(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 December 2008 (18.12.2008)

PCT

(10) International Publication Number
WO 2008/154119 A1

(51) International Patent Classification:
G06F 11/00 (2006.01) ***G06F 11/10*** (2006.01)

(21) International Application Number: PCT/US2008/063938

(22) International Filing Date: 16 May 2008 (16.05.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/759,585 7 June 2007 (07.06.2007) US

(71) Applicant *(for all designated States except US)*: **MICRON TECHNOLOGY, INC.** [US/US]; 8000 South Federal Way, P.O. Box 6, Boise, ID 83707 (US).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **LI, Tieniu** [CN/US]; 3254 Kilo Avenue, San Jose, CA 95124 (US).

(74) Agent: ALTMAN, Daniel, E.; Knobbe, Martens, Olson & Bear, LLP, 2040 Main Street, 14th Floor, Irvine, CA 92614 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL,

[Continued on next page]

(54) Title: EMERGING BAD BLOCK DETECTION

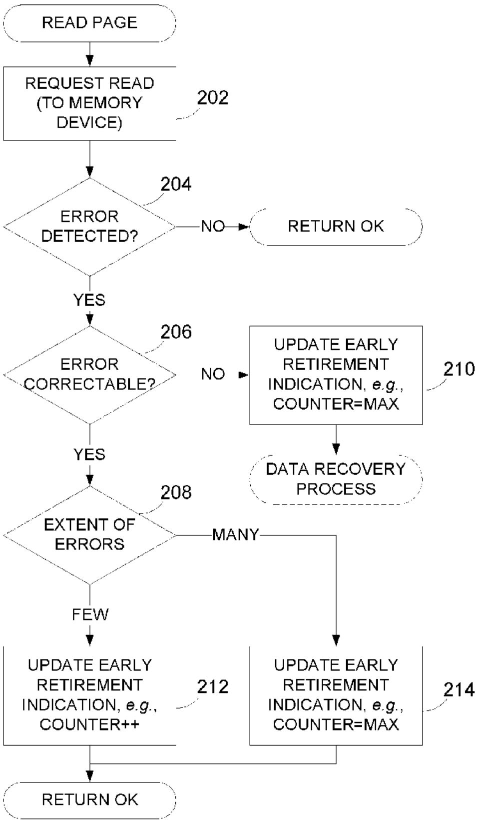

Fig. 2

(57) Abstract: Apparatus and methods, such as those that read data from non-volatile integrated circuit memory devices (106), such as NAND flash. For example, disclosed techniques can be embodied in a device driver (110) of an operating system (104). Errors are tracked during read operations. If sufficient errors are observed (204) during read operations, the block is then retired when it is requested to be erased or a page of the block is to be written (210), (212), (214), (310). One embodiment is a technique to recover data from uncorrectable errors. For example, a read mode can be changed (410) to a more reliable read mode to attempt to recover data. One embodiment further returns data (106) from the memory device regardless of whether the data was correctable (430) by decoding of error correction code data or not.

WO 2008/154119 A1

NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

- *with international search report*
- *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

EMERGING BAD BLOCK DETECTION

Background

Field of the Invention

[0001] Embodiments of the invention generally relate to integrated circuits. In particular, embodiments relate to software or hardware for memory circuits.

Description of the Related Art

[0002] Flash memory is a form of erasable and reprogrammable non-volatile integrated circuit memory. In a flash memory, memory cells are arranged in "blocks" for erasure. After a block has been erased, it is ready for programming if desired. NOR flash memory is a type of flash memory that offers access to individual bytes for retrieval of data, but has relatively low density.

[0003] NAND flash memory is a type of flash memory that offers relatively high density. The high density is achieved in part by forming columns of cells connected in series. In addition, with NAND flash memory, data is programmed and accessed in relatively large groups of bytes, such as a page of data. For example, a page can correspond to a row or to a portion of a row in an array.

[0004] Data is typically written to and read from a NAND flash memory array a "page" of data at a time. For example, a page can have 2,112 bytes of which 2,048 are data bytes and 64 are spare bytes. The spare bytes are typically used for error correction codes (ECC), wear-leveling information, or other overhead data. Error correction codes increase the robustness of the stored data. Typically, a form of block code is used to generate the error correction codes, such as cyclic redundancy check (CRC) checksums, Hamming codes, Reed-Solomon error correction, or the like. These error correction codes detect if there were errors in the reading of the data bytes and can typically correct errors in the data bytes provided that the errors do not exceed the capability of the error correction code.

[0005] Flash memory has many uses. Examples include flash memory hard drives (replacements for hard drives), USB flash drives or thumb drives, mobile phones, digital cameras, digital media players, games, memory cards, navigation devices, personal digital assistants, computers, or the like. Within limits, the error correction codes can correct many errors in the data in the data bytes. However, beyond these limits, data with errors cannot typically be corrected. One disadvantage of conventional techniques is that by the time errors become uncorrectable, it is often too late.

[0006] Many of the devices that use flash memory for data storage also use an operating system. The operating system serves as an abstraction layer between hardware and other software. For example, a file system and a device driver of the operating system typically provide access to data stored on a memory device. There can be additional layers within an operating system.

