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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 October 2010 (21.10.2010)

PCT

(10) International Publication Number
WO 2010/120475 A2

(51) **International Patent Classification**:
G06F 12/16 (2006.01) *G06F 11/08* (2006.01)
G06F 3/06 (2006.01)

(21) **International Application Number**:
PCT/US2010/028853

(22) **International Filing Date**:
26 March 2010 (26.03.2010)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
12/424,766 16 April 2009 (16.04.2009) US

(71) **Applicant** *(for all designated States except US)*: **MICRON TECHNOLOGY, INC.** [US/US]; 8000 South Federal Way, Ms 525, Boise, Idaho 83716 (US).

(72) **Inventors; and**
(75) **Inventors/Applicants** *(for US only)*: **LARSEN, Troy** [US/US]; 520 East 3550 North, North Ogden, Utah 84414 (US). **CULLEY, Martin** [US/US]; 7336 W. Coho Drive, Boise, Idaho 83709 (US). **MANNING, Troy** [US/US]; 4753 S. Ten Mile Rd., Meridian, Idaho 83642 (US).

(74) **Agent**: **BOLVIN, Kenneth W.**; Leffert Jay & Polglaze, P.A., P.O. Box 581009, Minneapolis, Minnesota 55458-1009 (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, CL, 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, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, 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, LR, 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, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) **Title**: DATA RECOVERY IN A SOLID STATE STORAGE SYSTEM

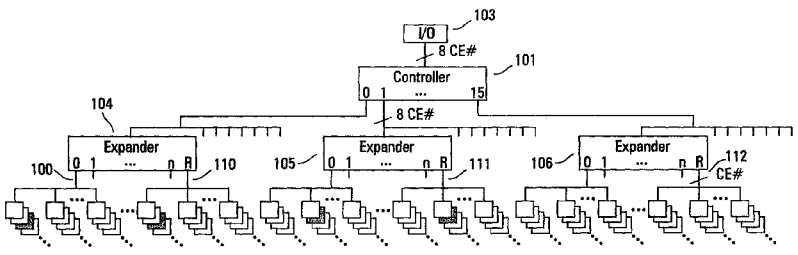


FIG. 1

(57) **Abstract**: Methods for data recovery and memory systems are provided. According to at least one such method, when defective data is read from a memory location, the data is recovered by an XOR operation on the remaining good data and associated RAID data to reconstruct the defective data. The defective data is excluded from the XOR operation.

WO 2010/120475 A2

DATA RECOVERY IN A SOLID STATE STORAGE SYSTEM

TECHNICAL FIELD OF THE INVENTION

The present invention relates generally to memory and in a particular embodiment the present invention relates to non-volatile memory.

BACKGROUND OF THE INVENTION

Memory is typically provided as internal, semiconductor, integrated circuits in computers or other electronic devices. There are many different types of memory including random access memory (RAM), read only memory (ROM), dynamic random access memory (DRAM), synchronous dynamic random access memory (SDRAM), and flash memory.

A flash memory is a type of memory that can be erased and reprogrammed in blocks instead of one byte at a time. A typical flash memory comprises a memory array that includes a large number of memory cells. Each of the memory cells includes a floating gate field-effect transistor capable of holding a charge. The cells are usually grouped into blocks of 64 pages of single level cells (SLC) or 128 pages of multilevel cells (MLC), where each page is typically 2048 bytes of data on 32 word lines. Each of the cells within a block can be electrically programmed on a random basis by charging the floating gate.

Non-volatile memory can be incorporated into solid state storage devices such as solid state drives. Solid state drives, incorporating a large number of individual non-volatile memory devices, such as a large number of memory die, can be used to replace the hard disk drives in computers that typically use magnetic or optical disks for storing large amounts of data. A solid state drive does not use moving parts whereas a hard disk drive requires a complex and sensitive drive and read/write head assembly to interact with the magnetic/optical disk. Thus, the solid state drives are more resistant to damage and loss of data through vibration and impacts.

As with most semiconductor memory, non-volatile memory can have a number of failed memory locations (e.g., bad cells, bad pages, bad blocks, etc.). The failed memory locations may be defectively fabricated or can also fail during use. If a memory location fails during operation, the data stored in that location or locations may be lost.

For the reasons stated above, and for other reasons stated below which will become apparent to those skilled in the art upon reading and understanding the present specification, there is a need in the art for a way to recover data from defective memory locations.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a block diagram of one embodiment of a solid state storage system.

Figure 2 shows a block diagram of one embodiment of a memory communication channel coupled to a plurality of memory devices in accordance with the solid state storage system of Figure 1.

Figure 3 shows a block diagram of one embodiment of a non-volatile memory device that uses a memory communication channel.

Figure 4 shows a schematic diagram of one embodiment of a portion of a non-volatile memory array in accordance with a non-volatile memory device of Figure 3.

Figure 5 shows a conceptual block diagram of one embodiment of a data programming operation utilizing RAID functions.

Figure 6 shows a conceptual block diagram of one embodiment of a RAID read page and data reconstruction.

Figure 7 shows a flowchart of one embodiment of a method for data recovery in a solid state storage system in accordance with the system of Figure 1.

DETAILED DESCRIPTION

In the following detailed description of the invention, reference is made to the accompanying drawings that form a part hereof, and in which is shown, by way of illustration, specific embodiments in which the invention may be practiced. In the drawings, like numerals describe substantially similar components throughout the several views. These embodiments are described in sufficient detail to enable those skilled in the art to practice the invention. Other embodiments may be utilized and structural, logical, and electrical changes may be made without departing from the scope of the present invention. The following detailed description is, therefore, not to be taken in a limiting sense, and the scope of the present invention is defined only by the appended claims and equivalents thereof.

Figure 1 illustrates a block diagram of one embodiment of a solid state storage system, such as a solid state drive, that incorporates non-volatile memory. Such a system can emulate a hard disk drive in a computer system. Alternate embodiments can use the solid state storage system in other applications.

