20427 U.S. PTO
101305

113256 U.S. PTO
11/250238
101305

PTO/SB/05 (09-04)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office. U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No.	SNDK.156US2
First Inventor	Kevin M. Conley
Title	Partial Block Data Programming...
Express Mail Label No.	EV663653274US

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO:
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

1. ☑ **Fee Transmittal Form** (e.g., PTO/SB/17)
   (Submit an original and a duplicate for fee processing)
2. ☐ **Applicant claims small entity status.**
   See 37 CFR 1.27.
3. ☑ **Specification** [*Total Pages* 21]
   Both the claims and abstract must start on a new page
   (For information on the preferred arrangement, see MPEP 608.01(a))
4. ☑ **Drawing(s)** (*35 U.S.C. 113*) [*Total Sheets* 9]
5. **Oath or Declaration** [*Total Sheets* 1]
   - a. ☐ Newly executed (original or copy)
   - b. ☑ A copy from a prior application (37 CFR 1.63(d))
     (for continuation/divisional with Box 18 completed)
     - i. ☐ DELETION OF INVENTOR(S)
       Signed statement attached deleting inventor(s) name in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).
6. ☑ **Application Data Sheet.** See 37 CFR 1.76
7. ☐ **CD-ROM or CD-R** in duplicate, large table or Computer Program *(Appendix)*
   ☐ Landscape Table on CD
8. **Nucleotide and/or Amino Acid Sequence Submission**
   (if applicable, items a. – c. are required)
   - a. ☐ Computer Readable Form (CRF)
   - b. Specification Sequence Listing on:
     - i. ☐ CD-ROM or CD-R (2 copies); or
     - ii. ☐ Paper
   - c. ☐ Statements verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

9. ☐ **Assignment Papers** (cover sheet & document(s))
   Name of Assignee ____________________
10. ☐ **37 CFR 3.73(b) Statement** *(when there is an assignee)* ☐ **Power of Attorney**
11. ☐ **English Translation Document** *(if applicable)*
12. ☑ **Information Disclosure Statement** (PTO/SB/08 or PTO-1449)
    ☐ Copies of citations attached
13. ☐ **Preliminary Amendment**
14. ☑ **Return Receipt Postcard** (MPEP 503)
    (*Should be specifically itemized)*
15. ☐ **Certified Copy of Priority Document(s)**
    (if foreign priority is claimed)
16. ☐ **Nonpublication Request** under 35 U.S.C. 122(b)(2)(B)(i). Applicant must attach form PTO/SB/35 or equivalent.
17. ☐ Other: ____________________

18. If a CONTINUING APPLICATION, *check appropriate box, and supply the requisite information below and in the first sentence of the specification following the title, or in an Application Data Sheet under 37 CFR 1.76:*

☑ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No.: 10/841,388

Prior application information: Examiner Dinh, Ngoc V. *Art Unit:* 2187

19. CORRESPONDENCE ADDRESS

☑ The address associated with Customer Number: 36257 **OR** ☐ Correspondence address below

Name					
Address					
City		State		Zip Code	
Country		Telephone		Fax	

Signature	Gerald P. Parsons	Date	October 13, 2005
Name (Print/Type)	Gerald P. Parsons	Registration No. (Attorney/Agent)	24,486

This collection of information is required by 37 CFR 1.53(b). The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PTO/SB/17 (12-04v2)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number

Effective on 12/08/2004.
Fees pursuant to the Consolidated Appropriations Act, 2005 (H.R. 4818).

FEE TRANSMITTAL
For FY 2005

☐ Applicant claims small entity status. See 37 CFR 1.27

TOTAL AMOUNT OF PAYMENT ($) 1,000.00

Complete if Known	
Application Number	
Filing Date	
First Named Inventor	Kevin M. Conley
Examiner Name	
Art Unit	
Attorney Docket No.	SNDK.156US2

METHOD OF PAYMENT (check all that apply)

☐ Check ☐ Credit Card ☐ Money Order ☐ None ☐ Other (please identify):

☑ Deposit Account Deposit Account Number: 502664 Deposit Account Name: Parsons Hsue & de Runtz

For the above-identified deposit account, the Director is hereby authorized to: (check all that apply)

☑ Charge fee(s) indicated below
☐ Charge fee(s) indicated below, **except for the filing fee**
☑ Charge any additional fee(s) or underpayments of fee(s) under 37 CFR 1.16 and 1.17
☑ Credit any overpayments

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

FEE CALCULATION

1. BASIC FILING, SEARCH, AND EXAMINATION FEES

	FILING FEES		SEARCH FEES		EXAMINATION FEES		
Application Type	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fees Paid ($)**
Utility	300	150	500	250	200	100	1,000.00
Design	200	100	100	50	130	65	
Plant	200	100	300	150	160	80	
Reissue	300	150	500	250	600	300	
Provisional	200	100	0	0	0	0	

2. EXCESS CLAIM FEES

Fee Description	**Fee ($)**	**Small Entity Fee ($)**
Each claim over 20 (including Reissues)	50	25
Each independent claim over 3 (including Reissues)	200	100
Multiple dependent claims	360	180

Total Claims		**Extra Claims**		**Fee ($)**		**Fee Paid ($)**
3	- 20 or HP =	0	x	50	=	0.00

HP = highest number of total claims paid for, if greater than 20.

Indep. Claims		**Extra Claims**		**Fee ($)**		**Fee Paid ($)**
2	- 3 or HP =	0	x	200	=	0.00

HP = highest number of independent claims paid for, if greater than 3.

Multiple Dependent Claims

Fee ($)	**Fee Paid ($)**
360	0.00

3. APPLICATION SIZE FEE

If the specification and drawings exceed 100 sheets of paper (excluding electronically filed sequence or computer listings under 37 CFR 1.52(e)), the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).

Total Sheets		**Extra Sheets**		**Number of each additional 50 or fraction thereof**		**Fee ($)**		**Fee Paid ($)**
30	- 100 =	0	/ 50 =	0	(round **up** to a whole number) x	250	=	0.00

4. OTHER FEE(S)

	Fees Paid ($)
Non-English Specification, $130 fee (no small entity discount)	0.00
Other (e.g., late filing surcharge):	0.00

SUBMITTED BY

Signature		Registration No. (Attorney/Agent) 24,486	Telephone 415-318-1160
Name (Print/Type)	Gerald P. Parsons		Date October 13, 2005

This collection of information is required by 37 CFR 1.136. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

20427 U.S. PTO
101305

113256 U.S. PTO
11/250238
101305

PTO/SB/05 (09-04)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office. U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No.	SNDK.156US2
First Inventor	Kevin M. Conley
Title	Partial Block Data Programming...
Express Mail Label No.	EV663653274US

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO:
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

1. ☑ **Fee Transmittal Form** (e.g., PTO/SB/17)
   (Submit an original and a duplicate for fee processing)
2. ☐ **Applicant claims small entity status.**
   See 37 CFR 1.27.
3. ☑ **Specification** [*Total Pages* 21]
   Both the claims and abstract must start on a new page
   (For information on the preferred arrangement, see MPEP 608.01(a))
4. ☑ **Drawing(s)** (*35 U.S.C. 113*) [*Total Sheets* 9]
5. **Oath or Declaration** [*Total Sheets* 1]
   a. ☐ Newly executed (original or copy)
   b. ☑ A copy from a prior application (37 CFR 1.63(d))
   (for continuation/divisional with Box 18 completed)
      i. ☐ DELETION OF INVENTOR(S)
      Signed statement attached deleting inventor(s) name in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).
6. ☑ **Application Data Sheet.** See 37 CFR 1.76
7. ☐ **CD-ROM or CD-R** in duplicate, large table or Computer Program *(Appendix)*
   ☐ Landscape Table on CD
8. **Nucleotide and/or Amino Acid Sequence Submission**
   (if applicable, items a. – c. are required)
   a. ☐ Computer Readable Form (CRF)
   b. Specification Sequence Listing on:
      i. ☐ CD-ROM or CD-R (2 copies); or
      ii. ☐ Paper
   c. ☐ Statements verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

9. ☐ **Assignment Papers** (cover sheet & document(s))
   Name of Assignee__________
10. ☐ **37 CFR 3.73(b) Statement** *(when there is an assignee)* ☐ **Power of Attorney**
11. ☐ **English Translation Document** *(if applicable)*
12. ☑ **Information Disclosure Statement** (PTO/SB/08 or PTO-1449)
    ☐ Copies of citations attached
13. ☐ **Preliminary Amendment**
14. ☑ **Return Receipt Postcard** (MPEP 503)
    (*Should be specifically itemized)*
15. ☐ **Certified Copy of Priority Document(s)**
    (if foreign priority is claimed)
16. ☐ **Nonpublication Request** under 35 U.S.C. 122(b)(2)(B)(i). Applicant must attach form PTO/SB/35 or equivalent.
17. ☐ Other:__________

18. If a CONTINUING APPLICATION, *check appropriate box, and supply the requisite information below and in the first sentence of the specification following the title, or in an Application Data Sheet under 37 CFR 1.76:*

☑ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No.: 10/841,388

Prior application information: Examiner Dinh, Ngoc V. Art Unit: 2187

19. CORRESPONDENCE ADDRESS

☑ The address associated with Customer Number: 36257 **OR** ☐ Correspondence address below

Name					
Address					
City		State		Zip Code	
Country		Telephone		Fax	

Signature		Date	October 13, 2005
Name (Print/Type)	Gerald P. Parsons	Registration No. (Attorney/Agent)	24,486

This collection of information is required by 37 CFR 1.53(b). The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PTO/SB/17 (12-04v2)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number

Effective on 12/08/2004.
Fees pursuant to the Consolidated Appropriations Act, 2005 (H.R. 4818).

FEE TRANSMITTAL
For FY 2005

☐ Applicant claims small entity status. See 37 CFR 1.27

TOTAL AMOUNT OF PAYMENT ($) 1,000.00

Complete if Known	
Application Number	
Filing Date	
First Named Inventor	Kevin M. Conley
Examiner Name	
Art Unit	
Attorney Docket No.	SNDK.156US2

METHOD OF PAYMENT (check all that apply)

☐ Check ☐ Credit Card ☐ Money Order ☐ None ☐ Other (please identify): ____________

☑ Deposit Account Deposit Account Number: 502664 Deposit Account Name: Parsons Hsue & de Runtz

For the above-identified deposit account, the Director is hereby authorized to: (check all that apply)

☑ Charge fee(s) indicated below ☐ Charge fee(s) indicated below, **except for the filing fee**

☑ Charge any additional fee(s) or underpayments of fee(s) under 37 CFR 1.16 and 1.17 ☑ Credit any overpayments

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

FEE CALCULATION

1. BASIC FILING, SEARCH, AND EXAMINATION FEES

	FILING FEES		SEARCH FEES		EXAMINATION FEES		
Application Type	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fees Paid ($)**
Utility	300	150	500	250	200	100	1,000.00
Design	200	100	100	50	130	65	
Plant	200	100	300	150	160	80	
Reissue	300	150	500	250	600	300	
Provisional	200	100	0	0	0	0	

2. EXCESS CLAIM FEES

Fee Description	**Fee ($)**	**Small Entity Fee ($)**
Each claim over 20 (including Reissues)	50	25
Each independent claim over 3 (including Reissues)	200	100
Multiple dependent claims	360	180

Total Claims		**Extra Claims**		**Fee ($)**		**Fee Paid ($)**
3	- 20 or HP =	0	x	50	=	0.00

HP = highest number of total claims paid for, if greater than 20.

Indep. Claims		**Extra Claims**		**Fee ($)**		**Fee Paid ($)**
2	- 3 or HP =	0	x	200	=	0.00

HP = highest number of independent claims paid for, if greater than 3.

Multiple Dependent Claims

Fee ($)	**Fee Paid ($)**
360	0.00

3. APPLICATION SIZE FEE

If the specification and drawings exceed 100 sheets of paper (excluding electronically filed sequence or computer listings under 37 CFR 1.52(e)), the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).

Total Sheets		**Extra Sheets**		**Number of each additional 50 or fraction thereof**			**Fee ($)**		**Fee Paid ($)**
30	- 100 =	0	/ 50 =	0	(round **up** to a whole number)	x	250	=	0.00

4. OTHER FEE(S)

	Fees Paid ($)
Non-English Specification, $130 fee (no small entity discount)	0.00
Other (e.g., late filing surcharge):	0.00

SUBMITTED BY

Signature	P.	Registration No. (Attorney/Agent) 24,486	Telephone 415-318-1160
Name (Print/Type)	Gerald P. Parsons		Date October 13, 2005

This collection of information is required by 37 CFR 1.136. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

Inventor: Kevin M. Conley

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application is a continuation of application serial no. 10/841,388, filed May 7, 2004, which in turn is a continuation of application serial no. 09/766,436, filed January 19, 2001, now patent no. 6,763,424, which applications are incorporated herein in their entirety by this reference.

BACKGROUND OF THE INVENTION

[0002] This invention pertains to the field of semiconductor non-volatile data storage system architectures and their methods of operation, and has application to data storage systems based on flash electrically erasable and programmable read-only memories (EEPROMs).

[0003] A common application of flash EEPROM devices is as a mass data storage subsystem for electronic devices. Such subsystems are commonly implemented as either removable memory cards that can be inserted into multiple host systems or as non-removable embedded storage within the host system. In both implementations, the subsystem includes one or more flash devices and often a subsystem controller.

[0004] Flash EEPROM devices are composed of one or more arrays of transistor cells, each cell capable of non-volatile storage of one or more bits of data. Thus flash memory does not require power to retain the data programmed therein. Once programmed however, a cell must be erased before it can be reprogrammed with a new data value. These arrays of cells are partitioned into groups to provide for efficient implementation of read, program and erase functions. A typical flash memory architecture for mass storage arranges large groups of cells into erasable blocks, wherein a block contains the smallest number of cells (unit of erase) that are erasable at one time.

[0005] In one commercial form, each block contains enough cells to store one sector of user data plus some overhead data related to the user data and/or to the block in which it is stored. The amount of user data included in a sector is the standard 512 bytes in one class of such memory systems but can be of some other size. Because the isolation of individual blocks of cells from one another that is required to make them individually erasable takes space on the integrated circuit chip, another class of flash memories makes the blocks significantly larger so there is less space required for such isolation. But since it is also desired to handle user data in much smaller sectors, each large block is often further partitioned into individually addressable pages that are the basic unit for reading and programming user data (unit of programming and/or reading). Each page usually stores one sector of user data, but a page may store a partial sector or multiple sectors. A "sector" is used herein to refer to an amount of user data that is transferred to and from the host as a unit.

[0006] The subsystem controller in a large block system performs a number of functions including the translation between logical addresses (LBAs) received by the memory sub-system from a host, and physical block numbers (PBNs) and page addresses within the memory cell array. This translation often involves use of intermediate terms for a logical block number (LBN) and logical page. The controller also manages the low level flash circuit operation through a series of commands that it issues to the flash memory devices via an interface bus. Another function the controller performs is to maintain the integrity of data stored to the subsystem through various means, such as by using an error correction code (ECC).

[0007] In an ideal case, the data in all the pages of a block are usually updated together by writing the updated data to the pages within an unassigned, erased block, and a logical-to-physical block number table is updated with the new address The original block is then available to be erased. However, it is more typical that the data stored in a number of pages less than all of the pages within a given block must be updated. The data stored in the remaining pages of the given block remains unchanged. The probability of this occurring is higher in systems where the number of sectors of data stored per block is higher. One technique now used to accomplish such a partial block update is to write the data of the pages to be updated into a corresponding number of the pages of an unused erased block and then copy the unchanged pages from the original block into pages of the new block. The original block may then be erased and added to an inventory of unused blocks in which data may later be programmed. Another technique similarly writes the updated pages to a new block but eliminates the need to copy the other pages of data into the new block by changing the flags of the pages in the original block which are being updated to indicate they contain obsolete data. Then when the data are read, the updated data read from pages of the new block are combined with the unchanged data read from pages of the original block that are not flagged as obsolete.

SUMMARY OF THE INVENTION

[0008] According to one principal aspect of the present invention, briefly and generally, both the copying of unchanged data from the original to the new blocks and the need to update flags within the original block are avoided when the data of fewer than all of the pages within a block are being updated. This is accomplished by maintaining both the superceded data pages and the updated pages of data with a common logical address. The original and updated pages of data are then distinguished by the relative order in which they were programmed. During reading, the most recent data stored in the pages having the same logical address are combined with the unchanged pages of data while data in the original versions of the updated pages are ignored. The updated data can be written to either pages within a different block than the original data, or to available unused pages within the same block. In one specific implementation, a form of time stamp is stored with each page of data that allows determining the relative order that pages with the same logical address were written. In another specific implementation, in a system where pages are programmed in a particular order within the blocks, a form of time stamp is stored with each block of data, and the most recent copy of a page within a block is established by its physical location within the block.

[0009] These techniques avoid both the necessity for copying unchanged data from the original to new block and the need to change a flag or other data in the pages of the original block whose data have been updated. By not having to change a flag or other data in the superceded pages, a potential of disturbing the previously written data in adjacent pages of that same block that can occur from such a writing operation is eliminated. Also, a performance penalty of the additional program operation is avoided.

[0010] A further operational feature, which may be used in conjunction with the above summarized techniques, keeps track of the logical offset of individual pages of data within the individual memory cell blocks, so that the updated data need not be stored with the same physical page offset as the superceded data. This allows more efficient use of the pages of new blocks, and even allows the updated data to be stored in any erased pages of the same block as the superceded data.

[0011] Another principal aspect of the present invention groups together two or more blocks positioned in separate units of the memory array (also termed "sub-arrays") for programming and reading together as part of a single operation. Such a multiple block group is referenced herein as a "metablock." Its component blocks may be either all located on a single memory integrated circuit chip, or, in systems using more than one such chip, located on two or more different chips. When data in fewer than all of the pages of one of these blocks is updated, the use of another block in that same unit is normally required. Indeed, the techniques described above, or others, may be employed separately with each block of the metablock. Therefore, when data within pages of more than one block of the metablock are updated, pages within more than one additional block are required to be used. If there are four blocks of four different memory units that form the metablock, for example, there is some probability that up to an additional four blocks, one in each of the units, will be used to store updated pages of the original blocks. One update block is potentially required in each unit for each block of the original metablock. In addition, according to the present invention, updated data from pages of more than one of the blocks in the metablock can be stored in pages of a common block in only one of the units. This significantly reduces the number of unused erased blocks that are needed to store updated data, thereby making more efficient use of the available memory cell blocks to store data. This technique is particularly useful when the memory system frequently updates single pages from a metablock.

[0012] Additional aspects, features and advantages of the present invention are included in the following description of exemplary embodiments, which description should be read in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] Figure 1 is a block diagram of a typical prior art flash EEPROM memory array with memory control logic, data and address registers;

[0014] Figure 2 illustrates an architecture utilizing memories of Figure 1 with a system controller;

[0015] Figure 3 is a timing diagram showing a typical copy operation of the memory system of Figure 2;

[0016] Figure 4 illustrates an existing process of updating data in less than all of the pages of a multi-paged block;

[0017] Figures 5A and 5B are tables of corresponding logical and physical block addresses for each of the original and new blocks of Figure 4, respectively;

[0018] Figure 6 illustrates another existing process of updating data in less than all of the pages of a multi-paged block;

[0019] Figures 7A and 7B are tables of corresponding logical and physical page addresses for the original and new blocks of Figure 6, respectively;

[0020] Figure 8 illustrates an example of an improved process of updating data in less than all of the pages of a multi-paged block;

[0021] Figure 9 is a table of corresponding logical and physical page numbers for the new block of Figure 8;

[0022] Figure 10 provides an example of a layout of the data in a page shown in Figure 8;

[0023] Figure 11 illustrates a further development of the example of Figure 8;

[0024] Figure 12 is a table of corresponding logical and physical page numbers for the new block of Figure 11;

[0025] Figure 13 illustrates one way to read the updated data in the blocks of Figure 11;

[0026] Figure 14 is a flow diagram of a process of programming data into a memory system organized as illustrated in Figures 8 and 9;

[0027] Figure 15 illustrates an existing multi-unit memory with blocks from the individual units being linked together into a metablock and

[0028] Figure 16 illustrates an improved method of updating data of a metablock in the multi-unit memory of Figure 12 when the amount of updated data is much less that the data storage capacity of the metablock.

DESCRIPTION OF EXISTING LARGE BLOCK MANAGEMENT TECHNIQUES

[0029] Figure 1 shows a typical flash memory device internal architecture. The primary features include an input/output (I/O) bus 411 and control signals 412 to interface to an external controller, a memory control circuit 450 to control internal memory operations with registers for command, address and status signals. One or more arrays 400 of flash EEPROM cells are included, each array having its own row decoder (XDEC) 401 and column decoder (YDEC) 402, a group of sense amplifiers and program control circuitry (SA/PROG) 454 and a data register 404. Presently, the memory cells usually include one or more conductive floating gates as storage elements but other long term electron charge storage elements may be used instead. The memory cell array may be operated with two levels of charge defined for each storage element to therefore store one bit of data with each element. Alternatively, more than two storage states may be defined for each storage element, in which case more than one bit of data is stored in each element.

[0030] If desired, a plurality of arrays 400, together with related X decoders, Y decoders, program/verified circuitry, data registers, and the like are provided, for example as taught by U.S. Patent 5,890,192, issued March 30, 1999, and assigned to Sandisk Corporation, the assignee of this application, which is hereby incorporated by this reference. Related memory system features are described in co-pending patent application serial no. 09/505,555, filed February 17, 2000 by Kevin Conley et al., which application is expressly incorporated herein by this reference.

[0031] The external interface I/O bus 411 and control signals 412 can include the following:

CS - Chip Select.	Used to activate flash memory interface.
RS - Read Strobe.	Used to indicate the I/O bus is being used to transfer data from the memory array.
WS - Write Strobe.	Used to indicate the I/O bus is being used to transfer data to the memory array.
AS - Address Strobe.	Indicates that the I/O bus is being used to transfer address information.
AD[7:0] - Address/Data Bus	This I/O bus is used to transfer data between controller and the flash memory command, address and data registers of the memory control 450.

[0032] This interface is given only as an example as other signal configurations can be used to give the same functionality. Figure 1 shows only one flash memory array 400 with its related components, but a multiplicity of such arrays can exist on a single flash memory chip that share a common interface and memory control circuitry but have separate XDEC, YDEC, SA/PROG and DATA REG circuitry in order to allow parallel read and program operations.

[0033] Data is transferred from the memory array through the data register 404 to an external controller via the data registers' coupling to the I/O bus AD[7:0] 411. The data register 404 is also coupled the sense amplifier/programming circuit 454. The number of elements of the data register coupled to each sense amplifier/programming circuit element may depend on the number of bits stored in each storage element of the memory cells, flash EEPROM cells each containing one or more floating gates as the storage elements. Each storage element may store a plurality of bits, such as 2 or 4, if the memory cells are operated in a multi-state mode. Alternatively, the memory cells may be operated in a binary mode to store one bit of data per storage element.

[0034] The row decoder 401 decodes row addresses for the array 400 in order to select the physical page to be accessed. The row decoder 401 receives row addresses via internal row address lines 419 from the memory control logic 450. A column decoder 402 receives column addresses via internal column address lines 429 from the memory control logic 450.

[0035] Figure 2 shows an architecture of a typical non-volatile data storage system, in this case employing flash memory cells as the storage media. In one form, this system is encapsulated within a removable card having an electrical connector extending along one side to provide the host interface when inserted into a receptacle of a host. Alternatively, the system of Figure 2 may be embedded into a host system in the form of a permanently installed embedded circuit or otherwise. The system utilizes a single controller 301 that performs high level host and memory control functions. The flash memory media is composed of one or more flash memory devices, each such device often formed on its own integrated circuit chip. The system controller and the flash memory are connected by a bus 302 that allows the controller 301 to load command, address, and transfer data to and from the flash memory array. The controller 301 interfaces with a host system (not shown) with which user data is transferred to and from the flash memory array. In the case where the system of Figure 2 is included in a card, the host interface includes a mating plug and socket assembly (not shown) on the card and host equipment.

[0036] The controller 301 receives a command from the host to read or write one or more sectors of user data starting at a particular logical address. This address may or may not align with a boundary of a physical block of memory cells.

[0037] In some prior art systems having large capacity memory cell blocks that are divided into multiple pages, as discussed above, the data from a block that is not being updated needs to be copied from the original block to a new block that also contains the new, updated data being written by the host. This technique is illustrated in Figure 4, wherein two of a large number of blocks of memory are included. One block 11 (PBN0) is illustrated to be divided into 8 pages for storing one sector of user data in each of its pages. Overhead data fields contained within each page include a field 13 containing the LBN of the block 11. The order of the logical pages within a logical block is fixed with respect to the corresponding physical pages within a physical block. A second similarly configured block 15 (PBN1) is selected from an inventory of unused, erased blocks. Data within pages 3-5 of the original block 11 are being updated by three pages of new data 17. The new data is written into the corresponding pages 3-5 of the new block 15, and user data from pages 0-2, 6 and 7 of the block 11 are copied into corresponding pages of the new block 15. All pages of the new block 15 are preferably programmed in a single sequence of programming operations. After the block 15 is programmed, the original block 11 can be erased and placed in inventory for later use. The copying of data between the blocks 11 and 15, which involves reading the data from one or more pages in the original block and subsequently programming the same data to pages in a newly assigned block, greatly reduces the write performance and usable lifetime of the storage system.

[0038] With reference to Figures 5A and 5B, partial tables show mapping of the logical blocks into the original and new physical blocks 11 and 15 before (Figure 5A) and after (Figure 5B) the updating of data described with respect to Figure 4. Before the data update, the original block 11, in this example, stores pages 0-7 of LBN0 into corresponding pages 0-7 of PBN0. After the data update, the new block 15 stores pages 0-7 of LBN0 in corresponding pages 0-7 of PBN1. Receipt of a request to read data from LBN0 is then directed to the physical block 15 instead of the physical block 11. In a typical controller operation, a table in the form of that shown in Figures 5A and 5B is built from the LBN field 13 read from a physical page and knowledge of the PBN that is addressed when reading the data field 13. The table is usually stored in a volatile memory of the controller for ease of access, although only a portion of a complete table for the entire system is typically stored at any one time. A portion of the table is usually formed immediately in advance of a read or programming operation that involves the blocks included in the table portion.

[0039] In other prior art systems, flags are recorded with the user data in pages and are used to indicate that pages of data in the original block that are being superceded by the newly written data are invalid. Only the new data is written to the newly assigned block. Thus the data in pages of the block not involved in the write operation but contained in the same physical block as the superceded data need not be copied into the new block. This operation is illustrated in Figure 6, where pages 3-5 of data within an original block 21 (PBN0) are again being updated. Updated pages 3-5 of data 23 are written into corresponding pages of a new block 25. As part of the same operation, an old/new flag 27 is written in each of the pages 3-5 to indicate the data of those pages is old, while the flag 27 for the remaining pages 0-2, 6 and 7 remains set at "new". Similarly, the new PBN1 is written into another overhead data field of each of the pages 3-5 in the block 21 to indicate where the updated data are located. The LBN and page are stored in a field 31 within each of the physical pages.

[0040] Figures 7A and 7B are tables of the correspondence between the data LBN/page and the PBN/page before (Figure 7A) and after (Figure 7B) the data update is complete. The unchanged pages 0-2, 6 and 7 of the LBN remain mapped into PBN0 while the updated pages 3-5 are shown to reside in PBN1. The table of Figure 7B is built by the memory controller by reading the overhead data fields 27, 29 and 31 of the pages within the block PBN0 after the data update. Since the flag 27 is set to "old" in each of pages 3-5 of the original block PBN0, that block will no longer appear in the table for those pages. Rather, the new block number PBN1 appears instead, having been read from the overhead fields 29' of the updated pages. When data are being read from LBN0, the user data stored in the pages listed in the right column of Figure 7B are read and then assembled in the order shown for transfer to the host.

[0041] Various flags are typically located in the same physical page as the other associated overhead data, such as the LBN and an ECC. Thus, to program the old/new flags 27, and others, in pages where the data has been superceded requires that a page support multiple programming cycles. That is, the memory array must have the capability that its pages can be programmed in at least at least two stages between erasures. Furthermore, the block must support the ability to program a page when other pages in the block with higher offsets or addresses have been already programmed. A limitation of some flash memories however prevents the usage of such flags by specifying that the pages in a block can only be programmed in a physically sequential manner. Furthermore, the pages support a finite number of program cycles and in some cases additional programming of programmed pages is not permitted.

[0042] What is needed is a mechanism by which data that partially supercedes data stored in an existing block can be written without either copying unchanged data from the existing block or programming flags to pages that have been previously programmed.

DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE INVENTION

[0043] There are many different types of flash EEPROM, each of which presents its own limitations that must be worked around to operate a high performance memory system formed on a small amount of integrated circuit area. Some do not provide for writing any data into a page that has already been programmed, so updating flags in a page that contains superceded data, as described above, is not possible. Others allow such flags to be written but doing so in pages whose data is being superceded can disturb data in other pages of the same block that remain current.

[0044] An example memory system where this has been found to be a problem is a NAND type, where a column of memory cells is formed as a series circuit string between a bit line and a common potential. Each word line extends across a row of memory cells formed of one cell in each such string. Such a memory is particularly susceptible to such memory state disturbs when being operated in a multi-state mode to store more than one bit of data in each such cell. Such operation divides an available window of a memory cell transistor threshold voltage range into narrow non-overlapping voltage level ranges, each range becoming narrower as the number of levels, and thus the number of bits being stored in each cell, are increased. For example, if four threshold ranges are used, two bits of data are stored in each cell's storage element. And since each of the four threshold voltage ranges is necessarily small, the chance of the state of a cell being disturbed by programming other cells in the same block is increased with multi-state operation. In this case, the writing of the old/new or other flags, as described with respect to Figures 6, 7A and 7B, cannot be tolerated.

[0045] A common feature of each of the existing memory management techniques described above with respect to Figures 4-7B is that a logical block number (LBN) and page offset is mapped within the system to at most two physical block numbers (PBNs). One block is the original block and the other contains the updated page data. Data are written to the page location in the block corresponding to the low order bits of its logical address (LBA). This mapping is typical in various types of memory systems. In the techniques described below, pages containing updated data are also assigned the same LBN and page offsets as the pages whose data has been superceded. But rather than tagging the pages containing original data as being superceded, the memory controller distinguishes the pages containing the superceded data from those containing the new, updated version either (1) by keeping track of the order in which the pages having the same logical addresses were written, such as by use of a counter, and/or (2) from the physical page addresses wherein, when pages are written in order within blocks from the lowest page address to the highest, the higher physical address contains the most recent copy of the data. When the data is accessed for reading, therefore, those in the most current pages are used in cases where there are pages containing superceded data that have the same logical addresses, while the superceded data are ignored.

[0046] A first specific implementation of this technique is described with respect to Figures 8 and 9. The situation is the same in this example as that in the prior art techniques described with respect to Figures 4-7B, namely the partial re-write of data within a block 35, although each block is now shown to contain 16 pages. New data 37 for each of the pages 3-5 of the block 35 (PBN 35) is written into three pages of a new block 39 (PBN1) that has previously been erased, similar to that described previously. A LBN and page offset overhead data field 41 written into the pages of PBN1 that contain the updated data is the same as that in the pages of the superceded data in the initial block PBN0. The table of Figure 9, formed from the data within the fields 41 and 41', shows this. The logical LBN and page offsets, in the first column, are mapped into both the first physical block (PBN0), in the second column, and, for the pages that have been updated, also into the second physical block (PBN1) in the third column. The LBN and logical page offsets 41' written into each of the three pages of updated data within the new block PBN1 are the same as those 41 written into each of a corresponding logical page of the original block PBN0.

[0047] In order to determine which of two pages having the same LBN and page offset contains the updated data, each page contains another overhead field 43 that provides an indication of its time of programming, at least relative to the time that other pages with the same logical address are programmed. This allows the controller to determine, when reading the data from the memory, the relative ages of the pages of data that are assigned the same logical address.

[0048] There are several ways in which the field 43, which contains a form of time stamp, may be written. The most straight forward way is to record in that field, when the data of its associated page is programmed, the output of a real-time clock in the system. Later programmed pages with the same logical address then have a later time recorded in the field 43. But when such a real-time clock is not available in the system, other techniques can be used. One specific technique is to store the output of a modulo-N counter as the value of the field 43. The range of the counter should be one more than the number of pages that are contemplated to be stored with the same logical page number. When updating the data of a particular page in the original block PBN0, for example, the controller first reads the count stored in the field 43 of the page whose data are being updated, increments the count by some amount, such as one, and then writes that incremented count in the new block PBN1 as the field 43'. The counter, upon reaching a count of N+1, rolls over to 0. Since the number of blocks with the same LBN is less than N, there is always a point of discontinuity in the values of stored counts. It is easy then to handle the rollover with normalized to the point of discontinuity.

[0049] The controller, when called upon to read the data, easily distinguishes between the new and superceded pages' data by comparing the counts in the fields 43 and 43' of pages having the same LBA and page offset. In response to a need to read the most recent version of a data file, data from the identified new pages are then assembled, along with original pages that have not been updated, into the most recent version of the data file.

[0050] It will be noted that, in the example of Figure 8, the new data pages 37 are stored in the first three pages 0-2 of the new block PBN1, rather than in the same pages 3-5 which they replace in the original block PBN0. By keeping track of the individual logical page numbers, the updated data need not necessarily be stored in the same page offset of the new block as that of the old block where superceded data is contained. Page(s) of updated data can also be written to erased pages of the same block as the page of data being superceded.

[0051] As a result, there is no constraint presented by the techniques being described that limit which physical page new data can be written into. But the memory system in which these techniques are implemented may present some constraints. For example, one NAND system requires that the pages within the blocks be programmed in sequential order. That means that programming of the middle pages 3-5, as done in the new block 25 (Figure 6), wastes the pages 0-2, which cannot later be programmed. By storing the new data 37 in the first available pages of the new block 39 (Figure 8) in such a restrictive system, the remaining pages 3-7 are available for later use to store other data. Indeed, if the block 39 had other data stored in its pages 0-4 at the time the three pages of new data 37 were being stored, the new data could be stored in the remaining unused pages 5-7. This makes maximum use of the available storage capacity for such a system.

[0052] An example of the structure of data stored in an individual page of the blocks of Figure 8 is shown in Figure 10. The largest part is user data 45. An error correction code (ECC) 47 calculated from the user data is also stored in the page. Overhead data 49, including the LBN and page tag 41 (logical page offset), the time stamp 43 and an ECC 51 calculated from the overhead data are also stored in the page. By having an ECC 50 covering the overhead data that is separate from the user data ECC 47, the overhead 49 may be read separately from the user data and evaluated as valid without the need to transfer all of the data stored in the page. Alternatively, however, where the separate reading of the overhead data 49 is not a frequent event, all of the data in the page may be covered by a single ECC in order to reduce the total number of bits of ECC in a page.

[0053] A second specific implementation of the inventive technique can also be described with respect to Figure 8. In this example, the time stamp is used only to determine the relative age of the data stored in blocks, while the most recent pages among those that carry the same LBN and page number are determined by their relative physical locations. The time stamp 43 then does not need to be stored as part of each page. Rather, a single time stamp can be recorded for each block, either as part of the block or elsewhere within the non-volatile memory, and is updated each time a page of data is written into the block. Data is then read from pages in an order of descending physical address, starting from the last page of the most recently updated block containing data pages having the same LBN.

[0054] In Figure 8, for example, the pages are first read in the new block PBN1 from the last (page 15) to the first (page 0), followed by reading the pages of the original block PBN0 in the same reverse order. Once logical pages 3, 4 and 5 have been read from the new block PBN1, the superceded data in those pages of the original block PBN0 that are identified by the same logical page numbers can be skipped during the reading process. Specifically, physical pages 3, 4 and 5 of the old block PBN0 are skipped during reading, in this example, once the controller determines that their LBN/pages 41 are the same as those of the pages already read from the new block PBN1. This process can increase the speed of reading and reduce the number of overhead bits 49 that need to be stored for each page. Further, when this reverse page reading technique is employed, the table of Figure 9 used by the controller during a reading operation can be simplified into the form of Figures 5A and 5B. Only an identity of those physical blocks containing data of a common logical block and the relative times that the physical blocks were programmed need to be known in order to carry out this efficient reading process.

[0055] Figure 11 illustrates an extension of the example of Figure 8 by including a second update to the data originally written in the block PBN0. New data 51 for logical pages 5, 6, 7 and 8 is written to the respective physical pages 3, 4, 5 and 6 of the new block PBN1, along with their LBN and page number. Note, in this example, that the data of logical page 5 is being updated for the second time. During a reading operation that begins from the last page of the new block PBN1, the most recently written logical pages 8, 7, 6 and 5 of the data of interest are first read in that order. Thereafter, it will be noted that the LBN/page overhead field in physical page 2 of PBN1 is the same as that read from the physical page 3, so the user data of page 2 is not read. The physical pages 1 and 0 are then read. Next, the pages of the original block PBN0 are read, beginning with physical page 15. After reading physical pages 15-9, the controller will note that the LBN/page fields of each of pages 8-3 match those of pages whose data has already been read, so the old data need not be read from those pages. The efficiency of the reading process is thus improved. Finally, the original data of physical pages 2-0 are read since that data was not updated.

[0056] It will be noted that this example of reading pages in a reverse order efficiently sorts out the new data pages from the superceded data pages because data are written in physical page locations of an erased block in order from page 0 on. This technique is not limited to use with a memory system having such a specific programming constraint, however. So long as the order in which pages are programmed within a given block is known, the data from those pages may be read in the reverse order from which they were written. What is desired is that the most recently programmed pages having a common LBN with others that were earlier programmed be read first, and these are the most recently programmed pages. The most recent versions of updated pages are read first so that the superceded versions may easily be identified thereafter.

[0057] A table showing the correspondence between the logical data and physical page addresses for the example of Figure 11 is given in Figure 12. Although there have been two data updates, both are represented by the single column for the second block PBN1. The physical page noted in PBN1 for the logical page 5 is simply changed upon the second update to that page occurring. If the updating involves a third block, then another column is added for that other block. The table of Figure 12, constructed by reading the overhead data from each of the pages in blocks to which data of a common LBN has been written, can be used by the first implementation when the reverse page reading technique is not used. When the reverse page reading technique described above is used, the table of Figure 12 need be built only to identify a correspondence between an LBN and all PBNs containing data of that LBN.

[0058] An efficient way to organize pages of data being read from a physical block, where one or more of the pages has been updated, is illustrated by Figure 13. Enough space is provided in a volatile memory of the controller to buffer at least several pages of data at a time, and preferably a full block of data. That is what is shown in Figure 13. Sixteen pages of data, equal to the amount stored in a non-volatile memory block, are stored in the controller memory. Since the pages are most commonly read out of order, each page of data is stored in its proper position with respect to the other pages. For example, in the reverse page read operation of Figure 11, logical page 8 if the first to be read, so it is stored in position 8 of the controller memory, as indicated by the "1" in a circle. The next is logical page 7, and so forth, until all pages of data desired by the host are read and stored in the controller memory. The entire set of page data is then transferred to the host without having to manipulate the order of the data in the buffer memory. The pages of data have already be organized by writing them to the proper location in the controller memory.

[0059] A method of programming a non-volatile memory system that utilizes the techniques described with respect to Figures 8 and 9 is illustrated in the flow chart of Figure 14. Data for pages of an existing file to be updated are received from a host system, as indicated by the block 52. It is first determined by a step 53 whether the number of pages of updated data to be stored is equal to or greater than the storage capacity of a block of the system, 16 pages being shown as the block capacity, for simplicity, in the above described example. If so, one or more unused, erased blocks are addressed, in a step 55, and the new data pages are written to the addressed block(s), in a step 57. Typically, the updating of one block or more of data will result in one or more blocks storing the data that have been superceded by the new data. If so, as indicated by a step 59, those blocks with superceded data are identified for erasure. For the purpose of increasing performance, it is preferable that erase operations occur in the background, or when host requested programming or reading operations are not taking place. After being erased, the blocks are returned to the inventory of unused, erased blocks for further use. Alternatively, erasure of the blocks can be deferred until they are needed for programming operations.

[0060] If, on the other hand, in the step 53, it is determined that there are fewer pages of new data than will utilize the full storage capacity of a block, a next step 61 determines whether there are enough unused pages in a block having some pages programmed with other data. If so, such a block is addressed, in a step 63. If not, a totally unused, erased block is addressed, in a step 65. In either case, in a step 67, the new data are programmed into unused pages of the addressed block. As part of this programming process, the LBN and page offset is written into the fields 41, and the time stamp into the fields 43 of each of the pages (Figure 8) of the updated data, in the manner described above.

[0061] A desirable feature of the programming process is to make available for future programming any blocks that store only superceded data. So the question is asked, in a step 69, whether the data updating process has resulted in an entire block remaining with only superceded data. If so, such a block is queued for erasure, in a step 71, and the process is then completed. If not, the step 71 is omitted and the data update is finished.

Metablock Operation

[0062] In order to improve performance by reducing programming time, a goal is to program as many cells in parallel as can reasonably be done without incurring other penalties. One implementation divides the memory array into largely independent sub-arrays or units, such as multiple units 80-83 of Figure 15, each unit in turn being divided into a large number of blocks, as shown. Pages of data are then programmed at the same time into more than one of the units. Another configuration further combines one or more of these units from multiple memory chips. These multiple chips may be connected to a single bus (as shown in Figure 2) or multiple independent busses for higher data throughput. An extension of this is to link blocks from different units for programming, reading and erasing together, an example being shown in Figure 15. Blocks 85-88 from respective ones of the units 80-83 can be operated together as a metablock, for example. As with the memory embodiments described above, each block, the smallest erasable group of the memory array, is typically divided into multiple pages, a page containing the smallest number of cells that are programmable together within the block. Therefore, a programming operation of the metablock shown in Figure 15 will usually include the simultaneously programming of data into at least one page of each of the blocks 85-88 forming the metablock, which is repeated until the metablock is full or the incoming data has all been programmed. Other metablocks are formed of different blocks from the array units, one block from each unit.

[0063] In the course of operating such a memory, as with others, pages of data less than an entire block often need to be updated. This can be done for individual blocks of a metablock in the same manner as described above with respect to either of Figures 4 or 6, but preferably by use of the improved technique described with respect to Figure 8. When any of these three techniques are used to update data of one block of the metablock, an additional block of memory within the same unit is also used. Further, a data update may require writing new data for one or more pages of two or more of the blocks of a metablock. This can then require use of up to four additional blocks 90-93, one in each of the four units, to update a data file stored in the metablock, even though the data in only a few pages is being updated.

[0064] In order to reduce the number of blocks required for such partial block updates, according to another aspect of the present invention, updates to pages of data within any of the blocks of the illustrated metablock are made, as illustrated by Figure 16, to a single additional block 90 in the memory unit 80, so long as unused pages in the block 80 remain. If, for example, data in three pages of the block 86 and two pages of the block 88 are being updated at one time, all five pages of the new data are written into the block 90. This can save the use of one block of memory, thereby to effectively increase the number of available erased blocks by one block. This helps avoid, or at least postpone, the time when an inventory of erased blocks becomes exhausted. If one or more pages from each of the four blocks 85-88 are being updated, all of the new data pages are programmed in the single block 90, thereby avoiding tying up an additional three blocks of memory to make the update. If the number of pages of new data exceed the capacity of an unused block, pages that the block 90 cannot accept are written to another unused block which may be in the same unit 80 or one of the other units 81-83.

[0065] Although the invention has been described with respect to various exemplary embodiments, it will be understood that the invention is entitled to protection within the full scope of the appended claims.

IT IS CLAIMED:

1. A method of simultaneously storing original and replacement data in a non-volatile memory system, comprising:

identifying the original and replacement data by the same logical address, and

distinguishing the replacement data from the original data by keeping track of the relative times that the original and replacement data have been programmed into the memory.

2. In a non-volatile memory system having a plurality of blocks of memory storage elements that are individually organized into a plurality of pages of memory storage elements, a method of substituting new data for superceded data within at least one page of one of the plurality of blocks while data in at least another page of said one block is not replaced, comprising:

programming the new data into at least one page of said one or another of the plurality of blocks,

identifying the at least one page of superceded data and the at least one page of new data by a common logical address, and

recording a relative time of programming the new and the superceded data.

3. The method of claim 2, wherein the relative time of programming is recorded for the individual pages in which the new and superceded data are programmed, whereby the at least one page of new data is distinguishable from the at least one page of superceded data by their recorded relative times of programming.

PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

ABSTRACT OF THE DISCLOSURE

Data in less than all of the pages of a non-volatile memory block are updated by programming the new data in unused pages of either the same or another block. In order to prevent having to copy unchanged pages of data into the new block, or to program flags into superceded pages of data, the pages of new data are identified by the same logical address as the pages of data which they superceded and a time stamp is added to note when each page was written. When reading the data, the most recent pages of data are used and the older superceded pages of data are ignored. This technique is also applied to metablocks that include one block from each of several different units of a memory array, by directing all page updates to a single unused block in one of the units.

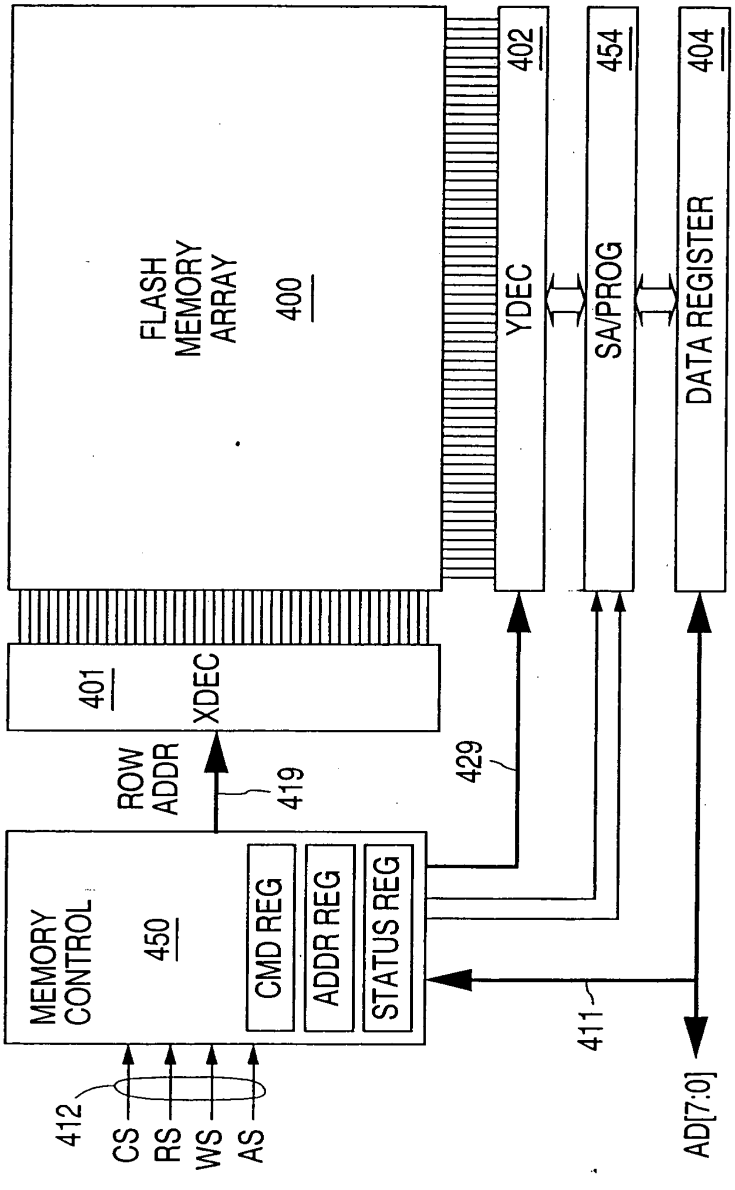

FLASH MEMORY ARRAY
400
401
XDEC
ROW ADDR
419
YDEC
402
SA/PROG
454
DATA REGISTER
404
429
MEMORY CONTROL
450
CMD REG
ADDR REG
STATUS REG
412
CS
RS
WS
AS
411
AD[7:0]

FIG. 1
(PRIOR ART)

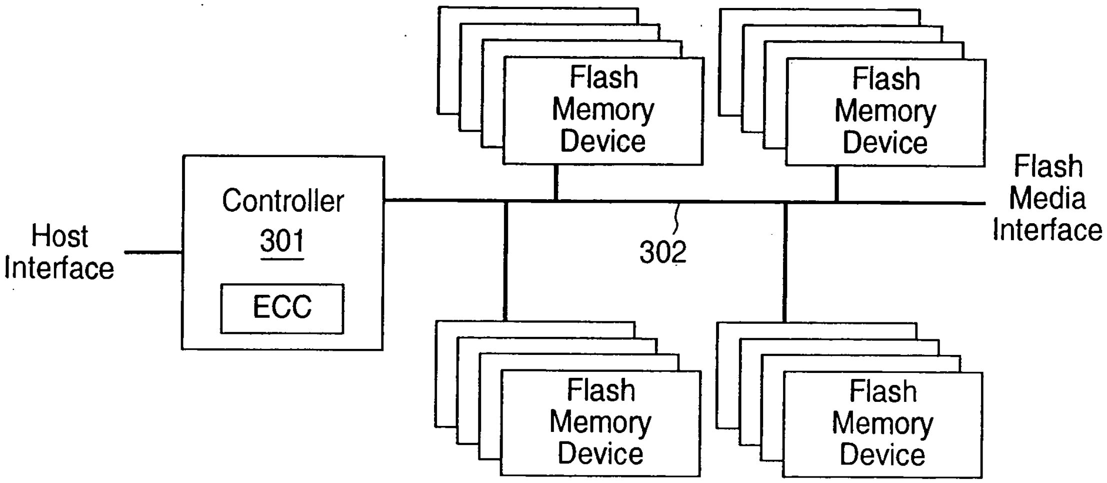
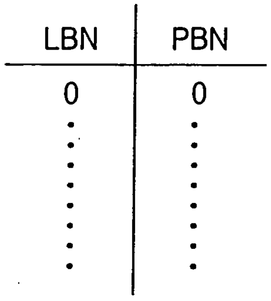

Flash Memory Device
Flash Memory Device
Controller 301
ECC
Host Interface
302
Flash Media Interface
Flash Memory Device
Flash Memory Device

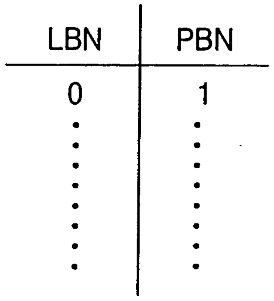
FIG. 2
(PRIOR ART)



LBN	PBN
0	0
⋮	⋮

Original Block 11

FIG. 5A



LBN	PBN
0	1
⋮	⋮

With New Block 15

FIG. 5B

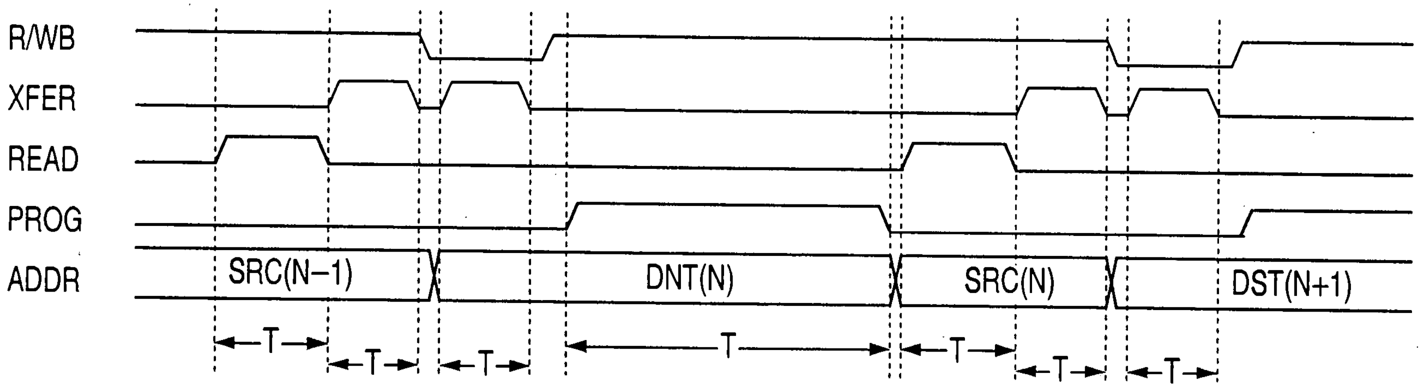

R/WB
XFER
READ
PROG
ADDR
SRC(N−1)
DNT(N)
SRC(N)
DST(N+1)
T
T
T
T
T
T
T

FIG. 3
(PRIOR ART)

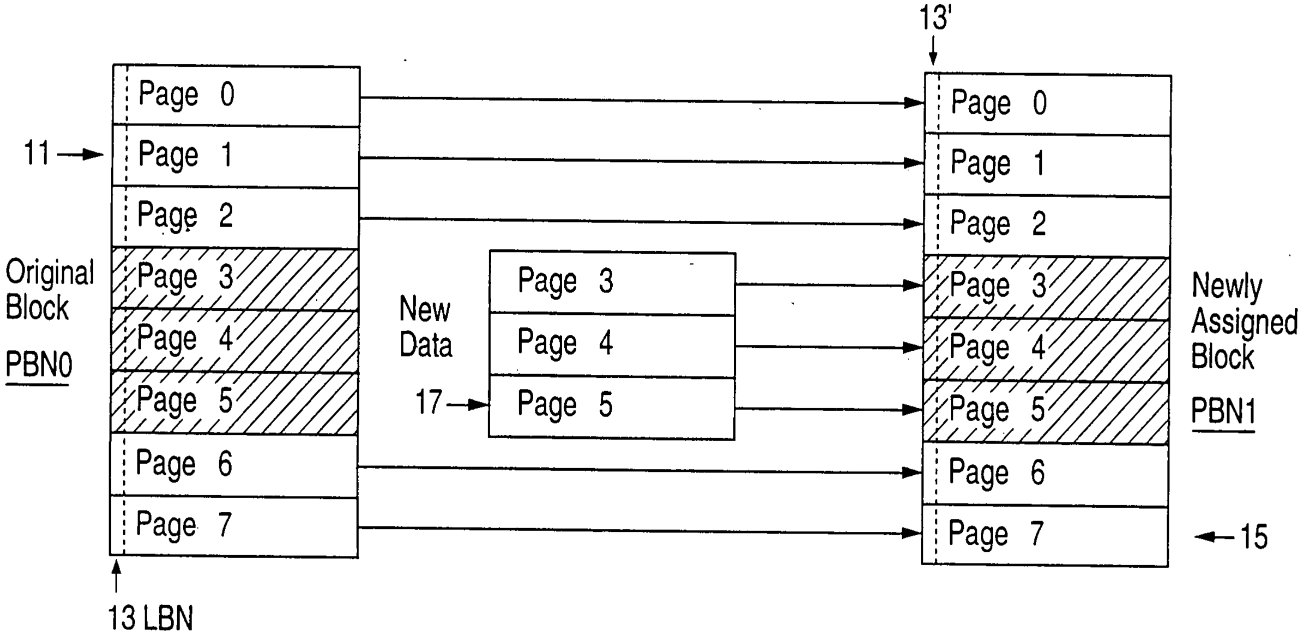

13'
11
Original Block
PBN0
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
13 LBN
New Data
17
Page 3
Page 4
Page 5
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Newly Assigned Block
PBN1
15

FIG. 4
(PRIOR ART)

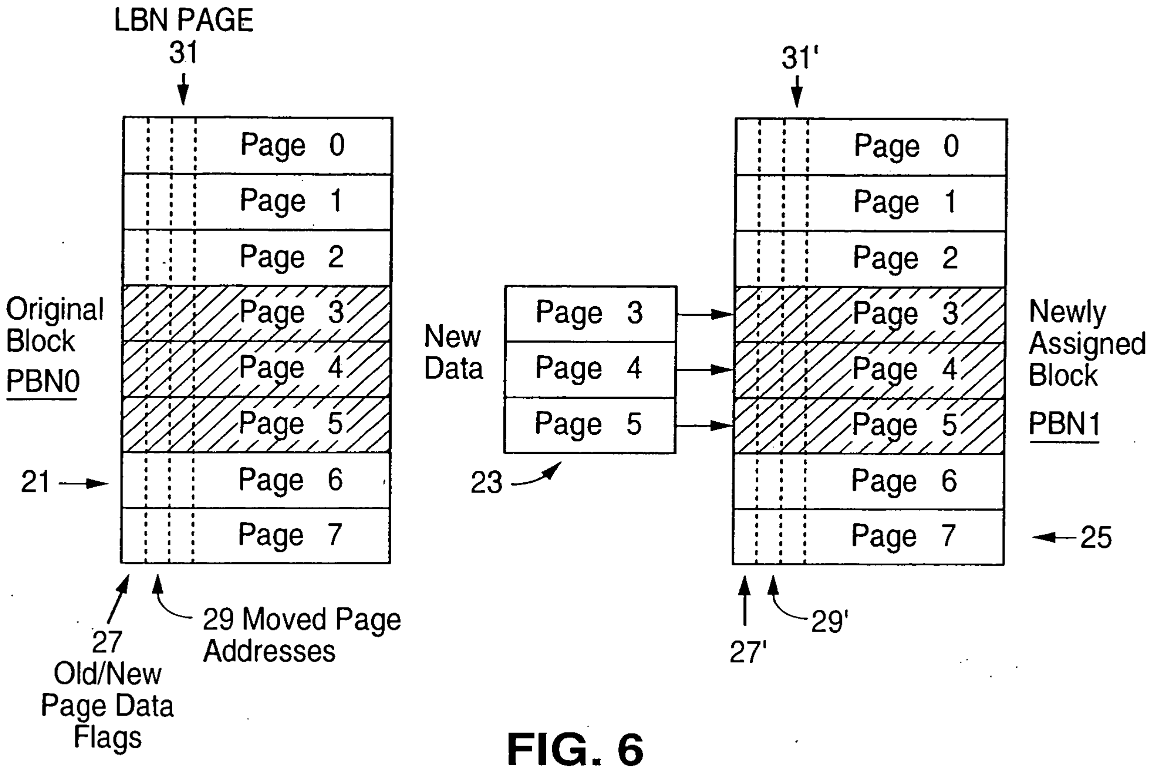

LBN PAGE
31
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Original Block
PBN0
21
27
Old/New Page Data Flags
29 Moved Page Addresses
New Data
Page 3
Page 4
Page 5
23
31'
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Newly Assigned Block
PBN1
25
27'
29'

FIG. 6
(PRIOR ART)

LBN	Page	PBN	Page
0	0	0	0
0	1	0	1
0	2	0	2
0	3	0	3
0	4	0	4
0	5	0	5
0	6	0	6
0	7	0	7
:	:	:	:

Original Block

FIG. 7A

LBN	Page	PBN	Page
0	0	0	0
0	1	0	1
0	2	0	2
0	3	1	3
0	4	1	4
0	5	1	5
0	6	0	6
0	7	0	7
:	:	:	:

With New Block

FIG. 7B

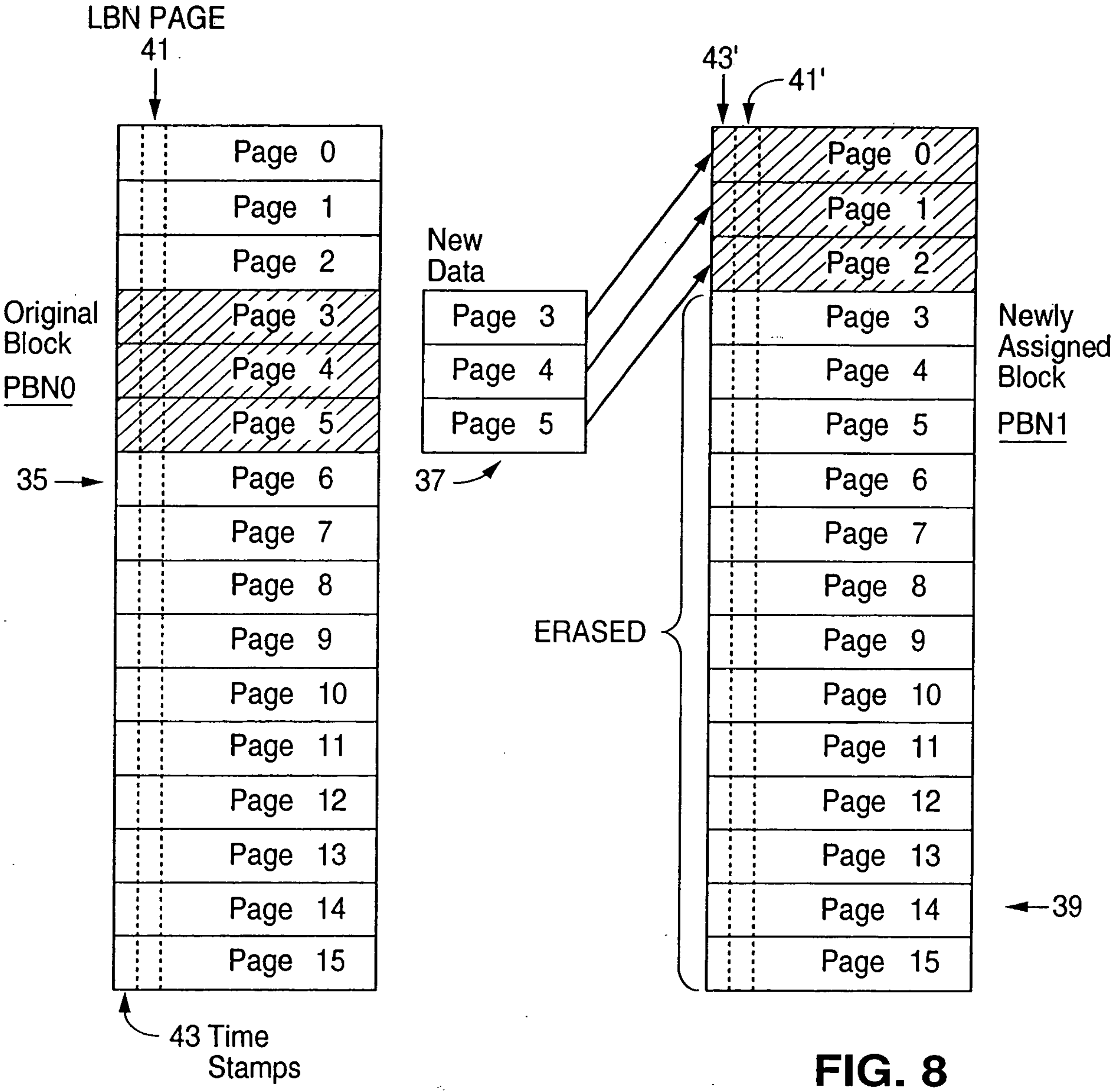

LBN PAGE
41
Original Block
PBN0
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
35
43 Time Stamps
New Data
Page 3
Page 4
Page 5
37
ERASED
43'
41'
Newly Assigned Block
PBN1
39

FIG. 8

LBN	Page	PBN0	Page	PBN1	Page
0	0	0	0		
0	1	0	1		
0	2	0	2		
0	3	0	3	1	0
0	4	0	4	1	1
0	5	0	5	1	2
0	6	0	6		
0	7	0	7		
:	:	:	:		

FIG. 9

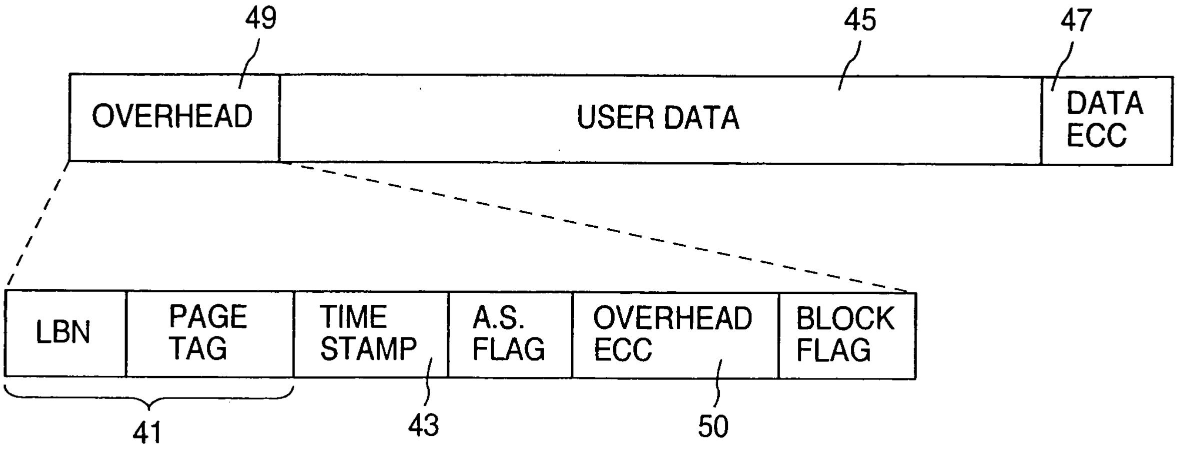

49
45
47
OVERHEAD
USER DATA
DATA ECC
LBN
PAGE TAG
TIME STAMP
A.S. FLAG
OVERHEAD ECC
BLOCK FLAG
41
43
50

FIG. 10

LBN	Page	PBN0	Page	PBN1	Page
0	0	0	0		
0	1	0	1		
0	2	0	2		
0	3	0	3	1	0
0	4	0	4	1	1
0	5	0	5	1	3
0	6	0	6	1	4
0	7	0	7	1	5
0	8	0	8	1	6
0	9	0	9		
:	:	:	:	:	

FIG. 12

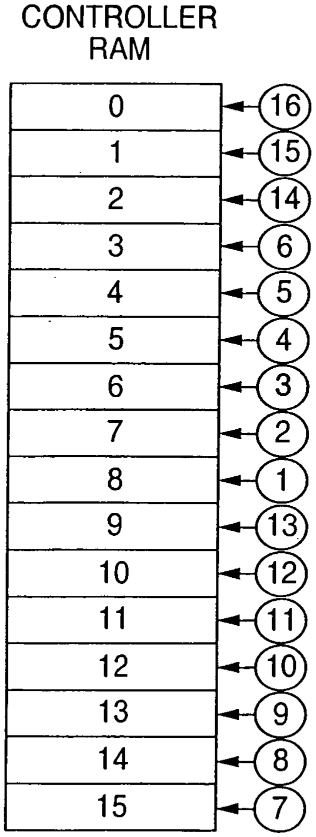

CONTROLLER RAM
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
15
14
6
5
4
3
2
1
13
12
11
10
9
8
7

FIG. 13

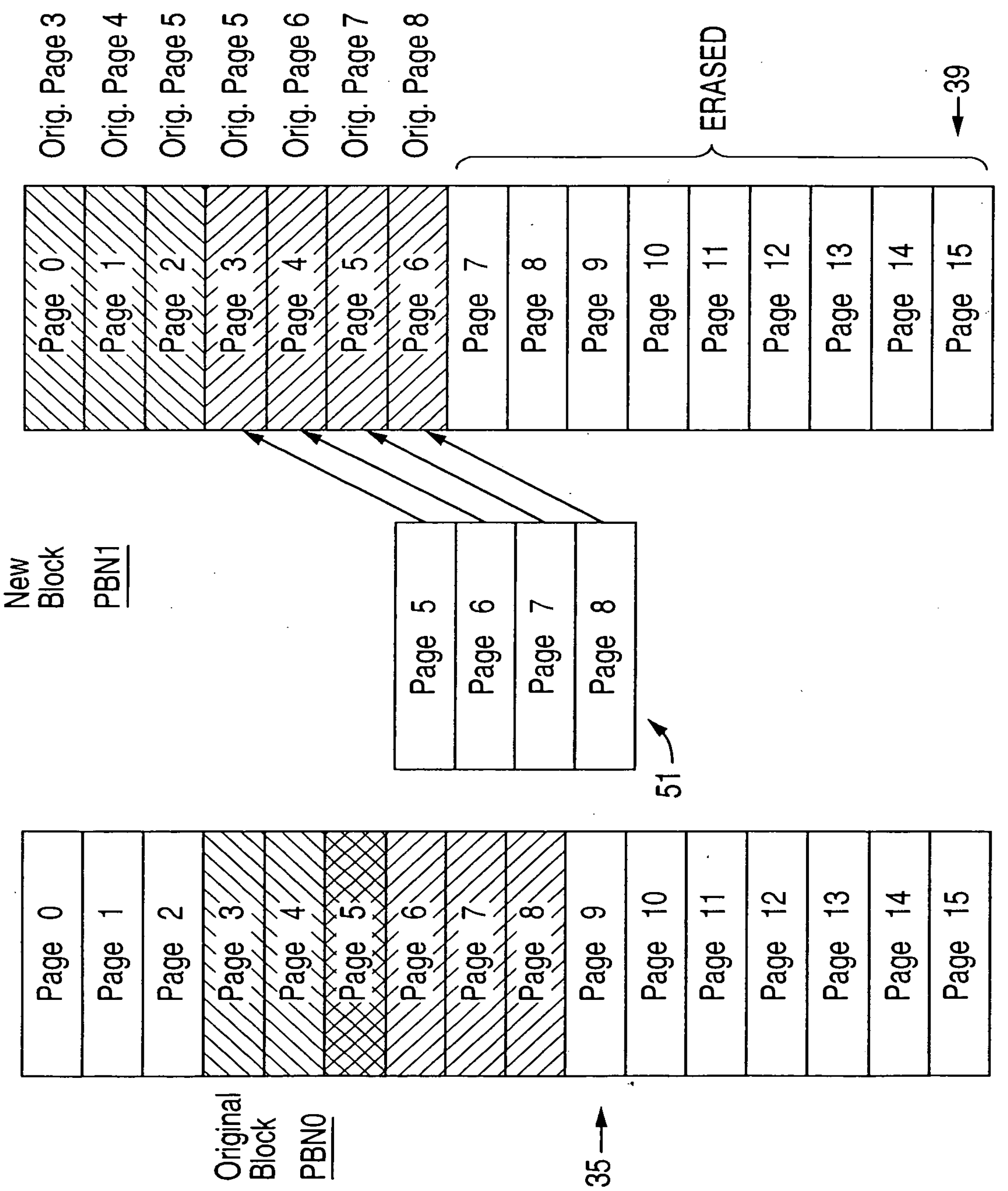

Original Block PBN0
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
35
Page 5
Page 6
Page 7
Page 8
51
New Block PBN1
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
Orig. Page 3
Orig. Page 4
Orig. Page 5
Orig. Page 5
Orig. Page 6
Orig. Page 7
Orig. Page 8
ERASED
39

FIG. 11

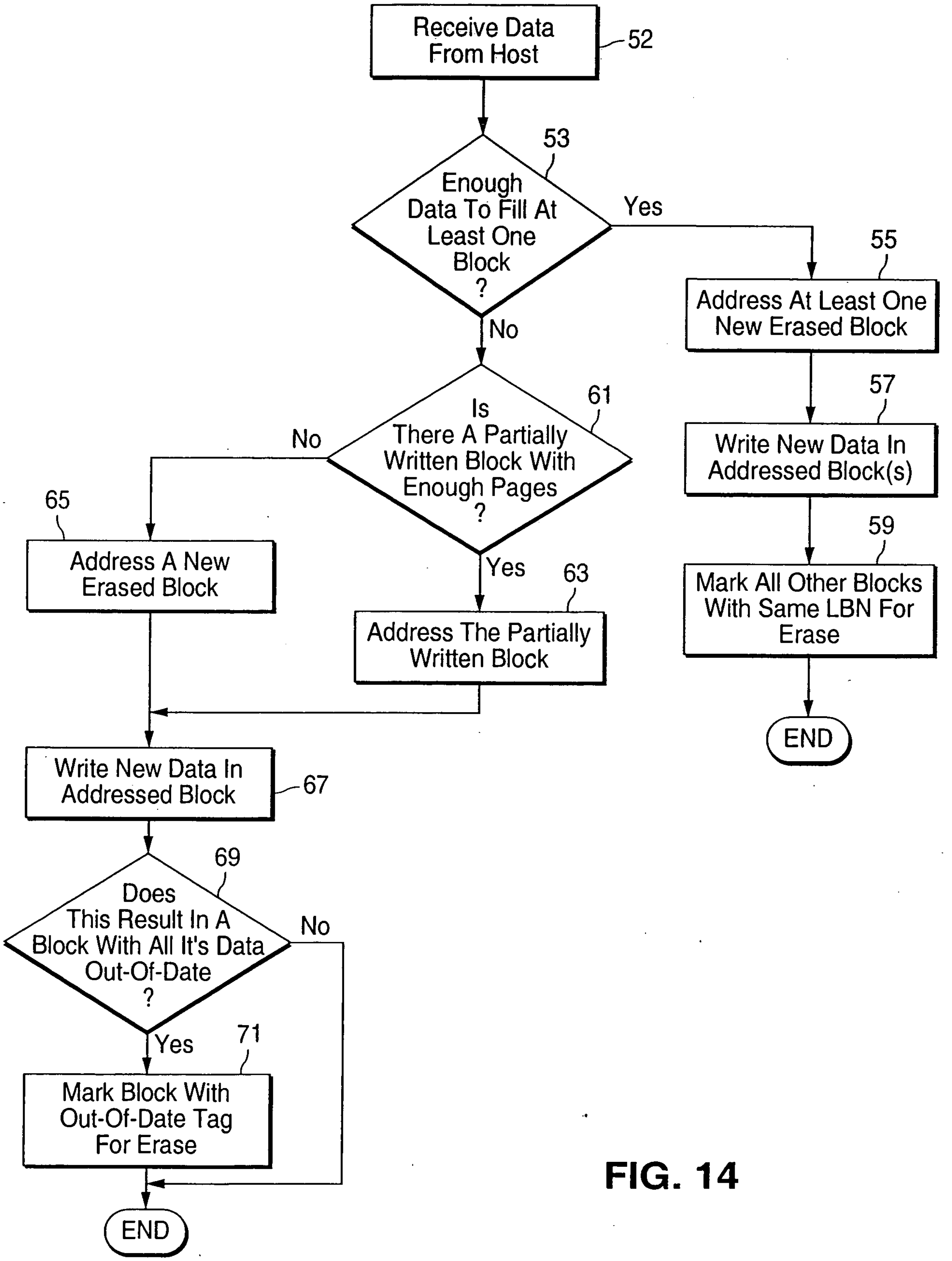
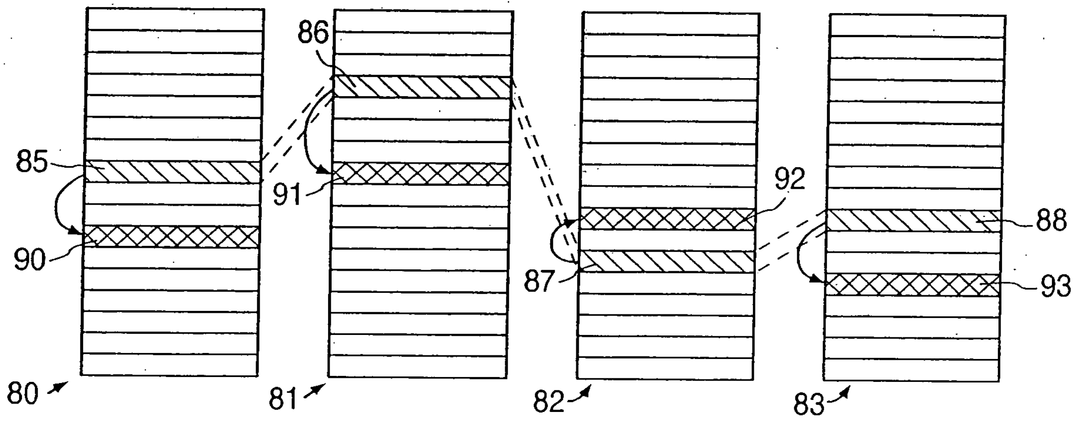
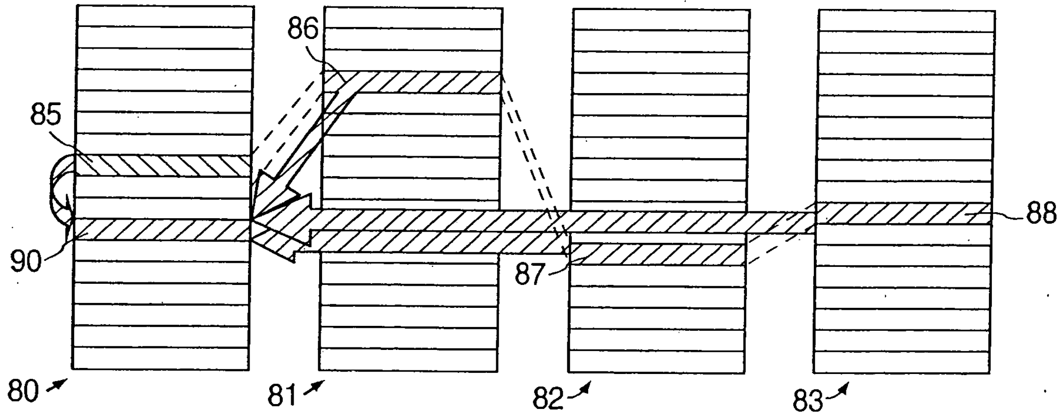

Receive Data From Host
52
Enough Data To Fill At Least One Block ?
53
Yes
No
Address At Least One New Erased Block
55
Write New Data In Addressed Block(s)
57
Mark All Other Blocks With Same LBN For Erase
59
END
Is There A Partially Written Block With Enough Pages ?
61
No
Yes
Address A New Erased Block
65
Address The Partially Written Block
63
Write New Data In Addressed Block
67
Does This Result In A Block With All It's Data Out-Of-Date ?
69
No
Yes
Mark Block With Out-Of-Date Tag For Erase
71
END

FIG. 14



86
85
91
92
88
90
87
93
80
81
82
83

FIG. 15



86
85
88
90
87
80
81
82
83

FIG. 16

COPY

Attorney Docket No.: 11587 M-10262 US

DECLARATION FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below adjacent to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of subject matter (process, machine, manufacture, or composition of matter, or an improvement thereof) which is claimed and for which a patent is sought by way of the application entitled

PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

which (check) ☒ is attached hereto.
☐ and is amended by the Preliminary Amendment attached hereto.
☐ was filed on as Application Serial No.
☐ and was amended on (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information, which is material to patentability as defined in Title 37, Code of Federal Regulations, § 1.56.

I declare that all statements made herein of my own knowledge are true, all statements made herein on information and belief are believed to be true, and all statements made herein are made with the knowledge that whoever, in any matter within the jurisdiction of the Patent and Trademark Office, knowingly and willfully falsifies, conceals, or covers up by any trick, scheme, or device a material fact, or makes any false, fictitious or fraudulent statements or representations, or makes or uses any false writing or document knowing the same to contain any false, fictitious or fraudulent statement or entry, shall be subject to the penalties including fine or imprisonment or both as set forth under 18 U.S.C. 1001, and that violations of this paragraph may jeopardize the validity of the application or this document, or the validity or enforceability of any patent, trademark registration, or certificate resulting therefrom.

Full name of sole (or ~~first joint~~ inventor) Kevin M. Conley
Inventor's Signature: Kevin Conley Date: 1/18/2001
Residence:
Post Office Address: 5983 Alvarado Ct
San Jose, CA 95120
Citizenship: U.S.A.

BEST AVAILABLE COPY

PATENT APPLICATION SERIAL NO ____________

U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE
FEE RECORD SHEET

```
10/18/2005 SSITHIB1 00000091 502664   11250238

01 FC:1011        300.00 DA
02 FC:1111        500.00 DA
03 FC:1311        200.00 DA
```

PTO-1556
(5/87)

U.S. Government Printing Office: 2002 — [illegible]

PTO/SB/06 (12-04)
Approved for use through 7/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875 Effective December 8, 2004

Application or Docket Number
11280238

APPLICATION AS FILED – PART I

	(Column 1)	(Column 2)	SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
FOR	NUMBER FILED	NUMBER EXTRA	RATE ($)	FEE ($)		RATE ($)	FEE ($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	150.00		N/A	300.00
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	$250		N/A	$500
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	$100		N/A	$200
TOTAL CLAIMS (37 CFR 1.16(i))	3 minus 20 =	*	X$ 25 =		OR	X$50 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X100 =			X200 =	
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))			+180=			+360=	
			TOTAL			TOTAL	1000

* If the difference in column 1 is less than zero, enter "0" in column 2.

APPLICATION AS AMENDED – PART II

AMENDMENT A		(Column 1)		(Column 2)	(Column 3)	SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	X$ 25 =		OR	X$50 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	X100 =		OR	X200 =	
	Application Size Fee (37 CFR 1.16(s))									
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					+180=		OR	+360=	
						TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B		(Column 1)		(Column 2)	(Column 3)					
		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	X$ 25 =		OR	X$50 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	X100 =		OR	X200 =	
	Application Size Fee (37 CFR 1.16(s))									
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					+180=		OR	+360=	
						TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

Application Data Sheet

Application Information

Application number::	
Filing Date::	10/13/05
Application Type::	Regular
Subject Matter::	Utility
Title::	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory
Attorney Docket Number::	SNDK.156US2
Request for Early Publication?::	No
Request for Non-Publication?::	No
Total Drawing Sheets::	9
Small Entity::	No
Petition included?::	No
Secrecy Order in Parent Appl.?::	No

Applicant Information

Applicant Authority type::	Inventor
Primary Citizenship Country::	US
Status::	Full Capacity
Given Name::	Kevin
Middle Name::	M.
Family Name::	Conley
City of Residence::	San Jose
State or Province of Residence::	CA
Country of Residence::	US
Street of mailing address::	5983 Alvarado Court

City of mailing address:: San Jose
State or Province of mailing address:: CA
Postal or Zip Code of mailing address:: 95120

Correspondence Information

Correspondence Customer Number::	36257
Name::	Parsons Hsue & de Runtz LLP
Street of mailing address::	595 Market Street Suite 1900
City of mailing address::	San Francisco
State or Province of mailing address::	CA
Postal or Zip Code of mailing address::	94105
Telephone::	(415) 318-1160, (415) 318-1163
Fax::	(415) 693-0194
E-Mail address::	gparsons@phdr-law.com

Representative Information

Representative Customer Number::	36257	

Domestic Priority Information

Application::	**Continuity Type::**	**Parent Application::**	**Parent Filing Date::**
This Application	Continuation of	10/841,388	05/07/04
10/841,388	Continuation of	09/766,436	01/19/01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	Unassigned	Filing Date:	Herewith
Examiner:	Unassigned	Group Art Unit:	Unassigned
Docket No.:	SNDK.156US2	Conf. No.:	Unassigned

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

INFORMATION DISCLOSURE STATEMENT

Dear Sir:

Pursuant to 37 C.F.R. §§ 1.56, 1.97 and 1.98, Applicant calls the documents listed on the enclosed Form PTO-1449 to the Examiner's attention in this patent application.

Copies of the documents listed on the accompanying Form PTO-1449 that are not enclosed were previously submitted in Application Numbers 10/841388 and 09/766,436, from which this Application claims an earlier effective filing date.

Citation of these documents shall not be construed as (1) an admission that the documents are prior art with respect to the invention or inventions claimed in this application, (2) a representation that a search has been made (other than as indicated by any cited document), or (3) an admission that the cited information is, or is considered to be, material to patentability as defined in § 1.56(b).

This information disclosure statement is submitted under 37 C.F.R. § 1.97(b) and consequently no fee should be required. The Commissioner is authorized, however, to charge any fee that may be required, or to credit any overpayment, against Deposit Account No. 502664.

EXPRESS MAIL LABEL NO:

EV663653274US

Respectfully submitted,

Gerald P. Parsons
Reg. No.: 24,486

October 13, 2005
Date

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	
(Form PTO-1449)	**Filing Date**	**Art Group**

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
	1	5,043,940	8/27/1991	Harari			
	2	5,172,338	12/15/1992	Mehrotra et al.			
	3	5,341,330	8/23/1994	Wells et al.			
	4	5,404,485	4/4/1995	Ban			
	5	5,457,658	10/10/1995	Niijima et al.			
	6	5,479,638	12/26/1995	Assar et al.			
	7	5,481,691	1/2/1996	Day, III et al.			
	8	5,485,595	1/16/1996	Assar et al.			
	9	5,598,370	1/28/1997	Niijima			
	10	5,648,929	7/15/1997	Miyamoto			
	11	5,649,200	7/15/1997	Leblang et al.			
	12	5,835,935	11/10/1998	Estakhri et al.			
	13	5,838,614	11/17/1998	Estakhri et al.			
	14	5,845,313	12/1/1998	Estakhri et al.			
	15	5,860,090	1/12/1999	Clark			
	16	5,890,192	3/30/1999	Lee et al.			
	17	5,896,393	4/20/1999	Yard et al.			
	18	5,907,856	5/25/1999	Estakhri et al.			
	19	5,986,933	11/16/1999	Takeuchi et al.			
	20	5,987,563	11/16/1999	Itoh et al.			
	21	5,999,947	12/7/1999	Zollinger et al.			
	22	6,034,897	3/7/2000	Estakhri et al.			
	23	6,040,997	3/21/2000	Estakhri et al.			
	24	6,115,785	9/5/2000	Estakhri et al.			
	25	6,122,195	9/19/2000	Estakhri et al.			
	26	6,125,435	9/26/2000	Estakhri et al.			
	27	6,134,151	10/17/2000	Estakhri et al.			
	28	6,151,247	11/21/2000	Estakhri et al.			
	29	6,161,163	12/12/2000	Komatsu et al.			
	30	6,219,768 B1	4/17/2001	Hirabayashi et al.			

Sheet 1 of 2
Express Mail No.: EV663653274US

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	
(Form PTO-1449)	**Filing Date**	**Art Group**

U.S. Patent Documents (continued)

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
	31	6,330,634	12/11/2001	Fuse et al.			
	32	6,426,893	7/30/2002	Conley et al.			
	33	6,449,625	9/10/2002	Wang			
	34	6,567,307	5/20/2003	Estakhri			
	35	6,763,424	7/13/2004	Conley			

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No
	36	WO 94/22075	9/29/1994	WIPO			X	
	37	WO 94/20906	9/15/1994	WIPO			X	
	38	WO 02/49309A2	20/06/2002	WIPO			X	
	39	FR 2,742,893	20/12/1995	France			Abstract	X
	40	WO 02/058074A2	7/25/2002	WIPO			X	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

	41	European Patent Office, "Communication Pursuant to Article 96(2) EPC" mailed in related European patent application no. 02703078.2-2210 on December 2, 2004, 3 pages.
	42	"International Preliminary Examination Report", European Patent Office, Corresponding Application PCT/US02/00366, July 4, 2003, 7 pages.
	43	"FTL Updates, PCMCIA Document Number 0165", Personal Computer Memory Card International Association, Version 002, Release 003, March 4, 1996, pp. 1-26.
	44	Mergi, Aryeh and Schneider, Robert, "M-Systems & SCM Flash Filing Software Flash Translation Layer - FTL", PCMCIA, July 1994, 15 pages.
	45	Petro Estakhri et al., "Moving Sectors Within a Block of Information in a Flash Memory Mass Storage Architecture", U.S. Patent Application No. 09/620,544, filing date July 21, 2000, 48 pages.
	46	PCT International Search Report, European Patent Office, corresponding PCT Application No. PCT/US02/00366, 7/4/2003, 5 pages.

Examiner	Date Considered

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 2 of 2
Express Mail No.: EV663653274US

PARSONS HSUE & DE RUNTZ LLP

595 Market Street, Suite 1900, San Francisco, CA 94105
tel 415.318.1160 fax 415.693.0194

RECEIVED
CENTRAL FAX CENTER
JAN 2 6 2006

Date	January 26, 2006
To	United States Patent and Trademark Office
Fax number	(571) 273-8300
Tel number	(800) 786-9199
From	Gerald P. Parsons, Reg. No. 24,486
Total pages	4

COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

Applicant:	Kevin M. Conley
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory
Application No.:	11/250,238
Filing Date:	October 13, 2005
Conf. No.:	7727
Atty Docket No.:	SNDK.156US2

ENCLOSED:

1. Transmittal Letter (1 page)
2. Power of Attorney by Assignee of Entire Interest (2 pages)

GPP/meb

Due Date: January 13, 2006
Transmission Date: January 26, 2006

THIS FACSIMILE IS SENT BY A LAW FIRM AND MAY CONTAIN INFORMATION THAT IS CONFIDENTIAL OR PRIVILEGED If you are not the intended recipient, please notify us immediately and return this facsimile and any attachments to us by mail.

PARSONS HSUE & DE RUNTZ LLP

595 Market Street, Suite 1900 San Francisco, CA 94105
tel 415.318.1160 fax 415.693.0194

RECEIVED
CENTRAL FAX CENTER
JAN 2 6 2006

January 26, 2006

Commissioner For Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Re:	Applicant:	Kevin M. Conley		
	Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
	Application No.:	11/250,238	Filing Date:	October 13, 2005
	Examiner:	Unassigned	Group Art Unit:	2189
	Docket No.:	SNDK.156US2	Conf. No.:	7727

Dear Sir:

Transmitted herewith are the following documents in the above-identified application:

(1) This Transmittal Letter (1 page); and
(2) Power of Attorney by Assignee of Entire Interest (2 pages).

☒ No additional fee is required.

Certificate of Transmission Under 37 CFR 1.8

I hereby certify that this correspondence is being facsimile transmitted to the United States Patent and Trademark Office on January 26, 2006

Mary E. Buggie
Mary E. Buggie

Respectfully submitted,

Gerald P. Parsons
Reg. No. 24,486

RECEIVED
CENTRAL FAX CENTER
JAN 2 6 2006

SDK0156.002US

BEST AVAILABLE COPY

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Unassigned	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

POWER OF ATTORNEY BY ASSIGNEE OF ENTIRE INTEREST

Sir:

SanDisk Corporation, a Delaware corporation, hereby revokes all powers of attorney previously given and appoints the following attorneys specifically and solely to prosecute the above-identified application and to transact all business in the United States Patent and Trademark Office in connection therewith:

Gerald P. Parsons, Reg. No. 24,486
James S. Hsue, Reg. No. 29,545
K. Alison de Runtz, Reg. No. 37,119

Please direct all communications in connection with the above-identified patent application to:

Customer Number: 36257

Parsons Hsue & de Runtz LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
Telephone No.: (415) 318-1160
Fax No.: (415) 693-0194

SDK0156.002US

BEST AVAILABLE COPY

ASSIGNEE CERTIFICATION UNDER 37 CFR 3.73(B)

The undersigned hereby certifies that SanDisk Corporation is the assignee of the entire right, title and interest in the above-identified patent application by virtue of a chain of title from the inventor to SanDisk Corporation, as shown by the Assignment from the inventor, which was separately recorded in the United States Patent and Trademark Office at Reel 011487, Frame 0823.

Each of the undersigned is empowered to sign this certificate on behalf of SanDisk Corporation.

1/17/06
Date

Name: Charles Van Orden
Title: Vice President and General Counsel

Jan 17, 2006
Date

Name: Megan Comport
Title: Assistant Secretary, SanDisk Corporation

PAGE 4/4 * RCVD AT 1/26/2006 5:16:39 PM [Eastern Standard Time] * SVR:USPTO-EFXRF-6/26 * DNIS:2738300 * CSID:4156930196 * DURATION (mm-ss):01-10

IFW

OIPE
MAR 13 2006
PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Unknown	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Certificate of Mailing Under 37 CFR 1.8

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on March 8, 2006

Mary E. Buggie

Mary E. Buggie

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT

Dear Sir:

Pursuant to 37 C.F.R. §§ 1.56, 1.97 and 1.98, Applicant calls the document listed on the enclosed Form PTO-1449 to the Examiner's attention in this patent application. A copy of the document listed on the accompanying Form PTO-1449 is enclosed.

Citation of these documents shall not be construed as (1) an admission that the documents are prior art with respect to the invention or inventions claimed in this application, (2) a representation that a search has been made (other than as indicated by any cited document), or (3) an admission that the cited information is, or is considered to be, material to patentability as defined in § 1.56(b).

This information disclosure statement is submitted under 37 C.F.R. § 1.97(c). No fee should be required because, in accordance with § 1.97(e)(1), each item contained in this information disclosure statement was first cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of this information disclosure statement. The Commissioner is authorized, however, to charge any fee that may be required, or to credit any overpayment, against Deposit Account No. 502664. This form is being submitted in duplicate.

Respectfully submitted,

March 8, 2006
Date

Gerald P. Parsons
Reg. No. 24,486

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

OIPE IAP1
MAR 13 2006
PATENT & TRADEMARK OFFICE

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate

U.S. Published Patent Application Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

	1	The Patent Office of the People's Republic of China, "Notification of the First Office Action," mailed in related Chinese Application No. 02803882.7 on January 27, 2006, 14 pages, including translation.

Examiner	Date Considered

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 1 of 1

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	1245	("same" identical common) near5 (logical adj2 address)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 10:28
L2	3	timestamp same 1	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:21
L3	6502	711/103 or 711/102 or 711/202 or 711/203 or 711/209	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:20
L4	42961	(sanitiz$4 eras$4 programm$4) near5 (part portion partial partially)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:20
L5	415128	(prom eeprom flash non$volatile)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:20
L6	4034	L4 same L5	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:20
L7	187996	(original old out-of-date superceded) same (replacement updated new)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:21
L8	1245	("same" identical common) near5 (logical adj2 address)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:24
L9	101	L7 same L8	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:21
L10	49	3 and 9	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:21
L11	32	timestamp and 1	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:22
L12	3	3 and 11	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:22
L13	363	711/103 and 6	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:22
L14	71	5 and 9	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:24

EAST Search History

S1	319157	prom eeprom flash	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 10:27
S2	4993	(partial partially portion) adj3 updat$4	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:00
S3	10402	(stor$4 updat$4) near10 ((original old) adj2 data)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:03
S4	7695	((original old) adj2 data) same ((updated new replacement) adj2 data)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:03
S5	4558	(stor$4 updat$4) same S4	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:03
S6	253	S5 same S1	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:04
S7	8675	(identical "same") adj3 (logical lbn)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:32
S8	10	S6 same S7	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:30
S9	42884	(sanitiz$4 eras$4 programm$4) near5 (part portion partial partially)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:31
S10	414090	(prom eeprom flash non$volatile)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:32
S11	4030	S9 same S10	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:20
S12	9	S7 same S11	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 18:34
S13	254	S7 and S11	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:45
S14	2	"10841118"	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:07

EAST Search History

S15	0	"20050144357".pub.	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:07
S16	1	"20050144357"	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:11
S17	1	"20060031627"	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:11
S18	187657	(original old out-of-date superceded) same (replacement updated new)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/26 19:47
S19	1245	("same" identical common) near5 (logical adj2 address)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 10:28
S20	101	S18 same S19	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2006/07/27 12:21

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/250,238	10/13/2005	Kevin M. Conley	SNDK.156US2	7727

36257 7590 08/08/2006

PARSONS HSUE & DE RUNTZ LLP
595 MARKET STREET
SUITE 1900
SAN FRANCISCO, CA 94105

EXAMINER
DINH, NGOC V

ART UNIT	PAPER NUMBER
2189	

DATE MAILED: 08/08/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Office Action Summary	**Application No.** 11/250,238	**Applicant(s)** CONLEY, KEVIN M.	
	Examiner NGOC V. DINH	**Art Unit** 2189	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 03 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 October 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-3* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-3* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *31 October 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *10/13/05 03/13/06*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This office action is a response to the continuation of copending U.S. application Serial No. 10/841388 filed on 05/07/2004 (PN 6,968,421), which is a continuation of application No 09/766436 filed on 01/19/2001 (PN 6,763,424).

2. Claims 1-3 are presented for examination.

INFORMATION DISCLOSURE STATEMENT

3. The Applicant's submission of the IDS filed 10/13/2005 and 03/13/2006 have been considered. However, the Examiner does not consider Prior art WO 02/058074A2 because the Applicant does not provide copies of this prior art. As required by M.P.E.P. 609 C(2), a copy of the PTOL-1449 is attached to the instant office action.

As required by M.P.E.P. 2001.06(b) and C.F.R 1.98(d) since the instant application has been identified as a continuation application of an earlier filed application No. 10/841,388 and 09/766436, now Patent No 6968421 and 6763424 and is relied upon for an earlier filing date under 35 U.S.C. 120, the Examiner has reviewed the prior art cited in the earlier related application as required by M.P.E.P 904, and as stated in M.P.E.P 2001.06(b) no separate citation of the same prior art need be made in the instant application.

SPECIFICATION

4. The Applicant is reminded to update the status of the applications on page 1 of the specification appropriately. The Applicant should cite the Patent number 6,968,421 of the parent application serial no. 10/841,388.

DOUBLE PATENTING

The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. See *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent is shown to be commonly owned with this application. See 37 CFR 1.130(b).

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

5. Claims 1-4 of Patent No 6,969,421 contain every element of claims 1-3 of the instant application and as such anticipate claims 1-3 of the instant application.

> " A later patent claim is not patentably distinct from an earlier patent claim if the later claim is obvious over, or **anticipated** by, the earlier claim. In re Longi, 759 F.2d at 896, 225 USPQ at 651 (affirming a holding of obviousness-type double patenting because the claims at issue were obvious over claims in four prior art patents); In re Berg, 140 F.3d at 1437, 46 USPQ2d at 1233 (Fed. Cir. 1998) (affirming a holding of obviousness-type double patenting where a patent application claim to a genus is anticipated by a patent claim to a species within that genus). " ELI LILLY AND COMPANY v BARR LABORATORIES, INC., United States Court of appeals for the Federal Circuit, ON PETITION FOR REHEARING EN BANC (DECIDED: May 30, 2001).

Claim 1 correspond to claim 1 of Patent No 6,968,421.
Claim 2 correspond to claims 2-3 of Patent No 6,968,421.
Claim 3 correspond to claim 4 of Patent No 6,968,421.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

6. Claims 1-3 are rejected under 35 U.S.C.102 (e) as being anticipated by Fuse et al. PN 6,330,634.

Per claim 1, Fuse teaches a method of simultaneously storing original and replacement data in a non-volatile memory system [fig. 2], comprising: identifying the original and replacement data by the same logical address [col. 12, lines 60-67]; and distinguishing the replacement data from the original data by keeping track of the relative times [new or old data stored in the blocks is identified by using the identification number, col. 12, lines 60-67; the identification number being incremented whenever data in the block is rewritten, col. 11, lines 5-10] that the original and replacement data have been programmed into the memory col. 2, lines 30-50; col. 11, lines 5-10; col. 12, lines 60-67].

Per claim 2, Fuse teaches in a nonvolatile memory system having a plurality of blocks of memory storage elements [fig. 2, 4A-4C] that are individually organized into a plurality of pages of memory storage elements, a method of substituting new data for superceded data within at least one page of one of the plurality of blocks while data in at least another page of said one block is replaced [col. 6, lines 19-50], comprising: programming the new data into at least one page of said one or another of the plurality of blocks [col. 8, lines 20-40]; identifying the at least one page of superceded data and the at least one page of new data by a common logical address, and recording a relative time of programming the new and the superceded data [col. 11, lines 1-15; col. 12, lines 60-67].

Per claim 3, Fuse teaches the relative time of programming is recorded for the individual page [col. 13, lines 5-10, 54-62] in which the new and superceded data are programmed, whereby the at least one page of new data is distinguishable from the at least one page of supercede data by their recorded relative times of programming [col. 11, lines 5-15; col. 12, lines 60-67].

Conclusion

5. Any response to this action should be mailed to:

Under Secretary of Commerce for intellectual Property and Director of the
United States Patent and Trademark Office
PO Box 1450
Alexandria, VA 22313-1450

or faxed to:

(571) 273-8300, (for Official communications intended for entry)

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PMR) system. Status information for published Applications may be obtained from either Private PMR or Public PMR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http//pak-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ngoc Dinh whose telephone number is (571) 272-4191. The examiner can normally be reached on Monday-Friday 8:30 AM-5:00 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Reginald Bragdon, can be reached on (571) 272-4204.

NGOC DINH
Patent Examiner
ART UNIT 2189
August 02, 2006

Reginald G. Bragdon
REGINALD BRAGDON
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

OIPE
MAR 1 3 2006
PATENT & TRADEMARK OFFICE

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate

U.S. Published Patent Application Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate

Foreign Patent Documents

		Document	Date	Country	Class	Subclass	Translation Yes	Translation No

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

NV	1	The Patent Office of the People's Republic of China, "Notification of the First Office Action," mailed in related Chinese Application No. 02803882.7 on January 27, 2006, 14 pages, including translation.

Examiner	Date Considered 07/28/06

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 1 of 1

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	
(Form PTO-1449)	**Filing Date**	**Art Group**
		2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
NR	1	5,043,940	8/27/1991	Harari			
	2	5,172,338	12/15/1992	Mehrotra et al.			
	3	5,341,330	8/23/1994	Wells et al.			
	4	5,404,485	4/4/1995	Ban			
	5	5,457,658	10/10/1995	Niijima et al.			
	6	5,479,638	12/26/1995	Assar et al.			
	7	5,481,691	1/2/1996	Day, III et al.			
	8	5,485,595	1/16/1996	Assar et al.			
	9	5,598,370	1/28/1997	Niijima			
	10	5,648,929	7/15/1997	Miyamoto			
	11	5,649,200	7/15/1997	Leblang et al.			
	12	5,835,935	11/10/1998	Estakhri et al.			
	13	5,838,614	11/17/1998	Estakhri et al.			
	14	5,845,313	12/1/1998	Estakhri et al.			
	15	5,860,090	1/12/1999	Clark			
	16	5,890,192	3/30/1999	Lee et al.			
	17	5,896,393	4/20/1999	Yard et al.			
	18	5,907,856	5/25/1999	Estakhri et al.			
	19	5,986,933	11/16/1999	Takeuchi et al.			
	20	5,987,563	11/16/1999	Itoh et al.			
	21	5,999,947	12/7/1999	Zollinger et al.			
	22	6,034,897	3/7/2000	Estakhri et al.			
	23	6,040,997	3/21/2000	Estakhri et al.			
	24	6,115,785	9/5/2000	Estakhri et al.			
	25	6,122,195	9/19/2000	Estakhri et al.			
	26	6,125,435	9/26/2000	Estakhri et al.			
	27	6,134,151	10/17/2000	Estakhri et al.			
	28	6,151,247	11/21/2000	Estakhri et al.			
	29	6,161,163	12/12/2000	Komatsu et al.			
NR	30	6,219,768 B1	4/17/2001	Hirabayashi et al.			

Sheet 1 of 2
Express Mail No.: EV663653274US

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	
(Form PTO-1449)	**Filing Date**	**Art Group**
		2189

U.S. Patent Documents (continued)

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
NP	31	6,330,634	12/11/2001	Fuse et al.			
	32	6,426,893	7/30/2002	Conley et al.			
	33	6,449,625	9/10/2002	Wang			
	34	6,567,307	5/20/2003	Estakhri			
NP	35	6,763,424	7/13/2004	Conley			

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No
NP	36	WO 94/22075	9/29/1994	WIPO			X	
	37	WO 94/20906	9/15/1994	WIPO			X	
	38	WO 02/49309A2	20/06/2002	WIPO			X	
NP	39	FR 2,742,893	20/12/1995	France			Abstract	X
	40	~~WO 02/058074A2~~	~~7/25/2002~~	~~WIPO~~			~~X~~	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

NP	41	European Patent Office, "Communication Pursuant to Article 96(2) EPC" mailed in related European patent application no. 02703078.2-2210 on December 2, 2004, 3 pages.
	42	"International Preliminary Examination Report", European Patent Office, Corresponding Application PCT/US02/00366, July 4, 2003, 7 pages.
	43	"FTL Updates, PCMCIA Document Number 0165", Personal Computer Memory Card International Association, Version 002, Release 003, March 4, 1996, pp. 1-26.
	44	Mergi, Aryeh and Schneider, Robert, "M-Systems & SCM Flash Filing Software Flash Translation Layer - FTL", PCMCIA, July 1994, 15 pages.
	45	Petro Estakhri et al., "Moving Sectors Within a Block of Information in a Flash Memory Mass Storage Architecture", U.S. Patent Application No. 09/620,544, filing date July 21, 2000, 48 pages.
NP	46	PCT International Search Report, European Patent Office, corresponding PCT Application No. PCT/US02/00366, 7/4/2003, 5 pages.

Examiner	Date Considered 07/28/06

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 2 of 2
Express Mail No.: EV663653274US

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

Bib Data Sheet

CONFIRMATION NO. 7727

SERIAL NUMBER 11/250,238	**FILING OR 371(c) DATE** 10/13/2005 **RULE**	**CLASS** 711	**GROUP ART UNIT** 2189	**ATTORNEY DOCKET NO.** SNDK.156US2

APPLICANTS

Kevin M. Conley, San Jose, CA;

** **CONTINUING DATA** *************************

This application is a CON of 10/841,388 05/07/2004 PAT 6,968,421 which is a CON of 09/766,436 01/19/2001 PAT 6,763,424

** **FOREIGN APPLICATIONS** ********************

IF REQUIRED, FOREIGN FILING LICENSE GRANTED
** 11/03/2005

Foreign Priority claimed ☐ yes ☒ no 35 USC 119 (a-d) conditions met ☐ yes ☒ no ☐ Met after Allowance Verified and Acknowledged ________ Examiner's Signature ________ Initials	**STATE OR COUNTRY** CA	**SHEETS DRAWING** 9	**TOTAL CLAIMS** 3	**INDEPENDENT CLAIMS** 2

ADDRESS
36257

TITLE

Partial block data programming and reading operations in a non-volatile memory

FILING FEE RECEIVED 1000	FEES: Authority has been given in Paper No. _________ to charge/credit DEPOSIT ACCOUNT No. _________ for following:	☐ All Fees ☐ 1.16 Fees (Filing) ☐ 1.17 Fees (Processing Ext. of time) ☐ 1.18 Fees (Issue) ☐ Other ____________ ☐ Credit

Index of Claims	**Application/Control No.**	**Applicant(s)/Patent under Reexamination**
	11/250,238	CONLEY, KEVIN M.
	Examiner	**Art Unit**
	NGOC V. DINH	2189

√	**Rejected**	–	**(Through numeral) Cancelled**	**N**	**Non-Elected**	**A**	**Appeal**
=	**Allowed**	÷	**Restricted**	**I**	**Interference**	**O**	**Objected**

Claim		Date								
Final	Original	07/28/06								
	1	✓								
	2	✓								
	3	✓								
	4									
	5									
	6									
	7									
	8									
	9									
	10									
	11									
	12									
	13									
	14									
	15									
	16									
	17									
	18									
	19									
	20									
	21									
	22									
	23									
	24									
	25									
	26									
	27									
	28									
	29									
	30									
	31									
	32									
	33									
	34									
	35									
	36									
	37									
	38									
	39									
	40									
	41									
	42									
	43									
	44									
	45									
	46									
	47									
	48									
	49									
	50									

Claim		Date								
Final	Original									
	51									
	52									
	53									
	54									
	55									
	56									
	57									
	58									
	59									
	60									
	61									
	62									
	63									
	64									
	65									
	66									
	67									
	68									
	69									
	70									
	71									
	72									
	73									
	74									
	75									
	76									
	77									
	78									
	79									
	80									
	81									
	82									
	83									
	84									
	85									
	86									
	87									
	88									
	89									
	90									
	91									
	92									
	93									
	94									
	95									
	96									
	97									
	98									
	99									
	100									

Claim		Date								
Final	Original									
	101									
	102									
	103									
	104									
	105									
	106									
	107									
	108									
	109									
	110									
	111									
	112									
	113									
	114									
	115									
	116									
	117									
	118									
	119									
	120									
	121									
	122									
	123									
	124									
	125									
	126									
	127									
	128									
	129									
	130									
	131									
	132									
	133									
	134									
	135									
	136									
	137									
	138									
	139									
	140									
	141									
	142									
	143									
	144									
	145									
	146									
	147									
	148									
	149									
	150									

Search Notes	**Application/Control No.**	**Applicant(s)/Patent under Reexamination**	
	11/250,238	CONLEY, KEVIN M.	
	Examiner	**Art Unit**	
	NGOC V. DINH	2189	

SEARCHED			
Class	Subclass	Date	Examiner

INTERFERENCE SEARCHED			
Class	Subclass	Date	Examiner

SEARCH NOTES (INCLUDING SEARCH STRATEGY)		
	DATE	EXMR
Limited classified search of711/103, 102, 202, 203, 209 class's/sub's in "SEARCHED" section.	7/28/2006	ND
EAST text search w/o classified/search. See prinout		
Inventor Search		

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	PATENT NUMBER	GROUP ART UNIT	FILE WRAPPER LOCATION
11/250,238		2189	37M1

Correspondence Address / Fee Address Change

The following fields have been set to Customer Number 66785 on 12/01/2006

- Correspondence Address
- Maintenance Fee Address

The address of record for Customer Number 66785 is:
PARSONS HSUE & DE RUNTZ, LLP - SANDISK CORPORATION
595 MARKET STREET
SUITE 1900
SAN FRANCISCO,CA 94105

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT

Dear Sir:

Pursuant to 37 C.F.R. §§ 1.56, 1.97 and 1.98, Applicants call the documents listed on the enclosed Form PTO-1449 to the Examiner's attention in this patent application.

According to 37 C.F.R. 1.98(2)(ii), copies of the U.S. Patents and U.S. Published Patent Applications documents are not required and are therefore not enclosed. A copy of the listed foreign patent document is enclosed.

Citation of these documents shall not be construed as (1) an admission that the documents are prior art with respect to the invention or inventions claimed in this application, (2) a representation that a search has been made (other than as indicated by any cited document), or

(3) an admission that the cited information is, or is considered to be, material to patentability as defined in § 1.56(b).

This information disclosure statement is submitted under 37 C.F.R. § 1.97(c). The fee of $180.00 has been authorized via EFS to Deposit Account 502664. The Commissioner is hereby authorized to charge any additional fees, which may be required, or credit any overpayment to Deposit Account 502664.

FILED VIA EFS

Respectfully submitted,

January 8, 2007

Gerald P. Parsons
Date
Reg. No. 24,486

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
	1	6,968,421 B2	11/22/2005	Conley			

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No
	2	WO 02/058074A2	7/25/2002	WIPO			X	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

Examiner	Date Considered

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 1 of 1
Express Mail No.: EV663653274US

Electronic Patent Application Fee Transmittal

Application Number:	11250238
Filing Date:	13-Oct-2005
Title of Invention:	Partial block data programming and reading operations in a non-volatile memory
First Named Inventor/Applicant Name:	Kevin M. Conley
Filer:	Gerald Paul Parsons/Mary Buggie (GPP)
Attorney Docket Number:	SNDK.156US2

Filed as Large Entity

Utility Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Statutory disclaimer	1814	1	130	130
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension - 2 months with $0 paid	1252	1	450	450
Miscellaneous:				
Submission- Information Disclosure Stmt	1806	1	180	180
		Total in USD ($)		**760**

Electronic Acknowledgement Receipt

EFS ID:	1422795
Application Number:	11250238
International Application Number:	
Confirmation Number:	7727
Title of Invention:	Partial block data programming and reading operations in a non-volatile memory
First Named Inventor/Applicant Name:	Kevin M. Conley
Customer Number:	66785
Filer:	Gerald Paul Parsons/Mary Buggie (GPP)
Filer Authorized By:	Gerald Paul Parsons
Attorney Docket Number:	SNDK.156US2
Receipt Date:	08-JAN-2007
Filing Date:	13-OCT-2005
Time Stamp:	14:35:24
Application Type:	Utility

Payment information:

Submitted with Payment	yes
Payment was successfully received in RAM	$760
RAM confirmation Number	1755
Deposit Account	502664

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)	Multi Part /.zip	Pages (if appl.)

1		SNDK156US2-Trans_Resp_ Term_Ext_IDS.pdf	590108	yes	15

Multipart Description/PDF files in .zip description

Document Description	Start	End
Miscellaneous Incoming Letter	1	1
Applicant Arguments/Remarks Made in an Amendment	2	2
Claims	3	5
Specification	6	6
Applicant Arguments/Remarks Made in an Amendment	7	9
Terminal Disclaimer Filed	10	11
Extension of Time	12	12
Information Disclosure Statement (IDS) Filed	13	15

Warnings:

Information:

2	Fee Worksheet (PTO-06)	fee-info.pdf	8484	no	2

Warnings:

Information:

Total Files Size (in bytes):	598592

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

PARSONS HSUE & DE RUNTZ LLP

595 Market Street, Suite 1900 San Francisco, CA 94105
tel 415.318.1160 fax 415.693.0194

January 8, 2007

Mail Stop Amendment
Commissioner For Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Customer No. 66785

Re:	Applicant:	Kevin M. Conley		
	Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
	Application No.:	11/250,238	Filing Date:	October 13, 2005
	Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
	Docket No.:	SNDK.156US2	Conf. No.:	7727

Dear Sir:

Transmitted herewith are the following documents in the above-identified application:

(1) This Transmittal Letter;
(2) Response to Office Action and Amendment (8 pages);
(3) Terminal Disclaimer (2 pages);
(4) Petition for Extension of Time (1 page);
(5) Supplemental Information Disclosure Statement and PTO Form 1449 (3 pages);
(6) Copy of 1 cited reference

☒ The fee has been calculated as shown below:

CLAIMS AS AMENDED

	Claims Remaining *After* Amendment		Highest No. Previously Paid For		Present Extra		Rate		Additional Fee
Total Claims	15	Minus	20	=	0	x	$50.00	$	0.00
Independent Claims	2	Minus	3	=	0	x	$200.00	$	0.00
☐ Fee of _____ for the first filing of one or more multiple dependent claims per application								$	
☒ Fee for Terminal Disclaimer								$	130.00
☒ Fee for Petition for Extension of Time								$	450.00
☒ Fee for Information Disclosure Statement								$	180.00
Total additional fee for this Amendment:								$	760.00

☒ Conditional Petition for Extension of Time: If an extension of time is required for timely filing of the enclosed document(s) after all papers filed with this transmittal have been considered, an extension of time is hereby requested.

☒ The fee of $760.00 has been authorized via EFS to Deposit Account 502664. The Commissioner is hereby authorized to charge any additional fees, which may be required, or credit any overpayment to Deposit Account 502664.

FILED VIA EFS

Respectfully submitted,

Gerald P. Parsons
Reg. No. 24,486

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RESPONSE TO OFFICE ACTION AND AMENDMENT

Sir:

This is in response to the non-final Office Action dated August 8, 2006.

Claim Amendments are reflected in the listing of claims, which begins on page 2 of this paper.

A Specification Amendment is on page 5 of this paper.

Remarks begin on page 6 of this paper.

Reconsideration is kindly requested in light of the following amendments and remarks.

CLAIM AMENDMENTS

Please amend the claims by canceling claims 1-3, without prejudice, and adding new claims 4-18, as indicated on the following listing of all the claims in the present application after this Amendment:

1. - 3. (Cancelled)

4. (New) A method of operating a memory system having an array of reprogrammable non-volatile charge storage elements organized in blocks of storage elements that are erasable together and in pages of storage elements within the blocks that are individually programmable as a unit, comprising:

as part of writing data into pages, recording an indication of a time that data are written into individual pages,

when updating data previously written into one or more initial pages, writing the updated data into one or more update pages and identify the initial and update data pages by the same logical addresses, and

when reading data of two or more pages having the same logical addresses, read the indications of the times that data have been stored in the two or more pages and use the data in the two or more pages having more recent time indications without using data in the two or more pages having older time indications.

5. (New) The method of claim 4, wherein recording an indication of a time that data are written into individual pages includes recording a value of a clock within the memory system.

6. (New) The method of claim 4, wherein recording an indication of a time that data are written into individual pages includes recording a different value of a sequence of numbers.

7. (New) The method of claim 4, wherein recording an indication of a time that data are written into individual pages includes recording the indication within the pages wherein the data are written.

8. (New) The method of claim 4, wherein updating data previously written into one or more initial pages is limited to updating data in a number of pages less than all of the pages of a block while not updating data in a remaining one or more pages of the same block.

9. (New) The method of claim 8, wherein the memory system in which the method is carried out utilizes electrically conductive floating gates as the charge storage elements.

10. (New) The method of claim 4, wherein as part of writing data into pages, logical addresses of the individual pages in which the data are written are also written in the individual pages.

11. (New) The method of claim 4, wherein as part of writing data into pages, data are written in individual storage elements of the pages with more than two storage states, thereby storing more than one bit of data in the individual storage elements.

12. (New) The method of claim 4, wherein the memory system in which the method is carried out utilizes electrically conductive floating gates as the charge storage elements.

13. (New) A method of operating a memory system having an array of reprogrammable non-volatile charge storage elements organized in blocks of storage elements that are erasable together and in pages of storage elements within the blocks that are individually programmable as a unit, comprising:

as part of writing data into pages, data are written into the pages of the blocks in sequence, updating data previously written into one or more initial pages by writing the updated data into one or more update pages and identify the initial and update data pages by the same logical addresses, and

reading data from the pages of the blocks in an order that is a reverse of the sequence in which they where written and ignore data in any page having the same logical address as a page from which data have already been read.

14. (New) The method of claim 13, wherein updating data previously written into one or more initial pages is limited to updating data in a number of pages less than all of the pages of a block while not updating data in a remaining one or more pages of the same block.

15. (New) The method of claim 14, wherein the memory system in which the method is carried out utilizes electrically conductive floating gates as the charge storage elements.

16. (New) The method of claim 13, wherein as part of writing data into pages, logical addresses of the individual pages in which the data are written are also written in the individual pages.

17. (New) The method of claim 13, wherein as part of writing data into pages, data are written in individual storage elements of the pages with more than two storage states, thereby storing more than one bit of data in the individual storage elements.

18. (New) The method of claim 13, wherein the memory system in which the method is carried out utilizes electrically conductive floating gates as the charge storage elements.

SPECIFICATION AMENDMENT

Please amend the Specification as follows:

[0001] This application is a continuation of application serial no. 10/841,388, filed May 7, 2004, now patent no. 6,968,421, which in turn is a continuation of application serial no. 09/766,436, filed January 19, 2001, now patent no. 6,763,424, which applications are incorporated herein in their entirety by this reference.

REMARKS

The original claims 1-3 of the present continuation application have been cancelled and replaced by new claims 4-18.

Double Patenting Rejection

Claims 1-3 were rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1-3 of U.S. Patent No. 6,968,421 to Conley ("Conley"). In order to facilitate the prosecution of the present application with the new claims, a Terminal Disclaimer is being filed herewith in order to overcome this ground of rejection.

Claim Rejections Under 35 U.S.C. §102

Claims 1-3 were rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,330,634 to Fuse et al. ("Fuse"). It is noted that claims in the two parent applications, now patents nos. 6,763,424 and 6,968,421, were also initially rejected over Fuse. It is submitted that new independent claims 4 and 13 are also allowable over Fuse. These claims are more specific in several ways from the initial claims 1-3 of the present continuation application. In particular, each of new claims 4 and 13 recite, in their last paragraphs, specifics of reading the data earlier stated to be programmed in a particular way.

New claims 4-18 are directed to a method of operating a memory by individually managing pages of storage capacity within erase blocks of data storage elements. Fuse describes a different operation of a similar type of memory system. Its memory cells are organized into erase blocks and pages within the blocks but nothing is found in Fuse that suggests the management of pages of data, rather than entire blocks. In two embodiments described by Fuse, one for boot data and the other for user data, an order of the programming of original and updated data are maintained at the block level, but not for individual pages within the blocks.

The Fuse reference does not address the problem of how to operate a memory system to update data in some pages of a block but not others. Indeed, the memory operation described in the Fuse reference seems to have significantly different goals, namely either to identify the most recent copy of boot data or to avoid the loss of user data being rewritten during a power failure. In the later case, the latest programmed blocks of data are not used for fear of data corruption but rather the original user data are maintained as valid. This is the reverse of the methods recited in the new claims 4-18, where the most recent data are read out of the memory without using the original data that have been updated.

The method of new independent claim 4 comprises recording the times that data are written into individual pages within erase blocks and storing updated data in pages that are given the same logical address as other pages containing the initial data that are being updated. Use is then specified to be made of those times during reading to assemble updated pages of valid data without using obsolete data in other pages having the same logical addresses. In Fuse, on the other hand, an identification number is stored in each block to indicate its relative time of programming, not in each page. (See, for example, Figure 6, showing that the identification number is maintained along with other block administrative information at the end of the last page of each block.) Further, Fuse's identification number is used to identify original data for reading, in case of a power failure, instead of identifying the most recent updated data for reading, as recited in all the new claims.

Although several of the dependent claims 5-12 add other features of novelty over Fuse, it appears to be sufficient to point out here that these claims are patentable for the same reasons as set forth above for claim 4.

The method of new claim 13 comprises reading pages of data in a reverse order from that in which they are written and ignoring data in any page having the same logical address as a page from which data were previously read. In addition to lacking disclosure of keeping track of pages of memory cells instead of blocks, nothing is found in Fuse that describes such a reverse order read in order to identify and utilize the most recent updated data.

Although several of the dependent claims 14-18 add other features of novelty over Fuse, it appears to be sufficient to point out here that these claims are patentable for the same reasons as set forth above for claim 13.

Information Disclosure Statement

A Supplemental Information Disclosure Statement is being filed herewith which includes the reference WO 02/058074 A2 not previously considered because a copy was not included. As stated in the Information Disclosure Statement filed October 13, 2005 that cited this reference, a copy was not included since it was cited and available in the parent case. However, a copy is now being provided for the Examiner's convenience. Parent patent no. 6,968,421 is also included in the Statement. It is respectfully requested that this Supplemental Information Disclosure Statement be considered and the PTO Form 1449 be initialed and returned with the next Action, in order to formally make these references of record and cause them to be printed on the patent granted from the present application.

Conclusion

Accordingly, it is believed that this application is now in condition for allowance and an early indication of its allowance is solicited. However, if the Examiner has any further matters that need to be resolved, a telephone call to the undersigned at 415-318-1163 would be appreciated.

FILED VIA EFS

Respectfully submitted,

January 8, 2007

Gerald P. Parsons
Reg. No. 24,486

Date

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TERMINAL DISCLAIMER UNDER 37 C.F.R. § 1.321(c)

Sir:

SanDisk Corporation ("the owner") owns the entire interest in and to both the above-identified continuation application ("instant application") and parent patent no. 6,968,421 ("prior patent") by way of a written assignment from the inventor of parent patent application Serial No. 09/766,436, filed January 19, 2001.

The owner hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application that would extend beyond the expiration date of the full statutory term defined in 35 U.S.C. §§ 154, 155, 156 and 173, as presently shortened by any terminal disclaimer, of the prior patent, if any such extended term would otherwise exist. The owner further agrees that any such patent granted on the instant application shall be enforceable only for and during such period that such patent and the prior patent are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the owner, its successors and assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of any patent granted on the instant application that would extend to the expiration date of the full statutory term as defined in 35 U.S.C. §§ 154, 155, 156 and 173 of the prior patent, as presently shortened by any terminal disclaimer, in the event that the prior patent later expires for failure to pay a maintenance fee, is held unenforceable, is found invalid by a court of competent jurisdiction, is statutorily disclaimed in whole or terminally disclaimed under 37 CFR § 1.321, has all claims canceled by a reexamination certificate, is reissued, or is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

The fee under 37 C.F.R. § 1.20(d) of $130.00 is being filed herewith. Please charge any additional fees required or credit any overpayment to our Deposit Account No. 502664.

FILED VIA EFS

Respectfully submitted,

January 8, 2007

Gerald P. Parsons
Reg. No. 24,486

Date

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

PETITION FOR EXTENSION OF TIME

Dear Sir:

Applicant respectfully petitions for a two-month extension of time within which to respond to the August 8, 2006 outstanding Office Action, such extension allowing the undersigned until January 8, 2006 to respond. The fee of $450.00 has been authorized via EFS to Deposit Account 502664. The Commissioner is hereby authorized to charge any additional fees, which may be required, or credit any overpayment to Deposit Account 502664.

FILED VIA EFS

Respectfully submitted,

Gerald P. Parsons
Reg. No. 24,486

January 8, 2007
Date

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 July 2002 (25.07.2002)

PCT

(10) International Publication Number
WO 02/058074 A2

(51) International Patent Classification⁷: **G11C 16/00**

(21) International Application Number: PCT/US02/00366

(22) International Filing Date: 7 January 2002 (07.01.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/766,436 19 January 2001 (19.01.2001) US

(71) Applicant: SANDISK CORPORATION [US/US]; 140 Caspian Court, Sunnyvale, CA 94089 (US).

(72) Inventor: CONLEY, Kevin, M.; 5983 Alvarado Court, San Jose, CA 95120 (US).

(74) Agent: PARSONS, Gerald, P.; Skjerven Morrill Macpherson LLP, Three Embarcadero Center, 28th Floor, San Francisco, CA 94111 (US).

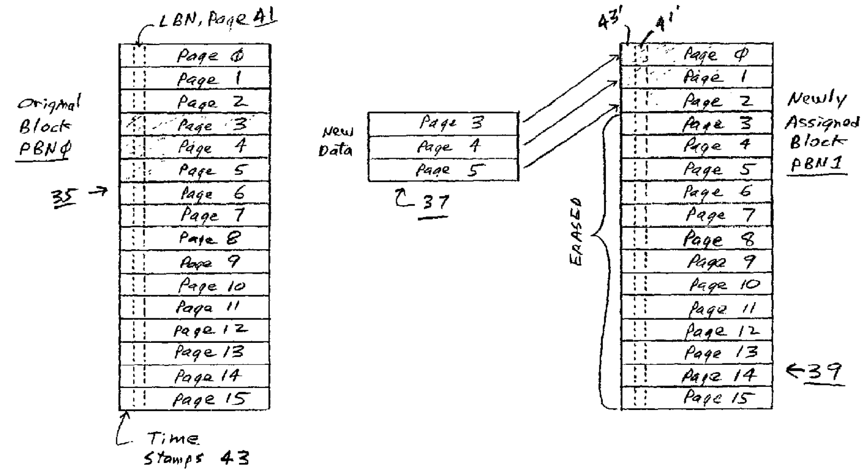
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (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*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY



(57) Abstract: Data in less than all of the pages of a non-volatile memory block are updated by programming the new data in unused pages of either the same or another block. In order to prevent having to copy unchanged pages of data into the new block, or to program flags into superceded pages of data, the pages of new data are identified by the same logical address as the pages of data which they superceded and a time stamp is added to note when each page was written. When reading the data, the most recent pages of data are used and the older superceded pages of data are ignored. This technique is also applied to metablocks that include one block from each of several different units of a memory array, by directing all page updates to a single unused block in one of the units.

WO 02/058074 A2

PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

BACKGROUND OF THE INVENTION

This invention pertains to the field of semiconductor non-volatile data storage system architectures and their methods of operation, and has application to data storage systems based on flash electrically erasable and programmable read-only memories (EEPROMs).

A common application of flash EEPROM devices is as a mass data storage subsystem for electronic devices. Such subsystems are commonly implemented as either removable memory cards that can be inserted into multiple host systems or as non-removable embedded storage within the host system. In both implementations, the subsystem includes one or more flash devices and often a subsystem controller.

Flash EEPROM devices are composed of one or more arrays of transistor cells, each cell capable of non-volatile storage of one or more bits of data. Thus flash memory does not require power to retain the data programmed therein. Once programmed however, a cell must be erased before it can be reprogrammed with a new data value. These arrays of cells are partitioned into groups to provide for efficient implementation of read, program and erase functions. A typical flash memory architecture for mass storage arranges large groups of cells into erasable blocks, wherein a block contains the smallest number of cells (unit of erase) that are erasable at one time.

In one commercial form, each block contains enough cells to store one sector of user data plus some overhead data related to the user data and/or to the block in which it is stored. The amount of user data included in a sector is the standard 512 bytes in one class of such memory systems but can be of some other size. Because the isolation of individual blocks of cells from one another that is required to make them individually erasable takes space on the integrated circuit chip, another class of flash memories makes the blocks significantly larger so there is less space required for such isolation. But since it is also desired to handle user data in much smaller sectors, each large block is often further partitioned into individually addressable pages that are the basic unit for reading and programming user data (unit of programming and/or reading). Each page usually stores one sector of user data, but a page may store a partial sector or multiple sectors. A "sector" is used herein to refer to an amount of user data that is transferred to and from the host as a unit.

The subsystem controller in a large block system performs a number of functions including the translation between logical addresses (LBAs) received by the memory sub-system from a host, and physical block numbers (PBNs) and page addresses within the memory cell array. This translation often involves use of intermediate terms for a logical block number (LBN) and logical page. The controller also manages the low level flash circuit operation through a series of commands that it issues to the flash memory devices via an interface bus. Another function the controller performs is to maintain the integrity of data stored to the subsystem through various means, such as by using an error correction code (ECC).

In an ideal case, the data in all the pages of a block are usually updated together by writing the updated data to the pages within an unassigned, erased block, and a logical-to-physical block number table is updated with the new address The original block is then available to be erased. However, it is more typical that the data stored in a number of pages less than all of the pages within a given block must be updated. The data stored in the remaining pages of the given block remains unchanged. The probability of this occurring is higher in systems where the number of sectors of data stored per block is higher. One technique now used to accomplish such a partial block update is to write the data of the pages to be updated into a corresponding number of the pages of an unused erased block and then copy the unchanged pages from the original block into pages of the new block. The original block may then be erased and added to an inventory of unused blocks in which data may later be programmed. Another technique similarly writes the updated pages to a new block but eliminates the need to copy the other pages of data into the new block by changing the flags of the pages in the original block which are being updated to indicate they contain obsolete data. Then when the data are read, the updated data read from pages of the new block are combined with the unchanged data read from pages of the original block that are not flagged as obsolete.

SUMMARY OF THE INVENTION

According to one principal aspect of the present invention, briefly and generally, both the copying of unchanged data from the original to the new blocks and the need to update flags within the original block are avoided when the data of fewer than all of the pages within a block are being updated. This is accomplished by maintaining both the superceded data pages and the updated pages of data with a common logical address. The original and updated pages of data are then distinguished by the relative order in which they were programmed. During reading, the most recent data stored in the pages having the same logical address are combined with the unchanged pages of data while data in the original versions of the updated pages are ignored. The updated data can be written to either pages within a different block than the original data, or to available unused pages within the same block. In one specific implementation, a form of time stamp is stored with each page of data that allows determining the relative order that pages with the same logical address were written. In another specific implementation, in a system where pages are programmed in a particular order within the blocks, a form of time stamp is stored with each block of data, and the most recent copy of a page within a block is established by its physical location within the block.

These techniques avoid both the necessity for copying unchanged data from the original to new block and the need to change a flag or other data in the pages of the original block whose data have been updated. By not having to change a flag or other data in the superceded pages, a potential of disturbing the previously written data in adjacent pages of that same block that can occur from such a writing operation is eliminated. Also, a performance penalty of the additional program operation is avoided.

A further operational feature, which may be used in conjunction with the above summarized techniques, keeps track of the logical offset of individual pages of data within the individual memory cell blocks, so that the updated data need not be stored with the same physical page offset as the superceded data. This allows more efficient use of the pages of new blocks, and even allows the updated data to be stored in any erased pages of the same block as the superceded data.

Another principal aspect of the present invention groups together two or more blocks positioned in separate units of the memory array (also termed "sub-arrays") for programming and reading together as part of a single operation. Such a multiple block group is referenced herein as a "metablock." Its component blocks may be either all located on a single memory integrated circuit chip, or, in systems using more than one such chip, located on two or more different chips. When data in fewer than all of the pages of one of these blocks is updated, the use of another block in that same unit is normally required. Indeed, the techniques described above, or others, may be employed separately with each block of the metablock. Therefore, when data within pages of more than one block of the metablock are updated, pages within more than one additional block are required to be used. If there are four blocks of four different memory units that form the metablock, for example, there is some probability that up to an additional four blocks, one in each of the units, will be used to store updated pages of the original blocks. One update block is potentially required in each unit for each block of the original metablock. In addition, according to the present invention, updated data from pages of more than one of the blocks in the metablock can be stored in pages of a common block in only one of the units. This significantly reduces the number of unused erased blocks that are needed to store updated data, thereby making more efficient use of the available memory cell blocks to store data. This technique is particularly useful when the memory system frequently updates single pages from a metablock.

Additional aspects, features and advantages of the present invention are included in the following description of exemplary embodiments, which description should be read in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of a typical prior art flash EEPROM memory array with memory control logic, data and address registers;

Figure 2 illustrates an architecture utilizing memories of Figure 1 with a system controller;

Figure 3 is a timing diagram showing a typical copy operation of the memory system of Figure 2;

Figure 4 illustrates an existing process of updating data in less than all of the pages of a multi-paged block;

Figures 5A and 5B are tables of corresponding logical and physical block addresses for each of the original and new blocks of Figure 4, respectively;

Figure 6 illustrates another existing process of updating data in less than all of the pages of a multi-paged block;

Figures 7A and 7B are tables of corresponding logical and physical page addresses for the original and new blocks of Figure 6, respectively;

Figure 8 illustrates an example of an improved process of updating data in less than all of the pages of a multi-paged block;

Figure 9 is a table of corresponding logical and physical page numbers for the new block of Figure 8;

Figure 10 provides an example of a layout of the data in a page shown in Figure 8;

Figure 11 illustrates a further development of the example of Figure 8;

Figure 12 is a table of corresponding logical and physical page numbers for the new block of Figure 11;

Figure 13 illustrates one way to read the updated data in the blocks of Figure 11;

Figure 14 is a flow diagram of a process of programming data into a memory system organized as illustrated in Figures 8 and 9; Figure 15 illustrates an existing multi-unit memory with blocks from the individual units being linked together into a metablock and

Figure 16 illustrates an improved method of updating data of a metablock in the multi-unit memory of Figure 12 when the amount of updated data is much less that the data storage capacity of the metablock.

DESCRIPTION OF EXISTING LARGE BLOCK MANAGEMENT TECHNIQUES

Figure 1 shows a typical flash memory device internal architecture. The primary features include an input/output (I/O) bus 411 and control signals 412 to interface to an external controller, a memory control circuit 450 to control internal memory operations with registers for command, address and status signals. One or more arrays 400 of flash EEPROM cells are included, each array having its own row decoder (XDEC) 401 and column decoder (YDEC) 402, a group of sense amplifiers and program control circuitry (SA/PROG) 454 and a data register 404. Presently, the memory cells usually include one or more conductive floating gates as storage elements but other long term electron charge storage elements may be used instead. The memory cell array may be operated with two levels of charge defined for each storage element to therefore store one bit of data with each element. Alternatively, more than two storage states may be defined for each storage element, in which case more than one bit of data is stored in each element.

If desired, a plurality of arrays 400, together with related X decoders, Y decoders, program/verified circuitry, data registers, and the like are provided, for example as taught by U.S. Patent 5,890,192, issued March 30, 1999, and assigned to Sandisk Corporation, the assignee of this application, which is hereby incorporated by this reference. Related memory system features are described in co-pending patent application serial no. 09/505,555, filed February 17, 2000 by Kevin Conley et al., which application is expressly incorporated herein by this reference.

The external interface I/O bus 411 and control signals 412 can include the following:

CS - Chip Select.	Used to activate flash memory interface.
RS - Read Strobe.	Used to indicate the I/O bus is being used to transfer data from the memory array.
WS - Write Strobe.	Used to indicate the I/O bus is being used to transfer data to the memory array.
AS - Address Strobe.	Indicates that the I/O bus is being used to transfer address information.
AD[7:0] - Address/Data Bus	This I/O bus is used to transfer data between controller and the flash memory command, address and data registers of the memory control 450.

This interface is given only as an example as other signal configurations can be used to give the same functionality. Figure 1 shows only one flash memory array 400 with its related components, but a multiplicity of such arrays can exist on a single flash memory chip that share a common interface and memory control circuitry but have separate XDEC, YDEC, SA/PROG and DATA REG circuitry in order to allow parallel read and program operations.

Data is transferred from the memory array through the data register 404 to an external controller via the data registers' coupling to the I/O bus AD[7:0] 411. The data register 404 is also coupled the sense amplifier/programming circuit 454. The number of elements of the data register coupled to each sense amplifier/programming circuit element may depend on the number of bits stored in each storage element of the memory cells, flash EEPROM cells each containing one or more floating gates as the storage elements. Each storage element may store a plurality of bits, such as 2 or 4, if the memory cells are operated in a multi-state mode. Alternatively, the memory cells may be operated in a binary mode to store one bit of data per storage element.

The row decoder 401 decodes row addresses for the array 400 in order to select the physical page to be accessed. The row decoder 401 receives row addresses via internal row address lines 419 from the memory control logic 450. A column decoder 402 receives column addresses via internal column address lines 429 from the memory control logic 450.

Figure 2 shows an architecture of a typical non-volatile data storage system, in this case employing flash memory cells as the storage media. In one form, this system is encapsulated within a removable card having an electrical connector extending along one side to provide the host interface when inserted into a receptacle of a host. Alternatively, the system of Figure 2 may be embedded into a host system in the form of a permanently installed embedded circuit or otherwise. The system utilizes a single controller 301 that performs high level host and memory control functions. The flash memory media is composed of one or more flash memory devices, each such device often formed on its own integrated circuit chip. The system controller and the flash memory are connected by a bus 302 that allows the controller 301 to load command, address, and transfer data to and from the flash memory array. The controller 301 interfaces with a host system (not shown) with which user data is transferred to and from the flash memory array. In the case where the system of Figure 2 is included in a card, the host interface includes a mating plug and socket assembly (not shown) on the card and host equipment.

The controller 301 receives a command from the host to read or write one or more sectors of user data starting at a particular logical address. This address may or may not align with a boundary of a physical block of memory cells.

In some prior art systems having large capacity memory cell blocks that are divided into multiple pages, as discussed above, the data from a block that is not being updated needs to be copied from the original block to a new block that also contains the new, updated data being written by the host. This technique is illustrated in Figure 4, wherein two of a large number of blocks of memory are included. One block 11 (PBN0) is illustrated to be divided into 8 pages for storing one sector of user data in each of its pages. Overhead data fields contained within each page include a field 13 containing the LBN of the block 11. The order of the logical pages within a logical block is fixed with respect to the corresponding physical pages within a physical block. A second similarly configured block 15 (PBN1) is selected from an inventory of unused, erased blocks. Data within pages 3-5 of the original block 11 are being updated by three pages of new data 17. The new data is written into the corresponding pages 3-5 of the new block 15, and user data from pages 0-2, 6 and 7 of the block 11 are copied into corresponding pages of the new block 15. All pages of the new block 15 are preferably programmed in a single sequence of programming operations. After the block 15 is programmed, the original block 11 can be erased and placed in inventory for later use. The copying of data between the blocks 11 and 15, which involves reading the data from one or more pages in the original block and subsequently programming the same data to pages in a newly assigned block, greatly reduces the write performance and usable lifetime of the storage system.

With reference to Figures 5A and 5B, partial tables show mapping of the logical blocks into the original and new physical blocks 11 and 15 before (Figure 5A) and after (Figure 5B) the updating of data described with respect to Figure 4. Before the data update, the original block 11, in this example, stores pages 0-7 of LBN0 into corresponding pages 0-7 of PBN0. After the data update, the new block 15 stores pages 0-7 of LBN0 in corresponding pages 0-7 of PBN1. Receipt of a request to read data from LBN0 is then directed to the physical block 15 instead of the physical block 11. In a typical controller operation, a table in the form of that shown in Figures 5A and 5B is built from the LBN field 13 read from a physical page and knowledge of the PBN that is addressed when reading the data field 13. The table is usually stored in a volatile memory of the controller for ease of access, although only a portion of a complete table for the entire system is typically stored at any one time. A portion of the table is usually formed immediately in advance of a read or programming operation that involves the blocks included in the table portion.

In other prior art systems, flags are recorded with the user data in pages and are used to indicate that pages of data in the original block that are being superceded by the newly written data are invalid. Only the new data is written to the newly assigned block. Thus the data in pages of the block not involved in the write operation but contained in the same physical block as the superceded data need not be copied into the new block. This operation is illustrated in Figure 6, where pages 3-5 of data within an original block 21 (PBN0) are again being updated. Updated pages 3-5 of data 23 are written into corresponding pages of a new block 25. As part of the same operation, an old/new flag 27 is written in each of the pages 3-5 to indicate the data of those pages is old, while the flag 27 for the remaining pages 0-2, 6 and 7 remains set at "new". Similarly, the new PBN1 is written into another overhead data field of each of the pages 3-5 in the block 21 to indicate where the updated data are located. The LBN and page are stored in a field 31 within each of the physical pages.

Figures 7A and 7B are tables of the correspondence between the data LBN/page and the PBN/page before (Figure 7A) and after (Figure 7B) the data update is complete. The unchanged pages 0-2, 6 and 7 of the LBN remain mapped into PBN0 while the updated pages 3-5 are shown to reside in PBN1. The table of Figure 7B is built by the memory controller by reading the overhead data fields 27, 29 and 31 of the pages within the block PBN0 after the data update. Since the flag 27 is set to "old" in each of pages 3-5 of the original block PBN0, that block will no longer appear in the table for those pages. Rather, the new block number PBN1 appears instead, having been read from the overhead fields 29' of the updated pages. When data are being read from LBN0, the user data stored in the pages listed in the right column of Figure 7B are read and then assembled in the order shown for transfer to the host.

Various flags are typically located in the same physical page as the other associated overhead data, such as the LBN and an ECC. Thus, to program the old/new flags 27, and others, in pages where the data has been superceded requires that a page support multiple programming cycles. That is, the memory array must have the capability that its pages can be programmed in at least at least two stages between erasures. Furthermore, the block must support the ability to program a page when other pages in the block with higher offsets or addresses have been already programmed. A limitation of some flash memories however prevents the usage of such flags by specifying that the pages in a block can only be programmed in a physically sequential manner. Furthermore, the pages support a finite number of program cycles and in some cases additional programming of programmed pages is not permitted.

What is needed is a mechanism by which data that partially supercedes data stored in an existing block can be written without either copying unchanged data from the existing block or programming flags to pages that have been previously programmed.

DESCRIPTION OF EXEMPLARY EMBODIMENTS OF THE INVENTION

There are many different types of flash EEPROM, each of which presents its own limitations that must be worked around to operate a high performance memory system formed on a small amount of integrated circuit area. Some do not provide for writing any data into a page that has already been programmed, so updating flags in a page that contains superceded data, as described above, is not possible. Others allow such flags to be written but doing so in pages whose data is being superceded can disturb data in other pages of the same block that remain current.

An example memory system where this has been found to be a problem is a NAND type, where a column of memory cells is formed as a series circuit string between a bit line and a common potential. Each word line extends across a row of memory cells formed of one cell in each such string. Such a memory is particularly susceptible to such memory state disturbs when being operated in a multi-state mode to store more than one bit of data in each such cell. Such operation divides an available window of a memory cell transistor threshold voltage range into narrow non-overlapping voltage level ranges, each range becoming narrower as the number of levels, and thus the number of bits being stored in each cell, are increased. For example, if four threshold ranges are used, two bits of data are stored in each cell's storage element. And since each of the four threshold voltage ranges is necessarily small, the chance of the state of a cell being disturbed by programming other cells in the same block is increased with multi-state operation. In this case, the writing of the old/new or other flags, as described with respect to Figures 6, 7A and 7B, cannot be tolerated.

A common feature of each of the existing memory management techniques described above with respect to Figures 4-7B is that a logical block number (LBN) and page offset is mapped within the system to at most two physical block numbers (PBNs). One block is the original block and the other contains the updated page data. Data are written to the page location in the block corresponding to the low order bits of its logical address (LBA). This mapping is typical in various types of memory systems. In the techniques described below, pages containing updated data are also assigned the same LBN and page offsets as the pages whose data has been superceded. But rather than tagging the pages containing original data as being superceded, the memory controller distinguishes the pages containing the superceded data from those containing the new, updated version either (1) by keeping track of the order in which the pages having the same logical addresses were written, such as by use of a counter, and/or (2) from the physical page addresses wherein, when pages are written in order within blocks from the lowest page address to the highest, the higher physical address contains the most recent copy of the data. When the data is accessed for reading, therefore, those in the most current pages are used in cases where there are pages containing superceded data that have the same logical addresses, while the superceded data are ignored.

A first specific implementation of this technique is described with respect to Figures 8 and 9. The situation is the same in this example as that in the prior art techniques described with respect to Figures 4-7B, namely the partial re-write of data within a block 35, although each block is now shown to contain 16 pages. New data 37 for each of the pages 3-5 of the block 35 (PBN 35) is written into three pages of a new block 39 (PBN1) that has previously been erased, similar to that described previously. A LBN and page offset overhead data field 41 written into the pages of PBN1 that contain the updated data is the same as that in the pages of the superceded data in the initial block PBN0. The table of Figure 9, formed from the data within the fields 41 and 41', shows this. The logical LBN and page offsets, in the first column, are mapped into both the first physical block (PBN0), in the second column, and, for the pages that have been updated, also into the second physical block (PBN1) in the third column. The LBN and logical page offsets 41' written into each of the three pages of updated data within the new block PBN1 are the same as those 41 written into each of a corresponding logical page of the original block PBN0.

In order to determine which of two pages having the same LBN and page offset contains the updated data, each page contains another overhead field 43 that provides an indication of its time of programming, at least relative to the time that other pages with the same logical address are programmed. This allows the controller to determine, when reading the data from the memory, the relative ages of the pages of data that are assigned the same logical address.

There are several ways in which the field 43, which contains a form of time stamp, may be written. The most straight forward way is to record in that field, when the data of its associated page is programmed, the output of a real-time clock in the system. Later programmed pages with the same logical address then have a later time recorded in the field 43. But when such a real-time clock is not available in the system, other techniques can be used. One specific technique is to store the output of a modulo-N counter as the value of the field 43. The range of the counter should be one more than the number of pages that are contemplated to be stored with the same logical page number. When updating the data of a particular page in the original block PBN0, for example, the controller first reads the count stored in the field 43 of the page whose data are being updated, increments the count by some amount, such as one, and then writes that incremented count in the new block PBN1 as the field 43'. The counter, upon reaching a count of N+1, rolls over to 0. Since the number of blocks with the same LBN is less than N, there is always a point of discontinuity in the values of stored counts. It is easy then to handle the rollover with normalized to the point of discontinuity.

The controller, when called upon to read the data, easily distinguishes between the new and superceded pages' data by comparing the counts in the fields 43 and 43' of pages having the same LBA and page offset. In response to a need to read the most recent version of a data file, data from the identified new pages are then assembled, along with original pages that have not been updated, into the most recent version of the data file.

It will be noted that, in the example of Figure 8, the new data pages 37 are stored in the first three pages 0-2 of the new block PBN1, rather than in the same pages 3-5 which they replace in the original block PBN0. By keeping track of the individual logical page numbers, the updated data need not necessarily be stored in the same page offset of the new block as that of the old block where superceded data is contained. Page(s) of updated data can also be written to erased pages of the same block as the page of data being superceded.

As a result, there is no constraint presented by the techniques being described that limit which physical page new data can be written into. But the memory system in which these techniques are implemented may present some constraints. For example, one NAND system requires that the pages within the blocks be programmed in sequential order. That means that programming of the middle pages 3-5, as done in the new block 25 (Figure 6), wastes the pages 0-2, which cannot later be programmed. By storing the new data 37 in the first available pages of the new block 39 (Figure 8) in such a restrictive system, the remaining pages 3-7 are available for later use to store other data. Indeed, if the block 39 had other data stored in its pages 0-4 at the time the three pages of new data 37 were being stored, the new data could be stored in the remaining unused pages 5-7. This makes maximum use of the available storage capacity for such a system.

An example of the structure of data stored in an individual page of the blocks of Figure 8 is shown in Figure 10. The largest part is user data 45. An error correction code (ECC) 47 calculated from the user data is also stored in the page. Overhead data 49, including the LBN and page tag 41 (logical page offset), the time stamp 43 and an ECC 51 calculated from the overhead data are also stored in the page. By having an ECC 50 covering the overhead data that is separate from the user data ECC 47, the overhead 49 may be read separately from the user data and evaluated as valid without the need to transfer all of the data stored in the page. Alternatively, however, where the separate reading of the overhead data 49 is not a frequent event, all of the data in the page may be covered by a single ECC in order to reduce the total number of bits of ECC in a page.

A second specific implementation of the inventive technique can also be described with respect to Figure 8. In this example, the time stamp is used only to determine the relative age of the data stored in blocks, while the most recent pages among those that carry the same LBN and page number are determined by their relative physical locations. The time stamp 43 then does not need to be stored as part of each page. Rather, a single time stamp can be recorded for each block, either as part of the block or elsewhere within the non-volatile memory, and is updated each time a page of data is written into the block. Data is then read from pages in an order of descending physical address, starting from the last page of the most recently updated block containing data pages having the same LBN.

In Figure 8, for example, the pages are first read in the new block PBN1 from the last (page 15) to the first (page 0), followed by reading the pages of the original block PBN0 in the same reverse order. Once logical pages 3, 4 and 5 have been read from the new block PBN1, the superceded data in those pages of the original block PBN0 that are identified by the same logical page numbers can be skipped during the reading process. Specifically, physical pages 3, 4 and 5 of the old block PBN0 are skipped during reading, in this example, once the controller determines that their LBN/pages 41 are the same as those of the pages already read from the new block PBN1. This process can increase the speed of reading and reduce the number of overhead bits 49 that need to be stored for each page. Further, when this reverse page reading technique is employed, the table of Figure 9 used by the controller during a reading operation can be simplified into the form of Figures 5A and 5B. Only an identity of those physical blocks containing data of a common logical block and the relative times that the physical blocks were programmed need to be known in order to carry out this efficient reading process.

Figure 11 illustrates an extension of the example of Figure 8 by including a second update to the data originally written in the block PBN0. New data 51 for logical pages 5, 6, 7 and 8 is written to the respective physical pages 3, 4, 5 and 6 of the new block PBN1, along with their LBN and page number. Note, in this example, that the data of logical page 5 is being updated for the second time. During a reading operation that begins from the last page of the new block PBN1, the most recently written logical pages 8, 7, 6 and 5 of the data of interest are first read in that order. Thereafter, it will be noted that the LBN/page overhead field in physical page 2 of PBN1 is the same as that read from the physical page 3, so the user data of page 2 is not read. The physical pages 1 and 0 are then read. Next, the pages of the original block PBN0 are read, beginning with physical page 15. After reading physical pages 15-9, the controller will note that the LBN/page fields of each of pages 8-3 match those of pages whose data has already been read, so the old data need not be read from those pages. The efficiency of the reading process is thus improved. Finally, the original data of physical pages 2-0 are read since that data was not updated.

It will be noted that this example of reading pages in a reverse order efficiently sorts out the new data pages from the superceded data pages because data are written in physical page locations of an erased block in order from page 0 on. This technique is not limited to use with a memory system having such a specific programming constraint, however. So long as the order in which pages are programmed within a given block is known, the data from those pages may be read in the reverse order from which they were written. What is desired is that the most recently programmed pages having a common LBN with others that were earlier programmed be read first, and these are the most recently programmed pages. The most recent versions of updated pages are read first so that the superceded versions may easily be identified thereafter.

A table showing the correspondence between the logical data and physical page addresses for the example of Figure 11 is given in Figure 12. Although there have been two data updates, both are represented by the single column for the second block PBN1. The physical page noted in PBN1 for the logical page 5 is simply changed upon the second update to that page occurring. If the updating involves a third block, then another column is added for that other block. The table of Figure 12, constructed by reading the overhead data from each of the pages in blocks to which data of a common LBN has been written, can be used by the first implementation when the reverse page reading technique is not used. When the reverse page reading technique described above is used, the table of Figure 12 need be built only to identify a correspondence between an LBN and all PBNs containing data of that LBN.

An efficient way to organize pages of data being read from a physical block, where one or more of the pages has been updated, is illustrated by Figure 13. Enough space is provided in a volatile memory of the controller to buffer at least several pages of data at a time, and preferably a full block of data. That is what is shown in Figure 13. Sixteen pages of data, equal to the amount stored in a non-volatile memory block, are stored in the controller memory. Since the pages are most commonly read out of order, each page of data is stored in its proper position with respect to the other pages. For example, in the reverse page read operation of Figure 11, logical page 8 if the first to be read, so it is stored in position 8 of the controller memory, as indicated by the

"1" in a circle. The next is logical page 7, and so forth, until all pages of data desired by the host are read and stored in the controller memory. The entire set of page data is then transferred to the host without having to manipulate the order of the data in the buffer memory. The pages of data have already be organized by writing them to the proper location in the controller memory.

A method of programming a non-volatile memory system that utilizes the techniques described with respect to Figures 8 and 9 is illustrated in the flow chart of Figure 14. Data for pages of an existing file to be updated are received from a host system, as indicated by the block 52. It is first determined by a step 53 whether the number of pages of updated data to be stored is equal to or greater than the storage capacity of a block of the system, 16 pages being shown as the block capacity, for simplicity, in the above described example. If so, one or more unused, erased blocks are addressed, in a step 55, and the new data pages are written to the addressed block(s), in a step 57. Typically, the updating of one block or more of data will result in one or more blocks storing the data that have been superceded by the new data. If so, as indicated by a step 59, those blocks with superceded data are identified for erasure. For the purpose of increasing performance, it is preferable that erase operations occur in the background, or when host requested programming or reading operations are not taking place. After being erased, the blocks are returned to the inventory of unused, erased blocks for further use. Alternatively, erasure of the blocks can be deferred until they are needed for programming operations.

If, on the other hand, in the step 53, it is determined that there are fewer pages of new data than will utilize the full storage capacity of a block, a next step 61 determines whether there are enough unused pages in a block having some pages programmed with other data. If so, such a block is addressed, in a step 63. If not, a totally unused, erased block is addressed, in a step 65. In either case, in a step 67, the new data are programmed into unused pages of the addressed block. As part of this programming process, the LBN and page offset is written into the fields 41, and the time stamp into the fields 43 of each of the pages (Figure 8) of the updated data, in the manner described above.

A desirable feature of the programming process is to make available for future programming any blocks that store only superceded data. So the question is asked, in a step 69, whether the data updating process has resulted in an entire block remaining with only superceded data. If so, such a block is queued for erasure, in a step 71, and the process is then completed. If not, the step 71 is omitted and the data update is finished.

METABLOCK OPERATION

In order to improve performance by reducing programming time, a goal is to program as many cells in parallel as can reasonably be done without incurring other penalties. One implementation divides the memory array into largely independent sub-arrays or units, such as multiple units 80-83 of Figure 15, each unit in turn being divided into a large number of blocks, as shown. Pages of data are then programmed at the same time into more than one of the units. Another configuration further combines one or more of these units from multiple memory chips. These multiple chips may be connected to a single bus (as shown in Figure 2) or multiple independent busses for higher data throughput. An extension of this is to link blocks from different units for programming, reading and erasing together, an example being shown in Figure 15. Blocks 85-88 from respective ones of the units 80-83 can be operated together as a metablock, for example. As with the memory embodiments described above, each block, the smallest erasable group of the memory array, is typically divided into multiple pages, a page containing the smallest number of cells that are programmable together within the block. Therefore, a programming operation of the metablock shown in Figure 15 will usually include the simultaneously programming of data into at least one page of each of the blocks 85-88 forming the metablock, which is repeated until the metablock is full or the incoming data has all been programmed. Other metablocks are formed of different blocks from the array units, one block from each unit.

In the course of operating such a memory, as with others, pages of data less than an entire block often need to be updated. This can be done for individual blocks of a metablock in the same manner as described above with respect to either of Figures 4 or 6, but preferably by use of the improved technique described with respect to Figure 8. When any of these three techniques are used to update data of one block of the metablock, an additional block of memory within the same unit is also used. Further, a data update may require writing new data for one or more pages of two or more of the blocks of a metablock. This can then require use of up to four additional blocks 90-93, one in each of the four units, to update a data file stored in the metablock, even though the data in only a few pages is being updated.

In order to reduce the number of blocks required for such partial block updates, according to another aspect of the present invention, updates to pages of data within any of the blocks of the illustrated metablock are made, as illustrated by Figure 16, to a single additional block 90 in the memory unit 80, so long as unused pages in the block 80 remain. If, for example, data in three pages of the block 86 and two pages of the block 88 are being updated at one time, all five pages of the new data are written into the block 90. This can save the use of one block of memory, thereby to effectively increase the number of available erased blocks by one block. This helps avoid, or at least postpone, the time when an inventory of erased blocks becomes exhausted. If one or more pages from each of the four blocks 85-88 are being updated, all of the new data pages are programmed in the single block 90, thereby avoiding tying up an additional three blocks of memory to make the update. If the number of pages of new data exceed the capacity of an unused block, pages that the block 90 cannot accept are written to another unused block which may be in the same unit 80 or one of the other units 81-83.

Although the invention has been described with respect to various exemplary embodiments, it will be understood that the invention is entitled to protection within the full scope of the appended claims.

IT IS CLAIMED:

1. A method of simultaneously storing original and replacement data in a non-volatile memory system, comprising:

identifying the original and replacement data by the same logical address, and

distinguishing the replacement data from the original data by keeping track of the relative times that the original and replacement data have been programmed into the memory.

2. A method of storing and retrieving original and replacement data in a non-volatile memory system, comprising:

identifying units of the original and the replacement data by the same logical address,

reading units of data in an inverse order from an order in which they were programmed into the memory, and

distinguishing units of replacement data from units of original data having the same logical address by the order in which they are read.

3. In a non-volatile memory system having a plurality of blocks of memory storage elements that are individually organized into a plurality of pages of memory storage elements, a method of substituting new data for superceded data within at least one page of one of the plurality of blocks while data in at least another page of said one block is not replaced, comprising:

programming the new data into at least one page of said one or another of the plurality of blocks,

identifying the at least one page of superceded data and the at least one page of new data by a common logical address, and

recording a relative time of programming the new and the superceded data.

4. The method of claim 3, wherein the relative time of programming is recorded for the individual pages in which the new and superceded data are programmed, whereby the at least one page of new data is distinguishable from the at least one page of superceded data by their recorded relative times of programming.

5. The method of claim 3, wherein the relative time of programming is recorded for the individual blocks, thereby to identify an order of programming of individual blocks having data with a common logical address, and further wherein pages within the individual blocks are programmed in a designated order, whereby the new pages of data are distinguishable from superceded pages of data within an individual block by their relative positions within the block.

6. The method of claim 3, wherein the data in at least another page of said one block that is not replaced are not copied into said one or another block as part of substituting the new data for the superceded data.

7. The method of claim 3, wherein nothing is written into the at least one page of superceded data as part of substituting the new data for the superceded data.

8. The method of claim 4, wherein recording a relative time of programming the new and superceded data includes storing a value of a clock at each of the times that the new and superceded data are programmed.

9. The method of claim 4, wherein recording a relative time of programming the new and superceded data includes storing a different value of a sequence of numbers at each of the times that the new and superceded data are programmed.

10. The method of either of claims 8 or 9, wherein storing the value indicating a relative time of programming the new and superceded data includes storing the individual values within the same pages as the new and superceded data to which the values relate.

11. The method of claim 3, wherein programming the new data into at least one page of another said one or another of the plurality of blocks includes programming the new data into the first available unused pages within said one or another block in a predefined order.

12. The method of claim 3, wherein identifying the at least one page of superceded data and the at least one page of new data by a common logical address includes recording at least part of the common logical address in the individual pages as overhead data.

13. The method of claim 12, including building a table in volatile memory including multiple physical block addresses for the common logical address.

14. A method of reading data that has been updated according to claim 4, comprising:

reading pages of data from said one block and, if new data has been programmed thereinto, said another block,

identifying any multiple pages of data that have the same logical address,

utilizing the recorded relative time of programming the new and superceded data to identify the most current of any pages having the same logical address, and

assembling data in the most current of any pages having the same logical address along with pages in said at least another page of said one block that have not been updated.

15. A method of reading data that has been updated according to claim 5, comprising:

reading pages of data within said one and, if new data has been programmed thereinto, another block in a reverse order from which they were programmed, and

passing over any pages of data so read which have the same logical page address as a page whose data has already been read.

16. The method of either one of claims 14 or 15, additionally comprising operating the individual memory storage elements with more than two storage states, thereby storing more than one bit of data in each storage element, and reading pages of data includes reading the more than two storage states from the individual memory storage elements.

17. The method of any one of claims 3-9, additionally comprising operating storage elements of the individual memory cells with more than two storage states, thereby storing more than one bit of data in each storage element.

18. The method of claim 17, wherein the storage elements include individual floating gates.

19. The method of any one of claims 3-9, wherein the non-volatile memory system is formed within an enclosed card having an electrical connector along one edge thereof that operably connects with a host system.

20. A method of operating a non-volatile memory system having an array of memory storage elements organized into at least two sub-arrays, wherein the individual sub-arrays are divided into a plurality of non-overlapping blocks of storage elements wherein a block contains the smallest group of memory storage elements that are erasable together, and the individual blocks are divided into a plurality of pages of storage elements wherein a page is the smallest group of memory storage elements that are programmable together, comprising:

linking at least one block from individual ones of said at least two sub-arrays to form a metablock wherein its component blocks are erased together as a unit, and

updating pages of original data within any of the metablock component blocks less than all the pages within the block by programming replacement data into pages within another at least one block in only a designated one of the sub-arrays regardless of which sub-array the data being updated is stored.

21. The method of claim 20, wherein storing the original and replacement data includes:

identifying the original and replacement data by the same logical address to the memory system, and

distinguishing the replacement data from the original data by keeping track of the relative times that the original and replacement data have been programmed their respective pages of the memory.

22. A non-volatile memory system, comprising:

an array of non-volatile memory storage elements organized in blocks of storage elements, wherein an individual block contains the smallest group of storage elements that is erasable, and

a programming mechanism that writes into a first block an updated version of less than all of original data stored in a second block along with an indication of the later writing of the updated version,

an address mechanism that logically addresses both the original data and the updated version with the same address, and

a reading mechanism that distinguishes the updated version from the original data at least in part by the relative time by said indication of the later writing of the updated version.

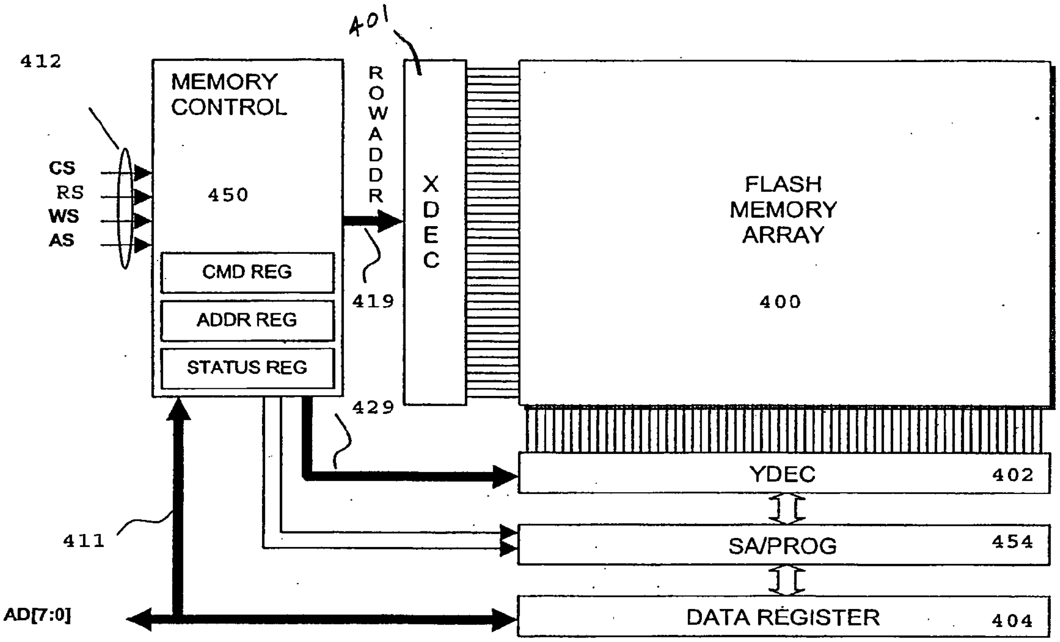

412
MEMORY CONTROL
450
CS
RS
WS
AS
CMD REG
ADDR REG
STATUS REG
401
ROW ADDR
419
XDEC
FLASH MEMORY ARRAY
400
429
YDEC
402
411
SA/PROG
454
AD[7:0]
DATA REGISTER
404

Fig. 1 (Prior Art)

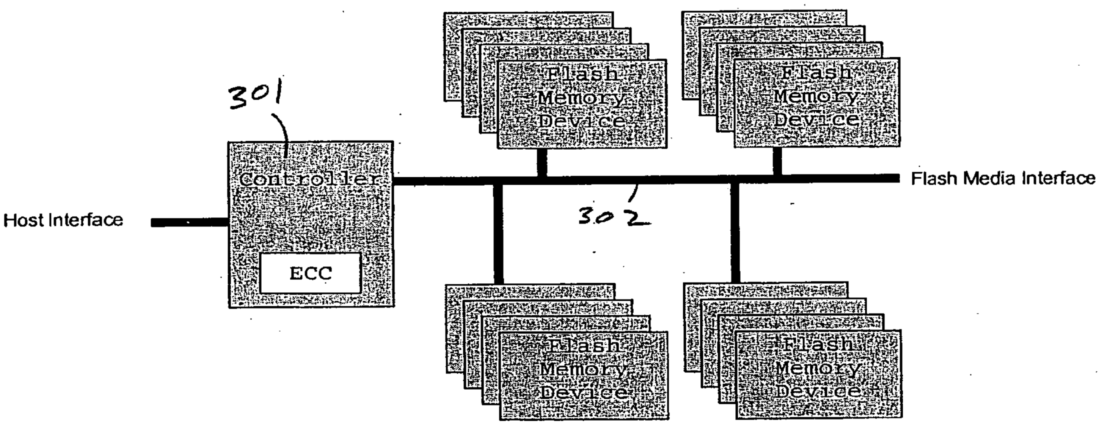

301
Controller
ECC
Host Interface
Flash Memory Device
Flash Memory Device
Flash Media Interface
302
Flash Memory Device
Flash Memory Device

Fig. 2 (Prior Art)

M-10262
(2 of 7)

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS
AMPAD

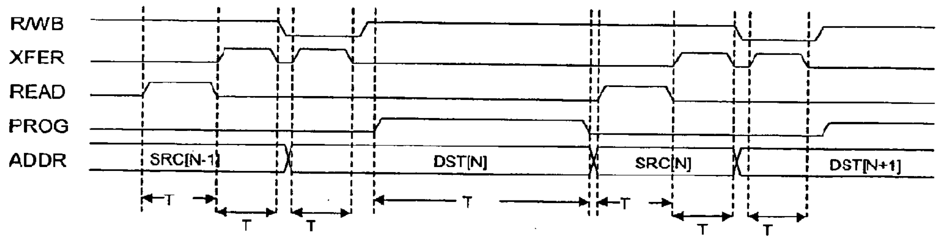

R/WB
XFER
READ
PROG
ADDR
SRC[N-1]
DST[N]
SRC[N]
DST[N+1]
T
T
T
T
T
T
T

Fig. 3 (Prior Art)

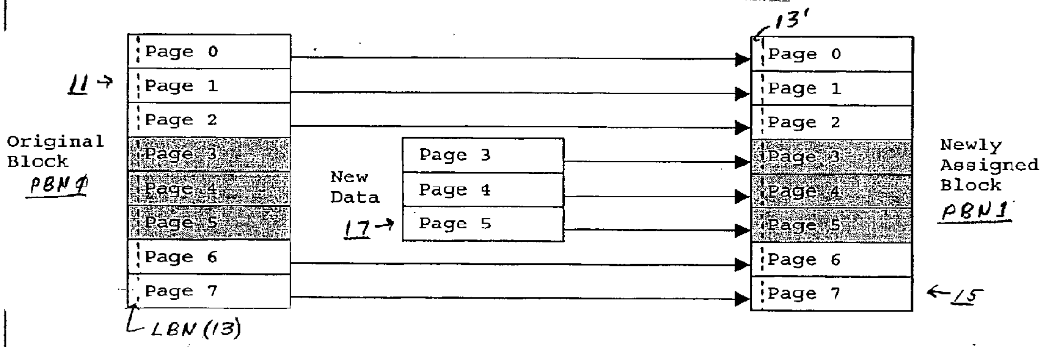

13'
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
11
Original Block
PBN 0
New Data
17
Newly Assigned Block
PBN 1
15
LBN (13)

Fig. 4 (Prior Art)

LBN	PBN
0	0

FIG.5A
Original Block 11

LBN	PBN
0	1

FIG. 5B
with New Block 15

M-10262
(3 of 7)

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS
AMPAD

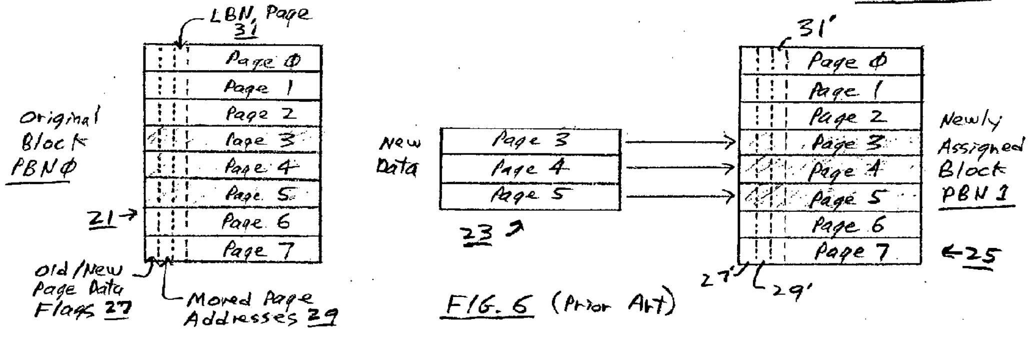

LBN, Page 31
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Original Block PBN 0
21
Old/New Page Data Flags 27
Moved Page Addresses 29
New Data
Page 3
Page 4
Page 5
23
31'
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Newly Assigned Block PBN 1
25
27'
29'

FIG. 6 (Prior Art)

FIG. 7A (original Block)

LBN	Page	PBN	Page
0	0	0	0
0	1	0	1
0	2	0	2
0	3	0	3
0	4	0	4
0	5	0	5
0	6	0	6
0	7	0	7
⋮	⋮	⋮	⋮

FIG. 7B (with New Block)

LBN	Page	PBN	Page
0	0	0	0
0	1	0	1
0	2	0	2
0	3	1	3
0	4	1	4
0	5	1	5
0	6	0	6
0	7	0	7
⋮	⋮	⋮	⋮

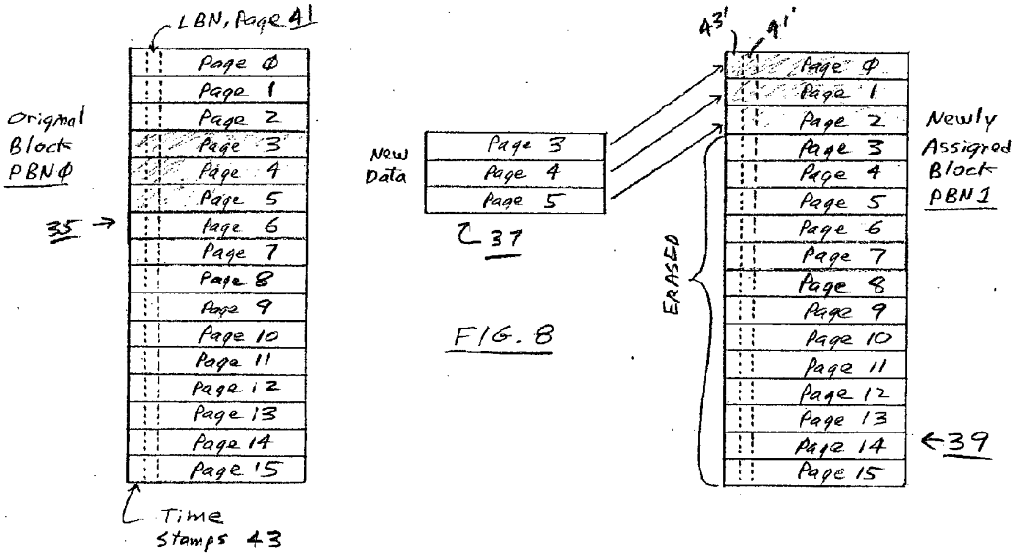

LBN, Page 41
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
Original Block PBN 0
35
Time Stamps 43
New Data
Page 3
Page 4
Page 5
37
43'
41'
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
ERASED
Newly Assigned Block PBN 1
39

FIG. 8

M-10262
(4 of 7)

LBN, Page		PBN0, Page		PBN1, Page	
0	0	0	0		
0	1	0	1		
0	2	0	2		
0	3	0	3	1	0
0	4	0	4	1	1
0	5	0	5	1	2
0	6	0	6		
0	7	0	7		
⋮	⋮	⋮	⋮		

FIG. 9

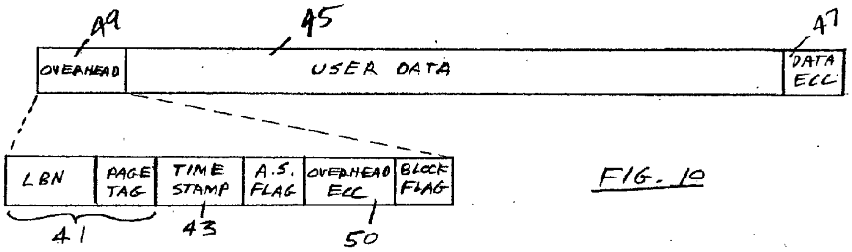

49
45
47
OVERHEAD
USER DATA
DATA ECC
LBN
PAGE TAG
TIME STAMP
A.S. FLAG
OVERHEAD ECC
BLOCK FLAG
41
43
50

FIG. 10

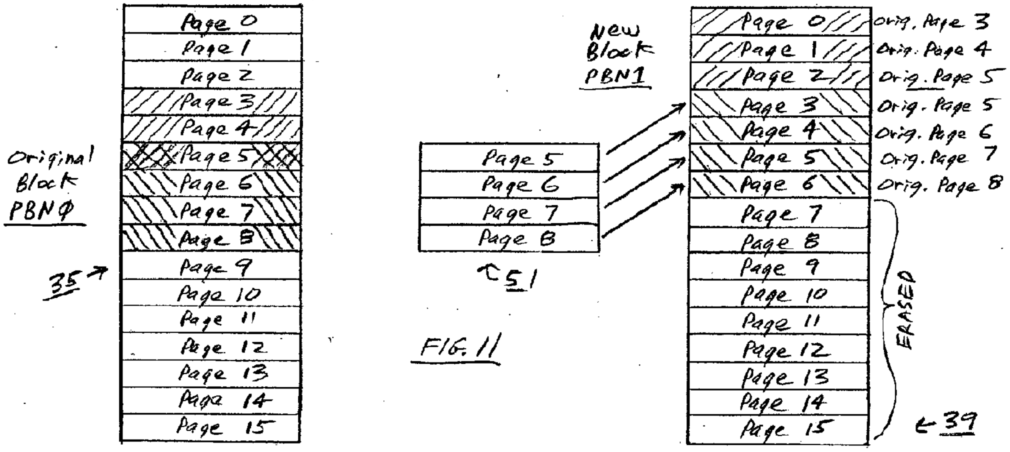

Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
Original Block PBN0
35
Page 5
Page 6
Page 7
Page 8
51
New Block PBN1
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Page 10
Page 11
Page 12
Page 13
Page 14
Page 15
Orig. Page 3
Orig. Page 4
Orig. Page 5
Orig. Page 5
Orig. Page 6
Orig. Page 7
Orig. Page 8
ERASED
39

FIG. 11

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS
AMPAD

M-1026Z
(5 of 7)

AMPAD 22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS

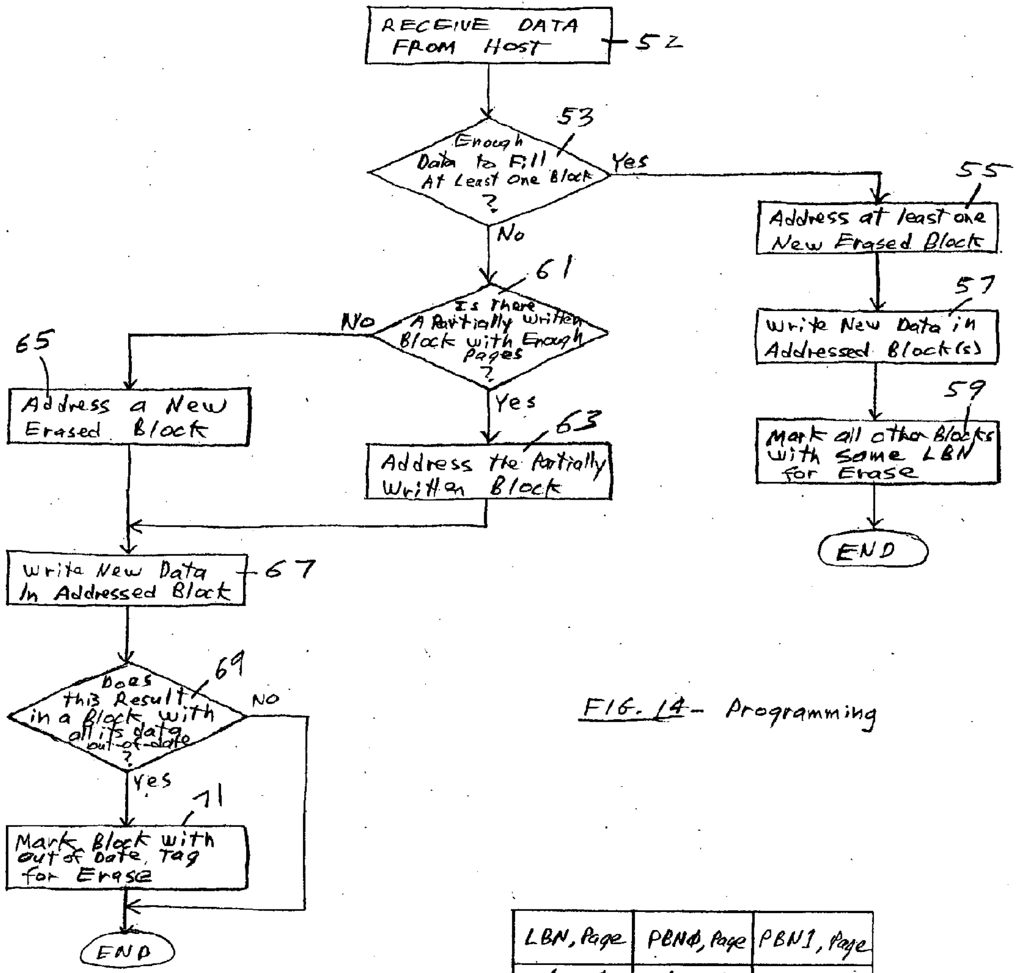

RECEIVE DATA FROM HOST
52
Enough Data to Fill At Least One Block ?
53
Yes
No
55
Address at least one New Erased Block
57
Write New Data in Addressed Block(s)
59
Mark all other Blocks with same LBN for Erase
END
61
Is There A Partially Written Block with Enough Pages ?
No
Yes
65
Address a New Erased Block
63
Address the Partially Written Block
Write New Data In Addressed Block
67
Does this Result in a Block with all its data out-of-date ?
69
NO
Yes
71
Mark Block with out of Date, Tag for Erase
END

FIG. 14 – Programming

LBN, Page	PBN0, Page	PBN1, Page
0 0	0 0	
0 1	0 1	
0 2	0 2	
0 3	0 3	1 0
0 4	0 4	1 1
0 5	0 5	1 ~~2~~3
0 6	0 6	1 4
0 7	0 7	1 5
0 8	0 8	1 6
0 9	0 9	
⋮ ⋮	⋮ ⋮	⋮

FIG. 12

M-10262
(6 of 7)

AMPAD 22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS

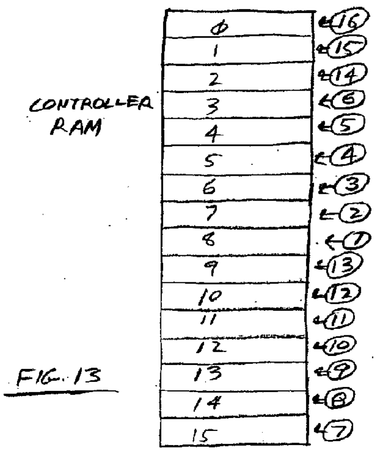

CONTROLLER RAM
0 ← ⑯
1 ← ⑮
2 ← ⑭
3 ← ⑥
4 ← ⑤
5 ← ④
6 ← ③
7 ← ②
8 ← ①
9 ← ⑬
10 ← ⑫
11 ← ⑪
12 ← ⑩
13 ← ⑨
14 ← ⑧
15 ← ⑦

FIG. 13

M-10262
(7 of 7)

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS
AMPAD

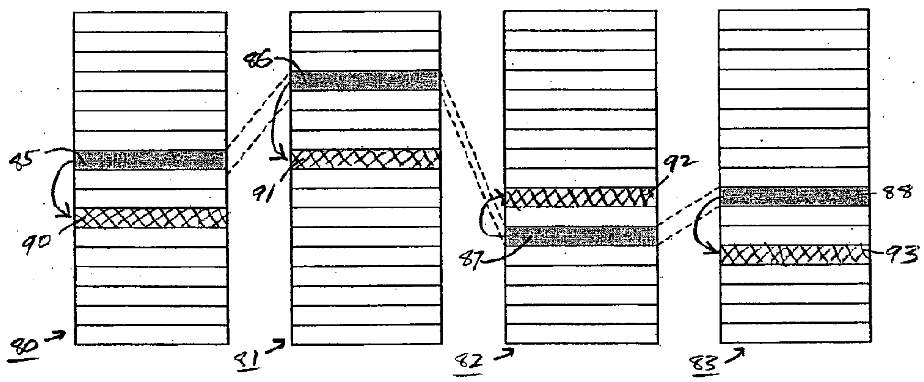

86
85
92
88
91
90
87
93
80
81
82
83

FIG. 15

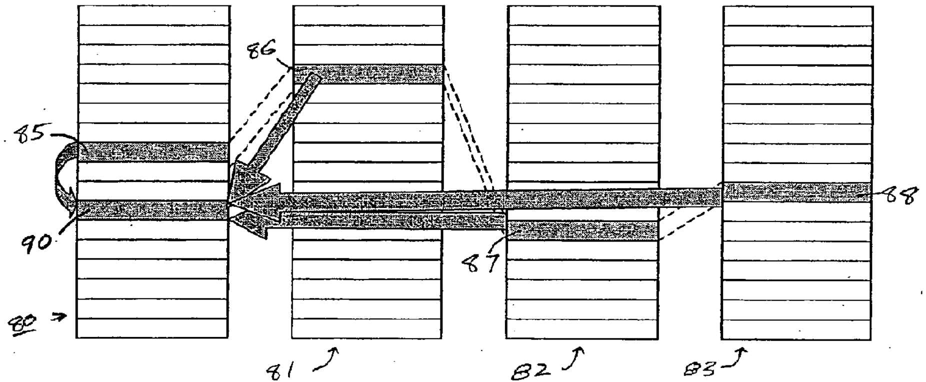

86
85
88
90
87
80
81
82
83

FIG. 16

Electronic Acknowledgement Receipt

EFS ID:	1422839
Application Number:	11250238
International Application Number:	
Confirmation Number:	7727
Title of Invention:	Partial block data programming and reading operations in a non-volatile memory
First Named Inventor/Applicant Name:	Kevin M. Conley
Customer Number:	66785
Filer:	Gerald Paul Parsons/Mary Buggie (GPP)
Filer Authorized By:	Gerald Paul Parsons
Attorney Docket Number:	SNDK.156US2
Receipt Date:	08-JAN-2007
Filing Date:	13-OCT-2005
Time Stamp:	14:41:05
Application Type:	Utility

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)	Multi Part /.zip	Pages (if appl.)
1	Foreign Reference	WO02058074.pdf	1465382	no	31
Warnings:					

Information:	
Total Files Size (in bytes):	1465382

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

PTO/SB/06 (12-04)
Approved for use through 7/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875 Effective December 8, 2004

Application or Docket Number: 11250238

APPLICATION AS FILED – PART I

FOR	NUMBER FILED (Column 1)	NUMBER EXTRA (Column 2)	SMALL ENTITY RATE ($)	SMALL ENTITY FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY FEE ($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	150.00		N/A	300.00
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	$250		N/A	$500
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	$100		N/A	$200
TOTAL CLAIMS (37 CFR 1.16(i))	3 minus 20 =	*	X$ 25 =		OR	X$50 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X100 =			X200 =	
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))			+180=			+360=	
			TOTAL			TOTAL	1000

* If the difference in column 1 is less than zero, enter "0" in column 2.

APPLICATION AS AMENDED – PART II

AMENDMENT A

1/8/07	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE ($)	SMALL ENTITY ADDITIONAL FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	15	Minus	20	=	X$ 25 =		OR	X$50 =	
Independent (37 CFR 1.16(h))	2	Minus	3	=	X100 =		OR	X200 =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					+180=		OR	+360=	
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE ($)	SMALL ENTITY ADDITIONAL FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	*	Minus	**	=	X$ 25 =		OR	X$50 =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	X100 =		OR	X200 =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))					+180=		OR	+360=	
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

his collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the ISPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, icluding gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments n the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent nd Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS DDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

EAST Search History

S43	0	S36 same S38	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:18
S44	296	S36 and S38	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:18
S45	253	S36 and S39	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:18
S46	322	S36 and S40	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:18
S47	91	S36 and S41	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:18
S48	1189	S36 and S42	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:19
S49	151	S38 and S39	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:19
S50	151	S49 and S40	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:19
S51	22	S40 same S41	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:19
S52	5074	(write writing written programm$3 eras$3) same (out$of$order (reverse near3 order))	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:24
S53	979	S37 and S52	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:20
S54	7	S40 and S53	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:21
S55	158	updat$3 with (reverse near3 order)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:21
S56	262	updat$3 same S40	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:22

EAST Search History

S57	3	S40 and S55	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:22
S58	24	S37 and S55	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:22
S59	45484	(old original) with (page block)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:23
S60	138072	(updated new) with (page block)	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:24
S61	4787	S37 and S59 and S60	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:24
S62	557	S61 and (out$of$order (reverse near3 order))	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:24
S63	5	S39 and S62	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:25
S64	14	S36 and S62	US-PGPUB; USPAT; EPO; IBM_TDB	OR	ON	2007/02/05 13:25

Application Number	Application/Control No. 11/250,238	Applicant(s)/Patent under Reexamination CONLEY, KEVIN M.	

Document Code - DISQ	**Internal Document – DO NOT MAIL**

TERMINAL DISCLAIMER	☒ APPROVED	☐ DISAPPROVED
Date Filed : 01/08/07	**This patent is subject to a Terminal Disclaimer**	

Approved/Disapproved by:
meason

U.S. Patent and Trademark Office

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL LETTER

Sir:

In response to a telephone call from Examiner Dinh on February 21, 2007, a second Terminal Disclaimer is being filed herewith, over the second parent patent No. 6,763,424. This Terminal Disclaimer is being filed in order to expedite formal allowance of the present application, without an evaluation of whether or not a sufficient basis exists for the request to file it.

The fee under 37 C.F.R. § 1.20(d) of $130.00 is being filed herewith via EFS. Please charge any additional fees required or credit any overpayment to our Deposit Account No. 502664.

FILED VIA EFS

Respectfully submitted,

February 21, 2007
Date

Gerald P. Parsons
Reg. No. 24,486
PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Kevin M. Conley		
Title:	Partial Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Dinh, Ngoc V.	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TERMINAL DISCLAIMER UNDER 37 C.F.R. § 1.321(c)

Sir:

SanDisk Corporation ("the owner") owns the entire interest in and to both the above-identified continuation application ("instant application") and parent patent no. 6,763,424 ("prior patent") by way of a written assignment from the inventor of parent patent application Serial No. 09/766,436, filed January 19, 2001.

The owner hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application that would extend beyond the expiration date of the full statutory term defined in 35 U.S.C. §§ 154, 155, 156 and 173, as presently shortened by any terminal disclaimer, of the prior patent, if any such extended term would otherwise exist. The owner further agrees that any such patent granted on the instant application shall be enforceable only for and during such period that such patent and the prior patent are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the owner, its successors and assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of any patent granted on the instant application that would extend to the expiration date of the full statutory term as defined in 35 U.S.C. §§ 154, 155, 156 and 173 of the prior patent, as presently shortened by any terminal disclaimer, in the event that the prior patent later expires for failure to pay a maintenance fee, is held unenforceable, is found invalid by a court of competent jurisdiction, is statutorily disclaimed in whole or terminally disclaimed under 37 CFR § 1.321, has all claims canceled by a reexamination certificate, is reissued, or is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

The fee under 37 C.F.R. § 1.20(d) of $130.00 is being filed herewith. Please charge any additional fees required or credit any overpayment to our Deposit Account No. 502664.

FILED VIA EFS

Respectfully submitted,

Gerald P. Parsons
Reg. No. 24,486

February 21, 2007
Date

PARSONS HSUE & DE RUNTZ LLP
595 Market Street, Suite 1900
San Francisco, CA 94105
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)

Electronic Patent Application Fee Transmittal

Application Number:	11250238
Filing Date:	13-Oct-2005
Title of Invention:	Partial block data programming and reading operations in a non-volatile memory
First Named Inventor/Applicant Name:	Kevin M. Conley
Filer:	Gerald Paul Parsons/Mary Buggie (GPP)
Attorney Docket Number:	SNDK.156US2

Filed as Large Entity

Utility Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Statutory disclaimer	1814	1	130	130
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
			Total in USD ($)	**130**

Electronic Acknowledgement Receipt

EFS ID:	1532533
Application Number:	11250238
International Application Number:	
Confirmation Number:	7727
Title of Invention:	Partial block data programming and reading operations in a non-volatile memory
First Named Inventor/Applicant Name:	Kevin M. Conley
Customer Number:	66785
Filer:	Gerald Paul Parsons/Mary Buggie (GPP)
Filer Authorized By:	Gerald Paul Parsons
Attorney Docket Number:	SNDK.156US2
Receipt Date:	21-FEB-2007
Filing Date:	13-OCT-2005
Time Stamp:	13:33:31
Application Type:	Utility

Payment information:

Submitted with Payment	yes
Payment was successfully received in RAM	$130
RAM confirmation Number	2041
Deposit Account	502664

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)	Multi Part /.zip	Pages (if appl.)

1		SNDK156US2-Trans_Term_Discl.pdf	118865	yes	3

Multipart Description/PDF files in .zip description

Document Description	Start	End
Miscellaneous Incoming Letter	1	1
Terminal Disclaimer Filed	2	3

Warnings:

Information:

2	Fee Worksheet (PTO-06)	fee-info.pdf	8189	no	2

Warnings:

Information:

Total Files Size (in bytes): 127054

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Application Number	**Application/Control No.** 11/250,238	**Applicant(s)/Patent under Reexamination** CONLEY, KEVIN M.	

Document Code - DISQ	**Internal Document – DO NOT MAIL**

TERMINAL DISCLAIMER	☐ APPROVED	☒ DISAPPROVED
Date Filed : 022107	**This patent is subject to a Terminal Disclaimer**	

Approved/Disapproved by:
jrm

U.S. Patent and Trademark Office

5

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

66785 7590 03/06/2007

PARSONS HSUE & DE RUNTZ, LLP - SANDISK CORPORATION
595 MARKET STREET
SUITE 1900
SAN FRANCISCO, CA 94105

EXAMINER	
DINH, NGOC V	
ART UNIT	PAPER NUMBER
2189	

DATE MAILED: 03/06/2007

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/250,238	10/13/2005	Kevin M. Conley	SNDK.156US2	7727

TITLE OF INVENTION: PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1400	$300	$0	$1700	06/06/2007

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

B. If the status above is to be removed, check box 5b on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above, or

If the SMALL ENTITY is shown as NO:

A. Pay TOTAL FEE(S) DUE shown above, or

B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check box 5a on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and 1/2 the ISSUE FEE shown above.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PTOL-85 (Rev. 07/06) Approved for use through 04/30/2007.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail **Mail Stop ISSUE FEE**
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
or Fax. (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

66785 7590 03/06/2007

PARSONS HSUE & DE RUNTZ, LLP - SANDISK CORPORATION
595 MARKET STREET
SUITE 1900
SAN FRANCISCO, CA 94105

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/250,238	10/13/2005	Kevin M. Conley	SNDK.156US2	7727

TITLE OF INVENTION: PARTIAL BLOCK DATA PROGRAMMING AND READING OPERATIONS IN A NON-VOLATILE MEMORY

APPLN. TYPE	SMALL ENTITY	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1400	$300	$0	$1700	06/06/2007

EXAMINER	ART UNIT	CLASS-SUBCLASS
DINH, NGOC V	2189	711-103000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list

(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,

(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ____________________
2 ____________________
3 ____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. The following fee(s) are submitted:

☐ Issue Fee
☐ Publication Fee (No small entity discount permitted)
☐ Advance Order - # of Copies ____________

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**

☐ A check is enclosed.
☐ Payment by credit card. Form PTO-2038 is attached.
☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number __________ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

☐ a. Applicant claims SMALL ENTITY status. See 37 CFR 1.27. ☐ b. Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature ____________________ Date ____________________

Typed or printed name ____________________ Registration No. ____________________

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PTOL-85 (Rev. 07/06) Approved for use through 04/30/2007. OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/250,238	10/13/2005	Kevin M. Conley	SNDK.156US2	7727

66785 7590 03/06/2007

PARSONS HSUE & DE RUNTZ, LLP - SANDISK CORPORATION
595 MARKET STREET
SUITE 1900
SAN FRANCISCO, CA 94105

EXAMINER	
DINH, NGOC V	
ART UNIT	PAPER NUMBER
2189	

DATE MAILED: 03/06/2007

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment to date is 0 day(s). If the issue fee is paid on the date that is three months after the mailing date of this notice and the patent issues on the Tuesday before the date that is 28 weeks (six and a half months) after the mailing date of this notice, the Patent Term Adjustment will be 0 day(s).

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

PTOL-85 (Rev. 07/06) Approved for use through 04/30/2007.

Notice of Allowability	Application No. 11/250,238	Applicant(s) CONLEY, KEVIN M.	
	Examiner NGOC V. DINH	Art Unit 2189	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *amendment filed 01/08/07*.

2. ☒ The allowed claim(s) is/are *4-18 (renumbered as 1-15).*.

3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   a) ☐ All b) ☐ Some* c) ☐ None of the:

   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.

   (a) ☐ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached

   1) ☐ hereto or 2) ☐ to Paper No./Mail Date _____.

   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. ☒ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 01/08/07
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. ☐ Notice of Informal Patent Application
6. ☒ Interview Summary (PTO-413), Paper No./Mail Date *01/30/2007* .
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. ☐ Other _____.

DETAILED ACTION

1. This Office Action is responsive to amendment and terminal disclaimer filed 01/08/2007. In this instant application claims 1-3 are canceled. Claims 4-15 are added.

Terminal Disclaimer

2. The terminal disclaimer filed on 01/08/2007 and 02/20/2007 disclaiming the terminal portion of any patent granted on this application which would extend beyond the expiration date of parent patent 6,968,421 and grand parent patent 6,763,424 have been reviewed and are accepted. The terminal disclaimers have been recorded.

Examiner's Amendment

3. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this Examiner's Amendment was given in a telephone interview with Mr. Gerald Parsons (Reg. 24,486) on February 5 and 21 of 2007.

In the claims:

a) Claim 4, lines 8-9, replace "the same" with --common --.

b) Claim 5, line 1:
replace "an indication" with -- the indication--.
replace "a time" with --the time--.

c) Claim 6, line 1:
replace "an indication" with -- the indication--.
replace "a time" with --the time--.

d) Claim 7, line 1:
replace "an indication" with -- the indication--.
replace "a time" with --the time--.

e) Claim 13, line 8, replace "the same" with --common --.

Reasons for allowance

3. The primary reasons for allowance of claim 4 in the instant application is the combination with the inclusion of at least the limitations of " identify the initial and update data pages by the same logical addresses, and when reading data of two or more pages having the same logical addresses, read the indications of the times that data have been stored in the two or more pages and use the data in the two or more pages having more recent time indications without using data in the two or more pages having older time indications".

The primary reasons for allowance of claim 13 in the instant application is the combination with the inclusion of at least the limitations of "identify the initial and update data pages by the same logical addresses, and reading data from the pages of the blocks in an order that is a reverse of the sequence in which they where written and ignore data in any page having the same logical address as a page from which data have already been read".

Because claims 5-12 and 14-18 depend directly or indirectly on claims 4 and 13. These claims are considered allowable for at least the same reasons noted above.

Conclusion

4. **Any response to this action should be mailed to:**
Under Secretary of Commerce for intellectual Property and Director of the
United States Patent and Trademark Office

PO Box 1450
Alexandria, VA 22313-1450
or faxed to:
(571) 273-8300, (for Official communications intended for entry)

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PMR) system. Status information for published Applications may be obtained from either Private PMR or Public PMR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http//pak-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ngoc Dinh whose telephone number is (571) 272-4191. The examiner can normally be reached on Monday-Friday 8:30 AM-5:00 PM. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Reginald Bragdon, can be reached on (571) 272-4204.

NGOC DINH
Patent Examiner
ART UNIT 2189
February 6, 2007

Reginald D. Bragdon
REGINALD BRAGDON
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Interview Summary	**Application No.** 11/250,238	**Applicant(s)** CONLEY, KEVIN M.
	Examiner NGOC V. DINH	**Art Unit** 2189

All participants (applicant, applicant's representative, PTO personnel):

(1) *NGOC V. DINH*. (3)_____.

(2) *Gerald Parsons (24,486)*. (4)_____.

Date of Interview: *5, 21 February 07*.

Type: a)☒ Telephonic b)☐ Video Conference
c)☐ Personal [copy given to: 1)☐ applicant 2)☐ applicant's representative]

Exhibit shown or demonstration conducted: d)☐ Yes e)☒ No.
If Yes, brief description: _____.

Claim(s) discussed: *4-7 and 13*.

Identification of prior art discussed: _____.

Agreement with respect to the claims f)☒ was reached. g)☐ was not reached. h)☐ N/A.

Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: *See Continuation Sheet*.

(A fuller description, if necessary, and a copy of the amendments which the examiner agreed would render the claims allowable, if available, must be attached. Also, where no copy of the amendments that would render the claims allowable is available, a summary thereof must be attached.)

THE FORMAL WRITTEN REPLY TO THE LAST OFFICE ACTION MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a reply to the last Office action has already been filed, APPLICANT IS GIVEN A NON-EXTENDABLE PERIOD OF THE LONGER OF ONE MONTH OR THIRTY DAYS FROM THIS INTERVIEW DATE, OR THE MAILING DATE OF THIS INTERVIEW SUMMARY FORM, WHICHEVER IS LATER, TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW. See Summary of Record of Interview requirements on reverse side or on attached sheet.

Reginald D. Bragdon
REGINALD BRAGDON
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Examiner Note: You must sign this form unless it is an Attachment to a signed Office action.

____
Examiner's signature, if required

Continuation of Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: The Examiner initiated a phone call to Mr. Parsons to notify him that there is a typographical error in the examiner previous office action in section 5, line 1, that the number of the patent (parent patent) which forms the basis for the double patent rejection is 6,968,421 instead of 6,969,42. This typographical error causes the terminal disclaimer not being accepted.
There is no need for the applicant to response to this action since the applicant terminal disclaimer filed on 01/08/2007 was appropriate (filed for 6,968,421). The examiner will correct this errror.
On February 21, 2007, the examiner requested the Applicant to file a Terminal Disclaimer for grandparent PN 6,763,424 in order to overcome the double patenting rejection based on the grandparent PN 6,763,424. The Applicant agreed to file the Terminal Disclaimer.

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicant	**Conf. No.**
(Use several sheets if necessary)	Kevin M. Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
ND	1	6,968,421 B2	11/22/2005	Conley			

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No
ND	2	WO 02/058074A2	7/25/2002	WIPO			X	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

Examiner Date Considered 02/05/07

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

Sheet 1 of 1

Express Mail No.: EV663653274US

Issue Classification	**Application/Control No.** 11/250,238	**Applicant(s)/Patent under Reexamination** CONLEY, KEVIN M.
	Examiner NGOC V. DINH	**Art Unit** 2189

ISSUE CLASSIFICATION

ORIGINAL		INTERNATIONAL CLASSIFICATION									
CLASS	SUBCLASS	CLAIMED					NON-CLAIMED				
711	103	G	00	F	12	/00					/
CROSS REFERENCES						/					/

CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)			
711	203			

Ngoc Dinh 02/06/2007 (Assistant Examiner) (Date)	Reginald D. Bragdon Reginald Bragdon 02/06/2007 (Primary Examiner) (Date)	**Total Claims Allowed: 15**
3/ / (Legal Instruments Examiner) (Date)		O.G. Print Claim(s): 1 — O.G. Print Fig.: 14

☐ **Claims renumbered in the same order as presented by applicant** ☐ CPA ☐ T.D. ☐ R.1.47

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1		31		61		91		121		151		181
	2		32		62		92		122		152		182
	3		33		63		93		123		153		183
1	4		34		64		94		124		154		184
2	5		35		65		95		125		155		185
3	6		36		66		96		126		156		186
4	7		37		67		97		127		157		187
5	8		38		68		98		128		158		188
6	9		39		69		99		129		159		189
7	10		40		70		100		130		160		190
8	11		41		71		101		131		161		191
9	12		42		72		102		132		162		192
10	13		43		73		103		133		163		193
11	14		44		74		104		134		164		194
12	15		45		75		105		135		165		195
13	16		46		76		106		136		166		196
14	17		47		77		107		137		167		197
15	18		48		78		108		138		168		198
	19		49		79		109		139		169		199
	20		50		80		110		140		170		200
	21		51		81		111		141		171		201
	22		52		82		112		142		172		202
	23		53		83		113		143		173		203
	24		54		84		114		144		174		204
	25		55		85		115		145		175		205
	26		56		86		116		146		176		206
	27		57		87		117		147		177		207
	28		58		88		118		148		178		208
	29		59		89		119		149		179		209
	30		60		90		120		150		180		210

U.S. Patent and Trademark Office

Part of Paper No. 20070130

Search Notes

Application/Control No. 11/250,238	**Applicant(s)/Patent under Reexamination** CONLEY, KEVIN M.	
Examiner NGOC V. DINH	**Art Unit** 2189	

SEARCHED

Class	Subclass	Date	Examiner

INTERFERENCE SEARCHED

Class	Subclass	Date	Examiner
interferrence search EAST		02/06/2007	ND

SEARCH NOTES (INCLUDING SEARCH STRATEGY)

	DATE	EXMR
Limited classified search of 711/103, 203 class's/sub's in "SEARCHED" section.	2/6/2007	ND
EAST text search w/o classified/search. See prinout	2/6/2007	ND

U.S. Patent and Trademark Office

Part of Paper No. 20070130

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

Bib Data Sheet

CONFIRMATION NO. 7727

SERIAL NUMBER	FILING OR 371(c) DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
11/250,238	10/13/2005 **RULE**	711	2189	SNDK.156US2

APPLICANTS

Kevin M. Conley, San Jose, CA;

** **CONTINUING DATA** ************************* yes NL

This application is a CON of 10/841,388 05/07/2004 PAT 6,968,421 which is a CON of 09/766,436 01/19/2001 PAT 6,763,424

** **FOREIGN APPLICATIONS** ********************** N/A NL

IF REQUIRED, FOREIGN FILING LICENSE GRANTED
** 11/03/2005

		STATE OR COUNTRY	SHEETS DRAWING	TOTAL CLAIMS	INDEPENDENT CLAIMS
Foreign Priority claimed	☐ yes ☒ no	CA	9	3	2
35 USC 119 (a-d) conditions met	☐ yes ☒ no ☐ Met after Allowance				
Verified and Acknowledged	Examiner's Signature ______ Initials NL				

ADDRESS
66785

TITLE

Partial block data programming and reading operations in a non-volatile memory

FILING FEE RECEIVED 1000	FEES: Authority has been given in Paper No. ________ to charge/credit DEPOSIT ACCOUNT No. ________ for following:	☐ All Fees ☐ 1.16 Fees (Filing) ☐ 1.17 Fees (Processing Ext. of time) ☐ 1.18 Fees (Issue) ☐ Other ______________ ☐ Credit

Application Number	**Application/Control No.** 11/250,238	**Applicant(s)/Patent under Reexamination** CONLEY, KEVIN M.	