[0007] Figure 1 illustrates an example of a portion of a processing environment including a microprocessor in the form of a CPU 102, an operating system 104, and a memory device 106. The memory device 106 can be a NAND flash memory device. The operating system 104 further includes a file system 108 and a device driver 110. It will be understood that the operating system 104 can have support for more than one file system and more than one device driver and other components not relevant to the present discussion. Also illustrated are a volatile memory device 114, such as DRAM, and a direct memory access (DMA) controller 116.

[0008] The CPU 102 executes instructions, including the code of the operating system 104. The code of the file system 108 provides abstraction between low-level information, such as logical addresses for the memory device 106, and high-level information, such as a file name and directory. The code for the device driver 110 typically handles low-level information for the data transfer to and from the memory device 106. The device driver 110 can provide code for the CPU 102 to directly access the memory device 106 (known as processors input/output) or can provide code that activates the DMA controller 116 to handle the bus control so that data is transferred to or from the memory device 106. The use of the DMA controller 116 frees up the CPU 102 to handle other tasks.

[0009] With both processor input/output (PIO) and DMA input/output, the device driver 110 can handle ECC information associated with write operations and read operations. In many operating systems, many device drivers exist to support reading to and writing from various different types of memory devices. In addition to the mapping by the file system 108, it should be noted that many NAND flash memory devices utilize virtual mapping, which can be referred to as a flash translation layer, between logical addresses and physical addresses for bad block management and wear out management.

Brief Description of the Drawings

[0010] These drawings and the associated description herein are provided to illustrate specific embodiments of the invention and are not intended to be limiting.

[0011] Figure 1 illustrates a processing environment, including a CPU, an operating system, and a memory device.

[0012] Figure 2 is a flowchart generally illustrating an embodiment of a process for reading a page of data from a memory device.

[0013] Figure 3 is a flowchart generally illustrating an embodiment of a process for proactive retirement of a degraded block.

[0014] Figure 4 is a flowchart generally illustrating an embodiment of a process for recovering data determined to have uncorrectable errors.

Detailed Description of Specific Embodiments

[0015] Apparatus and methods read data from non-volatile integrated circuit memory devices, such as NAND flash. For example, these techniques can be embodied in a device driver of an operating system. At least a partial history of errors is maintained (*e.g.*, tracked) during read operations. Read operations for a page typically occur much more frequently than erase operations for the corresponding block or programming of the page. The errors encountered and tracked during these read operations can provide an indication of the reliability of the page and block being read.

[0016] If sufficient errors are observed during read operations, the block is then indicated (*e.g.*,designated) for retirement. The block can be retired, for example, by adding the block to a bad block table when it is requested to be erased or written (programmed).

[0017] One embodiment includes a technique to recover data from uncorrectable errors. For example, a read mode can be changed to a more reliable read mode to attempt to recover data. This read mode can have less aggressive timing. One embodiment further returns data from the memory device regardless of whether or not the data was corrected by decoding of error correction code data. In contrast to hard disk devices, which typically fail catastrophically, an integrated circuit memory device typically has a small number of bit failures, which leaves most of the data usable even when error correction fails.

[0018] A device driver for an operating system embodying the disclosed techniques can be embodied in software (in code), in hardware (in modules), or in a combination of software and hardware. In addition, while described in the context of NAND flash memory, the principles and advantages described herein will be applicable to other forms of non-volatile integrated circuits as well. Other embodiments of the invention, including embodiments that do not provide all of the benefits and features set forth herein, will be apparent to those of ordinary skill in the art.

[0019] Figure 2 is a flowchart generally illustrating a process for reading a page of data from a memory device 106 (Figure 1) and maintaining a history of errors. In one embodiment, the process is performed by the device driver 110 (Figure 1) of an operating system, and the memory device 106 is a NAND flash memory device. It will be appreciated by the skilled practitioner that the illustrated process can be modified in a variety of ways. For example, in another embodiment, various portions of the illustrated process can be combined, can be rearranged in an alternate sequence, can be removed, and the like. At the start of the process, it is assumed that data has already been stored in the memory device along with corresponding error correction code data.

[0020] The illustrated embodiment of the process will generally be described in connection with reading a single page of a block of the memory device 106 (Figure 1). A counter is used to maintain the history of errors for the block. The memory device 106 has many blocks and can have one counter per block for the tracking of errors. In one embodiment, a counter array is used to store the counters. When a counter for a block reaches a particular threshold level, *MAX*, which can be a predetermined value such as a value of 10, the process determines that the block is to be retired early. For example, the block can be retired when it is to be erased or when a page of the block is to be written as will be described in greater detail later in connection with Figure 3. Volatile or non-volatile memories can be used to store the counters. For robustness, however, it is typically preferred to store this type of information on a different device than the memory device 106 itself. In one embodiment, the counters are reset to zero upon power up such that the history is reset upon power up. This resetting provides a relatively fast response to a sudden degradation in a block.

[0021] The process begins by receiving a request for a read of a particular page of the memory device 106 (Figure 1). The request can be via a subroutine call from a higher layer of the operating system 104 (Figure 1) than the device driver 110 (Figure 1), such as from the file system 108 (Figure 1). The process requests 202 the particular page using low-level instructions, such as machine code to allow the CPU 102 to communicate with the memory device 106 or instructions for the CPU 102 to instruct the DMA controller 116 to communicate with the memory device 106. The memory device 106 internally accesses the memory array(s) and provides data for the page. The ECC data is decoded to detect whether there were any errors and also to correct errors, should errors exist and be correctable. The decoding of ECC can be a software function or a hardware function. In one embodiment, the encoding and decoding of ECC is a function performed by the device driver 110. In an alternative embodiment, the encoding and decoding of ECC is a function of the DMA controller 116.

