claimed blocks.

ii. The prior art does not teach associating a group metadata value with each block group

Claim 1 recites "associating a group metadata value with each block group." The Examiner has admitted that Sinclair does not explicitly teach this feature (page 5, second paragraph). As discussed above, the Examiner has asserted that "said block metadata value being dependent upon a writing time of said each block" is taught by Sinclair because Sinclair teaches "indications of voltage levels to be used for programming erasing the erase block, paragraph 0058." Sinclair teaches the following in paragraph 0058:

> "[0058] The parameters 55 may also include an indication of the bit values assigned to each of the storage states of the memory cells, referred to as their "rotation". This also has a beneficial effect in wear leveling. One or more flags may also be included in the parameters 55 that indicate status or states. Indications of voltage levels to be used for programming and/or erasing the erase **block** can also be stored within the parameters 55, these voltages

being updated as the number of cycles experienced by the erase block and other factors change. Other examples of the parameters 55 include an identification of any defective cells within the erase block, the logical address of the data that is mapped into this physical block and the address of any substitute erase block in case the primary erase block is defective. The particular combination of parameters 55 that are used in any memory system will vary in accordance with the design. Also, some or all of the overhead data can be stored in erase **blocks** dedicated to such a function, rather than in the erase block containing the user data or to which the overhead data pertains.' (Paragraph [0058], emphasis added); and

"[0011] To further efficiently manage the memory, erase blocks may be linked together to form virtual blocks or metablocks" (paragraph [0011], emphasis added);

Sinclair teaches that erase blocks may be grouped into a metablock. Therefore, it may appear that the Examiner means that a metablock teaches a block group, and that an erase block teaches the claimed block. In addition, paragraph [0058] cited by the Examiner refers to Figure 3. Two paragraphs before paragraph [0058], Sinclair teaches the following:

"[0056] The size of the individual memory cell erase blocks of FIG. 2 can vary but one commercially practiced form includes a single sector of data in an individual erase block. The contents of such a data sector are illustrated in FIG. 3. User data 51 are typically 512 bytes. In addition to the user data 51 are overhead data that includes an ECC 53 calculated from the user data, parameters 55 relating to the sector data and/or the erase block

> in which the sector is programmed and an ECC 57 calculated from the parameters 55 and any other overhead data that might be included. Alternatively, a single ECC may be calculated from both user data 51 and parameters 55" (paragraph [0056], emphasis added).

Therefore, Figure 3 refers to the contents of data associated with an erase block. It is noted that the overhead data does not refer to a metablock, therefore, the overhead data is not kept in a metablock. In addition, Sinclair also teaches the following regarding parameters 55:

> "[0057] The parameters 55 may include a quantity related to the number of program/erase cycles experienced by the erase block …" (Paragraph [0057], emphasis added).

The parameters 55 of Sinclair refer to erase block parameters, and not parameters associated with the metablocks. The assertion by the Examiner that "metablocks have overhead data" is not supported by Sinclair, because Sinclair teaches that some of the overhead data for **a block** might be stored in another block. However, there is no teaching in Sinclair regarding overhead data for metablocks, much less overhead data corresponding to block parameters stored in the metablocks. Therefore, Sinclair does not teach that "metablocks have overhead data," and Sinclair does not teach a group metadata value, as claimed by Appellants.

In the Response to Arguments of the Office Action dated November 28, 2011, the Examiner has asserted the following:

> "Sinclair discloses a memory array wherein memory cells are grouped into erase blocks [Figure 2], and where the erase memory blocks include both sector data and overhead data (paragraph 0056]. Sinclair further discloses that this overhead data

> includes ECC calculated from the data, parameters relating the erase block in which the sector is programmed, ECC calculated from the parameters and any other overhead data that might be included [Figure 3, paragraph 0056]. This overhead information is clearly metadata pertaining to the erase blocks and user data" (page 2, last paragraph, emphasis added).

Here, it appears that the Examiner means that "memory cells" teach the claimed "blocks," and that an "erase block" teaches a "block group." The Examiner's rejection is confusing because there is no mention of "memory cells" in the 103 rejection, and they are only mentioned in the Response to Arguments where the Examiner mentions for the first time "memory cells." The Examiner's rejection lacks rational underpinnings and is therefore improper.

Furthermore, it appears that the "ECC" teaches the claimed "block metadata value," according to the Examiner. However, Sinclair is silent with reference to calculating ECC for the individual memory cells. But claim 1 specifies that "each block from the plurality of blocks has at least one block metadata value." The ECC overhead data can not refer to the block metadata value, because Sinclair does not calculate the ECC for each memory cell, but rather for the complete erase block. This means that Sinclair can not teach "associating a group metadata value with each block group," as claimed by Appellants.