Document Code - DISQ	**Internal Document – DO NOT MAIL**

TERMINAL DISCLAIMER	☒ APPROVED	☐ DISAPPROVED
Date Filed : 02/21/07	**This patent is subject to a Terminal Disclaimer**	

Approved/Disapproved by:
meason

U.S. Patent and Trademark Office

SDK0156.002US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s):	Conley		
Title:	Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Ngoc V. Dinh	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop RCE
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REQUEST FOR CONTINUED EXAMINATION (RCE)

Dear Sir:

This is a Request for Continued Examination (RCE) under 37 C.F.R. § 1.114 of the above-identified application. Please consider the Information DisclosureStatement, which is being filed herewith.

The RCE fee of $790.00 required under 37 C.F.R. § 1.17(e) has been authorized via EFS to Deposit Account 04-0258. The Commissioner is hereby authorized to charge any additional fees, which may be required, or credit any overpayment to Deposit Account 04-0258. Please contact the undersigned with any questions concerning this request or the above-identified patent application.

FILED VIA EFS

Respectfully submitted,

June 4, 2007

Gerald P. Parsons
Reg. No. 24,486

Date

DAVIS WRIGHT TREMAINE LLP
505 Montgomery Street, Suite 800
San Francisco, California 94111-6533
Telephone: (415) 276-6500
Facsimile: (415) 276-6599
Email: geraldparsons@dwt.com

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s):	Conley		
Title:	Block Data Programming and Reading Operations in a Non-Volatile Memory		
Application No.:	11/250,238	Filing Date:	October 13, 2005
Examiner:	Ngoc V. Dinh	Group Art Unit:	2189
Docket No.:	SNDK.156US2	Conf. No.:	7727

Mail Stop RCE
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

INFORMATION DISCLOSURE STATEMENT

Dear Sir:

Pursuant to 37 C.F.R. §§ 1.56, 1.97 and 1.98, Applicant(s) call(s) the documents listed on the enclosed Form PTO-1449 to the Examiner's attention in this patent application.

According to 37 C.F.R. 1.98(2)(ii), copies of the U.S. Patents and U.S. Published Patent Applications documents are not required and are therefore not enclosed. Copies of the listed foreign patent documents and/or Other Art are enclosed.

Citation of these documents shall not be construed as (1) an admission that the documents are prior art with respect to the invention or inventions claimed in this application, (2) a representation that a search has been made (other than as indicated by any cited document), or (3) an admission that the cited information is, or is considered to be, material to patentability as defined in § 1.56(b).

This information disclosure statement is submitted under 37 C.F.R. § 1.97(b) and consequently no fee should be required. The Commissioner is authorized, however, to charge any fee that may be required, or to credit any overpayment, against Deposit Account No. 04-0258.

FILED VIA EFS

Respectfully submitted,

Gerald P. Parsons
Reg. No. 24,486

June 4, 2007
Date

DAVIS WRIGHT TREMAINE LLP
505 Montgomery Street, Suite 800
San Francisco, California 94111-6533
Telephone: (415) 276-6500
Facsimile: (415) 276-6599
Email: geraldparsons@dwt.com

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicants	**Conf. No.**
(Use several sheets if necessary)	Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

U.S. Patent Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate
	1	5,924,092	7/13/99	Johnson			
	2	5,388,083	2/7/95	Assar et al.			
	3	5,479,638	12/26/95	Assart et al.			
	4	5,568,439	10/22/96	Harari			
	5	5,835,935	11/10/98	Estakhri et al.			
	6	5,838,614	11/17/98	Estakhri et al.			
	7	5,860,124	1/12/99	Matthews et al.			
	8	5,924,113	7/13/99	Estakhri et al.			
	9	5,937,425	8/10/99	Ban			
	10	5,598,370	1/28/97	Niijima et al.			
	11	6,023,423	2/8/00	Aritome			

U.S. Published Patent Application Documents

*Examiner Initial		Document Number	Date	Name	Class	Subclass	Filing Date If Appropriate

Foreign Patent Documents

							Translation	
		Document	Date	Country	Class	Subclass	Yes	No
	12	WO 98/44420	10/8/98	PCT				
	13	0 250 876	5/27/87	EPO				
	14	WO 02/49039	6/20/02	PCT				
	15	FR 2 742 893	12/20/95	France			Abstract	
	16	WO 99/07000	2/11/99	PCT				
	17	EP 0 977 121	7/26/99	EPO				
	18	EP 0 896 280	02/10/99	EPO				
	19	H10-091490	4/10/98	Japan			Abstract	
	20	H10-091490	4/10/98	Japan			X	
	21	H08-221223	8/30/96	Japan			Abstract	
	22	H08-221223	8/30/96	Japan			X	

FILED VIA EFS
Sheet 1 of 2

U.S. Department of Commerce, Patent and Trademark	**Atty. Docket No.**	**Application No.**
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	SNDK.156US2	11/250,238
	Applicants	**Conf. No.**
(Use several sheets if necessary)	Conley	7727
(Form PTO-1449)	**Filing Date**	**Art Group**
	October 13, 2005	2189

	23	WO 99/21093	4/29/99	PCT				
	24	H06-250798	9/19/94	Japan			Abstract	
	25	H06-250798	9/19/94	Japan			X	
	26	H10-105661	4/24/98	Japan			Abstract	
	27	H10-105661	4/24/98	Japan			X	

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

	28	Notice of Opposition to a European Patent for SanDisk Corporation, Patent No. 1,352,394, Application No. 02703078.2, dated March 1, 2007, 17 pages.
	29	Notification of Reasons for Refusal for SanDisk Corporation, Japanese Patent Application No. 2002-558275, mailed February 23, 2007, 10 pages.

Examiner	Date Considered

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in conformance and not considered. Include copy of this form with your communication to applicant.

FILED VIA EFS
Sheet 2 of 2

D/10

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : **G06F 12/06**	**A1**	**(11) International Publication Number:** **WO 98/44420** **(43) International Publication Date:** 8 October 1998 (08.10.98)

(21) International Application Number: PCT/US98/06335

(22) International Filing Date: 31 March 1998 (31.03.98)

(30) Priority Data:

08/831,266	31 March 1997 (31.03.97)	US
08/858,847	19 May 1997 (19.05.97)	US

(71) Applicant *(for all designated States except US)*: LEXAR MEDIA, INC. [US/US]; 47421 Bayside Parkway, Fremont, CA 94538 (US).

(71) Applicant *(for US only)*: GANJUEI, Juomana (heir of the deceased inventor) [SA/US]; 4286 Garibaldi Place, Pleasanton, CA 94566 (US).

(72) Inventor: GANJUEI, Ali, R. (deceased).

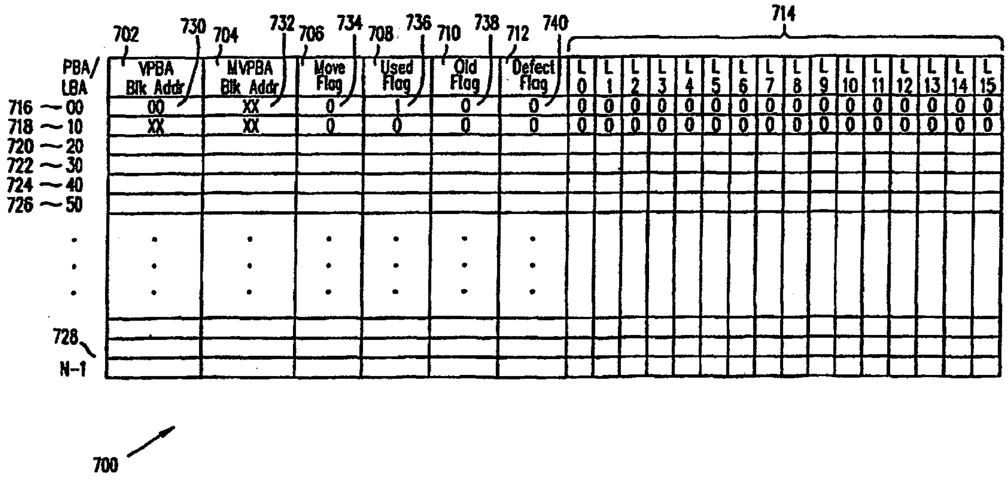
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: ESTAKHRI, Petro [US/US]; 7966 Foothill Knolls, Pleasanton, CA 94566 (US). IMAN, Berhanu [US/US]; 946 Iris Avenue, Sunnyvale, CA 94086 (US).

(74) Agents: HAMRICK, Claude, A., S. et al.; Oppenheimer Wolff & Donnelly LLP, Suite 600, Ten Almaden Boulevard, San Jose, CA 95113 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, 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 the receipt of amendments.

(54) Title: MOVING SECTORS WITHIN A BLOCK IN A FLASH MEMORY



PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20																						
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

(57) Abstract

A device (700) is disclosed for storing mapping information for mapping a logical block address identifying a block being accessed by a host to a physical block address, identifying a free area of nonvolatile memory, the block being selectively erasable and having one or more sectors that may be individually moved. The mapping information including a virtual physical block address (702) for identifying an "original" location within the nonvolative memory where a block is stored and a moved virtual physical block address (704) for identifying a "moved" location within the nonvolatile memory where one or more sectors of the stored block are moved. The mapping information further including status information (706, 712) for use of the "original" physical block address and the "moved" physical block address and for providing information (714) regarding "moved" sectors within the block being accessed.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

Specification

MOVING SECTORS WITHIN A BLOCK IN A FLASH MEMORY

BACKGROUND OF THE INVENTION

Cross Reference to Related Application

This application is a continuation-in-part of my prior application Serial No. 08/509,706, filed July 31, 1995, entitled "Direct Logical Block Addressing Flash Memory Mass Storage Architecture."

Field of the Invention

This invention relates to the field of mass storage for computers. More particularly, this invention relates to an architecture for replacing a hard disk with a semiconductor nonvolatile memory and in particular flash memory.

Computers conventionally use rotating magnetic media for mass storage of documents, data, programs and information. Though widely used and commonly accepted, such hard disk drives suffer from a variety of deficiencies. Because of the rotation of the disk, there is an inherent latency in extracting information from a hard disk drive.

Other problems are especially dramatic in portable computers. In particular, hard disks are unable to withstand many of the kinds of physical shock that a portable computer will likely sustain. Further, the motor for rotating the disk consumes significant amounts of power decreasing the battery life for portable computers.

Solid state memory is an ideal choice for replacing a hard disk drive for mass storage because it can resolve the problems cited above. Potential solutions have been proposed for replacing a hard disk drive with a semiconductor memory. For such a system to be truly useful, the memory must be nonvolatile and alterable. The inventors have determined that FLASH memory is preferred for such a replacement.

FLASH memory is a transistor memory cell which is programmable through hot electron, source injection, or tunneling, and erasable through Fowler-Nordheim tunneling. The programming and erasing of such a memory cell requires current to pass through the dielectric surrounding floating gate electrode. Because of this, such types of memory have a finite number of erase-write cycles. Eventually, the dielectric deteriorates. Manufacturers of FLASH cell devices specify the limit for the number of erase-write cycles between 100,000 and 1,000,000.

One requirement for a semiconductor mass storage device to be successful is that its use in lieu of a rotating media hard disk mass storage device be transparent to the designer and the user of a system using such a device. In other words, the designer or user of a computer incorporating such a semiconductor mass storage device could simply remove the hard disk and replace it with a semiconductor mass storage device. All presently available commercial software should operate on a system employing such a semiconductor mass storage device without the necessity of any modification.

SanDisk proposed an architecture for a semiconductor mass storage using FLASH memory at the Silicon Valley PC Design Conference on July 9, 1991. That mass storage system included read-write block sizes of 512 Bytes to conform with commercial hard disk sector sizes.

Earlier designs incorporated erase-before-write architectures. In this process, in order to update a file on the media, if the physical location on the media was previously programmed, it has to be erased before the new data can be reprogrammed.

This process would have a major deterioration on overall system throughput. When a host writes a new data file to the storage media, it provides a logical block address to the peripheral storage device associated with this data file. The storage device then translates this given logical block address to an actual physical block address on the media and performs the write operation. In magnetic hard disk drives, the new data can be written over the previous old data with no modification to the media. Therefore, once the physical block address is calculated from the given logical block address by the controller, it will simply write the data file into that location. In solid state storage, if the location associated with the calculated physical block address was previously programmed, before this block can be reprogrammed with the new data, it has to be erased. In one previous art, in erase-before-write architecture where the correlation between logical block address given by the host is one to one mapping with physical block address on the media. This method has many deficiencies. First, it introduces a delay in performance due to the erase operation before reprogramming the altered information. In solid state flash, erase is a very slow process.

Secondly, hard disk users typically store two types of information, one is rarely modified and another which is frequently changed. For example, a commercial spread sheet or word processing software program stored on a user's system are rarely, if ever, changed. However, the spread sheet data files or word processing documents are frequently changed. Thus, different sectors of a hard disk typically have dramatically different usage in terms of the number of times the information stored thereon is changed. While this disparity has no impact on a hard disk because of its insensitivity to data changes, in a FLASH memory device, this variance can cause sections of the mass storage to wear out and be unusable significantly sooner than other sections of the mass storage.

In another architecture, the inventors previously proposed a solution to store a table correlating the logical block address to the physical block address. The inventions relating to that solution are disclosed in U.S. Patent Application serial number 08/038,668 filed on March 26, 1993 and U.S. Patent Application serial number 08/037,893 also filed on March 26, 1993. Those applications are incorporated herein by reference.

The inventors' previous solution discloses two primary algorithms and an associated hardware architecture for a semiconductor mass storage device. It will be understood that "data file" in this patent document refers to any computer file including commercial software, a user program, word processing software document, spread sheet file and the like. The first algorithm in the previous solution provides means for avoiding an erase operation when writing a modified data file back onto the mass storage device. Instead, no erase is performed and the modified data file is written onto an empty portion of the mass storage.

The semiconductor mass storage architecture has blocks sized to conform with commercial hard disk sector sizes. The blocks are individually erasable. In one embodiment, the semiconductor mass storage can be substituted for a rotating hard disk with no impact to the user, so that such a substitution will be transparent. Means are provided for avoiding the erase-before-write cycle each time information stored in the mass storage is changed.

According to the first algorithm, erase cycles are avoided by programming an altered data file into an empty block. This would ordinarily not be possible when using conventional mass storage because the central processor and commercial software available in conventional computer systems are not configured to track continually changing physical locations of data files. The previous solution includes a programmable map to maintain a correlation between the logical address and the physical address of the updated information files.

All the flags, and the table correlating the logical block address to the physical block address are maintained within an array of CAM cells. The use of the CAM cells provides very rapid determination of the physical address desired within the mass storage, generally within one or two clock cycles. Unfortunately, as is well known, CAM cells require multiple transistors, typically six. Accordingly, an integrated circuit built for a particular size memory using CAM storage for the tables and flags will need to be significantly larger than a circuit using other means for just storing the memory.

The inventors proposed another solution to this problem which is disclosed in U.S. Patent Application serial number 08/131,495 filed on October 4, 1993. That application is incorporated herein by reference.

This additional previous solution invented by these same inventors is also for a nonvolatile memory storage device. The device is also configured to avoid having to perform an erase-before-write each time a data file is changed by keeping a correlation between logical block address and physical block address in a volatile space management RAM. Further, this invention avoids the overhead associated with CAM cell approaches which require additional circuitry.

Like the solutions disclosed above by these same inventors, the device includes circuitry for performing the two primary algorithms and an associated hardware architecture for a semiconductor mass storage device. In addition, the CAM cell is avoided in this previous solution by using RAM cells.

Reading is performed in this previous solutions by providing the logical block address to the memory storage. The system sequentially compares the stored logical block addresses until it finds a match. That data file is then coupled to the digital system. Accordingly, the performance offered by this solution suffers because potentially all of the memory locations must be searched and compared to the desired logical block address before the physical location of the desired information can be determined.

What is needed is a semiconductor hard disk architecture which provides rapid access to stored data without the excessive overhead of CAM cell storage.

SUMMARY OF THE INVENTION

The present invention is for a nonvolatile memory storage device. The device is configured to avoid having to perform an erase-before-write each time a data file is changed. Further, to avoid the overhead associated with CAM cells, this approach utilizes a RAM array. The host system maintains organization of the mass storage data by using a logical block address. The RAM array is arranged to be addressable by the same address as the logical block addresses (LBA) of the host. Each such addressable location in the RAM includes a field which holds the physical address of the data in the nonvolatile mass storage expected by the host. This physical block address (PBA) information must be shadowed in the nonvolatile memory to ensure that the device will still function after resuming operation after a power down because RAMs are volatile memory devices. In addition, status flags are also stored for each physical location. The status flags can be stored in either the nonvolatile media or in both the RAM and in the nonvolatile media.

The device includes circuitry for performing two primary algorithms and an associated hardware architecture for a semiconductor mass storage device. The first algorithm provides a means for mapping of host logical block address to physical block address with much improved performance and minimal hardware assists. In addition, the second algorithm provides means for avoiding an erase-before-write cycle when writing a modified data file back onto the mass storage device. Instead, no erase is performed and the modified data file is written onto an empty portion of the mass storage.

Reading is performed in the present invention by providing the logical block address to the memory storage. The RAM array is arranged so that the logical block address selects one RAM location. That location contains the physical block address of the data requested by the host or other external system. That data file is then read out to the host.
According to the second algorithm, erase cycles are avoided by programming an altered data file into an altered data mass storage block rather than itself after an erase cycle of the block as done on previous arts.

In an alternative embodiment of the present invention, a method and apparatus is presented for efficiently moving sectors within a block from a first area within the nonvolatile memory to an unused area within the nonvolatile memory and marking the first area as "used".

Briefly, a preferred embodiment of the present invention includes a method and apparatus for storing mapping information for mapping a logical block address identifying a block being accessed by a host to a physical block address, identifying a free area of nonvolatile memory, the block being selectively erasable and having one or more sectors that may be individually moved. The mapping information including a virtual physical block address for identifying an "original" location, within the nonvolatile memory, wherein a block is stored and a moved virtual physical block address for identifying a "moved" location, within the nonvolatile memory, wherein one or more sectors of the stored block are moved. The mapping information further including status information for use of the "original" physical block address and the "moved" physical block address and for providing information regarding "moved" sectors within the block being accessed.

IN THE DRAWINGS

Fig. 1 shows a schematic block diagram of an architecture for a semiconductor mass storage according to the present invention.

Fig. 2 shows an alternative embodiment to the physical block address 102 of the RAM storage of Figure 1.

Fig. 3 shows a block diagram of a system incorporating the mass storage device of the present invention.

Fig. 4 through 8 show the status of several of the flags and information for achieving the advantages of the present invention.

Fig. 9 shows a flow chart block diagram of the first algorithm according to the present invention.

Fig. 10 shows a high-level block diagram of a digital system, such as a digital camera, including a preferred embodiment of the present invention.

Figs. 11-21 illustrate several examples of the state of a mapping table that may be stored in the digital system of Figure 10 including LBA-PBA mapping information.

Fig. 22 depicts an example of a nonvolatile memory device employed in the preferred embodiment of Figure 10.

Fig. 23 shows a high-level flow chart of the general steps employed in writing a block of information to the nonvolatile devices of Figure 10.

Fig. 24 shows an example of the contents of flash memory devices in an alternative embodiment of the present invention using a novel defect flag and LBA address means.

Fig. 25 is an example of the contents of flash memory devices wherein another alternative embodiment of the present invention stores the LBA address for each block in two different sector locations within the block.

Fig. 26 includes Fig 26a and Fig. 26b, as shown in the key in Fig. 26b, and shows an example of yet another alternative embodiment of the present invention wherein the contents of the flash memory devices and the SPM RAM block are depicted to illustrate the correlation between the LBA and PBA addressing as employed by the system of Fig. 10.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 shows an architecture for implementation of a solid state storage media according to the present invention. The storage media is for use with a host or other external digital system. The mass storage is partitioned into two portions, a volatile RAM array 100 and a nonvolatile array 104. According to the preferred embodiment, all of the nonvolatile memory storage is FLASH. The FLASH may be replaced by EEPROM. The RAM can be of any convenient type.

The memory storage 104 is arranged into N blocks of data from zero through N-1. Each of the blocks of data is M Bytes long. In the preferred embodiment, each data block is 512 Bytes long to correspond with a sector length in a commercially available hard disk drive plus the extra numbers of bytes to store the flags and logical block address (LBA) information and the associated ECC. The memory 104 can contain as much memory storage as a user desires. An example of a mass storage device might include 100 M Byte of addressable storage.

There are a plurality of RAM locations 102. Each RAM location 102 is uniquely addressable by controller using an appropriate one of the logical block addresses provided by the host system or the actual physical address of the nonvolatile media. The RAM location 102 contains the physical block address of the data associated with the logical block address and the flags associated with a physical block address on the nonvolatile media.

It is possible that the physical block address (PBA) can be split into two fields as shown in Figure 2. These fields can be used for cluster addresses of a group of data blocks. The first such field 290 is used to select a cluster address and the second such field 292 can be used to select the start address of the logical block address associated with this cluster.

A collection of information flags is also stored for each nonvolatile memory location 106. These flags include an old/new flag 110, a used/free flag 112, a defect flag 114, and a single/sector flag 116. Additionally, there is also a data store 122.

When writing data to the mass storage device of the present invention, a controller determines the first available physical block for storing the data. The RAM location 102 corresponding to the logical block address selected by the host is written with the physical block address where the data is actually stored within the nonvolatile memory array in 104 (Figure 1).

Assume for example that a user is preparing a word processing document and instructs the computer to save the document. The document will be stored in the mass storage system. The host system will assign it a logical block address. The mass storage system of the present invention will select a physical address of an unused block or blocks in the mass storage for storing the document. The address of the physical block address will be stored into the RAM location 102 corresponding to the logical block address. As the data is programmed, the system of the present invention also sets the used free flag 112 in 104 and 293 to indicate that this block location is used. One used/free flag 112 is provided for each entry of the nonvolatile array 104.

Later, assume the user retrieves the document, makes a change and again instructs the computer to store the document. To avoid an erase-before-write cycle, the system of the present invention provides means for locating a block having its used/free flag 112 in 100 unset (not programmed) which indicates that the associated block is erased. The system then sets the used/free flag for the new block 112 of 106 and 293 of 100 and then stores the modified document in that new physical block location 106 in the nonvolatile array 104. The address of the new physical block location is also stored into the RAM location 102 corresponding the logical block address, thereby writing over the previous physical block location in 102. Next, the system sets the old/new flag 110 of the previous version of the document indicating that this is an old unneeded version of the document in 110 of 104 and 293 of 109. In this way, the system of the present invention avoids the overhead of an erase cycle which is required in the erase-before-write of conventional systems to store a modified version of a previous document.

Because of RAM array 100 will lose its memory upon a power down condition, the logical block address with the active physical block address in the media is also stored as a shadow memory 108 in the nonvolatile array 104. It will be understood the shadow information will be stored into the appropriate RAM locations 102 by the controller. During power up sequence, the RAM locations in 100 are appropriately updated from every physical locations in 104, by reading the information 106 of 104. The logical address 108 of 106 is used to address the RAM location of 100 to update the actual physical block address associated with the given logical block address. Also since 106 is the actual physical block address associated with the new data 122, the flags 110, 112, 114, and 116 arc updated in 293 of 102 with the physical block address of 106 in 100. It will be apparent to one of ordinary skill in the art that the flags can be stored in either the appropriate nonvolatile memory location 106 or in both the nonvolatile memory location and also in the RAM location 102 associated with the physical block address.

During power up, in order to assign the most recent physical block address assigned to a logical block address in the volatile memory 100, the controller will first read the Flags 110, 112, 114, and 116 portion of the nonvolatile memory 104 and updates the flags portion 293 in the volatile memory 100. Then it reads the logical block address 108 of every physical block address of the nonvolatile media 104 and by tracking the flags of the given physical block address in the volatile memory 100, and the read logical block address of the physical block address in the nonvolatile memory 104, it can update the most recent physical block address assigned to the read logical block address in the volatile memory 100.

Figure 3 shows a block diagram of a system incorporating the mass storage device of the present invention. An external digital system 300 such as a host computer, personal computer and the like is coupled to the mass storage device 302 of the present invention. A logical block address is coupled via an address bus 306 to the volatile RAM array 100 and to a controller circuit 304. Control signals are also coupled to the controller 304 via a control bus 308. The volatile RAM array 1.00 is coupled for providing the physical block address to the nonvolatile RAM array 400. The controller 304 is coupled to control both the volatile RAM 100, the nonvolatile array 104, and for the generation of all flags.

A simplified example, showing the operation of the write operation according to the present invention is shown in Figures 4 through 8. Not all the information flags are shown to avoid obscuring these features of the invention in excessive detail. The data entries are shown using decimal numbers to further simplify the understanding of the invention. It will be apparent to one of ordinary skill in the art that in a preferred embodiment binary counting will be used.

Figure 4 shows an eleven entry mass storage device according to the present invention. There is no valid nor usable data stored in the mass storage device of Figure 4. Accordingly, all the physical block addresses are empty. The data stored in the nonvolatile mass storage location '6' is filled and old. Additionally, location '9' is defective and cannot be used.

The host directs the mass storage device of the example to write data pursuant to the logical block address '3' and then to '4'. The mass storage device will first write the data associated with the logical block address '3'. The device determines which is the first unused location in the nonvolatile memory. In this example, the first empty location is location '0'.

Accordingly, Figure 5 shows that for the logical block address '3', the corresponding physical block address '0' is stored and the used flag is set in physical block address '0'. The next empty location is location '1'. Figure 6 shows that for the logical block address '4', the corresponding physical block address '1' is stored and the used flag is set in physical block address '1'.

The host instructs that something is to be written to logical block address '3' again. The next empty location is determined to be location '2'. Figure 7 shows that the old flag in location '0' is set to indicate that this data is no longer usable, the used flag is set in location '2' and the physical block address in location '3' is changed to '2'.

Next, the host instructs that something is to be written to logical block address '4' again. The next empty location is determined to be location '3'. Figure 8 shows that the old flag in location '1' is set to indicate that this data is no longer usable, the used flag is set in location '3' and the physical block address in location '4' is changed to '3'. (Recall that there is generally no relation between the physical block address and the data stored in the same location.).

Figure 9 shows algorithm 1 according to the present invention. When the system of the present invention receives an instruction to program data into the mass storage (step 200), then the system attempts to locate a free block (step 202), i.e., a block having an unset (not programmed) used/free flag. If successful, the system sets the used/free flag for that block and programs the data into that block (step 206).

If on the other hand, the system is unable to locate a block having an unset used/ free flag, the system erases the flags (used/free and old/new) and data for all blocks having a set old/new flag and unset defect flag (step 204) and then searches for a block having an unset used/free flag (step 202). Such a block has just been formed by step 204. The system then sets the used/flag for that block and programs the data file into that block (step 206).

If the data is a modified version of a previously existing file, the system must prevent the superseded version from being accessed. The system determines whether the data file supersedes a previous data file (step 208). If so, the system sets the old/new flag associated with the superseded block (step 210). If on the other hand, the data file to be stored is a newly created data file, the step of setting the old/new flag (step 210) is skipped because there is no superseded block. Lastly, the map for correlating the logical address 308- to the physical addresses updated (step 212).

By following the procedure outlined above, the overhead associated with an erase cycle is avoided for each write to the memory 104 except for periodically. This vastly improves the performance of the overall computer system employing the architecture of the present invention.

In the preferred embodiment of the present invention, the programming of the flash memory follows the procedure commonly understood by those of ordinary skill in the art. In other words, the program impulses are appropriately applied to the bits to be programmed and then compared to the data being programmed to ensure that proper programming has occurred. In the event that a bit fails to be erased or programmed properly, a defect flag 148 is set which prevent that block from being used again.

Fig. 10 depicts a digital system 500 such as a digital camera employing an alternative embodiment of the present invention. Digital system 500 is illustrated to include a host 502, which may be a personal computer (PC) or simply a processor of any generic type commonly employed in digital systems, coupled to a controller circuit 506 for storing in and retrieving information from non-volatile memory unit 508. The controller circuit 506 may be a semiconductor (otherwise referred to as an "integrated circuit" or "chip") or optionally a combination of various electronic components. In the preferred embodiment, the controller circuit is depicted as a single chip device. The non-volatile memory unit 508 is comprised of one or more memory devices, which may each be flash or EEPROM types of memory. In the preferred embodiment of Fig. 10, memory unit 508 includes a plurality of flash memory devices, 510-512, each flash device includes individually addressable locations for storing information. In the preferred application of the embodiment in Fig. 10, such information is organized in blocks with each block having one or more sectors of data. In addition to the data, the information being stored may further include status information regarding the data blocks, such as flag fields, address information and the like.

The host 502 is coupled through host information signals 504 to a controller circuit 506. The host information signals comprise of address and data busses and control signals for communicating command, data and other types of information to the controller circuit 506, which in turn stores such information in memory unit 508 through flash address bus 512, flash data bus 514, flash signals 516 and flash status signals 518 (508 and 512-516 collectively referred to as signals 538). The signals 538 may provide command, data and status information between the controller 506 and the memory unit 508.

The controller 506 is shown to include high-level functional blocks such as a host interface block 520, a buffer RAM block 522, a flash controller block 532, a microprocessor block 524, a microprocessor controller block 528, a microprocessor storage block 530, a microprocessor ROM block 534, an ECC logic block 540 and a space manager block 544. The host interface block 520 receives host information signals 504 for providing data and status information from buffer RAM block 522 and microprocessor block 524 to the host 502 through host information signals 504. The host interface block 520 is coupled to the microprocessor block 524 through the microprocessor information signals 526, which is comprised of an address bus, a data bus and control signals.

The microprocessor block 524 is shown coupled to a microprocessor controller block 528, a microprocessor storage block 530 and a microprocessor ROM block 534, and serves to direct operations of the various functional blocks shown in Fig. 10 within the controller 506 by executing program instructions stored in the microprocessor storage block 530 and the microprocessor ROM block 534. Microprocessor 524 may, at times, execute program instructions (or code) from microprocessor ROM block 534, which is a non-volatile storage area. On the other hand, microprocessor storage block 530 may be either volatile, i.e., read-and-write memory (RAM), or non-volatile, i.e., EEPROM, type of memory storage. The instructions executed by the microprocessor block 524, collectively referred to as program code, are stored in the storage block 530 at some time prior to the beginning of the operation of the system of the present invention. Initially, and prior to the execution of program code from the microprocessor storage location 530, the program code may be stored in the memory unit 508 and later downloaded to the storage block 530 through the signals 538. During this initialization, the microprocessor block 524 can execute instructions from the ROM block 534.

Controller 506 further includes a flash controller block 532 coupled to the microprocessor block 524 through the microprocessor information signals 526 for providing and receiving information from and to the memory unit under the direction of the microprocessor. Information such as data may be provided from flash controller block 532 to the buffer RAM block 522 for storage (may be only temporary storage) therein through the microprocessor signals 526. Similarly, through the microprocessor signals 526, data may be retrieved from the buffer RAM block 522 by the flash controller block 532.

ECC logic block 540 is coupled to buffer RAM block 522 through signals 542 and further coupled to the microprocessor block 524 through microprocessor signals 526. ECC logic block 540 includes circuitry for generally performing error coding and correction functions. It should be understood by those skilled in the art that various ECC apparatus and algorithms are commercially available and may be employed to perform the functions required of ECC logic block 540. Briefly, these functions include appending code that is for all intensive purposes uniquely generated from a polynomial to the data being transmitted and when data is received, using the same polynomial to generate another code from the received data for detecting and potentially correcting a predetermined number of errors that may have corrupted the data. ECC logic block 540 performs error detection and/or correction operations on data stored in the memory unit 508 or data received from the host 502.

The space manager block 544 employs a preferred apparatus and algorithm for finding the next unused (or free) storage block within one of the flash memory devices for storing a block of information, as will be further explained herein with reference to other figures. As earlier discussed, the address of a block within one of the flash memory devices is referred to as PBA, which is determined by the space manager by performing a translation on an LBA received from the host. A variety of apparatus and method may be employed for accomplishing this translation. An example of such a scheme is disclosed in U.S. Pat. No. 5,485,595, entitled "Flash Memory Mass Storage Architecture Incorporating Wear Leveling Technique Without Using CAM Cells", the specification of which is herein incorporated by reference. Other LBA to PBA translation methods and apparatus may be likewise employed without departing from the scope and spirit of the present invention.

Space manager block 544 includes SPM RAM block 548 and SPM control block 546, the latter two blocks being coupled together. The SPM RAM block 548 stores the LBA-PBA mapping information (otherwise herein referred to as translation table, mapping table, mapping information, or table) under the control of SPM control block 546. Alternatively, the SPM RAM block 548 may be located outside of the controller, such as shown in Fig. 3 with respect to RAM array 100.

In operation, the host 502 writes and reads information from and to the memory unit 508 during for example, the performance of a read or write operation through the controller 506. In so doing, the host 502 provides an LBA to the controller 506 through the host signals 504. The LBA is received by the host interface block 520. Under the direction of the microprocessor block 524, the LBA is ultimately provided to the space manager block 544 for translation to a PBA and storage thereof, as will be discussed in further detail later.

Under the direction of the microprocessor block 524, data and other information are written into or read from a storage area, identified by the PBA, within one of the flash memory devices 510-512 through the flash controller block 532. The information stored within the flash memory devices may not be overwritten with new information without first being erased, as earlier discussed. On the other hand, erasure of a block of information (every time prior to being written), is a very time and power consuming measure. This is sometimes referred to as erase-before-write operation. The preferred embodiment avoids such an operation by continuously, yet efficiently, moving a sector (or multiple sectors) of information, within a block, that is being re-written from a PBA location within the flash memory to an unused PBA location within the memory unit 508 thereby avoiding frequent erasure operations. A block of information may be comprised of more than one sector such as 16 or 32 sectors. A block of information is further defined to be an individually-erasable unit of information. In the past, prior art systems have moved a block stored within flash memory devices that has been previously written into a free (or unused) location within the flash memory devices. Such systems however, moved an entire block even when only one sector of information within that block was being re-written. In other words, there is waste of both storage capacity within the flash memory as well as waste of time in moving an entire block's contents when less than the total number of sectors within the block are being re-written. The preferred embodiments of the present invention, as discussed herein, allow for "moves" of less than a block of information thereby decreasing the number of move operations of previously-written sectors, consequently, decreasing the number of erase operations.

Referring back to Fig. 10, it is important to note that the SPM RAM block 548 maintains a table that may be modified each time a write operation occurs thereby maintaining the LBA-PBA mapping information and other information regarding each block being stored in memory unit 508. Additionally, this mapping information provides the actual location of a sector (within a block) of information within the flash memory devices. As will be further apparent, at least a portion of the information in the mapping table stored in the SPM RAM block 548 is "shadowed" (or copied) to memory unit 508 in order to avoid loss of the mapping information when power to the system is interrupted or terminated. This is, in large part, due to the use of volatile memory for maintaining the mapping information. In this connection, when power to the system is restored, the portion of the mapping information stored in the memory unit 508 is transferred to the SPM RAM block 548.

It should be noted, that the SPM RAM block 548 may alternatively be nonvolatile memory, such as in the form of flash or EEPROM memory architecture. In this case, the mapping table will be stored within nonvolatile memory thereby avoiding the need for "shadowing" because during power interruptions, the mapping information stored in nonvolatile memory will be clearly maintained.

When one or more sectors are being moved from one area of the flash memory to another area, the preferred embodiment of the present invention first moves the sector(s) from the location where they are stored in the flash memory devices, i.e., 510-512, to the buffer RAM block 522 for temporary storage therein. The moved sector(s) are then moved from the buffer RAM block 522 to a free area within one of the flash memory devices. It is further useful to note that the ECC code generated by the ECC logic block 540, as discussed above, is also stored within the flash memory devices 510-512 along with the data, as is other information, such as the LBA corresponding to the data and flag fields.

Figs. 11-21 are presented to show examples of the state of a table 700 in SPM RAM block 548 configured to store LBA-PBA mapping information for identification and location of blocks (and sectors within the blocks) within the memory unit 508. Table 700 in all of these figures is shown to include an array of columns and rows with the columns including virtual physical block address locations or VPBA block address locations 702, move virtual physical address locations or MVPBA block address locations 704, move flag locations 706, used/free flag locations 708, old/new flag locations 710, defect flag locations 712 and sector move status locations 714.

The rows of table include PBA/LBA rows 716, 718 through 728 with each row having a row number that may be either an LBA or a PBA depending upon the information that is being addressed within the table 700. For example, row 716 is shown as being assigned row number '00' and if PBA information in association with LBA '00' is being retrieved from table 700, then LBA '00' may be addressed in SPM RAM block 548 at row 716 to obtain the associated PBA located in 730. On the other hand, if status information, such as flag fields, 706-712, regarding a block is being accessed, the row numbers of rows 716 - 728, such as '00', '10', '20', '30', '40', '50', 'N-1' represent PBA, as opposed to LBA, values. Furthermore, each row of table 700 may be thought of as a block entry wherein each entry contains information regarding a block. Furthermore, each row of table 700 may be addressed by an LBA.

In the preferred embodiment, each block is shown to include 16 sectors. This is due to the capability of selectively erasing an entire block of 16 sectors (which is why the block size is sometimes referred to as an "erase block size". If an erase block size is 16 sectors, such as shown in Figs. 11-21, each block entry (or row) includes information regarding 16 sectors. Row 716 therefore includes information regarding a block addressed by LBA '00' through LBA '15' (or LBA '00' through LBA '0F' in Hex. notation). The next row, row 718, includes information regarding blocks addressed by LBA '16' (or '10' in Hex.) through LBA '31' (or '1F' in Hex.) The same is true for PBAs of each block.

It should be noted however, other block sizes may be similarly employed. For example, a block may include 32 sectors and therefore an erase block size 32. In the latter situation, each block entry or row, such as 716, 718, 720..., would include information regarding 32 sectors. where each block comprises of 16 sectors (other than 16-sector block sizes may be similarly employed).

The VPBA block address locations 702 of table 700 stores information generally representing a PBA value corresponding to a particular LBA value. The MVPBA block address locations 704 store information representing a PBA value identifying, within the memory unit 508, the location of where a block (or sector portions thereof) may have been moved. The move flag locations 706 store values indicating whether the block being accessed has any sectors that may have been moved to a location whose PBA is indicated by the value in the MVPBA block address location 704 (the PBA value within 704 being other than the value indicated in VPBA block address 702 wherein the remaining block address information may be located). The used/new flag location 708 stores information to indicate whether the block being accessed is a free block, that is, no data has been stored since the block was last erased. The old/new flag location 710 stores information representing the status of the block being accessed as to whether the block has been used and re-used and therefore, old. The defect flag location 712 stores information regarding whether the block is defective. If a block is declared defective, as indicated by the value in the defect flag location 712 being set, the defective block can no longer be used. Flags 708-712 are similar to the flags 110-114 shown and described with respect to Fig. 1.

Sector move status location 714 is comprised of 16 bits (location 714 includes a bit for each sector within a block so for different-sized blocks, different number of bits within location 714 are required) with each bit representing the status of a sector within the block as to whether the sector has been moved to another block within the memory unit 508. The moved block location within the memory unit 508 would be identified by a PBA that is other than the PBA value in VPBA block address location 702. Said differently, the status of whether a sector within a block has been moved, as indicated by each of the bits within 714, suggests which one of either the VPBA block address locations 702 or the MBPBA block address locations 704 maintain the most recent PBA location for that sector.

Referring still to Fig. 11, an example of the status of the table 700 stored in SPM RAM block 548 (in Fig. 10) is shown when, by way of example, LBA '0' is being written. As previously noted, in the figures presented herein, a block size of sixteen sectors (number 0-15 in decimal notation or 0-10 in hexadecimal notation) is used to illustrate examples only. Similarly, N blocks (therefore N LBAs) are employed, numbered from 0 - N-1. The block size and the number of blocks are both design choices that may vary for different applications and may depend upon the memory capacity of each individual flash memory device (such as 510 - 512) being employed. Furthermore, a preferred sector size of 512 bytes is used in these examples whereas other sector sizes may be employed without departing from the scope and spirit of the present invention.

Assuming that the operation of writing to LBA '0' is occurring after initialization or system power-up when all of the blocks within the flash memory devices 510-512 (in Fig. 10) have been erased and are thus free. The space manager block 548 is likely to determine that the next free PBA location is '00'. Therefore, '00' is written to 730 in VPBA block address 702 of row 716 wherein information regarding LBA '0' is maintained, as indicated in table 700 by LBA row number '00'. Since no need exists for moving any of the sectors within the LBA 0 block, the MVPBA block address 704 for row 716, which is shown as location 732 may include any value, such as an initialization value (in Fig. 11, 'XX' is shown to indicate a "don't care" state).

The value in 734 is at logic state '0' to show that LBA '0' block does not contain any moved sectors. Location 736 within the used flag 708 column of row 716 will be set to logic state '1' indicating that the PBA '0' block is in use. The state of location 738, representing the old flag 710 for row 716, is set to '0' to indicate that PBA '0' block is not "old" yet. Location 740 maintains logic state '0' indicating that the PBA '0' block is not defective and all of the bits in move status location 714 are at logic state '0' to indicate that none of the sectors within the LBA '0' through LBA '15' block have been moved.

In Fig. 11, the status information for LBA '0' in row 716, such as in move flag location 706, used flag location 708, old flag location 710, defect flag location 712 and move status location 714 for all remaining rows, 716-728, of table 700 are at logic state '0'. It is understood that upon power-up of the system and/or after erasure of any of the blocks, the entries for the erased blocks, which would be all blocks upon power-up, in table 700, are all set to logic state '0'.

At this time, a discussion of the contents of one of the flash memory devices within the memory unit 508, wherein the LBA '0' block may be located is presented for the purpose of a better understanding of the mapping information shown in table 700 of Fig. 11.

Turning now to Fig. 22, an example is illustrated of the contents of the flash memory device 510 in accordance with the state of table 700 (as shown in Fig. 11). LBA '0', which within the memory unit 508 is identified at PBA '0' by controller 506 (of Fig. 10) is the location wherein the host-identified block is written. A PBA0 row 750 is shown in Fig. 22 to include data in sector data location 752. An ECC code is further stored in ECC location 754 of PBA0 row 750. This ECC code is generated by the ECC logic block 540 in association with the data being written, as previously discussed. Flag field 756 in PBA0 row 750 contains the move, used, old and defect flag information corresponding to the sector data of the block being written. In this example, in flag field 756, the "used" flag and no other flag is set, thus, flag field 756 maintains a logic state of '0100' indicating that PBA '0' is "used" but not "moved", "old" or "defective".

PBA0 row 750 additionally includes storage location for maintaining in LBA address location 758, the LBA number corresponding to PBA '0', which in this example, is '0'. While not related to the example at hand, the remaining PBA locations of LBA '0' are stored in the next 15 rows following row 750 in the flash memory device 510.

It will be understood from the discussion of the examples provided herein that the information within a PBA row of flash memory device 510 is enough to identify the data and status information relating thereto within the LBA '0' block including any moves associated therewith, particularly due to the presence of the "move" flag within each PBA row (750, 762, 764, ...) of the flash memory. Nevertheless, alternatively, another field may be added to the first PBA row of each LBA location within the flash, replicating the status of the bits in the move status location 714 of the corresponding row in table 700. This field is optionally stored in sector status location 760 shown in Fig. 22 to be included in the first PBA row of each LBA block, such as row 750, 780 and so on. Although the information maintained in location 760 may be found by checking the status of the "move" flags within the flag fields 756 of each PBA row, an apparent advantage of using location 760 is that upon start-up (or power-on) of the system, the contents of table 700 in SPM RAM block 548 may be updated more rapidly due to fewer read operations (the reader is reminded that table 700 is maintained in SPM RAM 548, which is volatile memory whose contents are lost when the system is power-down and needs to be updated upon power-up from non-volatile memory, i.e. memory unit 508).

That is, rather than reading every PBA row (altogether 16 rows in the preferred example) to update each LBA entry of the table 700 upon power-up, only the first PBA row of each LBA must be read from flash memory and stored in SPM RAM 548 thereby saving time by avoiding needless read operations. On the other hand, clearly more memory capacity is utilized to maintain 16 bits of sector status information per LBA.

In the above example, wherein location 760 is used, the value in sector status location 760 would be all '0's (or '0000' in hexadecimal notation).

In flash memory device 510, each of the rows 750, 762, 764, 768.., is a PBA location with each row having a PBA row number and for storing data and other information (data and other information are as discussed above with respect to row 750) for a sector within a block addressed by a particular LBA. Furthermore, every sixteen sequential PBA rows represents one block of information. That is, PBA rows 750, 762, 764 through 768, which are intended to show 16 PBA rows correspond to LBA 0(shown as row 716 in table 700 of Fig. 11) and each of the PBA rows maintains information regarding a sector within the block. The next block of information is for the block addressed by LBA '10' (in Hex.) whose mapping information is included in row 718 of table 700, and which is stored in locations starting from '10' (in hexadecimal notation, or '16' in decimal notation) and ending at '1F' (in hexadecimal notation, or '31') in the flash memory device 510 and so on.

Continuing on with the above example, Fig. 12 shows an example of the state of table 700 when LBA 0 is again being written by the host. Since LBA 0 has already been written and is again being written without first being erased, another free location within the memory unit 508 (it may serve helpful to note here that the blocks, including their sectors, are organized sequentially and continuously through each of the flash memory devices of memory unit 508 according to their PBAs such that for example, the next flash memory device following device 510 picks up the PBA-addressed blocks where flash memory device 510 left off, an example of this is where flash memory device 510 includes PBAs of 0-FF (in Hex.) and the next flash memory device, which may be 512, may then include 100-1FF (in Hex.)) is located by space manager 544 for storage of the new information. This free location is shown to be PBA '10' (in Hexadecimal notation, or 16 in decimal notation). In row 718, where the entries for LBA '10' will remain the same as shown in Fig. 11 except the used flag in location 742 will be set (in the preferred embodiment, a flag is set when it is at logic state '1' although the opposite polarity may be used without deviating from the present invention) to indicate that the PBA '10' is now "in use".

The entries in row 716 are modified to show '10' in MVPBA block address location 732, which provides the PBA address of the moved portion for the LBA '00' block. The move flag in location 734 is set to logic state '1' to indicate that at least a portion (one or more sectors) of the LBA '00' block have been moved to a PBA location other than the PBA location indicated in location 730 of table 700. Finally, the bits of the move status location 714 in row 716 are set to '1000000000000000' (in binary notation, or '8000' in hexadecimal notation), reflecting the status of the moved sectors within the block LBA '00'. That is, in this example, '8000' indicates that the first sector, or sector '0', within LBA '00' block has been moved to a different PBA location.

Referring now to Fig. 22, the state of table 700 in Fig. 12 will affect the contents of the flash memory device 510 in that the moved sector of the LBA '0' block will now be written to PBA '10' in row 780. Row 780 will then include the data for the moved sector, which is 512 bytes in size. With respect to the moved sector information, row 780 further includes ECC code, a copy of the values in flag locations 734 - 740 of table 700 (in Fig. 12), and LBA '00' for indicating that the data in row 780 belongs to LBA '00' and may further include the move status for each of the individual sectors within the LBA '0' block.

While not specifically shown in the figure, the move flag within location 756 of PBA row 750 is set to indicate that at least a portion of the corresponding block has been moved. The value stored in the move status location 714 of row 716 (in Fig. 12), which is '8000' in Hex., is also stored within location 760 of the row 750. As earlier noted, this indicates that only sector '0' of PBA '0' was marked "moved" and the new block LBA '0' was written to PBA '10' in flash memory. Without further detailed discussions of Fig. 22, it should be appreciated that the examples to follow likewise affect the contents of the flash memory device 510.

Fig. 13 shows the status of table 700 when yet another write operation to LBA '00' is performed. The values (or entries) in row 716 remain the same as in Fig. 12 except that the value in location 732 is changed to '20' (in Hex. Notation) to indicate that the moved portion of block LBA '00' is now located in PBA location '20' (rather than '10' in Fig. 12). As in Fig. 12, the value in move status location 714, '8000', indicates that the first sector (with PBA '00') is the portion of the block that has been moved.

Row 718 is modified to show that the LBA '10' block is now old and can no longer be used before it is erased. This is indicated by the value in location 744 being set to logic state '1'. The entries for LBA '20', row 720, remain unchanged except that location 746 is modified to be set to logic state '1' for reflecting the state of the PBA '20' block as being in use. It is understood that as in Figs. 11 and 12, all remaining values in table 700 of Fig. 13 that have not been discussed above and are not shown as having a particular logic state in Fig. 13 are all unchanged (the flags are all set to logic state '0').

Continuing further with the above example, Fig. 14 shows the state of table 700 when yet another write to LBA '0' occurs. For ease of comparison, there is a circle drawn around the values shown in Fig. 14, which are at a different logic state with respect to their states shown in Fig. 13. In row 716, everything remains the same except for the new moved location, indicated as PBA '30', shown in location 732. PBA '30' was the next free location found by the space manager 544. As previously noted, this value indicates that a portion of the block of LBA '0' is now in PBA '30'; namely, the first sector (shown by the value in 714 of row 716 being '8000') in that block has been moved to PBA '30' in the flash memory device 510.

Row 718 remains the same until it is erased. The flags in locations 742 and 744 are set to logic state '0'. Row 720 also remains unchanged except for the value in its old flag 710 column being modified to '1' to show that the block of PBA '20' is also old and can not be used until first erased. Row 722 remains the same except for the value in its used flag 708 column being changed to logic state '1' to show that the block of LBA '30' is now in use.

Fig. 15 is another example of the state of table 700, showing the state of table 700 assuming that the table was at the state shown in Fig. 13 and followed by the host writing to LBA '5'. Again, the changes to the values in table 700 from Fig. 13 to Fig. 15 are shown by a circle drawn around the value that has changed, which is only one change.

When writing to LBA '5', it should be understood that the LBA entries of rows 716, 718, 720, etc. are only for LBA '00', LBA '10', LBA '20', so on, and therefore do not reflect an LBA '5' entry. The reader is reminded that each of the LBA row entries is for a block of information with each block being 16 sectors in the preferred embodiment. For this reason, LBA '5' actually addresses the fifth sector in row 716. Since PBA '20' was used to store LBA '0', only the sector within PBA '20', corresponding to LBA '5', is yet not written and "free". Therefore, the data for LBA '5' is stored in PBA '20' in sector '5'. The move status location 714 of row 716 will be modified to logic state '8400' (in Hex. Notation). This reflects that the location of the first and fifth sectors within LBA '0' are both identified at PBA '20' in the flash memory device 510. The remaining values in table 700 of Fig. 15 remain the same as those shown in Fig. 13.

Figs. 16-18 show yet another example of what the state of table 700 may be after either power-up or erasure of the blocks with the memory unit 508. In Figs. 16 and 17, the same write operations as those discussed with reference to Figs. 11 and 12 are performed. The state of table 700 in Figs. 16 and 17 resembles that of Figs. 11 and 12, respectively (the latter two figures have been re-drawn as Figs. 16 and 17 for the sole convenience of the reader). Briefly, Fig. 16 shows the state of table 700 after a write to LBA '0' and Fig. 17 shows the state of table 700 after another write to LBA '0'.

Fig. 18 picks up after Fig. 17 and shows the state of table 700 after the host writes to LBA '5'. As indicated in Fig. 18, LBA '5' has been moved to PBA '10' where LBA '0' is also located. To this end, MBPBA block address location 732 is set to '10' in row 716 and the move flag is set at location 734 in the same row. Moreover, the state of move status location 714 in row 716 is set to '8400' (in Hex.) indicating that LBA '0' and LBA '5' have been moved, or that the first and fifth sectors within LBA '00' are moved. Being that these two sectors are now located in the PBA '10' location of the flash memory device 510, the move flag for each of the these sectors are also set in the flash memory device 510. It should be understood that LBA '5' was moved to PBA '10' because remaining free sectors were available in that block. Namely, even with LBA '0' of that block having been used, 15 other sectors of the same block were available, from which the fifth sector is now in use after the write to LBA '5'.

Continuing on with the example of Fig. 18, in Fig. 19, the state of the table 700 is shown after the host writes yet another time to LBA '0'. According to the table, yet another free PBA location, '20', is found where both the LBA '5' and LBA '0' are moved. First, LBA '5' is moved to the location PBA '10' to PBA '20' and then the new block of location LBA '0' is written to PBA '20'. As earlier discussed, any time there is a move of a block (for example, here the block of LBA '5' is moved) it is first moved from the location within flash memory where it currently resides to a temporary location within the controller 506, namely with the buffer RAM block 522, and then it is transferred from there to the new location within the flash memory devices.

The used flag in location 746 of row 720 is set to reflect the use of the PBA '20' location in flash memory and the old flag in location 744 is set to discard use of PBA '10' location until it is erased. Again, in flash memory, the state of these flags as well as the state of the move flag for both the LBA '0' and LBA '5' sectors are replicated.

Fig. 20 picks up from the state of the table 700 shown in Fig. 18 and shows yet another state of what the table 700 may be after the host writes to LBA '5'. In this case, the block of LBA '0' is first moved from location PBA '10' within the flash memory device 510 wherein it is currently stored to location PBA '20' of the flash memory. Thereafter, the new block being written to LBA '5' by the host is written into location PBA '20' of the flash memory. The flags in both table 700 and corresponding locations of the flash memory device 510 are accordingly set to reflect these updated locations.

Fig. 21 also picks up from the state of the table 700 shown in Fig. 18 and shows the state of what the table 700 may be after the host writes to LBA '7'. In this case, the new block is simply written to location PBA '10' of the flash memory since that location has not yet been used. Additionally, three of the bits of the move status location 714 in row 716 are set to show that LBA '0', LBA '5' and LBA '7' have been moved to another PBA location within the flash memory. Location 732 shows that the location in which these three blocks are stored is PBA '10'.

As may be understood from the discussion presented thus far, at some point in time, the number of sectors being moved within a block makes for an inefficient operation. Thus, the need arises for the user to set a threshold for the number of sectors within a block that may be moved before the block is declared "old" (the old flag is set) and the block is no longer used, until it is erased. This threshold may be set at, for example, half of the number of sectors within a block. This is demonstrated as follows: For a block having 16 sectors, when 8 of the sectors are moved into another block, the "original" block and the "moved" block (the block in which the moved sectors reside) are combined into the same PBA block. The combined PBA block may be stored in a new block altogether or, alternatively, the "original" block may be combined with and moved into the "moved" block. In the latter case, the "original" block is then marked as "old" for erasure thereof. If the combined PBA block is stored in a new block, both of the "original" and the "moved" blocks are marked as "old".

Fig. 23 depicts a general flow chart outlining some of the steps performed during a write operation. It is intended to show the sequence of some of the events that take place during such an operation and is not at all an inclusive presentation of the method or apparatus used in the preferred embodiment of the present invention.

The steps as outlined in Fig. 23 are performed under the direction of the microprocessor block 524 as it executes program code (or firmware) during the operation of the system. When the host writes to a block of LBA M, step 800, the space manager block 544, in step 802, checks as to whether LBA M is in use by checking the state of the corresponding used flag in table 700 of the SPM RAM block 548. If not in use, in step 804, a search is performed for the next free PBA block in memory unit 508. If no free blocks are located, an "error" state is detected in 808. But where a free PBA is located, in step 806, its used flag is marked (or set) in table 700 as well as in flash memory. In step 810, the PBA of the free block is written into the VPBA block address 702 location of the corresponding LBA row in table 700.

Going back to step 802, if the LBA M block is in use, search for the next free PBA block is still conducted in step 812 and upon the finding of no such free block, at 814, an "error" condition is declared. Whereas, if a free PBA location is found, that PBA is marked as used in table 700 and flash memory, at step 816. Next, in step 818, the state of the block is indicated as having been moved by setting the move flag as well as the setting the appropriate bit in the move status location 714 of table 700. The new location of where the block is moved is also indicated in table 700 in accordance with the discussion above.

Finally, after steps 818 and 810, data and all corresponding status information, ECC code and LBA are written into the PBA location within the flash memory.

As earlier indicated, when a substantial portion of a block has sectors that have been moved (in the preferred embodiment, this is eight of the sixteen sectors), the block is declared "old" by setting its corresponding "old" flag. Periodically, blocks with their "old" flags set, are erased and may then be re-used (or re-programmed, or re-written).

As can be appreciated, an advantage of the embodiments of Figs. 10-23 is that a block need not be erased each time after it is accessed by the host because if for example, any portions (or sectors) of the block are being re-written, rather than erasing the block in the flash memory devices or moving the entire block to a free area within the flash, only the portions that are being re-written need be transferred elsewhere in flash, i.e. free location identified by MVPA block address. In this connection, an erase cycle, which is time consuming is avoided until later and time is not wasted in reading an entire block and transferring the same.

In an alternative embodiment, there may be less information pertaining to each sector that is stored in the flash memory. Likewise, the table 700 in the SPM RAM block 548 (in Fig. 10) maintains less information as explained below.

Fig. 24 shows an example of the information a flash memory chip (such as flash memory device 510) may store. In Fig. 24 is shown, N blocks, blocks 1000, 1002, ..., 1004, with each block having up to m sectors Using block 1000 as an example, each of the sectors 1006, 1008, ..., 1010 within block 1000 includes a data field 1052 and an ECC field 1014. In the preferred embodiment, the data field 1052 contains 512 bytes and the ECC field 1014 contains 4 bytes of information although other sizes of both data and ECC may be employed without departing from the spirit of the present invention.

The first sector in each block, for example sector 1006 of block 1000, additionally includes a defect flag 1016 indicating whether the block is defective or not. In the preferred embodiment, if any of the sectors within a block are defective, the entire block is marked defective as further explained below.

Each sector within a block may further contain a spare area 1018. In each block, there is included an LBA 1020 for identifying the block within the flash memory unit, as will be explained in greater detail shortly. It is however, important to note that the LBA 1020 while shown in Fig. 24 to be located within the last sector 1010 of block 1000, may be alternatively located within any other sector of block 1000 and further within any area within the sector in which it is located. For example, LBA 1020 may be located within the spare area 1022 of sector 1008 or it may be located at 1024, which is right before the location where data is stored for the sector 1008 of block 1000. The same sector organization that is used for block 1000 is also used for the remaining blocks such as blocks 1002 and 1004 within the flash memory devices.

Similarly, the location of the defect flag 1012 may be anywhere within the block. For example, the defect flag 1012 may be alternatively stored at 1024 or at 1022. But there is only one defect flag stored per block. An aspect of the present invention relates to the way in which the defect flag is employed.

During manufacturing of flash memory chips, defects within the memory are commonly identified and marked by the chip manufacturer. This is typically done by writing a predefined pattern (byte-wide) in a predetermined location within a defective block. That is, the manufacturer will erase the flash chip and a value, such as all '1's, will then be carried by each memory cell within the flash chip that was successfully erased. If a sector or a cell within a block is defective, the manufacturer will set a manufacturing defect flag located somewhere within the defective block (not shown in Fig. 24) to a predetermined value. This manufacturing defect flag is not the same as defect flag 1012 and its location is determined by the manufacturer and not by design choice. These manufacturer-identified defective blocks are then ignored (or not used) during system operation. That is, the space manager block 544 (in Fig. 10) keeps track of these defective blocks and knows not to use them when it searches for a free block within the flash memory devices.

Other than defects detected during manufacturing of flash chips, there may be additional defects developed during operation of the chips due to wearing as discussed earlier. These defects are sometimes referred to as "grown defects" by the industry at-large and must be accounted for by a system in which using flash memory devices are employed.

The case of "grown defects" will be explained in the context of system operations. In a preferred embodiment of the present invention, after erasure of the block 1000, the defect flag 1012 (in Fig. 24) is programmed to indicate whether the erased block is defective or not. That is, if the cells in the erased block 1000 were successfully erased, the defect flag 1012 is set to a predefined value such as a byte-wide value of '55' (in Hex.) indicating that the block is not defective. If on the other hand, the block is not successfully erased, a value other than '55', such as '00', is written to the defect flag 1012 or alternatively, no value is written and whatever the state of the defect flag was after erasure (commonly, an erased state is all '1's (or 'FF' in Hex.))

will be maintained. In the latter case however, the erased state of the defect flag must be a value other than '55' in order to distinguish successful erasures.

The use of the defect flag 1012 is especially noted after power-up of the system. The space manager block 544 takes note of which blocks are defective by quickly scanning the defect flags of each block and identifying those blocks that are not defective for later by searching for the value '55' in the defect flag 1012 of each block. If a block is not defective after power-up and later becomes defective, its defect flag is modified to so designate and the next time the system is powered-up, the defective block is noted by the space manager and any use of the block is avoided.

During a write operation, if a failure of one or more cells of a block occurs, the block is first erased and then marked as being defective by writing a value other than '55' (such as '00') to the defect flag 1012 of the defective block. Alternatively, there is no value written to the defect flag 1012 of the defective block.

One of the advantages of having a defect flag such as described thus far is that during system power-up, the space manager block 544 (in Fig. 10) is able to quickly find blocks that are not defective.

In an alternative embodiment, as shown in Fig. 25, the LBA for each block within the flash memory devices is stored within two different sectors of the same block. For example, after block 1000 has been erased and during the first write operation of block 1000 following its erasure, the space manager block 544 determines that block 1000 or a portion thereof is free to be written into. The LBA associated with block 1000 is then programmed into two locations 1030 and 1020 (within sectors 1006 and 1010, respectively) of block 1000.

Thereafter, each time power to the system is temporarily interrupted, or upon power-up, the two LBAs in locations 1030 and 1020 are compared to each other and if they match and the defect flag 1012 indicates that the block is not defective, the block continues to be used for information storage under the direction of the space manager block 544. If however, the defect flag 1012 indicates that the block is not defective but the two LBAs do not match, then the block 1000 will still continue to be used but it will first be erased prior to further re-use. The latter case may arise, for example, when the system was writing to block 1000 and the first several sectors starting with sector 0 were written but then there was a power interruption and information was therefore not written to the remaining sectors of block 1000. In this case, the LBA values in locations 1030 and 1020 will likely not match because sector m-1, or sector 1010, of block 1000 was not written into prior to the power interruption. Upon power restoration, since not completely written, the block 1000 should not be used in its state as it was prior to the power interruption. Accordingly, upon detecting a mismatch between the two LBA values in locations 1030 and 1020, the controller 506 (in Fig. 10) will erase the block 1000 prior to its re-use.

Where only one LBA is used per block, as shown in Fig. 24, obviously, the comparison of two LBAs in each block is not performed yet detection of blocks that are defective is maintained through the use of the defect flag as explained earlier. Additionally, the LBA may be written last, that is, after all of the sectors of the block are written. Writing the LBA last is an added measure of successfully having written to the block.

To illustrate another aspect of the present invention, Fig. 26 shows, on the left-hand side, an example of the contents of several blocks of the flash memory device (this is the same information that is illustrated in Fig. 24) and, on the right-hand side, an example of the contents of the table 700, which corresponds to the block information that is on the left-hand side of the figure. The purpose of showing the block contents is only to serve as convenience to the reader in understanding the correlation between the LBA-PBA addressing of the blocks with respect to the table 700. The example shown in Fig. 26 is to illustrate the mapping of this alternative embodiment between the addresses of the blocks as they are stored in flash memory and the addresses of the same blocks as that information is stored in the SPM RAM block 548.
In the table 700, there is shown to be stored in a column, a virtual PBA field 1036 and a flag field 1038 including a defect flag 1012 for each row. Rows 1040, 1042, 1044, ..., 1050 each correspond to an LBA and are addressed by '0', '1','2',...'Z', respectively. The virtual PBA field 1036 for each row actually contains an LBA address pointer that points to a block within the flash memory having 16 sectors stored therein. In this respect, the virtual PBA serves as the PBA for 16 sectors (the number of sectors may be other than sixteen in alternative embodiments).

If the host sends a command to read for example, LBA 05, the controller 506, in Fig. 10, first masks the four least significant bits (LSBs) of the value '05' to obtain the value '00'. Then using '00' as the row address for table 700, the row 1040 in Fig. 26 is addressed. Note that the reason the four LSBs of the LBA value sent by the host is masked is because there are 16 sectors being represented by each row of table 700. 16 sectors is two to the power of 4 in binary terms which translates to 4 bits. If for example each row represented 32 sectors, then the 5 LSBs of the LBA sent by the host would be masked.

Once the row 1040 in the SPM RAM block 548 has been addressed, the value in the virtual PBA field 1036 for the row 1040 is retrieved, which in this case is '00'. '00' is then used as the pointer to the blocks stored within the flash memory devices. That is, in this example, block 0, or the block 1000 (on the left-hand side of Fig. 26) is addressed. But to get to the sector that was intended to be read, the 4 bits that were initially masked from the LBA value sent by the host, are used to specifically address sector '5' (not shown), which is the sixth sector, within the block 1040. The data stored in the data field 1052 of sector 5 (not shown) is then retrieved or read. Next, the ECC field 1014 of that same sector is read.

The LBA location 1020 should have the value '00', which is the same address value that is stored in the virtual PBA field 1036 of the row 1040 of table 700. This is intended for identifying the block 1000 in the flash memory devices as block '00' during the power-up routine, to update the space manager. Although, the LBA location 1020 does not represent the LBA value sent by the host as noted above. To avoid such confusion, the value in location 1020 may be referred to as a 'cluster' address rather than an LBA address.

To illustrate another example in the context of a write operation, if the host commands the controller 506 to write to LBA '17' (in decimal notation), the controller masks the LSBs of the hexadecimal notation of '17' (which is '11'). The masked version of '17' is '1' in hexadecimal notation. Thus, the row 1042 is addressed in table 700 and a the address of a free block, as found by the space manager block 544, is placed in the virtual PBA field 1036 of the row 1042. In this case, block 2 is found as the next free block. Within the flash memory devices, block 2, which is the block 1002, is addressed. The sector where information is actually written into is sector 1 of the block 1002 (the second sector in that block) because the 4 LSBs of the LBA value '11' sent by host. The data and ECC are then written into the sector 1 of block 1002.

If the preferred embodiment example of Fig. 26 is used with the embodiment discussed with reference to Fig. 25, the LBA (or cluster) address is written in two places within each of the blocks.

Although the present invention has been described in terms of specific embodiments, it is anticipated that alterations and modifications thereof will no doubt become apparent to those skilled in the art. It is therefore intended that the following claims be interpreted as covering all such alterations and modification as fall within the true spirit and scope of the invention.

What is claimed is:

CLAIMS

1. A storage device having nonvolatile memory and being coupled to a host for storing information, organized in blocks, in the nonvolatile memory, each block having associated therewith a logical block address (LBA) and a physical block address (PBA), the LBA provided by the host to the storage device for identifying a block to be accessed, the PBA developed by the storage device for identifying a free location within the nonvolatile memory wherein the accessed block is be stored, each stored block within the nonvolatile being selectively erasable and further having one or more sectors, the storage device comprising:

(a) a memory device for storing a table defined by n LBA rows, each of the LBA rows being uniquely addressable by an LBA and for storing a virtual PBA for identifying the location of the stored block, a move virtual PBA for identifying the location of a portion of the stored block, and status information including flag means for indicating whether any sectors of the stored block have been moved to the move virtual PBA location within the nonvolatile memory;

(b) means for receiving from the host, a block of information identified by a particular LBA to be stored in the nonvolatile memory, for developing the virtual PBA if the particular LBA is "unused" and for developing the move virtual PBA if the particular LBA is "used",

wherein portions of a block may be stored in more than one PBA-identified location within the nonvolatile memory to avoid an erase operation each time the host writes to the storage device and to avoid transfer of the entire block to a free location within the nonvolatile memory each time a portion of the block is being re-written.

2. A storage device as recited in claim 1, wherein the status information of each of the LBA rows further includes a sector move status field for identifying one or more of the sectors of the stored block that are not stored in the location of the nonvolatile memory identified by the virtual PBA.

3. A storage device as recited in claim 1, wherein the status information of each of the LBA rows further includes a sector move status field for identifying which of the sectors of the stored block are no longer in the location identified by the virtual PBA.

4. A storage device as recited in claim 1 further including one or more nonvolatile memory devices, each device having a plurality of storage locations with each location uniquely addressable by a PBA for storing a sector of a block.

5. A storage device as recited in claim 4, wherein each of the storage locations of the nonvolatile memory device is further for storing a copy of said flag means.

6. A storage device as recited in claim 4, wherein each storage location of the nonvolatile devices includes a sector move status location for storing sector move information to identify which sectors of the stored block are moved.

7. A storage device as recited in claim 3 and 6, wherein upon power-up, the sector move information for each PBA is read from the nonvolatile memory devices and stored in the sector move status field of each corresponding LBA row of the table.

8. A storage device as recited in claim 4, wherein the particular block is stored in the storage location of one of the nonvolatile devices addressed by the virtual PBA.

9. A storage device as recited in claim 4, wherein the particular block is stored in the storage location of one of the nonvolatile devices addressed by the move virtual PBA.

10. A storage device as recited in claim 1 wherein said flag means further includes an "old" flag means for identifying a corresponding block as being "old" when set and a "used" flag means for identifying a corresponding block as being "in use" when set.

11. A storage device as recited in claims 3 and 10 wherein if the sector move status field indicates that half of the sectors of the stored block are no longer in the virtual PBA location, the corresponding "old" flag means of the stored block is set.

12. A storage device as recited in claim 1 wherein said memory device is comprised of volatile memory.

13. A storage device as recited in claim 12 wherein said volatile memory is RAM.

14. In a storage device for use with a host, the storage device having a controller and one or more flash memory devices, a method for accessing information within the flash memory devices under the direction of the controller, the information organized in sectors with one or more sectors defining a block wherein each block is selectively erasable having associated therewith an LBA provided by the host and a PBA developed by the controller for identifying an unused location within the flash memory devices wherein a sector may be stored, the method comprising:

(a) allocating a table within the controller defined by rows, each row being individually addressable by an LBA and configured to store a virtual PBA, a moved virtual PBA, and status information for use in determining the location of a sector that has been moved from the virtual PBA to the moved virtual PBA within the flash memory devices;

(b) providing to the controller, a particular LBA identifying a block being accessed by the host;

(c) developing a PBA in association with the particular LBA;

(d) using the particular LBA to address an LBA row;

(e) storing the PBA associated with the particular LBA in the addressed LBA row;

(f) upon further access of the block identified by the particular LBA, developing a moved virtual PBA;

(g) storing the moved virtual PBA in the addressed LBA row; and

(h) modifying the status information of the addressed LBA row to indicate whether any sectors within the addressed LBA row have been moved,

wherein sectors of a block are moved to unused locations within the flash memory devices to avoid erase-before-write operations each time the block is accessed.

15. A method for accessing information as recited in claim 14 wherein the status information further includes a move flag means for indicating whether any portion of the particular block is stored in a location within the flash memory devices identified by the moved virtual PBA.

16. A method for accessing information as recited in claim 14, further including storing the table in volatile memory.

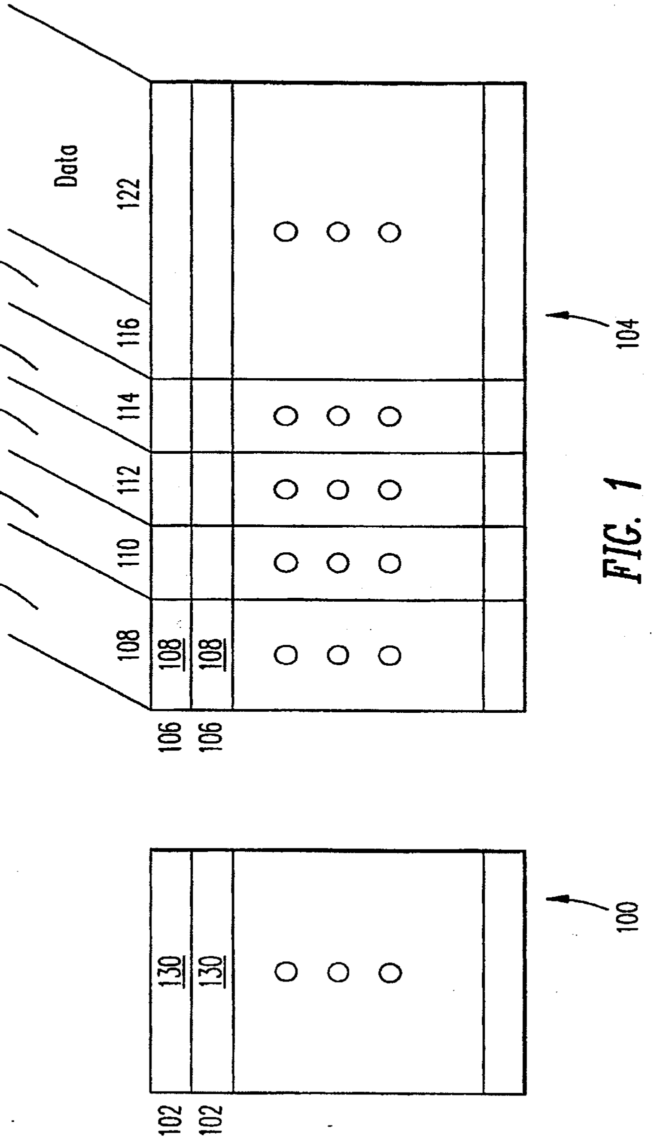
17. A storage device having nonvolatile memory and being coupled to a host for storing

information, organized in blocks, in the nonvolatile memory, each block having associated therewith a logical block address (LBA) and a physical block address (PBA), the LBA provided by the host to the storage device for identifying a block to be accessed, the PBA developed by the storage device for identifying a free location within the nonvolatile memory wherein the accessed block is to be stored, each stored block within the nonvolatile being selectively erasable and further having one or more sectors, the storage device comprising:

a memory device for storing a table defined by an LBA rows, each of the LBA rows being uniquely addressable by an LBA and for storing a virtual PBA for identifying the location of the stored block, a move virtual PBA for identifying the location of a portion of the stored block, and status information including flag means for indicationg wherher any sectors of the stored block have been moved to the move virtual PBA location within the nonvolatile memory;

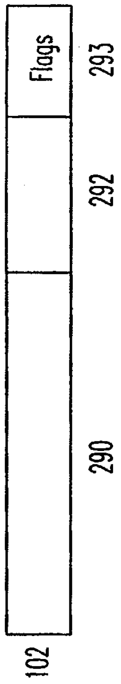
means for receiving from the host, a block of information identified by a particular LBA to be stored in the nonvolatile memory, for developing the virtual PBA if the particular LBA is "unused" and for developing the move virtual PBA if the particular LBA is "used"; defect flag stored within each of the blocks and being set to a predetermined value for identifying successfully erased blocks,

wherein portions of a block may be stored in more than one PBA-identified location within the nonvolatile memory to avoid an erase operation each time the host writes to the storage device and to avoid transfer of the entire block to a free location within the nonvolatile memory each time a portion of the block is being re-written.



Logical Address
Old/New Flag
Used/Free Flag
Defect Flag
Single/Sector Flag
Data

FIG. 1



Flags

FIG. 2

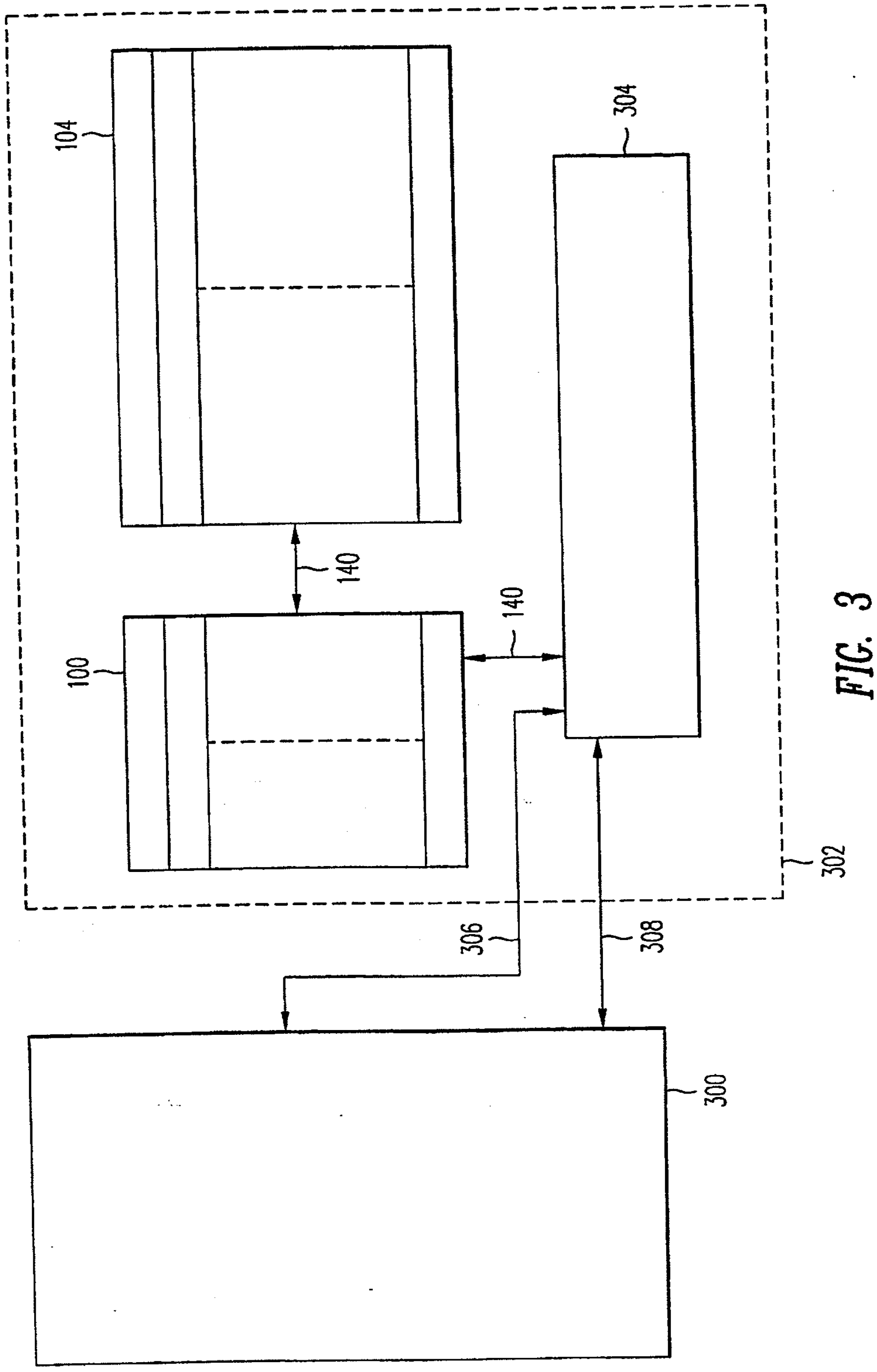

104
304
140
140
100
302
306
308
300

FIG. 3

Actual LBA/ Actual PBA	Virtual PBA	old	used	def
0		0	0	0
1		0	0	0
2		0	0	0
3		0	0	0
4		0	0	0
5		0	0	0
6		1	1	0
7		0	0	0
8		0	0	0
9		0	0	1
10		0	0	0

FIG. 4

Actual LBA/ Actual PBA	Virtual PBA	old	used	def
0		0	1	0
1		0	0	0
2		0	0	0
3	0	0	0	0
4		0	0	0
5		0	0	0
6		1	1	0
7		0	0	0
8		0	0	0
9		0	0	1
10		0	0	0

FIG. 5

Actual LBA/ Actual PBA	Virtual PBA	old	used	def
0		0	1	0
1		0	1	0
2		0	0	0
3	0	0	0	0
4	1	0	0	0
5		0	0	0
6		1	1	0
7		0	0	0
8		0	0	0
9		0	0	1
10		0	0	0

FIG. 6

Actual LBA/ Actual PBA	Virtual PBA	old	used	def
0		1	1	0
1		0	1	0
2		0	1	0
3	2	0	0	0
4	1	0	0	0
5		0	0	0
6		1	1	0
7		0	0	0
8		0	0	0
9		0	0	1
10		0	0	0

FIG. 7

Actual LBA/ Actual PBA	Virtual PBA	old	used	def
0		1	1	0
1		1	1	0
2		0	1	0
3	2	0	1	0
4	3	0	0	0
5		0	0	0
6		1	1	0
7		0	0	0
8		0	0	0
9		0	0	1
10		0	0	0

FIG. 8

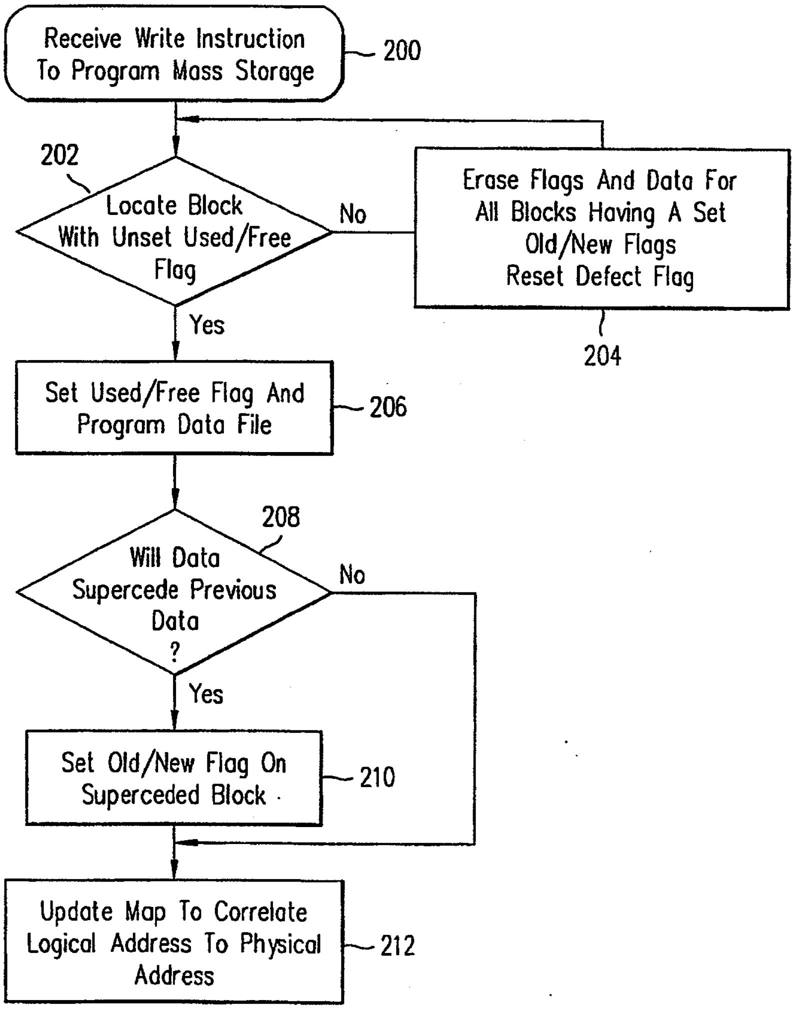

Receive Write Instruction To Program Mass Storage
200
202
Locate Block With Unset Used/Free Flag
No
Erase Flags And Data For All Blocks Having A Set Old/New Flags Reset Defect Flag
204
Yes
Set Used/Free Flag And Program Data File
206
208
Will Data Supercede Previous Data ?
No
Yes
Set Old/New Flag On Superceded Block
210
Update Map To Correlate Logical Address To Physical Address
212

FIG. 9

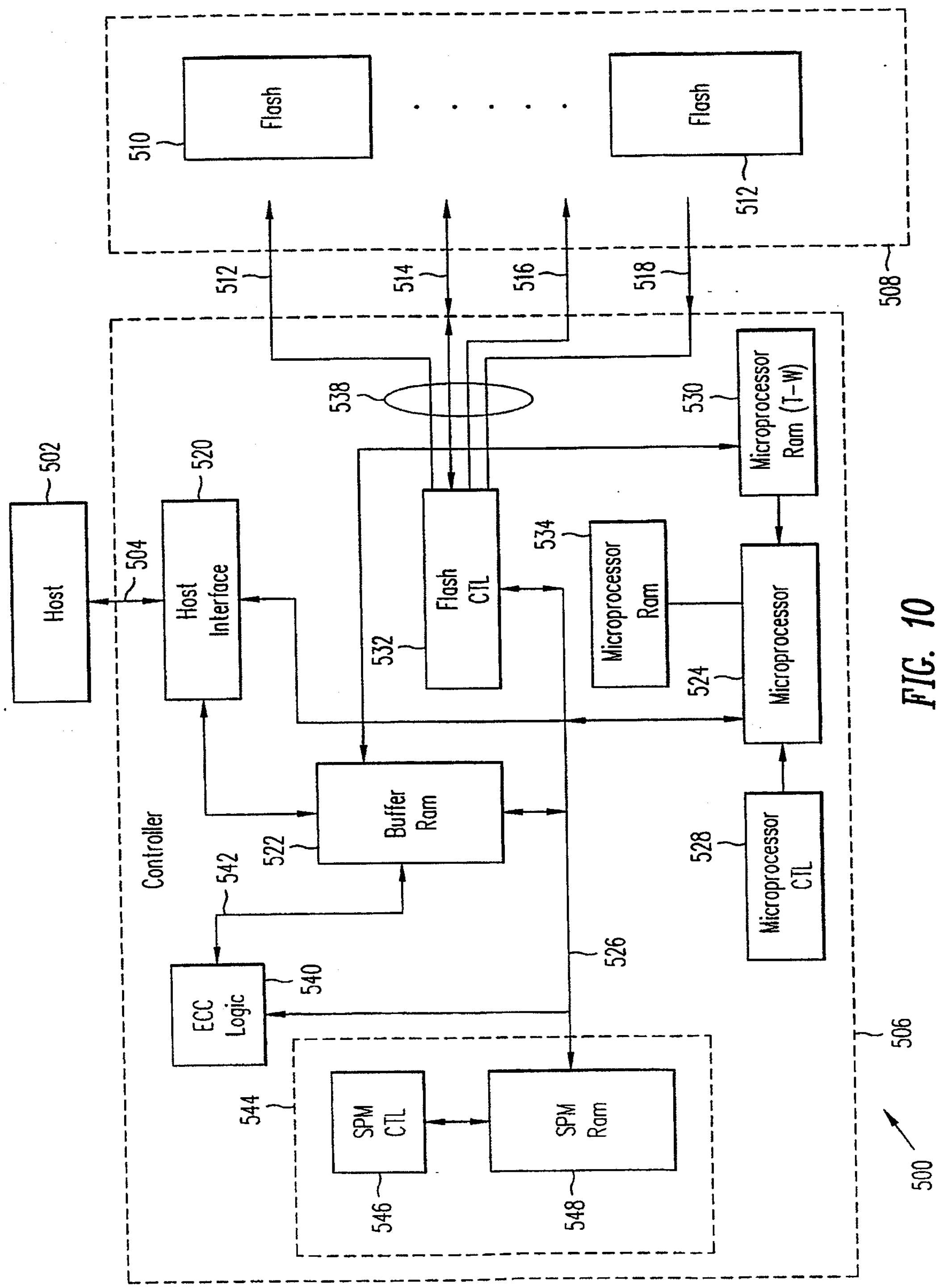
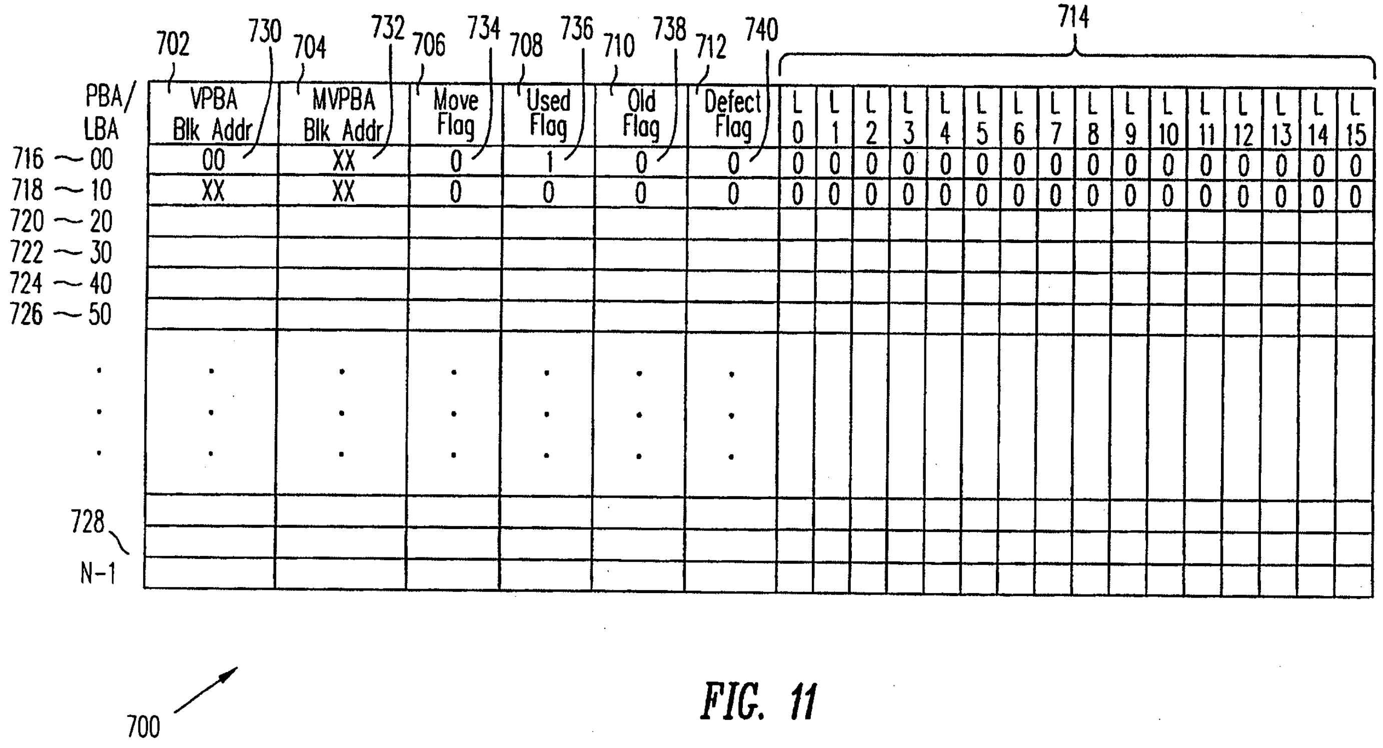
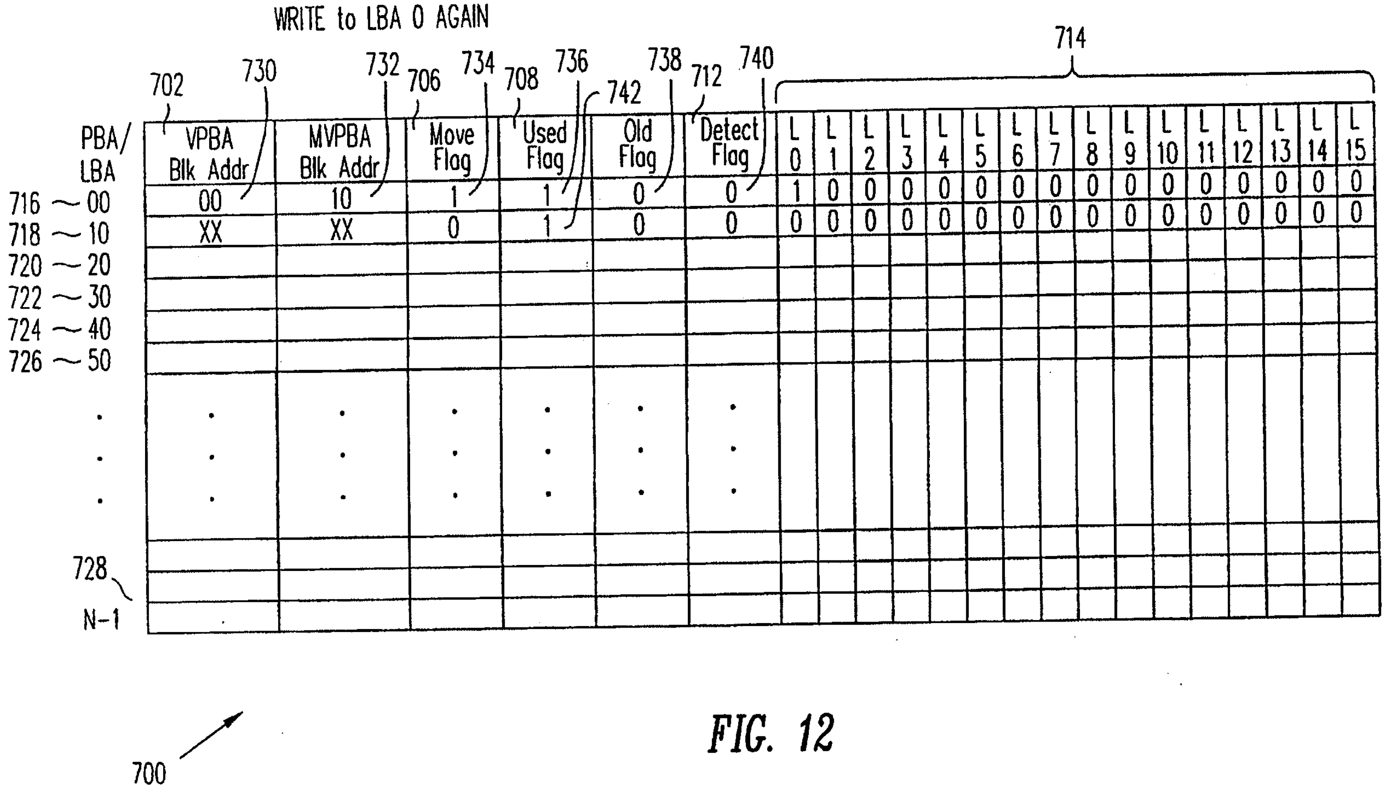

Flash
Flash
Microprocessor Ram (T-W)
Host
Host Interface
Flash CTL
Microprocessor Ram
Microprocessor
Buffer Ram
Controller
Microprocessor CTL
ECC Logic
SPM CTL
SPM Ram

FIG. 10


702
730
704
732
706
734
708
736
710
738
712
740
714
716
718
720
722
724
726
728
700

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20																						
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

FIG. 11

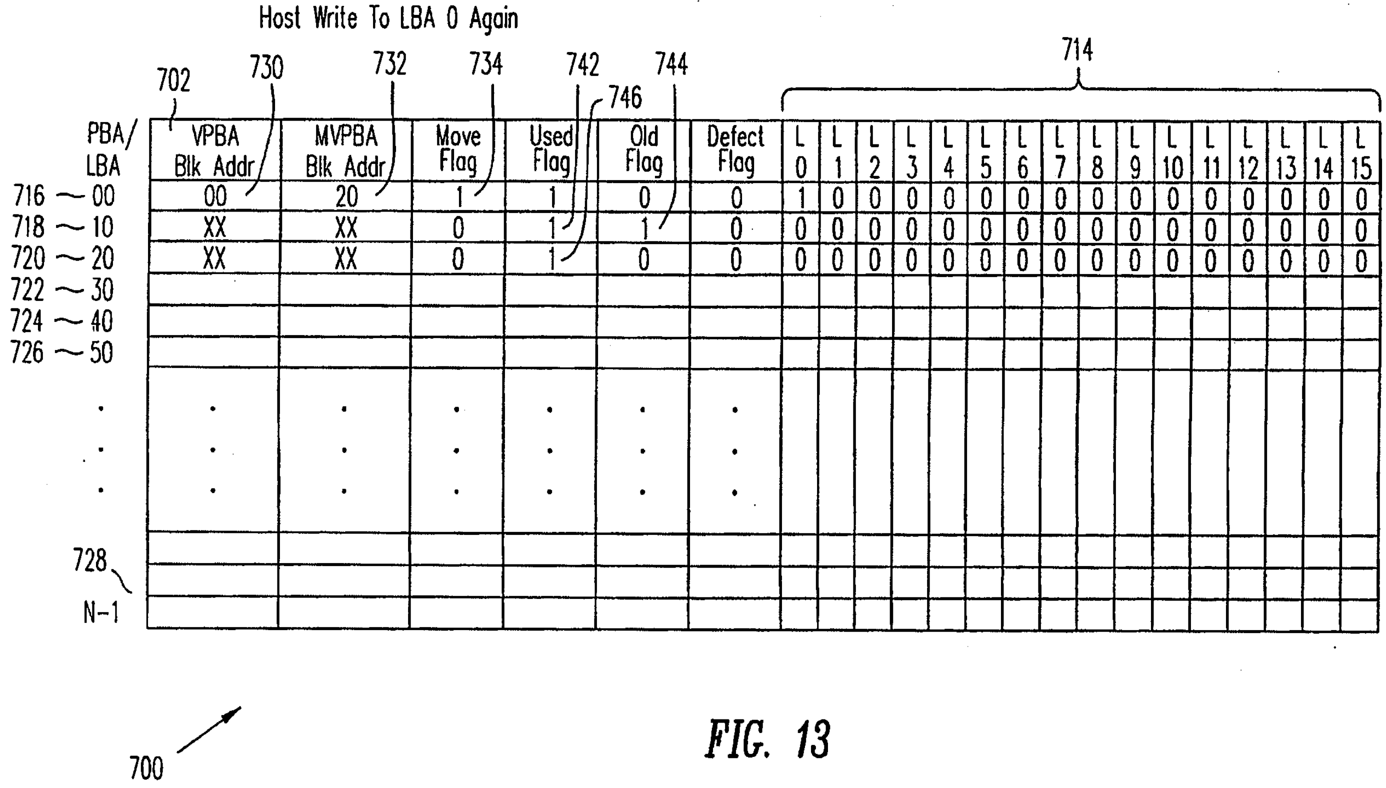

702
730
732
706
734
708
736
742
738
712
740
714
716
718
720
722
724
726
728
700

WRITE to LBA 0 AGAIN

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Detect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	10	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20																						
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

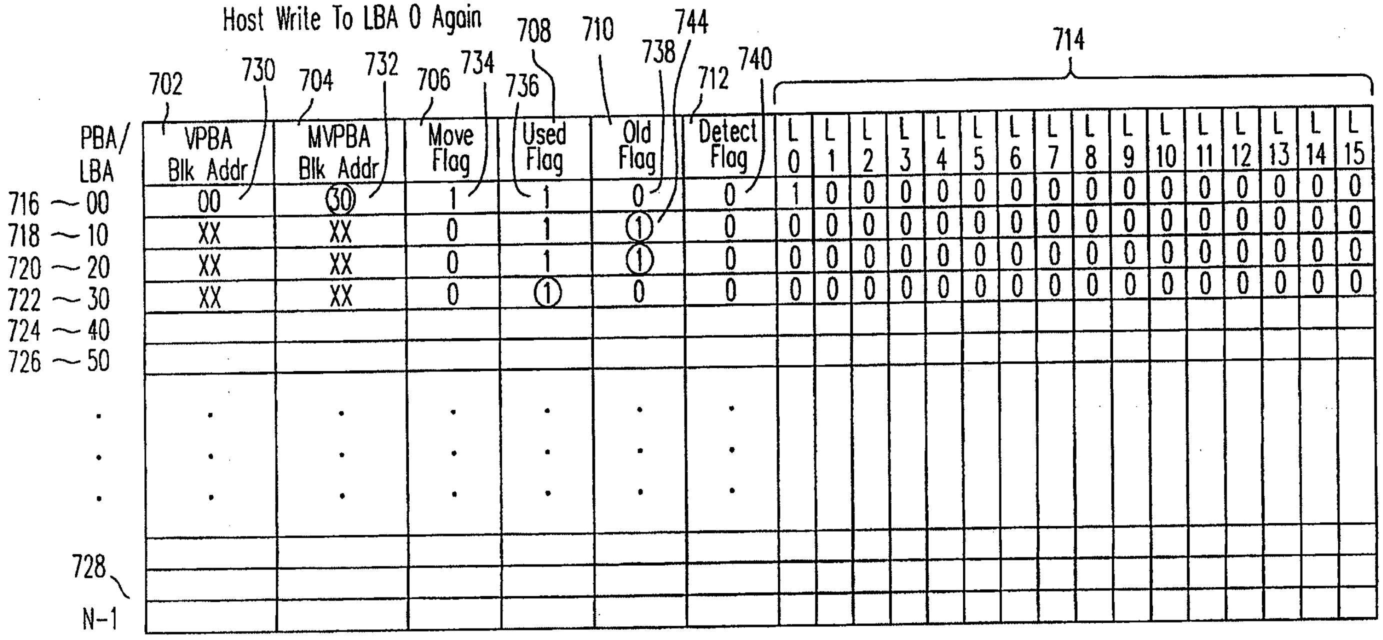
FIG. 12

Host Write To LBA 0 Again


702
730
732
734
742
746
744
714
716
718
720
722
724
726
728
700

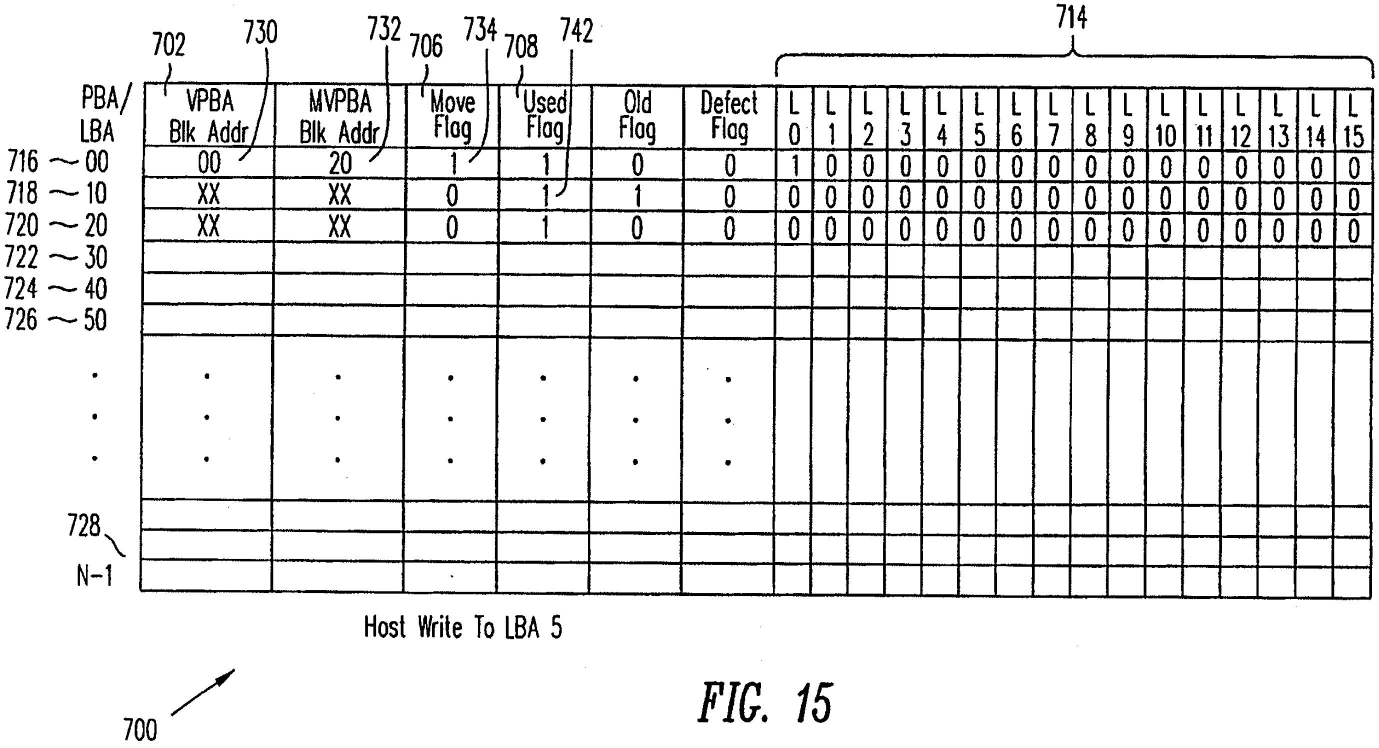
PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	20	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

FIG. 13


Host Write To LBA 0 Again
702
730
704
732
706
734
708
736
710
744
738
712
740
714
716
718
720
722
724
726
728
700

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Detect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	㉚	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	①	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	XX	XX	0	1	①	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	XX	XX	0	①	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

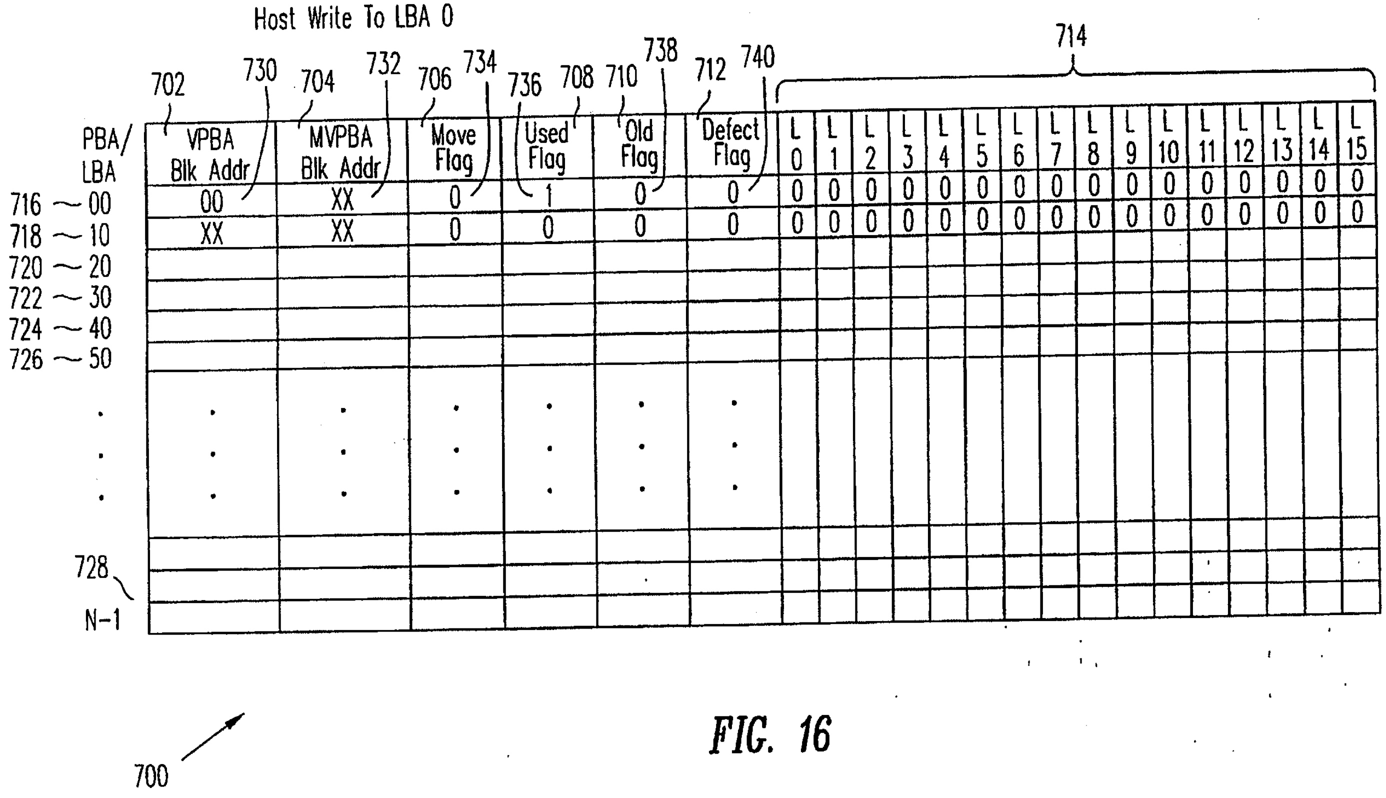
FIG. 14


702
730
732
706
734
708
742
714
716
718
720
722
724
726
728
700

PBA/LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	20	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

Host Write To LBA 5

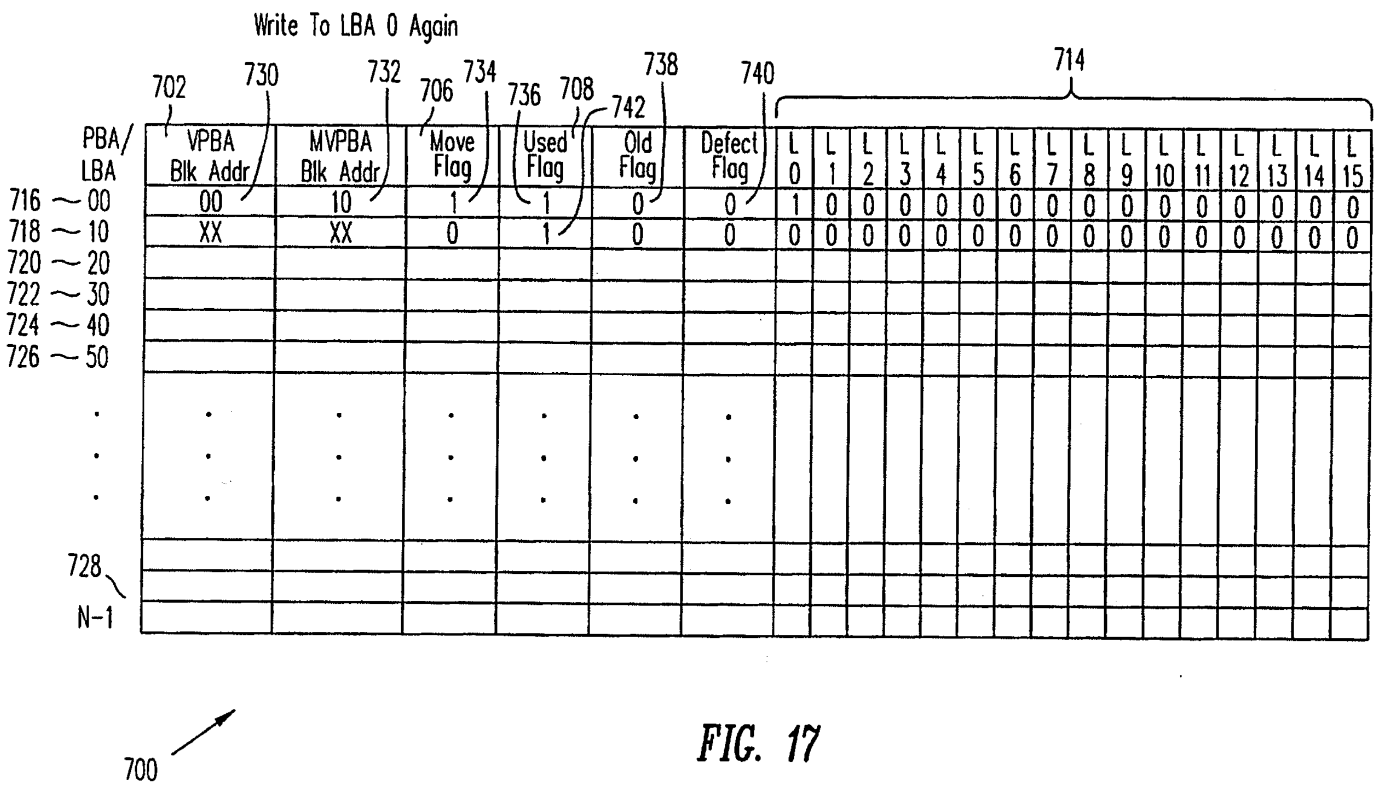
FIG. 15


702
730
704
732
706
734
736
708
710
738
712
740
714
716
718
720
722
724
726
728
700

Host Write To LBA 0

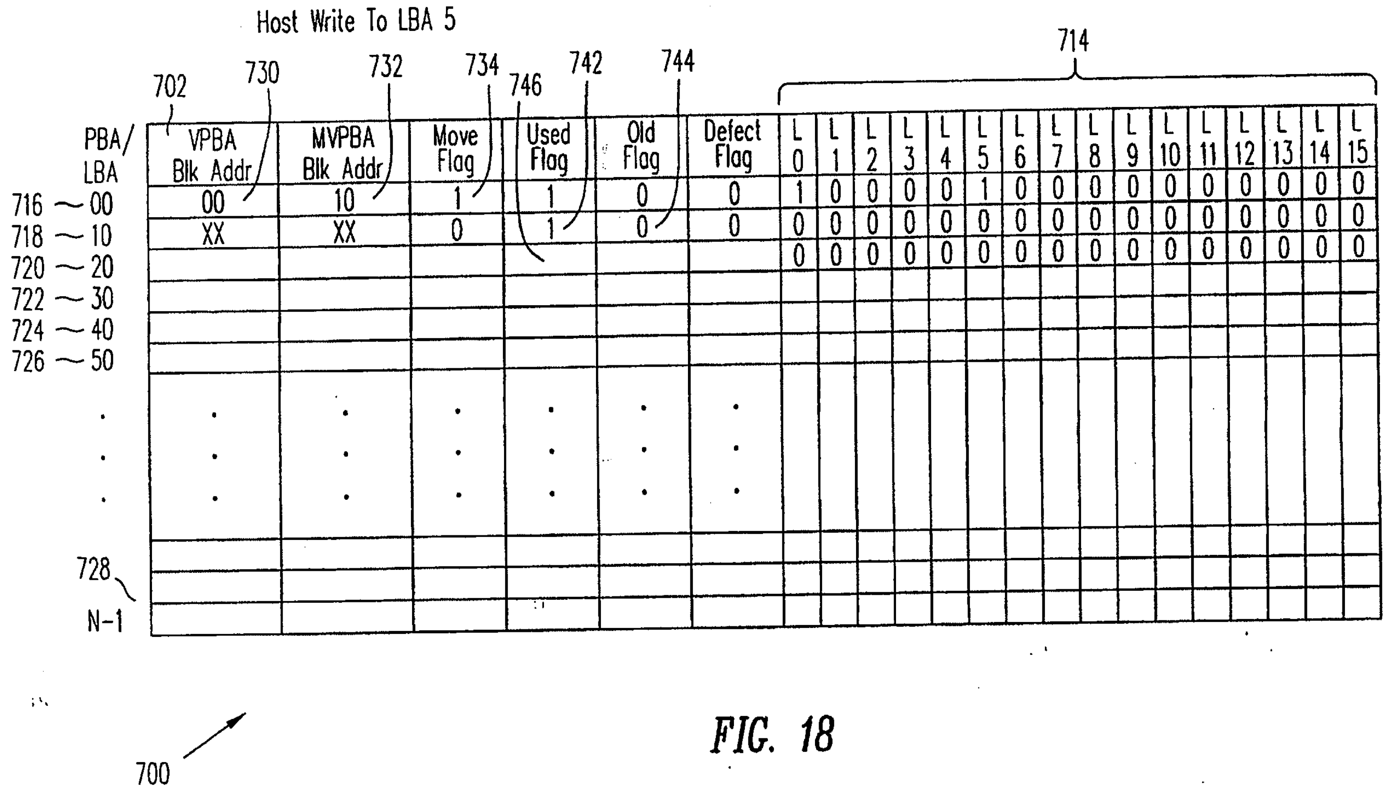
PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20																						
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

FIG. 16


Write To LBA 0 Again
702
730
732
706
734
736
708
742
738
740
714
716
718
720
722
724
726
728
700

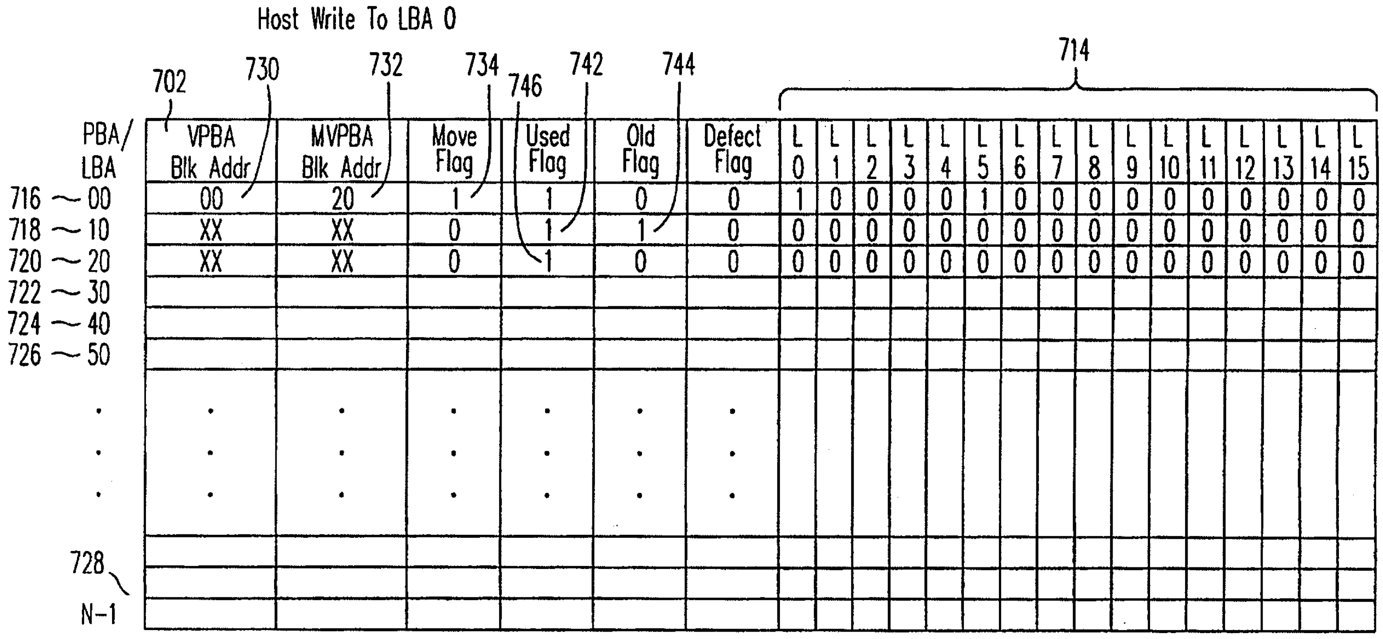
PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	10	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20																						
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

FIG. 17


Host Write To LBA 5
702
730
732
734
746
742
744
714
716
718
720
722
724
726
728
700

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	10	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

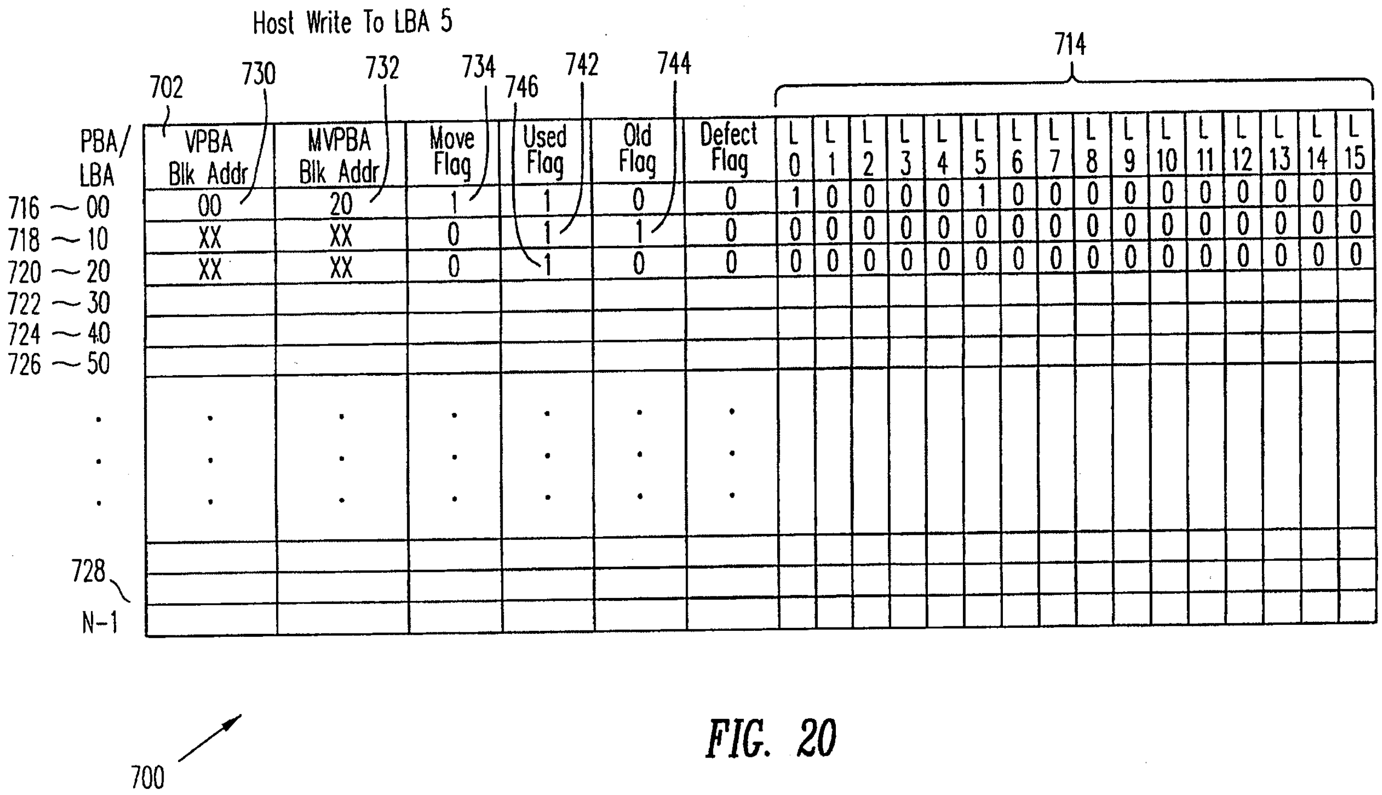
FIG. 18


Host Write To LBA 0
702
730
732
734
746
742
744
714
716
718
720
722
724
726
728
700

Host Write To LBA 0

PBA/LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
716 ~ 00	00	20	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
718 ~ 10	XX	XX	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
720 ~ 20	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
722 ~ 30																						
724 ~ 40																						
726 ~ 50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
728																						
N-1																						

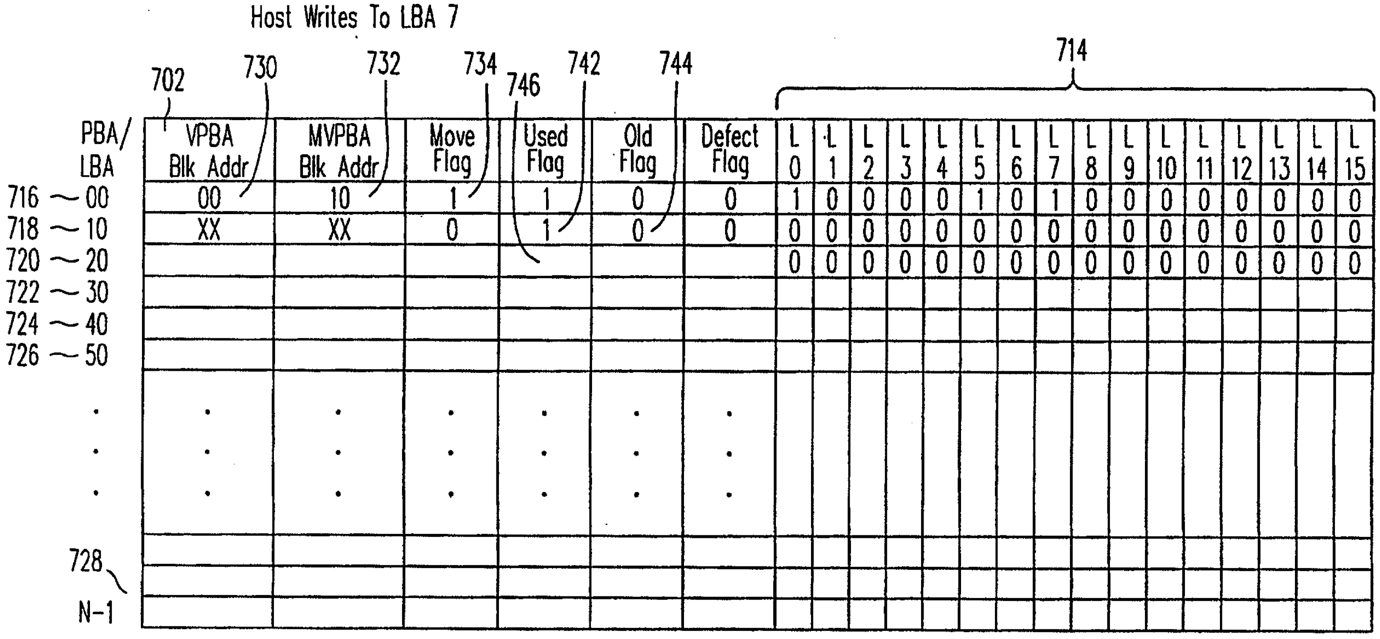
FIG. 19


Host Write To LBA 5
702
730
732
734
746
742
744
714
716
718
720
722
724
726
728
700

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	20	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	XX	XX	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

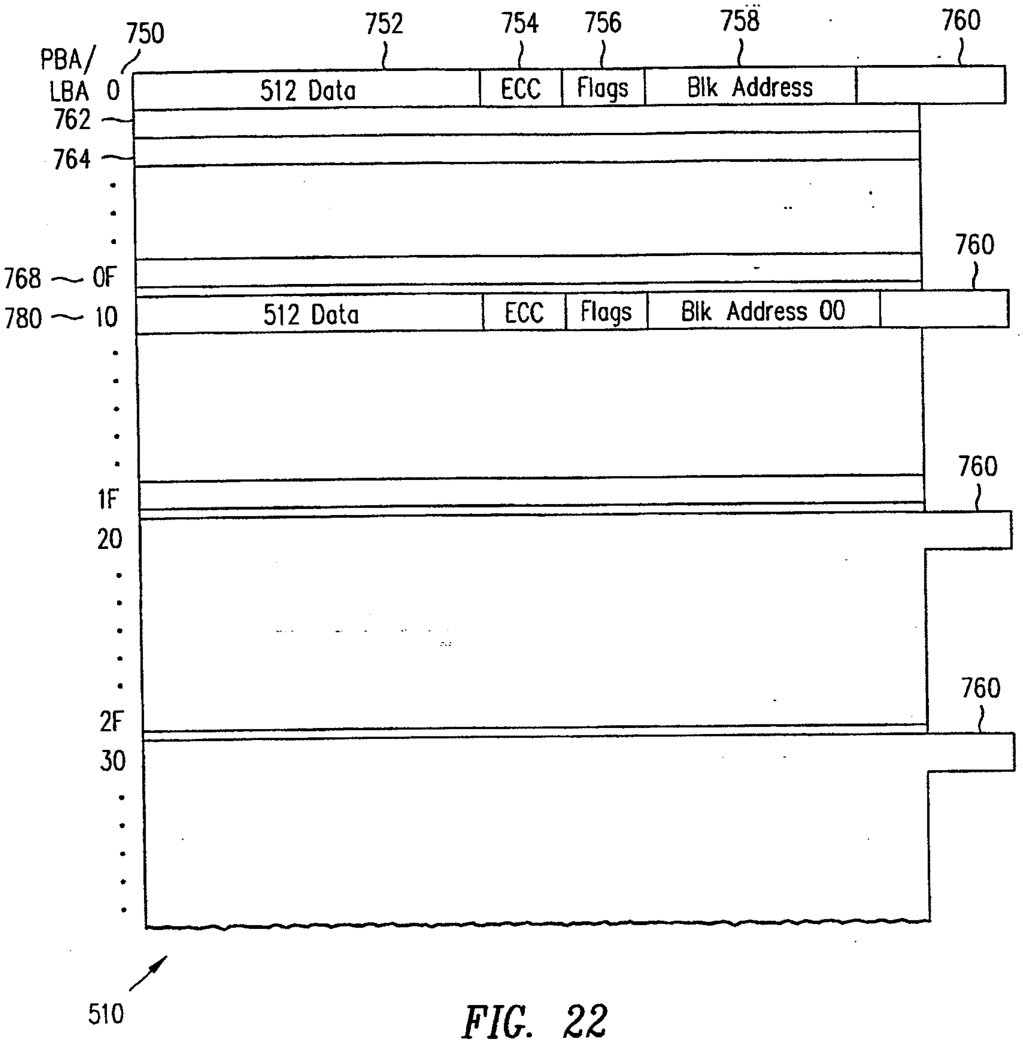
FIG. 20

Host Writes To LBA 7


702
730
732
734
746
742
744
714
716
718
720
722
724
726
728
700

PBA/ LBA	VPBA Blk Addr	MVPBA Blk Addr	Move Flag	Used Flag	Old Flag	Defect Flag	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	L 8	L 9	L 10	L 11	L 12	L 13	L 14	L 15
00	00	10	1	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
10	XX	XX	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30																						
40																						
50																						
⋮	⋮	⋮	⋮	⋮	⋮	⋮																
N-1																						

FIG. 21



750
752
754
756
758
760
PBA/
LBA 0
512 Data
ECC
Flags
Blk Address
762
764
768
0F
760
780
10
512 Data
ECC
Flags
Blk Address 00
760
1F
20
760
2F
30
510

FIG. 22

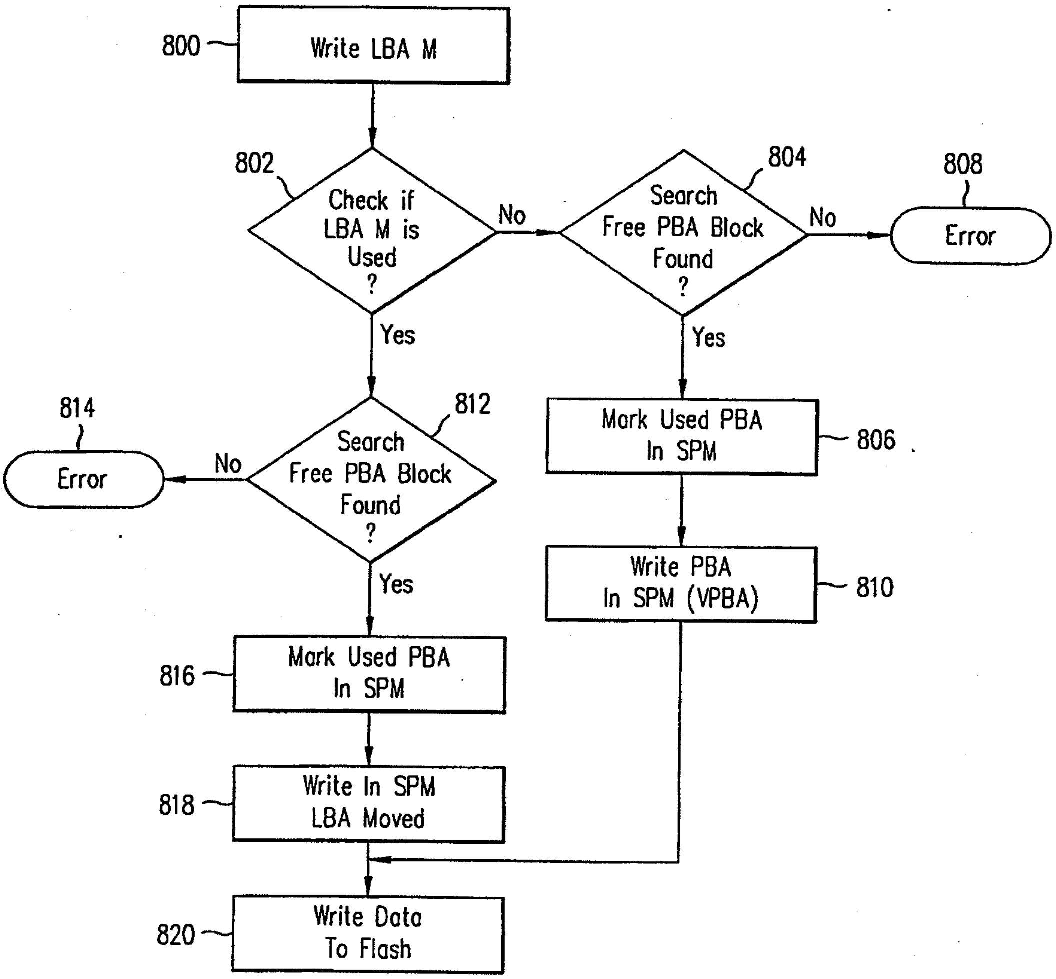

800
Write LBA M
802
Check if LBA M is Used ?
No
804
Search Free PBA Block Found ?
No
808
Error
Yes
Yes
814
Error
No
812
Search Free PBA Block Found ?
Mark Used PBA In SPM
806
Write PBA In SPM (VPBA)
810
Yes
816
Mark Used PBA In SPM
818
Write In SPM LBA Moved
820
Write Data To Flash

FIG. 23

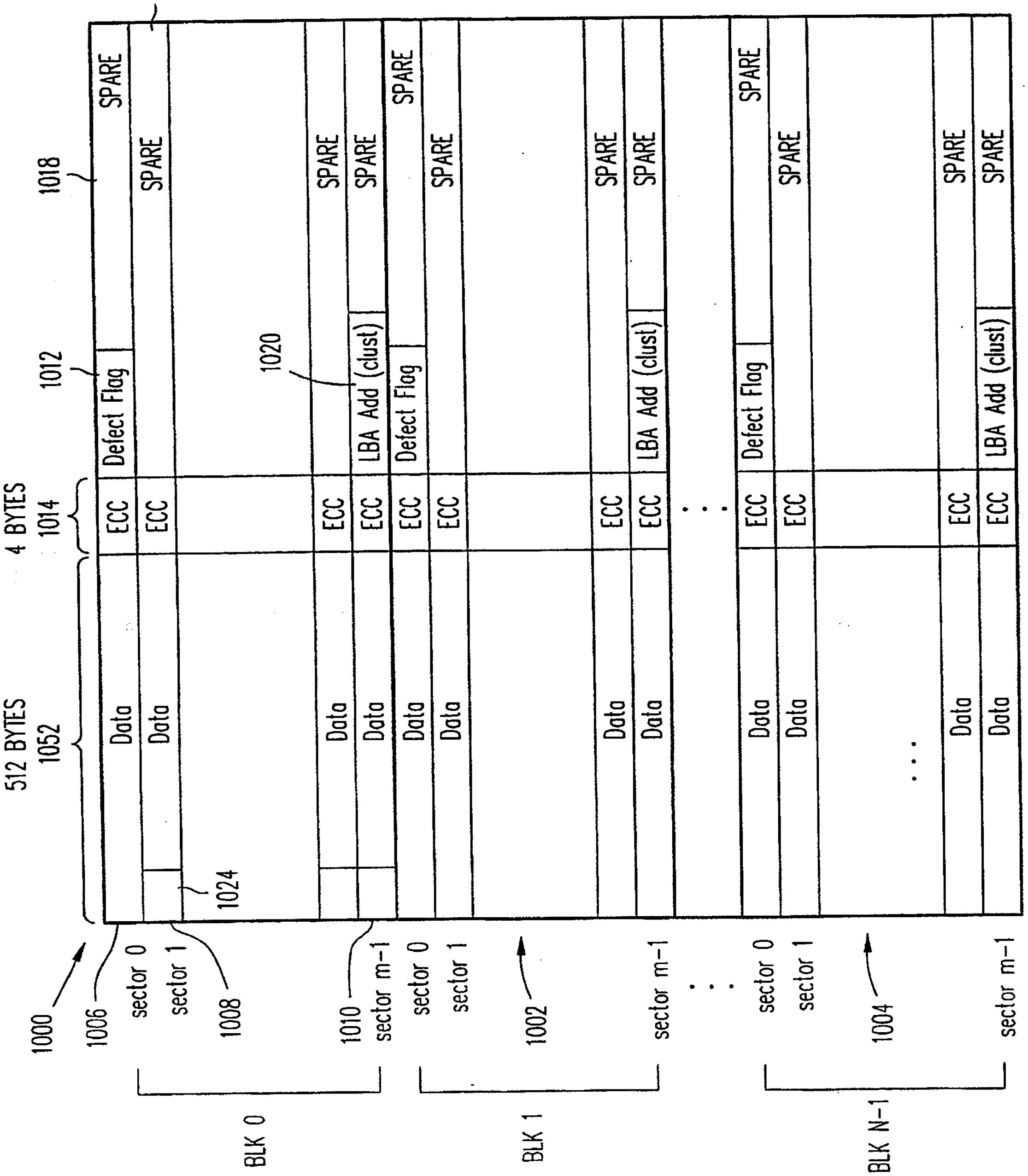

512 BYTES
1052
4 BYTES
1014
1018
1012
1022
SPARE
Defect Flag
ECC
Data
LBA Add (clust)
1020
1024
1000
1006
sector 0
sector 1
1008
1010
sector m−1
1002
1004
BLK 0
BLK 1
BLK N−1

FIG. 24

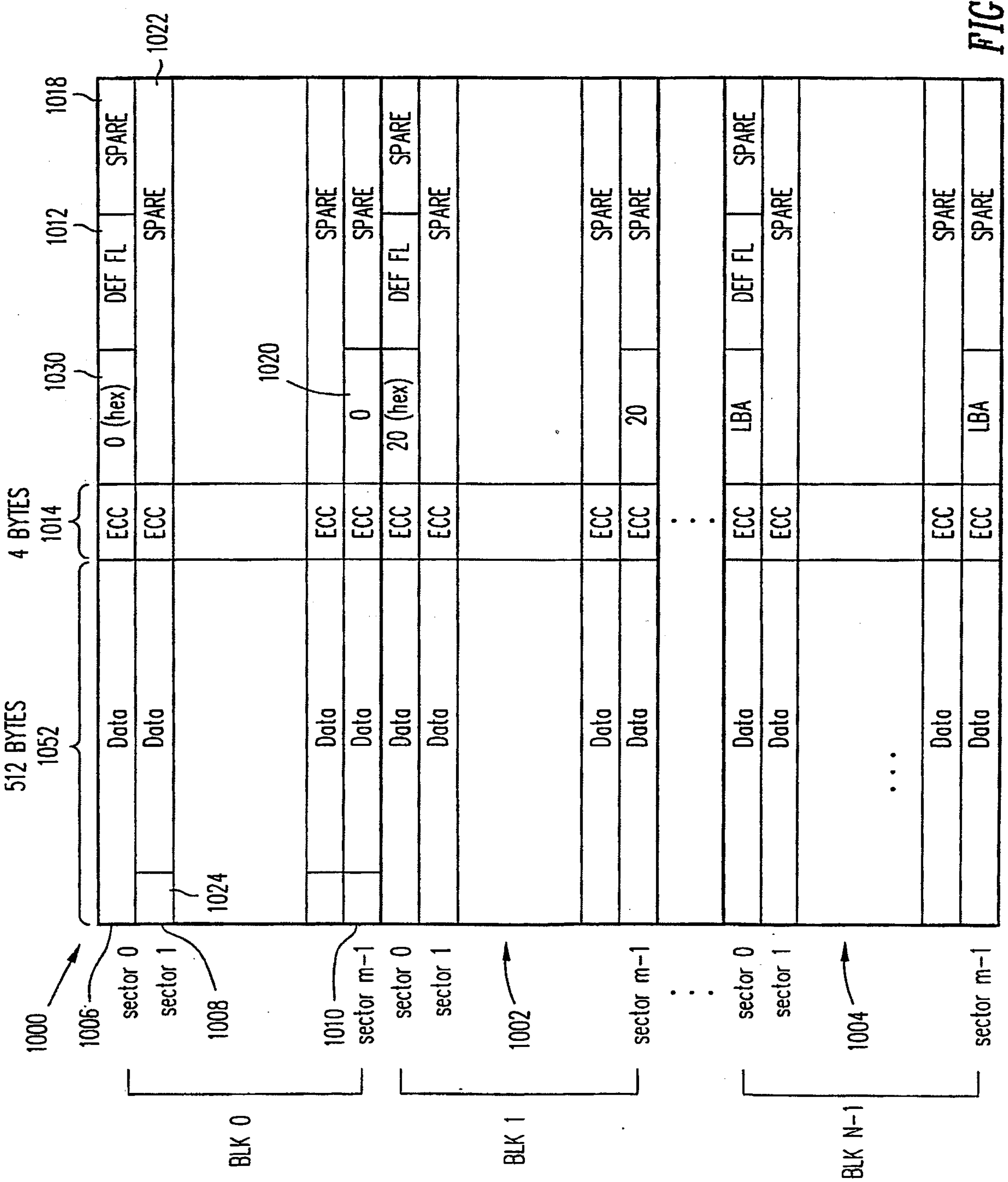

1000
1006
sector 0
sector 1
1008
1010
sector m-1
sector 0
sector 1
1002
sector m-1
sector 0
sector 1
1004
sector m-1
BLK 0
BLK 1
BLK N-1
512 BYTES
1052
4 BYTES
1014
1030
1012
1018
1022
1020
1024
Data
ECC
0 (hex)
DEF FL
SPARE
0
20 (hex)
20
LBA
. . .

FIG. 25

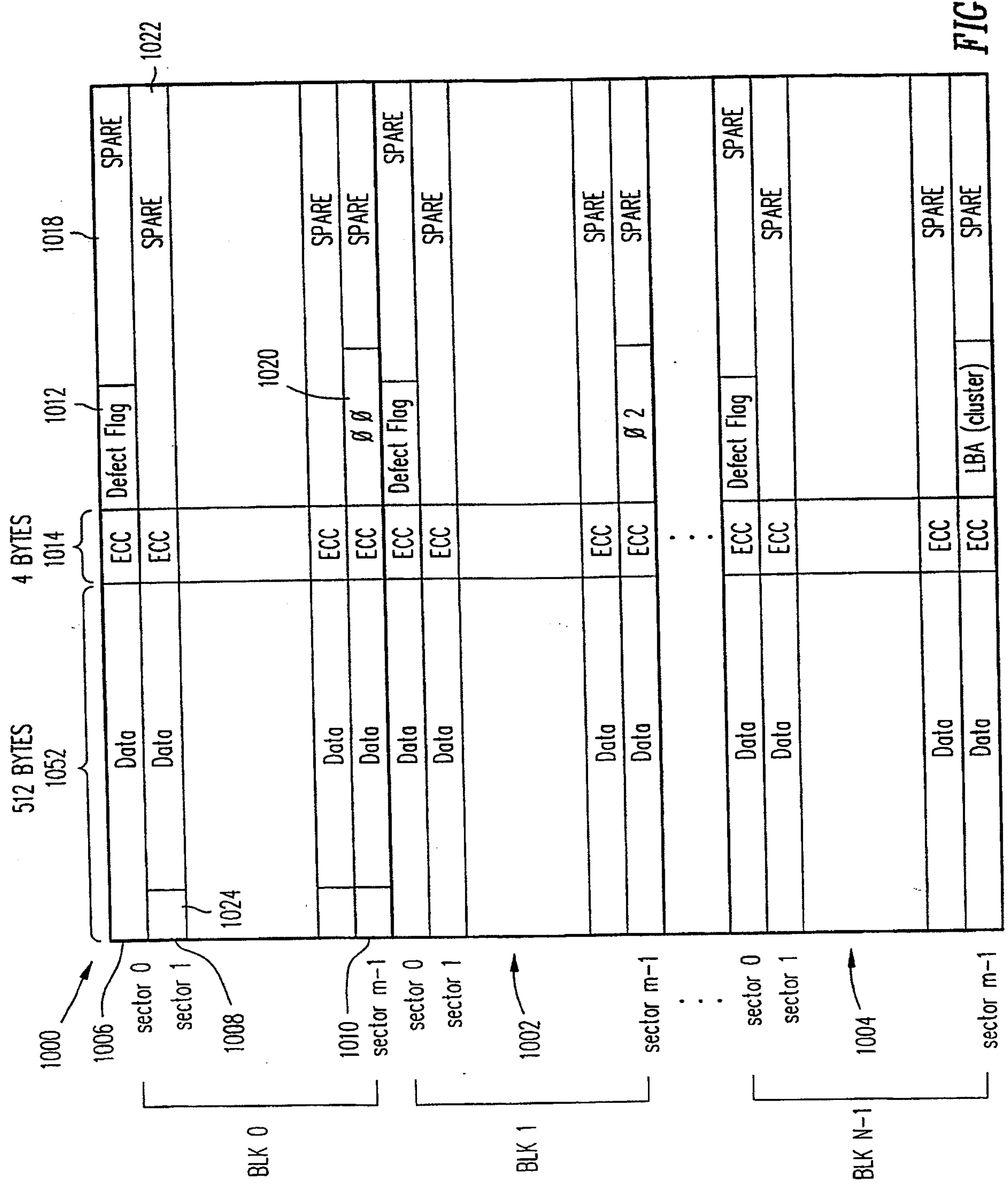
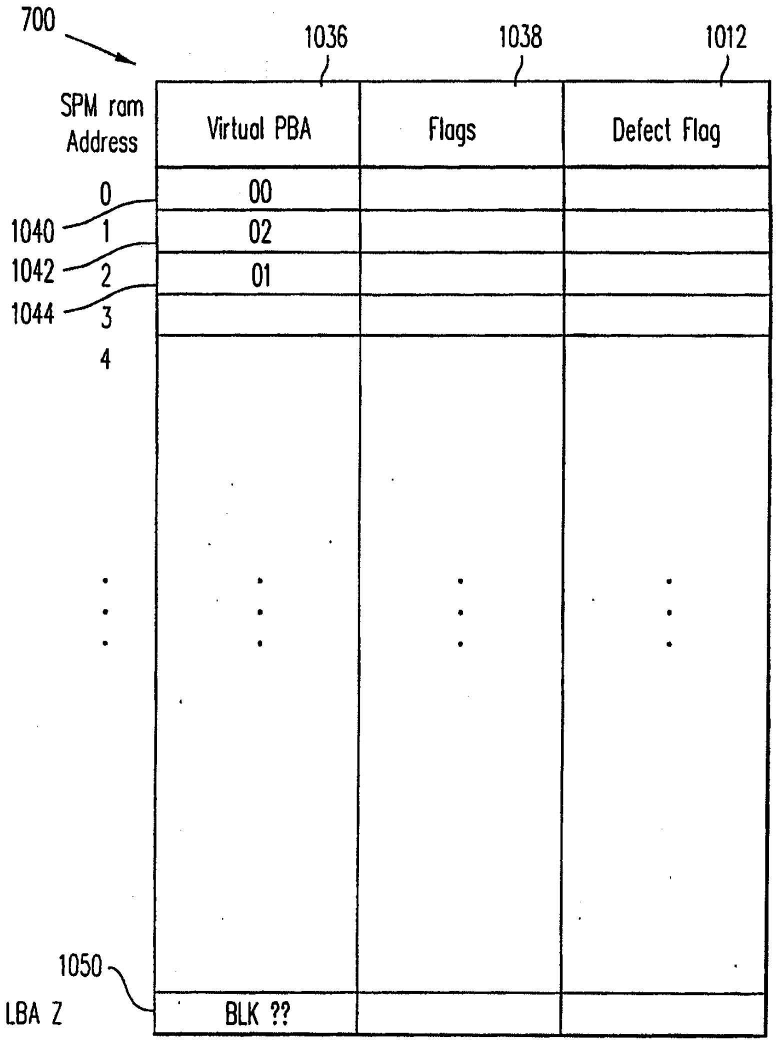

512 BYTES
1052
4 BYTES
1014
1012
1018
1022
1020
1024
SPARE
Defect Flag
ECC
Data
Ø Ø
Ø 2
LBA (cluster)
1000
1006
sector 0
sector 1
1008
1010
sector m-1
1002
1004
BLK 0
BLK 1
BLK N-1

FIG. 26a

PBA(BLK)	LBA
0	LBA= 0...1.5
1	LBA= 16...31
2	LBA = 32...47
3	LBA = 47...63
4	


700
1036
1038
1012
1040
1042
1044
1050
LBA Z

SPM ram Address	Virtual PBA	Flags	Defect Flag
0	00		
1	02		
2	01		
3			
4			
⋮	⋮	⋮	⋮
	BLK ??		

FIG. 26b

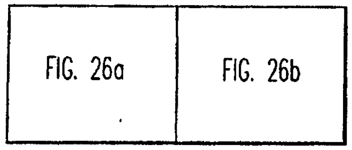


FIG. 26a
FIG. 26b

Key to
FIG. 26

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/06335

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 12/06
US CL :711/103, 156, 165, 203, 206
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)

U.S. : 711/103, 156, 165, 203, 206

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 practicable, search terms used)

APS, JPO ABSTRACTS, EPO ABSTRACTS, IEEE PERIODICALS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,479,638 (ASSAR et al) 26 December 1995, figure 1.	1-17
A	US 5,586,285 (HASBUN et al) 17 December 1996, figure 4.	1-17

☐ 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 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
30 JUNE 1998

Date of mailing of the international search report
01 SEP 1998

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
REGINALD G. BRAGDON
Telephone No. (703) 305-3823

Form PCT/ISA/210 (second sheet)(July 1992)★

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/06335

Box I Observations where certain claims were found unsearchable (Continuation of item 1 of first sheet)

This international 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.: 7 and 11
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box II Observations where unity of invention is lacking (Continuation of item 2 of first sheet)

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

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

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

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers 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.:

Remark on Protest ☐ The additional search fees were accompanied by the applicant's protest.
☐ No protest accompanied the payment of additional search fees.

Form PCT/ISA/210 (continuation of first sheet(1))(July 1992)★

[redacted] [redacted] [redacted]

D11

⑲ Europäisches Patentamt
European Patent Office
Office européen des brevets

⑪ Publication number: **0 250 876 A2**

⑫ **EUROPEAN PATENT APPLICATION**

㉑ Application number: 87107745.9

㉒ Date of filing: 27.05.87

(51) Int. Cl.⁴: G06F 12/12 , G06F 12/08

The title of the invention has been amended (Guidelines for Examination in the EPO, A-III, 7.3).

㉚ Priority: 30.05.86 US 869146

㊸ Date of publication of application:
07.01.88 Bulletin 88/01

(84) Designated Contracting States:
AT BE CH DE ES FR GB GR IT LI NL SE

(71) Applicant: **Honeywell Bull Inc.**
3800 W. 80th Street
Minneapolis Minnesota 55431(US)

(72) Inventor: **Morganti, Victor M.**
101 Lexington Road
Lincoln Massachusetts 01773(US)
Inventor: **Geyer, James B.**
5 Craft Road
Natick Massachusetts 01760(US)

(74) Representative: **Frohwitter, Bernhard, Dipl.-Ing. et al**
Bardehle-Pagenberg-Dost-Altenburg & Partner Patent- und Rechtsanwälte
Galileiplatz 1
8000 München 80(DE)

(54) Apparatus and method for page replacement in a data-processing system having a virtual memory.

(57) In a multiprocessor, multiprogrammed data processing system employing virtual addressing, apparatus and method are provided for selecting a page frame in main memory to be replaced by a new page of logic signal groups required by a processor. Rather than utilize a algorithm implemented in a series of logical decisions determined by a software procedure, the present invention provides for a single instruction that uses the status signals included with a page descriptor to address an entry in a table of resulting status signals. The relationship between the status signals and the table entries implements the algorithm. The table with entries of resulting status signals is associated with the instruction and is stored in the processor when the instruction is prepared for execution by the processor. The resulting status signals are stored with the page descriptor. The resulting status signals are analysed by a software procedure, the software procedure implementing the page replacement and executing other activity indicated by the resulting status signals.

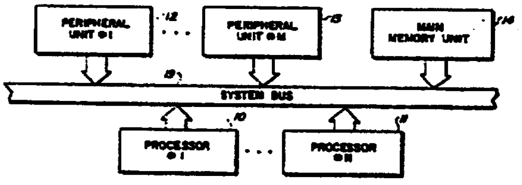


Fig. 1

EP 0 250 876 A2

Xerox Copy Centre

[redacted]

[redacted] [redacted] [redacted]

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to data processing systems and, more particularly, to data processing systems that have a plurality of central processing units using the techniques of virtual addressing to interact with a main memory unit.

2. Discussion of the Related Art

In the modern data processing system, a hierarchical memory system is typically used. The bulk of the data capable of being accessed by a data processing system is stored on mass storage media such as magnetic tape, magnetic disks or other media capable of storing large amounts of digital information. These media are typically too slow to permit the direct addressing of the stored data by a central processing unit. A memory unit, generally referred to as the main memory unit, is implemented with a faster technology and stores portions of the data required by the data processing system. (Although not directly related to the present invention, a memory of yet faster technology utilized for storing data of immediate importance to the central processing unit and physically located within the central processing is generally included in the data processing system. The memory apparatus in the processor is preferred to as the buffer or cache memory.)

Each processor, by means of the operating system, has access to all of the available data associated with the data processing system and can utilize this data as required. As a practical matter, because of the size of data available to the processor, the retrieval of the required data from the mass storage devices would have a detrimental impact on the performance of the process execution. The data having an immediate requirement by the data processing system is stored in the main memory unit. Because storing all of the data in the main memory is impractical, the data is divided into data blocks, called pages, that are entered into the main memory unit as an entity.

In the virtual memory techniques, at the start of operation of the data processing unit, the operating system allocates the space available in the main memory unit. Main memory space will be allocated to programs, tables, and portions of the operating system required for the operation of the data processing system. This area in memory does not have the contents replaced such as is typical of the remainder of the data stored in main memory. The remainder of the memory unit is divided in equal blocks where groups of related data signals that are important to the data processing system can be stored. These data signal blocks are referred to as page frames. Associated with each page frame is a group of data signals in a table of related signals referred to as the page frame descriptor. The operating system can reference the page frame descriptor and the page frame descriptor identifies or points to the location of the associated page descriptor. The page descriptor includes information relating the virtual or symbolic address manipulated by the processors of a data processing system to the actual or physical address where the original information is stored. Also associated with each page descriptor are signals relating to status information. The page frame descriptor also includes data signals in a location referred to as a linker indicative of the order of the page frames according to a preselected algorithm defining how order of the page frames is to be defined.

After initialization of the data processing system, a processor will require a group of data signals. A software procedure will provide the information relating the physical address to the symbolic address for the required group of data signals. The information relating these quantities is stored in the page descriptor. The appropriate information is entered in the linker portion of the page frame descriptor indicating that this is the first of the sequence of page frames. As additional page frames are added, the linker information will identify the order of the page frame in the sequence. After the page frames are all occupied with data, a processor will continue to require access to new data, requiring that a page frame already containing data will have new data replace the old data. It is implementation of this page frame data replacement that the present invention relates.

From the linker information, the next possible page frame in the sequence of page frames can be determined. Based on the linker information, the page descriptor associated with that page frame can be identified and the page descriptor can be entered in the processor. The processor, under software control in the prior art, examines the values of the status signals in the page descriptor and, based on the values of the signals, a decision is made as to the whether the data in that page frame can be replaced. For example, a status signal that is frequently used relates to the experience that the optimum strategy for the replacement of data is to replace the least most recently used data. To implement the least most recently used strategy requires an unacceptable amount of pro-

[redacted] [redacted]

cessing overhead. A typical strategy is to remove the "data used" signal according to a predetermined procedure, and to set the "data used" in the page descriptor whenever the data associated with a page frame is used. In this manner, the software procedures can determine that the data has been used within a preselected period of time.

In addition, still other status signals can be associated with the page frame. Because of the multiplicity of status signals, a software procedure providing a decision with respect to the replacement of the associated group of data can be complex and require an unacceptable amount of processing activity.

When the program controlling the operation of the data processing system requires data signal groups not currently stored in the main memory, the replacement algorithm is invoked. In the prior art, the replacement algorithm was executed by a software process, requiring an analysis of the several status signals. During the determination of the page frame data to be replaced, access to the descriptor was prevented to prevent the use of data that could be in the process of change. One technique to prevent access during this period of possible data change was to provide the memory of portions of the memory with a memory "lock", the memory lock preventing access to the main memory or the selected portions thereof. This technique was effective in insuring that proper data was used by the processors, but, because of the relative slow execution of the software replacement algorithm, the performance of the entire data processing system could be severely impacted. (As will be clear to those skilled in the art, for practical reasons, the memory lock typically involves a plurality of memory locations. The performance will therefore be impacted even if the particular location being analyzed by the replacement algorithm is not accessed.

In order to eliminate the reservation of a main memory portion during execution of the replacement algorithm, techniques have been used that permit the execution of the replacement without reservation of the main memory portion. According to this technique, a determination is made after a replacement page frame selection has been made for a particular location, if a change has occurred in the status signals of the page frame header during the execution of the replacement algorithm. This technique has required additional complexity in the data processing system.

A need has therefore been felt for a technique that permits the determination of a main memory location suitable for having the present data stored therein replaced by new data required by the data processing system.

OBJECTS OF THE INVENTION

It is an object of the present invention to provide an improved data processing system.

It is another object of the present invention to provide for a replacement of a group of data in the main memory of a data processing system according to preselected criteria based on status signals associated with the group of data.

It is yet another object of the present invention to provide for an analysis or a group of status signals without using a software procedure for the analysis.

SUMMARY OF THE INVENTION

The aforementioned and other objects are accomplished, according to the present invention, by providing the data processing system with an instruction that generates new status signals based on the original status signals while minimizing the impact on the availability of the main memory unit. During a page frame replacement procedure, the page descriptor associated with the next page frame to be examined for suitability for replacement is transferred to the processor. Associated with the instruction executing the replacement determination algorithm is a table of resulting status signal values. The status signals of the page descriptor are used to select an appropriate entry in the instruction table. The selected resulting status signals indicate whether the associated page frame data should be replaced, and the selected status signals can be used to indicate other activity required by the processor. The selected status signals are stored in the descriptor and stored in main memory. These and other features of the present invention will be understood upon reading of the following description along with the figures.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of a data processing system capable of using the present invention.

Figure 2 is a diagram illustrating the the division of the main memory unit into a plurality memory location groups.

Figure 3a is a diagram illustrating the relationship between the page frame descriptors table and the page table, Figure 3b illustrates the allocation of the bit positions in the page descriptor and Figure 3c illustrates the identification of the status signals in the page descriptor.

Figure 4 is a diagram of the bit positions of the replacement instruction.

Figure 5a illustrates the entries in the translation table, while Figure 5b provides examples of the use of the translation table.

Figure 6 is a block diagram of the apparatus in the execution unit that executes the replacement instruction.

Figure 7is a flow diagram illustrating how the replacement instruction can be implemented by a software procedure.

Figure 8 illustrates the separation of the main memory unit into sectors for purposes of limiting access.

DESCRIPTION OF THE PREFERRED EMBODIMENT

1.Detailed Description of the Figures

Referring now to Fig. 1, a data processing system capable of advantageously utilizing the present invention is shown. The data processing system includes at least one processor (illustrated by processor 10 and processor 11, a main memory unit 15, at least one peripheral interface units (illustrated by peripheral interface unit 12 and peripheral interface unit 13) and a system bus 19. In the data processing system illustrated in Figure 1, the main memory stores the data signal groups to be manipulated by the processors. In the main memory unit 15 during initialization of the system, the storage space of the main memory is divided into a multiplicity of regions, each region for storing a preestablished quantity of logic signals. In addition, the main memory unit has certain regions reserved for files, the files being accessible to the processors. The use of a common main memory unit for a plurality of processors is described as a "tightly coupled" system.

Referring now to Figure 2, a diagrammatic representation of the organization of the main memory is shown. This organization is typically established during initialization of the data processing system. Region 21 is generally reserved for operating system, programs and tables required by the data processing system. The remainder of the main memory locations are divided into groups of main memory locations referred to as page frames. Fig. 2 illustrates page frame #1 22 through page frame #Q. It will be clear that Fig. 2 is an idealized representation of the organization of the main memory location. Typically, region 21 would be fragmented and dispersed throughout the main memory unit. The numbering of the page frames is arbitrary and designates, at best, only a physical location in the main memory unit.

Referring next to Fig. 3a, 3b and 3c, an organization of tables of particular importance to the present invention is shown. In Fig. 2, a table of page frame descriptors 31 is established during initialization of the data processing system. Each page frame descriptor is associated with one of the page frames 22 through 23 of Fig. 2. The purpose of the page frame descriptor, as it relates to the present invention, is to identify the physical address of the data currently stored in the associated page frame. This task is accomplished by providing a pointer to the page table 34and more particularly to the page descriptor 35 associated with the data currently stored in the page frame. The page frame descriptor also includes a group of locations that defines the order of the page frame relative to the other page frame. When the data in a page frame is replaced, the linker, implementing the ordering of the page frames, insures that all the other page frames in the main memory will be reviewed for replacement of data before the page frame with the just replaced data is again reviewed. Fig. 3b illustrates the structure of the page descriptor 35 shown in Fig. 3a. The page descriptor includes status signals 37 in the bit positions 0-3 and a page frame number identifying the page frame in bit positions 4-23. Fig. 3c provides an identification of the status bits in the preferred embodiment. The first position is the valid bit and indicates when the data page is in a page frame in main memory data in the page frame. The second position is the use bit and is set each time the data in the associated page frame is accessed. Position 3 is the modified bit and indicates that there is a difference between the data in the page frame and the data from which the page frame was extracted. The status bit in the fourth position is currently reserved for future interpretation.

Referring next to Fig. 4, the structure of the instruction executing the replacement algorithm is shown. The first 16 bit positions identify the operation code of the instruction. The second 16 bits and possibly additional 16 bit data signal groups provide a pointer to the page descriptor. In the final four groups of 16 bits is contained the translation table.

Referring to Fig 5a, the translation table entries, according to the preferred embodiment, are shown. The table entries are indicated in hexadecimal notation. The actual translation is illustrated in Fig. 5b showing how the status signals of the page descriptor are modified by the replacement algorithm.

Referring next to Fig. 6, the apparatus implementing the instruction is illustrated. In the execution unit of a processor, the instruction register 61 is loaded with data signals determined by the operation code of of the instruction. The signals

[redacted] [redacted] [redacted]

from the instruction register are applied to the decode and control apparatus 62. The decode and control apparatus 62 applies signals to addressing apparatus 63. Under the control of the decode and control apparatus 62, the status signals from the retrieved page descriptor which are stored in register 64 of the working registers 60 are used as the address signal to address an entry in the translation table associated with the instruction. The entry addressed by the status signals via the addressing apparatus 63 provide the new status signals to be stored in the page descriptor. The signal analysis unit 66 is used to indicate that a replacement page frame has been identified and to begin the procedure to store the desired information page in the page frame.

Referring next to Fig. 7, the flow chart for examining page descriptors to determine if the associated page frame should be replaced. After beginning the process in step 700, for example by executing the instruction illustrated in Fig. 4, the next page descriptor is selected. This selection is determined by the linker memory locations 34 illustrated in Fig. 3a. The linker, in the preferred embodiment, selects a page descriptor that has remained unexamined by the replacement algorithm. Upon selection of the page descriptor to be examined, the status signals forming a portion of the page descriptor are retrieved from main memory in step 702. In step 703 the valid status signal is examined. If the valid signal is a negative value, the page descriptor is not changed in step 704 and the procedure returns to step 701 to select a next page descriptor. When the valid signal is a positive value, the use signal is examined in step 705. When the use signal has a positive value, the use bit position is changed to a negative value and the new status signals are stored in the appropriate page descriptor positions in main memory in step 706. The procedure then returns to step 701 to select a next descriptor for examination. When the use signal has a negative value, then the modified status signal is examined in step 707. When the modified status signal has a positive value, then the modified bit is changed to a negative value while a command is issued to bring the data in the page frame in correspondence with (or purify) the data from which the page frame data was derived in step 708. When the modified status signal is a negative value, then the associated page frame is one suitable for replacement. Step 709 changes the status signals to all negative values and stores these values in appropriate positions in the page descriptor. The page descriptor is invalidated and the data in the page frame is replaced with the requested data page The page descriptor is modified to represent the new data stored in the page frame via step 710. The page descriptor as replaced in the sequence of page frames via the linker in the page frame descriptor and the process of replacing data in a page frame has been completed, i.e. step 711.

Referring next to Fig. 8, in the preferred embodiment, the main memory unit is controlled by two controllers, A and B. Each memory controller controls one half of the memory locations. In each controller, the associated memory locations are divided into a plurality of sectors 83. Access to each sector can be controlled by the associated memory controller. Therefore, when for example a memory location is in the process of being altered, the controller can prevent access to the particular sector without limiting access to the entire region associated with the memory controller. Because of the expense and complexity, the access control to individual memory locations is seldom implemented. For this reason, the limiting of access to a particular memory location results in the limiting of access to a multiplicity of memory locations and any lengthy limitation of access can potentially impact the performance of the data processing system.

2. Operation of the Preferred Embodiment

The present invention resolves the problem of the access to the main memory by processors during a possible change in the location contents by providing a efficient determination of the page replacement algorithm. In this manner, the portion of the main memory that is reserved during the execution of the algorithm is reserved for only a short period of time, thereby minimizing the impact on the performance of the data processing system. In place of a software procedure that can limit access to sections of the main memory for an unacceptable period of time, the determination of the suitability of a page frame for replacement is determined by an instruction in which the status signals associated with the page frame are used as address signals to determine an entry into a table. The table is associated with the instruction itself. The entries in the instruction can be determined in a relatively short period of time. These new or resulting status signals can be immediately stored in the descriptor. Because the new status signals prohibit use of the associated page frame, the memory mechanism limiting access to the main memory sector including the page descriptor can be removed. The relatively short time required for this procedure minimizes impact on the main memory accessibility and consequently on the data processing system performance.

An important feature of the implementation of an algorithm in a software procedure is the flexibility of the procedure. If, for example a different interpretation of a status signal is desired, the consequences of this change in interpretation can be provided by associated changes in the steps of the procedure. It will be clear that this flexibility is retained in the present invention by the location in the instruction to which the group of status signals is directed. By changing resultant signals, the flexibility can be maintained.

The foregoing description is included to illustrate the operation of the preferred embodiment and is not intended to limit the scope of the invention. The scope of the invention is to be limited only by the following claims. From the foregoing discussion, many variations will be apparent to those skilled in the art that would yet be encompassed by the spirit and scope of the invention.

Claims

1. In a data processing system in which each of a plurality of memory locations has a plurality of status signals associated therewith, apparatus for determining new status signals resulting from applying an algorithm to said status signals, comprising:
an instruction for executing said algorithm, said instruction storing said new status signals therein;
a first storage unit for storing said status signals;
a second storage unit for storing said new status signals in a table; and
addressing apparatus responsive to said instruction for using said stored status signals to address an entry in said table configuration, a relation between said stored status signals and said table configuration entry implementing said algorithm.

2. The apparatus for determining new status signals of Claim 1 wherein said status signals are associated with a page frame of data, said algorithm identifying when a page frame of data can be replaced with a new page frame of data.

3. The apparatus for determining new status signals of Claim 2 further including apparatus responsive to selected new status signals for replacing said page frame of data associated with said stored status signals with a new page frame of data.

4. The apparatus for determining new status signals of Claim 2 wherein a first of said status signals related to usage of said associated page frame of data and a second of said status signals relates to modification of said associated page frame of data.

5. In a data processing system, a method of determining when to replace a page frame in a main memory of said data processing system comprising the steps of:
retrieving a first instruction from main memory to perform said replacement determination, said first instruction having associated therewith groups of status signals;
storing said first instruction status signal groups in a table;
retrieving a page descriptor associated with a selected page frame;
using status signals associated with said descriptor to access an entry in said instruction status signal table; and
identifying when said accessed instruction status signal group indicates said associated page frame can be replaced.

6. The method of determining when to replace a page frame of Claim 5 further comprising the step of replacing said status signals with said instruction table entry in said descriptor.

7. The method of determining when to replace a page frame of Claim 5 further comprising the step of retrieving a descriptor that has not been examined for replacement for the longest period of time when the retrieved page descriptor is not suitable for replacement.

8. The method of determining when to replace a page frame of Claim 5 further including the step of associating said status signals with usage, validity and modification of said page frame.

9. In a data processing system having page frames storing data in a main memory unit, apparatus in a processor for determining when a page of data signals can be replaced comprising:
retrieval apparatus associated with said processor for retrieving an instruction, said instruction including a plurality of entries, wherein said plurality of entries is stored in said processor, said retrieval apparatus for retrieving at least a preselected field of a descriptor associated with a one of said data signal pages, wherein said preselected field of said descriptor is stored in said processor;
addressing apparatus responsive to a preselected field of said store selected descriptor field for forming an address field, said addressing apparatus using said address field to address a predetermined one of entries, said predetermined entry being a replacement descriptor field; and
decision apparatus responsive to said replacement field for providing a signal to said processor indicating when a data signal page can be replaced.

10. The page replacement apparatus of Claim 9 further including apparatus for storing said replacement field in said descriptor in main memory.

11. The page replacement apparatus of Claim 9 wherein said retrieving apparatus at least a preselected field of a next descriptor when said decision apparatus indicates that the data page field can not be replaced.

12. The page replacement apparatus of Claim 11 wherein said next descriptor is the descriptor that has not had said at least one field replaced for the longest interval.

13. The page replacement apparatus of Claim 9 wherein said at least one field includes a plurality of status signals, a one of said status signals signifying use of said data signal page.

14. The page replacement apparatus of Claim 10 wherein said portion of main memory storing said descriptor is made unavailable to a remainder of said data processing system during activity of said instruction.

15. In a data processing system, an instruction for causing a processor to determine if a selected page frame can be replaced without intervention of a software program comprising:
a first field;
a second field, said first field causing said processor to retrieve and to store therein at least a first field of a descriptor associated with said page frame, a location of said descriptor being identified by said second field; and
a third field including a plurality entries, said third field being stored in said processor, said first field causing said processor to select a one of said instruction entries, said selected entry determining when said associated page frame is to be replaced.

16. The instruction for determination of a replacement of a page frame of Claim 15 wherein said selected entry is a replacement field for said first descriptor field, said replacement field being stored with said descriptor in said main memory.

17. The instruction for determination of a replacement of a page frame of Claim 15 wherein said first descriptor field includes a multiplicity of logic signals defining a status of said page frame.

18. The instruction for determination of a replacement of a page frame of Claim 17 wherein said status signals are indicative of a validity of data in said page frame, are indicative of use of said page frame and are indicative of modification of said page frame.

19. The instruction for determination of a replacement of a page frame of Claim 15 wherein said instruction entries are stored in a memory unit, said first field causing said said processor to select an address in said memory unit determined by logic signals in said first descriptor field.

[redacted] [redacted] [redacted]

Neu eingereicht / Newly filed
Nouvellement déposé

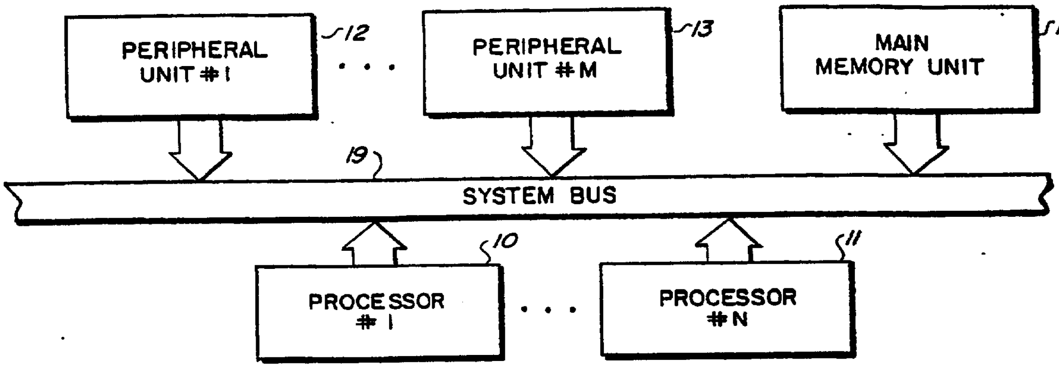

PERIPHERAL UNIT # I
12
PERIPHERAL UNIT # M
13
MAIN MEMORY UNIT
14
19
SYSTEM BUS
10
11
PROCESSOR # I
PROCESSOR # N

FIG. 1

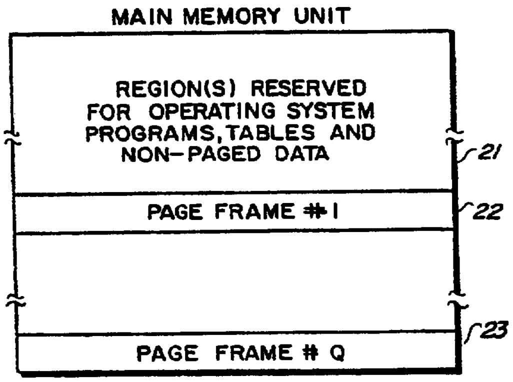

MAIN MEMORY UNIT
REGION(S) RESERVED FOR OPERATING SYSTEM PROGRAMS, TABLES AND NON-PAGED DATA
21
PAGE FRAME # I
22
PAGE FRAME # Q
23

FIG. 2

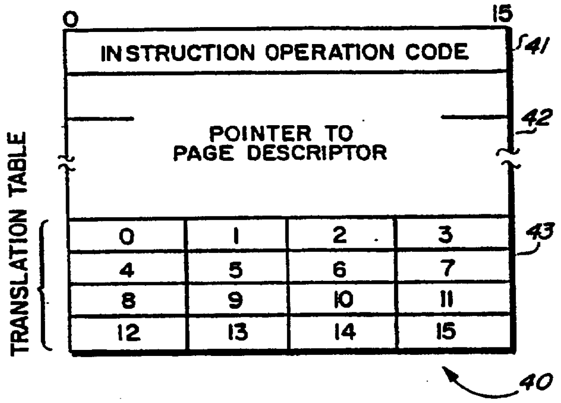

0
15
INSTRUCTION OPERATION CODE
41
POINTER TO PAGE DESCRIPTOR
42
TRANSLATION TABLE
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
43
40

FIG. 4

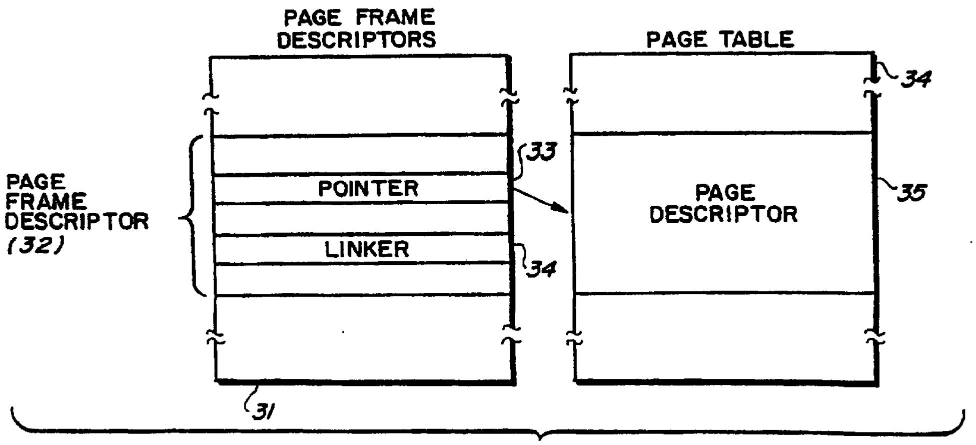

PAGE FRAME DESCRIPTORS
PAGE TABLE
34
33
POINTER
PAGE FRAME DESCRIPTOR (32)
LINKER
34
PAGE DESCRIPTOR
35
31

FIG. 3A

[redacted] [redacted]

Neu eingereicht / Newly filed
Nouvellement déposé

0	1	2	3
4	5	6	7
0	1	8	9
8	9	A	B

TRANSLATION TABLE ENTRIES

FIG. 5A

VUMX	TABLE ENTRY
111X	101X
110X	100X
101X	100X
100X	000X

FIG. 5B

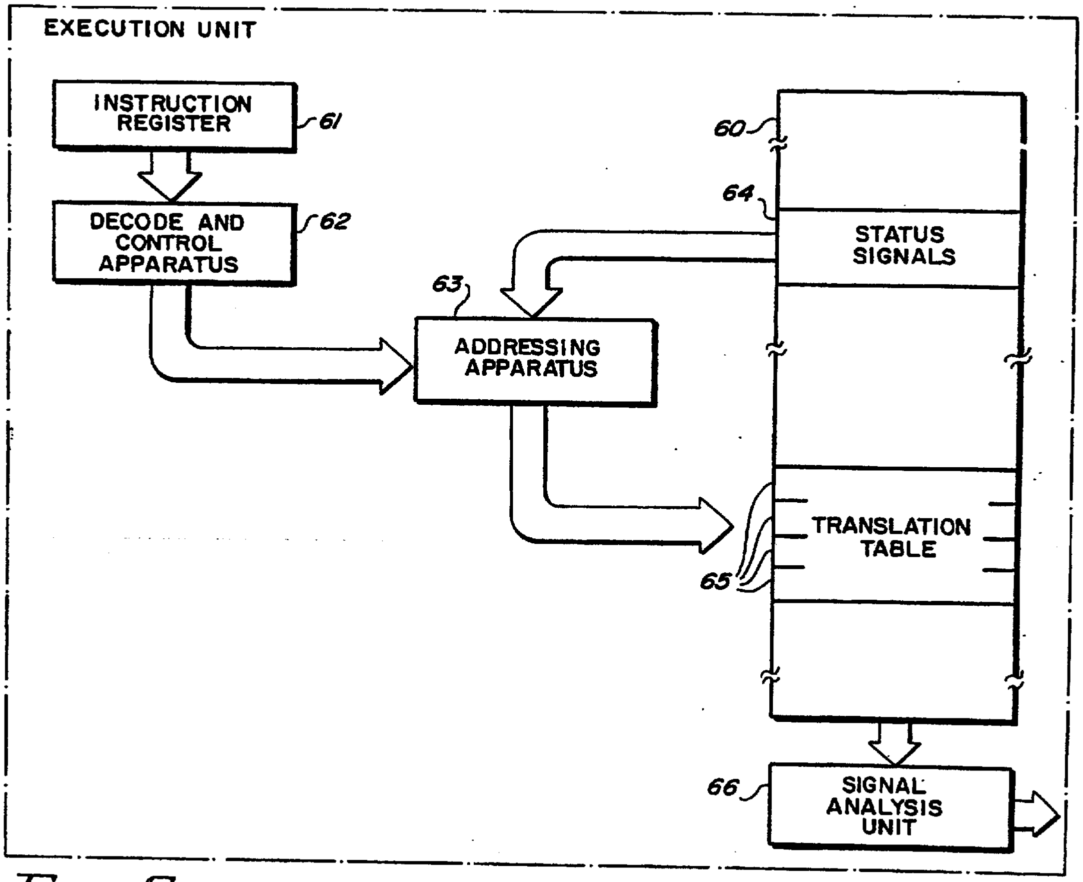
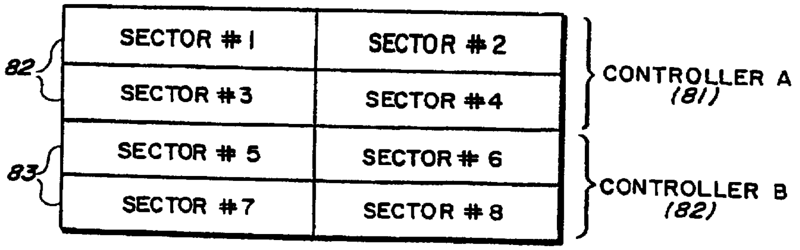

EXECUTION UNIT
INSTRUCTION REGISTER
61
DECODE AND CONTROL APPARATUS
62
63
ADDRESSING APPARATUS
60
64
STATUS SIGNALS
TRANSLATION TABLE
65
66
SIGNAL ANALYSIS UNIT

FIG. 6

FIG. 8


SECTOR #1
SECTOR #2
SECTOR #3
SECTOR #4
SECTOR #5
SECTOR #6
SECTOR #7
SECTOR #8
82
83
CONTROLLER A (81)
CONTROLLER B (82)

Neu eingereicht / Newly filed
Nouvellement déposé

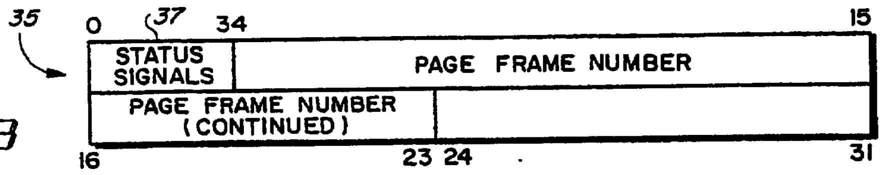

35
0
37
34
15
STATUS SIGNALS
PAGE FRAME NUMBER
PAGE FRAME NUMBER (CONTINUED)
16
23
24
31

FIG. 3B

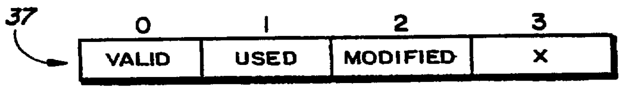

37
0
1
2
3
VALID
USED
MODIFIED
X

FIG. 3C

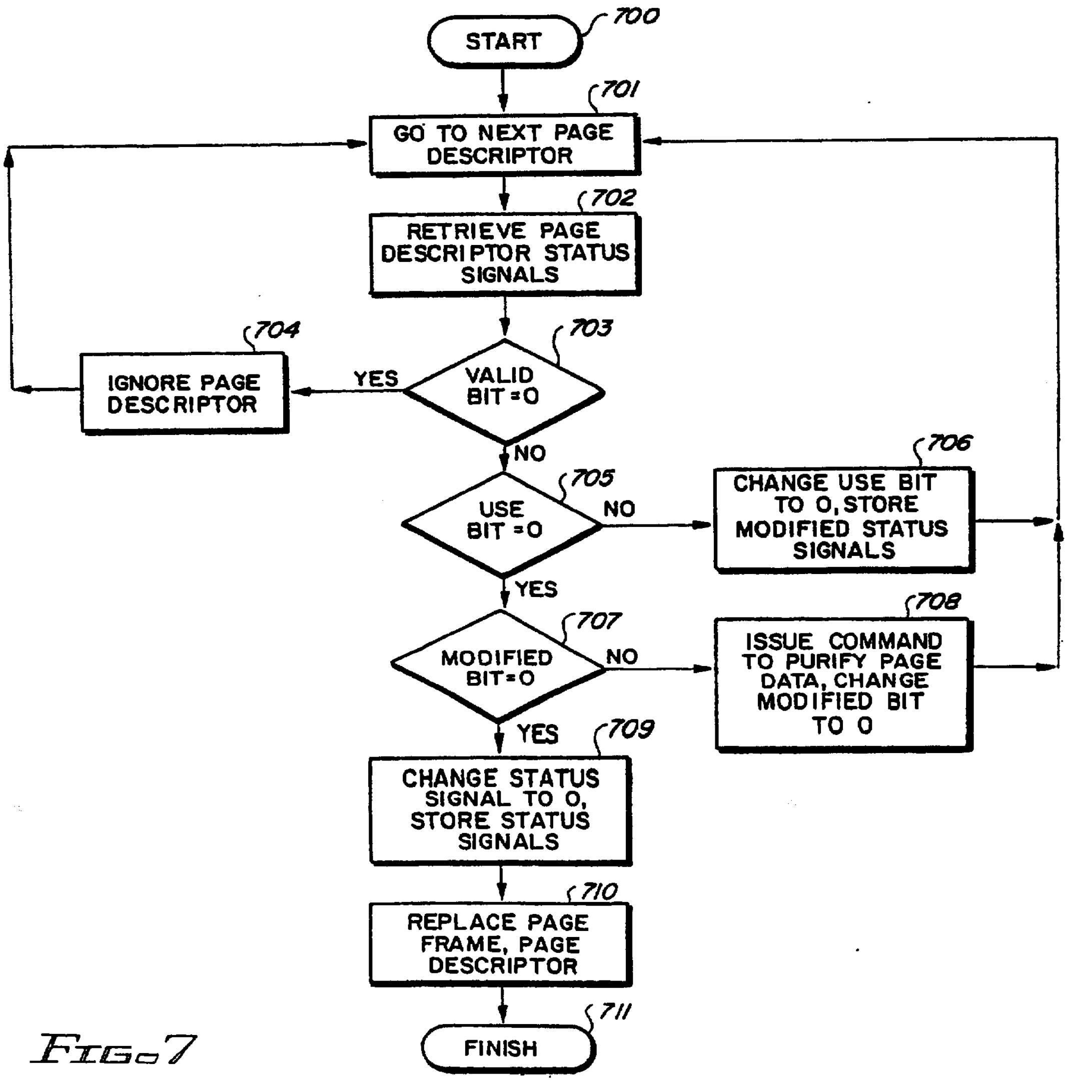

START
700
GO TO NEXT PAGE DESCRIPTOR
701
RETRIEVE PAGE DESCRIPTOR STATUS SIGNALS
702
VALID BIT = 0
703
YES
IGNORE PAGE DESCRIPTOR
704
NO
USE BIT = 0
705
NO
CHANGE USE BIT TO 0, STORE MODIFIED STATUS SIGNALS
706
YES
MODIFIED BIT = 0
707
NO
ISSUE COMMAND TO PURIFY PAGE DATA, CHANGE MODIFIED BIT TO 0
708
YES
CHANGE STATUS SIGNAL TO 0, STORE STATUS SIGNALS
709
REPLACE PAGE FRAME, PAGE DESCRIPTOR
710
FINISH
711

FIG. 7

[redacted] [redacted] [redacted]

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT) 1/13

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 June 2002 (20.06.2002)

PCT

(10) International Publication Number
WO 02/49039 A2

(51) International Patent Classification⁷: G11C 16/00

(21) International Application Number: PCT/US01/47166

(22) International Filing Date: 13 November 2001 (13.11.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/718,802 22 November 2000 (22.11.2000) US

(71) Applicant: SANDISK CORPORATION [US/US]; 140 Caspian Court, Sunnyvale, CA 94089 (US).

(72) Inventors: GONZALEZ, Carlos; 249 Mattson Avenue, Los Gatos, CA 95032 (US). CONLEY, Kevin, M.; 5983 Alvarado Court, San Jose, CA 95120 (US). HARARI, Eliyahou; 104 Auzerais Court, Los Gatos, CA 95030 (US).

(74) Agent: PARSONS, Gerald, P.; Skjerven Morrill MacPherson LLP, Three Embarcadero Center, 28th Floor, San Francisco, CA 94111 (US).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

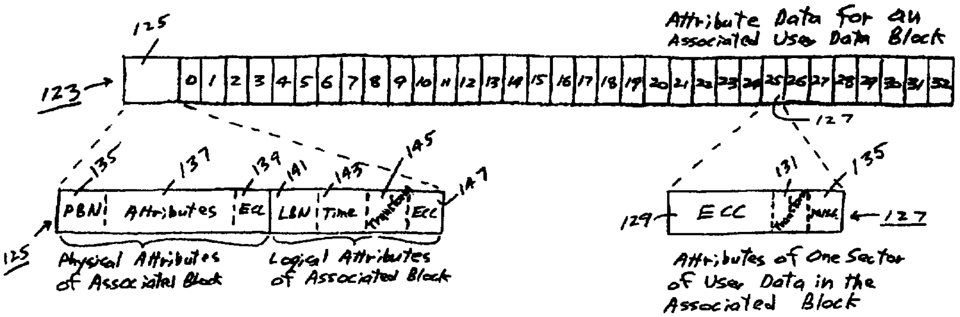
(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 02/49039 A2

(54) **Title:** TECHNIQUES FOR OPERATING NON-VOLATILE MEMORY SYSTEMS WITH DATA SECTORS HAVING DIFFERENT SIZES THAN THE SIZES OF THE PAGES AND/OR BLOCKS OF THE MEMORY


125
Attribute Data for an Associated User Data Block
123
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
127
135
137
139
141
143
145
147
PBN
Attributes
ECC
LBN
Time
ECC
125
Physical Attributes of Associated Block
Logical Attributes of Associated Block
131
135
129
ECC
127
Attributes of One Sector of User Data in the Associated Block

(57) **Abstract:** A non-volatile memory system, such as a flash EEPROM system, is disclosed to be divided into a plurality of blocks and each of the blocks into one or more pages, with sectors of data being stored therein that are of a different size than either the pages or blocks. One specific technique packs more sectors into a block than pages provided for that block. Error correction codes and other attribute data for a number of user data sectors are preferably stored together in different pages and blocks than the user data.

[redacted] [redacted]

TECHNIQUES FOR OPERATING NON-VOLATILE MEMORY SYSTEMS WITH DATA SECTORS HAVING DIFFERENT SIZES THAN THE SIZES OF THE PAGES AND/OR BLOCKS OF THE MEMORY

BACKGROUND OF THE INVENTION

This invention relates to semiconductor memory systems, particularly to non-volatile memory systems, and has application to flash electrically-erasable and programmable read-only memories (EEPROMs).

Flash EEPROM systems are being applied to a number of applications, particularly when packaged in an enclosed card that is removably connected with a host system. Current commercial memory card formats include that of the Personal Computer Memory Card International Association (PCMCIA), CompactFlash (CF), MultiMediaCard (MMC) and Secure Digital (SD). One supplier of these cards is SanDisk Corporation, assignee of this application. Host systems with which such cards are used include personal computers, notebook computers, hand held computing devices, cameras, audio reproducing devices, and the like. Flash EEPROM systems are also utilized as bulk mass storage embedded in host systems.

Such non-volatile memory systems include an array of floating-gate memory cells and a system controller. The controller manages communication with the host system and operation of the memory cell array to store and retrieve user data. The memory cells are grouped together into blocks of cells, a block of cells being the smallest grouping of cells that are simultaneously erasable. Prior to writing data into one or more blocks of cells, those blocks of cells are erased. User data are typically transferred between the host and memory array in sectors. A sector of user data can be any amount that is convenient to handle, preferably less than the capacity of the memory block, often being equal to the standard disk drive sector size, 512 bytes. In one commercial architecture, the memory system block is sized to store one sector of user data plus overhead data, the overhead data including information such as an error correction code (ECC) for the user data stored in the block, a history of use of the block, defects and other physical information of the memory cell block. Various implementations of this type of non-volatile memory system are described in the following United States patents and pending applications assigned to SanDisk Corporation, each of which is incorporated herein in its entirety by this reference:

[redacted] [redacted] [redacted]

Patents nos. 5,172,338, 5,602,987, 5,315,541, 5,200,959, 5,270,979, 5,428,621, 5,663,901, 5,532,962, 5,430,859 and 5,712,180, and application serial nos. 08/910,947, filed August 7, 1997 and 09/343,328, filed June 30, 1999.

Another type of non-volatile memory system utilizes a larger size memory cell block that each stores multiple pages per block, a page being a minimum unit of data that is programmed as part of a single programming operation. One sector of user data, along with overhead data related to the user data and the block in which such data is being stored, are typically included in a page. Yet another specific system that has been commercially available from SanDisk Corporation for more than one year from the filing date hereof, stores overhead data related to the user data being stored, such as ECC, along with the user data in a common sector, while overhead data related to the block in which the sector is stored is written as part of a different sector in a different block. An example of this system is given in patent application serial no. 09/505,555, filed February 17, 2000, which application is incorporated herein in its entirety by this reference.

One architecture of the memory cell array conveniently forms a block from one or two rows of memory cells that are within a sub-array or other unit of cells and which share a common erase gate. United States patents nos. 5,677,872 and 5,712,179 of SanDisk Corporation, which are incorporated herein in their entirety, give examples of this architecture. Although it is currently most common to store one bit of data in each floating gate cell by defining only two programmed threshold levels, the trend is to store more than one bit of data in each cell by establishing more than two floating-gate transistor threshold ranges. A memory system that stores two bits of data per floating gate (four threshold level ranges or states) is currently available, with three bits per cell (eight threshold level ranges or states) and four bits per cell (sixteen threshold level ranges) being contemplated for future systems. Of course, the number of memory cells required to store a sector of data goes down as the number of bits stored in each cell goes up. This trend, combined with a scaling of the array resulting from improvements in cell structure and general semiconductor processing, makes it practical to form a memory cell block in a segmented portion of a row of cells. The block structure can also be formed to enable selection of operation of each of the memory cells in two states (one data bit per cell) or in some multiple such as four states (two data bits per cell), as described in SanDisk Corporation

[redacted] [redacted]

United States patent no. 5,930,167, which is incorporated herein in its entirety by this reference.

Since the programming of data into floating-gate memory cells can take significant amounts of time, a large number of memory cells in a row are typically programmed at the same time. But increases in this parallelism cause increased power requirements and potential disturbances of charges of adjacent cells or interaction between them. United States patent no. 5,890,192 of SanDisk Corporation, which is incorporated herein in its entirety, describes a system that minimizes these effects by simultaneously programming multiple chunks of data into different blocks of cells located in different operational memory cell units (sub-arrays).

The foregoing referenced patents describe a memory array design wherein the individual memory cells are connected between adjacent bit lines in rows that include word lines, a "NOR" architecture. A "NAND" architecture is also commercially popular for non-volatile memory arrays, wherein a string of many memory cells are connected together in series between individual bit lines and a reference potential, rows of cells being formed from one cell of each of many such strings. Other specific architectures are also suggested in the literature. The foregoing referenced patents also describe the use of a type of non-volatile memory cell utilizing one or two floating gates made of a conductive material on which a level of electron charge is stored to control the effective threshold level of the cell. Alternative technologies for storing electrons, useful in various memory array architectures, includes those that trap electrons in a dielectric layer between two dielectric layers, rather than using conductive floating gates. Additionally, the foregoing referenced patents further describe the use of erase gates to which charge is removed from the cells' storage elements during an erasure of one or more blocks at a time. An alternative technique erases electrons from the storage element to the substrate as an erase electrode.

SUMMARY OF THE INVENTION

According to one aspect of the present invention, briefly and generally, rather than constraining one or some integer number of sectors of user data and accompanying overhead data to fill the individual data pages, at least some such sectors of data are split between two or more pages of memory. In one configuration, one or more sectors of user data, along with all the accompanying overhead data, a portion of the overhead data or none of the overhead data, are programmed together in one page of the memory while other sectors similarly constituted are divided and programmed into two or more pages in a manner that effectively utilizes the storage capacity of the pages. In a another configuration, the individual sectors of user data, with or without at least some part of its overhead data, are larger than the capacity of the pages in which they are stored, resulting in virtually every such sector being split between two pages. These approaches open up many possibilities for efficient use of a particular memory block and page architecture with an improved performance that are not possible when the memory system is constrained to store an integer number of data sectors in each page or block. The techniques have applications both in systems where blocks each contain many pages and in systems where the individual blocks contain a single page.

According to another aspect of the present invention, a sector contains primarily only user data while overhead data of both that user data and of the block in which it is written are stored as part of one or more other pages in other block(s) of memory cells. This has an advantage of increasing the amount of user data that may be stored in a block of a given size, having a particular application to the reading or programming of streaming data such as occurs when the content of the data is music, other audio, or video information. This technique also has the advantage that different frequencies of updating the user data and the overhead data may be accommodated without the updating of one necessitating the updating of the other. This improves system performance, particularly by reducing reading and/or programming times. Further, this allows the amount of overhead data that is stored to be independent of the size of the pages and blocks in which user data and overhead data are stored in existing systems.

In a specific example of a system incorporating both of these aspects of the present invention, a memory system having pages designed to store sectors of user data and accompanying overhead data are used differently than for which the system has been designed. A number of user data sectors, without most or all of the overhead data, are packed into a fewer number of memory pages, and the corresponding overhead data for a number of such user data sectors are combined together to form overhead data sectors that are stored in other pages of the memory. This has a particular advantage in long programming operations since the overhead data corresponding to user data sectors being written to the non-volatile memory may be accumulated in a faster buffer memory that is part of the controller, and then written from the buffer to the non-volatile memory all at once. It also increases speed during reading operations, since sectors of overhead data corresponding to user data sectors to be read are first read into the faster buffer memory of the controller. Overhead data that are necessary as part of reading the user data sectors can then be read faster from the buffer that they could from the non-volatile memory directly. This is a particular advantage with the ECCs of the user data sectors, which may be processed directly from the buffer memory as the user data sectors are read.

Additional aspects, features and advantages of the present invention are included in the following description of its exemplary embodiments, which description should be taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a schematic block diagram of a memory system which may be operated in accordance with the present invention;

Figure 2 illustrates typical components of a sector of data that is stored in present non-volatile memories;

Figure 3 illustrates one existing way of storing the data of Figure 2;

Figure 4 illustrates another existing way of storing the data of Figure 2;

Figure 5 illustrates a further existing way of storing the data of Figure 2;

Figures 6A and 6B illustrate alternate ways of storing the data of Figure 2, according to the present invention;

Figure 7 shows, in more detail, the architecture and use of a specific existing commercial memory system;

Figure 8 shows an example of packing sectors of user data into a block of the memory across its pages;

Figure 9 shows an example of the storage of pages of overhead data in a block different from that of corresponding user data of Figure 8;

Figure 10 illustrates and example page of overhead data of Figure 9;

Figure 11 illustrates a table that is built by the memory system controller from data within the overhead data stored in accordance with Figures 8 and 9;

Figure 12 shows a portion of the architecture of another specific existing commercial memory system; and

Figure 13 illustrates an example of packing sectors of user data into blocks of the memory system of Figure 12.

DESCRIPTION OF REPRESENTATIVE EMBODIMENTS

Figure 1 provides a diagram of the major components of a non-volatile memory system that are relevant to the present invention. A controller 11 communicates with a host system (not shown) over lines 13. The controller 11, illustrated to occupy one integrated circuit chip, communicates over lines 15 to one or more non-volatile memory cell arrays, one such array 17 being illustrated, each of the arrays usually formed on a separate integrated circuit chip. The illustrated controller is usually contained on a single integrated circuit chip, either without a flash EEPROM array (the example shown) or with some or all of the system's memory cell array. Even if a memory cell array is included on the controller circuit chip, an additional one or more chips that each contain only a memory array and associated circuitry will often be included in the system.

User data is transferred between the controller 11 and the memory array 17, in this example, over the lines 15. The memory array is addressed by the controller 11 also, in this example, over the lines 15. The memory system shown in Figure 1 can be embedded as part of a host system or packaged into a card, such as a card following one of the card standards previously described, or some other standard. In the case of a card, the lines 13 terminate in external terminals on the card for mating with a complementary socket within a host system. Although use of one controller chip and multiple memory chips is typical, the trend is, of course, to use fewer separate chips for such a system by combining their circuits. An example capacity of one of the illustrated memory chips is 256 Mbits, thus requiring only two such memory chips, plus the controller chip, to form a non-volatile memory system having a data capacity of 64 megabytes. Use of a single smaller capacity memory chip results in a memory system of lesser capacity, an 8 megabyte system being a marketable example. Conversely, use of memory chips with a higher bit storage density and/or use of more memory array chips in a system will result in a higher capacity systemThe controller 11 includes a micro-processor or micro-controller 23 connected through controller interface logic 25 to internal memories and interfaces with external components. A program memory 27 stores the firmware and software accessed by the micro-controller 23 to control the memory system operation to read data from the connected memory array(s) and transmit that data to the host, to write data from the host to the memory chip(s), and to carry out numerous other monitoring and controlling functions. The memory 27 can be a volatile re-programmable random-access-memory (RAM), a non-volatile memory that is not re-programmable (ROM), a one-time programmable memory (OTP) or a re-programmable flash EEPROM system. If the memory 27 is re-programmable, the controller can be configured to allow the host system to program it. A random-access-memory (RAM) 29 (such as a dynamic RAM (DRAM) or static RAM (SRAM)) is used to store, among other data, tables formed from data in the non-volatile memory 17 that are accessed during reading and writing operations. The RAM 29 also includes a number of registers used by the controller processor 23.

A logic circuit 31 interfaces with the host communication lines 13, while another logic circuit 33 interfaces with the memory array(s) through the lines 15. Another memory 35 is used as a buffer to temporarily store user data being transferred between the host system and non-volatile memory. The memories in the controller are usually volatile, since memories with fast access and other characteristics desired for efficient controller access have that characteristic, and may be combined physically into a single memory. A dedicated processing circuit 37 accesses the streaming user data being transferred between the controller and flash interfaces 25 and 33 for generating an ECC, or other type of redundancy code, from the user data. During programming, the generated ECC is stored in the memory array 17, along with the data from which is was calculated. During a reading operation, the ECC generated by the circuit 37 is compared with that read from the memory array 17 along with the data from which the read ECC was calculated during programming.

The flash memory 17 includes an array of memory cells that may be one of the types and according to one of the architectures that are described in the Background, or some other type and/or architecture. Such an array is physically divided into distinct blocks of memory cells that are simultaneously erasable, where a block is the smallest unit of memory cells that is erasable. The blocks individually contain the same number of memory cells, one size storing 528 bytes of data and another storing 4096 bytes of data, as examples. Of course, the number of memory cells required to store a given amount of data depends upon the number of bits of data that are stored in each cell. Multiple blocks of cells are usually addressed at one time for erasure together. The larger sized blocks are typically divided, in turn, into multiple pages of memory cells that are distinct, where a page is the smallest unit of memory cells that is programmable in a single program operation. As many memory cells as practical are programmed at the same time in order to reduce time necessary to program a given amount of data. In some systems, all of the cells in a page are programmed simultaneously, and in other systems, a distinct chunk of cells of a page are programmed one at a time, in a manner to minimize disturbing the charge on other cells, until all chunks have been programmed. In one specific system, there are four chunks in each page. A larger number of cells than in a single programming chunk are usually read in parallel.