[0022] The process determines 204 whether or not one or more errors were detected when the page was read. Errors can be correctable or not. If no error was encountered during the error detection process, the process proceeds to exit (without updating a counter for the block corresponding to the page) and returns a status of "OK" to the calling routine of the device driver 110 (Figure 1), such as a file system 108 (Figure 1). The requested data is available at an output buffer of the memory device 106 (Figure 1), at a logical address for a data bus, can be transferred to another location such as to another buffer, or the like.

[0023] If the process determines 204 that an error(s) has occurred, the process proceeds to determine 206, whether or not the error(s) is correctable via decoding of the error correction code (ECC) data. If the error(s) is correctable within the limitations of the ECC, the process proceeds to determine 208 the extent to which the error(s) that had been corrected. For example, in one embodiment, the process compares the uncorrected data with the corrected data to obtain a count of the number of error(s). Otherwise, if the errors were too numerous for correction by the error correction code, the process indicates the block for early retirement 210. In one embodiment, the counter is set 210 to the threshold value, *MAX*, to provide the indication of early retirement. Indicating that a block is to be retired early is not the same as actually retiring the block.

[0024] A block can be retired by marking the block as bad, such as by adding the block to a bad block table. The bad block table is typically stored on the memory device 106 (Figure 1) itself. However, until data from the block is no longer desired, the block should not yet be marked as bad. Marking of the block as bad will prevent read access to the block. Accordingly, in one embodiment, the process waits until an erase operation is requested or a write operation is requested to mark the block as bad, as will be described in greater later in connection with Figure 3. In one embodiment, the process returns an error to a calling routine, such as a file system 108 (Figure 1), which then initiates procedures to retire the block.

[0025] In one embodiment, the process advances from the indicate early retirement state 210 to a data recovery process to attempt to recover the data. One embodiment of such a data recovery process will be described in greater detail later in connection with Figure 4. In an alternative embodiment, the process advances from the indicate early retirement state 210 and returns a "read error" status to the calling routine of the device driver, such as the file system 108 (Figure 1), and exits.

[0026] Returning now to the determination 206 of whether the error(s) can be corrected, if the error(s) can be corrected, the process determines 208 the extent to which the ECC corrected the error(s). The number of errors that are considered to be relatively few or relatively many can depend on the page size and the characteristics of the particular memory device 106 (Figure 1). Appropriate thresholds will be readily determined by one of ordinary skill in the art. In addition, while the illustrated determination 208 is binary, *i.e.*, two possible decisions, it will be understood that one or more intermediate levels can also be used.

[0027] In one embodiment, with a 2048-byte page (not including ECC), "few" is considered one correctable error on the page, and "many" is considered two or more correctable errors. In this example, when there is one correctable error on the page, the process proceeds from the determination 208 to an increment counter state 212, and when there is more than one correctable error, the process proceeds from the determination 208 to an indicate early retirement state 214.

[0028] In the increment counter state 212, the process increments the counter by a count of 1 for the particular block of the page being read. A counter array can store various counts for the multiple blocks of the memory device 106 (Figure 1), *e.g.*, one counter per block of the memory device 106. As a block contains many pages, errors from multiple pages can contribute to the count for a particular block. The process then proceeds to return a status of "OK" to the calling routine, *e.g.*, the file system 108 (Figure 1), of the device driver 110 (Figure 1) and exits. As will be observed, if the same page or other pages of the block are repeatedly read with error(s), the count stored in the counter for the block will increase. The count stores a history of errors encountered while reading page(s) of the block.

[0029] Returning now to the path of the determination 208 used when relatively many errors, *e.g.*, 2 or more, were encountered, in the indicate early retirement state 214, the process updates the counter to designate the block for early retirement. For example, the counter can be set to the threshold level, *MAX*, as described in connection with the state 210. In the illustrated embodiment, the state 210 and the state 214 each set the counter to the threshold level, *MAX*, to designate the block for early retirement. In an alternative embodiment, the state 214 adjusts the counter by an increment that is equal to or larger than the increment of the state 212, but potentially smaller than a jump in counter value from the state 210. The process then proceeds to return a status of "OK" to the calling routine, *e.g.*, the file system 108 (Figure 1), of the device driver 110 (Figure 1) and exits. The process illustrated in Figure 2 identifies blocks that are degrading and should be retired. It can be advantageous for reliability purposes to recognize degrading blocks relatively early, such as when data is being read from a block, rather than relatively late, such as when new data is being written to the block.

[0030] Figure 3 is a flowchart generally illustrating a process for the proactive retirement of a degraded block. In one embodiment, the process is performed by a device driver of an operating system, and the memory device is a NAND flash memory device. It will be appreciated by the skilled practitioner that the illustrated process can be modified in a variety of ways. For example, in another embodiment, various portions of the illustrated process can be combined, can be rearranged in an alternate sequence, can be removed, or the like. In a NAND flash memory, while reading or writing is performed at the page level, erasing is performed at the block level.