The memory system is comprised of a system controller 101 that controls operation of the solid state storage system and enables the system to emulate a hard disk drive as well as perform other tasks. The controller 101 has a plurality of communication channels that allow it to communicate with and control a plurality of memory targets. In the illustrated embodiment, the controller 101 has 16 communication channels and each communication channel is comprised of eight chip enables (i.e., $\overline{CE0}$ - $\overline{CE7}$).

The controller 101 is coupled to an I/O interface 103 such as a peripheral component interconnect express (PCIe) interface or some other interface. The I/O interface 103 enables the controller 101 to communicate with external systems, such as a computer system, by handling standardized bus communication signals between the solid state storage system and the external system.

The controller 101 is also coupled to a plurality of expander blocks 104 – 106 that expand the communication channels usable by the controller 101. Each communication channel from the controller 101 can be coupled to a respective one of eight different expander blocks. Each expander block, such as blocks 104 – 106, has multiple communication channels 100, where each of the expander block communication channels 100 can be coupled to multiple memory targets. One example of such an expander block communication channel 100 is illustrated in Figure 2 that is described subsequently.

An expander block communication channel might contain a plurality of distinct chip enable signals, such as $\overline{CE0}$ - $\overline{CE7}$. Each of these chip enable signals may in turn be coupled to a respective one of a plurality of memory targets 201-208. A memory target can refer to one or more logical units (LUNs) of memory. In some embodiments, a memory target might be a single memory package that includes multiple LUNs. Typically, a LUN corresponds to a single memory device, such as a single die.

Figure 2 depicts an embodiment where an expander block communication channel 100 consists of eight chip enable signals respectively coupled to eight memory targets 201-208, where each of the depicted memory targets corresponds to a single memory device.

LUN can also contain multiple planes of memory device dies. Each LUN in a particular memory target, such as one of memory targets 201 – 208, might share the same chip enable signal (e.g., all of the LUNs in memory target 201 might share chip enable signal $\overline{CE0}$) so that each logical unit in that memory target is accessed by a respective address bit to distinguish the multiple LUNs (e.g., multiple dies) in the target. The multiple LUNs may operate in parallel on different commands at the same time.

Each expander block 104 – 106 incorporates an extra communication channel 110 – 112 that is used as a RAID channel by the controller 101. The RAID channel is coupled to one or more additional memory targets that are used by the system controller 101 for storing data redundant to that stored in other targets coupled to that expander block for each of the other communication channels coupled to that expander block.

Since NAND flash memory targets do not typically have address pins, addresses are loaded through a five-cycle sequence during command processing. Two address types are used: a column address and a row address. The column address is used to access bytes or words within a page. The row address is used to address pages, blocks, and/or LUNs. The row address structure, in one embodiment, is comprised of lower order row address bits for page addressing, middle row address bits for block addressing, and the most significant row address bits for LUN addressing.

Figure 2 illustrates a block diagram of one embodiment of an expander block communication channel 100 comprising a plurality of memory targets 201 – 208. This figure shows the address/data bus 210, Read/ $\overline{Write}$ control signal 211, and chip enable signals 212 that make up one of the expander block communication channels. The illustrated embodiment includes eight separate memory targets so that eight chip enable signals ($\overline{CE0}$ – $\overline{CE7}$) are used. In Figure 2, each of the memory targets are depicted as respective memory device 201 – 208 (e.g., respective separate die) which may be stacked with one or more of the other memory devices.

The embodiment of Figure 2 is for purposes of illustration only. A solid state storage system may use only one memory device 201 or multiple memory devices. For example, a solid state storage system could be comprised of thousands of non-volatile memory devices organized into groups of non-volatile memory devices 201, 202 in which

each memory device in the target shares a single chip enable line. Each of the plurality of chip enable lines are coupled to a different group of non-volatile memory devices.

Figure 3 illustrates a functional block diagram of a single LUN, such as a non-volatile memory device 300 that can be incorporated on an integrated circuit die. The non-volatile memory device 300, in one embodiment, is a NAND flash memory. The non-volatile memory device 300 has been simplified to focus on features of the memory that are helpful in understanding the present programming embodiments.

The non-volatile memory device 300 includes an array 330 of non-volatile memory cells such as the floating gate memory cells that are illustrated in Figure 4 and discussed previously. The memory array 330 is arranged in banks of access line (e.g., word line) rows and data line (e.g., bit line) columns. In one embodiment, the columns of the memory array 330 are comprised of series strings of memory cells. As is well known in the art, the connections of the cells to the bit lines determines whether the array is a NAND architecture, an AND architecture, or a NOR architecture.

The memory array 330 can be organized into memory blocks. The quantity of memory blocks is typically determined by the size of the memory device (i.e., 512 MB, 1GB). In one embodiment, each memory block can be organized into 64 pages of single level cells (SLC). In an alternate embodiment, each memory block can be organized into 128 pages of multiple level cells (MLC). Each page of data, in one embodiment, can be comprised of 2048 bytes of data on 32 word lines.

Address buffer circuitry 340 is provided to latch address signals provided through the I/O circuitry 360. Address signals are received and decoded by a row decoder 344 and a column decoder 346 to access the memory array 330. It will be appreciated by those skilled in the art, with the benefit of the present description, that the number of address input connections depends on the density and architecture of the memory array 330. That is, the number of addresses increases with both increased memory cell counts and increased bank and block counts. Data is also input and output through the I/O circuitry 360 based on the timing of the control signals provided on control bus 372.

The non-volatile memory device 300 reads data in the memory array 330 by a sense operation that senses voltage or current changes in the memory array columns using sense circuitry 350. The sense circuitry 350, in one embodiment, is coupled to read and latch a row

of data from the memory array 330. Data input and output buffer circuitry 360 is included for bidirectional data communication as well as address communication over a plurality of data connections 362 with an external controller. Write circuitry 355 is provided to write data to the memory array.

The memory control circuitry 370 decodes signals provided on control bus 372 from an external controller and/or expander block. These signals can include read/write (R/$\overline{W}$), chip enable (CE), command latch enable (CLE), address latch enable (ALE) as well as other control signals that are used to control the operations on the memory array 330 as well as other circuitry of the memory device 300. In one embodiment, these signals are active low but alternate embodiments can use active high signals. The memory control circuitry 370 may be a state machine, a sequencer, or some other type of controller to generate the memory control signals.