With reference to Figure 2, components of a sector of data usually include a large number of bytes 43 of user data, a few bytes 45 of attributes of the user data and a few byte 47 of attributes of the page and/or block in which the entire sector is being stored. That is, a stream or file of user data is divided into user data sectors 43, and then the attribute data fields 45 and 47 are added to each user data sector to form a complete data sector for storage. A typical amount of user data which is included in the sector 43 is 512 bytes, the same as the amount of user data in a disk storage system sector. The attributes 45 of the user data usually include an ECC that is calculated by the controller from the user data stored in the same sector, both during programming and reading of the sector. The attributes 47 of the physical block (some of which may alternative be stored for each page) in which the sector is stored often include a physical address for the block, a logical address for the block, a number of times that the block has been erased, a programming and/or erase voltage that is to be applied to cells of the block, and any other such characteristics of the block. Another ECC is usually calculated from the attribute data and stored as part of the attributes fields.

The incoming data to a memory system is often transformed in some manner before being stored. For example, in a binary system, incoming data can be inverted prior to its programming in order to avoid repeatedly programming a static pattern in a given block and then changed back after a while, in order to even the wear of the block's memory cells. In a multi-state system, the data is rotated (translated) between its multiple states in some predetermined order. When the data is transformed, a flag indicating the transformation that has been applied is stored as part of either the data attributes 45, if the transform is made on a page basis, or as part of the block attributes 47, if the transform is made on a block basis. When such data is read from the memory, the stored transform flag then allows the controller to apply an inverse of the transform in order to translate the read data back to its form initially received before transmitting the data to the host. Another example of a transform that may be made is encryption of the user data, wherein the transform flag can then include an encryption key for use during decryption.

Figure 3 shows generally the data content of a block of memory cells that has been used for many years in the products of SanDisk Corporation, as an specific example of what has been done before. Each individual block 49 includes enough memory cells to store one sector's worth of data, namely 528 bytes. User data 51 is 512 bytes. In addition to the user data and block attributes 53 and 55, the remaining 16 bytes of storage space in the block 49 includes spare cells. Spare cells are provided in those systems that replace defective cells within a block. When this is done, addresses of defective cells are also part of the block attribute data 55.

A more recent variation used in a SanDisk product is illustrated in Figure 4. A primary difference with the data storage format of Figure 3 is that the block attributes are not stored in the same block as the user data. In a block 59, for example, 512 bytes of user data 61 are stored, and 8 bytes of ECC and flags are stored as user data attributes 63. This leaves a number of spare memory cells 65 in the block 59 that is sufficient to store 8 bytes of user and/or attribute data, to replace any defective cells within the block 59 in which user or attribute data would normally be stored. Attribute data 67 of the block 59 is stored in another block 69, and requires only 4 bytes. Indeed, the block 69 includes a number of such block attribute records for other blocks that contain user data. This data architecture is further described in aforementioned application serial no. 09/505,555.

Figure 5 illustrates one block 71 a memory system according to yet a different architecture that has been commercially used by others for some time. The block 71 here is much larger than that of the systems of Figures 3 and 4, typically having a data storage capacity of 16 or 32 kilobytes. The block 71 is divided into a multiple pages, such as 16, 32, or 64 pages. One page includes 512 bytes of user data 73 and 16 bytes total of user data attributes 75 and block attributes 77, in this specific example. No spare cells are provided. A page is a unit of programming and reading, while the larger block remains the unit for erase. An entirely programmed block is erased before data can be written to any one page within the block.

In each of the systems of Figures 3-5, the units of data stored are constrained to match the size of the blocks or pages of physical memory that are provided. Heretofore, a change in the data sector structure was thought to require a change in the physical memory structure to accept it. Such a change is quite expensive, takes a great deal of time, and necessarily discourages making changes to the data sector structure that may otherwise be desirable.

Therefore, according to one aspect of the present invention, a data sector structure is adapted to a different physical memory block and page structure without having to change the physical structure, as is very generally illustrated in Figure 6. Figure 6A shows the case where a sector or other unit of data being stored is smaller than the capacity of the individual memory cell blocks. A data sector 81, for example, fits totally within a page or block 83, while another data sector 85 is stored partially in the page or block 83 and partially in the page or block 87. Since data is written into and read from each of the pages or blocks in separate operations, the data sector 85 is written in two such operations. The portion of the data sector 85 that is written into the page or block 83 is preferably combined with the data sector 81 in a controller memory prior to such writing, in order to program the page or block 83 in a single operation. Data is preferably similarly combined from two adjacent data sectors prior to programming each of the pages or blocks.

Figure 6B illustrates an alternative situation, where the data sector is larger in size than can be stored in an individual page or block. A data sector 89 is stored in both of the pages or blocks 91 and 93. Another data sector 95 is stored in a remaining portion of the page or block 93 and in another page or block. Programming of pages or blocks that contain portions of two data sectors, such as 93, is preferably performed after the data from each of the two data sectors being partially stored therein, such as the data sectors 89 and 95, are assembled in a controller memory. Conversely, during reading of data sectors, the data in each of the two pages or blocks containing data from the desired sector are read and the data assembled in a controller memory into the units of the two data sectors. These two data sectors are then transferred to the host system by the controller.

The data sectors illustrated in Figure 6 may include (1) only user data, (2) a combination of user data and user data attributes, without block attributes, or (3) a combination of all of user data, user data attributes and block attributes. Any remaining attribute data may be stored in different blocks with a link provided between the two. The user data of a sector need not all be stored in a single physical page or block of the memory array.

A First System Embodiment

Figure 7 illustrates the array and data architectures of one specific existing commercial memory system, and Figure 8-11 show modifications made to this system in accordance with the present invention. The existing memory array 101 (Figure 7) is divided into a large number of blocks B, 4096 of them in this case. Each of these blocks, such as a block 103, is divided into a number of pages P, in this case 32 pages per block. Each page, such as the page 105, is configured to store 512 bytes of user data 107 and 16 bytes of attribute (overhead) data 109. The page attribute data 109 includes an ECC 111 calculated from at least the user data 107, a logical block number (LBN) 113 in which the page is located, and two flags 115 and 117. The ECC 111 is normally an attribute of the user data. A separate ECC (not shown) may be calculated from the remaining page attribute data and stored within the page attribute data 109 but some fields, such as the fields 113, 115 and 117, are often stored without a full ECC. The LBN 113 is an attribute of the block in which the page is located, and, as can be noted, is repeated in each of the 32 pages within a block.

Figure 8 illustrates a modified use of the block 103 to store an additional sector of user data but without the overhead data now included in each page. Thirty-three sectors S of user data (S0-S32) are stored in the thirty-two pages (P0-P31) of the block 103 of memory. Each user data sector contains 512 bytes of data. The first sector S0 fills all but the memory cells of the first page P0 that normally store the 16 bytes of overhead data. That 16 bytes of capacity is used to store the first 16 bytes of the next user data sector S1, the remaining of the data sector S1 being stored in the second page P1. That then leaves 32 bytes of capacity in the second page P1 that is used to store the beginning of the third data sector S2, the remaining portion of S2 being stored in the third page P2, and so on. It works out, because of the physical and data sector numbers, that each block can be fully filled with 33 sectors of user data. However, if such an even allocation does not exist in another system with different physical page, block and data sector sizes, that

[redacted] [redacted] [redacted]

remaining capacity can be used for a portion of another data sector, with the remaining portion of the data sector being stored in another page in a different block.

Since a page of the block 103 is the smallest unit of memory that may be written in one programming operation, user data from the various sectors S0-S32 is preferably assembled in a memory of the controller into the pages shown in Figure 8. That is, before the first page P0 is written to the block 103, the first 16 bytes of the second user data sector S1 is attached to the end of the sector S0, and the page is then programmed. Similarly, for the second page P1, the remaining data of the sector S1 is combined with the first 32 bytes of data from the sector S2 in a non-volatile memory of the controller, and the combination then written from that memory into the second page P1 of the non-volatile memory.

A different data structure of another block 121 is illustrated in Figure 9. Each page of that block, such as a page 123, stores a number of records of attribute (overhead) data that accompany the sectors of user data stored in other blocks such as the block 133 of Figure 8. In one embodiment, the page 123 stores all, or nearly all, of the storage block attribute data for the block 133 and the user data attributes for each of the 33 user data sectors stored in that block. A specific example of the data structure for the page 123, and every other page written to the block 121, is shown in Figure 10. Since there are 32 pages in the block 121, there will be at least one attribute data block of the type of block 121 for every 32 user data blocks of the type of block 103.

The attribute page shown in Figure 10 includes a field 125 that stores the attributes of the block of the form shown in Figure 8 in which the corresponding user data is stored. A separate field is provided for each of the data sectors stored in the block 103 of Figure 8, a field 127 (Figure 10) storing the attributes of the user data sector S25 of the block 103 (Figure 8). Each of the user data sector attribute records includes an ECC 129 calculated from the corresponding sector of user data. The rotation of that user data can be stored at 131, and other attributes of the user data of the corresponding sector potentially being stored at 133.

The attributes 125 (Figure 10) of the block 103 (Figure 8) are shown divided into physical and logical attributes, for purposes of discussion. The physical attributes, in this example, include a physical address (PBN) 135 of the user data block 103, and one or more other attributes 137 that can include an experience count of the number of times that the block has been erased, a programming voltage, an

[redacted] [redacted]

erase voltage, and the like. An ECC 139 may be provided for the fields 135 and 137. The logical attributes of the block 103 can include its logical address (LBN) 141 of the user data block 103, a time 143 at which the attribute page 123 was last written to the block 121, a transform flag 145 of the user data or the data within the page 123, and an ECC 147 of the fields 141, 143 and 145. The fields 131 and 135 can be provided without an ECC.

If the transform flag field 131 is not included in the user data attributes, the transform field 145 of the block attributes will specify the transform of the data in all of the user data sectors of the corresponding block 103 and of the attribute data page 123. If the transform field 131 is included, the transform field 145 only specifies the transform of the data in its page 123.

The time field 143 records some indication of when the corresponding user data block 103 has last been updated. If the memory system includes a real time clock, the time at which an update occurs is stored. But since most systems do not have such a clock, the field 143 may record a value indicative of when data of a corresponding user data block is updated. In a specific implementation, a separate count is maintained in the field 143 for each LBN, with a count being read and incremented each time the user data of its corresponding logical block is rewritten. The content of the field 143 is then used by the system controller to determine which of two blocks containing the same data or identified by the same LBN is the most recent.

An advantage of storing the attributes separate from the block and user data sectors is that the attribute data for an entire block of user data need be written only once. Further, the attributes of the corresponding user data block are stored only once as part of the attribute data page for that block, rather than being duplicated as part of each page of user data, as is currently being done. During programming, the page 123 (Figure 10) is formed in a memory of the controller as user data is written to its corresponding block 103. The page 123 is then written into the block 121 after all the user data of a file being stored has been written to the block 103. If the size of a file requires writing user data to more than one user data block, a distinct attribute record page is formed in the controller memory for each such block, and all the attribute record pages are then written to the non-volatile memory after all the user data of the file has been programmed.

When there is a change to the attribute data page 123 that needs to be recorded, for example, that page is read by the controller into one of its memories, changes are made to its data and the page then written back to an unused page of the block 121, such as a page 151. The time field 143 (Figure 10) of the attribute data is updated, so that the processor can distinguish between the current and old versions of that data. Only the current version is used. If there is no unwritten page in the block 121, the updated attribute data is written to some other block of attribute pages that has a spare. When a sufficient number of old attribute records exist in the block 121 and others like it, the current records are read into a memory of the controller, the block may be erased and the current records written back into the block. Since the memory of the controller will usually be a volatile type, in order to avoid loss of data resulting from a loss of power, the current records of a block may alternatively be copied to a different previously erased non-volatile memory block prior to erasure of the original block. In either case, the result is the creation of spare, unwritten pages in either the original or a different block for use in the future to store updated or new attribute pages.

The very specific example being described with respect to Figures 8-11 can be modified in many ways, while still implementing the present invention. For instance, an attribute data page can include records of more than one user data block if the size of each of the records is reduced, or if a different array architecture stores fewer sectors of user data, either because the blocks are smaller or the sectors are larger, or if the size of the pages is smaller, etc. Conversely, a still different architecture or larger attribute record size can require more than one page of attribute data per block of user data. The present invention is applicable to non-volatile memory systems within a wide range of physical and data architectures.

When a request is received from a host to read sectors of user data from the non-volatile memory, a beginning logical block address (LBN) and a number of blocks containing data to be read are determined by the controller. The attribute data pages, such as those described with respect to Figures 9 and 10, are then scanned by the controller to identify the page or pages (Figure 10) whose LBN fields 141 are within the range of LBNs specified by the controller for reading. A table is then built in the form of Figure 11 and stored in volatile controller memory. One column 153 lists the PBNs in the attribute fields 135 whose LBNs 141 are within that range, the LBNs then becoming addresses to the table, as illustrated by a column 155. The

[redacted] [redacted] [redacted]

physical address (PBN) and page number for each attribute page from which a user data PBN is present in column 153 is included in a line of the table of Figure 11 that contains that user data PBN, thus forming columns 157 and 159.

This table is then accessed during a read operation by the user data LBN to determine the physical block location (PBN) where the requested user data reside, and the physical address of the corresponding attribute data page. The attribute data pages for the entire read operation are then read into a memory of the controller, if the controller memory has sufficient capacity, or, alternatively, read into the controller memory one or a few pages at a time as needed to execute the read operation. As the user data are then read from the non-volatile memory, the ECCs of the user data attribute data records (such as record 127 of Figure 10) are read by the controller and processed with the read user data, in order to identify and correct any errors before sending the read user data to the host.

The forgoing example provides for storing only user data in one block and all related attribute data in another. However, there may be instances where it is preferable to include a user data attribute as part of the individual user data sectors, or as a separate record within a block that otherwise stores only user data sectors. One such attribute is the transform flag of the user data, so that the controller need not access the separate attribute data page before being able to read those sectors. If the transform flag is included with the user data, the controller then knows how to apply the inverse transform to read that data without having to separately access the corresponding attribute page. Such inclusion is preferably limited to specific applications where doing so improves the performance of the memory system in terms of reading time.

A Second System Embodiment

Application of the invention to another type of non-volatile memory system with a much different architecture is briefly described with respect to Figures 12 and 13. The memory array is divided into an even number of units, such as eight, two such units 0 and 1 being illustrated in Figure 2. A pair of adjacent units, termed a plane, may share peripheral circuits, such as word line decoders. Each unit contains a large number of blocks of memory, such as a block 161 in the unit 0 and a block 163 in the unit 1. The individual blocks are in turn divided into multiple pages of memory.

[redacted] [redacted]

Rather than forming a full page from memory cells of a single block, which is usually done, the example shown puts one-half 165 of a given page in the block 161 of the unit 0 and the other one-half 167 of the same page in a block of the unit 1. The two blocks may have the same relative address within the units, and may be formed of a single row of memory cells that has a common word line extending through the two blocks. It may be preferable, however, for the corresponding one-half pages to be in blocks that do not share a word line. In either event, this page segmentation allows the simultaneous programming of an increased number of a page's cells in parallel.

The memory pages of this different architecture may be used in the same manner as described above with respect to the first example. Sectors of user data smaller or larger than the capacity of the pages are stored across adjacent pages within the individual blocks, except that one-half of each user data sector is stored in one-half of the page in one unit's block and the other one-half of the sector is stored in the remaining one-half of the page in the other unit's block. Similarly, for those blocks containing pages devoted to the storage of attribute records of the user data and the blocks in which such data are stored, one-half of a page of attribute data is stored in one block and the other one-half in its partner block. The operation of such a memory system is similar to that described above for the first system embodiment, except that individual pages are read from half-page portions in two blocks.

It should be noted that the memory systems of any of the forgoing applications may be implemented by storing one bit of data per memory cell storage element or, alternatively, by storing two or more bits per storage element. One bit is stored when the memory cell is operated in two states with only two threshold voltage levels (binary), and more than one bit is stored when operated in more than two states by operating the cells with more than two threshold voltage levels (multi-state). Many of the patents and applications identified above as being incorporated herein describe further aspects of binary and multi-state operation. Each of the two or more bits stored in an individual cell can either be addressed in a common page or in different pages within a memory system organized in the manner described above.

Although the various aspects of the present invention have been described with respect to specific implementation examples, it will be understood that the invention is entitled to protection within the full scope of the appended claims.

IT IS CLAIMED:

1. A method of operating a memory system having a plurality of memory cells organized into a plurality of blocks that individually contain the smallest group of memory cells that are simultaneously erasable by addressing individual blocks, said blocks being individually programmable in units of an integer number of a plurality of pages of a given amount of data per page, comprising programming sectors of data individually containing less than said given amount of data across boundaries of said pages within individual blocks, wherein more sectors of data are programmed into a block than a number of pages in the block.

2. The method of claim 1, wherein said sectors of data individually contain all of user data, data of attributes of the user data and data of attributes of the block in which said sectors of data are programmed.

3. The method of claim 1, wherein said sectors of data individually contain both user data and data of attributes of the user data, with data of attributes of the block in which said sectors of data are programmed being stored as part of different data sectors.

4. The method of claim 1, wherein said sectors of data individually contain user data, with data of attributes of the user data and data of attributes of the block in which the individual sectors of data are being programmed being stored as part of different data sectors.

5. The method of either one of claims 3 or 4, wherein said different data sectors are stored in different blocks than said sectors of data to which the data of attributes pertains.

6. The method of any one of claims 1-4, additionally comprising operating the memory cells with a plurality of effective threshold levels in excess of two that correspond to a plurality of alterable states of the individual cells in excess of two, whereby storage elements of the cells individually store more than one bit of data.

7. A method of operating a memory system having a plurality of memory cells organized into a plurality of blocks that individually contain the smallest group of memory cells that are simultaneously erasable by addressing individual blocks, said blocks being individually programmable in units of one or more integer numbers of a pages of a given amount of data per page, comprising programming sectors of data individually containing more than said given amount of data across boundaries of said pages.

8. The method of claim 7, wherein said sectors of data individually contain all of user data, data of attributes of the user data and data of attributes of the block in which said sectors of data are programmed.

9. The method of claim 7, wherein said sectors of data individually contain both user data and data of attributes of the user data, with data of attributes of the block in which said sectors of data are programmed being stored as part of different data sectors.

10. The method of claim 7, wherein said sectors of data individually contain user data, with data of attributes of the user data and data of attributes of the block in which the individual sectors of data are being programmed being stored as part of different data sectors.

11. The method of either one of claims 9 or 10, wherein said different data sectors are stored in different blocks than said sectors of data to which the data of attributes pertains.

12. The method of any one of claims 7-10, wherein said blocks individually include only one page.

13. The method of any one of claims 7-10, wherein said blocks individually include a plurality of pages.

14. The method of any one of claims 7-10, additionally comprising operating the memory cells with a plurality of effective threshold levels in excess of two that correspond to a plurality of alterable states of the individual cells in excess of two, whereby storage elements of the cells individually store more than one bit of data.

15. In a non-volatile memory system having memory cells organized into a plurality of blocks that are individually addressable for simultaneously erasing the memory cells within a block, wherein the blocks individually store a plurality of pages of data, the pages being designated to individually store at least one sector of user data and associated overhead data including at least one attribute of the associated user data stored in the page and at least one physical attribute of the block in which the page is stored, an improved method of operating the memory system, comprising:

storing user data in portions of the pages designated to store overhead data in a manner that at least one additional sector of user data is stored in individual ones of the blocks without the storage of overhead data in said individual user data blocks, and

storing said overhead data for a plurality of the user data blocks as corresponding individual records in blocks distinct from those storing the user data.

16. The method according to claim 15, wherein storing said overhead data includes storing overhead data records that individually include a field of attributes of a corresponding one of the user data blocks and a field of user data attributes for individual ones of the user data sectors stored in the corresponding user data block.

17. The method according to claim 16, wherein storing said overhead data includes storing both physical and logical attributes in the field of block attributes.

18. The method according to claim 16, wherein storing said overhead data includes storing within individual ones of the overhead data records logical and physical addresses of the corresponding block.

[redacted] [redacted] [redacted]

19. The method according to claim 16, wherein the field of user data attributes includes an error correction code calculated from the user data stored in a corresponding one of the pages in the corresponding block.

20. The method according to claim 16, wherein the field of physical block attributes includes a count of a number of times that the corresponding block has been erased.

21. A method of operating a memory system having a plurality of memory cells organized into a plurality of blocks that are individually addressable for simultaneously erasing the memory cells within a block, comprising:

storing multiple sectors of user data within individual ones of a first group of blocks, and

storing a plurality of records in individual ones of a second group of blocks different from the first group of blocks, wherein said plurality of records individually include overhead information of attributes of a corresponding one of the first group of blocks and the user data stored therein.

22. The method of claim 21, wherein storing records including overhead information of attributes includes storing logical and physical addresses of the corresponding user data block.

23. The method of claim 22, additionally comprising forming in volatile memory a temporary table of the logical and physical addresses of the corresponding user data blocks by reading the corresponding overhead records.

24. The method of claim 21, wherein storing records including overhead information of attributes includes storing an error correction code that has been calculated from the user data stored in a corresponding one of the first set of blocks.

[redacted] [redacted]

25. The method of claim 21, wherein storing records including overhead information of attributes includes storing a count of a number of times that the corresponding one of the first set of blocks has been programmed.

26. The method of any one of claims 21-25, wherein said individual ones of the first group of blocks do not contain any of said overhead information.

[illegible]

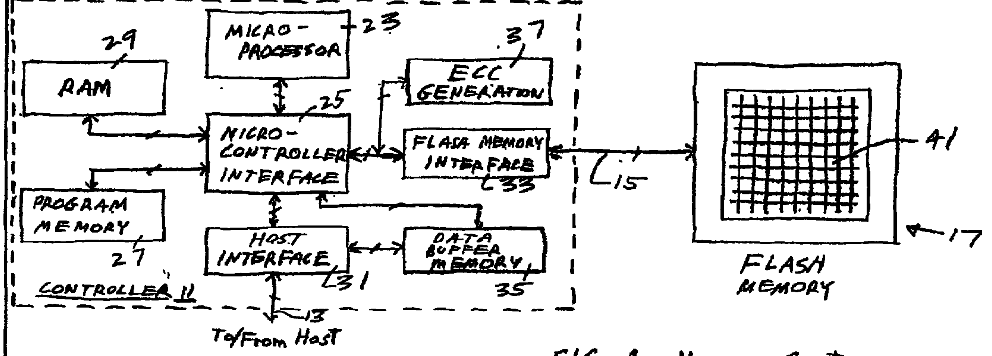

MICRO-PROCESSOR
23
29
RAM
37
ECC GENERATION
25
MICRO-CONTROLLER INTERFACE
FLASH MEMORY INTERFACE
33
15
41
PROGRAM MEMORY
27
HOST INTERFACE
31
DATA BUFFER MEMORY
35
CONTROLLER 11
13
To/From Host
17
FLASH MEMORY

FIG. 1 - Memory System

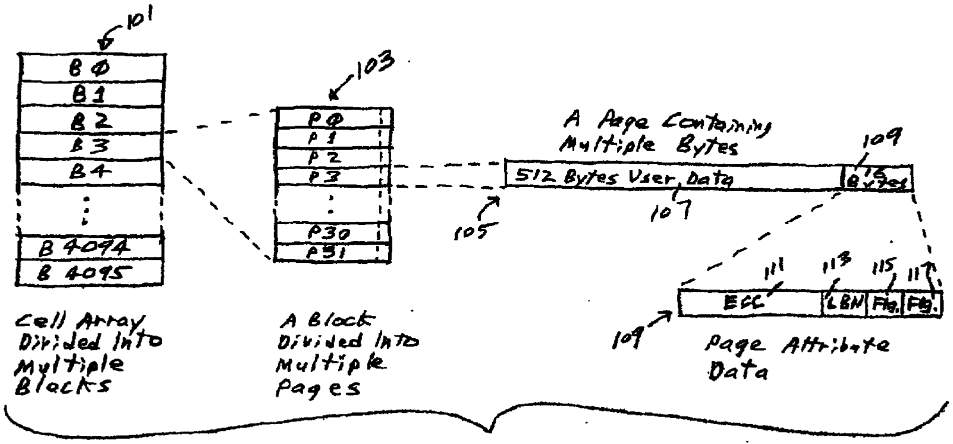

101
B0
B1
B2
B3
B4
B 4094
B 4095
103
P0
P1
P2
P3
P30
P31
A Page Containing Multiple Bytes
109
512 Bytes User Data
16 Bytes
105
107
111
113
115
117
ECC
LBN
Flg.
Flg.
109
Cell Array Divided Into Multiple Blocks
A Block Divided Into Multiple Pages
Page Attribute Data

FIG. 7 - PRIOR ART

[redacted] [redacted] [redacted]

MARI 234US0

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS

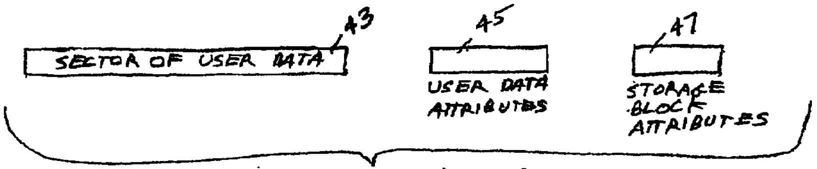

SECTOR OF USER DATA
43
45
USER DATA ATTRIBUTES
47
STORAGE BLOCK ATTRIBUTES

FIG. 2 - Prior Art

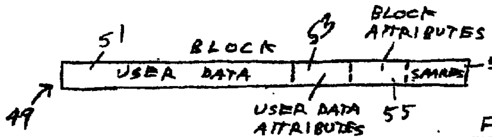

51
BLOCK
53
BLOCK ATTRIBUTES
USER DATA
SPARES
57
49
USER DATA ATTRIBUTES
55

FIG. 3 - Prior Art

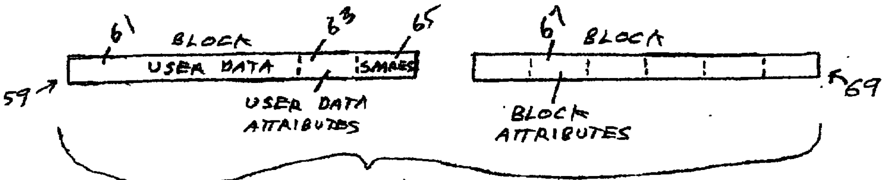

61
BLOCK
63
65
USER DATA
SPARES
59
USER DATA ATTRIBUTES
67
BLOCK
69
BLOCK ATTRIBUTES

Fig. 4 - Prior Art

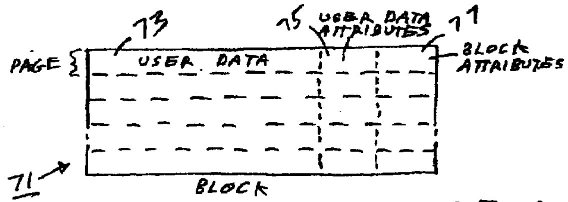

73
75
USER DATA ATTRIBUTES
77
PAGE
USER DATA
BLOCK ATTRIBUTES
71
BLOCK

FIG. 5 - Prior Art

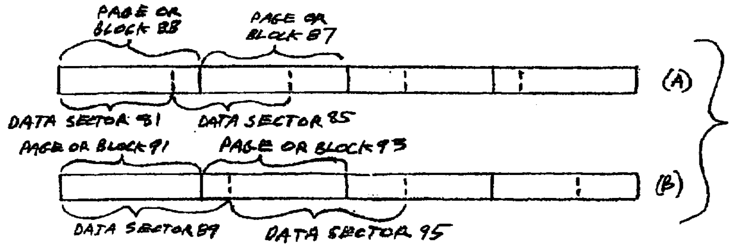

PAGE OR BLOCK 83
PAGE OR BLOCK 87
(A)
DATA SECTOR 81
DATA SECTOR 85
PAGE OR BLOCK 91
PAGE OR BLOCK 93
(B)
DATA SECTOR 89
DATA SECTOR 95

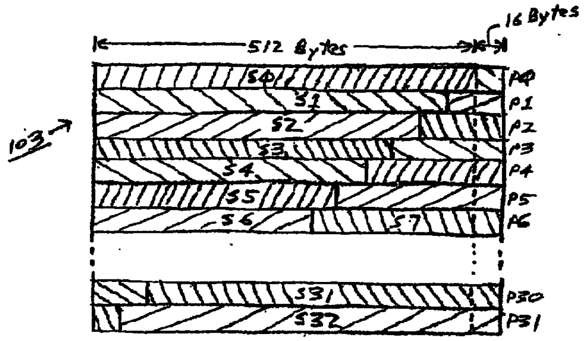
FIG. 6

[redacted] [redacted]

[redacted] [redacted] [redacted]

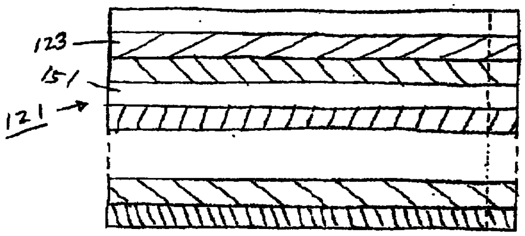
HART, [illegible]

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS


16 Bytes
512 Bytes
103
S0
S1
S2
S3
S4
S5
S6
S7
S31
S32
P0
P1
P2
P3
P4
P5
P6
P30
P31

A Block Storing More User Data Sectors them Physical Pages In the Block

FIG. 8



A Block Storing Attribute Data For One User Data Block In an Individual Page

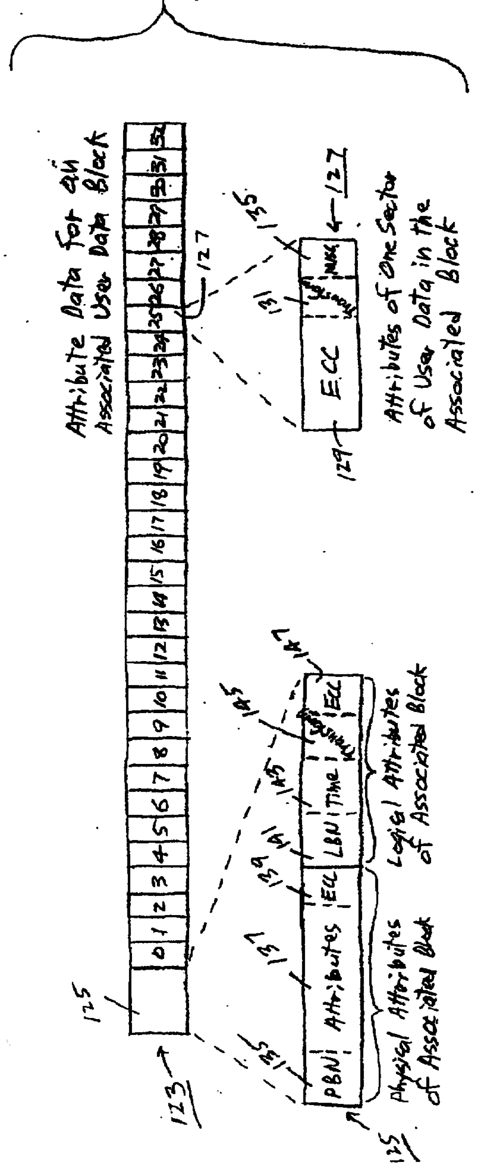
FIG. 9

[redacted] [redacted] [redacted]

HARF.ZB4450

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS

Fig. 10


Attribute Data for an Associated User Data Block
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
123
125
127
ECC
129
131
135
127
Attributes of One Sector of User Data in the Associated Block
PBN
Attributes
ECC
LBN
TIME
ECC
125
133
137
139
141
143
145
147
Physical Attributes of Associated Block
Logical Attributes of Associated Block

[redacted]

HABI 234US0

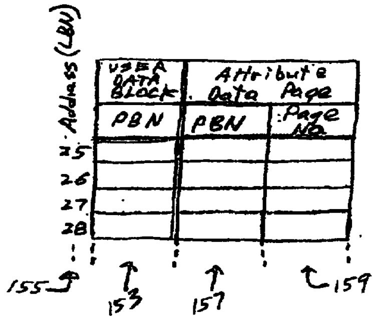

Address (LBN)
USER DATA BLOCK
Attribute Data Page
PBN
PBN
Page No.
25
26
27
28
155
153
157
159

FIG. 11 — Table in Controller RAM Built from Fig. 10 Attribute Records

22-141 50 SHEETS
22-142 100 SHEETS
22-144 200 SHEETS
AMPAD

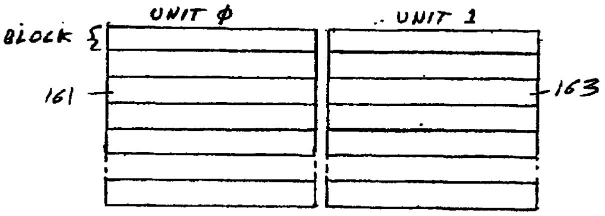

UNIT 0
UNIT 1
BLOCK
161
163

FIG. 12 — Prior Art

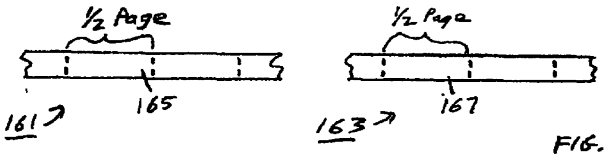

½ Page
½ Page
161
165
163
167

FIG. 13

DIALOG(R)File 351: Derwent WPI
(c) 2007 The Thomson Corporation. All rights reserved.

0008245975 *Drawing available*
WPI Acc no: 1997-353244/199733
XRPX Acc No: N1997-292674

Writing to re-writable memory e.g. memory card, smart card or non-contact card - has multiple zones in memory with flags associated with each zone and uses sequence or discontinuity of flag identifiers to indicate current zone and next zone for writing

Patent Assignee: SCHLUMBERGER IND SA (SLMB)
Inventor: DETURCHE J C

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
FR 2742893	A1	19970627	FR 199515186	A	19951220	199733	B

Priority Applications (no., kind, date): FR 199515186 A 19951220

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes
FR 2742893	A1	FR	14	3	

Alerting Abstract FR A1
The method of writing to EPROM consists of writing current data successively to different zones of memory, and programming flags to designate zones containing the most recent data written to memory. The memory is decomposed to N zones, each zone having a first space for the flags and a second space for the data.
The current zone is indicated either by a discontinuity in the sequence of flags or by the last zone number in the absence of discontinuity. A new data set is indicated in the zone indicated by the next flag identifier in the sequence. The flag is updated when the data is written.
USE/ADVANTAGE - Writing to EEPROM or flash EPROM. Simplified memory management in writing and verification of EPROM. Simplified error recovery. Increased number of re-writes of memory.

Title Terms /Index Terms/Additional Words: WRITING; MEMORY; CARD; SMART; NON; CONTACT; MULTIPLE; ZONE; FLAG; ASSOCIATE; SEQUENCE; DISCONTINUE; IDENTIFY; INDICATE; CURRENT; EEPROM; FLASH; EPROM

[redacted] [redacted] [redacted]

(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 742 893**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **95 15186**

(51) **Int Cl⁶ : G 06 F 12/02, G 11 C 8/00, 16/02**

(12) DEMANDE DE BREVET D'INVENTION **A1**

(22) **Date de dépôt** : 20.12.95.

(30) **Priorité** :

(43) **Date de la mise à disposition du public de la demande :** 27.06.97 Bulletin 97/26.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Se reporter à la fin du présent fascicule.*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *SCHLUMBERGER INDUSTRIES SA SOCIETE ANONYME* — FR.

(72) **Inventeur(s) :** DETURCHE JEAN CLAUDE.

(73) **Titulaire(s) :**

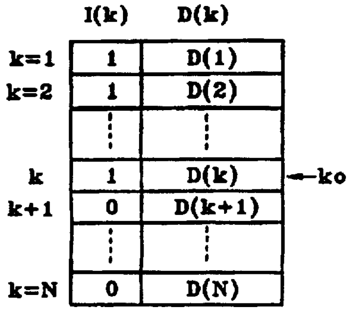
(74) **Mandataire :** SCHLUMBERGER INDUSTRIES.

(54) PROCEDE D'INSCRIPTION D'UNE DONNEE DANS UNE MEMOIRE REINSCRIPTIBLE.

(57) Procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celles desdites zones contenant la dernière valeur courante inscrite de la donnée D.

Selon l'invention, ladite mémoire est décomposée en N zones k (k=1,..., N), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur I(k) et un deuxième espace-mémoire affecté à une valeur courante D(k) de la donnée D, la zone ko contenant la dernière valeur courante D(ko) inscrite étant définie, soit par une discontinuité dans la suite des indicateurs I(k), soit par la dernière zone N(ko=N) en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko+1 avec mise à jour de l'indicateur I (ko + 1).

Application aux cartes à mémoire électronique telles que les cartes à mémoire simple, les cartes à microprocesseur et les cartes sans contact.



	I(k)	D(k)	
k=1	1	D(1)	
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	←ko
k+1	0	D(k+1)	
	⋮	⋮	
k=N	0	D(N)	

FR 2 742 893 - A1

[redacted]

I(k)	D(k)	CRC(D(k))
1	D(1)	CRC(D(1))
1	D(2)	CRC(D(2))
0	D(3)	CRC(D(3))
	⋮	⋮
0	D(N)	CRC(D(N))

FIG. 3

puisse désigner la zone dans laquelle est inscrite la dernière valeur courante de la donnée D.

L'inconvénient de ce type de procédé est qu'il nécessite plusieurs opérations d'inscription qui peuvent, chacune, être le siège d'une corruption. Le logiciel gérant les inscriptions dans la mémoire doit donc être complexe et, de là, consommateur de temps et d'espace-mémoire.

Aussi, le problème technique à résoudre par l'objet de la présente invention est de proposer un procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celle desdites zones contenant la dernière valeur courante inscrite de la donnée D, procédé qui permettrait de simplifier la gestion des opérations d'inscription tant en assurant l'augmentation de la durée de vie des cellules de la mémoire ainsi que l'intégrité de la donnée D au moment de l'inscription de la dernière valeur courante.

La solution au problème technique posé consiste, selon la présente invention, ce que ladite mémoire est décomposée en N zones k (k= 1, ..., N), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur I (k) et un deuxième espace-mémoire affecté à une valeur courante D (k) de la donnée D, la zone ko contenant la dernière valeur courante D (ko) inscrite étant définie, soit par une discontinuité dans la suite des indicateurs I(k), soit par la dernière zone N (ko = N) en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko + 1 avec mise à jour de l'indicateur I (ko + 1).

Le procédé de l'invention présente plusieurs avantages.

En premier lieu, le gestionnaire de mémoire se trouve simplifié puisqu'après avoir inscrit dans un premier temps la dernière valeur courante de la donnée D et vérifié que cette inscription est correcte, il suffit de mettre à jour l'indicateur correspondant. Sinon, l'indicateur n'est pas mis à jour et la valeur courante précédente reste "active". Il

	I(k)	D(k)	
k=1	1	D(1)	
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	← ko
k+1	0	D(k+1)	
	⋮	⋮	
k=N	0	D(N)	

FIG. 1a

I(k)	D(k)	
1	D(1)	
1	D(2)	
⋮	⋮	
1	D(k)	
1	D(k+1)	
⋮	⋮	
1	D(N)	← ko

FIG. 1b

k=1	0	D'(1)	← ko
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	
k+1	1	D(k+1)	
	⋮	⋮	
k=N	1	D(N)	

FIG. 1c

7	D(1)	← ko
6	D(2)	
⋮	⋮	
6	D(k)	
6	D(k+1)	
⋮	⋮	
6	D(N)	

FIG. 1d

Les figures 1a à 1c représentent une mémoire réinscriptible à différents stades d'inscription selon un premier procédé conforme à l'invention.

La figure 1d représente la mémoire réinscriptible des figures 1a et 1c inscrites selon une variante du premier procédé d'inscription.

Les figures 2a à 2d représentent une mémoire réinscriptible à différents stades d'inscription selon un deuxième procédé conforme à l'invention.

La figure 3 représente une mémoire réinscriptible analogue à celle des figures 1a à 1c, complétée par une information de contrôle de cohérence.

La mémoire réinscriptible montrée sur les figures 1a à 1c est destinée à recevoir N valeurs courantes successives $D(k)$ $(k=1, ..., N)$ d'une donnée D selon un procédé d'inscription qui consiste notamment à décomposer ladite mémoire en N zones k comprenant, chacune, un premier espace mémoire affecté à un indicateur $Z(k)$ et un deuxième espace-mémoire, contigu à la première, dans laquelle est inscrite une valeur courante $D(k)$, de la donnée D. Le rôle des indicateurs $I(k)$ est de pouvoir désigner à un moment donné celle desdites zones, notée ko, contenant la dernière valeur courante $D(ko)$ inscrite dans la mémoire, et, par voie de conséquence, à repérer la zone dans laquelle devra être inscrite une nouvelle valeur courante de la donnée D.

Comme on peut le voir sur la figure 1a qui décrit un stade quelconque d'inscription de la mémoire, les valeurs courantes $D(k)$ de la donnée D sont inscrites successivement, dans l'ordre, dans les différentes zones k correspondantes de la mémoire, c'est-à-dire la valeur $D(1)$ en zone $z=1$, la valeur $D(2)$ en zone $z=2$, etc... La zone ko contenant la dernière valeur courante $D\ (k=ko)$ inscrite est définie par une discontinuité dans la suite des indicateurs $I(k)$.

D'une manière générale, pour repérer ladite discontinuité, et donc la zone ko, on définit une relation $I(k+1) = f[I(k), k]$ qui doit être normalement vérifiée entre l'indicateur $I(k+1)$ et l'indicateur précédent $I(k)$, la discontinuité apparaissant précisément à la zone ko pour laquelle la relation précitée n'est pas vérifiée : $I(ko + 1) = f\ (I(ko), ko)$.

REVENDICATIONS

1. Procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celle desdites zones contenant la dernière valeur courante inscrite de la donnée D, caractérisé en ce que ladite mémoire est décomposée en N zones k ($k=1, \ldots, N$), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur $I(k)$ et un deuxième espace-mémoire affecté à une valeur courante $D(k)$ de la donnée D, la zone ko contenant la dernière valeur courante $D(ko)$ inscrite étant définie, soit par une discontinuité dans la suite des indicateurs $I(k)$, soit par la dernière zone $N(ko=N)$ en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko+1 avec mise à jour de l'indicateur $I\ (ko + 1)$.
2. Procédé selon la revendication 1, caractérisé en ce que ladite discontinuité consiste dans le fait que pour ladite zone zo une relation $I(k+1) = f[I(k), k]$ entre l'indicateur $I(k+1)$ et l'indicateur précédent $I(k)$ n'est pas vérifiée : $I\ (ko +1) = f[I(ko), ko]$.
3. Procédé selon la revendication 2, caractérisé en ce que la fonction f est définie par $I(k+1)=I(k)$.
4. Procédé selon la revendication 3, caractérisé en ce que la mise à jour de l'indicateur $I(ko+1)$ est réalisée par le passage d'un nombre binaire 0 ou 1 à l'autre : $I'(ko+1) = I(ko+1)$.
5. Procédé selon la revendication 4, caractérisé en ce que le nombre de valeurs courantes $D(1)$ ayant été inscrites dans la zone $z=1$ est stocké dans un compteur.
6. Procédé selon la revendication 3, caractérisé en ce que ladite mise à jour d'indicateur $I(ko+1)$ est réalisée par incrémentation d'une unité : $I'(ko + 1) = I\ (ko + 1) +1$, $I(k)$ représentant le nombre de valeurs courantes $D(k)$ ayant été inscrites dans la zone k.
7. Procédé selon la revendication 2, caractérisé en ce que la fonction f est définie par $I(k+1) = I(k) + g(k)$.

(ko+1)+1. Dans ce cas, l'indicateur I(k), outre sa fonction d'indication, représente le nombre de valeurs courantes D(k) ayant été inscrites dans la zone k. Il n'est alors plus besoin de compteur spécifique.

Sur les figures 2a à 2d est représenté un deuxième procédé d'inscription conforme à l'invention caractérisé par le fait que, d'une façon générale, la fonction f de régularité est définie par I(k+1)=I(k) +g(k), g(k) étant égal à 1 dans l'exemple proposé. Dans la suite normale des indicateurs I(k), un indicateur donné se déduit de l'indicateur précédent par incrémentation d'une unité, la discontinuité apparaît lorsque deux indicateurs consécutifs ne satisfont pas cette condition, en pratique cette situation se produit pour l'égalité de deux indicateurs consécutifs. Comme pour le premier procédé précédemment décrit, en l'absence de discontinuité, la zone ko de la dernière inscription sera la dernière zone ko = N.

La figure 2a montre l'état initial de la mémoire : aucune inscription n'y est portée et les indicateurs ne présentent aucune discontinuité vérifiant tous la relation de régularité. Par conséquent, en vertu de la règle énoncée plus haut, la première valeur courante est inscrite dans la zone k=1, l'indicateur I(1) étant mis à jour par incrémentation d'une unité en passant de 0 à 1. Cette opération, illustrée sur la figure 2b, fait apparaître une discontinuité entre les indicateurs I(1) et I(2) permettant de repérer la zone ko = 1 comme celle contenant la dernière valeur courante D(1) de la donnée D.

Il en résulte qu'une nouvelle valeur courante sera inscrite dans la zone k=2 avec mise à jour de l'indicateur I(2) par passage de la valeur 1 à la valeur 2 et apparition d'une discontinuité entre les indicateurs I(2)et I(3), et ainsi de suite jusqu'à aboutir à la situation de la figure 2c où la mémoire est complètement remplie.

Si une nouvelle valeur courante doit alors être inscrite, elle le sera en zone k=1, la nouvelle valeur D'(1) remplaçant l'ancienne valeur D(1). Cette substitution s'accompagne d'une mise à jour de l'indicateur I(1) qui passe de 1à 2, comme on peut le voir sur la figure 2d.

On notera qu'avec ce procédé, il est très facile d'établir le nombre de valeurs courantes D(k) ayant été inscrites dans la zone k, ce nombre étant donné par I(k) - k+1.

(ko+1)+1. Dans ce cas, l'indicateur I(k), outre sa fonction d'indication, représente le nombre de valeurs courantes D(k) ayant été inscrites dans la zone k. Il n'est alors plus besoin de compteur spécifique.

Sur les figures 2a à 2d est représenté un deuxième procédé d'inscription conforme à l'invention caractérisé par le fait que, d'une façon générale, la fonction f de régularité est définie par I(k+1)=I(k) +g(k), g(k) étant égal à 1 dans l'exemple proposé. Dans la suite normale des indicateurs I(k), un indicateur donné se déduit de l'indicateur précédent par incrémentation d'une unité, la discontinuité apparaît lorsque deux indicateurs consécutifs ne satisfont pas cette condition, en pratique cette situation se produit pour l'égalité de deux indicateurs consécutifs. Comme pour le premier procédé précédemment décrit, en l'absence de discontinuité, la zone ko de la dernière inscription sera la dernière zone ko = N.

La figure 2a montre l'état initial de la mémoire : aucune inscription n'y est portée et les indicateurs ne présentent aucune discontinuité vérifiant tous la relation de régularité. Par conséquent, en vertu de la règle énoncée plus haut, la première valeur courante est inscrite dans la zone k=1, l'indicateur I(1) étant mis à jour par incrémentation d'une unité en passant de 0 à 1. Cette opération, illustrée sur la figure 2b, fait apparaître une discontinuité entre les indicateurs I(1) et I(2) permettant de repérer la zone ko = 1 comme celle contenant la dernière valeur courante D(1) de la donnée D.

Il en résulte qu'une nouvelle valeur courante sera inscrite dans la zone k=2 avec mise à jour de l'indicateur I(2) par passage de la valeur 1 à la valeur 2 et apparition d'une discontinuité entre les indicateurs I(2)et I(3), et ainsi de suite jusqu'à aboutir à la situation de la figure 2c où la mémoire est complètement remplie.

Si une nouvelle valeur courante doit alors être inscrite, elle le sera en zone k=1, la nouvelle valeur D'(1) remplaçant l'ancienne valeur D(1). Cette substitution s'accompagne d'une mise à jour de l'indicateur I(1) qui passe de 1à 2, comme on peut le voir sur la figure 2d.

On notera qu'avec ce procédé, il est très facile d'établir le nombre de valeurs courantes D(k) ayant été inscrites dans la zone k, ce nombre étant donné par I(k) - k+1.

REVENDICATIONS

1. Procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celle desdites zones contenant la dernière valeur courante inscrite de la donnée D, caractérisé en ce que ladite mémoire est décomposée en N zones k ($k=1,\ldots, N$), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur $I(k)$ et un deuxième espace-mémoire affecté à une valeur courante $D(k)$ de la donnée D, la zone ko contenant la dernière valeur courante $D(ko)$ inscrite étant définie, soit par une discontinuité dans la suite des indicateurs $I(k)$, soit par la dernière zone $N(ko=N)$ en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko+1 avec mise à jour de l'indicateur $I(ko+1)$.
2. Procédé selon la revendication 1, caractérisé en ce que ladite discontinuité consiste dans le fait que pour ladite zone zo une relation $I(k+1)=f[I(k), k]$ entre l'indicateur $I(k+1)$ et l'indicateur précédent $I(k)$ n'est pas vérifiée : $I(ko+1) = f[I(ko), ko]$.
3. Procédé selon la revendication 2, caractérisé en ce que la fonction f est définie par $I(k+1)=I(k)$.
4. Procédé selon la revendication 3, caractérisé en ce que la mise à jour de l'indicateur $I(ko+1)$ est réalisée par le passage d'un nombre binaire 0 ou 1 à l'autre : $I'(ko+1) = I(ko+1)$.
5. Procédé selon la revendication 4, caractérisé en ce que le nombre de valeurs courantes $D(1)$ ayant été inscrites dans la zone $z=1$ est stocké dans un compteur.
6. Procédé selon la revendication 3, caractérisé en ce que ladite mise à jour d'indicateur $I(ko+1)$ est réalisée par incrémentation d'une unité : $I'(ko+1) = I(ko+1)+1$, $I(k)$ représentant le nombre de valeurs courantes $D(k)$ ayant été inscrites dans la zone k.
7. Procédé selon la revendication 2, caractérisé en ce que la fonction f est définie par $I(k+1) = I(k) + g(k)$.

Les figures 1a à 1c représentent une mémoire réinscriptible à différents stades d'inscription selon un premier procédé conforme à l'invention.

La figure 1d représente la mémoire réinscriptible des figures 1a et 1c inscrites selon une variante du premier procédé d'inscription.

Les figures 2a à 2d représentent une mémoire réinscriptible à différents stades d'inscription selon un deuxième procédé conforme à l'invention.

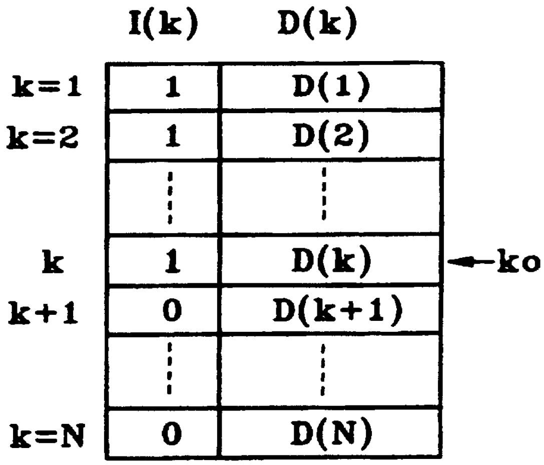
La figure 3 représente une mémoire réinscriptible analogue à celle des figures 1a à 1c, complétée par une information de contrôle de cohérence.

La mémoire réinscriptible montrée sur les figures 1a à 1c est destinée à recevoir N valeurs courantes successives $D(k)$ $(k=1, \ldots, N)$ d'une donnée D selon un procédé d'inscription qui consiste notamment à décomposer ladite mémoire en N zones k comprenant, chacune, un premier espace mémoire affecté à un indicateur $Z(k)$ et un deuxième espace-mémoire, contigu à la première, dans laquelle est inscrite une valeur courante $D(k)$, de la donnée D. Le rôle des indicateurs $I(k)$ est de pouvoir désigner à un moment donné celle desdites zones, notée ko, contenant la dernière valeur courante $D(ko)$ inscrite dans la mémoire, et, par voie de conséquence, à repérer la zone dans laquelle devra être inscrite une nouvelle valeur courante de la donnée D.

Comme on peut le voir sur la figure 1a qui décrit un stade quelconque d'inscription de la mémoire, les valeurs courantes $D(k)$ de la donnée D sont inscrites successivement, dans l'ordre, dans les différentes zones k correspondantes de la mémoire, c'est-à-dire la valeur $D(1)$ en zone $z=1$, la valeur $D(2)$ en zone $z=2$, etc... La zone ko contenant la dernière valeur courante D $(k=ko)$ inscrite est définie par une discontinuité dans la suite des indicateurs $I(k)$.

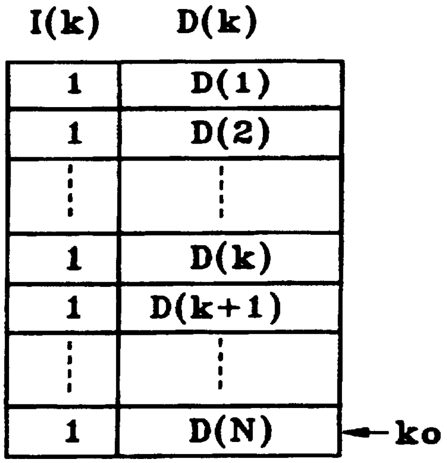
D'une manière générale, pour repérer ladite discontinuité, et donc la zone ko, on définit une relation $I(k+1) = f[I(k), k]$ qui doit être normalement vérifiée entre l'indicateur $I(k+1)$ et l'indicateur précédent $I(k)$, la discontinuité apparaissant précisément à la zone ko pour laquelle la relation précitée n'est pas vérifiée : $I(ko + 1) = f\,(I(ko), ko)$.

[redacted] [redacted] [redacted]



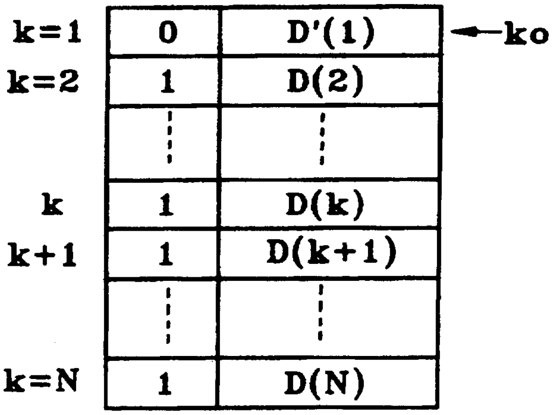
	I(k)	D(k)	
k=1	1	D(1)	
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	←ko
k+1	0	D(k+1)	
	⋮	⋮	
k=N	0	D(N)	

FIG. 1a



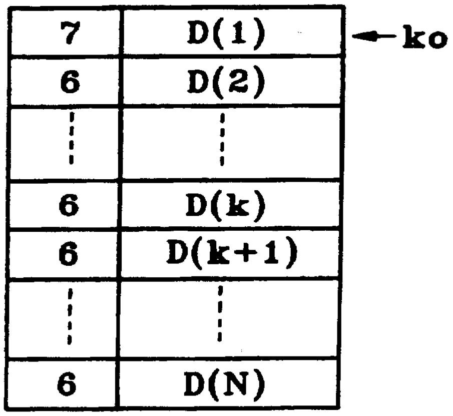
I(k)	D(k)	
1	D(1)	
1	D(2)	
⋮	⋮	
1	D(k)	
1	D(k+1)	
⋮	⋮	
1	D(N)	←ko

FIG. 1b



k=1	0	D'(1)	←ko
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	
k+1	1	D(k+1)	
	⋮	⋮	
k=N	1	D(N)	

FIG. 1c



7	D(1)	←ko
6	D(2)	
⋮	⋮	
6	D(k)	
6	D(k+1)	
⋮	⋮	
6	D(N)	

FIG. 1d

[redacted] [redacted]

puisse désigner la zone dans laquelle est inscrite la dernière valeur courante de la donnée D.

L'inconvénient de ce type de procédé est qu'il nécessite plusieurs opérations d'inscription qui peuvent, chacune, être le siège d'une corruption. Le logiciel gérant les inscriptions dans la mémoire doit donc être complexe et, de là, consommateur de temps et d'espace-mémoire.

Aussi, le problème technique à résoudre par l'objet de la présente invention est de proposer un procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celle desdites zones contenant la dernière valeur courante inscrite de la donnée D, procédé qui permettrait de simplifier la gestion des opérations d'inscription tant en assurant l'augmentation de la durée de vie des cellules de la mémoire ainsi que l'intégrité de la donnée D au moment de l'inscription de la dernière valeur courante.

La solution au problème technique posé consiste, selon la présente invention, ce que ladite mémoire est décomposée en N zones k (k= 1, ..., N), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur I (k) et un deuxième espace-mémoire affecté à une valeur courante D (k) de la donnée D, la zone ko contenant la dernière valeur courante D (ko) inscrite étant définie, soit par une discontinuité dans la suite des indicateurs I(k), soit par la dernière zone N (ko = N) en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko + 1 avec mise à jour de l'indicateur I (ko + 1).

Le procédé de l'invention présente plusieurs avantages.

En premier lieu, le gestionnaire de mémoire se trouve simplifié puisqu'après avoir inscrit dans un premier temps la dernière valeur courante de la donnée D et vérifié que cette inscription est correcte, il suffit de mettre à jour l'indicateur correspondant. Sinon, l'indicateur n'est pas mis à jour et la valeur courante précédente reste "active". Il

[redacted] [redacted] [redacted]

I(k)	D(k)	CRC(D(k))
1	D(1)	CRC(D(1))
1	D(2)	CRC(D(2))
0	D(3)	CRC(D(3))
	⋮	⋮
0	D(N)	CRC(D(N))

FIG. 3

[redacted] [redacted]

[redacted] [redacted] [redacted]

D14

(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 742 893**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **95 15186**

(51) **Int Cl⁶ : G 06 F 12/02, G 11 C 8/00, 16/02**

(12) **DEMANDE DE BREVET D'INVENTION** **A1**

(22) **Date de dépôt** : 20.12.95.

(30) **Priorité** :

(43) **Date de la mise à disposition du public de la demande :** 27.06.97 Bulletin 97/26.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Se reporter à la fin du présent fascicule.*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *SCHLUMBERGER INDUSTRIES SA SOCIETE ANONYME* — FR.

(72) **Inventeur(s) :** DETURCHE JEAN CLAUDE.

(73) **Titulaire(s) :**

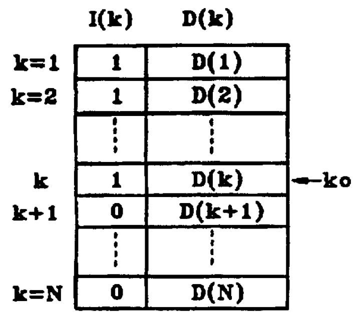
(74) **Mandataire :** SCHLUMBERGER INDUSTRIES.

(54) **PROCEDE D'INSCRIPTION D'UNE DONNEE DANS UNE MEMOIRE REINSCRIPTIBLE.**

(57) Procédé d'inscription de valeurs courantes successives d'une donnée D dans une mémoire réinscriptible, consistant à inscrire lesdites valeurs courantes successivement en différentes zones de la mémoire et à programmer des indicateurs aptes à désigner celles desdites zones contenant la dernière valeur courante inscrite de la donnée D.

Selon l'invention, ladite mémoire est décomposée en N zones k (k=1,..., N), chaque zone k comprenant un premier espace-mémoire affecté à un indicateur I(k) et un deuxième espace-mémoire affecté à une valeur courante D(k) de la donnée D, la zone ko contenant la dernière valeur courante D(ko) inscrite étant définie, soit par une discontinuité dans la suite des indicateurs I(k), soit par la dernière zone N(ko=N) en l'absence de discontinuité, et en ce qu'une nouvelle valeur courante de la donnée D est inscrite dans la zone suivante ko+1 avec mise à jour de l'indicateur I (ko + 1).

Application aux cartes à mémoire électronique telles que les cartes à mémoire simple, les cartes à microprocesseur et les cartes sans contact.



	I(k)	D(k)	
k=1	1	D(1)	
k=2	1	D(2)	
	⋮	⋮	
k	1	D(k)	← ko
k+1	0	D(k+1)	
	⋮	⋮	
k=N	0	D(N)	

FR 2 742 893 - A1

[redacted] [redacted]

D15

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G11C	A2	**(11) International Publication Number:** WO 99/07000 **(43) International Publication Date:** 11 February 1999 (11.02.99)

(21) International Application Number: PCT/IL98/00363

(22) International Filing Date: 2 August 1998 (02.08.98)

(30) Priority Data:
08/905,286 1 August 1997 (01.08.97) US

(71) Applicant *(for all designated States except US)*: SAIFUN SEMICONDUCTORS LTD. [IL/IL]; Bet HaSofer, Hamelacha Street 65, 42504 Industrial Area South Netanya (IL).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: EITAN, Boaz [IL/IL]; Achi Dakar Street 4, 43259 Ra'anana (IL).

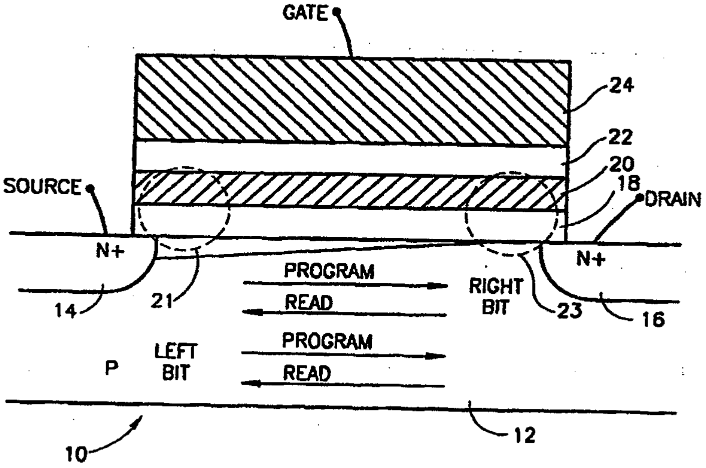
(74) Agent: EITAN, PEARL, LATZER & COHEN-ZEDEK; Lumir House, Maskit Street 22, 46733 Herzelia (IL).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: TWO BIT NON-VOLATILE ELECTRICALLY ERASABLE AND PROGRAMMABLE SEMICONDUCTOR MEMORY CELL UTILIZING ASYMMETRICAL CHARGE TRAPPING

(57) Abstract



A non-volatile electrically erasable programmable read only memory (EEPROM) capable of storing two bit of information having a nonconducting charge trapping dielectric, such as silicon nitride, sandwiched between two silicon dioxide layers acting as electrical insulators is disclosed. The invention includes a method of programming, reading and erasing the two bit EEPROM device. The nonconducting dielectric layer functions as an electrical charge trapping medium. A conducting gate layer is placed over the upper silicon dioxide layer. A left and a right bit are stored in physically different areas of the charge trapping layer, near left and right regions of the memory cell, respectively. Each bit of the memory device is programmed in the conventional manner, using hot electron programming, by applying programming voltages to the gate and to either the left or the right region while the other region is grounded. Hot electrons are accelerated sufficiently to be injected into the region of the trapping dielectric layer near where the programming voltages were applied to. The device, however, is read in the opposite direction from which it was written, meaning voltages are applied to the gate and to either the right or the left region while the other region is grounded. Two bits are able to be programmed and read due to a combination of relatively low gate voltages with reading in the reverse direction. This greatly reduces the potential across the trapped charge region. This permit much shorter programming times by amplifying the effect of the charge trapped in the localized trapping region associated with each of the bits. In addition, both bits of the memory cell can be individually erased by applying suitable erase voltages to the gate and either left or right regions so as to cause electrons to be removed from the corresponding charge trapping region of the nitride layer.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

TWO BIT NON-VOLATILE ELECTRICALLY ERASABLE AND PROGRAMMABLE SEMICONDUCTOR MEMORY CELL UTILIZING ASYMMETRICAL CHARGE TRAPPING

FIELD OF THE INVENTION

The present invention relates generally to semiconductor memory devices and more particularly to multi-bit flash electrically erasable programmable read only memory (EEPROM) cells that utilize the phenomena of hot electron injection to trap charge within a trapping dielectric material within the gate.

BACKGROUND OF THE INVENTION

Memory devices for non-volatile storage of information are currently in widespread use today, being used in a myriad of applications. A few examples of non-volatile semiconductor memory include read only memory (ROM), programmable read only memory (PROM), erasable programmable read only memory (EPROM), electrically erasable programmable read only memory (EEPROM) and flash EEPROM.

Semiconductor ROM devices, however, suffer from the disadvantage of not being electrically programmable memory devices. The programming of a ROM occurs during one of the steps of manufacture using special masks containing the data to be stored. Thus, the entire contents of a ROM must be determined before manufacture. In addition, because ROM devices are programmed during manufacture, the time delay before the finished product is available could be six weeks or more. The advantage, however, of using ROM for data storage is the low cost per device. However, the penalty is the inability to change the data once the masks are committed to. If mistakes in the data programming are found they are typically very costly to correct. Any inventory that exists having incorrect data programming is instantly obsolete and probably cannot be used. In addition, extensive time delays are incurred because new masks must first be generated from scratch and the entire manufacturing process repeated. Also, the cost savings in the use of ROM memories only exist if large quantities of the ROM are produced.

Moving to EPROM semiconductor devices eliminates the necessity of mask programming the data but the complexity of the process increases drastically. In addition, the die size is larger due to the addition of programming circuitry and there are more processing and testing steps involved in the manufacture of these types of memory devices. An advantage of EPROMs are that they are electrically programmed, but for erasing, EPROMs require exposure to ultraviolet (UV) light. These devices are constructed with windows transparent to UV light to allow the die to be exposed for erasing, which must be performed before the device can be programmed. A major drawback to these devices is that they lack the ability to be electrically erased. In many circuit designs it is desirable to have a non-volatile memory device that can be erased and reprogrammed in-circuit, without the need to remove the device for erasing and reprogramming.

Semiconductor EEPROM devices also involve more complex processing and testing procedures than ROM, but have the advantage of electrical programming and erasing. Using EEPROM devices in circuitry permits in-circuit erasing and reprogramming of the device, a feat not possible with conventional EPROM memory. Flash EEPROMs are similar to EEPROMs in that memory cells can be programmed (i.e., written) and erased electrically but with the additional ability of erasing all memory cells at once, hence the term flash EEPROM. The disadvantage of flash EEPROM is that it is very difficult and expensive to manufacture and produce.

The widespread use of EEPROM semiconductor memory has prompted much research focusing on constructing better memory cells. Active areas of research have focused on developing a memory cell that has improved performance characteristics such as shorter programming times, utilizing lower voltages for programming and reading, longer data retention times, shorter erase times and smaller physical dimensions. One such area of research involves a memory cell that has an insulated gate. The following prior art reference is related to this area.

U.S. Patent No. 4,173,766, issued to Hayes, teaches a metal nitride oxide semiconductor (MNOS) constructed with an insulated gate having a bottom silicon dioxide layer and a top nitride layer. A conductive gate electrode, such as polycrystalline silicon or metal, is placed on top of the nitride layer. A major disadvantage of this device is the difficulty in using it to construct a flash EEPROM. A consequence of using an oxide-nitride structure as opposed to an oxide-nitride-oxide structure is that during programming the charge gets distributed across the entire nitride layer. The absence of the top oxide layer lowers the ability to control where the charge is stored in the nitride layer.

Further, in the memory cell disclosed in Hayes, the nitride layer is typically 350 Angstroms thick. A thick nitride layer is required in Hayes' device in order to achieve sufficient charge retention. Since the nitride can only tolerate relatively small internal electric fields, a thick layer of nitride is required to compensate. Due to the thick nitride layer, very high vertical voltages are needed for erasing. The relatively thick nitride layer causes the distribution of charge, i.e., the charge trapping region, to be very wide and a wider charge trapping region makes erasing the cell via the drain extremely difficult if not impossible. In addition, drain erasing is made difficult because of the increased thickness of the charge trapping layer. Thus, the memory cell taught by Hayes must have a thick nitride layer for charge retention purposes but at the expense of making it extremely difficult to erase the device via the drain, thus making the device impractical for flash EEPROM applications.

To erase the memory cell of Hayes, the electrons previously trapped in the nitride must be neutralized either by moving electrons out of the nitride or by transferring holes into the nitride. Hayes teaches an erase mode for his memory cell whereby the information stored on the nitride is erased by grounding the gate and applying a sufficient potential to the drain to cause avalanche breakdown. Avalanche breakdown involves hot hole injection into the nitride in contrast to electron injection. Avalanche breakdown, however, requires relatively high voltages and high currents for the phenomena to occur. To lower the avalanche breakdown voltage, a heavily doped impurity is implanted into the channel between the source and the drain.

The hot holes are generated and caused to surmount the hole potential barrier of the bottom oxide and recombine with the electrons in the nitride. This mechanism, however, is very complex and it is difficult to construct memory devices that work in this manner. Another disadvantage of using hot hole injection for erase is that since the PN junction between the drain and the channel is in breakdown, very large currents are generated that are difficult to control. Further, the number of program/erase cycles that the memory cell can sustain is limited because the breakdown damages the junction area. The damage is caused by the very high local temperatures generated in the vicinity of the junction when it is in breakdown.

In addition, it is impractical to use the memory device of Hayes in a flash memory array architecture. The huge currents generated during erase using avalanche breakdown would cause significant voltage (i.e., IR), drops along the bit line associated with the memory cell in breakdown.

Another well known technique of erasing is to inject holes from the gate into the nitride layer. This mechanism, however, is very complex and difficult to control due to the higher mobility of holes versus electrons in the nitride. With elevated temperatures, the higher mobility of holes causes a large loss of charge retention and consequently lower threshold voltage deltas from the original programming threshold. Deep depletion phenomena create the need for a companion serial device to control the programming/erase process.

U.S. Patent No. 5,168,334, issued to Mitchell et al., teaches a single transistor EEPROM memory cell. Mitchell, however, teaches an oxide-nitride-oxide (ONO) EEPROM memory cell wherein oxide-nitride-oxide layers are formed above the channel area and between the bit lines for providing isolation between overlying polysilicon word lines. The nitride layer provides the charge retention mechanism for programming the memory cell.

Although the memory device of Mitchell includes a top oxide layer, it is not very well suited for flash EEPROM applications. This is due to the very wide charge trapping region that must be programmed in order to achieve a sufficient delta in the threshold voltage between programming and reading. The Mitchell device is programmed and read in the forward direction. Since reading in the forward direction is less effective than reading in the reverse direction, the charge trapping region must be wider by default in order to distinguish between the programmed and unprogrammed states. A wider charge trapping region, however, makes the memory device very difficult to erase, thus making this device inefficient for flash EEPROM applications.

A single transistor ONO EEPROM device is disclosed in the technical article entitled "A True Single-Transistor Oxide-Nitride-Oxide EEPROM Device," T.Y. Chan, K.K. Young and Chenming Hu, IEEE Electron Device Letters, March 1987. The memory cell is programmed by hot electron injection and the injected charges are stored in the oxide-nitride-oxide (ONO) layer of the device. This article teaches programming and reading in the forward direction. Thus, as in Mitchell, a wider charge trapping region is required to achieve a sufficiently large difference in threshold voltages between programming and reading. This, however, makes it much more difficult to erase the device.

Multi-bit transistors are known in the art. Most multi-bit transistors utilize multi-level thresholds to store more than one bit with each threshold level representing a different state. A memory cell having four threshold levels can store two bits. This technique has been implemented in a ROM by using implanting techniques and has also been attempted in FLASH and EEPROM memory. The multi-level threshold technique has not been applied to EPROM memory due to the fact that if the window of a threshold defining a given state is exceeded, a UV erase cycle must be performed which is very cumbersome and costly. In addition, to perform the UV erase, the chip must first be removed from the system which can be very problematic.

Achieving multiple thresholds in FLASH and EEPROM requires an initial erase cycle to bring all the memory cells below a certain threshold. Then, utilizing a methodical programming scheme, the threshold of each cell is increased until the desired threshold is reached. A disadvantage with this technique is that the programming process requires constant feedback which causes multi-level programming to be slow.

In addition, using this technique causes the window of operation to decrease meaning the margins for each state are reduced. This translates to a lower probability of making good dies and a reduction in the level of quality achieved. If it is not desired to sacrifice any margins while increasing the reliability of the cell, than the window of operation must be increased by a factor of two. This means operating at much higher voltages which is not desirable because it lowers the reliability and increases the disturbances between the cells. Due to the complexity of the multi-threshold technique, it is used mainly in applications where missing bits can be tolerated such as in audio applications.

Another problem with this technique is that the threshold windows for each state may change over time reducing the reliability. It must be guaranteed that using the same word line or bit line to program other cells will not interfere with or disturb the data in cells already programmed. In addition, the programming time itself increases to accommodate the multitude of different programming thresholds. Thus, the shifting of threshold windows for each state over time reduces the window of operation and consequently increases the sensitivity to disturbs.

The reduced margins for the threshold windows for the multiple states results in reduced yield. Further, in order to maintain quality and threshold margins, higher voltages are required. This implies higher electric fields in the channel which contributes to lower reliability of the memory cell.

In order to construct a multi-bit ROM memory cell, the cell must have four distinct levels that can be programmed. In the case of two levels, i.e., conventional single bit ROM cell, the threshold voltage programmed into a cell for a '0' bit only has to be greater than the maximum gate voltage, thus making sure the cell does not conduct when it is turned on during reading. It is sufficient that the cell conducts at least a certain amount of current to distinguish between the programmed and unprogrammed states. The current through a transistor can be described by the following equation.

$$I = \frac{1}{Leff} K(V_G - V_T)$$

L_{eff} is the effective channel length, K is a constant, V_G is the gate voltage and V_T is the threshold voltage. However, in the multi-bit case, different thresholds must be clearly distinguishable which translates into sensing different read currents and slower read speed. Further, for two bits, four current levels must be sensed, each threshold having a statistical distribution because the thresholds cannot be set perfectly. In addition, there will be a statistical distribution for the effective channel length which will further widen the distribution of the read currents for each threshold level.

The gate voltage also affects the distribution of read currents. For the same set of threshold levels, varying the gate voltage directly results in a variation of the ratio between the read currents. Therefore the gate voltage must be kept very stable. In addition, since there are multiple levels of current, sensing becomes more complex than in the two level, i.e., single bit, cell.

The following prior art references are related to multi-bit semiconductor memory cells.

U.S. Patent No. 5,021,999, issued to Kohda et al., teaches a non-volatile memory cell using an MOS transistor having a floating gate with two electrically separated segmented floating gates. The memory cell can store three levels of data: no electrons on either segment, electrons injected into either one of the two segments and electrons injected into both segments.

U.S. Patent No. 5,214,303, issued to Aoki, teaches a two bit transistor which comprises a semiconductor substrate, a gate electrode formed on the substrate, a pair of source/drain regions provided in the substrate and an offset step portion formed in at least one of the source/drain regions and downwardly extending into the substrate in the vicinity of the gate electrode.

U.S. Patent No. 5,394,355, issued to Uramoto et al., teaches a ROM memory having a plurality of reference potential transmission lines. Each reference potential transmission line represents a different level or state. Each memory cell includes a memory cell transistor able to connect one the reference potential transmission lines to the corresponding bit line.

U.S. Patent No. 5,414,693, issued to Ma et al., teaches a two bit split gate flash EEPROM memory cell structure that uses one select gate transistor and two floating gate transistors. In this invention essentially each bit is stored in a separate transistor.

U.S. Patent No. 5,434,825, issued to Harari, teaches a multi-bit EPROM and EEPROM memory cell which is partitioned into three or more ranges of programming charge. The cell's memory window is widened to store more than two binary states. Each cell is programmed to have one of the programmed states. To achieve more than two binary states, multiple negative and positive threshold voltages are used. The cell basically comprises a data storage transistor coupled to a series pass transistor. The data transistor is programmed to one of the predefined threshold states. Sensing circuitry distinguishes the different current levels associated with each programmed state.

SUMMARY OF THE INVENTION

The present invention discloses an apparatus for and method of programming, reading and erasing a two bit flash electrically erasable programmable read only memory (EEPROM). The two bit flash EEPROM memory cell is constructed having a charge trapping non-conducting dielectric layer sandwiched between two silicon dioxide layers. The nonconducting dielectric layer functions as an electrical charge trapping medium. The charge trapping layer is sandwiched between two layers of silicon dioxide which act as electrical insulators. A conducting gate layer is placed over the upper silicon dioxide layer. The two individual bits, i.e., left and right bits, are stored in physically different areas of the charge trapping region.

A novel aspect of the memory device is that while both bits are programmed in the conventional manner, using hot electron programming, each bit is read in a direction opposite that in which it was programmed with a relatively low gate voltage. For example, the right bit is programmed conventionally by applying programming voltages to the gate and the drain while the source is grounded. Hot electrons are accelerated sufficiently to be injected into a region of the trapping dielectric layer near the drain. The device, however, is read in the opposite direction from which it was written, meaning voltages are applied to the gate and the source while the drain is grounded. The left bit is similarly programmed and read by swapping the functionality of source and drain terminals. Programming one of the bits leaves the other bit with its information intact and undisturbed. Programming one of the bits does, however, have a very small effect on the other bit, e.g., slightly slower programming speed for the second bit.

Reading in the reverse direction is most effective when relatively low gate voltages are used. A benefit of utilizing relatively low gate voltages in combination with reading in the reverse direction is that the potential drop across the portion of the channel beneath the trapped charge region is significantly reduced. A relatively small programming region or charge trapping region is possible due to the lower channel potential drop under the charge trapping region. This permits much faster programming times because the effect of the charge trapped in the localized trapping region is amplified. Programming times are reduced while the delta in threshold voltage between the programmed versus unprogrammed states remains the same as when the device is read in the forward direction.

Another major benefit is that the erase mechanism of the memory cell is greatly enhanced. Both bits of the memory cell can be erased by applying suitable erase voltages to the gate and the drain for the right bit and to the gate and the source for the left bit so as to cause electrons to be removed from the charge trapping region of the nitride layer. Electrons move from the nitride through the bottom oxide layer to the drain or the source for the right and the left bits, respectively. Another benefit includes reduced wearout from cycling thus increasing device longevity. An effect of reading in the reverse direction is that a much higher threshold voltage for the same amount of programming is possible. Thus, to achieve a sufficient delta in the threshold voltage between the programmed and unprogrammed states of the memory cell, a much smaller region of trapped charge is required when the cell is read in the reverse direction than when the cell is read in the forward direction.

The erase mechanism is enhanced when the charge trapping region is made as narrow as possible. Programming in the forward direction and reading in the reverse direction permits limiting the width of the charge trapping region to a narrow region near the drain (right bit) or the source. This allows for much more efficient erasing of the memory cell.

Further, utilizing a thinner silicon nitride charge trapping layer than that disclosed in the prior art helps to confine the charge trapping region to a laterally narrower region near the drain. Further, the thinner top and bottom oxide sandwiching the nitride layer helps in retention of the trapped charge.

In addition, unlike prior art floating gate flash EEPROM memory cells, the bottom and top oxide thickness can be scaled due to the deep trapping levels that function to increase the potential barrier for direct tunneling. Since the electron trapping levels are so deep, thinner bottom and top oxides can be used without compromising charge retention.

Another benefit of localized charge trapping is that during erase, the region of the nitride away from the drain does not experience deep depletion since the erase occurs near the drain only. The final threshold of the cell after erasing is self limited by the device structure itself. This is in direct contrast to conventional singe transistor floating gate flash memory cells which are plagued with deep depletion problems. To overcome these problems, manufacturers include complex circuitry to control the erase process in order to prevent or recover from deep depletion.

Another approach previously employed in the prior art to solve the deep depletion problem was to design the floating gate flash memory cell using a split gate design forming multiple transistors per cell. The split gate or double transistor constructions were necessary because the information carrying transistor, i.e., the floating gate transistor, potentially could be over erased. An over erase condition caused the threshold voltage of the cell to go too low. The second transistor, acting as a control transistor, prevented the floating gate transistor from being over erased.

Two bit considerations of the memory cell of the present invention are described hereinbelow. The first is the fact that reading in the reverse direction permits read through of the trapping region associated with the other bit. The second is that programming the device to a low V_T, by clamping the word line voltage V_{WL}, further enhances the margin for each bit. The margin is defined as the parameters that will program one of the bits without affecting the other.

Further, the locality of the trapped charge due to hot electron injection in combination with reverse reading permits two distinct charge trapping regions to be formed in a relatively short device whose L_{eff} is approximately 0.2 microns. Also, utilizing a combination of positive V_D and either zero or negative V_G permits each bit to be erased separately.

The memory device also exhibits little or no disturb during programming. This is because during programming the drain voltage is only applied to the junction adjacent to the region where charge trapping is to occur.

It is important to note that a memory cell constructed in accordance with the present invention cannot store two bits utilizing programming in the forward direction and reading in the forward direction. This is due to the forward read requiring a wider charge trapping region to be programmed in order to achieve a sufficient delta in read currents for a one and a zero. Once one of the bits is programmed, the wider charge trapping region prevents read through to the other bit.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

Fig. 1 illustrates a sectional view of a single bit flash EEPROM cell of the prior art utilizing Oxide-Nitride-Oxide (ONO) as the gate dielectric;

Fig. 2 illustrates a sectional view of a two bit flash EEPROM cell constructed in accordance with an embodiment of the present invention utilizing ONO as the gate dielectric;

Fig. 3 illustrates a sectional view of a two bit flash EEPROM cell constructed in accordance with an embodiment of the present invention utilizing a silicon rich silicon dioxide with buried polysilicon islands as the gate dielectric;

Fig. 4 is a graph illustrating the threshold voltage as a function of programming time for reading in the forward and reverse directions of a selected memory cell in accordance with this invention;

Fig. 5A illustrates a sectional view of a flash EEPROM cell of the prior art showing the area of charge trapping under the gate;

Fig. 5B illustrates a sectional view of a flash EEPROM cell constructed in accordance with an embodiment of the present invention showing the area of charge trapping under the gate;

Fig. 6 is a graph illustrating the difference in threshold voltage in the forward and reverse directions as a function of drain voltage for a flash EEPROM cell of the present invention that has been programmed;

Fig. 7 is a graph illustrating the difference in drain current in the forward and reverse directions as a function of drain voltage for a flash EEPROM cell of the present invention that has been programmed;

Fig. 8 is a graph illustrating the threshold voltage of a flash EEPROM cell of the present invention as a function of programming time for reading in the forward and reverse directions;

Fig. 9 is a graph illustrating the leakage current through the region of trapped charge as a function of the voltage across the charge trapping region while reading in the reverse direction;

Fig. 10 is a graph illustrating the gate voltage required to sustain a given voltage in the channel beneath the edge of the region of trapped charge while reading in the reverse direction;

Fig. 11 is a graph illustrating the effect of the gate voltage applied during reading on the difference in drain current between reading in the forward versus the reverse direction;

Fig. 12 is a graph illustrating the effect of the gate voltage (as measured by threshold channel current I_{TH})on the difference in threshold voltage between the forward read and reverse read directions;

Fig. 13 is a graph illustrating the effect programming one of the bits has on the other bit that has not been previously programmed;

Fig. 14 is a graph illustrating the effect programming one of the bits has on the other bit that has been previously programmed;

Fig. 15 is a sectional view of a two bit EEPROM cell constructed in accordance with an embodiment of the present invention showing the area of charge trapping under the gate for both the right and the left bits;

Fig. 16 is a graph illustrating the effect of a low drain voltage on the read through of a programmed bit;

Fig. 17 is a graph illustrating the effect of programming on erase for the forward and reverse directions;

Fig. 18 is a graph illustrating the separate bit erase capability of the two bit EEPROM memory cell of the present invention;

Fig. 19 is a graph illustrating the effect of cycling on the program and erase ability of the two bit EEPROM cell of the present invention;

Fig. 20 is a graph illustrating the effect of over programming on the ability to erase for the forward and reverse directions;

Fig. 21 is a graph illustrating the programming and erasing curves for using oxide versus TEOS as the material used as the top oxide;

Fig. 22 is a graph illustrating the erase curves for two different values of drain voltage while the gate is held at ground potential;

Fig. 23 is a graph illustrating the erase curve for two different values of gate voltage;

Fig. 24A illustrates a sectional view of a flash EEPROM cell of the prior art showing the area of charge trapping under the gate after being programmed for a period of time; and

Fig. 24B illustrates a sectional view of a flash EEPROM cell constructed in accordance with an embodiment of the present invention showing the area of charge trapping under the gate after being programmed for a sufficient time to achieve the same threshold voltage of the cell illustrated in Figure 24A.

DETAILED DESCRIPTION OF THE INVENTION

The two bit flash EEPROM cell of the present invention can best be understood with an understanding of how present day single bit charge trapping dielectric flash EEPROM memory cells are constructed, programmed and read. Thus, prior art single bit ONO EEPROM memory cells and the conventional method used to program, read and erase them are described in some detail. Illustrated in Figure 1 is a cross section of a conventional ONO EEPROM memory cell as disclosed in the technical article entitled "A True Single-Transistor Oxide-Nitride-Oxide EEPROM Device," T.Y. Chan, K.K. Young and Chenming Hu, IEEE Electron Device Letters, March 1987, incorporated herein by reference. The memory cell, generally referenced 41, comprises a P-type silicon substrate 30, two PN junctions between N+ source and drain regions 32, 34 and P type substrate 30, a non conducting nitride layer 38 sandwiched between two oxide layers 36, 40 and a polycrystalline conducting layer 42.

Programming Prior Art Single Bit Memory Devices

The operation of the prior art memory cell 41 will now be described. To program or write the cell, voltages are applied to the drain 34 and the gate 42 and the source 32 is grounded. For example, 10V is applied to the gate and 9V is applied to the drain. These voltages generate a vertical and lateral electric field along the length of the channel from the source to the drain. This electric field causes electrons to be drawn off the source and begin accelerating towards the drain. As they move along the length of the channel, they gain energy. If they gain enough energy they are able to jump over the potential barrier of the oxide layer 36 into the silicon nitride layer 38 and become trapped. The probability of this occurring is a maximum in the region of the gate next to the drain

34 because it is near the drain that the electrons gain the most energy. These accelerated electrons are termed hot electrons and once injected into the nitride layer they become trapped and remain stored there. The trapped electrons cannot spread through the nitride layer because of the low conductivity of the nitride layer and the low lateral electric field in the nitride. Thus, the trapped charge remains in a localized trapping region in the nitride typically located close to the drain.

In U.S. Patent No. 4,173,766, issued to Hayes, the nitride layer is described as typically being about 350 Angstroms thick (see column 6, lines 59 to 61). Further, the nitride layer in Hayes has no top oxide layer. A top oxide layer would serve as a low conductivity layer to prevent holes from moving into the nitride from the overlying gate and combining with electrons trapped in the nitride which reduces the charge stored in the nitride. If the memory cell of Hayes used a thinner nitride layer, then electrons trapped in the nitride layer would be lost to holes entering from the overlying conductive gate. The conductive gate permits the electrons in the nitride to be removed. Further, once the electrons are trapped in a given region of the nitride associated with a single cell, the programming of adjacent cells can cause an electric field to be generated with respect to the electrons in the trapped region of the single cell causing further dissipation of the electrons from the trapped region. During life testing, where the device is subjected to elevated temperatures typically in the range from about 150 degrees Centigrade to 250 degrees Centigrade, holes from the gate can enter the nitride and combine with the electrons to further reduce the amount of charge trapped in the nitride. Although lateral fields exist in the nitride of the Hayes structure as they do in any ONO structure used as gate insulation in an MOS device, the relatively thick nitride layer such as disclosed by

Hayes causes the electrons to move laterally in response to this lateral field and come to rest either in traps between the conduction and valence bands or in localized regions of positive charge in the nitride layer. Such movement of electrons, commonly known as electron hopping, can readily occur in a relatively thick nitride layer such as disclosed by Hayes. Such hopping diffuses and thus reduces the localized intensity of the trapped charge.

As previously described, in order to achieve an effective delta in threshold voltage between the unprogrammed and the programmed state of each cell, the charge trapping region of prior art flash EEPROM cells must be made fairly wide. Thus, electrons are trapped in areas far from the drain which directly affects the effectiveness of the erase. In some cases, the device cannot be erased at all because the charge trapping region was programmed too wide.

In memory cells constructed using a conductive floating gate, the charge that gets injected into the gate is distributed equally across the entire gate. The threshold voltage of the entire gate increases as more and more charge is injected into the gate. The threshold voltage increases because the electrons that become stored in the gate screen the gate voltage from the channel.

With reference to Figure 1, in devices with low conductivity or non conductive floating gates, the injection of hot electrons into the silicon nitride layer causes the gate threshold voltage to increase only in the localized trapping region. This is in contrast to the conductive floating gate memory cells of EPROMs and EEPROMs wherein the gate threshold voltage of the entire channel rises as programming time increases. In both conductive and non conductive floating gate memory cell designs, an increase in the gate threshold voltage causes the current flowing through the channel to decrease for a given gate voltage. This reduces programming efficiency by lengthening the programming time. However, due to the localized electron trapping in the non conductive floating gate memory cell design, the programming time is reduced less than with the conductive floating gate memory cell design. The technique of programming flash EEPROM memory cells with either conductive or low conductivity or non conductive floating gates is well known in the art and is currently used to program EEPROM and flash EEPROM memory cells.

Reading Prior Art Single Bit Memory Devices

The method of reading prior art flash EEPROM memory cells will now be described. The conventional technique of reading both prior art conductive floating gate and non conductive localized trapping gate EEPROM or flash EEPROM memory is to apply read voltages to the gate and drain and to ground the source. This is similar to the method of programming with the difference being that lower level voltages are applied during reading than during programming. Since the floating gate is conductive, the trapped charge is distributed evenly throughout the entire floating conductor. In a programmed device, the threshold is therefore high for the entire channel and the process of reading becomes symmetrical. It makes no difference whether voltage is applied to the drain and the source is grounded or vice versa. A similar process is also used to read prior art non conductive localized gate flash EEPROM devices.

The process of programming typically includes writing followed by reading. This is true for all EPROM and EEPROM memory devices. A short programming pulse is applied to the device followed by a read. The read is actually used to effectively measure the gate threshold voltage. By convention, the gate threshold voltage is measured by applying a voltage to the drain and a separate voltage to the gate, with the voltage on the gate being increased from zero while the channel current flowing from drain to source is measured. The gate voltage that provides 1 p.A of channel current is termed the threshold voltage.

Typically, programming pulses (i.e., write pulses) are followed by read cycles wherein the read is performed in the same direction that the programming pulse is applied. This is termed symmetrical programming and reading. Programming stops when the gate threshold voltage has reached a certain predetermined point (i.e., the channel current is reduced to a sufficiently low level). This point is chosen to ensure that a '0' bit can be distinguished from a ' 1' bit and that a certain data retention time has been achieved.

The Two Bit Memory Device of the Present Invention

A sectional view of a two bit flash EEPROM cell constructed in accordance with an embodiment of the present invention utilizing ONO as the gate dielectric is shown in Figure 2. The flash EEPROM memory cell, generally referenced 10, comprises a P-type substrate 12 having two buried PN junctions, one being between the source 14 and substrate 12, termed the left junction and the other being between the drain 16 and the substrate 12, termed the right junction. Above the channel is a layer of silicon dioxide 18, preferably between approximately 60 to 100 Angstroms thick, which forms an electrical isolation layer over the channel. On top of the silicon dioxide layer 18 is a charge trapping layer 20 constructed preferably in the range of 20 to 100 Angstroms thick and preferably comprised of silicon nitride, Si_3N_4. The hot electrons are trapped as they are injected into the charge trapping layer. In this fashion, the charge trapping layer serves as the memory retention layer. Note that the programming, reading and erasing of the memory cell of the present invention is based on the movement of electrons as opposed to movement of holes. The charge trapping dielectric can be constructed using silicon nitride, silicon dioxide with buried polysilicon islands or implanted oxide, for example. In the third listed alternative, the oxide can be implanted with arsenic, for example. The thickness of layer 18 is chosen to be in excess of 50 angstrom to prevent electrons from tunneling through the oxide and leaving charge trapping layer 20 during the operation of the cell. Thus the lifetime of the cell of this invention is greatly extended relative to prior art MNOS devices. The memory cell 10 is capable of storing two bits of data, a right bit represented by the dashed circle 23 and a left bit represented by the dashed circle 21.

It is important to note that the two bit memory cell of the present invention is a symmetrical device. Therefore, the terms source and drain as used with conventional one bit devices may be confusing. In reality, the left junction serves as the source terminal and the right junction serves as the drain terminal for the right bit. Similarly, for the left bit, the right junction serves as the source terminal and the left junction serves as the drain terminal. Thus, to avoid confusion, the terms left or first junction and right or second junction are utilized most of the time rather than source and drain. When the distinction between left and right bits is not crucial to the particular discussion, the terms source and drain are utilized. However, it should be understood that the source and drain terminals for the second bit are reversed compared to the source and drain terminals for the first bit.

Another layer of silicon dioxide 22 is formed over the charge trapping layer, (i.e., silicon nitride layer), and is preferably between approximately 60 to 100 Angstroms thick. The silicon dioxide layer 22 functions to electrically isolate a conductive gate 24 formed over the silicon dioxide layer 22 from charge trapping layer 20. The thickness of gate 24 is approximately 4,000 Angstroms. Gate 24 can be constructed from polycrystalline silicon, commonly known as polysilicon.

Charge trapping dielectric materials other than nitride may also be suitable for use as the asymmetric charge trapping medium. One such material is silicon dioxide with buried polysilicon islands. The silicon dioxide with polysilicon islands is sandwiched between two layers of oxide in similar fashion to the construction of the ONO memory cell in Figure 2. A sectional view of a two bit flash EEPROM cell constructed in accordance with a preferred embodiment of the present invention utilizing a silicon rich silicon dioxide layer 54 with buried polysilicon islands 57 as the gate dielectric is illustrated in Figure 3. Note that for simplicity, only a few polysilicon islands are numbered. A P-type substrate 62 has buried N+ source 58 and N+ drain 60 regions. The silicon dioxide 54 with buried polysilicon islands 57 is sandwiched between two layers of silicon dioxide 52, 56. Covering oxide layer 52 is polysilicon gate 50. Gate 50 is typically heavily doped with an N-type impurity such as phosphorus in the 10^{19} to 10^{20} atom/cc range. Similar to the two bit memory cell of Figure 2, the memory cell of Figure 3 is capable of storing two data bits, a right bit represented by the dashed circle 55 and a left bit represented by the dashed circle 53. The operation of the memory cell of Figure 3 is similar to that of the memory cell illustrated in Figure 2 with programming and reading occurring in opposite directions for each bit.

Alternatively, the charge trapping dielectric can be constructed by implanting an impurity, such as arsenic, into a middle layer 54 of silicon dioxide deposited on top of the bottom oxide 56.

A key aspect of the present invention lies in the manner in which the flash EEPROM memory cell 10 (Figure 2) is programmed and read. Rather than performing symmetrical programming and reading, the flash EEPROM memory cell of the present invention is programmed and read asymmetrically. This means that programming and reading occur in opposite directions. The arrows labeled PROGRAM and READ for each bit (i.e. the left bit and the right bit) in Figure 2 point in opposite directions to signify this asymmetry. Thus, programming is performed in what is termed the forward direction and reading is performed in what is termed the opposite or reverse direction.

It is noted that throughout the discussion of the EEPROM memory cell of the present invention presented below, the voltage levels discussed in connection therewith are assumed to be independent of the power supply voltage. Thus, the power supply voltages supplied to the chip embodying the EEPROM memory device may vary while the voltages applied to the gate, drain and source thereof will be supplied from regulated voltage sources.

Programming One Bit in the Forward Direction

As previously mentioned, the flash EEPROM memory cell 10 of Figure 2 is programmed similarly to the prior art flash EEPROM memory cell of Figure 1. Voltages are applied to the gate 24 and drain 16 creating vertical and lateral electrical fields which accelerate electrons from the source 14 along the length of the channel. As the electrons move along the channel some of them gain sufficient energy to jump over the potential barrier of the bottom silicon dioxide layer 18 and become trapped in the silicon nitride layer 20. For the right bit, for example, the electron trapping occurs in a region near the drain 16 indicated by the dashed circle 23 in Figure 2. Thus the trapped charge is self-aligned to the junction between the drain 16 and the substrate. Electrons are trapped in the portion of nitride layer 20 near but above and self-aligned with the drain region 16 because the electric fields are the strongest there. Thus, the electrons have a maximum probability of being sufficiently energized to jump the potential barrier of the silicon dioxide layer 18 and become trapped in the nitride layer 20 near the drain 16. The threshold voltage of the portion of the channel between the source 14 and drain 16 under the region of trapped charge increases as more electrons are injected into the nitride layer 20.

It is important to note that in order to be able to subsequently erase memory device 10 effectively, the programming time period must be limited. As the device continues to be programmed, the width of the charge trapping region increases. If programming continues past a certain point the charge trapping region becomes too wide whereby erasing is ineffective in removing trapped charge from the nitride layer 20.

However, by reading in the reverse direction, programming times can be shortened. This permits a much narrower charge trapping region. This in turn greatly increases the erase efficiency since fewer electrons need to be removed to erase the device. In addition, the trapped electrons are stored in a narrower region near the drain also improving the effectiveness of the erase.

Reading One Bit in the Forward Direction

If the flash EEPROM memory cell 10 is read using the conventional technique of reading in the same direction as programming, the time needed to program the device greatly increases to achieve the same threshold voltage. Reading in the same direction as programming means the device is programmed and read in the same forward direction.

During reading, voltages having levels lower than the voltages applied during programming are applied to the gate and drain and the channel current are sensed. If device 10 is programmed (i.e., a logic '0') the channel current should be very low and if the device is not programmed (i.e., a logic '1') there should be significant channel current generated. Preferably, the difference in the channel current between the '0' and '1' logic states should be maximized in order to better distinguish between the '0' and '1' logic states.

Illustrated in Figure 4 is a graph showing the rise in gate threshold voltage as a function of programming time for reading in the forward direction (curve labeled FORWARD READ) and for reading in the reverse direction (curve labeled REVERSE READ). Apparent from the graph in Figure 4 is the several orders of magnitude reduction in programming time achieved when reading in the reverse direction versus reading in the forward direction. As is described in more detail below, this dramatic reduction in programming time is due to amplification of the effect of the trapped charge injected into the nitride layer brought about by reading the memory cell in the opposite direction from which it was programmed.

As stated above, the time needed to program the flash EEPROM memory cell greatly increases when reading occurs in the same direction (i.e., the forward direction) as programming. The reason for this will now be explained in more detail with reference to Figures 5A and 5B. Figure 5A illustrates a sectional view of a flash EEPROM cell of the prior art showing the area 66 of charge trapping under the gate 42. Figure 5B illustrates a sectional view of a flash EEPROM cell constructed in accordance with an embodiment of the present invention showing the area 68 of charge trapping under the gate 24 for the right bit.

A description of what occurs during programming is presented first followed by what occurs during reading. Note that the description that follows also pertains to the memory cell of Figure 3 comprising the silicon dioxide layer 54 having buried polysilicon islands 57 substituting for the nitride layer 20 of Figure 2. During programming, hot electrons are injected into the nitride layer 20, as described above. Since the nitride 20 is a nonconductor, the trapped charge remains localized to the region near the drain 34 (Figure 5A) or 16 (Figure 5B). The region of trapped charge is indicated by the cross hatched area 66 in Figure 5A and by the cross hatched area 68 in Figure 5B. Thus, the threshold voltage rises, for example, to approximately 4 V, only in the portion of the channel under the trapped charge. The threshold voltage of the remainder of the channel under the gate remains at, for example, approximately 1 V. If the device is now read in the conventional forward direction (i.e., voltages are applied to the gate and drain as indicated by the arrow in Figure 5A), electrons move off the source and begin traveling toward the drain. When a logic '0' is programmed, there can be little or no channel current through the device when it is read. Thus, only if a sufficient portion of the channel is turned off, can the electron current be stopped. If the channel cannot be completely turned off, the electrons will reach the drain. Whether the electrons reach the drain will be determined by, among other things, the length of the trapping area. If the memory cell is programmed for a sufficiently long period, eventually, the channel stops conducting when read in the forward direction. If the trapped charge region (the programmed area) 66 (Figure 5A) is not long enough, electrons can punch through to the drain 34 in the depletion region under the trapped charge 66.

When the device is read in the forward direction, a voltage is applied to the drain and the gate, for example 2V and 3V, respectively, and the source is grounded. Full inversion occurs in the channel under the area of the nitride 38 that does not have trapped charge. A vertical electric field exists in the channel that spans the length of the channel up to the region of the channel underneath the trapped charge 66. In the inversion region, electrons travel in a linear fashion up to the edge 35 of the inversion region which is beneath the left edge 35 of the trapped charge region 66. This is indicated by the line shown in the channel region in Figure 5A that extends from the source to just beneath the edge 33 of the region of trapped charge 66. Due the fact that the device is in inversion (i.e., the channel is in a conductive state), the potential in the inversion layer is pinned to ground potential because the source is grounded. The voltage in the inverted channel near the trapped charge (i.e., just to the left of the right edge 35 of the channel inversion region) is approximately zero. Thus, the voltage across the region of trapped charge is close to the full drain potential of 2 V. Due to the drain potential across the channel region beneath the trapped charge 66 some of the electrons punch through across the trapped region to the drain, resulting in a channel current.

The diagonal line under the channel in Figures 2 and 5A indicate the reduction in the number of electrons in the channel as a function of channel distance. The channel region under the trapped charge is off (i.e., not inverted) due to the high threshold voltage required to invert this region under the trapped charge. However, the channel region inside the dashed circle 23 in Figure 2 and under the region 66 in Figure 5A is a depletion region because the device is in saturation (a device will be in saturation when V_{DS}, the voltage from drain to source, is higher than V_{DSAT}, the saturation voltage). Due to the voltage on the drain 34, a lateral electric field exists in this portion of the channel under region 66. As a result of this lateral electric field, any electron arriving at the edge of the depletion region will be swept through and pulled to the drain 34. As described earlier, this phenomena is called punch through. Punch through occurs if the lateral electric field is strong enough to draw electrons through to the drain, regardless of the threshold level. In order to prevent punch through from occurring during a read, the prior art memory cells require a much longer programming time than does the memory cell of this invention because the prior art memory cells are read in the forward direction. As the memory device is programmed for a longer and longer time, more and more electrons are injected into the nitride, increasing the length of the programmed portion 66 (Fig. 5A) of the channel. The memory cell must be programmed for an amount of time that yields a trapped charge region 66 of sufficient length to eliminate the punch through of electrons. When this occurs, the lateral electric field is too weak for electrons to punch through to the drain under normal operating conditions. As an example, for the threshold voltage equaling 3V during read in the forward direction, Fig. 4 shows that at programming time of approximately 3 milliseconds is required.

Reading in the Reverse Direction

However, if the flash EEPROM memory cell 10 (Figure 5B) is read in the reverse direction, a very different scenario exists. Reading in the reverse direction means reading in a direction opposite that of programming. In other words, voltages are applied to the source 14 and the gate 24 and the drain 16 is grounded. Similar to the prior art memory device of Figure 5A, the memory device of Figure 5B is programmed in the forward direction by injecting hot electrons into region 68 of the nitride layer 20. Since nitride 20 is a nonconductor, the trapped charge remains localized to the region near the drain, for the right bit, for example. The left bit is similar except that source and drain functionality are reversed. The region of trapped charge is indicated by the cross hatched area 68 in Figure 5B. Thus, the threshold voltage rises, for example, to approximately 4V only in the portion of the channel under the trapped charge 68. The threshold voltage of the remainder of the channel remains at, for example, approximately 1 V.

To read the right bit of the device of Figure 5B in the reverse direction, a voltage is applied to the source 14 and the gate 24, for example 2V and 3V, respectively, and the drain 16 is grounded. A major difference between reading in the forward direction and reading in the reverse direction is that when reading in the reverse direction, the gate voltage required to put the channel of the memory device into inversion increases significantly. For the same applied gate voltage of 3V, for example, there will be no inversion but rather the channel of the memory device will be in depletion. The reason for this is that the channel region next to the drain 16 (which functions as the source in read) is not inverted due to the electron charge in region 68 of the nitride 20. The channel adjacent the source 14 (which functions as the drain in read) is not inverted because 2V is applied to the source 14 and the channel, to be inverted, must be inverted relative to 2 V. In the case of reading in the reverse direction, in order to sustain a higher voltage in the channel, a much wider depletion region must be sustained. A wider depletion region translates to more fixed charge that must be compensated for before there can be inversion. When reading in the reverse direction in accordance with the present invention, to achieve a voltage drop across the charge trapping region 66 of the prior art memory device shown in Figure 5A similar to the voltage drop achieved when reading the same device in the forward direction, a higher gate voltage is required, for example, 4 V. This is in contrast to the prior art memory device where the source was grounded and a lower gate voltage was required to invert the channel. In the memory device of the present invention, a much higher gate voltage is required to pin the voltage in the channel to a higher voltage, i.e., the 2V that is applied to the source terminal rather than ground. In other words, the present invention recognizes and takes advantage of the fact that for the same magnitude potential across the drain and the source, the voltage across the portion of the channel under the trapped charge region 68 (Figure 5B) is significantly reduced when reading occurs in a reverse direction to writing (programming) directly resulting in less punch through and greater impact of the programming charge injected in region 68 of the nitride layer 20 (Figure 5B) on the threshold voltage of the transistor. As an example, for the threshold voltage V_T equaling 3v during reverse read, Fig. 4 shows that a programming time of approximately 2 microseconds is required. This programming time is three orders of magnitude less than the programming time required for the same threshold voltage when the cell is read in the forward direction.

In the prior art, memory cells utilizing the ONO structure have had difficulty retaining the localized charge in the nitride layer. This is because such memory cells are programmed in a first forward direction and then read in the same direction. The reading of the programmed cell in the forward direction requires a significant amount of charge to be stored on the nitride to provide the desired increase in threshold voltage associated with the programmed cell. However, in accordance with this invention, by reading in the reverse direction, significantly less charge is required to be stored on the nitride to achieve the same increase in threshold voltage in a programmed cell. Fig. 4 shows the difference in charge (measured as a function of programming time required to achieve a given threshold voltage V_T) for reading in the reverse direction versus the forward direction. In the prior art, the charge retention in a localized region of the silicon nitride layer was difficult if not impossible to achieve because the lateral electric field generated by the charge dispersed the charge laterally in the nitride layer. Such dispersion particularly occurred during the high temperature retention bake required for quality control and reliability. The high temperature retention bake typically requires temperatures between 150 degrees Centigrade to 250 degrees Centigrade for at least 12 to 24 hours. The charge in the prior art devices typically dispersed through the nitride during the high temperature bake causing the performance of prior art devices using the nitride layer as a charge retention material to be less than satisfactory. Accordingly, prior art devices that use the nitride layer for charge retention are not widely used. In addition, charge stored on the nitride layer in prior art memory cells is particularly prone to lateral diffusion and dispersion through the nitride layer in response to the retention bake due to the internal fields causing what is known as electron hopping. The phenomena of electron hopping is exponentially dependent on the field strength. In the case of charge in the nitride layer the internally generated electric field is directly related to the amount of charge stored on the nitride layer. Because electron hopping is exponentially dependent upon the electric field strength, the additional charge required to obtain a given threshold voltage change or shift when the memory cell is read in the same direction as it was programmed causes a very significant change in the charge distribution in the nitride layer. This change in the charge distribution seriously degrades the threshold voltage from the intended (i.e., design) threshold voltage. Consequently, prior art ONO devices have not been successful.

In accordance with the present invention, by reading the memory cell in the reverse direction from which the memory cell is programmed, the amount of charge required to achieve a given threshold voltage is reduced in some cases by a factor of two or three times the amount of charge required to obtain the same threshold voltage shift when the memory cell is read in the forward direction. Accordingly, the internal electric fields generated by the charge in the nitride when the memory cell is to be read in the reverse direction are much less than the internal electric fields associated with the charge stored on the nitride when the memory cell is to be read in the forward direction. Consequently electron hopping is exponentially reduced and the small amount of charge stored in the nitride does not disperse laterally through the nitride due to the internally self generated electric fields even during retention bake. Consequently, the memory cell of the present invention does not suffer the degradation in performance and reliability of prior art ONO memory cells which are programmed and read in the same direction.

Sample Flash EEPROM Device Data

Data obtained from flash EEPROM devices constructed in accordance with the present invention will now be presented to help illustrate the principles of operation thereof. A graph illustrating the difference in threshold voltage in the forward and reverse directions as a function of drain voltage for a flash EEPROM cell of the present invention that has been previously programmed is shown in Figure 6. The memory cell used to obtain the data presented in Figures 6, 7 and 8 was constructed with a bottom oxide layer 18, a top oxide 22 and a nitride layer 20, each 100 Angstroms thick. The drawn width of the channel measures 0.6 microns and the drawn length of the channel measures 0.65 microns.

While reading in the forward direction, the threshold voltage is approximately the same as the threshold voltage when reading in the reverse direction for low drain voltages. At low drain voltages there is insufficient potential for punch through to occur. However, as the drain voltage increases while reading in the forward direction, the punch through region increases resulting in lower threshold voltage. At a high enough drain voltage, the entire portion of the channel under the trapped charge in region 68 of nitride layer 20 (Figure 5B) is punched through and the threshold voltage levels off at the original threshold voltage of the channel.

However, while reading in the reverse direction, the V_T versus V_D) curve appears to follow the V, versus VD curve while reading in the forward direction at low drain voltages. However, the curves rapidly diverge for higher drain voltages and the threshold voltage for reading in the reverse direction levels off at approximately 4V. At a gate voltage V_G of approximately 4V and a drain voltage V_D of 1.2V, the device has reached saturation (V_{DSAT}). At this gate voltage, any further increase in V_D cannot be transferred through the inversion layer thus establishing the maximum potential drop across the portion of the channel beneath the charge trapping region 68. The V_T then becomes independent of further increases in V_D. For example, at a drain voltage of 1.6V, the difference in V_T between reverse and forward read is almost 2V.

A graph illustrating the difference in drain current in the forward and reverse directions as a function of drain voltage for a flash EEPROM cell of the present invention that has been programmed is shown in Figure 7. In Figure 7, rather than measure threshold voltage, the drain current is measured while keeping the gate voltage constant. In the forward direction, as expected, the drain current I_D increases as the drain voltage V_D increases. The curve labeled FORWARD also resembles the I_D curve for reading an unprogrammed cell in the reverse direction.

The drain current while reading in the reverse direction also increases with increasing drain voltage (measured at the source which functions as the drain when reading in the reverse direction) but the drain current levels off at a much lower current than when reading in the forward direction. The difference between drain currents at a V_D of 2V is on the order of approximately 1000 times. If the logic threshold for this memory cell is set to 10 µA, the forward curve can represent a logic '0' and the reverse curve a logic '1'.

The Voltage V_X in the Channel

The voltage V_X is defined as the voltage in the channel at a distance X from the source. Using the example presented above, the voltage V_X that exists in the channel of the memory cell of the present invention (Figure 5B, for example) will not be 2V because the device is in depletion rather than inversion. On the other hand, the voltage V_X must be larger than 0 because a gate voltage of only 1.5V is able to sustain approximately 0.4V in the channel. The actual voltage in the channel varies across the channel length because of the lateral electric field set up between the source and the drain. The threshold voltage, however, varies as a function of the voltage in the channel.

With reference to Figure 5B, the channel will be in saturation as long as the gate voltage V_G is higher than the threshold voltage V_T and the voltage V_X at any point in the channel is given by

$$V_X = V_{DSAT}$$

with

$$V_{DSAT} = V_G - V_T = V_G - V_T(V_{DSAT})$$

and

$$V_T(V_X) = V_{T0} + \Delta V_T(V_X)$$

As is shown in the above equations, the threshold voltage in the channel is equal to the threshold voltage with the source at zero potential V_{T0} plus a delta threshold voltage ΔV_T which is itself a function of the voltage in the channel.

The leakage current through the channel under the region 68 of trapped charge, plotted as a function of the voltage V_{TC}, across the portion of the channel under the charge trapping region 68 while reading in the reverse direction, is shown in Figure 9. From the graph, one can see that the approximate leakage current I_L through the channel when V_{TC} is 2V is 10^{-5} A. In the case of the prior art memory cell read in the forward direction, the voltage across the portion of the channel under region 68 of trapped charge is approximately 2V. In contrast, the voltage V_X in the channel of the memory device of the present invention at location 27 beneath the edge 25 of the region 68 of trapped charge is not 2V but something less, 1V for example. The leakage current I_L corresponding to 1V across the trapped charge region is approximately 10^{-7} A, a whole two orders of magnitude smaller.

Of importance, the edge of the region of trapped charge formed in the nitride layer during programming is the portion of the trapped charge that begins to affect the gate voltage required to invert the channel beneath that point.

A graph illustrating the gate voltage required to sustain a given voltage in the channel, V_X, spanning the distance from the drain to the edge 27 of the channel under the edge 25 of the charge trapping area for one of the two bits while reading in the reverse direction is shown in Figure 10. The gate voltage V_G that is required to sustain a particular V_X at the point 27 in the channel under the edge 25 of the charge trapping area 68 (Figure 5B) is a function of the number of acceptors N_A in the substrate and the thickness of the oxide T_{OX} and is represented by the dashed/dotted line. The solid line represents the threshold voltage in the channel that exists when the back bias effect on the threshold voltage is zero. In this case, the threshold voltage is constant along the entire channel. However, once there is a voltage in the channel, the threshold voltage is not constant along the channel. As shown in the graph, the threshold voltage increases nonlinearly as the voltage in the channel increases. The relationship between the incremental increase in threshold voltage as a function of channel voltage is well known in the art. A more detailed discussion of this relationship can be found in Chapter 2 of "The Design and Analysis of VLSI Circuits" by L.A. Glasser and D.W. Dobberpuhl, incorporated herein by reference.

It is important to emphasize that the advantages and benefits of reading in the reverse direction are achieved only when combined with the use of relatively low gate voltages. For a particular drain voltage, e.g., 2V, applying a high enough V_G such as 5V, for example, causes the differences in threshold voltages between forward and reverse reading to fade. A graph illustrating the effect of the gate voltage V_G applied during reading on the difference in drain current I_D between reading in the forward direction versus reading in the reverse direction for one of the two bits is shown in Figure 11. The reverse V_T of the device used to generate the curves in the Figure is 3.5V. From Figure 11 it can be seen that as V_G is increased while V_D is kept constant, the I_D curves for the reverse read begin to resemble the curves for the forward read. For example, comparing the forward and reverse read curves when V_G equals 2.5V shows the read current in the reverse direction being about four orders of magnitude lower. At a gate voltage V_G of 3V, the difference in read current between the forward and reverse directions drops to a little more than two orders of magnitude. At a gate voltage of 5V, the difference in read current is only approximately 15%. These curves clearly show that large differences in I_D between the forward and reverse read directions are only obtained when V_G is chosen to be low enough. Thus, the benefits of reading in the reverse direction are only achieved when suitably low gate voltages are used for reading. There is an optimum range within which V_G should lie. If V_G is too low, insufficient current is developed in the channel. On the other hand, if V_G is chosen too high, the differences between reading in the reverse and forward directions are greatly diminished.

A graph illustrating the effect of the gate voltage on the difference in threshold voltage between the forward and reverse directions is shown in Figure 12. The device used to generate the curves in Figure 12 was programmed once to a V_T of 3.5V using a V_D of 1.6V and an I_{TH} of 1 μA. The V_T as a function of V_D during reading was subsequently measured. As labeled in Figure 12, the I_{TH} level for the lower two curves is 1 μA, and is 40 μA for the upper two curves. The effect of raising the I_{TH} is to force the V_T, measurement to be at a higher V_G level even though the amount of charge trapped in the silicon nitride layer is identical for all measurements. For the lower two curves (I_{TH} of 1 μA) the forward and reverse threshold voltages start to separate from each other at a V_D of approximately 50 mV while the V_T, for the reverse saturates at approximately 0.6 V. For the upper two curves (I_{TH} of 40 μA) the forward and reverse threshold voltages start to separate from each other at a V_D of approximately 50 mV while the V_T, for the reverse saturates at approximately 0.6V. For the upper two curves (ITH of 40 μA) the forward and reverse threshold voltages start to separate from each other at a V_D of approximately 0.35V while the V_T for the reverse saturates at approximately 1.35V. Thus, these curves clearly show that the effect of the trapped charge depends heavily on the choice of V_G.

Programming the Two Bit Cell

With reference to Figure 2, programming the two bit EEPROM cell of the present invention will now be described. In programming the two bit cell, each bit, i.e., the left and right bit, is treated as if the device was a single bit device. In other words, both the left and right bits are programmed as described in the section entitled "Programming One Bit in the Forward Direction." For the right bit, for example, programming voltages are applied to the gate 24 and drain 16 and hot electrons are injected into and trapped in the charge trapping layer 20 in the region near the drain defined by the dashed circle 23. Correspondingly, the threshold voltage of the portion of the channel under the trapped charge increases as more and more electrons are injected into the nitride layer. The programming of the right bit is represented in Figure 2 by the right-pointing arrow labeled 'PROGRAM.' This arrow represents the flow of electrons to the right during programming of the right bit.

Similarly, the left bit is programmed by applying programming voltages to the gate 24 and source 14, which now functions as the drain for the left bit. Hot electrons are injected into and trapped in the charge trapping layer 20 in the region defined by the dashed circle 21. The threshold voltage of the portion of the channel under the trapped charge comprising the left bit increases as more and more electrons are injected into the nitride layer. The programming of the left bit is represented in Figure 2 by the left-pointing arrow labeled 'PROGRAM.' This arrow represents the flow of electrons to the left during programming of the left bit.

A graph illustrating the effect programming one of the bits has on the other bit which has not been previously programmed is shown in Figure 13. In this particular example, the right bit is shown being programmed while the left bit is read. The threshold voltage V_T for the right bit assumes that the right bit is read in the reverse direction to the programming direction. Thus the threshold voltage for a programmed left bit will be relatively low compared to the threshold voltage for the right bit and thus the state of the right bit can be read without interference from the left bit. It is clear from the curves that during programming of the right bit, the unprogrammed left bit remains unprogrammed. This graph also illustrates the read through of the programmed right bit in order to perform a read of the left bit.

A graph illustrating the effect programming one of the bits has on the other bit which has been previously programmed is shown in Figure 14. This graph was generated in two passes. Each curve is labeled either PASS #1 or PASS #2. During the first pass, the right bit was programmed while reading the unprogrammed left bit, as shown by the curves labeled RIGHT BIT-PASS #1 and LEFT BIT-PASS #1. These curves are similar to the curves of Figure 13. During the second pass, once the right bit is programmed, the left bit, previously unprogrammed, is now programmed. At the same time, the right bit is read. The second pass is represented by the curves RIGHT BIT-PASS #2 and LEFT BIT-PASS #2.

As shown in Figure 14, during the first pass, the left bit remains unprogrammed during the programming of the right bit. Programming the right bit does not affect the unprogrammed left bit. During the second pass, the left bit is programmed and the right bit remains programmed and can still be read. The gate voltage during programming is sufficiently high (typically around 10V) that the programmed right bit does not interfere with the programming of the left bit except to increase somewhat the time required to reach a given threshold voltage relative to the time required to reach the same threshold voltage for the right bit when the right bit is programmed. The graph also shows that the right bit can be programmed through during programming of the left bit. Further, the programming of the left bit does not disturb the programmed right bit. This is possible because program through (i.e. the programming of the one bit substantially without interference from the other bit when the other bit is programmed) and read through (i.e. the reading of one bit without interference from the other bit when the other bit is programmed) occurs through both the left and the right bits.

Program through and read through are possible due to the relatively low gate voltages required to turn on each programmed bit when read in the forward direction as occurs when the other bit is read in the reverse direction. Another way to look at this is that a narrow charge trapping region permits punch through to be more effective. Thus the small amount of charge 68 trapped on the right edge of charge trapping layer 20 (Figure 15) and self-aligned with the junction between region 16 and the substrate 12 and a comparable amount of charge 70 trapped on the left edge of charge trapping layer 20 and self-aligned with the junction between region 14 and the substrate 12 cause a narrow charge trapping region to be formed at both the right side and the left side of charge trapping layer 20 which is easy to be punched through when the bit is read in the forward direction. Thus when left bit 70 (the charge trapping region 70 is referred to as a bit because the presence or absence of charge in region 70 would represent either a zero or a one) is read in the forward direction, bit 68 is being read in the reverse direction. The punch-through under charge trap region 70 is quite easily achieved with a low gate voltage thereby allowing the charge trapped in bit 68 to control the state of the signal read out of the device. Thus for equal amounts of charge trapped in regions 70 and 68, reading a bit in the reverse direction results in the opposite bit having no effect on the state of the signal being read.

Another reason that the bit not being programmed is not disturbed is that the programming voltage is not being applied to the drain for the bit previously programmed. When programming the other bit, the programming voltage is applied to the drain for the bit on the other side of the device.

As discussed earlier, the programming duration must be limited for each bit in order that the other bit can still be read. For example, in the case when the right bit is programmed, i.e., a logic '0', and the left bit is not programmed, i.e., a logic '1', if the right bit was programmed for too long a time then when the left bit is read, there may be insufficient current for the sense amps to detect a logic '1' because the channel is not sufficiently conductive. In other words, if the right bit is programmed too long, a left logic '1' bit becomes slower, i.e., takes longer to read due to lower channel current, or, in the worst case, may appear to be a logic '0' because the over-programmed right bit prevents the left bit from being read. Thus, a window exists in the programming time within which a logic '0' bit must fall. One of the variable parameters is the voltage that is applied to the functional drain region during read. As the drain voltage is increased, a longer programming time, i.e., longer area of trapped charge, is required in order to avoid punch through. Thus, a longer trapping region is equivalent to increasing the programming time. The upper limit of the programming time for the window is the programming time such that a forward read does not change the read current by more than a predetermined percentage compared to the read current for a reverse read. Preferably, the percentage change to the read current should be limited to 10%. This percentage, although not arbitrary, can be optimized according to the design goals of the chip designer. For example, a designer may wish to have three orders of magnitude margin between the threshold voltage of a forward read and the threshold for a reverse read. To achieve this, the gate voltage, drain voltage and implant level are all adjusted accordingly to determine a maximum programming time.

The effect of programming one of the bits is that both programming and reading for the second bit is slowed somewhat. The second bit can be programmed as long as the gate voltage during programming is higher than the threshold voltage of the channel with the first bit programmed and sufficient voltage is placed on the drain. The channel resistance, however, is raised due to the programming of the fist bit. As long as programming parameters are tuned properly, the higher channel resistance does not prevent the second bit from being programmed and read. The higher channel resistance, however, does cause programming and reading of the second bit to take longer.

Reading the Two Bit Memory Cell

Reading the two bit EEPROM cell of the present invention will now be described. In reading the two bit cell, as in programming, each bit is treated as if the device was a single bit device. A sectional view of a two bit EEPROM cell constructed in accordance with a preferred embodiment of the present invention showing the area of charge trapping under the gate for both the right and the left bits is shown in Figure 15. The area of trapping for the right bit is referenced 68 and that of the left bit is referenced 70. Also shown in Figure 15 are two arrows labeled 'READ', one pointed in the left direction indicating the direction for reading of the right bit and one pointed in the right direction indicating the direction for reading of the left bit.

As described in the section entitled "Reading One Bit in the Reverse Direction" the right bit is read in the reverse direction by applying read voltages to the source 14 and the gate 24 and grounding the drain 16. For example, a gate voltage of 3V and a source voltage of 2V is applied. The resulting voltage in the channel V_X will be something less than two volts in accordance with the graph in Figure 10 and as described in detail above. Similarly, to read the left bit in the reverse direction, read voltages are applied to the gate 24 and to the drain 16 and the source 14 is grounded, e.g., 3V on the gate and 2V on the drain.

A graph illustrating the effect of a low drain voltage on the read through of a programmed bit is shown in Figure 16. This graph is similar to that of Figure 14 with the addition of the top two curves above 5.1V. The four lower curves were generated using a V_D of 1.6V. The two upper curves were generated by reading the unprogrammed bit after the other bit was programmed using a V_D of 50 mV. These curves show that if V_D is made too low and the first bit is programmed, insufficient voltage exists in the channel for read through to occur. They also show that the second bit to be programmed, in this case the left bit, experiences slower programming due to the increased series resistance of the channel. Even if the second bit is unprogrammed, when the drain voltage is too low and the first bit is programmed, the second bit cannot be read properly. Insufficient voltage exists in order for punch through to occur. If punch though does not occur, the second bit looks as if it is programmed whether it really is or not.

Punch through is very sensitive to the length of the trapped charge region, such as regions 68 and 70 of the structure shown in Fig. 15. Should these regions be too wide or not self-aligned with the appropriate region 16 or 14 (depending on whether the charge represents the right bit 68 or the left bit 70), then punch through would not be able to be guaranteed to occur and this concept would not work. Thus, the self-alignment of the trapped charge to the junction between region 16 and the substrate (for the trapped charge 68) and region 14 and the substrate (for the trapped charge region 70) is crucial to the functioning of this invention.

A read of the two bit memory device of the present invention falls into one of three cases: (1) neither of the two bits are programmed (2) one of the bits is programmed and the other is not or (3) both of the bits are programmed. The first case does not require a read through. The second case requires reading through the programmed bit to read the unprogrammed bit. In this case the margin is the delta between reading a single bit in the forward direction versus the reverse direction. An example of the margin can be seen in Figures 6 and 7 which illustrate the difference in V_T and read current between the forward and the reverse directions for a single bit.

The third case requires read through to read both programmed bits. Programming the second bit, in fact, improves the conditions for reading the first bit. This is so because the voltage in the channel is further reduced over the case of reading a single bit. This increases the read margins between programmed and unprogrammed bits.

It is important to note that although the EEPROM cell of the present invention stores two bits, support circuitry and concepts designed to work with single bit memory cells can still be used. For example, the sense amplifier circuitry needed for the two bit memory cell is basically no different than that for the single bit memory cell. In the single bit memory cell, the sense amplifier circuitry is required to distinguish between two states, the programmed and unprogrammed states. Likewise, in the two bit memory cell of the present invention, the sense amplifiers must also distinguish between only two states: programmed and unprogrammed. This is in direct contrast to the prior art approaches to multi-bit memory cells wherein multiple thresholds are used which require multiple current levels to be detected by the sense amps. Accurately detecting multiple current levels in a memory device is a complex and difficult task to accomplish. The memory cell of the present invention, however, requires that the sense amps only distinguish between two states as in the single bit memory cell.

In the case when one of the bits is unprogrammed, i.e., no charge injected into charge trapping layer for that bit, a read of the other bit will be unaffected by this unprogrammed bit. On the other hand, however, in the case when one bit is programmed, a read of the other bit will be affected by this other programmed bit to some extent. Depending on various process parameters, the programmed bit may cause the channel to be less conductive. However, as long as the channel is sufficiently conductive both bits can be still be programmed and read correctly. This is discussed in more detail in the section titled "Optimization Parameters" presented below.

With reference to Figure 15, the two bit memory device of the present invention utilizes a punch through or read through technique to read one bit when the other bit is in a programmed state. In order to read, for example, the right bit 68, the read current must be able to read through or punch through the left bit 70, assuming that both the left bit and the right bit have been programmed. Thus, there is a limit on the length of the charge trapping region that can be programmed. The charge trapping region must be short enough to permit punch through of the bit not being read. If a bit is in the unprogrammed state, there is no constraint on the read current of the other bit from the unprogrammed bit.

It is important to note that when a semiconductor device is scaled, the channel lengths become shorter and short channel effects take hold. Thus, in the two bit memory cell, because each bit is stored in different areas of the transistor, short channel effects may become prevalent sooner than in the case of the single bit transistor. In order to retain the usable range of drain voltage, the two bit transistor may need to be scaled by a smaller factor.

Criteria Necessary For Two Bit Operation

A key concept associated with the two bit EEPROM memory cell of the present invention is that for the device to operate properly, both bits must be able to be written and read. If one of the bits is programmed, a reverse read on the programmed bit must sense a high V_T, i.e., a '0' and a reverse read on the unprogrammed bit must sense a low V_T, i.e., a '1'. Thus, a reverse read on the unprogrammed bit, which is equivalent to a forward read on the programmed bit, must punch through the region of trapped charge in order to generate a high enough read current. If this does not happen, the unprogrammed bit will not be able to be read as a '1', i.e., a conductive bit.

In order to achieve this goal, a sufficient margin is generated between reading in the forward and reverse directions. With reference to Figure 11, in order to store two bits, there must be sufficient difference between forward read of one of the bits and reverse read of the other bit. In addition, the reverse read current for one of the bits when the other bit is and is not programmed should be sufficient to distinguish between the two bits. For example, in Figure 11, for a gate voltage of 3V, punch through for reading in the reverse direction occurs at approximately 1V. Thus, a drain voltage of 1.6V creates a suitable safety margin ensuring that the second bit can be read when the first bit is programmed.

There are two parameters that can be used to ensure punch through of the charge trapping region. The first is the V_G, applied during reading and the second is the width of the charge trapping region. A low V_G used during reading combined with a narrow charge trapping region makes punch through more effective. The lower gate voltage produces a weaker vertical electric field which causes the lateral electric field to be stronger.

It is more important to use a low V_G during reading in the two bit memory cell than in the single bit memory cell. In the single bit case, it only had to be ensured that the reverse read was better than the forward read, meaning that the V_T of a given bit during forward reading was lower than the V_T of this bit during reverse reading. In the two bit case, however, it is not enough that the V_T drops in the forward case, it must drop sufficiently to be able to punch through when reading the other bit. If the delta V_T between the forward and reverse read is not sufficient, one bit cannot be read when the other bit is programmed.

Erasing Prior Art Memory Devices

As discussed previously in connection with U.S. Patent No. 4,173,766, issued to Hayes, a major disadvantage of the Hayes prior art insulated gate device is the difficulty in using the Hayes device to construct a flash EEPROM. A consequence of using an oxide-nitride structure as opposed to an oxide-nitride-oxide structure is that during programming the charge gets distributed across the entire nitride layer. The absence of the top oxide layer lowers the ability to control where the charge is stored in the nitride layer and allows holes from the gate to neutralize charge in the nitride layer. A thick nitride layer is required in order to generate sufficient charge retention in the device. However, the relatively thick nitride layer causes the charge trapping region to be very wide thus making erasing the cell difficult if not impossible. Thus there is a tradeoff between charge retention and sufficiently large threshold voltage deltas on the one hand and the ability to erase the device on the other hand.

Some of the prior art devices that use hot electron programming utilize an erase mechanism whereby the electrons previously trapped in the nitride are neutralized (i.e., erased) by transferring holes into the nitride. The information is erased by grounding the gate and applying a sufficient potential to the drain to cause avalanche breakdown. Avalanche breakdown involves hot hole injection and requires relatively high voltages on the drain for the phenomena to occur. The hot holes are generated and caused to jump over the hole potential barrier of the bottom oxide between the channel and the nitride and recombine with the electrons in the nitride. This mechanism, however, is very complex and it is difficult to construct memory devices that work in this manner. Another disadvantage of using hot hole injection for erasing is that since the drain/substrate junction is in breakdown, very large currents are generated that are difficult to control. Further, the number of program/erase cycles that the memory cell can sustain is limited because the breakdown damages the junction area. The damage is caused by very high local temperatures generated in the vicinity of the junction when it is in breakdown.

Erasing the Two Bit Memory Cell

The erase mechanism of the two bit flash EEPROM memory cell 10 (Figure 15) will now be described in more detail. The mechanism used to erase the two bit flash EEPROM memory cell of the present invention involves the movement of electrons as opposed to the movement of holes. For the right bit, an erase is performed by removing electrons from the charge trapping nitride region 68 either through the gate 24 via the top oxide 22 or through the drain 16 via the bottom oxide 18. For the left bit, an erase is performed by removing electrons from the charge trapping nitride region 70 either through the gate 24 via the top oxide 22 or through the source 14 via the bottom oxide 18.

Using the right bit as an example, one technique of erasing is to simultaneously apply a negative potential to the gate 24 and a positive potential to the drain 16 such that electron tunneling occurs from the charge trapping nitride layer 20 to the drain 16 via the bottom oxide 18. The left bit is erased in a similar fashion except that a positive potential is applied to the source 14 rather than the drain 16. The electron tunneling is substantially confined to a local area near the drain 16. To facilitate the erasing of the memory cell 10 using this technique, the thickness of the bottom oxide layer 18 is suitably constructed (i.e., has a thickness of about seventy (70) Angstroms) to optimize the removal of electrons from the nitride charge trapping layer 20 into the drain 16.

Using the right bit as an example, a second well known technique is to simultaneously apply a positive voltage potential to the gate 24 and zero potential, i.e., ground, to the drain 16 such that electron tunneling occurs from the charge trapping nitride layer 20 through the top oxide 22 to the gate 24. The right bit is erased in a similar fashion with zero potential applied to the source 14. In this case, the top oxide 22 is suitably constructed (again with a thickness of about seventy (70) Angstroms) to optimize the tunneling of electrons from the nitride charge trapping layer 20 into the gate 24 in order to facilitate the erasing of the memory cell 10. In one embodiment, the top oxide 22 has a thickness of 50 Angstroms to 80 Angstroms for a voltage on gate 24 of 10 to 18 volts.

A graph illustrating the effect of programming on erase time for reading in the forward and reverse directions is shown in Figure 17. Figure 17 shows the times necessary to program the device to a threshold voltage of four (4) volts for reading in both the reverse (10^{-5} seconds) and forward ($3x10^{-3}$ seconds) directions. The graph presented in Figure 17 is based on data obtained from a memory cell constructed in accordance with the present invention. In the first pass, the device was programmed to be read in the reverse direction and then erased. In the second pass, the device was programmed to be read in the forward direction and then erased. The erase processes for the charges associated with reverse read and forward read used the same drain voltage and gate voltage, namely a V_D of 5.5V and a V_G of -8V. The thickness of the top oxide, bottom oxide and nitride layers are all 100 Angstroms. Programming for forward reading and reverse reading utilized a V_D of 5.5V and V_G of 10V. Only the programming times differed. The forward and reverse programming curves are identical to those illustrated in the graph of Figure 8.

As can be seen from Figure 17, even when the device is programmed to the same threshold voltage, the time to complete the reverse erase is much less than the time to complete the forward erase. The forward erase (i.e. the time to remove the trapped charge associated with a given threshold voltage when the device is read in the forward direction) is slower than the reverse erase (i.e. the time to remove the trapped charge associated with a given threshold voltage when the device is read in the forward direction). In addition, there is residual charge left in the charge trapping region as shown in the small gap between the reverse and forward erase curves at the one (1) second mark. This is due to the larger wider charge trapping region formed during the forward programming that was required to generate a threshold voltage of 4V. From the curves, the forward erase is approximately an order of magnitude slower than the reverse erase. The abrupt increase in threshold voltage for the curve labeled 'FORWARD ERASE' is due to the reverse read used to measure the threshold voltage. For the same amount of charge trapping, the equivalent threshold voltage for reverse reading is much higher than that for forward reading. As can be seen in Figure 17, the slopes of the forward and reverse erase curves are different. Reading in the reverse direction requires trapped charge so much smaller than does reading in the forward direction that the erase of the trapped charge is approximately 10 to 20 times faster. Also apparent from Figure 17 is that the cell does not enter deep depletion. Even at the 1 second erase mark, the threshold voltage (about 2v) is no lower than that of an unprogrammed cell. This is a huge advantage of the memory cell of the present invention over prior art memory cells especially floating gate cells where over-erase can cause a failure of the memory array due to deep depletion of the charge on the floating gate.

The erase mechanism in the memory cell is self limiting due to the fact that as the memory cell is erased, more and more positive charge is stored in the trapping region 68 (Figure 15) (for the right bit) of the nitride layer thereby neutralizing the negative charge stored there while the remainder of the nitride layer 20 remains unaffected. Thus, the threshold voltage of the channel keeps dropping until it levels off at the threshold voltage of an unprogrammed memory cell which is the threshold voltage of the larger majority of the channel closer to the source. Over-erasing the memory cell of the present invention only affects (i.e., lowers) the threshold voltage of the portion of the channel under the charge trapping region 68 which is a relatively narrow region while leaving the threshold voltage of the remainder of the channel at its normal value. A graph illustrating the separate bit erase capability of the two bit EEPROM memory cell of the present invention is shown in Figure 18. The graph was generated in two passes and initially, both the right and the left bit are programmed each with an amount of trapped charge to achieve a given threshold voltage when read in the reverse direction. During the first pass, the right bit was erased while the left bit was read, as represented by the curves labeled RIGHT BIT-PASS #1 and LEFT BIT-PASS #1. During the second pass, the left bit was erased while the right bit was read, as represented by the curves labeled RIGHT BIT-PASS #2 and LEFT BIT-PASS #2. The graph shows that erasing of one of the bits does not affect the other bit. This is due to the fact that the erase voltage is localized to the junction adjacent to the bit that is to be erased. The difference in location between the curve labeled "Left Bit-Pass #2" and the curve labeled "Right Bit-Pass #1" is of no significance being well within the tolerance of the measurements.

A graph illustrating the effect of cycling on the program and erase ability of the two bit EEPROM cell of the present invention is shown in Figure 19. The graph shows the V_T of a bit associated with a given amount of trapped charge for reading in the reverse direction (top line) and the forward direction (bottom line). The gradual increase in threshold voltage V_T for reading in both the forward and reverse directions reflects the lack of complete erasure of all the stored charge during each erase such that the amount of trapped charge gradually increases with time after programming and erasing for 1000 cycles.

As explained previously, a result of reading in the reverse direction is that a narrower charge trapping region is required due to the higher efficiency of the reverse read. Since erasing is always performed through the effective drain region 16 (for trapped charge 68 and region 14 for trapped charge 70), less charge needs to be moved off the charge trapping layer 20 and directed through the drain 16 (charge 68) or effective drain 14 (charge 70). Thus, reading the memory cell 10 in the reverse direction enables much faster erase times. This makes the entire erase process much easier than in prior art memory devices. In the prior art memory device (i.e., forward programming/forward read), the charge trapping region 66 (Figure 5A) was much bigger and wider to achieve the desired change in threshold voltage, thus making the erase process more difficult. To erase the cell 41, a larger amount of charge spread out over a wider trapping region 66 must be directed through the drain 34. The danger with this lies in that if the charge trapping region 66 becomes too wide, the cell 41 may never be able to be completely erased. The charge trapping region 66 may become too wide if the device is overprogrammed which is a real possibility when programming and reading in the forward direction.

A graph illustrating the effects associated with over programming on the ability to erase in the forward and reverse directions is shown in Figure 20. The graph presented in Figure 20 was constructed using data obtained from a memory cell 10 (Figures 5B and 15) constructed in accordance with the present invention. The top oxide 22 (Figure 15), bottom oxide 18 and nitride layer 20 are each 100 Angstroms thick for a total ONO thickness of 300 Angstroms. Programming utilized a V_D of 5.0V and V_G of 10V. Erasing utilized a V_D of 5.0V and a V_G of-8V. Note that programming and erasing are both in the forward direction. Reading, however, is either in the forward or reverse direction. It is the reverse direction read in conjunction with the careful control of the gate voltage to be within a selected range, that yields the advantages of this invention.

In this case, the memory cell, which has been programmed for 100 milliseconds, does not fully erase in a reasonable time (shown in Figure 20 as 100 milliseconds) with V_T being approximately 7V after 100 milliseconds of erase for reading in both the forward and reverse directions. The cell 10 cannot be erased because it has been over programmed, meaning the charge trapping region was made too wide to effectively erase. After 100 milliseconds of programming, the charge trapping region is very wide. The 13V (V_D of 5V and V_G of-8 v) that is applied across the charge trapping region 68 (Figure 5B) to erase the trapped charge is effective in removing the electrons that are close to the drain 16. However, the electrons that are trapped further away from drain 16 towards the middle of the channel cannot be effectively removed because the electric field created by the 13V potential difference between the drain and the gate is much weaker at that point.

As is apparent from Figures 17 and 20, the slopes of the threshold voltage V_T versus program time curves for forward read and reverse read (labeled "forward program" and "reverse program" in Figure 20) are different. After approximately one millisecond, the forward program curve exhibits a higher slope than the reverse program curve. This shows that reading in the reverse direction is more tolerant to over programming than reading in the forward direction in the sense that a given uncertainty in programming time causes a bigger uncertainty in threshold voltage V_T when reading in the forward direction than when reading in the reverse direction. When reading in the reverse direction, a V_T, of about 4V is reached after approximately 100 microseconds of programming. Even if programming continues up until a millisecond, a factor of 10X, the V_T, for reading in the reverse direction is only approximately 4.5V. For reading in the forward direction, a V_T of 4V is reached only after approximately 7 milliseconds of programming. If programming is off by only 3X, the V_T increases to approximately 8.3V. At this high V_T it is not likely that the device can be erased.

Thus, it is important to stress that reading the memory device in the reverse direction does not just enable simpler and faster erasing, but in fact, if the device is to be read in the forward direction and the trapped charge is so adjusted to give the desired threshold voltage V_T, erasing is likely to be not possible at all. This is because much more charge must be trapped on the dielectric 20 beneath the gate 24 to achieve a usable difference in threshold voltage V_T between the programmed and the unprogrammed state when reading in the forward direction than when reading in the reverse direction. This makes erasing the memory device at best difficult if not impossible thus making the forward programming/forward read impractical for this type of memory device which must be erasable.

The graph of Figure 20 also illustrates the higher effectiveness during erase of the voltage on the drain versus the voltage on the gate. The gate voltage is not as effective due to the distance of the gate from the trapped charge which includes the thickness' of the top oxide 22 and the nitride layer 20. The drain voltage is more effective since it is more proximate to the region 68 of trapped charge. However, the gate voltage is more crucial when the width of the trapped charge region 68 is narrow. In this case, the gate voltage will be effective in creating an electric field that covers the entire charge trapping region 68 making the removal of electrons more efficient. The trapped charge region can only be made sufficiently narrow if the device is read in the reverse direction because only when the device is read in the reverse direction does a relatively small amount of charge stored on the dielectric under the gate yield a sufficiently larger difference in threshold voltage VT to allow the programmed state (i.e., charge stored on the gate) and the unprogrammed state (i.e., no charge stored on the gate) to be differentiated. As discussed previously, if the device is read in the forward direction, the charge trapping region must be made wide enough to generate a sufficient threshold voltage to differentiate between the programming and the unprogrammed states. Charge trapped far from the drain cannot be compensated for by lowering the voltage on the gate. In addition, the drain voltage cannot be increased beyond approximately 2V due to read disturb. The read disturb refers to slow programming of the bit during read. While the programming occurs very slowing, constantly reading the same cell over an extended period of time can cause programming of the bit to occur.

A graph illustrating the programming and erasing curves representing the use of oxide versus TEOS as the dielectric on top of the nitride is shown in Figure 21. The graph presented in Figure 21 was constructed using data obtained from two memory cells constructed in accordance with the present invention, one memory cell using TEOS to form the oxide layer 22 (Figure 15) on top of the nitride and the other memory cell using thermal oxidation of the nitride to form the top oxide layer 22. The thickness' of the top oxide layer 22, bottom oxide layer 18 and nitride layer 20 are 70, 100, 80 Angstroms, respectively. The width/length ratio for each memory cell channel is 0.6/0.65 microns. Programming (which is always done in the forward direction) utilized a V_D, of 5.0V and a V_G of 10V. Erasing (which is also always done in the forward direction) utilized a V_D of 5.0V and a V_G of-6V. This graph shows that there is little difference in the programming and erase characteristics when either oxide or TEOS is placed on top of the nitride.

A graph illustrating erase times for a gate voltage of zero with two different values of drain voltage is shown in Figure 22. Both curves were generated by first programming in the forward direction for about 10 microseconds, until the threshold voltage V_T equals about 4V and then erasing in the forward direction. For the upper curve, the gate 24 (Figure 15) was grounded and 6.0V applied to the drain 16. For the lower curve, the gate 24 was grounded and 6.5V applied to the drain. For both curves, the threshold voltage is raised during programming from nearly 1.5V to approximately 4V. Erasing then brings the VT back down to approximately 1.7V. Note that the time to erase the charge from the dielectric decreases as the drain voltage increases. The curves show that it takes about 100 seconds with a gate voltage of 6.5V to erase (i.e., remove) sufficient charge from the dielectric to bring the threshold voltage of the device down to about 1.9V and that it takes about 1000 seconds with a gate voltage of 6.0V to achieve the same threshold voltage.

A graph illustrating the erase curve for two different values of negative gate voltage is shown in Figure 23. The graph presented in Figure 23 was constructed using data obtained from a memory cell constructed in accordance with the present invention. The thickness of each of the top oxide 22 (Figure 15), bottom oxide 18 and nitride 20 layers is 100 Angstroms for a total dielectric thickness of 300 Angstroms. The channel width/length ratio is 0.6/0.65 microns. Erasing for the reverse direction utilized a constant V_D of 5.5V and a V_G of -5V versus a V_G of -7.5V. The graph shows that drain and gate voltages on the order of 5V and -5V respectively, are sufficient to enable an effective erase. This is a big advantage over the prior art where erase voltages of around -10V on the gate are more typical. The graph also shows that lowering V_G to -7.5V is effective to erase the device approximately 20 times faster while still retaining a V_G less than 10V.

Benefits of Reading in the Reverse Direction

Reading the graph in Figure 10, one can see that to achieve a V_X equal to approximately 2V in the channel (i.e., the same conditions as the prior art memory device with 3V applied to the gate) when reading in the reverse direction, approximately 4V must be applied to the gate. When, for example, 3V is applied to the gate and the device is read in the reverse direction, only approximately 1.2V is generated in the channel. This is in direct contrast to the prior art reading in the forward direction wherein the potential across the trapped charge region was almost the full potential applied to the drain (i.e., 2V). This significant benefit of reading in the reverse direction is that for the same gate voltage a much lower voltage is present across the portion of the channel under the region of trapped charge. This results in dramatically less leakage current for the same charge trapping length. Or stated another way, a shorter charge trapping region is needed in the gate dielectric to achieve an equivalent amount of leakage current. A shorter charge trapping region translates through an exponential function to shorter programming times. A discussion of the variation in programming time as a function of various parameters, voltage and temperature is given in a paper entitled "Hot-Electron Injection Into the Oxide in n-Channel MOS Devices," B. Eitan and D. Frohman-Bentchkowsky, IEEE Transactions on Electron Devices, March 1981, incorporated herein by reference.

The effect of reading the memory device in the reverse direction is to amplify the effect of the trapped charge (i.e., the programmed region or the localized trapping region) on the threshold voltage thereby allowing much less charge to be trapped to achieve the same difference in threshold voltage between the programmed state (i.e., charge stored in the charge trapping region of the gate dielectric) and the unprogrammed state (i.e., no charge stored in the charge trapping region of the gate dielectric) of the device. For the same programming time (meaning the same length of trapped charge in the nitride, for example as shown in Figures 5A and 5B), device 10, when read in the reverse direction, exhibits a leakage current I_L approximately two orders of magnitude less than that of a prior art memory cell. As previously discussed, by reading in the reverse direction, a major benefit is that the programming time can be reduced because the leakage current is significantly less and thus less trapped charge is required to achieve the same leakage current as when reading in the forward direction. Thus, the size of the trapping region does not have to be as large as with prior art memory cells which translates exponentially into shorter programming times.

A key advantage of reading in the opposite direction from programming is that the effect of the lateral electric field next to the charge trapping region is minimized. In addition, the gate voltage can be reduced to further minimize the potential in the channel. In fact, the gate voltage can be set to achieve the desired voltage in the channel. This was described previously with reference to Figure 10.

The area of charge trapping necessary to program memory cell 41 of the prior art is illustrated in Figure 24A and the area of charge trapping necessary to program memory cell 10 of the present invention is illustrated in Figure 24B. The trapping region 68 of device 10 is shown much smaller than trapping region 66 of the prior art device. As described earlier, reading in the reverse direction permits a shorter charge trapping region. This results in much more efficient programming by reducing, through an exponential function, the programming time of the device.

Programming a smaller, narrower region of trapped charge has numerous benefits. One major benefit is that programming times are reduced while the delta in threshold voltage between the programmed versus unprogrammed states remains the same. Thus, short programming times are achieved by taking advantage of the asymmetric characteristics of the trapping dielectric flash EEPROM memory cell. Another major benefit is that the erase mechanism of the memory cell is greatly enhanced.

The erase mechanism is enhanced when the charge trapping region is made as narrow as possible. Programming in the forward direction and reading in the reverse direction enables limiting the width of the charge trapping region to a narrow region near the drain. This allows for much more rapid and thus more efficient erasing of the memory cell.

Yet another benefit of reading in the reverse direction, as described above, is that a narrow charge trapping region increases the effectiveness of the drain voltage during erase when combined with relatively low applied gate voltages. A narrow charge trapping region is allowed only by reading in the reverse direction while applying low gate voltages during the read.

Further, utilizing a thinner silicon nitride charge trapping layer than disclosed in the prior art helps to confine the charge trapping region to a region near the drain that is laterally narrower than in the prior art. This improves the retention characteristic of the memory cell. Further, the thinner top and bottom oxide sandwiching the nitride layer helps retain the vertical electric field.

In addition, when the memory cell is read in the reverse direction, it is more tolerant of over programming. Reading in the forward direction causes the threshold voltage of the memory cell to be very sensitive to inaccuracies in programming time while reading in the reverse direction reduces this sensitivity of threshold voltage to programming time. Over programming while programming to allow reading in the forward direction can potentially cause the device to become non erasable.

The voltage V_X in the channel is a function of the gate voltage and the impurity level in the channel. V_X is the voltage in the channel just beneath the edge of the trapped charge region above the channel (Figure 5B). A higher gate voltage translates to a higher voltage in the channel. When the device is N channel, the impurity in the channel region before inversion is usually boron. The voltage V_X is generally independent of the boron impurity level over a normal range of values in the forward reading mode, but V_X is dependent on the impurity level in the reverse direction, becoming smaller as the impurity level goes up. Indeed in the reverse direction the voltage V_X in the channel just beneath the edge of the trapped charge region is given by the following expression

$$V_X = V_G - (V_T + \Delta V_T)$$

where V_T is the device threshold voltage for zero substrate bias and ΔV_T is the incremental increase in threshold voltage due to substrate back bias caused by a finite value for V_X when the channel is just inverted.

Various thickness' were tried for the second oxide layer 22 in the ONO structure of Figures 5B and 24B. The following table presents the combinations of thickness' for the ONO layers that were constructed for three embodiments of the memory cell of this invention. Note that all thickness' are in Angstroms in the table below.

Layer	Embodiment # 1	Embodiment #2	Embodiment #3
Top Oxide ('O' Layer 22)	150	100	70
Nitride ('N' Layer 20)	50	50	50
Bottom Oxide ('O' Layer 18)	70	70	70
Total Thickness	270	220	190

The nitride layer 20 retains the stored charge. By employingthe reverse read as opposed to the forward read, the amount of charge required to be retained for a given shift in threshold voltage is reduced by a factor typically of two or more. By making the nitride layer 20 thinner and the top oxide layer 22 thicker, the amount of charge required to be stored on the nitride layer 20 for a given threshold voltage shift is also reduced.

It is also noted that as the thickness of the top oxide layer 22 increased, the lateral fields associated with the charge stored on the 50 Angstrom thick nitride layer 20 decreased slightly. It is also observed that as the thickness of the bottom oxide layer 18 was made thinner, the erase of the charge stored on the nitride layer 20 becomes easier.

For a 70 Angstrom thick bottom oxide layer 18, the charge stored on the nitride layer 20 is more easily erased than if the bottom oxide layer 18 is 100 Angstroms thick.

Thus, the conclusion is that the thinner the nitride the better for the purposes of the present invention. Nitride layers as thin as 20 Angstroms are believed possible with this invention. The thinner nitride reduces the lateral field associated with a given charge stored in a portion of the nitride layer and thus reduces the lateral dispersion of the stored charge as a result of the internally generated electric field associated with the stored charge.

Optimization Parameters

In terms of optimization, three parameters can be varied to give the quickest programming time and the widest margins. The first parameter is the channel length. A longer channel length, for a given programming time when reading in the reverse direction, increases the distance between the drain and the trapped charge (effectively, the source and drain designations are flipped). This lowers the level of the lateral electric field even lower.

The second parameter, as described previously, is the gate voltage which can be set to minimize the voltage drop in the channel across the channel region beneath the trapped charge. This further reduces the lateral electric field in the channel beneath the trapped charge. Within limits, the voltage in the channel can be 'dialed in' by varying the voltage on the gate. This allows control over the voltage drop in the channel beneath the region of trapped charge. If the gate voltage is made too low then reading a logic '1', i.e., the unprogrammed state, becomes problematic. The gate voltage for reading a logic '1' must be still high enough to generate inversion in order to produce sufficient read current for each sense amplifier. Thus, a lower limit for the gate voltage is approximately 1V above the threshold voltage. The lower limit for the gate voltage is determined by the maximum time required to sense the channel current which represents one state of the memory cell. For example, for fast access time, the maximum time would be in the range of 10 to 30 nanoseconds while for a mass storage device the maximum access time could be as high as 1 microsecond. The actual gate voltage to achieve these maximum times would depend upon the device structure, the dielectric thickness, the bit line capacitance, the doping concentration in the channel and other parameters associated with the device. An upper limit on the gate voltage is the voltage at which the voltage in the channel just beneath the edge of the region of trapped charge is just below the voltage potential applied to the source terminal during reading in the reverse direction. A too high gate voltage will cause inversion in the channel and the benefits of the present invention are lost. Thus, it is not recommended to apply a gate voltage that generates such a high voltage in the channel beneath the edge of the charge trapping region because it defeats the benefits of having a lower potential across the portion of the channel beneath this charge trapping region with the accompanying reduction in leakage current and shortened programming time. In a preferred embodiment of the present invention, the gate voltage used for reading is approximately 3V which represents an optimized tradeoff between programming time and leakage current.

The third optimization method, previously described and which is known in the art, is to vary the boron doping of the channel region under the gate. An increase in the doping concentration results in a higher threshold voltage V_T and a lower voltage generated in the channel. This is due to the reduction in the width of the depletion region formed.

Thus, a higher doping concentration permits a higher gate voltage to be applied for the same voltage across the portion of the channel beneath the charge trapping region.

In addition, an increase in the N_A doping concentration for the same length trapping region will improve the punch through behavior of the device. By varying the level of boron implanted in the channel region, the width of the depletion region under the gate can be varied. An increase in the doping concentration results in a reduction in the width of the depletion region for the same applied gate voltage. The reduction in the width of the depletion region occurs because there is now more fixed charge in the substrate. Thus, varying the doping concentration can be used to limit the length of the pinchoff region under the gate. In addition, the doping concentration can be used to increase or decrease the initial threshold voltage of the device.

Optimization parameters specific to programming and reading two bits in the memory cell of the present invention will now be described. The optimizations for programming include utilizing a longer minimum effective channel length Leer in order to physically separate the two bits better. In addition, the implant level can be reduced in the channel in order to increase the delta between forward and reverse programming. On the other hand, the implant level can be increased in the channel in order to reduce the impact of the first bit on the programming of the second bit. Thus, the implant level in the channel is a compromise between the forward and reverse delta on the one hand and the programming speed on the other hand.

The optimizations for reading include lowering the gate voltage in order to enhance the punch through during reading. As described previously, punch through is necessary to program and read the second bit. A lower implant level in the channel serves to increase punch through. Also, a higher drain voltage during read functions to increase punch through. These three optimizations relate to reading in the forward direction, which is equivalent to reading the second bit in the reverse.

In addition, a lower gate voltage reduces the number of electrons that need to be injected into the charge trapping region. This improves erasing because it eliminates residual charge remaining trapped after erasure. Any residual charge that remains in the charge trapping layer after erasure degrades cycling.

While the invention has been described with respect to a limited number of embodiments, it will be appreciated that many variations, modifications and other applications of the invention may be made.

CLAIMS

1. An electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, comprising:

a semiconducting substrate of a first conductivity type;

a first region comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate;

a second region, spaced from said first region, comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate, a channel being formed in the space between said first region and said second region within said semiconducting substrate;

a first insulating layer overlaying and covering said channel portion of said semiconducting substrate;

a nonconducting charge trapping layer formed on and overlaying said first insulating layer;

a second insulating layer formed on and overlaying said nonconducting charge trapping layer;

a gate comprising an electrically conductive material formed on and overlaying said second insulating layer;

wherein said charge trapping layer is formed so as to receive and retain electrons injected into said charge trapping layer in a charge storage region close to said first region forming a first bit, the quantity of electrons so stored being selected so as to provide said cell with a first threshold voltage greater than a first selected value when said memory cell is read in a first direction opposite to that in which it was programmed and to provide said cell with a second threshold voltage when said memory cell is read in a second direction which is the same direction in which it was programmed, said second threshold voltage being sufficiently less than said first threshold voltage as to allow said cell to store said first bit of information when the cell is read in the first direction but to not store said first bit of information with the same probability of detecting said first bit when the cell is read in the second direction; and

wherein said charge trapping layer is formed so as to receive and retain electrons injected into said charge trapping layer in a charge storage region close to said second region forming a second bit, the quantity of electrons so stored being selected so as to provide said cell with a third threshold voltage greater than a third selected value when said memory cell is read in a third direction opposite to that in which it was programmed and to provide said cell with a fourth threshold voltage when said memory cell is read in a fourth direction which is the same direction in which it was programmed, said fourth threshold voltage being sufficiently less than said third threshold voltage as to allow said cell to store said second bit of information when the cell is read in the third direction but to not store said second bit of information with the same probability of detecting said right bit when the cell is read in the fourth direction.

2. The memory cell according to claim 1, wherein said first bit and said second bit are erased by removing electrons from said region of trapped charge in said nonconducting charge trapping layer previously stored therein during programming.

3. The memory cell according to claim 1, wherein the amount of charge stored in said region of trapped charge for said first bit and said second bit in said non conducting charge trapping layer is such that the change in threshold voltage for reading in the reverse direction relative to reading in the forward direction is greater than a factor of ten.

4. The memory cell according to claim 1, wherein the amount of charge stored in said region of trapped charge for said first bit and said second bit in said non conducting charge trapping layer is such that the change in threshold voltage for reading in the reverse direction relative to reading in the forward direction is greater than a factor of five.

5. The memory cell according to claim 1, wherein the amount of charge stored in said region of trapped charge for said first bit and said second bit in said non conducting charge trapping layer is such that the change in threshold voltage for reading in the reverse direction relative to reading in the forward direction is greater than a factor of two.

6. The memory cell according to claim 1, wherein the amount of charge stored in said region of trapped charge for said first bit and said second bit in said nonconducting charge trapping layer is such that the change in threshold voltage for reading in the reverse direction relative to reading in the forward direction is greater than a factor of one and one half.

7. The memory cell according to claim 1, wherein a lower limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which sufficient inversion is generated in the channel whereby the unprogrammed state can be sensed and which permits read through of the bit not being read, an upper limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below the voltage applied to said first region during reading for said first bit and below the voltage applied to said second region during reading for said second bit.

8. The memory cell according to claim 1, wherein said cell is adapted such that programming said first bit comprises applying programming voltages to said first region and said gate, grounding said second region and measuring the resulting channel current, wherein reading said first bit comprises applying reading voltage to said second region and said gate, grounding said first region and measuring the resulting channel current, wherein said cell is adapted such that programming said second bit comprises applying programming voltages to said second region

and said gate, grounding said first region and measuring the resulting channel current and wherein reading said second bit comprises applying reading voltage to said first region and said gate, grounding said second region and measuring the resulting channel current.

9. The memory cell according to claim 1, wherein said first bit of said cell is adapted to be erased by applying erasing voltages to said gate and said first region and wherein said second bit of said cell is adapted to be erased by applying erasing voltages to said gate and said second region.

10. The memory cell according to claim 1, wherein said first bit of said cell is adapted to be erased by applying a first voltage on said gate and ground potential on said first region such that electrons are removed from said charge trapping region via said gate and wherein said second bit of said cell is adapted to be erased by applying a second voltage on said gate and ground potential on said second region such that electrons are removed from said charge trapping region via said gate.

11. The memory cell according to claim 1, wherein said first bit of said cell is adapted to be erased by applying a first voltage to said gate and a second voltage to said first region such that electrons are removed from said charge trapping region via said first region and wherein said second bit of said cell is adapted to be erased by applying a third voltage to said gate and a fourth voltage to said second region such that electrons are removed from said charge trapping region via said second region.

12. The memory cell according to claim 1, wherein said first and second insulating layers comprise silicon dioxide.

13. The memory cell according to claim 1, wherein said charge trapping layer comprises silicon nitride.

14. The memory cell according to claim 1, wherein said charge trapping layer comprises silicon dioxide with buried polysilicon islands.

15. The memory cell according to claim 1 wherein said charge trapping layer comprises silicon dioxide with impurities therein.

16. The memory cell according to claim 1, wherein said semiconducting substrate comprises P type semiconductor material.

17. The memory cell according to claim 1, wherein said source and said drain comprise N+ semiconductor material.

18. An electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, comprising:

a semiconducting substrate of a first conductivity type;

a first region comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate;

a second region, spaced from said first region, comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate, a channel being formed between said first region and said second region within said semiconducting substrate;

a first insulating layer overlaying and covering said channel portion of said semiconducting substrate;

a non conducting charge trapping layer formed on and overlaying said first insulating layer;

a second insulating layer formed on and overlaying said non conducting charge trapping layer;

a gate comprising an electrically conductive material formed on and overlaying said second insulating layer;

wherein said cell is adapted to receive and retain electrons injected into said non conducting charge trapping layer in a portion close to said first region forming a first bit and in a portion close to said second region forming a second bit; and

wherein said memory cell is adapted to be read in a direction opposite to that in which it was programmed, such that a lower limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which sufficient inversion is generated in said channel whereby an unprogrammed state can be sensed and which permits read through of the bit not being read, an upper limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below the voltage applied to said first region during reading for said first bit and below the voltage applied to said second region during reading for said second bit.

19. The memory cell according to claim 18, wherein said first bit and said second bit of said cell are erased by removing electrons from respective regions of trapped charge in said non conducting charge trapping layer previously stored therein during the programming of each bit.

20. The memory cell according to claim 18, wherein said first bit of said cell is adapted to be programmed by applying programming voltages to said first region and said gate, grounding said second region and measuring the resulting channel current, wherein said first bit of said cell is adapted to be read by applying reading voltages to said second region and said gate, grounding said first region and measuring the resulting channel current, wherein said second bit of said cell is adapted to be programmed by applying programming voltages to said second region and said gate, grounding said first region and measuring the resulting channel current, wherein said second bit of said cell is adapted to be read by applying reading voltages to said first region and said gate, grounding said second region and measuring the resulting channel current.

21. The memory cell according to claim 18, wherein said first bit of said cell is adapted to be erased by applying erasing voltages to said gate and said first region and wherein said second bit of said cell is adapted to be erased by applying erasing voltages to said gate and said second region.

22. The memory cell according to claim 18, wherein said first bit of said cell is adapted to be erased by applying a first voltage to said gate and ground to said first region such that electrons are removed from said charge trapping region via said gate and wherein said second bit of said cell is adapted to be erased by applying a second voltage to said gate and ground to said second region such that electrons are removed from said charge trapping region via said gate.

23. The memory cell according to claim 18, wherein said first bit of said cell is adapted to be erased by applying a first voltage to said gate and a second voltage to said first region such that electrons are removed from said charge trapping region via said first region and wherein said second bit of said cell is adapted to be erased by applying a third voltage to said gate and a fourth voltage to said second region such that electrons are removed from said charge trapping region via said second region.

24. The memory cell according to claim 18, wherein said first and second insulating layers comprise silicon dioxide.

25. The memory cell according to claim 18, wherein said charge trapping layer comprises silicon nitride.

26. The memory cell according to claim 18, wherein said charge trapping layer comprises silicon dioxide with buried polysilicon islands.

27. The memory cell according to claim 18, wherein said charge trapping layer comprises silicon dioxide with impurities therein.

28. The memory cell according to claim 18, wherein said semiconducting substrate comprises P-type semiconductor material.

29. The memory cell according to claim 18, wherein said source and said drain comprise N+ semiconductor material.

30. An electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, comprising:

a semiconducting substrate of a first conductivity type;

a first region comprising a portion of said semiconducting substrate doped to have a conductivity type opposite that of said semiconducting substrate;

a second region, spaced from said first region, comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate, a channel being formed between said first region and said second region within said semiconducting substrate;

a first insulating layer overlaying and covering said channel portion of said semiconducting substrate;

a non conducting charge trapping layer formed on and overlaying said first insulating layer;

a second insulating layer formed on and overlaying said non conducting charge trapping layer;

a gate comprising an electrically conductive layer formed on and overlaying said second insulating layer;

wherein said memory cell is adapted to be programmed by hot electron injection into said non conducting charge trapping layer in a region close to said first region forming a first bit and in a region close to said second region forming a second bit;

wherein said memory cell is adapted to be read in a manner opposite from that in which it was programmed, said cell being adapted such that a lower limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which sufficient inversion is generated in said channel whereby an unprogrammed state can be sensed and which permits read through of the bit not being read, an upper limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below the voltage applied to said first region during reading for said first bit and below the voltage applied to said second region during reading for said second bit; and

wherein said cell is adapted to be erased by applying ground potential to said gate and removing electrons from said respective regions of trapped charge in said non conducting charge trapping layer previously stored therein during programming via said first region to erase said first bit and via said second region to erase said second bit.

31. An electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, comprising:

a semiconducting substrate of a first conductivity type;

a first region comprising a portion of said semiconducting substrate doped to have a conductivity type opposite that of said semiconducting substrate;

a second region, spaced from said first region, comprising a portion of said semiconducting substrate doped to have a conductivity type opposite that of said semiconducting substrate, a channel being formed in the space between said first region and said second region within said semiconducting substrate;

a first insulating layer overlaying and covering said channel portion of said semiconducting substrate;

a non conducting charge trapping layer formed on and overlaying said first insulating layer;

a second insulating layer formed on and overlaying said non conducting charge trapping layer;

a gate comprising an electrically conductive layer formed on and overlaying said second insulating layer;

wherein said memory cell is programmed by hot electron injection into a region of said non conducting charge trapping layer close to said first region forming a first bit and close to said second region forming a second bit;

wherein said memory cell is adapted to be read in a manner opposite from that in which it was programmed, said cell being adapted such that a lower limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which sufficient inversion is generated in said channel whereby an unprogrammed state can be sensed and which permits read through of the bit not being read, an upper limit for the voltage applied to said gate during reading for either said first bit or said second bit is the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below the voltage applied to said first region during reading for said first bit and below the voltage applied to said second region during reading for said second bit; and

wherein said cell is adapted to be erased by applying a negative potential to said gate and removing electrons from said respective regions of trapped charge in said non conducting charge trapping layer previously stored therein during programming via said first region to erase said first bit and via said second region to erase said second bit.

32. A method of programming, reading and erasing an electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, said memory cell having a first region and a second region with a channel therebetween and a gate above said channel but separated therefrom by a non conducting charge trapping material sandwiched between first and second silicon dioxide layers, said method comprising the steps of:

programming in the forward direction by injecting electrical charge into said charge trapping material utilizing hot electron injection for a sufficient time that electrical charge becomes trapped asymmetrically in a charge trapping region of said charge trapping material close to said first region forming a first bit and close to said second region forming a second bit, said electrical charge being injected until the threshold voltage of said gate reaches a predetermined level when said memory cell is read in the reverse direction from which it was programmed, said asymmetrical charge injection for said first bit being generated by applying programming voltages to said first region and said gate and grounding said second region, said asymmetrical charge injection for said second bit being generated by applying programming voltages to said second region and said gate and grounding said first region;

reading said first bit in the reverse direction by applying read voltages to said second region and said gate and grounding said first region, and subsequently sensing whether or not current flows through said memory cell from said second region to said first region;

reading said second bit in the reverse direction by applying read voltages to said first region and said gate and grounding said second region, and subsequently sensing whether or not current flows through said memory cell from said first region to said second region;

erasing said first bit of said memory cell by applying erasing voltages to said gate and said first region so as to cause electrons to be removed from said charge trapping region; and

erasing said second bit of said memory cell by applying erasing voltages to said gate and said second region so as to cause electrons to be removed from said charge trapping region.

33. The method according to claim 32, wherein the gate voltage during reading in the reverse direction for either said first bit or said second bit is between the gate voltage at which sufficient inversion is generated in the channel whereby the unprogrammed state can be sensed and which permits read through of the bit not being read and the gate voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below the voltage applied to said first region during reading for said first bit and below the voltage applied to said second region during reading for said second bit.

34. A method of programming, reading and erasing an electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, said memory cell having a first region, a second region spaced from said first region, a channel between said first region and said second region and a gate and utilizing a non conducting charge trapping material sandwiched between first and second silicon dioxide layers formed between said gate and said channel, said method comprising the steps of:

programming in the forward direction by injecting electrical charge into said charge trapping material utilizing hot electron injection for a sufficient time that electrical charge becomes trapped asymmetrically in a charge trapping region of said charge trapping material close to said first region forming a first bit and close to said second region forming a second bit, said electrical charge being injected until the threshold voltage of said gate reaches a predetermined level when said memory cell is read in the reverse direction from which it was programmed, said asymmetrical charge injection for said first bit being generated by applying programming voltages to said first region and said gate and grounding said second region, said asymmetrical charge injection for said second bit being generated by applying programming voltages to said second region and said gate and grounding said first region;

reading said first bit in the reverse direction by applying read voltages to said second region and said gate and grounding said first region, and subsequently sensing whether or not current flows through said memory cell from said second region to said first region;

reading said second bit in the reverse direction by applying read voltages to said first region and said gate and grounding said second region, and subsequently sensing whether or not current flows through said memory cell from said first region to said second region;

erasing said first bit of said memory cell by applying a selected potential to said gate so as to cause electrons to be removed from said charge trapping region via said first region; and

erasing said second bit of said memory cell by applying a selected potential to said gate so as to cause electrons to be removed from said charge trapping region via said second region.

35. The method according to claim 34, wherein the selected potential applied to said gate is ground potential.

36. The method according to claim 34, wherein the selected potential applied to said gate is a negative potential.

37. A method of programming, reading and erasing an electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, said memory cell having a semiconducting substrate of a first conductivity type, a first region of a second conductivity type opposite to said first conductivity type and forming a first bit, a second region of said second conductivity type, said second region being spaced from said first region and forming a second bit, a channel formed in said substrate between said first region and said second region, a conductive gate and a non conducting charge trapping material sandwiched between first and second silicon dioxide layers formed between said gate and said channel, said method comprising the steps of:

programming said first bit in the forward direction by: applying a first programming voltage to said gate; applying a second programming voltage to said first region; and coupling said second region to ground;

thereby to inject electrical charge into said charge trapping material utilizing hot electron injection for a time sufficient to cause enough electrical charge to become trapped asymmetrically in a charge trapping region of said charge trapping material in close vicinity to said first region such that the threshold voltage of said cell is at least at a predetermined level when said memory cell is read in the reverse direction from which it was programmed;

programming said second bit in the forward direction by:

applying a third programming voltage to said gate;

applying a fourth programming voltage to said second region; and coupling said first region to ground;

thereby to inject electrical charge into said charge trapping material utilizing hot electron injection for a time sufficient to cause enough electrical charge to become trapped asymmetrically in a charge trapping region of said charge trapping material in close vicinity to said second region such that the threshold voltage of said cell is at least at a predetermined level when said memory cell is read in the reverse direction from which it was programmed;

reading said first bit in the reverse direction by:

applying a first read voltage to said gate;

applying a second read voltage to said second region; coupling said first region to ground; sensing whether or not current flows through said memory cell from said second region to said first region;

wherein said first read voltage is between the voltage at which sufficient inversion is generated in the channel whereby the unprogrammed state can be sensed and which permits read through of the bit not being read and the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below said second read voltage;

reading said second bit in the reverse direction by:

applying a third read voltage to said gate;

applying a fourth read voltage to said first region;

coupling said second region to ground;

sensing whether or not current flows through said memory cell from said first region to said second region;

wherein said third read voltage is between the voltage at which sufficient inversion is generated in the channel whereby the unprogrammed state can be sensed and which permits read through of the bit not being read and the voltage at which the voltage across a region of said channel beneath the trapped charge in said charge storage region is below said fourth read voltage;

erasing said first bit of said memory cell by:

applying a first erase voltage to said gate;

applying a second erase voltage to said first region;

whereby said first and second erase voltages are sufficient to cause electrons to be removed from said charge trapping region;

erasing said second bit of said memory cell by:

applying a third erase voltage to said gate;

applying a fourth erase voltage to said second region; and

whereby said third and fourth erase voltages are sufficient to cause electrons to be removed from said charge trapping region.

38. An electrically erasable programmable read only memory (EEPROM) cell capable of storing two bits of information, comprising:

a semiconductor substrate of a first conductivity type containing therein a first region and a second region each of a second conductivity type opposite to said first conductivity type and separated from each other by a channel region normally of said first conductivity type;

a dielectric formed over said channel region, said dielectric being capable of holding selected charge in a portion thereof above but adjacent to said first region forming a first bit and in a portion thereof above but adjacent to said second region forming a second bit, said dielectric including a first layer of silicon oxide, a second layer of silicon oxide and a charge trapping material formed between said first layer of silicon oxide and said second layer of silicon oxide;

a gate comprising an electrically conductive layer formed on and overlaying said dielectric;

a first voltage source capable of being connected to said first region, a second voltage source capable of being connected to said gate and a third voltage source capable of being connected to said second region;

a first control both for causing said first voltage source to apply a first voltage to said first region, said second voltage source to apply a second voltage to said gate, and said third voltage source to apply a third voltage to said second region, thereby to cause electrons to be injected by hot electron injection into a portion of said dielectric close to said first region and for causing said first voltage source to apply a fourth voltage to said first region, said second voltage source to apply a fifth voltage to said gate and said third voltage source to apply a sixth voltage to said second region thereby to cause said memory cell to read said first bit in the reverse direction from the direction in which the first bit was programmed; and

a second control both for causing said first voltage source to apply a seventh voltage to said second region, said second voltage source to apply an eighth voltage to said gate, and said third voltage source to apply a ninth voltage to said first region, thereby to cause electrons to be injected by hot electron injection into a portion of said dielectric close to said second region and for causing said first voltage source to apply a tenth voltage to said second region, said second voltage source to apply an eleventh voltage to said gate and said third voltage source to apply a twelfth voltage to said first region thereby to cause said memory cell to read said second bit in the reverse direction from the direction in which the second bit was programmed.

39. The memory cell according to claim 38, wherein said charge trapping material comprises silicon nitride, said layer of silicon nitride being of such a thickness as to receive and retain a selected charge in a localized portion of said silicon nitride near said first region for said first bit and near said second region for said second bit, and wherein the remainder of said dielectric is adapted to assist in retaining the charge in the silicon nitride in said portion of the silicon nitride in which the charge is formed.

40. The memory cell according to claim 39, wherein said first layer of silicon oxide formed between said silicon nitride layer and said semiconductor substrate, said second layer of silicon oxide formed between said silicon nitride layer and said gate.

41. The memory cell according to claim 38, wherein said semiconductor substrate comprises silicon, said first layer of silicon oxide formed by thermally oxidizing said semiconductor substrate.

42. The memory cell according to claim 38, wherein said charge trapping region comprises silicon nitride, said second layer of silicon oxide formed at least in part by thermally oxidizing a top portion of the layer of said silicon nitride.

43. The memory cell according to claim 38, wherein said second layer of silicon oxide comprises at least a portion formed from the deposition of silicon dioxide.

44. The memory cell according to claim 38, wherein said second layer of silicon oxide formed at least in part by the deposition of silicon dioxide from TEOS.

45. The memory cell according to claim 39, wherein said second layer of silicon oxide formed by at least one of the following: thermal oxidation of the top surface of said silicon nitride layer, the chemical vapor deposition of silicon dioxide from selected reactants or the deposition of silicon dioxide from the decomposition of TEOS.

46. The memory cell according to claim 38, wherein said dielectric comprises a selected layer of silicon dioxide formed with selected pockets of polycrystalline silicon dispersed therein, said pockets of polycrystalline silicon being capable of retaining a charge lodged in said polycrystalline silicon.

47. The memory cell according to claim 38, wherein said dielectric comprises selected impurities which are capable of retaining a charge.

48. The memory cell according to claim 38, wherein said dielectric comprises a pocket of polycrystalline silicon located adjacent the drain region of said cell, said pocket of polycrystalline silicon being capable of storing trapped charges.

49. The memory cell according to claim 38, wherein said dielectric comprises at least a portion thereof formed by depositing silicon dioxide in a silicon rich environment such that precipitates of silicon form randomly but are dispersed throughout a portion of the silicon dioxide layer, said precipitates of silicon serving to hold trapped charge injected into said portion of said silicon dioxide layer.

50. A semiconductor memory cell capable of storing two bits of information, comprising:

a substrate of a first conductivity type including a first region of a second conductivity type opposite said first conductivity type and a second region of said second conductivity type, said first region and second region being respectively spaced from each other by a channel region normally of said first conductivity type formed therebetween;

a dielectric capable of holding an electrical charge in a charge trapping region thereof formed over said channel region near said first region forming a first bit and near said second region forming a second bit;

a conductive gate formed over said dielectric;

means for applying a first voltage to said first region and a second voltage to said gate region thereby to cause electrons to be lodged on and stored in said charge trapping region near said first region to form said first bit;

means for applying a third voltage to said second region and said second voltage to said gate region thereby to cause electrons to be lodged on and stored in said charge trapping region near said second region to form said second bit;

means for applying a fourth voltage to said second region and a fifth voltage to said gate thereby to cause a current to be read indicating the presence or absence of a stored charge in said dielectric near said first region representing said first bit, the fifth voltage being between a sixth voltage sufficient to invert said channel with no charge in said charge trapping region near said first region and a seventh voltage sufficient to create a voltage beneath said fourth voltage at a point in said channel beneath an edge of said charge trapping region with charge stored therein; and

means for applying an eighth voltage to said first region and a ninth voltage to said gate thereby to cause a current to be read indicating the presence or absence of a stored charge in said dielectric near said second region representing said second bit, the ninth voltage being between a tenth voltage sufficient to invert said channel with no charge in said charge trapping region near said second region and an eleventh voltage sufficient to create a voltage beneath said ninth voltage at a point in said channel beneath an edge of said charge trapping region with charge stored therein.

51. The memory cell according to claim 50, wherein said dielectric further comprises:

a first layer of silicon dioxide formed on said substrate;

a layer of silicon nitride formed on said first layer of silicon dioxide, said layer of silicon nitride being capable of retaining charge in a first selected portion thereof forming said first bit and in a second selected portion thereof forming said second bit; and

a second layer of silicon dioxide formed on said silicon nitride, said second layer of silicon dioxide assisting said silicon nitride in retaining the charge trapped in the charge trapping region thereof despite the electric fields generated therein.

52. .A method of operating a non-volatile electrically erasable and programmable semiconductor memory cell capable of storing two bits of information utilizing asymmetrical charge trapping, said memory cell comprising a semiconductor substrate of a first conductivity type having formed therein a first region and a second region each of a second conductivity type opposite the said first conductivity type, said memory cell further having formed therein a channel between said first and second regions, a dielectric overlying said channel, said dielectric including at least a silicon nitride layer for the capture and retention of localized charge in a first portion of said silicon nitride layer closest to and above said first region forming a first bit and in a second portion of said silicon nitride layer closest to and above said second region forming a second bit and a conductive gate overlying said dielectric, said method comprising the steps of:

placing charge on said first portion and said second portion of said silicon nitride layer;

applying a first voltage to said second region greater than the voltage on said first region and applying a second voltage to said conductive gate, said second voltage being less than the voltage across the portion of said channel beneath said first portion of said silicon nitride layer holding said charge, said second voltage causing a first current to be read by said device when no charge has been placed on said first portion of said silicon nitride layer and causing a second current, less than said first current, or no current, to be read when a localized charge has been placed on said first portion of said silicon nitride layer, said localized charge being substantially less than the localized charge required to achieve the same threshold voltage for the device when the device is to be read by applying a voltage to the second region and a voltage to the gate region; and

applying a third voltage to said first region greater than the voltage on said second region and applying a fourth voltage to said conductive gate, said fourth voltage being less than the voltage across the portion of said channel beneath said second portion of said silicon nitride layer holding said charge, said fourth voltage causing a third current to be read by said device when no charge has been placed on said second portion of said silicon nitride layer and causing a fourth current, less than said third current, or no current, to be read when a localized charge has been placed on said second portion of said silicon nitride layer, said localized charge being substantially less than the localized charge required to achieve the same threshold voltage for the device when the device is to be read by applying a voltage to the first region and a voltage to the gate region.

53. An electrically erasable programmable read only memory (EEPROM) cell capable storing two bits of information, comprising:

a semiconducting substrate of a first conductivity type;

a first region comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate;

a second region spaced from said first region, comprising a portion of said semiconducting substrate doped to have a conductivity opposite that of said semiconducting substrate;

a channel formed in the space between said first region and said second region within said semiconducting substrate;

a first insulating layer overlaying and covering said channel;

a non conducting charge trapping layer formed on and overlaying said first insulating layer;

a second insulating layer formed on and overlaying said non conducting charge trapping layer;

a gate comprising an electrically conductive material formed on and overlaying said second insulating layer;

wherein said charge trapping layer is formed so as to receive and retain a first selected amount of charge in a region of said non conducting charge trapping layer close to and above said first region forming a first bit and a second selected amount of charge in a region of said non conducting charge trapping layer close to and above said second region forming a second bit, said charge trapping layer comprising a layer of silicon nitride having a thickness selected to ensure that the lateral electric field associated with the trapped charge forming either said first bit or said second bit is below the lateral electric field which would cause significant lateral diffusion of the stored charge and said first selected amount of charge and said second selected amount of charge is sufficient to cause a desired increase in the threshold voltage required to invert said channel when said cell is read in the reverse direction but is not sufficient to cause the same desired increase in the threshold voltage required to invert the channel when the cell is read in the forward direction.

54. The memory cell according to claim 53, wherein the thickness of the silicon nitride is selected to reduce the lateral electric field within the charge storage region associated with each of said first bit and said second bit to beneath a selected value thereby to reduce the lateral diffusion of the stored charge in the silicon nitride and thereby to limit the reduction in the threshold voltage of the corresponding portion of the channel beneath said charge storage region due to this lateral diffusion to less than a selected amount.

55. The memory cell according to claim 54, wherein the reduction in the threshold voltage of the portion of the channel beneath said charge storage region for each of said first bit and said second bit due to lateral diffusion of the stored charge is less than ten percent of the threshold voltage of the corresponding portion of the channel before the lateral diffusion.

56. The memory cell according to claim 54, wherein the reduction in the threshold voltage of the portion of the channel beneath said charge storage region for each of said first bit and said second bit due to lateral diffusion of the stored charge is less than five percent of the threshold voltage of the corresponding portion of the channel before the lateral diffusion.

57. An electrically erasable programmable read only memory cell capable of storing two bits of information, comprising:

a semiconductor substrate of a first conductivity type;

a first region and a second region of a second conductivity type opposite to said first conductivity type formed in said semiconductor substrate and separated from each other by a channel region;

a multi-layer dielectric formed over said channel region between said first region and said second region, said multi-layer dielectric having a first end adjacent said first region and a second adjacent said second region;

a conductive gate formed over said multi-layer dielectric thereby to control the voltage in said channel region; and a first amount of charge trapped at said first end of said multi-layer dielectric and a second amount of charge trapped at said second end of said dielectric, said first amount of charge and said second amount of charge together representing two bits of information capable of being stored in said cell.

58. Structure as in Claim 57 wherein said first amount of charge represents a binary 1 and said second amount of charge represents a binary 0.

59. Structure as in Claim 57 wherein said first amount of charge represents a binary 0 and said second amount of charge represents a binary 1.

60. Structure as in Claim 57 wherein said first amount of charge represents a binary 1 and said second amount of charge represents a binary 1.

61. Structure as in Claim 57 wherein said first amount of charge represents a binary 0 and said second amount of charge represents a binary 0.

62. Structure as in Claim 57 wherein said first amount of charge in said multi-layer dielectric is self-aligned to a junction between said first region and said substrate and said second amount of charge in said multi-layer dielectric is self-aligned to a junction between said second region and said substrate.

63. Structure as in Claim 62 wherein the width of said first amount of charge trapped in said multi-layer dielectric and the width of said second amount of charge trapped in said multi-layer dielectric is such that punch through occurs in the channel beneath said first amount of charge when the state of said second amount

of charge is being read from said memory cell and punch through of the channel region beneath said second amount of charge occurs when the state of said first amount of charge is being read from said memory cell.

64. Structure as in Claim 57 wherein said first amount of charge is capable of being erased by the application of a low voltage to said conductive gate and a positive voltage to said first region and said second amount of charge is capable of being erased by the application of a low voltage to said conductive gate and a positive voltage to said second region.

65. Structure as in Claim 57 wherein the state of said first amount of charge is read by applying a first selected voltage to said conductive gate and a second selected voltage to the second region, while said first region is held at ground thereby to cause the channel between said first region and said second region to conduct a current above a threshold level when said first amount of charge is below a given amount and to not conduct a current above said threshold level when said first amount of charge is above said given amount.

66. Structure as in Claim 57 wherein the state of said second amount of charge trapped at said second end of said dielectric is read by applying a first selected voltage to said conductive gate and a second selected voltage to said first region while said second region is held at ground thereby to cause the channel between said first region and said second region to conduct a current above a threshold level when said second amount of charge is below a given amount and not

conduct a current above said threshold level when said second amount of charge is above said given amount.

67. The method of storing two bits of information in a memory cell, the memory cell comprising a semiconductor substrate of a first conductivity type with a first region and a second region of second conductivity type opposite to said first conductivity type formed therein, a channel in said substrate separating said first region from said second region; a multi-layer dielectric having a first end and a second end formed over said channel and a conductive layer formed over said multi-layer dielectric, said method comprising:

placing a first selected charge at said first end of said dielectric and a second selected charge at said second end of said dielectric, wherein the first selected charge and the second selected charge represent a first bit and a second bit capable of being stored in said memory cell, and the state of said first bit in said memory cell is read by:

applying a first selected voltage to said second region;

applying a second selected voltage to said conductive gate; and

applying a ground potential to said first region;

wherein a current above a threshold current flows in said channel when said first selected charge is below a selected amount and no current or a current beneath said threshold current flows in said channel when the first selected charge is above said selected amount; and wherein the state of said second bit in said memory cell is read by:

applying said first selected voltage to said first region;

applying said second selected voltage to said conductive gate; and

applying a ground potential to said second region;

wherein a current above a threshold current flows in said channel when said second selected charge is below said selected amount and no current or a current beneath said threshold current flows in said channel when said second selected charge is above said selected amount.

68. The method of Claim 67 wherein said first amount of charge and said second amount of charge stored in said memory cell are erased by:

applying a relatively low voltage to said conductive gate; and

applying a positive voltage to said first region thereby to remove the first amount of charge trapped at the first end of said multi-layer dielectric; and

applying a relatively low voltage to said conductive gate; and

applying a positive voltage to said second region thereby to remove the second amount of charge stored at the second end of said multi-layer dielectric.