[0031] The process is initiated by a request to erase a block from the memory device or by a request to write to a page of a block. For example, the request can come from the file system 108 (Figure 1) of the operating system 104 (Figure 1). The request can be in the context of updating data, can be for deleting data, can be for formatting, or the like.

[0032] The process retrieves 310 an error history associated with the block to be erased or the corresponding block of the page to be written. For example, the error history can be the count described earlier in connection with Figure 2.

[0033] The process proceeds to determine whether the block is reliable 320. If the block is deemed to be reliable, the process proceeds to instruct the memory device 106 (Figure 1) to erase 330 the block. The process determines 350 whether or not an error occurred during the erasing process or the writing process within the memory device 106. Assuming that the memory device 106 erases the block or writes to the page without error, the process then exits with a status of "OK" for the calling routine, such as the file system 108 (Figure 1). If there is an error in the erase or write process, such as stuck bits, then the process can exit with an error status.

[0034] If the block is deemed to be unreliable, the process proceeds to refuse 360 the requested erase or write operation and exits with an error status, such as an I/O error. For example, the block can be deemed to be unreliable if the counter is at or above the threshold value for identification of a potentially unreliable block. In response to the error status, a higher layer calling routine such as the file system 108 (Figure 1) can then reclaim previously written pages, if any, of the block before requesting the device driver 110 (Figure 1) to mark the block as bad. The previously written pages as well as a current page to be written can be written to a different block. After a block is marked as bad, the block is not usable. In some memory devices, marking a block as bad is accomplished by adding the block to a bad block table in the memory device 106 (Figure 1). In one embodiment, the addition of the block to the bad block table is also handled by the device driver 110 (Figure 1) at the request of the file system 108. The early retirement of the block can advantageously permit the block to be retired before the block goes bad. This can help to avoid the future loss of data.

[0035] Figure 4 is a flowchart generally illustrating a process for recovering data determined to have uncorrectable errors. In one embodiment, the process is performed by a device driver 110 (Figure 1) of an operating system 104 (Figure 1), and the memory device

106 (Figure 1) is a NAND flash memory device. It will be appreciated by the skilled practitioner that the illustrated process can be modified in a variety of ways. For example, in another embodiment, various portions of the illustrated process can be combined, can be rearranged in an alternate sequence, can be removed, and the like.

[0036] In one embodiment, the process starts when a data read from the memory device resulted in uncorrectable errors. An uncorrectable error can result when there are too many errors for the error correction codes (ECC) to correct. For example, the process can begin following the indicate early retirement state 210 of Figure 2.

[0037] In the state 410, the process changes a read mode of operation to attempt to read the data in a correctable manner. Most computer systems are configured to access memory devices, such as the memory device 106 (Figure 1) at the highest compatible speed during normal operation (fast read mode). In one embodiment, the read mode of operation is changed to a reliable read mode. This can include, for example, slower bus speeds, longer latencies, or the like. In another example, the read mode can be changed from a direct memory access (DMA) mode to a mode using the microprocessor to control access, which is also known as processor input/output or PIO. Typically, the DMA mode is preferred because its use frees up the microprocessor to perform other tasks. However, with PIO, the microprocessor can perform a read operation with more control than with DMA. For example, the process can request the temporary disabling of a device that generates noise, such as a speaker of a cell phone. In another example, the process can deliberately slow down timing for the memory device 106. Other techniques will be readily determined by one of ordinary skill in the art. Various features of these techniques can also be combined.

[0038] The process proceeds to request a read 420 of the page previously read from the memory device. This read corresponds to a re-read. The error correction code data are also retrieved and reapplied to determine 430 if the data is now correctable. In many circumstances, the data can be correctable after the read using a more reliable mode. If the data is correctable, the process proceeds to provide 440 the corrected data, and the process exits with a status of "OK."

[0039] If the data is uncorrectable, the process determines 450 whether to retry or to provide data 460 with errors. The determination 450 to retry or to provide data can be based on, for example, a loop counter, a timer, and the like. The process can return to the state 410 or the read state 420 to re-read data. If the process returns to the state 410, a different (*e.g.*, third, fourth, etc.) read mode can also be tried. For example, if more than one reliable read mode exists, the features of various read modes can be tried separately and/or in combination.

[0040] If the data is still not correctable, one embodiment nevertheless provides data 460 with the errors. Unlike hard disk devices, which tend to fail catastrophically, errors in solid-state devices such as NAND flash are often characterized by single-bit failures. While the number of errors will typically be greater than one bit when the page is uncorrectable, the return of data having an error can be better than no return of data. For example, in many instances, the portion of the data with the error will be unnoticeable. For example, if program code is stored in the page, the portion with the error can correspond to a portion of the code that is not always executed. By contrast, when no data is provided, then the entire page's data is not available, in which case, a program nearly always crashes.

[0041] One embodiment is a method of monitoring a block of a non-volatile integrated circuit memory device, wherein the method includes: decoding error correction codes to determine whether or not a page of data of the block of the memory device was read with at least one error; and maintaining a history of errors for the block of the memory device corresponding to the page that was read at least when the page was read with at least one error, wherein the history of errors is used to indicate whether or not to retire the block early.

[0042] One embodiment is a method of managing blocks for a non-volatile integrated circuit memory device, wherein the method includes: receiving a command to erase a block of memory or write to a page of the block of memory; and disabling an erase or write based at least in part on a history of errors associated with read operations for one or more pages of the block.