The non-volatile memory device 300 communicates with an external controller and/or expander block over a channel 390. In one embodiment, the channel 390 is comprised of the memory address, data, and control signals between the external controller and/or expander block and the memory device 300. The embodiment of Figure 3 shows the address and data being coupled as one bus to the I/O circuitry 360. In an alternate embodiment, the address and data buses are separate inputs/outputs with the memory device 300.

Figure 4 illustrates a schematic diagram of a portion of a NAND architecture memory array comprising series strings of non-volatile memory cells. While the subsequent discussions refer to a NAND memory device, the present embodiments are not limited to such an architecture.

The memory array is comprised of an array of non-volatile memory cells 401 (e.g., floating gate) arranged in columns such as series strings 404, 405. Each of the cells 401 are coupled drain to source in each series string 404, 405. A word line WL0 – WL31 that spans across multiple series strings 404, 405 is connected to the control gates of each memory cell in a row in order to bias the control gates of the memory cells in the row. Bit lines BL1, BL2 are eventually connected to sense circuitry (not shown) that detect the state of each cell by sensing current on a particular bit line.

Each series string 404, 405 of memory cells is coupled to a source line 406 by a source select gate 416, 417 and to an individual bit line BL1, BL2 by a drain select gate 412,

413. The source select gates 416, 417 are controlled by a source select gate control line SG(S) 418 coupled to their control gates. The drain select gates 412, 413 are controlled by a drain select gate control line SG(D) 414.

Each memory cell can be programmed as an SLC device or an MLC device. Each cell's threshold voltage (V_t) is indicative of the data that is stored in the cell. For example, in an SLC, a V_t of 0.5V might indicate a programmed cell while a V_t of –0.5V might indicate an erased cell. The MLC may have multiple V_t ranges that each indicate a different state. Multiple level cells can take advantage of the analog nature of a traditional flash cell by assigning a bit pattern to a specific voltage range stored on the cell. This technology permits the storage of two or more bits per cell, depending on the quantity of voltage ranges assigned to the cell.

Redundant Array of Independent Disks (RAID) is a technology that employs the simultaneous use of two or more media (e.g., non-volatile memory devices) to achieve greater levels of performance, reliability, and/or larger data volume sizes. The RAID support provided by the solid state storage system of Figure 1 divides and/or replicates data among multiple memory targets that are addressed in unison. The multiple memory targets are seen by a computer operating system coupled to the solid state storage system of Figure 1 as one single hard disk drive.

The solid state storage system of Figure 1 can use a page addressing mode so that the expander blocks provide data protection by employing RAID-type hardware support. With the RAID function, corresponding pages in multiple memory targets are addressed in unison. These pages may not be located at the same physical address within each of the memory targets but the translated (i.e., logical) address at which the controller has positioned the data are associated together so that they are programmed and read in unison. There is a one-to-one correspondence across the RAID codeword. The system uses the RAID functions to provide data back-up in that if a failure is experienced in one memory target, the data that was stored in that memory target can be recovered, for example, by using the corresponding page data and the associated RAID data (e.g., redundancy data) in the corresponding memory targets.

In order to recover the data from the memory target experiencing the failure, the failed target is identified. In one embodiment, the error detection is accomplished by memory

control circuitry receiving an error correction coding error that is not correctable after subsequent failed attempts to read the bad page.

Figure 5 illustrates a conceptual block diagram of a data programming operation utilizing the RAID functions of Figure 1. In this figure, each data register 0 – n (501 – 504) represents a different expander block communication channel coupled to a separate group of memory targets. Each group of memory targets 510 – 513 is associated with a different expander block communication channel.

RAID channel register 500 is part of the RAID channel that stores the RAID redundancy data. This RAID channel has its own associated group of memory targets 520 for storing the RAID redundancy data. Figure 5 shows that as the data to be programmed to a respective memory target in each of groups 510 – 513 is sequentially written into the expander registers 501 – 504, the RAID redundancy data is calculated and stored in RAID channel register 500. After all of the page data is received and moved to the expander registers 501 – 504, the same addresses in the corresponding memory targets are programmed in unison. The programming can be performed on only a subset of the memory targets and is not required to be performed on all of the memory targets of the system.

A typical error condition may not include an entire memory target experiencing the failure. A failure may simply be only a page or block of a location that cannot be read. When single pages or blocks are found to be unreadable, the target control circuitry can set and clear the notification to its associated expander. If an entire memory device/die/LUN is found to be unreadable, the error notification may be left on continuously when the failed location is being accessed.

The defective data is corrected by using the remaining good data, including the data from the RAID channels, and performing a bit-wise exclusive OR (XOR) operation of the same memory addresses in each memory target. Since the failed memory location is known, it is excluded from the XOR operation. A conceptual block diagram of data read and reconstruction operation is illustrated in Figure 6.

The data from each corresponding memory target in each of groups 510 – 513 can be substantially simultaneously written into the expander registers 501 – 504. The RAID redundancy data from the memory target in RAID group 520 is also written into the RAID channel register 500 at this time. As the data is received from the targets, it can be serially

read by the system controller. If it has been determined that the data cannot be read from a particular target and an associated status bit has been set in the control register 651, as the data is re-read it is reconstructed by an XOR operation 650 of the remaining good data, excluding the failed memory location. The XOR operation 650 is performed on the data from corresponding locations in each accessed memory target, including the corresponding memory target in RAID group 520.

For example, if a page of data starting at location 003Fh of one memory target in group 512 is found to have failed, the XOR operation is performed on the pages of data starting at location 003Fh of each of the other groups 510, 511, 513, and 520 coupled to the expander block. The XOR operation can be performed by data reconstruction logic 650 or a controller performing an XOR operation. The reconstructed data can then be passed to the system controller through the combination logic 630 – 633 with the other data.

The expander block associated with the failed memory device is notified of the failed read operation. In one embodiment, this is accomplished by the associated expander block receiving a command that specifically identifies the particular channel that is associated with the memory device that cannot be read. Once the associated expander block is informed of the failed channel, the memory control circuitry of the memory device that has failed can issue a re-read of the failed page. The data can then be read without regard to the fact that data for the failed memory device will be reconstructed by the expander block. There is no impact on performance when reading reconstructed data.