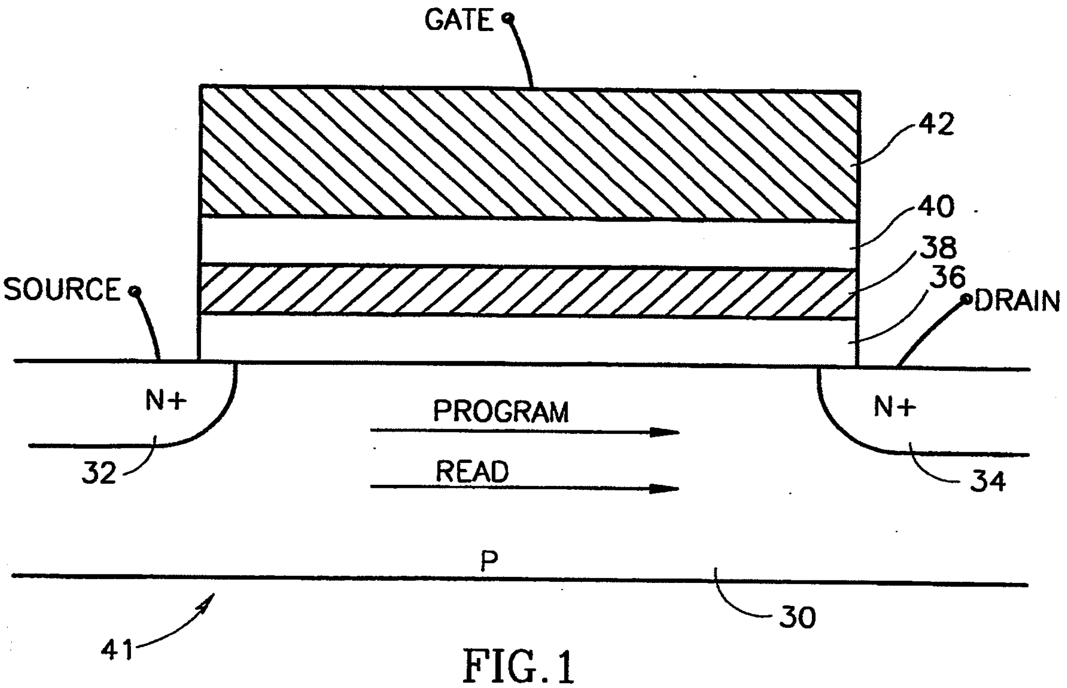

GATE
42
40
38
36
SOURCE
DRAIN
N+
N+
PROGRAM
READ
32
34
P
30
41

FIG. 1
PRIOR ART

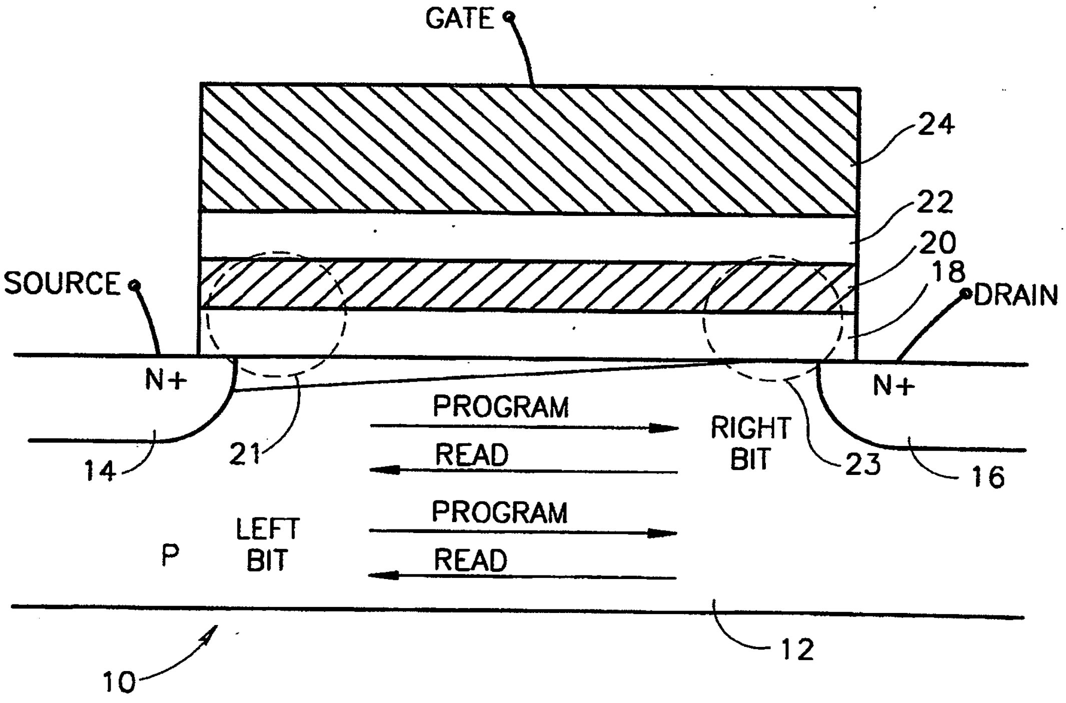

GATE
24
22
20
18
SOURCE
DRAIN
N+
N+
PROGRAM
READ
RIGHT BIT
14
21
23
16
P
LEFT BIT
PROGRAM
READ
12
10

FIG. 2

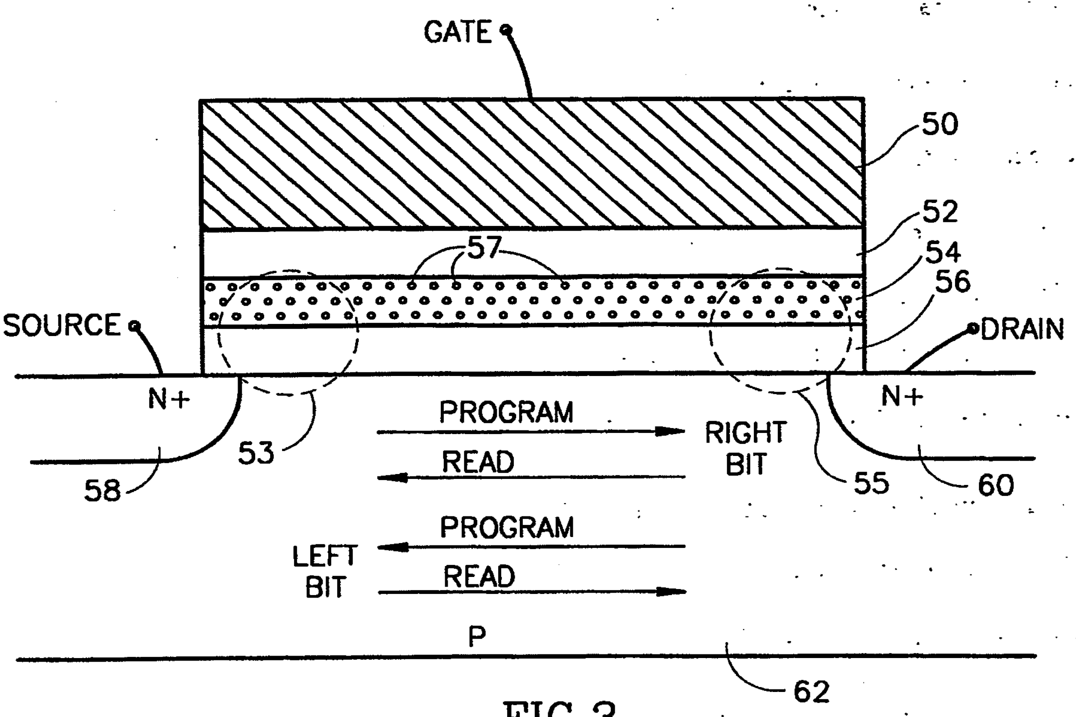

GATE
50
52
57
54
56
SOURCE
DRAIN
N+
N+
53
58
PROGRAM
READ
RIGHT
BIT
55
60
PROGRAM
LEFT
BIT
READ
P
62

FIG.3

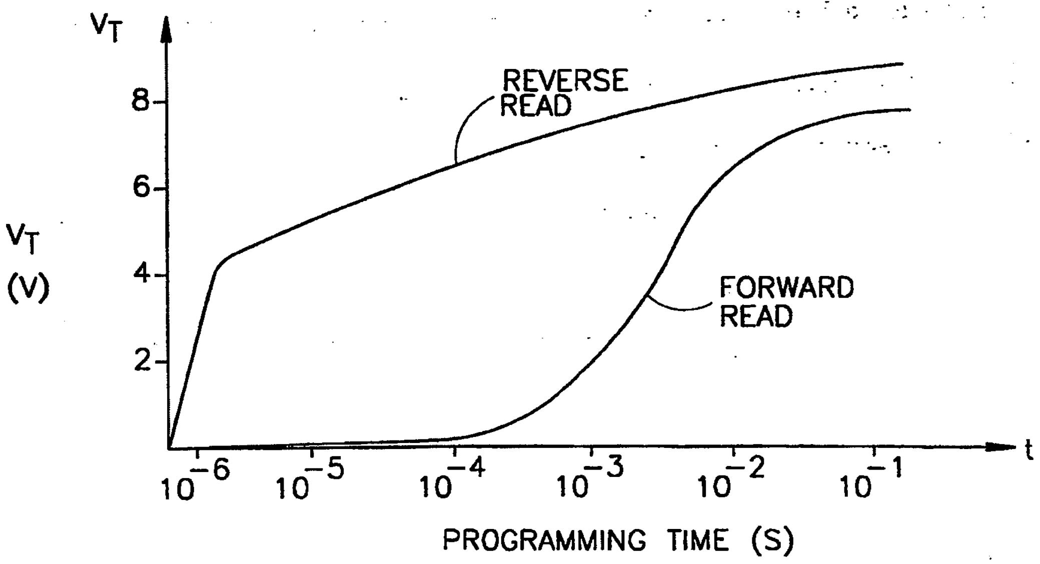

V_T
8
6
4
2
V_T
(V)
REVERSE
READ
FORWARD
READ
t
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
PROGRAMMING TIME (S)

FIG.4

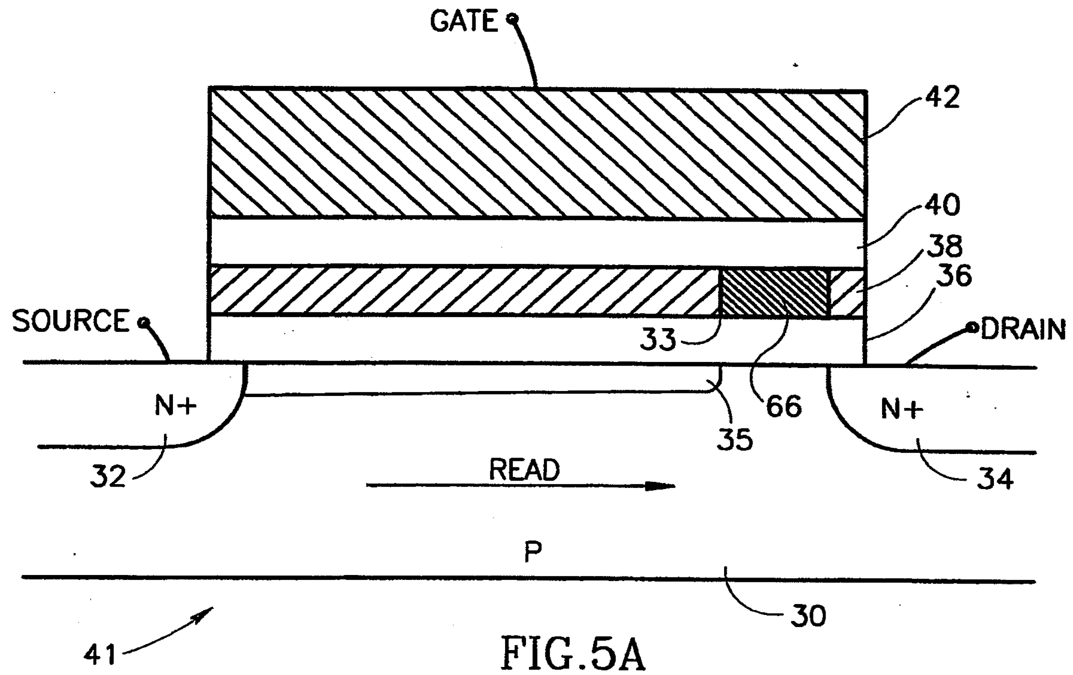

GATE
42
40
38
36
SOURCE
DRAIN
33
66
N+
N+
35
32
34
READ
P
30
41

FIG.5A
PRIOR ART

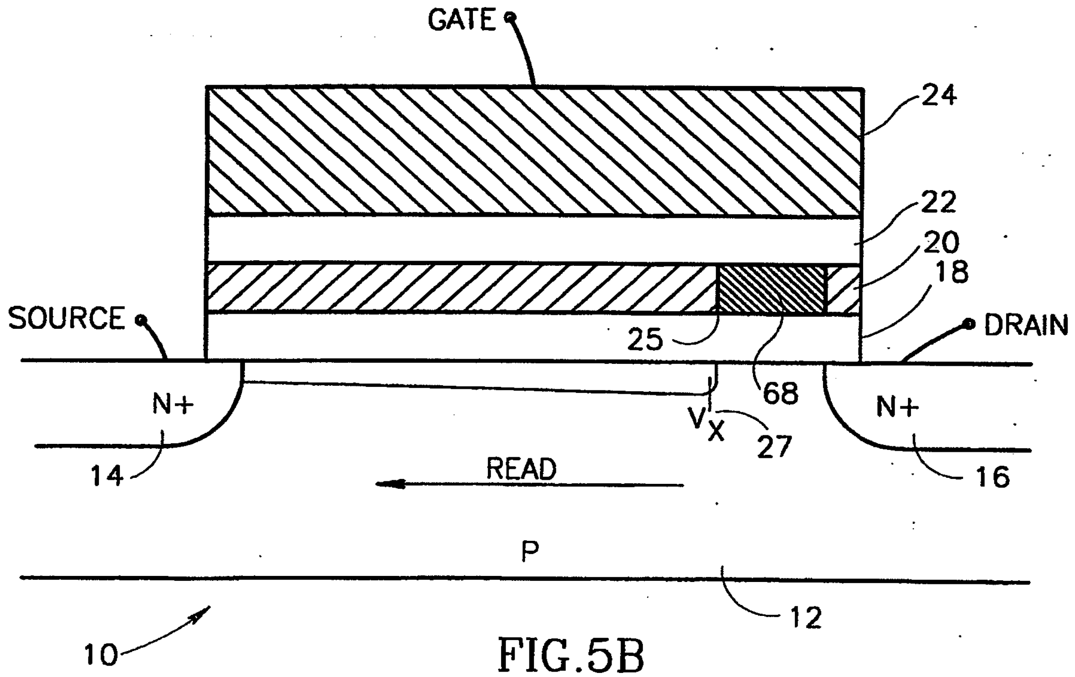

GATE
24
22
20
18
SOURCE
DRAIN
25
68
N+
N+
V_X
27
14
16
READ
P
12
10

FIG.5B

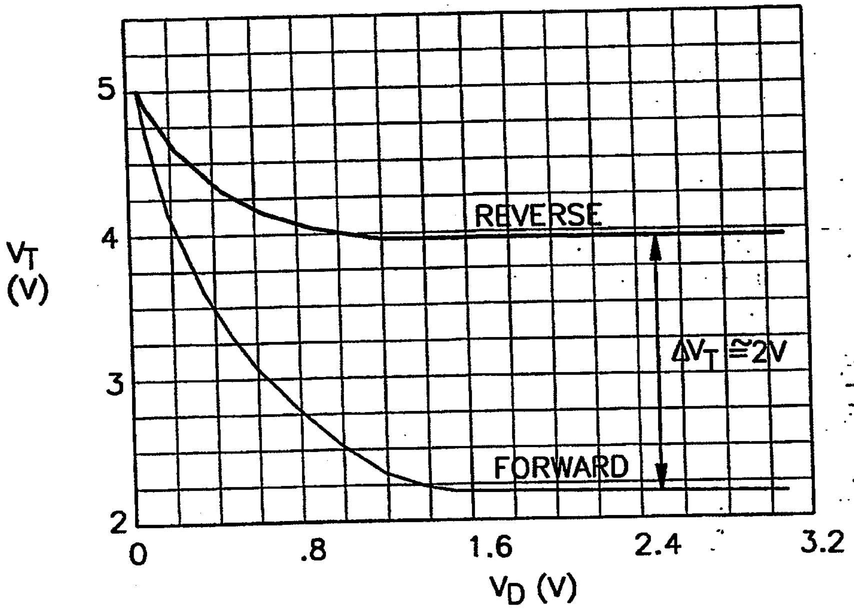

5
4
3
2
V_T (V)
REVERSE
FORWARD
$\Delta V_T \cong 2V$
0
.8
1.6
2.4
3.2
V_D (V)

FIG.6

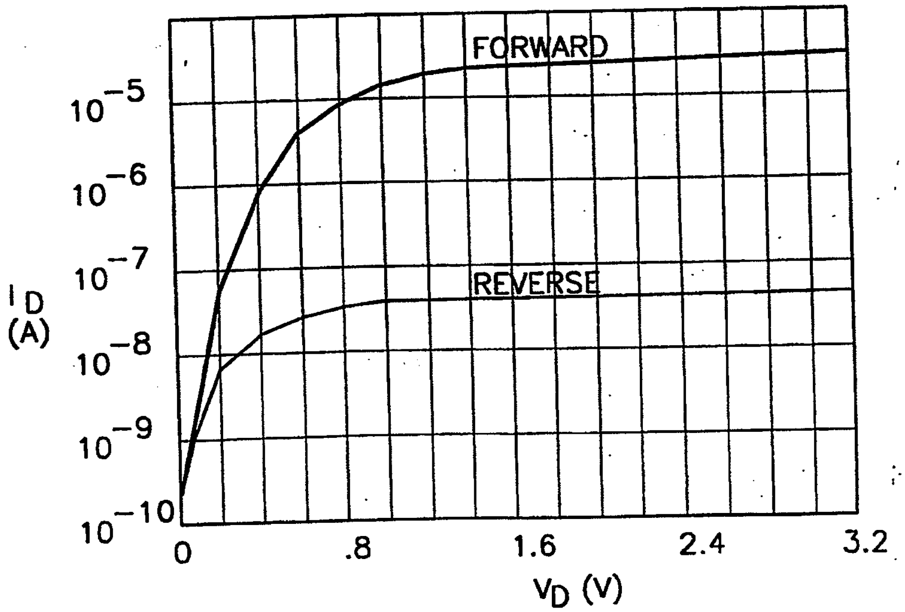

10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
I_D (A)
FORWARD
REVERSE
0
.8
1.6
2.4
3.2
V_D (V)

FIG.7

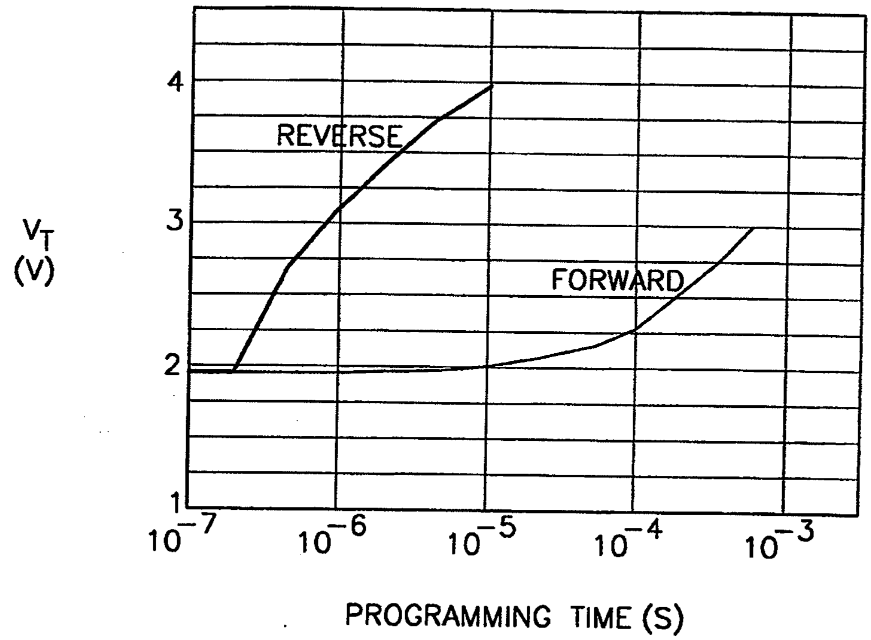
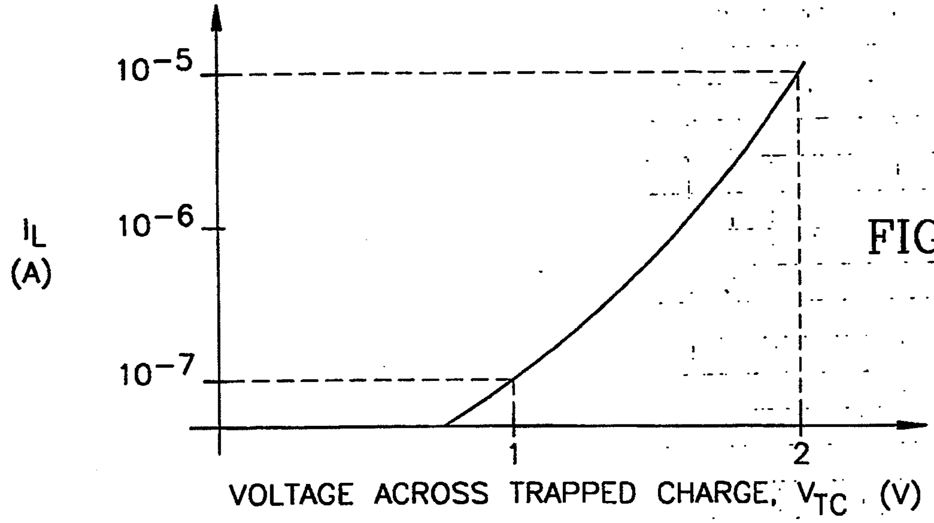
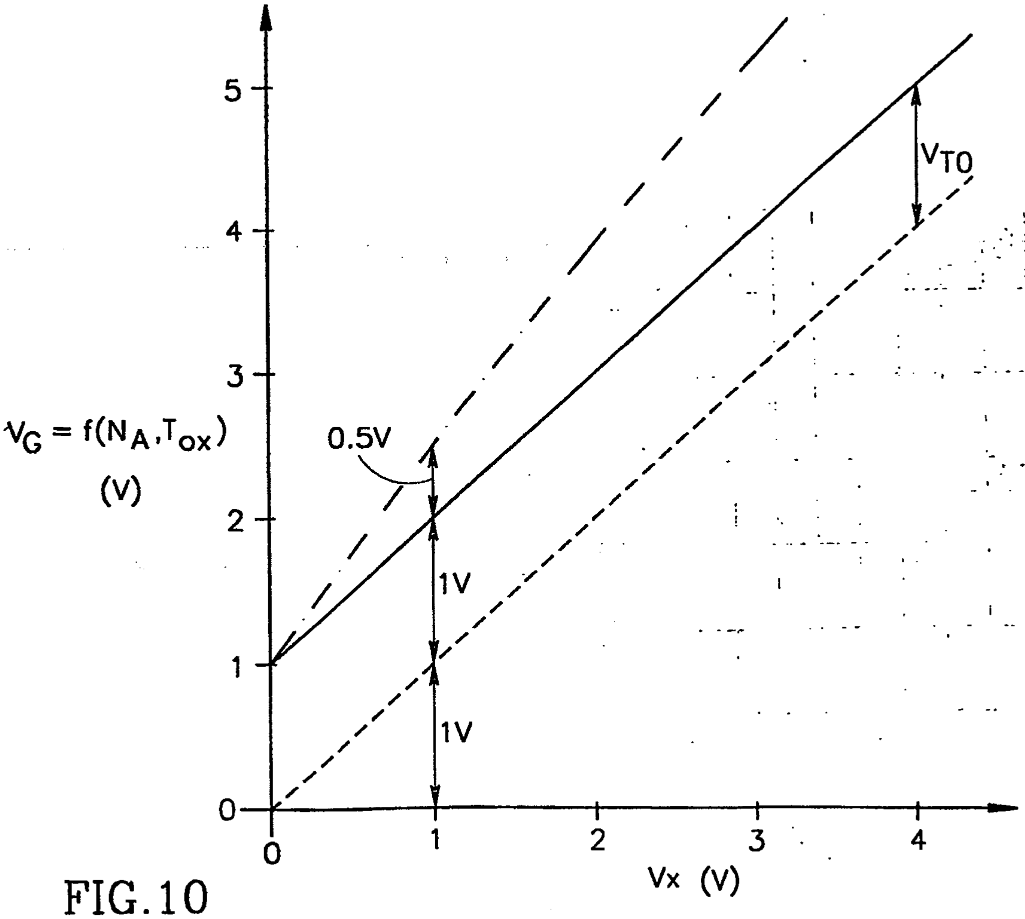

4
3
2
1
V_T
(V)
REVERSE
FORWARD
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
PROGRAMMING TIME (S)

FIG.8



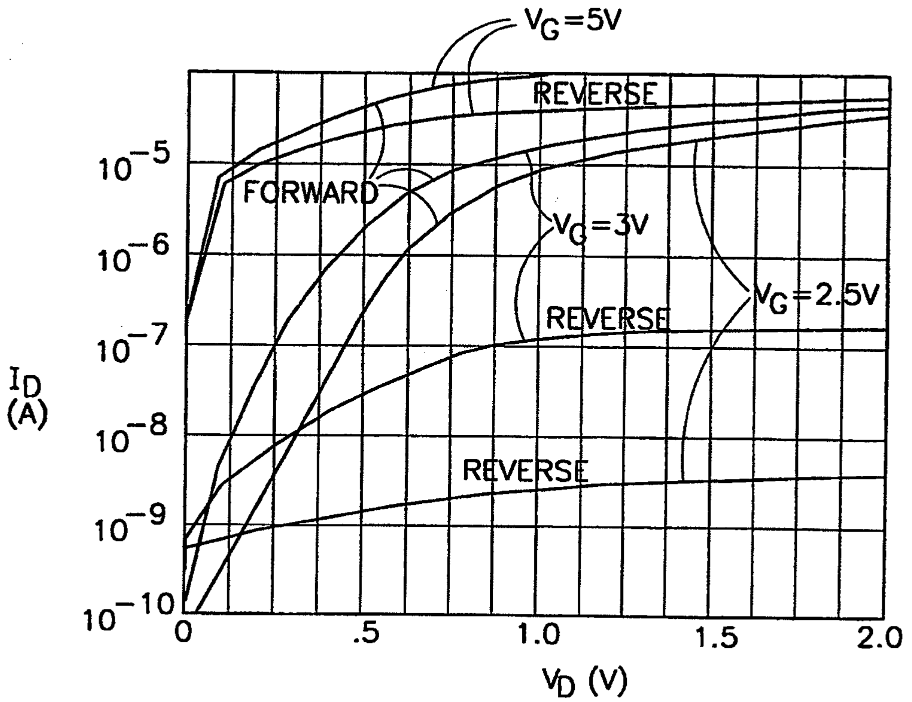
I_L (A)
10^{-5}
10^{-6}
10^{-7}
1
2
VOLTAGE ACROSS TRAPPED CHARGE, V_{TC} (V)

FIG.9



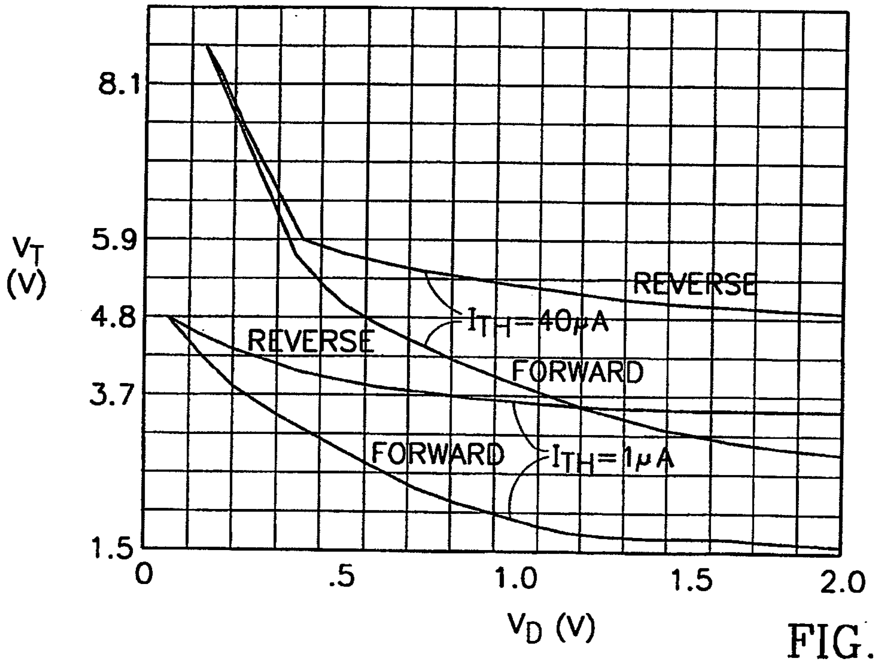
$V_G = f(N_A, T_{ox})$ (V)
0
1
2
3
4
5
0.5V
1V
1V
V_{T0}
0
1
2
3
4
Vx (V)

FIG.10



V_G=5V
REVERSE
FORWARD
V_G=3V
REVERSE
V_G=2.5V
REVERSE
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
I_D (A)
0
.5
1.0
1.5
2.0
V_D (V)

FIG. 11



8.1
5.9
4.8
3.7
1.5
V_T (V)
REVERSE
I_{TH}=40μA
REVERSE
FORWARD
FORWARD
I_{TH}=1μA
0
.5
1.0
1.5
2.0
V_D (V)

FIG. 12

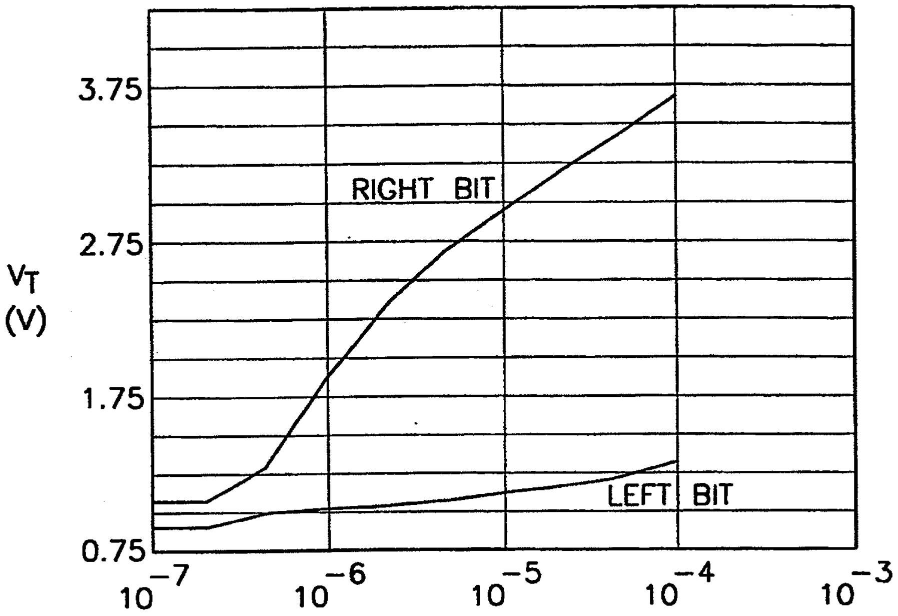

3.75
2.75
1.75
0.75
V_T
(V)
RIGHT BIT
LEFT BIT
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
PROGRAMMING TIME (S)

FIG.13

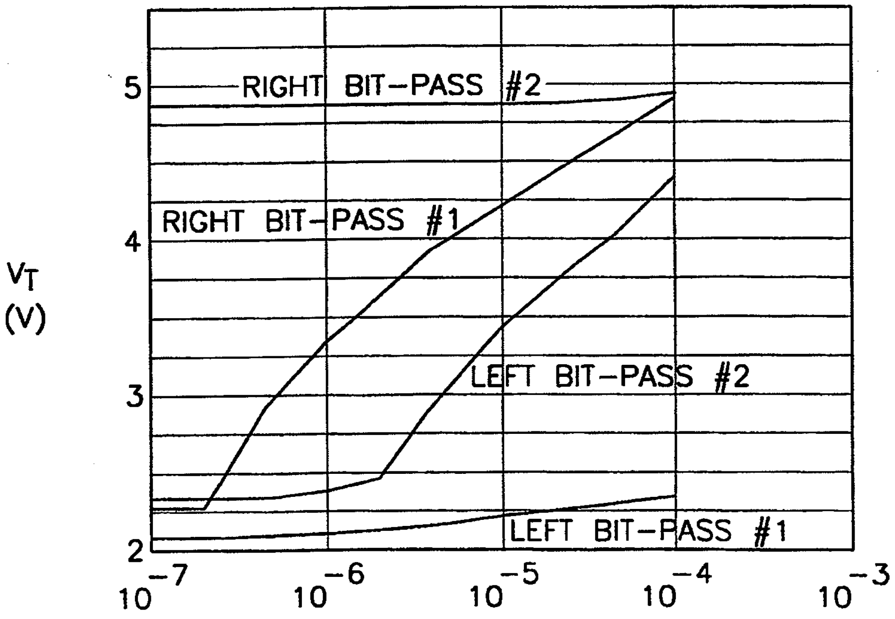

5
4
3
2
V_T
(V)
RIGHT BIT-PASS #2
RIGHT BIT-PASS #1
LEFT BIT-PASS #2
LEFT BIT-PASS #1
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
PROGRAMMING TIME (S)

FIG.14

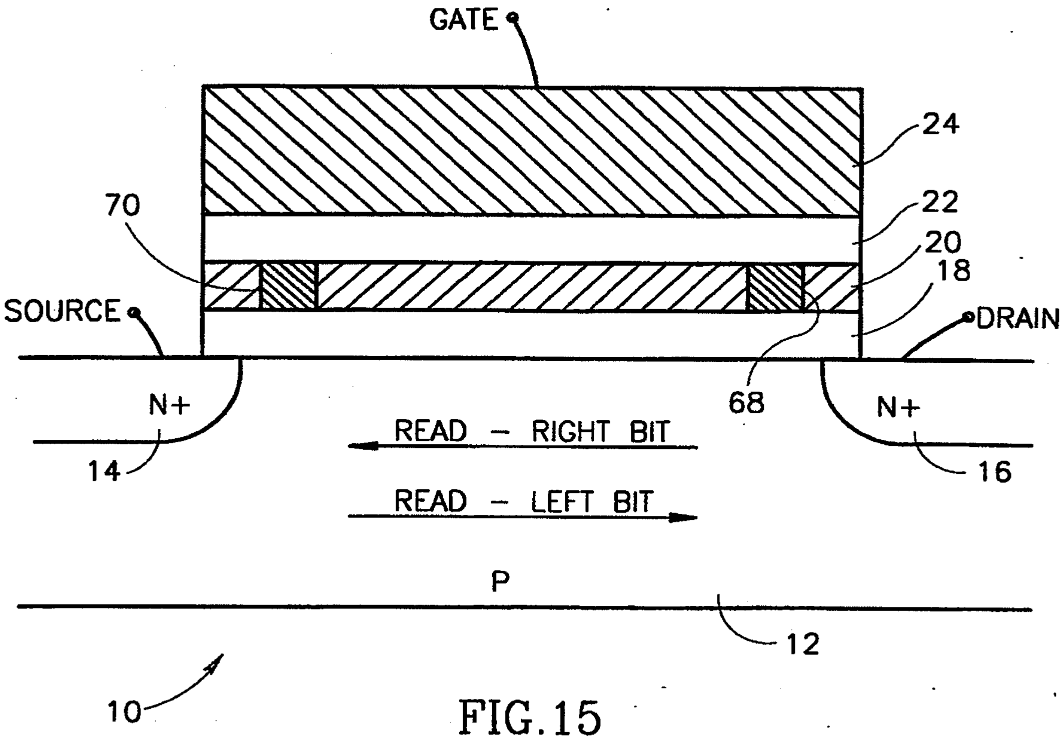

GATE
24
22
20
18
70
SOURCE
DRAIN
68
N+
N+
14
16
READ - RIGHT BIT
READ - LEFT BIT
P
12
10

FIG.15

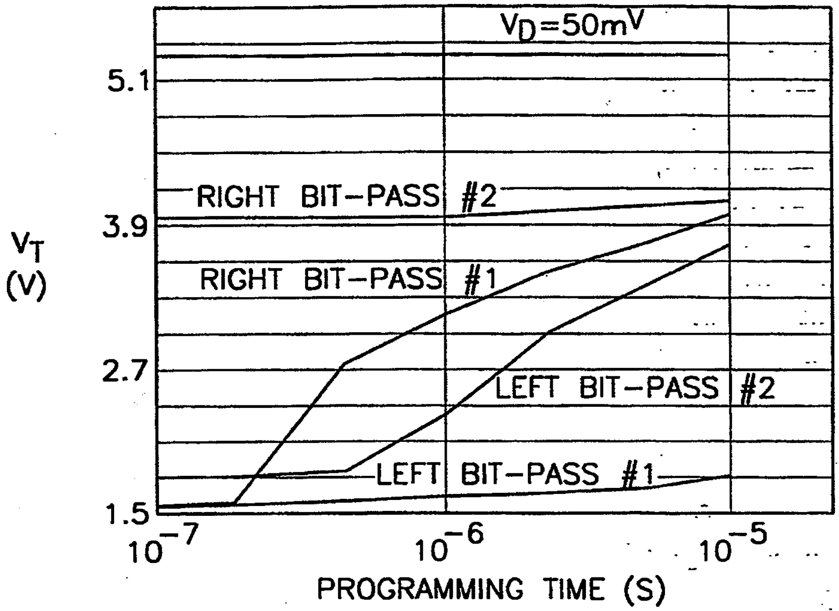

VD=50mV
5.1
3.9
2.7
1.5
VT
(V)
RIGHT BIT-PASS #2
RIGHT BIT-PASS #1
LEFT BIT-PASS #2
LEFT BIT-PASS #1
10^-7
10^-6
10^-5
PROGRAMMING TIME (S)

FIG.16

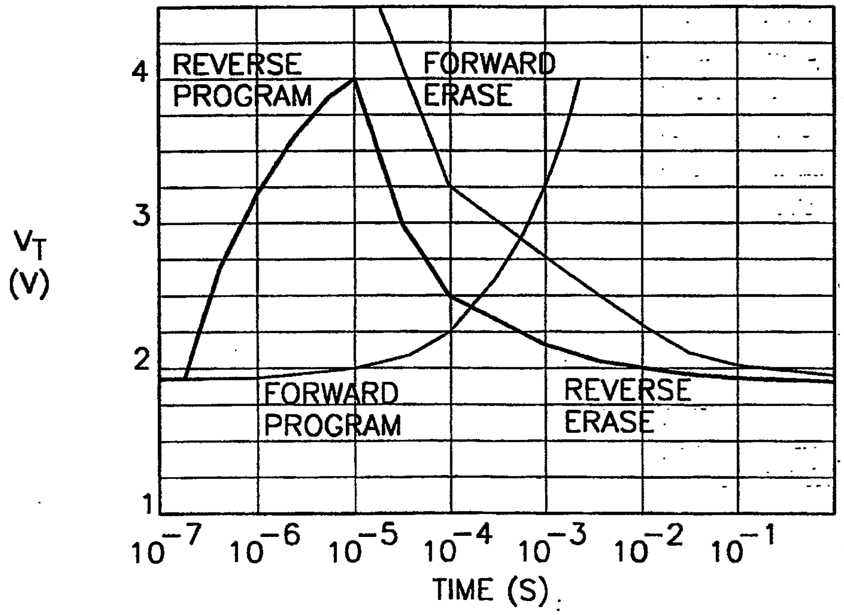

4
3
2
1
VT
(V)
REVERSE PROGRAM
FORWARD ERASE
FORWARD PROGRAM
REVERSE ERASE
10^-7
10^-6
10^-5
10^-4
10^-3
10^-2
10^-1
TIME (S)

FIG.17

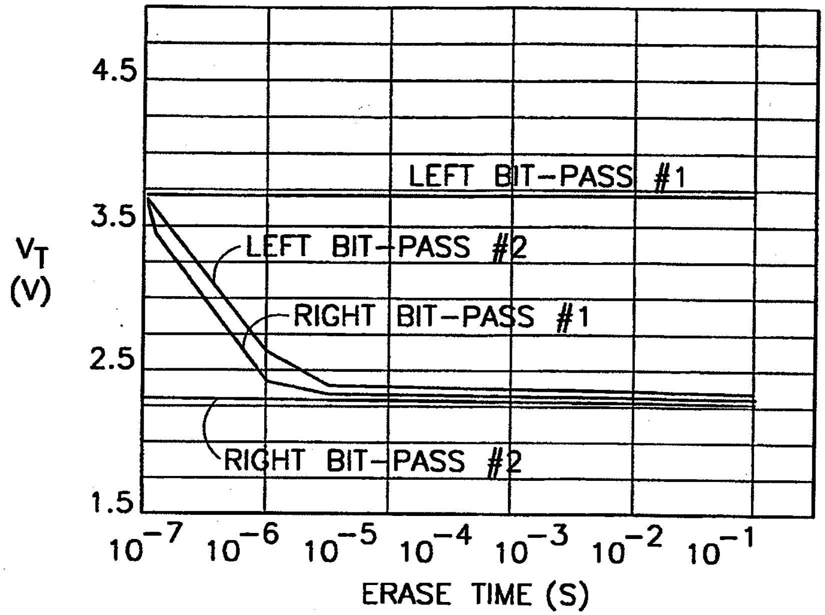

4.5
3.5
2.5
1.5
V_T (V)
LEFT BIT-PASS #1
LEFT BIT-PASS #2
RIGHT BIT-PASS #1
RIGHT BIT-PASS #2
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
ERASE TIME (S)

FIG.18

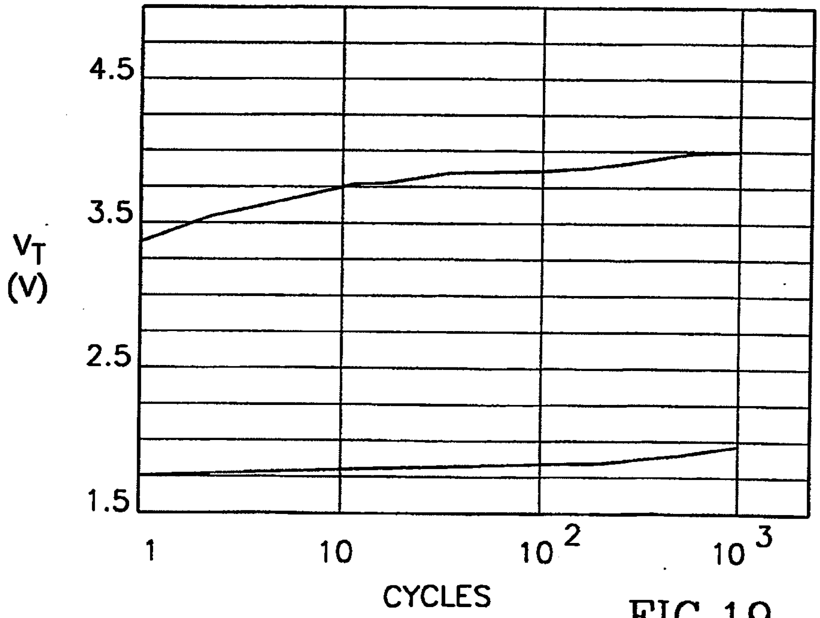

4.5
3.5
2.5
1.5
V_T (V)
1
10
10^2
10^3
CYCLES

FIG.19

12/15

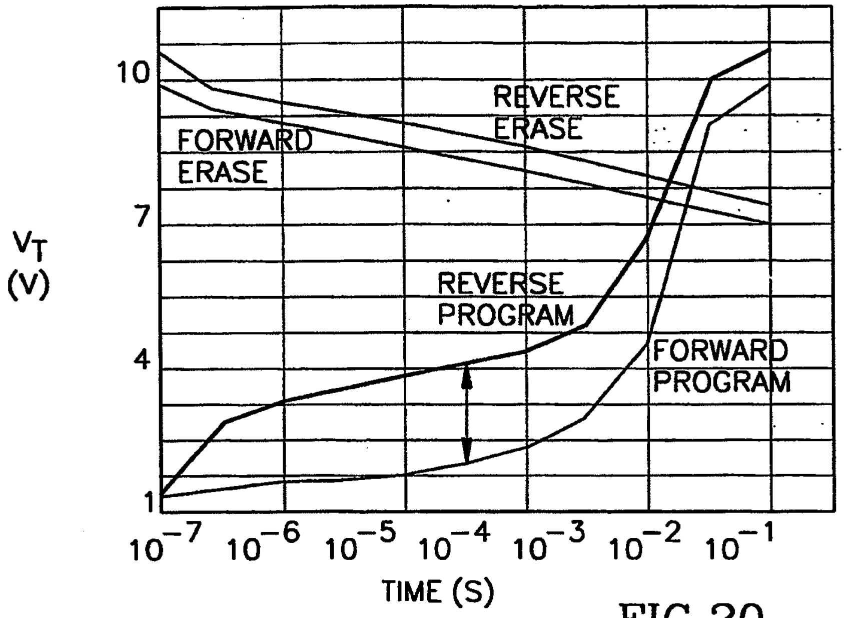


FIG.20

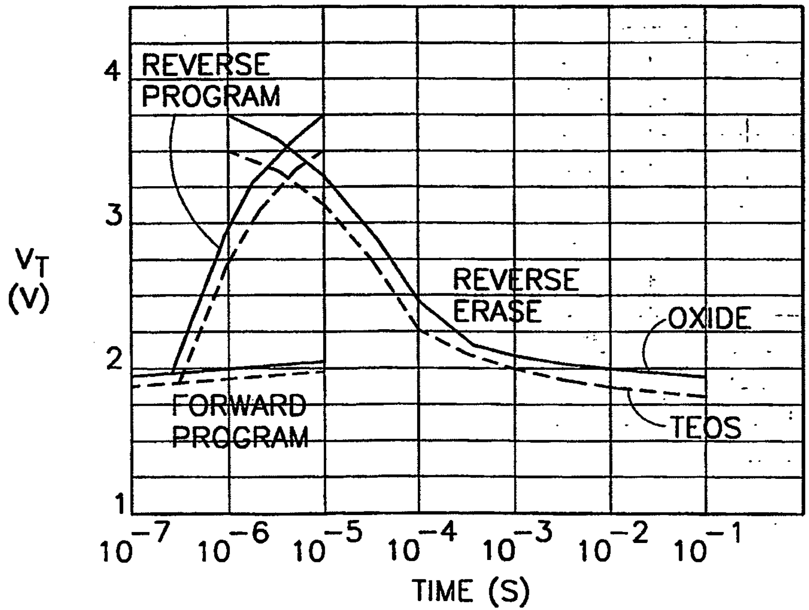


FIG.21

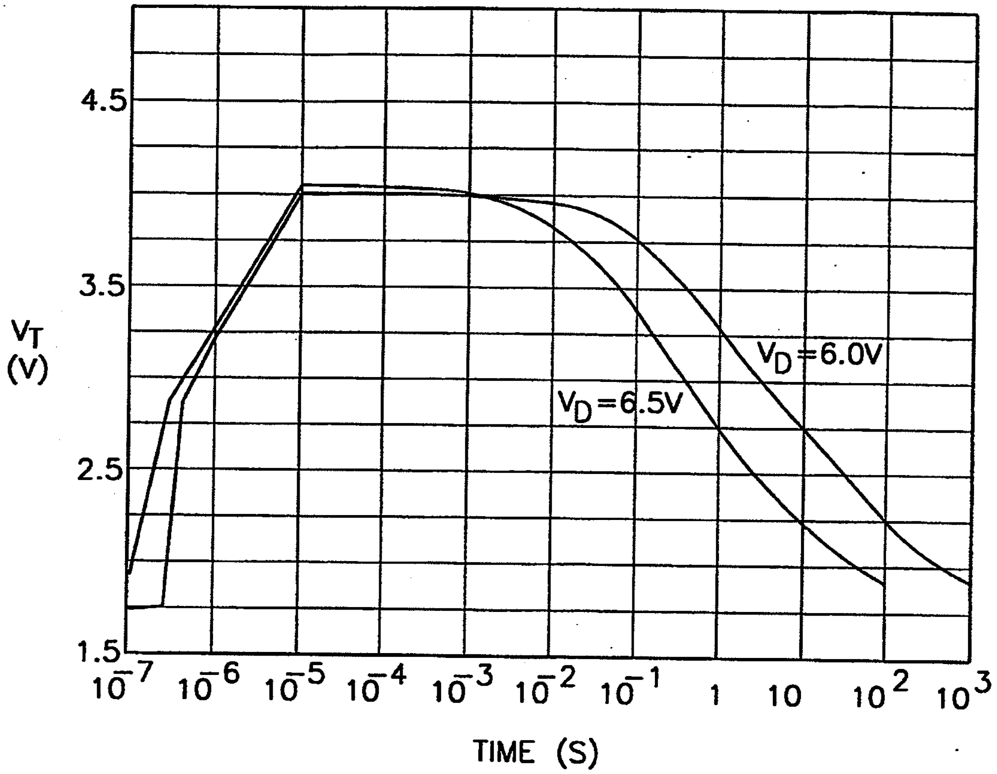
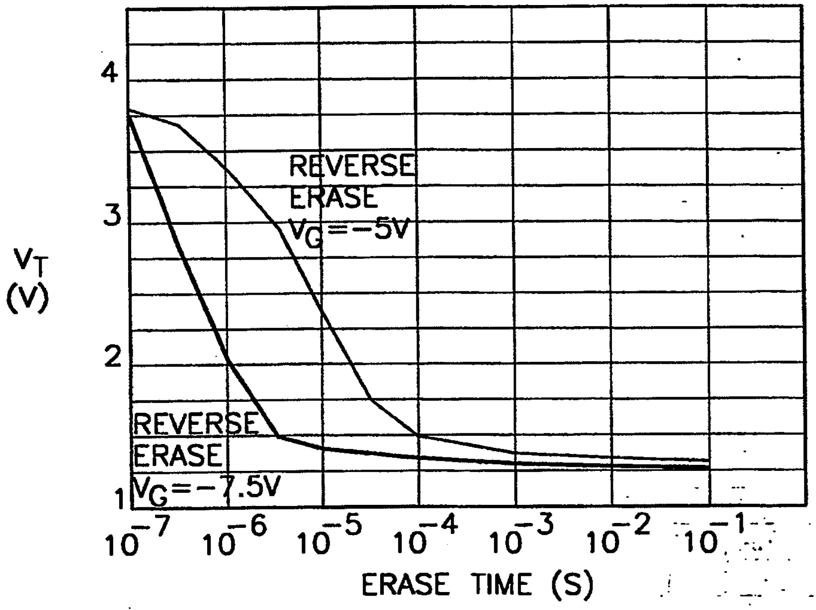

4.5
3.5
2.5
1.5
V_T
(V)
V_D=6.0V
V_D=6.5V
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
1
10
10^2
10^3
TIME (S)

FIG.22



4
3
2
1
V_T (V)
REVERSE ERASE V_G=−5V
REVERSE ERASE V_G=−7.5V
10^{-7}
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
ERASE TIME (S)

FIG.23

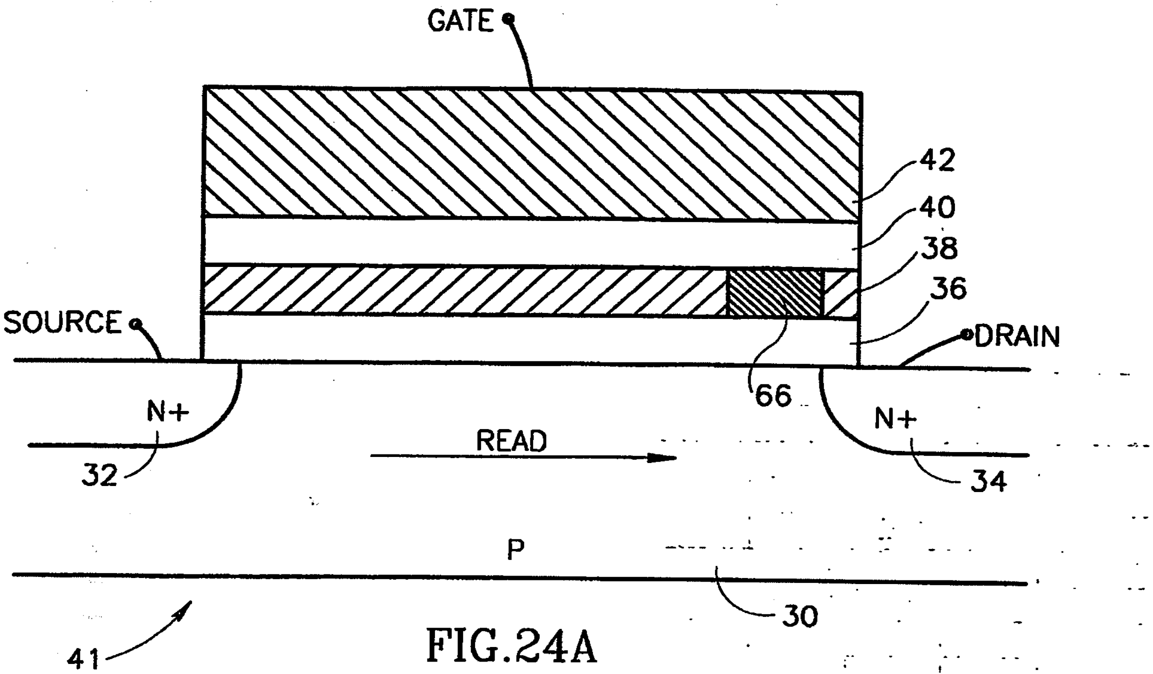

GATE
42
40
38
36
SOURCE
DRAIN
66
N+
N+
READ
32
34
P
30
41

FIG.24A
PRIOR ART

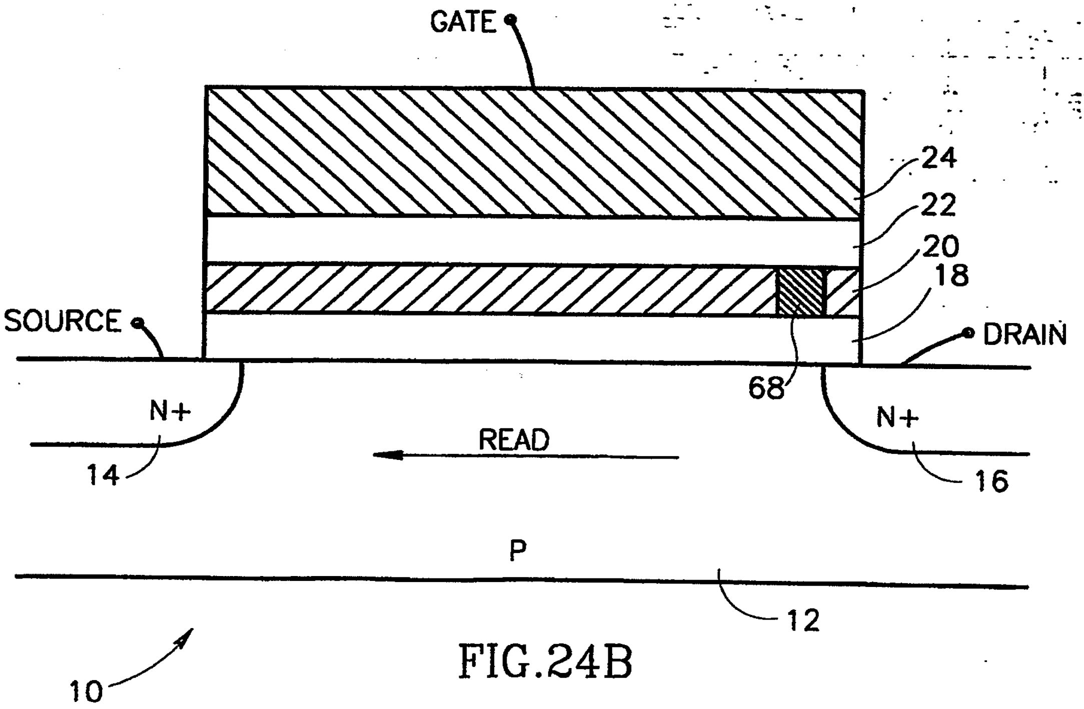

GATE
24
22
20
18
SOURCE
DRAIN
68
N+
N+
READ
14
16
P
12
10

FIG.24B

[redacted] [redacted] [redacted]

D16

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 977 121 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
02.02.2000 Bulletin 2000/05

(51) Int Cl.^{7}: G06F 12/02

(21) Application number: 99305882.5

(22) Date of filing: 26.07.1999

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 28.07.1998 JP 21263098

(71) Applicant: SONY CORPORATION
Tokyo (JP)

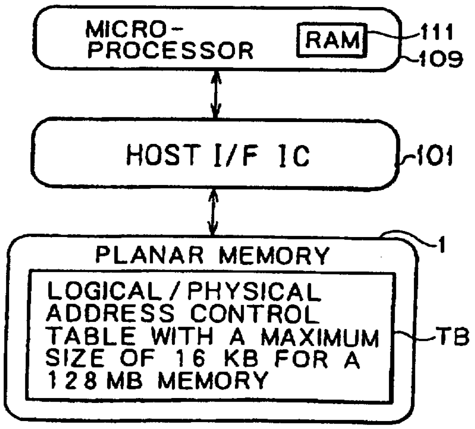
(72) Inventors:
- Iida, Kenichi
  Shinagawa-ku, Tokyo (JP)
- Koga, Noriyuki
  Shinagawa-ku, Tokyo (JP)
- Yamada, Eiichi
  Shinagawa-ku, Tokyo (JP)
- Sugiura, Mari
  Shinagawa-ku, Tokyo (JP)
- Obayashi, Shuji
  Shinagawa-ku, Tokyo (JP)

(74) Representative: Ayers, Martyn Lewis Stanley
J.A. KEMP & CO.
14 South Square
Gray's Inn
London WC1R 5LX (GB)

(54) Non-volatile memory, recording apparatus and recording method

(57) The object of the present invention is to provide a non-volatile memory including a logical/physical address control table for controlling data recorded discretely in the non-volatile memory composed of a plurality of blocks each serving as a data deletion unit and comprising adjacent pages which each have a fixed length and serve as a data read/write unit, and to provide a recording apparatus as well as a recording method for generating control data cataloged in the logical/physical address control table and used in making an access to the non-volatile memory.

FIG.4


MICRO-PROCESSOR
RAM
111
109
HOST I/F IC
101
PLANAR MEMORY
1
LOGICAL / PHYSICAL ADDRESS CONTROL TABLE WITH A MAXIMUM SIZE OF 16 KB FOR A 128 MB MEMORY
TB

EP 0 977 121 A2

Printed by Jouve, 75001 PARIS (FR)

[redacted] [redacted]

[redacted] [redacted] [redacted]

Description

[0001] The present invention relates to a non-volatile memory including a logical/physical address control table for controlling the non-volatile memory in which data is recorded discretely, composed of a plurality of blocks each serving as a data deletion unit and comprising adjacent pages each of which has a fixed length and serves as a data read/write unit, and relates to a recording apparatus as well as a recording method for generating control data cataloged in the logical/physical address control table and used in making an access to the non-volatile memory.

[0002] In recent years, there has been developed a compact storage device (or storage medium) which includes a solid storage device such as a flash memory and is mounted on various kinds of equipment such as a video camera to store video, audio and computer data.

[0003] Since such a storage device is more compact than the typical storage medium such as a 3.5-inch floppy disc and requires a drive with a small size, the device can be suitably mounted on equipment such as a video camera, a recording apparatus and a portable computer apparatus.

[0004] By the way, a flash memory exhibits a characteristic showing that the length of the life thereof is affected by the number of repeated writing and erasing operations. With regard to a file system for writing and reading out data into and from a storage device utilizing a flash memory like the one described above, the concept of logical and physical addresses has been introduced. In a configuration with this concept introduced, operations to write and read out data into and from the storage device are carried out by utilizing logical and physical addresses.

[0005] With such a configuration adopted in a system, in order to make an access to the storage device implemented by a flash memory in the system in an operation to write or read out data into or from the memory, it is necessary to provide a table showing relations between logical addresses and physical addresses. Such a table is referred to hereafter as the logical/physical address control table.

[0006] In the conventional system, a logical/physical address control table is provided in the main apparatus of equipment utilizing the storage device.

[0007] By the way, a large logical/physical address control table has a typical data size of about 16 KB depending on the storage capacity of the flash memory. On the other hand, the storage capacity of a RAM (Random Access Memory) embedded in a 1-chip microprocessor employed in the main apparatus is only several tens of KB at the most. Thus, if the logical/physical address control table is included in the RAM embedded in a microprocessor employed in the main apparatus, most of the storage area of the RAM will be occupied by the logical/physical address control table. Thus, it is quite difficult to store the logical/physical address control table in the RAM embedded in the microprocessor without sacrificing the processing performance of the microprocessor. In addition, a low-cost microprocessor may have a RAM capacity of only about 10 KB. In this case, it is impossible to store the logical/physical address control table in the embedded RAM of such a microprocessor due to the fact that the size of the logical/physical address control table is larger than the RAM capacity.

[0008] In order to solve the problems described above, the main apparatus utilizing a storage device implemented by a flash memory is provided with an external RAM which can be used for storing a logical/physical address control table.

[0009] However, a RAM provided externally causes problems of a rising cost and increased power consumption by additional power required to drive the external RAM. In particular, if the main apparatus is a portable apparatus powered by a battery, the problem of increased power consumption raises another serious problem affecting the life of the battery.

[0010] In addition, information recorded in the logical/physical address control table stored in the external RAM is cleared when the storage device is taken out from the main apparatus. Information is recorded in a logical/physical address control table normally each time the storage device is mounted on the main apparatus.

[0011] In generation of a logical/physical address control table, the microprocessor of the main apparatus checks the internal state of a storage device mounted on the main apparatus, constructing information in the logical/physical address control table as part of a file-management system. Then, the logical/physical address control table is stored in the external RAM.

[0012] The time it takes to carry out such preparation processing is at least about several seconds. In the case of a low-cost microprocessor with a low processing ability, a processing time with a length of a multiple of these several seconds is required. For example, since an access to write or read out data into or from the storage device can be made only after the preparation processing is completed, the time it takes to carry out the preparation processing appears to the user as a waiting time. If the use of the equipment in a way the user likes is taken into consideration, the time it takes to carry out such preparation processing needs to be shortened as much as possible.

[0013] It is thus an object of the present invention to provide a non-volatile memory including a logical/physical address control table and a recording apparatus as well as a recording method for generating control data cataloged in the logical/physical address control table and used in making an access to the non-volatile memory wherein the non-volatile memory allows a microprocessor having only a small work memory to use the logical/physical address control table in an access to the non-volatile memory.

[0014] According to the first aspect of the present in-

[redacted] [redacted]

[redacted] [redacted] [redacted]

vention, there is provided a non-volatile memory which allows a microprocessor having only a small work memory to use the logical/physical address control table in an access to the non-volatile memory wherein a storage area of the non-volatile memory comprises a main-data area comprising any one of the blocks comprising a plurality of the adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording main data; and a control-data area comprising any one of the blocks comprising a plurality of the adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording control data representing relations associating logical addresses with physical addresses wherein the logical addresses are assigned to pieces of data written into the blocks and the physical addresses show a physical layout order of the blocks.

[0015] According to the second aspect of the present invention, there is provided a recording apparatus which generates control data cataloged in the logical/physical address control table and used in making an access to the non-volatile memory comprising: an attribute determining means for determining whether data to be written into the non-volatile memory is main data or control data; an identifier generating means for generating an identifier indicating whether the data to be written into the non-volatile memory is main data or control data in accordance with a result of determination output by the attribute determining means; and a memory control means for synthesizing the data to be written into the nonvolatile memory and the identifier output by the identifier generating means and writing synthesized data into the non-volatile memory.

[0016] According to the third aspect of the present invention, there is provided a recording method which generates control data cataloged in the logical/physical address control table and used in making an access to the non-volatile memory comprising: attribute determining step of determining whether data to be written into the non-volatile memory is main data or control data; an identifier generating step of generating an identifier indicating whether the data to be written into the non-volatile memory is main data or control data in accordance with a result of determination output at the attribute determining step; and a step of synthesizing the data to be written into the nonvolatile memory and the identifier output at the identifier generating step and writing synthesized data into the non-volatile memory.

[0017] Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Fig. 1 is a block diagram showing the configuration of a conventional system comprising a main apparatus and the planar memory for a purpose of comparison with an embodiment of the present invention;

Fig. 2 is an explanatory diagram conceptually showing an interface between a microprocessor employed in the main apparatus and the planar memory in the conventional system for a purpose of comparison with the embodiment;

Fig. 3 is a block diagram showing the configuration of a system comprising a main apparatus and the planar memory as implemented by the embodiment of the present invention;

Fig. 4 is an explanatory diagram conceptually showing an interface between a microprocessor employed in the main apparatus and the planar memory in the embodiment;

Fig. 5A is a diagram showing a front view of the external shape of the planar memory;

Fig. 5B is a diagram showing a top view of the external shape of the planar memory;

Fig. 5C is a diagram showing a side view of the external shape of the planar memory;

Fig. 5D is a diagram showing a bottom view of the external shape of the planar memory;

Fig. 6 is an explanatory diagram showing a processing hierarchy of a file system provided by the embodiment;

Fig. 7A is an explanatory diagram showing a segment in a physical data structure of a flash memory;

Fig. 7B is an explanatory diagram showing a boot block in the physical data structure of the flash memory;

Fig. 7C is an explanatory diagram showing a backup of a boot block in the physical data structure of the flash memory;

Fig. 7D is an explanatory diagram showing a block in the physical data structure of the flash memory;

Fig. 7E is an explanatory diagram showing a page in the physical data structure of the flash memory;

Fig. 7F is an explanatory diagram showing a redundant portion of the page in the physical data structure of the flash memory;

Fig. 8 is an explanatory diagram showing the contents of a management flag;

Fig. 9A is an explanatory diagram showing a preprocessing state in description of the concept of processing to update data in a flash memory, a logical address and a physical address;

Fig. 9B is an explanatory diagram showing a postprocessing state in the description of the concept of processing to update data in a flash memory, a logical address and a physical address;

Fig. 10 is an explanatory diagram used for conceptually describing the concept of managing a logical/physical address control table;

Fig. 11A is a diagram showing the whole data structure of the logical/physical address control table provided by this embodiment;

Fig. 11B is a diagram showing a data structure of one segment of the logical/physical address control table provided by this embodiment;

[redacted] [redacted]

Fig. 12A is an explanatory diagram used for describing management of unused blocks by using the logical/physical address control table provided by the embodiment;

Fig. 12B is an explanatory diagram used for describing management of unused blocks by using the logical/physical address control table in the conventional system;

Fig. 13 is an explanatory diagram showing relations between the storage capacity of a flash memory, the number of blocks, the size of a block, the size of a page and the size of the logical/physical address control table;

Fig. 14A is an explanatory diagram showing the physical data structure of a segment in a flash memory provided by the embodiment;

Fig. 14B is an explanatory diagram showing the physical data structure of a main-data block in the flash memory provided by the embodiment;

Fig. 14C is an explanatory diagram showing the physical data structure of a control-data block in the flash memory provided by the embodiment;

Fig. 14D is an explanatory diagram showing the structure of the logical/physical address control table in the flash memory provided by the embodiment;

Fig. 15 shows a flowchart representing a recording method adopted in a recording apparatus implemented by the embodiment;

Fig. 16 shows a flowchart representing a method of determining a block to be used in an operation to rewrite data and related processing carried out on the logical/physical address control table in the recording apparatus implemented by the embodiment;

Fig. 17 shows a flowchart representing a method of rewriting main data in the recording apparatus implemented by the embodiment; and

Fig. 18 shows a flowchart representing a method of rewriting control data in the recording apparatus implemented by the embodiment.

[0018] An embodiment of the present invention will be described below. It should be noted that a storage device provided by the embodiment is a planar memory having a planar external shape.

[0019] The embodiment will be described in the following order:

1. External Shape of the Memory
2. Memory Format
    - 2-1. Processing Hierarchy of a Memory File System
    - 2-2. Physical Data Structure
    - 2-3. Concept of Physical and Logical Addresses
    - 2-4. Logical Physical Address Control table of the Embodiment
3. System Configuration

1. External Shape of the Memory

[0020] The description begins with the planar shape of a planar memory 1, which is a storage device provided by an embodiment of the present invention, with reference to Figs. 5A, 5B, 5C and 5D.

[0021] The planar memory 1 is implemented by a memory device which is enclosed in a planar case like one shown in Figs. 5A, 5B, 5C and 5D and typically has a predetermined storage capacity. In this embodiment, the memory device is a flash memory.

[0022] Figs. 5A, 5B, 5C and 5D are diagrams showing respectively a top view, a front view, a side view and a bottom view of the case which is for example, formed as a plastic mold having a typical length W11 of 60 mm and a typical width W12 of 20 mm as shown in Fig. 5B and a typical height W13 of 2.8 mm as shown in Fig. 5A.

[0023] The terminal unit 2 which is formed on the surface of the case has 9 electrodes disposed in such a manner as to extend from the lower portion of the front side to the bottom side. Data is written into or read out from the internal memory device through the terminal unit 2.

[0024] A cut 3 is formed at the upper left corner in the diagram showing the top view of the case. The cut 3 prevents for example the planar memory 1 from being inserted into a case mounting/dismounting mechanism of a drive of the main apparatus in a wrong insertion direction.

[0025] On the bottom of the case, a bumpy surface 4 is created to prevent from slipping of the case which improves usability of the case.

[0026] A slide switch 5 is further formed on the bottom to prevent inadvertent erasure of data stored in the internal memory device.

2. Memory Format

2-1. Processing Hierarchy of a Memory File System

[0027] The next description explains a format adopted in the system wherein the planar memory 1 is used as a recording medium.

[0028] Fig. 6 is an explanatory diagram showing a file-system processing hierarchy of the system wherein the planar memory 1 is used as a recording medium.

[0029] As shown in Fig. 6, the file-system processing hierarchy comprises an application processing layer at the top followed sequentially by a file-management processing layer, a logical-address layer, a physical-address layer and a flash-memory access layer at the bottom of the hierarchy. The file-management processing layer in the hierarchy is the so-called FAT (File Allocation Table). As is also obvious from Fig. 6, the file system of the embodiment introduces the concept of logical and physical addresses which will be described later.

2-2. Physical Data Structure

[0030] Figs. 7A to 7F are diagrams showing a physical data structure of a flash memory which is used as the storage device of the planar memory 1.

[0031] The storage area of a flash memory is divided into segments, that is, basic data units each having a fixed length. The size of a segment is prescribed to be 4 MB or 8 MB. Thus, the number of segments constituting a flash memory varies depending on the capacity of the flash memory.

[0032] As shown in Fig. 7A, a segment is further divided into blocks each of which is a data unit prescribed to have a length of 8 KB or 16 KB. Basically, a segment is divided into 512 blocks, namely, blocks 0 to n where n = 511 as shown in Fig. 7A. However, a flash memory is allowed to include a defect area comprising up to a predetermined number of blocks. A defect area is a damaged area into which data can not be written. Thus, the number of effective blocks into which data can be actually written is smaller than 512, that is, in actuality, n is smaller than 511.

[0033] As shown in Fig. 7A, 2 blocks at the head of blocks 0 to n, namely, blocks 0 and 1, are called boot blocks. Actually, however, the 2 blocks at the head of the effective blocks are used as boot blocks. Thus, there is no assurance boot blocks are always blocks 0 and 1.

[0034] The remaining blocks are user blocks for storing user data.

[0035] As shown in Fig. 7D, a block is further divided into pages 0 to m. As shown in Fig. 7E, a page comprises a data area of 512 bytes and a redundant portion of 16 bytes to give a fixed size of 528 bytes. It should be noted that the structure of the redundant portion will be described later with reference to Fig. 7F.

[0036] The number of pages in a block is 16 for a block size of 8 KB and 32 for a block size of 16 KB.

[0037] The block structure shown in Fig. 7D and the page structure shown in Fig. 7E apply to both the boot blocks and the user blocks.

[0038] Data is written into and read out from a flash memory in page units. However, data is erased from a flash memory in block units. A flash memory is characterized in that data can not be written into an area in which other data has already been written before. Thus, new or replaced data has to be written into a page which is shown by the file management system as an unused area. The file management system changes the status of a block from 'used' status to 'unused' status by merely changing particular data for the block in the table controlling status of blocks to a new value indicating that the block is an unused block without erasing the contents of the block. For this reason, before writing data into this unused block, it is necessary to erase the contents thereof. Since contents can be erased in block units only as described before, new or replaced data is actually written into the flash memory in block units instead of page units.

[0039] As shown in Fig. 7B, a header is stored on page 0 of the first boot block. Information indicating an address indicating the position of initial bad data is stored on page 1. On page 2, information called a CIS/IDI (Card Information Structure/Identify Drive Information) is stored.

[0040] As shown in Fig. 7C, the second boot block is used as a backup area.

[0041] The 16-byte redundant portion shown in Fig. 7E has a structure shown in Fig. 7F.

[0042] As shown in Fig. 7F, the first 3 bytes of the redundant portion, namely, bytes 0 to 2, is an overwrite area which can be rewritten depending on updating of the contents of the data area. To be more specific, byte 0 is used for storing block status and byte 1 is used for storing data status (Block Flag Data). A predetermined number of high-order bits in byte 2 are used for storing an update status (update status).

[0043] Basically, contents of bytes 3 to 15 are fixed in accordance with data stored on the page. That is to say, these bytes are an area for storing information that can not be rewritten.

[0044] To be more specific, a management flag (Block Info) is stored in byte 3 and a logical address (Logic Address) is stored in an area comprising the following 2-byte area, namely, bytes 4 and 5.

[0045] The following 5-byte area comprising bytes 6 to 10 is used as a format reserve area. The following 2-byte area comprising bytes 11 and 12 is used for storing distributed information ECC (Error Correction Code) for error correcting for the format reserve.

[0046] The remaining bytes 13 to 15 are used for storing data ECC for error correction for data stored in the data area shown in Fig. 7E.

[0047] As shown in Fig. 8, contents of bits 7 to 0 of the management flag stored in byte 3 of the redundant portion shown in Fig. 7F are defined individually.

[0048] Bits 7, 6, 1 and 0 are undefined reserved bits.

[0049] Bit 5 includes a flag indicating whether an allowance of access to the block is valid or invalid. To be more specific, a value of 1 indicates that an access to the block can be made freely while a value of 0 indicates that the block is read protected. Bit 4 includes a copy prohibited specification flag with a value of 1 meaning that a copy operation is allowed while a value of 0 meaning that a copy operation is prohibited.

[0050] Bit 3 is a control table flag indicating whether or not the block is a block for storing a logical/physical address control table to be described later. To be more specific, a value of 0 set in bit 3 indicates that the block is block for storing a logical/physical address control table. A value of 1 set in bit 3, on the other hand, indicates a denial, that is, the block is not a block for storing a logical/physical address control table.

[0051] Bit 2 is a system flag. A value of 1 indicates

[redacted] [redacted] [redacted]

that the block is a user block while a value of 0 indicates that the block is a boot block.

[0052] Next, a relation between the storage capacity of a flash memory and the number of blocks or the number of segments is explained with reference to Fig. 13.

[0053] As shown in the figure, the flash-memory storage capacity of the planar memory 1 is prescribed to be 4 MB, 8 MB, 16 MB, 32 MB, 64 MB or 128 MB.

[0054] In the case of the minimum storage capacity of 4 MB, the block size is prescribed to be 8 KB and the number of blocks is 512. That is to say, the storage capacity of 4 MB is just equal to the size of a segment. A planar memory 1 with a flash-memory storage capacity of 8 MB comprises 1,024 blocks each prescribed to have a size of 8 KB as described above. The 1,024 blocks constitute 2 segments. In addition, as described above, a 8-KB block comprises 16 pages.

[0055] In the case of a planar memory 1 with a flash-memory storage capacity of 16 MB, however, the size of a block can be 8 KB or 16 KB. Thus, the planar memory 1 can comprise 2,048 8-KB blocks (or 4 segments) or 1,024 16-KB blocks (or 2 segments). A 16-KB block comprises 32 pages.

[0056] In the case of a planar memory 1 with a flash-memory storage capacity of 32 MB, 64 MB or 128 MB, the size of a block is prescribed to be 16 KB only. Thus, a planar memory 1 with a flash-memory storage capacity of 32 MB comprises 2,048 blocks (or 4 segments) and a planar memory 1 with a flash-memory storage capacity of 64 MB comprises 4,096 blocks (or 8 segments). On the other hand, a planar memory 1 with a flash-memory storage capacity of 128 MB comprises 8,192 blocks (or 16 segments).

2-3. Concept of Physical and Logical Addresses

[0057] The following description explains a concept of physical and logical addresses adopted in a file system provided by this embodiment by showing an operation as shown in Figs. 9A and 9B to update data in the aforementioned physical data structure of a flash memory.

[0058] Fig. 9A is a diagram showing 4 blocks extracted from a segment as a model.

[0059] A physical address is assigned to each of the blocks. As shown in the figure, the physical address increases in accordance with the physical layout of the blocks in the memory. The relation between a block and a physical address assigned to the block is fixed. The values of the physical addresses assigned to the 4 blocks shown in Fig. 9A are 105 for the top block, 106, 107 and 108 following in order. It should be noted that an actual physical address is 2 bytes in length.

[0060] In the example shown in Fig. 9A, the blocks with the physical addresses 105 and 106 are used blocks in which data is stored. On the other hand, the blocks with the physical addresses 107 and 108 are unused blocks or unrecorded areas, from which data was erased.

[0061] A logical address is an address assigned to data written into a block. A logical address is an address used by the FAT file system.

[0062] In the example shown in Fig. 9A, the values of the logical addresses assigned to pieces of data in the 4 blocks are 102 for the data in the top block, 103, 104 and 105 following in order. It should be noted that an actual logical address is also 2 bytes in length.

[0063] In the state shown in Fig. 9A, for example, data stored at the physical address 105 is updated, that is, its contents are rewritten or erased partially.

[0064] In such a case, in the file system of the flash memory, updated data is not rewritten in the same block to be updated. Instead, the updated data is written into an unused block.

[0065] That is to say, as shown in Fig. 9B, at the processing (1), the data stored at the physical address 105 is erased and then the updated data is written into a block at the physical address 107 which has been an unused block so far.

[0066] Then, in processing (2), assignment of logical addresses is changed so that the logical address 102 which has been assigned to the physical address 105 in the state before the data updating process shown in Fig. 9A is reassigned to the physical address 107 assigned to the block in which the updated data was written as shown in Fig. 9B. With this, the logical address 104 which has been assigned to the physical address 107 before the data updating process is reassigned to the physical address 105.

[0067] That is to say, a physical address is assigned to a block permanently while a logical address can be regarded as an address assigned permanently to a data which has the size of a block unit and is once written into a block.

[0068] By swapping blocks as described above, an access is not made in a concentrated manner repeatedly to the same storage area (block), making it possible to prolong the life of the flash memory which is determined by the number of write operations carried out thereon.

[0069] In swapping logical addresses between blocks by treating logical address in the processing (2) as described above, data is moved from the physical address of a block occupied by the data prior to the data updating process to the physical address of a block occupied by the updated data. To the FAT file system, however, the data appears to remain at the same logical address, allowing subsequent accesses thereto to be made properly and correctly.

[0070] It should be noted that, in order to simplify control to update information stored in the logical/physical address control table, the processing to swap logical addresses between blocks is prescribed as processing to swap logical addresses only between blocks pertaining to the same segment. To put it differently, logical addresses are not swapped among blocks of different seg-

[redacted]

ments.

2-4. Logical/Physical Address Control table of the Embodiment

[0071] As is obvious from the explanation with reference to Figs. 9A and 9B, the swapping of logical addresses between blocks changes the assignment of a logical address to a physical address. A logical/physical address control table is used for storing information on assignments of logical addresses to physical addresses. Thus, to implement an access to the flash memory to write or read out data into or from the flash memory, the logical/physical address control table is required. To put it in detail, the FAT file system refers to the logical/physical address control table in order to identify a physical address associated with the logical address specified in the access by the FAT file system. The access is then made to a block at the identified physical address. Conversely speaking, an access by the FAT file system to the flash memory can not be made without the logical/physical address control table.

[0072] In the conventional system, when the planar memory 1 is mounted on the main apparatus, a logical/physical address control table is constructed by a microprocessor employed in the main apparatus by checking logical addresses of redundant portions shown in Fig. 7F for all pages in the planar memory 1 and then stored in a RAM also employed in the main apparatus. That is to say, the planar memory 1 does not include information stored in the logical/physical address control table.

[0073] In the case of the embodiment, on the contrary, a logical/physical address control table is stored in the planar memory 1 as will be described later.

[0074] Fig. 10 is a diagram conceptually showing the construction of a logical/physical address control table to be stored in the planar memory 1 provided by the present invention.

[0075] In this embodiment, logical addresses put typically in an ascending order are assigned to 2-byte physical addresses as indicated by information on assignment stored in the constructed logical/physical address control table.

[0076] It should be noted that, actually the logical and physical addresses are each expressed by 2 bytes as described before. The number of bits in the 2 bytes is large enough to cover 8,192 blocks composing a flash memory with a maximum storage capacity of 128 MB.

[0077] Much like the physical addresses, each of the logical addresses shown in Fig. 10 is actually 2 bytes in length. In addition, the 2-byte logical/physical addresses are each represented in a hexadecimal format. That is to say, a number following the notation '0X' comprises hexadecimal digits. It should be noted that the hexadecimal representation using the notation '0X' is used to express a hexadecimal number in the same way throughout the rest of the description.

[0078] Figs. 11A and 11B are diagrams showing a typical structure of the logical/physical address control table based on the concept shown in Fig. 10 as provided by this embodiment. In order to make the description easy to understand, the 16-byte redundant portion shown in Figs. 7E and 7F is omitted here.

[0079] The logical/physical address control table provided by this embodiment is stored in a certain block of the flash memory as shown in Figs. 11A and 11B. It should be noted that a block for storing the logical/physical address control table is prescribed to be always a block in the last segment.

[0080] As shown in Fig. 11A, the first 2 pages of the block, namely, pages 0 and 1, are used as an area for storing information of the logical/physical address control table for segment 0. In the case of a flash memory with a storage capacity of 4 MB shown in Fig. 13, only pages 0 and 1 are used as an area for storing information of the logical/physical address control table for the only 1 segment existing in the flash memory.

[0081] A flash memory with a storage capacity of 8 MB comprises 2 segments. In this case, pages 0 and 1 are used as an area for storing information of the logical/physical address control table for segment 0 whereas pages 2 and 3 are used as an area for storing information of the logical/physical address control table for segment 1.

[0082] As the storage capacity of the flash memory increases thereafter, the following 2 pages are used as an area for storing information of the logical/physical address control table for an additional segment of the flash memory. Finally, a flash memory with the maximum storage capacity of 128 MB comprises 16 segments. In this case, all pages are used as areas for storing information of the logical/physical address control table for all the segments including the last segment, that is, segment 15. Thus, in a flash memory with the maximum storage capacity of 128 MB, the 32 pages of the block are all used. In Fig. 11A, the last page of the block is page N where N is 31.

[0083] As is obvious from the above explanation, information of the logical/physical address control table is controlled in segment units.

[0084] Fig. 11B is a diagram showing the structure of 2-page information extracted from the logical/physical address control table for 1 segment. Since the data area of 1 page is 512 bytes in size as shown in Fig. 7E, the information shown in Fig. 11B is 1,024 (= 512 × 2) bytes in length.

[0085] As shown in Fig. 11B, the 2-page data area comprising 1,024 bytes is delimited into sub-areas each composed of 2 bytes. The sub-areas starting with byte 0 and ending with byte 991 are prescribed as sub-areas permanently allocated to logical address 0, logical address 1 and so on up to logical address 495. Thus, logical address 495 is associated with the last 2 bytes, namely, byte 990 and byte 991. Each of these 2-byte sub-areas is used for storing a physical address asso-

[redacted] [redacted] [redacted]

ciated with a logical address to which the sub-area is allocated. Thus, logical addresses are swapped between blocks as part of an operation to update actual data, to which one of the logical addresses is assigned as described above, by changing assignments of the logical addresses to physical addresses stored as information in the logical/physical address control table provided by this embodiment. To put it concretely, the assignments of the logical addresses to physical addresses are changed by swapping the physical addresses in the 2-page data area shown in Fig. 11B between 2-byte sub-areas permanently allocated to the logical addresses.

[0086] The remaining 32-byte area in the 2-page data area from byte 992 to byte 1023 is used for storing physical addresses of the remaining 16 blocks in the segment. Thus, the physical addresses of the remaining 16 blocks can be controlled. A remaining block serves typically as the so-called work block which is, for example, used for temporarily saving data to be rewritten in an operation to update data in block units.

[0087] By the way, in spite of the fact that 1 segment comprises 512 blocks as described earlier, in the table structure shown in Figs. 11A and 11B, only 496 blocks are controllable blocks, being allocated to logical addresses 0 to 495. This is because the rest is used for storing physical addresses of the remaining blocks described above and, in a flash memory, the existence of some defective blocks treated as unusable blocks is unavoidable. As a matter of fact, the remaining blocks includes a fairly large number of defective blocks.

[0088] Actually, 496 controllable blocks are sufficient for use as valid blocks which data can be written into or erased from.

[0089] As described above, the block for storing the logical/physical address control table comprises pages each including a redundant portion for storing a management flag shown in Fig. 8. Bit 3 of management flag is set at 0 to indicate that the block is used for storing a logical/physical address control table.

[0090] When the block containing the logical/physical address control table is updated, that is, when the contents of the logical/physical address control table are changed, the processing to swap logical addresses between blocks explained earlier with reference to Figs. 9A and 9B is carried out without exception. That is to say, the block for storing the logical/physical address control table is not determinate or it is impossible to prescribe a block to be dedicated for storing the logical/physical address control table.

[0091] Thus, the FAT file system makes accesses to the flash memory to search the memory for a block with a 0 set in bit 3 of the management flag thereof and recognizes such a block as a block for storing the logical/physical address control table. In order to make it easy for the FAT file system to search the flash memory for a block for storing the logical/physical address control table, the logical/physical address control table is prescribed in this embodiment to be always stored in a block pertaining to the last segment of the flash memory, that is, a segment identified by the last segment number. Accordingly, the FAT file needs to search only the last segment for a block for storing the logical/physical address control table. That is to say, it is not necessary for the FAT file to search all segments in the flash memory for a block for storing the logical/physical address control table.

[0092] The logical/physical address control table shown in Figs. 11A and 11B is stored in a block typically during a process of manufacturing the planar memory 1.

[0093] The following description explains an example with reference to Figs. 14A, 14B, 14C and 14D in order to help the reader understand better what has been described so far. Figs. 14A, 14B, 14C and 14D are diagrams showing a flash memory with a storage capacity of 4 MB. As explained earlier with reference to Fig. 13, the 4-MB flash memory includes only 1 segment comprising 512 blocks each having 16 pages.

[0094] As shown in Fig. 14A, the segment of the flash memory is segment 0, blocks 0 and 1 which each serve as a boot block. Since segment 0 is also the last segment, both main data and control data are stored therein. The physical address of block 0 is 0×0000 associated with no logical address. By the same token, the physical address of block 1 is 0×0001 also associated with no logical address.

[0095] Block 2 at a physical address of 0×0002 is used for storing main data to which a logical address of 0×0001 is assigned. Likewise, block 3 at a physical address of 0×0003 is used for storing main data to which a logical address of 0×0000 is assigned. Similarly, block 4 at a physical address of 0×0004 is used for storing main data to which a logical address of 0×0004 is assigned and block 5 at a physical address of 0×0005 is used for storing main data to which a logical address of 0×0003 is assigned. By the same token, block 6 at a physical address of 0×0006 is used for storing main data to which a logical address of 0×0002 is assigned and block 511 at a physical address of 0×01FF is used for storing main data to which a logical address of 0×1FD is assigned. On the other hand, block 123 at a physical address of 0×007B is used for storing control data to which a logical address of 0×0005 is assigned.

[0096] Fig. 14B is a diagram showing block 2 used for storing main data and Fig. 14C is a diagram showing block 123 used for storing control data. As shown in Figs. 14B and 14C, there is no difference in configuration at all between blocks used for storing main data and control data. The difference between the 2 blocks is the value of a control table flag stored in the redundant portion of page 0 in each of the blocks. Let us take the blocks shown in Figs. 14B and 14C as an example. The value of the control table flag in block 2 shown in Fig. 14B is 1 to indicate that this block is used for storing main data. On the other hand, the value of the control table flag in block 123 shown in Fig. 14C is 0 to indicate

[redacted]

[redacted] [redacted] [redacted]

that this block is used for storing control data. As shown in Figs. 14B and 14C, a logical-address field of the redundant portion in a block is used for storing a logical address assigned to data stored in the block.

[0097] In order to make the figures simple, the logical address in the redundant portion shown in Fig. 14B, the logical address in the redundant portion shown in Fig. 14C and physical addresses in the logical/physical address control table shown in Fig. 14D are each expressed in the hexadecimal format omitting the code 0X. Not specially used, hatched portions in the redundant portions in the blocks shown in Figs. 14B and 14C each have indeterminate contents.

[0098] As described above, the block shown in Fig. 14C is used for storing control data for a flash memory of 4 MB in this example. As shown in Fig. 14C, only 2 pages of the block are required for storing control data representing relations between logical addresses and physical addresses for segment 0 of the 4-MB flash memory including the extra blocks. The remaining pages of the block are not used. The data structure of the logical/physical address control table stored in these 2 pages is shown in Fig. 14D. 2 pages are required for storing control data representing relations between logical addresses and physical addresses for 1 segment. In the case of a flash memory comprising more segments, more pages in the block are thus used even though there are always unused pages in the block as shown in Fig. 14C except for a memory flash with a storage capacity of 128 MB shown in Fig. 13.

[0099] Fig. 14D is a diagram showing the logical/physical address control table containing control data stored in pages 0 and 1 of the block shown in Fig. 14C. It should be noted that the logical address 0×0000 is abbreviated simply to logical address 0 in order to make Fig. 14D simple and this abbreviation is applied to other logical addresses shown in Fig. 14D. Fig. 14D does not show explicitly which of the 2 bytes is used for storing the high-order byte of the physical address and which of the 2 bytes is used for storing the low-order byte of the physical address.

[0100] Physically, a flash memory with a storage capacity of 4 MB comprises 512 blocks as shown in Fig. 13. Since there are defective blocks in the flash memory, only a maximum of 493 blocks excluding the 2 boot blocks are allocated to logical addresses as shown in Fig. 14D.

[0101] As shown in Fig. 14A, the logical address 0×0000 is assigned to data stored in a block at the physical address 0×0003. In this case, the physical address 0×0003 is stored in 2 bytes in the logical/physical address control table of Fig. 14D which are allocated to the logical address 0. By the same token, the physical address 0×0002 is stored in the logical/physical address control table which are allocated to the logical address 1 and the physical address 0×0006 is stored in the logical/physical address control table which are allocated to the logical address 2. Similarly, the physical address 0×0005 is stored in the logical/physical address control table which are allocated to the logical address 3 and the physical address 0×0004 is stored in the logical/physical address control table which are allocated to the logical address 4. Likewise, the physical address 0×007b is stored in the logical/physical address control table which are allocated to the logical address 5 and the physical address 0×01FF is stored in the logical/physical address control table which are allocated to the logical address 493.

[0102] In an access to data in the planar memory 1, the file system converts a logical address specified in the access into a physical address of a block containing the data by using the logical/physical address control table. Assume, for example, that an application program issues an instruction to read out data sequentially from the logical addresses 0×0002, 0×0003 and 0× 0004 to the file system. In this case, the file system processes the instruction in accordance with the following procedure.

[0103] First of all, the planar memory 1 shown in Fig. 14A is searched for a block with a 0 control table flag in the redundant portion of page 0 thereof. As a result of the search, block 123 for storing the logical/physical address control table is found. In this case, since the logical addresses 0×0002, 0×0003 and 0×0004 are obviously associated with the first part of the logical/physical address control table and since the size of an embedded RAM for temporarily storing control data of the logical/physical address control table is small, only control data in page 0 of block 123 is read out. Then, by using the control data, the logical addresses 0×0002, 0×0003 and 0×0004 are converted into the physical addresses 0×0006, 0×0005 and 0×0004 respectively. Finally, the file system reads out pieces of data sequentially from blocks at the physical addresses 0×0006, 0×0005 and 0×0004 as requested by the instruction issued by the application program.

[0104] Relations between the storage capacity of a flash memory and the size of a logical/physical address control table are explained with reference back to Fig. 13.

[0105] As has been explained earlier with reference to Figs. 11A and 11B, the amount of information stored in the logical/physical address control table for controlling 1 segment is 1,024 bytes (or 1 KB) corresponding to 2 pages. Thus, for controlling a flash memory of 1 segment with a storage capacity of 4 MB as shown in Fig. 13, the size of the logical/physical address control table is 1 KB. For controlling a flash memory of 2 segments with a storage capacity of 8 MB, the logical/physical address control table occupies an area of 2 KB corresponding to 4 pages.

[0106] For controlling a flash memory of 4 segments (= 2,048 blocks) with a storage capacity of 16 MB, the logical/physical address control table occupies an area of 4 KB corresponding to 8 pages. For controlling a flash memory of 2 segments (= 1,024 blocks) with a storage

[redacted] [redacted]

[redacted] [redacted] [redacted]

capacity of 16 MB, the logical/physical address control table occupies an area of 2 KB corresponding to 4 pages.

[0107] For controlling a flash memory of 4 segments with a storage capacity of 32 MB, the logical/physical address control table occupies an area of4 KB corresponding to 8 pages. For controlling a flash memory of 8 segments with a storage capacity of 64 MB, the logical/physical address control table occupies an area of 8 KB corresponding to 16 pages. For controlling a flash memory of 16 segments with a storage capacity of 128 MB, the logical/physical address control table occupies an area of 16 KB corresponding to 32 pages.

[0108] By the way, in the logical/physical address control table of a file system with the conventional configuration for a flash memory, a virtually indeterminate value is used as a physical address associated with an unused logical address.

[0109] To put it concretely, take a logical/physical address control table shown in Fig. 12B as an example. In this example, logical addresses of 0X0000, 0X0001, 0X0002 and 0X0003 are already used and assigned to physical addresses of 0X0002, 0X0006, 0X0007 and 0X0008 respectively. That is to say, the physical addresses 0X0002, 0X0006, 0X0007 and 0X0008 at which pieces of data have already been stored are associated with the logical addresses 0X0000, 0X0001, 0X0002 and 0X0003 respectively.

[0110] On the other hand, if a logical address of 0X0004 is not used, an invalid value of 0XFFFF is used as a physical address associated with the logical address 0X0004. The invalid value 0XFFFF set as a physical address indicates that a storage area at this physical address is not used.

[0111] Thus, in an attempt to newly write data in the unused storage area allocated to the logical address 0X0004 by referring to the logical/physical address control table shown in Fig. 12B, the FAT file system typically searches for a physically unused block at a hierarchical level different from the logical/physical address control table before executing an operation to write the data into the block found in the search. Then, the contents of the logical/physical address control table are updated by cataloging the physical address of the block, in which the data was newly written, into the logical/physical address control table at a table entry associated with the logical address 0X0004.

[0112] However, the following problem is expected to arise in such an implementation of the logical/physical address control table.

[0113] Assume that data handled by the main apparatus is the so-called real time data observed along the time axis such as motion-picture data or audio data of a piece of music or the like.

[0114] In the main apparatus, the input data observed along the time axis is subjected to signal processing carried out in a real-time manner before being recorded in to the planar memory 1 as recording data.

[0115] If the implementation of the logical/physical address control table explained earlier with reference to Fig. 12B is adopted, in the operation to record the data into the planar memory 1, the planar memory 1 naturally needs to be searched for an unused block described above. In an operation to record data observed along the time axis such as the one described above, it is necessary to write the input data into the planar memory 1 at such an average speed that no data overflow occurs. The search carried out at that time for an unused block is extremely tough processing to be carried out by the microprocessor 109.

[0116] That is to say, in the present state of the art, it is very difficult to record real-time data into the planar memory 1. Practically, such data is merely recorded into a still-picture file or a text file which imposes no real-time requirements.

[0117] In order to solve the problem described above, in the logical/physical address control table provided by this embodiment, the physical address of a block controlled as an unused area is associated with an unused logical address. An example of the logical/physical address control table provided by this embodiment is shown in Fig. 12A.

[0118] In this example, logical addresses of 0X0000, 0X0001, 0X0002 and 0X0003 are already used and assigned to physical addresses of 0X0002, 0X0006, 0X0007 and 0X0008 respectively. That is to say, the physical addresses 0X0002, 0X0006, 0X0007 and 0X0008 at which pieces of data have already been stored are associated with the logical addresses 0X0000, 0X0001, 0X0002 and 0X0003 respectively as is the case with the example shown in Fig. 12B. In addition, a logical address of 0X0004 is not used as is the case with the example shown in Fig. 12B.

[0119] As shown in Fig. 12A, in this embodiment, however, a physical address of 0X0009 of a typical unused block replacing the physical address 0XFFFF is associated with the unused logical address 0X0004. In this example, only one unused block allocated to one unused logical address is shown. It should be noted that other unused blocks can be allocated to other unused logical addresses and the physical addresses of the other unused blocks are associated with the other unused logical addresses in the same way as the physical address 0X0009 is associated with the logical address 0X0004.

[0120] In a logical/physical address control table constructed in this way, an area at a physical address associated with a logical address can interpreted as a free area allocated to the logical address.

[0121] Thus, the FAT file system is capable of determining a physical address of an unused block allocated to a logical address in advance with reference to the logical/physical address control table in a recording operation and it is no longer necessary to execute processing to search for an unused block as is the case with the implementation of the logical/physical address control

[redacted] [redacted]

table shown in Fig. 12B. That is to say, with reference to the logical/physical address control table, it is possible to obtain a physical address associated with a logical address assigned to a free area by the FAT file system. Then, data is written into an unused block at the physical address by making an access to the block. As a result, the processing load borne by the microprocessor employed in the main apparatus is reduced substantially and, for example, the operation to record data observed along the time axis as described above can be carried out with ease. Also in an operation to record data requiring no real-time processing such as text-file data and still-picture data, the time it takes to write the data into the flash memory can of course be shortened by adopting the file system provided by the embodiment as shown in Fig. 12A in comparison with the conventional system.

[0122] A procedure of recording data executed by a recording/playback apparatus implemented by the embodiment is explained with reference to a flowchart shown in Fig. 15.

[0123] As shown in Fig. 15, the flowchart begins with a step S1 at which the non-volatile memory implemented as the planar memory 1 to undergo a recording operation is searched for a block used as a control-data area for storing control data of the logical/physical address control table before the recording operation is started. Actually, only the last segment of the non-volatile memory containing the logical/physical address control table is searched for a block with a 0 control table flag in the redundant portion of the first page or page 0 thereof shown in Fig. 14C.

[0124] The flow of the procedure then goes on to a step S2 to make a judgment as to whether or not the local memory of the recording/playback apparatus already contains some of the control data of the logical/physical address control table with an amount large enough for making an access to the non-volatile memory. If the outcome of the judgment indicates that the local memory already contains some of the control data of the logical/physical address control table with an amount large enough for making an access to the non-volatile memory, the flow of the processing goes on to a step S5. If the outcome of the judgment made at the step S2 indicates that the local memory does not contain control data of the logical/physical address control table necessary for making an access to the non-volatile memory, on the other hand, the flow of the processing goes on to a step S17. At the steps S17, S18 and S19, part of the control data currently stored in the local memory is swapped with other control data. At the step S17, a redundant portion of the control data currently stored in the local memory is generated with the control table flag in the case of this embodiment to indicate that the data to be written is control data of the logical/physical address control table. The control data is a part of the logical/physical address control table. A logical address assigned in advance to the logical/physical address control table is also recorded into the redundant portion.

[0125] At the step S18, the control data of the logical/physical address control table currently stored in the local memory and the redundant portion generated at the step S17 are written into the non-volatile memory at the logical address. As described earlier, the logical address is converted by the file system by using the logical/physical address control table into a physical address in the non-volatile memory at which the control data and the redundant portion are actually stored.

[0126] At the step S 19, some control data of the logical/physical address control table required for making an access to the non-volatile memory as judged at the step S2 is transferred from the non-volatile memory to the local memory. After the processing of the step S19 is completed, the flow of the procedure proceeds to the step S5.

[0127] At the step S5, an attribute of data to be written into the non-volatile memory is examined to make a judgment as to whether the data is control data or main data. If the outcome of the judgment indicates an attribute of main data, the flow of the procedure goes on to a step S8. If the outcome of the judgment indicates an attribute of control data, on the other hand, the flow of the procedure goes on to a step S3.

[0128] At the step S3, a redundant portion of the control data recognized at the step S5 is generated with the control table flag of the redundant portion in the case of this embodiment to indicate that the data to be written is control data of the logical/physical address control table. The control data is a part of the logical/physical address control table. A logical address assigned in advance to the logical/physical address control table is also recorded into the redundant portion.

[0129] The flow of the procedure than goes on to a step S4 at which the control data recognized at the step S5 and the redundant portion generated at the step S3 are written into the non-volatile memory at the logical address. As described earlier, the logical address is converted by the file system by using the logical/physical address control table into a physical address in the non-volatile memory at which the control data and the redundant portion are actually stored.

[0130] At the step S8, a next block into which the main data is to be written is determined by using part of the logical/physical address control table currently stored in the local memory. The flow of the procedure then goes on to a step S9 at which the number of a page into which the main data is to be written is initialized at 0.

[0131] The flow of the procedure then goes on to a step S10 at which main data of 1 page is input from the DSP 102.

[0132] At a step S11 a judgment is made as to whether or not the number of a page into which the main data is to be written is 0. If the page number is 0, the flow of the procedure goes on to a step S12. If the page number is not 0, on the other hand, the flow of the procedure goes on to a step S13. In this embodiment, a non-zero flag

[redacted] [redacted] [redacted]

number has a value in the range 1 to 15.

[0133] At the step 12, a redundant portion of the main data recognized is generated with the control table flag of the redundant portion set at 1 in the case of this embodiment to indicate that the data to be written is main data. A logical address assigned in advance to the logical/physical address control table is also recorded into the redundant portion.

[0134] The step S13 is similar to the step S12 except that the contents of a created redundant portion are arbitrary. The contents may be the values set at the step S12.

[0135] After the processing of the step S12 or S13 is completed, the flow of the procedure goes on to a step S14 at which the redundant portion generated at the step S12 or S13 and the 1-page main data obtained at the step S10 are written into a page of the block in the non-volatile memory. The block was determined at the step S8 and the page is indicated by a page number. which was initialized at the step S9.

[0136] The flow of the procedure then goes on to the step S15 at which the page number is incremented.

[0137] The flow of the procedure then goes on to a step S16 to make a judgment as to whether or not the page number incremented at the step S15 has reached the number of pages per block in the non-volatile memory. In this embodiment, the number of pages per block in the non-volatile memory is 16. Thus, a page number equal to 16 indicates that an operation to write data into a block unit has been completed. In this case, the flow of the procedure goes back to the step S2. If the page number is found smaller than 16 at the step S16, on the other hand, the flow of the procedure goes back to the step S10.

[0138] By carrying out the operations described above, the recording/playback apparatus is capable of recording main data into the planar memory 1.

[0139] The following description explains an operation to rewrite main data or control data with reference to Figs. 16, 17 and 18. In the following description, the main data and the control data are referred to simply as data.

[0140] In this embodiment employing the planar memory 1 implemented by a non-volatile flash memory, data is rewritten in block units. This is because, unlike an operation to write new data, in a flash memory, data is always rewritten into a block from which data has been erased before. As a characteristic of a flash memory, smallest physical storage units of erased data which are each called a cell are all set to "1". Thus, the smallest physical storage unit can be regarded as a bit of logic data. In an operation to write data into a flash memory, 0s are written only into cells corresponding to 0 bits of the data. To be more specific, such cells are each put into a state of electrically 0. Once a cell has been put into a state of electrically 0, the cell can not be restored to a state of electrically 1 even if a bit having a value of 1 is written into the cell. Such a cell can be restored to a state of electrically 1 only in a block erase operation. That is why updated data can not be stored correctly unless the data is written into an erased area with all cells or bits thereof restored to the initial value of 1. Also as described earlier, in this embodiment, updated data is not rewritten into the same area in order to prolong the life of the flash memory which may otherwise be shortened by repeated operations to write data into the same area. Instead, updated data is rewritten into a block which is currently unused. That is to say, in a rewrite operation, the updated data is moved or copied to the unused block from the block occupied by the data so far.

[0141] As described before, the logical/physical address control table has a configuration wherein each logical address is assigned permanently to a table entry and a dynamically variable physical address associated with the logical address is stored in the table entry. In an operation to rewrite data into an unused block, it is necessary to know the physical address of the unused block. As shown in Fig. 11B, a portion of the logical/physical address control table for segment 1 comprises 2-byte table entries 0 to 495 allocated permanently to logical addresses of 0 to 495 respectively. A physical address associated with a logical address assigned to a table entry is recorded in the table entry. Table entries after the table entry 495 are each used for storing the physical address of an extra block. The head of the table entries for storing physical addresses of extra blocks is univocally determined by a segment number.

[0142] Fig. 16 shows a flowchart representing a method of determining a block to be used in an operation to rewrite data as a write target block and related processing carried out on the logical/physical address control table.

[0143] As shown in the figure, the flowchart begins with a step S21 at which a physical address is selected arbitrarily from the table entries for extra blocks shown in Fig. 11B as the physical address of a write target block. A write target block is a block into which data is to be actually written.

[0144] At a step S22, the physical address associated with the logical block assigned to a block subjected to the rewrite operation is found from the logical/physical address control table shown in Fig. 11B. It should be noted that data will be actually written into a write target block instead of the block subjected to the rewrite operation.

[0145] At a step S23, the physical address of the write target block selected at the step S21 is cataloged into a table entry from which the physical address of the block subjected to the rewrite operation was found at the step S22. At a step S24, the physical address of the block subjected to the rewrite operation found at the step S22 is cataloged into a table entry for an extra block from which the physical address of the write target block was found at the step S21.

[0146] Next, a procedure for rewriting main data is ex-

[redacted] [redacted]

[redacted] [redacted] [redacted]

plained with reference to a flowchart shown in Fig. 17.

[0147] As shown in the figure, the flowchart begins with a step S31 at which a write target block is found in the same way as the step S21 of the flowchart shown in Fig. 16. At a step S32, data is erased from the write target block. Data needs to be erased from the write target block for the reason already described earlier.

[0148] At a step S33, an update status flag of the write source block shown in Fig. 7F is set. The write source block is the block subjected to the rewrite operation. The processing of the step S33 is carried out to cope with an accident such as a power-supply failure. In the event of a power-supply failure, even if there are blocks which have same logical address in the same segment, with the update status flag set, the write source block can be identified with ease. Once the write source block is identified, it will be easy to re-determine the write target block.

[0149] At a step S34, the contents of the logical/physical address control table are updated in the same way as the steps S22, S23 and S24 of the flowchart shown in Fig. 16. In this way, the physical address of the write target block is cataloged a table entry of the logical/physical address control table as a physical address associated with a logical address assigned to the table entry.

[0150] At a step S35, the original data is updated while the updated data is being rewritten into the write target block.

[0151] Next, a procedure for rewriting the control data itself, that is, the logical/physical control table, is explained with reference to a flowchart shown in Fig. 18.

[0152] As shown in the figure, the flowchart begins with a step S41 at which a write target block is found in the same way as the step S21 of the flowchart shown in Fig. 16. At a step S42, data is erased from the write target block. Data needs to be erased from the write target block for the reason already described earlier.

[0153] At a step S43, an update status flag of the write source block shown in Fig. 7F is set. The processing of the step S43 is carried out to cope with an accident such as a power-supply failure. In the event of a power-supply failure, even if there are blocks which have same logical address in the same segment, with the update status flag set, the write source block can be identified with ease. Once the write source block is identified, it will be easy to re-determine the write target block.

[0154] At a step S44, the control table flag in the redundant portion of page 0 of the write target block is reset to 0 to indicate that this write target block is a block for storing the logical/physical address control table. As described earlier, a flash memory is characterized in that, once a cell of the flash memory is reset to 0, it can not be restored to 1 unless data in the block including the cell is deleted by a block erasure. In this way, this write target block is recognized as a block for storing the logical/physical address control table till the table is erased from the block.

[0155] At a step S45, the contents of the logical/physical address control table are updated in the same way as the steps S22, S23 and S24 of the flowchart shown in Fig. 16. In this way, the physical address of the write target block is cataloged in a table entry in the logical/physical address control table as a physical address associated with a logical address assigned to the table entry.

[0156] At a step S46, the original data is updated while the updated data is being rewritten into the write target block.

3. System Configuration

[0157] Fig. 3 is a block diagram showing the configuration of a main apparatus which is capable of writing and reading out data into and from the planar memory 1 provided by the embodiment of the present invention explained so far. The main apparatus 100 as shown in Fig. 3 and the planar memory 1 constitute an electronic equipment system implemented by the embodiment. In this case, the main apparatus 100 is capable of writing and reading out at least audio data into and from the planar memory 1.

[0158] The configuration of the main apparatus 100 includes a case mounting/dismounting mechanism 120 for mounting and dismounting the planar memory 1 onto and from the main apparatus 100. Data is exchanged between the planar memory 1 mounted on the case mounting/dismounting mechanism 120 and the microprocessor 109 through a host interface IC 101.

[0159] In addition, the main apparatus 100 also has typically a microphone 103 for inputting an audio signal representing voice or sound. The analog audio signal is then supplied to a DSP (Digital Signal Processor) 102 by way of a microphone amplifier 104. In the DSP 102, the input analog audio signal is converted into digital audio data subjected to necessary signal processing such as an encoding process before being supplied to the microprocessor 109 as recording data.

[0160] The microprocessor 109 is capable of carrying out processing to record the recording data into the planar memory 1 by way of the host interface IC 101.

[0161] In addition, the microprocessor 109 reads out audio data recorded in the planar memory 1 through the host interface IC 101 and supplies the data to the DSP 102.

[0162] In the DSP 102, the data received from the microprocessor 109 is subjected to necessary signal processing such as demodulation. The DSP 102 finally supplies an analog audio signal obtained as a result of the processing to a speaker amplifier 105. The speaker amplifier 105 amplifies the analog audio signal received from the DSP 102 and supplies an amplified signal to a speaker 106. In this way, a playback audio signal is output.

[0163] By controlling a display driver 107, the microprocessor 109 is capable of displaying a desired picture

[redacted] [redacted]

on a display unit 108. Assume that picture data representing a motion picture or a still picture has been stored in the planar memory 1. In this case, the microprocessor is capable of displaying the picture data read out from the planar memory 1 on the display unit 108.

[0164] An operation unit 112 is provided with a variety of keys to be used by the user to carry out a variety of operations for the main apparatus 100. The microprocessor 109 receives a command entered by the user by operating the operation unit 112 and executes necessary control processing in accordance with the command.

[0165] It should be noted that the configuration of the main apparatus 100 as shown in Fig. 3 is typical to the bitter end. That is to say, the main apparatus 100 is not limited to the typical configuration shown in the figure. In other words, the main apparatus 100 can be implemented as an electronic apparatus of any type as long as the electronic apparatus is capable of exchanging data with the planar memory I provided by the embodiment.

[0166] In order to implement operations to record and play back (or write and read out) data into and from the aforementioned planar memory 1 by means of the main apparatus 100 with a configuration shown in Fig. 3, the logical/physical address control table is required to be referred to by the FAT file system as described above.

[0167] Fig. 4 is an explanatory diagram conceptually showing an interface between the microprocessor 109 employed in the main apparatus 100 based on the configuration shown in Fig. 3 and the logical/physical address control table stored in the planar memory 1.

[0168] For example, when the planar memory 1 provided by the embodiment is mounted on the main apparatus 100, the microprocessor 109 reads out necessary data in the logical/physical address control table TB from the planar memory 1 through the host interface IC 101 and stores the data into an internal RAM 111.

[0169] The configuration of the conventional system is shown in Fig. 1 to be compared with the system provided by the embodiment shown in Fig. 3. In the conventional system, the logical/physical address control table is not stored in the planar memory 1A as is the case with the configuration shown in Fig. 1. It should be noted that components of the configuration shown in Fig. 1 identical with those shown in Fig. 3 are denoted by the same reference numerals as the latter and their explanation is not repeated.

[0170] The system configuration shown in Fig. 1 is different from that shown in Fig. 3 in that, in the case of the former, an external RAM 113 is provided in the main apparatus 100A. The RAM 113 is connected to the microprocessor 109.

[0171] For a purpose of comparison with the interface of the embodiment shown in Fig. 4, Fig. 2 shows an interface between the microprocessor 109 employed in the main apparatus 100A and the planar memory 1A in the conventional system configuration shown in Fig. 1.

[0172] The RAM 113 is used for storing the logical/physical address control table. When the planar memory 1A with no logical/physical address control table stored therein is mounted, the microprocessor 109 makes an access to the planar memory 1A by way of the host interface IC 101 to check data contents of the memory 1A in order to execute processing to construct a logical/physical address control table. The logical/physical address control table TB constructed in this way is then stored in the RAM 113.

[0173] Typically, the RAM 111 embedded in the microprocessor 109 has a storage capacity of about several tens of KB at the most. It is thus absolutely impossible to store a logical/physical address control table with a size up to 16 KB in the RAM 111 since the existence of a logical/physical address control table in the RAM 111 will provide a hindrance to other processing. Some microprocessors 109 even have a RAM 111 with a size smaller than the logical/physical address control table. That is to say, in the case of a configuration wherein a logical/physical address control table is constructed and saved in the main apparatus, it is not realistic to store the logical/physical address control table in the RAM 111. That is why the external RAM 113 is required.

[0174] On the other hand, the embodiment adopts a configuration wherein the logical/physical address control table is stored in the planar memory 1. In this case, only some necessary data of the logical/physical address control table is simply read out from the planar memory 1 and stored in the embedded RAM 111 as is explained earlier with reference to Fig. 4. For example, the microprocessor 109 needs only data of the logical/physical address control table for 1 segment which occupies an area of 1,024 bytes in the logical/physical address control table as shown in Fig. 11. The size of such data will hardly have an impact on the RAM 111 that causes some problems.

[0175] For this reason, the external RAM 113 can be eliminated from the embodiment shown in Fig. 3. As a result, the cost of the main apparatus 100 can be reduced and the power consumption can also be decreased by the amount of power required to drive the external RAM 113.

[0176] In addition, in the case of the embodiment, the microprocessor 109 employed in the main apparatus 100 is relieved from the processing to construct a logical/physical address control table. Thus, there is no longer required a time to wait for the processing to construct a logical/physical address control table to be completed. As a result, for example, the embodiment manages to shorten the time it takes to carry out build-up processing of the file system when the planar memory 1 in comparison with the conventional system.

[0177] Furthermore, in the logical/physical address control table provided by the embodiment, the physical address of each unused block is associated with an unused logical address as described earlier with reference to Figs. 12A and 12B. Thus, an access to an unused block can be made through the FAT file system in simple processing and in a short period of time in comparison with the conventional system. This fast processing is particularly effective for a configuration of Fig. 3 adopted by the main apparatus 100 for recording data requiring real-time processing such as audio data.

[0178] It should be noted that the embodiment of the present invention is not limited to what is described above. If necessary, changes and modifications can be made to the embodiment. For example, the storage device provided by the present invention is not limited to the external shape shown in Figs. 5A, 5B, 5C and 5D. The storage device can be designed into any other external shape. In addition, for example, detailed prescriptions of the format of the file system described above can also be changed in accordance with actual applications. Moreover, variations of the storage capacity of the flash memory are not limited to the data shown in Fig. 13.

[0179] As described above, the logical/physical address control table is stored in the storage device.

[0180] Thus, it is not necessary to carry out processing to construct a logical/physical address control table. As a result, at least, the time it takes to complete the build-up process of the file system can be shortened. To put it concretely, while the user normally has to wait for the main apparatus to enter a state of being capable of writing and reading data into and from the storage device, for example, after the storage device is mounted on the main apparatus, the time it takes to wait for such a state can therefore be shortened in the case of the embodiment. As a result, the user is allowed to use the electronic equipment system more in a way the user likes.

[0181] In addition, with such a configuration, for example, the main apparatus merely needs to read out only some necessary data of the logical/physical address control table from the storage device and store the data typically in a storage area of the RAM embedded in the microprocessor employed in the main apparatus wherein the table occupies only a small area of the embedded RAM so that the operation to obtain the necessary data almost provides no additional load to the microprocessor.

[0182] Thus, since it is not necessary to provide the main apparatus with an external RAM including a storage area allocated to all data of the logical/physical address control table, the cost of the main apparatus can be reduced accordingly. In addition, the power consumption can also be decreased by an amount of electric power required to drive the external RAM.

[0183] In the present invention, data is written into a page which is one of storage units of the flash memory in a uniform format without regard to the type of the data. Each page always contains data recorded therein and a redundant portion showing attributes of the data as a pair. A plurality of adjacent pages constitute a block. Pieces of data stored in pages constituting a block have the same attributes. Thus, by checking only the redundant portion for storing data attributes in page 0 which is the first page of a block, for example, it is possible to know the attributes of all pieces of data in the block. In addition, since the data format is uniform regardless of the data attributes, it is not necessary to provide a means and a method for generating a page for each data attribute. Moreover, also with regard to generation of a redundant portion to be stored in a page, it is not necessary to provide a means and a method for generating redundant portion for each data attributes forming a pair with data since the format of the redundant portion is uniform. On the top of that, the data structures of blocks and pages are uniform independently of the attributes of data stored therein. It is thus not necessary to provide a plurality of reading means and reading methods each suitable for a block when designing a playback apparatus for reproducing data from pages and blocks. This also means that the number of circuit blocks and the number of program processing steps can be reduced substantially. Thus, there is exhibited an effect of addition of functions to a playback apparatus in equipment designed to have a small size, a small weight and little power consumption like the apparatus provided by the present invention.

[0184] Furthermore, according to the present invention, information for identifying the logical/physical address control table is also recorded in a block of the storage device for storing the logical/physical address control table. Thus, in an operation to search the storage device for a block containing the logical/physical address control table, the logical/physical address control table can be identified among data stored in the storage device.

[0185] Moreover, according to the present invention, an unused logical address is cataloged in the logical/physical address control table stored in the storage device by assigning the unused logical address to the physical address of an unused block, that is, a block with no data recorded therein. Thus, the location of an unused block can be determined readily by referencing the physical address of the block without the need to search for an unused block at another hierarchical layer. That is to say, since the processing to search for an unused block is not required in an operation to write data, the data can be written into an unused block at a high speed as such a light processing load. This fast processing is particularly effective for a case in which data to be recorded is data observed along the time axis such as audio data or motion-picture data which requires real-time processing.

Claims

1. A non-volatile memory having of a plurality of blocks each serving as a data deletion unit and comprising a plurality of adjacent pages each having a fixed

[redacted] [redacted] [redacted]

length and serving as a data read/write unit wherein a storage area of said non-volatile memory comprises:

a main-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording main data; and
a control-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording control data representing relations associating logical addresses with physical addresses wherein said logical addresses are assigned to pieces of data written into said blocks and said physical addresses show a physical layout order of said blocks.

2. A non-volatile memory according to claim 1 wherein the length of main data occupying said pages of said main-data area is equal to the length of control data occupying said pages of said control-data area.

3. A non-volatile memory according to claim 1 or 2, wherein said control data is an array of physical addresses laid out in an order determined by logical addresses with which said physical addresses are associated.

4. A non-volatile memory according to claim 1, 2 or 3, wherein said control data is stored in one of said blocks in close proximity to the end of said storage area of said non-volatile memory.

5. A non-volatile memory according to claim 1, 2, 3 or 4, wherein an attribute storing area of a storage area containing said control data includes an attribute indicating that data stored in said storage area is said control data.

6. A non-volatile memory according to any one of the preceding claims, wherein a block from which data was deleted is treated as an unused block.

7. A non-volatile memory according to claim 6 wherein the physical address of a block treated as an unused block is associated with a logical address assigned to said unused block.

8. A non-volatile memory according to claim 6 or 7, wherein, when new data is written supposedly into an occupied block in which old data has already been written to replace said old data, said new data is written into an unused block and said old data is deleted from said occupied block.

9. A recording apparatus for recording data into a non-volatile memory having a plurality of blocks each serving as a data deletion unit and comprising a plurality of adjacent pages each having a fixed length and serving as a data read/write unit wherein a storage area of said non-volatile memory comprises:

a main-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording main data; and
a control-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording control data representing relations associating logical addresses with physical addresses wherein said logical addresses are assigned to pieces of data written into said blocks and said physical addresses show a physical layout order of said blocks,
said recording apparatus comprising:
an attribute determining means for determining whether data to be written into said non-volatile memory is main data or control data;
an identifier generating means for generating an identifier indicating whether said data to be written into said non-volatile memory is main data or control data in accordance with a result of determination output by said attribute determining means; and
a memory control means for synthesizing said data to be written into said non-volatile memory and said identifier output by said identifier generating means and writing synthesized data into said non-volatile memory.

10. A recording apparatus according to claim 9 further including a local memory for temporarily holding a portion of said control data read out from said non-volatile memory.

11. A recording apparatus according to claim 10 wherein the amount of said portion of said control data held in said local memory is large enough for making an access to said non-volatile memory.

12. A recording apparatus according to claim 9, 10 or 11, further having a search means for searching said non-volatile memory for one of said pages treated as an unused page wherein, when first data existing in said non-volatile memory is replaced by second data, said search means searches said non-volatile memory for an unused page and then said memory control means writes said second data into said unused page found by said search means and deletes said first data.

[redacted] [redacted]

[redacted] [redacted] [redacted]

13. A recording method for recording data into a non-volatile memory having of a plurality of blocks each serving as a data deletion unit and comprising a plurality of adjacent pages each having a fixed length and serving as a data read/write unit wherein a storage area of said non-volatile memory comprises:

    a main-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording main data; and
    a control-data area comprising any one of said blocks comprising a plurality of said adjacent pages each used for recording an identifier for distinguishing main data and control data from each other and for recording control data representing relations associating logical addresses with physical addresses wherein said logical addresses are assigned to pieces of data written into said blocks and said physical addresses show a physical layout order of said blocks, said recording method comprising:
    an attribute determining step of determining whether data to be written into said non-volatile memory is main data or control data;
    an identifier generating step of generating an identifier indicating whether said data to be written into said non-volatile memory is main data or control data in accordance with a result of determination output at said attribute determining step; and
    a step of synthesizing said data to be written into said non-volatile memory and said identifier output at said identifier generating step and writing synthesized data into said non-volatile memory.

14. A recording method according to claim 13, further comprising the step of using a local memory for temporarily holding a portion of said control data read out from said non-volatile memory.

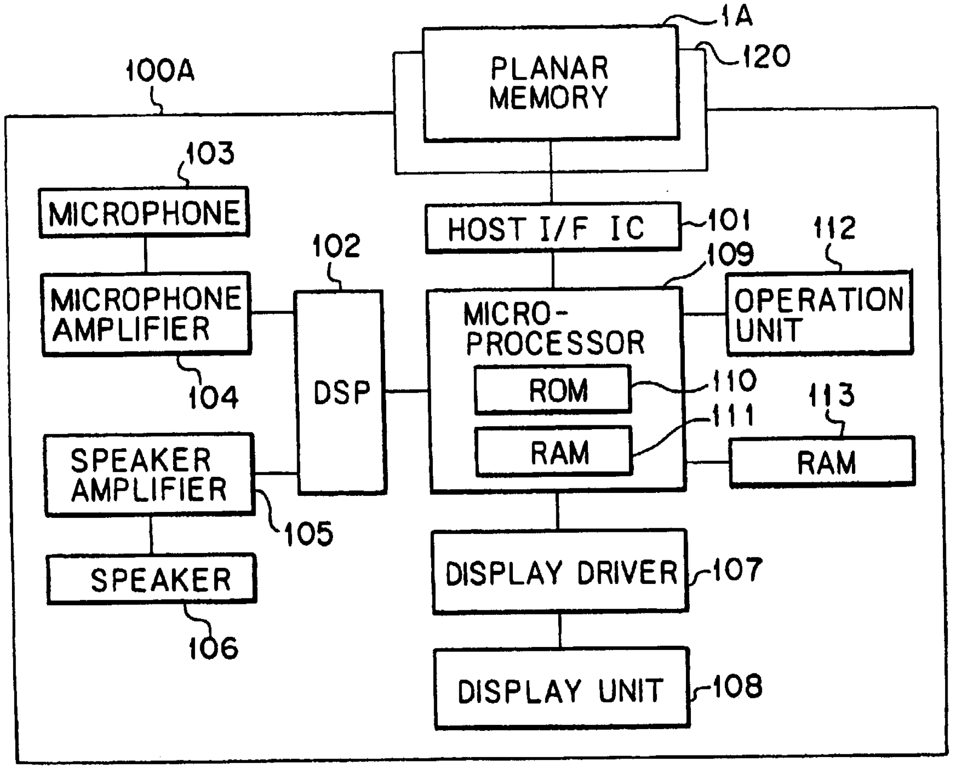
15. A recording method according to claim 14 wherein the amount of said portion of said control data held in said local memory is large enough for making an access to said non-volatile memory.

16. A recording method according to claim 13, 14 or 15, whereby first data existing in said non-volatile memory is replaced by second data by executing the steps of:

    searching said non-volatile memory for one of said pages treated as an unused page; and
    writing said second data into said unused page found at said searching step and deleting said first data.

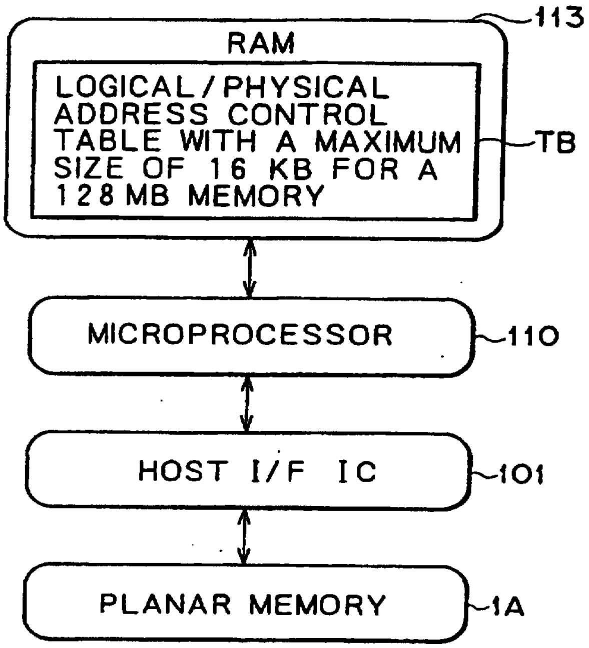
[redacted] [redacted]

FIG.1
PRIOR ART

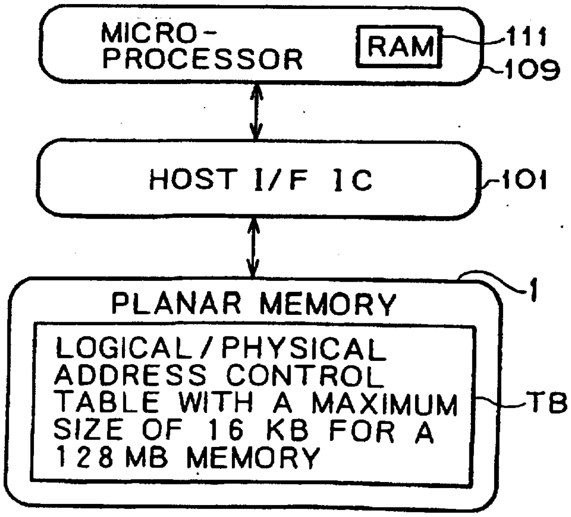

1A
120
PLANAR MEMORY
100A
103
MICROPHONE
HOST I/F IC
101
109
112
102
MICROPHONE AMPLIFIER
DSP
MICRO-PROCESSOR
OPERATION UNIT
ROM
110
111
113
104
RAM
RAM
SPEAKER AMPLIFIER
105
SPEAKER
DISPLAY DRIVER
107
106
DISPLAY UNIT
108

[redacted] [redacted] [redacted]

F I G. 2
PRIOR ART


RAM
LOGICAL/PHYSICAL ADDRESS CONTROL TABLE WITH A MAXIMUM SIZE OF 16 KB FOR A 128 MB MEMORY
113
TB
MICROPROCESSOR
110
HOST I/F IC
101
PLANAR MEMORY
1A

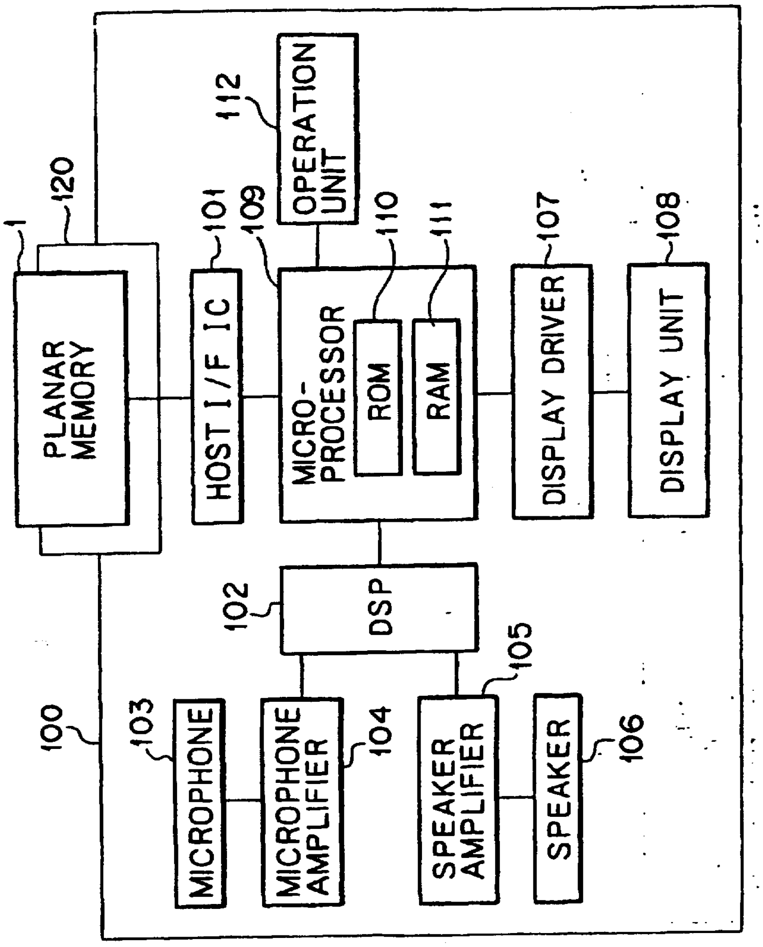
F I G. 4


MICRO-PROCESSOR
RAM
111
109
HOST I/F IC
101
PLANAR MEMORY
LOGICAL/PHYSICAL ADDRESS CONTROL TABLE WITH A MAXIMUM SIZE OF 16 KB FOR A 128 MB MEMORY
1
TB

[redacted] [redacted]

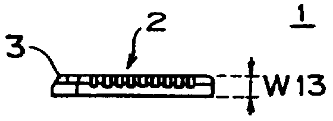
[redacted] [redacted] [redacted]

F I G. 3

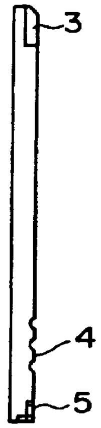

1
120
PLANAR MEMORY
101
HOST I/F IC
109
MICRO-PROCESSOR
110
ROM
111
RAM
112
OPERATION UNIT
107
DISPLAY DRIVER
108
DISPLAY UNIT
102
DSP
100
103
MICROPHONE
104
MICROPHONE AMPLIFIER
105
SPEAKER AMPLIFIER
106
SPEAKER

[redacted] [redacted]

FIG. 5A

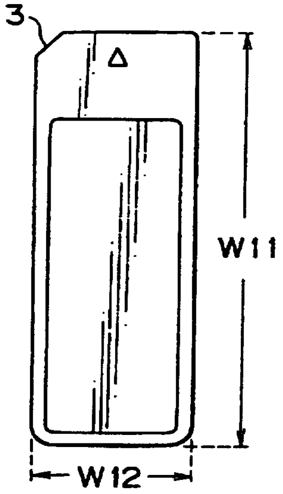


FRONT VIEW



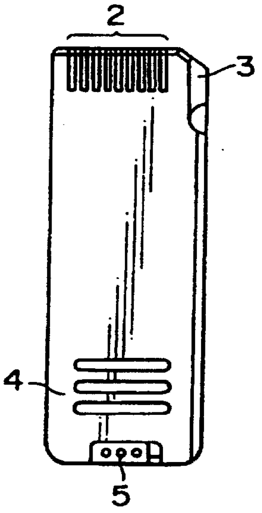
SIDE VIEW

FIG. 5C



TOP VIEW

FIG. 5B

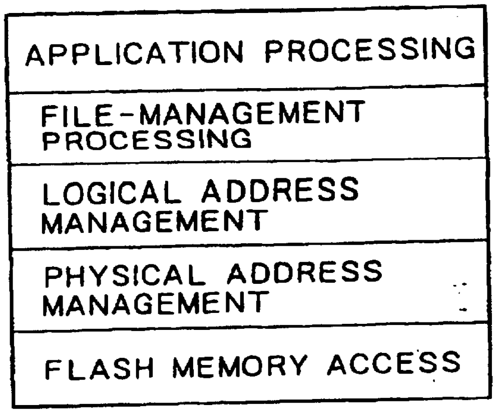


BOTTOM VIEW

FIG. 5D

[redacted] [redacted] [redacted]

F I G. 6


APPLICATION PROCESSING
FILE-MANAGEMENT PROCESSING
LOGICAL ADDRESS MANAGEMENT
PHYSICAL ADDRESS MANAGEMENT
FLASH MEMORY ACCESS

PROCESSING HIERARCHY OF A FILE SYSTEM

F I G. 8

MANAGEMENT FLAG

BIT	DEFINITION
7	RESERVED
6	RESERVED
5	ACCESS PERMISSION (1 : FREE, 0 : READ PROTECTED)
4	COPY PROHIBITED SPECIFICATION (1 : OK, 0 : NG)
3	CONTROL-TABLE FLAG (1 : NOT A TABLE BLOCK, 0 : TABLE BLOCK) * VALID ONLY FOR THE LAST SEGMENT
2	SYSTEM FLAG (1 : USER BLOCK, 0 : BOOT BLOCK)
1	RESERVED
0	RESERVED

[redacted] [redacted]

[redacted] [redacted] [redacted]

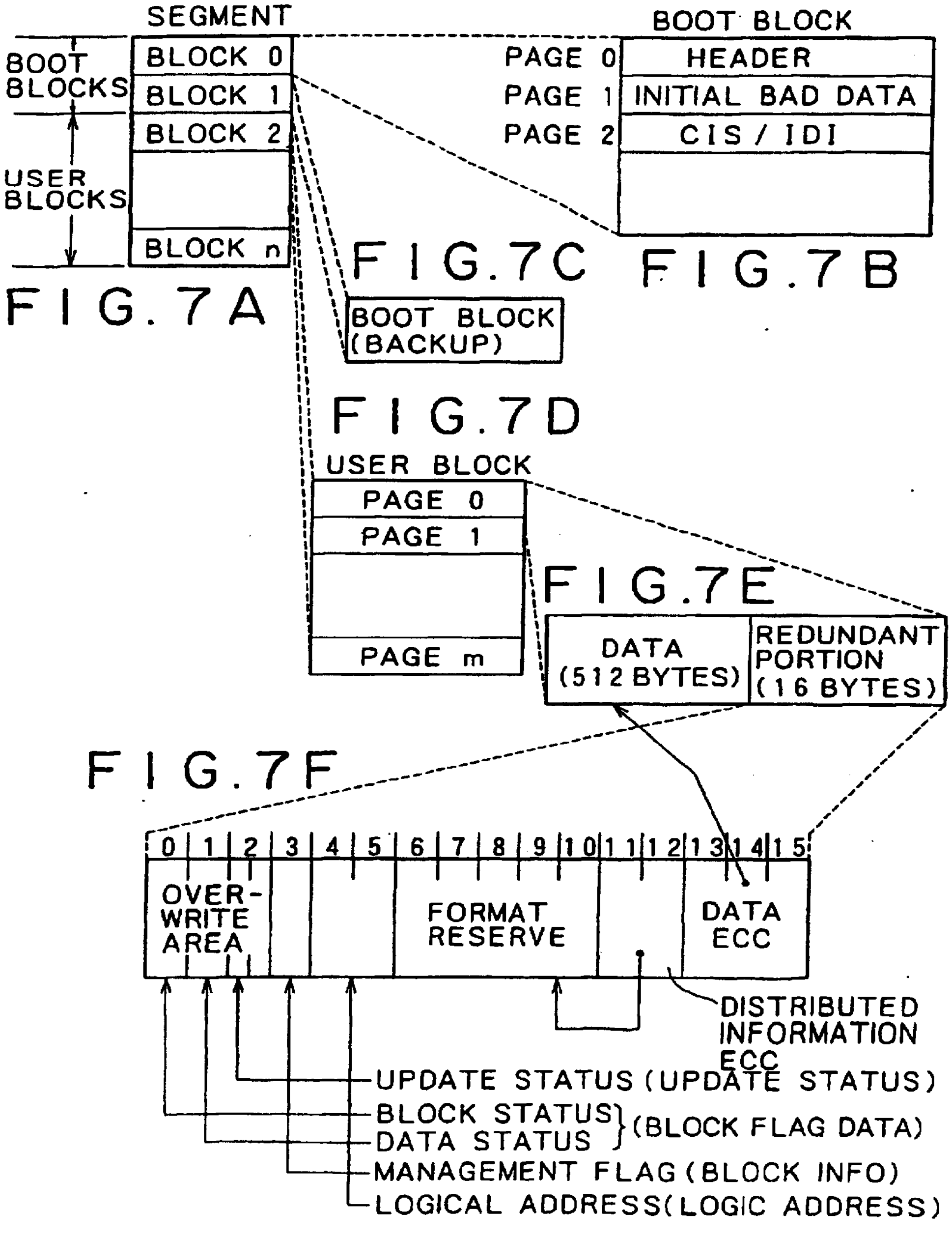


FIG.7A

FIG.7B

FIG.7C

FIG.7D

FIG.7E

FIG.7F

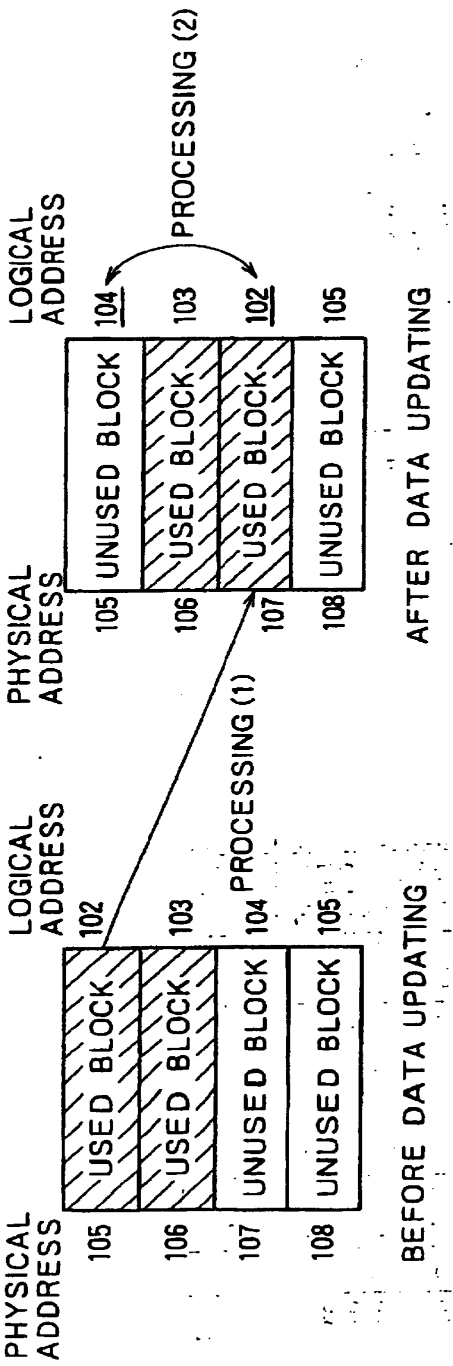
SEGMENT
BOOT BLOCKS
USER BLOCKS
BLOCK 0
BLOCK 1
BLOCK 2
BLOCK n
BOOT BLOCK
PAGE 0
PAGE 1
PAGE 2
HEADER
INITIAL BAD DATA
CIS / IDI
BOOT BLOCK (BACKUP)
USER BLOCK
PAGE 0
PAGE 1
PAGE m
DATA (512 BYTES)
REDUNDANT PORTION (16 BYTES)
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
OVER-WRITE AREA
FORMAT RESERVE
DATA ECC
DISTRIBUTED INFORMATION ECC
UPDATE STATUS (UPDATE STATUS)
BLOCK STATUS
DATA STATUS
(BLOCK FLAG DATA)
MANAGEMENT FLAG (BLOCK INFO)
LOGICAL ADDRESS(LOGIC ADDRESS)

[redacted] [redacted]

[redacted] [redacted] [redacted]

FIG.9A

FIG.9B


PHYSICAL ADDRESS
105
106
107
108
USED BLOCK
USED BLOCK
UNUSED BLOCK
UNUSED BLOCK
LOGICAL ADDRESS
102
103
104
105
PROCESSING (1)
BEFORE DATA UPDATING
PHYSICAL ADDRESS
105
106
107
108
UNUSED BLOCK
USED BLOCK
USED BLOCK
UNUSED BLOCK
LOGICAL ADDRESS
104
103
102
105
PROCESSING (2)
AFTER DATA UPDATING

[redacted] [redacted]

[redacted] [redacted] [redacted]

F I G.10

PHYSICAL ADDRESS (2 BYTES)			LOGICAL ADDRESS (2 BYTES)
0×00	0×03	←	LOGICAL ADDRESS 0×0000
0×00	0×04	←	LOGICAL ADDRESS 0×0001
0×00	0×04	←	LOGICAL ADDRESS 0×0002
0×00	0×05	←	LOGICAL ADDRESS 0×0003
0×01	0×A8	←	LOGICAL ADDRESS 0×0004
0×00	0×06	←	LOGICAL ADDRESS 0×0005 ↓

PHYSICAL ADDRESSES ASSOCIATED WITH LOGICAL ADDRESSES ARE STORED IN AN ORDER THE LOGICAL ADDRESSES ARE ARRANGED.

[redacted] [redacted]

FIG.11 A

LOGICAL/PHYSICAL ADDRESS CONTROL TABLE

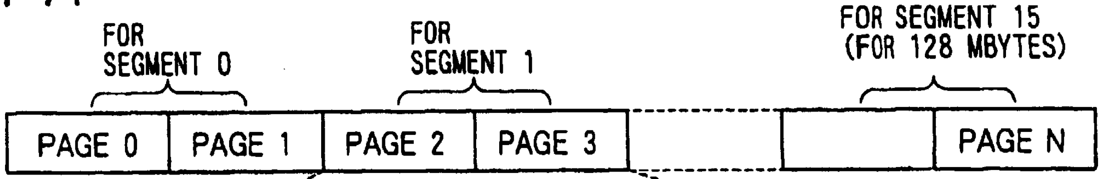
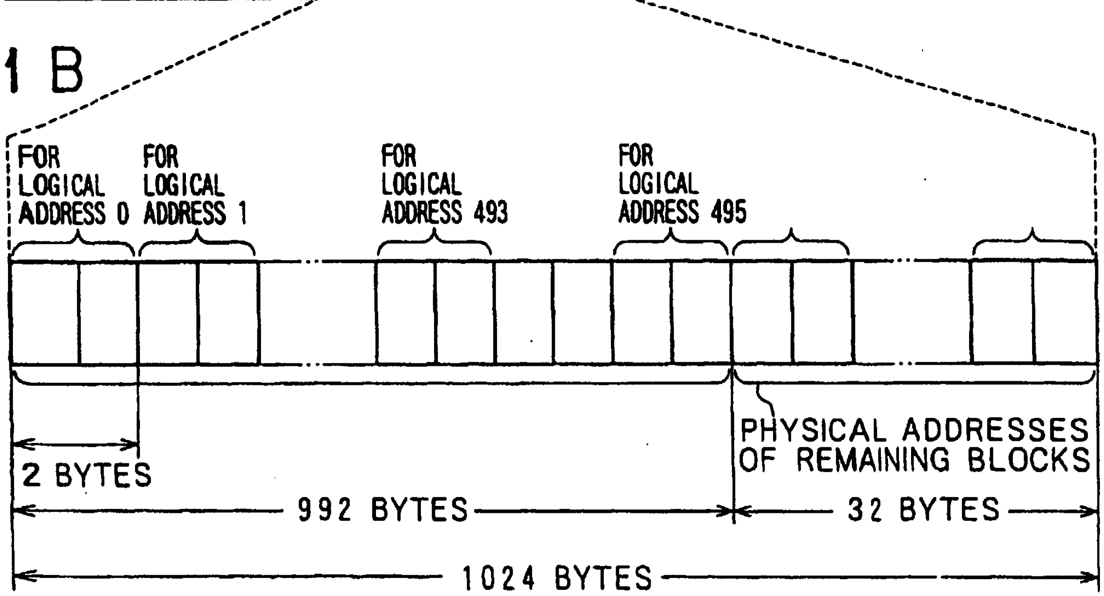

FOR SEGMENT 0
FOR SEGMENT 1
FOR SEGMENT 15 (FOR 128 MBYTES)
PAGE 0
PAGE 1
PAGE 2
PAGE 3
PAGE N

FIG.11 B


FOR LOGICAL ADDRESS 0
FOR LOGICAL ADDRESS 1
FOR LOGICAL ADDRESS 493
FOR LOGICAL ADDRESS 495
PHYSICAL ADDRESSES OF REMAINING BLOCKS
2 BYTES
992 BYTES
32 BYTES
1024 BYTES

[redacted] [redacted] [redacted]

FIG.12A

EMBODIMENT

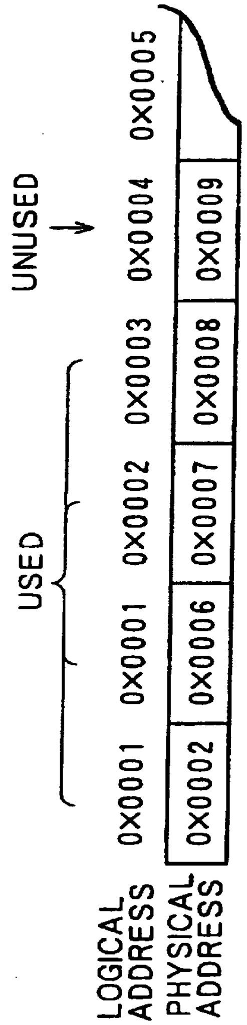
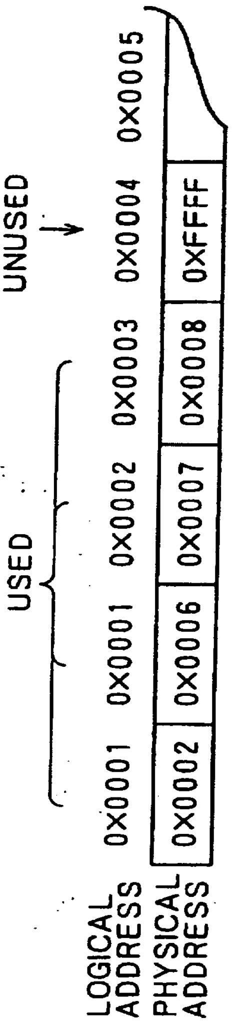

USED
UNUSED
LOGICAL ADDRESS
PHYSICAL ADDRESS
0X0001
0X0001
0X0002
0X0003
0X0004
0X0005
0X0002
0X0006
0X0007
0X0008
0X0009

FIG.12B

PRIOR ART


USED
UNUSED
LOGICAL ADDRESS
PHYSICAL ADDRESS
0X0001
0X0001
0X0002
0X0003
0X0004
0X0005
0X0002
0X0006
0X0007
0X0008
0XFFFF

[redacted] [redacted]

F I G.13

FLASH MEMORY CAPACITY	BLOCK COUNT	BLOCK SIZE	PAGE SIZE	LOGICAL/PHYSICAL ADDRESS CONTROL TABLE SIZE
4MB	512(1 SEGMENT)	8KB(16pages)	(512+16)B	1KB(2pages)
8MB	1024(2 SEGMENTS)	8KB(16pages)	(512+16)B	2KB(4pages)
16MB	2048(4 SEGMENTS)	8KB(16pages)	(512+16)B	4KB(8pages)
	1024(2 SEGMENTS)	16KB(32pages)	(512+16)B	2KB(4pages)
32MB	2048(4 SEGMENTS)	16KB(32pages)	(512+16)B	4KB(8pages)
64MB	4096(8 SEGMENTS)	16KB(32pages)	(512+16)B	8KB(16pages)
128MB	8192(16 SEGMENTS)	16KB(32pages)	(512+16)B	16KB(32pages)

[redacted]

[redacted]

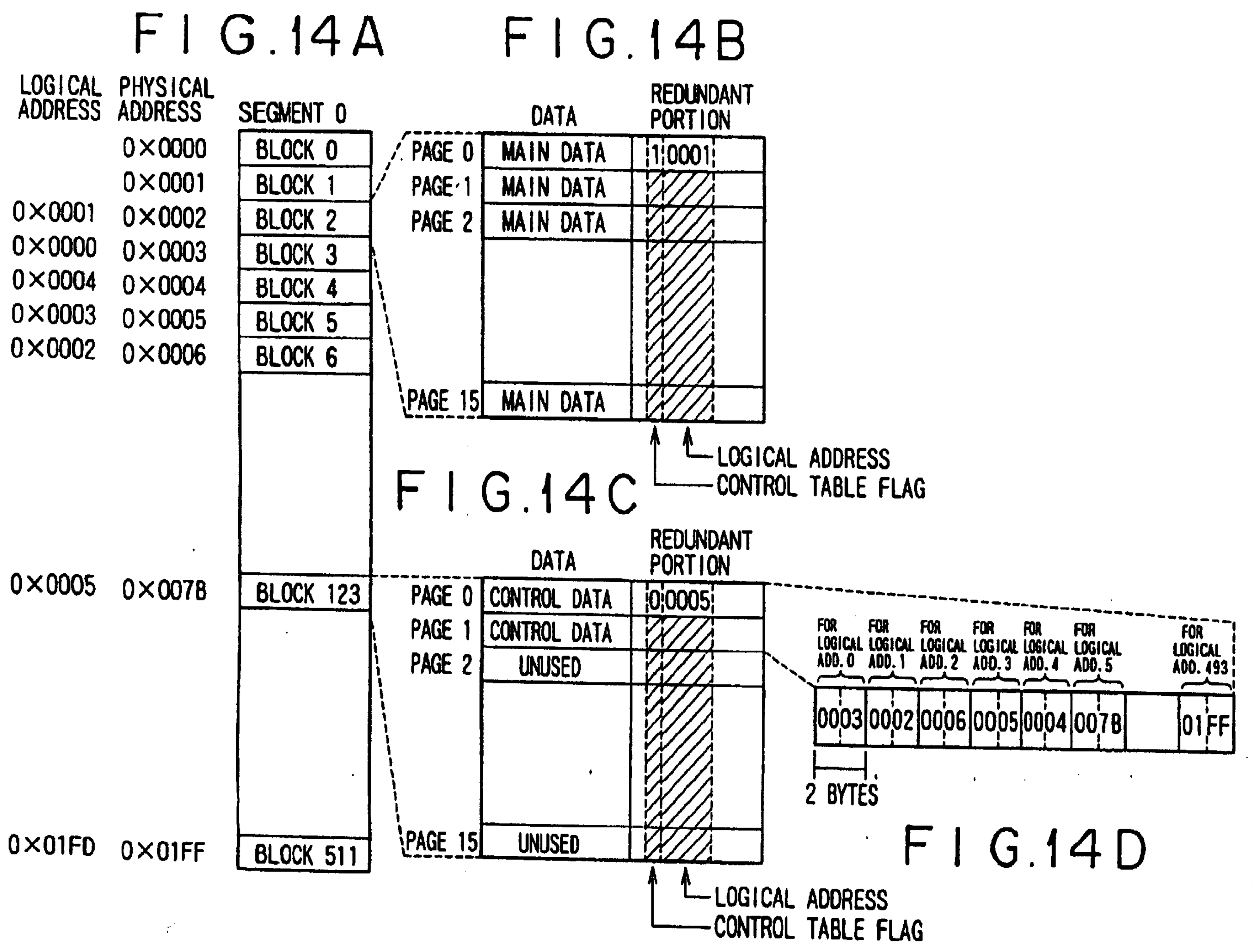
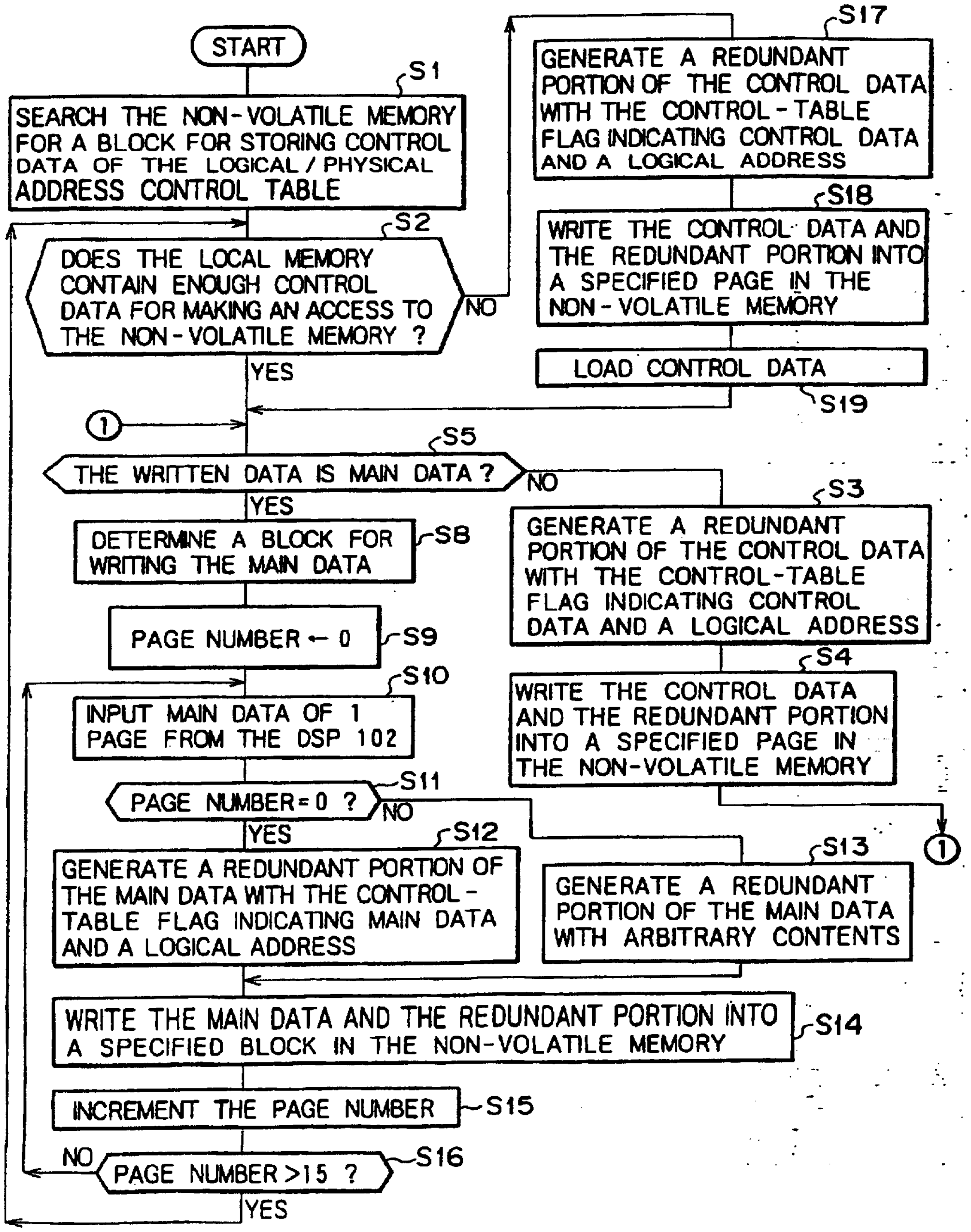


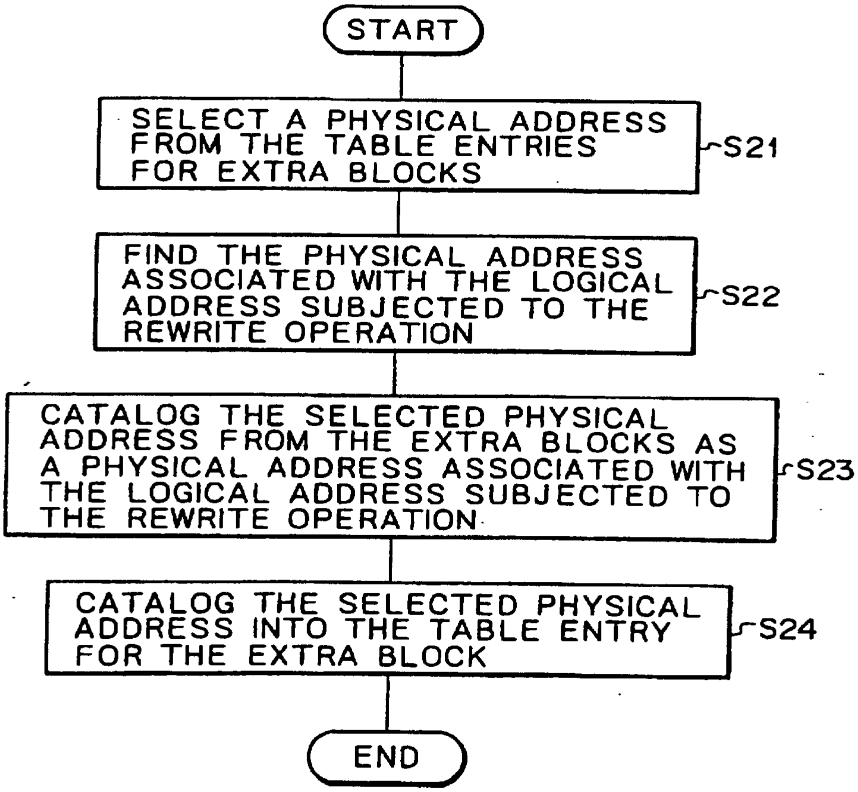
FIG.14A
FIG.14B
FIG.14C
FIG.14D
LOGICAL ADDRESS
PHYSICAL ADDRESS
SEGMENT 0
0×0000
0×0001
0×0001 0×0002
0×0000 0×0003
0×0004 0×0004
0×0003 0×0005
0×0002 0×0006
0×0005 0×007B
0×01FD 0×01FF
BLOCK 0
BLOCK 1
BLOCK 2
BLOCK 3
BLOCK 4
BLOCK 5
BLOCK 6
BLOCK 123
BLOCK 511
DATA
REDUNDANT PORTION
PAGE 0
PAGE 1
PAGE 2
PAGE 15
MAIN DATA
1 0001
LOGICAL ADDRESS
CONTROL TABLE FLAG
CONTROL DATA
UNUSED
0 0005
FOR LOGICAL ADD. 0
FOR LOGICAL ADD. 1
FOR LOGICAL ADD. 2
FOR LOGICAL ADD. 3
FOR LOGICAL ADD. 4
FOR LOGICAL ADD. 5
FOR LOGICAL ADD. 493
0003
0002
0006
0005
0004
007B
01FF
2 BYTES

F I G.15


START
S1
SEARCH THE NON-VOLATILE MEMORY FOR A BLOCK FOR STORING CONTROL DATA OF THE LOGICAL / PHYSICAL ADDRESS CONTROL TABLE
S2
DOES THE LOCAL MEMORY CONTAIN ENOUGH CONTROL DATA FOR MAKING AN ACCESS TO THE NON-VOLATILE MEMORY ?
NO
YES
S17
GENERATE A REDUNDANT PORTION OF THE CONTROL DATA WITH THE CONTROL-TABLE FLAG INDICATING CONTROL DATA AND A LOGICAL ADDRESS
S18
WRITE THE CONTROL DATA AND THE REDUNDANT PORTION INTO A SPECIFIED PAGE IN THE NON-VOLATILE MEMORY
LOAD CONTROL DATA
S19
①
S5
THE WRITTEN DATA IS MAIN DATA ?
NO
YES
S3
GENERATE A REDUNDANT PORTION OF THE CONTROL DATA WITH THE CONTROL-TABLE FLAG INDICATING CONTROL DATA AND A LOGICAL ADDRESS
S4
WRITE THE CONTROL DATA AND THE REDUNDANT PORTION INTO A SPECIFIED PAGE IN THE NON-VOLATILE MEMORY
①
DETERMINE A BLOCK FOR WRITING THE MAIN DATA
S8
PAGE NUMBER ← 0
S9
S10
INPUT MAIN DATA OF 1 PAGE FROM THE DSP 102
S11
PAGE NUMBER = 0 ?
NO
YES
S12
GENERATE A REDUNDANT PORTION OF THE MAIN DATA WITH THE CONTROL-TABLE FLAG INDICATING MAIN DATA AND A LOGICAL ADDRESS
S13
GENERATE A REDUNDANT PORTION OF THE MAIN DATA WITH ARBITRARY CONTENTS
WRITE THE MAIN DATA AND THE REDUNDANT PORTION INTO A SPECIFIED BLOCK IN THE NON-VOLATILE MEMORY
S14
INCREMENT THE PAGE NUMBER
S15
NO
PAGE NUMBER >15 ?
S16
YES

[redacted] [redacted] [redacted]

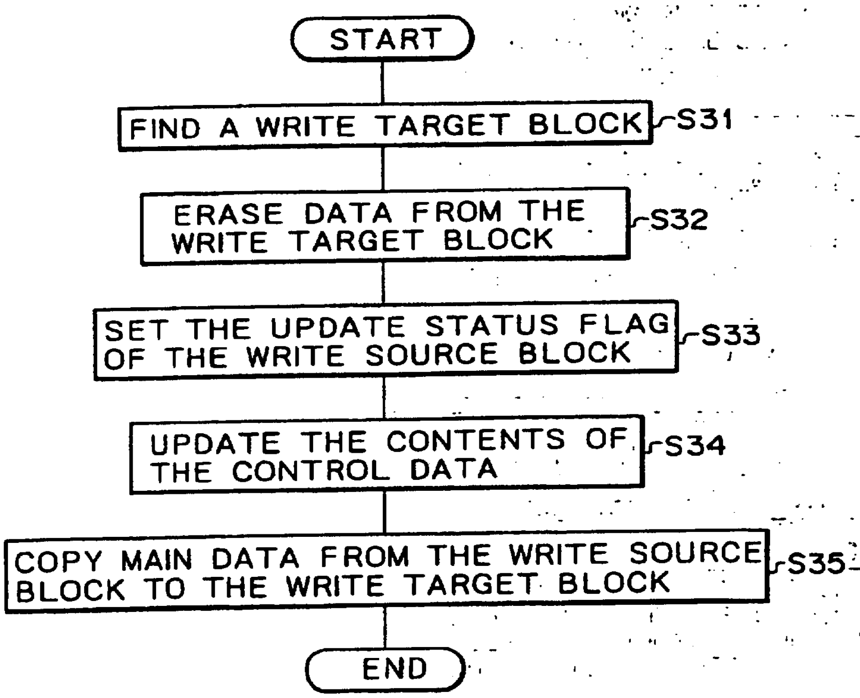
FIG.16


START
SELECT A PHYSICAL ADDRESS FROM THE TABLE ENTRIES FOR EXTRA BLOCKS
S21
FIND THE PHYSICAL ADDRESS ASSOCIATED WITH THE LOGICAL ADDRESS SUBJECTED TO THE REWRITE OPERATION
S22
CATALOG THE SELECTED PHYSICAL ADDRESS FROM THE EXTRA BLOCKS AS A PHYSICAL ADDRESS ASSOCIATED WITH THE LOGICAL ADDRESS SUBJECTED TO THE REWRITE OPERATION
S23
CATALOG THE SELECTED PHYSICAL ADDRESS INTO THE TABLE ENTRY FOR THE EXTRA BLOCK
S24
END

[redacted] [redacted]

[redacted] [redacted] [redacted]

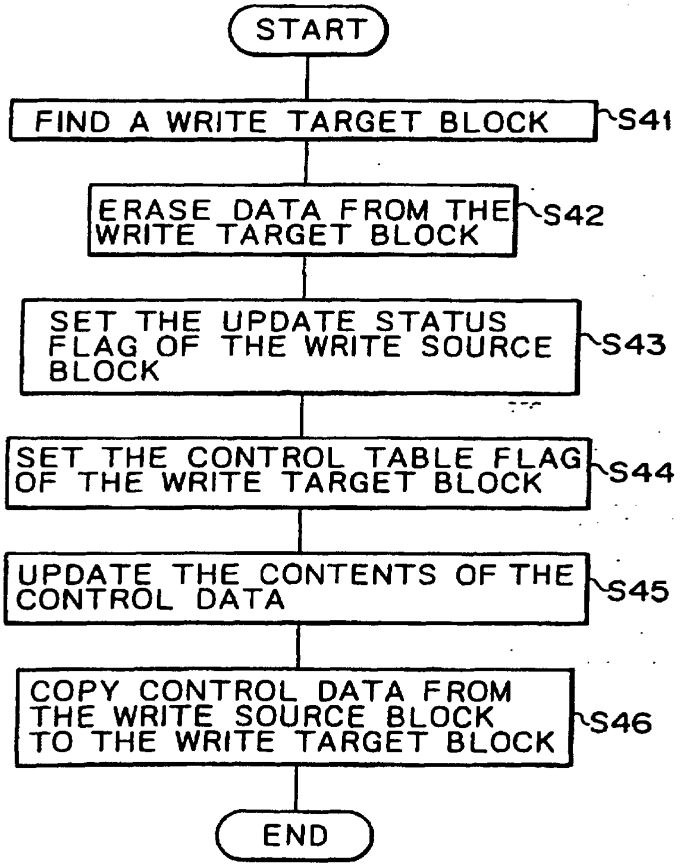
F I G.17


START
FIND A WRITE TARGET BLOCK
S31
ERASE DATA FROM THE WRITE TARGET BLOCK
S32
SET THE UPDATE STATUS FLAG OF THE WRITE SOURCE BLOCK
S33
UPDATE THE CONTENTS OF THE CONTROL DATA
S34
COPY MAIN DATA FROM THE WRITE SOURCE BLOCK TO THE WRITE TARGET BLOCK
S35
END

[redacted] [redacted]

[redacted] [redacted] [redacted]

FIG.18


START
FIND A WRITE TARGET BLOCK
S41
ERASE DATA FROM THE WRITE TARGET BLOCK
S42
SET THE UPDATE STATUS FLAG OF THE WRITE SOURCE BLOCK
S43
SET THE CONTROL TABLE FLAG OF THE WRITE TARGET BLOCK
S44
UPDATE THE CONTENTS OF THE CONTROL DATA
S45
COPY CONTROL DATA FROM THE WRITE SOURCE BLOCK TO THE WRITE TARGET BLOCK
S46
END

[redacted] [redacted]

D1/18

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 896 280 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
10.02.1999 Bulletin 1999/06

(51) Int. Cl.^{6}: **G06F 13/16**

(21) Application number: 98114979.2

(22) Date of filing: 10.08.1998

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 08.08.1997 JP 214561/97
28.04.1998 JP 119099/98

(71) Applicant:
KABUSHIKI KAISHA TOSHIBA
Kawasaki-shi Kanagawa-ken Tokyo (JP)

(72) Inventors:
- Tanaka, Yoshiyuki
Yokohama-shi, Kanagawa-ken (JP)
- Yatabe, Makoto
Yokohama-shi, Kanagawa-ken (JP)
- Sato, Takeaki
Yokohama-shi, Kanagawa-ken (JP)
- Kawamoto, Kazuya
Sagamihara-shi, Kanagawa-ken (JP)

(74) Representative: HOFFMANN - EITLE
Patent- und Rechtsanwälte
Arabellastrasse 4
81925 München (DE)

(54) Method for controlling nonvolatile semiconductor memory system

(57) In a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, the storage medium has a GROUND terminal, a power supply terminal, a control terminal and a data input/output terminal, and the connector has a function of being sequentially connected to each of the terminals. When the storage medium is inserted into the connector, the GROUND terminal and control terminal of the storage medium are connected to corresponding terminals of the connector before the power supply terminal and data input/output terminal of the storage medium are connected to corresponding terminals of the connector. Thus, it is possible to improve the stability when a memory card is inserted into or ejected from the memory system.

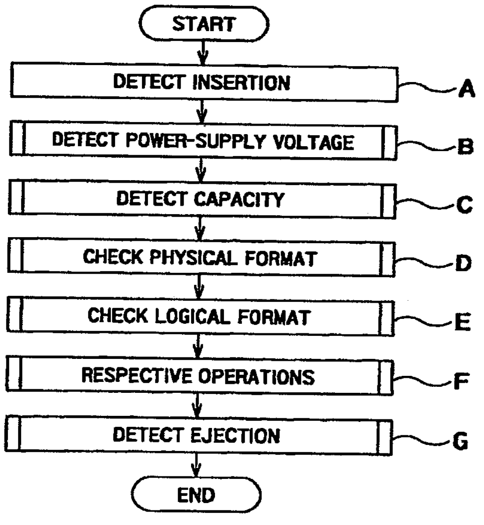


FIG.22

EP 0 896 280 A2

Printed by Xerox (UK) Business Services
2.16.7/3.6

Description

BACKGROUND OF THE INVENTION

Field of The Invention

[0001] The present invention relates generally to a method for controlling a memory system. More specifically, the invention relates to a method for controlling a non-volatile semiconductor memory system, such as a non-volatile semiconductor memory card.

Related Background Art

[0002] In recent years, a flash memory card shown in FIG. 1 has been widely noticed as a storage medium for a portable information apparatus, such as a digital steel camera and a PDA. As flash memories, so-called NAND-type flash memories have been used and sold.

[0003] As shown in FIG. 1, a flash memory card of this type has a thin plastic package 1 having a slightly recessed portion, in which a flash memory 2 having a flat electrode with 22 pins is mounted. This flash memory card is electrically connected to a host system via a dedicated connector to input and output data.

[0004] As an example of a flash memory, a 16-Mbit NAND-type flash memory is divided into 512 physical memory blocks as shown in FIG. 2. Each of these blocks is a minimum unit of erase. One block is divided into 16 pages. One page is a basic unit of writing and read-out. One page comprises 264 bytes. Among the 264 bytes, 256 bytes are used for a user data area (data division), and the remaining 8 bytes (redundant division) are used for storing an error correcting code and a management information service.

[0005] On the other hand, on the side of a personal computer, data are managed by logical blocks shown in FIG. 3. As the logical blocks (logical block address (LBA)), 500 logical blocks are set. One logical block corresponds to continuous 8 sectors. That is, logical block 0 means logical sectors 0 to 7.

[0006] In ordinary personal computers, data are managed every sector (512 bytes). Therefore, the memory card stores therein data of one sector of a logical block using 2 pages of a physical block as a pair to basically carry out data management every 512 bytes. FIG. 4 shows a concrete data storing method.

[0007] Unused normal blocks of both of the data division and redundant division are set to be "FFh". The meanings of the respective bytes will be described below. In a Data Area-1, the data of the first half 0 to 255 bytes of the data of 512 bytes are stored. In a Data Area-2, the data of the second half 256 to 511 bytes of the data of 512 bytes are stored. The data of a User Data Area are open to a user, so that the use thereof is entrusted to the user. A Data Status Area indicates whether data are normal. Although an "FFh" is usually set, a "00h" is set when abnormal data are written. A Block Status Area indicates whether a block is good or defective. Although the "FFh" is usually set, the "00h" (initial defective block) or an "F0h" (acquired defective block) is set in the case of a defective block. When the block has "0"s of two bits or more, it is determined that the block is a defective block. Furthermore, the same values for all the data are written in the same block. A Block Address Area-1 indicates a logical address information of a block. Furthermore, since 8 sectors forming one logical block correspond to one of 512 physical blocks, the same values for all the data are written in the same block. Similarly, in a Block Address Area-2, the same contents as the data of the Block Address Area-1 are written. An Ecc Area-1 is a 3-byte ECC code of even page data (256 bytes). An Ecc Area-2 is a 3-byte ECC code of odd page data (256 bytes).

[0008] As an example of a flash memory, a 64-Mbit NAND-type flash memory is divided into 1024 physical memory blocks as shown in FIG. 5. Each of these blocks is a minimum unit of erase. One block is divided into 16 pages. One page is a basic unit of writing and read-out. One page comprises 528 bytes. Among the 528 bytes, 512 bytes are used for a user data area (data division), and the remaining 16 bytes (redundant division) are used for storing an error correcting code and a management information service. As shown in FIG. 6, 1000 logical blocks are set. One logical block corresponds to continuous 16 sectors. That is, logical block 0 means logical sectors 0 to 15. FIG. 7 shows a method for storing data in the 64-Mbit NAND-type flash memory.

[0009] The control of such a memory card adopts an additional writing system for writing updated data in an erased area when data are updated and for erasing an area, in which the original data exist. Therefore, a physical block, in which data corresponding to a certain logical block exist, is not fixed and is always moving in the memory. Therefore, as shown in FIG. 8, the redundant division of the physical block stores therein a logical block address information indicating which logical block the data held in the physical block correspond to. The Block Address Area-1 and the Block Address Area-2 in FIGS. 4 and 7 correspond to the corresponding logical addresses.

[0010] Therefore, the memory system searches areas for storing the logical block address information of all the physical blocks to prepare a translation table between logical blocks and physical blocks on a system RAM, usually when a power supply is turned on. Alter the tables are prepared once, it is possible to immediately determine the physical block corresponding to the logical block when referring to the tables. Therefore, the searching operation for all the blocks is carried out once when the power supply is turned on. Naturally, if the position of the corresponding physical block changes after the data are updated, the memory system updates a logical address/physical address translation table to get ready for the next access.

[0011] However, in conventional memory systems, there is a first problem in that the RAM area required for the logical address/physical address translation table are large. The contents thereof will be described in detail below.

[0012] FIG. 9 shows a logical address/physical address translation table of a conventional 16-Mbit NAND-type flash memory. As described above, the data of one logical block, i.e., the data of continuous 8 sectors, exist in any one of 512 physical blocks in the flash memory. In order to select one block from the 512 physical blocks, 9 bits are required. If the table is formed so that offset directly indicates a physical block for convenience of a software, 2 bytes are required for one logical block, so that a RAM area of 1 KB in total is required. For example, the address of a physical block, in which the information of logical block 5 is stored, is an address offset by 5 words (10 bytes) from the top of the table.

[0013] Thus, in the conventional method, there is a problem in that the RAM area required for the logical address/physical address translation table is very large. In general, a general purpose CUP usually used well has a RAM of about 1 KB on board as an integrated RAM. Therefore, conventionally, the logical address/physical address translation table must use 1 KB, and system configuration can not be carried out only by means of the integrated RAM, so that an external RAM must be provided as a system. This is a great factor in an increase in costs.

[0014] FIG. 10 shows a logical address/physical address translation table of a conventional 64-Mbit NAND-type flash memory. In this case, the data of one logical block, i.e., the data of continuous 16 sectors, exist in any one of 1024 physical blocks in the flash memory. The selection of one physical block from 1024 physical blocks needs 10 bits, so that a RAM area of 2 KB in total is required. For that reason, a very large RAM area is required similar to the 16-bit NAND-type flash memory.

[0015] This problem is more serious as the capacity of the flash memory increases. For example, the number of blocks is 8192 in the 1-Gbit period, so that a RAM capacity of 16 KB is required.

[0016] If the memory capacity increases more, there is a second problem in that the logical address can not be stored in the redundant division of the physical block of the flash memory. The Block Address Area of the redundant division of a physical block of the 16-Mbit NAND-type flash memory shown in FIG. 7 stores therein a logical block address information indicating which logical block the data held in the physical block correspond to. FIG. 11 shows the structure of the Block Address Area of the redundant division of each physical block. In FIG. 11, four bits of D4 through D7 of a number 262 byte of an even page and a number 259 byte of an odd page are "0", "0", "0" and "1", and one bit of D0 of a number 263 byte of an even page and a number 260 byte of an odd page has a fixed value "1". Therefore, the maximum value of a block address capable of being stored is 2047 expressed by BA0 through BA10. Since 4096 physical blocks exist in a 512-Mbit NAND-type flash memory, it is not possible to store the address unless the description method for the Block Address Area is changed. Since the method for controlling the redundant division is different from those for conventional flash memories, it is not possible to control a flash memory of a wide-area capacity unless the host prepares two kinds of programs, so that there is a problem in that the capacity of the program storing area of the host system is pressed.

[0017] The writing and erase for a flash memory will be briefly described below. The writing in a flash memory is carried out in a lump every page. In the case of a 64-Mbit NAND-type EEPROM, one page has 528 bytes. In addition, erase is carried out every block. In the case of the 64-Mbit NAND-type EEPROM, one block is formed by 16 pages. Thus, in the NAND EEPROM, the unit of writing is different from the unit of erase. Therefore, it is not possible to erase only a certain page to update data.

[0018] When a flash memory card is used for a personal computer, it is generally treated as a drive under the control of the DOS. FIGS. 12(a) and 12(b) show conventional DOS format parameters, wherein FIG. 12(a) shows the parameters in the case of a cluster size of 4 KB and FIG. 12(b) shows the parameters in the case of a cluster size of 8 KB. The term "cluster" means a basic minimum unit of file management on the DOS. Even if the file size is very small, the capacity of one cluster is occupied. When the file size is large, the file is managed as a chain of a plurality of clusters, and its management information service is stored in a FAT (file allocation table). The size of the cluster, the management method for the FAT and so forth are managed in a sector called a boot sector. When one device is managed as a plurality of drives, its information is stored in a master boot sector. In order to carry out the writing in a file, a write command is issued from the OS every cluster.

[0019] FIG. 12(a) shows the case of a cluster size of 4 KB. A master boot sector is arranged in logical sector 0, and a boot sector is arranged in logical sector 16. In addition, FATs are arranged in logical sectors 17 through 22, and the copies of the FATs are arranged in logical sectors 23 through 28. Moreover, directories are arranged in logical sectors 29 through 44, and file data areas are arranged in and after logical sector 45.

[0020] FIG. 12(b) shows the case of cluster size of 8 KB. A master boot sector is arranged in logical sector 0, and a boot sector is arranged in logical sector 16. In addition, FATs are arranged in logical sectors 17 through 19, and the copies of the FATs are arranged in logical sectors 20 through 22. Moreover, directories are arranged in logical sectors 23 through 38, and file data areas are arranged in and after logical sector 39.

[0021] First, referring to FIG. 13, a conventional rewrite sequence in the case of a cluster size of 4 KB will be described. Since the cluster size is 4 KB, a write command for continuous 8 sectors is issued from the OS. At this time, the writing (data update) in logical sectors 45 through 52 (cluster A) is carried out.

(1) An erased new area is searched, and logical sectors 32 through 44 are copied on the new area NAND Block C from the original block.
(2) The new data of logical sectors 45 through 47 are written in the new area NAND Block C.
(3) The original block NAND Block A is erased.
(4) The logical address/physical address translation table is updated.
(5) An erased new area is searched, and the new data of logical sectors 48 through 52 are written in the new area NAND Block D.
(6) The data of logical sectors 53 through 63 of the original block NAND Block B are copied on the new area NAND Block D.
(7) The original block NAND block B is erased.
(8) The logical address/physical address translation table is updated.
Therefore, when 8 sectors are rewritten if viewed from the outside, the write operation in logical sectors 32 through 63, i.e., 32 sectors in total (32 pages), and the erase operation from the NAND Block A and the NAND Block B, i.e., 2 blocks in total, are carried out as an actual device.

[0022] Referring to FIG. 14, a writing sequence in cluster B will be described below. In this case, the writing (data update) in logical sectors 53 through 60 (cluster B) is carried out.

(1) An erased new area is searched, and logical sectors 48 through 52 are copied on the new area NAND Block C from the original block NAND Block B.
(2) The new data of logical sectors 53 through 60 are written in the new area NAND Block C.
(3) Logical sectors 61 through 63 are copied on the new area NAND Block C from the original block NAND Block B.
(4) The original block NAND Block B is erased.
(5) The logical address/physical address translation table is updated.
Therefore, when 8 sectors are rewritten if viewed from the outside, the write operation in logical sectors 48 through 63, i.e., 16 sectors in total (16 pages), and the erase operation from the NAND Block A, i.e., one block, are carried out as an actual device.

[0023] Referring to FIG. 15, a conventional rewriting sequence in the case of a cluster size of 8 KB will be described. Since the cluster size is 8 KB, a write command for continuous 16 sectors is issued from the OS. At this time, the writing (data update) in logical sectors 39 through 54 (cluster A) is carried out.

(1) An erased new area is searched, and logical sectors 32 through 38 are copied on the new area NAND Block C from the original block NAND Block A.
(2) The new data of logical sectors 39 through 47 are written in the new area NAND Block C.
(3) The original block NAND Block A is erased.
(4) The logical address/physical address translation table is updated.
(5) An erased new area is searched, and the new data of logical sectors 48 through 54 are written in the new area NAND Block D.
(6) The data of logical sectors 55 through 63 of the original block NAND Block B are copied on the new area NAND Block D.
(7) The original block NAND block B is erased.
(8) The logical address/physical address translation table is updated.
Therefore, when 16 sectors are rewritten if viewed from the outside, the write operation in logical sectors 32 through 63, i.e., 32 sectors in total (32 pages), and the erase operation from the NAND Block A and the NAND Block B, i.e., 2 blocks in total, are carried out as an actual device.

[0024] Comparing the cluster of 4 KB with the cluster of 8 KB when the same 8 KB data are written, in the case of the cluster size of 4 KB, the processing is divided into two write operations, so that the write operation in 48 sectors in total and the erase operation from three blocks are carried out. On the other hand, comparing with the cluster size of 8 KB in the case of the cluster size of 8 KB, the processing is concentrated on one writing, so that the write operation in 32 sectors in total and the erase operation from two blocks are carried out.

[0025] Thus, in the conventional memory system, when viewed from the outside, the number of the write and erase operations actually executed in the device is far greater than the number of updated sectors, so that there is the second problem in that the rewriting speed viewed from the outside decreases.

[0026] The operation of the conventional memory system when executing a file erase command will be described below. In an ordinary DOS file system, when the file erase command is executed, a mark indicating that a corresponding file is invalid is put on a directory, and a memory area having been occupied by the corresponding file is open on a FAT (file allocation table). Therefore, the data division of the file body remains on the flash memory without being erased. FIG. 16 shows the relationship between a management area and a data area when an erase command is executed. In FIG. 16, for example, when erase commands for File-1 and File-4 are executed, the File-1 and File-4 are open, and a del.mark is put. At this time, the File-1 and the File-4 are not erased from the data area.

[0027] For that reason, when a subsequent write command is executed, it is first required to carry out the erase operation of the flash memory when a data division of a new file is written in the open area. For that reason, the erase operation of the flash memory must be always carried out when the file writing is carried out, so that there is a third problem in that the file writing speed deteriorates.

[0028] The ECC Area-1 shown in FIG. 4 is a 3-byte ECC code of an even page data (256 bytes). The ECC Area-2 is a 3-byte ECC code of an odd page data (256 bytes).

[0029] The ECC (error correcting code) means a code for correcting an error. The system utilizes this error correcting code to determines whether a read data has an error. When an error exists, the system can correct the error. The required error correcting capability depends on the reliability of the flash memory itself, e.g., the cell structure of the memory. Flash memories have a plurality of data storing methods. When these flash memories are used for a system, such as a digital steel camera and a PDA, error correction will be considered.

[0030] For example, a first flash memory card holds binary values "0" and "1" corresponding to a threshold of a memory cell as shown in FIG. 17, and applies a 1-bit error correcting code to one page (256 bytes). A second flash memory card holds four-valued values "00", "01"," 10" and "11" (2 bits) corresponding to a threshold of a memory cell as shown in FIG. 18, and applies a 2-bit error correcting code to one page (256 bytes) since there is a possibility that 2-bit data may be broken when one memory cell is broken. Algorithm for generating a code and detecting and correcting an error for 1-bit error correction is different from that for 2-bit error correction.

[0031] Conventional systems (e.g., digital steel cameras, PDAs) have only one kind of error correcting algorithm on board. For that reason, there is a fourth problem in that it is possible to read only one of the above described first and second memory cards. This is an obstacle to the enhancement of flexibility of flash memories on the market.

SUMMARY OF THE INVENTION

[0032] It is therefore an object of the present invention to eliminate the aforementioned problems and to provide a method for controlling a memory system, which improve the stability of operation when the insertion/ejection of a memory card is carried out.

[0033] It is another object of the present invention to provide a method for reducing the capacity of a RAM required for a logical address/physical address translation table to control a flash memory by means of only an integrated RAM of a general purpose CPU, thereby dispensing with an external RAM, which has been conventionally required, to considerably reduce costs.

[0034] It is another object of the present invention to provide a method for controlling a memory system so as to prevent the delimiter of a cluster serving as a basic unit of file management on the DOS from straddling a block serving as a unit of erase, thereby providing a high-speed data writing.

[0035] It is a further object of the present invention to open a management area for a physical block while erasing its data area when an erase command is executed, thereby improving a processing speed when a subsequent write command is executed.

[0036] It is a still further object of the present invention to provide a system capable of supporting any one of two kinds of flash memories, such as binary/multi-valued flash memories, or a plurality of flash memories.

[0037] In order to accomplish the aforementioned and other objects, according to the present invention, there is provided a method for controlling a memory system, which uses the terminals of a connector sequentially connected to the terminals of a memory card (a storage medium), which is disclosed in Japanese Patent Laid-Open Nos. 8-90969 and 8-202509.

[0038] According to a first aspect of the present invention, there is provided a method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, power supply terminals, control terminals and data input/output terminals, said connector having a function of being sequentially connected to each of said terminals, wherein when said storage medium is inserted into said connector, said ground terminals and control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminals of said storage medium are connected to corresponding terminals of said connector.

[0039] According to a second aspect of the present invention, there is provided a method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, power supply terminals, control terminals and data input/output terminals, said connector having a function of being sequentially disconnected from each of said terminals, wherein when said storage medium is ejected from said connector, said power supply terminals and data input/output terminals of said storage medium are disconnected from corresponding terminals of said connector before said ground terminals and control terminals of said storage medium are disconnected from corresponding terminals of said connector.

[0040] According to a third aspect of the present invention, there is provided a method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, power supply terminals, a first control terminal, a second control terminal and data input/output terminals, said connector having a function of being sequentially connected to each of said terminals, wherein when said storage medium is inserted into said connector, said ground terminals and first control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminal of said storage medium are connected to corresponding terminals of said connector, and said second control terminal being connected to a corresponding terminal of said connector before said data input/output terminals are connected to a corresponding terminal of said connector.

[0041] According to a fourth aspect of the present invention, there is provided a method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, a power supply terminals, a first control terminal, a second control terminal and data input/output terminals, said connector having a function of being sequentially connected to each of said terminals, wherein when said storage medium is ejected from said connector, said data input/output terminals are disconnected from corresponding terminals of said connector before said second control terminal is disconnected from a corresponding terminal of said connector, and said power supply terminals and data input/output terminals of said storage medium being disconnected from corresponding terminals of said connector before said ground terminals and first control terminal of said storage medium are disconnected from corresponding terminals of said connector.

[0042] According to a fifth aspect of the present invention, there is provided a method for controlling a memory system having a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, power supply terminals, first control terminal, a second control terminal, data input/output terminals and an insertion/ejection detecting terminal, said connector having a function of being sequentially connected to each of said terminals, wherein when said storage medium is inserted into said connector, said ground terminals and first control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminals of said storage medium are connected to corresponding terminals of said connector, said second control terminal of said storage medium being connected to a corresponding terminal of said connector before said data input/output terminals of said storage medium are connected to corresponding terminals of said connector, and said insertion/ejection detecting terminal being connected to a corresponding terminal of said connector after all of said terminals of said storage medium are inserted.

[0043] According to a sixth aspect of the present invention, there is provided a method for controlling a memory system having a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto, said storage medium having ground terminals, power supply terminals, a first control terminal, a second control terminal, data input/output terminals and an insertion/ejection detecting terminal, said connector having a function of being sequentially connected to each of said terminals, wherein when said storage medium is ejected from said connector, said insertion/ejection detecting terminal is disconnected from a corresponding terminal of said connector before said data input/output terminals of said storage medium are disconnected from corresponding terminals of said connector, said data input/output terminals of said storage medium being disconnected from corresponding terminals of said connector before said power supply terminals and second control terminal of said storage medium are disconnected from corresponding terminals of said connector, and said power supply terminals of said storage medium being disconnected from corresponding terminals of said connector before said ground terminals and first control terminal of said storage medium are disconnected from corresponding terminals of said connector.

[0044] According to a seventh aspect of the present invention, there is provided a method for controlling a memory system which includes:

[0045] logical blocks managed by the system; physical blocks for storing therein data corresponding to said logical blocks, said physical blocks comprising a plurality of memory cells; redundant divisions included in a corresponding said physical blocks for storing therein addresses of corresponding said logical blocks; and physical block areas formed by at least two of said physical blocks, wherein a logical address/physical address translation table is prepared for managing corresponding relationships between said logical blocks and said physical blocks.

[0046] According to an eighth aspect of the present invention, there is provided a method for controlling a memory system, which is a semiconductor memory system for storing a file managed by a first predetermined units, in a storage area divided into second predetermined units,

[0047] According to a ninth aspect of the present invention, there is provided a method for controlling a memory system which includes:

> files managed by the system; data areas for storing therein the contents of said files; and management areas for storing therein a corresponding relationships between said files and said data areas, wherein when said file is erased, it is marked that said data areas corresponding to said management areas are empty area, to erase said corresponding data areas.

[0048] According to a tenth aspect of the present invention, there is provided a method for controlling a memory system which includes:

> files managed by the system; data areas for storing therein the contents of said files; and management areas for storing therein a corresponding relationships between said files and said data areas, wherein when said file is erased, it is marked that said data areas corresponding to said management areas are empty area, and the contents of said management areas are detected, on the basis of signals inputted to said memory system, to erase said data areas.

[0049] According to an eleventh aspect of the present invention, there is provided a method for controlling a non-volatile semiconductor memory system, which comprises the steps of:

> dividing a cell array into a plurality of physical blocks; storing each information corresponding to relationship between said physical block and logical block which is managed by said system, in each said physical block; and in order to form a table for managing corresponding relationships between said logical blocks and said physical blocks, in a random access memory in said system, sequentially preparing required corresponding relationships of corresponding relationships between said logical blocks and said physical blocks, in said random access memory in said system in accordance with accesses from a host.

[0050] According to a twelfth aspect of the present invention, there is provided a method for controlling a non-volatile semiconductor memory system which comprises the steps of:

> dividing a cell array of non-volatile semiconductor memory cells into a plurality of physical blocks; storing each information corresponding to relationship between said physical block and logical block address in each said physical block, which logical blocks are managed by said system, in order to form an address translation table for managing corresponding relationships between said physical blocks and said logical block addresses, in a random access memory in said memory system, forming a plurality of areas, each area being formed by an aggregate of at least one of said plurality of physical blocks, controlling said system so that data in predetermined address of said logical block are stored in said predetermined area, forming an address translation table corresponding to said predetermined area in which data in said predetermined address of said logical block are stored, if necessary, when said non-volatile semiconductor memory is accessed.

[0051] According to a thirteenth aspect of the present invention, there is provided a method for controlling a non-volatile semiconductor memory system which comprises the steps of:

> dividing a cell array of non-volatile memory cells into a plurality of physical blocks; storing each information corresponding to relationship between said physical block and logical block in a storage region of each of said physical blocks, which logical blocks are managed by said system; and forming a table for managing a corresponding relationship between said logical blocks and said physical blocks of a flash memory, in a random access memory of said memory system, said method further comprising the steps of:
> ensuring an area formed by one or a plurality of physical blocks, on a cell array of said flash memory; in every memory access time, searching said object area of physical blocks, forming said table for managing said corresponding relationship between said logical blocks and said physical blocks, on said random access memory of said system, allowing to select physical blocks corresponding to said logical blocks by using said table.

[0052] According to a fourteenth aspect of the present invention, there is provided a method for controlling a non-volatile semiconductor memory system, as set forth in claim 20, which further comprises the steps of:

providing a function of selectively replacing a defective physical blocks including defective cells with redundant physical blocks; and

managing said functions, for said each area, so that the number of defective physical blocks is less than or equal to a predetermined number.

[0053] According to a fifteenth further aspect of the present invention, there is provided a method for controlling a non-volatile semiconductor memory system, which allows any one of various memory units to be detachably mounted in a body of said memory system, said method comprising a step of selecting a corresponding one of various error correcting means of said body in accordance with said one of various memory units mounted in said body, to carry out error correction.

BRIEF DESCRIPTION OF THE DRAWINGS

[0054] The present invention will be understood more fully from the detailed description given herebelow and from the accompanying drawings of the preferred embodiments of the invention. However, the drawings are not intended to imply limitation of the invention to a specific embodiment, but are for explanation and understanding only.

[0055] In the drawings:

FIG. 1 is an outside drawing of a memory card;
FIG. 2 is a table showing a physical block construction of a flash memory;
FIG. 3 is a table showing data in a personal computer;
FIG. 4 is a table showing a method for storing data in a flash memory;
FIG. 5 is a table showing a physical block construction of a flash memory;
FIG. 6 is a table showing a physical block construction of a flash memory;
FIG. 7 is a table showing a method for storing data in a flash memory;
FIG. 8 is a conventional logical/physical block translation table;
FIG. 9 is a logical address/physical address translation table in a conventional memory system;
FIG. 10 is a logical address/physical address translation table in a 64-Mbit NAND-type flash memory system;
FIG. 11 is a table showing a data configuration of a block address of a flash memory;
FIG. 12 is a table showing parameters in a conventional DOS format;
FIG. 13 is a chart showing a conventional rewrite sequence;
FIG. 14 is a chart showing a conventional rewrite sequence;
FIG. 15 is a chart showing a conventional rewrite sequence;
FIG. 16 is a table showing a conventional relationship between a management area and a data area when am erase command is executed;
FIG. 17 is a chart showing an example of data stored in a flash memory;
FIG. 18 is a chart showing an example of data stored in a flash memory;
FIG. 19 is a table showing a conventional system;
FIG. 20 is an outside drawing of a memory card for use in a memory system according to the present invention;
FIG. 21 is a table showing signal names of the respective terminals of the card of FIG. 20;
FIG. 22 is a main flow chart of a memory system using a method for controlling a memory system according to the present invention;
FIGs. 23(a) and 23(b) are a schematic view showing the relationship between a power supply voltage and an appearance of a memory card;
FIGs. 24(a) and 24(b) are a schematic view showing a method for electrically detecting a power supply voltage of a memory card;
FIG. 25 is an outside drawing of a 5 V dedicated connector;
FIG. 26 is an outside drawing of a 3.3 V dedicated connector;
FIG. 27 is an outside drawing of a 5 V/ 3.3 V dedicated connector;
FIG. 28 is a schematic view showing a method for detecting the insertion/ejection of a memory card;
FIG. 29 is a schematic view showing a method for detecting the insertion/ejection of a memory card when the memory card corresponds to a PC card adapter;
FIG. 30 is a table showing an ECC data construction in a memory system according to the present invention;
FIG. 31 is a view showing conditions for calculating an ECC code in a memory system according to the present invention;
FIG. 32 is a control flow chart of a memory system according to the present invention when a power supply is turned on;
Fig. 33 is a flow chart for preparing a logical address/physical address translation table in a memory system accord-

ing to the present invention;

FIG. 34 is a logical address/physical address translation table in a memory system according to the present invention;

FIG. 35 is a flow chart of a memory system according to the present invention when read-out is carried out;

FIG. 36 is a flow chart of a memory system according to the present invention writing is carried out;

FIG. 37 is a view showing parameters of a DOS format according to the present invention;

FIG. 38 is a chart showing a rewrite sequence according to the present invention;

FIG. 39 is a chart showing a rewrite sequence according to the present invention;

FIG. 40 is a chart showing a rewrite sequence according to the present invention;

FIG. 41 is a table showing the relationship between a management area and a data area according to the present invention when an erase command is executed;

FIGs. 42(a) and 42(b) are a table of an example of a logical block/physical block translation table in a preferred embodiment of the present invention;

FIG. 43 is a flop chart of an example of a method for preparing a logic block/physical block translation table in this preferred embodiment;

FIG. 44 is a logical/physical block translation table in the preferred embodiment;

FIG. 45 is a table showing a method for expressing physical block addresses in the preferred embodiment;

FIG. 46 is a table for explaining a conventional embodiment when a defective block is replaced with a redundant block;

FIG. 47 is a table for explaining a conventional example of a replacement of a defective block with a redundant block;

FIG. 48 is a table for explaining a preferred embodiment when a defective block is replaced with a redundant block in view of a zone division control as a premise;

FIG. 49 is a schematic view of a PC card adapter according to the present invention;

FIG. 50 is a flow chart when a flash memory card is inserted into a PC card adapter;

FIG. 51 is a table showing parameters of another preferred embodiment of a DOS format of a flash memory according to the present invention; and

FIG. 52 is a table showing a method for using an ECC code area.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0056] Referring now to the accompanying drawings, the preferred embodiments of a memory system control method according to the present invention will be described below.

[0057] FIG. 20 shows a flash memory card for use in a memory system according to the present invention, and FIG. 21 shows signal names of the respective pins of a flat electrode of the memory card.

[0058] FIG. 22 shows a main control flow for use in the flash memory card. The operations of the respective steps in this flow will be described below.

[0059] First, an insertion detecting step A will be described below.

[0060] As described in the description of the prior art, in order to start the processing for a memory card, it is premised that the memory card is normally inserted into a connector. Because there is a possibility that physical or data destruction may be caused if the system accesses the memory card when the memory card is incompletely inserted. Therefore, in the control flow for the memory system according to the present invention, there is provided means for detecting whether a memory card is normally inserted. This means may be structural or electrical means.

[0061] For example, a detecting switch may be provided in the connector at a position (e.g., the innermost part of the connector) so that the detecting switch can transmit a detection signal to the system after the memory card is held at a normal contact position.

[0062] Alternatively, there may be provided a system mechanism for ensuring that the memory card is held at a normal position, not for detecting insertion. For example, a lid may be provided in an inlet for the memory card so that the lid pushes the memory card to a home position when the lid is closed.

[0063] In addition, means for electrically detecting the insertion of the memory cart may be provided.

[0064] A power supply voltage detecting step B will be described in detail below.

[0065] The memory cards include a product having a power supply voltage of 5 V and a product having a power supply voltage of 3.3 V. If a voltage of 5 V is applied to a memory card operated by a power supply voltage of 3.3 V, there is a possibility that voltage destruction and so forth may be caused. In order to avoid this, the system is designed to detect a power supply voltage for a memory card. Methods for detecting the power supply voltage may include a method for determining the power supply voltage on the basis of appearance of the memory card, and a method for electrically determining the power supply voltage.

[0066] FIGS. 23(a) and 23(b) show an example of a method for determining a power supply voltage for a memory card on the basis of appearance of the memory card. The 5 V product has a cutout on the left side of the upper end thereof as shown in FIG. 23(a), and the 3.3 V product has a cutout on the right side of the upper end thereof as shown in FIG. 23(b). Thus, it is possible to easily visually determine the power supply voltage for the memory card.

[0067] FIGS. 24(a) and 24(b) show an example of a method for electrically detecting a power supply voltage for a memory card. In this case, a number 17 pin is used for detecting the power supply voltage. For example, in the 5 V product, the number 17 pin is electrically on the floating condition as shown in FIG. 24(a), and in the 3.3 V product, the number 17 pin is short-circuited to a VCC line on the surface of the memory card. On the system side, the number 17 pin is pulled down to a VSS via a resistor. On the system side, a voltage of 3.3 V is supplied to a number 12 pin and a number 22 pin as a VCC. At this stage, a voltage of 5 V is not applied to a power supply voltage terminal. The system monitors the level of the number 17 pin. When the level of the number 17 pin is a "L" level, the system determines that the memory card is the 5 V product, and when the level of the number 17 pin is a "H" level, the system determines that the memory card is the 3.3 product. In accordance with the determined results, a normal power supply voltage is applied to the power supply voltage terminal.

[0068] The electrical detection may be omitted by the improvement of a connector. FIG. 25 shows an example of a dedicated system for the 5 V product. The 5 V dedicated system may use a connector with a cutout detecting mechanism so as to prevent the 3.3 V product from being inserted. In this case, it is also possible to prevent the memory card serving as the 5 V product from being inserted upside down. Although the 3.3 V product can be inserted upside down, it is possible to prevent the 3.3 V product from electrically contacting the system. Although FIG. 25 is simplified in order to facilitate better understanding thereof, the following consideration is given. In the case of a system wherein a supply voltage of 5 V is always supplied to a connector, a misinsertion preventing mechanism utilizing the cutout must function before the terminals of the memory card contact the terminals of the connector. For example, the misinsertion preventing mechanism may be positioned in the vicinity of the inlet of the connector so as to prevent the memory cart from electrically contacting the system when misinsertion is carried out.

[0069] FIG. 26 shows an example of a 3.3 V dedicated system. The 3.3 V dedicated system uses a connector with a cutout detecting mechanism so as to prevent the 5 V product from being inserted. In this case, it is also possible to prevent the 3.3 V product from being inserted upside down. Although the 5 V product can be inserted upside down, it is possible to prevent the 5 V product from electrically contacting the system.

[0070] FIG. 27 shows an example of a 5 V/3.3 V dual purpose system. In the case of the dual purpose system, either of the 5 V and 3.3 V products may be inserted into the system. Therefore, this system is designed to prevent a power supply of 5 V from being applied to the 3.3 V product. That is, a power supply voltage of 5 V may not be outputted when a memory card is inserted. The supply voltage of 5 V is not applied to the device until the power supply voltage is electrically detected to verify that the inserted memory card is the 5 V product.

[0071] A capacity detecting step C will be described below.

[0072] There are various flash memories having different memory capacities and interface specifications. When a memory card is inserted into the system, a maker code, device code or the like of the device is determined to inhibit new access when an unsupported code is determined. In order to read the maker cord, device code or the like, a normal power supply voltage is inputted.

[0073] A physical format checking step D will be described below.

[0074] The memory card has a physical format for storing data. When a memory card is inserted into the system, the physical format thereof is checked. If the memory card has an unsupported format, it is rejected without breaking data. If the system executes a physical format again when an unknown physical format product is inserted, there is a risk that the processing for innate and acquired defective blocks of the flash memory may be incomplete, so that care should be taken. For example, a 16-Mbit NAND-type flash memory is divided into 512 physical blocks. In the top block thereof, a physical format, an attribute information of the card and so forth are written. Since the remaining blocks are used as data areas, it may be determined whether a supportable physical format is formed by determining data of the top block.

[0075] A logical formal step E is carried out to allow the memory card to be accessed as a device on the DOS. It is also required to check whether the logical format of the memory card is supported. If the delimiter of cluster of the DOS is coincident with the delimiter of the physical block of the NAND flash memory, it is possible to carry out a higher speed operation.

[0076] At step F, the respective operations indicate the memory access operations of the system, and include readout, write and erase operations.

[0077] At an ejection detecting step G, election is detected similar to the above described insertion detection. When ejection is detected, the operation of the memory system ends.

[0078] In the above described control flow for the memory card, methods for electrically detecting the insertion/ejection of a memory card are disclosed in Japanese Patent Laid-Open Nos. 8-90969 and 8-202509. A method for controlling a memory system using connector terminals sequentially connected to the terminals of a memory card (a storage medium) will be described below.

[0079] Referring to FIG. 28, a method for detecting the insertion/ejection of a memory card in the memory system using the connector will be described below.

[0080] A number 1 pin terminal and number 10 pin terminal of the connector are connected to a VSS, and a number 11 pin terminal of the connector is pulled up to a VCC level. When the number 11 pin terminal of the memory card does not contact a connector terminal, the potential level of the connector terminal is a "H" level by means of a pull-up resistor. When the number 11 pin of the memory card contacts the connector terminal, the number 11 is connected to the VSS via the number 1 pin and number 10 pin, so that the potential level is changed to a "L" level. Therefore, it is possible to electrically detect the insertion of the memory card on the basis of the transition of the potential of the connector terminal of the number 11 pin to the "L" level, and the ejection thereof on the basis of the transition from the "L" level to the "H" level. In order to electrically detect the insertion by means of the number 11 pin, it is required to allow terminals other than the number 11 pin to contact the terminals of the connector if the number 11 pin contacts the connector terminal. The value of the pull-up resistor may be adjusted so as to meet the required specifications of the system.

[0081] In the above described memory system using the connector, a control method during the insertion/ejection of the memory card will be described in detail below. In this preferred embodiment, as a practical example, a hot-line insertion/ejection will be described. The term "hot-line insertion/ejection" means the insertion/ejection of a memory cart while a power supply voltage is applied to a connector. In order to prevent malfunction and electrical destruction of the memory, the following care should be taken.

[0082] If control signals (/CE, /WE, CLE, etc.) and an I/O pin are undefined while a power supply voltage is supplied, there is a possibility that an unintended write or erase command or the like may be received. In addition, if /CE = "L" and /RE = "L" while a power supply voltage is supplied, the device condition is a data output condition. At this time, if the system side condition is also a data output condition, there is a possibility that collision occurs on a data bus to have an undesired influence on the device. Generally, in a CMOS device, if a potential higher than the power supply voltage is applied to an I/O pin, there is a possibility that current flows into the device to cause the latch up and so forth. An example of a method for avoiding the above described matters to be attended to will be described below. For example, the contact and removal sequences for connector terminals are provided, and the contact and removal are sequentially carried out, so that it is possible to achieve the hot-line insertion/ejection. For example, the undermentioned four-stage sequence is suitable for use in a PC card adapter or the like since there is a little limit to the system side. The contact sequence for the connector and the limitations on the system side are as follows.

TABLE 1

Connector Contact Sequence		System Side Setting Condition
Contact Sequence	Contact Pin	
First Stage	VSS(1.10),CLE,ALE./WP	When Not Accessing Memory, Each Control Signal (/CE./WE./RE.CLE. ALE, etc.) Should Be Held On Inactive Stage
Second Stage	VCC./CE./RE./WE	
Third Stage	Pin At First, Second, Third Stages or More	
Fourth Stage	11 Pin	

[0083] In this case, at the first stage, the contact with a VCC terminal is made to establish the ground level of the chip, and the level of a CLE terminal is fixed to a "L" level (an inactive state) so as to avoid the problem that an erroneous command is inputted with noises and so forth at a subsequent sequence to erroneously erase the device. At the second stage, the /CE terminal is fixed to the "H" (an inactive state) so that the state of the output pin of the device is a Hi-z state, thereby preventing the collision of data on the bus regardless of the bus control state on the system side. At the third stage, the problem that current flows into the output pin from the bus line is avoided since the VCC reaches a predetermined level at the second stage. At the fourth stage, the number 11 pin finally contacts the connector terminal. As described above, the number 11 pin serves as an insertion/ejection detecting terminal.

[0084] In addition to the hot-line insertion/ ejection in the four-stage sequence, a hot-line insertion/ ejection in a two-stage sequence may be used. In this case, the contact sequence for the connector and the limitations on the system side are as follows.

TABLE 2

Connector Contact Sequence		System Side Setting Condition
Contact Sequence	Contact Pin	
First Stage	VSS(1.10),CLE,ALE./WP	When Not Accessing Memory, Each Control Signal (/CE./WE./RE.CLE. ALE, etc.) Should Be Held On Inactive Stage. I/O Pin Should Be Held In Hi-z State
Second Stage	Pin other than Pin Contacting At First Stage	

[0085] In this case, at the first stage, the contact with the VCC terminal is made to establish the ground level of a chip, and the level of the CLE terminal is fixed to the "L" level (the inactive state) so as to avoid the problem that an erroneous command is inputted with noises and so forth in a subsequent sequence to erroneously erase the device. At the second stage, the VCC, /CE, I/O pins and so forth do not completely simultaneously contact the connector terminals, and any one of unintended contacts may occur at the contact timing. However, if the I/O bus is held in the Hi-z state on the system side, it is possible to avoid the collision of data and the flow of current into the pin on the bus.

[0086] The connector for the four-stage contact sequence may be also applied to a PC card type adapter. A typical PC card has two pins, -CD1 and -CD2 pins, which are defined as card detecting pins, and the insertion/ejection of the PC card is typically detected on the system side while both pins are pulled up. As a signal generating method for the -CD1 and -CD2 pins in the PC card adapter, an example of a method using the 11 pin of the memory card is shown in FIG. 29. In this case, a standard sequence for a personal computer will be described below. When only a PC card adapter is inserted into a personal computer, the personal computer determines that no card is inserted. When a memory card is inserted into the personal computer, the VSS (1, 10 pins) first contacts a four-stage contact sequence connector. At this stage, GRD level is supplied from the personal computer. After sequential connection proceeds and all the pins are connected, the 11 pin is finally connected. At this stage, the levels of the -CD1 and -CD2 are reduced to the "L", and the personal computer side recognizes that the card is inserted, so that the initializing processing is started and a power supply is turned on the personal computer side.

[0087] The processing performed when a memory card is ejected while accessing the memory card will be described. Since this action should be basically an inhibit action, it is naturally conceived that means, such as an access lamp, for informing the user that the personal computer is accessing the memory card be provided. However, if the memory card is erroneously ejected during erase or write operation, there is a possibility that data other than those of an accessed block may be broken. For that reason, the following sequence is carried out to enhance the possibility that data other than those of a selected block may be broken. An example of a connector of a four-stage insertion/ejection sequence will be described below.

[0088] When a memory card is ejected, the memory card is removed in the reverse sequence to the above described contact sequence. First, the ejection of the memory card is detected by means of the 11 pin. Then, a write enable signal /WP is enabled to reset the erase or write operation. The time required for reset is 1.5 seconds or less at the maximum in an erase mode. In order to normally carry out the reset operation, the power supply voltage must be supplied to the memory card, and the logic in the memory card must be normally operated. When the four-stage removal sequence is used, it is ideal that the reset operation is completed before the VCC pin is removed and that the reset operation is completed before the I/O pin and so forth are removed at the third stage.

[0089] In addition, the second-stage removal sequence can be achieved by removing the pins in the reverse sequence to the above described two-stage insertion sequence.

[0090] Moreover, a three-stage contact/removal sequence for simultaneously carrying out the third and fourth stages of the four-stage contact/removal sequence is effective in data protection during the hot-line insertion/ ejection.

[0091] As described above, according to the memory system control method of the present invention, it is possible to improve the stability of operation during the insertion/ejection of the memory card when the insertion/ejection of the memory card is detected, and it is possible to protect data during the hot-line insertion/ejection.

[0092] The second preferred embodiment of the present invention will be described in detail below.

[0093] In this preferred embodiment, there is provided a flash memory card, which has a reduced capacity of a RAM area necessary for a logical address/physical address translation table.

[0094] First, the structure of a physical block and the relationship with a logical block in a memory system according to the present invention will be described below.

[0095] The structure of a physical block and the data construction in the physical block in a memory system according to the present invention are the same as those described in the description of the prior art, and shown in FIGS. 2 and 4. Referring to FIG. 4, the meanings of the respective bytes in a physical block in a memory system according to the present invention will be described.

[0096] In a Data Area-1, the data of the first half 0 to 255 bytes of the data of 512 bytes are stored. In a Data Area-2, the data of the second half 256 to 511 bytes of the data of 512 bytes are stored. The data of a User Data Area are open to a user, and the use thereof is entrusted to the user. A Data Status Area indicates whether data are normal. Although an "FFh" is usually set, a "00h" is set when abnormal data are written. A Block Status Area indicates whether a block is good or defective. Although the "FFh" is usually set, the "00h" (an initial defective block) or an "F0h" (an acquired defective block) is set in the case of a defective block. When a block has "0"s of two bits or more, it is determined that the block is a defective block. Furthermore, the same value for all the data is written in the same block. A Block Address Area-1 indicates a logical address information of a block. Furthermore, since 8 sectors forming one logical block correspond to one of 512 physical blocks, the same value for all the data is written in the same block. Similarly, in a Block Address-2, the same contents as the data of the Block Address Area-1 are written. An Ecc Area-1 is a 3-byte ECC code of even page data (256 bytes). An Ecc Area-2 is a 3-byte ECC code of odd page data (256 bytes).

[0097] The ECC will be described below. In this preferred embodiment, an ECC code is generated every data of 256 bytes (2048 bits). ECC data of 22 bits are used for 256 bytes so as to have a correction function of 1 bit. As shown in FIG. 30, 256 bytes are arranged on data.

[0098] That is, bit 0 of the input at the first byte is the first bit (address: 00000000 000) of 2048 bits, and bit 7 of the input at the 255th byte is the 2048th bit (address: 11111111 111) of 2048 bits.

[0099] The ECC code (line parity (LP) and column parity (CP)) is calculated as an odd parity of 1024 bits meeting the conditions shown in FIG. 31.

[0100] Column parities CP0 through CP5 are updated each time data of 1 byte (8 bits) are inputted. When the ECC code is generated by a software, there may be provided a method, which previously has the calculated results of column parities for an input of 1 byte (256 ways) on an ROM in the system. By this method, it is not required to carry out calculation every bit, so that it is possible to considerably reduce the time required for calculation. Alternatively, there may be provided a method for deriving the calculated results of column parities for an input of 1 byte (256 ways) in a lump when a power supply is turned on, to hold the calculated results on a RAM. In comparison with the former method, it is required to provide the RAM area although it is not required to provide a ROM.

[0101] FIG. 32 is a control flow chart for the second preferred embodiment of a flash memory card according to the present invention when a power supply is turned on. This control flow will be described below.

(Step S1) A power supply voltage is detected in response to the insertion of the memory card into the connector.
(Step S2) An ID code of the memory card is read out, and the storage capacity is read.
(Step S3) If an ID code, which is not supported by the system, is read out, it is rejected.
(Step S4) The physical format is checked. The information on the top block of the physical address is read.
(Step S5) If a format, which is not supported by the system, is carried out, it is rejected.
(Step S6) A logical address/physical address translation table is prepared, and an erased area for write in the next write operation is selected.

[0102] FIG. 33 shows a flow chart for preparing a logical address/physical address translation table, which is shown at the aforementioned step S6 and which is prepared when a power supply is turned on.

(Step S1) A RAM area for storing therein a logical address/physical address translation table is reset.
(Step S2) A table area for storing therein an erased block used for the next data writing is reset.
(Step S3) Search is started from physical block 1.
(Step S4) A redundant division of the block is read out.
(Step S5) On the basis of data in a predetermined area, it is determined whether the block is a normal block. When the block is a defective block, the subsequent processing is not required, and the search for the next block is carried out.
(Step S6) It is determined whether the area is an erased area.
(Step S7) If it is an erased block, it is stored in the table as a proposed block used when the next write is carried out.
(Step S8) If it is not the erased area, a logical address information area is extracted. At this time, a parity check is carried out to check validity.
(Step S9) On the basis of the above contents, a logical address/physical address translation table is prepared.
(Step S10) Physical block number is counted up.
(Step S11) After 512 blocks are searched, the routine ends.

[0103] FIG. 34 shows a logical address/physical address translation table prepared according to the aforementioned flow. The term "physical block area" shown in FIG. 34 means an aggregate of continuous two physical blocks. For example, physical block area 0 indicates physical block 0 and physical block 1. In this table, one physical block area is allocated to one logical block. For example, when physical block area 5 is allocated to logical block 0, actual data of logical block 0 are stored in physical block 10 or 11. Therefore, when access is actually carried out, it is required to search for a data area indicating the relationship with the logical addresses of the redundant divisions of the physical blocks 10 and 11 to determine which actually stores data of the logical block 0. However, since it is enough to read only a very limited area, there is little influence on the performance of memory access.

[0104] At this time, 256 (512/2) physical block areas exist as a whole and can be described by 8-bit data. If the table is formed so that offset directly indicates a physical block for convenience of the software, it is required to provide 1 byte for one block, i.e., a RAM area of 0.5 KB in total. For example, the address of a physical block area storing therein the information on physical block 5 is stored in the fifth byte offset from the top of the table.

[0105] The 0.5 KB RAM capacity is a half of 1 KB RAM capacity conventionally required. Usually, a general purpose CPU has a RAM area of about 1 KB. According to this preferred embodiment, it is possible to reduce the RAM area by 0.5 KB to obtain a great advantage.