[0043] One embodiment is a method of accessing a page of data from a non-volatile integrated circuit memory device, wherein the method includes: requesting a read of the page of data from the memory device; inspecting error correction codes (ECC) associated with the page; and providing a status to an operating system that indicates that the page of data was read correctly regardless of whether or not the page of data included any uncorrectable errors.

[0044] One embodiment is a method of maintaining one or more indications for indicating that a block of a non-volatile integrated circuit memory device, wherein the method includes preventing further erasing or reprogramming of the block at least partially in response to at least one of: determining that an error detected as part of at least one read operation performed with respect to at least one page of the block of memory is not correctable via decoding of an error correction code; or determining that a history of errors collected for the blocks of the memory device indicates that at least one error has occurred during a prior read operation for the block.

[0045] One embodiment is an apparatus for monitoring a block of a non-volatile integrated circuit memory device, wherein the apparatus includes: a module configured to decode error correction codes to determine whether or not a page of data of the block of the memory device was read with at least one error; and a module configured to maintain a history of errors for the block of the memory device corresponding to the page that was read at least when the page was read with at least one error, wherein the history of errors is used to indicate whether or not to retire the block early.

[0046] Various embodiments have been described above. Although described with reference to these specific embodiments, the descriptions are intended to be illustrative and are not intended to be limiting. Various modifications and applications may occur to those skilled in the art without departing from the true spirit and scope of the invention as defined in the appended claims.

I CLAIM:

1. A method of monitoring a block of a non-volatile integrated circuit memory device, the method comprising:

    decoding error correction codes to determine whether or not a page of data of the block of the memory device was read with at least one error; and

    maintaining a history of errors for the block of the memory device corresponding to the page that was read at least when the page was read with at least one error, wherein the history of errors is used to indicate whether or not to retire the block early.

2. The method of Claim 1, wherein maintaining the history further comprises:

    maintaining a count, wherein the count is incremented at least partially in response to determining that the at least one error corresponds to one error;

    wherein early retirement for the block is indicated in response to the count reaching a threshold value.

3. The method of Claim 2, further comprising resetting the count upon power up of a device using the memory device.

4. The method of Claim 1, wherein maintaining the history further comprises updating the history to indicate early retirement for the block when multiple errors are encountered when reading the page of the block, wherein a count of the number of errors is determined by comparing uncorrected data and corrected data.

5. The method of Claim 1, wherein maintaining the history comprises setting a count for the history to at least a threshold value, wherein the block is indicated for early retirement if the count reaches at least the threshold value.

6. The method of Claim 1, wherein the at least one error comprises multiple errors that are uncorrectable, further comprising initiating a data recovery process to attempt to recover the data.

7. The method of Claim 1, wherein the at least one error comprises multiple errors that are uncorrectable by decoding of error correction codes (ECC), and wherein the page was read using a first read mode, further comprising changing to a different read mode and re-reading the page.

8. The method of Claim 1, wherein the at least one error comprises uncorrectable errors further comprising returning a status to an operating system of a successful read operation even when an uncorrectable error is read.

9. The method of Claim 1, further comprising maintaining the history of errors for the block of the memory device on a different device than the memory device itself.

10. The method of Claim 1, wherein the memory device comprises a NAND flash memory device.

11. The method of Claim 1, wherein the method is performed by an operating system.

12. A method of managing blocks for a non-volatile integrated circuit memory device, the method comprising:

receiving a command to erase a block of memory or write to a page of the block of memory; and

disabling an erase or write based at least in part on a history of errors associated with read operations for one or more pages of the block.

13. The method of Claim 12, further comprising reclaiming previously written pages of the block, and then disabling use of the block.

14. The method of Claim 12, wherein disabling comprises adding the block to a bad block table in the memory device to mark the block as bad.

15. The method of Claim 12, further comprising inspecting a count of errors encountered during read operations for one or more pages of the block.

16. A method of accessing a page of data from a non-volatile integrated circuit memory device, the method comprising:

requesting a read of the page of data from the memory device;

inspecting error correction codes (ECC) associated with the page; and

providing a status to an operating system that indicates that the page of data was read correctly regardless of whether or not the page of data included any uncorrectable errors.

17. The method of Claim 16, wherein at least a first read mode of operation and a second read mode of operation slower than the first read mode of operation are selectable for reading data from the device, wherein requesting the read of the page further comprises:

determining that a decoding of an initial read has resulted in an uncorrectable error, wherein the initial read was performed using the first read mode of operation;

changing from the first read mode of operation to the second read mode of operation; and

requesting another read of the page.

18. The method of Claim 17, wherein the first read mode of operation comprises direct memory access (DMA) and wherein the second read mode of operation comprises using the microprocessor to control access.

19. The method of Claim 17, further comprising determining whether to request another read of the page based at least partly on a loop counter.

20. The method of Claim 17, further comprising determining whether to request another read of the page based at least partly on a timer.

21. The method of Claim 16, further comprising performing the method by a driver of an operating system.

22. A method of maintaining one or more indications for indicating that a block of a non-volatile integrated circuit memory device, the method comprising:

preventing further erasing or reprogramming of the block at least partially in response to at least one of:

determining that an error detected as part of at least one read operation performed with respect to at least one page of the block of memory is not correctable via decoding of an error correction code; or

determining that a history of errors collected for the blocks of the memory device indicates that at least one error has occurred during a prior read operation for the block.