Figure 7 illustrates a flowchart of one embodiment of a method for data recovery in a solid state storage system, such as the system of Figure 1. The method can be executed by the solid state storage system controller, any of the individual memory device control circuitry, or any other controller. The method involves detection by a controller that the data stored in a particular memory target cannot be correctly read back out and has too many errors for correction by an error correction coding (ECC) scheme (e.g., Hamming code, BCH code, Turbo code, LDPC code, Reed-Muller code, Binary Golay code, Trellis Code Modulation).

The method begins by reading data into the appropriate controller 701. In one embodiment, a page of data is read into the controller. Alternate embodiments might read in other amounts of data. The controller then determines if the data is correct 703. This

determination can be performed by an ECC verification, a parity check or some other error checking scheme.

If the data does not contain errors, the method is done 704. If the data has errors, it is determined if the errors can be corrected by an ECC 705. If the number of errors is less than or equal to the ECC threshold, the errors are corrected 707. If the number of errors in the data is more than can be corrected by the ECC scheme, the controller and the associated expander block are informed that the data has been found to be unrecoverable 709. This can be accomplished by setting a bit in a control register, sending a command to the control circuitry of the memory target in which the defective data is stored, and/or sending an error indication to the system controller. The error indication can inform the controller that a particular cell is bad, a particular page is bad, a particular block is bad, or an entire memory device is bad.

When the data is unrecoverable by ECC, the defective data is reconstructed by performing a bit-wise XOR operation on the remaining good data and the RAID redundancy data 711 as described with reference to Figure 6. The location of the defective data can be used by the controller performing the operation to know which target's data stream to exclude from the XOR operation and to which page the regenerated data should be associated. The same addresses of each corresponding memory target coupled to a particular expander block are XOR'ed during the data regeneration process.

CONCLUSION

In summary, one or more embodiments of the present invention provide a method for recovering defective data in a memory system such as a solid state drive. Once the location of the defective data is known, good data can be XOR'ed with RAID data to reconstruct the defective data. The defective data is excluded from this logical combining operation.

Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement that is calculated to achieve the same purpose may be substituted for the specific embodiments shown. Many adaptations of the invention will be apparent to those of ordinary skill in the art. Accordingly, this application is intended to cover any adaptations or variations of the invention. It is manifestly intended that this invention be limited only by the following claims and equivalents thereof.

What is claimed is:

1. A method for data recovery in a memory system, the method comprising:
determining data to be defective at a location within a memory target in the memory system; and
logically combining data from a corresponding location in corresponding memory targets in the memory system to reconstruct the data determined to be defective wherein the memory target having the defective data is excluded from the logically combining.

2. The method of claim 1 wherein determining data to be defective at a location comprises determining that the memory target is experiencing a failure at the location.

3. The method of claim 2, wherein determining that the memory target is experiencing a failure at the location comprises detecting an error after reading data from the location.

4. The method of claim 1 wherein logically combining comprises performing an XOR operation on data read from the corresponding location in each of the corresponding memory targets in the memory system.

5. The method of claim 1 wherein logically combining comprises logically combining RAID data from the corresponding location in a memory target coupled to a RAID channel of the memory system.

6. The method of claim 1 and further comprising:
determining if data sensed from the corresponding location in the memory target determined to be defective is correctable; and
correcting the data sensed from the corresponding location in the memory target determined to be defective by error correction coding and not performing the bit-wise exclusive OR operation if the data is correctable.

7. The method of claim 6 wherein determining if data sensed from the corresponding location in the memory target determined to be defective is correctable comprises determining if a quantity of errors is greater than an error correction code threshold.

8. The method of claim 1 and further comprising:
programming data to the memory target prior to determining data to be defective; and
storing, in an associated RAID register, RAID data for the data being programmed in the memory target.

9. A memory system comprising:
a controller for controlling operation of the memory system, the controller having a plurality of communication channels; and
a plurality of memory targets coupled to the controller;
wherein each of the plurality of communication channels is coupled to a different subset of memory targets and wherein the memory system is configured to perform a data recovery operation for defective data read from a location in one of the memory targets by logically combining data read from corresponding locations in other ones of the memory targets, wherein the defective data is excluded from the logical combining.

10. The memory system of claim 9 wherein the system controller is configured to perform the data recovery operation.

11. The memory system of claim 9 wherein the memory system is a solid state drive.

12. The memory system of claim 9 and further including a plurality of expander blocks, each of the expander blocks coupling a respective one of the different subsets of memory targets to a respective one of the plurality of communication channels of the controller.

13. The memory system of claim 12 wherein each of the expander blocks has a plurality of expander block communication channels and each expander block communication channel is coupled to a respective at least one of the memory targets of the respective

subset of memory targets, wherein at least one of the expander block communication channels comprises a RAID channel coupled to a RAID register.

14. The memory system of claim 9 wherein each of the memory targets comprises a plurality of flash memory dies.

15. The memory system of claim 9 wherein each of the memory targets is configured to perform error detection on data read from that target.

16. The memory system of claim 9 and further comprising

a plurality of communication channels comprising a plurality of expander block communication channels, wherein the expander block communication channels include a RAID channel;

a plurality of data registers, each data register coupled to a different expander block communication channel of the plurality of expander block communication channels wherein each data register is configured to store page data;

a RAID register coupled to the RAID channel, wherein the RAID register is configured to store RAID data in response to the page data; and

wherein each subset of memory targets is coupled to each expander block communication channel wherein a corresponding memory location in a first memory target of each of the expander block communication channels is programmed with the page data from its respective data register and the corresponding memory location in the RAID register is programmed with RAID data corresponding to the page data.

17. The solid state drive of claim 16 wherein page data to be programmed to the first memory target in each group of memory targets is sequentially written into the plurality of data registers.

18. The solid state drive of claim 16 wherein the RAID register is configured to store RAID data that is calculated in response to the page data in each respective data register.

19. The solid state drive of claim 18 wherein the RAID data and the corresponding memory locations of the first memory targets of each of the expander channels are programmed substantially in unison.