[0106] That is, it is possible to form the system by using an empty area of 0.5 KB without the need of an external RAM increasing costs, so that it is possible to reduce costs.

[0107] The present invention should not be limited to the above described preferred embodiment, but four physical blocks may be defined as a physical block area, or a larger number of blocks may be supposed.

[0108] FIG. 35 is an operation flow chart of a memory card in this preferred embodiment when read-out is carried out. This flow will be described below.

(Step S1) The top sector address for carrying out read-out and the number of transfer sectors are received from the host.
(Step S2) It is verified whether the read-out range is a valid range.
(Step S3) The sector is converted to a logical block. Since one block comprises 8 sectors in the case of a 16-Mbit product, one block is divided by 8.
(Step S4) A physical block area, in which the corresponding logical block exists, is obtained by referring to a logical address/physical address translation table.
(Step S5) The logical address information areas of two physical blocks in the block area are examined, and it is examined which stores therein data of the designated logical block.
(Step S6) The data of one sector is read out of the identified physical block. For example, when the sector number

is 0, the data of the top 2 pages of the physical block are read, and when the sector number is 7, the data of the end 2 pages of the physical block are read. In one physical block, the data of 8 sectors are arranged in sequence.

(Step S7) Error check for the read-out data is carried out, and the presence of an error is checked.
(Step S8) When an error is detected, it is determined whether the data can be corrected.
(Step S9) When an error is detected and when the data can be corrected, the data is corrected.
(Step S10) After sectors of a number required by the host are read out, the routine ends.
(Step S11) It is determined whether the next read sector exceeds the boundary between physical blocks. For example, when the sector is transferred from sector 7 to sector 8, data exist in different physical blocks, so that the system refers to the logical address/physical address translation table again.
(Step S12) When the read-out is continued in the same block, pages to be read out are counted up.
(Step S13) When the block to be read is moved to another block, the logical block is counted up, and the count of pages is also reset.

[0109] The write operation in this preferred embodiment will be described below.

[0110] Basically, the write operation is roughly divided into the following three processes. The case where, e.g., logical sector 3, is rewritten will be described. Although only sector 3 is updated, the processing for one block is required since data of 8 sectors, sectors 0 to 7, exist in the same block.

[0111] First, since the data of logical sectors 0, 1 and 2 are not updated, the data of logical sectors 0, 1 and 2 are copied on a physical block to be newly written from the originally stored physical block.

[0112] Secondly, since logical sector 3 is updated, it is not required to copy the original data, and the data supplied from the host are written in a block to be newly written.

[0113] Thirdly, since the data of logical sectors 4 through 7 are not updated, the data of logical sectors 4 through 7 are copied on a physical block to be newly written from the originally stored physical block.

[0114] As described above, the operations of copy/updated data writing/copy for one block are basically carried out. When write in sectors 0 through 7 is carried out, all the data of one block are updated, so that the copying operation is clearly unnecessary. The branch of the undermentioned flow chart proceeds mainly while determining whether a sector to be written is updated data or copying operation.

[0115] FIG. 36 is an operation flow chart of a flash memory card in this preferred embodiment when write is carried out. This flow will be described below.

(Step S1) The top sector address for carrying out read-out and the number of transfer sectors are received from the host.
(Step S2) It is converted to a logical block number to refer to the logical address/physical address translation table. An actual physical block is selected from two blocks of a logical block area similar to the read-out operation. Data to be copied are sucked from the selected block.
(Step S3) The processing is started from the top of the physical block.
(Step S4) It is determined whether the copying operation in the first half of the block or the update of data is carried out.
(Step S5) When the copy is carried out, data are read out of the original block, and write is carried out in a new block.
(Step S6) The processing for the next sector is carried out.
(Step S7) When it is determined at step S4 that the area is an updated area, the write is carried out on the basis of the updated data received from the host.
(Step S8) The processing for the next sector is carried out.
(Step S9) It is checked whether write has been carried out for sectors of a number required by the host.
(Step S10) It is determined at step S10 that write has been carried out for the required number of sectors, it is determined whether it is the boundary between blocks. If unwritten areas remain, the copying operation on the second half of the block is carried out. If it is the boundary between blocks, it is not required to carry out the copying operation.
(Step S11) Data are read from the original block to be written in a new block.
(Step S12) The processing for the next sector is carried out.
(Step S13) When it is determined at step S4 that the write in sectors of a number required by the host is not completed, it is required to more carry out write. However, when it is the boundary between blocks, the processing for the next physical block is carried out.
(Step S14) Before the processing for the next block is carried our or before the processing is completed, the logical address/physical address translation table is updated on the basis of the written results, and the physical block, in which the original data exist, is erased. In addition, a proposed area is registered as a new write area for the next processing.

(Step S15) The processing for the next block is carried out.

[0116] As described above, according to the memory system control method in this preferred embodiment, it is possible to considerably reduce the RAM area. In the conventional memory card system, the RAM capacity required for the logical address/physical address translation table is large, and the system can not be constructed only by the integrated RAM of the general purpose CPU, so that the external RAM must be provided. On the other hand, the memory card system of the present invention uses a flash memory control method, which can reduce the RAM capacity required for the logical address/physical address translation table and which can control only the integrated RAM of the general purpose CPU. Thus, the external RAM, which has been conventionally required, is not required, so that it is possible to considerably reduce costs.

[0117] The third preferred embodiment of the present invention will be described in detail below.

[0118] In this preferred embodiment, there is provided a flash memory card, which can carry out the write operation of data at a higher speed when it is used in the DOS format.

[0119] FIGS. 37(a) and 37(b) show DOS format parameters in this preferred embodiment, wherein FIG. 37(a) shows the case of a cluster size of 4 KB, and FIG. 37(b) shows the case of a cluster size of 8 KB. When the cluster size is 4 KB, a master boot sector is arranged in logical master 0, and a boot sector is arranged in logical sector 19. In addition, FATs are arranged in logical sectors 20 through 25, and the copies of the FATs are arranged in logical sectors 26 through 31. Moreover, directories are arranged in logical sectors 32 through 47, and file data areas are arranged in and after logical sector 48. When the cluster size is 8 KB, a master boot sector is arranged in logical sector 0, and a boot sector is arranged in logical sector 25. In addition, FATs are arranged in logical sectors 26 through 28, and the copies of the FATs are arranged in logical sectors 29 through 31. Moreover, directories are arranged in logical sectors 32 through 47, and file data areas are arranged in and after logical sector 49. Thus, in either case of cluster sizes of 4 KB and 8 KB, parameters are set so that the delimiter of cluster does not straddle the delimiter of physical block. This is achieved by adjusting a place, in which a boot sector is arranged, of the DOS format parameters.

[0120] First, referring to FIG. 38, an example of a write sequence in the case of a cluster size of 4 KB will be described. Since the cluster size is 4 KB, a write command for continuous 8 sectors is issued from the OS. At this time, write (data update) in logical sectors 48 through 55 (cluster A) is carried out.

(1) An erased new area is searched, and new data of logical sectors 48 through 55 are written in a new area NAND Block C.
(2) The original data of logical sector 56 through 63 are copied on the new area NAND Block C.
(3) The original block NAND Block B is erased.
(4) The logical address/physical address translation table is updated.

[0121] Therefore, if the rewrite is carried out in 8 sectors when viewed from the outside, the write operation in logical sectors 48 through 63, i.e., 16 sectors in total (16 pages), and the erase operation from the NAND Block B, i.e., one block, have been carried out as an actual device.

[0122] Referring to FIG. 39, the write sequence in cluster B will be described below. In this case, the write (data update) in logical sectors 56 through 63 (cluster B)is carried out.

(1) An erased new area is searched, and the original data of logical sectors 48 through 55 are copied on the new area NAND Block C.
(2) Hew data of logical sectors 56 through 63 are written in the new area NAND Block C.
(3) The original block NAND Block B is erased.
(4) The logical address/physical address translation table is updated.

[0123] Therefore, if rewrite is carried out in 8 sectors when viewed from the outside, the write operation in logical sectors 48 through 63, i.e., 16 sectors in total (16 pages), and the erase operation from the NAND Block B, i.e., one block, have been carried out as an actual device.

[0124] Referring to FIG. 40, an example of a write sequence in the case of a cluster size of 8 KB will be described below. Since the cluster size is 8 KB, a write command for continuous 16 sectors is issued from the OS. At this time, write (data update) in logical sectors 48 through 55 (cluster A) is carried out.

(1) An erased new area is searched, and new data of logical sectors 48 through 63 are written in a new area NAND Block C.
(2) The original block NAND Block B is erased.
(3) The logical address/physical address translation table is updated.

[0125] Therefore, if rewrite is carried out in 16 sectors when viewed from the outside, the write operation in logical sectors 48 through 63, i.e., 16 sectors in total (16 pages), and the erase operation from the NAND Block B, i.e., one block, have been carried out as an actual device.

[0126] Comparing the cluster of 4 KB with the cluster of 8 KB when the same 8 KB data are written, in the case of the cluster size of 4 KB, the processing is divided into two write operations, so that the write operation in 32 sectors in total and the erase operation from two blocks are carried out. On the other hand, in the case of the cluster size of 8 KB, the processing is concentrated on one writing, so that the write operation in 16 sectors in total and the erase operation from one block are carried out.

[0127] Comparing with the rewriting speed of the conventional memory system shown in FIGS. 12 through 15, in the case of the cluster of 4 KB, the write operation in 48 sectors and the erase operation from three blocks are carried out in order to update 8 KB data in the conventional memory system, whereas the write operation in 32 sectors and the erase operation from two blocks are carried out in the memory system of the present invention, so that the rewriting time can be reduced to two thirds according to the present invention. In the case of the cluster of 8 KB, the write operation in 32 sectors and the erase operation from two blocks are carried out in order to update 8 KB data in the conventional memory system, whereas the write operation in 16 sectors and the erase operation from one block are carried out in the memory system of the present invention, so that the rewriting time can be reduced to half according to the present invention.

[0128] Thus, the delimiter of cluster serving as a unit of file management on the DOS does not straddle the boundary between physical blocks of the flash memory, so that the rewriting speed can be increased.

[0129] In addition, in the case of a cluster of 4 KB, the write operation in 32 sectors and the erase operation from two blocks are carried out in order to update 8 KB data, whereas in the case of a cluster of 8 KB, the write operation in 16 sectors and the erase operation from one block are carried out in order to update 8 KB data, so that the rewriting time is reduced to half. That is, if the size of a cluster is the same size as that of a physical block, a high-speed write can be achieved. Also, if the size of a cluster is an integer times as large as the size of a physical block, the same advantage can be obtained.

[0130] The fourth preferred embodiment of the present invention will be described below.

[0131] In this preferred embodiment, there is provided a flash memory card, which improves the processing speed when a write command is executed after erase.

[0132] In the flash memory card system in this preferred embodiment, unlike the file erase in the ordinary DOS, a mark indicating that a corresponding file is invalid is put on a directory, and a memory area having been occupied by the corresponding file is not only open on a FAT (file allocation table), but the data division of the file body is erased on the flash memory. That is, when the erase of the file is commanded, the erase operation from the open cluster area is carried out.

[0133] FIG. 41 shows the relationship between a management area and a data area when an erase command is executed. In FIG. 41, for example, when erase commands for File-1 and File 4 are executed, the File-1 and File-4 in the management area are open, and a del.mark is put. In addition, the areas having stored File-1 and File-4 in the data area are erased.

[0134] Therefore, since the cluster selected when the next new file write command occurs has been erased, write can be immediately carried out, so that the file writing speed can be improved. Since the erasing time is generally longer than the writing time in the flash memory, it is possible remarkably improve the file writing speed according to the present invention.

[0135] As can be seen from the above described third preferred embodiment, the advantage of this preferred embodiment is most remarkable when the cluster size is the same as the block size of the flash memory. When the cluster size is smaller than the block size of the flash memory, a part of a block is erased. In this case, the processing is not only complicated, but there are also some cases where it is not possible to erase only a part of a block on the specification. If the cluster size is the same as the block size, the cluster can be open by simply erasing the block. Also if the cluster size is an integer times as large as the physical block size, the same advantage can be obtained.

[0136] The present invention can be embodied in various ways without departing from the principle of the invention. For example, in this preferred embodiment, while the erase operation from the corresponding cluster in the data area has been executed when the file is erased, the execution timing for erase operation should be limited thereto. For example, the erase operation from all the clusters may be executed when the format operation is executed. The memory card may be shipped after erasing the cluster in the data area. When a memory card is shipped, a shipping test for the memory card is generally carried out. If directories and FATs are rewritten when this test is completed, it is not only possible to obtain no file state, but it is also possible to achieve a high-speed file writing without the need of end user's correction when the memory card is delivered to the end user if the erase operation from the data area is executed according to this preferred embodiment.

[0137] In a case where the memory card is used for a digital steel camera or the like, when operations including the erase of a file in the camera and the erase of an image file, such as a reformat, are carried out, if a cluster area having stored the file body is erased simultaneously when the FAT and so forth are rewritten, it is possible to achieve a high-speed writing in the subsequent image writing sequence, so that it is possible to continuously take pictures and to capture moving pictures. In this case, if the cluster size is an integer times as large as the block size and if the delimiter of cluster is coincident with the delimiter of block size, the file body part can be easily erased. In addition, unnecessary file parts may be automatically erased when a power supply is turned on in a digital camera or the like. Thus, it is also possible to increase the writing speed without giving the user trouble, with respect to a memory card wherein the file erase has been carried out by simply updating the FATs and so forth in a personal computer. This timing should not be limited to the time when the power supply is turned on, but it may be any time. The writing speed can also be increased when the memory card of the present invention is used for a computer system or the like.

[0138] In addition, according to this preferred embodiment, since it is possible to reduce the numbers of erase and write operations occurring when data are rewritten, it is also possible to increase the life of a flash memory when the flash memory having a limited number of rewrite operations is used.

[0139] As described above, according to the memory system control method in this preferred embodiment, the delimiter of cluster serving as a unit of file management on the DOS does not straddle the boundary between physical blocks, so that it is possible to reduce the numbers of erase and write operations occurring when data are rewritten, thereby increasing the rewriting speed. Moreover, when a flash memory having a limited number of rewrite operations is used, it is possible to increase the life of the memory.

[0140] The fifth preferred embodiment of the present invention will be described in detail below.

[0141] In this preferred embodiment, there is provided a flash memory card, which reduces the capacity of a RAM area required for a logical address/physical address translation table.

[0142] In this preferred embodiment, when a logical address/physical address translation table is prepared, access from a host is divided into two kinds. For example, it is assumed that case 1 is a state accessing the first half 250 logical blocks of 500 logical blocks and that case 2 is a state accessing the second half 250 logical blocks of 500 logical blocks. In the case 1, a table of the first half 250 blocks is held in the logical address/physical address translation table. In the case 2, a table of the second half 250 blocks is held in the logical address/physical address translation table. FIG. 42(a) shows the state of the table in the case 1, and FIG. 42(b) shows the state of the table in the case 2.

[0143] Assuming that the table of the first half 250 logical blocks exists on the table in a certain moment, if the range accessed from the host is a range of the first half 0 through 249 logical blocks, it is possible to search the correspondence between the logical blocks and the physical blocks by using the existing table.

[0144] Similarly, assuming that the table of the second half 250 logical blocks exists on the table in a certain moment, if the range accessed from the host is a range of the second half 250 through 499 logical blocks, it is possible to search the correspondence between the logical blocks and the physical blocks by using the existing table.

[0145] Then, assuming that the table of the first half 250 logical blocks exists on the table in a certain moment, if the range accessed from the host is a range of the second half 250 through 499 logical blocks, it is not possible to search the correspondence between the logical blocks and the physical blocks by using the existing table. Therefore, in this case, the logical address/physical address translation table corresponding to the second half 250 logical blocks is remade. This needs to refer to all the areas of the flash memory again.

[0146] Similarly, assuming that the table of the second half 250 logical blocks exists on the table in a certain moment, if the range accessed from the host is a range of the first half 0 through 249 logical blocks, it is not possible to search the correspondence between the logical blocks and the physical blocks by using the existing table. Therefore, in this case, the logical address/physical address translation table corresponding to the first half 250 logical blocks is remade. This needs to refer to all the areas of the flash memory again.

[0147] Thus, if the logical address/physical address translation table corresponding to the area accessed from the host does not exist on the RAM, the required logical address/physical address translation table is remade by referring to all the areas of the flash memory again.

[0148] FIG. 43 shows a flow chart for preparing the logical address/physical address translation table in this case.

> (Step S0) The presence of a required logical address/physical address translation table is checked at the beginning of access, and if it is required, the routine goes to a table preparing routine.
> (Step S1) A RAM area for storing therein a logical address/physical address translation table is reset.
> (Step S2) Search is started from the top of a physical block.
> (Step S3) The redundant division of the block is read out.
> (Step S4) On the basis of data in a predetermined area, it is determined whether the block is a normal block.

[0149] When the block is a defective block, the subsequent processes are not required, and the search for the next block is carried out.

> (Step S5) It is determined whether the block is an erased area.

(Step S6) If the block is an erased area, it is stored on the table as a proposed block used for the next write.

(Step S7) If the block is not an erased area, a logical address information area is extracted.

(Step S8) A logical address/physical address translation table is prepared on the basis of the above described contents.

(Step S9) The physical block number is counted up. After the search for all the blocks is completed, the routine ends.

[0150] In accordance with the above described operations, the logical address/physical address translation table is prepared if necessary.

[0151] The present invention should not be limited to the above described preferred embodiment, but the invention can be embodied in various ways without departing from the principle of the invention.

[0152] For example, in the above described preferred embodiment, while the flash memory has been divided into two parts, the first and second halves, the present invention should not be limited thereto, but the flash memory may be divided into an optional number of parts.

[0153] In addition, in the above described preferred embodiment, while the flash memory has been divided into the first and second halves having the same size, the present invention should not be limited thereto. Thus, the number and size of divided parts may be optionally determined.

[0154] It is not always required to hold only one table when the flash memory is divided into three or more areas. For example, it is assumed that the flash memory card serves as a device of the DOS to store therein an image file or the like. In the top of an ordinary device, there are file management areas, i.e., a master boot sector, a partition boot sector, a file allocation table (FAT) and a directory area. These file management areas are frequently accessed each time the update or access of the file is carried out. In this case, the area corresponding to the file management area is one area, and each of other file data storing areas are divided into two parts. The logical address/physical address translation table corresponding to the file management area may be always held. In this case, when a file is written in the second half of the logical address, it is not required to alternately frequently remake the file management area and the second half of the logical address, so that it is possible to prevent the deterioration of performance. While the file management area has been described, the same advantage can be obtained if the table is always held for areas other than the file management area.

[0155] As described above, according to this preferred embodiment, all of the correspondence relationships between logical blocks and physical blocks must not be always held on the RAM, and the correspondence relationship of only a required area is prepared on the RAM in the system one by one in accordance with the access from the host. Therefore, in comparison with the case where the correspondence relationships in all the areas are always held on the RAM, it is possible to reduce the minimum required RAM area, and it is possible to control the memory only by the integrated RAM of the general purpose CPU although this has not been able to be achieved. Thus, it is possible to greatly reduce costs in comparison with a conventional system having an external RAM.

[0156] The sixth preferred embodiment of the present invention will be described in detail below.

[0157] In this preferred embodiment, there is provided a flash memory card, which reduce the capacity of a RAM area required for a logical address/physical address translation table.

[0158] In this preferred embodiment, a physical block address of the flash memory is divided into a plurality of logical areas (which will be hereinafter referred to as "zones"). In this case, even if the number of physical blocks allocated to each zone, i.e., the capacity of each zone, may be uniform or ununiform. In addition, the number of zones may be one or plural. Moreover, the number of zones may be even or odd.

[0159] FIG. 44 shows the structure of a physical block and the relationship between logical addresses and physical addresses when a 16-Mbit NAND-type flash memory is divided into two zones having the same capacity.

[0160] When the host controls the flash memory, the number of required logical blocks is defined to be 500, and the values of logical block addresses are 0 through 499. The redundant division of a physical block stores therein a logical block address information indicating which logical block the data held in the physical block correspond to. In this preferred embodiment, logical block addresses are given from 0 in series for each zone. Therefore, logical block addresses 0 through 249 are allocated to zone 1, and each of logical blocks corresponds to any one of 256 physical blocks of physical block addresses 0 through 255. In addition, any one of logical block addresses 0 through 249 is stored in the redundant division of a physical block. Moreover, logical block addresses 250 through 499 are allocated to zone 2, and each of logical blocks corresponds to any one of 256 physical blocks of physical block addresses 256 through 511. In addition, any one of logical block addresses 0 through 249 is stored in the redundant division of a physical block.

[0161] An additional write system during data update is executed in a zone including a logical block address to be updated, and a physical block having data corresponding to a certain logical block is not fixed and is always moving in each zone.

[0162] In this preferred embodiment, usually when a power supply is turned on, the logical block address information stored in the redundant divisions of all the physical blocks in the zone is searched, and a translation table between logical blocks and physical blocks is prepared on a system RAM. The zone prepared on the RAM may be any one of zones. Usually, when the flash memory is used under the control of the DOS, the FATs and directory areas serving as management information services are arranged in the first zone, so that it is efficient to prepare the table of the top zone. In addition, the table prepared on the RAM should not be limited to one zone, but it may be prepared for a plurality of zones so far as the RAM capacity permits.

[0163] A process for preparing a translation table for an access demand from a host will be described below. An area including physical block addresses 0 through 255 is set to be called zone 1. In addition, an area including physical block addresses 256 through 511 is set to be called zone 2. The host remembers which zone the translation table currently prepared on the RAM belongs to.

[0164] When a translation table of zone 1 is prepared on the RAM and when an access demand for logical block address 128 is made, the procedure will be described blow.

(1) If n is derived so that logical block address $128 - 250 \times (n - 1) < 250$, $n = 1$, so that it can be seen that the translation table of zone 1 is required.
(2) Since the translation table on the RAM is zone 1, no table is prepared.
(3) Since the values of addresses on the translation table are 0 through 249, an address to be referred on the translation table is derived from logical block address 128 demanded by the host. From $128 - 250 \times (1 - 1) = 128$, a physical block address corresponding to address 128 on the translation table may be accessed. (4) When rewrite occurs, the translation table is updated to get ready for the next access.

[0165] Then, when a translation table of zone 1 has been prepared on the RAM and when an access demand for logical block address 324 is made, the procedure will be described blow.

(1) If n is derived so that logical block address $324 - 250 \times (n - 1) < 250$, $n = 2$, so that it can be seen that a translation table of zone 2 is required.
(2) Since the translation table on the RAM is zone 1, the logical block address information of the redundant divisions of physical block addresses 256 through 511 included in zone 2 is searched, and the translation table of zone 2 is prepared on the RAM.
(3) Since the values of addresses on the translation table are 0 through 249, an address to be referred on the translation table is derived from logical block address 324 demanded by the host. From $324 - 250 \times (2 - 1) = 74$, a physical block address corresponding to address 74 on the translation table may be accessed.
(4) When rewrite occurs, the translation table is updated to get ready for the next access.

[0166] As described above, if the translation table is prepared in accordance with the accessed logical block, it is possible to easily reduce the RAM area in comparison with the conventional RAM area. Also, when the table of zone 2 has been prepared on the system RAM, it is possible to easily access a target address by the similar process.

[0167] When the logical address/physical address translation table is prepared, the translation table between logical blocks and physical blocks is prepared on the RAM. At this time, since a zone number to be prepared is known, the values of physical block addresses on the translation table may be 0 through 255. When the physical block is actually accessed, it is possible to easily obtain a new physical block address to be inputted to the flash memory by adding 256 × zone number to the physical address on the translation table as OFFSET.

[0168] When a 16-Mbit NAND-type flash memory is used, the conventional control method requires 9 bits to express a physical block address, and uses 2 bytes for convenience of a software. In this preferred embodiment, as shown in FIG. 45, a physical block address may be expressed by 8 bits, i.e., 1 byte. Therefore, the RAM capacity, which has conventionally required 1 Kbyte, can be reduced to half. If a logical block address increases, the capacity of the logical address/physical address translation table increases. Therefore, the advantages of this preferred embodiment increases as the capacity of the flash memory increases.

[0169] In addition, according to this preferred embodiment, a large-capacity flash memory can be controlled by a block address, which can be stored in the Block Address Area of the redundant division of the physical block shown in FIG. 11. That is, if one zone is divided so as to be formed by a physical block of a block address value, which can be stored in the Block Address Area of the redundant division, it is possible to correspond to a large-capacity logical address.

[0170] FIG. 46 shows the structure of a physical block when a 16-Mbit NAND-type flash memory is divided into four equal capacity zones.

[0171] When the host controls the flash memory, the number of required logical blocks is defined to be 500, so that the values of logical block addresses are 0 through 499. The redundant division of the physical block stores therein a logical block address information indicating which logical block the data held in the physical block correspond to. In this preferred embodiment, the logical block addresses are given in series from 0 for each zone. Physical block addresses corresponding to these logical block addresses are as follows. The logical block addresses 0 through 124 are allocated to zone 1, and each of the logical blocks corresponds to any one of 128 physical blocks having physical block addresses 0 through 127. The logical block addresses 125 through 249 are allocated to zone 2, and each of the logical blocks corresponds to any one of 128 physical blocks having physical block addresses 128 through 255. The logical block addresses 126 through 374 are allocated to zone 3, and each of the logical blocks corresponds to any one of 128 physical blocks having physical block addresses 256 through 383. The logical block addresses 384 through 499 are allocated to zone 4, and each of the logical blocks corresponds to any one of 128 physical blocks having physical block addresses 384 through 511.

[0172] The additional write system during data update is executed only in a zone including a logical block to be written, and a physical block storing therein data corresponding to a certain logical block address is not fixed and is always moving in the zone. The redundant division of each physical block address stores therein a logical block address information indicating which logical block address the data held in the physical block belong to.

[0173] As described above, 125 logical blocks are allocated to each zone, and 128 physical blocks are allocated to each zone. If a physical block is divided into four zones as described above, it is possible to obtain the same advantages as those in the case where a physical block is divided into two zones.

[0174] An example of redundancy operation in this preferred embodiment in the case of the physical block divided into four zones will be described below.

[0175] Before describing the redundancy operation in this preferred embodiment, the redundancy operation of a 16-Mbit NAND-type flash memory as an example of a conventional flash memory, which does not carry out the division of a physical block into zones, will be described. As shown in FIG. 2, the 16-Mbit NAND-type flash memory has 512 physical blocks, and as shown in FIG. 3, the number of logical blocks viewed from the host is defined to be 500. In addition, one block must be provided for storing therein information for easily constructing a PC card ATA interface, and one block must be provided for adopting the additional write system. Therefore, 502 blocks must be provided for controlling this flash memory. Accordingly, the 16-Mbit NAND-type flash memory permits the existence of up to 10 defective blocks.

[0176] However, the flash memory has the upper limit of the number of rewrite operations. When the flash memory is used, a defective block may be produced therein. Therefore, in order to ensure a sufficient storage region, the flash memory must have a writable and erasable effective block as a block replaced when the defective block is produced. In addition, there is a problem in that a flash memory having 10 or more defective blocks can not be shipped since it does not have a sufficient capacity when it is shipped, so that the yield of products is decreased.

[0177] For that reason, the 16-Mbit NAND-type flash memory is provided with a plurality of redundant blocks in addition to 512 body blocks, in order to ensure a large number of effective blocks to be prepared for the occurrence of defective blocks when the flash memory and in order to improve the yield of products when the flash memory is shipped. These redundant blocks are replaced with defective blocks, which have been produced in the 512 body blocks, by means of a redundant circuit as a hardware when the flash memory is shipped. When the redundant circuit is used, the redundant blocks are allocated to addresses of defective blocks. After the replacement as a hardware, if the address of a defective block is selected, the replaced redundant block is selected. Since the number of the redundant blocks is not infinite, a defective block, which has not been replaced when the flash memory is shipped, is treated as an innate defective block. In addition, a defective block, which is produced when the flash memory is used by the user, is treated as an acquired defective block. These innate and acquired defective blocks are relieved by the effective blocks.

[0178] The replacement of the body blocks and redundant blocks is usually carried out by means of the redundant circuit as a hardware. As shown in FIG. 47, the replacement is carried out sequentially from a defective block having a small block address or a large block address for convenience of replacement operation. Therefore, after the replacement of blocks, there are blocks, which can be written or erased sequentially from a small block address or a large block address.

[0179] Thus, as described above, the conventional flash memory control method requires 500 logical blocks, one block for storing therein information for easily constructing a PC card ATA interface, and one block as an empty area to use the additional write system, with respect to 512 physical blocks. Thus, the conventional flash memory can be controlled if it has 502 rewritable blocks. Therefore, the conventional control method permits 10 defective blocks with respect to 512 physical blocks.

[0180] The redundancy operation in this preferred embodiment will be described below. With respect to zone 1, one block for storing therein information for easily constructing a PC card ATA interface is required, and one block as an empty area is required since the additional write system is adopted. Therefore, with respect to 125 logical blocks, 127 rewritable physical blocks are required, and the number of defective blocks, which are permitted in the zone and which can not be rewritten, is up to 1. With respect to zones 2 through 4, since additional one block is required as an additionally writing empty block, 126 rewritable physical blocks are required, and the number of defective block, which are permitted in the zone and which can not be rewritten, is up to 2. In order to simplify the explanation, it is assumed that the number of defective blocks permitted with respect to zones 1 through 4 is 1. Therefore, the specification of the number of permitted defective blocks is very severe in comparison with the specification of the number of defective blocks permitted by the conventional control method.

[0181] In the 16-Mbit NAND-type flash memory shown in FIG. 46, it is assumed that, for example, 7 blocks having block addresses 2, 5, 129, 131, 132, 385 and 389 are defective blocks, and that the flash memory has four redundant blocks. As shown in FIG. 46, when the redundant circuit is used and when defective blocks are replaced with redundant blocks in sequence from a defective block having a small block address similar to the conventional method, the defective blocks after replacement are three blocks having physical block addresses 132, 385 and 389. These blocks are regarded as innate blocks, and innate defective block marks are put on the redundant divisions of the blocks. Since up to 10 innate defective blocks are permitted in the conventional control method, there is particularly no problem to control the flash memory. However, when it is premised that the flash memory is divided into zones to be controlled, two defective blocks having physical block addresses 385 and 389 exist in zone 4, so that it is not possible to control the flash memory and it is not possible to ship this product.

[0182] Thus, when defective blocks exist in the flash memory in this preferred embodiment, if the replacement is carried out sequentially from a defective block having a small block address or a large block address for simple convenience of replacement operation similar to the conventional flash memory, it is not possible to ensure required good blocks, and the possibility of occurrence of unusable zones is increased.

[0183] Thus, in this preferred embodiment, when the flash memory is divided into zones to be controlled, the defective blocks existing in each zone are replaced with redundant blocks so that all the zones meet the number of effective blocks.

[0184] FIG. 48 shows the state of a physical block when defective blocks existing in each zone are replaced with redundant blocks so that the number of defective blocks exceeds the number permitted in each zone. Similar to the flash memory shown in FIG. 46, the flash memory shown in FIG. 48 has seven defective blocks having block addresses 2, 5, 129, 131, 132, 385 and 389 and four redundant blocks. It can be seen that it is possible to ship products, which can not be shipped by the conventional replacement method, as good product. An example of replacement procedure will be described below.

(1) Zones 1 through 4 are searched, and the following variables are derived.
Defective physical block addresses are extracted.
Z(n) BA(m) (n = 1~4, m = 1~ the number of defective blocks in each zone)
The number of defective blocks for each zone is derived.
Z(n)BN (n=1~4).
(2) Even if all of redundant blocks are used with respect to one of n = 1~4 of Z(n) BN, if at least one does not meet the number of effective blocks, the replacement is not carried out and the operation ends.
(3) Among n = 1~4 of Z(n) BN, the maximum n is extracted.
(4) With respect to n extracted in (3), a block corresponding to a physical block stored in Z(n) BA(m) with respect to the minimum or maximum m for storing therein a block address information among Z(n) BA(m) (m = 1~the number of defective blocks in the zone) is replaced with a redundant block.
(5) Z(n) BN = Z(n) BN - 1
(6) With respect to m selected in (3), the block address information of Z(n) BA(m) is deleted.
(7) The processes (3) through (6) are repeated. After all the redundant blocks are used, the process (9) is carried out.
(8) If Z(n) BN (n = 1~4) are equal to each other, (3) through (7) are repeated with respect to the zone of the minimum or maximum n.
(9) Z(n) BN (n = 1~4) is checked. When n exceeding the specification exists, the product is regarded as a defective.
(10) End

[0185] Various methods for replacing body blocks with redundant blocks may be considered. While the flash memory has been divided into four zones in this preferred embodiment, it may be divided into two zones or odd zones. In addition, the capacities (the number of blocks) of divided zones may be different. In either case, in this preferred embodiment, body blocks are replaced with redundant blocks while monitoring the number of defective blocks existing in each zone so that the number of defective blocks existing in each zone after replacement operation exceeds the number of defective blocks permitted in each zone, and the replacement procedure may be modified without departing from the principle of the invention.

[0186] As described above, according to this preferred embodiment, since a table is prepared every zone serving as an object to be accessed unlike conventional methods using a table wherein logical blocks correspond to physical blocks of a flash memory by one to one, it is possible to reduce a RAM area required for the table, and it is possible to control a memory only by means of an integrated RAM of a general purpose CPU although this can not be achieved by the conventional method, so that it is possible to considerably reduce costs in comparison with conventional methods using an external RAM. In particular, these advantages are remarkable when a non-volatile semiconductor memory having a large number of physical blocks is controlled. In addition, it is possible to cope with a large-capacity logical address by a bit number determined by a Block Address Area of a redundant division of a physical block.

[0187] In addition, according to this preferred embodiment, in a flash memory for use in a system using a control method for allocating a plurality of physical blocks to a plurality of logical zones to prepare a translation table between logical blocks and the physical blocks every zone to carry out memory access, defective blocks of the body are replaced with redundant blocks while monitoring the number of defective blocks existing in each zone so that the number of defective blocks existing in each zone after replacement exceeds the number of defective blocks permitted for each zone, although the replacement has been carried out sequentially from a small (or large) address in conventional methods. Therefore, it is possible to reduce the number of products, which can not be shipped since the products have unusable zones, so that it is possible to improve the yield of products.

[0188] The seventh preferred embodiment of the present invention will be described below.

[0189] In this preferred embodiment, there is provided a flash memory card, which can support any types of binary/multi-valued flash memories.

[0190] FIG. 49 is a schematic view of a PC card adapter, to which this preferred embodiment is applied. A flash memory card 101A is a binary flash memory card shown in FIG. 17, and a flash memory card 101B is a four-valued flash memory card shown in FIG. 18. A PC card adapter 102 is provided for transferring the data of the flash memory cards 101A and 101B. The PC card adapter 102 has a 68-pin connector for a PC card slot, and a connector for a flash memory card. In the PC card adapter 102, there is provided a controller 103 for controlling the flash memory card and for electrically interfacing with the PC card slot, an oscillator 104 for a CPU provided in the controller, a RAM 105 for buffer and so forth. In the controller 103, there is provided circuits of two types for error correction, an ECC circuit 1 and an ECC circuit 2, which are directly related to the present invention.

[0191] FIG. 50 shows an example of control when a power supply is turned on.

(Step S1) Detection of Power Supply Voltage

The memory cards include a product having a power supply voltage of 5 V and a product having a power supply voltage of 3.3 V. When a power supply voltage of 5 V is applied to a memory card operated by a power supply voltage of 3.3 V, there is a possibility that voltage destruction and so forth may be caused. In order to avoid this, the system detects the power supply voltage.

(Steps S2 and S3) ID Check

There are various kinds of flash memory cards having different storage capacities or interface specifications. When a memory card is inserted into the system, the system determines a maker code, a device code or the like of the device. When the determined code is an unsupported code, a new access is not carried out. In order to read the maker code, the device code or the like, a normal power supply voltage is inputted.

(Steps S4 through S8) ECC Check

The system of ECC is checked herein.

[0192] Examples of methods for recognizing the ECC system will be described below. A first method is a method for determining a device code similar to the above described steps S2 and S3. For example, if flash memories have the same capacity, the device code may be changed by the memory construction.

[0193] FIG. 51 shows the constructions of data and redundant divisions of a 16-Mbit flash memory in this preferred embodiment. The different point from the conventional memory shown in FIG. 4 is that a User Data Area of the redundant division is allocated to each of an ECC Flag Area and an ECC Area-3. As shown in FIG. 51, three areas of ECC Area-1 through ECC Area-3 are defined as areas for storing ECC codes. Referring to FIG. 52, a method for using the three areas will be described. Information on the ECC system is stored in an ECC Flag Area byte. For example, in the case of ECC system 1, "AAh" is defined as Flag data, and in the case of ECC system 2, "55h" is defined as Flag data.

[0194] In the case of a flash memory card based on the ECC system 1, a 3-byte ECC code of an even page data (256 bytes) enters the ECC Area-1, and a 3-byte ECC code of an odd page data (256 bytes) enters the ECC Area-2. The ECC Area-3 is null. It is possible to correct a 2-bit error of each of 256 bytes in the ECC Area-1 and ECC Area-2.

[0195] In the case of a flash memory card based on the ECC system 2, ECC codes with respect to 512 bytes are dispersively stored in the ECC Area-1, ECC Area-2 and ECC Area-3. In this case, it is possible to correct a 2-bit error of 512-byte data.

[0196] The controller 103 in the PC card adapter 102 of FIG. 49 reads the ECC Flag Area out to determine the ECC system when the flash memory card is inserted into the adapter.

[0197] In FIG. 49, the code generation and an error detection circuit, which correspond to each of the ECC system 1 and ECC system 2, are selected. The CPU in the controller controls the flow of data between the host and the RAM for buffer and the flash memory. In FIG. 49, the ECC circuit must not always be a hardware. All the generation of the ECC codes and so forth may be carried out by a software.

[0198] The present invention should not be limited to the above described preferred embodiment. The present invention may be embodied in various ways without departing from the principle of the invention.

[0199] While the existence of two kinds of ECC systems has been assumed in the above described preferred embodiment, the present invention should not be limited thereto, but three or more kinds of ECC systems may be set.

[0200] The selection of the ECC system includes the selection that no ECC is used. In the fields of a flash memory card having very high reliability, and in the field of data, which does not requires particularly high reliability, such as data in voice field or the like, the ECC is not essential. In this case, it may be defined that the ECC is not used if the content of the above described ECC Flag Area is "FFh".

[0201] While the ECC system has been defined every flash memory in the above described preferred embodiment, the present invention should not be limited. For example, the ECC system may be switched every optional unit, such as every sector or block. In this case, the ECC system may be switched each time access is carried out every the above described unit without simply determining the ECC system when a power supply is turned on. In addition, for example, the operation for converting data read out by the ECC system 1 to the ECC system 2 to restore the converted data may be supposed.

[0202] While the PC card adapter has been used in the above described preferred embodiment, the present invention should not be limited thereto. The present invention may be applied to various apparatuses, such as a digital steel camera, a PDA, a word processor and a voice recorder. Thus, according to this preferred embodiment, since a flash memory card having a very wide use range may be treated by one system, the flexibility can be remarkably improved.

[0203] According to the present invention, when a storage medium is inserted into or ejected from a connector in an electronic apparatus, the contact and breakaway of various pins are carried out in sequence, so that it is possible to improve the stability of operation and to surely protect data.

[0204] In addition, according to the present invention, when the translation between logical addresses and physical addresses is carried out by means of a translation table, a plurality of physical blocks are allocated to one logical block, so that it is possible to reduce a RAM area required for the table.

[0205] Moreover, according to the present invention, the delimiter of cluster does not straddle a block serving as an erase unit, so that it is possible to decrease the numbers of erase and write operations occurring when data are rewritten.

[0206] In addition, according to the present invention, when an erase command is executed, the management area for physical blocks is open, and simultaneously, the data area is also erased, so that it is possible to improve the processing speed when a subsequent write command is executed.

[0207] In addition, according to the present invention, only the correspondence relationship in a required area of the correspondence relationships between logical blocks and physical blocks is prepared one by one, so that it is possible to reduce the RAM area required at the minimum.

[0208] In addition, according to the present invention, defective blocks are replaced with redundant blocks so that the number of defective blocks existing in each logical zone does not exceed a predetermined value after replacement, so that it is possible to improve the yield of products.

[0209] Moreover, according to the present invention, the error correction algorithm is selected in accordance with the kind of the used storage medium, so that it is possible to use various storage media to improve the flexibility.

[0210] While the present invention has been disclosed in terms of the preferred embodiment in order to facilitate better understanding thereof, it should be appreciated that the invention can be embodied in various ways without departing from the principle of the invention. Therefore, the invention should be understood to include all possible embodiments and modification to the shown embodiments which can be embodied without departing from the principle of the invention as set forth in the appended claims.

Claims

1. A method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

   said storage medium having ground terminals, power supply terminals, control terminals and data input/output terminals,
   said connector having a function of being sequentially connected to each of said terminals,
   wherein when said storage medium is inserted into said connector, said ground terminals and control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminals of said storage medium are connected to corresponding terminals of said connector.

2. A method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

said storage medium having ground terminals, power supply terminals, control terminals and data input/output terminals,
said connector having a function of being sequentially disconnected from each of said terminals,
wherein when said storage medium is ejected from said connector, said power supply terminals and data input/output terminals of said storage medium are disconnected from corresponding terminals of said connector before said ground terminals and control terminals of said storage medium are disconnected from corresponding terminals of said connector.

3. A method for controlling a memory system as set forth in claim 1, wherein a signal transmitted to said control terminal is a command latch enable signal.

4. A method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

said storage medium having ground terminals, power supply terminals, a first control terminal, a second control terminal and data input/output terminals,
said connector having a function of being sequentially connected to each of said terminals,
wherein when said storage medium is inserted into said connector, said ground terminals and first control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminal of said storage medium are connected to corresponding terminals of said connector, and
said second control terminal being connected to a corresponding terminal of said connector before said data input/output terminals are connected to a corresponding terminal of said connector.

5. A method for controlling a memory system using a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

said storage medium having ground terminals, a power supply terminals, a first control terminal, a second control terminal and data input/output terminals,
said connector having a function of being sequentially connected to each of said terminals,
wherein when said storage medium is ejected from said connector, said data input/output terminals are disconnected from corresponding terminals of said connector before said second control terminal is disconnected from a corresponding terminal of said connector, and
said power supply terminals and data input/output terminals of said storage medium being disconnected from corresponding terminals of said connector before said ground terminals and first control terminal of said storage medium are disconnected from corresponding terminals of said connector.

6. A method for controlling a memory system having a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

said storage medium having ground terminals, power supply terminals, first control terminal, a second control terminal, data input/output terminals and an insertion/ejection detecting terminal,
said connector having a function of being sequentially connected to each of said terminals,
wherein when said storage medium is inserted into said connector, said ground terminals and first control terminals of said storage medium are connected to corresponding terminals of said connector before said power supply terminals and data input/output terminals of said storage medium are connected to corresponding terminals of said connector,
said second control terminal of said storage medium being connected to a corresponding terminal of said connector before said data input/output terminals of said storage medium are connected to corresponding terminals of said connector, and
said insertion/ejection detecting terminal being connected to a corresponding terminal of said connector after all of said terminals of said storage medium are inserted.

7. A method for controlling a memory system having a storage medium, which is inserted into an electronic apparatus via a connector to add a memory function thereto,

said storage medium having ground terminals, power supply terminals, a first control terminal, a second control terminal, data input/output terminals and an insertion/ejection detecting terminal,

said connector having a function of being sequentially connected to each of said terminals,
wherein when said storage medium is ejected from said connector, said insertion/ ejection detecting terminal is disconnected from a corresponding terminal of said connector before said data input/output terminals of said storage medium are disconnected from corresponding terminals of said connector,
said data input/output terminals of said storage medium being disconnected from corresponding terminals of said connector before said power supply terminals and second control terminal of said storage medium are disconnected from corresponding terminals of said connector, and
said power supply terminals of said storage medium being disconnected from corresponding terminals of said connector before said ground terminals and first control terminal of said storage medium are disconnected from corresponding terminals of said connector.

8. A method for controlling a memory system as set forth in claim 7, wherein said storage medium completes the reset of write or erase operation before said power supply terminals of said storage medium are disconnected from the corresponding terminals of said connector after said insertion/ejection detecting terminal of said storage medium is disconnected from the corresponding terminal of said connector.

9. A method for controlling a memory system as set forth in claim 4, wherein a signal transmitted to said first control terminal is a signal for receiving a command, and a signal transmitted to said second control terminal is a signal for inactivating a data output terminal.

10. A method for controlling a memory system which includes:

logical blocks managed by the system;
physical blocks for storing therein data corresponding to said logical blocks, said physical blocks comprising a plurality of memory cells;
redundant divisions included in a corresponding said physical blocks for storing therein addresses of corresponding said logical blocks; and
physical block areas formed by at least two of said physical blocks,
wherein a logical address/physical address translation table is prepared for managing corresponding relationships between said logical blocks and said physical blocks.

11. A method for controlling a memory system as set forth in claim 10, wherein when a memory access is carried out, said physical blocks corresponding to said logical blocks are selected by referring to said logical address/physical address translation table to read addresses of said physical block areas corresponding to said logical blocks and to read addresses of said corresponding logical blocks stored in said redundant divisions of at least two of said physical blocks forming said physical block area.

12. A method for controlling a memory system as set forth in claim 10, wherein said logical address/physical address translation table is prepared when a power supply is turned on.

13. A method for controlling a memory system, which is a semiconductor memory system for storing a file managed by a first predetermined units, in a storage area divided into second predetermined units,

wherein said semiconductor memory system is controlled so that boundaries between said first predetermined units are arranged on boundaries between said second predetermined units.

14. A method for controlling a memory system as set forth in claim 13, wherein said second predetermined unit is an erase unit.

15. A method for controlling a memory system which includes:

files managed by the system;
data areas for storing therein the contents of said files; and
management areas for storing therein a corresponding relationships between said files and said data areas,
wherein when said file is erased, it is marked that said data areas corresponding to said management areas are empty area, to erase said corresponding data areas.

16. A method for controlling a memory system which includes:

files managed by the system;
data areas for storing therein the contents of said files; and
management areas for storing therein a corresponding relationships between said files and said data areas, wherein when said file is erased, it is marked that said data areas corresponding to said management areas are empty area, and the contents of said management areas are detected, on the basis of signals inputted to said memory system, to erase said data areas.

17. A method for controlling a non-volatile semiconductor memory system, which comprises the steps of:

dividing a cell array into a plurality of physical blocks;
storing each information corresponding to relationship between said physical block and logical block which is managed by said system, in each said physical block; and
in order to form a table for managing corresponding relationships between said logical blocks and said physical blocks, in a random access memory in said system, sequentially preparing required corresponding relationships of corresponding relationships between said logical blocks and said physical blocks, in said random access memory in said system in accordance with accesses from a host.

18. A method for controlling a non-volatile semiconductor memory system as set forth in claim 17, wherein said physical blocks are comprised of non-volatile semiconductor memories as flash memories.

19. A method for controlling a non-volatile semiconductor memory system which comprises the steps of:

dividing a cell array of non-volatile semiconductor memory cells into a plurality of physical blocks;
storing each information corresponding to relationship between said physical block and logical block address in each said physical block, which logical blocks are managed by said system,
in order to form an address translation table for managing corresponding relationships between said physical blocks and said logical block addresses, in a random access memory in said memory system,
forming a plurality of areas, each area being formed by an aggregate of at least one of said plurality of physical blocks,
controlling said system so that data in predetermined address of said logical block are stored in said predetermined area,
forming an address translation table corresponding to said predetermined area in which data in said predetermined address of said logical block are stored, if necessary, when said non-volatile semiconductor memory is accessed.

20. A method for controlling a non-volatile semiconductor memory system which comprises the steps of:

dividing a cell array of non-volatile memory cells into a plurality of physical blocks;
storing each information corresponding to relationship between said physical block and logical block in a storage region of each of said physical blocks, which logical blocks are managed by said system; and
forming a table for managing a corresponding relationship between said logical blocks and said physical blocks of a flash memory, in a random access memory of said memory system,
said method further comprising the steps of:
ensuring an area formed by one or a plurality of physical blocks, on a cell array of said flash memory;
in every memory access time, searching said object area of physical blocks,
forming said table for managing said corresponding relationship between said logical blocks and said physical blocks, on said random access memory of said system,
allowing to select physical blocks corresponding to said logical blocks by using said table.

21. A method for controlling a non-volatile semiconductor memory system, as set forth in claim 20, which further comprises the steps of:

providing a function of selectively replacing a defective physical blocks including defective cells with redundant physical blocks; and
managing said functions, for said each area, so that the number of defective physical blocks is less than or equal to a predetermined number.

22. A method for controlling a non-volatile semiconductor memory system, which allows any one of various memory

units to be detachably mounted in a body of said memory system, said method comprising a step of selecting a corresponding one of various error correcting means of said body in accordance with said one of various memory units mounted in said body, to carry out error correction.

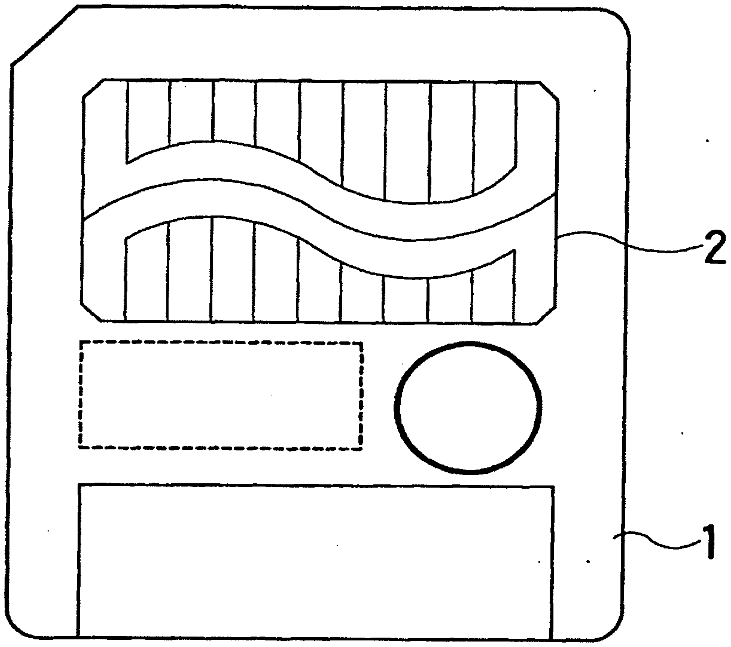


FIG. 1

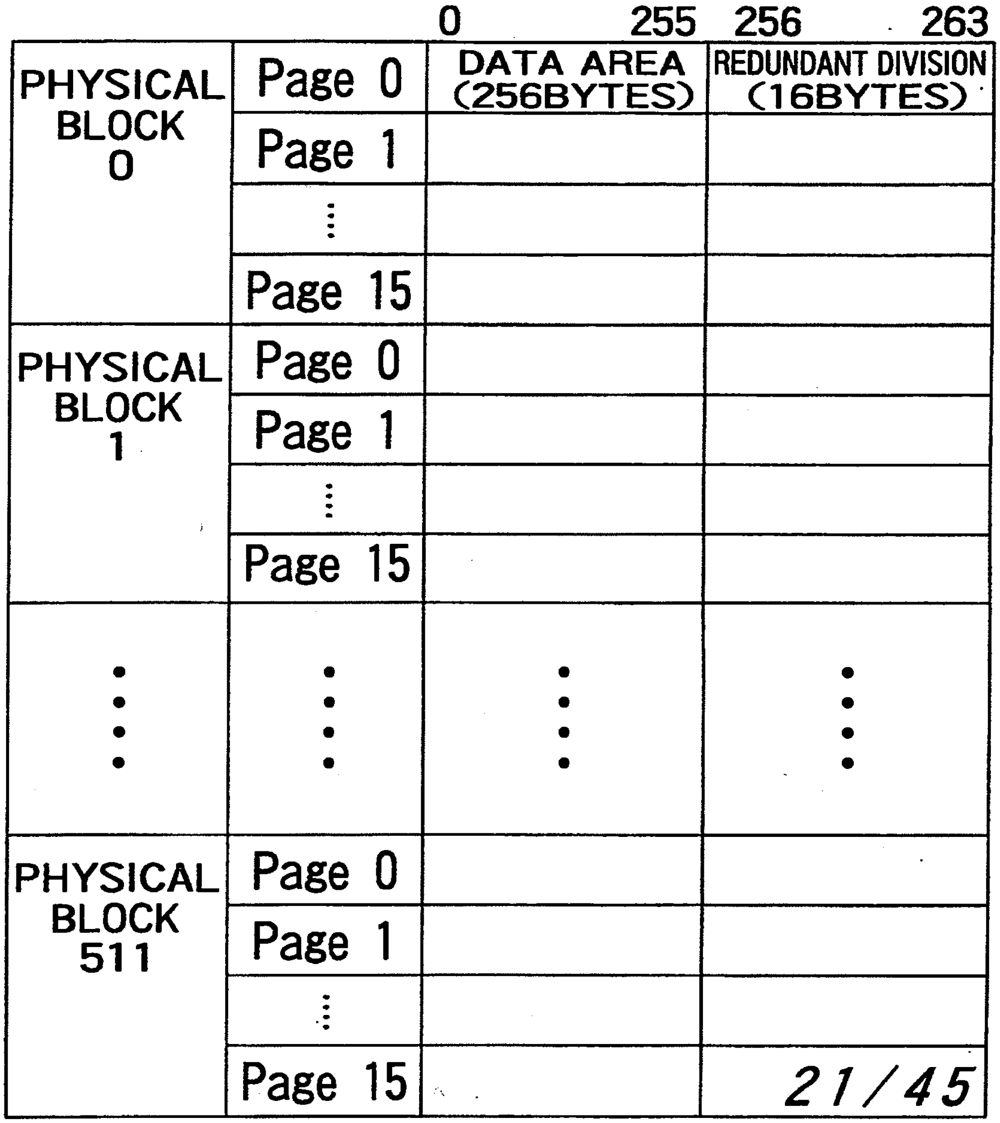

0
255
256
263
PHYSICAL BLOCK 0
Page 0
Page 1
Page 15
DATA AREA (256BYTES)
REDUNDANT DIVISION (16BYTES)
PHYSICAL BLOCK 1
Page 0
Page 1
Page 15
PHYSICAL BLOCK 511
Page 0
Page 1
Page 15

FIG.2

PHYSICAL BLOCK 0	SECTOR 0	512 BYTES
	SECTOR 1	
	⋮	
	SECTOR 7	
PHYSICAL BLOCK 1	SECTOR 8	
	SECTOR 9	
	⋮	
	SECTOR 15	
⋮	⋮	⋮
PHYSICAL BLOCK 499	SECTOR 3992	
	SECTOR 3993	
	⋮	
	SECTOR 3999	

FIG.3

DATA DIVISION

BYTE	PAGE 0(EVEN PAGE)	PAGE 1(ODD PAGE)
0~255	DATA Area-1	DATA Area-2

REDUNDANT DIVISION

BYTE	EVEN PAGE	ODD PAGE
256	User Data Area	ECC Area-2
257		
258		
259		Block Address Area-2
260	Data Status Area	
261	Block Status Area	ECC Area-1
262	Block Address Area-1	
263		

FIG.4

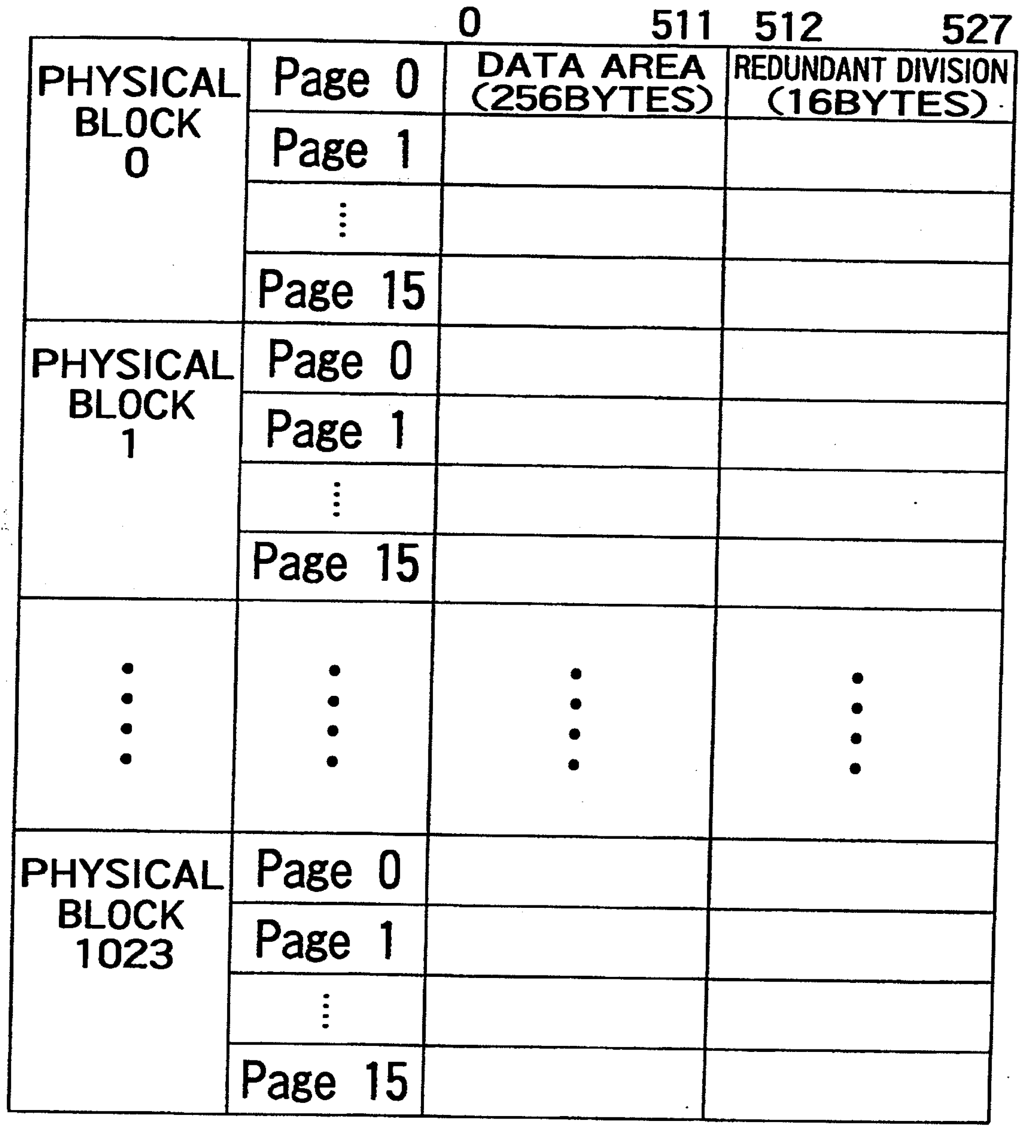

0
511
512
527
PHYSICAL BLOCK 0
Page 0
Page 1
Page 15
DATA AREA (256BYTES)
REDUNDANT DIVISION (16BYTES)
PHYSICAL BLOCK 1
Page 0
Page 1
Page 15
PHYSICAL BLOCK 1023
Page 0
Page 1
Page 15

FIG.5

LOGICAL BLOCK 0	SECTOR 0	512 BYTES
	SECTOR 1	
	⋮	
	SECTOR 15	
LOGICAL BLOCK 1	SECTOR 16	
	SECTOR 17	
	⋮	
	SECTOR 31	
⋮	⋮	⋮
LOGICAL BLOCK 999	SECTOR 15984	
	SECTOR 15985	
	⋮	
	SECTOR 15999	

FIG.6

DATA DIVISION

BYTE	
0~511	DATA Area

REDUNDANT DIVISION

BYTE	
512	User Data Area
513	
514	
515	
516	Data Status Area
517	Block Status Area
518	Block Address Area-1
519	
520	ECC Area-2
521	
522	
523	Block Address Area-2
524	
525	ECC Area-1
526	
527	

FIG.7

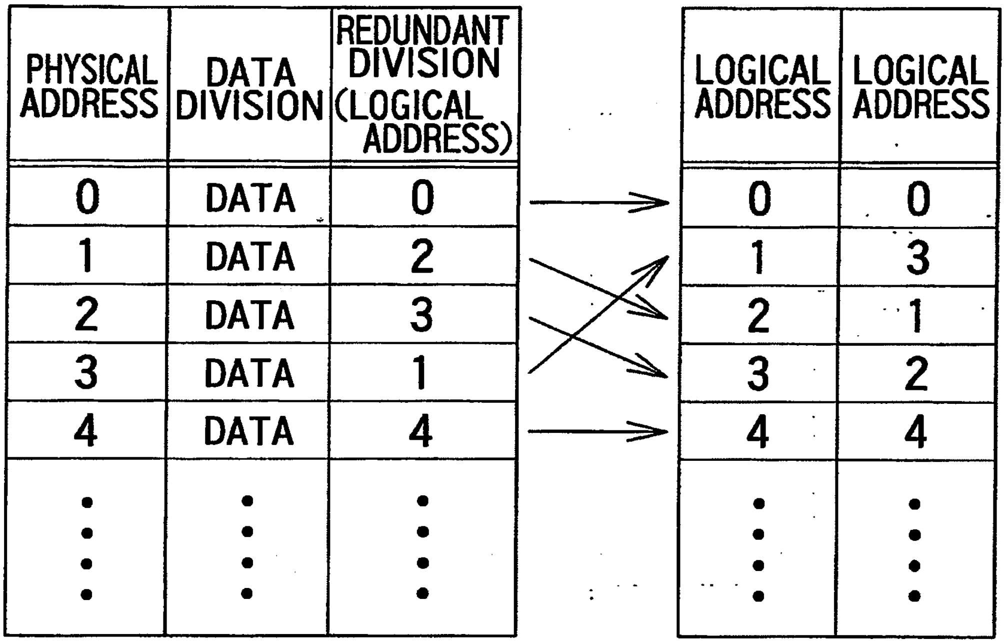

PHYSICAL ADDRESS
DATA DIVISION
REDUNDANT DIVISION (LOGICAL ADDRESS)
0 DATA 0
1 DATA 2
2 DATA 3
3 DATA 1
4 DATA 4
LOGICAL ADDRESS
LOGICAL ADDRESS
0 0
1 3
2 1
3 2
4 4

FIG.8

OFFSET (LOGICAL BLOCK ADDRESS)	PHYSICAL BLOCK ADDRESS	PHYSICAL BLOCK ADDRESS (BINARY DATA)			
		OPPER BYTE		LOWER BYTE	
word0(LBA=0)	0	0000	0000	0000	0000
word1(LBA=1)	500	0000	0001	1111	0100
word2(LBA=2)	327	0000	0001	0100	0111
⋮	⋮	⋮	⋮	⋮	⋮
word497(LBA=497)	244	0000	0000	1111	0100
word498(LBA=498)	249	0000	0001	1110	1111
word499(LBA=499)	128	0000	0001	1000	0000

FIG.9

OFFSET (LOGICAL BLOCK ADDRESS)	PHYSICAL BLOCK ADDRESS	PHYSICAL BLOCK ADDRESS (BINARY DATA)			
		OPPER BYTE		LOWER BYTE	
word0(LBA=0)	0	0000	0000	0000	0000
word1(LBA=1)	1000	0000	0011	1110	1000
word2(LBA=2)	654	0000	0010	1000	1110
⋮	⋮	⋮	⋮	⋮	⋮
word997(LBA=997)	488	0000	0001	1110	1000
word998(LBA=998)	498	0000	0001	1111	0010
word999(LBA=999)	256	0000	0001	0000	0000

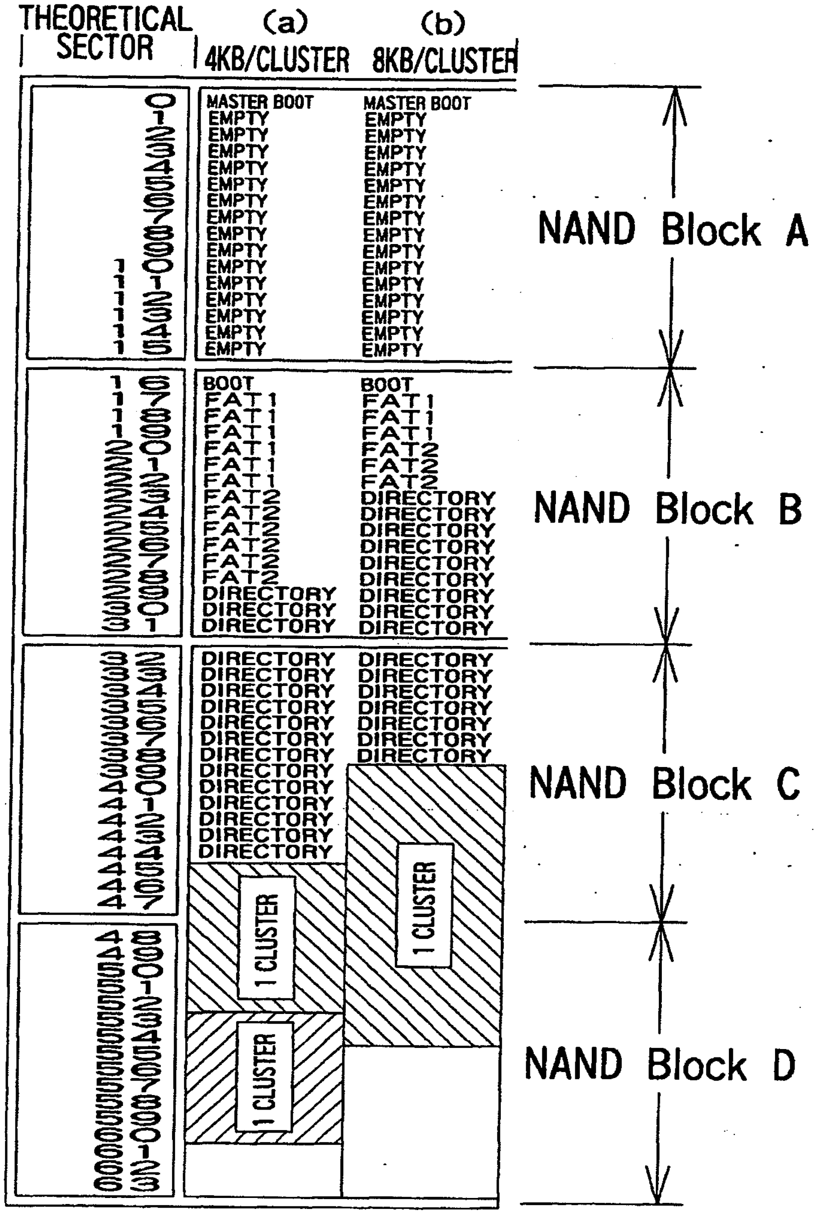
FIG.10

D7	D6	D5	D4	D3	D2	D1	D0	256 + 8 BYTE/PAGE
0	0	0	1	BA10	BA9	BA8	BA7	262 BYTE(EVEN PAGE) 259 BYTE(ODD PAGE)
BA6	BA5	BA4	BA3	BA2	BA1	BA0	P	263 BYTE(EVEN PAGE) 260 BYTE(ODD PAGE)

BA10~BA0:LOGICAL BLOCK ADDRESS

P EVEN PARITY BIT "1" FIXED VALUE

FIG.11



THEORETICAL SECTOR	(a) 4KB/CLUSTER	(b) 8KB/CLUSTER
0	MASTER BOOT	MASTER BOOT
1	EMPTY	EMPTY
2	EMPTY	EMPTY
3	EMPTY	EMPTY
4	EMPTY	EMPTY
5	EMPTY	EMPTY
6	EMPTY	EMPTY
7	EMPTY	EMPTY
8	EMPTY	EMPTY
9	EMPTY	EMPTY
10	EMPTY	EMPTY
11	EMPTY	EMPTY
12	EMPTY	EMPTY
13	EMPTY	EMPTY
14	EMPTY	EMPTY
15	EMPTY	EMPTY
16	BOOT	BOOT
17	FAT1	FAT1
18	FAT1	FAT1
19	FAT1	FAT1
20	FAT1	FAT2
21	FAT1	FAT2
22	FAT1	FAT2
23	FAT2	DIRECTORY
24	FAT2	DIRECTORY
25	FAT2	DIRECTORY
26	FAT2	DIRECTORY
27	FAT2	DIRECTORY
28	FAT2	DIRECTORY
29	DIRECTORY	DIRECTORY
30	DIRECTORY	DIRECTORY
31	DIRECTORY	DIRECTORY
32	DIRECTORY	DIRECTORY
33	DIRECTORY	DIRECTORY
34	DIRECTORY	DIRECTORY
35	DIRECTORY	DIRECTORY
36	DIRECTORY	DIRECTORY
37	DIRECTORY	DIRECTORY
38	DIRECTORY	DIRECTORY
39	DIRECTORY	1 CLUSTER
40	DIRECTORY	
41	DIRECTORY	
42	DIRECTORY	
43	DIRECTORY	
44	DIRECTORY	
45	1 CLUSTER	
46		
47		
48		
49		
50		
51		
52		
53	1 CLUSTER	
54		
55		
56		
57		
58		
59		
60		
61		
62		
63		

FIG. 12

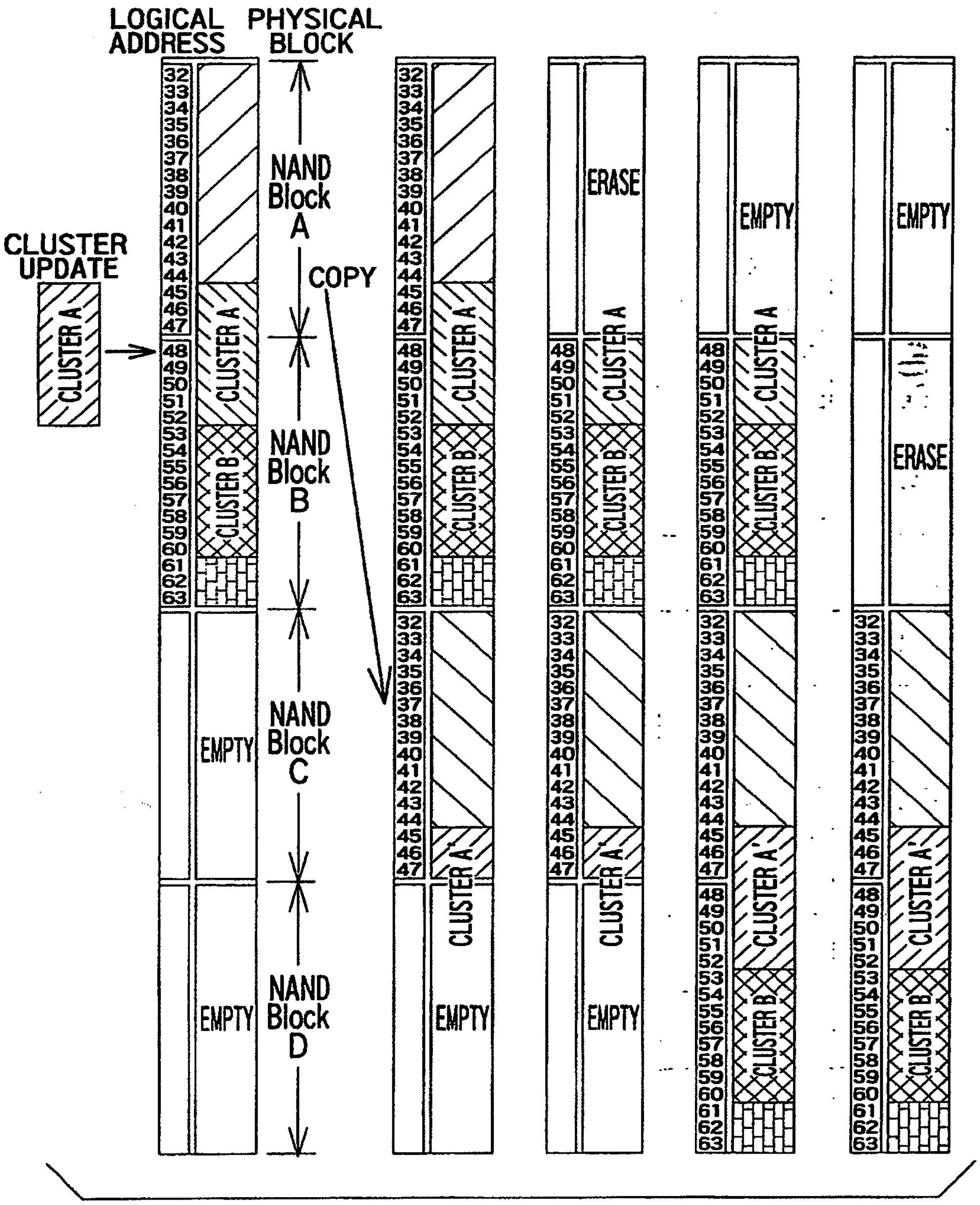

LOGICAL ADDRESS
PHYSICAL BLOCK
CLUSTER UPDATE
CLUSTER A
NAND Block A
COPY
NAND Block B
NAND Block C
NAND Block D
CLUSTER B
CLUSTER A'
EMPTY
ERASE
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

FIG.13

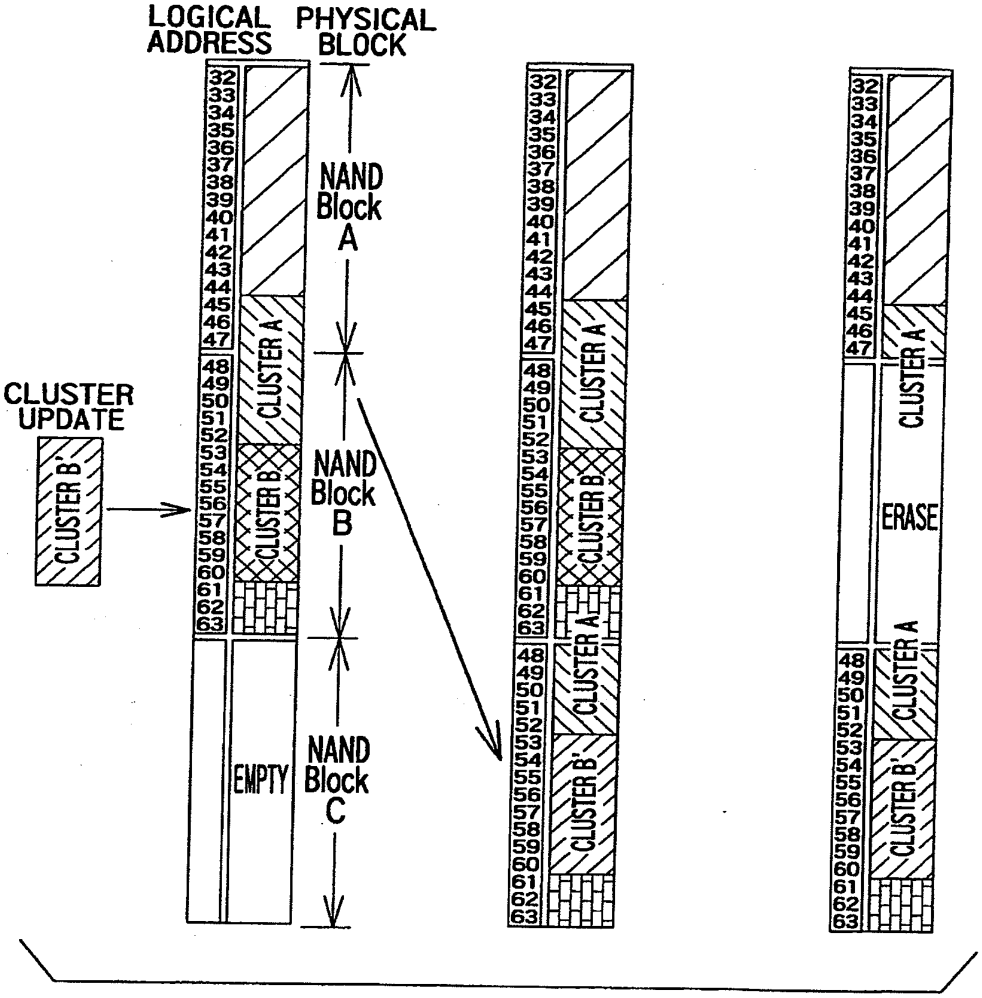

LOGICAL ADDRESS
PHYSICAL BLOCK
NAND Block A
NAND Block B
NAND Block C
CLUSTER A
CLUSTER B
CLUSTER B'
CLUSTER UPDATE
EMPTY
ERASE

FIG. 14

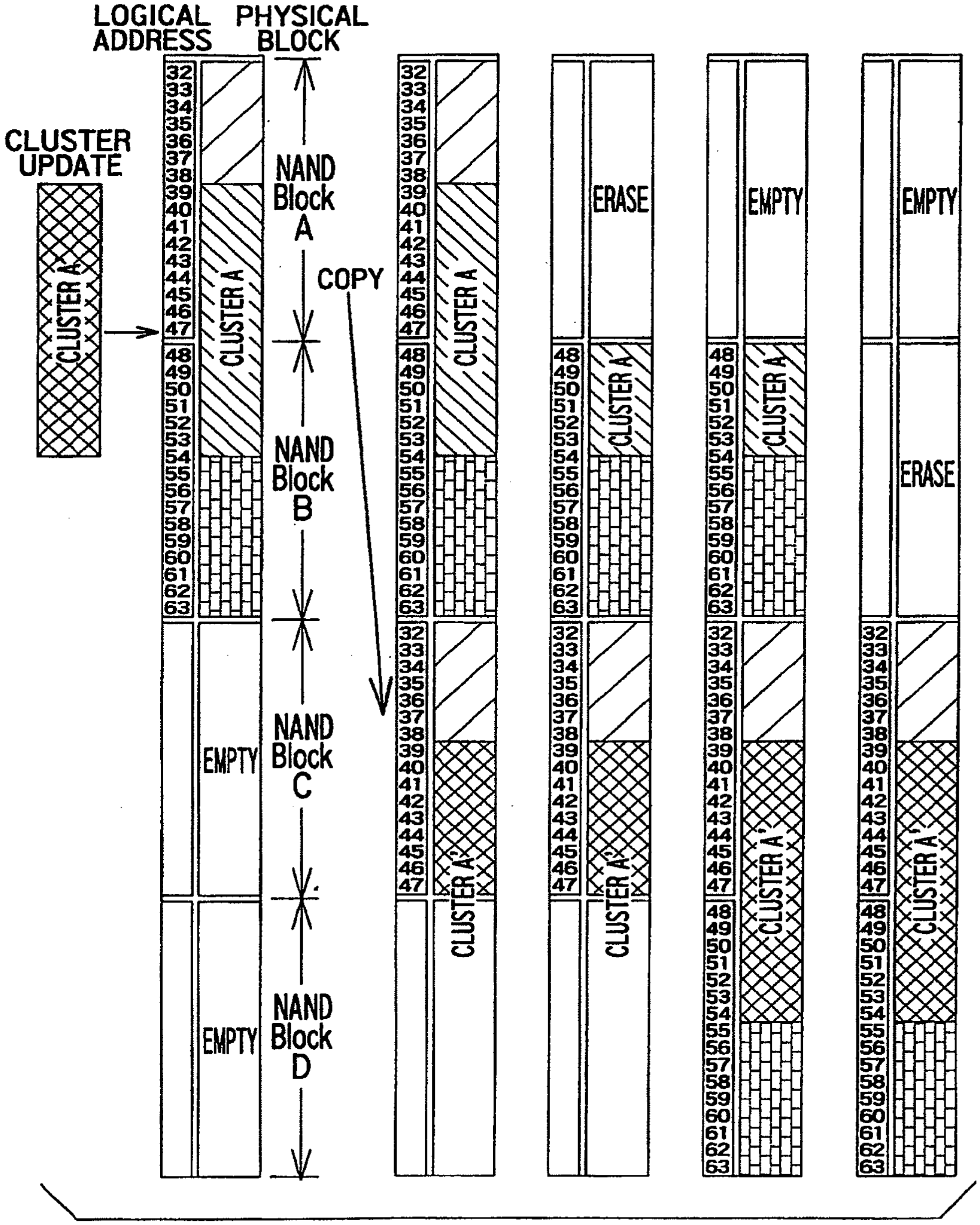

LOGICAL ADDRESS
PHYSICAL BLOCK
CLUSTER UPDATE
CLUSTER A'
CLUSTER A
NAND Block A
NAND Block B
NAND Block C
NAND Block D
COPY
EMPTY
ERASE
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

FIG. 15

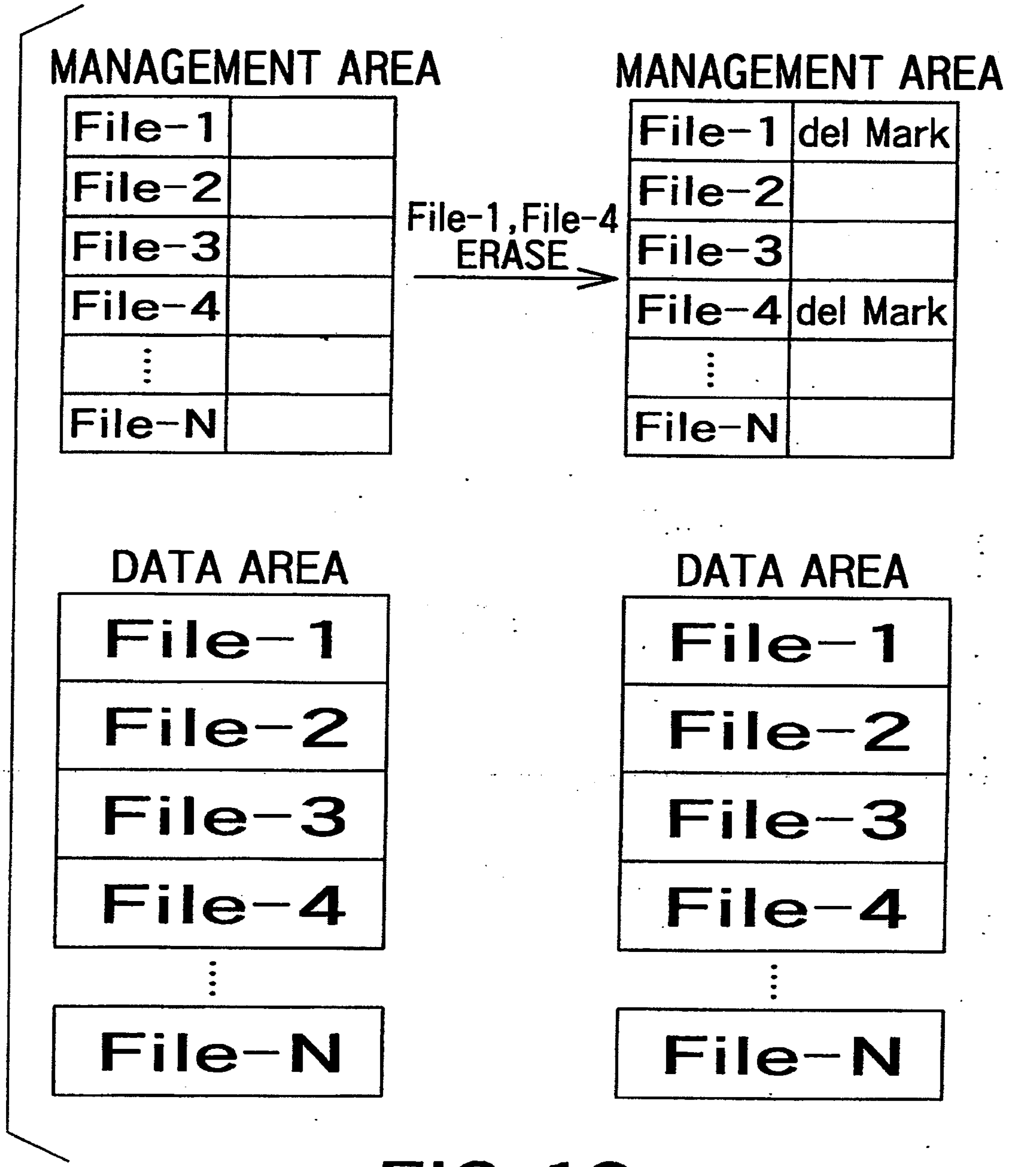

MANAGEMENT AREA
File-1
File-2
File-3
File-4
File-N
File-1,File-4
ERASE
MANAGEMENT AREA
File-1
del Mark
File-2
File-3
File-4
del Mark
File-N
DATA AREA
File-1
File-2
File-3
File-4
File-N
DATA AREA
File-1
File-2
File-3
File-4
File-N

FIG. 16

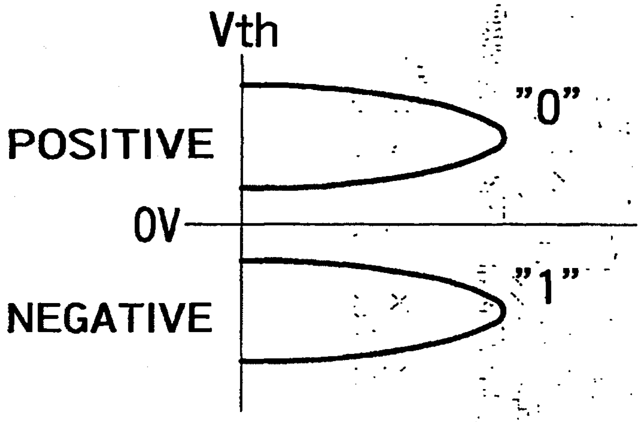

Vth
POSITIVE
"0"
0V
NEGATIVE
"1"

FIG.17

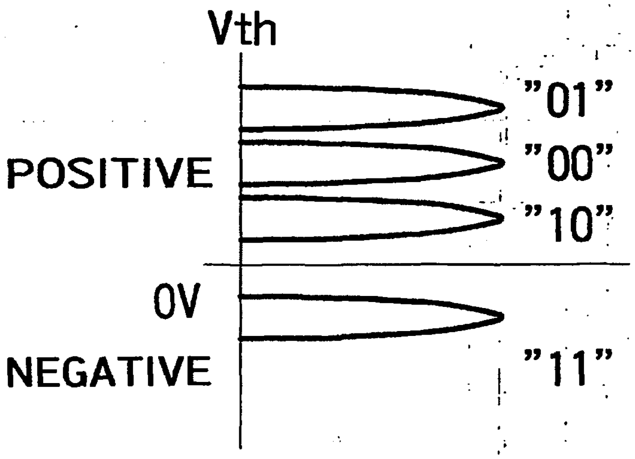
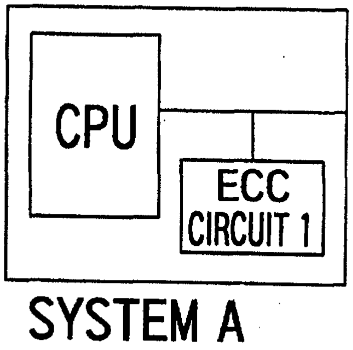
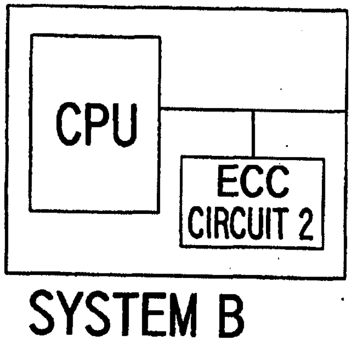

Vth
POSITIVE
"01"
"00"
"10"
0V
NEGATIVE
"11"

FIG.18

	CARD IN FIG.2(a)	CARD IN FIG.2(b)
	AVAILABLE	VNAVAILABLE
	VNAVAILABLE	AVAILABLE

CPU
ECC CIRCUIT 1
SYSTEM A
CPU
ECC CIRCUIT 2
SYSTEM B

FIG.19

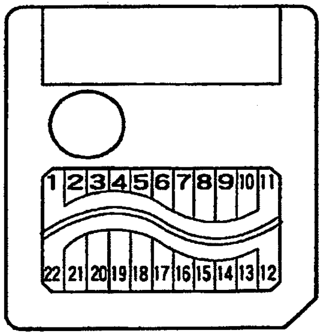


FIG.20

1,10,11	Vss	POWER SUPPY(GND)
2	CLE	COMMAND LATCH ENABLE
3	ALE	ADDRESS LATCH ENABLE
4	$\overline{WE}$	WRITE ENABLE
5	$\overline{WP}$	WRITE PROTECT
6−9	I/O_{1-4}	ADDRESS DATA COMMAND INPUT-OUTPUT PORT
13-16	I/O_{5-8}	ADDRESS DATA COMMAND INPUT-OUTPUT PORT
17	NC	N_C
18	GND	GND LEVEL INPUT
19	$R/\overline{B}$	READY BUSY OUTPUT
20	$\overline{RE}$	READ ENABLE
21	$\overline{CE}$	CHIP ENABLE
22,23	Vcc	POWER SUPPY

FIG.21

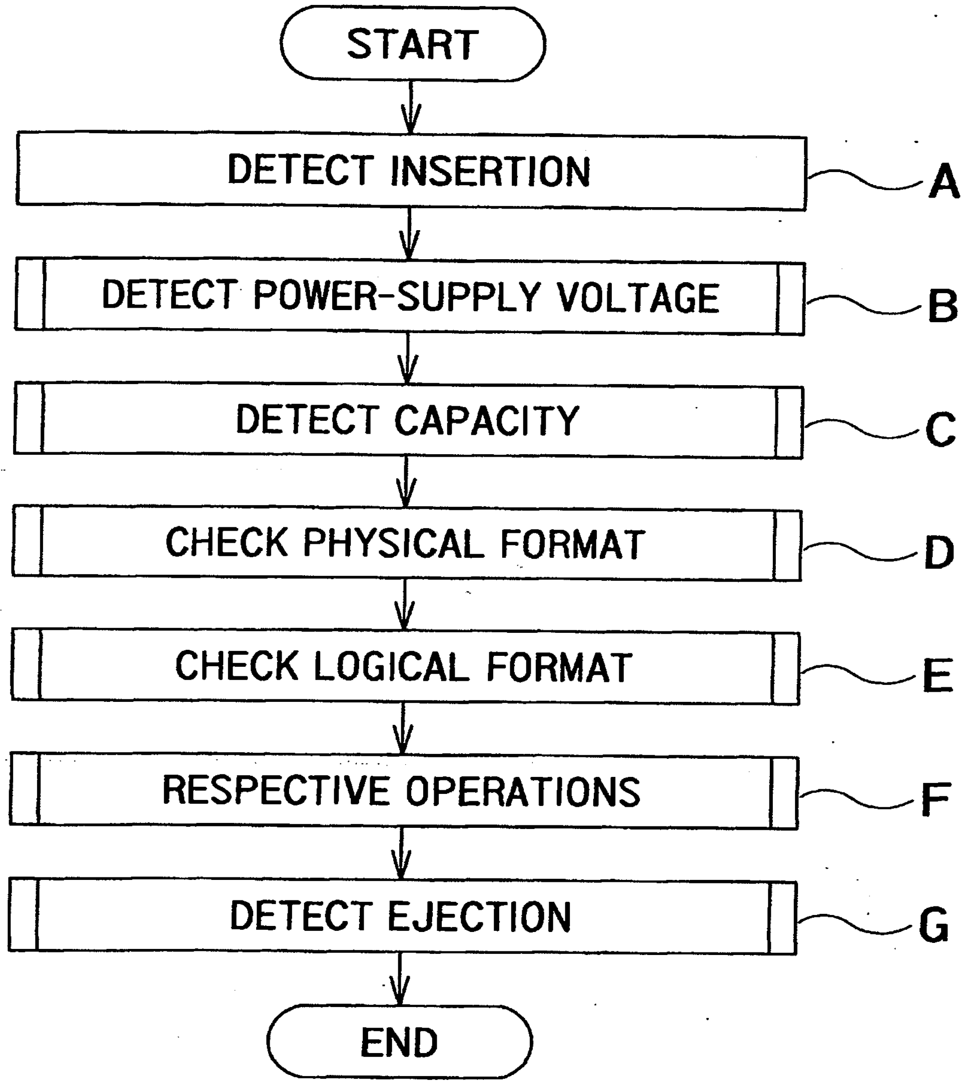
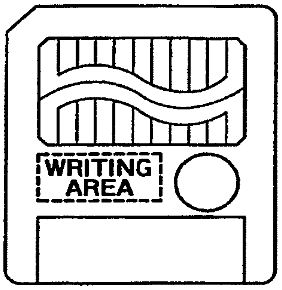
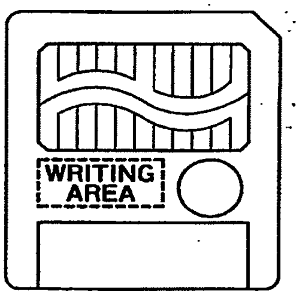

START
DETECT INSERTION
A
DETECT POWER-SUPPLY VOLTAGE
B
DETECT CAPACITY
C
CHECK PHYSICAL FORMAT
D
CHECK LOGICAL FORMAT
E
RESPECTIVE OPERATIONS
F
DETECT EJECTION
G
END

FIG.22


WRITING
AREA

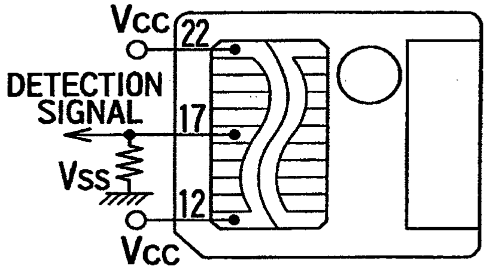
5V PRODUCT

FIG.23(a)


WRITING
AREA

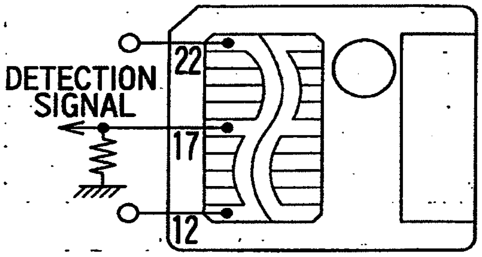
3.3V PRODUCT

FIG.23(b)


Vcc
22
DETECTION
SIGNAL
17
Vss
12
Vcc

5V PRODUCT

FIG.24(a)


DETECTION
SIGNAL
22
17
12

3.3V PRODUCT

FIG.24(b)

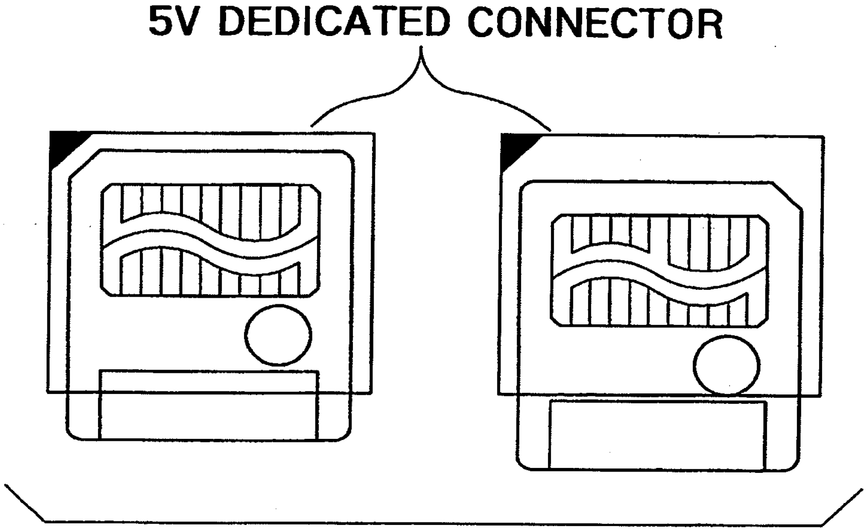

5V DEDICATED CONNECTOR

FIG.25

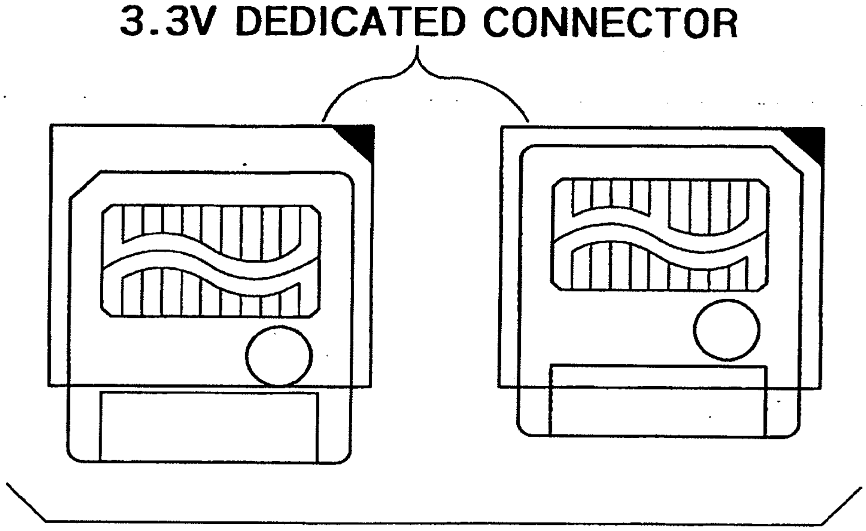

3.3V DEDICATED CONNECTOR

FIG.26

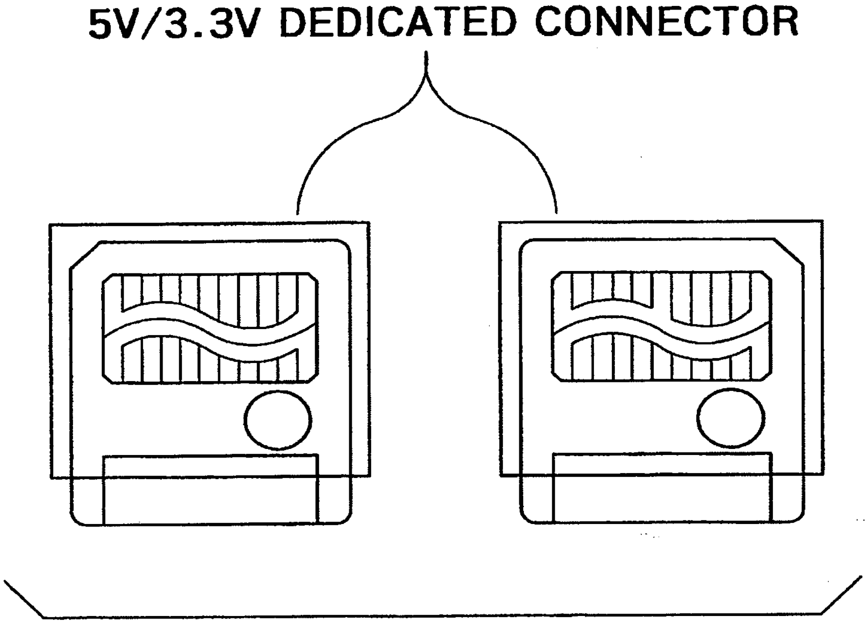

5V/3.3V DEDICATED CONNECTOR

FIG.27

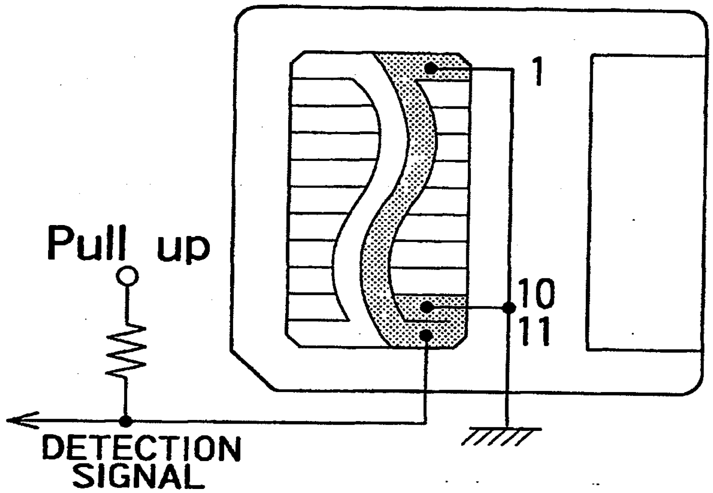

1
Pull up
10
11
DETECTION SIGNAL

FIG.28

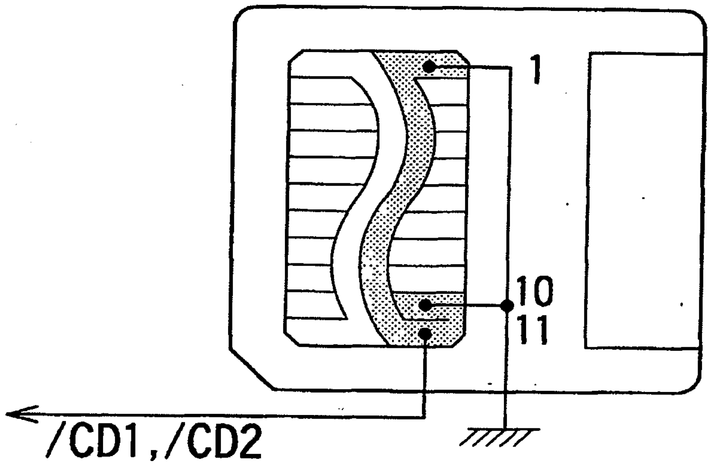

1
10
11
/CD1,/CD2

FIG.29

	bit7	bit6					bit1	bit0
1stByte	00000000 111	00000000 110					00000000 001	00000000 000
2ndByte	00000000 111	00000001 110					00000001 001	00000001 000
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
255thByte	11111110 111	11111110 110					11111110 001	11111110 000
266thByte	11111111 111	11111111 110					11111111 001	11111111 000

FIG.30

LP00=D(*******0、 ***) 、LP01=D(*******1、 ***)

LP02=D(******0*、 ***) 、LP03=D(******1*、 ***)

LP04=D(*****0**、 ***) 、LP05=D(*****1**、 ***)

LP06=D(****0***、 ***) 、LP07=D(****1***、 ***)

LP08=D(***0****、 ***) 、LP09=D(***1****、 ***)

LP010=D(**0*****、 ***) 、LP011=D(**1*****、 ***)

LP012=D(*0******、 ***) 、LP013=D(*1******、 ***)

LP014=D(0*******、 ***) 、LP015=D(1*******、 ***)

LP00=D(********、 **0) 、LP01=D(********、 **1)

LP02=D(********、 *0*) 、LP03=D(********、 *1*)

LP04=D(********、 0**) 、LP05=D(********、 1**)

FIG.31

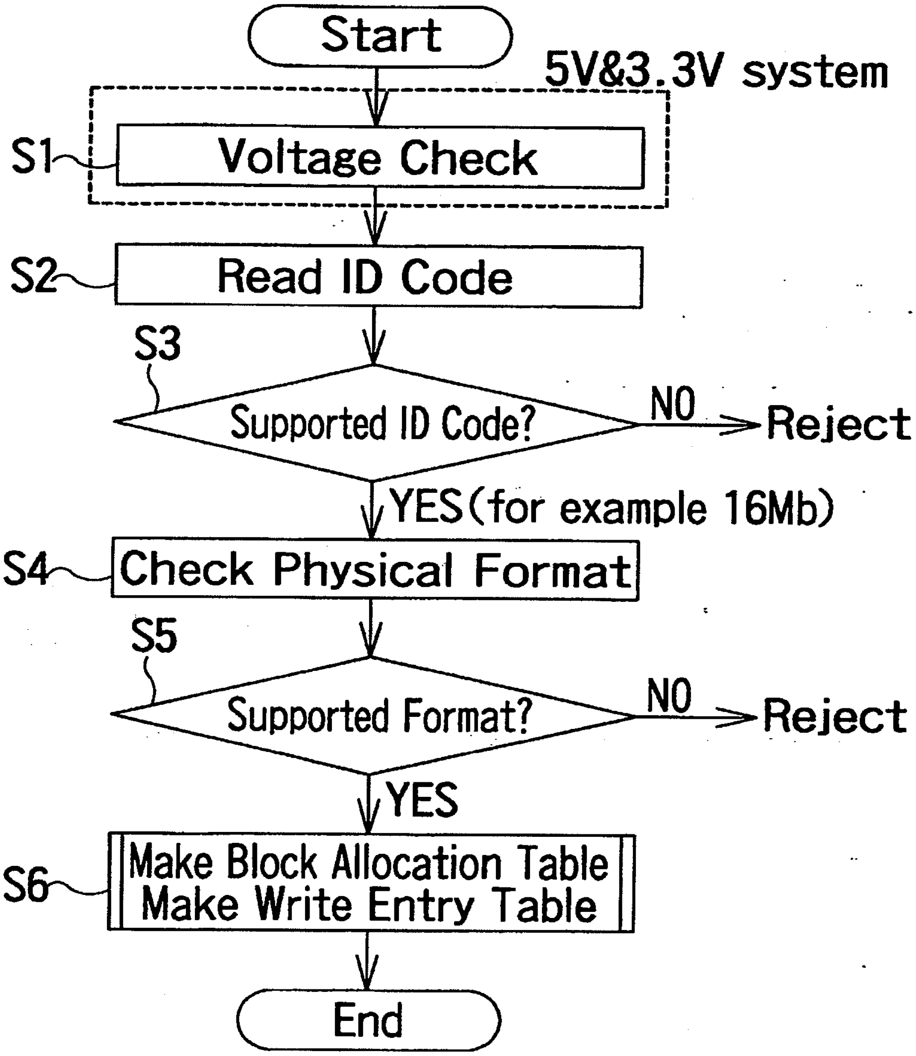

Start
5V&3.3V system
S1
Voltage Check
S2
Read ID Code
S3
Supported ID Code?
NO
Reject
YES(for example 16Mb)
S4
Check Physical Format
S5
Supported Format?
NO
Reject
YES
S6
Make Block Allocation Table
Make Write Entry Table
End

FIG.32

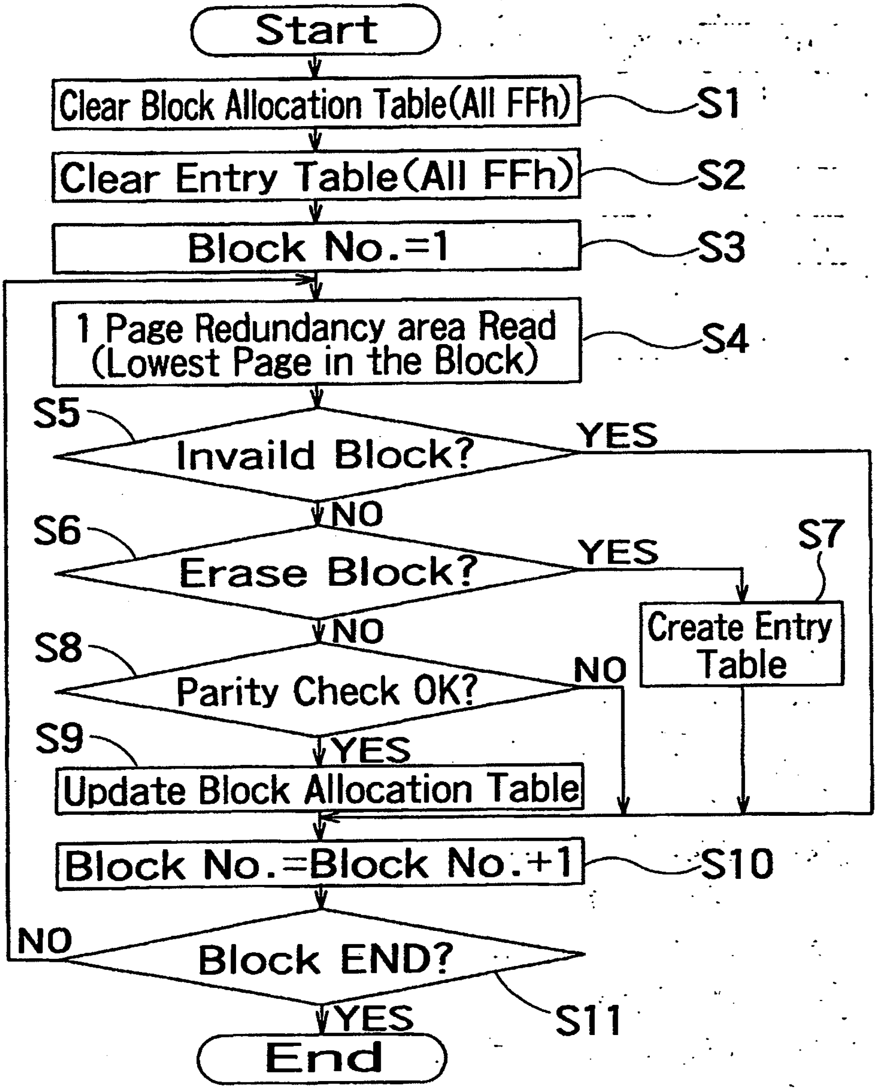

Start
Clear Block Allocation Table(All FFh)
S1
Clear Entry Table(All FFh)
S2
Block No.=1
S3
1 Page Redundancy area Read (Lowest Page in the Block)
S4
S5
Invaild Block?
YES
NO
S6
Erase Block?
YES
S7
Create Entry Table
NO
S8
Parity Check OK?
NO
YES
S9
Update Block Allocation Table
Block No.=Block No.+1
S10
NO
Block END?
S11
YES
End

FIG.33

OFFSET (LOGICAL BLOCK ADDRESS)	PHYSICAL BLOCK AREA ADDRESS	PHYSICAL BLOCK AREA ADDRESS (BINARY DATA)	
Word0(LBA=0)	0	0000	0000
Word1(LBA=1)	250	1111	1010
Word2(LBA=2)	163	1010	0011
⋮	⋮	⋮	⋮
Word497(LBA=497)	122	0111	1010
Word498(LBA=498)	248	1010	1000
Word499(LBA=499)	64	0100	0000

1 PHYSICAL BLOCK AREA=2 PHYSICAL BLOCK

FIG.34

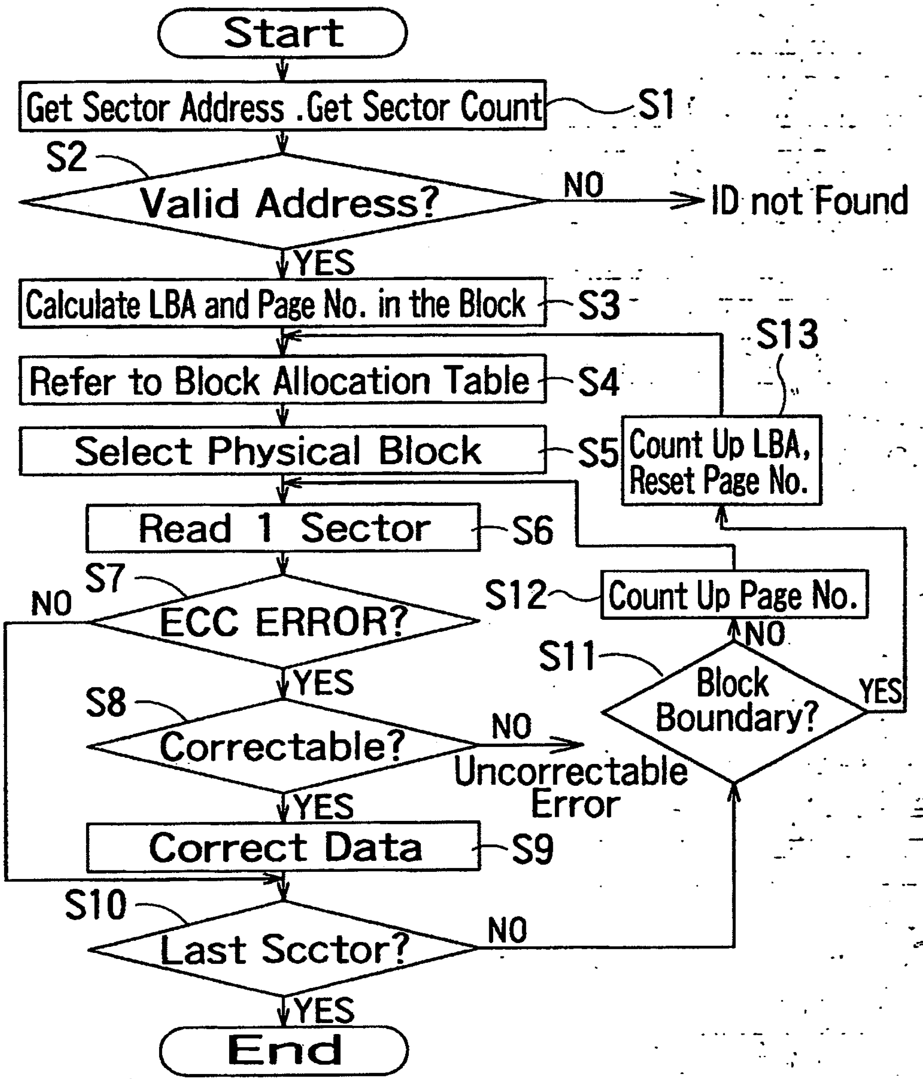

Start
Get Sector Address .Get Sector Count
S1
S2
Valid Address?
NO
ID not Found
YES
Calculate LBA and Page No. in the Block
S3
S13
Refer to Block Allocation Table
S4
Select Physical Block
S5
Count Up LBA, Reset Page No.
Read 1 Sector
S6
S7
NO
ECC ERROR?
S12
Count Up Page No.
NO
S11
Block Boundary?
YES
YES
S8
Correctable?
NO
Uncorrectable Error
YES
Correct Data
S9
S10
Last Scctor?
NO
YES
End

FIG.35

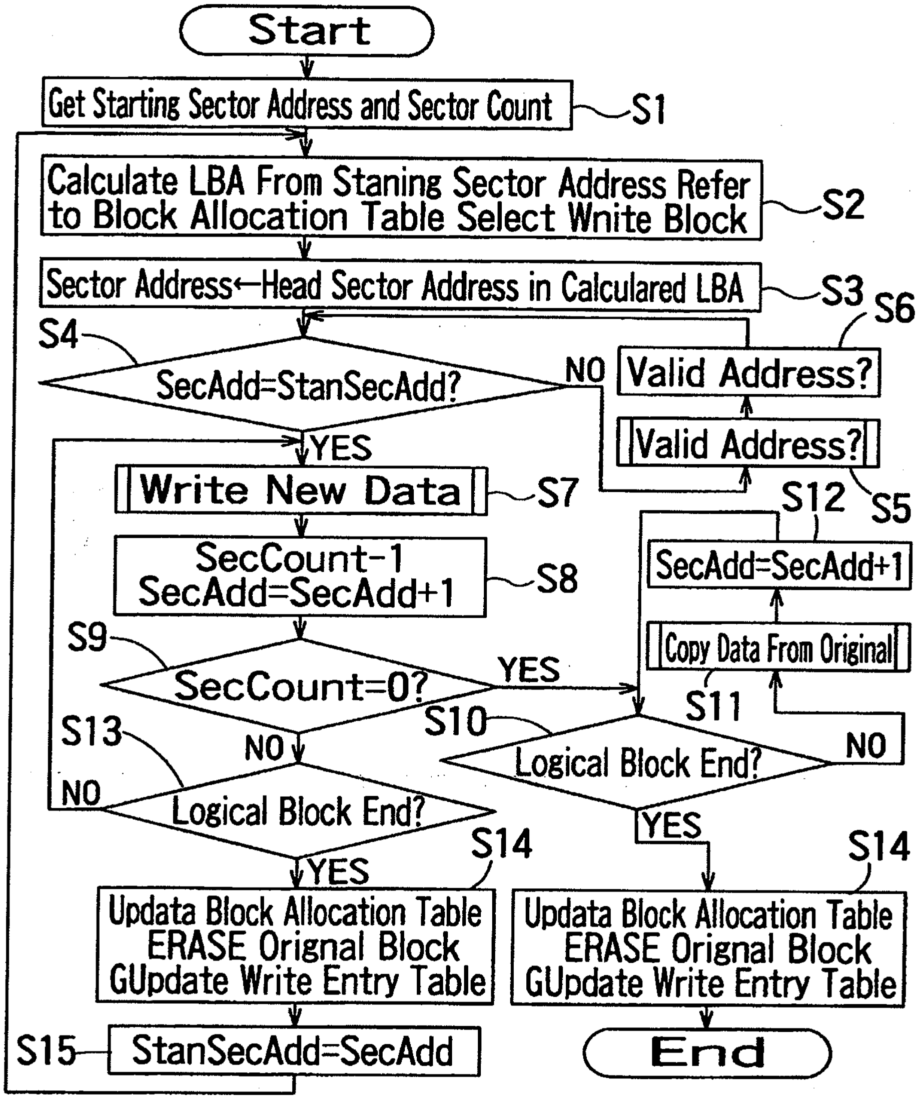
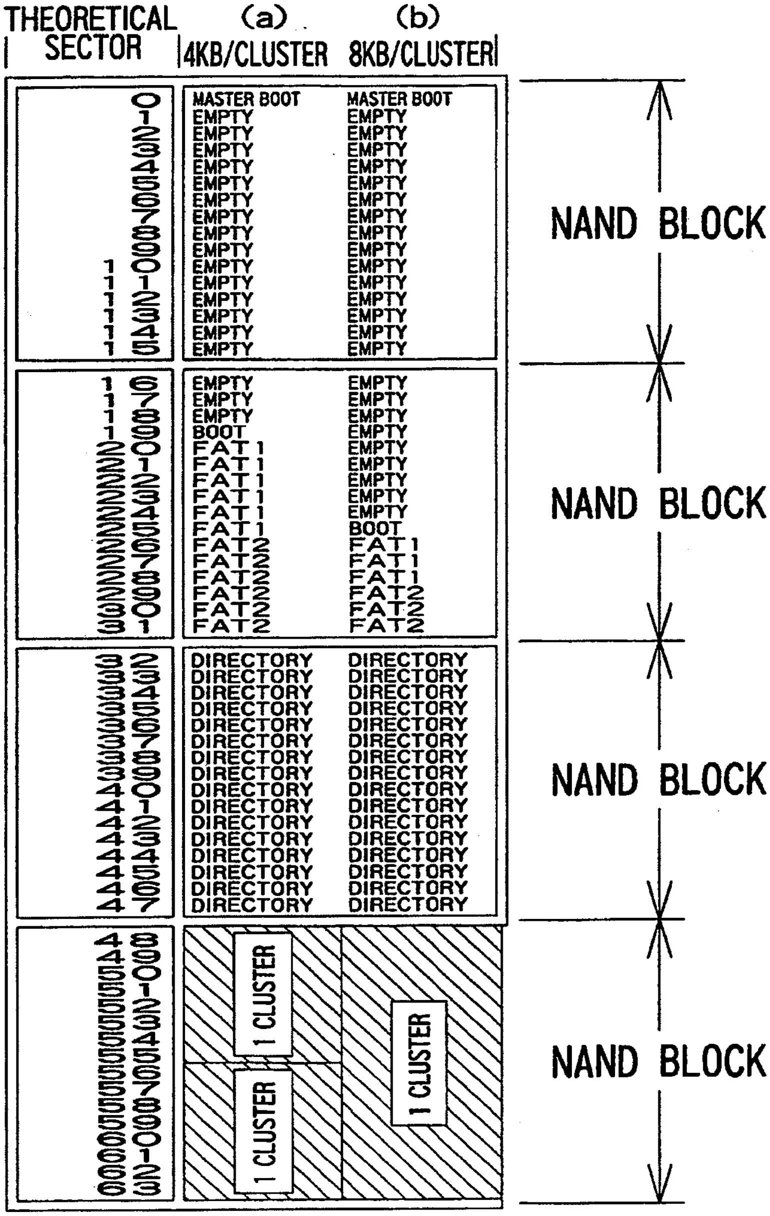

Start
Get Starting Sector Address and Sector Count
S1
Calculate LBA From Staning Sector Address Refer to Block Allocation Table Select Wnite Block
S2
Sector Address←Head Sector Address in Calculared LBA
S3
S6
S4
SecAdd=StanSecAdd?
NO
Valid Address?
Valid Address?
YES
Write New Data
S7
S12
S5
SecCount-1
SecAdd=SecAdd+1
S8
SecAdd=SecAdd+1
S9
Copy Data From Original
SecCount=0?
YES
S11
S10
S13
NO
Logical Block End?
NO
NO
Logical Block End?
YES
S14
YES
S14
Updata Block Allocation Table
ERASE Orignal Block
GUpdate Write Entry Table
Updata Block Allocation Table
ERASE Orignal Block
GUpdate Write Entry Table
S15
StanSecAdd=SecAdd
End

FIG.36



THEORETICAL SECTOR	(a) 4KB/CLUSTER	(b) 8KB/CLUSTER
0	MASTER BOOT	MASTER BOOT
1	EMPTY	EMPTY
2	EMPTY	EMPTY
3	EMPTY	EMPTY
4	EMPTY	EMPTY
5	EMPTY	EMPTY
6	EMPTY	EMPTY
7	EMPTY	EMPTY
8	EMPTY	EMPTY
9	EMPTY	EMPTY
10	EMPTY	EMPTY
11	EMPTY	EMPTY
12	EMPTY	EMPTY
13	EMPTY	EMPTY
14	EMPTY	EMPTY
15	EMPTY	EMPTY
16	EMPTY	EMPTY
17	EMPTY	EMPTY
18	EMPTY	EMPTY
19	BOOT	EMPTY
20	FAT 1	EMPTY
21	FAT 1	EMPTY
22	FAT 1	EMPTY
23	FAT 1	EMPTY
24	FAT 1	EMPTY
25	FAT 1	BOOT
26	FAT2	FAT 1
27	FAT2	FAT 1
28	FAT2	FAT 1
29	FAT2	FAT2
30	FAT2	FAT2
31	FAT2	FAT2
32	DIRECTORY	DIRECTORY
33	DIRECTORY	DIRECTORY
34	DIRECTORY	DIRECTORY
35	DIRECTORY	DIRECTORY
36	DIRECTORY	DIRECTORY
37	DIRECTORY	DIRECTORY
38	DIRECTORY	DIRECTORY
39	DIRECTORY	DIRECTORY
40	DIRECTORY	DIRECTORY
41	DIRECTORY	DIRECTORY
42	DIRECTORY	DIRECTORY
43	DIRECTORY	DIRECTORY
44	DIRECTORY	DIRECTORY
45	DIRECTORY	DIRECTORY
46	DIRECTORY	DIRECTORY
47	DIRECTORY	DIRECTORY
48	1 CLUSTER	1 CLUSTER
49		
50		
51		
52		
53		
54		
55		
56	1 CLUSTER	
57		
58		
59		
60		
61		
62		
63		

FIG.37

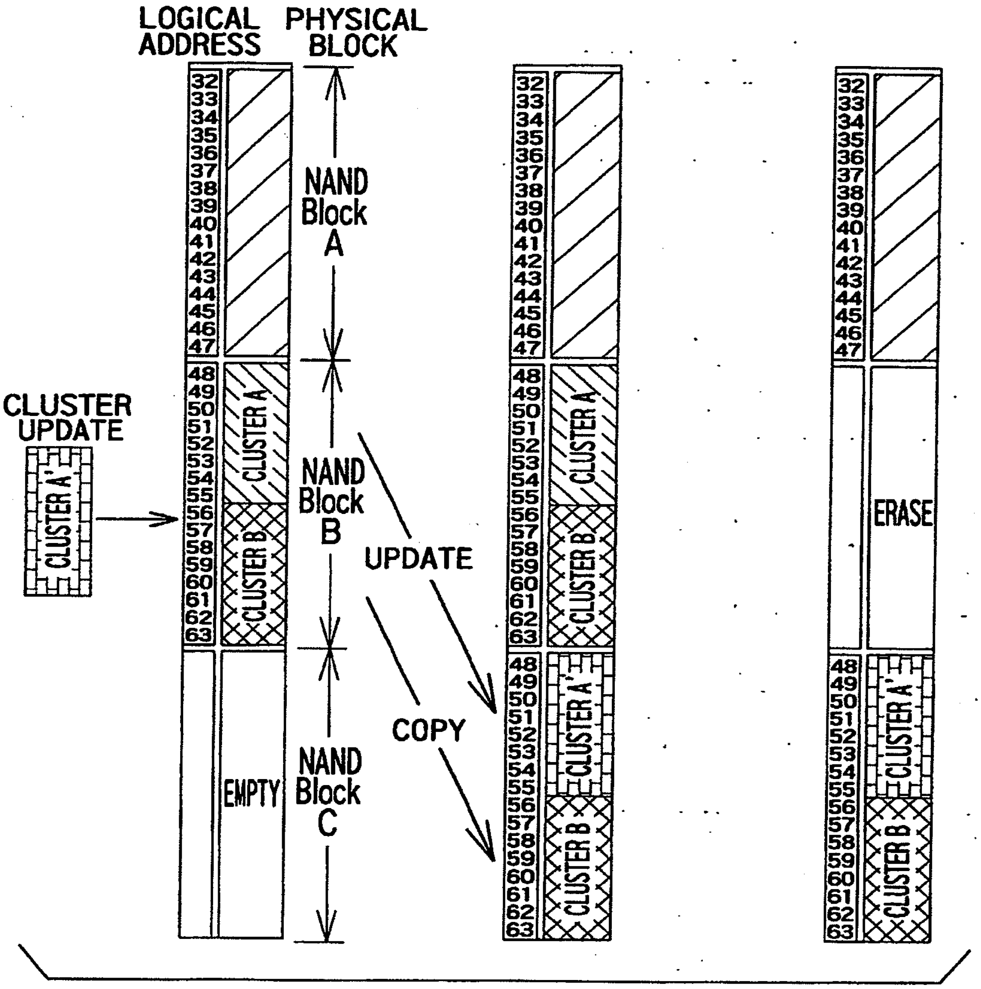

LOGICAL ADDRESS
PHYSICAL BLOCK
NAND Block A
NAND Block B
NAND Block C
CLUSTER A
CLUSTER B
EMPTY
CLUSTER UPDATE
CLUSTER A'
UPDATE
COPY
ERASE

FIG. 38

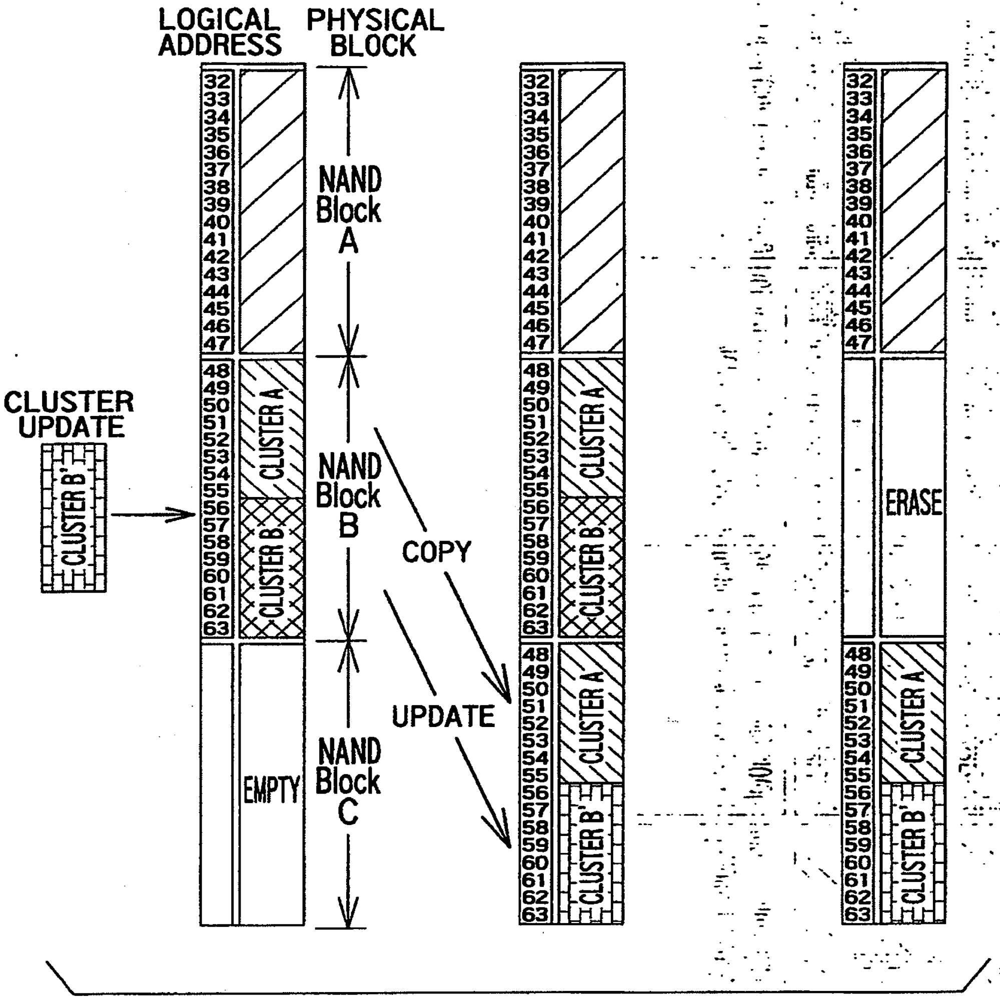

LOGICAL ADDRESS
PHYSICAL BLOCK
CLUSTER UPDATE
CLUSTER B'
NAND Block A
NAND Block B
NAND Block C
CLUSTER A
CLUSTER B
EMPTY
COPY
UPDATE
ERASE

FIG. 39

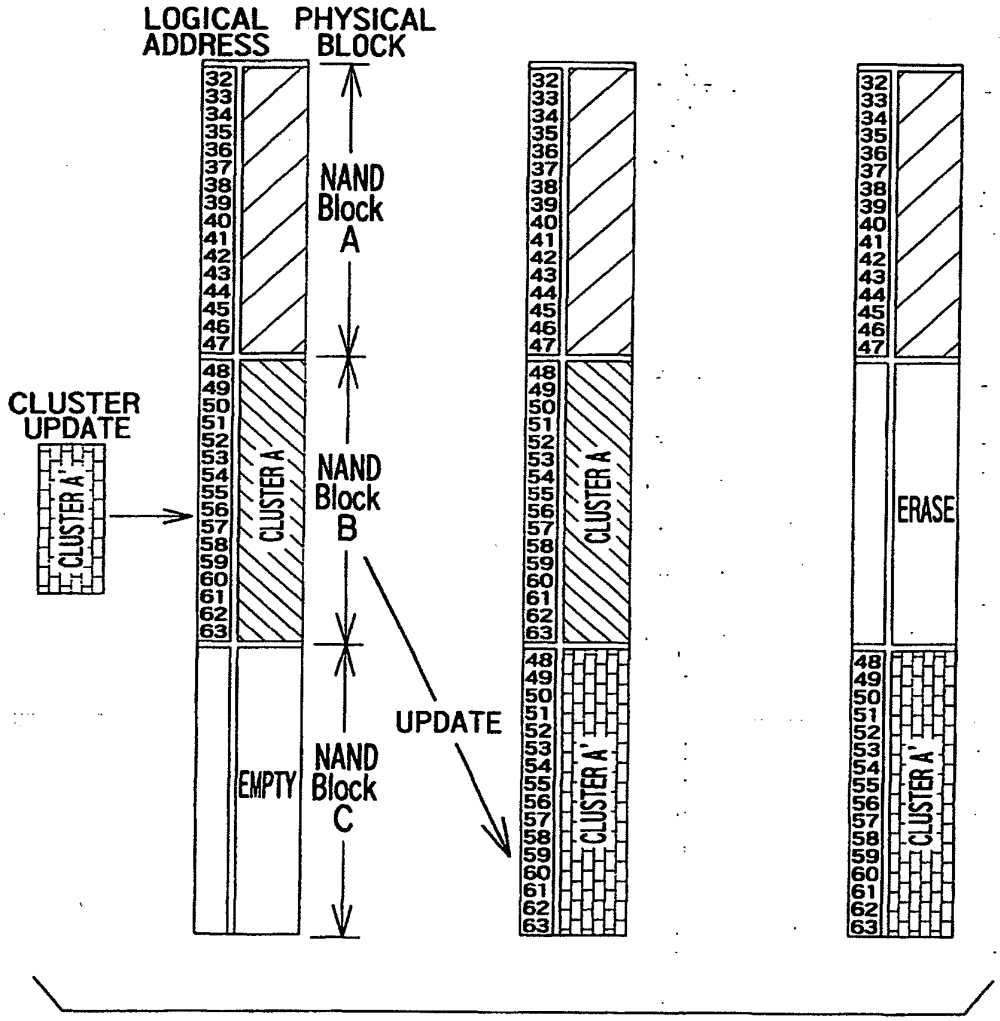

LOGICAL ADDRESS
PHYSICAL BLOCK
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
NAND Block A
NAND Block B
NAND Block C
CLUSTER UPDATE
CLUSTER A'
CLUSTER A
EMPTY
UPDATE
ERASE

FIG.40

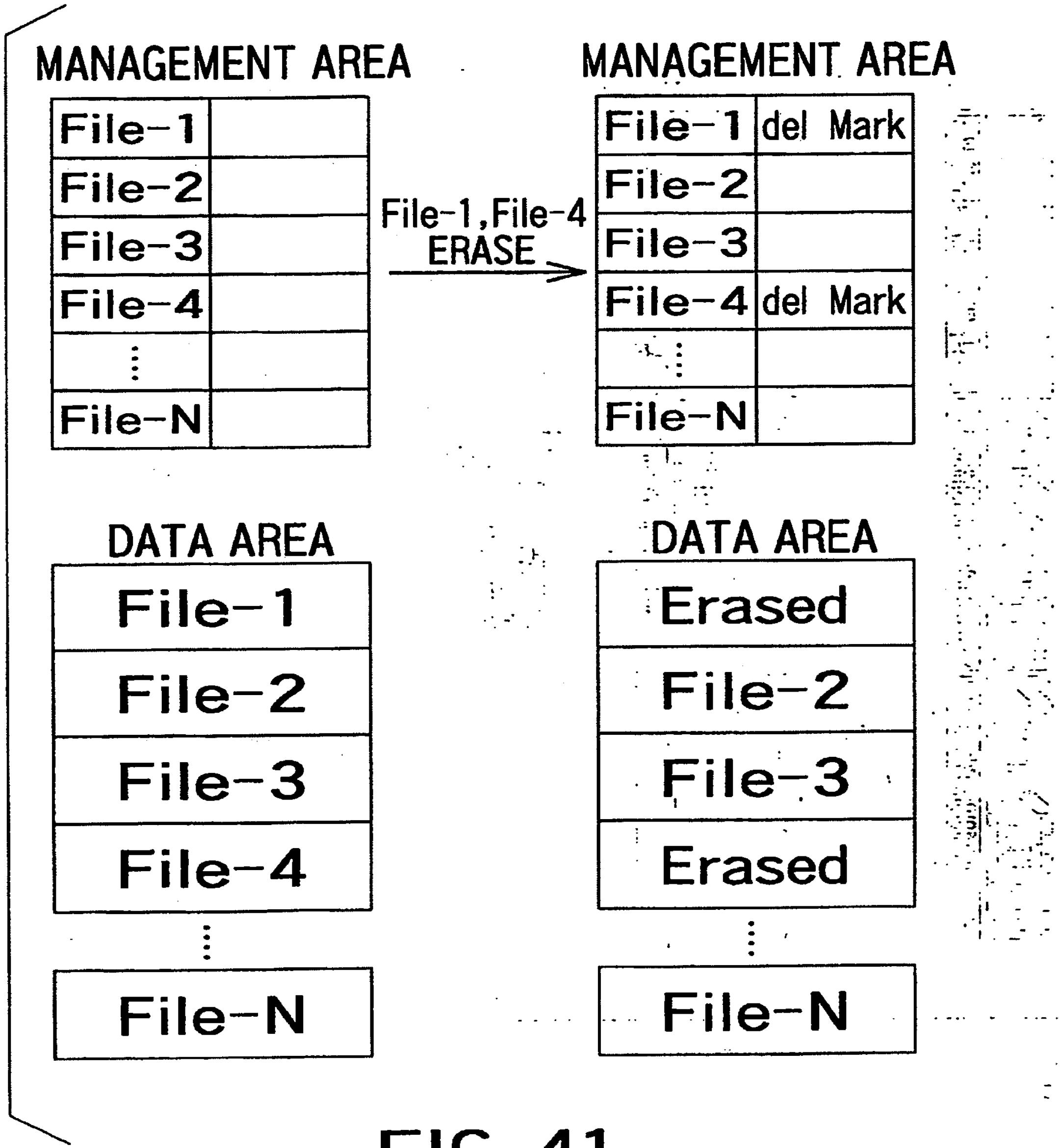

MANAGEMENT AREA
File-1
File-2
File-3
File-4
File-N
File-1,File-4
ERASE
MANAGEMENT AREA
File-1 del Mark
File-2
File-3
File-4 del Mark
File-N
DATA AREA
File-1
File-2
File-3
File-4
File-N
DATA AREA
Erased
File-2
File-3
Erased
File-N

FIG.41

OFFSET (LOGICAL BLOCK ADDRESS)	(PHYSICAL BLOCK ADDRESS)	
	Upper Byte	Lower Byte
Word0(LBA=0)	Physical Block Upper Address	Physical Block Lower Address
Word1(LBA=1)	Physical Block Upper Address	Physical Block Lower Address
Word2(LBA=2)	Physical Block Upper Address	Physical Block Lower Address
⋮		
Word247(LBA=247)	Physical Block Upper Address	Physical Block Lower Address
Word248(LBA=248)	Physical Block Upper Address	Physical Block Lower Address
Word249(LBA=249)	Physical Block Upper Address	Physical Block Lower Address

FIG.42(a)

OFFSET (LOGICAL BLOCK ADDRESS)	(PHYSICAL BLOCK ADDRESS)	
	Upper Byte	Lower Byte
Word0(LBA=250)	Physical Block Upper Address	Physical Block Lower Address
Word1(LBA=251)	Physical Block Upper Address	Physical Block Lower Address
Word2(LBA=252)	Physical Block Upper Address	Physical Block Lower Address
⋮		
Word247(LBA=497)	Physical Block Upper Address	Physical Block Lower Address
Word248(LBA=498)	Physical Block Upper Address	Physical Block Lower Address
Word249(LBA=499)	Physical Block Upper Address	Physical Block Lower Address

FIG.42(b)

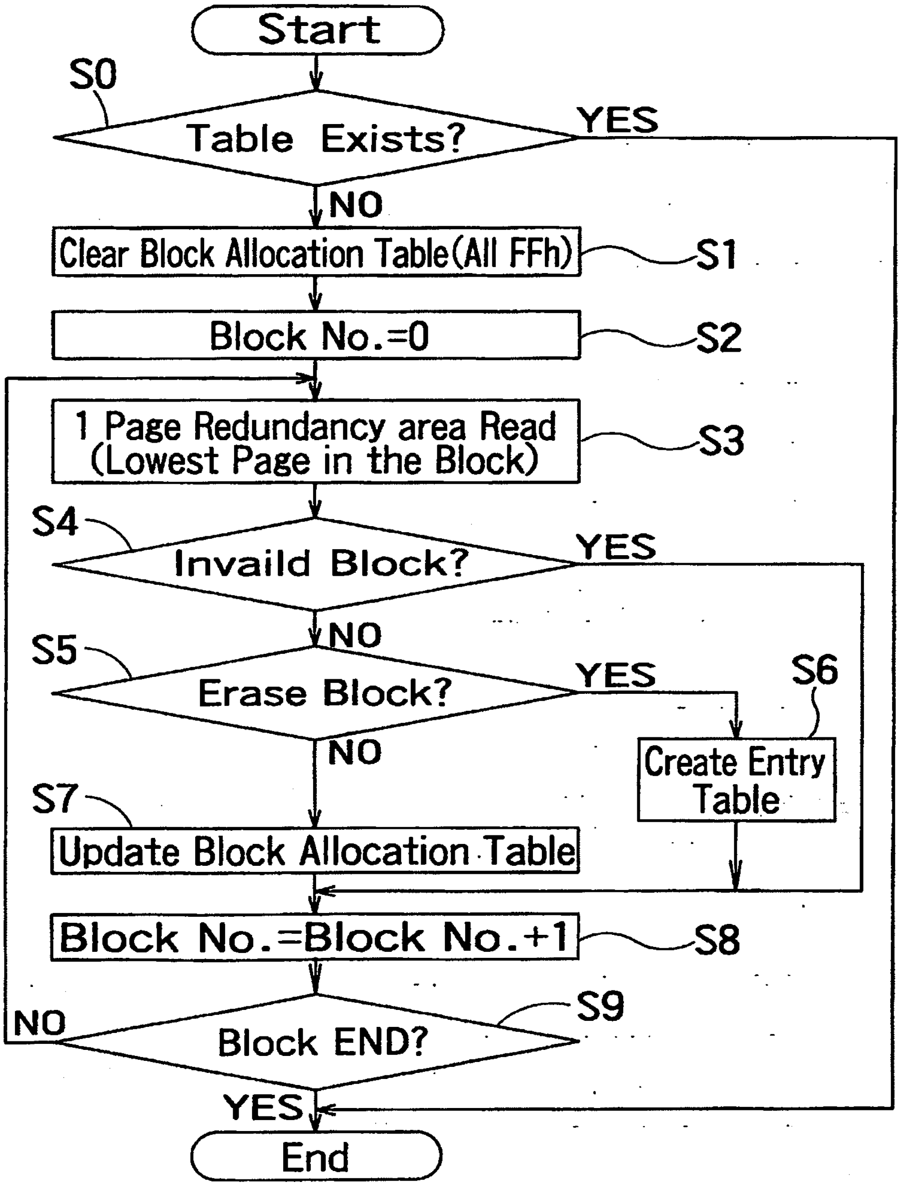
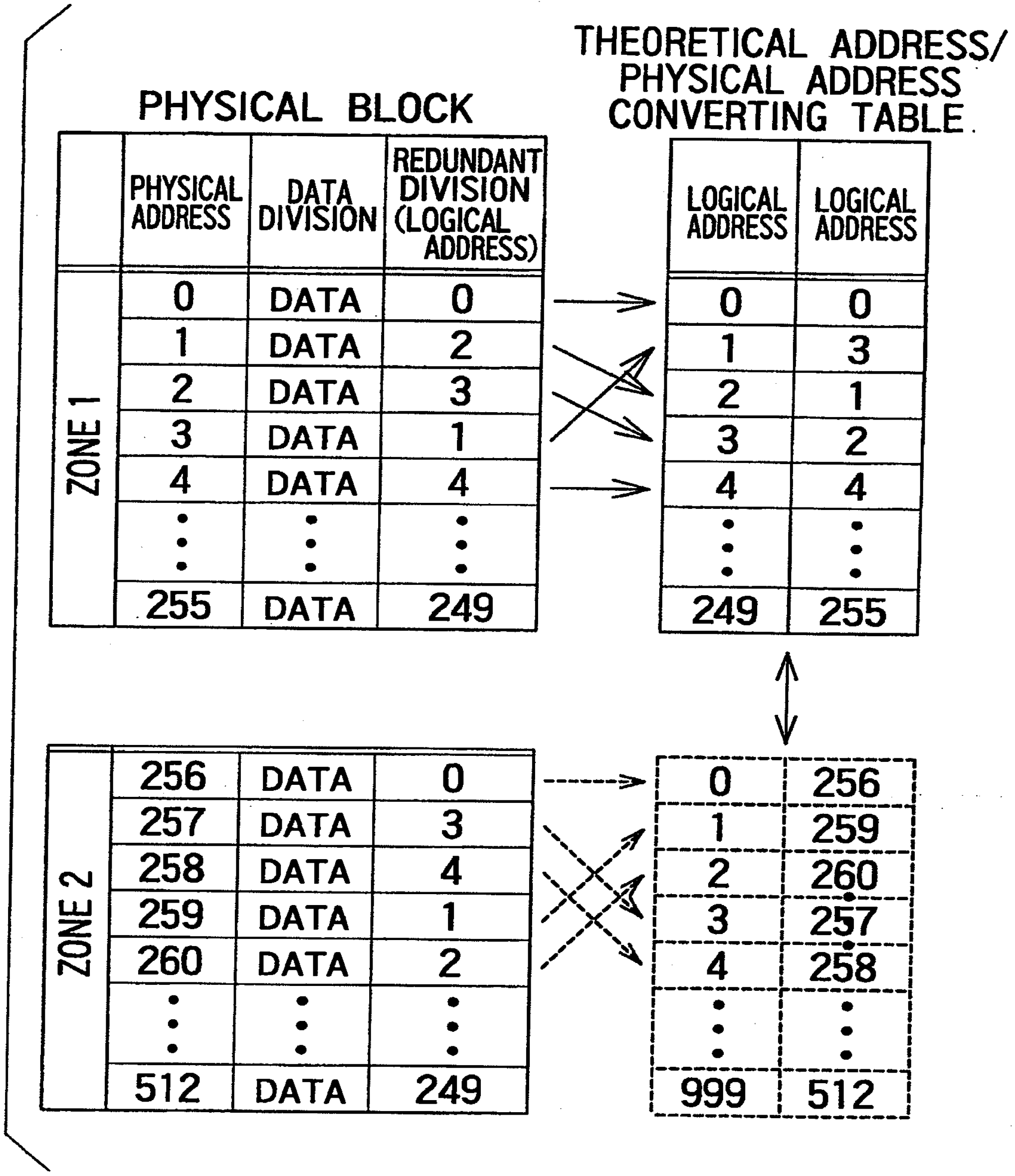

Start
S0
Table Exists?
YES
NO
Clear Block Allocation Table(All FFh)
S1
Block No.=0
S2
1 Page Redundancy area Read
(Lowest Page in the Block)
S3
S4
Invaild Block?
YES
NO
S5
Erase Block?
YES
S6
Create Entry Table
NO
S7
Update Block Allocation Table
Block No.=Block No.+1
S8
S9
NO
Block END?
YES
End

FIG.43



PHYSICAL BLOCK

	PHYSICAL ADDRESS	DATA DIVISION	REDUNDANT DIVISION (LOGICAL ADDRESS)
ZONE 1	0	DATA	0
	1	DATA	2
	2	DATA	3
	3	DATA	1
	4	DATA	4
	⋮	⋮	⋮
	255	DATA	249

ZONE 2	256	DATA	0
	257	DATA	3
	258	DATA	4
	259	DATA	1
	260	DATA	2
	⋮	⋮	⋮
	512	DATA	249

THEORETICAL ADDRESS/ PHYSICAL ADDRESS CONVERTING TABLE.

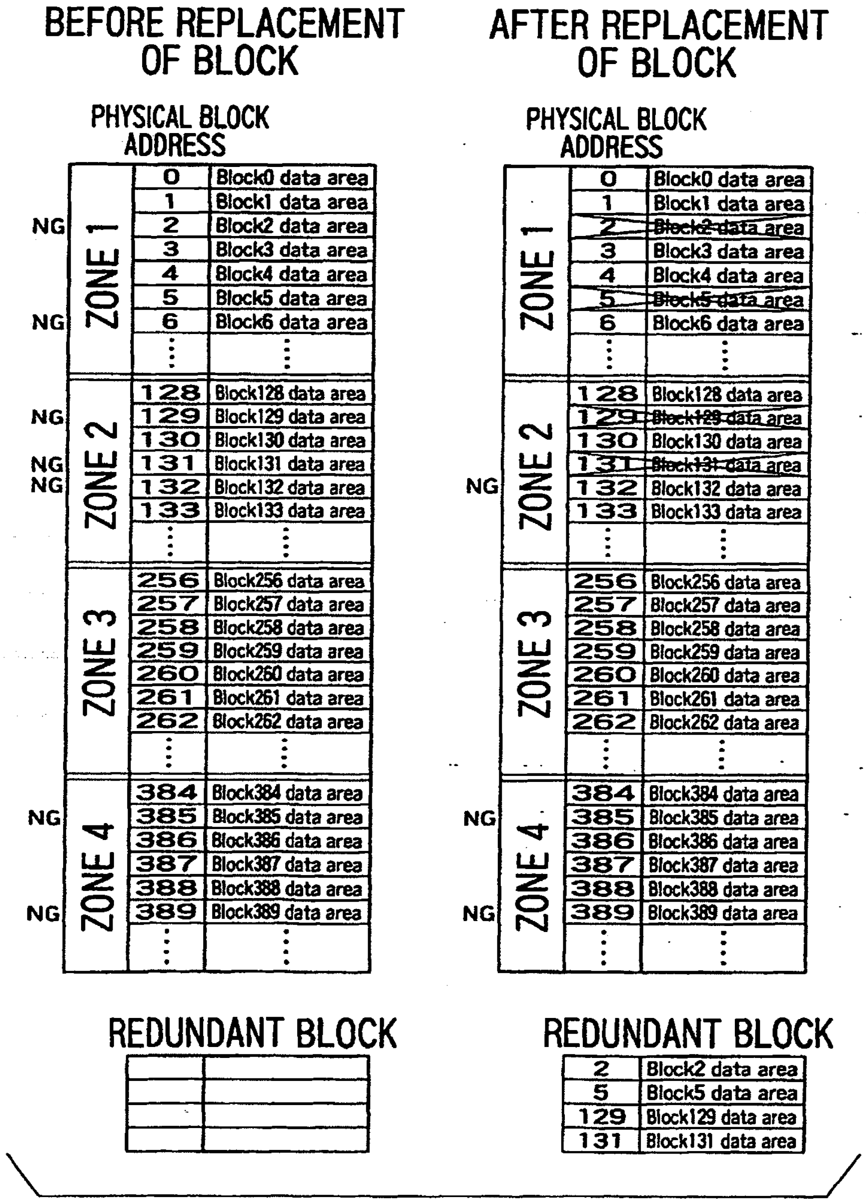
LOGICAL ADDRESS	LOGICAL ADDRESS
0	0
1	3
2	1
3	2
4	4
⋮	⋮
249	255

0	256
1	259
2	260
3	257
4	258
⋮	⋮
999	512

FIG. 44

	OFFSET (LOGICAL BLOCK ADDRESS)	PHYSICAL BLOCK ADDRESS	PHYSICAL BLOCK ADDRESS (BINARY DATA)		
ZONE 1	Word0 (LBA=0)	0	0000	0000	0000
	Word2 (LBA=2)	227	0000	1110	0011
	⋮	⋮	⋮	⋮	⋮
	Word254 (LBA=254)	244	0000	1111	0100
	Word255 (LBA=255)	128	0000	1000	0111
ZONE 2	Word256 (LBA=256)	256(256-256=0)	0000	0000	0000
	Word257 (LBA=257)	327(327-256=71)	0000	0100	0111
	⋮	⋮	⋮	⋮	⋮
	Word499 (LBA=499)	500(500-256=244)	0000	1110	0000
	Word500 (LBA=500)	428(428-256=172)	0000	1010	1100

FIG. 45



BEFORE REPLACEMENT OF BLOCK

		PHYSICAL BLOCK ADDRESS	
	ZONE 1	0	Block0 data area
		1	Block1 data area
NG		2	Block2 data area
		3	Block3 data area
		4	Block4 data area
		5	Block5 data area
NG		6	Block6 data area
		⋮	⋮
	ZONE 2	128	Block128 data area
NG		129	Block129 data area
		130	Block130 data area
NG		131	Block131 data area
NG		132	Block132 data area
		133	Block133 data area
		⋮	⋮
	ZONE 3	256	Block256 data area
		257	Block257 data area
		258	Block258 data area
		259	Block259 data area
		260	Block260 data area
		261	Block261 data area
		262	Block262 data area
		⋮	⋮
	ZONE 4	384	Block384 data area
NG		385	Block385 data area
		386	Block386 data area
		387	Block387 data area
		388	Block388 data area
NG		389	Block389 data area
		⋮	⋮

REDUNDANT BLOCK

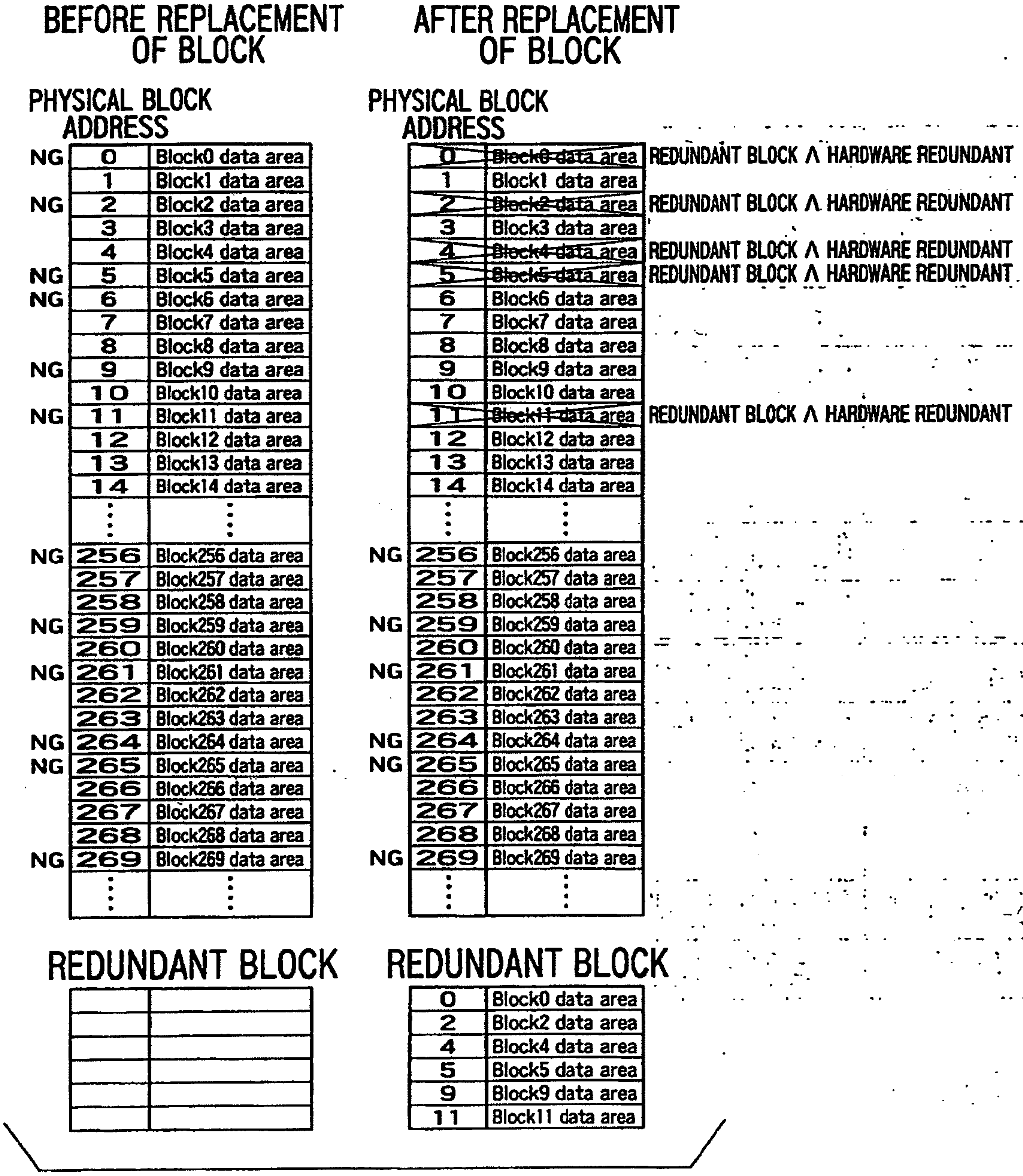
AFTER REPLACEMENT OF BLOCK

		PHYSICAL BLOCK ADDRESS	
	ZONE 1	0	Block0 data area
		1	Block1 data area
		~~2~~	~~Block2 data area~~
		3	Block3 data area
		4	Block4 data area
		~~5~~	~~Block5 data area~~
		6	Block6 data area
		⋮	⋮
	ZONE 2	128	Block128 data area
		~~129~~	~~Block129 data area~~
		130	Block130 data area
		~~131~~	~~Block131 data area~~
NG		132	Block132 data area
		133	Block133 data area
		⋮	⋮
	ZONE 3	256	Block256 data area
		257	Block257 data area
		258	Block258 data area
		259	Block259 data area
		260	Block260 data area
		261	Block261 data area
		262	Block262 data area
		⋮	⋮
	ZONE 4	384	Block384 data area
NG		385	Block385 data area
		386	Block386 data area
		387	Block387 data area
		388	Block388 data area
NG		389	Block389 data area
		⋮	⋮

REDUNDANT BLOCK

2	Block2 data area
5	Block5 data area
129	Block129 data area
131	Block131 data area

FIG.46



BEFORE REPLACEMENT OF BLOCK

PHYSICAL BLOCK ADDRESS

NG	0	Block0 data area
	1	Block1 data area
NG	2	Block2 data area
	3	Block3 data area
	4	Block4 data area
NG	5	Block5 data area
NG	6	Block6 data area
	7	Block7 data area
	8	Block8 data area
NG	9	Block9 data area
	10	Block10 data area
NG	11	Block11 data area
	12	Block12 data area
	13	Block13 data area
	14	Block14 data area
	⋮	⋮
NG	256	Block256 data area
	257	Block257 data area
	258	Block258 data area
NG	259	Block259 data area
	260	Block260 data area
NG	261	Block261 data area
	262	Block262 data area
	263	Block263 data area
NG	264	Block264 data area
NG	265	Block265 data area
	266	Block266 data area
	267	Block267 data area
	268	Block268 data area
NG	269	Block269 data area
	⋮	⋮

REDUNDANT BLOCK

AFTER REPLACEMENT OF BLOCK

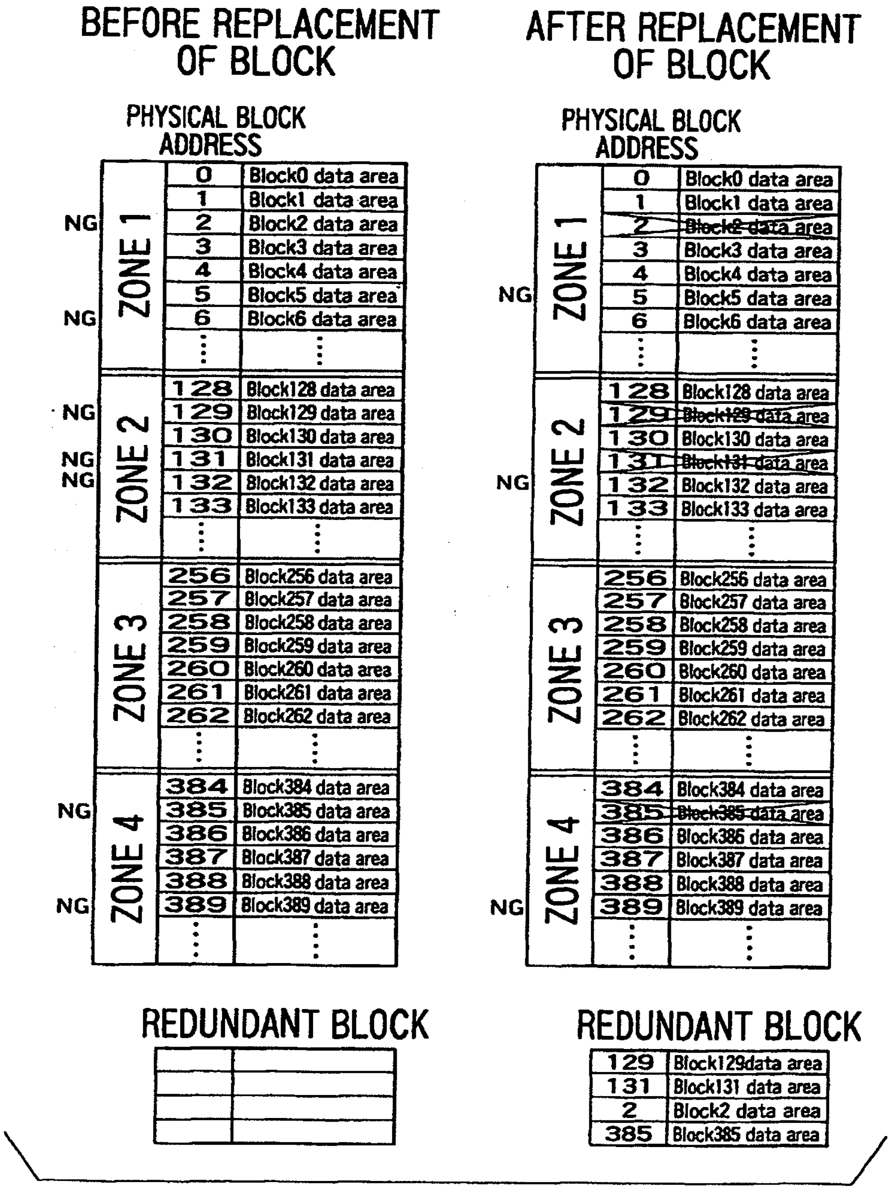
PHYSICAL BLOCK ADDRESS

	~~0~~	~~Block0 data area~~	REDUNDANT BLOCK Λ HARDWARE REDUNDANT
	1	Block1 data area	
	~~2~~	~~Block2 data area~~	REDUNDANT BLOCK Λ HARDWARE REDUNDANT
	3	Block3 data area	
	~~4~~	~~Block4 data area~~	REDUNDANT BLOCK Λ HARDWARE REDUNDANT
	~~5~~	~~Block5 data area~~	REDUNDANT BLOCK Λ HARDWARE REDUNDANT
	6	Block6 data area	
	7	Block7 data area	
	8	Block8 data area	
	9	Block9 data area	
	10	Block10 data area	
	~~11~~	~~Block11 data area~~	REDUNDANT BLOCK Λ HARDWARE REDUNDANT
	12	Block12 data area	
	13	Block13 data area	
	14	Block14 data area	
	⋮	⋮	
NG	256	Block256 data area	
	257	Block257 data area	
	258	Block258 data area	
NG	259	Block259 data area	
	260	Block260 data area	
NG	261	Block261 data area	
	262	Block262 data area	
	263	Block263 data area	
NG	264	Block264 data area	
NG	265	Block265 data area	
	266	Block266 data area	
	267	Block267 data area	
	268	Block268 data area	
NG	269	Block269 data area	
	⋮	⋮	

REDUNDANT BLOCK

0	Block0 data area
2	Block2 data area
4	Block4 data area
5	Block5 data area
9	Block9 data area
11	Block11 data area

FIG.47



BEFORE REPLACEMENT OF BLOCK

		PHYSICAL BLOCK ADDRESS	
	ZONE 1	0	Block0 data area
		1	Block1 data area
NG		2	Block2 data area
		3	Block3 data area
		4	Block4 data area
		5	Block5 data area
NG		6	Block6 data area
		⋮	⋮
	ZONE 2	128	Block128 data area
NG		129	Block129 data area
		130	Block130 data area
NG		131	Block131 data area
NG		132	Block132 data area
		133	Block133 data area
		⋮	⋮
	ZONE 3	256	Block256 data area
		257	Block257 data area
		258	Block258 data area
		259	Block259 data area
		260	Block260 data area
		261	Block261 data area
		262	Block262 data area
		⋮	⋮
	ZONE 4	384	Block384 data area
NG		385	Block385 data area
		386	Block386 data area
		387	Block387 data area
		388	Block388 data area
NG		389	Block389 data area
		⋮	⋮

REDUNDANT BLOCK

AFTER REPLACEMENT OF BLOCK

		PHYSICAL BLOCK ADDRESS	
	ZONE 1	0	Block0 data area
		1	Block1 data area
		~~2~~	~~Block2 data area~~
		3	Block3 data area
		4	Block4 data area
NG		5	Block5 data area
		6	Block6 data area
		⋮	⋮
	ZONE 2	128	Block128 data area
		~~129~~	~~Block129 data area~~
		130	Block130 data area
		~~131~~	~~Block131 data area~~
NG		132	Block132 data area
		133	Block133 data area
		⋮	⋮
	ZONE 3	256	Block256 data area
		257	Block257 data area
		258	Block258 data area
		259	Block259 data area
		260	Block260 data area
		261	Block261 data area
		262	Block262 data area
		⋮	⋮
	ZONE 4	384	Block384 data area
		~~385~~	~~Block385 data area~~
		386	Block386 data area
		387	Block387 data area
		388	Block388 data area
NG		389	Block389 data area
		⋮	⋮

REDUNDANT BLOCK

129	Block129data area
131	Block131 data area
2	Block2 data area
385	Block385 data area

FIG. 48

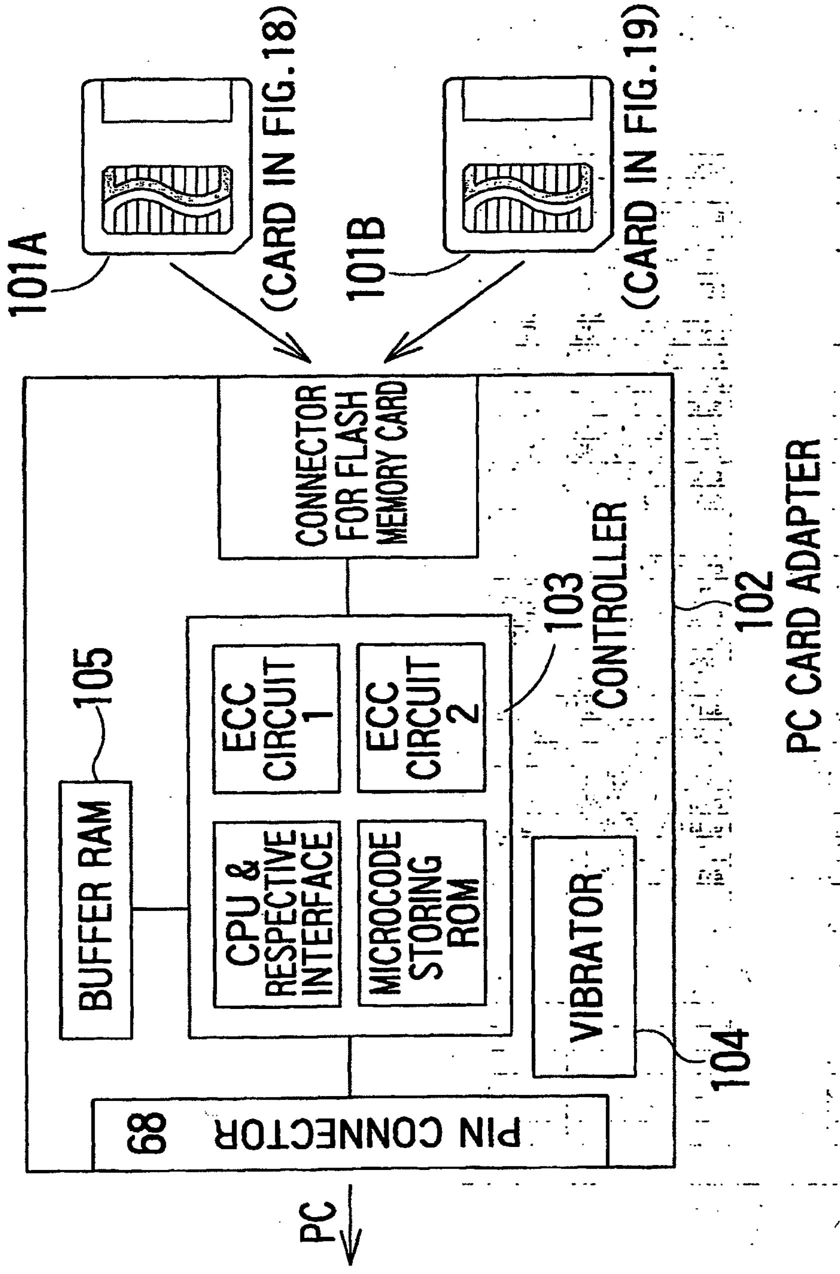

101A
(CARD IN FIG.18)
101B
(CARD IN FIG.19)
CONNECTOR FOR FLASH MEMORY CARD
BUFFER RAM
105
ECC CIRCUIT 1
ECC CIRCUIT 2
CPU & RESPECTIVE INTERFACE
MICROCODE STORING ROM
103
CONTROLLER
102
PC CARD ADAPTER
VIBRATOR
104
68 PIN CONNECTOR
PC

FIG. 49

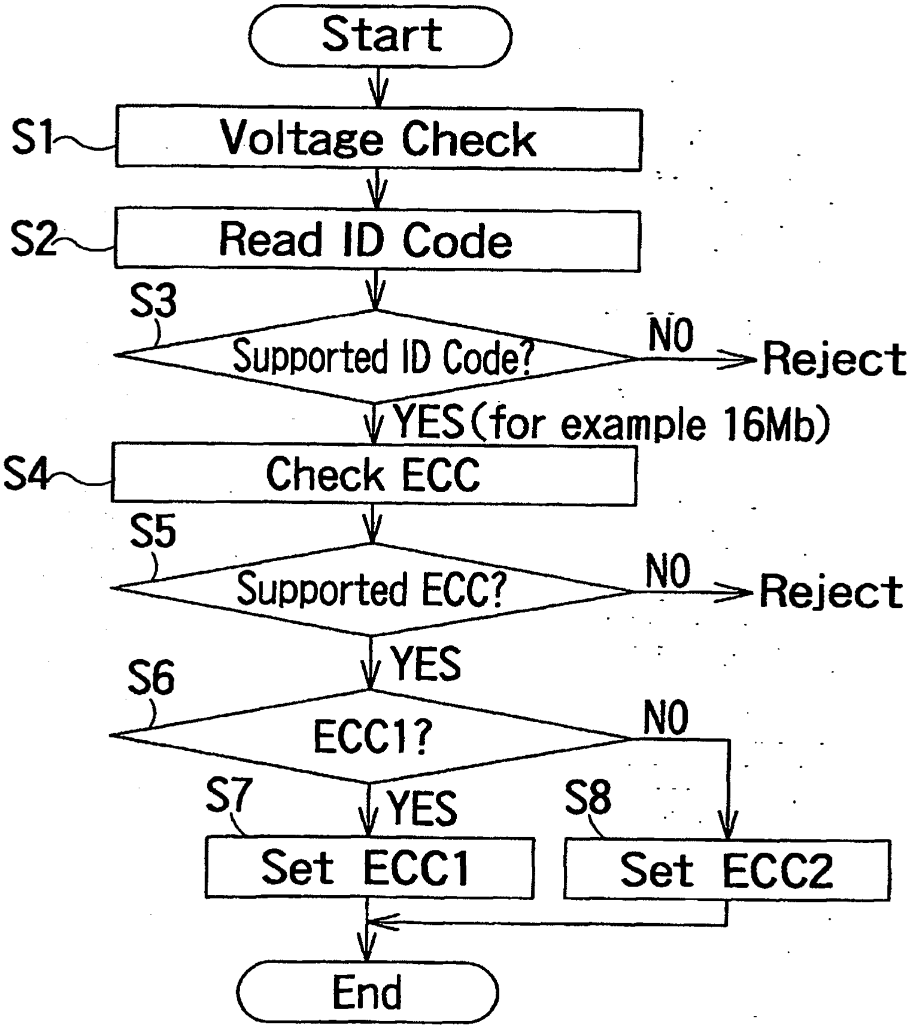

Start
S1
Voltage Check
S2
Read ID Code
S3
Supported ID Code?
NO
Reject
YES(for example 16Mb)
S4
Check ECC
S5
Supported ECC?
NO
Reject
YES
S6
ECC1?
NO
S7
YES
S8
Set ECC1
Set ECC2
End

FIG.50

DATA DIVISION

BYTE	PAGE 0(EVEN PAGE)	PAGE 1(ODD PAGE)
0~255	DATA Area-1	DATA Area-2

REDUNDANT DIVISION

BYTE	EVEN PAGE	ODD PAGE
256	ECC Flag Area	ECC Area-2
257	ECC Area-3	
258		
259		Block Address Area-2
260	Data Status Area	
261	Block Status Area	ECC Area-1
262	Block Address Area-1	
263		

FIG.51

	ECC-AREA1	ECC-AREA2	ECC-AREA3	ECC-AREA4
ECC METHOD 1	ECC CODE FOR DATA AREA-1	ECC CODE FOR DATA AREA-2	NULL (ALL"FFh")	ECC1-FLAG
ECC METHOD 2	ECC CODE FOR DATA AREA-1,2	ECC CODE FOR DATA AREA-1,2	ECC CODE FOR DATA AREA-1,2	ECC2-FLAG

FIG.52

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-091490
(43)Date of publication of application : 10.04.1998

---

(51)Int.Cl. G06F 12/00
G11C 16/06

---

(21)Application number : 08-242124
(22)Date of filing : 12.09.1996

(71)Applicant : SANYO ELECTRIC CO LTD
TOTTORI SANYO ELECTRIC CO LTD
(72)Inventor : KANBARA HARUAKI
KAHARA SHIGEMI
YAMASHITA TAKAHIRO
AMISAKI MASAYA
UMEZAWA ISAO
URAKAWA FUMITAKA
MAEDA KOJI

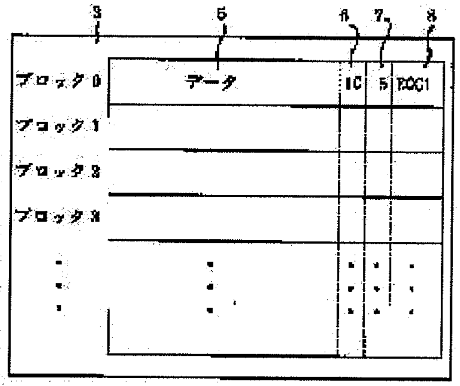
---

(54) STORAGE DEVICE USING FLASH MEMORY

(57)Abstract:

PROBLEM TO BE SOLVED: To improve data updating speed by forming plural history information areas for storing history information storing a logical address which is permitted to correspond to respective data areas and also is stored in an allocation area corresponding to the respective data areas inside a flash memory.

SOLUTION: The respective blocks of the flash memory 3 are divided into a data area 5 storing 512 byte data, an allocation area 6 storing a two-byte logical address and a history information area 7 storing information which indicates times for storing the same two-byte logical address and remaining 12 bytes are used as the storing area 8 of the ECC data, etc., as shown in terms of a type. Then, a flash memory control part reads and writes data in the flash memory 3 based on address data supplied under the control of a main control part and new data so as to be constituted by exclusive gate array and a one-chip microcomputer. Thus, data is change without requiring removal.



LEGAL STATUS

[Date of request for examination]	03.02.2000
[Date of sending the examiner's decision of rejection]	05.04.2005
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	

[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－91490

(43)公開日　平成10年(1998)４月10日

(51)Int.Cl.^{6}		識別記号	ＦＩ		
Ｇ０６Ｆ	12/00	５１０	Ｇ０６Ｆ	12/00	５１０Ａ
Ｇ１１Ｃ	16/06		Ｇ１１Ｃ	17/00	５１０Ａ
					５３０Ｃ

審査請求　未請求　請求項の数６　ＯＬ　（全 ６ 頁）

(21)出願番号　特願平8－242124

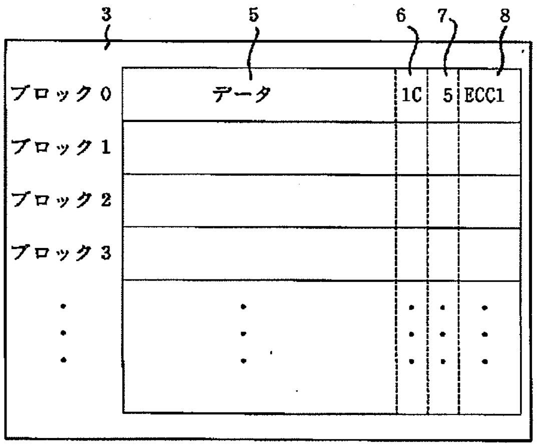
(22)出願日　平成８年(1996) ９月12日

(71)出願人　000001889
三洋電機株式会社
大阪府守口市京阪本通２丁目５番５号
(71)出願人　000214892
鳥取三洋電機株式会社
鳥取県鳥取市南吉方３丁目201番地
(72)発明者　神原　春明
鳥取県鳥取市南吉方３丁目201番地　鳥取三洋電機株式会社内
(72)発明者　花原　成美
鳥取県鳥取市南吉方３丁目201番地　鳥取三洋電機株式会社内
(74)代理人　弁理士　安富　耕二　（外１名）

最終頁に続く

(54)【発明の名称】　フラッシュメモリを利用した記憶装置

(57)【要約】
【課題】　ＮＡＮＤ型のフラッシュメモリにおいてもデータ更新を短時間で行なえるようにする。
【解決手段】　データを格納するためのデータエリア５と、該データエリアの各々に対応付けられ、上記データエリア個々の論理アドレスを格納するアロケーションエリア６と、該エリア６に格納された論理アドレスの格納履歴情報を格納する履歴情報エリア７とをフラッシュメモリ３内に形成する。



【特許請求の範囲】

【請求項１】　データを格納するための複数のデータエリアと、該データエリアの各々に対応付けられ、上記データエリア個々の論理アドレスを格納するための複数のアロケーションエリアと、上記各データエリアに対応付けられ、各データエリアと対応する上記アロケーションエリアに格納された論理アドレスの格納履歴情報を格納するための複数の履歴情報エリアとをフラッシュメモリ内に形成したことを特徴とするフラッシュメモリを利用した記憶装置。

【請求項２】　データを格納するための複数のデータエリアと、該データエリアの各々に対応付けられ、上記データエリア個々の論理アドレスを格納するための複数のアロケーションエリアと、上記各データエリアに対応付けられ、各データエリアと対応する上記アロケーションエリアに格納された論理アドレスの格納履歴情報を格納するための複数の履歴情報エリアとを有するフラッシュメモリと、該メモリ中の空きデータエリアを検出する空きエリア検出手段と、入力されたアドレスデータと同一の論理アドレスデータが格納されている上記アロケーションエリアを検索するエリア検索手段と、該検索手段で検索されたアロケーションエリアと対応する履歴情報エリアに格納されている履歴情報に従って新たな履歴情報を作成する手段と、上記空きデータエリア検出手段で検出したデータエリアと対応するアロケーションエリア及び履歴情報エリアに夫々上記アドレスデータ及び上記作成手段で作成された新たな履歴情報を書込む手段とを備えたことを特徴とするフラッシュメモリを利用した記憶装置。

【請求項３】　請求項２の上記書込手段は、上記アドレスデータと共に入力された新規データを上記空きデータエリア検出手段で検出したデータエリアに書込むことを特徴とするフラッシュメモリを利用した記憶装置。

【請求項４】　請求項１～３のフラッシュメモリはＮＡＮＤ型であることを特徴とするフラッシュメモリを利用した記憶装置。

【請求項５】　請求項４において、対応関係にある上記データエリア、アロケーションエリア及び履歴情報エリアは書込単位となる同一ブロック内に位置することを特徴とするフラッシュメモリを利用した記憶装置。

【請求項６】　請求項１～５において、上記履歴情報は同一の論理アドレスが格納された回数を示す情報であることを特徴とするフラッシュメモリを利用した記憶装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はフラッシュメモリを利用した記憶装置に関する。

【０００２】

【従来の技術】フラッシュメモリはバッテリバックアップなしに半永久的にメモリ内容を保持可能であり、かつ、アクセスに関しては半導体メモリと同様に電気的に行なえるため、ＦＤ（フロッピィディスク）やＨＤ（ハードディスク）等のディスク型の記憶装置に換わる記憶装置として期待されている。

【０００３】しかし、フラッシュメモリのデータ消去は最低でも数Ｋバイト単位で行わなければならず、１バイトのデータを消去又は変更する際には、まず変更対象となる１バイトを含む数Ｋバイトのデータを半導体ＲＡＭ等の別のメモリに待避した後、上記数Ｋバイトの領域を消去し、その後待避した数Ｋバイトのデータをフラッシュメモリ上に書込まなければならず、時間がかかるという問題があった。

【０００４】そこで、米国特許第５，４０４，４８５号明細書には、フラッシュメモリ中のデータを格納すべきデータエリア毎に対応するアロケーションエリアを設け、このアロケーションエリアに対応するデータエリアの論理アドレス又は更新済であることを示すデータを格納可能としておき、データエリア中の既格納データに対する書替は、この既格納データエリアに対して書替処理を行なうのではなく、空きのデータエリアに新規データを書込み、このデータエリアと対応するアロケーションエリアに上記既格納データの論理アドレスを書込むと共に既格納データに対応したアロケーションエリアのデータを更新済であることを示すデータに書替えることによりデータエリアの書替処理を省略することによりデータ更新速度を向上する方法が開示されている。

【０００５】

【発明が解決しようとする課題】しかし、この米国特許に開示された方法は、消去単位は数Ｋバイトであるが、データの書き込みは１バイト毎に行なえる所謂ＮＯＲ型と称されるフラッシュメモリを対象としているものであり、書込み単位が数百バイトとなるＮＡＮＤ型のフラッシュメモリには適用できないのである。

【０００６】即ち、フラッシュメモリでは、ＮＯＲ型、ＮＡＮＤ型共に電気的に無チャージ状態の論理「１」からチャージ状態の論理「０」への変更は、チャージ状態から無チャージ状態にする所謂「データ消去」に比して容易に行なえるという特徴があり、従って、上記更新済を示すデータを例えばオール「０」のデータとして規定しておけば、ＮＯＲ型ではアロケーションエリアの論理アドレスデータをオール「０」の更新済データに変更することは容易に行なえる。しかし、ＮＡＮＤ型では、１バイト単位での書込みが行なえないため、上記米国特許に記載の発明は採用できないのである。

【０００７】

【課題を解決するための手段】本発明は上述の課題に鑑みてなされたもので、その基本的な特徴は、データを格納するための複数のデータエリアと、該データエリアの各々に対応付けられ、上記データエリア個々の論理アド

レスを格納するための複数のアロケーションエリアと、上記各データエリアに対応付けられ、各データエリアと対応する上記アロケーションエリアに格納された論理アドレスの格納履歴情報を格納するため複数の履歴情報エリアとをフラッシュメモリ内に形成したことにある。

【０００８】

【発明の実施の形態】図１は、本発明を適用したシステムの主要回路構成を示すブロック図であり、マイクロコンピュータからなる主制御部１は内蔵の制御プログラムに基づいて例えば半導体ランダムアクセスメモリＲＡＭ２に対するアクセスを行なったり、フラッシュメモリ３を制御するフラッシュメモリ制御部４をコントロールする。

【０００９】本実施例が対象とするフラッシュメモリ３は、例えば図２に示す如く（５１２＋１６）バイトからなるブロック単位での書込み及び８ブロック（４．１２５Ｋバイト）を１単位とする単位毎の消去しか行なえないＮＡＮＤ型である。また、上記各ブロックには図３に模式的に示す如く５１２バイトのデータが格納されるデータエリア５、２バイトの論理アドレスを格納するアロケーションエリア６、２バイトの同一の論理アドレスが格納された回数を示す情報が格納される履歴情報エリア７に分割され、残りの１２バイトはこのＥＣＣデータ等の格納エリア８として利用される。

【００１０】フラッシュメモリ制御部４は、主制御部１の制御の下で供給されたアドレスデータや新規データに基づいて上記フラッシュメモリ３へのデータ読出・書込を行なうもので、専用のゲートアレイやワンチップマイコンで構成される。

【００１１】次に本実施例の動作について説明する。図４はフラッシュメモリ制御部４のフラッシュメモリ３へのデータ書込時の制御動作を示すフローチャートである。

【００１２】フラッシュメモリ制御部４は、主制御部１より論理アドレスａに新規データＤＮを書込むように指令を受けると、Ｓ１ステップにおいて新規データ書込み（含：更新）のための初期化を行なう。

【００１３】具体的には、フラッシュメモリ３中の各ブロックを特定するための変数ｉに先頭ブロック番号である「０」をセットし、また、履歴情報エリア８中のデータを格納するための変数ｃｎｔに「－１」をセットし、更にブロック番号を格納するための変数ｂｌｋに「－１」をセットする。尚、上記各変数ｉ、ｃｎｔ及びｂｌｋは、フラッシュメモリ制御部４中のＲＡＭ（図示せず）に保持される。

【００１４】続くＳ２ステップでは、フラッシュメモリ３中のブロックｉのアロケーションエリア（以下、ａｌｌｏｃ［ｉ］と称す）中に格納されている論理アドレスが論理アドレスａと一致するか否かを判定する。

【００１５】一致した場合には、Ｓ３ステップにおいて、フラッシュメモリ３中のブロックｉの履歴情報エリア（以下、ｃｏｕｎｔ［ｉ］と称す）中のデータと変数ｃｎｔとを比較し、「ｃｏｕｎｔ［ｉ］＞ｃｎｔ」の際にはＳ４ステップにおいて変数ｂｌｋの値を変数ｉの値に置換すると共に、変数ｃｎｔの値をｃｏｕｎｔ［ｉ］の値に置換し、処理をＳ５ステップに進める。

【００１６】一方、Ｓ２ステップにおいて不一致と判定された場合、及び、Ｓ３ステップにおいて「ｃｏｕｎｔ［ｉ］＞ｃｎｔ」と判定されなかった場合には、直ちに処理をＳ５ステップに進める。

【００１７】Ｓ５ステップで変数ｉをインクリメントし、続くＳ６ステップでは変数ｉの値がフラッシュメモリ３中のブロック数（ｍａｘ　ｂｌｋ）を超えたか否かを判定し、超えていないと判定すると処理をＳ２ステップに戻す。

【００１８】一方、超えたと判定した場合、即ちフラッシュメモリ３の全てのブロックのアロケーションエリア６に格納されている論理アドレスと論理アドレスａとの比較が終了しているので、最終的な処理を行なうためＳ７ステップに進む。

【００１９】Ｓ７ステップでは、変数ｂｌｋの値が「－１」であるか否かを判定する。変数ｂｌｋの値が「－１」である場合、Ｓ８ステップにおいて変数ｃｎｔに「０」をセットし、一方、変数ｂｌｋの値が「－１」でない場合、Ｓ９ステップにおいて変数ｃｎｔの値をインクリメントした後、処理をＳ１０ステップに進める。

【００２０】Ｓ１０ステップでは、フラッシュメモリ３中の空きブロックを検出し、そのブロックのデータエリア５、アロケーションエリア６及び履歴情報エリア８に夫々新規データＤＮ、論理アドレスａ及び変数ｃｎｔの値を書込み、処理を終了する。

【００２１】尚、上記空きブロックの検出は、上記フラッシュメモリ３のブロックのデータが消去された状態では全エリアとも無チャージ状態の論理「１」となっているので、これを検出することにより行なえる。また、書込対象となる全エリアは同一ブロック上にあるため、ＮＡＮＤ型のものであっても書込は簡単に行なえる。

【００２２】次に、図４のフローチャートの制御動作をより理解し易くするために一具体例を使って説明する。

【００２３】今、図５に示す如くフラッシュメモリ３にはブロック０～７までの８個のブロックが存在し、かつ、全ブロックのエリアは全て消去されオール「１」となっているものとする。

【００２４】この状態において、主制御部１より論理アドレスａ１に新規データｄ１を書込むように指示があると、フラッシュメモリ制御部４は、図４のフローを実行するが、この時点ではいずれのブロックも空きであるためＳ２、Ｓ５、Ｓ６ステップを８回繰返し実行した後、Ｓ７ステップに処理を進める。また、この時点では変数ｂｌｋは「－１」のままであるため、Ｓ８、Ｓ１０ステ

ップが順次処理され、図６に示す如く空きブロックであるブロック０のデータエリア５、アロケーションエリア６及び履歴情報エリア７に夫々データｄ１、論理アドレスａ１及び変数ｃｎｔの値である数値「０」が格納される。

【００２５】また、続いて異なる論理アドレスａ２、ａ３に夫々新規データｄ２、ｄ３（ｄ１＝ｄ２＝ｄ３であっても良い）を書込むよう指示があると、フラッシュメモリ３のアロケーションエリア６には同一のアドレスが存在しないので、上述のケースと同様にＳ１、Ｓ２、Ｓ５、Ｓ６、Ｓ７、Ｓ８、Ｓ１０ステップのみが処理され、図７に示す如くデータｄ２、ｄ３及びアドレスａ２、ａ３が空き領域であったブロック１及び２に格納される。

【００２６】次に、主制御部１より論理アドレスａ１に新規データｄ４を書込むように指令を受けると、ブロック０のアロケーションエリア中の論理アドレス（ａｌｌｏｃ［０］）のみが「ａ１」となっているため、Ｓ２ステップでこれを判定した際、Ｓ３及びＳ４ステップでの処理により変数ｂｌｋにはブロック０のブロック番号を示す数値「０」がセットされ、変数ｃｎｔにはブロック０の履歴情報エリアの値「０」がセットされる。従って、Ｓ２～Ｓ６ステップにおける全ブロックに対する判定を行なった後、Ｓ７、Ｓ９、Ｓ１０ステップが順次処理されることとなるので、図８に示す如くブロック３のデータエリア５、アロケーションエリア６及び履歴情報エリア７に夫々データｄ４、論理アドレスａ１及び数値「１」が書込まれることとなる。

【００２７】更に、再々度論理アドレスａ１に新規データｄ５を書込むように主制御部１より指令を受けると、変数ｉが「０」及び「３」となった際にＳ２ステップで一致と判定され、変数ｉが「０」の時に変数ｂｌｋ及びｃｎｔに夫々数値「０」がセットされ、変数ｉが「０」の時には変数ｂｌｋ及びｃｎｔに夫々数値「３」及び「１」がセットされることとなる。従って、Ｓ７ステップ以下を処理した結果、図９に示す如くブロック４のデータエリア５、アロケーションエリア６及び履歴情報エリア７には、夫々データｄ５、論理アドレスａ１及び数値「２」が書込まれることとなる。

【００２８】このように、本実施例装置では、新規及び更新データは空きブロックに論理アドレスと共に順次書込まれ、また更新データの新旧は各ブロックの履歴情報エリア７の数値で判定できる。

【００２９】図１０は、フラッシュメモリ制御部４のフラッシュメモリ３に対するデータ読出制御動作を示すフローチャートである。

【００３０】フラッシュメモリ制御部４は、主制御部１より論理アドレスａと対応するデータを読み出すように指令を受けると、図１０に示すＳ２０ステップ以下を処理する。

【００３１】Ｓ２０～Ｓ２５ステップでは、上述のＳ１～Ｓ６ステップと同様にフラッシュメモリ３中の全てのブロックを順次サーチし、アロケーションエリア６に上記論理アドレスａと同一のアドレスが格納されているブロックを検出し、そのブロック番号及び履歴情報エリア７内のデータ（ｃｏｕｎｔ［ｉ］）を夫々変数ｂｌｋ及びｃｎｔにセットする。尚、検出ブロックが複数ある場合には、最も履歴情報の数値が大きいもの、即ち最新の変更データが格納されたブロックのものがセットされる。

【００３２】Ｓ２６ステップでは、Ｓ７ステップと同様に変数ｂｌｋの値を判定する。このとき、変数ｂｌｋがＳ２０ステップで初期化された「－１」のままである場合、論理アドレスａが格納されたブロックが発見されなかったことを意味するので、エラーとして処理する。

【００３３】一方、変数ｂｌｋの値が「－１」以外であれば、論理アドレスａが格納されたブロックが発見され、上記Ｓ２３ステップにおいてそのブロック番号に変更されたことを意味するため、続くＳ２７ステップにおいて変数ｂｌｋの値と同一のブロック番号のブロックのデータエリア５中のデータを読み出し主制御部１に出力し処理を終了する。

【００３４】

【発明の効果】本発明によれば、ＮＡＮＤ型のフラッシュメモリにおいても消去を必要とすることなくデータ変更が可能となるので、データ更新速度を向上できる。

【図面の簡単な説明】

【図１】本発明を適用した装置の主要部を示すブロック回路図である。

【図２】本発明が対象とするＮＡＮＤ型フラッシュメモリの構成を示す模式図である。

【図３】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図４】本発明の一実施例における動作を説明するためのフローチャートである。

【図５】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図６】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図７】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図８】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図９】本発明の一実施例におけるフラッシュメモリを示す模式図である。

【図１０】本発明の一実施例における動作を説明するためのフローチャートである。

【符号の説明】

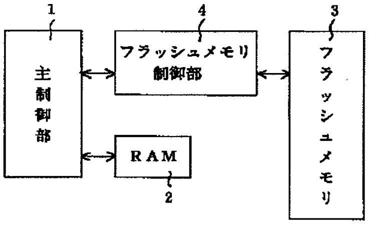
３　フラッシュメモリ
４　フラッシュメモリ制御部

5　データエリア

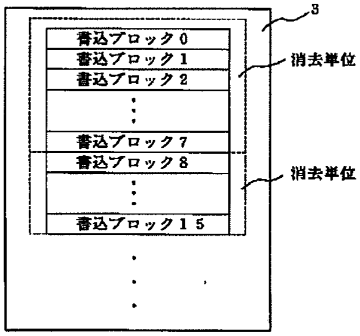
6　アロケーションエリア　＊

＊７　履歴情報エリア

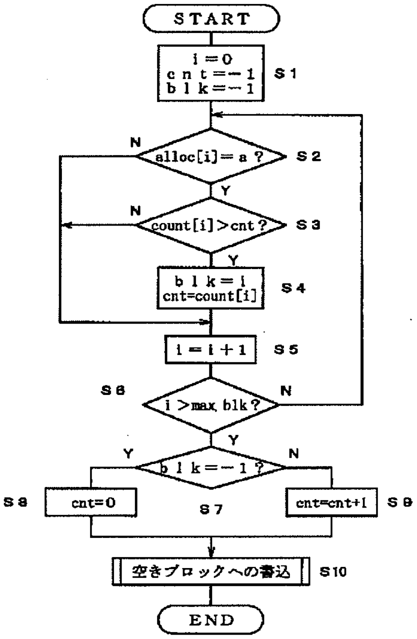
【図１】


1
主制御部
4
フラッシュメモリ制御部
RAM
2
3
フラッシュメモリ

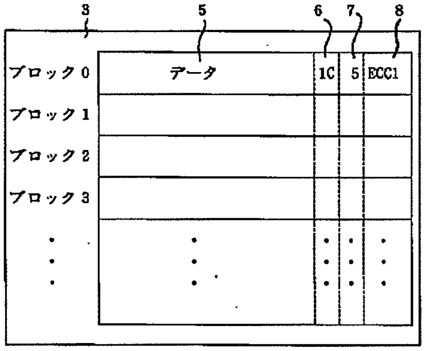
【図２】


3
書込ブロック０
書込ブロック１
書込ブロック２
書込ブロック７
書込ブロック８
書込ブロック１５
消去単位
消去単位

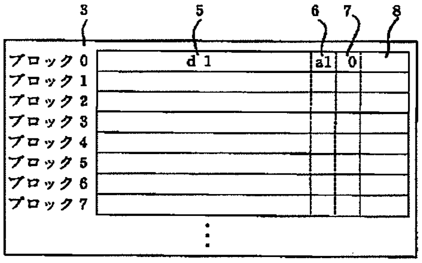
【図４】


START
i＝0
cnt＝−1
blk＝−1
S1
N
alloc[i]＝a？
S2
Y
N
count[i]＞cnt？
S3
Y
blk＝i
cnt=count[i]
S4
i＝i＋1
S5
S6
i＞max blk？
N
Y
Y
blk＝−1？
N
S8
cnt=0
S7
cnt=cnt+1
S9
空きブロックへの書込
S10
END

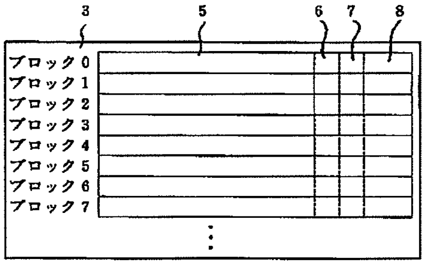
【図３】


3
5
6
7
8
ブロック０
データ
1C
5
ECC1
ブロック１
ブロック２
ブロック３

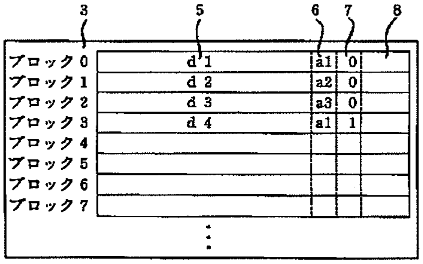
【図６】


3
5
6
7
8
ブロック０
d 1
a1
0
ブロック１
ブロック２
ブロック３
ブロック４
ブロック５
ブロック６
ブロック７

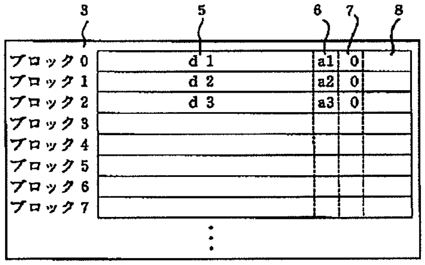
【図５】


3
5
6
7
8
ブロック０
ブロック１
ブロック２
ブロック３
ブロック４
ブロック５
ブロック６
ブロック７

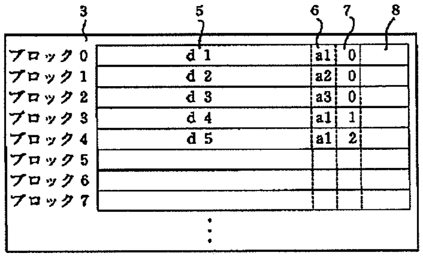
【図８】


3
5
6
7
8
ブロック０
d 1
a1
0
ブロック１
d 2
a2
0
ブロック２
d 3
a3
0
ブロック３
d 4
a1
1
ブロック４
ブロック５
ブロック６
ブロック７

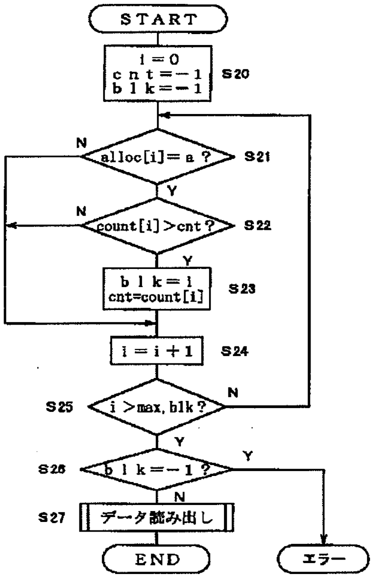
【図７】


3
5
6
7
8
ブロック0
ブロック1
ブロック2
ブロック3
ブロック4
ブロック5
ブロック6
ブロック7
d 1
d 2
d 3
a1
a2
a3
0
0
0

【図９】


3
5
6
7
8
ブロック0
ブロック1
ブロック2
ブロック3
ブロック4
ブロック5
ブロック6
ブロック7
d 1
d 2
d 3
d 4
d 5
a1
a2
a3
a1
a1
0
0
0
1
2

【図１０】


START
i＝0
cnt＝−1
blk＝−1
S20
N
alloc[i]＝a ?
S21
Y
N
count[i]＞cnt ?
S22
Y
blk＝i
cnt=count[i]
S23
i＝i＋1
S24
S25
i＞max, blk ?
N
Y
S26
blk＝−1 ?
Y
N
S27
データ読み出し
END
エラー

フロントページの続き

(72)発明者　山下　隆弘
鳥取県鳥取市南吉方３丁目201番地　鳥取
三洋電機株式会社内

(72)発明者　網崎　真哉
鳥取県鳥取市南吉方３丁目201番地　鳥取
三洋電機株式会社内

(72)発明者　梅沢　功
鳥取県鳥取市南吉方３丁目201番地　鳥取
三洋電機株式会社内

(72)発明者　浦川　文隆
鳥取県鳥取市南吉方３丁目201番地　鳥取
三洋電機株式会社内

(72)発明者　前田　浩司
鳥取県鳥取市南吉方３丁目201番地　鳥取
三洋電機株式会社内

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] The step which detects the command from a predetermined user, and the 1st field which memorizes one or more users' data, The step which processes said command using the storage section which is used by said one or more users memorized to said 1st field, and in which the 2nd field set as the free area which is not used as said 1st field is formed, The information processing approach characterized by having the step which outputs the result of said processing.
[Claim 2] Said 2nd field is the information processing approach according to claim 1 which has one or more blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said 2nd field as some said user's data.
[Claim 3] The step which detects the command from a predetermined user, and the 1st field which memorizes one or more users' data, The step which processes said command using the storage section which is used by said one or more users memorized to said 1st field, and in which the 2nd field set as the free area which is not used as said 1st field is formed, The transmission medium characterized by transmitting a program equipped with the step which outputs the result of said processing.
[Claim 4] A detection means to detect the input signal from the outside, and the 1st field which memorizes one or more users' data, A storage means used by said one or more users memorized to said 1st field by which the 2nd field set as the free area which is not used as said 1st field is formed, The information processor characterized by having a processing means to perform processing corresponding to said input signal using said storage means, and an output means to output the result of processing of said processing means outside.
[Claim 5] Said 2nd field is an information processor according to claim 4 which has one or more blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said 2nd field as some said user's data.
[Claim 6] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, In the information processing approach in an information processor equipped with a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per physical block of predetermined magnitude is formed The step at which said processing means assigns a logical-block number to the data memorized by said physical block, The information processing approach characterized by equipping said storage means with the step which memorizes the new data which have a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[Claim 7] Said 2nd field is the information processing approach according to claim 6 which has one or more physical blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said

2nd field as some said user's data.
[Claim 8] The data memorized by said physical block have further the identification information which identifies the newness of data which has said same logical-block number. Said storage means The information processing approach according to claim 6 characterized by memorizing the new data which have a predetermined logical-block number with reference to the value of said identification information to physical blocks other than the physical block the newest data of the data which have said same logical-block number are remembered to be.
[Claim 9] Said identification information is the information processing approach according to claim 8 characterized by being the value of the counter at the time of the data storage which has the time of day or said logical-block number at the time of the data storage which has the value of the counter showing the number of updating of the data which have said logical-block number, and said logical-block number.
[Claim 10] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, Are used by said one or more users memorized to said 1st field. In the transmission medium which transmits the program used for an information processor equipped with a storage means by which the 2nd field managed per physical block of predetermined magnitude is formed The step at which said processing means assigns a logical-block number to the data memorized by said physical block, The transmission medium characterized by transmitting the program said whose storage means is equipped with the step which memorizes the new data which have a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[Claim 11] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, In an information processor equipped with a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per physical block of predetermined magnitude is formed Said processing means assigns a logical-block number to the data memorized by said physical block. Said storage means The information processor characterized by memorizing the new data which have a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[Claim 12] Said 2nd field is an information processor according to claim 11 which has one or more physical blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said 2nd field as some said user's data.
[Claim 13] It is the information processor according to claim 11 which the data memorized by said physical block have further the identification information which identifies the newness of data which has said same logical-block number, and is characterized by for said storage means to memorize the new data which have a predetermined logical-block number with reference to the value of said identification information to physical blocks other than the physical block the newest data of the data which have said same logical-block number are remembered to be.
[Claim 14] Said identification information is an information processor according to claim 13 characterized by being the value of the counter at the time of the data storage which has the time of day or said logical-block number at the time of the data storage which has the value of the counter showing the number of updating of the data which have said logical-block number, and said logical-block number.
[Claim 15] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, In the information processing approach in an information processor equipped with a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is

formed The data of a predetermined block of said 2nd field The information processing approach characterized by having a recognition number, comparing the recognition number which said command with which said processing means was supplied by said user has with the recognition number which said data have, and having the step which processes said command corresponding to the comparison result.
[Claim 16] Said 2nd field is the information processing approach according to claim 15 which has one or more blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said 2nd field as some said user's data.
[Claim 17] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, Are used by said one or more users memorized to said 1st field. In the transmission medium which transmits the program used for an information processor equipped with a storage means by which the 2nd field managed per block of predetermined magnitude is formed The data of a predetermined block of said 2nd field It is the transmission medium characterized by having a recognition number, and for said processing means comparing the recognition number which said command supplied by said user has with the recognition number which said data have, and transmitting a program equipped with the step which processes said command corresponding to the comparison result.
[Claim 18] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, In an information processor equipped with a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed The data of a predetermined block of said 2nd field It is the information processor characterized by having a recognition number, and for said processing means comparing the recognition number which said command supplied by said user has with the recognition number which said data have, and processing said command corresponding to the comparison result.
[Claim 19] Said 2nd field is an information processor according to claim 18 which has one or more blocks and is characterized by memorizing the number corresponding to the block of the head of the field used by the user, and the number corresponding to the last block to said 1st field in said 2nd field as some said user's data.
[Claim 20] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, In the information processing approach in an information processor equipped with a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed The step at which said processing means assigns the number corresponding to the sequence memorized to the data memorized by said block, In said storage means which memorized the number corresponding to the block of the head of the field which said user uses for said 1st field, and the number corresponding to the last block When the block which has the number of said last is a block of said last, The information processing approach characterized by having the step which memorizes new data to the block of said head, and memorizes said new data to the block next to the block which has the number of said last when the block which has the number of said last is not a block of said last.
[Claim 21] It is the information processing approach according to claim 20 characterized by not memorizing said new data when there is a block which has the same data as said new data.
[Claim 22] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, Are used by said one or more users memorized to said 1st field. In the transmission medium which transmits the program used for an information processor equipped with a storage means by which the 2nd field managed per block of predetermined magnitude is formed The step at which said processing means assigns the number

corresponding to the sequence memorized to the data memorized by said block, In said storage means which memorized the number corresponding to the block of the head of the field which said user uses for said 1st field, and the number corresponding to the last block When the block which has the number of said last is a block of said last, The block which memorizes new data to the block of said head, and has the number of said last The transmission medium characterized by transmitting a program equipped with the step which memorizes said new data to the block next to the block which has the number of said last when it is not the block of said last.
[Claim 23] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, It has a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed. Said processing means The number corresponding to the sequence memorized is assigned to the data memorized by said block. Said storage means The number corresponding to the block of the head of the field which said user uses for said 1st field, and the number corresponding to the last block are memorized. When the block which has the number of said last is a block of said last, The information processor characterized by memorizing new data to the block of said head, and memorizing said new data to the block next to the block which has the number of said last when the block which has the number of said last is not a block of said last.
[Claim 24] It is the information processor according to claim 23 characterized by not memorizing said new data when there is a block which has the same data as said new data.
[Claim 25] The step which detects the command from a predetermined user, and the 1st field which memorizes one or more users' data, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by said one or more users memorized to said 1st field is formed, and / said 2nd field] And the information processing approach characterized by having the step which processes said command, and the step which outputs the result of said processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to said 1st field.
[Claim 26] The step which detects the command from a predetermined user, and the 1st field which memorizes one or more users' data, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by said one or more users memorized to said 1st field is formed, and / said 2nd field] And the transmission medium characterized by transmitting a program equipped with the step which processes said command, and the step which outputs the result of said processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to said 1st field.
[Claim 27] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes one or more users' data, It has a storage means used by said one or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed. Said storage means The information processor characterized by memorizing two or more data which specify an access privilege different, respectively to said 1st field to each user as opposed to the predetermined field in said 2nd field.
[Claim 28] The step which detects the command from a predetermined user, and the 1st field which memorizes two or more users' data, Are used by said two or more users memorized to said 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in said 2nd field jointly to said 1st field is used. The information processing approach characterized by having the step which processes said command, and the step which outputs the result of said processing.
[Claim 29] The step which detects the command from a predetermined user, and the 1st field which memorizes two or more users' data, Are used by said two or more users memorized to said 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in said 2nd field jointly to

said 1st field is used. The transmission medium characterized by transmitting a program equipped with the step which processes said command, and the step which outputs the result of said processing.