Further yet, the Examiner has asserted the following:

> "Gorobets discloses that the voltage levels, referred to as threshold levels (V_t), shift as a result of charge disturbing programming, reading or erasing operations performed in the

> neighboring or other related memory cells, pages or blocks [paragraph 0013]. Gorobets further teaches a scrub operation to reduce the corruption of data that happens over time due to V_t shifting [paragraph 0015]. Finally Gorobets discloses that Flash memories place time constraints on reading and other operations [paragraph 0024]." (bottom of page 5-top of page 6).

Appellants respectfully assert that the teachings of Gorobets do not cure the deficiencies in the teachings of Sinclair. Even assuming that the statement by the Examiner is correct, the excerpts referenced by the Examiner merely recite properties of memory devices. Gorobets is silent with reference to a group metadata value, much less, metadata value being dependent upon a writing time of said each block, because the threshold levels (V_t) do not depend upon the writing time of the block, and much less associating a group metadata value with each block group, as claimed by Appellants. Since neither reference teaches a group metadata value, or "associating a group metadata value with each block group," the combination cannot teach this feature either. For all these reasons, the prior art does not teach the aforementioned feature.

iii. The prior art does not teach that each block from the plurality of blocks does not store a dedicated copy of said block metadata value and each block utilizes the group metadata value instead of the block metadata value

Present claim 1 recites that "each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value." The Examiner has admitted that Sinclair does not teach this feature. Further, the Examiner has not provided a rejection for this feature as the Examiner has not stated how Gorobets may teach that each block utilizes the group metadata value instead of the block metadata value. As discussed above, ECC does not

teach the claimed block metadata value, because the ECC is calculated for the erase block, and the same ECC can not be used for the memory cells that are part of the erase block, as it would not reflect the correct ECC for the memory cell.

In the Response to Arguments of the Office Action dated November 28, 2011, the Examiner has asserted the following:

"Sinclair discloses that the erase block overhead data (metadata) can be stored in erase blocks dedicated to such actions, rather than in the erase block containing the user data or to which the overhead data (metadata pertains [paragraph 0058]. Thus the metadata is stored in a dedicated block and not on each block from the plurality of blocks" (page 4, first paragraph).

Appellants respectfully disagree. It appears that here, the Examiner believes that a block is taught by a memory cell and that the block group is taught by an erase block. However, the erase block overhead data described in paragraph [0058] refers to data associated with the erase block. Sinclair is silent with reference to overhead data for the memory cells (which teach a block). Claim 1 recites that "each block from the plurality of blocks has at least one block metadata value associated with said each block." Although Sinclair teaches values associated with the erase block, Sinclair does not teach values associated with the memory cells, only with the erase blocks. For all these reasons, the prior art does not teach the aforementioned feature.

2. Claims 4 and 16

i. The prior art does not teach that said step of grouping is based on a tag that is common to all blocks in a block group, wherein said tag is associated with an approximate relative time for writing said blocks of said block group

Claims 4 and 16 recite that "said step of grouping is based on a tag that is common to all blocks in a block group, wherein said tag is associated with an approximate relative time for writing said blocks of said block group. The Examiner has asserted that "V_t shifts as a result of charge disturbing programming, Gorobets paragraph 0013, therefore V_t shift is associate *[sic]* to the time of writing/programming" (page 7, third paragraph). Appellants respectfully disagree.

Gorobets teaches the following:

> "[0015] A comprehensive scrub operation is included in a flash memory system to compensate for disturbs of storage levels in one group of memory cells that are caused by programming, reading or erasing operations performed on another group of memory cells on the same integrated circuit chip." (Paragraph [0015], emphasis added).

Therefore, the V_t shift is associated with operations performed on another group of memory cells, and not on the "time for writing said blocks" (taught by memory cells according to the Examiner)," as claimed by Appellants.

3. Claims 5, 10, and 17

i. The prior art does not teach that a time since power-up of said memory is divided into time intervals, and wherein blocks written during different time intervals are associated with different tags

Claims 5, 10, and 17 recite that "a time since power-up of said memory is divided into time intervals, and wherein blocks written during different time intervals are associated with different tags." The Examiner has asserted that this feature it taught by Gorobets in a "timestamp" of paragraph [0026].