20. The solid state drive of claim 16 wherein the corresponding memory locations of the first memory targets of only a subset of the expander block communication channels are programmed substantially in unison.

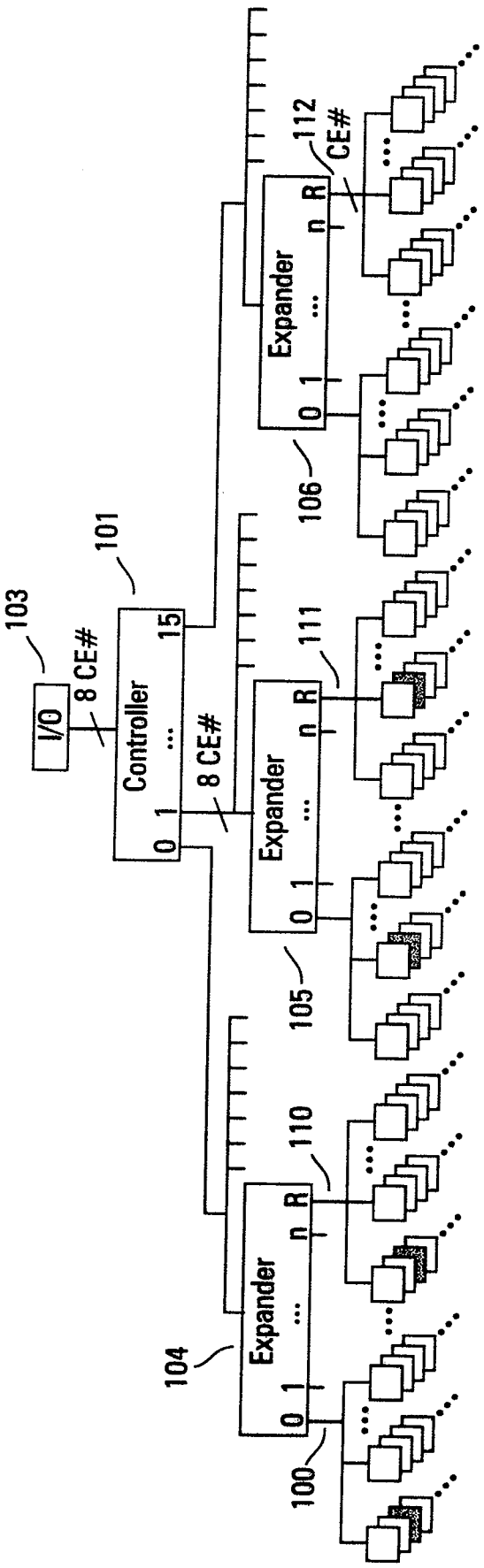

I/O
103
8 CE#
Controller
101
0 1 ... 15
8 CE#
Expander
0 1 ... n R
100
104
Expander
0 1 ... n R
105
110
Expander
0 1 ... n R
106
111
Expander
0 1 ... n R
112
CE#

FIG. 1

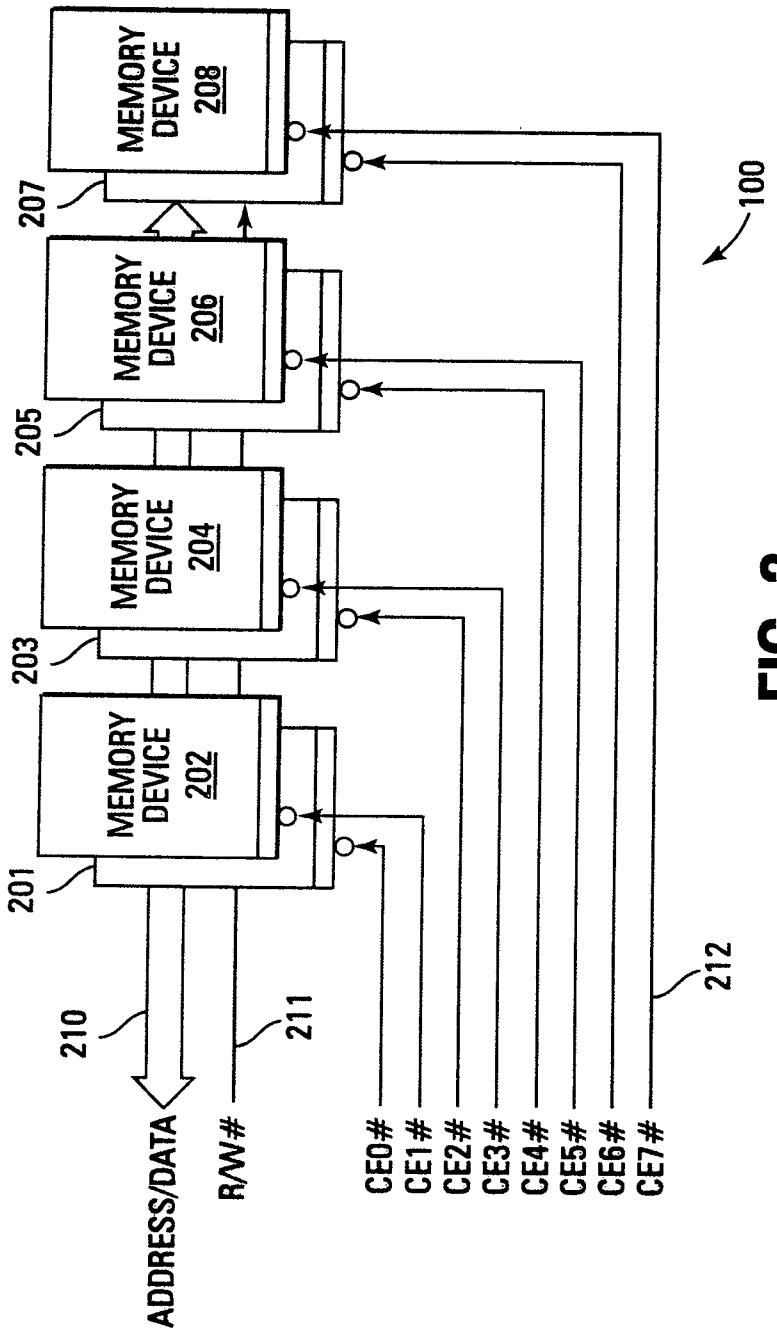

MEMORY DEVICE 208
MEMORY DEVICE 206
MEMORY DEVICE 204
MEMORY DEVICE 202
207
205
203
201
100
210
211
212
ADDRESS/DATA
R/W#
CE0#
CE1#
CE2#
CE3#
CE4#
CE5#
CE6#
CE7#