23. An apparatus for monitoring a block of a non-volatile integrated circuit memory device, the apparatus comprising:

a module configured to decode error correction codes to determine whether or not a page of data of the block of the memory device was read with at least one error; and

a module configured to maintain a history of errors for the block of the memory device corresponding to the page that was read at least when the page was read with at least one error, wherein the history of errors is used to indicate whether or not to retire the block early.

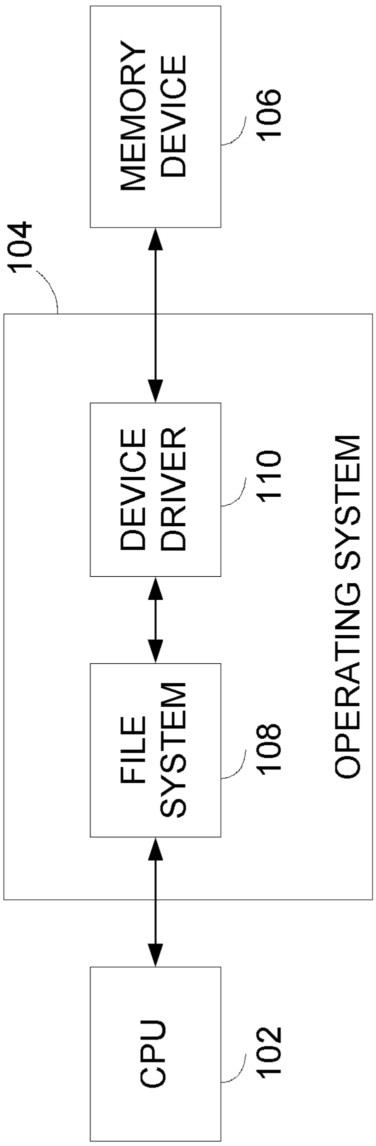
CPU
102
FILE SYSTEM
108
DEVICE DRIVER
110
OPERATING SYSTEM
104
MEMORY DEVICE
106

Fig. 1

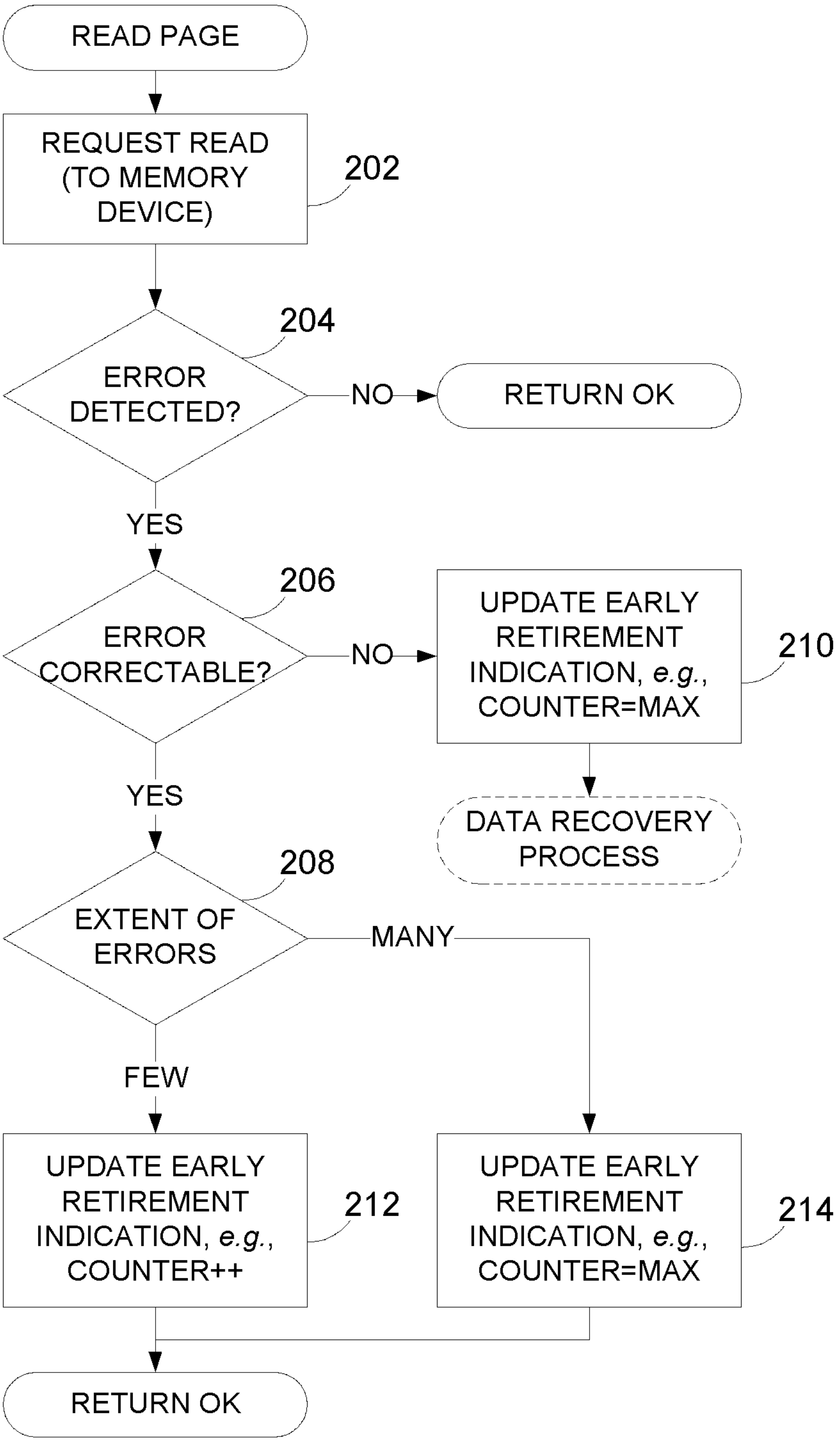
READ PAGE
REQUEST READ (TO MEMORY DEVICE)
202
204
ERROR DETECTED?
NO
RETURN OK
YES
206
ERROR CORRECTABLE?
NO
UPDATE EARLY RETIREMENT INDICATION, e.g., COUNTER=MAX
210
DATA RECOVERY PROCESS
YES
208
EXTENT OF ERRORS
MANY
FEW
UPDATE EARLY RETIREMENT INDICATION, e.g., COUNTER++
212
UPDATE EARLY RETIREMENT INDICATION, e.g., COUNTER=MAX
214
RETURN OK