[Claim 30] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes two or more users' data, It has a storage means used by said two or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed. Said storage means The information processor characterized by memorizing the data with which two or more users use the predetermined field in said 2nd field jointly to said 1st field.

[Claim 31] The step which detects the command from a predetermined user, and the 1st field which memorizes two or more users' data, Are used by said two or more users memorized to said 1st field. A predetermined field [in / the 2nd field managed per block of predetermined magnitude is formed and / said 2nd field], And the information processing approach characterized by having the step which processes said command, and the step which outputs the result of said processing using the storage section which memorizes two or more data which specify the access privilege from which two or more users differ, respectively to said 1st field.

[Claim 32] The step which detects the command from a predetermined user, and the 1st field which memorizes two or more users' data, Are used by said two or more users memorized to said 1st field. A predetermined field [in / the 2nd field managed per block of predetermined magnitude is formed and / said 2nd field], And the transmission medium characterized by transmitting a program equipped with the step which processes said command, and the step which outputs the result of said processing using the storage section which memorizes two or more data which specify the access privilege from which two or more users differ, respectively to said 1st field.

[Claim 33] A detection means to detect the command from a predetermined user, and a processing means to process said command, An output means to output the result of processing of said processing means, and the 1st field which memorizes two or more users' data, It has a storage means used by said two or more users memorized to said 1st field by which the 2nd field managed per block of predetermined magnitude is formed. Said storage means The information processor characterized by memorizing the predetermined field in said 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to said 1st field.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention relates to a transmission medium at the information processing approach of receiving the command from a predetermined user in the information processing approach and an information processor, and a list especially, processing the command in them, and transmitting the result of processing to them about a transmission medium and an information processor, and a list.
[0002]
[Description of the Prior Art] The IC card (smart card) used by the cybermoney system or the security system is developed.
[0003] Such an IC card contains the memory which memorizes CPU which performs various processings, data required for processing, etc., and is transmitting and receiving data in the condition of having made predetermined reader/writer (R/W) contacting.
[0004] Moreover, there is also an IC card of the dc-battery loess mold which does not have the dc-battery in an IC card itself. Power is supplied to the IC card of such a dc-battery loess mold from R/W.
[0005]
[Problem(s) to be Solved by the Invention] However, in such an IC card, since it is premised on using it in the condition of having made R/W contacting, when using it by non-contact, it has the problem that it is difficult to acquire power.
[0006] Moreover, although how to supply power required for an IC card by the electromagnetic wave is also considered while transmitting and receiving data between an IC card and R/W by non-contact using an electromagnetic wave While having accessed the memory which an IC card contains in such an approach When the receive state of an electromagnetic wave becomes a defect, it has the problem that there is possibility that sufficient power will no longer be obtained and a defect will arise for the adjustment of the data in memory (memory colla tempestade PUSHON (Memory Corruption) arises).
[0007] Furthermore, if information is held like FAT (FileAllocation Table) of MS-DOS (Microsoft-Disc Operating System) in every [data are remembered to be] unit (it is a sector when it is MS-DOS), the field proportional to the area size data are remembered to be is needed for data control, and it has the problem that the use effectiveness of memory falls. Moreover, if a storage region is managed in the predetermined unit data are remembered to be, when memorizing the data of the magnitude with which the unit is not filled, the storage region which is not used occurs and it has further the problem that the use effectiveness of memory falls.
[0008] Furthermore, in the above-mentioned IC card, since uniform processing is performed to R/W, it has the problem that it is difficult to perform processing according to individual corresponding to two or more R/W.
[0009] The 1st field which this invention was made in view of such a situation, and memorizes two or more users' data, While using the storage section including the 2nd field which is used by two or more users memorized to the 1st field, and is managed per physical block of predetermined magnitude A logical-block number is assigned to the data memorized by the physical block. Memorize the data to

physical blocks other than the physical block the data which have the logical-block number are remembered to be, or The number corresponding to the sequence memorized is assigned to the data memorized by the physical block. When the physical block which has the last number is the last physical block, By memorizing the data to a top physical block, and memorizing the data to the next physical block of the physical block which has a number at the tail end, when the physical block which has the last number is not the last physical block Generating of memory colla tempestade PUSHON in memory is controlled logically.
[0010] Moreover, this invention is not the area size used by the user, but is the information on the amount proportional to the number of users (the number corresponding to a top physical block, and number corresponding to the last physical block), and enables it to manage data by holding the number corresponding to the physical block of the head of the field used by each user, and the number corresponding to the last physical block.
[0011] Furthermore, a predetermined field [in / on the above-mentioned storage section and / in this invention / the 2nd field], By and the thing for which the data which specify a predetermined field [in / for two or more data which specify an access privilege different, respectively / in memorizing to the 1st field **** / the 2nd field] corresponding to one user are memorized to the 1st field corresponding to two or more users It enables it to perform processing according to individual corresponding to two or more users (R/W).
[0012]
[Means for Solving the Problem] The step to which the information processing approach according to claim 1 detects the command from a predetermined user, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It is characterized by having the step which processes a command using the storage section in which the 2nd field set as the free area which is not used as the 1st field is formed, and the step which outputs the result of processing.
[0013] The step to which a transmission medium according to claim 3 detects the command from a predetermined user, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It is characterized by transmitting a program equipped with the step which processes a command using the storage section in which the 2nd field set as the free area which is not used as the 1st field is formed, and the step which outputs the result of processing.
[0014] A detection means by which an information processor according to claim 4 detects the input signal from the outside, A storage means by which the 2nd field which is used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field and which is set as the free area which is not used as the 1st field is formed, It is characterized by having a processing means to perform processing corresponding to an input signal using a storage means, and an output means to output the result of processing of a processing means outside.
[0015] The information processing approach according to claim 6 is characterized by equipping the step at which a processing means assigns a logical-block number to the data memorized by the physical block, and a storage means with the step memorized to physical blocks other than the physical block the data which have are remembered [data / which have a predetermined logical-block number / new] to be in the logical-block number.
[0016] A transmission medium according to claim 10 is characterized by transmitting the program which the step at which a processing means assigns a logical-block number to the data memorized by the physical block, and a storage means equip with the step memorized to physical blocks other than the physical block the data which have are remembered [data / which have a predetermined logical-block number / new] to be in the logical-block number.
[0017] An information processor according to claim 11 assigns a logical-block number to the data with which a processing means is memorized by the physical block, and a storage means is characterized by memorizing the new data which have a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[0018] The information processing approach according to claim 15 is characterized by for the data of a predetermined block of the 2nd field having a recognition number, comparing the recognition number

which the command with which the processing means was supplied by the user has with the recognition number which data have, and equipping them with the step which processes a command corresponding to the comparison result.

[0019] It is characterized by for the data of a predetermined block of the 2nd field having a recognition number in a transmission medium according to claim 17, and for a processing means comparing the recognition number which the command supplied by the user has with the recognition number which data have, and transmitting a program equipped with the step which processes a command corresponding to the comparison result.

[0020] It is characterized by for the data of a predetermined block of the 2nd field having a recognition number in an information processor according to claim 18, and for a processing means comparing the recognition number which the command supplied by the user has with the recognition number which data have, and processing a command corresponding to the comparison result.

[0021] The step at which, as for the information processing approach according to claim 20, a processing means assigns the number corresponding to the sequence memorized to the data memorized by block, In the storage means which memorized the number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block When the block which has the last number is the last block, new data It is characterized by having the step which memorizes to a top block, and memorizes new data to the block next to the block which has the last number when the block which has the last number is not the last block.

[0022] The step at which a transmission medium according to claim 22 assigns the number corresponding to the sequence memorized to the data with which a processing means is memorized by block, In the storage means which memorized the number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block When the block which has the last number is the last block, new data It is characterized by transmitting a program equipped with the step which memorizes to a top block, and memorizes new data to the block next to the block which has the last number when the block which has the last number is not the last block.

[0023] A detection means by which an information processor according to claim 23 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed. A processing means The number corresponding to the sequence memorized is assigned to the data memorized by block. A storage means The number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block are memorized. It is characterized by memorizing new data to a top block, when the block which has the last number is the last block, and memorizing new data to the block next to the block which has the last number, when the block which has the last number is not the last block.

[0024] The step to which the information processing approach according to claim 25 detects the command from a predetermined user, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And it is characterized by having the step which processes a command, and the step which outputs the result of processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to the 1st field.

[0025] The step to which a transmission medium according to claim 26 detects the command from a predetermined user, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And it is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to the 1st field.

[0026] A detection means by which an information processor according to claim 27 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing two or more data which specify an access privilege different, respectively to the 1st field to each user as opposed to the predetermined field in the 2nd field.

[0027] The step to which the information processing approach according to claim 28 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field is used. It is characterized by having the step which processes a command, and the step which outputs the result of processing.

[0028] The step to which a transmission medium according to claim 29 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field is used. It is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing.

[0029] A detection means by which an information processor according to claim 30 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.

[0030] The step to which the information processing approach according to claim 31 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the predetermined field in the 2nd field and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field is used. It is characterized by having the step which processes a command, and the step which outputs the result of processing.

[0031] The step to which a transmission medium according to claim 32 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the predetermined field in the 2nd field and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field is used. It is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing.

[0032] A detection means by which an information processor according to claim 33 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.

[0033] In the information processing approach according to claim 1, a transmission medium according to claim 3, and an information processor according to claim 4 A command is processed using the storage section in which the 2nd field set as the 1st field which memorizes one or more users' data, and the free area which is used by one or more users memorized to the 1st field, and is not used as the 1st field is formed.

[0034] In the information processing approach according to claim 6, a transmission medium according to claim 10, and an information processor according to claim 11, a processing means assigns a logical-block number to the data memorized by the physical block, and the new data with which a storage means has a predetermined logical-block number are memorized to physical blocks other than the physical block the data which have the logical-block number are remembered to be.

[0035] In the information processing approach according to claim 15, a transmission medium according to claim 17, and an information processor according to claim 18, it has a recognition number, a processing means compares the recognition number which the command supplied by the user has with the recognition number which data have, and the data of a predetermined block of the 2nd field process a command corresponding to the comparison result.

[0036] In the information processing approach according to claim 20, a transmission medium according to claim 22, and an information processor according to claim 23 The number corresponding to the sequence memorized can be assigned to the data memorized by block with a processing means. With a storage means When the block which has the last number is the last block, new data are memorized by top block, and new data are memorized by the block next to the block which has the last number when the block which has the last number is not the last block.

[0037] In the information processing approach according to claim 25, a transmission medium according to claim 26, and an information processor according to claim 27 As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And a command is processed using the storage section which memorizes two or more data which specify an access privilege which is different to each user, respectively to the 1st field.

[0038] In the information processing approach according to claim 28, a transmission medium according to claim 29, and an information processor according to claim 30 The 1st field which memorizes two or more users' data, and the 2nd field which is used by two or more users memorized to the 1st field, and is managed per block of predetermined magnitude are formed, and it sets to the 1st field. A command is processed using the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.

[0039] In the information processing approach according to claim 31, a transmission medium according to claim 32, and an information processor according to claim 33 It is used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and a command is processed using the storage section which memorizes the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.

[0040]

[Embodiment of the Invention] Although the gestalt of operation of this invention is explained below, it is as follows, when the gestalt (however, an example) of operation [/ in the parenthesis after each means] is added and the description of this invention is described, in order to clarify correspondence relation between each means of invention given in a claim, and the gestalt of the following operations. However, of course, this publication does not mean limiting to what indicated each means.

[0041] A detection means by which an information processor according to claim 4 detects the input signal from the outside (for example, BPSK demodulator circuit 62 of drawing 3), Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. A storage means by which the 2nd field set as the free area which is not used as the 1st field is formed (for example, EEPROM66 of drawing 3), It is characterized by having a processing means (for example, sequencer 91 of drawing 3) to perform processing corresponding to an input signal using a storage means, and an output means (for example, BPSK modulation circuit 68 of drawing 3) to output the result of processing of a processing means outside.

[0042] A processing means (for example, sequencer 91 of drawing 3) assigns a logical-block number to the data memorized by the physical block, and an information processor according to claim 11 is characterized by memorizing the new data with which a storage means (for example, EEPROM66 of drawing 3) has a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.

[0043] An information processor according to claim 18 is characterized by for the data of a predetermined block of the 2nd field having a recognition number, comparing the recognition number which the command with which the processing means (for example, sequencer 91 of drawing 3) was supplied by the user has with the recognition number which data have, and processing a command corresponding to the comparison result.

[0044] A detection means by which an information processor according to claim 23 detects the command from a predetermined user, A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A processing means The number corresponding to the sequence memorized is assigned to the data memorized by block. A storage means The number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block are memorized. It is characterized by memorizing new data to a top block, when the block which has the last number is the last block, and memorizing new data to the block next to the block which has the last number, when the block which has the last number is not the last block.

[0045] A detection means by which an information processor according to claim 27 detects the command from a predetermined user, A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing two or more data which specify an access privilege different, respectively to the 1st field to each user as opposed to the predetermined field in the 2nd field.

[0046] A detection means by which an information processor according to claim 30 detects the command from a predetermined user (for example, BPSK demodulator circuit 62 of drawing 3), A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means (for example, BPSK modulation circuit 68 of drawing 3), Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.

[0047] A detection means by which an information processor according to claim 33 detects the command from a predetermined user (for example, BPSK demodulator circuit 62 of drawing 3), A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means (for example, BPSK modulation circuit 68 of drawing 3), Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.

[0048] Drawing 1 shows an example using R/W1 and IC card 2 of a non-contact card system. Using an electromagnetic wave, R/W1 and IC card 2 are non-contact, and transmit and receive data.

[0049] If R/W1 transmits a predetermined command to IC card 2, IC card 2 receives the command and is made as [perform / processing corresponding to the command].

[0050] If R/W1 transmits data to IC card 2, the command is received, and IC card 2 which is the gestalt of 1 operation of the information processor of this invention processes the received command, and is made as [transmit / to R/W1 / the response data corresponding to the processing result].
[0051] Moreover, it connects with a controller 3 through a predetermined interface (for example, RS-485A), and a predetermined control signal is supplied to R/W1 from a controller 3, and it is made as [process] according to the control signal.
[0052] Drawing 2 shows the configuration of R/W1.
[0053] In IC21 Processing of data The communication link with SPU (Signal Processing Unit)32 and the controller 3 which process the data received from the data and IC card 2 which are transmitted to DPU (Data Processing Unit)31 to perform and IC card 2 It is ** to SCC (Serial Communication Controller)33 which carries out, and processing of data. The ROM section 41 which has memorized **** information beforehand, The memory section 34 which consists of RAM sections 42 which memorize the data in the middle of processing temporarily is connected through the bus.
[0054] Moreover, the flash memory 22 which memorizes predetermined data is also connected to this bus.
[0055] DPU31 is made as [receive / from SPU32 / the response data received from IC card 2] while outputting the command transmitted to IC card 2 to SPU32.
[0056] SPU32 is made in the response data transmitted with IC card 2 as [perform / to reception and its data / from a demodulator circuit 25 / predetermined processing] while outputting to a modulation circuit 23, after performing predetermined processing (for example, BPSK (BiPhase Shift Keying) modulation (after-mentioned)) to the command transmitted to IC card 2.
[0057] A modulation circuit 23 is data to which the subcarrier of the predetermined frequency (for example, 13.56MHz) supplied from the oscillator 26 was supplied from SPU32, and is made as [output / through an antenna 27 / the generated modulated wave / carry out an ASK (AmplitudeShift Keying) modulation and / to IC card 2 / as an electromagnetic wave]. At this time, a modulation circuit 23 makes a modulation factor less than one, and performs an ASK modulation. That is, when data are a low level, it is made for the maximum amplitude of a modulated wave not to become zero.
[0058] The demodulator circuit 24 is made as [output / restore to the modulated wave (ASK modulated wave) which received through the antenna 27, and / to SPU32 / the data to which it restored].
[0059] Drawing 3 shows the example of a configuration of IC card 2. In this IC card 2, IC51 is made as [receive / through an antenna 53 / the modulated wave transmitted by R/W1]. In addition, a capacitor 52 constitutes LC circuit with an antenna 53, and is made as [side / with the electromagnetic wave of a predetermined frequency (carrier frequency)].
[0060] In IC51 RF interface section 61 While detecting the modulated wave (ASK modulated wave) which received through the antenna 53, getting over in the ASK recovery section 81 and outputting the data after a recovery to the BPSK demodulator circuit 62 and the PLL (Phase Locked Loop) section 63 At a voltage regulator 82, it is ASK **. The signal which the tone section 81 detected is stabilized and it is made as [supply / each circuit / as direct current power].
[0061] Moreover, RF interface section 61 oscillates the signal of the same frequency as the clock frequency of data in an oscillator circuit 83, and is made as [output / to the PLL section 63 / the signal].

[0062] And the ASK modulation section 84 of RF interface section 61 Corresponding to the data supplied from operation part 64, fluctuate the load of the antenna 53 as a power source of IC card 2. (For example, correspond to data and a predetermined switching element is made to turn on / turn off, and only when a switching element is an ON state, a predetermined load is connected to an antenna 53 at juxtaposition) By things The modulated wave which has received through an antenna 53 (when transmitting data from IC card 2) maximum amplitude of a modulated wave is fixed -- **** -- an ASK modulation is carried out and the modulation component is transmitted to R/W1 through an antenna 53 -- it is made like (the terminal voltage of the antenna 27 of R/W1 is fluctuated).
[0063] From the data supplied from the ASK recovery section 81, the PLL section 63 generates the clock signal which synchronized with the data, and is made as [output / to the BPSK demodulator

circuit 62 and the BPSK modulation circuit 68 / the clock signal].
[0064] The BPSK demodulator circuit 62 is made as [output / according to the clock signal supplied from the PLL section 63, / restore to the data and / to operation part 64 / the data to which it restored], when the BPSK modulation of the data to which it restored in the ASK recovery section 81 is carried out.
[0065] When the data supplied from the BPSK demodulator circuit 62 are enciphered, operation part 64 is made as [process / the data / by the sequencer 91 / as a command], after decrypting the data in a code / decode section 92. In addition, when data are not enciphered, the data supplied from the BPSK demodulator circuit 62 are directly supplied to a sequencer 91 without a code / decode section 92.
[0066] The sequencer 91 is made as [perform / processing corresponding to the supplied command]. For example, a sequencer 91 processes the data memorized by EEPROM66 at this time.
[0067] The parity operation part 93 of operation part 64 is made as [compute / a Reed Solomon code] as parity from the data memorized by EEPROM66 and the data memorized by EEPROM66.
[0068] Furthermore, operation part 64 is made as [output / to the BPSK modulation circuit 68 / the response data (data transmitted to R/W1) corresponding to the processing], after performing processing predetermined by the sequencer 91.
[0069] The BPSK modulation circuit 68 carries out the BPSK modulation of the data supplied from operation part 64 (after-mentioned), and is made as [output / to the ASK modulation section 84 of RF interface section 61 / the data after a modulation].
[0070] RAM67 is made as [memorize / the data in the middle of processing etc. / temporarily], when a sequencer 91 processes.
[0071] EEPROM (Electrically Erasable and Programmable ROM)66 is the memory of a non-volatile, and even after IC card 2 ends the communication link with R/W1 and an electric power supply stops, it is made as [memorize / data / continue]. The fundamental program required for a sequencer 91 to process the command from R/W1 is memorized by ROM65.
[0072] Drawing 4 shows an example of assignment of the memory of EEPROM66.
[0073] EEPROM66 has 40 bytes of 256 physical blocks. Each physical block consists of a total of 40 bytes of 32 bytes of data division (D00 thru/or D1f), 2 bytes of attribute section (AT1, AT2), and 6 bytes of parity section (P0 thru/or P5).
[0074] The physical block number ffH (H expresses the hexadecimal) of EEPROM66 is assigned to the system ID block. The system ID block has memorized the information about the security of IC card 2.
[0075] Next, the physical block is assigned to the common area definition block (Common Area Definition Block) (the 1st field) or the provider domain-defined block (Provider Area Definition Block) (the 1st field) one by one toward 00H from the physical block number fdH.
[0076] When IC card 2 is published by EEPROM66, those (provider) who offer the system using this IC card 2 with predetermined equipment (issue machine) are registered into it. An issue machine is 1 physical block per one provider, carries out sequential use of the provider domain-defined block toward 00H from the physical block number fdH, and registers a provider.
[0077] The common area definition block and the provider domain-defined block have memorized the information on the location of the storage region which a provider uses etc.
[0078] And the physical block which is not used as a system ID block, a common area definition block, and a provider domain-defined block is assigned to the user block (User Block) used by the provider.
[0079] Drawing 5 shows an example of the assignment of each data to a system ID block.
[0080] As for D00 thru/or D0f of data division, the manufacture ID at the time of manufacture of EEPROM66 (Manufacture ID) (IDm) is memorized. A field D00 thru/or D03, a field D04 or D07, a field D08 or D0b and field D0c thru/or D0f have memorized the IC code of EEPROM66, the code (Manufacture Equipment Code) of the manufacture machine which created EEPROM66, the manufacture date (Manufacture Date) of EEPROM66, and the manufacture serial number (Manufacture Serial Number) of EEPROM66, respectively.
[0081] By using this information on IDm, all IC cards 2 (EEPROM66) are discriminable. In addition, a manufacture date sets January 1, 2000 to 0000H, and makes it the days from January 1, 2000. In

addition, when a manufacture date is the 1990 set, a manufacture date is expressed as negative days from January 1, 2000 using a two's complement.

[0082] The issue ID (Issue ID) (IDi) when D10 thru/or D1f of data division publishes this ID card 2 is memorized. A field D10 thru/or D13, a field D14 or D17, a field D18 or D1b and field D1c thru/or 1f of the codes of the issue machine which published the category / group number which shows the category and group to whom IC card 2 belongs, and this IC card 2, the dates which published IC card 2, and the expiration dates of IC card 2 are memorized, respectively.

[0083] Drawing 6 shows the attribute section of a system ID block. The attribute section has memorized the number of providers registered. In case an issue machine registers one provider, one physical block is used for it and it updates the value of this attribute section then.

[0084] The value of the attribute section is set as zero at the time of manufacture, and when an issue machine registers a provider into IC card 2 after that, it updates the value of the attribute section by the number of providers registered.

[0085] The parity section of a system ID block has memorized the Reed Solomon code (RS sign) calculated by the parity operation part 93 from the value of each bit of data division and the attribute section. Therefore, the value of the parity section is recalculated whenever data division or the attribute section is updated.

[0086] Drawing 7 shows an example of a common area definition block and a provider domain-defined block. In addition, these blocks are beforehand written in by the issue machine, when IC card 2 is published.

[0087] The common area definition block has been arranged at the physical block number feH of EEPROM66, and has memorized a setup of the storage region (common area (Common Area)) (the 2nd field) used by all providers.

[0088] From the physical block number fdH of EEPROM66, the provider domain-defined block has been arranged toward 00H, is 1 physical block per one provider, and has memorized a provider's information.

[0089] As shown in drawing 7 , the data division D00 of a domain-defined block (a common area definition block and provider domain-defined block) thru/or the fields D00 and D01 of D1f have memorized the provider code (Provider Code) which shows provider's class. In the common area definition block, the value of fields D00 and D01 is set to 0000H, and, in the provider domain-defined block, let the value of fields D00 and D01 be the value of either 0001H thru/or FFFFH.

[0090] The field D02 of the data division of a domain-defined block thru/or D05 have memorized the allocation table (Allocation Table) which consists of numbers BN1 (fields D04 and D05) (BN1> BN0) of the next physical block of the number BN0 (fields D02 and D03) of the physical block of the head of the storage region (provider field (Provider Area)) (the 2nd field) which this provider uses, and the number of the physical block of an end. a provider field is set as the position (the physical block number BN0 -- or (BN 1-1)) of EEPROM66 except a system block (a system ID block, domain-defined block), as shown in drawing 8 .

[0091] Thus, since the provider field is specified by BN0 and BN1, it is not the area size used by the provider (user), and for the information on the amount proportional to the number of providers, data can be managed and use effectiveness of memory can be made high.

[0092] The field D06 of the data division of a domain-defined block thru/or D09 have remembered the partition table (Partition Table) which consists of the block counts BRW (fields D08 and D09) of the read/write block in a random access field to be block count BRA (fields D06 and D07) of a random access field (after-mentioned) among the storage regions which a provider uses. At this time, block count BRA of a random access field is set as formula BRA=0 or the value with which are satisfied of formula 2xn<=BRA<=BN1-BN0 (n is the number of light buffers (after-mentioned)), and the block count BRW of a read/write block is set as BRW=0, when it is BRA=0, and when it is BRA!=0, it is set as the value with which are satisfied of formula n<=BRW<=BRA-n.

[0093] Field D0a of the data division of a domain-defined block and D0b have memorized several n of the light buffer of a random access field. A n light buffer is used when making coincidence memorize n

data to logical-block number 00H thru/or (00+n (hexadecimal display)) H of a random access field. In addition, when memorizing data to the physical block which has other logical-block numbers among random access fields, only one light buffer is used.
[0094] as mentioned above, according to a domain-defined block, it is shown in drawing 8 -- as -- the physical block number BN0 -- a field (a provider field or common area) is assigned to the provider specified in provider code, further, the physical block of the BRA individual of the field (a provider field or common area) is assigned to a random access field, and the remaining physical blocks are assigned to the sequential access field (after-mentioned) for or (BN 1-1).
[0095] Furthermore, according to the domain-defined block, as shown in drawing 8 , the random access field is logically assigned to the read/write block of a BRW individual, the read-only block, and the n light buffer. In addition, a read/write block and physical blocks other than a light buffer are assigned to a read-only block.
[0096] Field D0c of the data division of a domain-defined block and D0d have memorized the Perth block permission which has the information on the access privilege to the Perth block (Purse Block) (after-mentioned) in the storage region (random access field) which this provider uses.
[0097] Drawing 9 shows an example of the Perth block permission.
[0098] The Perth block permission (16 bits, b0, or bf) is read to the Perth block, and shows authorization or the disapproval of an add instruction and a subtraction instruction.
[0099] The Perth block permission of a common area definition block has memorized whether the Perth block is used in the storage region (common area) set up with a common area definition block to Field (bit) bb. That is, in the case of bb=0, the Perth block is not used. In the case of bb=1, the Perth block is used. And especially the field (bit) of others in the Perth block permission of a common area definition block is not used. In addition, in the case of bb=1, the read/write block whose logical-block number is 00H is used as a Perth block.
[0100] Next, in the Perth block permission of a provider domain-defined block, it has memorized whether the Perth block is used in the storage region set up with this provider domain-defined block to the field b3. That is, in the case of b3=0, the Perth block is not used. In the case of b3=1, the Perth block is used. In addition, in the case of b3=1, the read/write block whose logical-block number is 00H is used as a Perth block.
[0101] And the propriety of an add instruction to the Perth block was memorized to the field b2, the propriety of a subtraction instruction to the Perth block was memorized to the field b1, and the propriety of read-out to the Perth block is memorized to the field b0 (in the case of bi=1 (1 i= 0, 2), the instruction is permitted and, in the case of bi=0, the instruction is not permitted). Moreover, it has memorized to Field bb whether the Perth block is used in the storage region set up with a common area definition block. In addition, the same value as bb of the Perth block permission of a common area definition block is memorized by bb.
[0102] Furthermore, the propriety of an add instruction to the Perth block was memorized to Field ba, the propriety of a subtraction instruction to the Perth block was memorized to the field b9, and the propriety of read-out to the Perth block is memorized to the field b8 (in the case of bi=1 (8 i= 9a), the instruction is permitted and, in the case of bi=0, the instruction is not permitted).
[0103] Field D0e of the data division of a domain-defined block of drawing 7 and D0f memorized the version number of the security key (a common key and provider key) used for encryption and a decryption at a provider's (R/W1) authentication, and a list, and a field D10 thru/or 1f of the security key are memorized.
[0104] In addition, when R/W1 polls, IC card 2 returns the version number of these two keys (a common key and provider key). Therefore, in authentication between R/W1 and IC card 2, the security key of two or more versions can be used properly.
[0105] And the attribute sections AT1 and AT2 of a domain-defined block are formed as a reserve, and especially information is not memorized. The parity section of a domain-defined block has memorized the parity (RS sign) calculated from the value of all the bits of data division and the attribute section.
[0106] Thus, the domain-defined block set up by the issue machine has memorized a provider code, an

allocation table, a partition table, the Perth block permission, the security key version, and the security key.

[0107] Drawing 10 shows an example of a user block. As mentioned above with reference to drawing 4 , physical blocks other than a system ID block, a common area definition block, and a provider domain-defined block are used by the provider as a user block among the rooms of EEPROM66.

[0108] For example, if eight providers are registered when room consists of 256 blocks as shown in drawing 4 , 246 (= 256-10) blocks of those other than the system block of a total of ten (= 1+1+8) individuals of a system ID block, a common area definition block, and eight provider domain-defined blocks will be used as a user block. Moreover, if 40 providers are registered, a system block will become a total of 42 (= 1+1+40) individuals, and the user block of 214 (= 256-42) individuals will be secured.

[0109] A user block is assigned to each provider according to the allocation table (drawing 7) of a domain-defined block. In addition, since a provider uses the user block currently assigned beforehand with reference to an allocation table, he does not access other than the field (a provider field or common area) assigned on the allocation table.

[0110] The user block of the field (a provider field or common area) assigned on the allocation table is assigned to the random access field and the sequential access field according to the above-mentioned partition table (drawing 7).

[0111] Furthermore, the user block of a random access field is used as either a read/write block, a read-only block and a light buffer, and the number of these blocks is set up as mentioned above according to the number of a partition table and light buffers.

[0112] Thus, the data division D00 thru/or D1f of a user block currently assigned is used according to processing by the provider to whom the user block is assigned.

[0113] The attribute section of a user block of a random access field has memorized the incremental counter (Incremental Counter) (bits bf and be) and the logical-block number (Bit bd thru/or b0), as shown in drawing 11 .

[0114] A logical-block number and an incremental counter are used when accessing the user block of a random access field.

[0115] When reading the data memorized to the random access field, it is a logical-block number, and the data (physical block) to read are searched and the newest data are read with reference to the incremental counter of the data which have the logical-block number.

[0116] On the other hand, when memorizing data to a random access field, after using as a light buffer the physical block (after-mentioned) which became unnecessary with reference to the logical-block number and incremental counter of data which have already been memorized to the random access field, data are written in the light buffer.

[0117] In addition, when the Perth block permission of an above-mentioned domain-defined block is set up so that the Perth block may be used, the read/write block whose logical-block number is 00H is used as a Perth block.

[0118] The Perth block is used to read the value already memorized when performing addition and subtraction of data frequently when setting up the access privilege to data finely (since possibility that information will be revealed increases).

[0119] Drawing 12 shows an example of the Perth block. The data division D00 of the Perth block the field D00 of D1f thru/or D07 are used as Perth data division. The data division D00 of the Perth block the field D08 of D1f thru/or D0f have memorized Execution ID (Execution ID). In addition, the field D10 thru/or D1f of data division of the Perth block is set up read-only, although used as user data division.

[0120] The Perth data division have memorized predetermined data. Execution ID is referred to when the add instruction or subtraction instruction to the Perth block is executed, and it is compared with the execution ID contained in the add instruction or a subtraction instruction.

[0121] On the other hand, the attribute section of a user block of a sequential access field has memorized the lap round number (Bit bf thru/or b0), as shown in drawing 13 . In the sequential access field, if data (sequentially) are memorized in an order from the physical block of the head of a field and data are memorized to the physical block of the last of a field, data are again memorized in an order from the physical block of the head of a field (overwritten). The lap round number has memorized the sequence.

[0122] Therefore, when accessing the user block of a sequential access field, while being used, when memorizing data to a sequential access field, sequential reference of the lap round number is carried out. And data are memorized by the next physical block of the physical block which has a lap round number at the tail end till then. At this time, the lap round number of the physical block data were remembered to be is set as the number which added 1 to the lap round number at the tail end till then.

[0123] In addition, a failure occurs in the middle of a store at the time of the last store, and when the parity error (physical memory colla tempestade PUSHON) has arisen in the physical block which has a lap round number at the tail end, new data are memorized by the physical block, for example. Moreover, new data are memorized by the physical block of the head of a sequential access field when the physical block which has a lap round number at the tail end is a physical block of a sequential access end-of-region rate.

[0124] As mentioned above, EEPROM66 is suitably used for each provider.

[0125] Next, with reference to the flow chart of drawing 14 , and the timing chart of drawing 15 , actuation of IC card 2 and R/W1 is explained.

[0126] First, a predetermined electromagnetic wave is emitted from an antenna 27, the loaded condition of an antenna 27 is supervised, IC card 2 approaches, and R/W1 corresponding to the provider registered into IC card 2 in step S1 stands by until change of loaded condition is detected. In addition, R/W1 emits the electromagnetic wave which carried out the ASK modulation by the data of a short predetermined pattern, and you may make it repeat the appeal to IC card 2 in step S1 until the response from IC card 2 is obtained in fixed time amount.

[0127] When R/W1 detects approach of IC card 2 in step S1 (time of day t0 of drawing 15), it progresses to step S2. SPU32 of R/W1 The square wave of a predetermined frequency (a clock frequency twice the frequency [for example,] of data) as shown in drawing 16 (a) is made into a subcarrier. By the data (command corresponding to the processing which IC card 2 is made to perform) (for example, data shown in drawing 16 (b)) transmitted to IC card 2, a BPSK modulation is performed and the generated modulated wave (BPSK modulating signal) (drawing 16 (c)) is outputted to a modulation circuit 23.

[0128] In addition, as shown in drawing 16 (c) using differential conversion at the time of a BPSK modulation A value makes the same thing as the last BPSK modulating signal ("1", "0" or "0", "1") a BPSK modulating signal, when the data of 0 appear. The value makes what reversed the phase of the last BPSK modulating signal (thing which made "0" reverse "1" and made "1" reverse "0") the BPSK modulating signal, when the data of 1 appear.

[0129] Thus, since it gets over to the original data also when a BPSK modulating signal is reversed by holding data by change of the phase of a modulated wave using differential conversion, when getting over, the need of considering the polarity of a modulated wave is lost.

[0130] And a modulation circuit 23 is the BPSK modulating signal, the ASK modulation of the predetermined subcarrier is carried out with less than (for example, 0.1) one modulation factor (= maximum amplitude of the maximum amplitude/subcarrier of a data signal), and the generated modulated wave (ASK modulated wave) is transmitted to IC card 2 through an antenna 27 (during the time of day t0 of drawing 15 thru/or time of day t1).

[0131] In addition, when not transmitting, the modulation circuit 23 is made as [generate / with the high level of the two level (high level and low level) of a digital signal / a modulated wave].

[0132] Next, in step S3, IC cards 2 are an antenna 53 and a capacitor 52, transform into an electrical signal a part of electromagnetic wave which the antenna 27 of R/W1 emitted, and output the electrical signal (modulated wave) to RF interface section 61 of IC51. and the ASK recovery section 81 of RF interface section 61 controls the dc component of the generated signal, extracts a data signal, and outputs the data signal to the BPSK demodulator circuit 62 and the PLL section 63 while it supplies the signal generated in the modulated wave by rectifying and carrying out smooth (namely, envelope detection -- carrying out) to a voltage regulator 82.

[0133] A voltage regulator 82 stabilizes the signal supplied from the ASK recovery section 81, generates direct current power, and supplies it to each circuit.
[0134] In addition, the terminal voltage V0 of an antenna 53 is as follows, for example at this time.
$V0=V10(1+kxVs (t)) cos (omegat)$
[0135] It is here, and in V10, k shows a modulation factor and Vs (t) shows the signal component for the amplitude of a carrier component, respectively.
[0136] Moreover, the value VLR of a low level in the electrical potential difference V1 after rectification by the ASK recovery section 81 is as follows, for example.
$VLR=V10(1+kx (-1))-Vf$[0137] Here, Vf shows the voltage drop in the diode D of a rectifier circuit. Usually, Vf is about 0.7 volts.
[0138] And a voltage regulator 82 stabilizes rectification and the signal by which smooth was carried out by the ASK recovery section 81, and supplies it to each circuits including operation part 64 as direct current power. In addition, since the modulation factor k of a modulated wave is less than one, its voltage variation after rectification (difference of high level and a low level) is small. Therefore, a voltage regulator 82 can generate direct current power easily.
[0139] When a modulation factor k receives 5% of modulated wave so that V10 may become 3 volts or more, for example, the low-level electrical potential difference VLR after rectification It becomes more than 2.15 (=3x(1-0.05)-0.7) bolts. A voltage regulator 82 While being able to supply electrical potential difference sufficient as a power source to each circuit, amplitude 2xkxV10 (Peak-to-Peak value) of the alternating current component (data component) of the electrical potential difference V1 after rectification It becomes more than 0.3 (= 2x0.05x3) bolts, and the ASK recovery section 81 can restore to data by the sufficiently high S/N ratio.
[0140] Thus, when a modulation factor k uses less than one ASK modulated wave, while performing the communication link with a low (in high condition of a S/N ratio) error rate, direct current voltage sufficient as a power source is supplied to IC card 2.
[0141] And according to the clock signal supplied from the PLL section 63, the BPSK demodulator circuit 62 restores to the data signal (BPSK modulating signal) from the ASK recovery section 81, and outputs the data to which it restored to operation part 64.
[0142] Next, in step S4, when the data supplied from the BPSK demodulator circuit 62 are enciphered, after decrypting operation part 64 in a code / decode section 92, it supplies the data (command) to a sequencer 91, and performs processing corresponding to the command (during the time of day t1 of drawing 15 thru/or time of day t2). In addition, while the value had transmitted the data of 1, R/W1 is standing by, until it receives the answerback from this period 2, i.e., an IC card. Therefore, in this period, IC card 2 has received the modulated wave with fixed maximum amplitude.
[0143] Next, in step S5, the sequencer 91 of operation part 64 outputs data (data transmitted to R/W1), such as a processing result, to the BPSK modulation circuit 68. Like SPU32 of R/W1, after the BPSK modulation circuit 68 carries out the BPSK modulation of the data, it is outputted to the ASK modulation section 84 of RF interface section 61.
[0144] The ASK modulation section 84 and by fluctuating the load connected to the both ends of an antenna 53 according to data using a switching element The modulated wave which has received (in the time of transmission of IC card 2) the maximum amplitude of a modulated wave becomes fixed -- **** -- an ASK modulation is carried out according to the data to transmit, the terminal voltage of the antenna 27 of R/W1 is fluctuated according to it, and the data is transmitted to R/W1 (during the time of day t2 of drawing 15 thru/or time of day t3).
[0145] In step S6, as for the modulation circuit 23 of R/W1, the value is continuing transmission of the data of 1 (high-level) at the time of reception of the data from IC card 2. And a demodulator circuit 25 detects the data transmitted with IC card 2 from minute fluctuation (for example, dozens of microvolts) of the terminal voltage of the antenna 27 of IC card 2, and the antenna 27 combined in electromagnetism.
[0146] And it gets over and a demodulator circuit 25 outputs the generated digital data to SPU32, after amplifying the detected signal (ASK modulated wave) with the amplifier of high interest profit.

[0147] And in step S7, after restoring to the data (BPSK modulating signal), it outputs to DPU31 and, as for SPU32 of R/W1, DPU31 processes the data (during the time of day t3 of drawing 15 thru/or time of day t4).

[0148] Furthermore, in step S8, when it judges whether a communication link is ended according to a processing result and it is judged again that it communicates, to step S2, DPU31 of R/W1 is return, step S2, or step S7, and communicates the following data (command) (time of day t4 thru/or time of day t8 of drawing 15). On the other hand, when it is judged that a communication link is ended, R/W1 ends the communication link with IC card 2.

[0149] As mentioned above, using the ASK modulation whose modulation factor k is less than one, R/W1 transmits a predetermined command to IC card 2, and IC card 2 performs reception and processing corresponding to the command for the command, and it returns the data corresponding to the result of the processing to R/W1.

[0150] Next, the actuation when writing in data to EEPROM66 as an example of processing by IC card 2 in above-mentioned step S4 is explained with reference to the flow chart of drawing 17 thru/or drawing 21 .

[0151] First, with reference to the flow chart of drawing 17 thru/or drawing 19 , the actuation when writing data in the random access field of EEPROM66 is explained.

[0152] In step S21, a sequencer 91 judges whether it is that the physical block which writes in data is a read/write block (the Perth block is not included) (the block to a BRW individual is considered as a read/write block from BN0 at sequence as shown in drawing 8), and when it is judged that it is a read/write block, it progresses to step S22.

[0153] A sequencer 91 progresses to step S23 (drawing 18), when it judges whether it is using the Perth block (b3=1) and the Perth block is not being used with reference to the Perth block permission (drawing 9) of a provider domain-defined block which has the provider code of R/W1 (in the case of b3=0).

[0154] On the other hand, when it is judged that the Perth block is used in step S22, a sequencer 91 judges whether the read/write block which writes in whether the logical-block number of the data (it writes in) to memorize is 00H and data in step S24 has lapped with the Perth block, and when it is judged that the read/write block which writes in data has not lapped with the Perth block, it progresses to step S23.

[0155] When it is judged that the read/write block which writes in data has lapped with the Perth block, a sequencer 91 ends processing in step S25, after performing error processing.

[0156] Moreover, when it is judged that the physical block which writes in data in step S21 is not a read/write block, it progresses to step S26, the physical block in which a sequencer 91 writes data judges whether it is the Perth block, and when it is judged that it is the Perth block, it progresses to step S27.

[0157] When it is judged that the physical block which writes in data is not the Perth block, a sequencer 91 ends processing in step S28, after performing error processing.

[0158] In step S27, in a random access field, a sequencer 91 progresses to step S29, when the Perth block (physical block whose logical-block number is 00H) is looked for and the Perth block is discovered.

[0159] Since the store to the Perth block cannot be performed when the Perth block is not discovered at step S27, a sequencer 91 ends processing in step S30, after performing error processing.

[0160] or [next, / that the add instruction is permitted with reference to the Perth block permission of a provider domain-defined block in step S29 by the instruction (command) to the Perth block judging whether it is an add instruction, and a sequencer 91 progressing to step S31 when it is judged that it is an add instruction] (b2=1) -- it judges whether it is no.

[0161] And when a sequencer 91 judges that the add instruction to the Perth block is permitted at step S31, it progresses to step S23.

[0162] On the other hand, when it is judged that the add instruction to the Perth block is not permitted at step S31, a sequencer 91 ends processing, after performing error processing in step S32, without executing an add instruction (when it is b2=0).

[0163] Moreover, in step S29, when the instruction to as opposed to [when it is judged that the instruction to the Perth block is not an add instruction / progress to step S33 and] the Perth block in a sequencer 91 judges that it judges whether it is a subtraction instruction and is a subtraction instruction, it progresses to step S34.
[0164] And in step S34, a sequencer 91 judges whether it is that the subtraction instruction is permitted (b1=1) with reference to the Perth block permission of a provider domain-defined block, and when it is judged that the subtraction instruction to the Perth block is permitted, it progresses to step S23.
[0165] On the other hand, when it is judged that the subtraction instruction to the Perth block is not permitted at step S34, a sequencer 91 ends processing, after performing error processing in step S35, without executing a subtraction instruction (when it is b1=0).
[0166] Moreover, in step S33, when it is judged that the instruction to the Perth block is not a subtraction instruction, a sequencer 91 ends processing in step S36, after performing error processing.
[0167] Next, in step S23 of drawing 18 , a sequencer 91 searches the physical block of a random access field, and looks for the physical block which has the same logical-block number as the logical-block number of the data which write in.
[0168] And in step S37, a sequencer 91 judges whether the number of the physical blocks discovered at step S23 is two. That is, in this system, the last data and the data before last are memorized at least about each logical block. And when memorizing still newer data, new data are memorized on the data before last (it may memorize on the data of other logical-block numbers before last). When two physical blocks of the same logical-block number exist, it progresses to step S38, and the value (00, 01, 10, or 11) of the incremental counter in the two physical blocks is read and compared.
[0169] And make a physical block with the large value of an incremental counter into the physical block (new physical block) new data are remembered to be, and let a physical block with the small value of an incremental counter be the physical block (old physical block) old data are remembered to be.
[0170] However, when the values of two incremental counters are 00 and 11, make into a new physical block the physical block whose value of an incremental counter is 00, and let the physical block whose value of an incremental counter is 11 be an old physical block.
[0171] A sequencer 91 memorizes the number (physical block number) of a new physical block to RAM67 as a variable Y between two physical blocks, and makes RAM67 memorize the number of an old physical block in step S39 as a variable W (number of the physical block used as a light block).
[0172] Thus, a sequencer 91 progresses to step S49, after making Variable Y and Variable W memorize.

[0173] On the other hand, when it is judged that the number of the physical blocks discovered at step S23 is not two in step S37, it progresses to step S40 and a sequencer 91 judges whether the number of the physical blocks discovered at step S23 is one. And when it is judged that it is one piece, it progresses to step S41.
[0174] In step S40, when a sequencer 91 judges that the number of the physical blocks discovered at step S23 is not one, after performing error processing, processing is ended in step S42.
[0175] As for saying [that the same logical block exists only in one piece], the data before last will not exist for a certain reason. Then, in this case, it is the physical block of other logical-block numbers, and the physical block (namely, physical block whose number of the physical blocks which have the same logical-block number is two) which has data the last and before last is searched, and the physical block before last of them is used as a light block. For this reason, in step S41, a sequencer 91 progresses to step S43, after making RAM67 memorize the number of the discovered physical block (one piece) as a variable Y.
[0176] In step S43, a sequencer 91 searches the physical block of a random access field, and looks for two physical blocks which have the same predetermined logical-block number (arbitration) (logical-block number unrelated to the logical-block number now made into the write-in object).
[0177] In addition, since it retrieves sequentially from logical-block number 00H when searching a physical block, a smaller number, then retrieval time can be shortened for the logical-block number of the data which perform write-in processing frequently.

[0178] In step S44 and a sequencer 91 It judges whether the physical block whose logical-block number is the two same pieces was discovered at step S43. When it is judged that it was discovered, the incremental counter of two physical blocks discovered by progressing to step S45 is referred to. It progresses to step S49 (drawing 19), after making RAM67 memorize the number of the physical block of the older one as a variable W (number of a light block) between two physical blocks.
[0179] On the other hand, when it is judged in step S44 that two physical blocks were not discovered at step S43, it progresses to step S46, and a sequencer 91 carries out sequential count of the parity of each physical block of a random access field, and looks for the physical block which has started the parity error as compared with the value memorized by the parity section of each physical block.
[0180] And when it judges whether there is any physical block which has started the parity error and it is judged that there is a physical block which has started the parity error, it progresses to step S47, and a sequencer 91 progresses to step S49, after making RAM67 memorize the number of the physical block as a variable W (number of a light block).
[0181] In step S46, when it is judged that there is no physical block which has started the parity error, a sequencer 91 ends processing in step S48, after performing error processing.
[0182] In step S49 of drawing 19 next, a sequencer 91 It judges whether the physical block which writes in data is the Perth block (physical block whose logical-block number is 00H). When it is judged that it is the Perth block, the execution ID of the instruction which progresses to step S50 and is performed to the Perth block When it judges whether it is the same as that of the execution ID of the physical block of the number memorized as a variable Y at step S39 or step S41 (drawing 12) and it is judged that it is the same, it judges that this instruction is already processed and processing is ended.
[0183] Thus, since IC card 2 does not process the command when R/W1 carries out the retry of the same command by using Execution ID and the command is already processed, the same command is not processed twice.
[0184] In step S50, when the instruction with which a sequencer 91 is performed to the Perth block in step S51 when the execution ID of the instruction performed to the Perth block judges that it is not the same as that of the execution ID of the physical block of the number memorized as a variable Y judges whether it is an add instruction and it is an add instruction, it progresses to step S52.
[0185] In step S52, a sequencer 91 reads the Perth data of the physical block of the number of Variable Y, calculates the sum of the Perth data and the data contained in the instruction performed to the Perth block, and uses the sum as the Perth data (new Perth data) in new block data. Thus, it progresses to step S54, after processing. In addition, let execution ID of the physical block of the number of Variable Y be the execution ID of new block data at this time. This prevents processing of a duplex.
[0186] On the other hand, when it is judged that the instruction performed to the Perth block is not an add instruction in step S51 (that is, it is a subtraction instruction), it progresses to step S53, and a sequencer 91 reads the Perth data of the physical block of the number of Variable Y, calculates the difference of the Perth data and the data contained in the instruction performed to the Perth block, and uses the difference as the Perth data (new Perth data) in new block data. Thus, it progresses to step S54, after processing. In addition, let execution ID of the physical block of the number of Variable Y be the execution ID of new block data at this time. This prevents processing of a duplex.
[0187] Moreover, in step S49, a sequencer 91 progresses to step S54, when the physical block which writes in data judges that it is not the Perth block (that is, it is a read/write block).
[0188] And in step S54, a sequencer 91 makes the number which added 1 to the value of the incremental counter of the physical block of the number of Variable Y the value of the incremental counter of new block data. However, when the value of the incremental counter of the physical block of the number of Variable Y is 11, a sequencer 91 sets the value of the incremental counter of new block data to 00.
[0189] Next, in step S55, a sequencer 91 makes the parity of the data newly written in the parity operation part 93, an incremental counter, and a logical-block number calculate, and makes the value of the parity the value of the parity section of new block data.
[0190] And a sequencer 91 makes the physical block (light buffer) of the number of the variable W memorized at either step S39, step S45 or step S47 memorize new block data (the newly memorized

data (for them to be the Perth data and Execution ID in the Perth block), its logical-block number, incremental counters, and such parity) in step S56.
[0191] As mentioned above, since the data of the same logical-block number as the logical-block number of the data are left behind to memory by choosing the physical block (light buffer) which remembers data to be a logical-block number using an incremental counter when a failure occurs in the midst of the store of data, logically, memory colla tempestade PUSHON does not occur.
[0192] Although the incremental counter was used with the gestalt of the above-mentioned implementation in order to distinguish the block with which new data are recorded among the same logical blocks of a random access field, it is also possible to, distinguish the block with which new data are recorded by [at the time of record] securing 4 bytes of field in a random access field, and making time of day (the date, time of day, or value of a counter) then, record to it absolutely for example.
[0193] Next, with reference to the flow chart of drawing 20 and drawing 21 , the actuation when writing data in the sequential access field of EEPROM66 is explained.
[0194] A sequencer 91 makes RAM67 memorize the number of the physical block of the head of a sequential access field as a variable Z in step S61.
[0195] Next, a sequencer 91 reads the lap round number of the physical block whose physical block number is Z, while making RAM67 memorize, it reads the lap round number of the physical block whose physical block number is Z+1, and RAM67 is made to memorize it as a variable B as a variable A in step S62.
[0196] And in step S63, a sequencer 91 judges whether the difference (A-B) of the value of Variable A and the value of Variable B is 1, judges that it is the physical block the physical block of the physical block number Z remembers the data which have a lap round number at the tail end to be when it is not 1, and progresses to step S66.
[0197] When a sequencer 91 judges whether the physical block number Z is the same as the number of the physical block of a sequential access end-of-region rate in step S64 when it is judged that the difference (A-B) of the value of Variable A and the value of Variable B is 1, and it is judged that it is the same, the data with which the physical block of a sequential access end-of-region rate has a lap round number at the tail end are judged to be the physical block to memorize, and it progresses to step S66.
[0198] In step S64, when the physical block number Z judges that it is not the same as that of the number of the physical block of a sequential access end-of-region rate, a sequencer 91 returns to step S62 in step S65, after only 1 makes the value of the variable Z which RAM67 was made to memorize increase. And processing of step S62 thru/or step S65 is repeated successively, changing the value (value of the physical block number to search) of Variable Z.
[0199] Thus, the tail end of the lap round number of the data memorized sequentially is discovered. And in step S66, a sequencer 91 performs the parity check of a block of the number (number of the physical block at the tail end of = lap round number) of Variable Z.
[0200] And in step S67, it judges whether the parity error has produced the sequencer 91 in the physical block, and when it is judged that the parity error has arisen, it progresses to step S68.
[0201] In step S68 a sequencer 91 The value of Variable Z judges whether it is the same as that of the number of the physical block of the head of a sequential access field. When it is judged that it is the same, the tail end of data (what has started the parity error does not contain) judges that it is the physical block of a sequential access end-of-region rate, and sets to step S70. It progresses to step S72 (drawing 21), after making RAM67 memorize the number of the physical block of a sequential access end-of-region rate as a new variable Y.
[0202] When it is judged that the value of Variable Z is not the same as that of the number of the physical block of the head of a sequential access field, after a sequencer 91 makes RAM67 memorize the value (Z-1) which computed the number of the physical block at the tail end of data by having subtracted 1 from the value of Variable Z, and computed it as a variable Y in step S71, it progresses to step S72.
[0203] On the other hand, in step S69, when it is judged that the parity error has not arisen at step S67, a sequencer 91 progresses to step S72, after making RAM67 memorize the number (value of Variable Z in

this case) of the physical block at the tail end of data as a variable Y.
[0204] Next, in step S72, when it judges whether a sequencer 91 has the number (value of Variable Y) of the physical block at the tail end of data, and the same number of the physical block of a sequential access end-of-region rate and it is judged that it is the same, it progresses to step S73.
[0205] And in step S73, it progresses to step S75, after a sequencer's 91 making the number of the physical block of the head of a sequential access field the number of the physical block which writes in new data and making RAM67 memorize by making the number into Variable W.
[0206] It progresses to step S75, after a sequencer's 91 making the number which added 1 to the value of Variable Y the number of the physical block which writes in new data, making the number Variable W in step S74 and making RAM67 memorize, when it is judged in step S72 that the number (value of Variable Y) of the physical block at the tail end of data and the number of the physical block of a sequential access end-of-region rate are not the same.
[0207] Next, in step S75, since the newly memorized data and the data which judge whether the physical block (data at the tail end) of the number of Variable Y is the same, and are newly memorized when the same are already memorized, a sequencer 91 ends processing.
[0208] When it is judged that the physical block (data at the tail end) of the data newly memorized on the other hand and the number of Variable Y is not the same, in step S76, a sequencer 91 reads the lap round number of the physical block of the number of Variable Y, and makes the number which added 1 to the value the lap round number of the data (new block data) newly memorized.
[0209] Next, in step S77, a sequencer 91 makes the parity operation part 93 calculate the parity of the data to memorize and a lap round number (new block data), and writes new block data in it in step S78 at the physical block of a number W.
[0210] Thus, since the data of a lap round number smaller than the lap round number of the data which were being written in in the midst of the store of new data when a failure occurred since the lap round number in the data memorized sequentially is retrieved sequentially and new data are memorized to the next physical block (or physical block of the head of a sequential access field) of the data at the tail end remain, logically, memory colla tempestade PUSHON is not generated.
[0211] As mentioned above, EEPROM66 is made as [control / generating of memory colla tempestade PUSHON] using the information on the attribute section while being able to offer a storage region independently to two or more providers.
[0212] In addition, the same user block can also be assigned to two or more providers. In that case, the same user block is assigned on the allocation table of the provider domain-defined block with which those providers (overlap provider) are registered. at this time, a different access privilege (read/write -- or read-only) for every provider can be set up to the same user block by setting up the partition table of a provider domain-defined block for every provider. Furthermore, a predetermined provider can write in data to the user data division (read-only to other providers) of the Perth block which other providers use by setting up so that the Perth block may not be used to a predetermined provider, and setting up so that the Perth block may be used to other providers.
[0213] Moreover, the value of field D0e of a domain-defined block, and D0f (field where the version number of a security key is usually memorized) By setting it as a predetermined value (for example, FFFFH), and memorizing a predetermined provider's provider code (a maximum of eight pieces) to the field D10 thru/or D1f of a domain-defined block further That provider (local common provider) can use the user block assigned on the allocation table of this domain-defined block as a common area.
[0214] Moreover, the access privilege to the user block can be set up for every local common provider by registering a local common provider into two domain-defined blocks which assign the same user block, and setting up a different access privilege for every domain-defined block.
[0215] Thus, corresponding to two or more providers (namely, R/W), processing according to individual can be performed by setting up an overlap provider and a local common provider.
[0216] In addition, this invention can be applied also when delivering and receiving a signal in the condition (contact) of having been combined physically besides in the case of delivering and receiving a signal through radio (non-contact). In the case of interruption of service, or the equipment which

operates by the cell, when the cell has been removed, data can be secured.
[0217] Moreover, the program which performs each above-mentioned processing is recorded on the transmission medium which consists of record media, such as a magnetic disk and CD-ROM, it provides for a user, or is transmitted to a user through transmission media, such as a network, is recorded on the transmission medium which consists of record media, such as a hard disk and solid-state memory, and can be made to use.
[0218]
[Effect of the Invention] According to the information processing approach according to claim 1, a transmission medium according to claim 3, and the information processor according to claim 4, like the above It is used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. Since a command is processed using the storage section including the 2nd field set as the free area which is not used as the 1st field, use effectiveness of memory can be made high.
[0219] Since the new data which have a predetermined logical-block number were memorized to physical blocks other than the physical block the data which have the logical-block number are remembered to be while assigning the logical-block number to the data memorized by the physical block according to the information processing approach according to claim 6, the transmission medium according to claim 10, and the information processor according to claim 11, generating of memory colla tempestade PUSHON can be controlled logically.
[0220] Since the data of a predetermined block of the 2nd field compare the recognition number which has a recognition number and the command supplied by the user has with the recognition number which data have and it is made not to repeat and process the same command according to the information processing approach according to claim 15, a transmission medium according to claim 17, and the information processor according to claim 18, it can avoid carrying out multiple-times processing accidentally [command / each].
[0221] While assigning the number corresponding to the sequence memorized to the data memorized by block according to the information processing approach according to claim 20, a transmission medium according to claim 22, and the information processor according to claim 23 When the block which has the last number is a block of the last of the assigned field, Since new data were memorized to the top block, and new data were memorized to the block next to the block which has the last number when the block which has the last number was not the last block, generating of memory colla tempestade PUSHON can be controlled logically.
[0222] According to the information processing approach according to claim 25, a transmission medium according to claim 26, and the information processor according to claim 27 Since the command was processed using the storage section which memorizes the predetermined field in the 2nd field, and two or more data which specify an access privilege which is different to one user, respectively to the 1st field Two or more access privileges which can be set to a predetermined storage region can be granted to a predetermined user.
[0223] Since the command was processed using the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field according to the information processing approach according to claim 28, the transmission medium according to claim 29, and the information processor according to claim 30, the same storage region can be assigned to two or more users.
[0224] According to the information processing approach according to claim 31, a transmission medium according to claim 32, and the information processor according to claim 33 Since the command was processed using the storage section which memorizes the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field A different access privilege in a predetermined storage region can be granted to two or more users.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL FIELD

[Field of the Invention] This invention relates to a transmission medium at the information processing approach of receiving the command from a predetermined user in the information processing approach and an information processor, and a list especially, processing the command in them, and transmitting the result of processing to them about a transmission medium and an information processor, and a list.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

PRIOR ART

[Description of the Prior Art] The IC card (smart card) used by the cybermoney system or the security system is developed.
[0003] Such an IC card contains the memory which memorizes CPU which performs various processings, data required for processing, etc., and is transmitting and receiving data in the condition of having made predetermined reader/writer (R/W) contacting.
[0004] Moreover, there is also an IC card of the dc-battery loess mold which does not have the dc-battery in an IC card itself. Power is supplied to the IC card of such a dc-battery loess mold from R/W.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EFFECT OF THE INVENTION

[Effect of the Invention] According to the information processing approach according to claim 1, a transmission medium according to claim 3, and the information processor according to claim 4, like the above It is used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. Since a command is processed using the storage section including the 2nd field set as the free area which is not used as the 1st field, use effectiveness of memory can be made high.
[0219] Since the new data which have a predetermined logical-block number were memorized to physical blocks other than the physical block the data which have the logical-block number are remembered to be while assigning the logical-block number to the data memorized by the physical block according to the information processing approach according to claim 6, the transmission medium according to claim 10, and the information processor according to claim 11, generating of memory colla tempestade PUSHON can be controlled logically.
[0220] Since the data of a predetermined block of the 2nd field compare the recognition number which has a recognition number and the command supplied by the user has with the recognition number which data have and it is made not to repeat and process the same command according to the information processing approach according to claim 15, a transmission medium according to claim 17, and the information processor according to claim 18, it can avoid carrying out multiple-times processing accidentally [command / each].
[0221] While assigning the number corresponding to the sequence memorized to the data memorized by block according to the information processing approach according to claim 20, a transmission medium according to claim 22, and the information processor according to claim 23 When the block which has the last number is a block of the last of the assigned field, Since new data were memorized to the top block, and new data were memorized to the block next to the block which has the last number when the block which has the last number was not the last block, generating of memory colla tempestade PUSHON can be controlled logically.
[0222] According to the information processing approach according to claim 25, a transmission medium according to claim 26, and the information processor according to claim 27 Since the command was processed using the storage section which memorizes the predetermined field in the 2nd field, and two or more data which specify an access privilege which is different to one user, respectively to the 1st field Two or more access privileges which can be set to a predetermined storage region can be granted to a predetermined user.
[0223] Since the command was processed using the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field according to the information processing approach according to claim 28, the transmission medium according to claim 29, and the information processor according to claim 30, the same storage region can be assigned to two or more users.
[0224] According to the information processing approach according to claim 31, a transmission medium according to claim 32, and the information processor according to claim 33 Since the command was processed using the storage section which memorizes the predetermined field in the 2nd field, and two

or more data which specify the access privilege from which two or more users differ, respectively to the 1st field A different access privilege in a predetermined storage region can be granted to two or more users.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] However, in such an IC card, since it is premised on using it in the condition of having made R/W contacting, when using it by non-contact, it has the problem that it is difficult to acquire power.
[0006] Moreover, although how to supply power required for an IC card by the electromagnetic wave is also considered while transmitting and receiving data between an IC card and R/W by non-contact using an electromagnetic wave While having accessed the memory which an IC card contains in such an approach When the receive state of an electromagnetic wave becomes a defect, it has the problem that there is possibility that sufficient power will no longer be obtained and a defect will arise for the adjustment of the data in memory (memory colla tempestade PUSHON (Memory Corruption) arises).
[0007] Furthermore, if information is held like FAT (FileAllocation Table) of MS-DOS (Microsoft-Disc Operating System) in every [data are remembered to be] unit (it is a sector when it is MS-DOS), the field proportional to the area size data are remembered to be is needed for data control, and it has the problem that the use effectiveness of memory falls. Moreover, if a storage region is managed in the predetermined unit data are remembered to be, when memorizing the data of the magnitude with which the unit is not filled, the storage region which is not used occurs and it has further the problem that the use effectiveness of memory falls.
[0008] Furthermore, in the above-mentioned IC card, since uniform processing is performed to R/W, it has the problem that it is difficult to perform processing according to individual corresponding to two or more R/W.
[0009] The 1st field which this invention was made in view of such a situation, and memorizes two or more users' data, While using the storage section including the 2nd field which is used by two or more users memorized to the 1st field, and is managed per physical block of predetermined magnitude A logical-block number is assigned to the data memorized by the physical block. Memorize the data to physical blocks other than the physical block the data which have the logical-block number are remembered to be, or The number corresponding to the sequence memorized is assigned to the data memorized by the physical block. When the physical block which has the last number is the last physical block, By memorizing the data to a top physical block, and memorizing the data to the next physical block of the physical block which has a number at the tail end, when the physical block which has the last number is not the last physical block Generating of memory colla tempestade PUSHON in memory is controlled logically.
[0010] Moreover, this invention is not the area size used by the user, but is the information on the amount proportional to the number of users (the number corresponding to a top physical block, and number corresponding to the last physical block), and enables it to manage data by holding the number corresponding to the physical block of the head of the field used by each user, and the number corresponding to the last physical block.
[0011] Furthermore, a predetermined field [in / on the above-mentioned storage section and / in this invention / the 2nd field], By and the thing for which the data which specify a predetermined field [in / for two or more data which specify an access privilege different, respectively / in memorizing to the 1st

field **** / the 2nd field] corresponding to one user are memorized to the 1st field corresponding to two or more users It enables it to perform processing according to individual corresponding to two or more users (R/W).

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

MEANS

[Means for Solving the Problem] The step to which the information processing approach according to claim 1 detects the command from a predetermined user, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It is characterized by having the step which processes a command using the storage section in which the 2nd field set as the free area which is not used as the 1st field is formed, and the step which outputs the result of processing.
[0013] The step to which a transmission medium according to claim 3 detects the command from a predetermined user, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It is characterized by transmitting a program equipped with the step which processes a command using the storage section in which the 2nd field set as the free area which is not used as the 1st field is formed, and the step which outputs the result of processing.
[0014] A detection means by which an information processor according to claim 4 detects the input signal from the outside, A storage means by which the 2nd field which is used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field and which is set as the free area which is not used as the 1st field is formed, It is characterized by having a processing means to perform processing corresponding to an input signal using a storage means, and an output means to output the result of processing of a processing means outside.
[0015] The information processing approach according to claim 6 is characterized by equipping the step at which a processing means assigns a logical-block number to the data memorized by the physical block, and a storage means with the step memorized to physical blocks other than the physical block the data which have are remembered [data / which have a predetermined logical-block number / new] to be in the logical-block number.
[0016] A transmission medium according to claim 10 is characterized by transmitting the program which the step at which a processing means assigns a logical-block number to the data memorized by the physical block, and a storage means equip with the step memorized to physical blocks other than the physical block the data which have are remembered [data / which have a predetermined logical-block number / new] to be in the logical-block number.
[0017] An information processor according to claim 11 assigns a logical-block number to the data with which a processing means is memorized by the physical block, and a storage means is characterized by memorizing the new data which have a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[0018] The information processing approach according to claim 15 is characterized by for the data of a predetermined block of the 2nd field having a recognition number, comparing the recognition number which the command with which the processing means was supplied by the user has with the recognition number which data have, and equipping them with the step which processes a command corresponding to the comparison result.
[0019] It is characterized by for the data of a predetermined block of the 2nd field having a recognition number in a transmission medium according to claim 17, and for a processing means comparing the recognition number which the command supplied by the user has with the recognition number which

data have, and transmitting a program equipped with the step which processes a command corresponding to the comparison result.
[0020] It is characterized by for the data of a predetermined block of the 2nd field having a recognition number in an information processor according to claim 18, and for a processing means comparing the recognition number which the command supplied by the user has with the recognition number which data have, and processing a command corresponding to the comparison result.
[0021] The step at which, as for the information processing approach according to claim 20, a processing means assigns the number corresponding to the sequence memorized to the data memorized by block, In the storage means which memorized the number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block When the block which has the last number is the last block, new data It is characterized by having the step which memorizes to a top block, and memorizes new data to the block next to the block which has the last number when the block which has the last number is not the last block.
[0022] The step at which a transmission medium according to claim 22 assigns the number corresponding to the sequence memorized to the data with which a processing means is memorized by block, In the storage means which memorized the number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block When the block which has the last number is the last block, new data It is characterized by transmitting a program equipped with the step which memorizes to a top block, and memorizes new data to the block next to the block which has the last number when the block which has the last number is not the last block.
[0023] A detection means by which an information processor according to claim 23 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed. A processing means The number corresponding to the sequence memorized is assigned to the data memorized by block. A storage means The number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block are memorized. It is characterized by memorizing new data to a top block, when the block which has the last number is the last block, and memorizing new data to the block next to the block which has the last number, when the block which has the last number is not the last block.
[0024] The step to which the information processing approach according to claim 25 detects the command from a predetermined user, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And it is characterized by having the step which processes a command, and the step which outputs the result of processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to the 1st field.
[0025] The step to which a transmission medium according to claim 26 detects the command from a predetermined user, As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And it is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing to each user using the storage section which memorizes two or more data which specify an access privilege different, respectively to the 1st field.
[0026] A detection means by which an information processor according to claim 27 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing two or more data which specify an access privilege different, respectively

to the 1st field to each user as opposed to the predetermined field in the 2nd field.
[0027] The step to which the information processing approach according to claim 28 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field is used. It is characterized by having the step which processes a command, and the step which outputs the result of processing.
[0028] The step to which a transmission medium according to claim 29 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field is used. It is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing.
[0029] A detection means by which an information processor according to claim 30 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.
[0030] The step to which the information processing approach according to claim 31 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the predetermined field in the 2nd field and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field is used. It is characterized by having the step which processes a command, and the step which outputs the result of processing.
[0031] The step to which a transmission medium according to claim 32 detects the command from a predetermined user, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and the storage section which memorizes the predetermined field in the 2nd field and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field is used. It is characterized by transmitting a program equipped with the step which processes a command, and the step which outputs the result of processing.
[0032] A detection means by which an information processor according to claim 33 detects the command from a predetermined user, A processing means to process a command, and an output means to output the result of processing of a processing means, Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.
[0033] In the information processing approach according to claim 1, a transmission medium according to claim 3, and an information processor according to claim 4 A command is processed using the storage section in which the 2nd field set as the 1st field which memorizes one or more users' data, and the free area which is used by one or more users memorized to the 1st field, and is not used as the 1st field is formed.
[0034] In the information processing approach according to claim 6, a transmission medium according to claim 10, and an information processor according to claim 11, a processing means assigns a logical-block number to the data memorized by the physical block, and the new data with which a storage means has a predetermined logical-block number are memorized to physical blocks other than the

physical block the data which have the logical-block number are remembered to be.
[0035] In the information processing approach according to claim 15, a transmission medium according to claim 17, and an information processor according to claim 18, it has a recognition number, a processing means compares the recognition number which the command supplied by the user has with the recognition number which data have, and the data of a predetermined block of the 2nd field process a command corresponding to the comparison result.
[0036] In the information processing approach according to claim 20, a transmission medium according to claim 22, and an information processor according to claim 23 The number corresponding to the sequence memorized can be assigned to the data memorized by block with a processing means. With a storage means When the block which has the last number is the last block, new data are memorized by top block, and new data are memorized by the block next to the block which has the last number when the block which has the last number is not the last block.
[0037] In the information processing approach according to claim 25, a transmission medium according to claim 26, and an information processor according to claim 27 As opposed to a predetermined field [in / the 2nd field managed per block of the predetermined magnitude used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field is formed, and / the 2nd field] And a command is processed using the storage section which memorizes two or more data which specify an access privilege which is different to each user, respectively to the 1st field.
[0038] In the information processing approach according to claim 28, a transmission medium according to claim 29, and an information processor according to claim 30 The 1st field which memorizes two or more users' data, and the 2nd field which is used by two or more users memorized to the 1st field, and is managed per block of predetermined magnitude are formed, and it sets to the 1st field. A command is processed using the storage section which memorizes the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.
[0039] In the information processing approach according to claim 31, a transmission medium according to claim 32, and an information processor according to claim 33 It is used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. The 2nd field managed per block of predetermined magnitude is formed, and a command is processed using the storage section which memorizes the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.
[0040]
[Embodiment of the Invention] Although the gestalt of operation of this invention is explained below, it is as follows, when the gestalt (however, an example) of operation [/ in the parenthesis after each means] is added and the description of this invention is described, in order to clarify correspondence relation between each means of invention given in a claim, and the gestalt of the following operations. However, of course, this publication does not mean limiting to what indicated each means.
[0041] A detection means by which an information processor according to claim 4 detects the input signal from the outside (for example, BPSK demodulator circuit 62 of drawing 3), Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. A storage means by which the 2nd field set as the free area which is not used as the 1st field is formed (for example, EEPROM66 of drawing 3), It is characterized by having a processing means (for example, sequencer 91 of drawing 3) to perform processing corresponding to an input signal using a storage means, and an output means (for example, BPSK modulation circuit 68 of drawing 3) to output the result of processing of a processing means outside.
[0042] A processing means (for example, sequencer 91 of drawing 3) assigns a logical-block number to the data memorized by the physical block, and an information processor according to claim 11 is characterized by memorizing the new data with which a storage means (for example, EEPROM66 of drawing 3) has a predetermined logical-block number to physical blocks other than the physical block the data which have the logical-block number are remembered to be.
[0043] An information processor according to claim 18 is characterized by for the data of a predetermined block of the 2nd field having a recognition number, comparing the recognition number

which the command with which the processing means (for example, sequencer 91 of drawing 3) was supplied by the user has with the recognition number which data have, and processing a command corresponding to the comparison result.

[0044] A detection means by which an information processor according to claim 23 detects the command from a predetermined user, A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A processing means The number corresponding to the sequence memorized is assigned to the data memorized by block. A storage means The number corresponding to the block of the head of the field which a user uses for the 1st field, and the number corresponding to the last block are memorized. It is characterized by memorizing new data to a top block, when the block which has the last number is the last block, and memorizing new data to the block next to the block which has the last number, when the block which has the last number is not the last block.

[0045] A detection means by which an information processor according to claim 27 detects the command from a predetermined user, A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means, Are used by one or more users memorized to the 1st field which memorizes one or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing two or more data which specify an access privilege different, respectively to the 1st field to each user as opposed to the predetermined field in the 2nd field.

[0046] A detection means by which an information processor according to claim 30 detects the command from a predetermined user (for example, BPSK demodulator circuit 62 of drawing 3), A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means (for example, BPSK modulation circuit 68 of drawing 3), Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed, and a storage means is characterized by memorizing the data with which two or more users use the predetermined field in the 2nd field jointly to the 1st field.

[0047] A detection means by which an information processor according to claim 33 detects the command from a predetermined user (for example, BPSK demodulator circuit 62 of drawing 3), A processing means (for example, sequencer 91 of drawing 3) to process a command, and an output means to output the result of processing of a processing means (for example, BPSK modulation circuit 68 of drawing 3), Are used by two or more users memorized to the 1st field which memorizes two or more users' data, and the 1st field. It has a storage means (for example, EEPROM66 of drawing 3) by which the 2nd field managed per block of predetermined magnitude is formed. A storage means It is characterized by memorizing the predetermined field in the 2nd field, and two or more data which specify the access privilege from which two or more users differ, respectively to the 1st field.

[0048] Drawing 1 shows an example using R/W1 and IC card 2 of a non-contact card system. Using an electromagnetic wave, R/W1 and IC card 2 are non-contact, and transmit and receive data.

[0049] If R/W1 transmits a predetermined command to IC card 2, IC card 2 receives the command and is made as [perform / processing corresponding to the command].

[0050] If R/W1 transmits data to IC card 2, the command is received, and IC card 2 which is the gestalt of 1 operation of the information processor of this invention processes the received command, and is made as [transmit / to R/W1 / the response data corresponding to the processing result].

[0051] Moreover, it connects with a controller 3 through a predetermined interface (for example, RS-485A), and a predetermined control signal is supplied to R/W1 from a controller 3, and it is made as [process] according to the control signal.

[0052] Drawing 2 shows the configuration of R/W1.
[0053] In IC21 Processing of data The communication link with SPU (Signal Processing Unit)32 and the controller 3 which process the data received from the data and IC card 2 which are transmitted to DPU (Data Processing Unit)31 to perform and IC card 2 It is ** to SCC (Serial Communication Controller)33 which carries out, and processing of data. The ROM section 41 which has memorized **** information beforehand, The memory section 34 which consists of RAM sections 42 which memorize the data in the middle of processing temporarily is connected through the bus.
[0054] Moreover, the flash memory 22 which memorizes predetermined data is also connected to this bus.
[0055] DPU31 is made as [receive / from SPU32 / the response data received from IC card 2] while outputting the command transmitted to IC card 2 to SPU32.
[0056] SPU32 is made in the response data transmitted with IC card 2 as [perform / to reception and its data / from a demodulator circuit 25 / predetermined processing] while outputting to a modulation circuit 23, after performing predetermined processing (for example, BPSK (BiPhase Shift Keying) modulation (after-mentioned)) to the command transmitted to IC card 2.
[0057] A modulation circuit 23 is data to which the subcarrier of the predetermined frequency (for example, 13.56MHz) supplied from the oscillator 26 was supplied from SPU32, and is made as [output / through an antenna 27 / the generated modulated wave / carry out an ASK (AmplitudeShift Keying) modulation and / to IC card 2 / as an electromagnetic wave]. At this time, a modulation circuit 23 makes a modulation factor less than one, and performs an ASK modulation. That is, when data are a low level, it is made for the maximum amplitude of a modulated wave not to become zero.
[0058] The demodulator circuit 24 is made as [output / restore to the modulated wave (ASK modulated wave) which received through the antenna 27, and / to SPU32 / the data to which it restored].
[0059] Drawing 3 shows the example of a configuration of IC card 2. In this IC card 2, IC51 is made as [receive / through an antenna 53 / the modulated wave transmitted by R/W1]. In addition, a capacitor 52 constitutes LC circuit with an antenna 53, and is made as [side / with the electromagnetic wave of a predetermined frequency (carrier frequency)].
[0060] In IC51 RF interface section 61 While detecting the modulated wave (ASK modulated wave) which received through the antenna 53, getting over in the ASK recovery section 81 and outputting the data after a recovery to the BPSK demodulator circuit 62 and the PLL (Phase Locked Loop) section 63 At a voltage regulator 82, it is ASK **. The signal which the tone section 81 detected is stabilized and it is made as [supply / each circuit / as direct current power].
[0061] Moreover, RF interface section 61 oscillates the signal of the same frequency as the clock frequency of data in an oscillator circuit 83, and is made as [output / to the PLL section 63 / the signal].

[0062] And the ASK modulation section 84 of RF interface section 61 Corresponding to the data supplied from operation part 64, fluctuate the load of the antenna 53 as a power source of IC card 2. (For example, correspond to data and a predetermined switching element is made to turn on / turn off, and only when a switching element is an ON state, a predetermined load is connected to an antenna 53 at juxtaposition) By things The modulated wave which has received through an antenna 53 (when transmitting data from IC card 2) maximum amplitude of a modulated wave is fixed -- **** -- an ASK modulation is carried out and the modulation component is transmitted to R/W1 through an antenna 53 -- it is made like (the terminal voltage of the antenna 27 of R/W1 is fluctuated).
[0063] From the data supplied from the ASK recovery section 81, the PLL section 63 generates the clock signal which synchronized with the data, and is made as [output / to the BPSK demodulator circuit 62 and the BPSK modulation circuit 68 / the clock signal].
[0064] The BPSK demodulator circuit 62 is made as [output / according to the clock signal supplied from the PLL section 63, / restore to the data and / to operation part 64 / the data to which it restored], when the BPSK modulation of the data to which it restored in the ASK recovery section 81 is carried out.
[0065] When the data supplied from the BPSK demodulator circuit 62 are enciphered, operation part 64

is made as [process / the data / by the sequencer 91 / as a command], after decrypting the data in a code / decode section 92. In addition, when data are not enciphered, the data supplied from the BPSK demodulator circuit 62 are directly supplied to a sequencer 91 without a code / decode section 92.

[0066] The sequencer 91 is made as [perform / processing corresponding to the supplied command]. For example, a sequencer 91 processes the data memorized by EEPROM66 at this time.

[0067] The parity operation part 93 of operation part 64 is made as [compute / a Reed Solomon code] as parity from the data memorized by EEPROM66 and the data memorized by EEPROM66.

[0068] Furthermore, operation part 64 is made as [output / to the BPSK modulation circuit 68 / the response data (data transmitted to R/W1) corresponding to the processing], after performing processing predetermined by the sequencer 91.

[0069] The BPSK modulation circuit 68 carries out the BPSK modulation of the data supplied from operation part 64 (after-mentioned), and is made as [output / to the ASK modulation section 84 of RF interface section 61 / the data after a modulation].

[0070] RAM67 is made as [memorize / the data in the middle of processing etc. / temporarily], when a sequencer 91 processes.

[0071] EEPROM (Electrically Erasable and Programmable ROM)66 is the memory of a non-volatile, and even after IC card 2 ends the communication link with R/W1 and an electric power supply stops, it is made as [memorize / data / continue]. The fundamental program required for a sequencer 91 to process the command from R/W1 is memorized by ROM65.

[0072] Drawing 4 shows an example of assignment of the memory of EEPROM66.

[0073] EEPROM66 has 40 bytes of 256 physical blocks. Each physical block consists of a total of 40 bytes of 32 bytes of data division (D00 thru/or D1f), 2 bytes of attribute section (AT1, AT2), and 6 bytes of parity section (P0 thru/or P5).

[0074] The physical block number ffH (H expresses the hexadecimal) of EEPROM66 is assigned to the system ID block. The system ID block has memorized the information about the security of IC card 2.

[0075] Next, the physical block is assigned to the common area definition block (Common Area Definition Block) (the 1st field) or the provider domain-defined block (Provider Area Definition Block) (the 1st field) one by one toward 00H from the physical block number fdH.

[0076] When IC card 2 is published by EEPROM66, those (provider) who offer the system using this IC card 2 with predetermined equipment (issue machine) are registered into it. An issue machine is 1 physical block per one provider, carries out sequential use of the provider domain-defined block toward 00H from the physical block number fdH, and registers a provider.

[0077] The common area definition block and the provider domain-defined block have memorized the information on the location of the storage region which a provider uses etc.

[0078] And the physical block which is not used as a system ID block, a common area definition block, and a provider domain-defined block is assigned to the user block (User Block) used by the provider.

[0079] Drawing 5 shows an example of the assignment of each data to a system ID block.

[0080] As for D00 thru/or D0f of data division, the manufacture ID at the time of manufacture of EEPROM66 (Manufacture ID) (IDm) is memorized. A field D00 thru/or D03, a field D04 or D07, a field D08 or D0b and field D0c thru/or D0f have memorized the IC code of EEPROM66, the code (Manufacture Equipment Code) of the manufacture machine which created EEPROM66, the manufacture date (Manufacture Date) of EEPROM66, and the manufacture serial number (Manufacture Serial Number) of EEPROM66, respectively.

[0081] By using this information on IDm, all IC cards 2 (EEPROM66) are discriminable. In addition, a manufacture date sets January 1, 2000 to 0000H, and makes it the days from January 1, 2000. In addition, when a manufacture date is the 1990 set, a manufacture date is expressed as negative days from January 1, 2000 using a two's complement.

[0082] The issue ID (Issue ID) (IDi) when D10 thru/or D1f of data division publishes this ID card 2 is memorized. A field D10 thru/or D13, a field D14 or D17, a field D18 or D1b and field D1c thru/or 1f of the codes of the issue machine which published the category / group number which shows the category and group to whom IC card 2 belongs, and this IC card 2, the dates which published IC card 2, and the

expiration dates of IC card 2 are memorized, respectively.
[0083] Drawing 6 shows the attribute section of a system ID block. The attribute section has memorized the number of providers registered. In case an issue machine registers one provider, one physical block is used for it and it updates the value of this attribute section then.
[0084] The value of the attribute section is set as zero at the time of manufacture, and when an issue machine registers a provider into IC card 2 after that, it updates the value of the attribute section by the number of providers registered.
[0085] The parity section of a system ID block has memorized the Reed Solomon code (RS sign) calculated by the parity operation part 93 from the value of each bit of data division and the attribute section. Therefore, the value of the parity section is recalculated whenever data division or the attribute section is updated.
[0086] Drawing 7 shows an example of a common area definition block and a provider domain-defined block. In addition, these blocks are beforehand written in by the issue machine, when IC card 2 is published.
[0087] The common area definition block has been arranged at the physical block number feH of EEPROM66, and has memorized a setup of the storage region (common area (Common Area)) (the 2nd field) used by all providers.
[0088] From the physical block number fdH of EEPROM66, the provider domain-defined block has been arranged toward 00H, is 1 physical block per one provider, and has memorized a provider's information.
[0089] As shown in drawing 7 , the data division D00 of a domain-defined block (a common area definition block and provider domain-defined block) thru/or the fields D00 and D01 of D1f have memorized the provider code (Provider Code) which shows provider's class. In the common area definition block, the value of fields D00 and D01 is set to 0000H, and, in the provider domain-defined block, let the value of fields D00 and D01 be the value of either 0001H thru/or FFFFH.
[0090] The field D02 of the data division of a domain-defined block thru/or D05 have memorized the allocation table (Allocation Table) which consists of numbers BN1 (fields D04 and D05) (BN1> BN0) of the next physical block of the number BN0 (fields D02 and D03) of the physical block of the head of the storage region (provider field (Provider Area)) (the 2nd field) which this provider uses, and the number of the physical block of an end. a provider field is set as the position (the physical block number BN0 -- or (BN 1-1)) of EEPROM66 except a system block (a system ID block, domain-defined block), as shown in drawing 8 .
[0091] Thus, since the provider field is specified by BN0 and BN1, it is not the area size used by the provider (user), and for the information on the amount proportional to the number of providers, data can be managed and use effectiveness of memory can be made high.
[0092] The field D06 of the data division of a domain-defined block thru/or D09 have remembered the partition table (Partition Table) which consists of the block counts BRW (fields D08 and D09) of the read/write block in a random access field to be block count BRA (fields D06 and D07) of a random access field (after-mentioned) among the storage regions which a provider uses. At this time, block count BRA of a random access field is set as formula BRA=0 or the value with which are satisfied of formula 2xn<=BRA<=BN1-BN0 (n is the number of light buffers (after-mentioned)), and the block count BRW of a read/write block is set as BRW=0, when it is BRA=0, and when it is BRA!=0, it is set as the value with which are satisfied of formula n<=BRW<=BRA-n.
[0093] Field D0a of the data division of a domain-defined block and D0b have memorized several n of the light buffer of a random access field. A n light buffer is used when making coincidence memorize n data to logical-block number 00H thru/or (00+n (hexadecimal display)) H of a random access field. In addition, when memorizing data to the physical block which has other logical-block numbers among random access fields, only one light buffer is used.
[0094] as mentioned above, according to a domain-defined block, it is shown in drawing 8 -- as -- the physical block number BN0 -- a field (a provider field or common area) is assigned to the provider specified in provider code, further, the physical block of the BRA individual of the field (a provider field

or common area) is assigned to a random access field, and the remaining physical blocks are assigned to the sequential access field (after-mentioned) for or (BN 1-1).
[0095] Furthermore, according to the domain-defined block, as shown in drawing 8 , the random access field is logically assigned to the read/write block of a BRW individual, the read-only block, and the n light buffer. In addition, a read/write block and physical blocks other than a light buffer are assigned to a read-only block.
[0096] Field D0c of the data division of a domain-defined block and D0d have memorized the Perth block permission which has the information on the access privilege to the Perth block (Purse Block) (after-mentioned) in the storage region (random access field) which this provider uses.
[0097] Drawing 9 shows an example of the Perth block permission.
[0098] The Perth block permission (16 bits, b0, or bf) is read to the Perth block, and shows authorization or the disapproval of an add instruction and a subtraction instruction.
[0099] The Perth block permission of a common area definition block has memorized whether the Perth block is used in the storage region (common area) set up with a common area definition block to Field (bit) bb. That is, in the case of bb=0, the Perth block is not used. In the case of bb=1, the Perth block is used. And especially the field (bit) of others in the Perth block permission of a common area definition block is not used. In addition, in the case of bb=1, the read/write block whose logical-block number is 00H is used as a Perth block.
[0100] Next, in the Perth block permission of a provider domain-defined block, it has memorized whether the Perth block is used in the storage region set up with this provider domain-defined block to the field b3. That is, in the case of b3=0, the Perth block is not used. In the case of b3=1, the Perth block is used. In addition, in the case of b3=1, the read/write block whose logical-block number is 00H is used as a Perth block.
[0101] And the propriety of an add instruction to the Perth block was memorized to the field b2, the propriety of a subtraction instruction to the Perth block was memorized to the field b1, and the propriety of read-out to the Perth block is memorized to the field b0 (in the case of bi=1 (1 i= 0, 2), the instruction is permitted and, in the case of bi=0, the instruction is not permitted). Moreover, it has memorized to Field bb whether the Perth block is used in the storage region set up with a common area definition block. In addition, the same value as bb of the Perth block permission of a common area definition block is memorized by bb.
[0102] Furthermore, the propriety of an add instruction to the Perth block was memorized to Field ba, the propriety of a subtraction instruction to the Perth block was memorized to the field b9, and the propriety of read-out to the Perth block is memorized to the field b8 (in the case of bi=1 (8 i= 9a), the instruction is permitted and, in the case of bi=0, the instruction is not permitted).
[0103] Field D0e of the data division of a domain-defined block of drawing 7 and D0f memorized the version number of the security key (a common key and provider key) used for encryption and a decryption at a provider's (R/W1) authentication, and a list, and a field D10 thru/or 1f of the security key are memorized.
[0104] In addition, when R/W1 polls, IC card 2 returns the version number of these two keys (a common key and provider key). Therefore, in authentication between R/W1 and IC card 2, the security key of two or more versions can be used properly.
[0105] And the attribute sections AT1 and AT2 of a domain-defined block are formed as a reserve, and especially information is not memorized. The parity section of a domain-defined block has memorized the parity (RS sign) calculated from the value of all the bits of data division and the attribute section.
[0106] Thus, the domain-defined block set up by the issue machine has memorized a provider code, an allocation table, a partition table, the Perth block permission, the security key version, and the security key.
[0107] Drawing 10 shows an example of a user block. As mentioned above with reference to drawing 4 , physical blocks other than a system ID block, a common area definition block, and a provider domain-defined block are used by the provider as a user block among the rooms of EEPROM66.
[0108] For example, if eight providers are registered when room consists of 256 blocks as shown in

drawing 4 , 246 (= 256-10) blocks of those other than the system block of a total of ten (= 1+1+8) individuals of a system ID block, a common area definition block, and eight provider domain-defined blocks will be used as a user block. Moreover, if 40 providers are registered, a system block will become a total of 42 (= 1+1+40) individuals, and the user block of 214 (= 256-42) individuals will be secured.
[0109] A user block is assigned to each provider according to the allocation table (drawing 7) of a domain-defined block. In addition, since a provider uses the user block currently assigned beforehand with reference to an allocation table, he does not access other than the field (a provider field or common area) assigned on the allocation table.
[0110] The user block of the field (a provider field or common area) assigned on the allocation table is assigned to the random access field and the sequential access field according to the above-mentioned partition table (drawing 7).
[0111] Furthermore, the user block of a random access field is used as either a read/write block, a read-only block and a light buffer, and the number of these blocks is set up as mentioned above according to the number of a partition table and light buffers.
[0112] Thus, the data division D00 thru/or D1f of a user block currently assigned is used according to processing by the provider to whom the user block is assigned.
[0113] The attribute section of a user block of a random access field has memorized the incremental counter (Incremental Counter) (bits bf and be) and the logical-block number (Bit bd thru/or b0), as shown in drawing 11 .
[0114] A logical-block number and an incremental counter are used when accessing the user block of a random access field.
[0115] When reading the data memorized to the random access field, it is a logical-block number, and the data (physical block) to read are searched and the newest data are read with reference to the incremental counter of the data which have the logical-block number.
[0116] On the other hand, when memorizing data to a random access field, after using as a light buffer the physical block (after-mentioned) which became unnecessary with reference to the logical-block number and incremental counter of data which have already been memorized to the random access field, data are written in the light buffer.
[0117] In addition, when the Perth block permission of an above-mentioned domain-defined block is set up so that the Perth block may be used, the read/write block whose logical-block number is 00H is used as a Perth block.
[0118] The Perth block is used to read the value already memorized when performing addition and subtraction of data frequently when setting up the access privilege to data finely (since possibility that information will be revealed increases).
[0119] Drawing 12 shows an example of the Perth block. The data division D00 of the Perth block the field D00 of D1f thru/or D07 are used as Perth data division. The data division D00 of the Perth block the field D08 of D1f thru/or D0f have memorized Execution ID (Execution ID). In addition, the field D10 thru/or D1f of data division of the Perth block is set up read-only, although used as user data division.
[0120] The Perth data division have memorized predetermined data. Execution ID is referred to when the add instruction or subtraction instruction to the Perth block is executed, and it is compared with the execution ID contained in the add instruction or a subtraction instruction.
[0121] On the other hand, the attribute section of a user block of a sequential access field has memorized the lap round number (Bit bf thru/or b0), as shown in drawing 13 . In the sequential access field, if data (sequentially) are memorized in an order from the physical block of the head of a field and data are memorized to the physical block of the last of a field, data are again memorized in an order from the physical block of the head of a field (overwritten). The lap round number has memorized the sequence.
[0122] Therefore, when accessing the user block of a sequential access field, while being used, when memorizing data to a sequential access field, sequential reference of the lap round number is carried out. And data are memorized by the next physical block of the physical block which has a lap round number at the tail end till then. At this time, the lap round number of the physical block data were remembered

to be is set as the number which added 1 to the lap round number at the tail end till then.
[0123] In addition, a failure occurs in the middle of a store at the time of the last store, and when the parity error (physical memory colla tempestade PUSHON) has arisen in the physical block which has a lap round number at the tail end, new data are memorized by the physical block, for example. Moreover, new data are memorized by the physical block of the head of a sequential access field when the physical block which has a lap round number at the tail end is a physical block of a sequential access end-of-region rate.
[0124] As mentioned above, EEPROM66 is suitably used for each provider.
[0125] Next, with reference to the flow chart of drawing 14 , and the timing chart of drawing 15 , actuation of IC card 2 and R/W1 is explained.
[0126] First, a predetermined electromagnetic wave is emitted from an antenna 27, the loaded condition of an antenna 27 is supervised, IC card 2 approaches, and R/W1 corresponding to the provider registered into IC card 2 in step S1 stands by until change of loaded condition is detected. In addition, R/W1 emits the electromagnetic wave which carried out the ASK modulation by the data of a short predetermined pattern, and you may make it repeat the appeal to IC card 2 in step S1 until the response from IC card 2 is obtained in fixed time amount.
[0127] When R/W1 detects approach of IC card 2 in step S1 (time of day t0 of drawing 15), it progresses to step S2. SPU32 of R/W1 The square wave of a predetermined frequency (a clock frequency twice the frequency [for example,] of data) as shown in drawing 16 (a) is made into a subcarrier. By the data (command corresponding to the processing which IC card 2 is made to perform) (for example, data shown in drawing 16 (b)) transmitted to IC card 2, a BPSK modulation is performed and the generated modulated wave (BPSK modulating signal) (drawing 16 (c)) is outputted to a modulation circuit 23.
[0128] In addition, as shown in drawing 16 (c) using differential conversion at the time of a BPSK modulation A value makes the same thing as the last BPSK modulating signal ("1", "0" or "0", "1") a BPSK modulating signal, when the data of 0 appear. The value makes what reversed the phase of the last BPSK modulating signal (thing which made "0" reverse "1" and made "1" reverse "0") the BPSK modulating signal, when the data of 1 appear.
[0129] Thus, since it gets over to the original data also when a BPSK modulating signal is reversed by holding data by change of the phase of a modulated wave using differential conversion, when getting over, the need of considering the polarity of a modulated wave is lost.
[0130] And a modulation circuit 23 is the BPSK modulating signal, the ASK modulation of the predetermined subcarrier is carried out with less than (for example, 0.1) one modulation factor (= maximum amplitude of the maximum amplitude/subcarrier of a data signal), and the generated modulated wave (ASK modulated wave) is transmitted to IC card 2 through an antenna 27 (during the time of day t0 of drawing 15 thru/or time of day t1).
[0131] In addition, when not transmitting, the modulation circuit 23 is made as [generate / with the high level of the two level (high level and low level) of a digital signal / a modulated wave].
[0132] Next, in step S3, IC cards 2 are an antenna 53 and a capacitor 52, transform into an electrical signal a part of electromagnetic wave which the antenna 27 of R/W1 emitted, and output the electrical signal (modulated wave) to RF interface section 61 of IC51. and the ASK recovery section 81 of RF interface section 61 controls the dc component of the generated signal, extracts a data signal, and outputs the data signal to the BPSK demodulator circuit 62 and the PLL section 63 while it supplies the signal generated in the modulated wave by rectifying and carrying out smooth (namely, envelope detection -- carrying out) to a voltage regulator 82.
[0133] A voltage regulator 82 stabilizes the signal supplied from the ASK recovery section 81, generates direct current power, and supplies it to each circuit.
[0134] In addition, the terminal voltage V0 of an antenna 53 is as follows, for example at this time.
V0=V10(1+kxVs (t)) cos (omegat)
[0135] It is here, and in V10, k shows a modulation factor and Vs (t) shows the signal component for the amplitude of a carrier component, respectively.

[0136] Moreover, the value VLR of a low level in the electrical potential difference V1 after rectification by the ASK recovery section 81 is as follows, for example.
VLR=V10(1+kx (-1))-Vf[0137] Here, Vf shows the voltage drop in the diode D of a rectifier circuit. Usually, Vf is about 0.7 volts.
[0138] And a voltage regulator 82 stabilizes rectification and the signal by which smooth was carried out by the ASK recovery section 81, and supplies it to each circuits including operation part 64 as direct current power. In addition, since the modulation factor k of a modulated wave is less than one, its voltage variation after rectification (difference of high level and a low level) is small. Therefore, a voltage regulator 82 can generate direct current power easily.
[0139] When a modulation factor k receives 5% of modulated wave so that V10 may become 3 volts or more, for example, the low-level electrical potential difference VLR after rectification It becomes more than 2.15 (=3x(1-0.05)-0.7) bolts. A voltage regulator 82 While being able to supply electrical potential difference sufficient as a power source to each circuit, amplitude 2xkxV10 (Peak-to-Peak value) of the alternating current component (data component) of the electrical potential difference V1 after rectification It becomes more than 0.3 (= 2x0.05x3) bolts, and the ASK recovery section 81 can restore to data by the sufficiently high S/N ratio.
[0140] Thus, when a modulation factor k uses less than one ASK modulated wave, while performing the communication link with a low (in high condition of a S/N ratio) error rate, direct current voltage sufficient as a power source is supplied to IC card 2.
[0141] And according to the clock signal supplied from the PLL section 63, the BPSK demodulator circuit 62 restores to the data signal (BPSK modulating signal) from the ASK recovery section 81, and outputs the data to which it restored to operation part 64.
[0142] Next, in step S4, when the data supplied from the BPSK demodulator circuit 62 are enciphered, after decrypting operation part 64 in a code / decode section 92, it supplies the data (command) to a sequencer 91, and performs processing corresponding to the command (during the time of day t1 of drawing 15 thru/or time of day t2). In addition, while the value had transmitted the data of 1, R/W1 is standing by, until it receives the answerback from this period 2, i.e., an IC card. Therefore, in this period, IC card 2 has received the modulated wave with fixed maximum amplitude.
[0143] Next, in step S5, the sequencer 91 of operation part 64 outputs data (data transmitted to R/W1), such as a processing result, to the BPSK modulation circuit 68. Like SPU32 of R/W1, after the BPSK modulation circuit 68 carries out the BPSK modulation of the data, it is outputted to the ASK modulation section 84 of RF interface section 61.
[0144] The ASK modulation section 84 and by fluctuating the load connected to the both ends of an antenna 53 according to data using a switching element The modulated wave which has received (in the time of transmission of IC card 2) the maximum amplitude of a modulated wave becomes fixed -- **** -- an ASK modulation is carried out according to the data to transmit, the terminal voltage of the antenna 27 of R/W1 is fluctuated according to it, and the data is transmitted to R/W1 (during the time of day t2 of drawing 15 thru/or time of day t3).
[0145] In step S6, as for the modulation circuit 23 of R/W1, the value is continuing transmission of the data of 1 (high-level) at the time of reception of the data from IC card 2. And a demodulator circuit 25 detects the data transmitted with IC card 2 from minute fluctuation (for example, dozens of microvolts) of the terminal voltage of the antenna 27 of IC card 2, and the antenna 27 combined in electromagnetism.
[0146] And it gets over and a demodulator circuit 25 outputs the generated digital data to SPU32, after amplifying the detected signal (ASK modulated wave) with the amplifier of high interest profit.
[0147] And in step S7, after restoring to the data (BPSK modulating signal), it outputs to DPU31 and, as for SPU32 of R/W1, DPU31 processes the data (during the time of day t3 of drawing 15 thru/or time of day t4).
[0148] Furthermore, in step S8, when it judges whether a communication link is ended according to a processing result and it is judged again that it communicates, to step S2, DPU31 of R/W1 is return, step S2, or step S7, and communicates the following data (command) (time of day t4 thru/or time of day t8 of

drawing 15). On the other hand, when it is judged that a communication link is ended, R/W1 ends the communication link with IC card 2.
[0149] As mentioned above, using the ASK modulation whose modulation factor k is less than one, R/W1 transmits a predetermined command to IC card 2, and IC card 2 performs reception and processing corresponding to the command for the command, and it returns the data corresponding to the result of the processing to R/W1.
[0150] Next, the actuation when writing in data to EEPROM66 as an example of processing by IC card 2 in above-mentioned step S4 is explained with reference to the flow chart of drawing 17 thru/or drawing 21 .
[0151] First, with reference to the flow chart of drawing 17 thru/or drawing 19 , the actuation when writing data in the random access field of EEPROM66 is explained.
[0152] In step S21, a sequencer 91 judges whether it is that the physical block which writes in data is a read/write block (the Perth block is not included) (the block to a BRW individual is considered as a read/write block from BN0 at sequence as shown in drawing 8), and when it is judged that it is a read/write block, it progresses to step S22.
[0153] A sequencer 91 progresses to step S23 (drawing 18), when it judges whether it is using the Perth block (b3=1) and the Perth block is not being used with reference to the Perth block permission (drawing 9) of a provider domain-defined block which has the provider code of R/W1 (in the case of b3=0).
[0154] On the other hand, when it is judged that the Perth block is used in step S22, a sequencer 91 judges whether the read/write block which writes in whether the logical-block number of the data (it writes in) to memorize is 00H and data in step S24 has lapped with the Perth block, and when it is judged that the read/write block which writes in data has not lapped with the Perth block, it progresses to step S23. [0155] When it is judged that the read/write block which writes in data has lapped with the Perth block, a sequencer 91 ends processing in step S25, after performing error processing.
[0156] Moreover, when it is judged that the physical block which writes in data in step S21 is not a read/write block, it progresses to step S26, the physical block in which a sequencer 91 writes data judges whether it is the Perth block, and when it is judged that it is the Perth block, it progresses to step S27.
[0157] When it is judged that the physical block which writes in data is not the Perth block, a sequencer 91 ends processing in step S28, after performing error processing.
[0158] In step S27, in a random access field, a sequencer 91 progresses to step S29, when the Perth block (physical block whose logical-block number is 00H) is looked for and the Perth block is discovered.
[0159] Since the store to the Perth block cannot be performed when the Perth block is not discovered at step S27, a sequencer 91 ends processing in step S30, after performing error processing.
[0160] or [next, / that the add instruction is permitted with reference to the Perth block permission of a provider domain-defined block in step S29 by the instruction (command) to the Perth block judging whether it is an add instruction, and a sequencer 91 progressing to step S31 when it is judged that it is an add instruction] (b2=1) -- it judges whether it is no.
[0161] And when a sequencer 91 judges that the add instruction to the Perth block is permitted at step S31, it progresses to step S23.
[0162] On the other hand, when it is judged that the add instruction to the Perth block is not permitted at step S31, a sequencer 91 ends processing, after performing error processing in step S32, without executing an add instruction (when it is b2=0).
[0163] Moreover, in step S29, when the instruction to as opposed to [when it is judged that the instruction to the Perth block is not an add instruction / progress to step S33 and] the Perth block in a sequencer 91 judges that it judges whether it is a subtraction instruction and is a subtraction instruction, it progresses to step S34.
[0164] And in step S34, a sequencer 91 judges whether it is that the subtraction instruction is permitted (b1=1) with reference to the Perth block permission of a provider domain-defined block, and when it is judged that the subtraction instruction to the Perth block is permitted, it progresses to step S23.

[0165] On the other hand, when it is judged that the subtraction instruction to the Perth block is not permitted at step S34, a sequencer 91 ends processing, after performing error processing in step S35, without executing a subtraction instruction (when it is b1=0).
[0166] Moreover, in step S33, when it is judged that the instruction to the Perth block is not a subtraction instruction, a sequencer 91 ends processing in step S36, after performing error processing.
[0167] Next, in step S23 of drawing 18 , a sequencer 91 searches the physical block of a random access field, and looks for the physical block which has the same logical-block number as the logical-block number of the data which write in.
[0168] And in step S37, a sequencer 91 judges whether the number of the physical blocks discovered at step S23 is two. That is, in this system, the last data and the data before last are memorized at least about each logical block. And when memorizing still newer data, new data are memorized on the data before last (it may memorize on the data of other logical-block numbers before last). When two physical blocks of the same logical-block number exist, it progresses to step S38, and the value (00, 01, 10, or 11) of the incremental counter in the two physical blocks is read and compared.
[0169] And make a physical block with the large value of an incremental counter into the physical block (new physical block) new data are remembered to be, and let a physical block with the small value of an incremental counter be the physical block (old physical block) old data are remembered to be.
[0170] However, when the values of two incremental counters are 00 and 11, make into a new physical block the physical block whose value of an incremental counter is 00, and let the physical block whose value of an incremental counter is 11 be an old physical block.
[0171] A sequencer 91 memorizes the number (physical block number) of a new physical block to RAM67 as a variable Y between two physical blocks, and makes RAM67 memorize the number of an old physical block in step S39 as a variable W (number of the physical block used as a light block).
[0172] Thus, a sequencer 91 progresses to step S49, after making Variable Y and Variable W memorize.

[0173] On the other hand, when it is judged that the number of the physical blocks discovered at step S23 is not two in step S37, it progresses to step S40 and a sequencer 91 judges whether the number of the physical blocks discovered at step S23 is one. And when it is judged that it is one piece, it progresses to step S41.
[0174] In step S40, when a sequencer 91 judges that the number of the physical blocks discovered at step S23 is not one, after performing error processing, processing is ended in step S42.
[0175] As for saying [that the same logical block exists only in one piece], the data before last will not exist for a certain reason. Then, in this case, it is the physical block of other logical-block numbers, and the physical block (namely, physical block whose number of the physical blocks which have the same logical-block number is two) which has data the last and before last is searched, and the physical block before last of them is used as a light block. For this reason, in step S41, a sequencer 91 progresses to step S43, after making RAM67 memorize the number of the discovered physical block (one piece) as a variable Y.
[0176] In step S43, a sequencer 91 searches the physical block of a random access field, and looks for two physical blocks which have the same predetermined logical-block number (arbitration) (logical-block number unrelated to the logical-block number now made into the write-in object).
[0177] In addition, since it retrieves sequentially from logical-block number 00H when searching a physical block, a smaller number, then retrieval time can be shortened for the logical-block number of the data which perform write-in processing frequently.
[0178] In step S44 and a sequencer 91 It judges whether the physical block whose logical-block number is the two same pieces was discovered at step S43. When it is judged that it was discovered, the incremental counter of two physical blocks discovered by progressing to step S45 is referred to. It progresses to step S49 (drawing 19), after making RAM67 memorize the number of the physical block of the older one as a variable W (number of a light block) between two physical blocks.
[0179] On the other hand, when it is judged in step S44 that two physical blocks were not discovered at step S43, it progresses to step S46, and a sequencer 91 carries out sequential count of the parity of each

physical block of a random access field, and looks for the physical block which has started the parity error as compared with the value memorized by the parity section of each physical block.
[0180] And when it judges whether there is any physical block which has started the parity error and it is judged that there is a physical block which has started the parity error, it progresses to step S47, and a sequencer 91 progresses to step S49, after making RAM67 memorize the number of the physical block as a variable W (number of a light block).
[0181] In step S46, when it is judged that there is no physical block which has started the parity error, a sequencer 91 ends processing in step S48, after performing error processing.
[0182] In step S49 of drawing 19 next, a sequencer 91 It judges whether the physical block which writes in data is the Perth block (physical block whose logical-block number is 00H). When it is judged that it is the Perth block, the execution ID of the instruction which progresses to step S50 and is performed to the Perth block When it judges whether it is the same as that of the execution ID of the physical block of the number memorized as a variable Y at step S39 or step S41 (drawing 12) and it is judged that it is the same, it judges that this instruction is already processed and processing is ended.
[0183] Thus, since IC card 2 does not process the command when R/W1 carries out the retry of the same command by using Execution ID and the command is already processed, the same command is not processed twice.
[0184] In step S50, when the instruction with which a sequencer 91 is performed to the Perth block in step S51 when the execution ID of the instruction performed to the Perth block judges that it is not the same as that of the execution ID of the physical block of the number memorized as a variable Y judges whether it is an add instruction and it is an add instruction, it progresses to step S52.
[0185] In step S52, a sequencer 91 reads the Perth data of the physical block of the number of Variable Y, calculates the sum of the Perth data and the data contained in the instruction performed to the Perth block, and uses the sum as the Perth data (new Perth data) in new block data. Thus, it progresses to step S54, after processing. In addition, let execution ID of the physical block of the number of Variable Y be the execution ID of new block data at this time. This prevents processing of a duplex.
[0186] On the other hand, when it is judged that the instruction performed to the Perth block is not an add instruction in step S51 (that is, it is a subtraction instruction), it progresses to step S53, and a sequencer 91 reads the Perth data of the physical block of the number of Variable Y, calculates the difference of the Perth data and the data contained in the instruction performed to the Perth block, and uses the difference as the Perth data (new Perth data) in new block data. Thus, it progresses to step S54, after processing. In addition, let execution ID of the physical block of the number of Variable Y be the execution ID of new block data at this time. This prevents processing of a duplex.
[0187] Moreover, in step S49, a sequencer 91 progresses to step S54, when the physical block which writes in data judges that it is not the Perth block (that is, it is a read/write block).
[0188] And in step S54, a sequencer 91 makes the number which added 1 to the value of the incremental counter of the physical block of the number of Variable Y the value of the incremental counter of new block data. However, when the value of the incremental counter of the physical block of the number of Variable Y is 11, a sequencer 91 sets the value of the incremental counter of new block data to 00.
[0189] Next, in step S55, a sequencer 91 makes the parity of the data newly written in the parity operation part 93, an incremental counter, and a logical-block number calculate, and makes the value of the parity the value of the parity section of new block data.
[0190] And a sequencer 91 makes the physical block (light buffer) of the number of the variable W memorized at either step S39, step S45 or step S47 memorize new block data (the newly memorized data (for them to be the Perth data and Execution ID in the Perth block), its logical-block number, incremental counters, and such parity) in step S56.
[0191] As mentioned above, since the data of the same logical-block number as the logical-block number of the data are left behind to memory by choosing the physical block (light buffer) which remembers data to be a logical-block number using an incremental counter when a failure occurs in the midst of the store of data, logically, memory colla tempestade PUSHON does not occur.
[0192] Although the incremental counter was used with the gestalt of the above-mentioned

implementation in order to distinguish the block with which new data are recorded among the same logical blocks of a random access field, it is also possible to, distinguish the block with which new data are recorded by [at the time of record] securing 4 bytes of field in a random access field, and making time of day (the date, time of day, or value of a counter) then, record to it absolutely for example.
[0193] Next, with reference to the flow chart of drawing 20 and drawing 21 , the actuation when writing data in the sequential access field of EEPROM66 is explained.
[0194] A sequencer 91 makes RAM67 memorize the number of the physical block of the head of a sequential access field as a variable Z in step S61.
[0195] Next, a sequencer 91 reads the lap round number of the physical block whose physical block number is Z, while making RAM67 memorize, it reads the lap round number of the physical block whose physical block number is Z+1, and RAM67 is made to memorize it as a variable B as a variable A in step S62.
[0196] And in step S63, a sequencer 91 judges whether the difference (A-B) of the value of Variable A and the value of Variable B is 1, judges that it is the physical block the physical block of the physical block number Z remembers the data which have a lap round number at the tail end to be when it is not 1, and progresses to step S66.
[0197] When a sequencer 91 judges whether the physical block number Z is the same as the number of the physical block of a sequential access end-of-region rate in step S64 when it is judged that the difference (A-B) of the value of Variable A and the value of Variable B is 1, and it is judged that it is the same, the data with which the physical block of a sequential access end-of-region rate has a lap round number at the tail end are judged to be the physical block to memorize, and it progresses to step S66.
[0198] In step S64, when the physical block number Z judges that it is not the same as that of the number of the physical block of a sequential access end-of-region rate, a sequencer 91 returns to step S62 in step S65, after only 1 makes the value of the variable Z which RAM67 was made to memorize increase. And processing of step S62 thru/or step S65 is repeated successively, changing the value (value of the physical block number to search) of Variable Z.
[0199] Thus, the tail end of the lap round number of the data memorized sequentially is discovered. And in step S66, a sequencer 91 performs the parity check of a block of the number (number of the physical block at the tail end of = lap round number) of Variable Z.
[0200] And in step S67, it judges whether the parity error has produced the sequencer 91 in the physical block, and when it is judged that the parity error has arisen, it progresses to step S68.
[0201] In step S68 a sequencer 91 The value of Variable Z judges whether it is the same as that of the number of the physical block of the head of a sequential access field. When it is judged that it is the same, the tail end of data (what has started the parity error does not contain) judges that it is the physical block of a sequential access end-of-region rate, and sets to step S70. It progresses to step S72 (drawing 21), after making RAM67 memorize the number of the physical block of a sequential access end-of-region rate as a new variable Y.
[0202] When it is judged that the value of Variable Z is not the same as that of the number of the physical block of the head of a sequential access field, after a sequencer 91 makes RAM67 memorize the value (Z-1) which computed the number of the physical block at the tail end of data by having subtracted 1 from the value of Variable Z, and computed it as a variable Y in step S71, it progresses to step S72.
[0203] On the other hand, in step S69, when it is judged that the parity error has not arisen at step S67, a sequencer 91 progresses to step S72, after making RAM67 memorize the number (value of Variable Z in this case) of the physical block at the tail end of data as a variable Y.
[0204] Next, in step S72, when it judges whether a sequencer 91 has the number (value of Variable Y) of the physical block at the tail end of data, and the same number of the physical block of a sequential access end-of-region rate and it is judged that it is the same, it progresses to step S73.
[0205] And in step S73, it progresses to step S75, after a sequencer's 91 making the number of the physical block of the head of a sequential access field the number of the physical block which writes in new data and making RAM67 memorize by making the number into Variable W.

[0206] It progresses to step S75, after a sequencer's 91 making the number which added 1 to the value of Variable Y the number of the physical block which writes in new data, making the number Variable W in step S74 and making RAM67 memorize, when it is judged in step S72 that the number (value of Variable Y) of the physical block at the tail end of data and the number of the physical block of a sequential access end-of-region rate are not the same.

[0207] Next, in step S75, since the newly memorized data and the data which judge whether the physical block (data at the tail end) of the number of Variable Y is the same, and are newly memorized when the same are already memorized, a sequencer 91 ends processing.

[0208] When it is judged that the physical block (data at the tail end) of the data newly memorized on the other hand and the number of Variable Y is not the same, in step S76, a sequencer 91 reads the lap round number of the physical block of the number of Variable Y, and makes the number which added 1 to the value the lap round number of the data (new block data) newly memorized.

[0209] Next, in step S77, a sequencer 91 makes the parity operation part 93 calculate the parity of the data to memorize and a lap round number (new block data), and writes new block data in it in step S78 at the physical block of a number W.

[0210] Thus, since the data of a lap round number smaller than the lap round number of the data which were being written in in the midst of the store of new data when a failure occurred since the lap round number in the data memorized sequentially is retrieved sequentially and new data are memorized to the next physical block (or physical block of the head of a sequential access field) of the data at the tail end remain, logically, memory colla tempestade PUSHON is not generated.

[0211] As mentioned above, EEPROM66 is made as [control / generating of memory colla tempestade PUSHON] using the information on the attribute section while being able to offer a storage region independently to two or more providers.

[0212] In addition, the same user block can also be assigned to two or more providers. In that case, the same user block is assigned on the allocation table of the provider domain-defined block with which those providers (overlap provider) are registered. at this time, a different access privilege (read/write -- or read-only) for every provider can be set up to the same user block by setting up the partition table of a provider domain-defined block for every provider. Furthermore, a predetermined provider can write in data to the user data division (read-only to other providers) of the Perth block which other providers use by setting up so that the Perth block may not be used to a predetermined provider, and setting up so that the Perth block may be used to other providers.

[0213] Moreover, the value of field D0e of a domain-defined block, and D0f (field where the version number of a security key is usually memorized) By setting it as a predetermined value (for example, FFFFH), and memorizing a predetermined provider's provider code (a maximum of eight pieces) to the field D10 thru/or D1f of a domain-defined block further That provider (local common provider) can use the user block assigned on the allocation table of this domain-defined block as a common area.

[0214] Moreover, the access privilege to the user block can be set up for every local common provider by registering a local common provider into two domain-defined blocks which assign the same user block, and setting up a different access privilege for every domain-defined block.

[0215] Thus, corresponding to two or more providers (namely, R/W), processing according to individual can be performed by setting up an overlap provider and a local common provider.

[0216] In addition, this invention can be applied also when delivering and receiving a signal in the condition (contact) of having been combined physically besides in the case of delivering and receiving a signal through radio (non-contact). In the case of interruption of service, or the equipment which operates by the cell, when the cell has been removed, data can be secured.

[0217] Moreover, the program which performs each above-mentioned processing is recorded on the transmission medium which consists of record media, such as a magnetic disk and CD-ROM, it provides for a user, or is transmitted to a user through transmission media, such as a network, is recorded on the transmission medium which consists of record media, such as a hard disk and solid-state memory, and can be made to use.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is the block diagram showing an example using IC card 2 which is the gestalt of 1 operation of the information processor of this invention of a non-contact card system.
[Drawing 2] It is the block diagram showing the example of a configuration of the reader/writer 1 of drawing 1 .
[Drawing 3] It is the block diagram showing the configuration of IC card 2 which is the gestalt of 1 operation of the information processor of this invention.
[Drawing 4] It is drawing showing an example of assignment of the memory of EEPROM66 of drawing 3 .
[Drawing 5] It is drawing showing an example of assignment of each field of a system ID block of drawing 4 .
[Drawing 6] It is drawing showing an example of the attribute section of drawing 5 .
[Drawing 7] It is drawing showing an example of assignment of each field of a domain-defined block of drawing 3 .
[Drawing 8] It is drawing showing an example of assignment of a user block of drawing 3 .
[Drawing 9] It is drawing showing an example of the Perth block permission of drawing 7 .
[Drawing 10] It is drawing showing an example of assignment of each field of a user block of drawing 3 .
[Drawing 11] It is drawing showing an example of the attribute section of a user block of the random access field of drawing 8 .
[Drawing 12] It is drawing showing an example of assignment of each field of the Perth block.
[Drawing 13] It is drawing showing an example of the attribute section of a user block of the sequential access field of drawing 8 .
[Drawing 14] It is the flow chart which actuation of the non-contact card system of drawing 1 explains.
[Drawing 15] It is the timing chart which actuation of the non-contact card system of drawing 1 explains.
[Drawing 16] It is drawing showing an example of a BPSK modulation.
[Drawing 17] It is a flow chart explaining actuation of IC card 2 at the time of the store to the user block of the random access field of drawing 8 .
[Drawing 18] It is a flow chart explaining actuation of IC card 2 at the time of the store to the user block of the random access field of drawing 8 .
[Drawing 19] It is a flow chart explaining actuation of IC card 2 at the time of the store to the user block of the random access field of drawing 8 .
[Drawing 20] It is a flow chart explaining actuation of IC card 2 at the time of the store to the user block of the sequential access field of drawing 8 .
[Drawing 21] It is a flow chart explaining actuation of IC card 2 at the time of the store to the user block of the sequential access field of drawing 8 .
[Description of Notations]

1 Reader/writer 2 IC Card, 3 Controller 21 IC, 23 Modulation circuit 25 A demodulator circuit, 27 Antenna 51 IC and 52 capacitor 53 Antenna 61 RF interface section 62BPSK demodulator circuit, 63 The PLL section 64 Operation part, 65 ROM, 66 EEPROM, 67 RAM 68 BPSK modulation circuit 81 ASK recovery section 82 Voltage regulator 83 Oscillator circuit 84 ASK modulation section 91 Sequencer 92 A code / decode section 93 Parity operation part

---

[Translation done.]

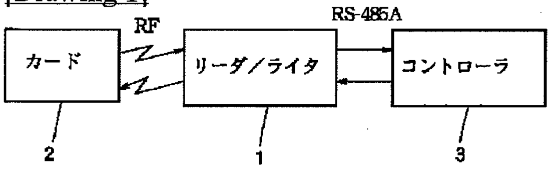
* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

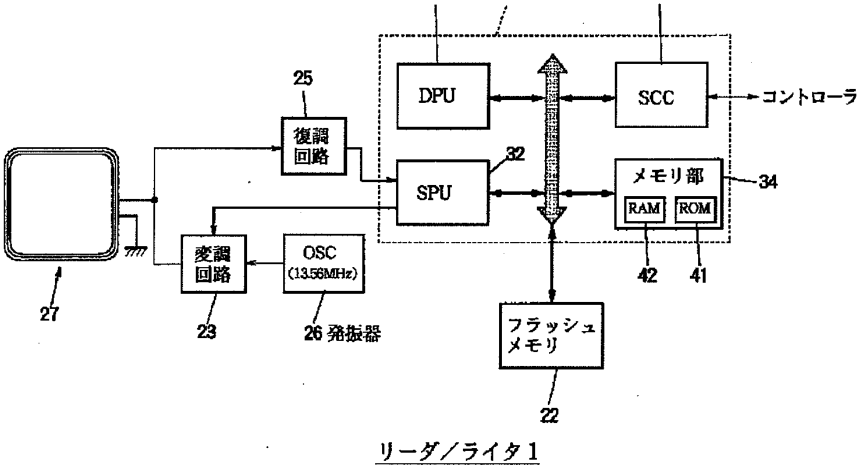
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

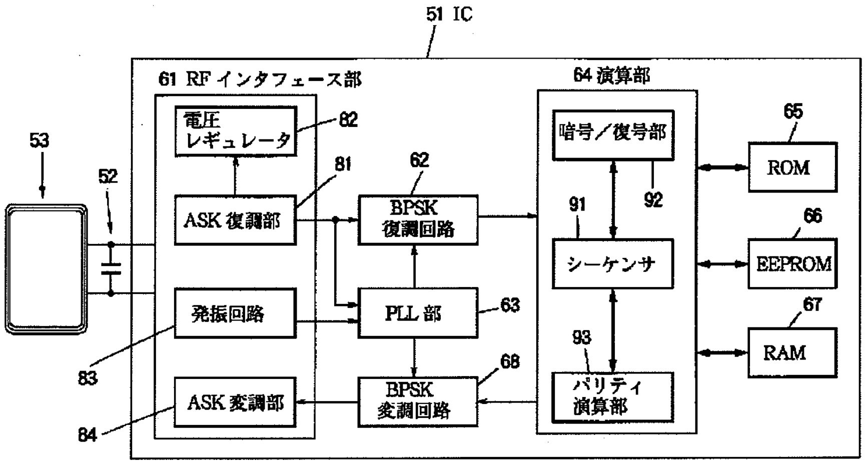
[Drawing 1]


RF
RS-485A
カード
リーダ/ライタ
コントローラ
2
1
3

[Drawing 2]

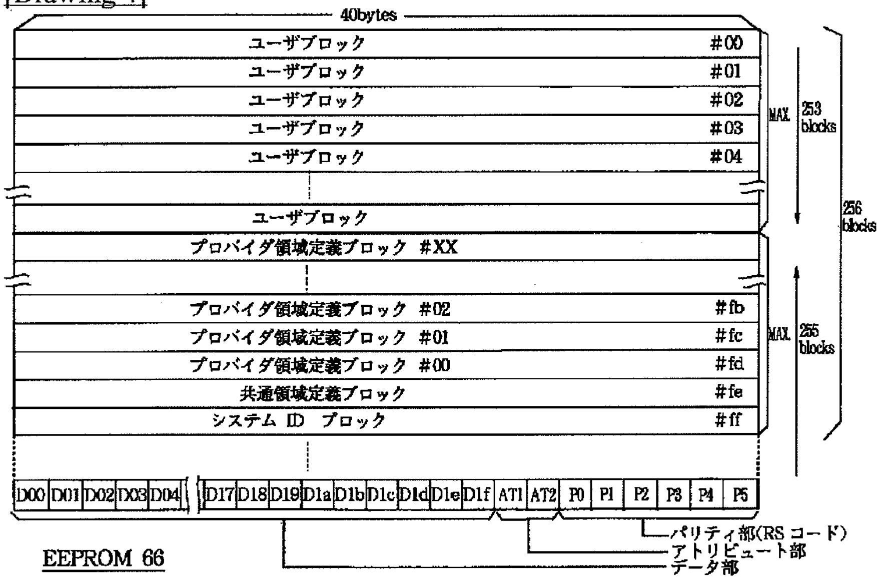

31
21 IC
33
25
DPU
SCC
コントローラ
復調回路
SPU
32
メモリ部
RAM
ROM
34
変調回路
OSC (13.56MHz)
42
41
27
23
26 発振器
フラッシュメモリ
22
リーダ/ライタ 1

[Drawing 3]


51 IC
61 RF インタフェース部
64 演算部
電圧
レギュレータ
82
53
52
81
62
ASK 復調部
BPSK
復調回路
暗号／復号部
91
92
シーケンサ
65
ROM
66
EEPROM
67
RAM
発振回路
PLL 部
63
93
68
83
ASK 変調部
BPSK
変調回路
パリティ
演算部
84

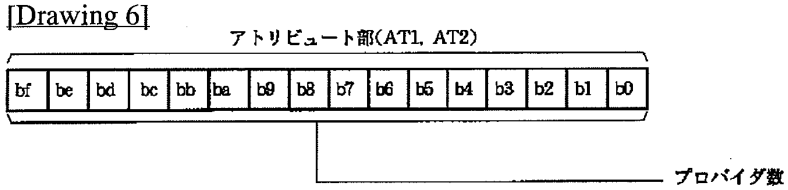
カード2

[Drawing 4]


40bytes
ユーザブロック #00
ユーザブロック #01
ユーザブロック #02
ユーザブロック #03
ユーザブロック #04
ユーザブロック
プロバイダ領域定義ブロック #XX
プロバイダ領域定義ブロック #02 #fb
プロバイダ領域定義ブロック #01 #fc
プロバイダ領域定義ブロック #00 #fd
共通領域定義ブロック #fe
システム ID ブロック #ff
MAX 253 blocks
MAX 255 blocks
256 blocks
D00 D01 D02 D03 D04 D17 D18 D19 D1a D1b D1c D1d D1e D1f AT1 AT2 P0 P1 P2 P3 P4 P5
パリティ部(RS コード)
アトリビュート部
データ部

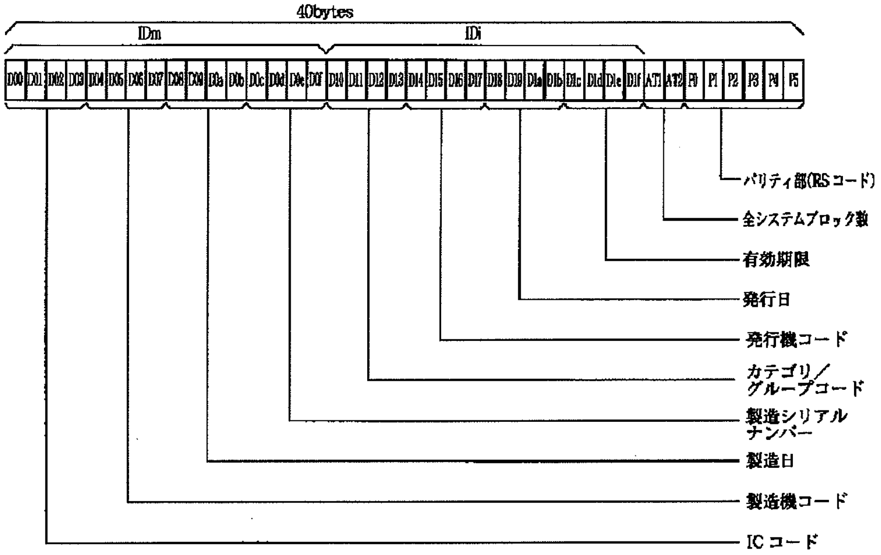
EEPROM 66

[Drawing 6]


アトリビュート部(AT1, AT2)
bf be bd bc bb ba b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
プロバイダ数

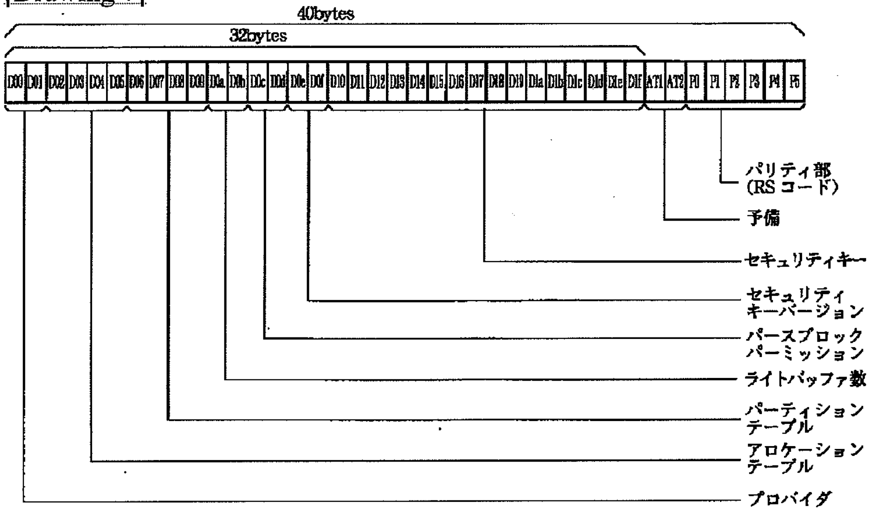
システム ID ブロック

[Drawing 5]


40bytes
IDm
IDi
D00 D01 D02 D03 D04 D05 D06 D07 D08 D09 D0a D0b D0c D0d D0e D0f D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D1a D1b D1c D1d D1e D1f AT1 AT2 P0 P1 P2 P3 P4 P5
パリティ部(RSコード)
全システムブロック数
有効期限
発行日
発行機コード
カテゴリ／グループコード
製造シリアルナンバー
製造日
製造機コード
IC コード

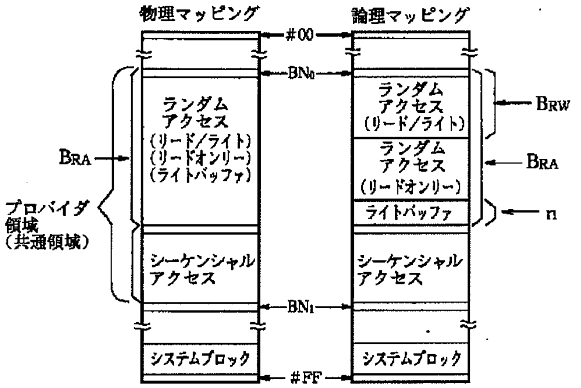
システム ID ブロック

[Drawing 7]

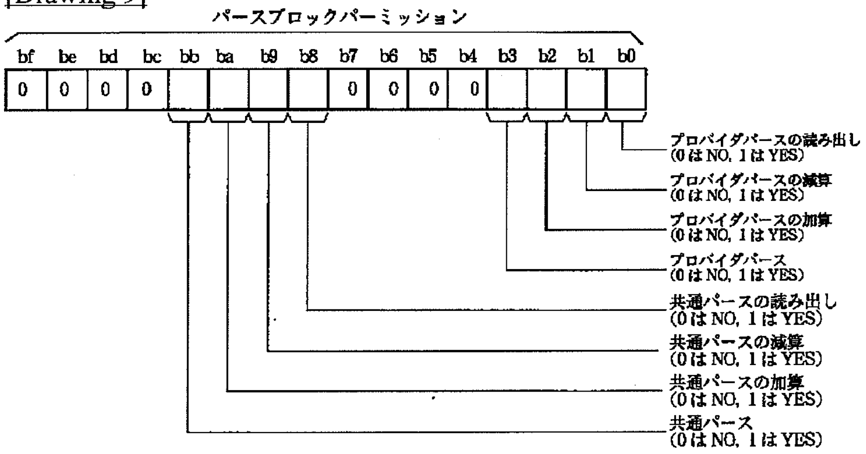

40bytes
32bytes
D00 D01 D02 D03 D04 D05 D06 D07 D08 D09 D0a D0b D0c D0d D0e D0f D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D1a D1b D1c D1d D1e D1f AT1 AT2 P0 P1 P2 P3 P4 P5
パリティ部(RS コード)
予備
セキュリティキー
セキュリティキーバージョン
パースブロックパーミッション
ライトバッファ数
パーティションテーブル
アロケーションテーブル
プロバイダコード

共通領域定義ブロック
（プロバイダ領域定義ブロック）

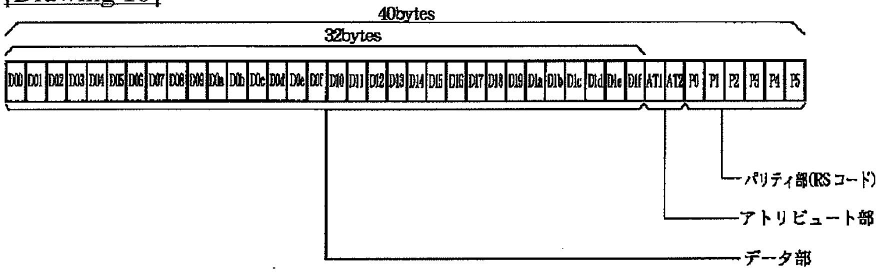
[Drawing 8]


物理マッピング
論理マッピング
#00
BN0
BN1
#FF
ランダム
アクセス
(リード/ライト)
(リードオンリー)
(ライトバッファ)
BRA
プロバイダ
領域
(共通領域)
シーケンシャル
アクセス
システムブロック
ランダム
アクセス
(リード/ライト)
BRW
ランダム
アクセス
(リードオンリー)
BRA
ライトバッファ
n
シーケンシャル
アクセス
システムブロック

[Drawing 9]

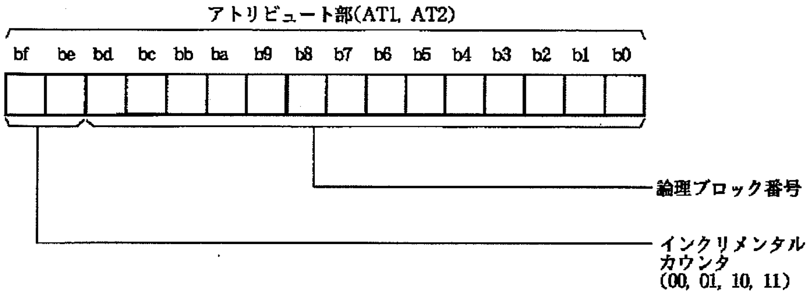

パースブロックパーミッション
bf
be
bd
bc
bb
ba
b9
b8
b7
b6
b5
b4
b3
b2
b1
b0
0
0
0
0
0
0
0
0
プロバイダパースの読み出し
(0はNO, 1はYES)
プロバイダパースの減算
(0はNO, 1はYES)
プロバイダパースの加算
(0はNO, 1はYES)
プロバイダパース
(0はNO, 1はYES)
共通パースの読み出し
(0はNO, 1はYES)
共通パースの減算
(0はNO, 1はYES)
共通パースの加算
(0はNO, 1はYES)
共通パース
(0はNO, 1はYES)

[Drawing 10]


40bytes
32bytes
D00 D01 D02 D03 D04 D05 D06 D07 D08 D09 D0a D0b D0c D0d D0e D0f D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D1a D1b D1c D1d D1e D1f AT1 AT2 P0 P1 P2 P3 P4 P5
パリティ部(RSコード)
アトリビュート部
データ部

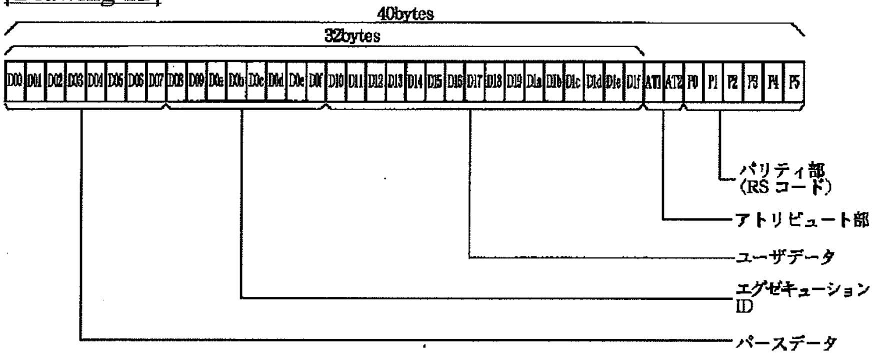
ユーザブロック

[Drawing 11]


アトリビュート部(AT1, AT2)
bf
be
bd
bc
bb
ba
b9
b8
b7
b6
b5
b4
b3
b2
b1
b0
論理ブロック番号
インクリメンタル
カウンタ
(00, 01, 10, 11)

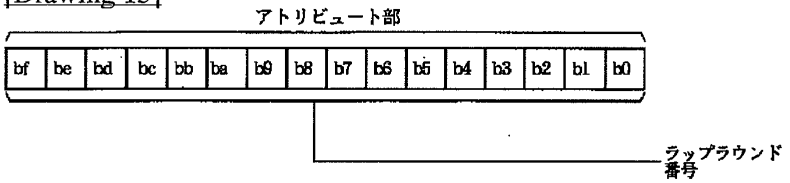
ランダムアクセス領域のユーザブロック

[Drawing 12]


40bytes
32bytes
D00
D01
D02
D03
D04
D05
D06
D07
D08
D09
D0a
D0b
D0c
D0d
D0e
D0f
D10
D11
D12
D13
D14
D15
D16
D17
D18
D19
D1a
D1b
D1c
D1d
D1e
D1f
AT1
AT2
P0
P1
P2
P3
P4
P5
パリティ部
(RS コード)
アトリビュート部
ユーザデータ
エグゼキューション
ID
パースデータ

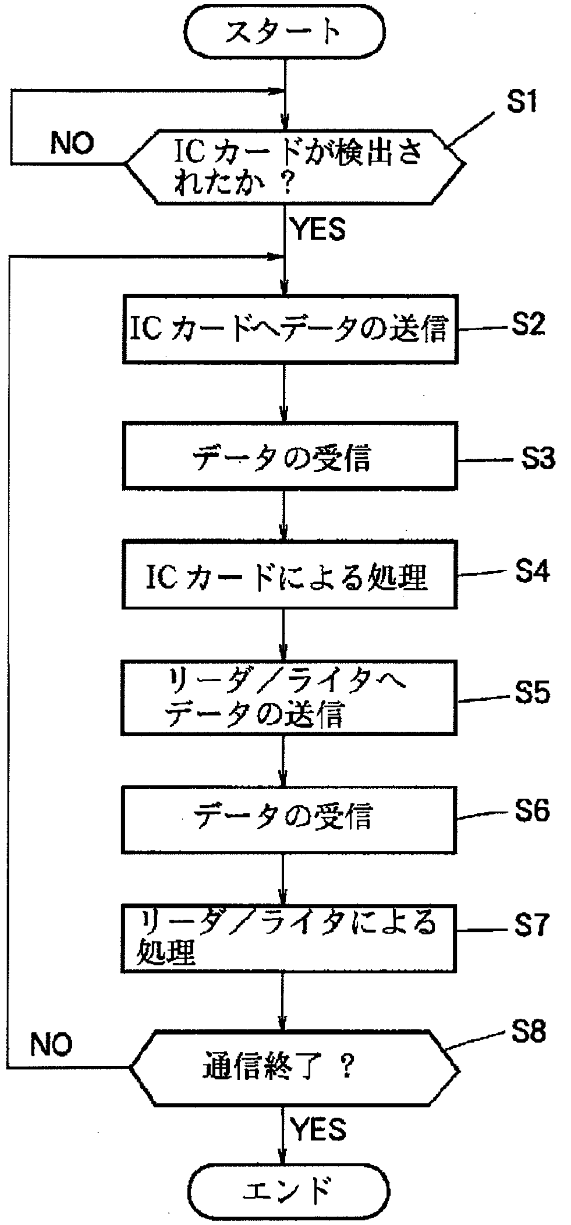
パースブロック

[Drawing 13]

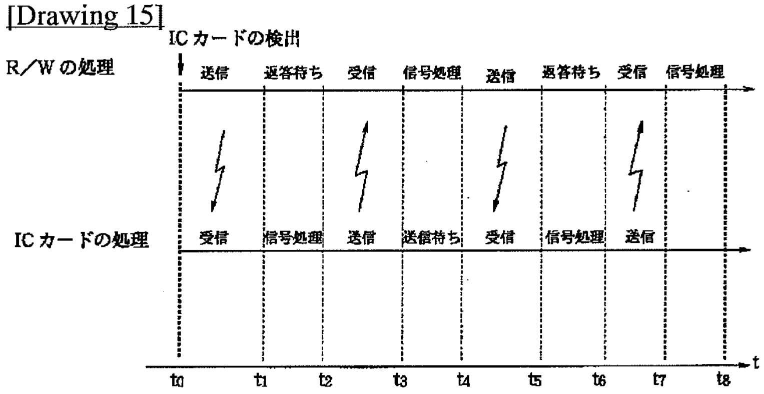

アトリビュート部
bf
be
bd
bc
bb
ba
b9
b8
b7
b6
b5
b4
b3
b2
b1
b0
ラップラウンド
番号

シーケンシャルアクセス領域のユーザブロック

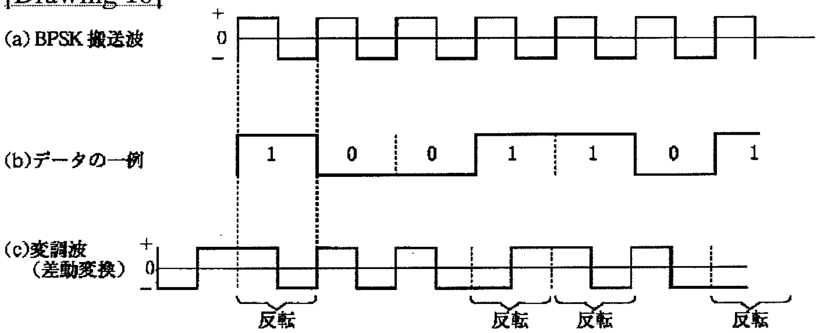
[Drawing 14]


スタート
S1
NO
ICカードが検出されたか？
YES
ICカードへデータの送信
S2
データの受信
S3
ICカードによる処理
S4
リーダ／ライタへデータの送信
S5
データの受信
S6
リーダ／ライタによる処理
S7
S8
NO
通信終了？
YES
エンド

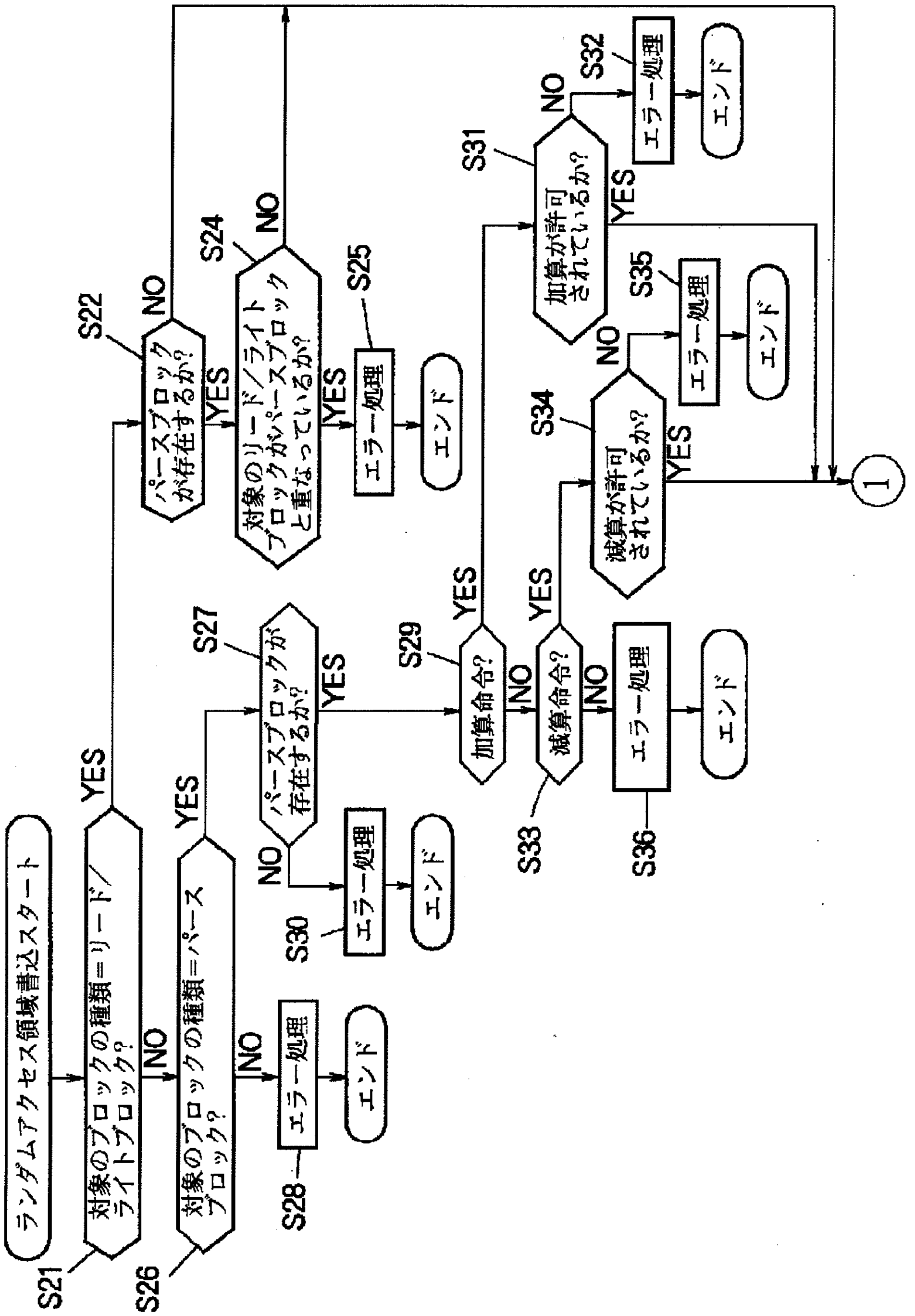
[Drawing 15]


ICカードの検出
R／Wの処理
送信
返答待ち
受信
信号処理
送信
返答待ち
受信
信号処理
ICカードの処理
受信
信号処理
送信
送信待ち
受信
信号処理
送信
t
t_0
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8

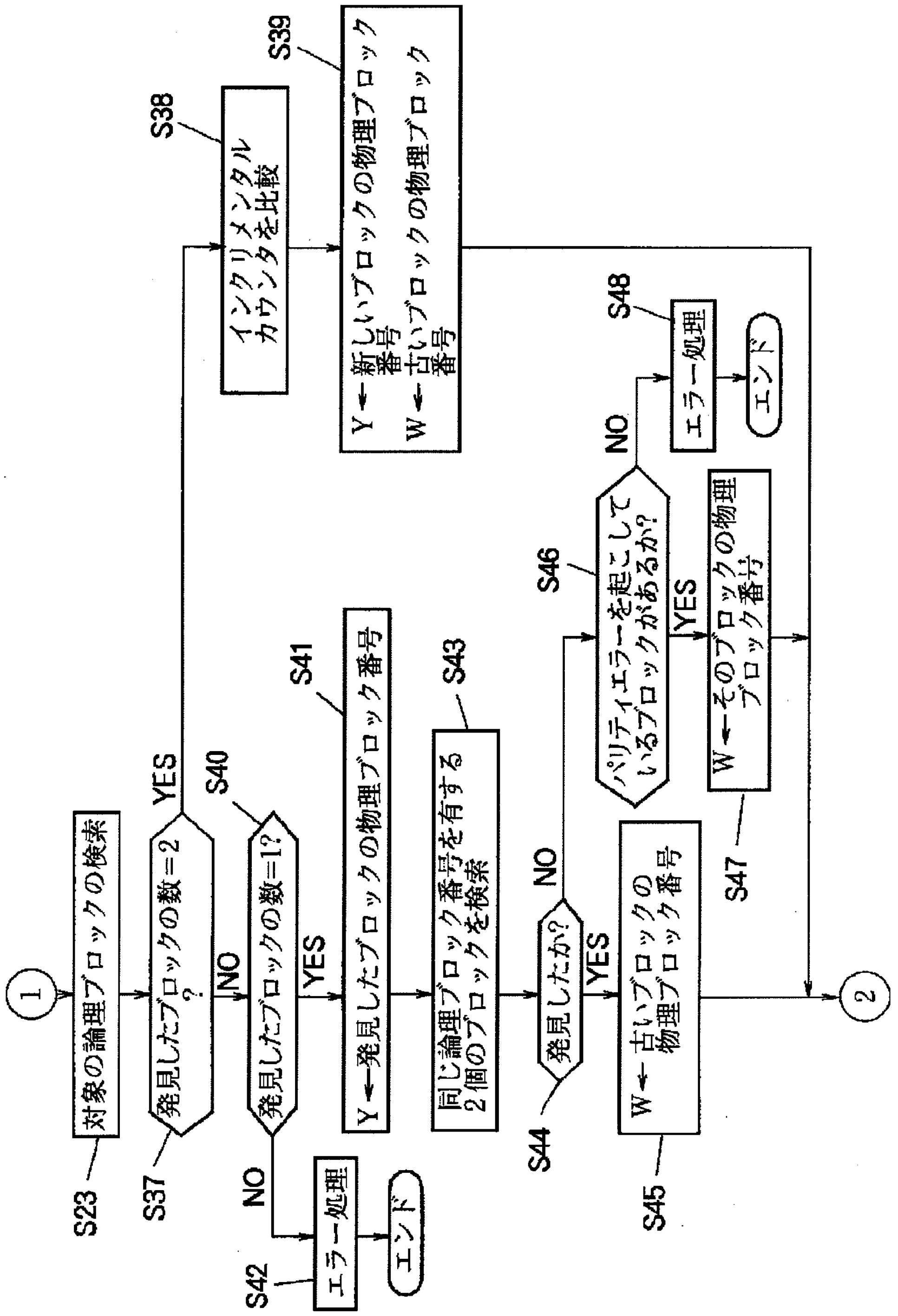
[Drawing 16]


(a) BPSK 搬送波
+
0
−
(b)データの一例
1
0
0
1
1
0
1
(c)変調波
（差動変換）
+
0
−
反転
反転
反転
反転

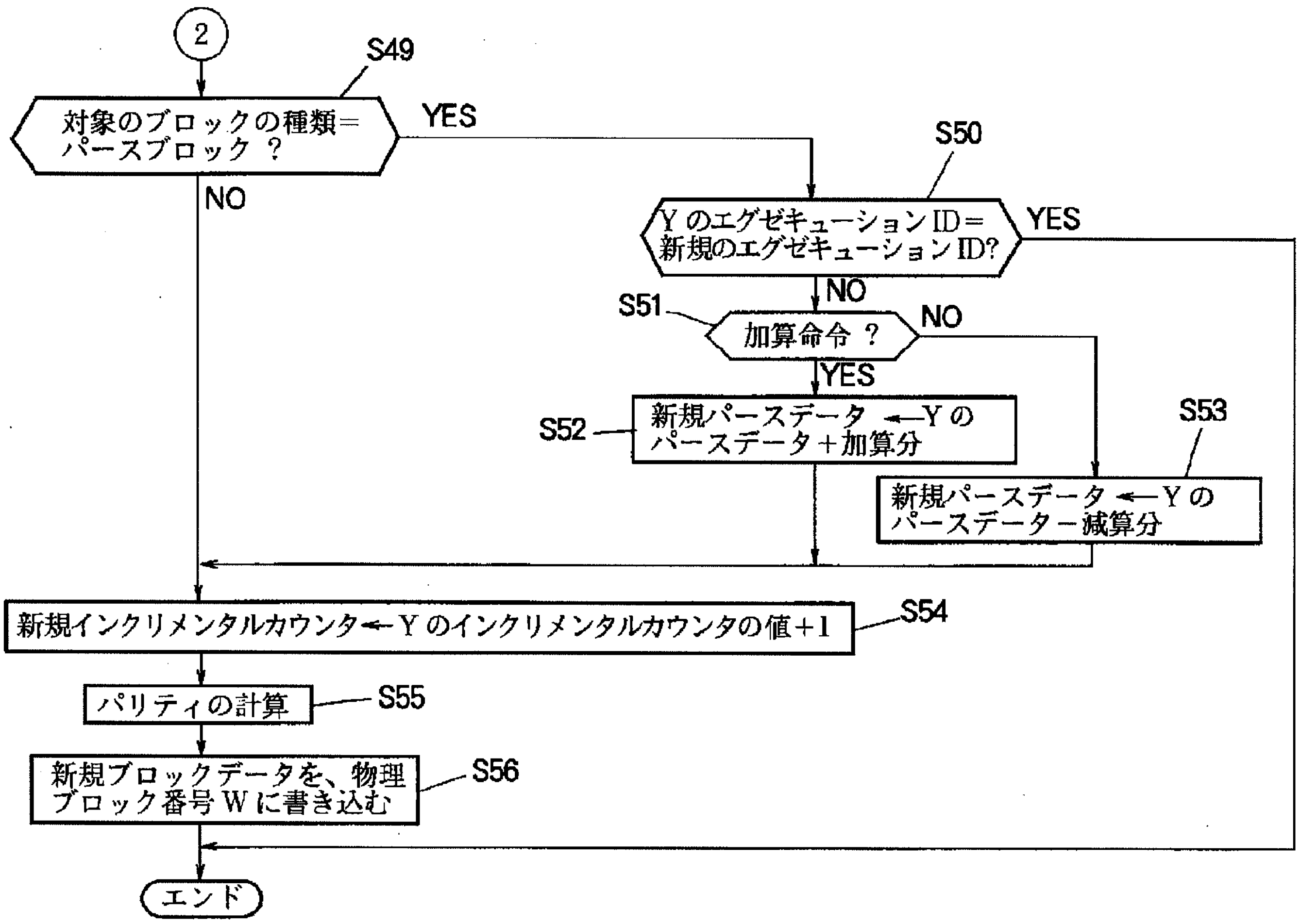
[Drawing 17]


ランダムアクセス領域書込スタート
S21
対象のブロックの種類＝リード／ライトブロック？
YES
NO
S22
パースブロックが存在するか？
NO
YES
S24
対象のリード／ライトブロックがパースブロックと重なっているか？
NO
YES
S25
エラー処理
エンド
S26
対象のブロックの種類＝パースブロック？
YES
NO
S28
エラー処理
エンド
S27
パースブロックが存在するか？
YES
NO
S30
エラー処理
エンド
S29
加算命令？
YES
NO
S31
加算が許可されているか？
YES
NO
S32
エラー処理
エンド
S33
減算命令？
YES
NO
S36
エラー処理
エンド
S34
減算が許可されているか？
YES
NO
S35
エラー処理
エンド
①

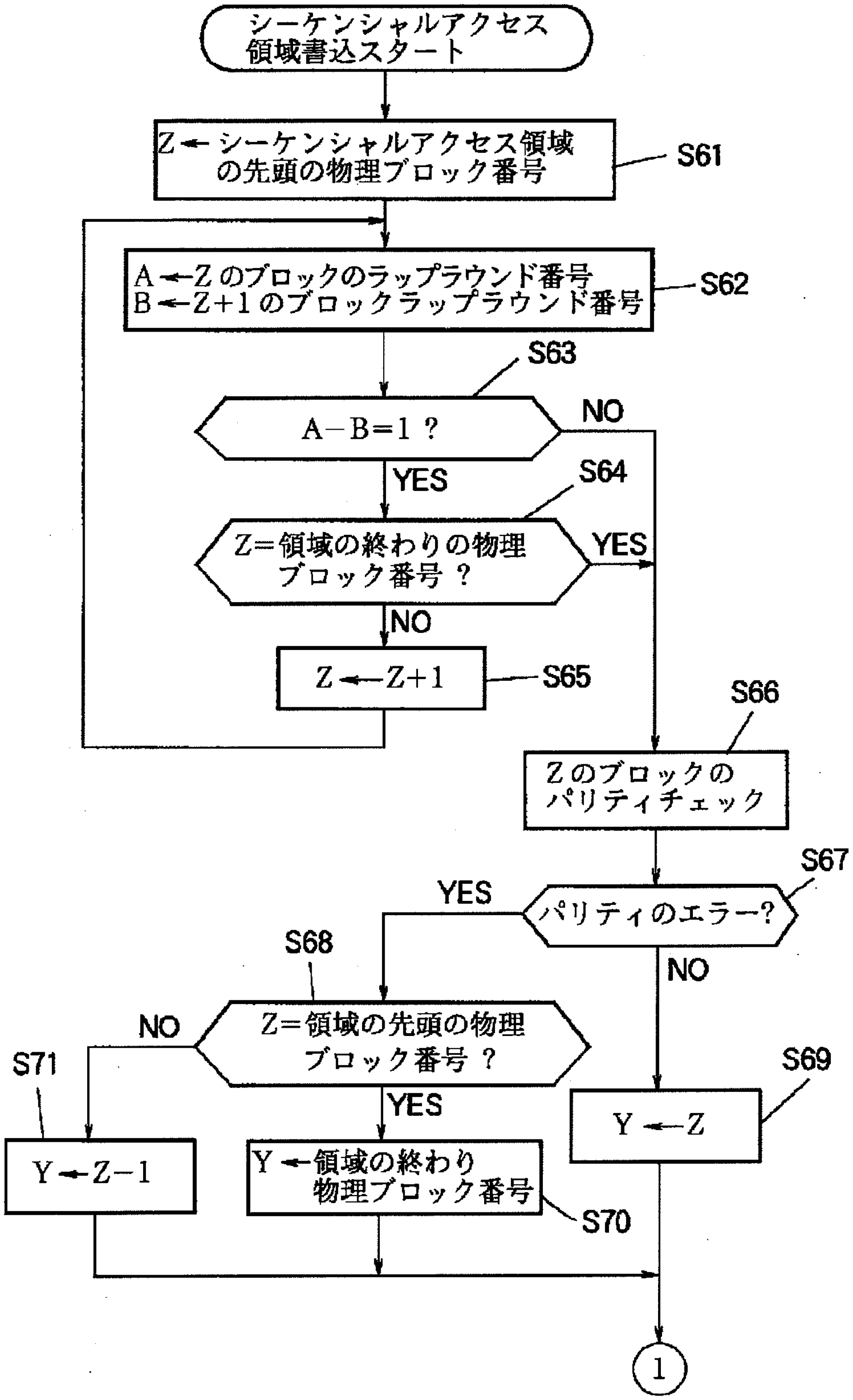
[Drawing 18]


①
S23
対象の論理ブロックの検索
S37
発見したブロックの数＝2？
YES
NO
S38
インクリメンタルカウンタを比較
S39
Y←新しいブロックの物理ブロック番号
W←古いブロックの物理ブロック番号
S40
発見したブロックの数＝1？
NO
S42
エラー処理
エンド
YES
S41
Y←発見したブロックの物理ブロック番号
S43
同じ論理ブロック番号を有する2個のブロックを検索
S44
発見したか？
YES
S45
W←古いブロックの物理ブロック番号
NO
S46
パリティエラーを起こしているブロックがあるか？
YES
S47
W←そのブロックの物理ブロック番号
NO
S48
エラー処理
エンド
②

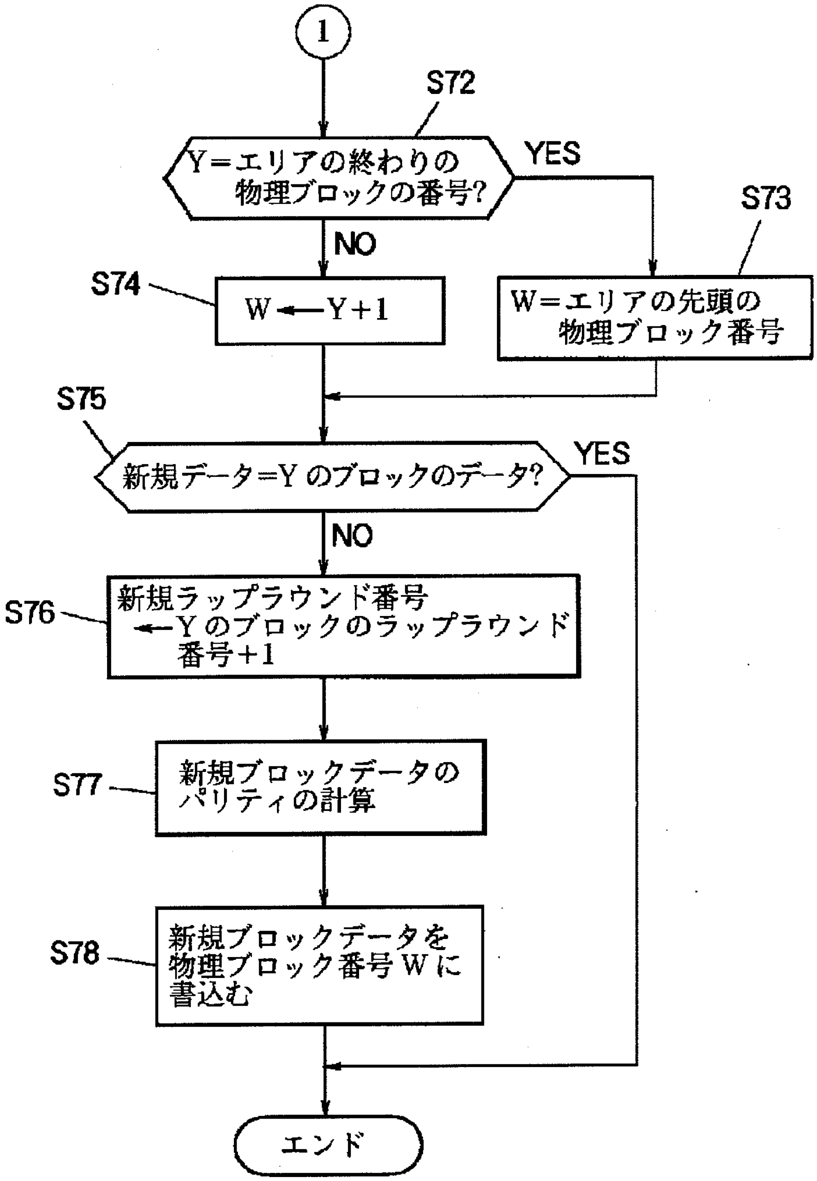
[Drawing 19]


2
S49
対象のブロックの種類＝
パースブロック ？
YES
NO
S50
Y のエグゼキューションID＝
新規のエグゼキューションID?
YES
NO
S51
加算命令 ？
NO
YES
S52
新規パースデータ ←Y の
パースデータ＋加算分
S53
新規パースデータ ←Y の
パースデーター減算分
新規インクリメンタルカウンタ←Y のインクリメンタルカウンタの値＋1
S54
パリティの計算
S55
新規ブロックデータを、物理
ブロック番号 W に書き込む
S56
エンド

[Drawing 20]


シーケンシャルアクセス
領域書込スタート
Z←シーケンシャルアクセス領域
の先頭の物理ブロック番号
S61
A←Zのブロックのラップラウンド番号
B←Z+1のブロックラップラウンド番号
S62
S63
A−B=1 ?
NO
YES
S64
Z=領域の終わりの物理
ブロック番号 ?
YES
NO
Z←Z+1
S65
S66
Zのブロックの
パリティチェック
S67
YES
パリティのエラー?
NO
S68
NO
Z=領域の先頭の物理
ブロック番号 ?
S71
YES
S69
Y←Z−1
Y←領域の終わり
物理ブロック番号
Y←Z
S70
1

[Drawing 21]


1
S72
Y=エリアの終わりの物理ブロックの番号?
YES
NO
S73
W=エリアの先頭の物理ブロック番号
S74
W←Y+1
S75
新規データ=Y のブロックのデータ?
YES
NO
S76
新規ラップラウンド番号←Y のブロックのラップラウンド番号+1
S77
新規ブロックデータのパリティの計算
S78
新規ブロックデータを物理ブロック番号 W に書込む
エンド

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CORRECTION OR AMENDMENT

[Kind of official gazette] Printing of amendment by the convention of 2 of Article 17 of Patent Law
[Section partition] The 3rd partition of the 6th section
[Publication date] May 19, Heisei 17 (2005. 5.19)

[Publication No.] JP,10-105661,A
[Date of Publication] April 24, Heisei 10 (1998. 4.24)
[Application number] Japanese Patent Application No. 9-171430
[The 7th edition of International Patent Classification]

```
G06K 17/00
G06F  3/06
G06F  3/08
G06F 12/02
G06K 19/07
```

[FI]

```
G06K 17/00       F
G06K 17/00       D
G06F  3/06   301 J
G06F  3/08       C
G06F 12/02   510 A
G06K 19/00       N
```

[Procedure revision]
[Filing Date] June 25, Heisei 16 (2004. 6.25)
[Procedure amendment 1]
[Document to be Amended] Specification
[Item(s) to be Amended] The name of invention
[Method of Amendment] Modification
[The contents of amendment]
[Title of the Invention] It is a record medium to an information processor and the information processing approach, reader/writer and the access approach, and a list.
[Procedure amendment 2]
[Document to be Amended] Specification
[Item(s) to be Amended] Claim
[Method of Amendment] Modification
[The contents of amendment]
[Claim(s)]
[Claim 1]

A receiving means to receive the command from the equipment of the provider who offers a predetermined system,
A processing means to process said command,
A transmitting means to transmit the result of said processing,
A storage means by which the user block field where one or more user blocks are managed in the unit of predetermined magnitude, and the domain-defined block field which defines use of said user block are formed while memorizing the data which said provider's equipment uses
Preparation,
Said domain-defined block field consists of two or more domain-defined blocks,
The data which specify the access privilege to the data which specify the user block to be used in each domain-defined block, and said user block are memorized,
Said command is processed using said data currently recorded on said domain-defined block field.
The information processor characterized by things.
[Claim 2]
Said user block field is assigned and formed in the free area which is not used as said domain-defined block field.
The information processor according to claim 1 characterized by things.
[Claim 3]
Said two or more domain-defined blocks specify an each of said provider access privilege which is different, respectively.
The information processor according to claim 1 characterized by things.
[Claim 4]
Said command is processed using two or more domain-defined blocks with which the data which specify said access privilege differ.
The information processor according to claim 3 characterized by things.
[Claim 5]
The data which specify said access privilege are data which specify access of either [to the user block of said user block field] read/write or a read-only either.
The information processor according to claim 4 characterized by things.
[Claim 6]
While memorizing the data which the equipment of the provider who offers a predetermined system uses The user block field where one or more user blocks are managed in the unit of predetermined magnitude, It is a storage means by which the domain-defined block field which defines use of said user block is formed. Said domain-defined block field In the information processing approach of the information processor which consists of two or more domain-defined blocks, and equips each domain-defined block with a storage means by which the data which specify the access privilege to the data which specify the user block to be used, and said user block are memorized,
The receiving step which receives the command from said provider's equipment,
The processing step which processes said command using said domain-defined block field,
The transmitting step which transmits the result of said processing
****** -- the information processing approach characterized by things.
[Claim 7]
While memorizing the data which the equipment of the provider who offers a predetermined system uses The user block field where one or more user blocks are managed in the unit of predetermined magnitude, It is a storage means by which the domain-defined block field which defines use of said user block is formed. Said domain-defined block field In the program for information processing of the information processor which consists of two or more domain-defined blocks, and equips each domain-defined block with a storage means by which the data which specify the access privilege to the data which specify the user block to be used, and said user block are memorized,
The reception-control step which controls reception of the command from said provider's equipment,
The processing step which processes said command using said domain-defined block field,

The transmission-control step which controls the transmission as a result of said processing
****** -- the record medium with which the program which the computer characterized by things can read is recorded.

[Claim 8]

It is the reader/writer for accessing said user block field of an information processor equipped with a storage means by which the user block field where the data which the equipment of the provider who one or more user blocks are managed per block of predetermined magnitude, and offers a predetermined system uses are memorized is formed,

A data-processing means to process the received data received from the transmit data transmitted to said information processor, and said information processor,

A modulation means to modulate said transmit data,

A recovery means to restore to said received data

Implication,

Said transmit data is a command for accessing said user block controlled based on the data which specify the access privilege to the data which specify said user block memorized by the domain-defined block of said storage means, and said user block.

Reader/writer characterized by things.

[Claim 9]

Said command is a command for accessing said user block field based on the data which specify said access privilege memorized by said two or more [mutually different] domain-defined blocks.

The information processor according to claim 8 characterized by things.

[Claim 10]

It is the access approach which accesses said user block field of an information processor equipped with a storage means by which the user block field where the data which the equipment of the provider who one or more user blocks are managed per block of predetermined magnitude, and offers a predetermined system uses are memorized is formed,

The appeal step which performs appeal to said information processor,

The transmitting step which transmits a command to said information processor which answered said appeal,

The receiving step which receives the transmitting result of said command processed by said information processor,

The processing step which processes said transmitting result

Implication,

Said transmit data is a command for accessing said user block controlled based on the data which specify the access privilege to the data which specify said user block memorized by the domain-defined block of said storage means, and said user block.

The access approach characterized by things.

[Claim 11]

Said command is a command for accessing said user block field based on the data which specify said access privilege memorized by said two or more [mutually different] domain-defined blocks.

The access approach according to claim 10 characterized by things.

[Claim 12]

The data which specify said access privilege are data which specify access of either [to said user block of said user block field] read/write or a read-only either.

The access approach according to claim 10 characterized by things.

[Claim 13]

It is the program for access processing which accesses said user block field of an information processor equipped with a storage means by which the user block field where the data which the equipment of the provider who one or more user blocks are managed per block of predetermined magnitude, and offers a predetermined system uses are memorized is formed,

The appeal step which performs appeal to said information processor,

The transmission-control step which controls transmission of a command to said information processor which answered said appeal,
The reception-control step which controls the reception of a transmitting result to said command processed by said information processor,
The processing step which processes said transmitting result
Implication,
Said transmit data is a command for accessing said user block controlled based on the data which specify the access privilege to the data which specify said user block memorized by the domain-defined block of said storage means, and said user block.
The record medium with which the program which the computer characterized by things can read is recorded.
[Procedure amendment 3]
[Document to be Amended] Specification
[Item(s) to be Amended] 0001
[Method of Amendment] Modification
[The contents of amendment]
[0001]
[Field of the Invention]
This invention relates to a record medium at the information processor and the information processing approach of receiving the command from a predetermined user in an information processor and the information processing approach, reader/writer and the access approach, and a list especially, processing the command in them, and transmitting the result of processing to them about a record medium, reader/writer and the access approach, and a list.
[Procedure amendment 4]
[Document to be Amended] Specification
[Item(s) to be Amended] 0012
[Method of Amendment] Modification
[The contents of amendment]
[0012]
[Means for Solving the Problem]
A receiving means to receive the command from the equipment of the provider whom an information processor according to claim 1 provides with a predetermined system, While remembering the data which a provider's equipment uses to be a processing means to process a command, and a transmitting means to transmit the result of processing The user block field where one or more user blocks are managed in the unit of predetermined magnitude, A storage means by which the domain-defined block field which defines use of a user block is formed is included. A domain-defined block field It consists of two or more domain-defined blocks. For each domain-defined block The data which specify the access privilege to the data and the user block which specify the user block to be used are memorized, and it is characterized by processing a command using the data currently recorded on the domain-defined block field.
[Procedure amendment 5]
[Document to be Amended] Specification
[Item(s) to be Amended] 0013
[Method of Amendment] Modification
[The contents of amendment]
[0013]
The information processing approach according to claim 6 is characterized by including the receiving step which receives the command from a provider's equipment, the processing step which processes a command using a domain-defined block field, and the transmitting step which transmits the result of processing.
[Procedure amendment 6]

[Document to be Amended] Specification
[Item(s) to be Amended] 0014
[Method of Amendment] Modification
[The contents of amendment]
[0014]
The program of a record medium according to claim 7 is characterized by including the reception-control step which controls reception of the command from a provider's equipment, the processing step which processes a command using a domain-defined block field, and the transmission-control step which controls the transmission as a result of processing.
[Procedure amendment 7]
[Document to be Amended] Specification
[Item(s) to be Amended] 0015
[Method of Amendment] Modification
[The contents of amendment]
[0015]
Transmit data is characterized by to be a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block including a data-processing means process the received data which received from the transmit data and the information processor which reader/writer according to claim 8 transmits to an information processor, a modulation means modulate transmit data, and a recovery means restore to received data.
[Procedure amendment 8]
[Document to be Amended] Specification
[Item(s) to be Amended] 0016
[Method of Amendment] Modification
[The contents of amendment]
[0016]
The appeal step to which the access approach according to claim 10 performs appeal to an information processor, The transmitting step which transmits a command to the information processor which answered appeal, The receiving step which receives the transmitting result of the command processed by the information processor, and the processing step which processes a transmitting result are included. Transmit data It is characterized by being a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block.
[Procedure amendment 9]
[Document to be Amended] Specification
[Item(s) to be Amended] 0017
[Method of Amendment] Modification
[The contents of amendment]
[0017]
The program of a record medium according to claim 13 The appeal step which performs appeal to an information processor, and the transmission-control step which controls transmission of a command to the information processor which answered appeal, The reception-control step which controls the reception of a transmitting result to the command processed by the information processor, and the processing step which processes a transmitting result are included. Transmit data It is characterized by being a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block.
[Procedure amendment 10]
[Document to be Amended] Specification
[Item(s) to be Amended] 0018

[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 11]
[Document to be Amended] Specification
[Item(s) to be Amended] 0019
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 12]
[Document to be Amended] Specification
[Item(s) to be Amended] 0020
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 13]
[Document to be Amended] Specification
[Item(s) to be Amended] 0021
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 14]
[Document to be Amended] Specification
[Item(s) to be Amended] 0022
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 15]
[Document to be Amended] Specification
[Item(s) to be Amended] 0023
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 16]
[Document to be Amended] Specification
[Item(s) to be Amended] 0024
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 17]
[Document to be Amended] Specification
[Item(s) to be Amended] 0025
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 18]
[Document to be Amended] Specification
[Item(s) to be Amended] 0026
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 19]
[Document to be Amended] Specification
[Item(s) to be Amended] 0027
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 20]
[Document to be Amended] Specification
[Item(s) to be Amended] 0028
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 21]
[Document to be Amended] Specification
[Item(s) to be Amended] 0029
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 22]
[Document to be Amended] Specification
[Item(s) to be Amended] 0030
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 23]
[Document to be Amended] Specification
[Item(s) to be Amended] 0031
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 24]
[Document to be Amended] Specification
[Item(s) to be Amended] 0032
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 25]
[Document to be Amended] Specification
[Item(s) to be Amended] 0033
[Method of Amendment] Modification
[The contents of amendment]
[0033]
In an information processor according to claim 1, the command from the equipment of the provider who offers a predetermined system is received, a command is processed, the result of processing is transmitted, and while memorizing the data which a provider's equipment uses, the user block field where one or more user blocks are managed in the unit of predetermined magnitude, and the domain-defined block field which defines use of a user block are formed. A domain-defined block field consists of two or more domain-defined blocks, the data which specify the access privilege to the data and the user block which specify the user block to be used for each domain-defined block are memorized, and a command is processed using the data currently recorded on the domain-defined block field.
[Procedure amendment 26]

[Document to be Amended] Specification
[Item(s) to be Amended] 0034
[Method of Amendment] Modification
[The contents of amendment]
[0034]
In the information processing approach and a record medium according to claim 7 according to claim 6, the command from a provider's equipment is received, a command is processed using a domain-defined block field, and the result of processing is transmitted.
[Procedure amendment 27]
[Document to be Amended] Specification
[Item(s) to be Amended] 0035
[Method of Amendment] Modification
[The contents of amendment]
[0035]
In reader/writer according to claim 8, processing of the received data received from the transmit data and the information processor which are transmitted to an information processor is performed, transmit data is modulated, and received data get over. Moreover, let transmit data be a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block.
[Procedure amendment 28]
[Document to be Amended] Specification
[Item(s) to be Amended] 0036
[Method of Amendment] Modification
[The contents of amendment]
[0036]
In the access approach and a record medium according to claim 13 according to claim 10, appeal is performed to an information processor, a command is transmitted to the information processor which answered appeal, the transmitting result of the command processed by the information processor is received, and a transmitting result is processed. Let transmit data be a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block.
[Procedure amendment 29]
[Document to be Amended] Specification
[Item(s) to be Amended] 0037
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 30]
[Document to be Amended] Specification
[Item(s) to be Amended] 0038
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 31]
[Document to be Amended] Specification
[Item(s) to be Amended] 0039
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 32]

[Document to be Amended] Specification
[Item(s) to be Amended] 0041
[Method of Amendment] Modification
[The contents of amendment]
[0041]

A receiving means to receive the command from the equipment of the provider whom an information processor according to claim 1 provides with a predetermined system (for example, the antenna 53, RF interface section 61, and the BPSK demodulator circuit 62 of drawing 3), A processing means to process a command (for example, sequence 91 of drawing 3), While remembering the data which a provider's equipment uses to be a transmitting means (for example, the antenna 53, RF interface section 61, and the BPSK modulation circuit 68 of drawing 3) to transmit the result of processing The user block field where one or more user blocks are managed in the unit of predetermined magnitude, A storage means (for example, EEPROM66 of drawing 3) by which the domain-defined block field which defines use of a user block is formed is included. A domain-defined block field It consists of two or more domain-defined blocks. For each domain-defined block The data which specify the access privilege to the data and the user block which specify the user block to be used are memorized, and it is characterized by processing a command using the data currently recorded on the domain-defined block field.

[Procedure amendment 33]
[Document to be Amended] Specification
[Item(s) to be Amended] 0042
[Method of Amendment] Modification
[The contents of amendment]
[0042]

Reader/writer according to claim 8 is managed per block of magnitude predetermined in one or more user blocks. A storage means by which the user block field where the data which the equipment of the provider who offers a predetermined system uses are memorized is formed It is the reader/writer which accesses the user block field of an information processor equipped with (for example, EEPROM66 of drawing 3). A data-processing means to process the received data received from the transmit data and the information processor which are transmitted to an information processor (for example, SPU32 of drawing 2), A modulation means (for example, modulation circuit 23 of drawing 2) to modulate transmit data, and a recovery means (for example, demodulator circuit 25 of drawing 2) to restore to received data are included. Transmit data It is characterized by being a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block.

[Procedure amendment 34]
[Document to be Amended] Specification
[Item(s) to be Amended] 0043
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 35]
[Document to be Amended] Specification
[Item(s) to be Amended] 0044
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 36]
[Document to be Amended] Specification
[Item(s) to be Amended] 0045
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 37]
[Document to be Amended] Specification
[Item(s) to be Amended] 0046
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 38]
[Document to be Amended] Specification
[Item(s) to be Amended] 0047
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 39]
[Document to be Amended] Specification
[Item(s) to be Amended] 0218
[Method of Amendment] Modification
[The contents of amendment]
[0218]
[Effect of the Invention]

Like the above, the command from the equipment of the provider who offers a predetermined system according to the information processor according to claim 1 is received, a command is processed, the result of processing is transmitted, and while memorizing the data which a provider's equipment uses, the user block field where one or more user blocks are managed in the unit of predetermined magnitude, and the domain-defined block field which defines use of a user block are formed. Moreover, a domain-defined block field consists of two or more domain-defined blocks, the data which specify the access privilege to the data and the user block which specify the user block to be used for each domain-defined block are memorized, and since the command was processed using the data currently recorded on the domain-defined block field, two or more access privileges which can be set to a predetermined storage region can be granted to a predetermined user. The same storage region can be assigned to two or more users. A different access privilege in a predetermined storage region can be granted to two or more users.

[Procedure amendment 40]
[Document to be Amended] Specification
[Item(s) to be Amended] 0219
[Method of Amendment] Modification
[The contents of amendment]
[0219]

Since according to the information processing approach and a record medium according to claim 7 according to claim 6 the command from a provider's equipment is received, a command is processed using a domain-defined block field and the result of processing was transmitted, two or more access privileges which can be set to a predetermined storage region can be granted to a predetermined user. The same storage region can be assigned to two or more users. A different access privilege in a predetermined storage region can be granted to two or more users.

[Procedure amendment 41]
[Document to be Amended] Specification
[Item(s) to be Amended] 0220
[Method of Amendment] Modification
[The contents of amendment]
[0220]

According to reader/writer according to claim 8, processing of the received data received from the transmit data and the information processor which are transmitted to an information processor is performed, transmit data is modulated, and received data get over. Since it was made for transmit data to be a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block, it can grant two or more access privileges which can be set to a predetermined storage region to a predetermined user. The same storage region can be assigned to two or more users. A different access privilege in a predetermined storage region can be granted to two or more users.

[Procedure amendment 42]
[Document to be Amended] Specification
[Item(s) to be Amended] 0221
[Method of Amendment] Modification
[The contents of amendment]
[0221]

According to the record medium of the access approach according to claim 10 and claim 13, appeal is performed to an information processor, a command is transmitted to the information processor which answered appeal, the transmitting result of the command processed by the information processor is received, and a transmitting result is processed. Moreover, since it was made for transmit data to be a command for accessing the user block controlled based on the data which specify the access privilege to the data and the user block which are memorized by the domain-defined block of a storage means, and which specify a user block, it can grant two or more access privileges which can be set to a predetermined storage region to a predetermined user. The same storage region can be assigned to two or more users. A different access privilege in a predetermined storage region can be granted to two or more users.

[Procedure amendment 43]
[Document to be Amended] Specification
[Item(s) to be Amended] 0222
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 44]
[Document to be Amended] Specification
[Item(s) to be Amended] 0223
[Method of Amendment] Deletion
[The contents of amendment]

[Procedure amendment 45]
[Document to be Amended] Specification
[Item(s) to be Amended] 0224
[Method of Amendment] Deletion
[The contents of amendment]

[Translation done.]

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-221223
(43)Date of publication of application : 30.08.1996

(51)Int.Cl.	G06F 3/08 G11C 16/06

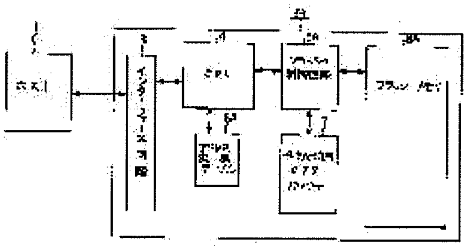
(21)Application number :	07-028287	(71)Applicant :	MITSUBISHI ELECTRIC CORP
(22)Date of filing :	16.02.1995	(72)Inventor :	KORIN MEESON

(54) SEMICONDUCTOR DISK DEVICE

(57)Abstract:

PURPOSE: To reduce address conversion table capacity for data control.

CONSTITUTION: This device is provided with a flash memory 8A having plural erasing blocks composed of erasing block information storage areas storing the numbers of times of erasure of erasing blocks, plural data storage areas storing data and the logical sector address storage area for every data storage area storing logical sector address, an address conversion table 5A for converting the logical sector address into a physical erasing block number, and a CPU 4 converting the logical sector address(LSA) inputted based on the address conversion table 5A, finding out the latest pertinent data storage area based on the logical sector address inputted within the pertinent physical erasing block on the flash memory 8A and reading the contents of the latest pertinent data storage area.