FIG. 2

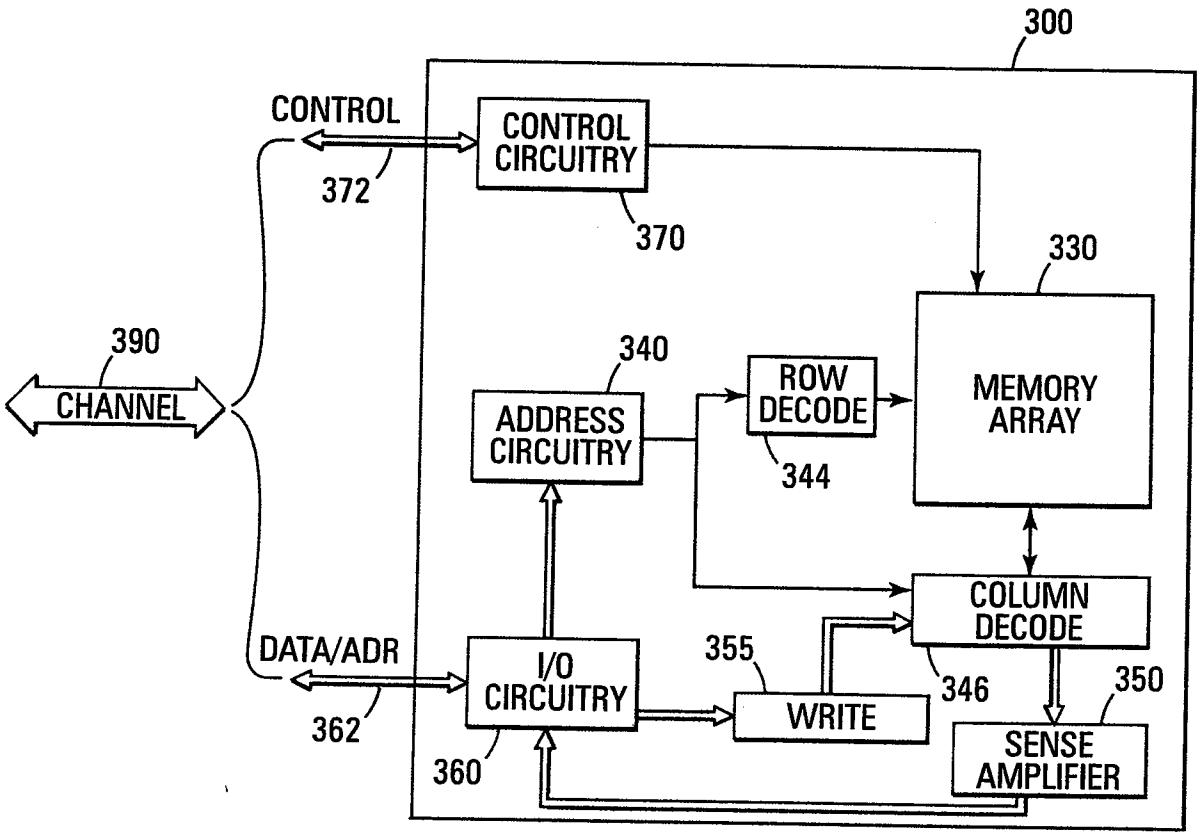

300
CONTROL
372
CONTROL CIRCUITRY
370
330
390
CHANNEL
340
ADDRESS CIRCUITRY
ROW DECODE
344
MEMORY ARRAY
COLUMN DECODE
DATA/ADR
362
I/O CIRCUITRY
360
355
WRITE
346
350
SENSE AMPLIFIER