Fig. 2

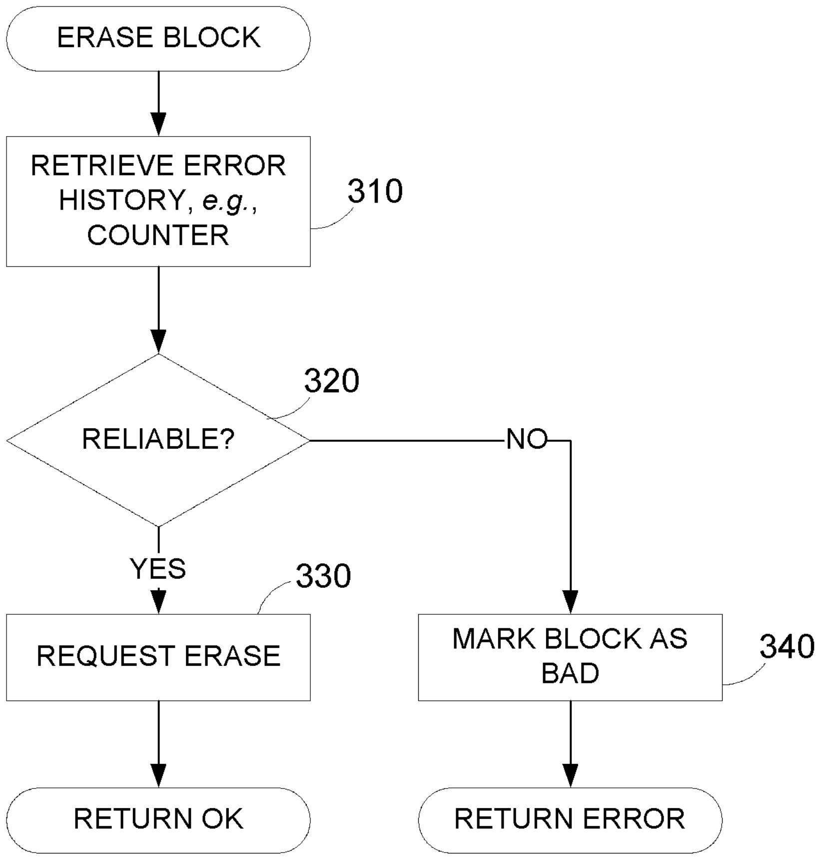
ERASE BLOCK
RETRIEVE ERROR HISTORY, *e.g.*, COUNTER
310
RELIABLE?
320
NO
YES
330
REQUEST ERASE
MARK BLOCK AS BAD
340
RETURN OK
RETURN ERROR