LEGAL STATUS

[Date of request for examination]	14.05.2001
[Date of sending the examiner's decision of rejection]	19.04.2005
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	3706167
[Date of registration]	05.08.2005
[Number of appeal against examiner's decision of rejection]	2005-009410
[Date of requesting appeal against examiner's decision of rejection]	19.05.2005
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平8－221223

(43)公開日　平成８年(1996)８月30日

(51)Int.Cl.^{6}	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　3/08			Ｇ０６Ｆ　3/08	Ｈ	
Ｇ１１Ｃ　16/06			Ｇ１１Ｃ　17/00	３０９Ｇ	

審査請求　未請求　請求項の数３　ＯＬ　（全 17 頁）

(21)出願番号　特願平7－28287

(22)出願日　平成７年(1995)２月16日

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号

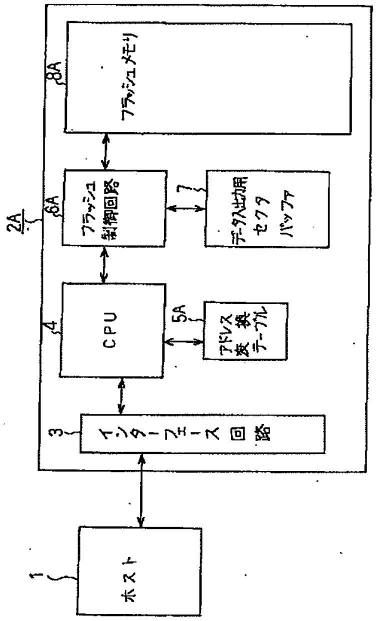
(72)発明者　コリン・メーソン
伊丹市瑞原４丁目１番地　三菱電機株式会社北伊丹製作所内

(74)代理人　弁理士　曾我　道照　（外６名）

(54)【発明の名称】　半導体ディスク装置

(57)【要約】
【構成】　当該消去ブロックの消去回数を格納する消去ブロック情報格納領域１０と、データを格納する複数のデータ格納領域１１と、論理セクタアドレスを格納する前記データ格納領域毎の論理セクタアドレス格納領域１２とから構成される消去ブロック９を複数有するフラッシュメモリ８Ａと、前記論理セクタアドレスを物理消去ブロック番号へ変換するためのアドレス変換テーブル５Ａと、前記アドレス変換テーブル５Ａに基づいて入力した論理セクタアドレス（ＬＳＡ）を物理消去ブロック番号（ＰＢＮ）へ変換し、前記フラッシュメモリ８Ａ上の該当物理消去ブロック内で前記入力した論理セクタアドレスに基づいて最新の該当データ格納領域１１を捜し出し、前記最新の該当データ格納領域１１の内容を読み出すＣＰＵ４とを備えた。

【効果】　データ管理用のアドレス変換テーブル５Ａを小さくできる。



【特許請求の範囲】

【請求項１】　当該消去ブロックの消去回数を格納する消去ブロック情報格納領域と、データを格納する複数のデータ格納領域と、論理セクタアドレスを格納する前記データ格納領域毎の論理セクタアドレス格納領域とから構成される消去ブロックを複数有するフラッシュメモリ、前記論理セクタアドレスを物理消去ブロック番号へ変換するためのアドレス変換テーブル、並びに前記物理消去ブロック番号に基づいて前記フラッシュメモリ上のデータを管理する制御手段を備えたことを特徴とする半導体ディスク装置。

【請求項２】　前記制御手段は、前記アドレス変換テーブルに基づいて入力した論理セクタアドレスを物理消去ブロック番号へ変換し、前記フラッシュメモリ上の該当物理消去ブロック内で前記入力した論理セクタアドレスに基づいて最新の該当データ格納領域を捜し出し、前記最新の該当データ格納領域の内容を読み出すことを特徴とする請求項１記載の半導体ディスク装置。

【請求項３】　前記制御手段は、該当消去ブロックの上から下へ連続してデータ格納領域にデータを書き込み、前記書き込んだデータ格納領域に対応する論理セクタアドレス格納領域に入力した論理セクタアドレスを書き込むとともに、前記アドレス変換テーブルの前記入力した論理セクタアドレスに対応する物理消去ブロック番号格納部に該当物理消去ブロック番号を書き込むことを特徴とする請求項１記載の半導体ディスク装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】この発明は、フラッシュメモリを記憶媒体として用いた半導体ディスクカード等の半導体ディスク装置に関するものである。

【０００２】

【従来の技術】今日、パーソナルコンピュータの分野において比較的大容量のデータを記憶させておく際には、ハードディスク装置などの磁気記憶媒体が用いられることが多い。これは、消費電力こそ大きいがコストパフォーマンスが非常によいためである。

【０００３】一方、フラッシュメモリ等の半導体メモリを上記ハードディスク装置のように動作させる半導体ディスク装置が出現した。この半導体ディスク装置は上記ハードディスク装置と違いモーターなどのメカニカルな部分が存在しないため、コストパフォーマンスは磁気記憶媒体に遅れをとるものの低消費電力、高信頼性という面を生かし携帯情報端末などに普及しつつある。

【０００４】なお、フラッシュメモリの特徴は以下のとおりである。第１に、電気的にデータの書き込み、消去が可能な不揮発性メモリである。第２に、データが既に書き込まれているメモリセルにデータを上書きすることはできない（このため、常に消去動作がつきまとう）。第３に、データの消去単位は、数Ｋ～数十ＫＢｙｔｅ単位である。第４に、書き込み、消去回数に制限がある。

【０００５】従来の半導体ディスク装置の構成について図１４、図１５、図１６及び図１７を参照しながら説明する。図１４は、従来の半導体ディスク装置の全体構成を示すブロック図である。図１５は、図１４のアドレス変換テーブルの内部構成を示す図である。図１６は、図１４のフラッシュメモリの内部構成を示す図である。また、図１７は、図１６の消去ブロックの内部構成を示す図である。

【０００６】図１４において、従来の半導体ディスク装置２は、インターフェイス回路３と、ＣＰＵ４と、アドレス変換テーブル５と、フラッシュ制御回路６と、データ入出力用セクタバッファ７と、フラッシュメモリ８とを備える。

【０００７】半導体ディスク装置２と接続するホスト１の代表的な例は、ノートパソコンや携帯情報端末である。リムーバブルタイプの半導体ディスク装置２は、現在の所、カード型が主流である。インターフェイス回路３は、ホスト１との情報をやりとりする。ＣＰＵ４は、データの入出力及びフラッシュメモリ８への命令を出力する。

【０００８】論理セクタ／物理セクタアドレス変換テーブル５は、論理セクタアドレスを物理セクタアドレスに変換するためのテーブルである。論理セクタアドレス（ＬＳＡ：Logical Sector Address）とはホスト１が半導体ディスク装置２に指定するセクタアドレスのことである。また、物理セクタアドレス（ＰＳＡ：Physical Sector Address）とは、半導体ディスク装置２内で使用されるフラッシュメモリ８のアドレスのことである。

【０００９】フラッシュ制御回路６は、複雑でないフラッシュメモリ８のデータ処理を行う。単純なデータの受け渡し等はフラッシュ制御回路６で行い、他の処理はＣＰＵ４で行う。データ入出力用セクタバッファ７は、データをフラッシュメモリ８からインターフェース回路３を通して出力、あるいはインターフェース回路３を通してフラッシュメモリ８にデータを入力する際に用いられる。

【００１０】図１５において、アドレス変換テーブル５は、論理セクタアドレス（ＬＳＡ）格納部と物理セクタアドレス（ＰＳＡ）格納部とから構成される。

【００１１】ＬＳＡ格納部には論理セクタアドレスが保存されている。内容は固定されている。なお、実際は、論理セクタアドレスがホスト１から送られてくると、論理セクタアドレスデータに基づき変換テーブル用の揮発性ＲＡＭのアドレスピンに電圧がかかりＰＳＡデータがでてくる構造になっている。説明しやすくするために、ＰＳＡ格納部との対応でＬＳＡ格納部があるとしている。ＰＳＡ格納部には任意の（図では１～ｎ）フラッシュメモリ８のセクタ番号が保存される。このアドレス変換テーブル５を用いることで、ホスト１が指定する論理

セクタアドレスに左右されることなく内部管理に都合のよい物理セクタアドレスにデータを保存することができる。このアドレス変換テーブル５は、頻繁に書き込み・消去されるのでＳＲＡＭやＤＲＡＭなどの揮発性のＲＡＭで構成するのが一般的である。

【００１２】このアドレス変換テーブル５の容量は、次のような条件のとき以下のようになる。２０メガバイト（MByte）のフラッシュメモリ８を使用し、データの入出力単位（セクタ）を５１２バイト（Byte）とすると、半導体ディスク装置２内のセクタ数は次のようになる。
半導体ディスク装置２内のセクタ数＝２０メガバイト÷５１２バイト＝４０９６０セクタ

【００１３】次に、「４０９６０」を２進数表現する際に必要なビット数は、ｌｎ４０９６０÷ｌｎ２＝１５．３となり、１６桁必要となる。

【００１４】これにより必要なアドレス変換テーブル５の容量は、４０９６０×１６＝６５５３６０ビットとなり、最終的に、８０キロバイト（KByte）必要となる。

【００１５】図１６において、フラッシュメモリ８は、複数の消去ブロック９と、予備の複数の消去ブロック９とから構成される。

【００１６】フラッシュメモリ８は電気的に書き込み消去可能な不揮発メモリである。不揮発であるためＤＲＡＭ・ＳＲＡＭのように電池によるバックアップの必要もなく、また電気的にデータの消去が可能なのでＥＰＲＯＭと違いボードから外すことなくデータを変更することができる。１セルで１ビットのデータを記憶することができるため、ＥＥＲＲＯＭより安価にメモリを作製することができる。以上の点がフラッシュメモリ８の長所にあたる。短所としては、消去回数に１万回～１０万回程度の上限があること、書き込みの際には必ず消去動作が必要なこと（このためデータがすでに書き込まれているセルに上書きすることは不可能）、消去単位は数Ｋ～数十KByteのブロック単位であること、等が上げられる。

【００１７】図１７において、１つの消去ブロック９は、先頭に消去ブロック情報格納領域１０と、複数のデータ格納領域１１と、データ格納領域１１毎のＬＳＡ格納領域１２とを有する。

【００１８】消去ブロック情報格納領域１０に現在のブロック消去回数を格納しておく。データ格納領域１１は、通常５１２バイト（＝１セクタ）の大きさである。ＬＳＡ格納領域１２はセクタごとに存在し、データを書き込む際にホスト１が指定したＬＳＡを格納しておく。これは、論理セクタ／物理セクタアドレス変換テーブル５を揮発性ＲＡＭで構成した際、電源オフと同時にデータが消えてしまうためである。電源をオンにしたときに全てのセクタのＬＳＡ格納領域１２を検索し揮発性ＲＡＭテーブル５を再構築する際に用いられる。

【００１９】つぎに、従来の半導体ディスク装置の動作について図１８、図１９及び図２０を参照しながら説明する。図１８は、従来の半導体ディスク装置の読み出し動作を説明するための図である。また、図１９及び図２０は、従来の半導体ディスク装置の書き込み動作を説明するための図である。

【００２０】フラッシュメモリ８を用いた半導体ディスク装置２はハードディスク装置とは異なり、データを上書きすることができない。従って、ホスト１から送られてくるデータの論理セクタアドレスとそのデータをフラッシュメモリ８のどの物理セクタアドレスに書き込むかを示すアドレス変換テーブル５を揮発性ＲＡＭ内に記憶させておくことが行われる。このアドレス変換テーブル５を用いることでＬＳＡに左右されることなくフラッシュメモリ８の記憶領域を有効に使用することが可能となる。

【００２１】まず、半導体ディスク装置２からのデータの読み出し動作を図１８で説明する。ホスト１は読み出したいデータのセクタアドレスを半導体ディスク装置２に送る。ホスト１から送られてくるアドレスデータには２種類ある。ＬＳＡ形式とＣＨＳ形式である。ＬＳＡ形式が１～ｎまでの通し番号でセクタを指定するのに対し、ＣＨＳ形式はハードディスク装置で使用されるシリンダ・ヘッド・セクタという３つのデータの組み合わせでデータ領域を指定する。半導体ディスク装置２内ではＬＳＡ／ＰＳＡアドレス変換テーブル５を用いるため、ホスト１からＣＨＳ形式のデータが入力された場合は、例えばインターフェース回路内でＬＳＡに変換し次の作業に移る。

【００２２】ＣＰＵ４は、アドレス変換テーブル５を用いてホスト１が指定したＬＳＡをＰＳＡにアドレス変換する。最後にＰＳＡに対応したフラッシュメモリ８内からデータが読み出される。

【００２３】例えば、ホスト１が指定したＬＳＡが「２」であった場合、アドレス変換テーブル５により「６」というＰＳＡに変換される。これにより、図１８に示すように、「Ａ」というデータが読み出されることになる。ＬＳＡ格納領域１２にはＬＳＡである「２」が格納されている。

【００２４】次に、半導体ディスク装置２へのデータの書き込み動作を図１９及び図２０で説明する。「Ａ」、「Ｂ」、「Ｃ」というデータがＰＳＡの「１」、「３」、「７」に格納されている状態を初期状態とする。データを書き込む際に注意しなければならないのはフラッシュメモリ８はデータの再書き込みができないという点である。上記初期状態の場合、ＰＳＡ「１」、「３」、「７」の領域が該当する。

【００２５】データが書き込まれていないＬＳＡをホスト１が指定してきた場合は、ＣＰＵ４は、フラッシュメモリ８内の適当な空き領域（ＰＳＡ「２」、「４」～「６」、「８」～「１２」）にデータを書き込み、アドレス変換テーブル５内のデータを更新する。図１９は、

ＬＳＡ「４」へ「Ｄ」というデータの書き込みをホストが指定した場合の例である。データ「Ｄ」とホスト１が指定したＬＳＡを空き領域ＰＳＡ「４」に書き込み、アドレス変換テーブル５のＬＳＡ「４」に対応したＰＳＡの部分にＰＳＡの値「４」を書き込む。

【００２６】ホスト１から、既にデータが書き込まれている領域への再書き込みが要求された場合（例えば、同名ファイルの上書き保存）であっても、再書き込みデータをフラッシュメモリ８の空き領域に書き込み、アドレス変換テーブル５を更新する。図２０は、ＬＳＡ「２」のデータを再書き込みした際の結果である。更新データ「Ｂ’」を空き領域ＰＳＡ「５」に書き込み、アドレス変換テーブル５のＬＳＡ「２」に対応するＰＳＡを「５」と更新する。なお、ＰＳＡ「３」が使用済みデータであることは、カード内のＣＰＵ４は認識しておかなければならない。

【００２７】

【発明が解決しようとする課題】上述したような従来の半導体ディスク装置では、アドレス変換テーブル５がセクタ（データ管理の最小単位）ごとに１つのＰＳＡを格納するメモリ領域が必要となるため、フラッシュメモリ８が大容量になるにつれアドレス変換テーブル５も大容量となるという問題点があった。

【００２８】この発明は、前述した問題点を解決するためになされたもので、従来装置のパフォーマンス、つまり読み出し速度と略同程度の読み出し速度を維持し、かつメモリ管理用のアドレス変換テーブルの容量を小さくできる半導体ディスク装置を得ることを目的とする。

【００２９】

【課題を解決するための手段】この発明に係る半導体ディスク装置は、当該消去ブロックの消去回数を格納する消去ブロック情報格納領域と、データを格納する複数のデータ格納領域と、論理セクタアドレスを格納する前記データ格納領域毎の論理セクタアドレス格納領域とから構成される消去ブロックを複数有するフラッシュメモリと、前記論理セクタアドレスを物理消去ブロック番号へ変換するためのアドレス変換テーブルと、前記物理消去ブロック番号に基づいて前記フラッシュメモリ上のデータを管理する制御手段とを備えたものである。

【００３０】また、この発明に係る半導体ディスク装置は、前記制御手段が、前記アドレス変換テーブルに基づいて入力した論理セクタアドレスを物理消去ブロック番号へ変換し、前記フラッシュメモリ上の該当物理消去ブロック内で前記入力した論理セクタアドレスに基づいて最新の該当データ格納領域を捜し出し、前記最新の該当データ格納領域の内容を読み出すものである。

【００３１】さらに、この発明に係る半導体ディスク装置は、前記制御手段が、該当消去ブロックの上から下へ連続してデータ格納領域にデータを書き込み、前記書き込んだデータ格納領域に対応する論理セクタアドレス格納領域に入力した論理セクタアドレスを書き込むとともに、前記アドレス変換テーブルの前記入力した論理セクタアドレスに対応する物理消去ブロック番号格納部に該当物理消去ブロック番号を書き込むものである。

【００３２】

【作用】この発明に係る半導体ディスク装置においては、当該消去ブロックの消去回数を格納する消去ブロック情報格納領域と、データを格納する複数のデータ格納領域と、論理セクタアドレスを格納する前記データ格納領域毎の論理セクタアドレス格納領域とから構成される消去ブロックを複数有するフラッシュメモリと、前記論理セクタアドレスを物理消去ブロック番号へ変換するためのアドレス変換テーブルと、前記物理消去ブロック番号に基づいて前記フラッシュメモリ上のデータを管理する制御手段とを備えたので、データ管理用のアドレス変換テーブルを小さくできる。

【００３３】また、この発明に係る半導体ディスク装置においては、前記制御手段が、前記アドレス変換テーブルに基づいて入力した論理セクタアドレスを物理消去ブロック番号へ変換し、前記フラッシュメモリ上の該当物理消去ブロック内で前記入力した論理セクタアドレスに基づいて最新の該当データ格納領域を捜し出し、前記最新の該当データ格納領域の内容を読み出すので、データ管理用のアドレス変換テーブルを小さくできる。

【００３４】さらに、この発明に係る半導体ディスク装置においては、前記制御手段が、該当消去ブロックの上から下へ連続してデータ格納領域にデータを書き込み、前記書き込んだデータ格納領域に対応する論理セクタアドレス格納領域に入力した論理セクタアドレスを書き込むとともに、前記アドレス変換テーブルの前記入力した論理セクタアドレスに対応する物理消去ブロック番号格納部に該当物理消去ブロック番号を書き込むので、データ管理用のアドレス変換テーブルを小さくできる。

【００３５】

【実施例】

実施例１．以下、この発明の実施例１の構成について図１、図２、図３及び図４を参照しながら説明する。図１は、この発明の実施例１の全体構成を示すブロック図である。図２は、図１のアドレス変換テーブルの内部構成を示す図である。図３は、図１のフラッシュメモリの内部構成を示す図である。図４は、図３の消去ブロックの内部構成を示す図である。なお、各図中、同一符号は同一又は相当部分を示す。

【００３６】図１において、この実施例１に係る半導体ディスク装置２Ａは、インターフェイス回路３と、ＣＰＵ４と、アドレス変換テーブル５Ａと、フラッシュ制御回路６Ａと、データ入出力用セクタバッファ７と、フラッシュメモリ８Ａとを備える。

【００３７】従来の半導体ディスク装置２との違いは、アドレス変換テーブル５Ａの容量が小さい点であり、そ

の分、フラッシュメモリ８Ａを大きくできる。また、フラッシュ制御回路６Ａは、従来のフラッシュ制御回路６の機能に加えて、ハード的な論理セクタアドレス比較回路６１を含む。さらに、ＣＰＵ４は、読み出し用と書き込み用のアドレスポインタを用いてフラッシュメモリ８Ａ上のデータ管理を行う。すなわち、半導体ディスク装置２Ａは、上記論理セクタアドレス比較回路６１と、アドレスポインタとを使用することで、従来と略同程度のパフォーマンスを維持しながら、アドレス変換テーブル５Ａの容量を小さくできる。なお、この発明に係る制御手段は、この実施例１では読み出し用と書き込み用のアドレスポインタを用いるＣＰＵ４と、論理セクタアドレス比較回路６１を含むフラッシュ制御回路６Ａと、データ入出力用セクタバッファ７とから構成される。

【００３８】アドレス変換テーブル５Ａは、論理セクタアドレスを物理消去ブロック番号に変換するためのテーブルである。物理消去ブロック番号（ＰＢＮ：Physical Block Number）とは、半導体ディスク装置２Ａ内で使用されるフラッシュメモリ８Ａのブロックアドレスのことである。

【００３９】図２において、アドレス変換テーブル５Ａは、論理セクタアドレス（ＬＳＡ）格納部と物理消去ブロック番号（ＰＢＮ）格納部とから構成される。

【００４０】ＬＳＡ格納部には論理セクタアドレスが保存されている。内容は固定されている。なお、実際は、論理セクタアドレスがホスト１から送られてくると、論理セクタアドレスデータに基づき変換テーブル用の揮発性ＲＡＭのアドレスピンに電圧がかかりＰＢＮデータがでてくる構造になっている。説明をしやすくするために、ＰＢＮ格納部との対応でＬＳＡ格納部があるとしている。ＰＢＮ格納部には任意のフラッシュメモリ８Ａの物理消去ブロック番号が保存される。このアドレス変換テーブル５Ａを用いることで、ホスト１が指定する論理セクタアドレスに左右されることなく内部管理に都合のよい物理消去ブロックにデータを保存することができる。このアドレス変換テーブル５Ａは、頻繁に書き込み・消去されるのでＳＲＡＭやＤＲＡＭなどの揮発性のＲＡＭで構成するのが一般的である。

【００４１】ただ、このアドレス変換テーブル５Ａでは、従来と違いＬＳＡのデータからＰＢＮしか解らない。例えば、消去ブロックのサイズが６４キロバイトのフラッシュメモリを用いた場合、内部にはおよそ１００前後のセクタ（＝５１２バイト）が存在する（消去ブロック情報格納領域１０、ＬＳＡ格納領域１２等がなければ最大１２８セクタが存在する）。このため目的のデータを検索するための工夫が必要となる。これについては後述する。

【００４２】このアドレス変換テーブル５Ａの容量は、以下のようになる。２０メガバイトのフラッシュメモリ８Ａを使用し、１セクタを５１２バイトとすると、半導体ディスク装置２Ａ内のセクタ数は従来と同様に４０９６０となる。しかしながら、アドレス変換テーブル５ＡのＰＢＮ格納部にはブロック番号を記憶させるので、半導体ディスク装置２Ａ内のブロック数は、１ブロック（消去ブロック）を６４キロバイトとすると以下のようになる。半導体ディスク装置２Ａ内のブロック数＝２０メガバイト÷６４キロバイト＝３２０ブロック

【００４３】次に、「３２０」を２進数表現する際に必要なビット数は、ｌｎ３２０÷ｌｎ２＝８．３となり、９桁必要となる。これにより必要なアドレス変換テーブル５Ａの容量は、４０９６０×９＝３６８６４０ビットとなり、最終的に、４５キロバイト必要となる。これは、従来の約１／２である。

【００４４】図３において、フラッシュメモリ８Ａは、複数の消去ブロック９Ａと、予備の複数の消去ブロック９Ａとから構成される。なお、データ（メモリ）管理の対象ブロックのサイズが消去単位と同じであるので、消去ブロックと称する。メインメモリに使用するフラッシュメモリ８Ａは、従来と同様ブロック消去型（消去ブロック単位は数Ｋ～数十Ｋバイト）のフラッシュメモリを用いる。

【００４５】図４において、１つの消去ブロック９Ａは、先頭に消去ブロック情報格納領域１０と、複数のデータ格納領域１１と、データ格納領域１１毎のＬＳＡ（論理セクタアドレス）格納領域１２と、データ格納領域１１毎の有効データ確認フラグ１３とを有する。有効データ確認フラグ１３以外は、従来と同様である。この有効データ確認フラグ１３は、ＣＰＵ４が有効データか、いつでも消去可能な無効データかを区別するためのものである。有効データは「ＦＦ」（１１１１１１１１）、無効データは「００」（００００００００）で表す。なお、逆の表現でもよい。ＣＰＵ４は、データの上書きや消去を行った場合、必要なくなったデータ格納領域１１に対応する有効データ確認フラグ１３を「ＦＦ」から「００」へ書き換える。

【００４６】つぎに、この実施例１の動作について図５から図１３までを参照しながら説明する。図５～図８は、この実施例１の書き込み動作を説明するための図である。図９及び図１０は、この実施例１の読み出し動作を説明するための図である。図１１及び図１２は、この実施例１の読み出し動作を示すフローチャートである。図１３は、この実施例１の書き込み動作を示すフローチャートである。

【００４７】まず、半導体ディスク装置２Ａへのデータ書き込み動作を図５～図８で説明する。従来との違いは、１つの消去ブロック内のセクタ（データ格納領域１１）が全て書き込み済みになるまで他の消去ブロックにはデータを書き込まないことである。また、データは、消去ブロックの上から下に連続して書き込む（ランダムに書き込まない）。これは、「書き込み用アドレスポイ

ンタ」を用い、書き込みの度に１つづつ更新（インクリメント）していく。ここで、「アドレスポインタ」とは、任意の物理セクタアドレス（ＰＳＡ）を記憶しておくためのものである。この実施例１では、データの読み出し時と、書き込み時にそれぞれ１つの「読み出し用アドレスポインタ」、「書き込み用アドレスポインタ」を使用する。

【００４８】半導体ディスク装置２Ａ内のフラッシュメモリ８Ａ内にデータが全く無い状態から動作の説明を進める。このとき、書き込み用アドレスポインタは「１」を指し示している。データ書き込みの際には、まずホスト１から書き込むべきデータと書き込みアドレスが送られてくる。アドレスは、ＣＨＳ形式の場合が考えられるので従来と同様に全てＬＳＡ形式に変換する。アドレス形式情報もホスト１から送られてくるため半導体ディスク装置２Ａ側で容易にＣＨＳ形式かＬＳＡ形式かを区別することができる。

【００４９】ＬＳＡがランダムに送られてきても、図５に示すように、書き込みはＰＢＮ「１」から行う。例えば、ホスト１からＬＳＡ「３」にデータ「Ａ」の書き込み要求があった場合、ＣＰＵ４は書き込み用アドレスポインタに従い、ＰＢＮ「１」の先頭のデータ格納領域１１にデータ「Ａ」を書き込み、対応する論理セクタアドレス格納領域１２にＬＳＡ「３」を書き込む。そして、書き込みアドレスポインタを更新する。つまり、書き込みアドレスポインタを「２」とする。さらに、アドレス変換テーブル５ＡのＬＳＡ「３」に対応するＰＢＮ格納部に「１」を書き込む。なお、図５～図９において、フラッシュメモリ８Ａの消去ブロック９Ａは、説明しやすいようにデータ格納領域１１を３つとしている。

【００５０】図６は、クリーンな消去ブロックがあと１ブロックになった状態である。このとき、書き込み用アドレスポインタは、「１」→「２」→「３」→「…」→「８」→「９」と更新されて、ＰＢＮ「４」の先頭のデータ格納領域１１である「１０」を指し示している。ここで、これ以上書き込むと、クリーンな消去ブロックを確保することができないためフラッシュメモリ８Ａの消去ブロックのクリーニングを行う。

【００５１】ＣＰＵ４は、全ての消去ブロック内の状態を確認し消去に最適なブロックを決定する。最適なブロックとは、有効データ確認フラグ１３に基づく、有効なデータがあまり存在しない消去ブロックや、消去ブロック情報格納領域１０の内容に基づく、消去回数が少ない消去ブロックなどである。図６では、各消去ブロックの消去回数は「０」で同じであり、ＰＢＮ「１」が有効なデータが少ないため、つまりＰＢＮ「１」のデータ「Ａ’」はデータ「Ａ」の上書きを意味しているため、このブロックを消去し、クリーンなセクタを確保することにする。

【００５２】まず、図７に示すように、有効なデータをクリーンなＰＢＮ「４」に退避し、次にアドレス変換テーブル５Ａを更新する。その後、図８に示すように、ＰＢＮ「１」をブロック消去し、ブロック消去回数を「１」だけ増やす。この作業により、クリーンなブロック１つと、クリーンなセクタ１つが確保されたことになる。このとき、書き込み用アドレスポインタは「１２」を指し示している。

【００５３】次に、半導体ディスク装置２Ａからのデータ読み出し動作を図９及び図１０で説明する。まず、ホスト１から送られてきたアドレスをＬＳＡ形式に統一する。次に、アドレス変換テーブル５Ａを用いてＬＳＡからＰＢＮを割り出す。

【００５４】次に、得られたＰＢＮ内の論理セクタアドレス格納領域１２に格納されたＬＳＡを下から順に確認していく。この際、図１０に示すように、フラッシュ制御回路６Ａ内の論理セクタアドレス比較回路６１で、ホスト１から送られてきたＬＳＡと、該当ＰＢＮ内の論理セクタアドレス格納領域１２に格納されたＬＳＡとを比較する。一致したときのデータ格納領域１１（セクタ）の内容が読み出すべきデータである。このとき、読み出し用アドレスポインタに一致したときのＰＳＡにプラス１した値をセットする。読み出しデータが複数のセクタの場合、次の読み出し時は読み出し用アドレスポインタが指し示すセクタアドレスからデータを読み出す。

【００５５】例えば、ホスト１からＬＳＡが「２」というデータが送られてきた場合、図９に示すように、アドレス変換テーブル５Ａにより、ＰＢＮ「３」を得る。次に、ＰＢＮ「３」の消去ブロック内の論理セクタアドレスを下から順に確認する。一番下のＬＳＡは「６」であるから該当せず。次に、次のＬＳＡが「２」であるから一致する。従って、読み出すべきデータは「Ｈ」ということになる。このとき、読み出し用アドレスポインタは「９」である。

【００５６】従来と違い、消去ブロックが大きくなると検索に要する時間がかかることになる。しかし、一般にデータは５１２バイト（１セクタ）以上のサイズのものが多いため、初めてのデータ検索には、１つの消去ブロック内に約１００セクタのデータエリアがあるため、最高１００ステップ程度の検索を必要とするが、２回目以降は前回読み出したセクタの次のセクタにデータがある可能性が極めて高いため、前回読み出したセクタの次のセクタのアドレスを読み出し用アドレスポインタに記憶しておくことで検索回数を大幅に減少させることができる。

【００５７】つづいて、半導体ディスク装置２Ａからのデータ読み出し動作を図１１及び図１２のフローチャートで説明する。１～数十のセクタからなるファイル（データ）を扱う場合、ファイルの先頭のセクタを読み出す際は読み出し用アドレスポインタが決定していないので、図１１の検索処理を行う。また、上記ファイルの次

のセクタのデータを読み出すときには読み出し用アドレスポインタが設定されているので、図１２の処理を行う。図１１の検索処理との違いは、図１２の処理はホスト１から読み出すべきデータのアドレス（ＬＳＡ）を受け取るが、ホスト１からのＬＳＡを使用することなくデータを読み出すことである。図１１の検索処理は、ファイルの先頭セクタのデータを読み出す場合の他に、読み出し用アドレスポインタが消去ブロック内の最終セクタアドレスまできた場合にも行う。これは、次に読み出すセクタが存在する消去ブロックが解らなくなるためである。ＣＰＵ４は、ファイル名等に基づきファイルの先頭とそれ以外を認識し、また、ブロックサイズ等に基づき最終セクタアドレスを認識する。

【００５８】初めに、ホスト１から読み出すべきデータのセクタ情報を受け取る（ステップ２０）。これはＬＳＡの形式かもしくはＣＨＳ形式で送られてくる。ＬＳＡ形式に統一するためにＣＨＳデータ形式で送られてきた場合はＬＳＡ形式に変換する（ステップ２１～２２）。この変換は、半導体ディスク装置２Ａ内のＣＰＵ４を用いてもかまわないし、専用の回路を半導体ディスク装置内部に持たせてもかまわない。

【００５９】次に、ＬＳＡをＰＢＮに変換する（ステップ２３）。これは、アドレス変換テーブル５Ａを用いる。次に、決定したＰＢＮ内の論理セクタアドレス格納領域１２内に格納されたＬＳＡを読み出す。この読み出したＬＳＡと、ホスト１からのＬＳＡとを論理セクタアドレス比較回路６１で比較し、不一致ならば次の論理セクタアドレス格納領域１２内に格納されたＬＳＡを読み出し、上記ＰＢＮ内を下から検索し、同様の比較を一致するまで行う。なお、検索の方向は、場合によっては上から行ってもよい（ステップ２４～２６）。

【００６０】ＬＳＡ同士が一致したら、対応するデータ格納領域１１からデータを読み出す（ステップ２７）。そして、読み出し用アドレスポインタに、上記のデータを読み出した次のセクタアドレスをセットする（ステップ２８）。

【００６１】つづいて、次のセクタアドレスのデータを読み出す場合は以下のとおりである。ステップ４０～４２は、上記ステップ２０～２２と同様である。これは、インターフェースの互換性を保つために、ホスト１からのＬＳＡを受け取り、処理した形にしなければならないからである。

【００６２】次に、読み出し用アドレスポインタが指し示すセクタアドレスからデータを読み出す（ステップ４３）。そして、読み出し用アドレスポインタに「１」を加算して更新する。ある１つのファイルを読み出す場合、従来の半導体ディスク装置ではファイルを構成しているセクタ数だけアドレス変換テーブル５を参照する。しかし、この実施例１では、初回の消去ブロック内のデータ検索に時間をとるものの、それ以降のセクタに関しては読み出し用アドレスポインタに従ってデータを読み出すため、２つ目以降のセクタに関しては上記従来装置よりも高速に処理できる。

【００６３】また、読み出し速度を犠牲にし、データ読み出しの信頼度を上げる場合には、ステップ４３の次で、ホスト１から受け取ったＬＳＡと、読み出し用アドレスポインタが指し示すセクタのＬＳＡ格納領域１２内のＬＳＡとを比較して確認する。なお、ここで不一致のときはステップ２３以降の検索処理を行う。

【００６４】つづいて、半導体ディスク装置２Ａへのデータ書き込み動作を図１３のフローチャートで説明する。図１３のステップ３０～３２は、図１１のステップ２０～２２と同様であるので説明を省略する。まず、書き込み用アドレスポインタの指示に基づき空きセクタを確認する（ステップ３３）。

【００６５】空きセクタがある場合は、そこにデータを書き込むとともに、アドレス変換テーブル５Ａを更新する（ステップ３４～３５）。つまり、データを書き込んだＰＢＮを該当するＬＳＡのＰＢＮ格納部に保存する。

【００６６】ステップ３４において、書き込み用アドレスポインタに基づき空きセクタがない場合には以下のように処理する（ステップ３６～３８）。まず、消去回数や無効データ数等に基づき消去するブロックを決定する。次に、有効データを空きブロックにコピーする。その後、決定したブロックを消去し、消去したブロックの消去回数を更新する。そして、ステップ３５に進む。

【００６７】データ更新の場合は、旧データの有効データ確認フラグ１３を「ＦＦ」から「００」へ更新する。また、書き込み用アドレスポインタを更新する（ステップ３９）。この実施例１では連続してデータを書き込むため、書き込み用アドレスポインタはデータ書き込み終了後、次のセクタアドレスを指し示すことになる。なお、１つの消去ブロック内のセクタ全てがデータで埋まったときは、次に書き込むべき消去ブロックをＣＰＵ４が決定し指し示すセクタアドレスを決める。例えば、複数の空きの（クリーンな）消去ブロックが存在する場合、ＣＰＵ４は、消去回数等に基づき次に書き込むべき空きの消去ブロックを決定する。

【００６８】この実施例１は、従来の半導体ディスクカード（装置）の読み出し速度等のパフオーマンスを落とすことなく、読み出し用及び書き込み用アドレスポインタと論理セクタアドレス比較回路６１を使用することにより、アドレス変換テーブル５Ａの容量を小さくすることができる。アドレス変換テーブル５Ａの容量を小さくすることで、無理なく半導体ディスクカードの大容量化を進めることができる。従来のアドレス変換テーブル５を用いると２０ＭＢの半導体ディスクで８０ＫＢのサイズが、４０ＭＢの半導体ディスクで１６０ＫＢ（１.２５Ｍｂｉｔ）のサイズが必要となる。これがおよそ１／２のサイズに小さくなれば揮発性ＲＡＭにかかっていた

コストを削減することができ、またアドレス変換テーブル用揮発性ＲＡＭメモリが搭載されていたスペースにフラッシュメモリを増設して搭載できるため半導体ディスク装置の容量を増大させることができる。

【００６９】

【発明の効果】この発明に係る半導体ディスク装置は、以上説明したとおり、当該消去ブロックの消去回数を格納する消去ブロック情報格納領域と、データを格納する複数のデータ格納領域と、論理セクタアドレスを格納する前記データ格納領域毎の論理セクタアドレス格納領域とから構成される消去ブロックを複数有するフラッシュメモリと、前記論理セクタアドレスを物理消去ブロック番号へ変換するためのアドレス変換テーブルと、前記物理消去ブロック番号に基づいて前記フラッシュメモリ上のデータを管理する制御手段とを備えたので、データ管理用のアドレス変換テーブルを小さくできるという効果を奏する。

【００７０】また、この発明に係る半導体ディスク装置は、以上説明したとおり、前記制御手段が、前記アドレス変換テーブルに基づいて入力した論理セクタアドレスを物理消去ブロック番号へ変換し、前記フラッシュメモリ上の該当物理消去ブロック内で前記入力した論理セクタアドレスに基づいて最新の該当データ格納領域を捜し出し、前記最新の該当データ格納領域の内容を読み出すので、データ管理用のアドレス変換テーブルを小さくできるという効果を奏する。

【００７１】さらに、この発明に係る半導体ディスク装置は、以上説明したとおり、前記制御手段が、該当消去ブロックの上から下へ連続してデータ格納領域にデータを書き込み、前記書き込んだデータ格納領域に対応する論理セクタアドレス格納領域に入力した論理セクタアドレスを書き込むとともに、前記アドレス変換テーブルの前記入力した論理セクタアドレスに対応する物理消去ブロック番号格納部に該当物理消去ブロック番号を書き込むので、データ管理用のアドレス変換テーブルを小さくできるという効果を奏する。

【図面の簡単な説明】

【図１】　この発明の実施例１の全体構成を示すブロック図である。

【図２】　この発明の実施例１のアドレス変換テーブルの構成を示す図である。

【図３】　この発明の実施例１のフラッシュメモリの構成を示す図である。

【図４】　この発明の実施例１の消去ブロックの内部構成を示す図である。

【図５】　この発明の実施例１のデータ書き込み動作を説明するための図である。

【図６】　この発明の実施例１のデータ書き込み動作を説明するための図である。

【図７】　この発明の実施例１のデータ書き込み動作を説明するための図である。

【図８】　この発明の実施例１のデータ書き込み動作を説明するための図である。

【図９】　この発明の実施例１のデータ読み出し動作を説明するための図である。

【図１０】　この発明の実施例１のデータ読み出し動作を説明するための図である。

【図１１】　この発明の実施例１のデータ読み出し動作を示すフローチャートである。

【図１２】　この発明の実施例１のデータ読み出し動作を示すフローチャートである。

【図１３】　この発明の実施例１のデータ書き込み動作を示すフローチャートである。

【図１４】　従来の半導体ディスク装置の全体構成を示すブロック図である。

【図１５】　従来の半導体ディスク装置のアドレス変換テーブルの構成を示す図である。

【図１６】　従来の半導体ディスク装置のフラッシュメモリの構成を示す図である。

【図１７】　従来の半導体ディスク装置のフラッシュメモリ内の消去ブロックの構成を示す図である。

【図１８】　従来の半導体ディスク装置のデータ読み出し動作を説明するための図である。

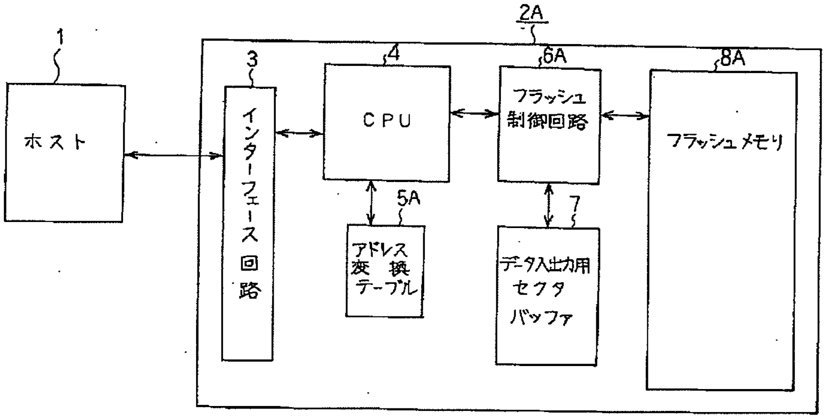
【図１９】　従来の半導体ディスク装置のデータ書き込み動作を説明するための図である。

【図２０】　従来の半導体ディスク装置のデータ書き込み動作を説明するための図である。

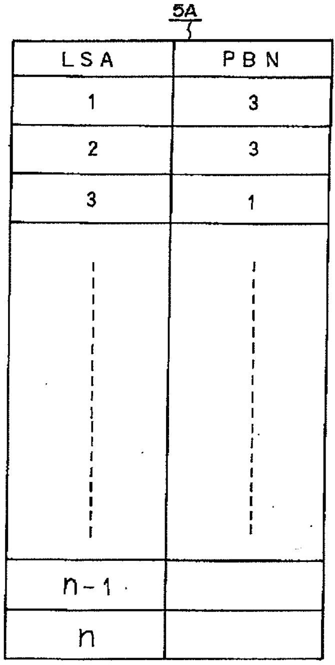
【符号の説明】

１　ホスト、２Ａ　半導体ディスク装置、３　インターフェース回路、４　ＣＰＵ、５Ａ　アドレス変換テーブル、６Ａ　フラッシュ制御回路、６１　論理セクタアドレス比較回路、７　データ入出力用セクタバッファ、８Ａ　フラッシュメモリ。

【図１】

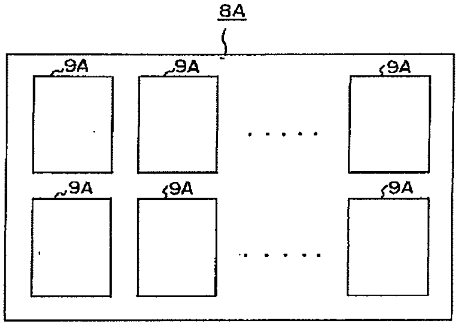

1
ホスト
2A
3
インターフェース回路
4
CPU
5A
アドレス変換テーブル
6A
フラッシュ制御回路
7
データ入出力用セクタバッファ
8A
フラッシュメモリ

【図２】

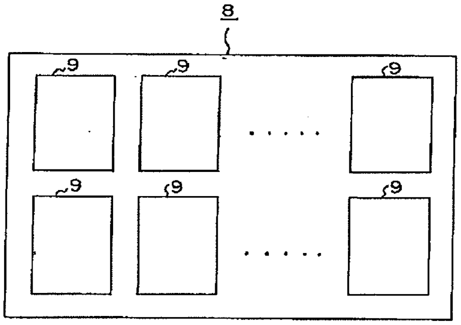


LSA	PBN
1	3
2	3
3	1
¦	¦
n-1	
n	

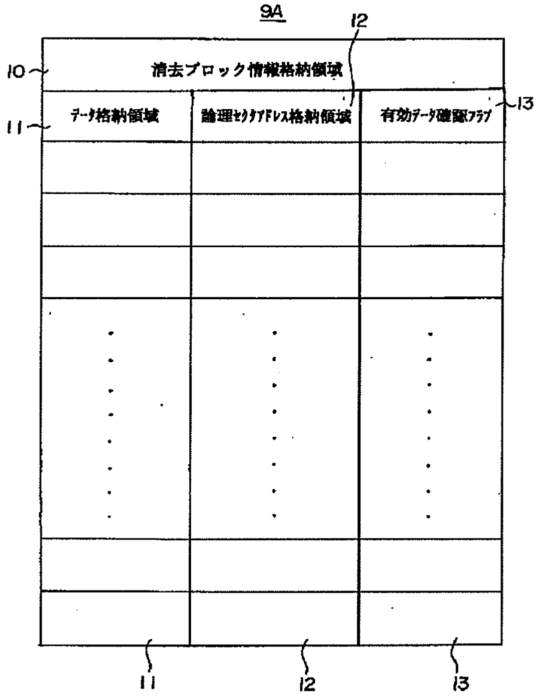
【図３】


8A
9A
9A
9A
9A
9A
9A
.....
.....

【図１６】



【図４】

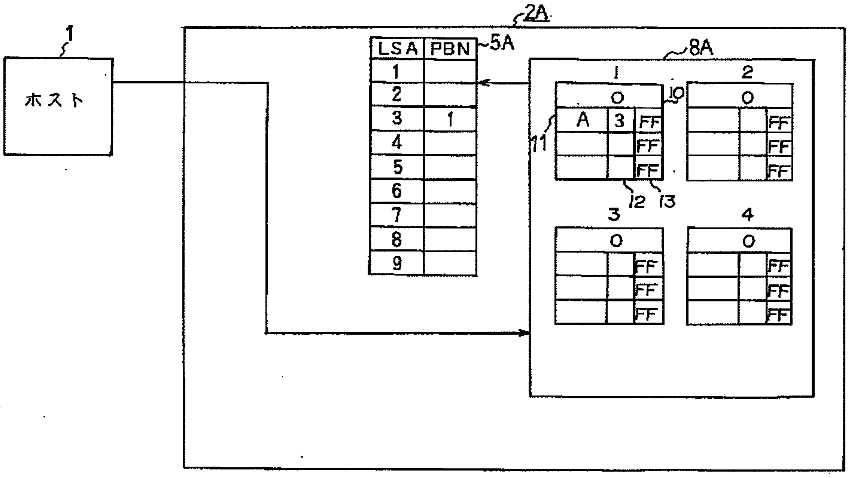

9A
10
11
12
13
消去ブロック情報格納領域
データ格納領域
論理セクタアドレス格納領域
有効データ確認フラグ
11
12
13

【図１５】

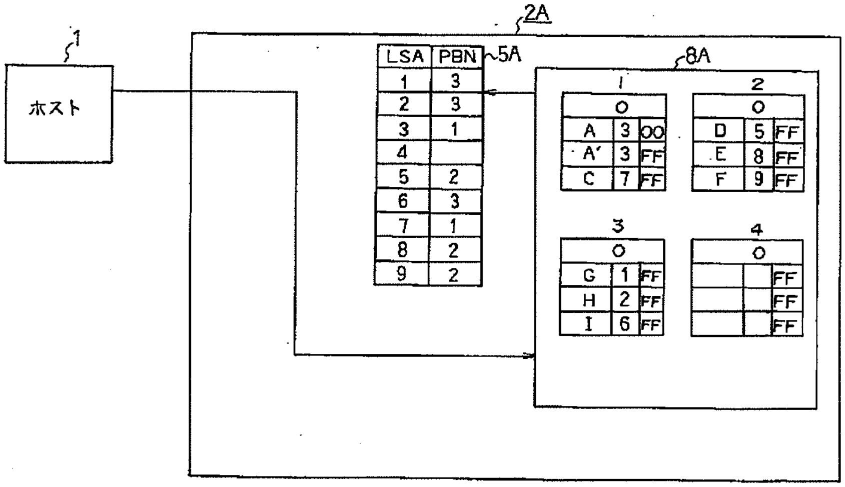
5

LSA	PSA
1	6
2	1
3	3
4	n
5	5
⋮	⋮
n−1	
n	

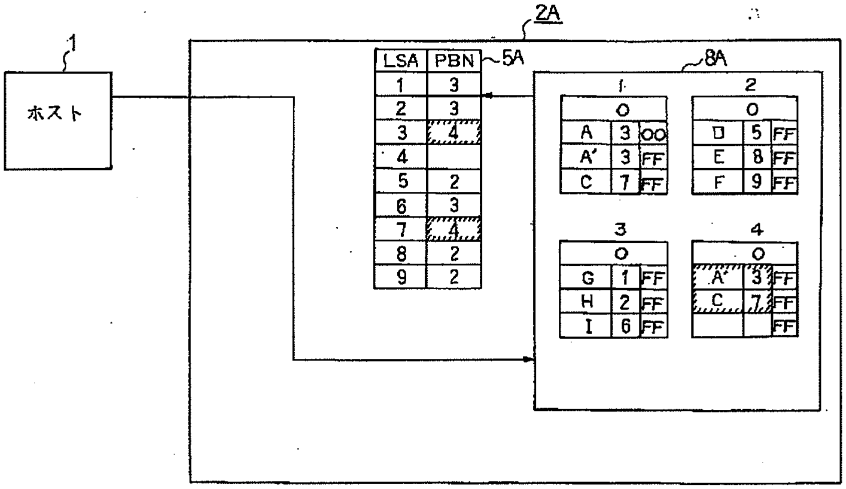
【図５】


1
ホスト
2A
5A
LSA
PBN
1
2
3
1
4
5
6
7
8
9
8A
1
2
0
0
10
11
A
3
FF
FF
FF
FF
FF
FF
12
13
3
4
0
0
FF
FF
FF
FF
FF
FF

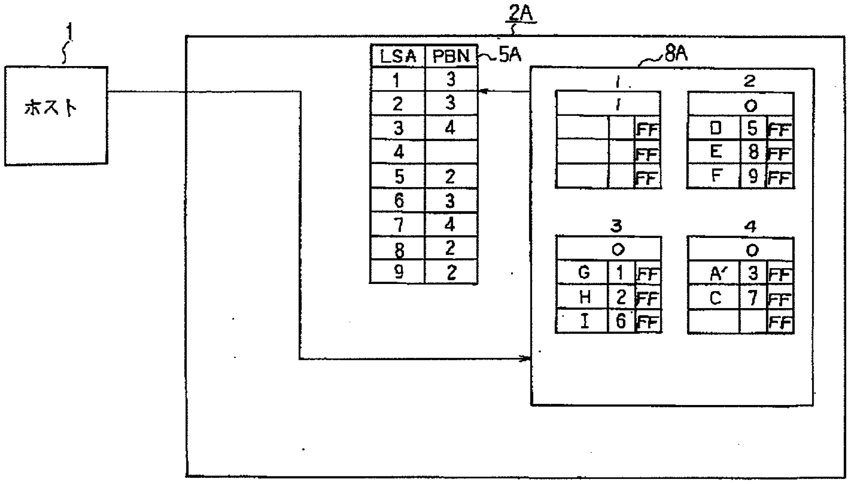
【図６】


2A
1
ホスト
5A
LSA PBN
1 3
2 3
3 1
4
5 2
6 3
7 1
8 2
9 2
8A
1
0
A 3 00
A' 3 FF
C 7 FF
2
0
D 5 FF
E 8 FF
F 9 FF
3
0
G 1 FF
H 2 FF
I 6 FF
4
0
FF
FF
FF

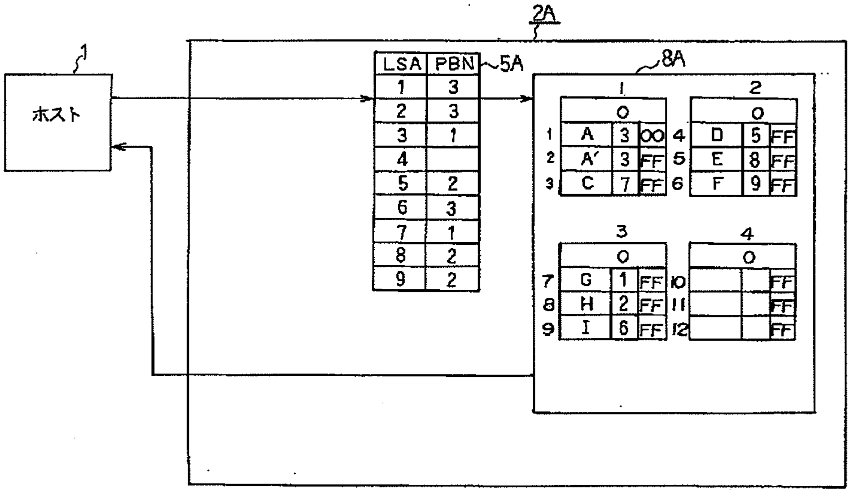
【図７】


2A
1
ホスト
5A
LSA PBN
1 3
2 3
3 4
4
5 2
6 3
7 4
8 2
9 2
8A
1
0
A 3 00
A' 3 FF
C 7 FF
2
0
D 5 FF
E 8 FF
F 9 FF
3
0
G 1 FF
H 2 FF
I 6 FF
4
0
A' 3 FF
C 7 FF
FF

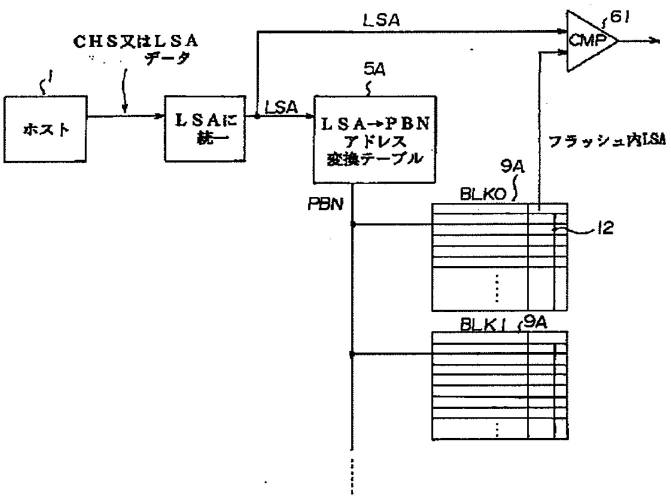
【図８】


2A
1
ホスト
LSA PBN
5A
1 3
2 3
3 4
4
5 2
6 3
7 4
8 2
9 2
8A
1
1
FF
FF
FF
2
0
D 5 FF
E 8 FF
F 9 FF
3
0
G 1 FF
H 2 FF
I 6 FF
4
0
A′ 3 FF
C 7 FF
FF

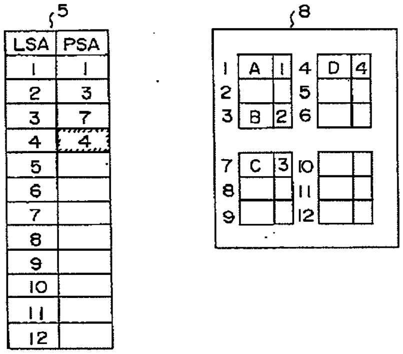
【図９】


2A
1
ホスト
LSA PBN
5A
1 3
2 3
3 1
4
5 2
6 3
7 1
8 2
9 2
8A
1
0
1 A 3 00
2 A′ 3 FF
3 C 7 FF
2
0
4 D 5 FF
5 E 8 FF
6 F 9 FF
3
0
7 G 1 FF
8 H 2 FF
9 I 6 FF
4
0
10 FF
11 FF
12 FF

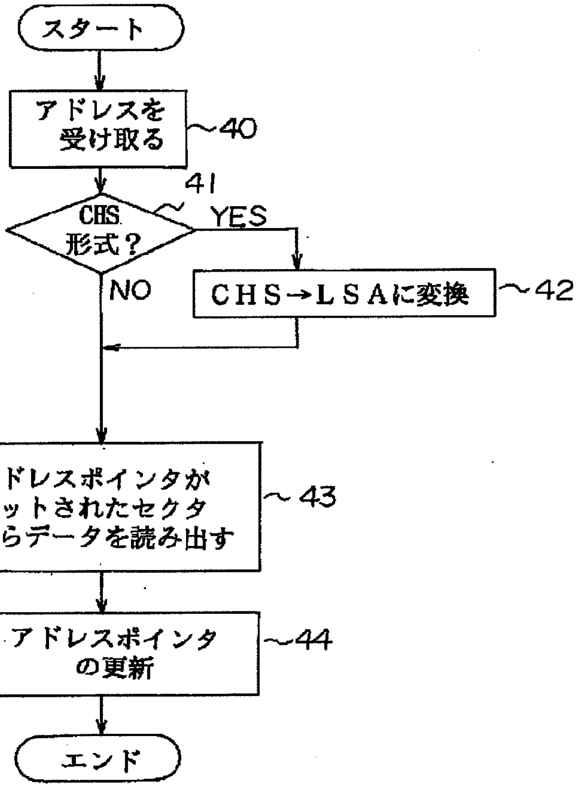
【図１０】


CHS又はLSA
データ
1
ホスト
LSAに
統一
LSA
LSA
5A
LSA→PBN
アドレス
変換テーブル
61
CMP
フラッシュ内LSA
PBN
9A
BLK0
12
BLK1
9A

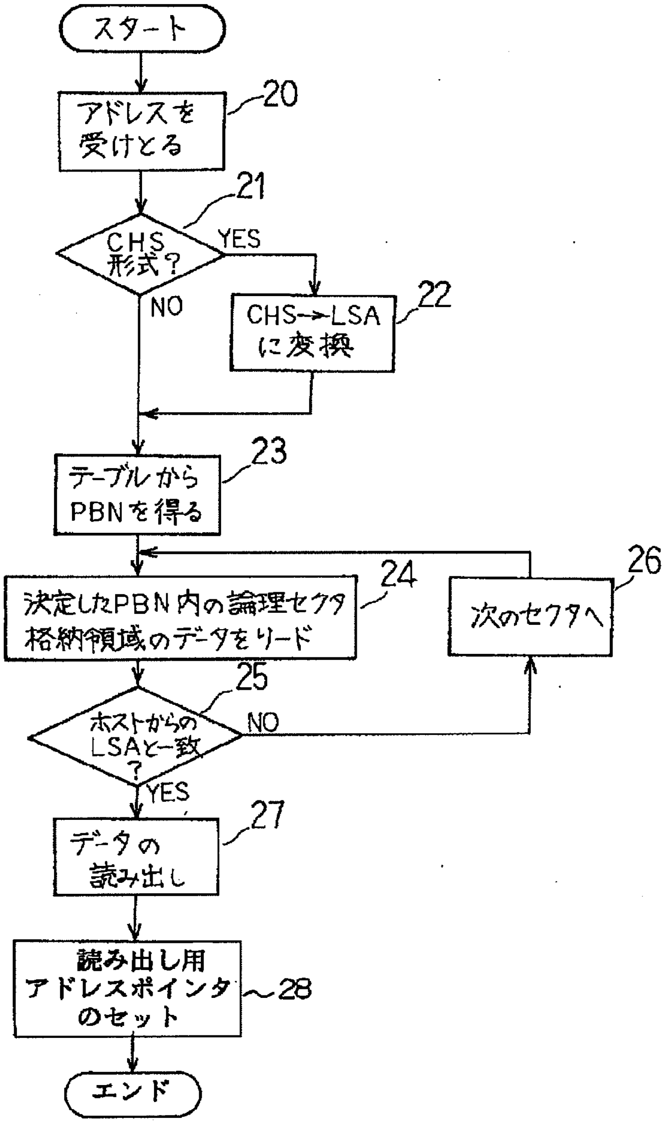
【図１９】


LSA
PSA
A
D
B
C

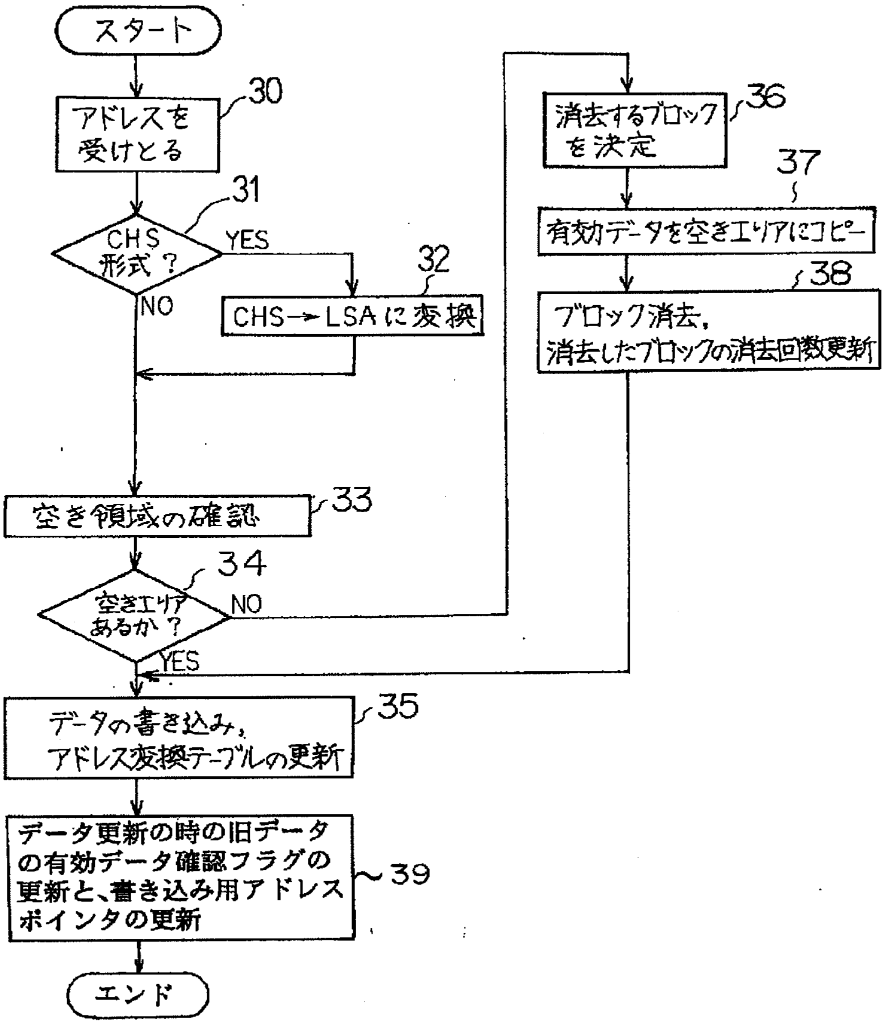
【図１２】


スタート
アドレスを
受け取る
40
41
CHS
形式？
YES
NO
CHS→LSAに変換
42
アドレスポインタが
セットされたセクタ
からデータを読み出す
43
アドレスポインタ
の更新
44
エンド

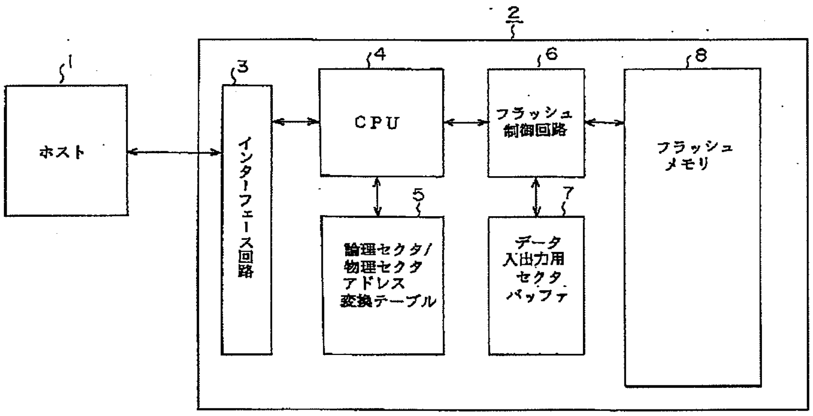
【図１１】


スタート
アドレスを受けとる
20
CHS形式？
21
YES
NO
CHS→LSAに変換
22
テーブルからPBNを得る
23
決定したPBN内の論理セクタ格納領域のデータをリード
24
次のセクタへ
26
ホストからのLSAと一致？
25
NO
YES
データの読み出し
27
読み出し用アドレスポインタのセット
28
エンド

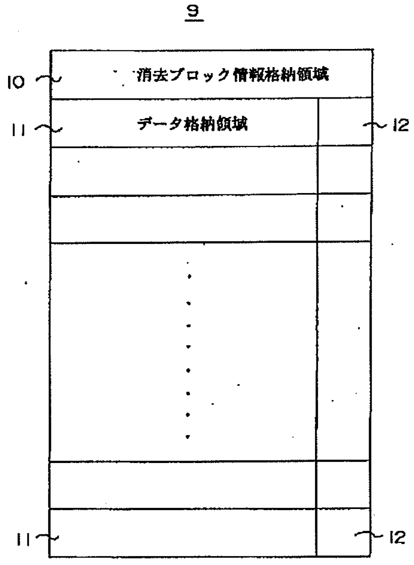
【図１３】


スタート
アドレスを受けとる
30
31
CHS形式？
YES
NO
32
CHS→LSAに変換
空き領域の確認
33
34
空きエリアあるか？
NO
YES
35
データの書き込み，アドレス変換テーブルの更新
データ更新の時の旧データの有効データ確認フラグの更新と、書き込み用アドレスポインタの更新
39
エンド
36
消去するブロックを決定
37
有効データを空きエリアにコピー
38
ブロック消去，消去したブロックの消去回数更新

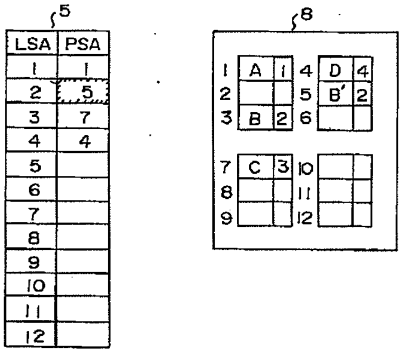
【図１４】


2
1
ホスト
3
インターフェース回路
4
CPU
5
論理セクタ/物理セクタアドレス変換テーブル
6
フラッシュ制御回路
7
データ入出力用セクタバッファ
8
フラッシュメモリ

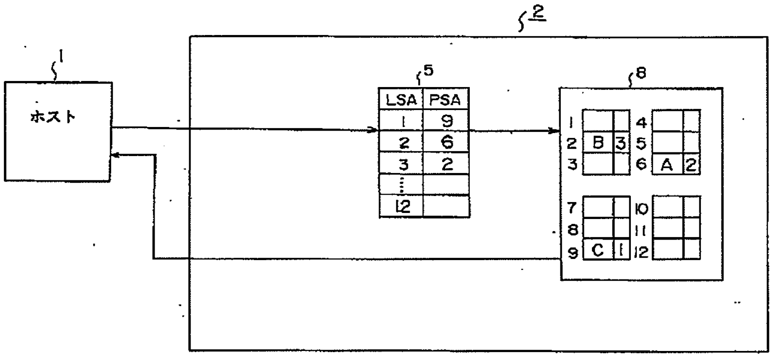
【図１７】


消去ブロック情報格納領域
データ格納領域

【図２０】


LSA
PSA
1 1
2 5
3 7
4 4
1 A 1
3 B 2
4 D 4
5 B′ 2
7 C 3

【図１８】


2
1
ホスト
5
LSA PSA
1 9
2 6
3 2
12
8
1 2 3
B 3
4 5 6
A 2
7 8 9
C 1
10 11 12

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] The address translation table for changing into a physical elimination block number the flash memory which has two or more elimination blocks which consist of an elimination block information storing field which stores the count of elimination of the elimination block concerned, two or more data-storage fields which store data, and a logic sector-address storing field for every data-storage field of said which stores a logic sector address, and said logic sector address, the RAM disk equipment which carry out [having had the control means which manages the data on said flash memory based on said physical elimination block number in a list, and] as the description.
[Claim 2] Said control means is RAM disk equipment according to claim 1 characterized by changing into a physical elimination block number the logic sector address inputted based on said address translation table, discovering the newest applicable data storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory, and reading the contents of said newest applicable data storage field.
[Claim 3] Said control means is RAM disk equipment according to claim 1 characterized by writing an applicable physics elimination block number in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table while writing in data from on an applicable elimination block to a data storage field continuously downward and writing in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field.

[Translation done.]

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application] This invention relates to RAM disk equipments, such as a RAM disk card which used the flash memory as a storage.
[0002]
[Description of the Prior Art] In case comparatively mass data are made to memorize in the field of a personal computer today, magnetic storage media, such as a hard disk drive unit, are used in many cases. Although just power consumption is large as for this, it is because cost performance is very good.
[0003] On the other hand, the RAM disk equipment which operates semiconductor memory, such as a flash memory, like the above-mentioned hard disk drive unit appeared. Since the mechanical motor part of this RAM disk equipment does not exist unlike the above-mentioned hard disk drive unit, although cost performance falls behind a magnetic storage medium, it employs the field of a low power and high-reliability efficiently, and is spreading through a Personal Digital Assistant etc.
[0004] In addition, the description of a flash memory is as follows. It is the nonvolatile memory in which the writing of data and elimination are possible electrically the 1st. Data cannot be overwritten at the memory cell in which data are already written [2nd] (for this reason, elimination actuation always hangs around). The elimination units of the 3rd data are several K - a dozens KByte unit. It writes in the 4th and the count of elimination has a limit.
[0005] It explains referring to drawing 14 , drawing 15 , drawing 16 , and drawing 17 about the configuration of conventional RAM disk equipment. Drawing 14 is the block diagram showing the conventional RAM disk equipment whole configuration. Drawing 15 is drawing showing the internal configuration of the address translation table of drawing 14 . Drawing 16 is drawing showing the internal configuration of the flash memory of drawing 14 . Moreover, drawing 17 is drawing showing the internal configuration of an elimination block of drawing 16 .
[0006] Conventional RAM disk equipment 2 is equipped with an interface circuitry 3, CPU4, an address translation table 5, the flash plate control circuit 6, the sector buffer 7 for data I/O, and a flash memory 8 in drawing 14 .
[0007] The typical examples of the host 1 linked to RAM disk equipment 2 are a notebook computer and a Personal Digital Assistant. Removable type RAM disk equipment 2 has a current place and a card mold in use. An interface circuitry 3 exchanges information with a host 1. CPU4 outputs I/O of data, and the instruction to a flash memory 8.
[0008] A logical sector / physical sector-address translation table 5 is tables for changing a logic sector address into a physical sector address. A logic sector address (LSA:Logical Sector Address) is a sector address which a host 1 specifies as RAM disk equipment 2. Moreover, a physical sector address (PSA:Physical Sector Address) is the address of the flash memory 8 used within RAM disk equipment 2.
[0009] The flash plate control circuit 6 performs data processing of the flash memory 8 which is not complicated. Delivery of simple data etc. is performed in the flash plate control circuit 6, and other processings are performed by CPU4. In data, the sector buffer 7 for data I/O is used, in case data are inputted into a flash memory 8 through an output or an interface circuitry 3 through an interface

circuitry 3 from a flash memory 8.
[0010] In drawing 15 , an address translation table 5 consists of the logic sector-address (LSA) storing section and the physical sector-address (PSA) storing section.
[0011] The logic sector address is saved in the LSA storing section. The contents are being fixed. In addition, in practice, if a logic sector address is sent by the host 1, it has structure out of which an electrical potential difference is built over the address pin of Volatility RAM for translation tables based on logic sector-address data, and PSA data come. In order to make it easy to explain, it is supposed that there is the LSA storing section by correspondence with the PSA storing section. In the PSA storing section, the sector number of the flash memory (drawing 1-n) 8 of arbitration is saved. By using this address translation table 5, data can be saved at the convenient physical sector address for internal management, without being influenced by the logic sector address specified by a host 1. Since this address translation table 5 is written in and eliminated frequently, constituting from volatile RAM, such as SRAM and DRAM, is common.
[0012] The capacity of this address translation table 5 is as follows at the time of the following conditions. When 20 megabytes (MByte) of flash memory 8 is used and the I/O unit (sector) of data is made into 512 bytes (Byte), the number of sectors in RAM disk equipment 2 is as follows.
Sector number =20 megabyte /512 byte in RAM disk equipment 2 = 40960 sectors [0013] Next, the number of bits required in case the binary number expression of "40960" is carried out serves as ln40960/ln 2= 15.3, and is needed 16 figures.
[0014] Thereby, the capacity of the required address translation table 5 becomes 40960x16=655360 bit, and, finally is needed 80 K bytes (KByte).
[0015] Setting to drawing 16 , a flash memory 8 consists of two or more elimination blocks 9 and two or more spare elimination blocks 9.
[0016] A flash memory 8 is written in electrically and is eliminable nonvolatile memory. Data can be changed without there being also no need for backup by the cell like DRAM-SRAM, and removing from a board unlike EPROM, since elimination of data is electrically possible since it did not volatilize. Since 1-bit data are memorizable in one cel, memory is more cheaply [than EERROM] producible. The above point hits the advantage of a flash memory 8. In case it is that 10,000 - about 100,000 times of upper limits are in the count of elimination, and writing as demerit, it is raised that elimination actuation is surely the need (data are unable to overwrite the cel already written in for this reason), that an elimination unit is a block unit of several K - dozens KByte(s), etc.
[0017] In drawing 17 , one elimination block 9 has the elimination block information storing field 10, two or more data storage fields 11, and the LSA storing field 12 for every data storage field 11 at the head.
[0018] The current count of block elimination is stored in the elimination block information storing field 10. The data storage field 11 is usually 512 bytes (= one sector) in magnitude. The LSA storing field 12 exists for every sector, and in case it writes in data, it stores LSA specified by a host 1. When this constitutes a logical sector / physical sector-address translation table 5 from volatility RAM, it is for data to disappear to power-source OFF and coincidence. It is used, in case the LSA storing field 12 of all sectors is searched and the volatile RAM table 5 is reconstructed, when a power source is turned ON.
[0019] Below, it explains, referring to drawing 18 , drawing 19 , and drawing 20 about actuation of conventional RAM disk equipment. Drawing 18 is drawing for explaining read-out actuation of conventional RAM disk equipment. Moreover, drawing 19 and drawing 20 are drawings for explaining write-in actuation of conventional RAM disk equipment.
[0020] Unlike a hard disk drive unit, the RAM disk equipment 2 using a flash memory 8 cannot overwrite data. Therefore, making the address translation table 5 showing whether the logic sector address and data of the data sent by the host 1 are written in the physical sector address of flash memory 8 throat memorize in Volatility RAM is performed. It becomes possible to use the storage region of a flash memory 8 effectively, without being influenced by LSA by using this address translation table 5.
[0021] First, drawing 18 explains read-out actuation of the data from RAM disk equipment 2. A host 1 sends the sector address of data to read to RAM disk equipment 2. There are two kinds of address data

sent by the host 1. They are a LSA format and a CHS format. A CHS format specifies a data area to a LSA format specifying a sector with the serial number to 1-n in the combination of three data called the cylinder head sector used with a hard disk drive unit. Within RAM disk equipment 2, in order to use the LSA/PSA address translation table 5, when the data of a CHS format are inputted from a host 1, it changes into LSA within an interface circuitry, and moves to the next activity.

[0022] CPU4 carries out address translation of the LSA which the host 1 specified using the address translation table 5 to PSA. Data are read from the inside of the flash memory 8 corresponding to PSA to the last.

[0023] For example, it is changed into the PSA "6" by the address translation table 5 when LSA specified by a host 1 is "2." By this, the data "A" will be read as shown in drawing 18 . "2" which is LSA is stored in the LSA storing field 12.

[0024] Next, drawing 19 and drawing 20 explain write-in actuation of the data to RAM disk equipment 2. The data "A", "B", and "C" make an initial state the condition of being stored in "1" of PSA, "3", and "7." That it must be careful in case data are written in is the point that the re-writing of data cannot do a flash memory 8. In the case of the above-mentioned initial state, the field of PSA "1", "3", and "7" corresponds.

[0025] When a host 1 specifies LSA in which data are not written, CPU4 writes data in the suitable free area in a flash memory 8 (PSA "2", "4" - "6" and "8" - "12"), and updates the data in an address translation table 5. Drawing 19 is an example when a host specifies the writing of the data "D" to LSA "4." LSA specified by data "D" and a host 1 is written in a free area PSA "4", and the value "4" of PSA is written in the part of PSA corresponding to LSA "4" of an address translation table 5.

[0026] Even if it is the case (for example, overwriting of a same name file) where the re-writing to the field [host / 1] where data are already written in is required, re-write-in data are written in the free area of a flash memory 8, and an address translation table 5 is updated. Drawing 20 is a result at the time of re-writing in the data of LSA "2." Updating data "B'" are written in a free area PSA "5", and PSA corresponding to LSA "2" of an address translation table 5 is updated with "5." In addition, that PSA "3" is used data must recognize CPU4 in a card.

[0027]

[Problem(s) to be Solved by the Invention] With conventional RAM disk equipment which was mentioned above, since the memory area where an address translation table 5 stores one PSA in every sector (smallest unit of data control) was needed, the address translation table 5 also had the trouble of becoming large capacity as the flash memory 8 became large capacity.

[0028] what was made in order that this invention might solve the trouble mentioned above -- it is -- the former -- the performance of equipment, i.e., a read-out rate, and abbreviation -- it aims at obtaining the RAM disk equipment which maintains a comparable read-out rate and can make small capacity of the address translation table for memory management.

[0029]

[Means for Solving the Problem] The elimination block information storing field where the RAM disk equipment concerning this invention stores the count of elimination of the elimination block concerned, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, It has an address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number.

[0030] Moreover, said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, and discovers the newest applicable data storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory, and the RAM disk equipment concerning this invention reads the contents of said newest applicable data storage field.

[0031] Furthermore, the RAM disk equipment concerning this invention writes an applicable physics elimination block number in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table while said control means writes in data from on an applicable elimination block to a data storage field continuously downward and writes in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field.

[0032]

[Function] In the RAM disk equipment concerning this invention The elimination block information storing field which stores the count of elimination of the elimination block concerned, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, Since it had the address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number, the address translation table for data control can be made small.

[0033] Moreover, in the RAM disk equipment concerning this invention, since said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, discovers the newest applicable data storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory and reads the contents of said newest applicable data storage field, the address translation table for data control can be made small.

[0034] Furthermore, in the RAM disk equipment concerning this invention, since an applicable physics elimination block number is written in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table while said control means writes in data from on an applicable elimination block to a data storage field continuously downward and writes in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field, the address translation table for data control can be made small.

[0035]

[Example]

It explains below example 1., referring to drawing 1 , drawing 2 , drawing 3 , and drawing 4 about the configuration of the example 1 of this invention. Drawing 1 is the block diagram showing the whole example 1 configuration of this invention. Drawing 2 is drawing showing the internal configuration of the address translation table of drawing 1 . Drawing 3 is drawing showing the internal configuration of the flash memory of drawing 1 . Drawing 4 is drawing showing the internal configuration of an elimination block of drawing 3 . In addition, the same sign shows the same or a considerable part among each drawing.

[0036] In drawing 1 , RAM disk equipment 2A concerning this example 1 is equipped with an interface circuitry 3, CPU4, address translation table 5A, flash plate control circuit 6A, the sector buffer 7 for data I/O, and flash memory 8A.

[0037] The capacity of address translation table 5A is a small point, and the difference from conventional RAM disk equipment 2 can enlarge the part and flash memory 8A. Moreover, in addition to the function of the conventional flash plate control circuit 6, flash plate control circuit 6A includes the hard logic sector-address comparator circuit 61. Furthermore, CPU4 is written in with the object for read-out, and performs data control on flash memory 8A using the address pointer of business. namely, the thing for which RAM disk equipment 2A uses the above-mentioned logic sector-address comparator circuit 61 and an address pointer -- it is -- the former and abbreviation -- capacity of address translation table 5A can be made small, maintaining comparable performance. In addition, in this example 1, the control means concerning this invention is read, is written in with business, and consists of CPU4 using the address pointer of business, flash plate control circuit 6A including the logic sector-address comparator circuit 61, and a sector buffer 7 for data I/O.

[0038] Address translation table 5A is a table for changing a logic sector address into a physical elimination block number. A physical elimination block number (PBN:Physical Block Number) is the block address of flash memory 8A used within RAM disk equipment 2A.

[0039] In drawing 2 , address translation table 5A consists of the logic sector-address (LSA) storing section and the physical elimination block number (PBN) storing section.

[0040] The logic sector address is saved in the LSA storing section. The contents are being fixed. In addition, in practice, if a logic sector address is sent by the host 1, it has structure out of which an electrical potential difference is built over the address pin of Volatility RAM for translation tables based on logic sector-address data, and PBN data come. In order to make it easy to explain, it is supposed that there is the LSA storing section by correspondence with the PBN storing section. In the PBN storing section, the physical elimination block number of flash memory 8A of arbitration is saved. By using this address translation table 5A, data can be saved at the convenient physical elimination block for internal management, without being influenced by the logic sector address specified by a host 1. Since this address translation table 5A is written in and eliminated frequently, constituting from volatile RAM, such as SRAM and DRAM, is common.

[0041] However, unlike the former, in this address translation table 5A, only PBN is understood from the data of LSA. For example, when the flash memory whose size of an elimination block is 64 K bytes is used, the sector (= 512 bytes) of about 100 order exists in the interior (if there are not the elimination block information storing field 10 and LSA storing field 12 grade, a maximum of 128 sector exists). For this reason, the device for searching the target data is needed. About this, it mentions later.

[0042] The capacity of this address translation table 5A is as follows. If flash memory [20 megabytes of] 8A is used and 1 sector is made into 512 bytes, the number of sectors in RAM disk equipment 2A will be set to 40960 as usual. However, since the PBN storing section of address translation table 5A is made to memorize a block number, the block count in RAM disk equipment 2A is as follows when 1 block (elimination block) is made into 64 K bytes. Block count =20 megabyte /64 K byte in RAM disk equipment 2A = 320 blocks [0043] Next, the number of bits required in case the binary number expression of "320" is carried out serves as ln320/ln 2= 8.3, and is needed 9 figures. Thereby, a required capacity of address translation table 5A becomes 40960x9=368640 bit, and, finally is needed 45 K bytes. This is conventional about 1/2.

[0044] In drawing 3 , flash memory 8A consists of two or more elimination block 9A and two or more spare elimination block 9A. In addition, since the size of an object block of data (memory) management is the same as an elimination unit, an elimination block is called. The flash memory of a block elimination mold (elimination block unit several K- dozens of K bytes) is used for flash memory 8A used for main memory as usual.

[0045] In drawing 4 , one elimination block 9A has the elimination block information storing field 10, two or more data storage fields 11, the LSA (logic sector address) storing field 12 for every data storage field 11, and the effective-data check flag 13 for every data storage field 11 at the head. Except effective-data check flag 13, it is the same as usual. This effective-data check flag 13 is for CPU4 to distinguish an effective data and an invalid data eliminable at any time. An effective data expresses with "FF" (11111111) and an invalid data is expressed with "00" (00000000). In addition, a reverse expression may be used. CPU4 rewrites the effective-data check flag 13 corresponding to the data storage field 11 which became unnecessary from "FF" to "00", when overwrite and elimination of data are performed.

[0046] Below, it explains, referring to from drawing 5 to drawing 13 about actuation of this example 1. Drawing 5 - drawing 8 are drawings for explaining write-in actuation of this example 1. Drawing 9 and drawing 10 are drawings for explaining read-out actuation of this example 1. Drawing 11 and drawing 12 are flow charts which show read-out actuation of this example 1. Drawing 13 is a flow chart which shows write-in actuation of this example 1.

[0047] First, drawing 5 - drawing 8 explain the data write-in actuation to RAM disk equipment 2A. The difference with the former is not writing data in other elimination blocks until all the sectors within one elimination block (data storage field 11) become write-in ending. Moreover, data are continued and written in downward from on an elimination block (it does not write in at random). This is updated using "the address pointer for writing" at [one / every] every writing (increment). Here, an "address pointer" is for memorizing the physical sector address (PSA) of arbitration. In this example 1, one "the address pointer for read-out" and "the address pointer for writing" are used, respectively at the time of read-out of data and writing.

[0048] Explanation of operation is advanced from the condition that there are no data, into flash memory 8A in RAM disk equipment 2A. At this time, the address pointer for writing is pointing to "1." In the case of data writing, it writes in with the data which should be first written in from a host 1, and the address is sent at it. Since the address can consider the case of a CHS format, it is altogether changed into a LSA format as usual. Since address-format information is also sent by the host 1, a CHS format or a LSA format is easily distinguishable by the RAM disk equipment 2A side.

[0049] Even if LSA is sent at random, as shown in drawing 5 , writing is performed from PBN "1." For example, when the write request of data "A" is in LSA "3" from a host 1, CPU4 writes LSA "3" in the logic sector-address storing field 12 which writes data "A" in the data storage field 11 of the head of PBN "1", and is equivalent to it according to the address pointer for writing. And a write-in address pointer is updated. That is, a write-in address pointer is set to "2." Furthermore, "1" is written in the PBN storing section corresponding to LSA "3" of address translation table 5A. In addition, in drawing 5 - drawing 9 , elimination block 9of flash memory 8A A is setting the data storage field 11 to three so that it may be easy to explain.

[0050] Drawing 6 is in the condition from which the clean elimination block turned into 1 more block. At this time, the address pointer for writing is updated with "1" ->"2" ->"3" ->"--" ->"8" -> "9", and is pointing to "10" which is the data storage field 11 of the head of PBN "4." Here, if it writes in more than this, since a clean elimination block is not securable, the elimination block of flash memory 8A will be cleaned.

[0051] CPU4 checks the condition within all elimination blocks, and determines the optimal block for elimination. The optimal blocks are the elimination block based on the effective-data check flag 13 with which effective data seldom exist, an elimination block with few counts of elimination based on the contents of the elimination block information storing field 10, etc. In drawing 6 , the count of elimination of each elimination block is the same at "0", and PBN "1" makes the data "A'" of PBN "1" since there is little effective data that is, to eliminate this block and secure a clean sector, since overwrite of data "A" is meant.

[0052] First, as shown in drawing 7 , effective data are evacuated to clean PBN "4", and then address translation table 5A is updated. Then, as shown in drawing 8 , block elimination of the PBN "1" is carried out, and only "1" increases the count of block elimination. It means that one clean block and one clean sector were secured according to this activity. At this time, the address pointer for writing is pointing to "12."

[0053] Next, drawing 9 and drawing 10 explain the data read-out actuation from RAM disk equipment 2A. First, the address sent by the host 1 is unified into a LSA format. Next, PBN is deduced from LSA using address translation table 5A.

[0054] Next, LSA stored in the logic sector-address storing field 12 in obtained PBN is checked sequentially from the bottom. Under the present circumstances, as shown in drawing 10 , LSA sent by the host 1 is compared with LSA stored in the logic sector-address storing field 12 in Relevance PBN in the logic sector-address comparator circuit 61 in flash plate control circuit 6A. It is data which the contents of the data storage field 11 (sector) when being in agreement should read. At this time, the value set to PSA when being in agreement with the address pointer for read-out plus 1 is set. When read-out data are two or more sectors, data are read from the sector address to which the address pointer for read-out points at the time of the next read-out.

[0055] For example, when the data [LSA] "2" have been sent from the host 1, as shown in drawing 9 , PBN "3" is obtained by address translation table 5A. Next, the logic sector address within the elimination block of PBN "3" is checked sequentially from the bottom. Since bottom LSA is "6", it does not correspond. Next, since the next LSA is "2", it is in agreement. Therefore, the data which should be read will be called "H." At this time, the address pointer for read-out is "9."

[0056] Unlike the former, when an elimination block becomes large, the time amount which retrieval takes will be taken. However, although they generally need the retrieval which is a maximum of 100 step extent for the first data retrieval since data have many things of the size more than 512 byte (one sector), and the data area of about 100 sectors is in one elimination block Since 2nd henceforth has very high possibility that data are in the next sector of the sector read last time, the count of retrieval can be sharply decreased by memorizing the address of the next sector of the sector read last time to the address pointer for read-out.

[0057] It continues and the flow chart of drawing 11 and drawing 12 explains the data read-out actuation from RAM disk equipment 2A. Since the address pointer for read-out is not determined in case the sector of the head of a file is read when treating the file (data) which consists of 1 - dozens of sectors, retrieval processing of drawing 11 is performed. Moreover, since the address pointer for read-out is set up when reading the data of the next sector of the above-mentioned file, drawing 12 is processed. The difference from retrieval processing of drawing 11 is reading data, without using LSA from a host 1, although processing of drawing 12 receives the address (LSA) of the data which should be read from a host 1. Retrieval processing of drawing 11 is performed also when the address pointer for read-out else [in the case of reading the data of the head sector of a file] comes to the last sector address within an elimination block. This is for the elimination block with which the sector read to a degree exists not to be clear anymore. CPU4 recognizes except [its] to be the head of a file based on a file name etc., and recognizes the last sector address based on a block size etc.

[0058] The sector information on the data which should be read from introduction and a host 1 is received (step 20). this -- the format of LSA -- or it is sent in a CHS format. In order to unify into a LSA format, when it has been sent in CHS data format, it changes into a LSA format (steps 21-22). This conversion may use CPU4 in RAM disk equipment 2A, and may give the circuit of dedication to the interior of RAM disk equipment.

[0059] Next, LSA is changed into PBN (step 23). Address translation table 5A is used for this. Next, LSA stored in the logic sector-address storing field 12 in determined PBN is read. This read LSA is compared with LSA from a host 1 in the logic sector-address comparator circuit 61, if inharmonious, LSA stored in the next logic sector-address storing field 12 will be read, the inside of Above PBN is searched from the bottom, and the same comparison is performed until it is in agreement. In addition, the direction of retrieval may be performed from a top depending on the case (steps 24-26).

[0060] If LSA(s) are in agreement, data will be read from the corresponding data storage field 11 (step 27). And the following sector address which read the above-mentioned data to the address pointer for read-out is set (step 28).

[0061] It is as follows, when continuing and reading the data of the following sector address. Steps 40-42 are the same as the above-mentioned steps 20-22. This is because LSA from a host 1 must be made into reception and the processed form in order to maintain the compatibility of an interface.

[0062] Next, data are read from the sector address to which the address pointer for read-out points (step 43). And "1" is added and updated to the address pointer for read-out. When you read one certain file, refer to the address translation table 5 only for the number of sectors which constitutes the file from conventional RAM disk equipment. However, in this example 1, although time amount is taken to the data retrieval within a first-time elimination block, since data are read according to the address pointer for read-out about the sector after it, about the sector after the 2nd, it can process at a high speed rather than equipment conventionally [above-mentioned].

[0063] Moreover, in raising the reliability of data read-out at the sacrifice of a read-out rate, it compares and checks LSA received from the host 1, and LSA in the LSA storing field 12 which is the sector to which the address pointer for read-out points by the degree of step 43. In addition, retrieval processing after step 23 is performed here at the time of an inequality.

[0064] It continues and the flow chart of drawing 13 explains the data write-in actuation to RAM disk equipment 2A. Since steps 30-32 of drawing 13 are the same as steps 20-22 of drawing 11 , explanation is omitted. First, an empty sector is checked based on directions of the address pointer for writing (step 33).

[0065] When there is an empty sector, while writing in data there, address translation table 5A is updated (steps 34-35). That is, PBN which wrote in data is saved in the PBN storing section of corresponding LSA.

[0066] In step 34, when there is no empty sector based on the address pointer for writing, it processes as follows (steps 36-38). First, the block eliminated based on the count of elimination, the number of invalid datas, etc. is determined. Next, an effective data is copied to a free block. Then, the determined block is eliminated and the eliminated count of elimination of a block is updated. And it progresses to step 35.

[0067] In renewal of data, the effective-data check flag 13 of the old data is updated from "FF" to "00." Moreover, the address pointer for writing is updated (step 39). In this example 1, in order to write in data continuously, the address pointer for writing will point to the following sector address after data write-in termination. In addition, when all the sectors within one elimination block are buried with data, the sector address to which CPU4 determines and points to the elimination block which should be written in a degree is decided. For example, when the elimination (it is clean) block of two or more openings exists, CPU4 determines the empty elimination block which should be written in a degree based on the count of elimination etc.

[0068] This example 1 can make small capacity of address translation table 5A by using the object for read-out and the address pointer for writing, and the logic sector-address comparator circuit 61, without dropping performance, such as a read-out rate of the conventional RAM disk card (equipment). By making small capacity of address translation table 5A, large capacity-ization of a RAM disk card can be advanced reasonable. When the conventional address translation table 5 is used, the size of 80KB is needed by 20MB of RAM disk, and the size of 160KB (1.25Mbit) is needed by 40MB of RAM disk. Since a flash memory can be extended and carried in the tooth space in which the cost concerning Volatility RAM could be reduced and the volatile RAM memory for address translation tables was carried if this becomes small in one half of sizes about, the capacity of RAM disk equipment can be increased.

[0069]

[Effect of the Invention] The elimination block information storing field which stores the count of elimination of the elimination block concerned as the RAM disk equipment concerning this invention was explained above, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, Since it had the address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number The effectiveness that the address translation table for data control can be made small is done so.

[0070] Moreover, since said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, discovers the newest applicable data-storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory and reads the contents of said newest applicable data-storage field as explained above, the RAM disk equipment concerning this invention does so the effectiveness that the address translation table for data control can be made small.

[0071] Furthermore, the RAM disk equipment concerning this invention Said control means writes in data from on an applicable elimination block to a data storage field continuously downward as explained above. While writing in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field Since an applicable physics elimination block number is written in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table, the effectiveness that the address translation table for data control can be made small is done so.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL FIELD

[Industrial Application] This invention relates to RAM disk equipments, such as a RAM disk card which used the flash memory as a storage.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

PRIOR ART

[Description of the Prior Art] In case comparatively mass data are made to memorize in the field of a personal computer today, magnetic storage media, such as a hard disk drive unit, are used in many cases. Although just power consumption is large as for this, it is because cost performance is very good.
[0003] On the other hand, the RAM disk equipment which operates semiconductor memory, such as a flash memory, like the above-mentioned hard disk drive unit appeared. Since the mechanical motor part of this RAM disk equipment does not exist unlike the above-mentioned hard disk drive unit, although cost performance falls behind a magnetic storage medium, it employs the field of a low power and high-reliability efficiently, and is spreading through a Personal Digital Assistant etc.
[0004] In addition, the description of a flash memory is as follows. It is the nonvolatile memory in which the writing of data and elimination are possible electrically the 1st. Data cannot be overwritten at the memory cell in which data are already written [2nd] (for this reason, elimination actuation always hangs around). The elimination units of the 3rd data are several K - a dozens KByte unit. It writes in the 4th and the count of elimination has a limit.
[0005] It explains referring to drawing 14 , drawing 15 , drawing 16 , and drawing 17 about the configuration of conventional RAM disk equipment. Drawing 14 is the block diagram showing the conventional RAM disk equipment whole configuration. Drawing 15 is drawing showing the internal configuration of the address translation table of drawing 14 . Drawing 16 is drawing showing the internal configuration of the flash memory of drawing 14 . Moreover, drawing 17 is drawing showing the internal configuration of an elimination block of drawing 16 .
[0006] Conventional RAM disk equipment 2 is equipped with an interface circuitry 3, CPU4, an address translation table 5, the flash plate control circuit 6, the sector buffer 7 for data I/O, and a flash memory 8 in drawing 14 .
[0007] The typical examples of the host 1 linked to RAM disk equipment 2 are a notebook computer and a Personal Digital Assistant. Removable type RAM disk equipment 2 has a current place and a card mold in use. An interface circuitry 3 exchanges information with a host 1. CPU4 outputs I/O of data, and the instruction to a flash memory 8.
[0008] A logical sector / physical sector-address translation table 5 is tables for changing a logic sector address into a physical sector address. A logic sector address (LSA:Logical Sector Address) is a sector address which a host 1 specifies as RAM disk equipment 2. Moreover, a physical sector address (PSA:Physical Sector Address) is the address of the flash memory 8 used within RAM disk equipment 2.
[0009] The flash plate control circuit 6 performs data processing of the flash memory 8 which is not complicated. Delivery of simple data etc. is performed in the flash plate control circuit 6, and other processings are performed by CPU4. In data, the sector buffer 7 for data I/O is used, in case data are inputted into a flash memory 8 through an output or an interface circuitry 3 through an interface circuitry 3 from a flash memory 8.
[0010] In drawing 15 , an address translation table 5 consists of the logic sector-address (LSA) storing section and the physical sector-address (PSA) storing section.

[0011] The logic sector address is saved in the LSA storing section. The contents are being fixed. In addition, in practice, if a logic sector address is sent by the host 1, it has structure out of which an electrical potential difference is built over the address pin of Volatility RAM for translation tables based on logic sector-address data, and PSA data come. In order to make it easy to explain, it is supposed that there is the LSA storing section by correspondence with the PSA storing section. In the PSA storing section, the sector number of the flash memory (drawing 1-n) 8 of arbitration is saved. By using this address translation table 5, data can be saved at the convenient physical sector address for internal management, without being influenced by the logic sector address specified by a host 1. Since this address translation table 5 is written in and eliminated frequently, constituting from volatile RAM, such as SRAM and DRAM, is common.
[0012] The capacity of this address translation table 5 is as follows at the time of the following conditions. When 20 megabytes (MByte) of flash memory 8 is used and the I/O unit (sector) of data is made into 512 bytes (Byte), the number of sectors in RAM disk equipment 2 is as follows.
Sector number =20 megabyte /512 byte in RAM disk equipment 2 = 40960 sectors [0013] Next, the number of bits required in case the binary number expression of "40960" is carried out serves as ln40960/ln 2= 15.3, and is needed 16 figures.
[0014] Thereby, the capacity of the required address translation table 5 becomes 40960x16=655360 bit, and, finally is needed 80 K bytes (KByte).
[0015] Setting to drawing 16 , a flash memory 8 consists of two or more elimination blocks 9 and two or more spare elimination blocks 9.
[0016] A flash memory 8 is written in electrically and is eliminable nonvolatile memory. Data can be changed without there being also no need for backup by the cell like DRAM-SRAM, and removing from a board unlike EPROM, since elimination of data is electrically possible since it did not volatilize. Since 1-bit data are memorizable in one cel, memory is more cheaply [than EERROM] producible. The above point hits the advantage of a flash memory 8. In case it is that 10,000 - about 100,000 times of upper limits are in the count of elimination, and writing as demerit, it is raised that elimination actuation is surely the need (data are unable to overwrite the cel already written in for this reason), that an elimination unit is a block unit of several K - dozens KByte(s), etc.
[0017] In drawing 17 , one elimination block 9 has the elimination block information storing field 10, two or more data storage fields 11, and the LSA storing field 12 for every data storage field 11 at the head.
[0018] The current count of block elimination is stored in the elimination block information storing field 10. The data storage field 11 is usually 512 bytes (= one sector) in magnitude. The LSA storing field 12 exists for every sector, and in case it writes in data, it stores LSA specified by a host 1. When this constitutes a logical sector / physical sector-address translation table 5 from volatility RAM, it is for data to disappear to power-source OFF and coincidence. It is used, in case the LSA storing field 12 of all sectors is searched and the volatile RAM table 5 is reconstructed, when a power source is turned ON.
[0019] Below, it explains, referring to drawing 18 , drawing 19 , and drawing 20 about actuation of conventional RAM disk equipment. Drawing 18 is drawing for explaining read-out actuation of conventional RAM disk equipment. Moreover, drawing 19 and drawing 20 are drawings for explaining write-in actuation of conventional RAM disk equipment.
[0020] Unlike a hard disk drive unit, the RAM disk equipment 2 using a flash memory 8 cannot overwrite data. Therefore, making the address translation table 5 showing whether the logic sector address and data of the data sent by the host 1 are written in the physical sector address of flash memory 8 throat memorize in Volatility RAM is performed. It becomes possible to use the storage region of a flash memory 8 effectively, without being influenced by LSA by using this address translation table 5.
[0021] First, drawing 18 explains read-out actuation of the data from RAM disk equipment 2. A host 1 sends the sector address of data to read to RAM disk equipment 2. There are two kinds of address data sent by the host 1. They are a LSA format and a CHS format. A CHS format specifies a data area to a LSA format specifying a sector with the serial number to 1-n in the combination of three data called the cylinder head sector used with a hard disk drive unit. Within RAM disk equipment 2, in order to use the

LSA/PSA address translation table 5, when the data of a CHS format are inputted from a host 1, it changes into LSA within an interface circuitry, and moves to the next activity.

[0022] CPU4 carries out address translation of the LSA which the host 1 specified using the address translation table 5 to PSA. Data are read from the inside of the flash memory 8 corresponding to PSA to the last.

[0023] For example, it is changed into the PSA "6" by the address translation table 5 when LSA specified by a host 1 is "2." By this, the data "A" will be read as shown in drawing 18 . "2" which is LSA is stored in the LSA storing field 12.

[0024] Next, drawing 19 and drawing 20 explain write-in actuation of the data to RAM disk equipment 2. The data "A", "B", and "C" make an initial state the condition of being stored in "1" of PSA, "3", and "7." That it must be careful in case data are written in is the point that the re-writing of data cannot do a flash memory 8. In the case of the above-mentioned initial state, the field of PSA "1", "3", and "7" corresponds.

[0025] When a host 1 specifies LSA in which data are not written, CPU4 writes data in the suitable free area in a flash memory 8 (PSA "2", "4" - "6" and "8" - "12"), and updates the data in an address translation table 5. Drawing 19 is an example when a host specifies the writing of the data "D" to LSA "4." LSA specified by data "D" and a host 1 is written in a free area PSA "4", and the value "4" of PSA is written in the part of PSA corresponding to LSA "4" of an address translation table 5.

[0026] Even if it is the case (for example, overwriting of a same name file) where the re-writing to the field [host / 1] where data are already written in is required, re-write-in data are written in the free area of a flash memory 8, and an address translation table 5 is updated. Drawing 20 is a result at the time of re-writing in the data of LSA "2." Updating data "B'" are written in a free area PSA "5", and PSA corresponding to LSA "2" of an address translation table 5 is updated with "5." In addition, that PSA "3" is used data must recognize CPU4 in a card.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EFFECT OF THE INVENTION

[Effect of the Invention] The elimination block information storing field which stores the count of elimination of the elimination block concerned as the RAM disk equipment concerning this invention was explained above, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, Since it had the address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number The effectiveness that the address translation table for data control can be made small is done so.
[0070] Moreover, since said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, discovers the newest applicable data-storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory and reads the contents of said newest applicable data-storage field as explained above, the RAM disk equipment concerning this invention does so the effectiveness that the address translation table for data control can be made small.
[0071] Furthermore, the RAM disk equipment concerning this invention Said control means writes in data from on an applicable elimination block to a data storage field continuously downward as explained above. While writing in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field Since an applicable physics elimination block number is written in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table, the effectiveness that the address translation table for data control can be made small is done so.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] With conventional RAM disk equipment which was mentioned above, since the memory area where an address translation table 5 stores one PSA in every sector (smallest unit of data control) was needed, the address translation table 5 also had the trouble of becoming large capacity as the flash memory 8 became large capacity.
[0028] what was made in order that this invention might solve the trouble mentioned above -- it is -- the former -- the performance of equipment, i.e., a read-out rate, and abbreviation -- it aims at obtaining the RAM disk equipment which maintains a comparable read-out rate and can make small capacity of the address translation table for memory management.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

MEANS

[Means for Solving the Problem] The elimination block information storing field where the RAM disk equipment concerning this invention stores the count of elimination of the elimination block concerned, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, It has an address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number.
[0030] Moreover, said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, and discovers the newest applicable data storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory, and the RAM disk equipment concerning this invention reads the contents of said newest applicable data storage field.
[0031] Furthermore, the RAM disk equipment concerning this invention writes an applicable physics elimination block number in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table while said control means writes in data from on an applicable elimination block to a data storage field continuously downward and writes in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

OPERATION

[Function] In the RAM disk equipment concerning this invention The elimination block information storing field which stores the count of elimination of the elimination block concerned, The flash memory which has two or more elimination blocks which consist of two or more data storage fields which store data, and a logic sector-address storing field for said every data storage field which stores a logic sector address, Since it had the address translation table for changing said logic sector address into a physical elimination block number, and the control means which manages the data on said flash memory based on said physical elimination block number, the address translation table for data control can be made small.
[0033] Moreover, in the RAM disk equipment concerning this invention, since said control means changes into a physical elimination block number the logic sector address inputted based on said address translation table, discovers the newest applicable data storage field based on said inputted logic sector address within the applicable physics elimination block on said flash memory and reads the contents of said newest applicable data storage field, the address translation table for data control can be made small.
[0034] Furthermore, in the RAM disk equipment concerning this invention, since an applicable physics elimination block number is written in the physical elimination block number storing section corresponding to said inputted logic sector address of said address translation table while said control means writes in data from on an applicable elimination block to a data storage field continuously downward and writes in the logic sector address inputted into the logic sector-address storing field corresponding to said written-in data storage field, the address translation table for data control can be made small.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EXAMPLE

[Example]
It explains below example 1., referring to drawing 1 , drawing 2 , drawing 3 , and drawing 4 about the configuration of the example 1 of this invention. Drawing 1 is the block diagram showing the whole example 1 configuration of this invention. Drawing 2 is drawing showing the internal configuration of the address translation table of drawing 1 . Drawing 3 is drawing showing the internal configuration of the flash memory of drawing 1 . Drawing 4 is drawing showing the internal configuration of an elimination block of drawing 3 . In addition, the same sign shows the same or a considerable part among each drawing.

[0036] In drawing 1 , RAM disk equipment 2A concerning this example 1 is equipped with an interface circuitry 3, CPU4, address translation table 5A, flash plate control circuit 6A, the sector buffer 7 for data I/O, and flash memory 8A.

[0037] The capacity of address translation table 5A is a small point, and the difference from conventional RAM disk equipment 2 can enlarge the part and flash memory 8A. Moreover, in addition to the function of the conventional flash plate control circuit 6, flash plate control circuit 6A includes the hard logic sector-address comparator circuit 61. Furthermore, CPU4 is written in with the object for read-out, and performs data control on flash memory 8A using the address pointer of business. namely, the thing for which RAM disk equipment 2A uses the above-mentioned logic sector-address comparator circuit 61 and an address pointer -- it is -- the former and abbreviation -- capacity of address translation table 5A can be made small, maintaining comparable performance. In addition, in this example 1, the control means concerning this invention is read, is written in with business, and consists of CPU4 using the address pointer of business, flash plate control circuit 6A including the logic sector-address comparator circuit 61, and a sector buffer 7 for data I/O.

[0038] Address translation table 5A is a table for changing a logic sector address into a physical elimination block number. A physical elimination block number (PBN:Physical Block Number) is the block address of flash memory 8A used within RAM disk equipment 2A.

[0039] In drawing 2 , address translation table 5A consists of the logic sector-address (LSA) storing section and the physical elimination block number (PBN) storing section.

[0040] The logic sector address is saved in the LSA storing section. The contents are being fixed. In addition, in practice, if a logic sector address is sent by the host 1, it has structure out of which an electrical potential difference is built over the address pin of Volatility RAM for translation tables based on logic sector-address data, and PBN data come. In order to make it easy to explain, it is supposed that there is the LSA storing section by correspondence with the PBN storing section. In the PBN storing section, the physical elimination block number of flash memory 8A of arbitration is saved. By using this address translation table 5A, data can be saved at the convenient physical elimination block for internal management, without being influenced by the logic sector address specified by a host 1. Since this address translation table 5A is written in and eliminated frequently, constituting from volatile RAM, such as SRAM and DRAM, is common.

[0041] However, unlike the former, in this address translation table 5A, only PBN is understood from

the data of LSA. For example, when the flash memory whose size of an elimination block is 64 K bytes is used, the sector (= 512 bytes) of about 100 order exists in the interior (if there are not the elimination block information storing field 10 and LSA storing field 12 grade, a maximum of 128 sector exists). For this reason, the device for searching the target data is needed. About this, it mentions later.
[0042] The capacity of this address translation table 5A is as follows. If flash memory [20 megabytes of] 8A is used and 1 sector is made into 512 bytes, the number of sectors in RAM disk equipment 2A will be set to 40960 as usual. However, since the PBN storing section of address translation table 5A is made to memorize a block number, the block count in RAM disk equipment 2A is as follows when 1 block (elimination block) is made into 64 K bytes. Block count =20 megabyte /64 K byte in RAM disk equipment 2A = 320 blocks [0043] Next, the number of bits required in case the binary number expression of "320" is carried out serves as ln320/ln 2= 8.3, and is needed 9 figures. Thereby, a required capacity of address translation table 5A becomes 40960x9=368640 bit, and, finally is needed 45 K bytes. This is conventional about 1/2.
[0044] In drawing 3 , flash memory 8A consists of two or more elimination block 9A and two or more spare elimination block 9A. In addition, since the size of an object block of data (memory) management is the same as an elimination unit, an elimination block is called. The flash memory of a block elimination mold (elimination block unit several K- dozens of K bytes) is used for flash memory 8A used for main memory as usual.
[0045] In drawing 4 , one elimination block 9A has the elimination block information storing field 10, two or more data storage fields 11, the LSA (logic sector address) storing field 12 for every data storage field 11, and the effective-data check flag 13 for every data storage field 11 at the head. Except effective-data check flag 13, it is the same as usual. This effective-data check flag 13 is for CPU4 to distinguish an effective data and an invalid data eliminable at any time. An effective data expresses with "FF" (11111111) and an invalid data is expressed with "00" (00000000). In addition, a reverse expression may be used. CPU4 rewrites the effective-data check flag 13 corresponding to the data storage field 11 which became unnecessary from "FF" to "00", when overwrite and elimination of data are performed.
[0046] Below, it explains, referring to from drawing 5 to drawing 13 about actuation of this example 1. Drawing 5 - drawing 8 are drawings for explaining write-in actuation of this example 1. Drawing 9 and drawing 10 are drawings for explaining read-out actuation of this example 1. Drawing 11 and drawing 12 are flow charts which show read-out actuation of this example 1. Drawing 13 is a flow chart which shows write-in actuation of this example 1.
[0047] First, drawing 5 - drawing 8 explain the data write-in actuation to RAM disk equipment 2A. The difference with the former is not writing data in other elimination blocks until all the sectors within one elimination block (data storage field 11) become write-in ending. Moreover, data are continued and written in downward from on an elimination block (it does not write in at random). This is updated using "the address pointer for writing" at [one / every] every writing (increment). Here, an "address pointer" is for memorizing the physical sector address (PSA) of arbitration. In this example 1, one "the address pointer for read-out" and "the address pointer for writing" are used, respectively at the time of read-out of data and writing.
[0048] Explanation of operation is advanced from the condition that there are no data, into flash memory 8A in RAM disk equipment 2A. At this time, the address pointer for writing is pointing to "1." In the case of data writing, it writes in with the data which should be first written in from a host 1, and the address is sent at it. Since the address can consider the case of a CHS format, it is altogether changed into a LSA format as usual. Since address-format information is also sent by the host 1, a CHS format or a LSA format is easily distinguishable by the RAM disk equipment 2A side.
[0049] Even if LSA is sent at random, as shown in drawing 5 , writing is performed from PBN "1." For example, when the write request of data "A" is in LSA "3" from a host 1, CPU4 writes LSA "3" in the logic sector-address storing field 12 which writes data "A" in the data storage field 11 of the head of PBN "1", and is equivalent to it according to the address pointer for writing. And a write-in address pointer is updated. That is, a write-in address pointer is set to "2." Furthermore, "1" is written in the PBN

storing section corresponding to LSA "3" of address translation table 5A. In addition, in drawing 5 - drawing 9 , elimination block 9of flash memory 8A A is setting the data storage field 11 to three so that it may be easy to explain.

[0050] Drawing 6 is in the condition from which the clean elimination block turned into 1 more block. At this time, the address pointer for writing is updated with "1" ->"2" ->"3" ->"--" ->"8" -> "9", and is pointing to "10" which is the data storage field 11 of the head of PBN "4." Here, if it writes in more than this, since a clean elimination block is not securable, the elimination block of flash memory 8A will be cleaned.

[0051] CPU4 checks the condition within all elimination blocks, and determines the optimal block for elimination. The optimal blocks are the elimination block based on the effective-data check flag 13 with which effective data seldom exist, an elimination block with few counts of elimination based on the contents of the elimination block information storing field 10, etc. In drawing 6 , the count of elimination of each elimination block is the same at "0", and PBN "1" makes the data "A'" of PBN "1" since there is little effective data that is, to eliminate this block and secure a clean sector, since overwrite of data "A" is meant.

[0052] First, as shown in drawing 7 , effective data are evacuated to clean PBN "4", and then address translation table 5A is updated. Then, as shown in drawing 8 , block elimination of the PBN "1" is carried out, and only "1" increases the count of block elimination. It means that one clean block and one clean sector were secured according to this activity. At this time, the address pointer for writing is pointing to "12."

[0053] Next, drawing 9 and drawing 10 explain the data read-out actuation from RAM disk equipment 2A. First, the address sent by the host 1 is unified into a LSA format. Next, PBN is deduced from LSA using address translation table 5A.

[0054] Next, LSA stored in the logic sector-address storing field 12 in obtained PBN is checked sequentially from the bottom. Under the present circumstances, as shown in drawing 10 , LSA sent by the host 1 is compared with LSA stored in the logic sector-address storing field 12 in Relevance PBN in the logic sector-address comparator circuit 61 in flash plate control circuit 6A. It is data which the contents of the data storage field 11 (sector) when being in agreement should read. At this time, the value set to PSA when being in agreement with the address pointer for read-out plus 1 is set. When read-out data are two or more sectors, data are read from the sector address to which the address pointer for read-out points at the time of the next read-out.

[0055] For example, when the data [LSA] "2" have been sent from the host 1, as shown in drawing 9 , PBN "3" is obtained by address translation table 5A. Next, the logic sector address within the elimination block of PBN "3" is checked sequentially from the bottom. Since bottom LSA is "6", it does not correspond. Next, since the next LSA is "2", it is in agreement. Therefore, the data which should be read will be called "H." At this time, the address pointer for read-out is "9."

[0056] Unlike the former, when an elimination block becomes large, the time amount which retrieval takes will be taken. However, although they generally need the retrieval which is a maximum of 100 step extent for the first data retrieval since data have many things of the size more than 512 byte (one sector), and the data area of about 100 sectors is in one elimination block Since 2nd henceforth has very high possibility that data are in the next sector of the sector read last time, the count of retrieval can be sharply decreased by memorizing the address of the next sector of the sector read last time to the address pointer for read-out.

[0057] It continues and the flow chart of drawing 11 and drawing 12 explains the data read-out actuation from RAM disk equipment 2A. Since the address pointer for read-out is not determined in case the sector of the head of a file is read when treating the file (data) which consists of 1 - dozens of sectors, retrieval processing of drawing 11 is performed. Moreover, since the address pointer for read-out is set up when reading the data of the next sector of the above-mentioned file, drawing 12 is processed. The difference from retrieval processing of drawing 11 is reading data, without using LSA from a host 1, although processing of drawing 12 receives the address (LSA) of the data which should be read from a host 1. Retrieval processing of drawing 11 is performed also when the address pointer for read-out else

[in the case of reading the data of the head sector of a file] comes to the last sector address within an elimination block. This is for the elimination block with which the sector read to a degree exists not to be clear anymore. CPU4 recognizes except [its] to be the head of a file based on a file name etc., and recognizes the last sector address based on a block size etc.

[0058] The sector information on the data which should be read from introduction and a host 1 is received (step 20). this -- the format of LSA -- or it is sent in a CHS format. In order to unify into a LSA format, when it has been sent in CHS data format, it changes into a LSA format (steps 21-22). This conversion may use CPU4 in RAM disk equipment 2A, and may give the circuit of dedication to the interior of RAM disk equipment.

[0059] Next, LSA is changed into PBN (step 23). Address translation table 5A is used for this. Next, LSA stored in the logic sector-address storing field 12 in determined PBN is read. This read LSA is compared with LSA from a host 1 in the logic sector-address comparator circuit 61, if inharmonious, LSA stored in the next logic sector-address storing field 12 will be read, the inside of Above PBN is searched from the bottom, and the same comparison is performed until it is in agreement. In addition, the direction of retrieval may be performed from a top depending on the case (steps 24-26).

[0060] If LSA(s) are in agreement, data will be read from the corresponding data storage field 11 (step 27). And the following sector address which read the above-mentioned data to the address pointer for read-out is set (step 28).

[0061] It is as follows, when continuing and reading the data of the following sector address. Steps 40-42 are the same as the above-mentioned steps 20-22. This is because LSA from a host 1 must be made into reception and the processed form in order to maintain the compatibility of an interface.

[0062] Next, data are read from the sector address to which the address pointer for read-out points (step 43). And "1" is added and updated to the address pointer for read-out. When you read one certain file, refer to the address translation table 5 only for the number of sectors which constitutes the file from conventional RAM disk equipment. However, in this example 1, although time amount is taken to the data retrieval within a first-time elimination block, since data are read according to the address pointer for read-out about the sector after it, about the sector after the 2nd, it can process at a high speed rather than equipment conventionally [above-mentioned].

[0063] Moreover, in raising the reliability of data read-out at the sacrifice of a read-out rate, it compares and checks LSA received from the host 1, and LSA in the LSA storing field 12 which is the sector to which the address pointer for read-out points by the degree of step 43. In addition, retrieval processing after step 23 is performed here at the time of an inequality.

[0064] It continues and the flow chart of drawing 13 explains the data write-in actuation to RAM disk equipment 2A. Since steps 30-32 of drawing 13 are the same as steps 20-22 of drawing 11 , explanation is omitted. First, an empty sector is checked based on directions of the address pointer for writing (step 33).

[0065] When there is an empty sector, while writing in data there, address translation table 5A is updated (steps 34-35). That is, PBN which wrote in data is saved in the PBN storing section of corresponding LSA.

[0066] In step 34, when there is no empty sector based on the address pointer for writing, it processes as follows (steps 36-38). First, the block eliminated based on the count of elimination, the number of invalid datas, etc. is determined. Next, an effective data is copied to a free block. Then, the determined block is eliminated and the eliminated count of elimination of a block is updated. And it progresses to step 35.

[0067] In renewal of data, the effective-data check flag 13 of the old data is updated from "FF" to "00." Moreover, the address pointer for writing is updated (step 39). In this example 1, in order to write in data continuously, the address pointer for writing will point to the following sector address after data write-in termination. In addition, when all the sectors within one elimination block are buried with data, the sector address to which CPU4 determines and points to the elimination block which should be written in a degree is decided. For example, when the elimination (it is clean) block of two or more openings exists, CPU4 determines the empty elimination block which should be written in a degree based on the count of elimination etc.

[0068] This example 1 can make small capacity of address translation table 5A by using the object for read-out and the address pointer for writing, and the logic sector-address comparator circuit 61, without dropping performance, such as a read-out rate of the conventional RAM disk card (equipment). By making small capacity of address translation table 5A, large capacity-ization of a RAM disk card can be advanced reasonable. When the conventional address translation table 5 is used, the size of 80KB is needed by 20MB of RAM disk, and the size of 160KB (1.25Mbit) is needed by 40MB of RAM disk. Since a flash memory can be extended and carried in the tooth space in which the cost concerning Volatility RAM could be reduced and the volatile RAM memory for address translation tables was carried if this becomes small in one half of sizes about, the capacity of RAM disk equipment can be increased.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is the block diagram showing the whole example 1 configuration of this invention.
[Drawing 2] It is drawing showing the configuration of the address translation table of the example 1 of this invention.
[Drawing 3] It is drawing showing the configuration of the flash memory of the example 1 of this invention.
[Drawing 4] It is drawing showing the internal configuration of an elimination block of the example 1 of this invention.
[Drawing 5] It is drawing for explaining data write-in actuation of the example 1 of this invention.
[Drawing 6] It is drawing for explaining data write-in actuation of the example 1 of this invention.
[Drawing 7] It is drawing for explaining data write-in actuation of the example 1 of this invention.
[Drawing 8] It is drawing for explaining data write-in actuation of the example 1 of this invention.
[Drawing 9] It is drawing for explaining data read-out actuation of the example 1 of this invention.
[Drawing 10] It is drawing for explaining data read-out actuation of the example 1 of this invention.
[Drawing 11] It is the flow chart which shows data read-out actuation of the example 1 of this invention.

[Drawing 12] It is the flow chart which shows data read-out actuation of the example 1 of this invention.

[Drawing 13] It is the flow chart which shows data write-in actuation of the example 1 of this invention.
[Drawing 14] It is the block diagram showing the conventional RAM disk equipment whole configuration.
[Drawing 15] It is drawing showing the configuration of the address translation table of conventional RAM disk equipment.
[Drawing 16] It is drawing showing the configuration of the flash memory of conventional RAM disk equipment.
[Drawing 17] It is drawing showing the configuration of the elimination block in the flash memory of conventional RAM disk equipment.
[Drawing 18] It is drawing for explaining data read-out actuation of conventional RAM disk equipment.
[Drawing 19] It is drawing for explaining data write-in actuation of conventional RAM disk equipment.
[Drawing 20] It is drawing for explaining data write-in actuation of conventional RAM disk equipment.
[Description of Notations]
1 Host, 2A RAM disk equipment, 3 An interface circuitry, 4 CPU, 5A Address translation table, 6A A flash plate control circuit, 61 A logic sector-address comparator circuit, 7 The sector buffer for data I/O, 8A Flash memory.

[Translation done.]

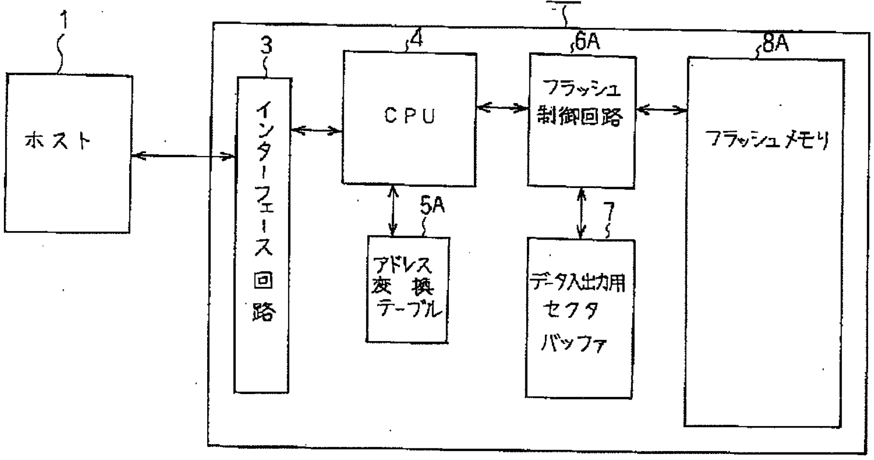
* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

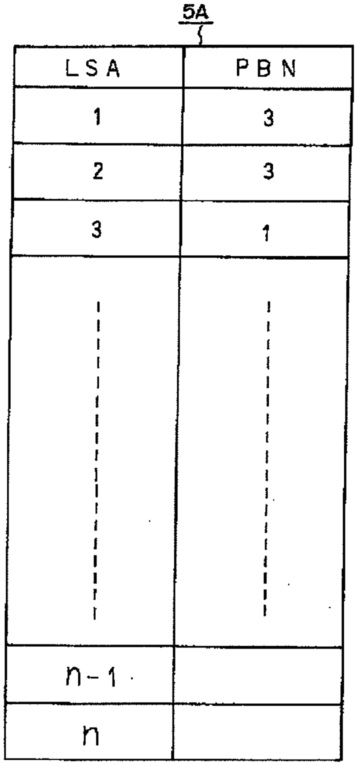
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

[Drawing 1]

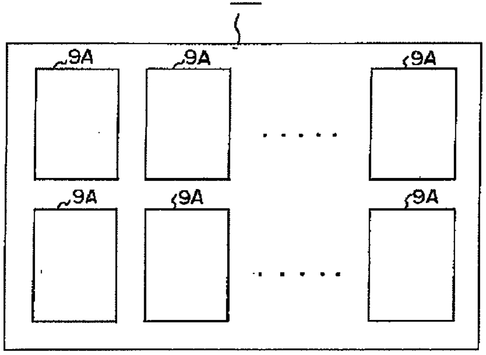

1
ホスト
2A
3
インターフェース回路
4
CPU
5A
アドレス変換テーブル
6A
フラッシュ制御回路
7
データ入出力用セクタバッファ
8A
フラッシュメモリ

[Drawing 2]

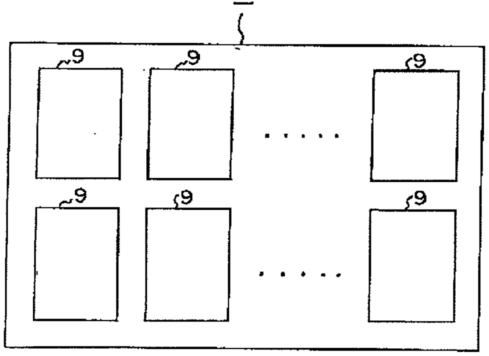


LSA	PBN
1	3
2	3
3	1
⋮	⋮
n-1	
n	

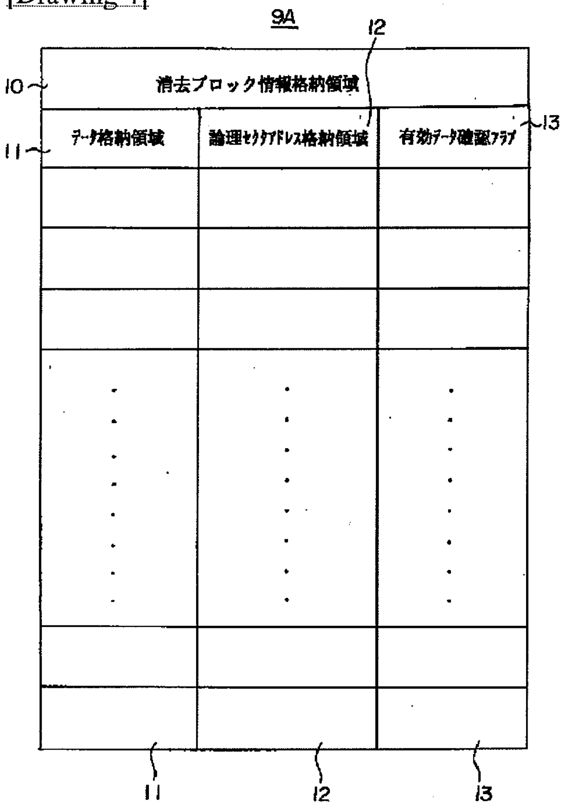
[Drawing 3]


8A
9A
9A
9A
9A
9A
9A

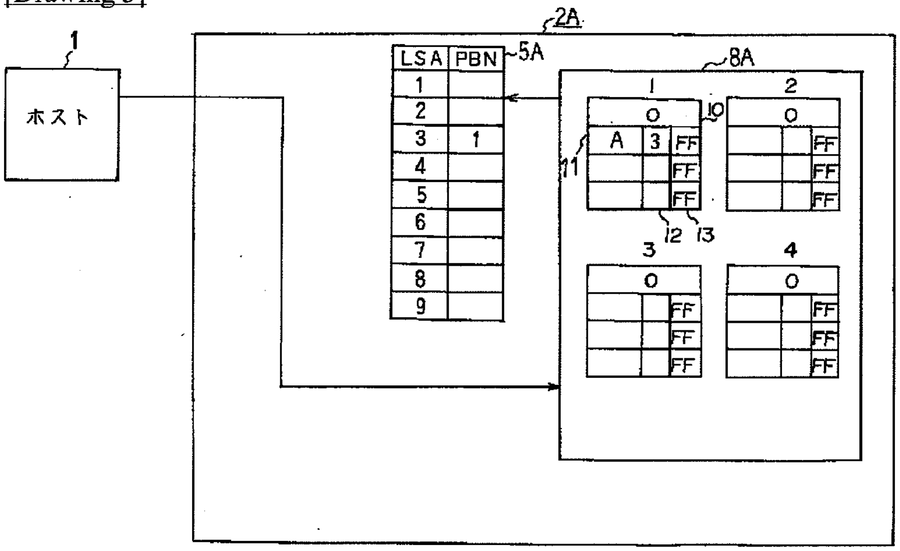
[Drawing 16]



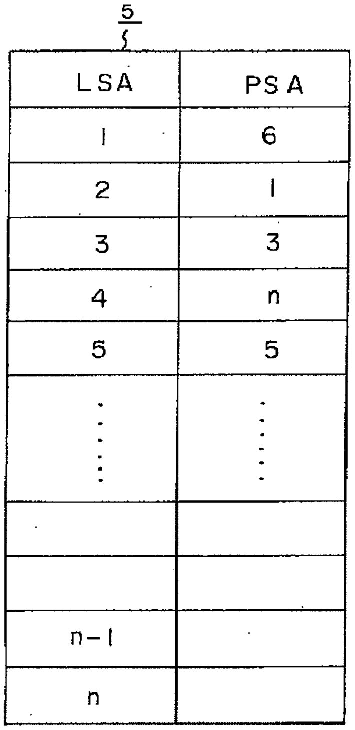
[Drawing 4]


9A
12
10
消去ブロック情報格納領域
11
ﾃﾞｰﾀ格納領域
論理ｾｸﾀｱﾄﾞﾚｽ格納領域
有効ﾃﾞｰﾀ確認ﾌﾗｸﾞ
13
11
12
13

[Drawing 5]

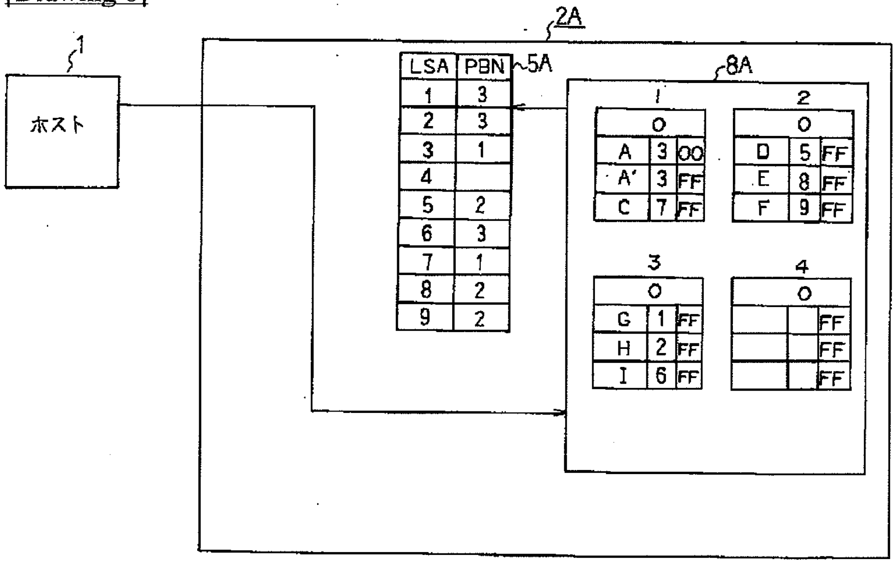

1
ホスト
2A
LSA
PBN
5A
1
2
3
1
4
5
6
7
8
9
8A
1
2
0
0
10
A
3
FF
FF
FF
FF
FF
FF
11
12
13
3
4
0
0
FF
FF
FF
FF
FF
FF

[Drawing 15]

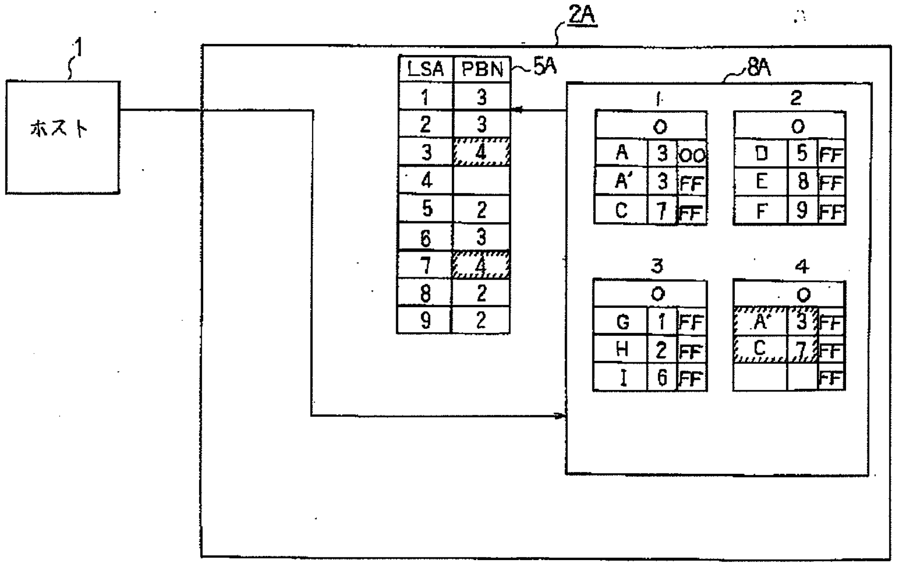


LSA	PSA
1	6
2	1
3	3
4	n
5	5
⋮	⋮
n−1	
n	

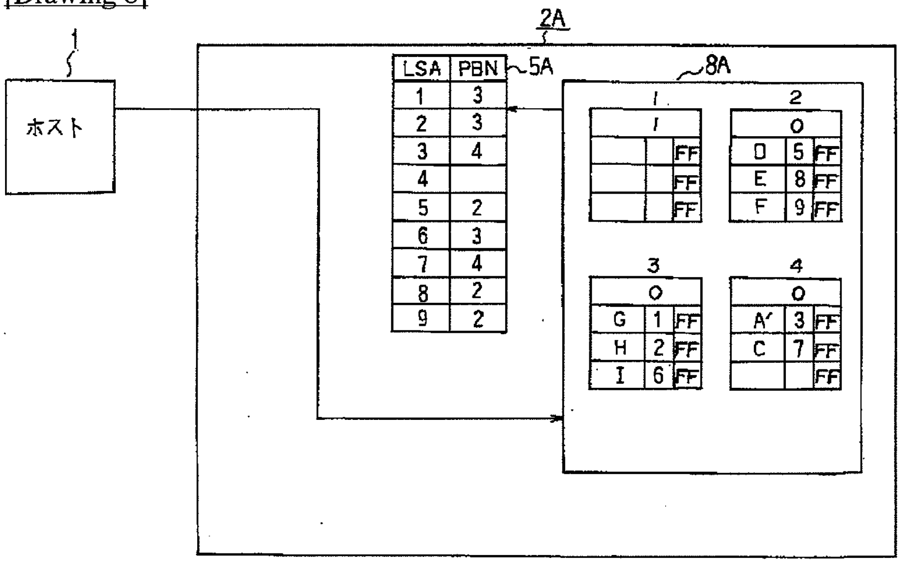
[Drawing 6]


1
ホスト
2A
LSA
PBN
5A
1 3
2 3
3 1
4
5 2
6 3
7 1
8 2
9 2
8A
1
0
A 3 00
A' 3 FF
C 7 FF
2
0
D 5 FF
E 8 FF
F 9 FF
3
0
G 1 FF
H 2 FF
I 6 FF
4
0
FF
FF
FF

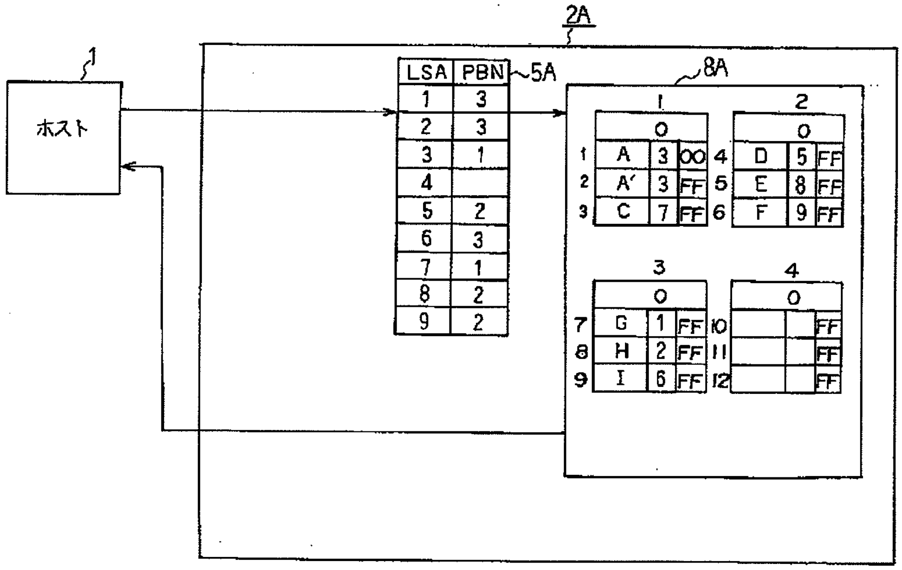
[Drawing 7]


2A
1
ホスト
LSA
PBN
5A
8A
1
2
3
4
0
A 3 00
A' 3 FF
C 7 FF
D 5 FF
E 8 FF
F 9 FF
G 1 FF
H 2 FF
I 6 FF
A' 3 FF
C 7 FF
FF
1 3
2 3
3 4
4
5 2
6 3
7 4
8 2
9 2

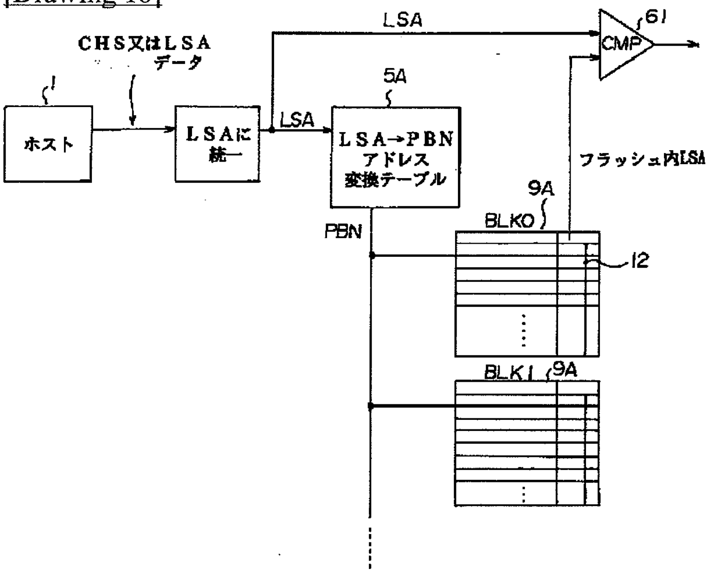
[Drawing 8]


2A
1
ホスト
LSA
PBN
5A
8A
1
2
3
4
1
0
FF
D 5 FF
E 8 FF
F 9 FF
G 1 FF
H 2 FF
I 6 FF
A' 3 FF
C 7 FF
1 3
2 3
3 4
4
5 2
6 3
7 4
8 2
9 2

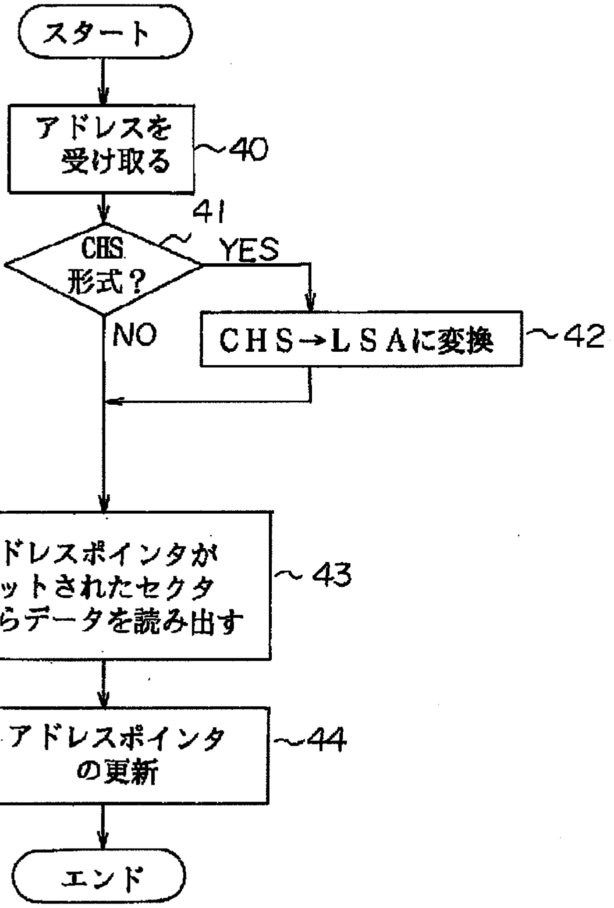
[Drawing 9]


2A
1
ホスト
LSA PBN
5A
1 3
2 3
3 1
4
5 2
6 3
7 1
8 2
9 2
8A
1
0
1 A 3 00
2 A' 3 FF
3 C 7 FF
2
0
4 D 5 FF
5 E 8 FF
6 F 9 FF
3
0
7 G 1 FF
8 H 2 FF
9 I 6 FF
4
0
10 FF
11 FF
12 FF

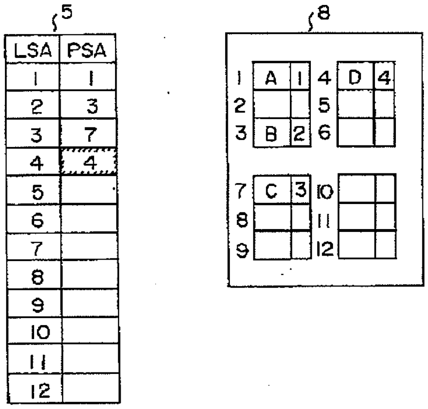
[Drawing 10]


LSA
61
CMP
CHS又はLSA
データ
1
ホスト
LSAに統一
LSA
5A
LSA→PBN
アドレス
変換テーブル
フラッシュ内LSA
PBN
9A
BLK0
12
BLK1
9A

[Drawing 12]

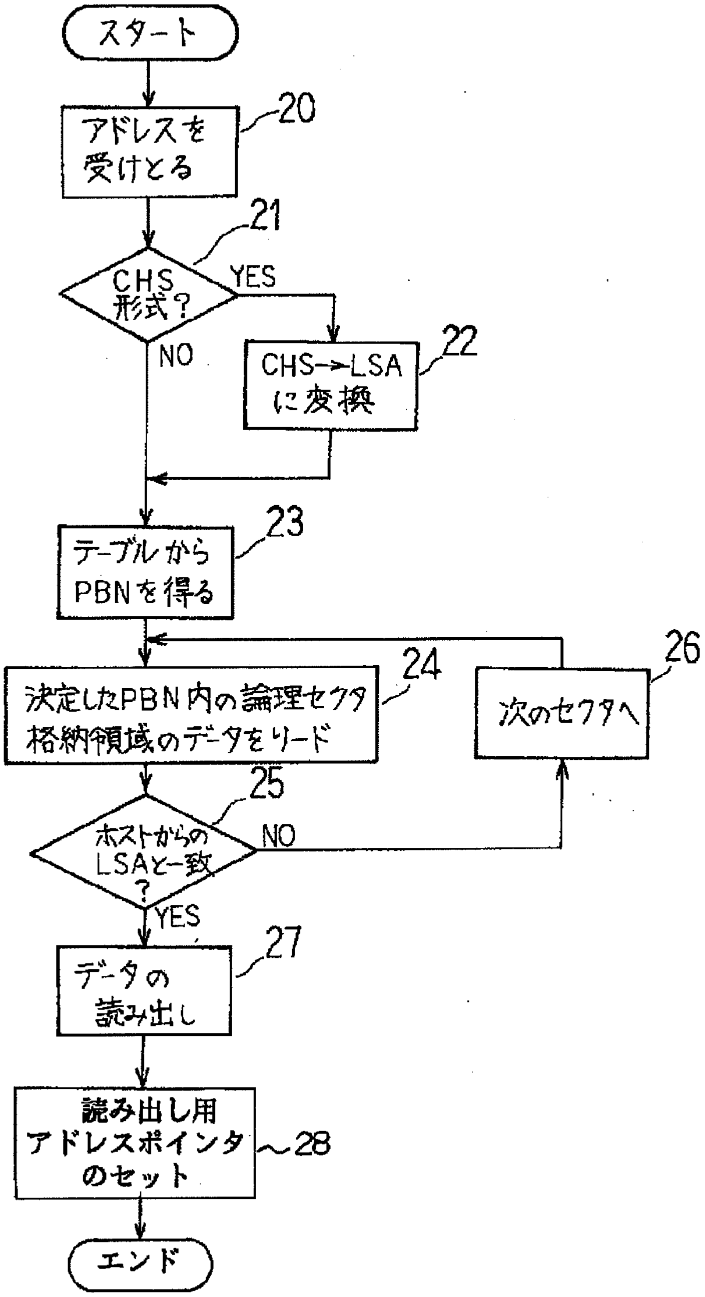

スタート
アドレスを受け取る
40
41
CHS形式？
YES
CHS→LSAに変換
42
NO
アドレスポインタがセットされたセクタからデータを読み出す
43
アドレスポインタの更新
44
エンド

[Drawing 19]

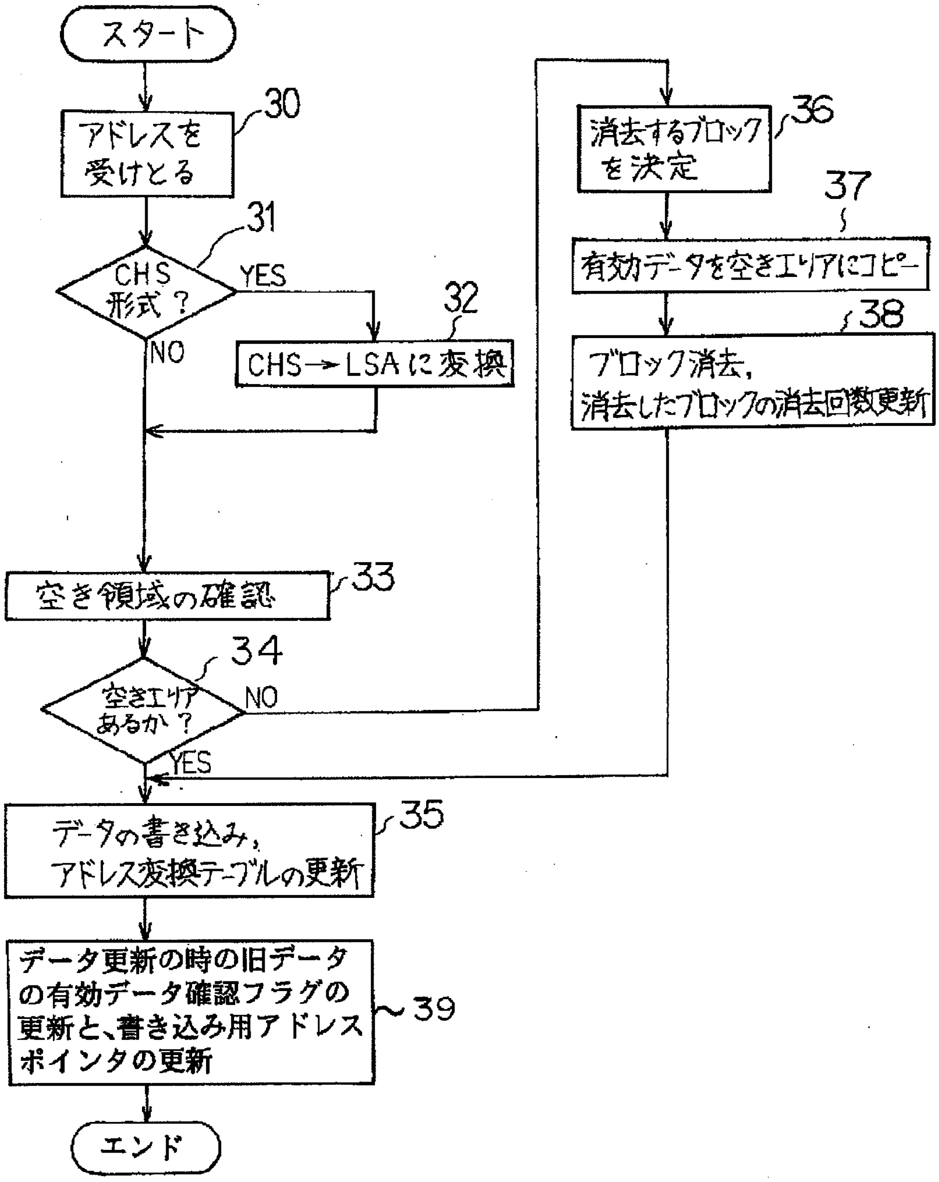

5
8
1
A
1
4
D
4
2
5
3
B
2
6
7
C
3
10
8
11
9
12

LSA	PSA
1	1
2	3
3	7
4	4
5	
6	
7	
8	
9	
10	
11	
12	

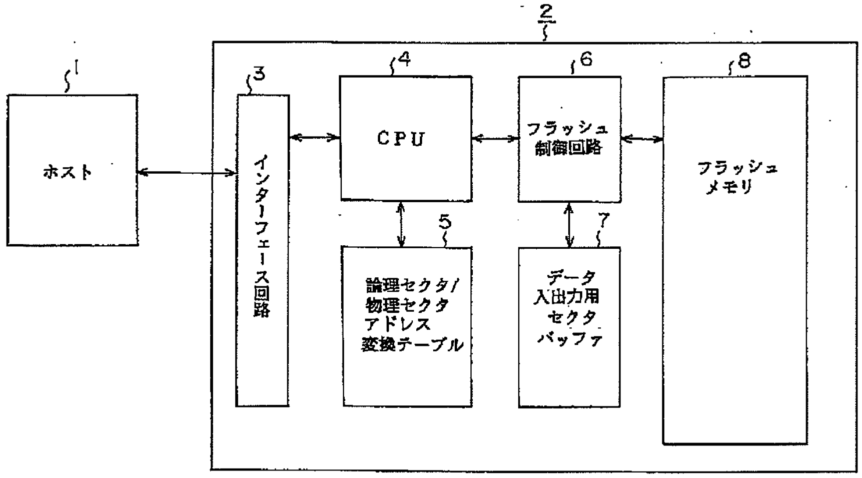
[Drawing 11]


スタート
アドレスを受けとる
20
CHS形式？
21
YES
NO
CHS→LSAに変換
22
テーブルからPBNを得る
23
決定したPBN内の論理セクタ格納領域のデータをリード
24
次のセクタへ
26
ホストからのLSAと一致？
25
NO
YES
データの読み出し
27
読み出し用アドレスポインタのセット
28
エンド

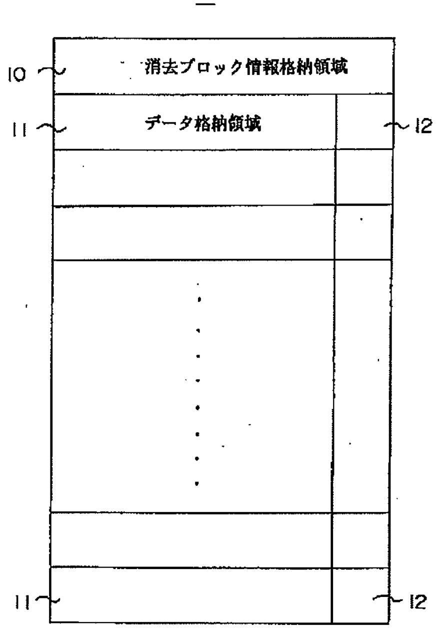
[Drawing 13]


スタート
アドレスを受けとる
30
CHS形式？
31
YES
NO
CHS→LSAに変換
32
空き領域の確認
33
空きエリアあるか？
34
NO
YES
データの書き込み，アドレス変換テーブルの更新
35
消去するブロックを決定
36
有効データを空きエリアにコピー
37
ブロック消去，消去したブロックの消去回数更新
38
データ更新の時の旧データの有効データ確認フラグの更新と、書き込み用アドレスポインタの更新
39
エンド

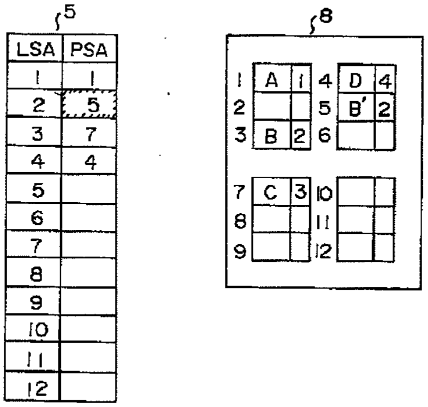
[Drawing 14]


2
1
ホスト
3
インターフェース回路
4
CPU
5
論理セクタ/物理セクタアドレス変換テーブル
6
フラッシュ制御回路
7
データ入出力用セクタバッファ
8
フラッシュメモリ

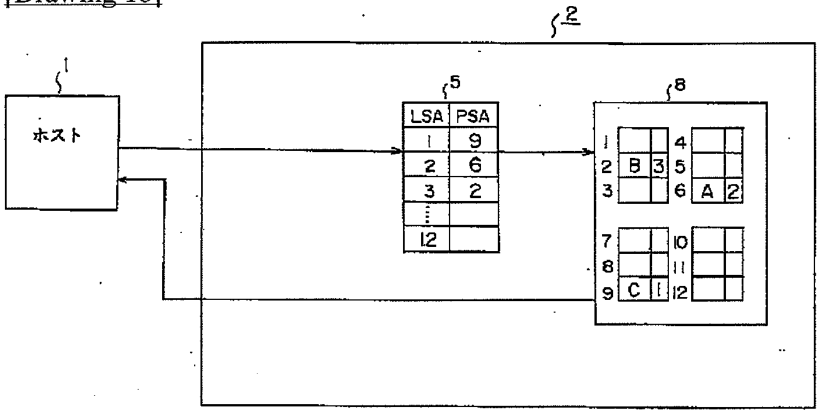
[Drawing 17]


9
10
消去ブロック情報格納領域
11
データ格納領域
12
11
12

[Drawing 20]


5
LSA
PSA
1 1
2 5
3 7
4 4
5
6
7
8
9
10
11
12
8
1 A 1
2
3 B 2
4 D 4
5 B' 2
6
7 C 3
8
9
10
11
12

[Drawing 18]


2
1
ホスト
5
LSA
PSA
1 9
2 6
3 2
⋮
12
8
1
2 B 3
3
4
5
6 A 2
7
8
9 C 1
10
11
12

[Translation done.]

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ^{6} :
G06F 12/12

A1

(11) International Publication Number: **WO 99/21093**

(43) International Publication Date: 29 April 1999 (29.04.99)

(21) International Application Number: PCT/US98/21017

(22) International Filing Date: 5 October 1998 (05.10.98)

(30) Priority Data:
08/951,644 16 October 1997 (16.10.97) US

(71) Applicant *(for all designated States except US)*: M-SYSTEMS FLASH DISK PIONEERS LTD. [IL/IL]; Building 7, Atidim Industrial Park, P.O. Box 58036, 61580 Tel Aviv (IL).

(71) Applicant *(for TJ only)*: FRIEDMAN, Mark, M. [US/IL]; Alharizi 1, 43406 Raanana (IL).

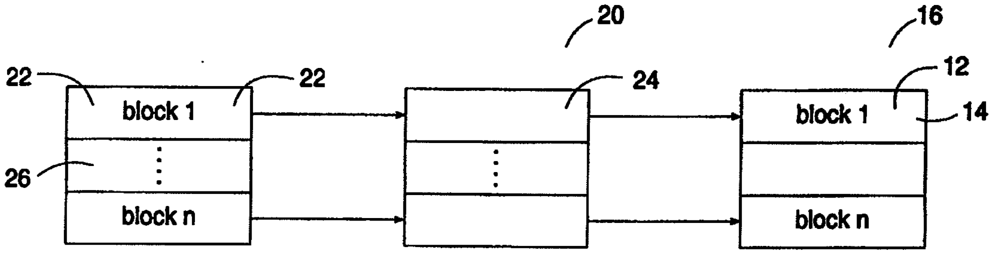
(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: BAN, Amir [IL/IL]; Yabok Street 4, 47205 Ramat Hasharon (IL).

(74) Common Representative: FRIEDMAN, Mark, M.; c/o CASTORINA, Anthony, Suite 207, 2001 Jefferson Davis Highway, Arlington, VA 22202 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, 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 the receipt of amendments.

(54) Title: IMPROVED FLASH FILE SYSTEM



(57) Abstract

A flash memory device, and methods for writing to the device and for reorganizing the device. The flash memory device (20) includes a physical device (10), a virtual device (22) and a virtual map (24) which relates the virtual addresses of the virtual device to the physical addresses of the physical device.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

IMPROVED FLASH FILE SYSTEM

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to a system of handling data storage on flash devices and, in particular, to a system which manages the storage and retrieval of information on page-mode flash devices, enabling them to behave as flash disks.

Flash devices include electrically erasable and programmable read-only memories (EEPROMs) made of flash-type, floating-gate transistors and are non-volatile memories similar in functionality and performance to EPROM memories, with an additional functionality that allows an in-circuit, programmable, operation to erase pages of the memory. Flash devices have the advantage of being relatively inexpensive and requiring relatively little power as compared to traditional magnetic storage disks. However, in a flash device, it is not practical to rewrite a previously written area of the memory without a preceding page erase of the area. This limitation of flash devices causes them to be incompatible with typical existing operating system programs, since data cannot be written to an area of memory within the flash device in which data has previously been written, unless the area is first erased.

Software products have been proposed in the prior art to allow a flash device to be managed by existing computer operating programs without modification of the operating system program. However, these prior art programs all have deficiencies. For example, one program operates the flash memory as a "write once read many" device. This prior art software product cannot recycle previously written memory locations. When all locations are eventually written the memory cannot be further used without specific user intervention. Other prior art programs, such as those proposed by SanDisk, erase and rewrite an entire memory page every time new data is to be written to the page. This system has the disadvantage of requiring multiple erase cycles, which are both relative slow and inefficient and which lead to a more rapid degradation of the physical media itself.

To overcome these deficiencies of the prior art, a flash File System (FFS) was disclosed in U.S. Patent No. 5,404,485, herein incorporated by reference. FFS provided a system of data storage and manipulation on flash devices which allowed these devices to emulate magnetic, disk-based data storage. As noted above, the relatively inexpensive cost and low power consumption of flash devices makes them a favorable choice for data storage, particularly for laptop, portable computers. FFS enhances the ability of flash devices to act as substitutes for magnetic disk storage. Indeed, FFS as disclosed in U.S. Patent No. 5,404,485 has proven to be so useful that the data layout specification was adopted by the PCMCIA [Personal Computer Memory Card International Association] and JEIDA [Japan Electronic Industry Development Association] committees as a standard called Flash Translation Layer (FTL).

FFS essentially describes a virtual mapping system for flash EEPROM devices. The virtual map is a table which relates the physical address of a read/write block within the flash device to the virtual address of that block. Since each of these blocks is relatively small, 512 bytes, the size of the virtual map itself is quite large. FFS also includes a method of storing and maintaining the bulk of the virtual map on a flash EEPROM device, minimizing the amount of other memory required for storage of the virtual map.

As noted above, FFS has proven particularly successful for transforming flash devices into emulators of magnetic disk storage, so much so that it has been adopted as an industry standard. However, FFS cannot fulfill all of the requirements of the newer flash device technologies. In particular, FFS is not as successful with the NAND and AND flash technologies.

Another example of an attempt to overcome certain deficiencies of prior art flash memory architectures, and in particular those of erase-before-write systems, is disclosed in U.S. Patent No. 5,479,638. In the system of U.S. Patent No. 5,479,638, the physical location of a particular read/write block is shifted if further programming to a written block is required. However, this system has the disadvantage of only being operable with those flash devices which can erase a single 512 byte read/write block at a time. Since such a requirement is implemented at the hardware level, this system also cannot be used with the newer NAND and AND flash technologies.

NAND and AND differ from older flash device technologies in a number of respects. First, the erasable unit size is smaller for NAND and AND, around 8 KB, as opposed to 64 KB for older flash devices. Second, the erase time is considerably faster for NAND and AND, even when measured as time required to erase a single byte. Third, the flash memory is divided into pages for NAND and AND which are 256 or 512 bytes in length, which is a fixed characteristic of the hardware devices themselves. It should be noted that the term "page" as used herein is roughly equivalent to the term "block" as used for older flash technologies, although the particular characteristics of a "page" and of a "block" differ somewhat. These features have a number of implications for the operation of flash devices based upon NAND and AND technologies.

First, page-mode memory has a fixed overhead for writing a page or any part of it. By contrast, the overhead for the writing operation in previous flash technologies was proportional to the number of bytes written. Second, the flash memory in NAND and AND is configured so that each page has several spare bytes which are specially addressable. These spare bytes are convenient locations for the storage of information related to the flash memory system. Finally, there is a limitation on the number of times a page may be written before it is erased. This limitation is relatively low, 8 or 10 times, after which further writing without prior erasing is unreliable. Thus, page-mode memory has both significant advantages and new challenges for successful data storage and retrieval.

Unfortunately, as noted above, the currently available prior art data handling system, FFS, has significant disadvantages for the operation of flash memory in page-mode. In particular, FFS demonstrates non-optimized performance on page-mode flash technologies such as NAND and AND because of the restrictions imposed by page-mode programming. Furthermore, the system disclosed in U.S. Patent No. 5,479,638 also cannot be used with such flash technologies, due to its requirement for a block-by-block erase operation.

There is therefore a need for, and it would be greatly advantageous to have, a system for handling data storage on a NAND or AND flash device which is optimized for performance on page-mode flash technologies, yet which is still usable on older, non-page mode flash devices.

SUMMARY OF THE INVENTION

The present invention provides a memory organization method for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, and in which the size of the memory portion for reading or writing data differs from the size of the smallest memory portion for erasing, the method including the steps of: (a) providing a plurality of physical units of the memory, each of the physical units being the smallest memory portion for erasing, each of the physical units being designated by a physical unit number and each of the physical units being divided into a plurality of physical blocks, each of the plurality of physical blocks being the memory portion for reading or writing data and each of the physical blocks being designated by a physical block offset within the physical unit; (b) providing a plurality of virtual units of the memory, each virtual unit being designated by a virtual unit number and each of the virtual units featuring a plurality of virtual blocks; each of the virtual blocks being designated by a virtual block offset within the virtual unit; (c) providing a virtual map for mapping each virtual unit to at least one physical unit; and (d) mapping each virtual block within the virtual unit to one physical block within the at least one physical unit.

Preferably, the method further includes the steps of: (e) receiving a write command to write data at a virtual block; (f) locating a virtual unit containing the virtual block; (g) locating a writable block within a physical unit mapped to the virtual unit; and (h) writing the data to the writable physical block. More preferably, the method further includes the steps of: (I) if an unwritten physical block in a written physical unit cannot be located, locating an unwritten physical unit; (j) writing the data to a writable physical block of the unwritten physical unit; and (k) updating the virtual map by additionally mapping the virtual unit to the unwritten physical unit, such that the virtual unit corresponds to the unwritten physical unit and to the written physical unit, the unwritten physical unit and the written physical unit forming a chain of physical units. Most preferably, the writable physical block of the unwritten physical unit has a physical block offset, and the physical block offset corresponds to the virtual block offset of the mapped virtual unit. Also most preferably, the method further includes the steps of: (l) if an unwritten physical block in a written physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain; (m) locating the last physical unit in the chain; (n) moving data from each of the physical blocks of the written physical unit to a writable physical block of the last physical unit, the writable physical block having substantially the same block offset as the physical block of the written physical unit; and (o) updating the virtual map by mapping the virtual unit to the last physical unit, such that the virtual unit corresponds substantially only to the last physical unit. Preferably, the method also further includes the step of: (p) erasing substantially all of the written physical units in the chain, with the exception of the last physical unit.

Alternatively and preferably, the method further includes the steps of: (l) if no unwritten physical unit is available for allocation for the chain, allocating an unwritten physical unit for reorganization; (m) moving data from each of the physical blocks of the written physical unit to a writable physical block of the unwritten physical unit; and (n) updating the virtual map by mapping the virtual unit to the unwritten physical unit, such that the virtual unit corresponds substantially only to the unwritten physical unit. More preferably, the method further includes the step of: (o) erasing all of the written physical units.

According to other preferred embodiments of the present invention, the method further includes the steps of: (i) if an unwritten physical block in a written physical unit cannot be located, allocating an unwritten physical unit to form a chain of physical units, such that the unwritten physical unit is a last physical unit in the chain; (j) writing the data to an unwritten physical block in the last physical unit; (k) moving data from each of the physical blocks to a writable physical block of the unwritten physical unit, except data written in step (j); and (m) updating the virtual map by mapping the virtual unit to the written physical unit, such that the virtual unit corresponds to the written physical unit.

According to another embodiment of the present invention, there is provided a method of writing data for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, the method including the steps of: (a) providing a plurality of physical units being divided into a plurality of physical blocks, each of the physical units having a physical unit number and each of the physical blocks having a physical block offset within the physical unit; (b) providing a plurality of virtual units being divided into a plurality of virtual blocks, each of the virtual units having a virtual unit number and each of the virtual blocks having a virtual block offset within the virtual unit, each virtual unit being mapped to at least one physical unit; (c) receiving a write command to write data at a virtual block; (d) determining a virtual unit containing the virtual block having a virtual block offset; (e) locating a physical unit corresponding to the virtual unit; (f) locating a physical block within the physical unit; (g) determining if the physical block is unwritten; (h) writing the data to the physical block only if the physical block is unwritten; (i) alternatively, if the physical block is not unwritten, allocating an unwritten physical unit; (j) locating a writable physical block within the unwritten physical unit, the writable physical block having a physical block offset; (k) writing the data to the writable physical block; and (l) additionally mapping the virtual unit to the unwritten physical unit containing the writable physical block, such that the virtual unit is additionally mapped to the unwritten physical unit to form a chain of physical units. Preferably, the physical block offset of the writable physical block has an identical block offset number as the virtual block offset.

Preferably, the method further includes the steps of: (m) if an unwritten physical block in a written physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain; (n) locating the last physical unit in the chain; (o) transferring all data within the physical blocks of the written physical unit to the physical blocks of the last physical unit; and (p) updating the virtual map such that the virtual unit corresponds only to the last physical unit. Most preferably, the method further includes the step of: (q) erasing all of the written physical units.

Alternatively and preferably, the method further includes the steps of: (l) if no unwritten physical unit is available for allocation, locating a last physical unit in the chain; (m) transferring all data within the physical blocks of the written physical units to the physical blocks of the last physical unit; and (n) updating the virtual map such that the virtual unit corresponds only to the last physical unit. More preferably, the method further includes the step of: (o) erasing substantially all of the written physical units, with the exception of the last physical unit.

DESCRIPTION OF THE INVENTION

The present invention is of a method of organizing a flash memory in which the size of the memory portion for reading or writing data, such as a block, differs from the size of the smallest portion for erasing, such as a unit. Examples of types of flash memory which can be organized according to the method of the present invention include, but are not limited to, page-mode devices exemplified by the NAND and AND technologies. Methods are also provided of reading and writing data to the flash memory, and of reorganizing the flash memory when no more unwritten physical units are available.

Hereinafter, the term "physical unit" is defined as a unit on the physical media or hardware of the memory which is the smallest portion of the memory which can be erased, or an integral multiple thereof. It is a portion of the memory which is contiguous, fixed in size and erasable. The term "physical block" is defined as being the portion of the memory for reading or writing data. Hereinafter, the term "virtual unit" is defined as the same size as the physical unit. For page-mode memory technologies such as NAND and AND, the smallest portion of the memory which can be erased is larger than the page size, typically about 8 KB. As used herein, the term "physical block" is equivalent to the term "page" for page-mode memory technologies. Thus, virtual units are as large as physical units.

Hereinafter, the term "virtual map" refers to a table which relates a virtual unit to at least one corresponding physical unit. As noted previously, each unit, virtual or physical, is composed of a plurality of blocks. The exact location of a block within a unit is determined according to one or more preset rules, as further described below.

Each physical unit is designated by a physical unit number. The location of each physical block is given by a physical block offset. Similarly, each virtual unit is designated by a virtual unit number. The location of each virtual block is given by a virtual block offset. It should be noted that each virtual unit number can correspond to one or more physical unit numbers. Thus, the mapping between virtual units and physical units can either be one-to-one or one-to-many.

Hereinafter, the term "writing data" describes the act of storing data on the flash memory. The term "reading data" describes the act of retrieving data from the flash memory. Hereinafter, the term "unwritten" indicates some portion of the memory, such as a physical block, which is capable of having data written to it. Thus, the term "unwritten" includes, but is not limited to, a portion of the memory which has just been erased.

In a computer or other electronic device having a flash memory organized according to the present invention, the operating system of that device interacts with the virtual units and virtual blocks for reading and writing data. The virtual media, which includes the virtual units and blocks, thus acts as an interface for the operating system to interact with the flash memory device. For example, the operating system issues a write command to write data to a virtual block at a virtual block offset. The virtual unit containing the virtual block is then located. The virtual map then locates a corresponding physical block within a physical unit of the memory, where the data are actually stored. Although the operating system issues read and write commands as though the virtual units and virtual blocks are the actual hardware of the flash memory, in reality the actual hardware is incorporated in the physical units and physical blocks of the flash memory. Thus, the operating system is only aware of the virtual units and blocks, and does not directly interact with the hardware itself.

The advantage of such an interface is that the inherent disadvantages of the flash memory, such as the requirement for an erase before further writing can occur, are overcome by the interactions of the operating system with the virtual memory. Additionally, the operating system of the electronic device does not have to organize the addresses of the flash memory. Furthermore, the operating system can interact with a variety of different flash memory technologies without requiring extensive modifications, since one interface can be used with multiple types of flash memory devices. Thus, the methods of the present invention permit the greatest flexibility for flash memory devices and the electronic devices which use them.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

FIG. 1 is a schematic diagram of a physical flash memory device according to the present invention;

FIG. 2 is a diagram of a basic system of organizing the flash memory device according to the present invention;

FIGS. 3A and 3B show AND and FMAX systems, respectively, according to the present invention;

FIGS. 4A and 4B show writing algorithms for the AND and FMAX systems, respectively, according to the present invention; and

FIGS. 5A and 5B show reorganization algorithms for the AND and FMAX systems, respectively, according to the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention provides a system for organizing a page-mode flash memory device, such as a NAND or AND flash device. This system includes methods for reading from, and writing to, such flash devices. Furthermore, the system also provides an interface which enables the flash device to successfully emulate magnetic disk storage. Such flash memory devices can be installed in a variety of host devices, such as a personal or laptop computer.

While this invention will be described in the context of a flash memory, those skilled in the art will understand that its teachings are also applicable to data storage devices with the same write, read, and unit erase characteristics as flash memories.

The principles and operation of a system for organizing a page-mode flash memory device according to the present invention may be better understood with reference to the drawings and the accompanying description.

Referring now to the drawings, Figure 1 schematically illustrates a prior art physical page-mode flash memory device according to a technology such as NAND, for example. The term "physical device" is hereinafter defined as the actual hardware itself which contains the physical media for the flash memory device. Such physical media is typically composed of flash EEPROM units, although any suitable non-volatile, programmable memory device could be substituted. The term "programmable" is hereinafter defined as alterable, such as by having data written to the memory device, for example.

A flash memory physical device **10** is shown, with at least one physical unit **12**. Physical unit **12** is the smallest segment of physical device **10** which can be erased. Physical unit **12** includes an integral number of blocks, individually designated as blocks 1 to *n*, where *n* is an integer, and collectively designated as block **14**. Block **14** is composed of a contiguous, fixed-length group of physical byte addresses and is a feature of the hardware.. Specifically, the size of block **14** is a characteristic of physical device **10**. User data can be stored in a primary data area **16** of block **14**. Each block **14** also has a control data area **18**. Control data area **18** is addressable by a separate algorithm from the main portion of block **14** and is not included in calculations of the size of block **14**. As further described below, control data area **18** is convenient for the storage of information related to the flash filing system itself. Each physical unit **12** is either an assigned unit or an unassigned unit. Each unassigned unit is free, contains no useful data and is ready to be allocated and assigned. Each assigned unit has been allocated and contains data.

Figure 2 illustrates a system for organizing a basic flash memory device. A system **20** controls both a virtual media **22** and physical device **10**, and relates virtual media **22** to physical device **10** through a virtual map **24**. Virtual media **22** includes a plurality of virtual units **26**. Each virtual unit **26** has a plurality of virtual blocks **28**. Each virtual unit **26** is designated by a virtual address. The virtual address includes a virtual unit number, designating a particular virtual unit **26**. Each virtual block **28** is designated by a virtual block offset. Similarly, each physical unit **12** has a physical address. The physical address includes a physical unit number, designating a particular physical unit **12**. Each physical block **14** has a physical block offset.

Virtual map **24** maps virtual unit **26**, which is an assigned virtual unit, to at least one physical unit **12**, which is an assigned physical unit. As for the physical units, if a virtual unit has been mapped to at least one physical unit, it is an assigned virtual unit. However, one virtual unit **26** can be mapped to more than one physical unit **12**. Thus, the correspondence between virtual unit **26** and physical unit **12** can be either one-to-one or one-to-many.

System **20** operates as follows. The operating system of an electronic device which contains a flash memory device (not shown) sends a command, such as a read command or a write command, to a particular virtual block **28** within a particular virtual unit **26**. Virtual map **24** then locates the corresponding physical block **14** within a physical unit **12**.

If the correspondence of the mapping is one-to-one as shown, with each virtual unit **26** being mapped to exactly one physical unit **12**, then the situation would appear to be relatively simple. However, as noted above, physical device **10** has particular physical constraints for reading and writing data, including the requirement for performing periodic erasures before additional data can be written. There are two possible solutions to these constraints which do not involve frequent, repeated erasures of physical device **10** or a portion thereof.

The first solution is shown in Figure 3A, in which the correspondence between virtual unit **26** and physical unit **12** is one-to-many, so that each virtual unit **26** corresponds to a plurality of physical units **12**. Virtual map **24** must hold the information required to perform such a mapping. An example of a portion of virtual map **24** is given in Figure 3A, and is usable with the AND system of the present invention.

The portion of virtual map **24** shows physical blocks **14** within physical units **12**, and virtual blocks **28** within virtual units **26**. In this example, one particular virtual unit **30** corresponds to two physical units **12**. The first physical unit **12** is a primary unit **32**. The second physical unit **12** is a replacement unit **34**. For each virtual unit **26** there can only be one primary unit **32**. However, there can be zero or more replacement units **34** associated with each virtual unit **26**. As an example, a virtual unit **36** corresponds only to a primary unit **38** and no replacement units **34**, so virtual unit **36** is an example of a non-replaced virtual unit.

The organization of virtual blocks **28** will depend upon the number of physical block(s) **14** corresponding to a particular virtual unit **26**. For virtual unit **30**, some virtual blocks **28** correspond to physical blocks **14** within primary unit **32**, while other virtual blocks **28** correspond to physical blocks **14** within replacement unit **34**. For virtual unit **36**, substantially all virtual blocks **28** correspond to physical blocks **14** within primary unit **38**.

In the simplest case, in which the virtual unit is a non-replaced unit, the procedure for locating a particular physical block **14** is as follows. A virtual unit **36** has a virtual unit number **44** which designates virtual unit **36**, and a virtual block offset **46** which designates virtual block **42**. Note that virtual block offset **46** is also a number. Physical unit number **50** designates primary unit **38**. A physical block offset **52** designates a physical block **54** within primary unit **38**. To locate physical block **54** for the purposes of reading or writing data, the first rule is to divide the desired virtual block offset **46** by the number of blocks per virtual unit to determine virtual unit number **44**. Virtual map **24** then maps virtual unit number **44** to a physical unit number **50**. The second rule is that the desired physical block **14**, in this case physical block **54**, can be located within physical unit **38** according to physical block offset **52** which must be the same number as virtual block offset **46**. Thus, virtual map **24** only contains information about virtual and physical units, but rules are used to determine the proper block offsets.

In the more complex case, each virtual unit corresponds to more than one physical unit. In this case, the group of two or more physical units is called a "chain". For example, a virtual unit number **72** designates virtual unit **30**, and a virtual block offset **74** designates virtual block **70**. A physical unit number **78** designates replacement unit **34** and a physical block offset **80** designates a physical block **82** within replacement unit **34**. Thus, virtual block **70** of virtual unit **30** corresponds to physical block **82** of replacement unit **34**.

To locate physical block **82** for the purposes of reading or writing data, again the first rule is to divide the desired virtual block offset **74** by the number of blocks per virtual unit to determine virtual unit number **72**. Virtual map **24** then maps virtual unit number **72** to physical unit number **78**. However, there is a problem. As stated previously, the second rule is that the desired physical block is located within the physical unit according to the physical block offset, which must be the same number as the virtual block offset. In this case, there are a plurality of physical blocks **14** in the chain. In order to determine which physical block **14** has the data, the third rule is that each physical block **14** having the same block offset as virtual block **70** is examined in each physical unit of the chain. The last non-free physical block **14**, in this case physical block **82** of replacement unit **34**, contains the desired data for reading. Conversely, for writing data, the first free physical block **14** is the desired block.

Since physical blocks are written in the order of the physical unit in the chain to which they belong, the term "last non-free physical block" refers to the physical block in a unit which is farthest down the chain but which is still non-free. Either there are no more units in the chain, or the physical block having the same block offset in the next unit in the chain is free. Similarly, to find the first free physical block, each physical block having a desired block offset is examined in each physical unit in the chain, starting with the primary unit and continuing down through each replacement unit in turn, until a free block is found.

By contrast, FMAX, although it uses a similar virtual map and addressing system, has only one replacement unit for each primary unit, as shown in Figure 3B. To accomplish this, FMAX uses simple and compound replacement (physical) units. A simple replacement unit is a unit in which substantially all of the physical block offsets of the physical unit correlate directly to the virtual block offsets of the corresponding virtual unit. A compound replacement unit is a unit in which such a direct correspondence between virtual block offsets and physical block offsets does not necessarily exist. Instead, if the physical block having the corresponding physical block offset is not available for writing, a different physical block is chosen. Control information is then written to the control data area in order to determine the actual correspondence between a virtual block and a physical block.

As shown in Figure 3B, primary unit **97** has a simple replacement unit **98**, which has a plurality of physical blocks **100**, each of which corresponds to a virtual block **102** in a virtual unit **104**. Each physical block offset corresponds to a virtual block offset which is the identical offset number.

However, if the physical block having the needed physical block offset is not available, then a different physical block in the same physical unit must be written and the replacement unit becomes a compound replacement unit. A second primary unit **109** has a compound physical unit **110**, which also has a plurality of physical blocks **112**, each of which corresponds to a virtual block **114** in a virtual unit **116**. However, one physical block offset could correspond to a virtual block offset which is the identical offset number, while a second physical block offset might correspond to a second virtual block offset which is not the identical offset number. To find a particular physical block, the control information written to the control data area must be examined. As further described below, this has significant consequences for both writing data and reorganizing the FMAX system when necessary.

Figure 4A shows a flow-chart for manipulating the virtual map of Figure 3A, while Figure 4B shows a flow-chart for manipulating the virtual map of Figure 3B. In the simplest case, where all replacement units are either simple units, or primary units with only one replacement unit, the same steps can be used for both AND and FMAX. First, the virtual unit number and the virtual block offset is calculated by dividing the number of the virtual block to be located into the number of blocks per virtual unit, giving the virtual unit number. The modulo, or division remainder, is the virtual block offset.

Next, the virtual map is examined to find the physical unit which corresponds to the virtual unit. If no physical unit can be found which corresponds to the virtual unit, then the required portion of physical memory does not exist on the flash device. As noted above, this simple scheme is only operative if all replacement units are simple units, or the primary units have only one replacement unit. However, this scheme does not work if the physical block to which data is to be written has already been programmed, or written, with other data. In this case, a replacement scheme is required which can handle the task of finding another physical block to which the data can be written.

Two different algorithms are illustrated in Figure 4A (ANAND) and Figure 4B (FMAX). Both algorithms start in the same manner. In step 1, the desired physical unit is located. In step 2, the physical block corresponding to the specified block offset is located within that physical unit. In step 3, if the block is unwritten, the data is written to the block. If the desired physical block is not available, then the two systems of the present invention, AND and FMAX, diverge in the way that each technology handles the situation where the desired physical block has already been written.

As shown in Figure 4A, the AND system will handle this situation by looking at the replacement unit(s). In step 4, the x^{th} replacement physical unit is examined, where x is an integer initially equal to 1. If that physical unit has an unwritten physical block with the desired physical block offset, the data is written to the physical block. If the block is not available, then as shown in step 5, x is incremented by one, and step 4 is repeated. Steps 4 and 5 are repeated until the data are either written to a block or no other replacement units in the chain are found. In step 6, an unassigned physical unit is assigned as a replacement unit, and the data are written to the block with the desired block offset.

The FMAX system handles this situation differently, as shown in Figure 4B. In step 4, a physical block with the same physical block offset in a replacement unit is located. If that physical block is unwritten, the data are written to that physical block. Otherwise, a physical block with a different physical block offset in the replacement unit is located, as in step 5. Step 5 is repeated until an unwritten physical block is located. The replacement unit is now a compound unit, since the virtual block offsets are no longer the same as the physical block offsets. In step 6, control information is appended to the control data area of the physical unit to enable the mapping scheme to find the correct location of any physical block within the compound unit.

However, even these replacement algorithms may not be sufficient to handle all of the different needs of flash devices. Both the AND and FMAX systems will eventually reach a situation where further data cannot be written to a block within a physical unit because such a physical block is not available.

In this situation, a virtual unit must be reorganized to restructure the data into its simplest state, which is a non-replaced primary unit. During this reorganization process, physical replacement units which previously belonged to the virtual unit representation are freed, thereby becoming unassigned or free physical units. For both AND replacement units, and simple FMAX replacement units, this reorganization process is called folding and is diagrammed in Figure 5A below.

Folding requires physical blocks to be written at the same physical block offset in the replacement unit as they would have been written in the primary unit, for reasons which will become more clear as the process is described. In the first step of folding, the last physical unit in the chain is identified, physical unit *x*, where *x* is an integer ranging from 1 to some predetermined implementation-dependent limit. Note that where *x* equals 1, the replacement unit is actually the primary unit and the rest of the algorithm is not performed. Also note that for FMAX, *x* equals 1 or 2.

In step 2, block *n* of unit *x* is examined, where *n* is an integer. If data are written to block *n*, *n* is incremented by 1. If not, then in step 3 *x* is decremented by 1. Steps 2 and 3 are repeated until either *x* is equal to 0 or a written block *n* is found. If a written block *n* is found, then the data are moved to block *n* of the last replacement unit in the chain, in step 4. Steps 2-4 are repeated until all data has been transferred to the last replacement unit, which then becomes the primary unit. All other units in the chain, if any, including the previous primary unit, are then freed and are available for allocation. The virtual map is also updated to reflect the fact that the virtual unit now corresponds to one physical unit.

Unfortunately, folding does not work for compound FMAX replacement units, since blocks within the replacement unit do not always have a physical block offset which is equal to the virtual block offset. A different process of reallocation is shown for such compound physical units in Figure 5B. In step 1, a new, unallocated physical unit is designated as the new primary physical unit. In step 2, block *n* of the compound physical unit is examined. If data are written to block *n* of the compound physical unit, the data are copied to the new primary unit in step 3. If not, data from block *n* of the old primary unit are written to the new primary unit. In step 4, *n* is incremented by 1. Steps 2-4 are repeated until all blocks have been copied. Once all of the blocks have been copied, the previous replacement unit, as well as the old primary unit, are freed and are then available for allocation. As in the previous procedure, the virtual map is updated to reflect the fact that the virtual unit now corresponds to only one physical unit.

A highly simplified embodiment of the reorganization scheme is also possible. In this simplified embodiment, the process of reorganization occurs immediately after a replacement unit is allocated. The replacement unit is therefore only a transient feature of the system, and in the quiescent state, in which the physical memory is not undergoing the process of writing, data exists exclusively in primary non-replaced units. The replacement unit exists only for the process of writing. At the end of the process, all of the information is transferred to a new unit, so that the replacement unit effectively disappears. This method has the advantage of simplicity of implementation and of control structures which are required to manage it. However, its disadvantage is that it is not efficient, reducing the write performance of the system.

All of the methods included in the present invention must be able to record control information on the physical flash device itself, in order to describe the state of the stored data. In particular, unit and block control information are preferably stored, although alternatively such data can be reconstructed from other types of data. Unit control information describes the physical unit number assigned to the physical unit, the status of the physical unit itself as a primary or replacement unit, and its position relative to other units. Block control information describes whether the physical block is occupied, freed or superseded by information residing in a different physical block.

One or both of these different types of information may be recorded in a special portion of the physical device. As noted above in Figure 1, preferably AND and FMAX systems divide each physical unit **12** into primary data areas **16**, containing the actual user data recorded on the physical flash device, and control data areas **18**, containing the control information. Although such areas are shown as subdivisions of block **16**, physical unit **12** could also be divided into primary data areas and control data areas which are substantially independent of divisions into blocks. It should be noted that control data areas **18** are not included within the block location scheme of primary data areas **16** and are also not included when calculating the overall size of the physical flash disk.

Since NAND and AND flash technologies have spare areas for each block of the memory, the control information is usually recorded in the spare area of the block, and the user data are located in the primary block area.

For those flash technologies in which no spare area is provided, every physical unit can be divided into a main area for storing user data, and an overhead section, for storing the required control information.

It will be appreciated that the above descriptions are intended only to serve as examples, and that many other embodiments are possible within the spirit and the scope of the present invention.

WHAT IS CLAIMED IS:

1. A memory organization method for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, and in which the size of the memory portion for reading or writing data differs from the size of the smallest memory portion for erasing, the method comprising the steps of:

(a) providing a plurality of physical units of the memory, each of said physical units being the smallest memory portion for erasing, each of said physical units being designated by a physical unit number and each of said physical units being divided into a plurality of physical blocks, each of said plurality of physical blocks being the memory portion for reading or writing data and each of said physical blocks being designated by a physical block offset within said physical unit;

(b) providing a plurality of virtual units of the memory, each virtual unit being designated by a virtual unit number and each of said virtual units featuring a plurality of virtual blocks, each of said virtual blocks being designated by a virtual block offset within said virtual unit;

(c) providing a virtual map for mapping each virtual unit to at least one physical unit; and

(d) mapping each virtual block within said virtual unit to one physical block within said at least one physical unit.

2. The method of claim 1, further comprising the steps of:

(e) receiving a write command to write data at a virtual block;

(f) locating a virtual unit containing said virtual block;

(g) locating a writable block within a physical unit mapped to said virtual unit; and

(h) writing said data to said writable physical block.

3. The method of claim 2, further comprising the steps of:

(i) if an unwritten physical block in a written physical unit cannot be located, locating an unwritten physical unit;

(j) writing said data to a writable physical block of said unwritten physical unit; and

(k) updating said virtual map by additionally mapping said virtual unit to said unwritten physical unit, such that said virtual unit corresponds to said unwritten physical unit and to said written physical unit, said unwritten physical unit and said written physical unit forming a chain of physical units.

4. The method of claim 3, wherein said writable physical block of said unwritten physical unit has a physical block offset, and said physical block offset corresponds to said virtual block offset of said mapped virtual unit.

5. The method of claim 3, further comprising the steps of:

(l) if an unwritten physical block in a written physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain;

(m) locating said last physical unit in said chain;

(n) moving data from each of said physical blocks of said written physical unit to a writable physical block of said last physical unit, said writable physical block having substantially the same block offset as said physical block of said written physical unit; and

(o) updating said virtual map by mapping said virtual unit to said last physical unit, such that said virtual unit corresponds substantially only to said last physical unit.

6. The method of claim 5, further comprising the step of:

(p) erasing substantially all of said written physical units in said chain, with the exception of said last physical unit.

7. The method of claim 3, further comprising the steps of:

(l) if no unwritten physical unit is available for allocation for said chain, allocating an unwritten physical unit for reorganization;

(m) moving data from each of said physical blocks of said written physical unit to a writable physical block of said unwritten physical unit; and

(n) updating said virtual map by mapping said virtual unit to said unwritten physical unit, such that said virtual unit corresponds substantially only to said unwritten physical unit.

8. The method of claim 7, further comprising the step of:

(o) erasing all of said written physical units.

9. The method of claim 2, further comprising the steps of:

(i) if an unwritten physical block in a written physical unit cannot be located, allocating an unwritten physical unit to form a chain of physical units, such that said unwritten physical unit is a last physical unit in said chain;

(j) writing said data to an unwritten physical block in said last physical unit;

(k) moving data from each of said physical blocks to a writable physical block of said unwritten physical unit, except data written in step (j); and

(m) updating said virtual map by mapping said virtual unit to said written physical unit, such that said virtual unit corresponds to said written physical unit.

10. A method of writing data for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, the method comprising the steps of:

(a) providing a plurality of physical units being divided into a plurality of physical blocks, each of said physical units having a physical unit number and each of said physical blocks having a physical block offset within said physical unit;

(b) providing a plurality of virtual units being divided into a plurality of virtual blocks, each of said virtual units having a virtual unit number and each of said virtual blocks having a virtual block offset within said virtual unit, each virtual unit being mapped to at least one physical unit;

(c) receiving a write command to write data at a virtual block;

(d) determining a virtual unit containing said virtual block having a virtual block offset;

(e) locating a physical unit corresponding to said virtual unit;

(f) locating a physical block within said physical unit;

(g) determining if said physical block is unwritten;

(h) writing said data to said physical block only if said physical block is unwritten;

(i) alternatively, if said physical block is not unwritten, allocating an unwritten physical unit;

(j) locating a writable physical block within said unwritten physical unit, said writable physical block having a physical block offset;

(k) writing said data to said writable physical block; and

(l) additionally mapping said virtual unit to said unwritten physical unit containing said writable physical block, such that said virtual unit is additionally mapped to said unwritten physical unit to form a chain of physical units.

11. The method of claim 10, wherein said physical block offset of said writable physical block has an identical block offset number as said virtual block offset.

12. The method of claim 10, further comprising the steps of:

(m) if an unwritten physical block in a written physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain;

(n) locating said last physical unit in said chain;

(o) transferring all data within said physical blocks of said written physical unit to said physical blocks of said last physical unit; and

(p) updating said virtual map such that said virtual unit corresponds only to said last physical unit.

13. The method of claim 12, further comprising the step of:

(q) erasing all of said written physical units.

14. The method of claim 10, further comprising the steps of:

(l) if no unwritten physical unit is available for allocation, locating a last physical unit in said chain;

(m) transferring all data within said physical blocks of said written physical units to said physical blocks of said last physical unit; and

(n) updating said virtual map such that said virtual unit corresponds only to said last physical unit.

15. The method of claim 14, further comprising the step of:

(o) erasing substantially all of said written physical units, with the exception of said last physical unit.

FIG.1

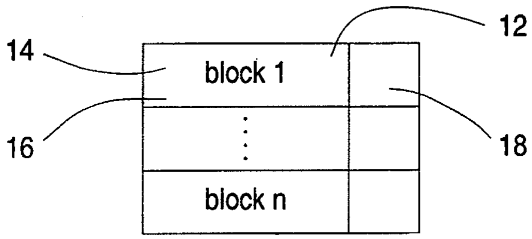

block 1
block n

FIG.2

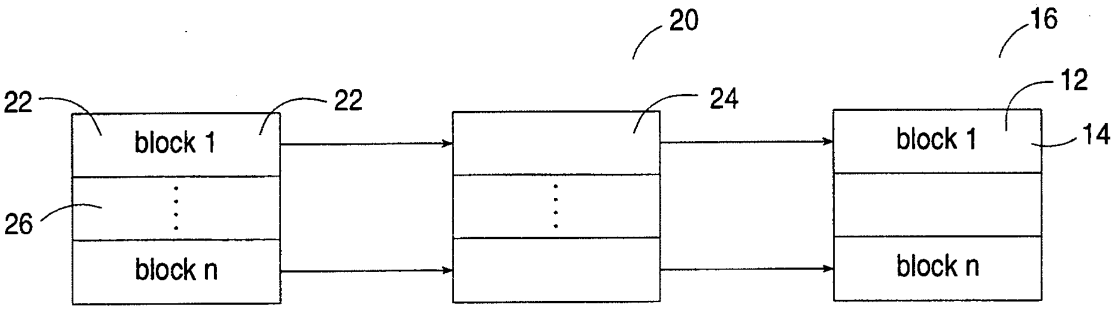

block 1
block 1
block n
block n

FIG.3A

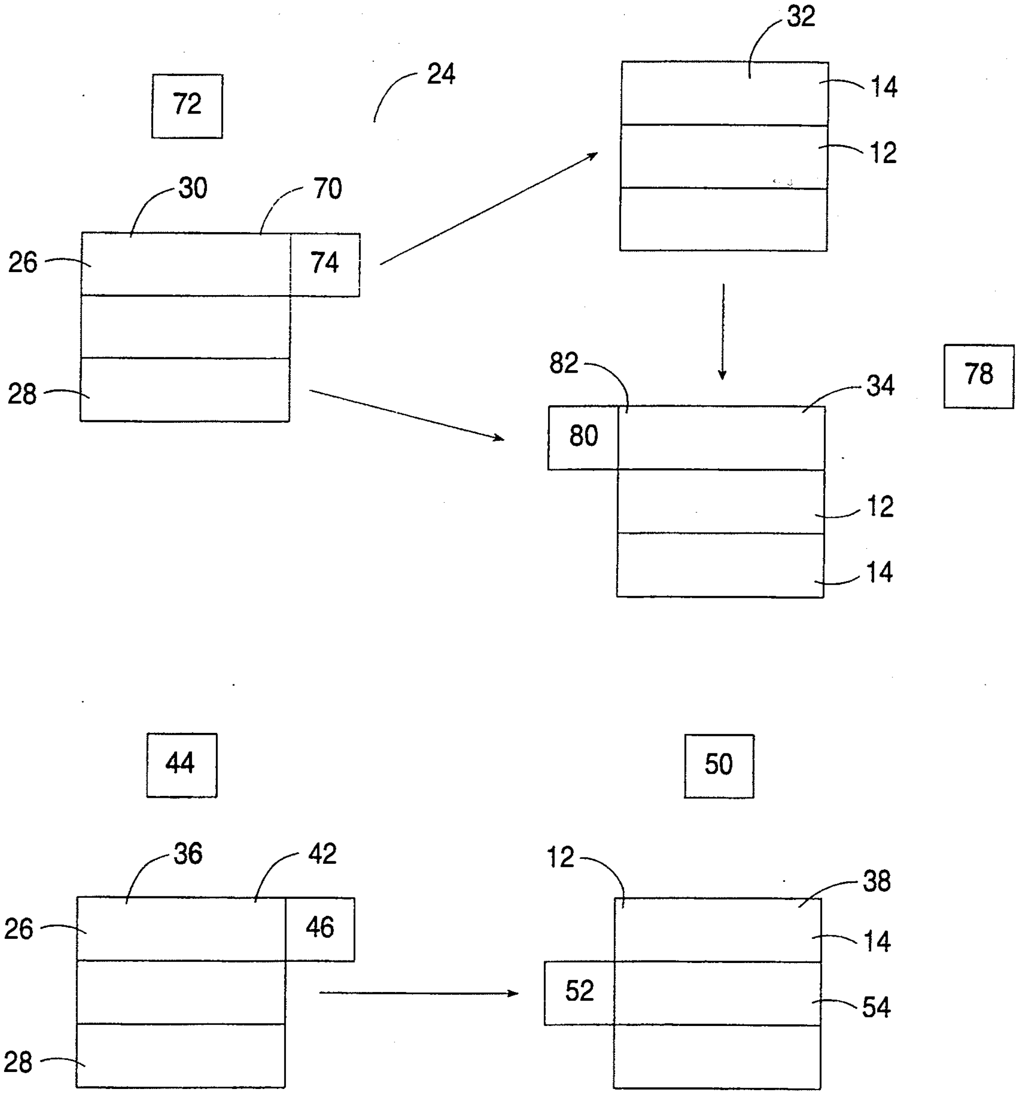


FIG.3B

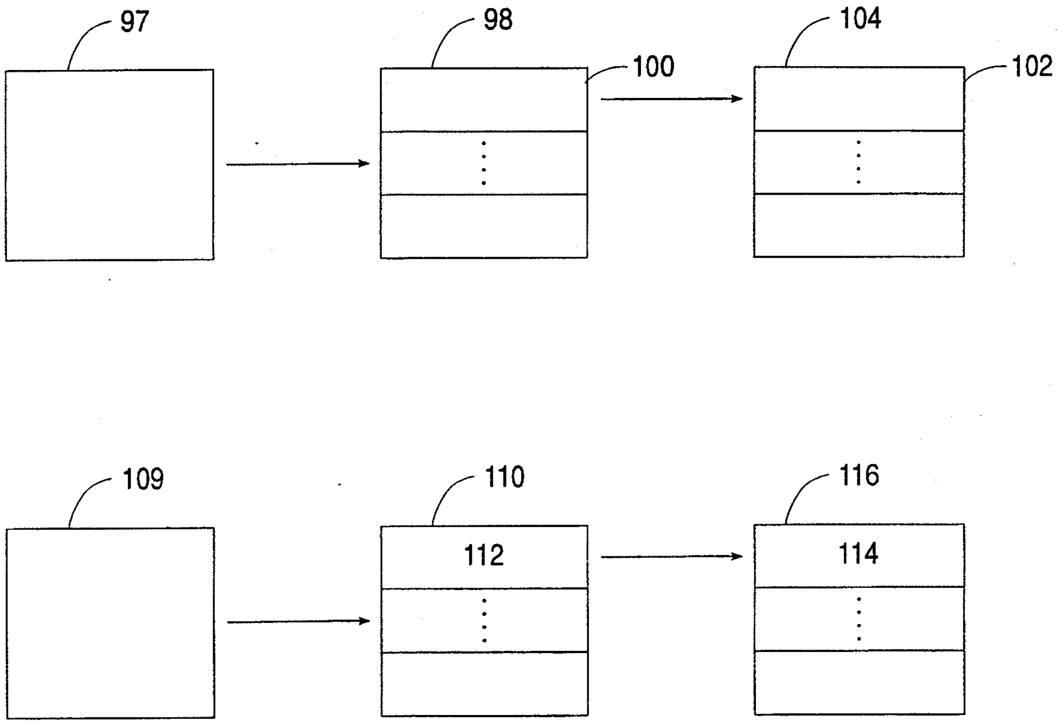

97
98
100
104
102
109
110
112
116
114

FIG.4A

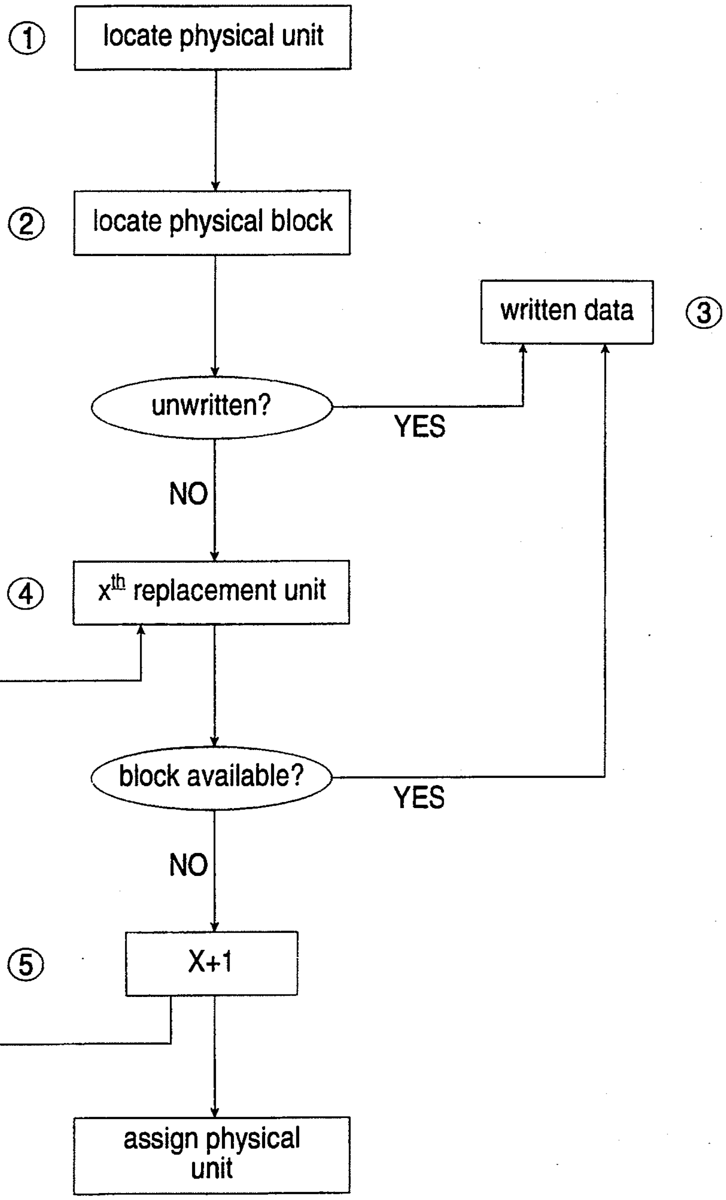
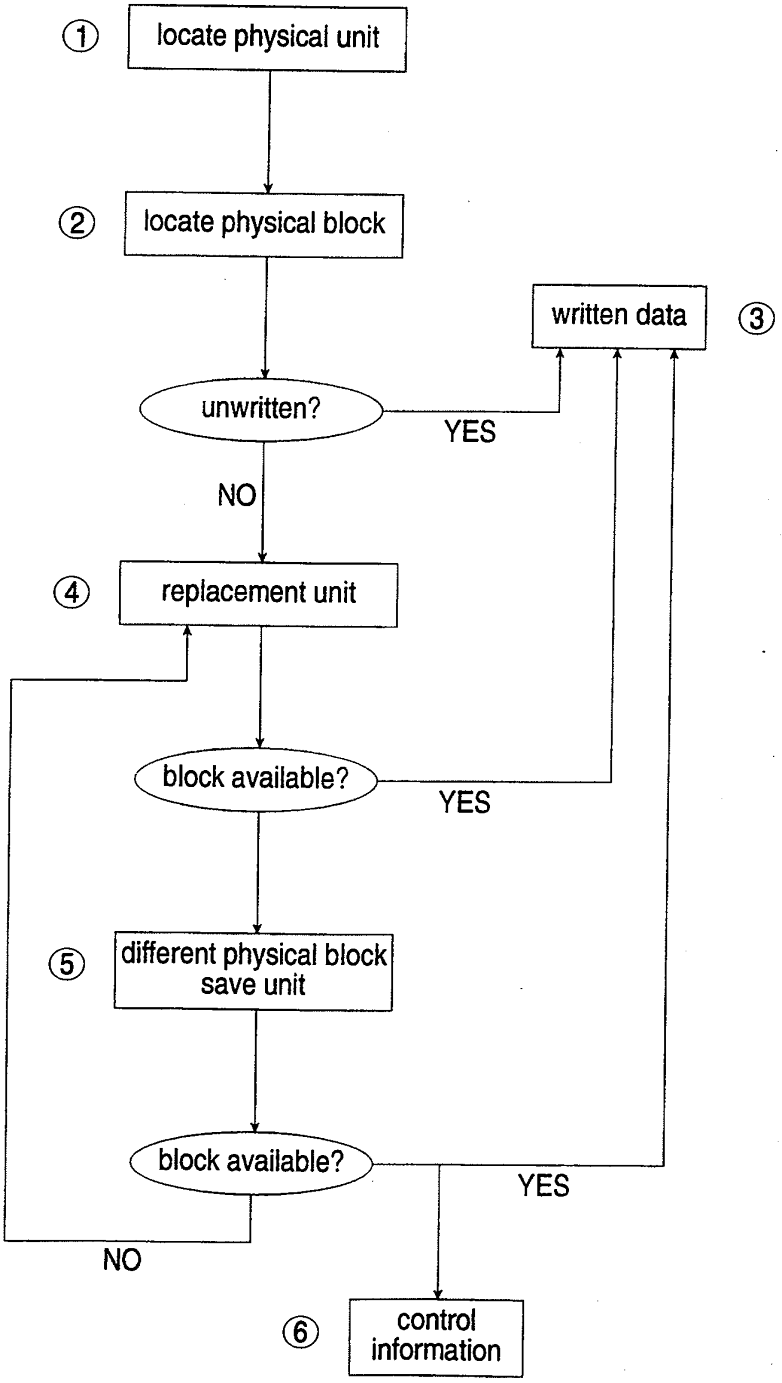

① locate physical unit
② locate physical block
③ written data
unwritten?
YES
NO
④ x^{th} replacement unit
block available?
YES
NO
⑤ X+1
assign physical unit

FIG.4B



①
locate physical unit
②
locate physical block
written data
③
unwritten?
YES
NO
④
replacement unit
block available?
YES
⑤
different physical block save unit
block available?
YES
NO
⑥
control information

FIG.5A

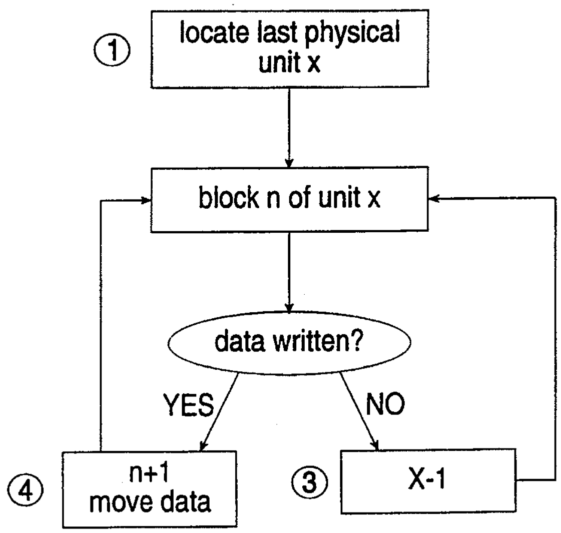
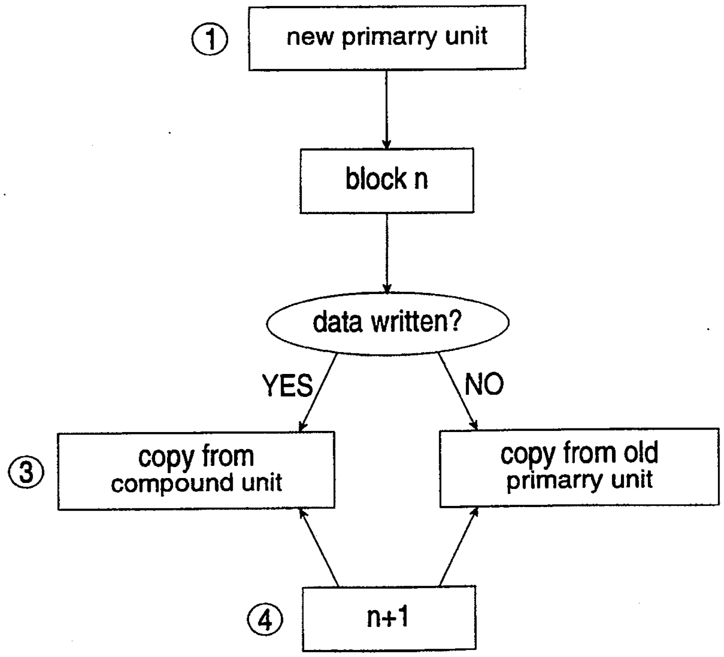

①
locate last physical unit x
block n of unit x
data written?
YES
NO
④
n+1
move data
③
X-1

FIG.5B


①
new primarry unit
block n
data written?
YES
NO
③
copy from compound unit
copy from old primarry unit
④
n+1

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/21017

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :GO6F 12/12
US CL : 711/103, 165, 202, 209
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)

U.S. : 711/103, 165, 202, 209

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 practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,404,485 A (BAN) 04 April 1995, column 2 (all), column 4 lines 11-66, column 5 line 36 to column 6 line 27.	1-4, 7-8, 10-11
Y	US 5,479,638 A (ASSAR et al) 26 December 1995, column 4 lines 1-58, column 5 line 53 to column 6 line 11.	1-4, 7-8, 10-11
Y	US 5,459,850 A (CLAY et al) 17 October 1995, column 3 lines 9-24, column 18 line 59 to column 21 line 44.	1-4, 7-11, 14-15
A	US 5,630,093 A (HOLZHAMMER et al) 13 May 1997.	1-15
A	US 5,644,539 A (YAMAGAMI et al) 01 July 1997.	1-15

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"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
"A"	document defining the general state of the art which is not considered to be of particular relevance		
"E"	earlier document published on or after the international filing date	"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
"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)	"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
"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	"&"	document member of the same patent family

Date of the actual completion of the international search 15 MARCH 1999	Date of mailing of the international search report 05 APR 1999
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer EDDIE P. CHAN Jon Hill Telephone No. (703) 305-3900

Form PCT/ISA/210 (second sheet)(July 1992)★

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ^{6} **:**
G06F 12/12

A1

(11) International Publication Number: **WO 99/21093**

(43) International Publication Date: 29 April 1999 (29.04.99)

(21) International Application Number: PCT/US98/21017

(22) International Filing Date: 5 October 1998 (05.10.98)

(30) Priority Data:
08/951,644 16 October 1997 (16.10.97) US

(71) Applicant *(for all designated States except US)*: M–SYSTEMS FLASH DISK PIONEERS LTD. [IL/IL]; Building 7, Atidim Industrial Park, P.O. Box 58036, 61580 Tel Aviv (IL).

(71) Applicant *(for TJ only)*: FRIEDMAN, Mark, M. [US/IL]; Alharizi 1, 43406 Raanana (IL).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: BAN, Amir [IL/IL]; Yabok Street 4, 47205 Ramat Hasharon (IL).

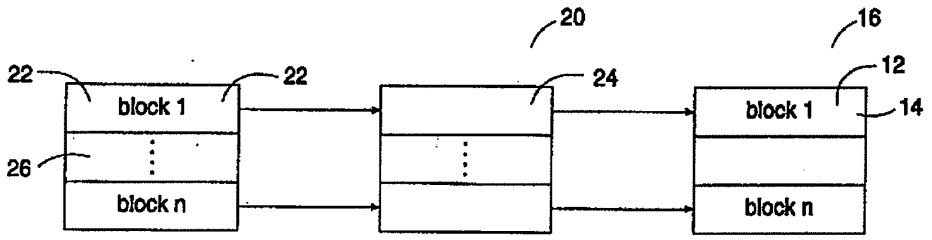
(74) Common Representative: FRIEDMAN, Mark, M.; c/o CASTORINA, Anthony, Suite 207, 2001 Jefferson Davis Highway, Arlington, VA 22202 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.
With amended claims.

Date of publication of the amended claims:
22 July 1999 (22.07.99)

(54) Title: IMPROVED FLASH FILE SYSTEM



(57) Abstract

A flash memory device, and methods for writing to the device and for reorganizing the device. The flash memory device (20) includes a physical device (10), a virtual device (22) and a virtual map (24) which relates the virtual addresses of the virtual device to the physical addresses of the physical device.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

AMENDED CLAIMS

[received by the International Bureau on 7 June 1999 (07.06.99);
original claims 1-15 replaced by amended claims 1-16 (7 pages)]

1. A memory organization method for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, and in which the size of the memory portion for reading or writing data differs from the size of the smallest memory portion for erasing, the method comprising the steps of:

(a) providing a plurality of physical units of the memory, each of said physical units being the size of the smallest memory portion for erasing, each of said physical units being designated by a physical unit number and each of said physical units being divided into a plurality of physical blocks, each of said plurality of physical blocks being the size of the memory portion for reading or writing data and each of said physical blocks being designated by a physical block offset within said physical unit;

(b) providing a plurality of virtual units of the memory, each virtual unit being designated by a virtual unit number and each of said virtual units featuring a plurality of virtual blocks, each of said virtual blocks being designated by a virtual block offset within said virtual unit;

(c) mapping each virtual unit to at least one physical unit to form a virtual map; and

(d) mapping each virtual block within said virtual unit to one physical block within said at least one physical unit according to said virtual map.

2. The method of claim 1, further comprising the step of:

(e) altering said virtual map to reflect a change in the memory by changing a correspondence between at least one of said plurality of virtual units of the memory and at least one of said plurality of physical units of the memory.

3. The method of claim 1, further comprising the steps of:

(e) receiving a write command to write data at a virtual block;

(f) locating a virtual unit containing said virtual block;

(g) locating a writable block within a physical unit mapped to said virtual unit; and

(h) writing said data to said writable physical block to form a written physical unit.

4. The method of claim 3, further comprising the steps of:

(i) if an unwritten physical block in said written physical unit cannot be located, locating a second physical unit;

(j) writing said data to a writable physical block of said second physical unit; and

(k) updating said virtual map by additionally mapping said virtual unit to said second physical unit, such that said virtual unit corresponds to said second physical unit and to said written physical unit, said second physical unit and said written physical unit forming a chain of physical units.

5. The method of claim 4, wherein said writable physical block of said second physical unit has a physical block offset, and said physical block offset corresponds to said virtual block offset of said mapped virtual unit.

6. The method of claim 4, wherein said writable physical block of said second physical unit has a physical block offset, and said phyiscal block offset is different than said virtual block offset of said mapped virtual unit.

7. The method of claim 4, further comprising the steps of:

(l) if an unwritten physical block cannot be located in any physical unit, locating a second virtual unit corresponding to a plurality of physical units in a chain;

(m) locating said last physical unit in said chain;

(n) moving data from each of said physical blocks of said written physical unit to a writable physical block of said last physical unit, said writable physical block having the same block offset as said physical block of said written physical unit; and

(o) updating said virtual map by mapping said virtual unit to said last physical unit, such that said virtual unit corresponds only to said last physical unit.

8. The method of claim 7, further comprising the step of:

(p) erasing all of said written physical units in said chain, with the exception of said last physical unit.

9. The method of claim 4, further comprising the steps of:

AMENDED SHEET (ARTICLE 19)

(l) if no unwritten physical unit is available for allocation for said chain, allocating an unwritten physical unit for reorganization;

(m) moving data from each of said physical blocks of said written physical unit to a writable physical block of said unwritten physical unit; and

(n) updating said virtual map by mapping said virtual unit to said unwritten physical unit, such that said virtual unit corresponds only to said unwritten physical unit.

10. A memory organization method for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, and in which the size of the memory portion for reading or writing data differs from the size of the smallest memory portion for erasing, the method comprising the steps of:

(a) providing a plurality of physical units of the memory, each of said physical units being the smallest memory portion for erasing, each of said physical units being designated by a physical unit number and each of said physical units being divided into a plurality of physical blocks, each of said plurality of physical blocks being the memory portion for reading or writing data and each of said physical blocks being designated by a physical block offset within said physical unit;

(b) providing a plurality of virtual units of the memory, each virtual unit being designated by a virtual unit number and each of said virtual units featuring a plurality of virtual blocks, each of said virtual blocks being designated by a virtual block offset within said virtual unit;

(c) providing a virtual map for mapping each virtual unit to at least one physical unit;

(d) mapping each virtual block within said virtual unit to one physical block within said at least one physical unit;

(e) receiving a write command to write data at a virtual block;

(f) locating a virtual unit containing said virtual block;

(g) locating a writable block within a physical unit mapped to said virtual unit;

(h) writing said data to said writable physical block;

AMENDED SHEET (ARTICLE 19)

(i) if an unwritten physical block in a written physical unit cannot be located, allocating an unwritten physical unit to form a chain of physical units, such that said unwritten physical unit is a last physical unit in said chain;

(j) writing said data to an unwritten physical block in said last physical unit;

(k) moving data from each of said plurality of physical blocks to a writable physical block of said last physical unit, except data written in step (j); and

(l) updating said virtual map by mapping said virtual unit to said written physical unit, such that said virtual unit corresponds to said written physical unit.

11. A method of writing data for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, the method comprising the steps of:

(a) providing a plurality of physical units being divided into a plurality of physical blocks, each of said physical units having a physical unit number and each of said physical blocks having a physical block offset within said physical unit;

(b) providing a plurality of virtual units being divided into a plurality of virtual blocks, each of said virtual units having a virtual unit number and each of said virtual blocks having a virtual block offset within said virtual unit, each virtual unit being mapped to at least one physical unit;

(c) receiving a write command to write data at a virtual block;

(d) determining a virtual unit containing said virtual block having a virtual block offset;

(e) locating a physical unit corresponding to said virtual unit;

(f) locating a physical block within said physical unit;

(g) determining if said physical block is unwritten;

(h) writing said data to said physical block only if said physical block is unwritten;

(i) alternatively, if said physical block is not unwritten, allocating second physical unit;

(j) locating a writable physical block within said second physical unit, said writable physical block having a physical block offset;

(k) writing said data to said writable physical block;

(l) additionally mapping said virtual unit to said second physical unit containing said writable physical block, such that said virtual unit is additionally mapped to said second physical unit to form a chain of physical units;

(m) if an unwritten physical block in a written physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain;

(n) locating a last physical unit in said chain;

(o) transferring all data within said physical blocks of said written physical unit to said physical blocks of said last physical unit; and

(p) updating said virtual map such that said virtual unit corresponds only to said last physical unit.

12. The method of claim 11, wherein said physical block offset of said writable physical block has an identical block offset number as said virtual block offset.

13. The method of claim 11, further comprising the step of:

(q) erasing all of said written physical units.

14. A method of writing data for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, the method comprising the steps of:

(a) providing a plurality of physical units being divided into a plurality of physical blocks, each of said physical units having a physical unit number and each of said physical blocks having a physical block offset within said physical unit;

(b) providing a plurality of virtual units being divided into a plurality of virtual blocks, each of said virtual units having a virtual unit number and each of said virtual blocks having a virtual block offset within said virtual unit, each virtual unit being mapped to at least one physical unit;

(c) receiving a write command to write data at a virtual block;

(d) determining a virtual unit containing said virtual block having a virtual block offset;

(e) locating a physical unit corresponding to said virtual unit;

(f) locating a physical block within said physical unit;

(g) determining if said physical block is unwritten;

(h) writing said data to said physical block only if said physical block is unwritten;

(i) alternatively, if said physical block is not unwritten, allocating a second physical unit;

(j) locating a writable physical block within said second physical unit, said writable physical block having a physical block offset;

(k) writing said data to said writable physical block; and

(l) additionally mapping said virtual unit to said second physical unit containing said writable physical block, such that said virtual unit is additionally mapped to said second physical unit to form a chain of physical units;

(m) if no unwritten physical unit is available for allocation, locating a last physical unit in said chain;

(n) transferring all data within said physical blocks of said written physical units to said physical blocks of said last physical unit; and

(o) updating said virtual map such that said virtual unit corresponds only to said last physical unit.