Appellants respectfully disagree. As the person skilled in the art would readily appreciate, a timestamp refers to the date and time of an event. For example, online-dictionary Merriam-Webster defines "time stamp" as "a device for recording the date and time of day that letters or papers are received or sent out"(emphasis added). On the other hand, a time interval defines a period of time, with a beginning and an end, which means a timestamp at the beginning and a timestamp at the end. Therefore, the timestamp does not teach a time interval, and the prior art does not teach the aforementioned feature.

4. Claims 2, 8, 13 and 20

i. The prior art does not teach the features of claims 2, 8, 13, and 20

Claims 2, 8, 13, and 20 recite that "said group metadata value is value selected from a group consisting of: a data age, an optimal-reading threshold-voltage, an optimal-verify threshold-voltage, a number of soft bits read, or data related to error correction." The Board is respectfully requested to consider the patentability of the features found in the recited dependent claims. These features, in combination with the features found in the independent claim from which it depends directly or indirectly, are submitted to be patentable over the prior art of record and the rejections made by the Examiner. Thus, a separate determination of patentability is requested from the Board for this dependent claim.

5. Claims 3, 9, 14, and 21

i. The prior art does not teach the features of claims 3, 9, 14, and 21

Claims 3, 9, 14, and 21 recite that "said group metadata value is stored in a block-group table." The Board is respectfully requested to consider the patentability of the

features found in the recited dependent claims. These features, in combination with the features found in the independent claims from which they depend directly or indirectly, are submitted to be patentable over the prior art of record and the rejections made by the Examiner. Thus, a separate determination of patentability is requested from the Board for these dependent claims.

6. Claims 6, 11, and 18

i. The prior art does not teach the features of claims 6, 11, and 18

Claims 6, 11, and 18 recite that "blocks written during two different power-up sessions of said memory are associated with different tags." The Board is respectfully requested to consider the patentability of the features found in the recited dependent claims. These features, in combination with the features found in the independent claims from which they depend directly or indirectly, are submitted to be patentable over the prior art of record and the rejections made by the Examiner. Thus, a separate determination of patentability is requested from the Board for these dependent claims.

7. Claims 15 and 22

i. The prior art does not teach the features of claims 15 and 22

Claims 15 and 22 recite that "each said word-line is contained in a block, and wherein at least one block contains two word-lines belonging to different word-line groups." The Board is respectfully requested to consider the patentability of the features found in the recited dependent claims. These features, in combination with the features found in the independent claims from which they depend directly or indirectly, are submitted to be patentable over the prior art of record and the rejections made by the

Examiner. Thus, a separate determination of patentability is requested from the Board for these dependent claims.

B. Conclusion

In view of the foregoing reasons, the Appellants submit that each of the claims 1-30 are patentable. Therefore, the Appellants respectfully request that the Board of Patent Appeals and Interferences reverse the Examiner's rejections of the claims on appeal.

Respectfully submitted,
MARTINE PENILLA GROUP, LLP

/Jose M. Nunez/

Jose M. Nunez, Esq.
Reg. No. 59,979

May 9, 2012

710 Lakeway Drive, Suite 200
Sunnyvale, CA 94085
Telephone: (408) 749-6900
Facsimile: (408) 749-6901
Customer Number 77215

V. CLAIMS APPENDIX

1. A method for storing data, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block;

(b) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks;

(c) associating a group metadata value with each block group; and

(d) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

2. The method of claim 1, wherein said group metadata value is value selected from a group consisting of: a data age, an optimal-reading threshold-voltage, an optimal-verify threshold-voltage, a number of soft bits read, or data related to error correction.

3. The method of claim 1, wherein said group metadata value is stored in a block-group table.

4. The method of claim 1, wherein said step of grouping is based on a tag that is common to all blocks in a block group, wherein said tag is associated with an approximate relative time for writing said blocks of said block group.

5. The method of claim 4, wherein a time since power-up of said memory is divided into time intervals, and wherein blocks written during different time intervals are associated with different tags.

6. The method of claim 5, wherein blocks written during two different power-up sessions of said memory are associated with different tags.

7. A method for storing data, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block;

(b) grouping said plurality of blocks into block groups, wherein at least one said block group contains at least two blocks;

(c) associating a sequential tag with each block group, wherein blocks in a first block group having a previous sequential tag are written before blocks in a second block group having a subsequent sequential tag; and

(d) associating a group metadata value with each respective sequential tag, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

8. The method of claim 7, wherein said group metadata value is selected from a group consisting of: a data age, an optimal-reading threshold-voltage, an optimal-verify threshold-voltage, a number of soft bits read, or data related to error correction.

9. The method of claim 7, wherein said group metadata value is stored in a block-group table.

10. The method of claim 7, wherein a time since power-up of said memory is divided into time intervals, and wherein blocks written during different time intervals are associated with different tags.