FIG. 3

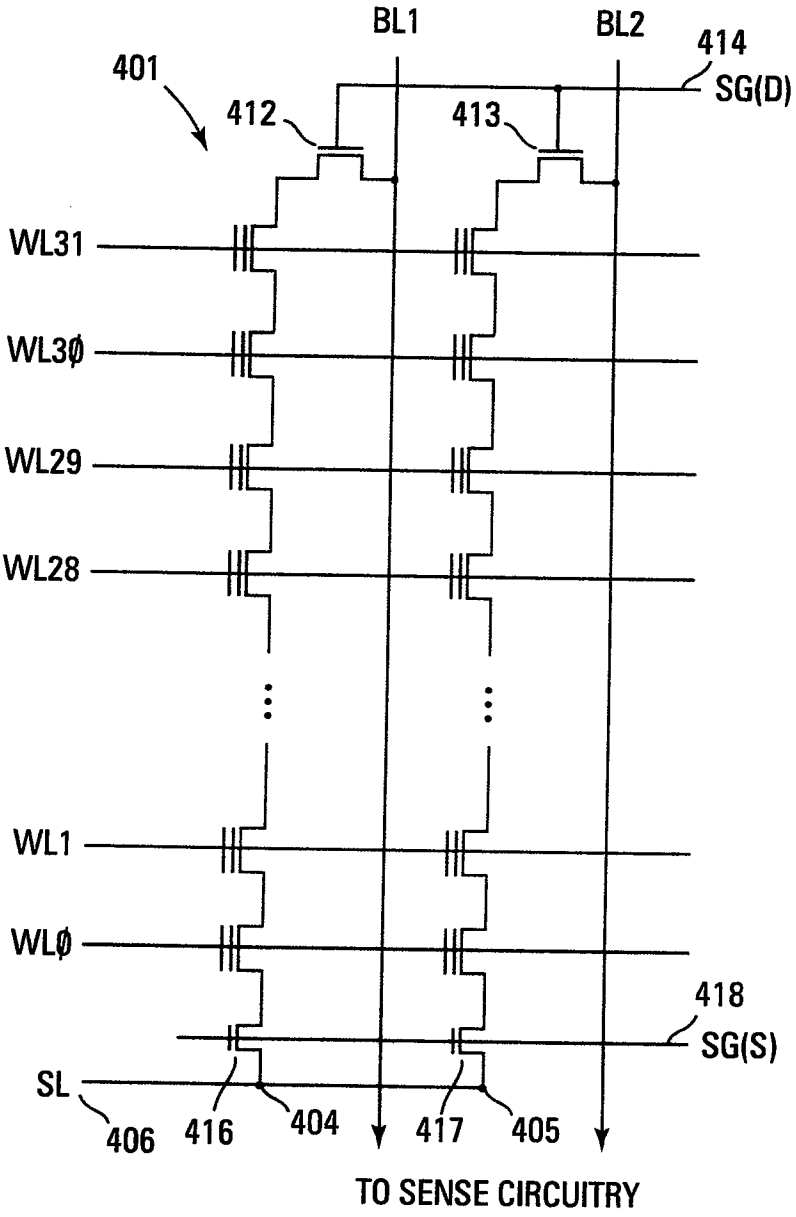

BL1
BL2
414
SG(D)
401
412
413
WL31
WL3Ø
WL29
WL28
WL1
WLØ
418
SG(S)
SL
406
416
404
417
405
TO SENSE CIRCUITRY

FIG. 4

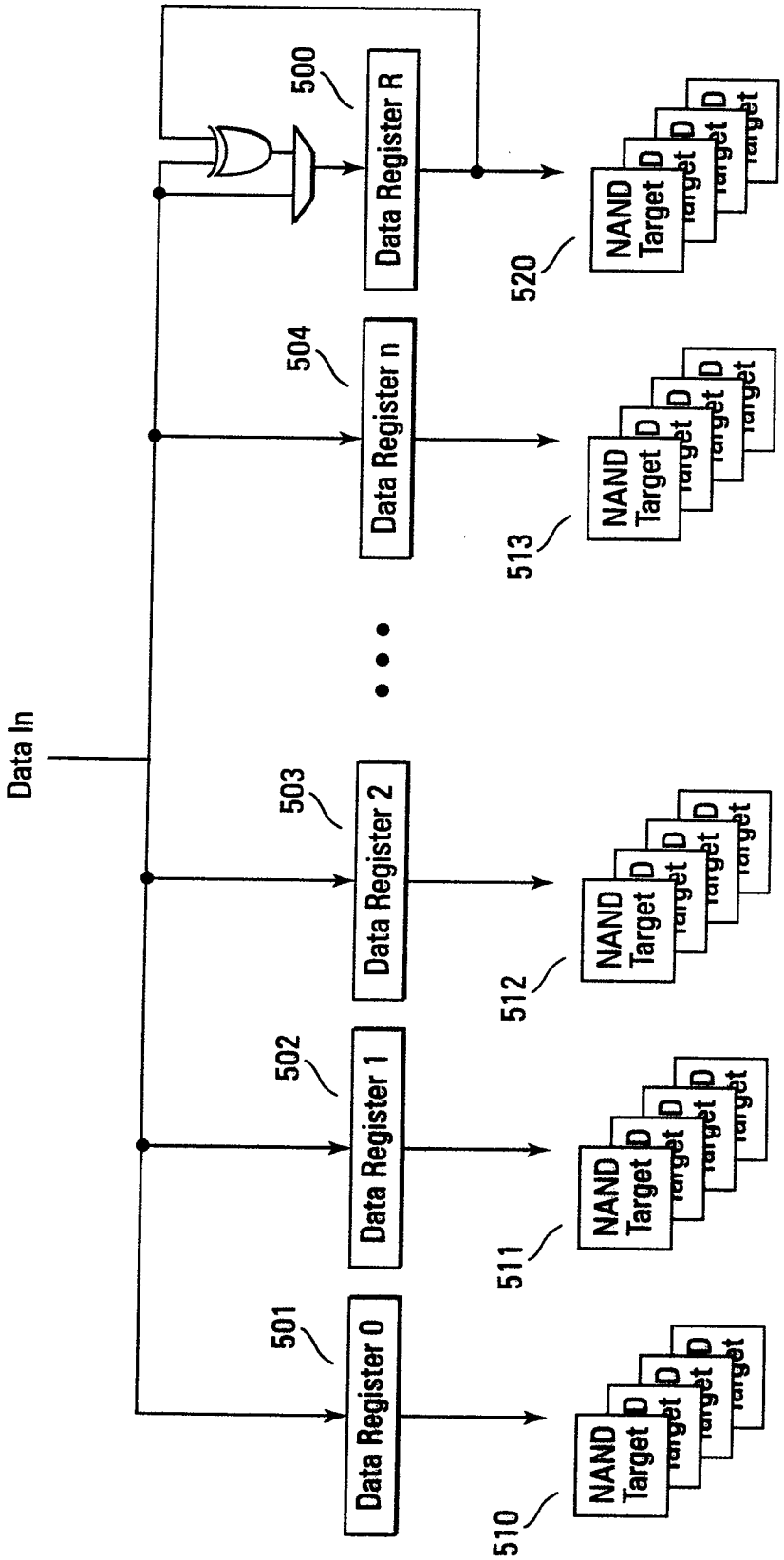

Data In
Data Register 0
Data Register 1
Data Register 2
Data Register n
Data Register R
501
502
503
504
500
NAND Target
510
511
512
513
520