Fig. 3

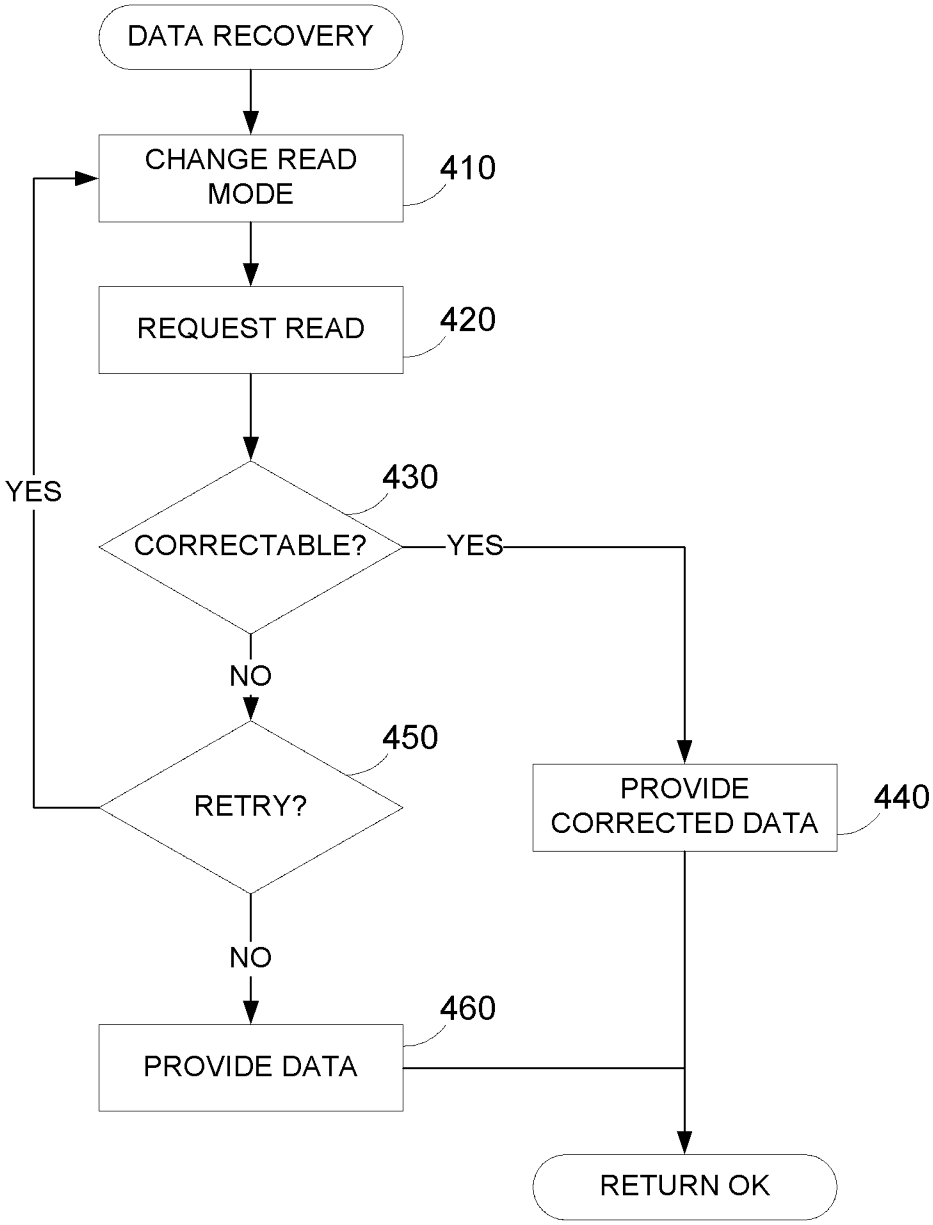
DATA RECOVERY
CHANGE READ MODE
410
REQUEST READ
420
CORRECTABLE?
430
YES
NO
RETRY?
450
YES
NO
PROVIDE CORRECTED DATA
440
PROVIDE DATA
460
RETURN OK

Fig. 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2008/063938

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F11/00 G06F11/10

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, INSPEC, COMPENDEX

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2003/225961 A1 (CHOW JAMES [US] ET AL) 4 December 2003 (2003-12-04) paragraphs [0012], [0019], [0027], [0041] - [0048], [0051] claim 1 figures 1,2 -----	1-11,23
X	US 6 339 546 B1 (KATAYAMA KUNIHIRO [JP] ET AL) 15 January 2002 (2002-01-15) column 1, line 49 - column 2, line 17 column 2, line 29 - line 44 column 3, line 1 - line 33 column 3, line 56 - column 5, line 45 column 5, line 66 - column 6, line 43 figures 1,2,7,8 -----	1-11,23

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 28 July 2008	Date of mailing of the international search report 23/10/2008
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Fax: (+31–70) 340–3016	Authorized officer Johansson, Ulf

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are ccordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers allsearchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search reportcovers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

1-11,23

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-11,23

Monitoring a block of a non-volatile integrated circuit memory device, the method comprising: decoding error correction codes to determine whether or not a page of data of the block of the memory device was read with at least one error; and maintaining a history of errors for the block of the memory device corresponding to the page that was read at least when the page was read with at least one error, wherein the history of errors is used to indicate whether or not to retire the block early.

---

2. claims: 12-15

Managing blocks for a non-volatile integrated circuit memory device, the method comprising: receiving a command to erase a block of memory or write to a page of the block of memory; and disabling an erase or write based at least in part on a history of errors associated with read operations for one or more pages of the block.

---

3. claims: 16-21

Accessing a page of data from a non-volatile integrated circuit memory device, the method comprising: requesting a read of the page of data from the memory device; inspecting error correction codes associated with the page; and providing a status to an operating system that indicates that the page of data was read correctly regardless of whether or not the page of data included any uncorrectable errors.

---

4. claim: 22

Maintaining one or more indications for indicating that a block of a non-volatile integrated circuit memory device, the method comprising: preventing further erasing or reprogramming of the block at least partially in response to at least one of: determining that an error detected as part of at least one read operation performed with respect to at least one page of the block of memory is not correctable via decoding of an error correction code; or determining that a history of errors collected for the blocks of the memory device indicates that at least one error has occurred during a prior read operation for the block.

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2008/063938

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003225961 A1	04-12-2003	AU 2003239201 A1	19-12-2003
		EP 1509845 A1	02-03-2005
		JP 2005528696 T	22-09-2005
		WO 03102782 A1	11-12-2003
US 6339546 B1	15-01-2002	AU 7313600 A	24-04-2001
		WO 0122232 A1	29-03-2001
		JP 3937214 B2	27-06-2007