11. The method of claim 10, wherein two blocks written during two different power-up sessions of said memory are associated with different tags.

12. A method for storing data, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line;

(b) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(c) associating a group metadata value with each word-line group; and

(d) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value.

13. The method of claim 12, wherein said group metadata value is selected from a group consisting of: a data age, an optimal-reading threshold-voltage, an optimal-verify threshold-voltage, a number of soft bits read, or data related to error correction.

14. The method of claim 12, wherein said group metadata value is stored in a word-line-group table.

15. The method of claim 12, wherein each said word-line is contained in a block, and wherein at least one block contains two word-lines belonging to different word-line groups.

16. The method of claim 12, wherein said step of grouping is based on a tag that is common to all word-lines in a word-line group, wherein said tag is associated with an approximate relative time for writing said word-lines of said word-line group.

17. The method of claim 16, wherein a time since power-up of said memory is divided into time intervals, and wherein word-lines written during different time intervals are associated with different tags.

18. The method of claim 16, wherein word-lines written during two different power-up sessions of said memory are associated with different tags.

19. A method for storing data, the method comprising the steps of:

(a) receiving data to be stored in a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each

word-line, said line metadata value being dependent upon a writing time of said each word-line;

(b) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(c) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag; and

(d) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value, and each word-line utilizes the group metadata value instead of the line metadata value.

20. The method of claim 19, wherein said group metadata value is selected from a group consisting of: a data age, an optimal-reading threshold-voltage, an optimal-verify threshold-voltage, a number of soft bits read, or data related to error correction.

21. The method of claim 19, wherein said group metadata value is stored in a word-line-group table.

22. The method of claim 19, wherein each said word-line is contained in a block, and wherein at least one block contains two word-lines belonging to different word-line groups.

23. A memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block, said controller configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks;

(ii) associating a group metadata value with each block group; and

(iii) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

24. A memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block, said controller configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks;

(ii) associating a sequential tag with each block group, wherein blocks in a first block group having a previous sequential tag are written before blocks in a second block group having a subsequent sequential tag; and

(iii) associating a group metadata value with each respective sequential tag, wherein each block from the plurality of blocks does not store a dedicated copy of

said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

25. A memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of word-lines, wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line, said controller configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(ii) associating a group metadata value with each word-line group; and

(iii) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value.

26. A memory device comprising:

(a) a controller for storing data in a memory that includes a plurality of word-lines, wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line, said controller configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(ii) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag; and

(iii) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value, and each word-line utilizes the group metadata value instead of the line metadata value.

27. A computer-readable non-transitory storage medium having computer-readable code embodied therein, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block, said driver code configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks;

(ii) associating a group metadata value with each block group; and

(iii) associating each group metadata value with each block in a respective block group, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

28. A computer-readable non-transitory storage medium having computer-readable code embodied therein, the computer-readable code comprising:

(a) driver code for a memory device having a that includes a plurality of blocks, wherein each block from the plurality of blocks has at least one block metadata value associated with said each block, said block metadata value being dependent upon a writing time of said each block, said driver code configured for:

(i) grouping said plurality of blocks into block groups, wherein at least one block group contains at least two blocks;

(ii) associating a sequential tag with each block group, wherein blocks in a first block group having a previous sequential tag are written before blocks in a second block group having a subsequent sequential tag; and

(iii) associating a group metadata value with each respective sequential tag, wherein each block from the plurality of blocks does not store a dedicated copy of said block metadata value, and each block utilizes the group metadata value instead of the block metadata value.

29. A computer-readable non-transitory storage medium having computer-readable code embodied therein, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line, said driver code configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(ii) associating a group metadata value with each word-line group; and

(iii) associating each group metadata value with each word-line in a respective word-line group, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said group metadata value.

30. A computer-readable non-transitory storage medium having computer-readable code embodied therein, the computer-readable code comprising:

(a) driver code for a memory device having a memory that includes a plurality of word-lines, and wherein each word-line has at least one line metadata value associated with said each word-line, said line metadata value being dependent upon a writing time of said each word-line, said driver code configured for:

(i) grouping said plurality of word-lines into word-line groups, wherein at least one word-line group contains at least two word-lines;

(ii) associating a sequential tag with each word-line group, wherein word-lines in a first word-line group having a previous sequential tag are written before blocks in a second word-line group having a subsequent sequential tag; and

(iii) associating a group metadata value with each respective sequential tag, wherein each word-line from the plurality of word-lines does not store a dedicated copy of said line metadata value, and each word-line utilizes the group metadata value instead of the line metadata value.