FIG. 5

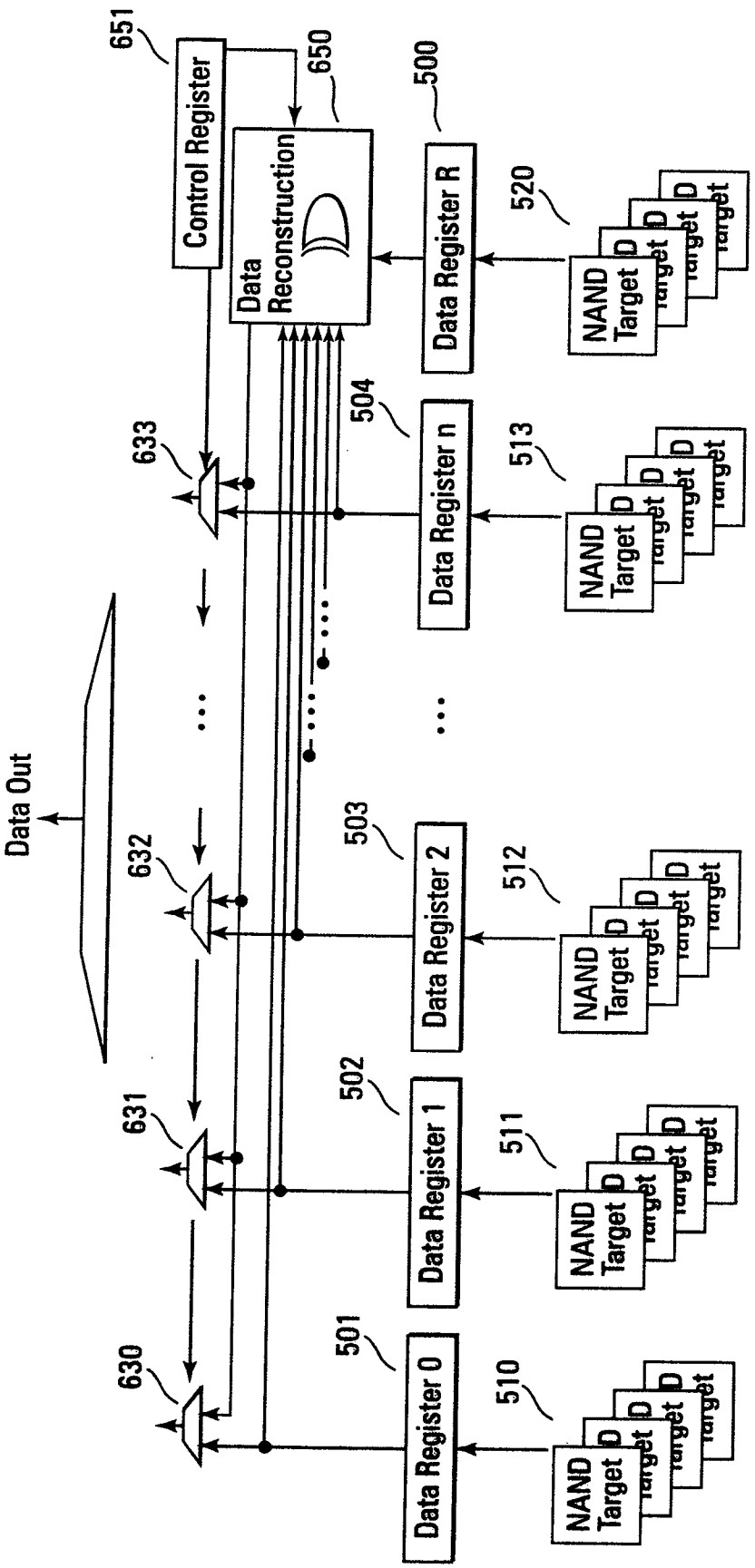

651
Control Register
650
Data Reconstruction
500
Data Register R
520
NAND Target
504
Data Register n
513
633
632
503
Data Register 2
512
631
502
Data Register 1
511
630
501
Data Register 0
510
Data Out

FIG. 6

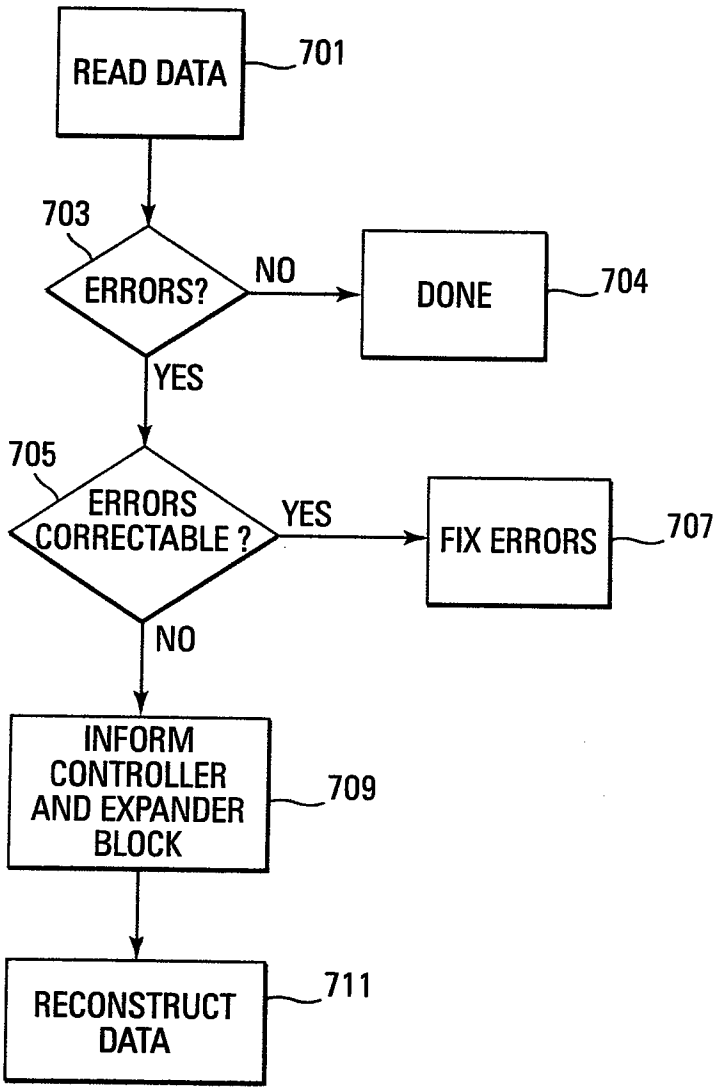

READ DATA
701
703
ERRORS?
NO
DONE
704
YES
705
ERRORS CORRECTABLE ?
YES
FIX ERRORS
707
NO
INFORM CONTROLLER AND EXPANDER BLOCK
709
RECONSTRUCT DATA
711

FIG. 7