15. The method of claim 14, further comprising the step of:

(p) erasing all of said written physical units, with the exception of said last physical unit.

16. A memory organization method for a memory in which data can only be written to an unwritten portion of the memory, such that a written portion of the memory must be erased to become unwritten, and in which the size of the memory portion for reading or writing data differs from the size of the smallest memory portion for erasing, the method comprising the steps of:

(a) providing a plurality of physical units of the memory, each of said physical units being the size of the smallest memory portion for erasing, each of said physical units being designated by a physical unit number and each of said physical units being divided into a plurality of physical blocks, each of said plurality of physical blocks being the size of the memory portion for reading or writing data and each of said physical blocks being designated by a physical block offset within said physical unit;

AMENDED SHEET (ARTICLE 19)

(b) providing a plurality of virtual units of the memory, each virtual unit being designated by a virtual unit number and each of said virtual units featuring a plurality of virtual blocks, each of said virtual blocks being designated by a virtual block offset within said virtual unit;

(c) providing a virtual map for mapping each virtual unit to at least one physical unit;

(d) mapping each virtual block within said virtual unit to one physical block within said at least one physical unit;

(e) receiving a write command to write data at a virtual block;

(f) locating a virtual unit containing said virtual block;

(g) locating a writable block within a physical unit mapped to said virtual unit;

(h) writing said data to said writable physical block to form a written physical unit;

(i) if an unwritten physical block in said written physical unit cannot be located, locating a second physical unit with a writable physical block;

(j) writing said data to said writable physical block of said second physical unit;

(k) updating said virtual map by additionally mapping said virtual unit to said second physical unit, such that said virtual unit corresponds to said second physical unit and to said written physical unit, said second physical unit and said written physical unit forming a chain of physical units;

(l) if an unwritten physical block in any physical unit cannot be located, locating a second virtual unit corresponding to a plurality of physical units in a chain;

(m) locating a last physical unit in said chain;

(n) moving data from each of said physical blocks of said written physical unit to a writable physical block of said last physical unit, said writable physical block having substantially the same block offset as said physical block of said written physical unit; and

(o) updating said virtual map by mapping said virtual unit to said last physical unit, such that said virtual unit corresponds only to said last physical unit.

(19)日本国特許庁（ＪＰ）

(12) **公表特許公報**（Ａ）

(11)特許出願公表番号
特表2001－521220
（P2001－521220A）

(43)公表日　平成13年11月６日(2001.11.6)

(51)Int.Cl.⁷		識別記号	ＦＩ		テーマコード＊(参考)	
Ｇ０６Ｆ	12/00	５４２	Ｇ０６Ｆ	12/00	５４２Ａ	５Ｂ０６０
		５９７			５９７Ｕ	５Ｂ０８２
	12/02	５７０		12/02	５７０Ａ	

審査請求　有　　予備審査請求　有　　（全 39 頁）

(21)出願番号　特願2000－517345(P2000－517345)
(86)(22)出願日　平成10年10月５日(1998.10.5)
(85)翻訳文提出日　平成12年４月11日(2000.4.11)
(86)国際出願番号　ＰＣＴ／ＵＳ９８／２１０１７
(87)国際公開番号　ＷＯ９９／２１０９３
(87)国際公開日　平成11年４月29日(1999.4.29)
(31)優先権主張番号　０８／９５１，６４４
(32)優先日　平成９年10月16日(1997.10.16)
(33)優先権主張国　米国（ＵＳ）

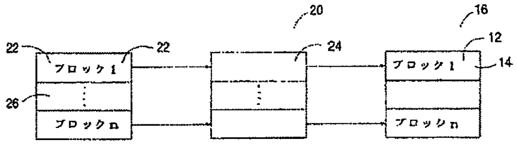
(71)出願人　エム　システムズ　フラッシュ　ディスク　パイオニアズ　リミテッド
Ｍ－ＳＹＳＴＥＭＳ　ＦＬＡＳＨ　ＤＩＳＫ　ＰＩＯＮＥＥＲＳ　ＬＴＤ．
イスラエル国、テル　アビブ　61580、ピー　オー　ボックス　58036、アティディム　インダストリアル　パーク、ビルディング　７
(72)発明者　パン　アミール
イスラエル国、ラマット　ハシャロン　47205、ヤボック　ストリート　４
(74)代理人　弁理士　大西　正悟

最終頁に続く

(54)【発明の名称】　改善されたフラッシュファイルシステム

(57)【要約】

フラッシュメモリデバイスおよびデバイスへの書込みならびにデバイスの再構成のための方法に関する。フラッシュメモリデバイス（２０）は、物理デバイス（１０）、仮想デバイス（２２）および仮想デバイスのアドレスを物理デバイスの物理アドレスに関連付ける仮想マップ（２４）を含む。



【特許請求の範囲】

【請求項１】　メモリのためのメモリ構成方法であって、前記メモリの未書込みの部分にのみデータを書込むことができるため、前記メモリの書込み済みの部分を未書込みの状態にするために消去しなければならず、読出しまたは書込みのデータ用のメモリ部分のサイズが、消去用の最小メモリ部分のサイズと異なるようになっているメモリ構成法であって、

（ａ）前記メモリの複数の物理ユニットを設けるステップであって、前記物理ユニットのそれぞれが消去用の最小メモリ部分のサイズであり、前記物理ユニットのそれぞれが物理ユニット番号によって表され、前記物理ユニットのそれぞれが複数の物理ブロックに分割され、前記複数の物理ブロックのそれぞれが読出しまたは書込みのデータ用のメモリ部分のサイズであり、前記物理ブロックのそれぞれが前記物理ユニット内部の物理ブロックオフセットによって表されるようになっているステップと、

（ｂ）前記メモリの複数の仮想ユニットを設けるステップであって、前記仮想ユニットのそれぞれが仮想ユニット番号によって表され、前記仮想ユニットのそれぞれが複数の仮想ブロックを特徴付け、前記仮想ブロックのそれぞれが前記仮想ユニット内部の仮想ブロックオフセットによって表されるようになっているステップと、

（ｃ）各仮想ユニットを少なくとも１つの物理ユニットにマッピングし、仮想マップを形成するステップと、

（ｄ）前記仮想マップによって、前記仮想ユニット内部の各仮想ブロックを前記少なくとも１つの物理ユニット内部の１つの物理ブロックにマッピングするステップと、を含む方法。

【請求項２】　（ｅ）前記メモリの前記複数の仮想ユニットの少なくとも１つと前記メモリの前記複数の物理ユニットの少なくとも１つとの対応関係を変更することによって、前記メモリにおける変更を反映するために前記仮想マップを変更するステップと、をさらに含む請求項１に記載の方法。

【請求項３】　（ｅ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、

（ｆ）前記仮想ブロックを含む仮想ユニットを突き止めるステップと、

（ｇ）前記仮想ユニットにマッピングされる物理ユニット内部の書込み可能なブロックを突き止めるステップと、

（ｈ）前記データを前記書込み可能な物理ブロックに書込み、書込み済みの物理ユニットを形成するステップと、をさらに含む請求項１に記載の方法。

【請求項４】　（ｉ）前記書込み済みの物理ユニットにおいて、未書込みの物理ブロックを突き止めることができない場合には、第２の物理ユニットを突き止めるステップと、

（ｊ）前記データを前記第２の物理ユニットの書込み可能な物理ブロックに書込むステップと、

（ｋ）前記仮想ユニットを前記第２の物理ユニットに追加的にマッピングすることによって、前記仮想マップを更新し、前記仮想ユニットが前記第２の物理ユニットおよび前記書込み済みの物理ユニットに対応するようにするステップであって、前記第２の物理ユニットおよび前記書込み済みの物理ユニットが物理ユニットのチェーンを形成するようになっているステップと、をさらに含む請求項３に記載の方法。

【請求項５】　前記第２の物理ユニットの前記書込み可能な物理ブロックが、物理ブロックオフセットを備え、前記物理ブロックオフセットが前記マッピングされた仮想ユニットの前記仮想ブロックオフセットに対応する請求項４に記載の方法。

【請求項６】　前記第２の物理ユニットの前記書込み可能な物理ブロックが物理ブロックオフセットを備え、前記物理ブロックオフセットが前記マッピングされた仮想ユニットの前記仮想ブロックオフセットとは異なる請求項４に記載の方法。

【請求項７】　（ｌ）任意の物理ユニットにおいて、未書込みの物理ブロックを突き止めることができない場合には、チェーンにおいて複数の物理ユニットに対応する第２の仮想ユニットを突き止めるステップと、

（ｍ）前記チェーンにおいて前記最終物理ユニットを突き止めるステップと、

（ｎ）前記書込み済みの物理ユニットの前記物理ブロックのそれぞれから前記

最終物理ユニットの書込み可能な物理ブロックまで、データを移動するステップであって、前記書込み可能な物理ブロックが前記書込み済みの物理ユニットの前記物理ブロックと同一のブロックオフセットを備えるようになっているステップと、

（ｏ）前記仮想ユニットを前記最終物理ユニットにマッピングすることによって、前記仮想マップを更新し、前記仮想ユニットが前記最終物理ユニットにのみ対応するようになっているステップと、をさらに含む請求項４に記載の方法。

【請求項８】　（ｐ）前記チェーンにおいて、前記最終物理ユニットを除き、前記書込み済みの物理ユニットのすべてを消去するステップをさらに含む請求項７に記載の方法。

【請求項９】　（ｌ）未書込みの物理ユニットが前記チェーンの割当てに利用可能でない場合には、再構成のために未書込みの物理ユニットを割当てるステップと、

（ｍ）前記書込み済みの物理ユニットの前記物理ブロックから前記未書込みの物理ユニットの書込み可能な物理ブロックまで、データを移動するステップと、

（ｎ）前記仮想ユニットを前記未書込みの物理ユニットにマッピングすることによって、前記仮想マップを更新し、前記仮想ユニットが前記未書込みの物理ユニットにのみ対応するようになっているステップと、をさらに含む請求項４に記載の方法。

【請求項１０】　メモリのためのメモリ構成方法であって、前記メモリの未書込みの部分にのみデータを書込むことができるため、前記メモリの書込み済みの部分を未書込みの状態にするために消去しなければならず、データの読出しまたは書込み用のメモリ部分のサイズが消去用の最小メモリ部分のサイズと異なる方法であって、

（ａ）前記メモリの複数の物理ユニットを設けるステップであって、前記物理ユニットのそれぞれが消去用の最小メモリ部分であり、前記物理ユニットのそれぞれが物理ユニット番号によって表され、前記物理ユニットのそれぞれが複数の物理ブロックに分割され、前記複数の物理ブロックのそれぞれが読出しまたは書込みのデータ用のメモリ部分であり、前記物理ブロックのそれぞれが前記物理ユ

ニット内部の物理ブロックオフセットによって表されるようになっているステップと、

（ｂ）前記メモリの複数の仮想ユニットを設けるステップであって、前記仮想ユニットのそれぞれが仮想ユニット番号によって表され、前記仮想ユニットのそれぞれが複数の仮想ブロックを特徴付け、前記仮想ブロックのそれぞれが前記仮想ユニット内部の仮想ブロックオフセットによって表されるようになっているステップと、

（ｃ）各仮想ユニットを少なくとも１つの物理ユニットにマッピングするための仮想マップを設けるステップと、

（ｄ）前記仮想ユニット内部の各仮想ブロックを前記少なくとも１つの物理ユニット内部の１つの物理ブロックにマッピングするステップと、

（ｅ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、

（ｆ）前記仮想ブロックを含む仮想ユニットを突き止めるステップと、

（ｇ）前記仮想ユニットにマッピングされる物理ユニット内部で書込み可能なブロックを突き止めるステップと、

（ｈ）前記データを前記書込み可能な物理ブロックに書込むステップと、

（ｉ）書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、物理ユニットのチェーンを形成するために未書込みの物理ユニットを割当て、前記未書込みの物理ユニットが前記チェーンにおいて最終物理ユニットであるようにするステップと、

（ｊ）前記データを前記最終物理ユニットの未書込みの物理ブロックに書込むステップと、

（ｋ）ステップ（ｊ）において書込まれたデータを除いて、データを前記複数の物理ブロックのそれぞれから前記最終物理ユニットの書込み可能な物理ブロックまで、移動するステップと、

（ｌ）前記仮想ユニットを前記書込み済みの物理ユニットにマッピングすることによって前記仮想マップを更新し、前記仮想ユニットが前記書込み済みの物理ユニットに対応するようになっているステップと、を含む方法。

【請求項１１】　メモリのためのデータ書込み方法であって、前記メモリの未書込みの部分にのみデータを書込むことができるため、前記メモリの書込み済みの部分を未書込みの状態にするために消去しなければならない方法であって、

（ａ）複数の物理ブロックに分割されることになる複数の物理ユニットを設けるステップであって、前記物理ユニットのそれぞれが物理ユニット番号を備え、前記物理ブロックのそれぞれが前記物理ユニット内部の物理ブロックオフセットを備えているステップと、

（ｂ）複数の仮想ブロックに分割されることになる複数の仮想ユニットを設けるステップであって、前記仮想ユニットのそれぞれが仮想ユニット番号を備え、前記仮想ブロックのそれぞれが前記仮想ユニット内部の仮想ブロックオフセットを備え、各仮想ユニットが少なくとも１つの物理ユニットにマッピングされるようになっているステップと、

（ｃ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、

（ｄ）仮想ブロックオフセットを有する前記仮想ブロックを含む仮想ユニットを決定するステップと、

（ｅ）前記仮想ユニットに対応する物理ユニットを突き止めるステップと、

（ｆ）前記物理ユニットの内部で物理ブロックを突き止めるステップと、

（ｇ）前記物理ブロックが未書込みであるかどうかを決定するステップと、

（ｈ）前記物理ブロックが未書込みの場合に限り、前記データを前記物理ブロックに書込むステップと、

（ｉ）別法として、前記物理ブロックが未書込みでない場合には、第２の物理ユニットを割当てるステップと、

（ｊ）前記第２の物理ユニット内部の書込み可能な物理ブロックを突き止めるステップであって、前記書込み可能な物理ブロックが物理ブロックオフセットを備えるステップと、

（ｋ）前記データを前記書込み可能な物理ブロックに書込むステップと、

（ｌ）前記仮想ユニットを前記書込み可能な物理ブロックを含む前記第２の物理ユニットに追加的にマッピングし、前記仮想ユニットが、物理ユニットのチェ

ーンを形成するために前記第２の物理ユニットに追加的にマッピングされるようになっているステップと、

（ｍ）書込まれた物理ユニットにおいて、未書込みの物理ブロックを突き止めることができない場合には、チェーンにおいて複数の物理ユニットに対応する第２の仮想ユニットを突き止めるステップと、

（ｎ）前記チェーンにおいて最終物理ユニットを突き止めるステップと、

（ｏ）前記書込み済みの物理ユニットの前記物理ブロック内部の全データを前記最終物理ユニットの前記物理ブロックに移動するステップと、

（ｐ）前記仮想ユニットが前記最終物理ユニットにのみ対応するように前記仮想マップを更新するステップと、

を含む方法。

【請求項１２】　前記書込み可能な物理ブロックの前記物理ブロックオフセットが、前記仮想ブロックオフセットと同一のブロックオフセット番号を有する請求項１１に記載の方法。

【請求項１３】　（ｑ）前記書込み済みの物理ユニットのすべてを消去するステップをさらに含む請求項１１に記載の方法。

【請求項１４】　メモリのためのデータの書込み方法であって、前記メモリの未書込みの部分にのみデータを書込むことができるため、前記メモリの書込み済みの部分を未書込みの状態にするために消去しなければならない方法であって、

（ａ）複数の物理ブロックに分割されることになる複数の物理ユニットを設けるステップであって、前記物理ユニットのそれぞれが物理ユニット番号を備え、前記物理ブロックのそれぞれが前記物理ユニット内部の物理ブロックオフセットを備えているステップと、

（ｂ）複数の仮想ブロックに分割されることになる複数の仮想ユニットを設けるステップであって、前記仮想ユニットのそれぞれが仮想ユニット番号を備え、前記仮想ブロックのそれぞれが前記仮想ユニット内部の仮想ブロックオフセットを備え、各仮想ユニットが少なくとも１つの物理ユニットにマッピングされるステップと、

（ｃ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、

（ｄ）仮想ブロックオフセットを有する前記仮想ブロックを含む仮想ユニットを決定するステップと、

（ｅ）前記仮想ユニットに対応する物理ユニットを突き止めるステップと、

（ｆ）前記物理ユニットの内部で物理ブロックを突き止めるステップと、

（ｇ）前記物理ブロックが未書込みであるかどうかを決定するステップと、

（ｈ）前記物理ブロックが未書込みの場合に限り、前記データを前記物理ブロックに書込むステップと、

（ｉ）別法として、前記物理ブロックが未書込みでない場合には、第２の物理ユニットを割当てるステップと、

（ｊ）前記第２の物理ユニット内部の書込み可能な物理ブロックを突き止めるステップであって、前記書込み可能な物理ブロックが物理ブロックオフセットを備えるステップと、

（ｋ）前記データを前記書込み可能な物理ブロックに書込むステップと、

（ｌ）前記仮想ユニットを前記書込み可能な物理ブロックを含む前記第２の物理ユニットに追加的にマッピングし、前記仮想ユニットが、物理ユニットのチェーンを形成するために前記第２の物理ユニットに追加的にマッピングされるようになっているステップと、

（ｍ）未書込みの物理ユニットが割当てに利用可能でない場合には、チェーンにおいて最終物理ユニットを突き止めるステップと、

（ｎ）前記書込み済みの物理ユニットの前記物理ブロック内部の全データを前記最終物理ユニットの前記物理ブロックに移動するステップと、

（ｏ）前記仮想ユニットが前記最終物理ユニットにのみ対応するように前記仮想マップを更新するステップと、を含む方法。

【請求項１５】　（ｐ）前記最終物理ユニットを除いて、前記書込み済みの物理ユニットのすべてを消去するステップをさらに含む請求項１４に記載の方法。

【請求項１６】　メモリのためのメモリ構成方法であって、前記メモリの未

書込みの部分にのみデータを書込むことができるため、前記メモリの書込み済みの部分を未書込みの状態にするために消去しなければならず、データの読出しまたは書込み用のメモリ部分のサイズが消去用の最小メモリ部分のサイズと異なる方法であって、

（ａ）前記メモリの複数の物理ユニットを設けるステップであって、前記物理ユニットのそれぞれが消去用の最小メモリ部分のサイズであり、前記物理ユニットのそれぞれが物理ユニット番号によって表され、前記物理ユニットのそれぞれが複数の物理ブロックに分割され、前記複数の物理ブロックのそれぞれが読出しまたは書込みのデータ用のメモリ部分のサイズであり、前記物理ブロックのそれぞれが前記物理ユニット内部の物理ブロックオフセットによって表されるようになっているステップと、

（ｂ）前記メモリの複数の仮想ユニットを設けるステップであって、前記仮想ユニットのそれぞれが仮想ユニット番号によって表され、前記仮想ユニットのそれぞれが複数の仮想ブロックを特徴付け、前記仮想ブロックのそれぞれが前記仮想ユニット内部の仮想ブロックオフセットによって表されるようになっているステップと、

（ｃ）各仮想ユニットを少なくとも１つの物理ユニットにマッピングするための仮想マップを設けるステップと、

（ｄ）前記仮想ユニット内部の各仮想ブロックを前記少なくとも１つの物理ユニット内部の１つの物理ブロックにマッピングするステップと、

（ｅ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、

（ｆ）前記仮想ブロックを含む仮想ユニットを突き止めるステップと、

（ｇ）前記仮想ユニットにマッピングされる物理ユニット内部で書込み可能なブロックを突き止めるステップと、

（ｈ）前記データを前記書込み可能な物理ブロックに書込み、書込み済みの物理ユニットを形成するステップと、

（ｉ）前記書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、書込み可能な物理ブロックを備えた第２の物理

ユニットを突き止めるステップと、

（ｊ）前記データを前記第２の物理ユニットの前記書込み可能な物理ブロックに書込むステップと、

（ｋ）前記仮想ユニットを前記第２の物理ユニットに追加的にマッピングすることによって、前記仮想マップを更新し、前記仮想ユニットが前記第２の物理ユニットおよび前記書込み済みの物理ユニットに対応し、前記第２の物理ユニットおよび前記書込み済みの物理ユニットが物理ユニットのチェーンを形成するようになっているステップと、

（ｌ）任意の物理ユニットにおいて、未書込みの物理ブロックを突き止めることができない場合には、チェーンにおいて複数の物理ユニットに対応する第２の仮想ユニットを突き止めるステップと、

（ｍ）前記チェーンにおいて最終物理ユニットを突き止めるステップと、

（ｎ）前記書込み済みの物理ユニットの前記物理ブロックのそれぞれから前記最終物理ユニットの書込み可能な物理ブロックまで、データを移動するステップであって、前記書込み可能な物理ブロックが前記書込み済みの物理ユニットの前記物理ブロックと実質的に同一のブロックオフセットを有するステップと、

（ｏ）前記仮想ユニットを前記最終物理ユニットにマッピングすることによって前記仮想マップを更新し、前記仮想ユニットが前記最終物理ユニットに対応するようになっているステップと、を含む方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、フラッシュデバイスのデータ格納の処理を行うシステムに関し、さらに詳細には、フラッシュディスクとして機能することができるページモードフラッシュデバイスで情報の格納および検索を行うシステムに関する。

【０００２】

【従来の技術】

フラッシュデバイスは、フラッシュ型浮遊ゲートトランジスタで形成される電気的消去可能プログラマブル読出し専用メモリ（ＥＥＰＲＯＭ）を含み、機能および性能に関してＥＰＲＯＭメモリと類似である不揮発性メモリであり、メモリのページを消去する回路内部においてプログラム可能な操作を実現することができる付加的な機能を備えている。フラッシュデバイスは、従来の磁気記憶ディスクに比べて、比較的廉価で、比較的電力が少なくて済むという利点を備えている。しかし、フラッシュデバイスにおいて、その領域の以前のページを消去せずに、以前に書込んだ領域に再書込みを行うことは実際的ではない。フラッシュデバイスにはこのような制限があるために代表的な既存のオペレーティングシステムプログラムと共存することはできない。データが以前に書込まれた領域を最初に消去しない限り、フラッシュデバイスの中にあるメモリの領域にデータを書込むことができないからである。

【０００３】

ソフトウェア製品が従来技術において提案してきたことは、オペレーティングシステムプログラムの修正を行うことなく、フラッシュデバイスを既存のコンピュータオペレーティングプログラムによって処理することができることであった。しかし、このような従来技術プログラムはすべて、欠点を持っている。たとえば、あるプログラムはフラッシュメモリを「追記型」装置として作動させる。この従来のソフトウェア製品は、以前に書込んだ記憶位置を再利用することができない。結局すべての位置が書込まれた場合には、特定の利用者の介在なしに、メモリをそれ以上使用することはできない。他の従来技術プログラムは、ＳａｎＤ ｉｓｋによって提案されたプログラムのように、新たなデータがページに書込まれることになるたびに、メモリページ全体を消去し、再書込みを行う。このようなシステムは複数の消去サイクルを必要とするという欠点があり、これらのサイクルは比較的遅い上、非効率的であり、物理的な媒体そのものの急速な劣化を引き起こす。

【０００４】

従来技術のこのような欠点を克服するために、フラッシュファイルシステム（ＦＦＳ）が米国特許第５，４０４，４８５号に開示され、参照として本願に包含される。ＦＦＳはフラッシュデバイスにデータ格納およびデータ操作のシステムを提供し、これらのデバイスに磁気ディスクを基にしたデータ格納をエミュレートすることができるようにした。上記のように、比較的廉価であると共に電力消費が少ないため、フラッシュデバイスはデータ格納、特にラップトップのポータブルコンピュータに好都合な選択となる。ＦＦＳは磁気ディスク記憶装置の代わりとして作用するフラッシュデバイスの能力を強化する。さらに言えば、ＦＦＳは、米国特許第５，４０４，４８５号に開示されたように、きわめて有用であることが判明したため、データ形式の仕様は、フラッシュ変換層（ＦＴＬ）と呼ばれる規格として、ＰＣＭＣＩＡ（Ｐｅｒｓｏｎａｌ Ｃｏｍｐｕｔｅｒ Ｍｅｍｏｒｙ Ｃａｒｄ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ａｓｓｏｃｉａｔｉｏｎ）およびＪＥＩＤＡ（日本電子工業振興協会）の委員会によって採用された。

【０００５】

ＦＦＳは本来、フラッシュＥＥＰＲＯＭ装置のための仮想マッピングシステムを表している。仮想マップは、フラッシュデバイス内部の読取り／書込みブロックの物理アドレスをそのブロックの仮想アドレスと関連付ける表である。これらのブロックのそれぞれは比較的小さいため、５１２バイト、すなわち仮想マップ自体のサイズは相当大きい。ＦＦＳはまた、フラッシュＥＥＰＲＯＭ装置に仮想マップの容量に格納および維持し、仮想マップの格納に必要な他のメモリの量を最小限に抑える方法を含む。

【０００６】

上記のように、ＦＦＳは、フラッシュデバイスを磁気ディスク記憶装置のエミ

ュレータに変形する場合に特にうまくいくことが判明し、産業規格として採用されたほどである。しかしながら、ＦＦＳは、さらに新たなフラッシュデバイス技術の要件をすべて満たすことはできない。特に、ＦＦＳはＮＡＮＤおよびＡＮＤフラッシュ技術の場合にはうまくいかない。

【０００７】

従来技術のフラッシュメモリアーキテクチャのある種の欠点、特に書込み前に消去するシステムの欠点を克服しようとする試みの別の例は、米国特許第５，４７９，６３８号に開示される。米国特許第５，４７９，６３８号のシステムにおいて、書込み済みのブロックに対してさらにプログラムの書込みがさらに必要である場合には、特定の読出し／書込みブロックの物理的な位置がシフトされる。しかし、このシステムは、一度に５１２バイトの単一の読出し／書込みブロックを消去することができるフラッシュデバイスを利用して操作されることができるにすぎないという欠点を持っている。このような要件はハードウェアレベルで実装されるため、このシステムはまた、さらに新たなＮＡＮＤおよびＡＮＤフラッシュ技術に対して使用されることはできない。

【０００８】

【発明が解決しようとする課題】

ＮＡＮＤおよびＡＮＤは多くの点に関して、以前のフラッシュデバイスと異なる。第一に、消去可能なユニットサイズは、以前のフラッシュデバイスの６４ＫＢとは対照的に、ＮＡＮＤおよびＡＮＤの場合には約８ＫＢと小さめである。第二に、単一バイトを消去するために必要な時間として測定された場合であっても、ＮＡＮＤおよびＡＮＤの場合、消去時間が相当速い。第三に、フラッシュメモリはＮＡＮＤおよびＡＮＤのために長さ２５６または５１２バイトのページに分割され、ハードウェアデバイス自体の不変の特性である。「ページ」および「ブロック」の特有の特性はある程度異なるが、ここで使用している「ページ」なる語は、以前のフラッシュ技術で使用される「ブロック」なる語と概ね同意義であることに留意すべきである。このような特徴は、ＮＡＮＤおよびＡＮＤ技術に基づくフラッシュデバイスの動作に非常に密接な関係がある。

【０００９】

第一に、ページモードメモリは、１ページまたは任意のページを書込むための固定オーバヘッドを備える。対照してみると、従来のフラッシュ技術における書込み動作のためのオーバヘッドは、書込まれたバイト数に比例した。第二に、各ページが特にアドレス呼出し可能である複数の予備バイトを有するように、ＮＡＮＤおよびＡＮＤにおけるフラッシュメモリは構成される。このような予備バイトは、フラッシュメモリシステムに関する情報の格納に好都合な位置にある。最後に、消去される前にページを書込むことができる回数に制限がある。事前の消去を行うことなくさらに書込む場合は信頼性に欠けることを考慮して、この制限は比較的低く、８または１０回である。したがって、ページモードメモリは、成功するデータの格納および検索に対して、重要な利点および新たな難題の両方を有する。

【００１０】

不幸なことに、上記のように、一般に利用可能な従来技術のデータ処理システム、すなわちＦＦＳは、ページモードにおけるフラッシュメモリの動作に関して、重大な欠点がある。特に、ページモードプログラミングによって課せられる制限のために、ＦＦＳはＮＡＮＤおよびＡＮＤなどのページモードフラッシュ技術では最適でない性能を例証する。さらに、ブロックごとの消去動作に必要な要件のために、米国特許第５，４７９，６３８号に開示されたシステムも、このようなフラッシュ技術に対して使用することができない。

【００１１】

したがって、以前の非ページモードフラッシュデバイスの上で未だ利用可能であるが、ページモードフラッシュ技術の性能を最適化するようなＮＡＮＤおよびＡＮＤフラッシュデバイスの上でデータ格納を処理するためのシステムが必要であり、そのようなシステムを備えることは大いに好都合であると思われる。

【００１２】

【課題を解決するための手段】

本発明は、メモリのためにメモリの構成方法であって、データをメモリの未書込みの部分にのみ書込むことができるため、メモリの書込み済みの部分を未書込みの状態にするために消去しなければならず、データの読込みまたは書込み用の

メモリ部分のサイズは、消去用の最小メモリ部分のサイズとは異なる方法を提供する。本方法は、（ａ）メモリの複数の物理ユニットを提供するステップであって、物理ユニットのそれぞれが消去用の最小メモリ部分であり、物理ユニットのそれぞれが物理ユニット番号によって表され、物理ユニットのそれぞれが複数の物理ブロックに分割され、複数の物理ブロックのそれぞれがデータの読出しおよび書込み用のメモリ部分であり、複数の物理ブロックのそれぞれが物理ユニット内部の物理ブロックオフセットによって表されるようなステップと、（ｂ）メモリの複数の仮想ユニットを提供するステップであって、各仮想ユニットが仮想ユニット番号によって表され、仮想ユニットのそれぞれが複数の仮想ブロックを特徴付け、仮想ブロックのそれぞれが仮想ユニット内部の仮想ブロックオフセットによって表されるようなステップと、（ｃ）各仮想ユニットを少なくとも１つの物理ユニットにマッピングするための仮想マップを提供するステップと、（ｄ）仮想ユニット内部の各仮想ブロックを少なくとも１つの物理ユニット内部の１つの物理ブロックにマッピングするステップと、を含む。

【００１３】

本方法はさらに、（ｅ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、（ｆ）仮想ブロックを含む仮想ユニットを突き止めるステップと、（ｇ）仮想ユニットにマッピングされる物理ユニット内部の書込み可能ブロックを突き止めるステップと、（ｈ）書込み可能な物理ブロックにデータを書込むステップと、を含むことが好ましい。本方法はさらに、（Ｉ）書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、未書込みの物理ユニットを突き止めるステップと、（ｊ）未書込みの物理ユニットの書込み可能な物理ブロックにデータを書込むステップと、（ｋ）仮想ユニットを未書込みの物理ユニットに追加的にマッピングすることによって仮想マップを更新するステップであって、仮想ユニットが未書込みの物理ユニットおよび書込み済みの物理ユニットに対応し、未書込みの物理ユニットおよび書込み済みの物理ユニットが物理ユニットのチェーンを形成するようになっているステップと、を含むことがさらに好ましい。未書込みの物理ユニットの書込み可能な物理ブロックが物理ブロックオフセットを備え、物理ブロックオフセットがマッ

ピングされる仮想ユニットの仮想ブロックオフセットに対応することが最も好ましい。また、本方法はさらに、（ｌ）書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、チェーンにおいて複数の物理ユニットに対応する第２の仮想ユニットを突き止めるステップと、（ｍ）チェーンにおいて最終物理ユニットを突き止めるステップと、（ｎ）書込み済みの物理ユニットの物理ブロックのそれぞれから最終物理ユニットの書込み可能な物理ブロックまで、データを移動するステップであって、書込み可能な物理ブロックが書込み済みの物理ユニットの物理ブロックと実質的に同一のブロックオフセットを備えるステップと、（ｏ）仮想ユニットを最終物理ユニットにマッピングすることによって仮想マップを更新するステップであって、仮想ユニットが実質的に最終物理ユニットにのみ対応するようになっているステップと、を含むことが最も好ましい。本方法はまた、（ｐ）最終物理ユニットを除いて、チェーンにおける書込み済みの物理ユニットの実質的にすべてを消去するステップをさらに含むことが好ましい。

【００１４】

別法として、本方法はさらに、（ｌ）チェーンのための割当てに未書込みの物理ユニットが利用可能でない場合には、再構成のために未書込みの物理ユニットを割当てるステップと、（ｍ）書込み済みの物理ユニットの物理ブロックのそれぞれから未書込みの物理ユニットの書込み可能な物理ブロックまでデータを移動するステップと、（ｎ）仮想ユニットを未書込みの物理ユニットに仮想ユニットをマッピングすることによって仮想マップを更新するステップであって、仮想ユニットが実質的に未書込みの物理ユニットにのみ対応するようになっているステップと、を含むことが好ましい。本方法はさらに、（ｏ）書込み済みの物理ユニットのすべてを消去するステップを含むことがさらに好ましい。

【００１５】

本発明の別の好ましい実施例によれば、本方法はさらに、（ｉ）書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、物理ユニットのチェーンを形成するために未書込みの物理ユニットを割当てるステップであって、未書込みの物理ユニットがチェーンの最終物理ユニット

であるようになっているステップと、（ｊ）最終物理ユニットにおける未書込みの物理ブロックにデータを書込むステップと、（ｋ）ステップ（ｊ）で書込まれたデータを除いて、物理ブロックのそれぞれから未書込みの物理ユニットの書込み可能なブロックにデータを移動するステップと、（ｍ）書込み済みの物理ユニットに仮想ユニットをマッピングすることによって仮想マップを更新するステップであって、仮想ユニットが書込み済みの物理ユニットに対応するようになっているステップと、を含む。

【００１６】

本発明の別の実施例によれば、メモリのためのデータの書込み方法であって、メモリの未書込みの部分にのみデータを書込むことができるため、未書込みの状態にするためにメモリの書込み済みの部分を消去しなければならない方法を提供する。本方法は、（ａ）複数の物理ブロックに分割されることになっている複数の物理ユニットを提供するステップであって、物理ユニットのそれぞれが物理ユニット番号を備え、物理ブロックのそれぞれが物理ユニット内部に物理ブロックオフセットを備えるステップと、（ｂ）複数の仮想ブロックに分割されることになっている複数の仮想ユニットを提供するステップであって、仮想ユニットのそれぞれが仮想ユニット番号を備え、仮想ブロックのそれぞれが仮想ユニット内部に仮想ブロックオフセットを備え、各仮想ユニットが少なくとも１つの物理ユニットにマッピングされるステップと、（ｃ）仮想ブロックでデータを書込むための書込み命令を受信するステップと、（ｄ）仮想ブロックオフセットを有する仮想ブロックを含む仮想ユニットを決定するステップと、（ｅ）仮想ユニットに対応する物理ユニットを突き止めるステップと、（ｆ）物理ユニット内部において物理ブロックを突き止めるステップと、（ｇ）物理ブロックが未書込みかどうかを決定するステップと、（ｈ）物理ブロックが未書込みの場合に限り、物理ブロックにデータを書込むステップと、（ｉ）別法として、物理ブロックが未書込みでない場合には、未書込みの物理ユニットを割当てるステップと、（ｊ）未書込みの物理ユニットの内部において書込み可能な物理ブロックを突き止めるステップであって、書込み可能な物理ブロックが物理ブロックオフセットを備えるステップと、（ｋ）書込み可能な物理ブロックにデータを書込むステップと、（ｌ）

仮想ユニットを書込み可能な物理ブロックを含む未書込みの物理ユニットに追加的にマッピングするステップであって、物理ユニットのチェーンを形成するために、仮想ユニットが未書込みの物理ユニットに追加的にマッピングされるようになっているステップと、を含む。

【００１７】

本方法はさらに、（ｍ）書込み済みの物理ユニットにおいて未書込みの物理ブロックを突き止めることができない場合には、チェーンにおいて複数の物理ユニットに対応する第２の仮想ユニットを突き止めるステップと、（ｎ）チェーンにおいて最終物理ユニットを突き止めるステップと、（ｏ）書込み済みの物理ユニットの物理ブロック内部の全データを最終物理ユニットの物理ブロックに移動するステップと、（ｐ）仮想ユニットが最終物理ユニットにのみ対応するように仮想マップを更新するステップと、を含むことが好ましい。本方法はさらに、（ｑ）書込み済みの物理ユニットのすべてを消去するステップを含むことがさらに好ましい。

【００１８】

別法として、本方法はさらに、（ｌ）未書込みの物理ユニットが割当てに利用可能でない場合には、チェーンにおいて最終物理ユニットを突き止めるステップと、（ｍ）書込み済みの物理ユニットの物理ブロック内部の全データを最終物理ユニットの物理ブロックに移動するステップと、（ｎ）仮想ユニットが最終物理ユニットにのみ対応するように仮想マップを更新するステップと、を含むことが好ましい。本方法はさらに、（ｏ）最終物理ユニットを除き、書込み済みの物理ユニットの実質的にすべてを消去するステップを含むことがさらに好ましい。

【００１９】

【発明の実施の形態】

本発明は、ブロックなどのデータの読出しおよび書込みのためのメモリ部分のサイズが、ユニットなどの消去のための最小部分のサイズと異なるようになっているフラッシュメモリを構成する方法に関する。本発明の方法によって構成されることができるフラッシュメモリの種類の例は、これに限定されるわけではないが、ＮＡＮＤおよびＡＮＤ技術によって例示されるページモードデバイスを含む

。方法はまた、フラッシュメモリにデータの読出しおよび書込みを行うための方法および未書込みの物理ユニットがそれ以上利用可能でない場合には、フラッシュメモリを再構成する方法である。

【００２０】

以下、「物理ユニット」なる語は、消去されることができるメモリの最小部分またはその整数倍であるメモリの物理的な媒体またはハードウェアにあるユニットとして定義される。それは連続的な固定サイズの消去可能なメモリの部分である。「物理ブロック」なる語は、データの読出しまたは書込み用のメモリの部分として定義される。以下、「仮想ユニット」なる語は、物理ユニットと同一のサイズとして定義される。ＮＡＮＤおよびＡＮＤなどのページモードメモリ技術のために、消去することができるメモリの最小部分はページサイズより大きく、通常は約８ＫＢである。ここで使用されるように、「物理ブロック」なる語は、ページモードメモリ技術のための「ページ」なる語と同意義である。したがって、仮想ユニットは物理ユニットと同一の大きさである。

【００２１】

以下、「仮想マップ」なる語は、仮想ユニットを少なくとも１つの対応する物理ユニットに関連付ける表を表す。上記のように、各ユニット、すなわち仮想ユニットまたは物理ユニットは、複数のブロックで構成される。以下でさらに説明するように、ユニット内のブロックの正確な位置は、１つ以上の予め定められた規則に基づいて決定される。

【００２２】

各物理ユニットは物理ユニット番号によって表される。各物理ブロックの位置は、物理ブロックオフセットによって与えられる。同様に、各仮想ユニットは仮想ユニット番号によって表される。各仮想ブロックの位置は、仮想ブロックオフセットによって与えられる。各仮想ユニット番号は、１つ以上の物理ユニット番号に対応することができることを留意すべきである。したがって、仮想ユニットと物理ユニットとの間のマッピングは、１対１または１対多のいずれかであってもよい。

【００２３】

以下、「書込みデータ」なる語は、フラッシュメモリの上にデータを格納する行為を表す。「読出しデータ」なる語は、フラッシュメモリからデータを検索する行為を表す。以下、「未書込み」なる語は、データを書込ませることができる物理ブロックなどのメモリの一部分を示す。したがって、「未書込み」なる語は、これに限定されるわけではないが、ちょうど消去されたばかりのメモリの部分を含む。

【００２４】

本発明によって構成されるフラッシュメモリを有するコンピュータまたは他の電子デバイスにおいて、そのデバイスのオペレーティングシステムが、読出しおよび書込みデータのために仮想ユニットおよび仮想ブロックに相互作用を及ぼす。仮想媒体は仮想ユニットおよびブロックを含むため、フラッシュメモリデバイスに相互作用を及ぼすオペレーティングシステムのためのインターフェースとして作用する。たとえば、オペレーティングシステムは、仮想ブロックオフセットで仮想ブロックにデータを書込むための書込み命令を発する。次に、仮想ブロックを含む仮想ユニットが突き止められる。次いで、仮想マップが、データが実際に格納されるメモリの物理ユニット内部の対応する物理ブロックを突き止める。オペレーティングシステムは、仮想ユニットおよび仮想ブロックがフラッシュメモリの実際のハードウェアであるかのように、読出しおよび書込み命令を発するが、実は実際のハードウェアはフラッシュメモリの物理ユニットおよび物理ブロックに組み込まれている。したがって、オペレーティングシステムは、仮想ユニットおよびブロックを認識するのみであり、ハードウェア自体に直接相互作用を及ぼさない。

【００２５】

このようなインターフェースの利点は、仮想メモリを用いてオペレーティングシステムが相互作用を及ぼすことによって、さらに書込み可能となる前に消去しなければならないという要件などのフラッシュメモリに固有の欠点を、克服することができることである。さらに、電子デバイスのオペレーティングシステムにはフラッシュメモリのアドレスを構成する必要がない。さらに、１つのインターフェースがフラッシュメモリデバイスの多数の種類に使用されることができるた

め、オペレーティングシステムは、著しい修整の必要がなく、さまざまな異なるフラッシュメモリ技術に相互作用を及ぼすことができる。したがって、本発明の方法は、フラッシュメモリデバイスおよびそれらを使用する電子デバイスに最大の適応性を許容することができる。

【００２６】

【実施例】

本発明は、ＮＡＮＤまたはＡＮＤフラッシュデバイスなどのページモードフラッシュメモリデバイスを構成するためのシステムを提供する。このシステムは、フラッシュデバイスなどから読み出したり、フラッシュデバイスなどに書込んだりするための方法を含む。さらに、このシステムはまた、フラッシュデバイスが磁気ディスク記憶装置をうまくエミュレートすることができるようなインターフェースも提供する。パーソナルコンピュータまたはラップトップコンピュータなどさまざまなホストデバイスに、このようなフラッシュメモリデバイスを取付けることができる。

【００２７】

本発明は、フラッシュメモリに関して説明するが、その教えはまた、フラッシュメモリなどの同様の書込み、読出しおよびユニット消去の特性を備えたデータ記憶デバイスに応用可能であることを当業者は理解されたい。

【００２８】

本発明によるページモードフラッシュメモリデバイスを構成するためのシステムの原理および動作は、図面および添付する詳細を参照すれば、よりよく理解されるであろう。

【００２９】

ここで図面を参照すると、図１は、たとえば、ＮＡＮＤなどの技術による従来技術の物理的なページモードフラッシュメモリデバイスを模式的に示している。「物理的なデバイス」なる語は、以下、フラッシュメモリデバイスのための物理的な媒体を含む実際のハードウェア自体として定義される。このような物理的な媒体は一般に、フラッシュＥＥＰＲＯＭ装置から構成されるが、適切な不揮発性のプログラム可能なメモリデバイスのいずれで代用してもよい。「プログラム可

能な」なる語は、以下、たとえば、データをメモリデバイスに書込ませることによってなど、変更可能であるとして定義される。

【００３０】

フラッシュメモリ物理デバイス１０は、少なくとも１つの物理ユニット１２を備えているように示される。物理ユニット１２は、消去可能な物理デバイス１０の最小のセグメントである。物理ユニット１２は整数倍のブロックを含み、個別にブロック１〜ｎとして表され、ここでｎは整数であり、集合的にはブロック１４として表される。ブロック１４は、連続した固定長のグループの物理的なバイトアドレスから構成され、ハードウェアの特徴である。具体的に言えば、ブロック１４のサイズは物理デバイス１０の特性である。ブロック１４の基本データ領域１６に、利用者データを格納することができる。各ブロック１４はまた、制御データ領域１８も有する。制御データ領域１８は、ブロック１４の主要部分から個別アルゴリズムによってアドレス呼出しを行うことができ、ブロック１４のサイズの計算には含まれない。以下でさらに説明するように、制御データ領域１８は、フラッシュファイリングシステム自体に関連する情報の格納に好都合である。各物理ユニット１２は、配分済みのユニットか未配分のユニットのいずれかである。未配分ユニットはそれぞれ、自由であり、利用データを含まず、配分および割当てが行われる用意がなされている。各配分済みのユニットは割当てされ、データを含む。

【００３１】

図２は、基本的なフラッシュメモリデバイスを構成するためのシステムを示している。システム２０は、仮想媒体２２および物理デバイス１０の両方を制御し、仮想マップ２４によって仮想媒体２２を物理デバイス１０に関連付ける。仮想媒体２２は複数の仮想ユニット２６を含む。各仮想ユニット２６は複数の仮想ブロック２８を含む。各仮想ユニット２６は仮想アドレスによって表される。仮想アドレスは、特定の仮想ユニット２６を表す仮想ユニット番号を含む。各仮想ブロック２８は仮想ブロックオフセットによって表される。同様に、各物理ユニット１２は物理アドレスを有する。物理アドレスは、特定の物理ユニット１２を表す物理ユニット番号を含む。各物理ブロック１４は物理ブロックオフセットを有

する。

【００３２】

仮想マップ２４は、配分済みの仮想ユニットである仮想ユニット２６を配分済みの物理ユニットである少なくとも１つの物理ユニット１２にマッピングする。物理ユニットに関して、仮想ユニットが少なくとも１つの物理ユニットにマッピングされた場合には、それは配分済みの仮想ユニットである。しかし、１つの仮想ユニット２６は１つ以上の物理ユニット１２にマッピングされることができる。したがって、仮想ユニット２６と物理ユニット１２との対応は、１対１または１対多のいずれであってもよい。

【００３３】

システム２０は以下のように作動する。フラッシュメモリデバイス（図示せず）を含む電子デバイスのオペレーティングシステムは、読出し命令または書込み命令などの命令を特定の仮想ユニット２６内部の特定の仮想ブロック２８に送信する。次に、仮想マップ２４が物理ユニット１２内部の対応する物理ブロック１４を突き止める。

【００３４】

示されているように、厳密に１つの物理ユニット１２にマッピングされる各仮想ユニット２６に対して、マッピングの対応が１対１である場合には、状況は比較的単純であるように思える。しかし、上記のように、物理デバイス１０は、追加的なデータを書込むことができる前に、定期的な消去を実行するための要件を含め、読出しおよび書込みのための特定の物理的な制約条件がある。このような制約条件に対して、物理デバイス１０またはデバイスの一部で頻繁に繰り返される消去を伴わない可能な解決法は２通りである。

【００３５】

第１の解決法は図３Ａに示されており、仮想ユニット２６と物理ユニット１２との対応が１対多であるため、各仮想ユニット２６が複数の物理ユニット１２に対応する。仮想マップ２４は、このようなマッピングを実行するために必要な情報を保持しなければならない。仮想マップ２４の一部の例が図３Ａに挙げられており、本発明のＡＮＤシステムを適用することができる。

【００３６】

仮想マップ２４の部分は、物理ユニット１２内部の物理ブロック１４および仮想ユニット２６内部の仮想ブロック２８を示す。この例では、ある特定の仮想ユニット３０が２つの物理ユニット１２に対応する。第１の物理ユニット１２は基本ユニット３２である。第２の物理ユニット１２は置換ユニット３４である。各仮想ユニット２６の場合には、１つの基本ユニット３２のみであってもよい。しかし、各仮想ユニット２６に関連するゼロ以上の置換ユニット３４であってもよい。たとえば、仮想ユニット３６は基本ユニット３８にのみ対応し、置換ユニット３４には対応しないため、仮想ユニット３６は非置換型仮想ユニットの例である。

【００３７】

仮想ブロック２８の構成は、特定の仮想ユニット２６に対応する物理ブロック１４の数に依存する。仮想ユニット３０の場合には、複数の仮想ブロック２８が基本ユニット３２内部の物理ブロック１４に対応する一方、他の仮想ブロック２８が置換ユニット３４内部の物理ブロック１４に対応する。仮想ユニット３６の場合には、実質的にすべての仮想ブロック２８が基本ユニット３８内部の物理ブロック１４に対応する。

【００３８】

最も簡素な例において、仮想ユニットは非置換型ユニットであり、特定の物理ブロック１４を突き止めるための手順は以下の通りである。仮想ユニット３６は、仮想ユニット３６を表す仮想ユニット番号４４および仮想ブロック４２を表す仮想ブロックオフセット４６を有する。仮想ブロックオフセット４６も番号であることを留意されたい。物理ユニット番号５０は基本ユニット３８を表す。物理ブロックオフセット５２は基本ユニット３８内部の物理ブロック５４を表す。データの読出しまたは書込みを行うための物理ブロック５４を突き止めるために、第１の規則は、仮想ユニット番号４４を決定するために仮想ユニットごとのブロックの数によって、所望の仮想ブロックオフセット４６を割り振ることである。次に、仮想マップ２４は仮想ユニット番号４４を物理ユニット番号５０にマッピングする。第２の規則は、仮想ブロックオフセット４６と同一の番号でなければ

ならない物理ブロックオフセット５２によって、所望の物理ブロック１４、この場合には物理ブロック５４を物理ユニット３８の内部で突き止めることができることである。したがって、仮想マップ２４は、仮想および物理ユニットに関する情報を含むだけであるが、適正なブロックオフセットを決定するために規則が使用される。

【００３９】

さらに複雑な場合には、各仮想ユニットは１つ以上の物理ユニットに対応する。この場合には、２つ以上の物理ユニットのグループが「チェーン」と呼ばれる。たとえば、仮想ユニット番号７２は仮想ユニット３０を表し、仮想ブロックオフセット７４が仮想ブロック７０を表す。物理ユニット番号７８は置換ユニット３４を表し、物理ブロックオフセット８０は置換ユニット３４内部の物理ブロック８２を表す。したがって、仮想ユニット３０の仮想ブロック７０は置換ユニット３４の物理ブロック８２に対応する。

【００４０】

データの読出しまたは書込みを行うための物理ブロック８２を突き止めるために、再び第１の規則は、仮想ユニット番号７２を決定するために仮想ユニットごとのブロックの数によって、所望の仮想ブロックオフセット７４を割り振ることである。次に、仮想マップ２４は仮想ユニット番号７２を物理ユニット番号７８にマッピングする。しかし、問題がある。前述したように、第２の規則は、仮想ブロックオフセットと同一の番号でなければならない物理ブロックオフセットによって、所望の物理ブロックが物理ユニットの中で突き止められることである。この場合には、チェーンに複数の物理ブロック１４がある。いずれの物理ブロック１４がデータを有するかを決定するために、第３の規則は、仮想ブロック７０と同一のブロックオフセットを有する各物理ブロック１４が、チェーンの各物理ユニット内部にあるかを調査することである。最終非自由物理ブロック１４、この場合には置換ユニット３４の物理ブロック８２が、読み出し用の所望のデータを含む。逆に、書込みデータの場合には、第１の自由物理ブロック１４が所望のブロックである。

【００４１】

物理ブロックは属するチェーンにおいて物理ユニットの順に書込まれるため、「最終非自由物理ブロック」なる語は、未だ自由ではないが、チェーンの中で最も遠い下にあるユニットの物理ブロックを呼ぶ。チェーンにはユニットがそれ以上存在しないか、またはチェーンにおける次のユニットに同一のブロックオフセットを有する物理ブロックが自由であるかのいずれかである。同様に、第１の自由物理ブロックを発見するために、所望のブロックオフセットを有する各物理ブロックがチェーンの各物理ユニットにあるかどうかを調査し、この調査は基本ユニットから始めて、今度は各置換ユニットを通じて下に続き、自由ブロックが発見されるまで続く。

【００４２】

ＦＭＡＸと対照してみると、ＦＭＡＸは同様の仮想マップおよびアドレス指定システムを使用するが、図３Ｂに示すように、各基本ユニットには１つの置換ユニットのみを備える。これを実現するために、ＦＭＡＸは単一および複合の置換（物理）ユニットを使用する。単一置換ユニットは、物理ユニットの物理ブロックオフセットの実質的にすべてが対応する仮想ユニットの仮想ブロックオフセットに直接的に相関されるユニットである。複合置換ユニットは、仮想ブロックオフセットと物理ブロックオフセットとのこのような直接の対応関係が必ずしも存在しないユニットである。代わりに、対応する物理ブロックオフセットを有する物理ブロックが書込みに利用可能でない場合には、異なる物理ブロックが選択される。次に、仮想ブロックと物理ブロックとの実際の対応関係を決定するために、制御情報が制御データ領域に書込まれる。

【００４３】

図３Ｂに示されるように、基本ユニット９７は、複数の物理ブロック１００を有し、それぞれのブロックが仮想ユニット１０４の仮想ブロック１０２に対応する単一置換ユニット９８を有する。各物理ブロックオフセットは、同一のオフセット番号である仮想ブロックオフセットに対応する。

【００４４】

しかしながら、必要とする物理ブロックオフセットを有する物理ブロックが利用可能でない場合には、同一の物理ユニットの異なる物理ブロックが書込まれな

ければならず、置換ユニットは複合置換ユニットになる。第２の基本ユニット１０９は、複数の物理ブロック１１２を有し、それぞれのブロックが仮想ユニット１１６の仮想ブロック１１４に対応する複合物理ユニット１１０を有する。しかし、１つの物理ブロックオフセットが同一のオフセット番号である仮想ブロックオフセットに対応することができる一方、第２の物理ブロックオフセットは同一のオフセット番号でない第２の仮想ブロックオフセットに対応してもよい。特定の物理ブロックを見つけるために、制御データ領域に書込まれた制御情報を調査しなければならない。以下にさらに説明するように、これは、データの書込みの場合および必要に応じてＦＭＡＸシステムを再構成する場合の両方の場合において、きわめて重要である。

【００４５】

図４Ａは図３Ａの仮想マップを操作するためのフローチャートを示し、図４Ｂは図３Ｂの仮想マップを操作するためのフローチャートを示す。最も簡素な場合、すなわちすべての置換ユニットが単一ユニットまたは１つのみの置換ユニットを備える基本ユニットである場合には、ＡＮＤおよびＦＭＡＸのいずれも同じステップを使用することができる。まず、突き止められる対象の仮想ブロックの数を、仮想ユニット番号を与える仮想ユニットごとのブロックの数で割ることによって、仮想ユニット番号および仮想ブロックオフセットが計算される。法または割算の剰余が仮想ブロックオフセットである。

【００４６】

次に、仮想マップが、仮想ユニットに対応する物理ユニットを発見するために調査される。仮想ユニットに対応する物理ユニットを発見することができない場合には、物理メモリの必要な部分はフラッシュデバイスに存在しない。上記のように、すべての置換ユニットが単一ユニットであるか、または基本ユニットが唯一の置換ユニットを有する場合にのみ、このような単一の方式が有効である。しかし、データが書込まれることになっている物理ブロックがすでにプログラムされているか、または他のデータで書込まれている場合には、この方式は作用しない。この場合には、データを書込むことができる別の物理ブロックを発見するタスクを処理することができるような置換方式が必要とされる。

【００４７】

２通りの異なるアルゴリズムが、図４Ａ（ＡＮＡＮＤ）および図４Ｂ（ＦＭＡＸ）に示されている。両方のアルゴリズムは同一の方式で始まる。ステップ１において、所望の物理ユニットが突き止められる。ステップ２において、特定のブロックオフセットに対応する物理ブロックが、その物理ユニットの内部で突き止められる。ステップ３において、ブロックが未書込みの場合には、データがブロックに書込まれる。所望の物理ブロックが利用可能でない場合には、本発明の２つのシステム、すなわちＡＮＤおよびＦＭＡＸは、各技術が所望の物理ブロックがすでに書込まれた状況に対処するような方法で分岐される。

【００４８】

図４Ａに示されるように、ＡＮＤシステムは、置換ユニットを見ることによってこの状況に対処する。ステップ４において、ｘ番目の置換物理ユニットが調査される。ここでｘは、最初は１に等しい整数である。その物理ユニットが所望の物理ブロックオフセットを備えた未書込みの物理ブロックを有する場合には、データが物理ブロックに書込まれる。ブロックが利用可能でない場合には、ステップ５に示されるように、ｘは１ずつ増分され、ステップ４が反復される。データがブロックに書込まれるか、またはチェーンの他の置換ユニットが発見されなくなるまで、ステップ４および５が反復される。ステップ６において、未配分の物理ユニットが置換ユニットとして配分され、データが所望のブロックオフセットを備えたブロックに書込まれる。

【００４９】

ＦＭＡＸシステムは、図４Ｂに示されるように、この状況に異なる方法で対処する。ステップ４において、置換ユニットにおける同一の物理ブロックオフセットを有する物理ブロックが突き止められる。その物理ブロックが未書込みの場合には、データがその物理ブロックに書込まれる。そうでない場合には、ステップ５のように、置換ユニットの中の異なる物理ブロックオフセットを備えた物理ブロックが突き止められる。未書込みの物理ブロックが突き止められるまで、ステップ５が反復される。今度は、仮想ブロックオフセットがもはや物理ブロックオフセットと同一でないため、置換ユニットが複合ユニットである。ステップ６に

おいて、マッピング方式が複合ユニット内部のいかなる物理ブロックの正確な位置も発見できるようにするために、制御情報が物理ユニットの制御データ領域に付加される。

【００５０】

しかしながら、これらの置換アルゴリズムも、フラッシュデバイスの異なる要求のすべてに対処するのに十分でないと思われる。ＡＮＤおよびＦＭＡＸシステムの両方とも、最後には物理ブロックが利用可能でないため、物理ユニット内部のブロックにさらなるデータを書込むことができない状況に達するであろう。

【００５１】

このような状況において、データを最も簡素な状態、すなわち非置換基本ユニットに再構築するために、仮想ユニットを再構成しなければならない。この再構成処理中、以前に仮想ユニット表示が属していた物理置換ユニットが解放され、それによって、割当てられていないまたは自由な物理ユニットとなる。ＡＮＤ置換ユニットおよび単一ＦＭＡＸ置換ユニットの両方に関して、この再構成処理は、フォールディングと呼ばれ、以下の図５Ａに図示される。

【００５２】

フォールディングは、置換ユニットにおいて基本ユニットにおいて書込まれたのと同一の物理ブロックオフセットで書込まれる対象の物理ブロックを必要とする。その理由については、処理が説明されるとさらに明らかになるであろう。フォールディングの第１のステップにおいて、チェーンの最終物理ユニットが物理ユニットｘと識別される。ここで、ｘは１からいくつかの予め決定された実装依存制限数までの整数である。ｘが１に等しい場合、置換ユニットが実際に基本ユニットであり、残りのアルゴリズムは実行されないことに留意されたい。また、ＦＭＡＸの場合には、ｘは１または２に等しいことにも留意されたい。

【００５３】

ステップ２において、ユニットｘのブロックｎが調査される。ここでｎは整数である。データがブロックｎに書込まれている場合には、ｎは１ずつ増分される。そうでない場合には、ステップ３においてｘが１ずつ減分される。ｘが０に等しいかまたは書込み済みのブロックｎが発見されるかのいずれかになるまで、ス

テップ２および３が反復される。書込み済みのブロックｎが発見された場合には、ステップ４において、データがチェーンの最終置換ユニットのブロックｎまで移動される。すべてのデータが次に基本ユニットとなる最終置換ユニットに移動されるまで、ステップ２～４が反復される。次に、事前の基本ユニットを含めるにしてもチェーンの他のすべてのユニットが解放され、割当てに利用可能となる。今度は仮想ユニットが１つの物理ユニットに対応するという事実を反映するために、仮想マップも更新される。

【００５４】

残念なことに、置換ユニット内部のブロックが常に、仮想ブロックオフセットに等しい物理ブロックオフセットを持っているとは限らないため、フォールディングは、複合ＦＭＡＸ置換ユニットの場合には作用しない。再割当ての異なる処理が、図５Ｂの複合物理ユニットのために示されている。ステップ１において、

新たな未割当ての物理ユニットが、新たな基本物理ユニットと呼ばれる。ステップ２において、複合物理ユニットのブロックｎが調査される。データが複合物理ユニットのブロックｎに書込まれている場合には、ステップ３において、データが新たな基本ユニットにコピーされる。そうでない場合には、古い基本ユニットのブロックｎからデータが新たな基本ユニットに書込まれる。ステップ４において、ｎが１ずつ増分される。すべてのブロックがコピーされるまで、ステップ２～４が反復される。一旦、ブロックのすべてがコピーされると、古い基本ユニットのほか以前の置換ユニットも解放され、割当てのために利用可能となる。以前の手順のように、今度は仮想ユニットが１つのみの物理ユニットに対応するという事実を反映するために、仮想マップが更新される。

【００５５】

再構成方式のきわめて簡略化した実施例も可能である。この簡略化した実施例において、置換ユニットが割当てられた直後に再構成の処理が行われる。したがって、置換ユニットはシステムの一時的な特徴にすぎず、静止状態、すなわち物理メモリが書込み処理を実行していない状態において、データは基本非置換ユニットにのみ存在する。置換ユニットは、書込み処理のためだけに存在する。処理の終了時に、情報のすべてが新たなユニットに移動されるため、置換ユニットは

実質的に消失する。この方法は、実装の容易さおよびそれを管理するために必要とされる制御構造の簡便さという利点を備える。しかし、その欠点は、この方法が効率的ではないため、システムの書込み性能を低下させることにある。

【００５６】

本発明に含まれる方法のすべては、格納されるデータの状態を記述するために、物理的なフラッシュデバイス自体に制御情報を記録することが可能でなければならない。特に、ユニットおよびブロックの制御情報は格納されることが好ましいが、別法としてそのようなデータは他の種類のデータから再構築することもできる。ユニット制御情報は、物理ユニットに配分された物理ユニット番号、基本または置換ユニットとしての物理ユニット自体の状態および他のユニットに対するそのユニットの位置を表示する。ブロック制御情報は、物理ブロックが使用されているか、解放されているかまたは異なる物理ブロックに存在する情報によって取り替えられているかどうかを表示する。

【００５７】

これらの異なる種類の１つまたは両方の情報を物理デバイスの特別な部分に、記録することができる。図１で上記のように、好ましくは、ＡＮＤおよびＦＭＡＸシステムは各物理ユニット１２を物理的なフラッシュデバイスに記録される実際の利用者データを含む基本データ領域１６および制御情報を含む制御データ領域１８に分割する。このような領域は、ブロック１６の下位区分として示されるが、物理ユニット１２はまた、ブロックへの分割に実質的に独立である基本データ領域および制御データ領域に分割されてもよい。制御データ領域１８は基本データ領域１６のブロック探索方式の中に含まれず、物理的なフラッシュディスクの全体サイズを計算する場合にも含まれないことに留意すべきである。

【００５８】

ＮＡＮＤおよびＡＮＤフラッシュ技術は、メモリの各ブロックに空白領域を有するため、制御情報が通常、ブロックの空白領域に記録され、利用者データが基本ブロック領域に配置される。

【００５９】

空白領域が設けられていないフラッシュ技術の場合には、利用者データを格納

するための主要領域および必要な制御情報を格納するためのオーバヘッド部分に、すべての物理ユニットを分割することができる。

【００６０】

上記の説明は例として使用することのみを目的とし、さまざまな他の実施例が本発明の精神および範囲の中で可能であることを認識されたい。

【図面の簡単な説明】

本発明は、添付図面に関して、例としてのみここには説明される。

【図１】　本発明による物理的なフラッシュメモリデバイスの概略図である。

【図２】　本発明によるフラッシュメモリデバイスを構成する基本システムの図である。

【図３Ａ】　本発明によるＡＮＤシステムを示す。

【図３Ｂ】　本発明によるＦＭＡＸシステムを示す。

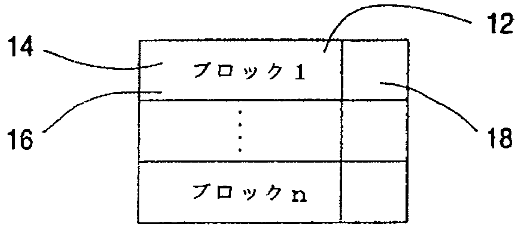
【図４Ａ】　本発明によるＡＮＤシステムのための書込みアルゴリズムを示す。

【図４Ｂ】　本発明によるＦＭＡＸシステムのための書込みアルゴリズムを示す。

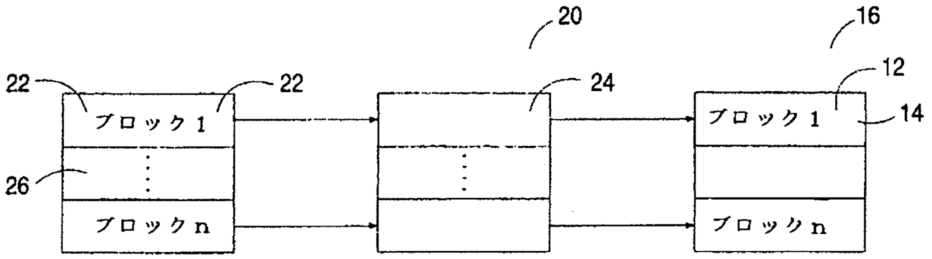
【図５Ａ】　本発明によるＡＮＤシステムのための再構成アルゴリズムを示す。

【図５Ｂ】　本発明によるＦＭＡＸシステムのための再構成アルゴリズムを示す。

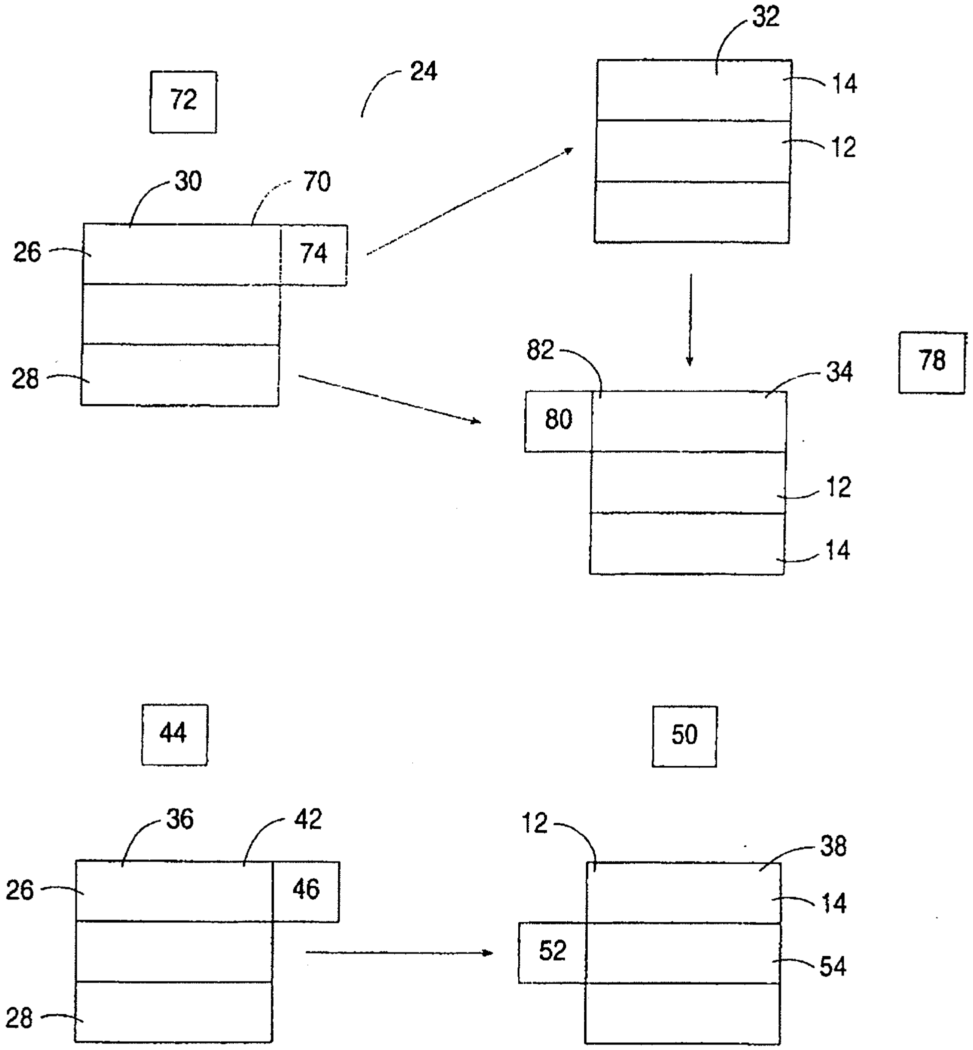
【図１】



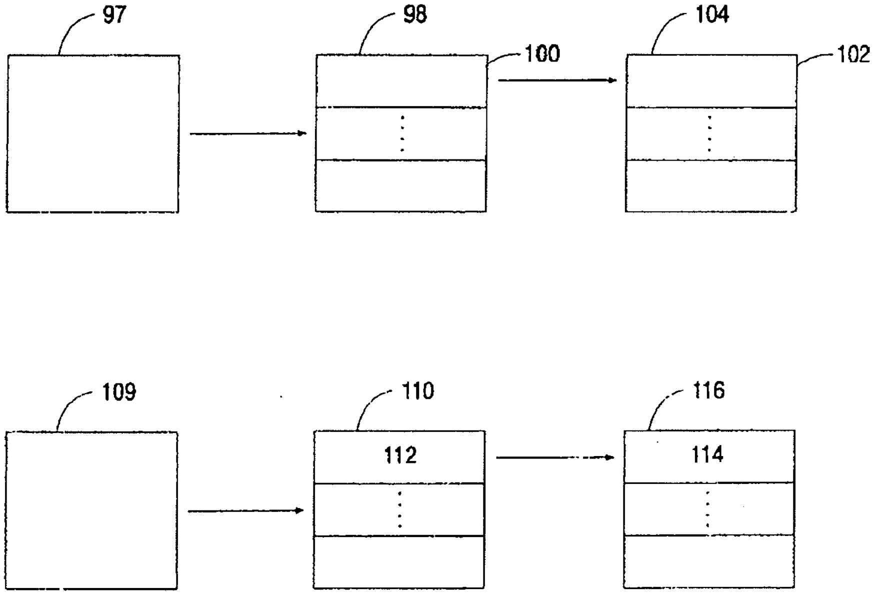
【図２】


ブロック１
ブロック１
ブロックｎ
ブロックｎ

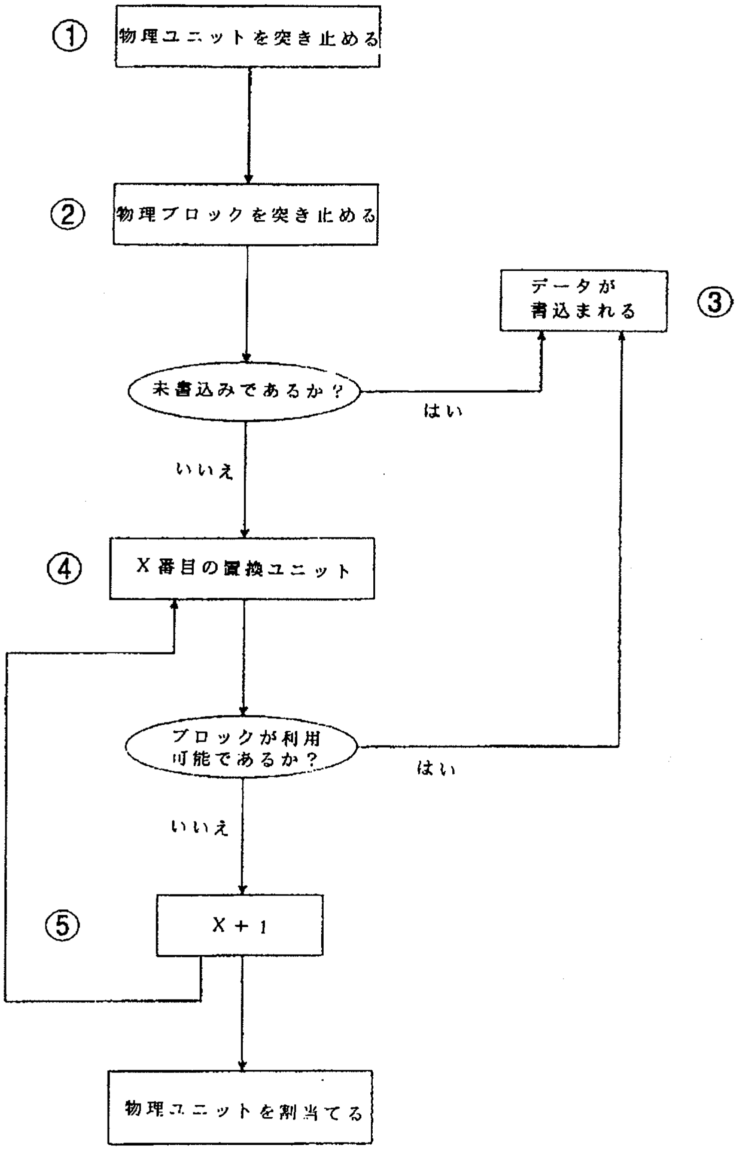
【図３Ａ】



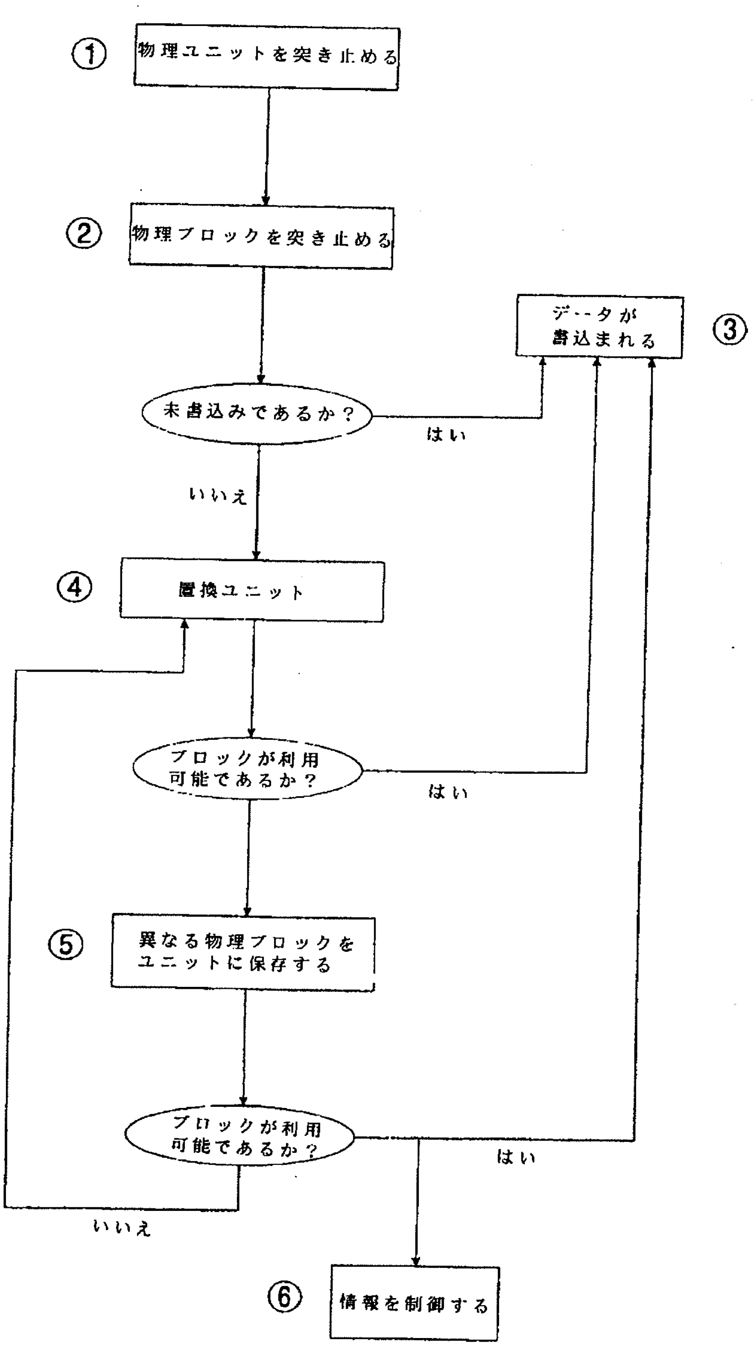
【図３Ｂ】


97
98
100
104
102
109
110
112
116
114

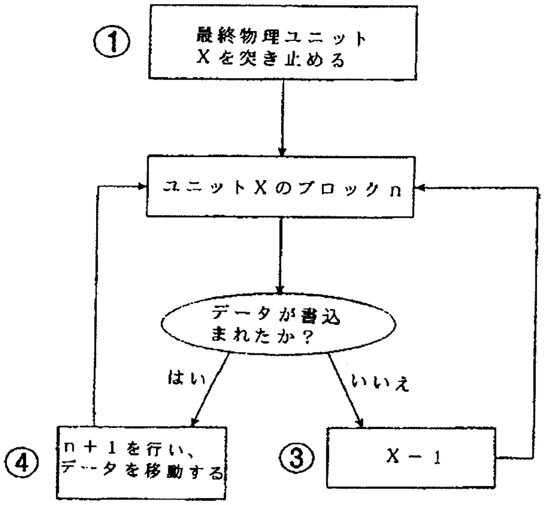
【図４Ａ】


①
物理ユニットを突き止める
②
物理ブロックを突き止める
データが書込まれる
③
未書込みであるか？
はい
いいえ
④
X番目の置換ユニット
ブロックが利用可能であるか？
はい
いいえ
⑤
X＋１
物理ユニットを割当てる

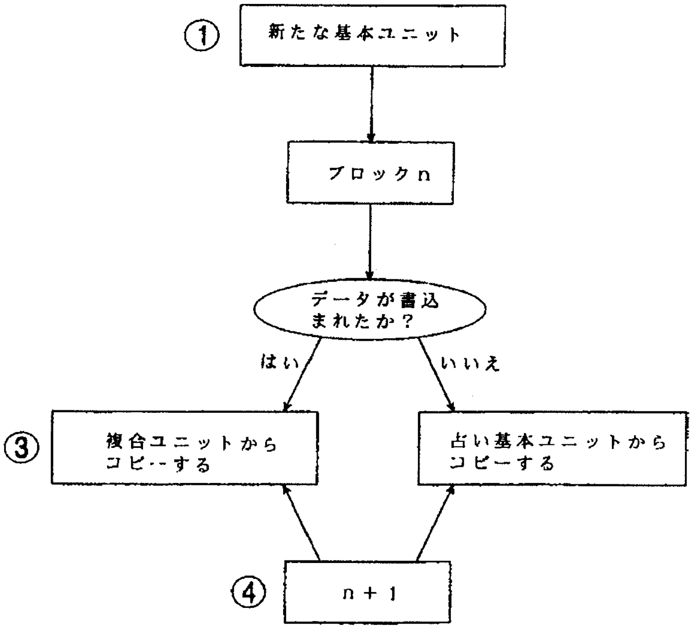
【図４Ｂ】


①
物理ユニットを突き止める
②
物理ブロックを突き止める
③
データが書込まれる
未書込みであるか？
はい
いいえ
④
置換ユニット
ブロックが利用可能であるか？
はい
⑤
異なる物理ブロックをユニットに保存する
ブロックが利用可能であるか？
はい
いいえ
⑥
情報を制御する

【図５Ａ】


①
最終物理ユニット
Xを突き止める
ユニットXのブロックn
データが書込
まれたか？
はい
いいえ
④
n+1を行い、
データを移動する
③
X－1

【図５Ｂ】


①
新たな基本ユニット
ブロックn
データが書込
まれたか？
はい
いいえ
③
複合ユニットから
コピーする
古い基本ユニットから
コピーする
④
n+1

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/21017

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 12/12
US CL : 711/103, 165, 202, 209
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)

U.S. : 711/103, 165, 202, 209

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 practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,404,485 A (BAN) 04 April 1995, column 2 (all), column 4 lines 11-66, column 5 line 36 to column 6 line 27.	1-4, 7-8, 10-11
Y	US 5,479,638 A (ASSAR et al) 26 December 1995, column 4 lines 1-58, column 5 line 53 to column 6 line 11.	1-4, 7-8, 10-11
Y	US 5,459,850 A (CLAY et al) 17 October 1995, column 3 lines 9-24, column 18 line 59 to column 21 line 44.	1-4, 7-11, 14-15
A	US 5,630,093 A (HOLZHAMMER et al) 13 May 1997.	1-15
A	US 5,644,539 A (YAMAGAMI et al) 01 July 1997.	1-15

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"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
"A"	document defining the general state of the art which is not considered to be of particular relevance		
"E"	earlier document published on or after the international filing date	"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
"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)	"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
"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	"&"	document member of the same patent family

Date of the actual completion of the international search 15 MARCH 1999	Date of mailing of the international search report 05 APR 1999
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer EDDIE P. CHAN Telephone No. (703) 305-3900

Form PCT/ISA/210 (second sheet)(July 1992)★

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, KE, LS, MW, SD, SZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW

(71)出願人 Building 7、Atidim Industrial Park、P. O. Box 58036、61580 Tel Aviv、Israel

Fターム(参考) 5B060 AB25 BA13
5B082 FA04 JA07

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-250798

(43)Date of publication of application : 09.09.1994

---

(51)Int.Cl. G06F 3/08
G11C 16/06

---

(21)Application number : 05-035228

(22)Date of filing : 24.02.1993

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>

(72)Inventor : NIIJIMA HIDETO
ASANO HIDEO
SAKAGAMI YOSHIISA
TOYOOKA TAKASHI

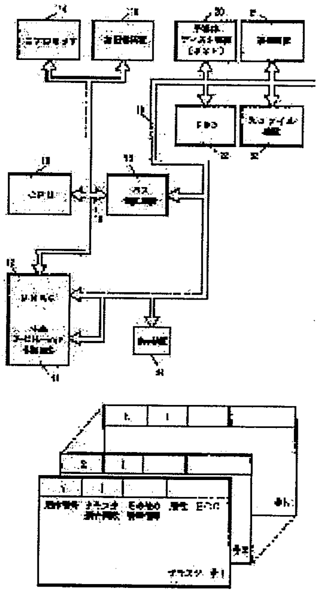
---

(54) BATCH DELETE TYPE NONVOLATILE MEMORY AND SEMICONDUCTOR DISK DEVICE USING THE SAME

(57)Abstract:

PURPOSE: To discriminate an invalid sector from a valid sector without using a method for overwriting by ensuring a cluster information sector in each cluster, applying a sequence number to the cluster so as not to be doubled, and writing the sequence number assigned to the cluster in the cluster information sector of each cluster.

CONSTITUTION: A batch delete type nonvolatile memory 20 can be deleted by each cluster unit, a cluster information sector is ensured in each of the N pieces of clusters, a sequence number is preliminarily applied to the N pieces of clusters so as not to be overlapped, and the sequence number of the cluster is written in the cluster information sector of each cluster. At the time of deleting a certain cluster, a controller 30 maintains the sequence number of the cluster at first, and at the time of initializing the deleted cluster, the controller 30 writes a value larger than the present maximum sequence number in the cluster information sector as the sequence number of the cluster. The controller 30 writes user data according to the sequence number of the physical address of the sector.



LEGAL STATUS

[Date of request for examination] 28.11.1996

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 2856621

[Date of registration] 27.11.1998

[Number of appeal against examiner's decision of

rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	27.11.2002

(19)日本国特許庁（ＪＰ）　　(12) **公開特許公報**（Ａ）　　(11)特許出願公開番号

特開平6－250798

(43)公開日　平成６年(1994) ９月９日

(51)Int.Cl.^{5}	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　3/08	Ｈ	7165－5B		
Ｇ１１Ｃ　16/06				
		6866－5L	Ｇ１１Ｃ　17/ 00　　３０９　Ｃ	

審査請求　未請求　請求項の数15　ＯＬ　（全 16 頁）

(21)出願番号　特願平5－35228

(22)出願日　平成５年(1993) ２月24日

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州アーモンク　（番地なし）

(72)発明者　新島　秀人
東京都千代田区三番町５－19　日本アイ・ビー・エム株式会社　東京基礎研究所内

(74)代理人　弁理士　頓宮　孝一　（外４名）

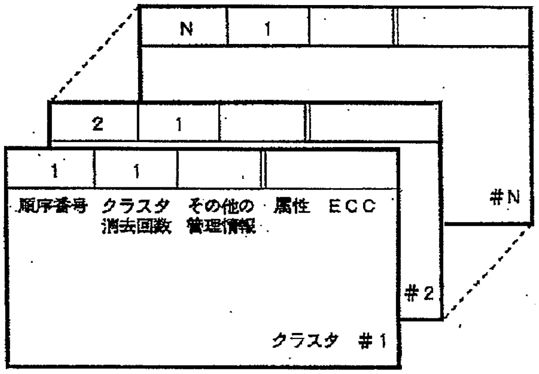
最終頁に続く

(54)【発明の名称】　一括消去型不揮発性メモリおよびそれを用いる半導体ディスク装置

(57)【要約】

【目的】重ね書きの手法を用いなくても無効セクタと有効セクタとを見分けることが可能な一括消去型不揮発性メモリ及びそれを用いた半導体ディスク装置を提供すること。

【構成】一括消去型不揮発性メモリは、クラスタ単位で消去することが可能であり、Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、予めＮ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれている。コントローラは、所与のクラスタを消去するとき、先に当該クラスタの順序番号を保管する。そして、所与の消去済みクラスタを初期設定するときには、現在の最大順序番号よりも大きな値を当該クラスタの順序番号としてそのクラスタ情報セクタに書き込む。このようにして初期設定されたクラスタのクラスタ情報セクタ以外のセクタに対して、コントローラは、それらセクタの物理アドレスの順番に従ってユーザ・データを書き込む。



【特許請求の範囲】

【請求項１】各々がM個のセクタからなるＮ個のクラスタを有し（M、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能であり、

上記Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、

上記Ｎ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれていることを特徴とする一括消去型不揮発性メモリ。

【請求項２】上記Ｎ個のクラスタ情報セクタの各々にはさらに当該クラスタの消去回数が書き込まれ、上記Ｎ個のクラスタに与えられた順序番号の最大値はそれらクラスタの消去回数の総和に等しいことを特徴とする請求項１記載の一括消去型不揮発性メモリ。

【請求項３】コントローラと、

各々がM個のセクタからなるＮ個のクラスタを有し（M、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能な一括消去型不揮発性メモリとを備え、

上記Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、

上記コントローラの動作に先立って、上記Ｎ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれており、

上記コントローラは、

所与のクラスタを消去するときは、先に当該所与のクラスタの順序番号を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号よりも大きな値を当該所与のクラスタの順序番号としてそのクラスタ情報セクタに書き込むことを特徴とする半導体ディスク装置。

【請求項４】上記コントローラは、所与のクラスタに対してユーザ・データの書込みを開始した後、当該クラスタに対するユーザ・データの書込みが終了するまで、他のクラスタへのユーザ・データの書込みを行わないことを特徴とする請求項３記載の半導体ディスク装置。

【請求項５】上記コントローラは、上記所与のクラスタのクラスタ情報セクタ以外のセクタに対してそのアドレスの順番に従ってユーザ・データを書き込む請求項３又は４記載の半導体ディスク装置。

【請求項６】コントローラと、

各々がM個のセクタからなるＮ個のクラスタを有し（M、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能な一括消去型不揮発性メモリとを備え、

上記Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、

上記コントローラの動作に先立って、上記Ｎ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当て

られた順序番号が書き込まれており、且つ、上記Ｎ個のクラスタ情報セクタの各々にはさらに当該クラスタの消去回数が書き込まれ、上記Ｎ個のクラスタに与えられた順序番号の最大値はそれらクラスタの消去回数の総和に等しく設定されており、

上記コントローラは、

所与のクラスタを消去するときは、先に当該所与のクラスタの順序番号と消去回数を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号に１を加算し、且つ消去前に保管された消去回数に１を加算し、加算結果をそれぞれ当該所与のクラスタの順序番号及び消去回数としてそのクラスタ情報セクタに書き込むことを特徴とする半導体ディスク装置。

【請求項７】プロセッサからの論理アドレスによってアクセスされる半導体ディスク装置であって、

上記プロセッサに接続されたコントローラと、

各々がM個のセクタからなるＮ個のクラスタを有し（M、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能な一括消去型不揮発性メモリと、

上記コントローラに接続されたランダム・アクセス・メモリとを備え、

上記一括消去型不揮発性メモリは、

上記Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、

上記コントローラの動作に先立って、上記Ｎ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれており、

上記コントローラは、

上記プロセッサのコマンドに含まれる論理アドレスを特定のセクタを指示する物理アドレスに変換するために上記ランダム・アクセス・メモリ上にアドレス変換表の領域を確保し、

上記プロセッサが所与の論理アドレスを指定して書込みを要求したときに、上記一括消去型不揮発性メモリの空白セクタを一つ選択し、上記アドレス変換表の当該所与の論理アドレスによって指示される項目に当該選択されたセクタの物理アドレスを書き込み、且つ当該選択されたセクタに当該所与の論理アドレスを逆参照ポインタとして書き込み、

所与のクラスタを消去するときは、先に当該クラスタの順序番号を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号よりも大きな値を当該クラスタの順序番号としてそのクラスタ情報セクタに書き込むことを特徴とする半導体ディスク装置。

【請求項８】上記コントローラは、所与のクラスタに対して書込みを開始した後、当該所与のクラスタに対する書込みが終了するまでの期間、上記プロセッサの書き込

み要求に応答して当該所与のクラスタから物理アドレスの順に空白セクタを選択することを特徴とする請求項７記載の半導体ディスク装置。

【請求項９】上記コントローラは、上記アドレス変換表を再構成するときに、上記不揮発性メモリのM×N個のセクタを読み出し、読み出されたセクタの逆参照ポインタによって指示される上記アドレス変換表の項目に当該セクタの物理アドレスを書き込み、

同じ逆参照ポインタを持つセクタが複数あるときには、それらセクタの属するクラスタの順序番号とクラスタ内での位置に従って、最も新しく書き込まれたセクタの物理アドレスを上記アドレス変換表に書き込むことを特徴とする請求項７又は８記載の半導体ディスク装置。

【請求項１０】上記コントローラは、上記アドレス変換表を再構成するときに、（ａ）上記M個のクラスタのクラスタ情報セクタを読み出し、（ｂ）順序番号が最小のクラスタを選択し、（ｃ）選択されたクラスタのセクタを順次読み出し、読み出されたセクタの逆参照ポインタによって指示される上記アドレス変換表の項目に当該セクタの物理アドレスを書き込み、（ｄ）順序番号の昇順に従って次のクラスタを選択し上記（ｃ）の動作を行うことを反復することを特徴とする請求項７又は８記載の半導体ディスク装置。

【請求項１１】上記コントローラは、上記アドレス変換表を再構成するときに、上記M個のクラスタの各々について、

当該クラスタのクラスタ情報セクタを読み出し、

当該クラスタのアドレスとその順序番号の対応関係を上記ランダム・アクセス・メモリに表形式で記憶し（以下、この表を順序番号表と呼ぶ）、

当該クラスタのクラスタ情報セクタ以外のセクタ（以下、データ・セクタと呼ぶ）を順次読み出し、

読み出されたデータ・セクタの逆参照ポインタによって指示されるアドレス変換表の項目を読み出し、

当該項目が空白であるならば、そこに当該読み出されたデータ・セクタの物理アドレスを書き込み、

当該項目が空白でないならば、そこに書き込まれた物理アドレスに位置するセクタの属するクラスタの順序番号を上記順序番号表を参照して求め、

上記求まった順序番号を現在読み出し中のクラスタの順序番号と比較し、

上記求まった順序番号の方が小であるなら、上記読み出されたデータ・セクタの物理アドレスを上記項目に書き込むことを特徴とする請求項７又は８記載の半導体ディスク装置。

【請求項１２】プロセッサと、

上記プロセッサに接続されたコントローラと、

各々がM個のセクタからなるN個のクラスタを有し（M、Nは各々２以上の整数）、クラスタ単位で消去することが可能な一括消去型不揮発性メモリと、

上記コントローラに接続されたランダム・アクセス・メモリとを備え、

上記一括消去型不揮発性メモリは、

上記N個のクラスタの各々にクラスタ情報セクタが確保され、

上記コントローラの動作に先立って、上記N個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれており、

上記コントローラは、

上記プロセッサのコマンドに含まれる論理アドレスを特定のセクタを指示する物理アドレスに変換するために上記ランダム・アクセス・メモリ上にアドレス変換表の領域を作成し、

上記プロセッサが所与の論理アドレスを指定して書込みを要求したときに、上記一括消去型不揮発性メモリの空白セクタを一つ選択し、上記アドレス変換表の当該所与の論理アドレスによって指示される項目に当該選択されたセクタの物理アドレスを書き込み、且つ当該選択されたセクタに当該所与の論理アドレスを逆参照ポインタとして書き込み、

所与のクラスタを消去するときは、先に当該クラスタの順序番号を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号よりも大きな値を当該クラスタの順序番号としてそのクラスタ情報セクタに書き込むことを特徴とするデータ処理システム。

【請求項１３】プロセッサと、

上記プロセッサに接続された表示装置と、

上記プロセッサに接続されたコントローラと、

各々がM個のセクタからなるN個のクラスタを有し（M、Nは各々２以上の整数）、クラスタ単位で消去することが可能な一括消去型不揮発性メモリとを備え、

上記N個のクラスタの各々にクラスタ情報セクタが確保され、

上記コントローラの動作に先立って、上記N個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれており、且つ、上記N個のクラスタ情報セクタの各々にはさらに当該クラスタの消去回数が書き込まれ、上記N個のクラスタに与えられた順序番号の最大値はそれらクラスタの消去回数の総和に等しく設定されており、

上記コントローラは、

上記プロセッサのコマンドに含まれる論理アドレスを特定のセクタを指示する物理アドレスに変換するために上記ランダム・アクセス・メモリ上にアドレス変換表を作成し、

所与のクラスタを消去するときは、先に当該所与のクラスタの順序番号と消去回数を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号に１を加算し、且つ消去前に保管された消去回数に１を加算し、それぞれの加算結果を当該所与のクラスタ順序番号及び消去回数としてそのクラスタ情報セクタに書き込み、

上記アドレス変換表作成時に最大順序番号と上記Ｎ個のクラスタのクラスタ情報セクタに記憶された消去回数の総和を比較し、不一致を検出したときには、上記表示装置にエラー・メッセージを表示させるように、上記プロセッサに対して要求することを特徴とするデータ処理システム。

【請求項１４】各々がＭ個のセクタからなるＮ個のクラスタを有し（Ｍ、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能な、一括消去型不揮発性メモリを用いた半導体ディスク装置の管理方法であって、

上記Ｎ個のクラスタに対して重複がないように順序番号を与え、各クラスタにその割り当てられた順序番号を書き込み、

所与のクラスタを消去するときは、当該クラスタの順序番号を保管した後、当該クラスタを消去し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号よりも大きな値を当該クラスタの順序番号としてそのクラスタに書き込むことを特徴とする半導体ディスク装置の管理方法。

【請求項１５】各々がＭ個のセクタからなるＮ個のクラスタを有し（Ｍ、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能な、一括消去型不揮発性メモリを用いた半導体ディスク装置の管理方法であって、

上記Ｎ個のクラスタに対して重複がないように順序番号を与え、クラスタの夫々にその割り当てられた順序番号と消去回数とを書き込み、上記Ｎ個のクラスタに与えられた順序番号の最大値はそれらクラスタの消去回数の総和に等しく設定し、

所与のクラスタを消去するときは、先に当該クラスタの順序番号と消去回数を保管し、

所与の消去済みクラスタを初期設定するときは、現在の最大順序番号に１を加算し、且つ消去前に保管された消去回数に１を加算し、加算結果をそれぞれ当該所与のクラスタの順序番号及び消去回数としてそのクラスタに書き込むことを特徴とする半導体ディスク装置の管理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、フラッシュＥＥＰＲＯＭ（以下ではフラッシュ・メモリと呼ぶ）等の一括消去型不揮発性メモリ及びそれを用いる半導体ディスク装置に係り、特に、動的セクタ割当ての可能な半導体ディスク装置に係る。

【０００２】

【従来の技術】ノートブック等の携帯可能なパーソナル

・コンピュータの普及に伴って、コンピュータ・システムの小型軽量化、低消費電力化に対する要求が強くなってきている。半導体メモリを用いた外部記憶システム、いわゆる半導体ディスク装置は、磁気ディスク装置のように駆動系を持たないため、消費電力が低く、高速動作が可能である。また、小さなメモリ・モジュールで構成されるため、磁気ディスク装置に比べて小型で軽く、形状に自由度が大きく、カード化も容易である。

【０００３】しかし、従来の半導体メモリにはコスト、容量、電池バックアップなどの点でまだ問題が多い。メモリとしてＳＲＡＭを使うと電池によるバックアップの時間は長くなるが、コストが高く、容量も小さくなってしまう。コスト、容量に優れたＤＲＡＭでは、スタンバイ時の消費電力が大きく、バックアップの時間が一週間程度に限られてしまう。電池系の事故によるデータ消失の危険もある。ＥＥＰＲＯＭは電池を必要としないが、コストが高すぎる。

【０００４】これらの問題を解決するメモリとして一括消去型のフラッシュ・メモリが開発されている。ＤＲＡＭと同じくトランジスタ１つで記憶素子が構成され、高密度化が可能で、将来の市場次第でＤＲＡＭと同等かそれ以下のビット単価（低コスト、大容量）になることが期待されている。記憶素子は不揮発性であり、電池バックアップの必要はない。消去は一般にチップ単位又はそれよりも小さなブロック単位で行われる。Richard D. Pashley 外の "Flash memories: the best of two worlds"、IEEE SPECTRUM 1989年12月、30～33頁は、このようなフラッシュ・メモリの概要を紹介している。パフォーマンスの点では、チップ消去型よりブロック消去型の方が優れている。

【０００５】ブロック消去型のフラッシュ・メモリを半導体ディスク装置に用いる場合は、ブロックの大きさをハード・ディスク装置のアクセス単位であるセクタに等しくしておくと、メモリ管理に都合がよい。例えば、ヨーロッパ公開特許出願第３９２８９５号はセクタ消去型のフラッシュＥＥＰＲＯＭシステムを開示している。このシステムは、消去単位であるセクタ毎にラッチを設けておき、消去したいセクタに対応するラッチをセットすることによって、任意の複数のセクタを同時に消去できるようにしている。複数セクタ分の容量（例えば４Ｋバイト）を持ったブロックを消去単位にしたフラッシュ・メモリも知られている。

【０００６】しかし、フラッシュ・メモリにはＳＲＡＭやＤＲＡＭにはない制限がある。まず、メモリ・ビットのプログラミングは一方通行で、０から１又は１から０へしか変えることができない。従って、既に書込まれている記憶位置に新たなデータを書込む場合は、その記憶位置を含むブロックを一括消去によって全０又は全１に設定した後に書込みを行う必要がある。消去及び書込みには、通常、数十ミリ秒から数秒の時間がかかる。ま

た、フラッシュ・メモリは消去及び書込みによって劣化し、現在のところ、数万回から数十万回の消去及び書込みで使用限度に達してしまう。

【０００７】このようなフラッシュ・メモリを半導体ディスク装置に用いた場合、同一の論理セクタを同一の物理セクタに割り当てていたのでは、書込みがメモリの一部に片寄ることが問題になる。例えば、ＤＯＳベースのパーソナル・コンピュータ・システムでは、ファイル・アロケーション・テーブル（ＦＡＴ）の書換えがしばしば行われる。ところが、ＦＡＴのアドレスは固定されているため、フラッシュ・メモリを用いた場合には、ＦＡＴの書換えの度にそれを記憶しているブロックの消去及び書込みを行う必要があり、その度に数十ミリ秒から数秒の時間がかかってしまう。また、このように書込み及び消去がメモリの一部のブロックに片寄ると、そのブロックが他のブロックに比べて早く使用限度に達してしまい、他のブロックがまだ使用可能であるにもかかわらず、メモリを交換しなければならなくなる。使用限度に達したブロックを無効化して、代替ブロックを使用するようにすれば、メモリの早期交換は避けられるが、書込みが集中するのが代替ブロックに変わっただけであり、根本的な解決にはなっていない。

【０００８】そこで、特願平３－１９７３１８号では、動的セクタ割当て法を用いることにより、この問題の解決に成功している。図１と図２を参照して、その概要を説明する。ＲＡＭにアドレス変換表が作成され、これを参照することにより、ホスト・プロセッサの指定するアドレス（論理アドレス）が半導体ディスク装置（ＳＳＦ）のセクタ（物理セクタ）を指定するアドレス（物理アドレス）に変換される。即ち、ホスト・プロセッサは、ヘッド番号、シリンダ番号、セクタ番号からなる論理アドレスでもってデータ書込み場所を指定する。アドレス変換表の論理アドレスで特定される項目には、当該論理アドレスに対応する物理アドレスが記憶される。物理アドレスによって指定されることになるＳＳＦの各セクタには、データを記憶するデータ領域の他に、逆参照ポインタ（ＲＰ）を記憶する領域とそのセクタの状況を記憶する領域が含まれる。

【０００９】今、ＳＳＦがホスト・プロセッサから論理アドレス（Ｈ，Ｃ，Ｓ）＝（１，４，５）に対する書込みコマンドを受け取ったときに、それまで空であった物理アドレスＡＢＣのセクタＹをこの論理アドレスに割り当てたとする。ＳＳＦのコントローラは、物理セクタＹのデータ領域にデータを書き込むとともに、ＲＰ領域（１，４，５）を書き込み、状況領域に有効であることを示すフラグを立てる。同時に、論理アドレス（１，４，５）によって特定される変換表の項目Ｘに物理アドレスＡＢＣを書き込む。以後、論理アドレス（１，４，５）からのデータの読出しが要求されたときは、アドレス変換表を使って、物理アドレスＡＢＣがアクセスされる（図１参照）。

【００１０】再びＳＳＦがホスト・プロセッサから論理アドレス（Ｈ，Ｃ，Ｓ）＝（１，４，５）に対する書込みコマンドを受け取ったとき、ＳＳＦのコントローラは、物理セクタＹを無効にし、それまで空であった物理セクタを論理アドレス（１，４，５）に割り当てる。例えば、アドレス変換表の項目ＸをＡＢＤに書き換え、ＳＳＦの物理アドレスＡＢＤのセクタＺのデータ領域にデータを書き込み、ＲＰ領域に（１，４，５）を書き込み、状況領域に有効であることを示すフラグを立てる。同時に、セクタＹの状況領域に無効であることを示すフラグを立てる。

【００１１】さて、パワーオフするとアドレス変換表は失われてしまうから、パワーオン時にこれを再構成する必要がある。そのときには、ＳＳＦの各セクタを読み、逆参照ポインタで指定されるアドレス変換表の項目に、当該セクタの物理アドレスを登録する。図２に示すように、同じＲＰを持つセクタが複数あるときは、有効であるセクタの物理アドレスを登録する。このように、ＳＳＦのセクタの有効・無効の情報は、動的割当ての要であるアドレス変換表の再構成に不可欠である。

【００１２】ところで、先に述べた通り、フラッシュ・メモリに於いてはブロックの消去後でなければそこに含まれるセクタにデータを書き込むことができないので、一般的にはセクタの状況を更新することは困難である。この問題に対して、特願平３－１９７３１８号では、一部のフラッシュ・メモリが有する、ビット変化が一方向に限定されている場合は重ね書きができるという特質に基づき、状況フラグビットを ″1111″ -> ″1110″ -> ″1100″ -> ″0000″ のように変化させることで、各セクタの「空白」、「有効」、「無効」及び「消去中」を示す方法を開示している。しかしながら、ＮＡＮＤ型のセル構造を持つフラッシュ・メモリの中には、全く重ね書きできないものがあり、状況フラグビットを用いた方法は使えない。

【００１３】

【発明が解決しようとする課題】本発明の目的は、重ね書きの手法を用いなくても無効セクタと有効セクタとを見分けることが可能な一括消去型不揮発性メモリ及びそれを用いた半導体ディスク装置を提供することを目的とする。

【００１４】

【課題を解決するための手段】本発明に従う一括消去型不揮発性メモリは、各々がＭ個のセクタからなるＮ個のクラスタを有し（Ｍ、Ｎは各々２以上の整数）、クラスタ単位で消去することが可能であり、上記Ｎ個のクラスタの各々にクラスタ情報セクタが確保され、予め上記Ｎ個のクラスタに対して重複がないように順序番号が与えられ、各クラスタのクラスタ情報セクタに当該クラスタに割り当てられた順序番号が書き込まれている。本発明

に従う半導体ディスク装置は、そのような一括消去型不揮発性メモリとそれに接続されたコントローラを含む。動的セクタ割当てを実行するため、コントローラは、ランダム・アクセス・メモリ上にアドレス変換表の領域を確保し、プロセッサが所与の論理アドレスを指定して書込みを要求したときに、空白セクタを一つ選択し、アドレス換表の当該所与の論理アドレスによって指示される項目に当該選択されたセクタの物理アドレスを　書き込み、且つ当該選択されたセクタに当該所与の論理アドレスを逆参照ポインタとして書き込む。

【００１５】コントローラはまた、所与のクラスタを消去するとき、先に当該クラスタの順序番号を他のクラスタ等の不揮発性の記憶領域に保管する。そして、所与の消去済みクラスタを初期設定するときには、現在の最大順序番号よりも大きな値を当該クラスタの順序番号としてそのクラスタ情報セクタに書き込む。

【００１６】このようにして初期設定されたクラスタのクラスタ情報セクタ以外のセクタに対して、コントローラは、それらセクタの物理アドレスの順番に従ってユーザ・データを書き込む。

【００１７】パワーオン時にアドレス変換表を再構成するとき、コントローラは、不揮発性メモリのM×N個のセクタを読み出し、読み出されたセクタの逆参照ポインタによって指示されるアドレス変換表の項目に当該セクタの物理アドレスを書き込む。同じ逆参照ポインタを持つセクタが複数あるときには、それらセクタの属するクラスタの順序番号とクラスタ内での位置に従って、最も新しく書き込まれたセクタの物理アドレスをアドレス変換表に書き込む。

【００１８】予め各クラスタ情報セクタにそれを含むクラスタの消去回数を書き込み、最大順序番号は全クラスタの消去回数の総和に等しくなるように設定しておいた場合には、クラスタ消去時に順序番号とともに消去回数も保管する。そして、消去済みのクラスタを初期設定するときに、保管されていた当該クラスタの消去回数を順序番号と同様にカウントアップしてそのクラスタ情報セクタに書き戻す。アドレス変換表再構成時において、全クラスタ情報セクタが読み出されることを利用して、最大順序番号と全クラスタの消去回数の総和が一致するかをチェックする。これによって、データの信頼性が維持されているか否かが判る。

【００１９】

【実施例】本発明の半導体ディスク装置として組み込んだコンピュータ・システムの一例を図３に示す。ＣＰＵ１０はシステム・バス１３を介して、主記憶装置１５、バス制御装置１６及びオプションの数値計算用コプロセッサ１４と通信する。ＣＰＵ１０及び関連する周辺装置の間の通信はバス制御装置１６を介して行われる。そのため、バス制御装置１６はファミリー・バス１８によって周辺装置に接続されている。周辺装置としては、本発明に従うフラッシュ・メモリ製の半導体ディスク装置（ＳＳＦ）２０が接続され、さらに、通信装置２１、フロッピー・ディスク・ドライブ（ＦＤＤ）２２、光ファイル装置２３、表示装置２４もファミリー・バス１８に接続されている。勿論、他の周辺装置も接続可能である。このようなコンピュータ・システムの一例はＩＢＭ　ＰＳ／２である。

【００２０】直接メモリ・アクセス制御装置（ＤＭＡＣ）１２は、これらの周辺装置の全部又は選択された何台かによるメモリ・アクセスを可能にすべく設けられる。そのため、ファミリー・バス１８は、少なくともその一部がＤＭＡＣ１２に分岐接続される。図には示していないが、ＤＭＡが可能な各周辺装置にはアービトレーション回路が設けられ、アービトレーション・レベル（優先順位）を割り当てられる。ＤＭＡＣ１２の側には、ＤＭＡを同時に要求している複数の周辺装置の間で調停作業を行って、どの周辺装置がＤＭＡを許可されたかをＤＭＡＣ１２に知らせる中央アービトレーション制御回路１１が設けられる。ＤＭＡＣ１２及び中央アービトレーション制御回路１１によるＤＭＡ制御の詳細は米国特許第４９０１２３４号明細書に記載されている。

【００２１】ＣＰＵ１０はＳＳＦ２０をハード・ディスク装置として扱う。従って、ＳＳＦ２０をアクセスするときは、ヘッド番号、シリンダ番号及びセクタ番号から成るいわゆる相対ブロック・アドレス（ＲＢＡ）がＳＳＦ２０に送られる。ＳＳＦ２０は動的セクタ割当てを行う。従って、ＣＰＵ１０から供給されるＲＢＡと、ＳＳＦ２０の実際にアクセスされるセクタのアドレス（物理アドレス）との間の関係は固定されておらず、書込みの度に変化する。そこで、それらの対応関係を明らかにするアドレス変換表が設けられる。即ち、ＣＰＵ１０からのＲＢＡは論理アドレスである。

【００２２】図４に、ＳＳＦ２０の概略的な構成を示す。このＳＳＦ２０は、ファミリー・バス１８に接続されたコントローラ３０と、内部バス３１を介してこのコントローラ３０に接続されたランダム・アクセス・メモリ（ＲＡＭ）３２、バス制御部３３及びフラッシュ・メモリ３４で構成される。ＲＡＭ３２は、アドレス変換表を記憶する領域３５及びバッファ領域３６を含む。ＲＡＭ３２はこの他に後述する最大順序番号（Ｍ）を記憶する領域も含む。バス制御部３３は、内部バス３１と、フラッシュ・メモリ３４に接続されたメモリ・バス３７とを相互接続するための周知のレシーバ／ドライバ構成を有する。

【００２３】本実施例では、ＣＰＵの指定する論理セクタのサイズは５１２バイトであり、ＣＰＵ１０のＳＳＦ２０に対する最小アクセス単位である物理セクタのサイズは５１２バイト＋αである（図５、図６参照）。１６Ｍビットのフラッシュ・メモリ・チップを用いる場合、１物理セクタはワード・ライン２本を占める。つまり、

２ページで１セクタを構成する。ＳＳＦ２０のセクタ（物理セクタ）は次のようにして管理される。

【００２４】１）実際の消去を行う論理的な集合をつくり、これをクラスタと呼ぶ。クラスタは物理的な消去単位であるブロックの１つ以上からなる。実施例では８セクタで１ブロックを構成し、８ブロックで１クラスタを構成する。各クラスタにクラスタ情報セクタを作成し、クラスタ消去回数及び順序番号の領域を確保する。クラスタ消去回数及び順序番号はクラスタ情報セクタ中の管理情報の一部として保管される。実施例では、各クラスタの先頭の物理アドレスに位置するセクタをクラスタ情報セクタに割り当てる。

【００２５】図６は、各クラスタのクラスタ情報セクタ以外のセクタ（以下ではデータ・セクタと呼ぶ）の構成を示す。図示のように、データ・セクタは、５１２バイトのユーザ・データを記憶するデータ領域の他に、属性及びエラー訂正符号（ＥＣＣ）を記憶する領域を含む。

【００２６】ここでは、特願平３－１９７３１８号とは違って、セクタに有効・無効のフラグを立てる状況領域がないことに注目されたい。なお、各セクタに共通に含まれる属性は、そのセクタがクラスタ情報セクタであるか否かを識別するのに用いられる。

【００２７】２）ＳＳＦ製造後のフラッシュ・メモリ・チップ全体の初期設定工程において、クラスタ別に初期順序番号を重複しないように与える。このとき、全クラスタの消去回数の総和が最大順序番号に等しいという条件を付すのが望ましい。

【００２８】図７に示すように、ＳＳＤ中にＮ個のクラスタが有り、各クラスタに１からＮまでのクラスタ番号を与えたとする。（実際には、クラスタ番号はアドレス・バスの上位複数ビットで指定される。）工場のコンピュータによって実行されるＳＳＤ初期設定プログラムは、各クラスタのクラスタ消去回数を”１”に設定する。同時にクラスタ番号ｉのクラスタの順序番号として”ｉ”を書き込み、順序番号が重複しないようにする。

【００２９】この例ではクラスタ消去回数の初期値として１を与えているが、本発明は実際の消去回数を書き込んだ場合でも実施可能である。また、この例では順序番号の初期値の順をクラスタ番号の順に一致させているが、本発明はそのような初期順序番号の与え方に限定されるものではなく、クラスタ番号に無関係に初期順序番号を割り当ててよい。

【００３０】セクタの有効・無効を判別するために必要とされる条件は、順序番号の重複がないようにすることである。実施例ではさらに順序番号の最大値（最大順序番号）が各クラスタの消去回数の総和に等しいように設定されている。以上の二つの条件は、クラスタ消去時を除き、ＳＳＤが稼働しているどの時点においても満たされなければならない。図７に示した例ではクラスタ消去回数の総和はＮであり、順序番号の最大値もＮであるので、条件を満たしている。

【００３１】もし、図７の例で、クラスタＮのクラスタ消去回数が２であり、その他のクラスタのクラスタ消去回数が１であるならば、クラスタ１からクラスタＮ－１に順序番号１からＮ－１を割り当て、クラスタＮに順序番号Ｎ＋１を割り当てれば、上記二つの条件を満たすことになる。

【００３２】３）図８を参照して、クラスタ消去時のコントローラ３０（図４）の動作を説明する。まず、消去するクラスタを決定する。様々な決定方法があるが、有効セクタの数が一定値を下回ったときに、そのクラスタを消去対象として決定するのが一般的である（ステップ８０）。次に、消去するクラスタをＸとすると、コントローラは、Ｘの有効データを他のクラスタのデータ・セクタに複写する（ステップ８１）。

【００３３】ステップ８０、８１を実行するためには、セクタの有効・無効の判別ができなければならないが、それは所与のセクタの逆参照ポインタが指示するアドレス変換表の項目を参照し、その項目に書き込まれているアドレスが当該セクタのアドレスと一致するかを調べればよい。一致すればそのセクタは有効であり、一致しなければ無効である。ステップ８１ではそのようにして有効セクタを検出する。また、一度クラスタごとに有効・無効セクタの数を調べて結果をＲＡＭ３２（図４）に設けた表（図示せず）に記録し、以後書き込みが行われる度にその表を更新するようにすれば、その表を定期的に参照することによって、消去すべきクラスタを判別することができる。

【００３４】次に、コントローラは、クラスタＸのクラスタ情報セクタを他の適当なクラスタのデータ・セクタや電池バックアップされたＲＡＭ等の不揮発性記憶領域に複写して、クラスタＸの現在の順序番号と消去回数を保管する（ステップ８２）。その後、クラスタＸを消去する（ステップ８３）。消去済みクラスタを初期設定する必要があるとき、即ち、データ・セクタが完全に空白であるクラスタがないときには、直ちに初期設定処理が行われるが（ステップ８５）、そうでなければ、ＳＳＤは通常のオペレーションを行う（ステップ８６）。ここでいう初期設定とは、消去済みのクラスタのクラスタ情報セクタに順序番号等を書き込み、データ・セクタに対して書き込み可能な状態にすることである。

【００３５】図９を参照して、消去済みクラスタ初期設定時のコントローラ３０（図４）の動作を説明する。まず、コントローラは、消去済みではあるがまだ初期設定されていない１以上のクラスタの中からクラスタ消去回数が最小のものを選択する（ステップ９０）。選択されたクラスタをＣとする。

【００３６】次に、現在の最大順序番号Ｍを求め、これに１を加えた値Ｍ＋１をクラスタＣの順序番号としてそ

のクラスタ情報セクタに書き込む（ステップ９１）。ここで、最大順序番号ＭはＲＡＭ３２の領域３８に記憶されているので、それをアクセスする。ステップ９２では、領域３８に値Ｍ＋１が書き込まれる。

【００３７】しかる後、ステップ９３において、クラスタＣを消去するときにステップ８２で保管した消去回数を読み出し、それに１を加えた値をクラスタＣのクラスタ情報セクタに書き込む。順序番号、クラスタ消去回数、ＥＣＣ以外のクラスタ管理情報は、クラスタＣを消去するときに保管した管理情報をそのまま書き込む。＊

＊【００３８】図８および図９に示した制御の流れは一例にすぎず、これを様々に変形することが可能である。例えば、消去したクラスタを直ちに初期設定しても差し支えなく、その場合には、図８のステップ８３の次に直ちに図９のステップ９１に飛ぶことになる。その結果、消去されたクラスタのクラスタ消去回数及び順序番号、及びＲＡＭに保存される最大順序番号は以下のように変化する。

【００３９】

	消去前の値	初期設定後の値
クラスタ消去回数	Ｅ	Ｅ＋１
順序番号	Ｓ	Ｍ＋１
最大順序番号	Ｍ	Ｍ＋１

【００４０】上の例では、現在の最大順序番号Ｍに１を加えた値をクラスタ情報セクタに書き込んだが、要は現在の最大順序番号Ｍよりも大きな値を書き込めばよいのであって、増分を１に限る必要はない。

【００４１】４）セクタへの書き込みは、動的セクタ割当て法を用いる。ただし、特願平３－１９７３１８号とは違って、有効・無効のフラグを立てる操作は行わない。同一のクラスタの中では、セクタはそのアドレスの昇順または降順に書き込まれていく。実施例ではクラスタ情報セクタをクラスタの先頭に置いたので、アドレスの昇順にセクタへデータを書き込むが、クラスタの末尾に置いた場合はアドレスの降順にセクタへデータを書き込むことになる。

【００４２】一つのクラスタに対する書き込みが終わるまで、即ちクラスタがデータで満たされるか、あるいは途中で書き込みを打ち切り、以降のセクタに書き込みを行わないと判断されるまで、他のクラスタにはデータを書き込まない。クラスタの途中から後のセクタが全て不良であるとき、データの書き込みは途中で打ち切られる。不良セクタの情報はＳＳＦ製造後の初期設定工程で予めフラッシュ・メモリ３４の一部に書き込まれているので、そのような打ち切りの判断は可能である。

【００４３】当初は２）で割り当てられた初期順序番号にしたがってクラスタへのユーザ・データの書き込みを行うが、一通り全クラスタにデータを書き込んだ後は、消去と初期設定の過程を経て最大順序番号が付与されたクラスタに対して書き込みを行う。

【００４４】一般的に全てのセクタの時間的前後関係を得ることができれば有効セクタと無効セクタとを見分けることはたやすい。しかしながら、全てのセクタに対して時間情報を書き込むことは、時間情報領域のオーバーヘッドが大きすぎて事実上不可能である。本発明ではこの時間情報を二段階の階層構造によって保持する。第一の階層はクラスタであり、第二の階層はクラスタに含まれるセクタである。３）で述べた消去方法によれば、クラスタ情報セクタに書かれた順序番号によってクラスタ間の時間的前後関係を決定することができ、４）で述べた書き込み方法により第二の階層であるクラスタ内セクタの時間情報がセクタの位置として保存される。これらを組み合わせることにより、全セクタの時間的前後関係を一意に決定することが可能であり、かつ、時間情報を書き込む領域のオーバーヘッドは非常に少ない。

【００４５】５）パワーオン時には、クラスタ情報セクタ内の順序番号とクラスタ内でのセクタの位置を手がかりにして、セクタの有効・無効を判別しつつ、アドレス変換表を再構成する。特定の論理セクタに対応する複数の物理セクタが存在する場合、最大の順序番号を持つクラスタにあるものを有効とする。同一のクラスタに、同一論理セクタに対応する複数の物理セクタが存在する場合は、その位置による時系列情報により有効セクタを決定する。

【００４６】以上を実現する方法としては二つの方法が考えられる。第一の方法は順序番号をまずソートしてその順にクラスタを走査する方法であり、第二の方法はＲＡＭに順序番号表を作るものである。前者はソートが完了してしまえば以後の処理は高速であり利点も多い。しかしながら、ソートに先立ち順序番号を全て読み込まなければならず、従ってクラスタの管理情報を二回読む事になる。さらに、高速なソートを行うには一般的に大きな作業領域を必要とするため状況によってはＳＳＦのコスト上昇を招く場合がある。

【００４７】図１０を参照して、第一の方法の処理の流れを説明する。まず、アドレス変換表の領域をＲＡＭに確保し、各項目の値を特別な値（例えばゼロ）に初期設定する（ステップ１００）。次に、すべてのクラスタ情報セクタを読み出し、順序番号を昇順にソートする（ステップ１０１、１０２）。しかる後、順序番号の小さなものから順にクラスタを選び、選んだクラスタのセクタをアドレスの順に読む出す。実施例では、クラスタ情報セクタから始まってアドレスの昇順にセクタを順次読むことになる。読み出されたセクタの逆参照ポインタによって指示されるアドレス変換表の項目に、当該セクタの

物理アドレスを書き込む。その項目に既に他のセクタのアドレスが書き込まれてあっても、現在読まれたセクタのアドレスを書き込む。（ステップ１０３～１０８）。

【００４８】図１１を参照して、第二の方法で用いる順序番号表について説明する。この方法では、第Ｎクラスタの順序番号を第Ｎ項目に格納するような順序番号表をＲＡＭ３２（図４）に作成する。１クラスタ当たり６４セクタである様な４０ＭビットのＳＳＦを考えると、クラスタ数は冗長分を除けば１２８０であり、順序番号表の１エントリーを５バイトとすれば順序番号表の領域は約６Ｋバイトとなる。また、動的セクタ割当て法使用時にはアドレス変換表がＲＡＭ上に存在するが、この表の各項目の値のうち、下位６ビットを除いた上位１８ビットがクラスタ番号に相当する（以下、上位１８ビットをクラスタポインタと呼ぶ）。今、第Ｓセクタのクラスタポインタの内容がＰであったとすると、順序番号表の第Ｐ項目を読むことにより、第Ｓセクタの存在するクラスタの順序番号を得ることが出来る。図１１の例だと、アドレス変換表は論理セクタＳの実体がクラスタＰのＱ番目に存在することを示し、順序番号表はクラスタＰの順序番号が１２３４であることを示している。なお、図１１において、Ｎｓは論理セクタの総数であり、Ｎｃはクラスタの総数である。

【００４９】次に、図１２を参照して、第二の方法の処理の流れを説明する。α］まず、アドレス変換表の領域をＲＡＭに確保し、各項目の値を特別な値（例えばゼロ）に初期設定する（ステップ１２０）。次に、クラスタを選び、そのクラスタの順序番号を順序番号表に登録する。実施例ではクラスタ番号の昇順にクラスタを選択しているが、クラスタを選択する順番は任意とすることができる（ステップ１２１、１２２、１２３、１３４）。選ばれたクラスタをＣとすると、クラスタＣ内では時系列順にセクタを走査し、各セクタのＲＰ領域の逆参照ポインタを読み出す（ステップ１２４、１２５、１３３）。

【００５０】β］特願平３－１９７３１８号特許出願の方法により、逆参照ポインタからアドレス変換表を再構成する。この時、逆参照ポインタによって指示されるアドレス変換表の項目に既に他のセクタの物理アドレスが書き込まれているか否かで動作が異なる（ステップ１２６、１２７）。

【００５１】◎　当該項目が空白である場合
当該項目に、ステップ１２５で読み出されたセクタＳの物理アドレスを書き込む（ステップ１３０）。

【００５２】◎　当該項目が空白でない場合
当該項目のクラスタポインタにより順序番号表を検索し、当該項目に既に登録されているセクタＳが属するクラスタの順序番号を求める（ステップ１２８）。これを現在読み出し中のクラスタＣの順序番号と大小比較する（ステップ１２９）。前者の方が小さければ当該項目にセクタＳのアドレスを書き込む（ステップ１３０）。そうでないときは何もせず既に登録されているセクタを有効のままにする。

【００５３】γ］以上を全てのクラスタ及びセクタについて繰り返すことによりアドレス変換表には有効セクタのみが登録される（ステップ１３１、１３２）。

【００５４】アドレス変換表を再構成した後、コントローラ３０（図４）は最大順序番号をＲＡＭ３２の作業域に保存し、順序番号表の領域を解放する。既に述べたように、最大順序番号は消去済みクラスタの初期設定で必要とされる。

【００５５】コントローラ３０はさらに、アドレス変換表再構成時に全クラスタのクラスタ情報セクタを読み出すことを利用して、全クラスタの順序番号の総和と最大順序番号とを比較する。ここで両者の値が不一致である場合は、データの信頼性が損なわれていることを意味するので、コントローラ３０はＣＰＵ１０（図３）に対して、表示装置２４にエラー・メッセージを表示させることを要求する。

【００５６】

【発明の効果】本発明に従えば、一括消去型不揮発性メモリを用いた半導体ディスク装置において重ね書きの手法を用いなくても無効セクタと有効セクタとを見分けることが可能になる。

【図面の簡単な説明】

【図１】特願平３－１９７３１８号で開示された動的セクタ割当ての説明図。

【図２】特願平３－１９７３１８号で開示された動的セクタ割当ての説明図。

【図３】本発明に従う半導体ディスク装置を組み込んだコンピュータ・システムの一例を示すブロック図。

【図４】半導体ディスク装置の概略構成を示す図。

【図５】クラスタ情報セクタの構成を示す図。

【図６】クラスタ情報セクタ以外のセクタ（データ・セクタ）の構成を示す図。

【図７】クラスタ情報セクタの初期値の設定を示す図。

【図８】クラスタ消去時のコントローラの動作を示すフローチャート。

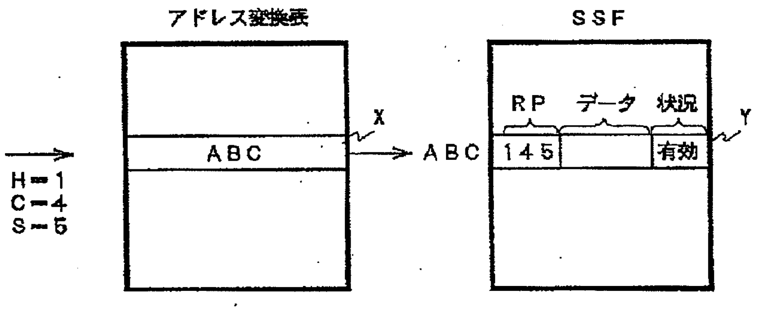
【図９】消去済みクラスタを初期設定するときのコントローラの動作を示すフローチャート。

【図１０】アドレス変換表再構成時のコントローラの動作の一例を示すフローチャート。

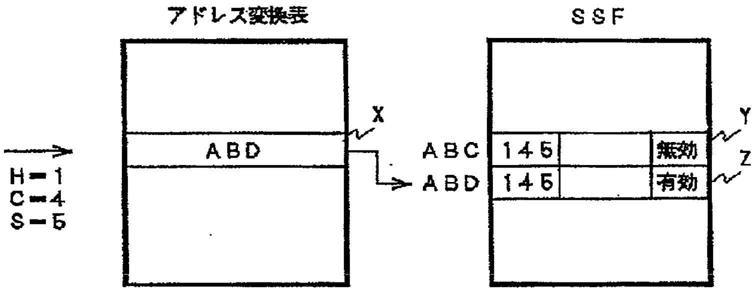
【図１１】アドレス変換表と順序番号表の関係を示す図。

【図１２】アドレス変換表再構成時のコントローラの動作の一例を示すフローチャート。

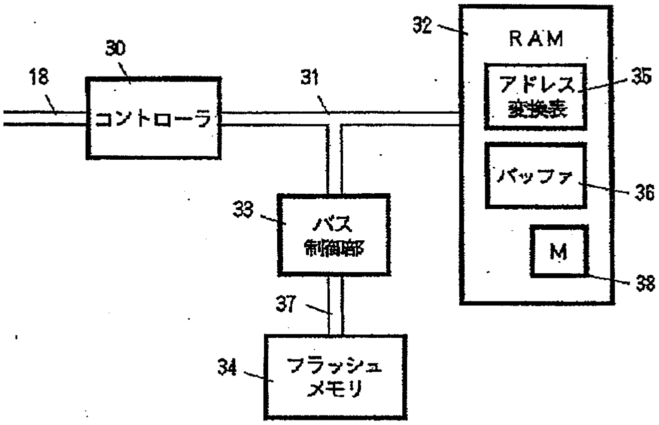
【図１】


アドレス変換表
SSF
RP
データ
状況
X
Y
ABC
ABC
145
有効
H=1
C=4
S=5

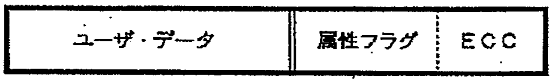
【図２】


アドレス変換表
SSF
X
Y
Z
ABD
ABC
145
無効
ABD
145
有効
H=1
C=4
S=5

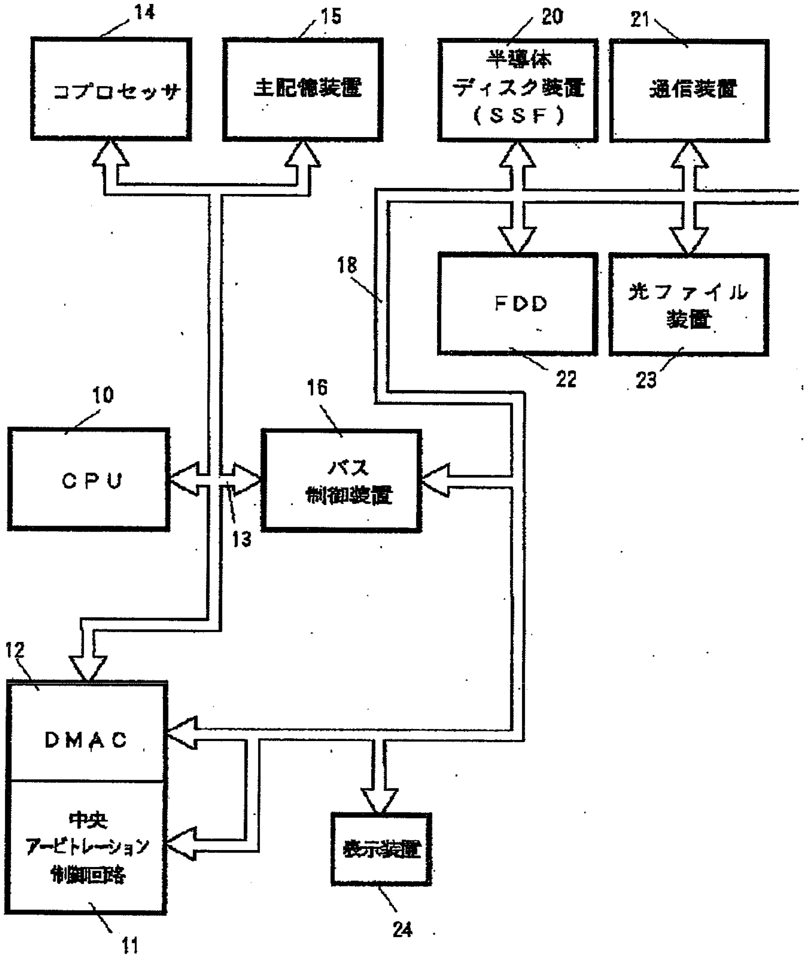
【図４】


32
RAM
30
18
31
コントローラ
アドレス変換表
35
バッファ
36
M
38
33
バス制御部
37
34
フラッシュメモリ

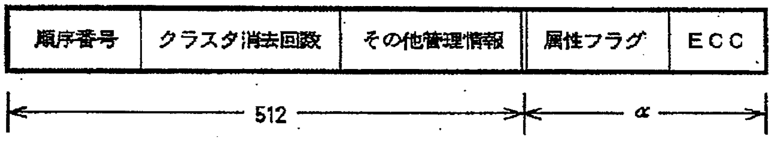
【図６】


ユーザ・データ
属性フラグ
ECC

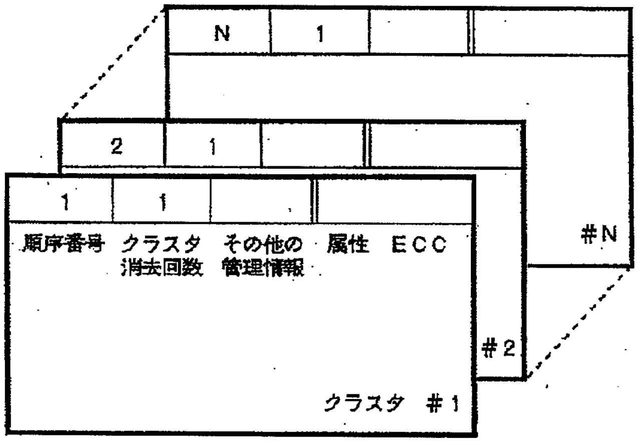
【図３】


14
15
20
21
コプロセッサ
主記憶装置
半導体
ディスク装置
（ＳＳＦ）
通信装置
18
ＦＤＤ
光ファイル
装置
22
23
10
16
ＣＰＵ
バス
制御装置
13
12
ＤＭＡＣ
中央
アービトレーション
制御回路
11
表示装置
24

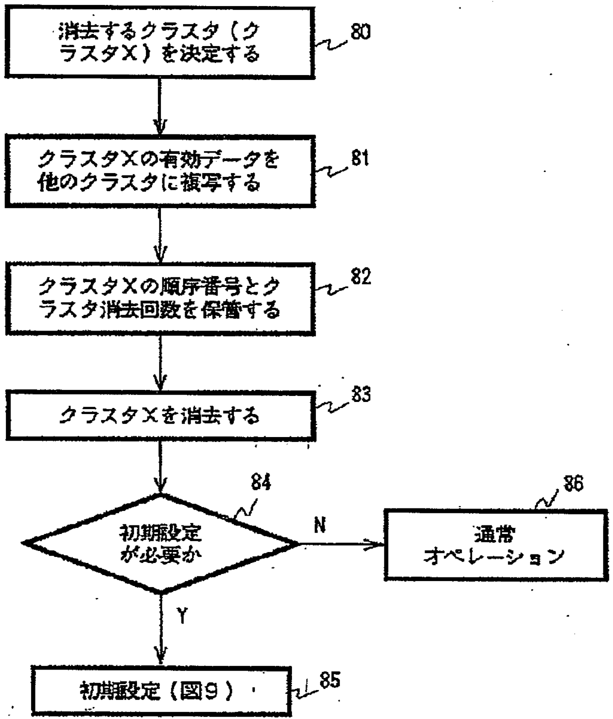
【図５】


順序番号
クラスタ消去回数
その他管理情報
属性フラグ
ＥＣＣ
512
α

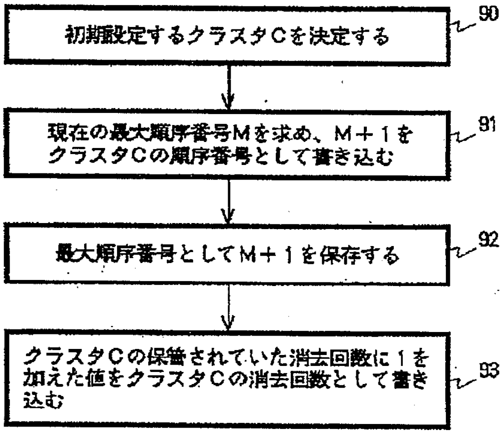
【図７】


N
1
2
1
1
1
順序番号
クラスタ消去回数
その他の管理情報
属性
ＥＣＣ
＃Ｎ
＃２
クラスタ　＃１

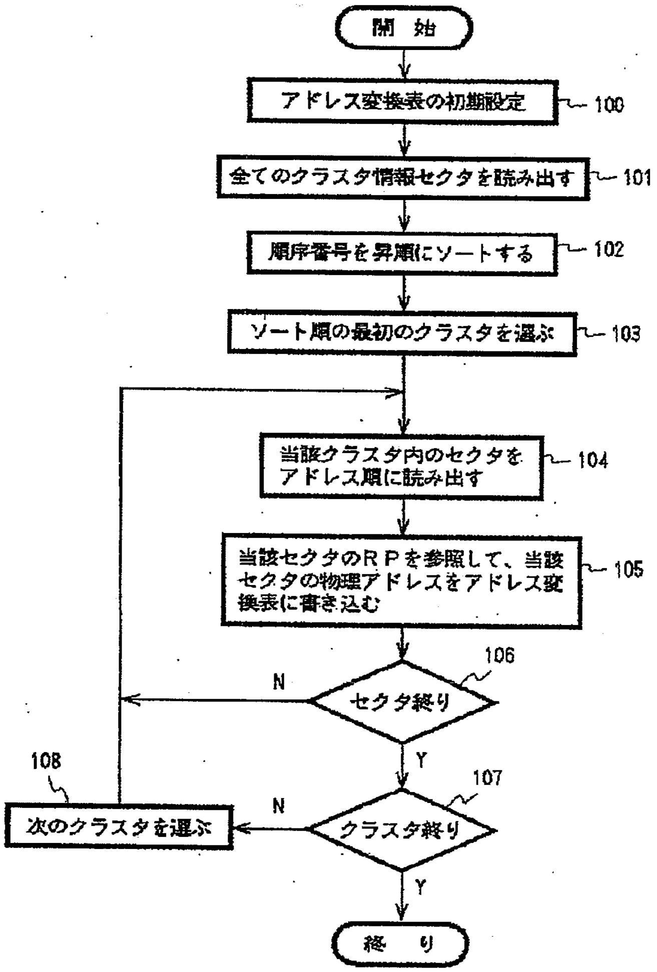
【図８】


消去するクラスタ（クラスタＸ）を決定する
80
クラスタＸの有効データを他のクラスタに複写する
81
クラスタＸの順序番号とクラスタ消去回数を保管する
82
クラスタＸを消去する
83
84
初期設定が必要か
N
通常オペレーション
86
Y
初期設定（図９）
85

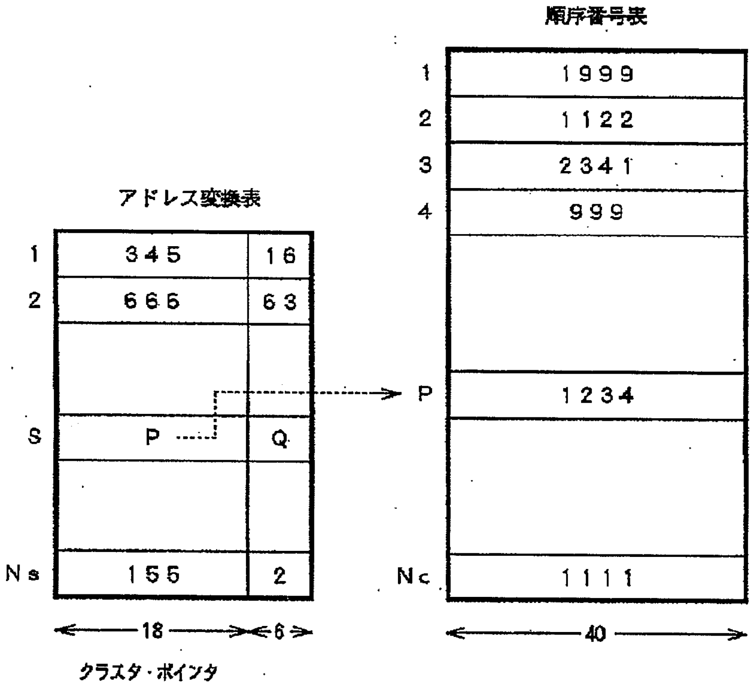
【図９】


初期設定するクラスタＣを決定する
90
現在の最大順序番号Ｍを求め、Ｍ＋１をクラスタＣの順序番号として書き込む
91
最大順序番号としてＭ＋１を保存する
92
クラスタＣの保管されていた消去回数に１を加えた値をクラスタＣの消去回数として書き込む
93

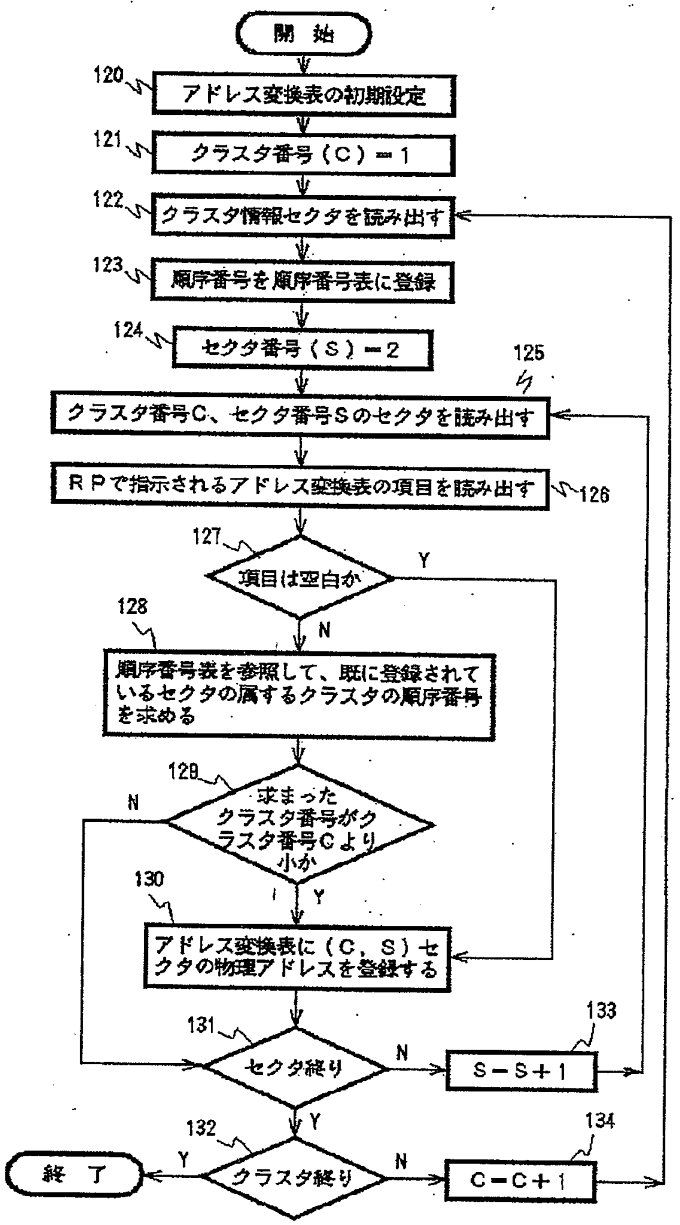
【図１０】


開　始
アドレス変換表の初期設定
100
全てのクラスタ情報セクタを読み出す
101
順序番号を昇順にソートする
102
ソート順の最初のクラスタを選ぶ
103
当該クラスタ内のセクタをアドレス順に読み出す
104
当該セクタのＲＰを参照して、当該セクタの物理アドレスをアドレス変換表に書き込む
105
セクタ終り
106
N
Y
クラスタ終り
107
N
Y
次のクラスタを選ぶ
108
終　り

【図１１】


順序番号表
1999
P
Nc
アドレス変換表
S
P
Q
Ns
クラスタ・ポインタ

【図１２】


開始
120
アドレス変換表の初期設定
121
クラスタ番号（C）－1
122
クラスタ情報セクタを読み出す
123
順序番号を順序番号表に登録
124
セクタ番号（S）－2
125
クラスタ番号C、セクタ番号Sのセクタを読み出す
RPで指示されるアドレス変換表の項目を読み出す
126
127
項目は空白か
Y
N
128
順序番号表を参照して、既に登録されているセクタの属するクラスタの順序番号を求める
129
求まったクラスタ番号がクラスタ番号Cより小か
N
Y
130
アドレス変換表に（C，S）セクタの物理アドレスを登録する
131
セクタ終り
N
133
S－S＋1
Y
132
クラスタ終り
Y
N
134
C－C＋1
終了

フロントページの続き

(72)発明者　浅野　秀夫
東京都千代田区三番町５－19　日本アイ・
ビー・エム株式会社　東京基礎研究所内

(72)発明者　坂上　好功
東京都千代田区三番町５－19　日本アイ・
ビー・エム株式会社　東京基礎研究所内

(72)発明者　豊岡　孝資
東京都千代田区三番町5－19　日本アイ・
ビー・エム株式会社　東京基礎研究所内

NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] Package elimination mold nonvolatile memory characterized by having the cluster of N individual which each becomes from M sectors (M and N being two or more integers respectively), eliminating per cluster being possible and a cluster information sector being secured to each of the cluster of the above-mentioned N individual, giving the sequence number so that there may be no duplication to the cluster of the above-mentioned N individual, and writing the sequence number assigned to the cluster concerned in the cluster information sector of each cluster.
[Claim 2] The maximum of the sequence number which the count of elimination of the cluster concerned was further written in each of the cluster information sector of the above-mentioned N individual, and was given to the cluster which is the above-mentioned N individual is package elimination mold nonvolatile memory according to claim 1 characterized by being equal to total of the count of elimination of these clusters.
[Claim 3] the cluster of N individual to which each serves as a controller from the sector of M individual -- having (M --) N is respectively equipped with two or more integers and the package elimination mold nonvolatile memory which can be eliminated per cluster. A cluster information sector is secured to each of the cluster of the above-mentioned N individual, and actuation of the above-mentioned controller is preceded. The sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual. The sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. The above-mentioned controller When keeping the sequence number of the given cluster concerned previously when eliminating a given cluster, and initializing a given eliminated cluster RAM disk equipment characterized by writing a bigger value than the current maximum sequence number in the cluster information sector as the sequence number of the given cluster concerned.
[Claim 4] The above-mentioned controller is RAM disk equipment according to claim 3 characterized by not writing in the user data to other clusters until the writing of user data to the cluster concerned is completed, after starting the writing of user data to a given cluster.
[Claim 5] The above-mentioned controller is RAM disk equipment according to claim 3 or 4 which writes in user data according to the sequence of the address to sectors other than the cluster information sector of the above-mentioned given cluster.
[Claim 6] the cluster of N individual to which each serves as a controller from the sector of M individual -- having (M --) N is respectively equipped with two or more integers and the package elimination mold nonvolatile memory which can be eliminated per cluster. A cluster information sector is secured to each of the cluster of the above-mentioned N individual, and actuation of the above-mentioned controller is preceded. The sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual. The sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. And the count of elimination of the cluster concerned is further written in each of the cluster information sector of the above-mentioned N individual. The maximum of the sequence number given to the cluster of the above-mentioned N individual is set up equally to total

of the count of elimination of these clusters. The above-mentioned controller When keeping the sequence number and the count of elimination of the given cluster concerned previously when eliminating a given cluster, and initializing a given eliminated cluster RAM disk equipment characterized by adding 1 to the count of elimination which added 1 to the current maximum sequence number, and was kept before elimination, and writing an addition result in the cluster information sector as the sequence number and the count of elimination of the given cluster concerned, respectively.

[Claim 7] The controller which is RAM disk equipment accessed by the logical address from a processor, and was connected to the above-mentioned processor, The package elimination mold nonvolatile memory which each has the cluster of N individual which consists of M sectors (M and N are two or more integers respectively), and can be eliminated per cluster, It has the random access memory connected to the above-mentioned controller. The above-mentioned package elimination mold nonvolatile memory A cluster information sector is secured to each of the cluster of the above-mentioned N individual, and actuation of the above-mentioned controller is preceded. The sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual. The sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. The above-mentioned controller In order to change the logical address included in the command of the above-mentioned processor into the physical address which directs a specific sector, the field of an address translation table is secured on the above-mentioned random access memory. When the above-mentioned processor specifies the given logical address and writing is required Choose one null sector of the above-mentioned package elimination mold nonvolatile memory, and the physical address of the selected sector concerned is written in the item directed by the given logical address concerned of the above-mentioned address translation table. And when writing the given logical address concerned in the selected sector concerned as a reverse reference pointer and eliminating a given cluster It is RAM disk equipment characterized by writing a bigger value than the current maximum sequence number in the cluster information sector as the sequence number of the cluster concerned when keeping the sequence number of the cluster concerned previously and initializing a given eliminated cluster.

[Claim 8] The above-mentioned controller is RAM disk equipment according to claim 7 characterized by answering a period until the writing to the given cluster concerned is completed, and the write request of the above-mentioned processor, and choosing a null sector from the given cluster concerned in order of a physical address after starting writing to a given cluster.

[Claim 9] The above-mentioned controller reads the sector of the MxN individual of the above-mentioned nonvolatile memory, when reconfigurating the above-mentioned address translation table. The physical address of the sector concerned is written in the item of the above-mentioned address translation table directed by the reverse reference pointer of the read sector. When there are two or more sectors with the same reverse reference pointer RAM disk equipment according to claim 7 or 8 characterized by writing the physical address of the sector written in most newly in the above-mentioned address translation table according to the sequence number of the cluster to which these sectors belong, and the location within a cluster.

[Claim 10] The above-mentioned controller reads the cluster information sector of the (a) above-mentioned M clusters, when reconfigurating the above-mentioned address translation table. (b) The sequence number chooses the minimum cluster and the sector of the cluster by which (c) selection was made is read one by one. The physical address of the sector concerned is written in the item of the above-mentioned address translation table directed by the reverse reference pointer of the read sector. (d) RAM disk equipment according to claim 7 or 8 characterized by repeating choosing the following cluster according to the ascending order of the sequence number, and operating the above (c).

[Claim 11] When reconfigurating the above-mentioned address translation table, the above-mentioned controller about each of the M above-mentioned clusters Read the cluster information sector of the cluster concerned and the correspondence relation of the address and the sequence number of the cluster concerned is memorized by the tabular format to the above-mentioned random access memory. Sectors other than (this table is hereafter called a sequence number table) and the cluster information sector of the cluster concerned If the item of the address translation table which reads one by one (it is hereafter called a data sector), and is directed by the reverse reference pointer of the read data sector is read and the item concerned is blank If the physical address of the read data sector concerned is written in there and the item concerned is not blank It asks for the sequence number of the cluster to which the sector located in the physical address written in there belongs with reference to the above-mentioned sequence number table. RAM disk equipment according to claim 7 or 8 characterized by writing the physical address which is the data sector by which reading appearance was carried out [above-mentioned] the account of a top about the sequence number which was able to be found the account of a top as compared with the sequence number of the cluster under present read-out if the direction of the sequence number which was able to be found is smallness in the above-mentioned item.

[Claim 12] A processor, the controller connected to the above-mentioned processor, and the package elimination mold nonvolatile memory which each has the cluster of N individual which consists of M sectors (M and N are two or more integers respectively), and can be eliminated per cluster, It has the random access memory connected to the above-mentioned controller. The above-mentioned package elimination mold nonvolatile memory A cluster information sector is secured to each of the cluster of the above-mentioned N individual, and actuation of the above-mentioned controller is preceded. The sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual. The sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. The above-mentioned controller In order to change the logical address included in the command of the above-mentioned processor into the physical address which directs a specific sector, the field of an address translation table is created on the above-mentioned random access memory. When the above-mentioned processor specifies the given logical address and writing is required Choose one null sector of the above-mentioned package elimination mold nonvolatile memory, and the physical address of the selected sector concerned is written in the item directed by the given logical address concerned of the above-mentioned address translation table. And when writing the given logical address concerned in the selected sector concerned as a reverse reference pointer and eliminating a given cluster It is the data processing system characterized by writing a bigger value than the current maximum sequence number in the cluster information sector as the sequence number of the cluster concerned when keeping the sequence number of the cluster concerned previously and initializing a given eliminated cluster.

[Claim 13] A processor, the display connected to the above-mentioned processor, and the controller connected to the above-mentioned processor, It has the cluster of N individual which each becomes from M sectors (M and N are two or more integers respectively). It has the package elimination mold nonvolatile memory which can be eliminated per cluster. A cluster information sector is secured to each of the cluster of the above-mentioned N individual, and actuation of the above-mentioned controller is preceded. The sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual. The sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. And the count of elimination of the cluster concerned is further written in each of the cluster information sector of the above-mentioned N individual. The maximum of the sequence number given to the cluster of the above-mentioned N individual is set up equally to total of the count of elimination of these clusters. The above-mentioned controller In order to change the logical address included in the command of the above-mentioned processor into the physical address which directs a specific sector, an address translation table is created on the above-mentioned random access memory. When keeping the sequence number and the count of elimination of the given cluster concerned previously when eliminating a given cluster, and initializing a given eliminated cluster 1 is added to the count of elimination which added 1 to the current maximum sequence number, and was kept before elimination. Each addition result is written in the cluster information sector as the given cluster sequence number concerned and a count of elimination. When total of the count of elimination memorized by the cluster information sector of the cluster of the maximum sequence number and the above-mentioned N individual at the time of the above-mentioned address translation tabulation is compared and an inequality is detected Data processing system characterized by what is demanded from the above-mentioned processor so that an error message may be displayed on the above-mentioned indicating equipment.

[Claim 14] It has the cluster of N individual which each becomes from M sectors (M and N are two or more integers respectively). It is the management method of the RAM disk equipment using the package elimination mold nonvolatile memory which can be eliminated per cluster. When giving the sequence number so that there may be no duplication to the cluster of the above-mentioned N individual, writing the assigned sequence number in each cluster and eliminating a given cluster It is the management method of the RAM disk equipment characterized by writing a bigger value than the current maximum sequence number in the cluster as the sequence number of the cluster concerned when eliminating the cluster concerned and initializing a given eliminated cluster after keeping the sequence number of the cluster concerned.

[Claim 15] It has the cluster of N individual which each becomes from M sectors (M and N are two or more integers respectively). It is the management method of the RAM disk equipment using the package elimination mold nonvolatile memory which can be eliminated per cluster. Give the sequence number so that there may be no duplication to the cluster of the above-mentioned N individual, and a cluster is alike, respectively, and the sequence number and count of elimination which were assigned are written in. When setting up equally to total of the count of elimination of these clusters the maximum of the sequence number given to the cluster of the above-mentioned N individual and eliminating a given cluster When keeping the sequence number and the count of elimination of the cluster concerned previously and initializing a given eliminated cluster The management method of the RAM disk equipment characterized by adding 1 to the count of elimination which added 1 to the current maximum sequence number, and was kept before elimination, and writing an addition result in the cluster as the sequence number and the count of elimination of the given cluster concerned, respectively.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application] This invention relates to the RAM disk equipment which uses package elimination mold nonvolatile memory, such as a flash EEPROM (it is called a flash memory below), and it, and relates to the possible RAM disk equipment of dynamic sector allocation especially.
[0002]
[Description of the Prior Art] The demand to the formation of small lightweight of computer system and low-power-izing is becoming strong with the spread of the portable personal computers of a notebook etc. Since the enternal memory system using semiconductor memory and the so-called RAM disk equipment do not have a drive system like a magnetic disk drive, its power consumption is low and high-speed operation is possible for them. Moreover, since it consists of small memory modules, compared with a magnetic disk drive, it is small, and light, and a degree of freedom is large in a configuration, and card-izing is also easy for it.
[0003] However, the conventional semiconductor memory still has many problems in respect of cost, capacity, cell backup, etc. Although the time amount of backup by the cell will become long if SRAM is used as memory, cost will be high and capacity will also become small. In DRAM excellent in cost and capacity, the power consumption at the time of standby will be large, and the time amount of backup will be restricted to about one week. There is also risk of data missing by the accident of a fuel cell subsystem. Cost is too high although EEPROM does not need a cell.
[0004] The flash memory of a package elimination mold is developed as memory which solves these problems. A storage element consists of one transistor as well as DRAM, densification is possible, and whether it is equivalent to DRAM and becoming a bit unit price not more than it (low cost, large capacity) according to a future commercial scene are expected. The storage element is nonvolatile and there is no need for cell backup. Generally elimination is performed in a chip unit or a block unit smaller than it. Richard D.Pashley Besides "Flash memories : 30-33 pages of outlines of such a flash memory are introduced in the best of two worlds", and IEEE SPECTRUM December, 1989. In respect of [mold / chip elimination] performance, the direction of a block elimination mold is excellent.
[0005] When using the flash memory of a block elimination mold for RAM disk equipment and magnitude of a block is made equal to the sector which is the access unit of hard disk equipment, it is convenient for memory management. For example, the Europe public presentation patent application No. 392895 is indicating the flash EEPROM system of a sector elimination mold. This system enables it to eliminate two or more sectors of arbitration to coincidence by setting the latch corresponding to the sector [prepares the latch for every sector and] which is an elimination unit eliminating. The flash memory which carried out the block with the capacity for two or more sectors (for example, 4 K bytes) per elimination is also known.
[0006] However, a flash memory has the limit which is not in SRAM or DRAM. First, programming of a memory bit is one-way traffic, and cannot be changed to 0 to 1, or 1-0. Therefore, to write new data in the storage location already written in, after setting the block including the storage location to a total a

total of of 0 or 1 by package elimination, it is necessary to write in. Elimination and writing usually take the time amount for several seconds from dozens mses. Moreover, a flash memory will deteriorate by elimination and writing, and, now, will reach an operating limit in tens of thousands of times to hundreds of thousands of elimination, and writing.

[0007] When such a flash memory is used for RAM disk equipment, in having assigned the same logical sector to the same physical sector, it becomes a problem that writing inclines toward a part of memory. For example, in the personal computer system of the DOS base, rewriting of a file allocation table (FAT) is often performed. However, since it is fixed, when a flash memory is used, the address of FAT will need to perform elimination and the writing of a block which have memorized it at every rewriting of FAT, and will require the time amount for several seconds for whenever [the] from dozens mses. The block reaches an operating limit early compared with other blocks, and although other blocks are still usable, it must stop moreover, having to exchange memory, if writing and elimination incline toward the block of a part of memory in this way. If the block which reached the operating limit is cancelled and an alternative block is used, although early exchange of memory is avoided, it only changed to the alternative block that writing concentrated, and has not become fundamental solution.

[0008] So, in Japanese Patent Application No. No. 197318 [three to], it has succeeded in solution of this problem by using the dynamic sector assigning method. The outline is explained with reference to drawing 1 and drawing 2 . An address translation table is created by RAM and the address (logical address) which a host processor specifies is changed into the address (physical address) which specifies the sector (physical sector) of RAM disk equipment (SSF) by referring to this. That is, a data write-in location is specified that a host processor is also in the logical address which consists of a head number, a cylinder number, and a sector number. The physical address corresponding to the logical address concerned is memorized by the item specified with the logical address of an address translation table. The field which memorizes the reverse reference pointer (RP) other than the data area which memorizes data, and the field which memorizes the situation of the sector are included in each sector of SSF which will be specified by the physical address.

[0009] When SSF receives the write-in command to logical address (H, C, S) = (1, 4, 5) from a host processor now, suppose that the sector Y of physical address ABC which was empty till then was assigned to this logical address. The controller of SSF writes in RP field (1, 4, 5), and sets the flag which shows that it is effective in a situation field while it writes data in the data area of physical sector Y. Physical address ABC is written in the item X of the conversion table specified as coincidence by the logical address (1, 4, 5). Henceforth, when read-out of the data from the logical address (1, 4, 5) is required, physical address ABC is accessed using an address translation table (refer to drawing 1).

[0010] When SSF receives the write-in command to logical address (H, C, S) = (1, 4, 5) from a host processor again, the controller of SSF makes physical sector Y an invalid, and assigns the physical sector which was empty till then to the logical address (1, 4, 5). For example, the item X of an address translation table is rewritten to ABD, data are written in the data area of the sector Z of the physical address ABD of SSF, (1, 4, 5) are written in RP field, and the flag which shows that it is effective in a situation field is set. The flag which shows an invalid thing to the situation field of Sector Y is set to coincidence.

[0011] Now, if power-off is carried out, since an address translation table will be lost, it needs to reconfigurate this at the time of power-on. Then each sector of SSF is read and the physical address of the sector concerned is registered into the item of the address translation table specified with a reverse reference pointer. As shown in drawing 2 , when there are two or more sectors with the same RP, the physical address of an effective sector is registered. Thus, the information on effective and the invalid of the sector of SSF is indispensable to reconstruction of the address translation table which is the important point of dynamic allocation.

[0012] By the way, since data cannot be written in the sector contained there if it is not after elimination of a block in a flash memory as stated previously, it is difficult to update the situation of a sector generally. As opposed to this problem in Japanese Patent Application No. No. 197318 [three to] It is based on the special feature that overwrite is made when bit change which some flash memories have is limited to the one direction. Situation flag bit "1111" - > "1110"-> "1100"-> "0000" By making it change like, the approach of showing the "null" of each sector, effective ["effective"], an "invalid", and "under elimination" is indicated. However, there are some flash memories with the cellular structure of a NAND mold which cannot carry out overwrite at all, and the approach using a situation flag bit cannot be used.

[0013]

[Problem(s) to be Solved by the Invention] Even if the technique of overwrite is not used for the purpose of this invention, it aims at offering the RAM disk equipment using the package elimination mold nonvolatile memory and it which can recognize an invalid sector and an effective sector.

[0014]

[Means for Solving the Problem] The package elimination mold nonvolatile memory according to this invention has the cluster of N individual which each becomes from M sectors (M and N are two or more integers respectively), and eliminating per cluster is possible, a cluster information sector is secured to each of the cluster of the above-mentioned N individual, the sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual beforehand, and the sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. The RAM disk equipment according to this invention contains the controller connected at such package elimination mold nonvolatile memory and it. It is the physical address of the selected sector concerned to the item which a controller chooses one null sector when the field of an address translation table is secured on random access memory, a processor specifies the given logical address and writing is required in order to perform dynamic sector allocation, and is directed by the given logical address concerned of an address ** table. It writes in and the given logical address concerned is written in the selected sector concerned as a reverse reference pointer.

[0015] A controller keeps the sequence number of the cluster concerned to the storage region of non-volatiles, such as other clusters, previously again, when eliminating a given cluster. And when initializing a given eliminated cluster, a bigger value than the current maximum sequence number is written in the cluster information sector as the sequence number of the cluster concerned.

[0016] Thus, a controller writes in user data to sectors other than the cluster information sector of the initialized cluster according to the sequence of the physical address of these sectors.

[0017] When reconfigurating an address translation table at the time of power-on, a controller writes the physical address of the sector concerned in the item of the address translation table which reads the sector of the MxN individual of nonvolatile memory and is directed by the reverse reference pointer of the read sector. When there are two or more sectors with the same reverse reference pointer, according to the sequence number of the cluster to which these sectors belong, and the location within a cluster, the physical address of the sector written in most newly is written in an address translation table.

[0018] The count of elimination of the cluster which contains it in each cluster information sector beforehand is written in, and the maximum sequence number also keeps the count of elimination with the sequence number at the time of cluster elimination, when it sets up so that it may become equal to total of the count of elimination of all clusters. And when initializing a cluster [finishing / elimination], the count of elimination of the cluster concerned currently kept is counted up like the sequence number, and it returns to the cluster information sector. It is confirmed using all cluster information sectors being read at the time of address translation table reconstruction whether total of the count of elimination of the maximum sequence number and all clusters is in agreement. This shows whether the dependability of data is maintained.

[0019]

[Example] An example of the computer system incorporated as RAM disk equipment of this invention is shown in drawing 3 . CPU10 communicates with main storage 15, bus control equipment 16, and the co-processor 14 for numerical calculation of an option through a system bus 13. The communication link between CPU10 and a related peripheral device is performed through bus control equipment 16. Therefore, bus control equipment 16 is connected to the peripheral device by the family bus 18. As a peripheral device, the RAM disk equipment (SSF) 20 made from a flash memory according to this invention is connected, and a communication device 21, a floppy disk drive (FDD) 22, optical file equipment 23, and an indicating equipment 24 are also further connected to the family bus 18. Of course, other peripheral devices are connectable. An example of such computer system is IBM. It is PS/2.

[0020] The direct memory access control unit (DMAC) 12 is formed that all of these peripheral devices or memory access by several selected sets should be made possible. Therefore, as for the family bus 18, multipoint connection of the part is carried out to DMAC12 at least. Although not shown in drawing, an Arbitration circuit is established in each peripheral device in which DMA is possible, and Arbitration level (priority) can be assigned. Mediation is performed among two or more peripheral devices which are demanding DMA of coincidence, and the central Arbitration control circuit 11 which tells DMAC12 about which peripheral device DMA was permitted is established in the DMAC12 side. The detail of the DMA control by DMAC12 and the central Arbitration control circuit 11 is indicated by the U.S. Pat. No. 4901234 specification.

[0021] CPU10 treats SSF20 as hard disk equipment. Therefore, when accessing SSF20, the so-called relative block address (RBA) which consists of a head number, a cylinder number, and a sector number is sent to SSF20. SSF20 performs dynamic sector allocation. Therefore, it is not fixed but the relation between the addresses (physical address) of the sector accessed by the actual condition of RBA and SSF20 supplied from CPU10 changes at every writing. Then, the address translation table which clarifies those correspondence relation is prepared. That is, RBA from CPU10 is the logical address.

[0022] The rough configuration of SSF20 is shown in drawing 4 . This SSF20 consists of the controller 30 connected to the family bus 18, random access memory (RAM) 32 connected to this controller 30 through the internal bus 31, the bus control section 33, and a flash memory 34. RAM32 includes the field 35 and buffer area 36 which memorize an address translation table. RAM32 also includes the field which memorizes the maximum sequence number (M) mentioned later in addition to this. The bus control section 33 has the receiver / driver configuration of the common knowledge for interconnecting an internal bus 31 and the memory bus 37 connected to the flash memory 34.

[0023] In this example, the size of the physical sector which the size of the logical sector which CPU specifies is 512 bytes, and is the minimum access unit over SSF20 of CPU10 is 512byte+alpha (refer to drawing 5 and drawing 6). When using a 16M bit flash memory chip, 1 physical sector occupies two Ward Rhine. That is, 1 sector consists of 2 pages. The sector (physical sector) of SSF20 is managed as follows.

[0024] 1) Build the logical set which performs actual elimination and call this a cluster. A cluster consists or more of one of the blocks which are physical elimination units. 1 block is constituted from 8 sectors and one cluster consists of examples at 8 blocks. A cluster information sector is created to each cluster, and the field of the count of cluster elimination and the sequence number is secured. The count of cluster elimination and the sequence number are kept as a part of management information in a cluster information sector. In the example, the sector located in the physical address of the head of each cluster is assigned to a cluster information sector.

[0025] Drawing 6 shows the configuration of sectors other than the cluster information sector of each cluster (below, it is called a data sector). Like illustration, a data sector includes the field which memorizes the attribute and error correction sign (ECC) other than a data area which memorize 512 bytes of user data.

[0026] Here, please note that it is different in Japanese Patent Application No. No. 197318 [three to], and there is no situation field which sets the flag of effective and an invalid into a sector. In addition, the attribute included common to each sector is used for identifying whether the sector is a cluster information sector.

[0027] 2) In the initialization process of the whole flash memory chip after SSF manufacture, give so that the initial sequence number may not be overlapped according to a cluster. At this time, it is desirable to attach the conditions that total of the count of elimination of all clusters is equal to the maximum sequence number.

[0028] As shown in drawing 7 , the cluster of N individual is in SSD and suppose that the cluster number from 1 to N was given to each cluster. (A cluster number is specified in high order two or more bits of an address bus in fact.) The SSD initialization program executed by computer of works sets the count of cluster elimination of each cluster as "1." "i" is written in coincidence as the sequence number of the cluster of cluster number i, and it is made for the sequence number not to overlap.

[0029] Although 1 is given as initial value of the count of cluster elimination in this example, this invention can be carried out even when the actual count of elimination is written in. Moreover, although the order of the initial value of the sequence number is made in agreement [in order of a cluster number] in this example, this invention is not limited to how to give such the initial sequence number, and may assign the initial sequence number regardless of a cluster number.

[0030] The conditions needed in order to distinguish effective and the invalid of a sector are making it there be no duplication of the sequence number. In the example, it is set up so that equally to the total whose maximum (the maximum sequence number) of the sequence number is the count of elimination of each cluster further. Except for the time of cluster elimination, the above two conditions must be fulfilled, when [which] SSD is working. Since total of the count of cluster elimination is N and the maximum of the sequence number is also N, conditions are filled with the example shown in drawing 7 .

[0031] If the count of cluster elimination of Cluster N is 2, the count of cluster elimination of other clusters is 1, N-1 will be assigned to a cluster N-1 from the sequence number 1 from a cluster 1 and the sequence number N+1 will be assigned to Cluster N, the two above-mentioned conditions will be filled with the example of drawing 7 .

[0032] 3) Explain actuation of the controller 30 (drawing 4) at the time of cluster elimination with reference to drawing 8 . First, the cluster to eliminate is determined. Although there are the various decision approaches, when the number of effective sectors is less than constant value, it is common to determine the cluster as a candidate for elimination (step 80). Next, if the cluster to eliminate is set to X, a controller will copy the effective data of X into the data sector of other clusters (step 81).

[0033] What is necessary is just to investigate whether it is in agreement with the address whose address currently written in the item is the sector concerned with reference to the item of the address translation table which the reverse reference pointer of a given sector directs, although distinction of the effective and the invalid of a sector must be able to be performed in order to perform steps 80 and 81. If in agreement, the sector is effective, and if it is not in agreement, it is invalid. At step 81, it is made such and an effective sector is detected. Moreover, if the table is updated whenever it records on the table (not shown) which investigated the number of effective and invalid sectors for every cluster once, and prepared the result in RAM32 (drawing 4) and writing is performed henceforth, the cluster which should be eliminated can be distinguished by referring to the table periodically.

[0034] Next, a controller copies the cluster information sector of Cluster X to non-volatile storage regions, such as a data sector of other suitable clusters, and RAM by which cell backup was carried out, and keeps the current sequence number and the current count of elimination of Cluster X (step 82). Then, Cluster X is eliminated (step 83). Although an initialization process is immediately performed when it is necessary to initialize an eliminated cluster (i.e., when a completely blank cluster does not have a data sector) (step 85), SSD performs the usual operation (step 86). Initial setting here is writing the sequence number etc. in the cluster information sector of a cluster [finishing / elimination], and changing into the condition which can be written in to a data sector.

[0035] With reference to drawing 9 , actuation of the controller 30 (drawing 4) at the time of eliminated cluster initialization is explained. First, although a controller is elimination ending, the count of cluster elimination chooses the minimum thing from one or more clusters which are not initialized yet (step 90). The selected cluster is set to C.

[0036] Next, it asks for the current maximum sequence number M, and the value M+1 which added 1 to this is written in the cluster information sector as the sequence number of Cluster C (step 91). Here, since the maximum sequence number M is memorized to the field 38 of RAM32, it is accessed. At step 92, a value M+1 is written in a field 38.

[0037] After an appropriate time, when eliminating Cluster C in step 93, the count of elimination kept at

step 82 is read, and the value which added 1 to it is written in the cluster information sector of Cluster C. Cluster management information other than the sequence number, the count of cluster elimination, and ECC writes in the management information kept when eliminating Cluster C as it is.
[0038] It is possible not to pass over the control flow shown in drawing 8 and drawing 9 to an example, but to transform this variously. For example, it will not interfere, even if it initializes the eliminated cluster immediately, and it will fly to the degree of step 83 of drawing 8 immediately in that case at step 91 of drawing 9 . Consequently, the count of cluster elimination of the eliminated cluster, the sequence number, and the maximum sequence number saved at RAM change as follows.
[0039]
Value before elimination Value after initialization Count of cluster elimination E E+1 Sequence number S M+1 The maximum sequence number M M+1 [0040] Although the value which added 1 to the current maximum sequence number M was written in the cluster information sector in the upper example, what is necessary is just to write in a value bigger in short than the current maximum sequence number M, and it is not necessary to restrict an increment to 1.
[0041] 4) The writing to a sector uses the dynamic sector assigning method. However, it is different in Japanese Patent Application No. No. 197318 [three to], and actuation of setting the flag of effective and an invalid is not performed. In the same cluster, the sector is written in the ascending order or descending order of the address. Although data are written in the ascending order of the address to a sector since the cluster information sector was placed at the head of a cluster in the example, when it places at the tail of a cluster, data will be written in the descending order of the address to a sector.
[0042] A cluster is filled with data, or writing is closed on the way until the writing to one cluster finishes, and data are not written in other clusters until it is judged that it does not write in subsequent sectors. From the middle of a cluster, as for the writing of data, all next sectors are closed on the way, when poor. Since the information on a bad sector is beforehand written in a part of flash memory 34 at the initialization process after SSF manufacture, decision of such the close is possible.
[0043] Although the user data to a cluster are written in at the beginning according to the initial sequence number assigned by 2, after writing data in all clusters briefly, it writes in to the cluster to which the maximum sequence number was given through the process of elimination and initial setting.
[0044] If the time context of all sectors can generally be obtained, it is easy to recognize an effective sector and an invalid sector. However, writing in a hour entry to all sectors has the too large overhead of a hour entry field, and it is impossible as a matter of fact. In this invention, this hour entry is held according to two steps of layered structures. The first hierarchy is a cluster and the second hierarchy is a sector contained in a cluster. According to the elimination approach stated by 3), the sequence number written to the cluster information sector can determine the time context between clusters, and the hour entry of the sector in a cluster which is the second hierarchy is saved as a location which is a sector by the approach of writing in stated by 4. By combining these, it is possible to determine the time context of all sectors as a meaning, and there are very few overheads of the field which writes in a hour entry.
[0045] 5) Reconfigurate an address translation table at the time of power-on, making the sequence number in a cluster information sector, and the location of the sector within a cluster into a key, and distinguishing effective and the invalid of a sector. When two or more physical sectors corresponding to a specific logical sector exist, the thing in a cluster with the greatest sequence number is confirmed. When two or more physical sectors corresponding to the same logical sector exist in the same cluster, the time series information by the location determines an effective sector.
[0046] Two approaches can be considered as an approach of realizing the above. A primary method is the approach of sorting the sequence number first and scanning a cluster in the order, and the second approach makes a sequence number table to RAM. If a sort completes the former, future processings are high-speed and there are also many advantages. However, all the sequence numbers must be read in advance of a sort, therefore the management information of a cluster will be read twice. Furthermore, since a big working area is generally needed to perform a high-speed sort, the cost rise of SSF may be caused depending on a situation.
[0047] The flow of processing of a primary method is explained with reference to drawing 10 . First, the

field of an address translation table is secured to RAM and the value of each item is initialized to a special value (for example, zero) (step 100). Next, all cluster information sectors are read and the sequence number is sorted in ascending order (steps 101 and 102). Sequentially from what has the small sequence number, it is begun after an appropriate time, for a cluster to be chosen and to read the sector of the selected cluster in order of the address. In the example, it will begin from a cluster information sector and a sector will be read one by one in ascending order of the address. The physical address of the sector concerned is written in the item of the address translation table directed by the reverse reference pointer of the read sector. Even if the address of other sectors is already written in the item, the address of the sector read now is written in. (Steps 103-108) .

[0048] With reference to drawing 11 , the sequence number table used by the second approach is explained. By this approach, a sequence number table which stores the sequence number of the Nth cluster in the Nth item is created to RAM32 (drawing 4). Considering 40M bit SSF which are 64 sectors per one cluster, if the number of clusters removes a part for redundancy, it will be 1280 and the field of 5 bytes, then a sequence number table will become about 6 K bytes about one entry of a sequence number table. Moreover, although an address translation table exists on RAM at the time of the dynamic sector assigning method use, 18 bits of high orders except 6 bits of low order are equivalent to a cluster number among the values of each item of this table (18 bits of high orders are hereafter called a cluster pointer). Now, supposing the contents of the cluster pointer of the Sth sector are P, the sequence number of the cluster in which the Sth sector exists can be obtained by reading the Pth item of a sequence number table. In case of the example of drawing 11 , as for an address translation table, the stereo of logical sector S shows that it exists in the Qth of Cluster P, and the sequence number table shows that the sequence number of Cluster P is 1234. In addition, in drawing 11 , Ns is the total of a logical sector and Nc is the total of a cluster.

[0049] Next, the flow of processing of the second approach is explained with reference to drawing 12 . alpha] First, the field of an address translation table is secured to RAM and the value of each item is initialized to a special value (for example, zero) (step 120). Next, a cluster is chosen and the sequence number of the cluster is registered into a sequence number table. Although the cluster is chosen as the ascending order of a cluster number in the example, suppose that the sequence which chooses a cluster is arbitrary (steps 121, 122, 123, and 134). If the selected cluster is set to C, within Cluster C, a sector will be scanned in order of time series, and the reverse reference pointer of RP field of each sector will be read (steps 124, 125, and 133).

[0050] By the approach of the beta] Japanese-Patent-Application-No. No. 197318 [three to] patent application, an address translation table is reconfigurated from a reverse reference pointer. At this time, actuation differs by whether the physical address of other sectors is already written in the item of the address translation table directed by the reverse reference pointer (steps 126 and 127).

[0051] O When the item concerned is blank, the physical address of the sector S read to the item concerned at step 125 is written in (step 130).

[0052] O When the item concerned is not blank, a sequence number table is searched with the cluster pointer of the item concerned, and it asks for the sequence number of the cluster to which the sector S already registered into the item concerned belongs (step 128). The size comparison of this is carried out with the sequence number of the cluster C under current read-out (step 129). If former one is small, the address of Sector S will be written in the item concerned (step 130). The sector which does nothing but has already been registered when that is not right is kept effective.

[0053] Only an effective sector is registered into an address translation table by repeating more than gamma] about all clusters and sectors (steps 131 and 132).

[0054] After reconfigurating an address translation table, a controller 30 (drawing 4) saves the maximum sequence number in the work area of RAM32, and releases the field of a sequence number table. As already stated, the maximum sequence number is needed by initial setting of an eliminated cluster.

[0055] A controller 30 compares total of the sequence number of all clusters with the maximum sequence number further using reading the cluster information sector of all clusters at the time of

address translation table reconstruction. Since it means that the dependability of data is spoiled here when both value is inharmonious, it is required that a controller 30 should display an error message on an indicating equipment 24 to CPU10 (drawing 3).
[0056]
[Effect of the Invention] If this invention is followed, even if it does not use the technique of overwrite in the RAM disk equipment using package elimination mold nonvolatile memory, it will become possible to recognize an invalid sector and an effective sector.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL FIELD

[Industrial Application] This invention relates to the RAM disk equipment which uses package elimination mold nonvolatile memory, such as a flash EEPROM (it is called a flash memory below), and it, and relates to the possible RAM disk equipment of dynamic sector allocation especially.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

PRIOR ART

[Description of the Prior Art] The demand to the formation of small lightweight of computer system and low-power-izing is becoming strong with the spread of the portable personal computers of a notebook etc. Since the enternal memory system using semiconductor memory and the so-called RAM disk equipment do not have a drive system like a magnetic disk drive, its power consumption is low and high-speed operation is possible for them. Moreover, since it consists of small memory modules, compared with a magnetic disk drive, it is small, and light, and a degree of freedom is large in a configuration, and card-izing is also easy for it.

[0003] However, the conventional semiconductor memory still has many problems in respect of cost, capacity, cell backup, etc. Although the time amount of backup by the cell will become long if SRAM is used as memory, cost will be high and capacity will also become small. In DRAM excellent in cost and capacity, the power consumption at the time of standby will be large, and the time amount of backup will be restricted to about one week. There is also risk of data missing by the accident of a fuel cell subsystem. Cost is too high although EEPROM does not need a cell.

[0004] The flash memory of a package elimination mold is developed as memory which solves these problems. A storage element consists of one transistor as well as DRAM, densification is possible, and whether it is equivalent to DRAM and becoming a bit unit price not more than it (low cost, large capacity) according to a future commercial scene are expected. The storage element is nonvolatile and there is no need for cell backup. Generally elimination is performed in a chip unit or a block unit smaller than it. Richard D.Pashley Besides "Flash memories : 30-33 pages of outlines of such a flash memory are introduced in the best of two worlds", and IEEE SPECTRUM December, 1989. In respect of [mold / chip elimination] performance, the direction of a block elimination mold is excellent.

[0005] When using the flash memory of a block elimination mold for RAM disk equipment and magnitude of a block is made equal to the sector which is the access unit of hard disk equipment, it is convenient for memory management. For example, the Europe public presentation patent application No. 392895 is indicating the flash EEPROM system of a sector elimination mold. This system enables it to eliminate two or more sectors of arbitration to coincidence by setting the latch corresponding to the sector [prepares the latch for every sector and] which is an elimination unit eliminating. The flash memory which carried out the block with the capacity for two or more sectors (for example, 4 K bytes) per elimination is also known.

[0006] However, a flash memory has the limit which is not in SRAM or DRAM. First, programming of a memory bit is one-way traffic, and cannot be changed to 0 to 1, or 1-0. Therefore, to write new data in the storage location already written in, after setting the block including the storage location to a total a total of of 0 or 1 by package elimination, it is necessary to write in. Elimination and writing usually take the time amount for several seconds from dozens mses. Moreover, a flash memory will deteriorate by elimination and writing, and, now, will reach an operating limit in tens of thousands of times to hundreds of thousands of elimination, and writing.

[0007] When such a flash memory is used for RAM disk equipment, in having assigned the same logical sector to the same physical sector, it becomes a problem that writing inclines toward a part of memory.

For example, in the personal computer system of the DOS base, rewriting of a file allocation table (FAT) is often performed. However, since it is fixed, when a flash memory is used, the address of FAT will need to perform elimination and the writing of a block which have memorized it at every rewriting of FAT, and will require the time amount for several seconds for whenever [the] from dozens mses. The block reaches an operating limit early compared with other blocks, and although other blocks are still usable, it must stop moreover, having to exchange memory, if writing and elimination incline toward the block of a part of memory in this way. If the block which reached the operating limit is cancelled and an alternative block is used, although early exchange of memory is avoided, it only changed to the alternative block that writing concentrated, and has not become fundamental solution.
[0008] So, in Japanese Patent Application No. No. 197318 [three to], it has succeeded in solution of this problem by using the dynamic sector assigning method. The outline is explained with reference to drawing 1 and drawing 2 . An address translation table is created by RAM and the address (logical address) which a host processor specifies is changed into the address (physical address) which specifies the sector (physical sector) of RAM disk equipment (SSF) by referring to this. That is, a data write-in location is specified that a host processor is also in the logical address which consists of a head number, a cylinder number, and a sector number. The physical address corresponding to the logical address concerned is memorized by the item specified with the logical address of an address translation table. The field which memorizes the reverse reference pointer (RP) other than the data area which memorizes data, and the field which memorizes the situation of the sector are included in each sector of SSF which will be specified by the physical address.
[0009] When SSF receives the write-in command to logical address (H, C, S) = (1, 4, 5) from a host processor now, suppose that the sector Y of physical address ABC which was empty till then was assigned to this logical address. The controller of SSF writes in RP field (1, 4, 5), and sets the flag which shows that it is effective in a situation field while it writes data in the data area of physical sector Y. Physical address ABC is written in the item X of the conversion table specified as coincidence by the logical address (1, 4, 5). Henceforth, when read-out of the data from the logical address (1, 4, 5) is required, physical address ABC is accessed using an address translation table (refer to drawing 1).
[0010] When SSF receives the write-in command to logical address (H, C, S) = (1, 4, 5) from a host processor again, the controller of SSF makes physical sector Y an invalid, and assigns the physical sector which was empty till then to the logical address (1, 4, 5). For example, the item X of an address translation table is rewritten to ABD, data are written in the data area of the sector Z of the physical address ABD of SSF, (1, 4, 5) are written in RP field, and the flag which shows that it is effective in a situation field is set. The flag which shows an invalid thing to the situation field of Sector Y is set to coincidence.
[0011] Now, if power-off is carried out, since an address translation table will be lost, it needs to reconfigurate this at the time of power-on. Then each sector of SSF is read and the physical address of the sector concerned is registered into the item of the address translation table specified with a reverse reference pointer. As shown in drawing 2 , when there are two or more sectors with the same RP, the physical address of an effective sector is registered. Thus, the information on effective and the invalid of the sector of SSF is indispensable to reconstruction of the address translation table which is the important point of dynamic allocation.
[0012] By the way, since data cannot be written in the sector contained there if it is not after elimination of a block in a flash memory as stated previously, it is difficult to update the situation of a sector generally. As opposed to this problem in Japanese Patent Application No. No. 197318 [three to] It is based on the special feature that overwrite is made when bit change which some flash memories have is limited to the one direction. Situation flag bit "1111" - > "1110"-> "1100"-> "0000" By making it change like, the approach of showing the "null" of each sector, effective ["effective"], an "invalid", and "under elimination" is indicated. However, there are some flash memories with the cellular structure of a NAND mold which cannot carry out overwrite at all, and the approach using a situation flag bit cannot be used.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EFFECT OF THE INVENTION

[Effect of the Invention] If this invention is followed, even if it does not use the technique of overwrite in the RAM disk equipment using package elimination mold nonvolatile memory, it will become possible to recognize an invalid sector and an effective sector.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] Even if the technique of overwrite is not used for the purpose of this invention, it aims at offering the RAM disk equipment using the package elimination mold nonvolatile memory and it which can recognize an invalid sector and an effective sector.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

MEANS

[Means for Solving the Problem] The package elimination mold nonvolatile memory according to this invention has the cluster of N individual which each becomes from M sectors (M and N are two or more integers respectively), and eliminating per cluster is possible, a cluster information sector is secured to each of the cluster of the above-mentioned N individual, the sequence number is given so that there may be no duplication to the cluster of the above-mentioned N individual beforehand, and the sequence number assigned to the cluster concerned is written in the cluster information sector of each cluster. The RAM disk equipment according to this invention contains the controller connected at such package elimination mold nonvolatile memory and it. It is the physical address of the selected sector concerned to the item which a controller chooses one null sector when the field of an address translation table is secured on random access memory, a processor specifies the given logical address and writing is required in order to perform dynamic sector allocation, and is directed by the given logical address concerned of an address ** table. It writes in and the given logical address concerned is written in the selected sector concerned as a reverse reference pointer.
[0015] A controller keeps the sequence number of the cluster concerned to the storage region of non-volatiles, such as other clusters, previously again, when eliminating a given cluster. And when initializing a given eliminated cluster, a bigger value than the current maximum sequence number is written in the cluster information sector as the sequence number of the cluster concerned.
[0016] Thus, a controller writes in user data to sectors other than the cluster information sector of the initialized cluster according to the sequence of the physical address of these sectors.
[0017] When reconfigurating an address translation table at the time of power-on, a controller writes the physical address of the sector concerned in the item of the address translation table which reads the sector of the MxN individual of nonvolatile memory and is directed by the reverse reference pointer of the read sector. When there are two or more sectors with the same reverse reference pointer, according to the sequence number of the cluster to which these sectors belong, and the location within a cluster, the physical address of the sector written in most newly is written in an address translation table.
[0018] The count of elimination of the cluster which contains it in each cluster information sector beforehand is written in, and the maximum sequence number also keeps the count of elimination with the sequence number at the time of cluster elimination, when it sets up so that it may become equal to total of the count of elimination of all clusters. And when initializing a cluster [finishing / elimination], the count of elimination of the cluster concerned currently kept is counted up like the sequence number, and it returns to the cluster information sector. It is confirmed using all cluster information sectors being read at the time of address translation table reconstruction whether total of the count of elimination of the maximum sequence number and all clusters is in agreement. This shows whether the dependability of data is maintained.

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EXAMPLE

[Example] An example of the computer system incorporated as RAM disk equipment of this invention is shown in drawing 3 . CPU10 communicates with main storage 15, bus control equipment 16, and the co-processor 14 for numerical calculation of an option through a system bus 13. The communication link between CPU10 and a related peripheral device is performed through bus control equipment 16. Therefore, bus control equipment 16 is connected to the peripheral device by the family bus 18. As a peripheral device, the RAM disk equipment (SSF) 20 made from a flash memory according to this invention is connected, and a communication device 21, a floppy disk drive (FDD) 22, optical file equipment 23, and an indicating equipment 24 are also further connected to the family bus 18. Of course, other peripheral devices are connectable. An example of such computer system is IBM. It is PS/2.

[0020] The direct memory access control unit (DMAC) 12 is formed that all of these peripheral devices or memory access by several selected sets should be made possible. Therefore, as for the family bus 18, multipoint connection of the part is carried out to DMAC12 at least. Although not shown in drawing, an Arbitration circuit is established in each peripheral device in which DMA is possible, and Arbitration level (priority) can be assigned. Mediation is performed among two or more peripheral devices which are demanding DMA of coincidence, and the central Arbitration control circuit 11 which tells DMAC12 about which peripheral device DMA was permitted is established in the DMAC12 side. The detail of the DMA control by DMAC12 and the central Arbitration control circuit 11 is indicated by the U.S. Pat. No. 4901234 specification.

[0021] CPU10 treats SSF20 as hard disk equipment. Therefore, when accessing SSF20, the so-called relative block address (RBA) which consists of a head number, a cylinder number, and a sector number is sent to SSF20. SSF20 performs dynamic sector allocation. Therefore, it is not fixed but the relation between the addresses (physical address) of the sector accessed by the actual condition of RBA and SSF20 supplied from CPU10 changes at every writing. Then, the address translation table which clarifies those correspondence relation is prepared. That is, RBA from CPU10 is the logical address.

[0022] The rough configuration of SSF20 is shown in drawing 4 . This SSF20 consists of the controller 30 connected to the family bus 18, random access memory (RAM) 32 connected to this controller 30 through the internal bus 31, the bus control section 33, and a flash memory 34. RAM32 includes the field 35 and buffer area 36 which memorize an address translation table. RAM32 also includes the field which memorizes the maximum sequence number (M) mentioned later in addition to this. The bus control section 33 has the receiver / driver configuration of the common knowledge for interconnecting an internal bus 31 and the memory bus 37 connected to the flash memory 34.

[0023] In this example, the size of the physical sector which the size of the logical sector which CPU specifies is 512 bytes, and is the minimum access unit over SSF20 of CPU10 is 512byte+alpha (refer to drawing 5 and drawing 6). When using a 16M bit flash memory chip, 1 physical sector occupies two Ward Rhine. That is, 1 sector consists of 2 pages. The sector (physical sector) of SSF20 is managed as follows.

[0024] 1) Build the logical set which performs actual elimination and call this a cluster. A cluster consists or more of one of the blocks which are physical elimination units. 1 block is constituted from 8 sectors and one cluster consists of examples at 8 blocks. A cluster information sector is created to each cluster, and the field of the count of cluster elimination and the sequence number is secured. The count of cluster elimination and the sequence number are kept as a part of management information in a cluster information sector. In the example, the sector located in the physical address of the head of each cluster is assigned to a cluster information sector.

[0025] Drawing 6 shows the configuration of sectors other than the cluster information sector of each cluster (below, it is called a data sector). Like illustration, a data sector includes the field which memorizes the attribute and error correction sign (ECC) other than a data area which memorize 512 bytes of user data.

[0026] Here, please note that it is different in Japanese Patent Application No. No. 197318 [three to], and there is no situation field which sets the flag of effective and an invalid into a sector. In addition, the attribute included common to each sector is used for identifying whether the sector is a cluster information sector.

[0027] 2) In the initialization process of the whole flash memory chip after SSF manufacture, give so that the initial sequence number may not be overlapped according to a cluster. At this time, it is desirable to attach the conditions that total of the count of elimination of all clusters is equal to the maximum sequence number.

[0028] As shown in drawing 7 , the cluster of N individual is in SSD and suppose that the cluster number from 1 to N was given to each cluster. (A cluster number is specified in high order two or more bits of an address bus in fact.) The SSD initialization program executed by computer of works sets the count of cluster elimination of each cluster as "1." "i" is written in coincidence as the sequence number of the cluster of cluster number i, and it is made for the sequence number not to overlap.

[0029] Although 1 is given as initial value of the count of cluster elimination in this example, this invention can be carried out even when the actual count of elimination is written in. Moreover, although the order of the initial value of the sequence number is made in agreement [in order of a cluster number] in this example, this invention is not limited to how to give such the initial sequence number, and may assign the initial sequence number regardless of a cluster number.

[0030] The conditions needed in order to distinguish effective and the invalid of a sector are making it there be no duplication of the sequence number. In the example, it is set up so that equally to the total whose maximum (the maximum sequence number) of the sequence number is the count of elimination of each cluster further. Except for the time of cluster elimination, the above two conditions must be fulfilled, when [which] SSD is working. Since total of the count of cluster elimination is N and the maximum of the sequence number is also N, conditions are filled with the example shown in drawing 7 .

[0031] If the count of cluster elimination of Cluster N is 2, the count of cluster elimination of other clusters is 1, N-1 will be assigned to a cluster N-1 from the sequence number 1 from a cluster 1 and the sequence number N+1 will be assigned to Cluster N, the two above-mentioned conditions will be filled with the example of drawing 7 .

[0032] 3) Explain actuation of the controller 30 (drawing 4) at the time of cluster elimination with reference to drawing 8 . First, the cluster to eliminate is determined. Although there are the various decision approaches, when the number of effective sectors is less than constant value, it is common to determine the cluster as a candidate for elimination (step 80). Next, if the cluster to eliminate is set to X, a controller will copy the effective data of X into the data sector of other clusters (step 81).

[0033] What is necessary is just to investigate whether it is in agreement with the address whose address currently written in the item is the sector concerned with reference to the item of the address translation table which the reverse reference pointer of a given sector directs, although distinction of the effective and the invalid of a sector must be able to be performed in order to perform steps 80 and 81. If in agreement, the sector is effective, and if it is not in agreement, it is invalid. At step 81, it is made such and an effective sector is detected. Moreover, if the table is updated whenever it records on the table (not shown) which investigated the number of effective and invalid sectors for every cluster once, and prepared the result in RAM32 (drawing 4) and writing is performed henceforth, the cluster which should be eliminated can be distinguished by referring to the table periodically.

[0034] Next, a controller copies the cluster information sector of Cluster X to non-volatile storage regions, such as a data sector of other suitable clusters, and RAM by which cell backup was carried out, and keeps the current sequence number and the current count of elimination of Cluster X (step 82). Then, Cluster X is eliminated (step 83). Although an initialization process is immediately performed when it is necessary to initialize an eliminated cluster (i.e., when a completely blank cluster does not have a data sector) (step 85), SSD performs the usual operation (step 86). Initial setting here is writing the sequence number etc. in the cluster information sector of a cluster [finishing / elimination], and changing into the condition which can be written in to a data sector.

[0035] With reference to drawing 9 , actuation of the controller 30 (drawing 4) at the time of eliminated cluster initialization is explained. First, although a controller is elimination ending, the count of cluster elimination chooses the minimum thing from one or more clusters which are not initialized yet (step 90). The selected cluster is set to C.

[0036] Next, it asks for the current maximum sequence number M, and the value M+1 which added 1 to this is written in the cluster information sector as the sequence number of Cluster C (step 91). Here, since the maximum sequence number M is memorized to the field 38 of RAM32, it is accessed. At step 92, a value M+1 is written in a field 38.

[0037] After an appropriate time, when eliminating Cluster C in step 93, the count of elimination kept at step 82 is read, and the value which added 1 to it is written in the cluster information sector of Cluster C. Cluster management information other than the sequence number, the count of cluster elimination, and ECC writes in the management information kept when eliminating Cluster C as it is.

[0038] It is possible not to pass over the control flow shown in drawing 8 and drawing 9 to an example, but to transform this variously. For example, it will not interfere, even if it initializes the eliminated cluster immediately, and it will fly to the degree of step 83 of drawing 8 immediately in that case at step 91 of drawing 9 . Consequently, the count of cluster elimination of the eliminated cluster, the sequence number, and the maximum sequence number saved at RAM change as follows.

[0039]

Value before elimination Value after initialization Count of cluster elimination E E+1 Sequence number S M+1 The maximum sequence number M M+1 [0040] Although the value which added 1 to the current maximum sequence number M was written in the cluster information sector in the upper example, what is necessary is just to write in a value bigger in short than the current maximum sequence number M, and it is not necessary to restrict an increment to 1.

[0041] 4) The writing to a sector uses the dynamic sector assigning method. However, it is different in Japanese Patent Application No. No. 197318 [three to], and actuation of setting the flag of effective and an invalid is not performed. In the same cluster, the sector is written in the ascending order or descending order of the address. Although data are written in the ascending order of the address to a sector since the cluster information sector was placed at the head of a cluster in the example, when it places at the tail of a cluster, data will be written in the descending order of the address to a sector.

[0042] A cluster is filled with data, or writing is closed on the way until the writing to one cluster finishes, and data are not written in other clusters until it is judged that it does not write in subsequent sectors. From the middle of a cluster, as for the writing of data, all next sectors are closed on the way, when poor. Since the information on a bad sector is beforehand written in a part of flash memory 34 at the initialization process after SSF manufacture, decision of such the close is possible.

[0043] Although the user data to a cluster are written in at the beginning according to the initial sequence number assigned by 2, after writing data in all clusters briefly, it writes in to the cluster to which the maximum sequence number was given through the process of elimination and initial setting.

[0044] If the time context of all sectors can generally be obtained, it is easy to recognize an effective sector and an invalid sector. However, writing in a hour entry to all sectors has the too large overhead of a hour entry field, and it is impossible as a matter of fact. In this invention, this hour entry is held according to two steps of layered structures. The first hierarchy is a cluster and the second hierarchy is a

sector contained in a cluster. According to the elimination approach stated by 3), the sequence number written to the cluster information sector can determine the time context between clusters, and the hour entry of the sector in a cluster which is the second hierarchy is saved as a location which is a sector by the approach of writing in stated by 4. By combining these, it is possible to determine the time context of all sectors as a meaning, and there are very few overheads of the field which writes in a hour entry.
[0045] 5) Reconfigurate an address translation table at the time of power-on, making the sequence number in a cluster information sector, and the location of the sector within a cluster into a key, and distinguishing effective and the invalid of a sector. When two or more physical sectors corresponding to a specific logical sector exist, the thing in a cluster with the greatest sequence number is confirmed. When two or more physical sectors corresponding to the same logical sector exist in the same cluster, the time series information by the location determines an effective sector.
[0046] Two approaches can be considered as an approach of realizing the above. A primary method is the approach of sorting the sequence number first and scanning a cluster in the order, and the second approach makes a sequence number table to RAM. If a sort completes the former, future processings are high-speed and there are also many advantages. However, all the sequence numbers must be read in advance of a sort, therefore the management information of a cluster will be read twice. Furthermore, since a big working area is generally needed to perform a high-speed sort, the cost rise of SSF may be caused depending on a situation.
[0047] The flow of processing of a primary method is explained with reference to drawing 10 . First, the field of an address translation table is secured to RAM and the value of each item is initialized to a special value (for example, zero) (step 100). Next, all cluster information sectors are read and the sequence number is sorted in ascending order (steps 101 and 102). Sequentially from what has the small sequence number, it is begun after an appropriate time, for a cluster to be chosen and to read the sector of the selected cluster in order of the address. In the example, it will begin from a cluster information sector and a sector will be read one by one in ascending order of the address. The physical address of the sector concerned is written in the item of the address translation table directed by the reverse reference pointer of the read sector. Even if the address of other sectors is already written in the item, the address of the sector read now is written in. (Steps 103-108) .
[0048] With reference to drawing 11 , the sequence number table used by the second approach is explained. By this approach, a sequence number table which stores the sequence number of the Nth cluster in the Nth item is created to RAM32 (drawing 4). Considering 40M bit SSF which are 64 sectors per one cluster, if the number of clusters removes a part for redundancy, it will be 1280 and the field of 5 bytes, then a sequence number table will become about 6 K bytes about one entry of a sequence number table. Moreover, although an address translation table exists on RAM at the time of the dynamic sector assigning method use, 18 bits of high orders except 6 bits of low order are equivalent to a cluster number among the values of each item of this table.(18 bits of high orders are hereafter called a cluster pointer). Now, supposing the contents of the cluster pointer of the Sth sector are P, the sequence number of the cluster in which the Sth sector exists can be obtained by reading the Pth item of a sequence number table. In case of the example of drawing 11 , as for an address translation table, the stereo of logical sector S shows that it exists in the Qth of Cluster P, and the sequence number table shows that the sequence number of Cluster P is 1234. In addition, in drawing 11 , Ns is the total of a logical sector and Nc is the total of a cluster.
[0049] Next, the flow of processing of the second approach is explained with reference to drawing 12 . alpha] First, the field of an address translation table is secured to RAM and the value of each item is initialized to a special value (for example, zero) (step 120). Next, a cluster is chosen and the sequence number of the cluster is registered into a sequence number table. Although the cluster is chosen as the ascending order of a cluster number in the example, suppose that the sequence which chooses a cluster is arbitrary (steps 121, 122, 123, and 134). If the selected cluster is set to C, within Cluster C, a sector will be scanned in order of time series, and the reverse reference pointer of RP field of each sector will be read (steps 124, 125, and 133).
[0050] By the approach of the beta] Japanese-Patent-Application-No. No. 197318 [three to] patent

application, an address translation table is reconfigurated from a reverse reference pointer. At this time, actuation differs by whether the physical address of other sectors is already written in the item of the address translation table directed by the reverse reference pointer (steps 126 and 127).

[0051] O When the item concerned is blank, the physical address of the sector S read to the item concerned at step 125 is written in (step 130).

[0052] O When the item concerned is not blank, a sequence number table is searched with the cluster pointer of the item concerned, and it asks for the sequence number of the cluster to which the sector S already registered into the item concerned belongs (step 128). The size comparison of this is carried out with the sequence number of the cluster C under current read-out (step 129). If former one is small, the address of Sector S will be written in the item concerned (step 130). The sector which does nothing but has already been registered when that is not right is kept effective.

[0053] Only an effective sector is registered into an address translation table by repeating more than gamma] about all clusters and sectors (steps 131 and 132).

[0054] After reconfigurating an address translation table, a controller 30 (drawing 4) saves the maximum sequence number in the work area of RAM32, and releases the field of a sequence number table. As already stated, the maximum sequence number is needed by initial setting of an eliminated cluster.

[0055] A controller 30 compares total of the sequence number of all clusters with the maximum sequence number further using reading the cluster information sector of all clusters at the time of address translation table reconstruction. Since it means that the dependability of data is spoiled here when both value is inharmonious, it is required that a controller 30 should display an error message on an indicating equipment 24 to CPU10 (drawing 3).

[Translation done.]

* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] The explanatory view of the dynamic sector allocation indicated by Japanese Patent Application No. No. 197318 [three to].
[Drawing 2] The explanatory view of the dynamic sector allocation indicated by Japanese Patent Application No. No. 197318 [three to].
[Drawing 3] The block diagram showing an example of the computer system incorporating the RAM disk equipment according to this invention.
[Drawing 4] Drawing showing the outline configuration of RAM disk equipment.
[Drawing 5] Drawing showing the configuration of a cluster information sector.
[Drawing 6] Drawing showing the configuration of sectors other than a cluster information sector (data sector).
[Drawing 7] Drawing showing a setup of the initial value of a cluster information sector.
[Drawing 8] The flow chart which shows actuation of the controller at the time of cluster elimination.
[Drawing 9] The flow chart which shows actuation of the controller when initializing an eliminated cluster.
[Drawing 10] The flow chart which shows an example of actuation of the controller at the time of address translation table reconstruction.
[Drawing 11] Drawing showing the relation between an address translation table and a sequence number table.
[Drawing 12] The flow chart which shows an example of actuation of the controller at the time of address translation table reconstruction.

[Translation done.]

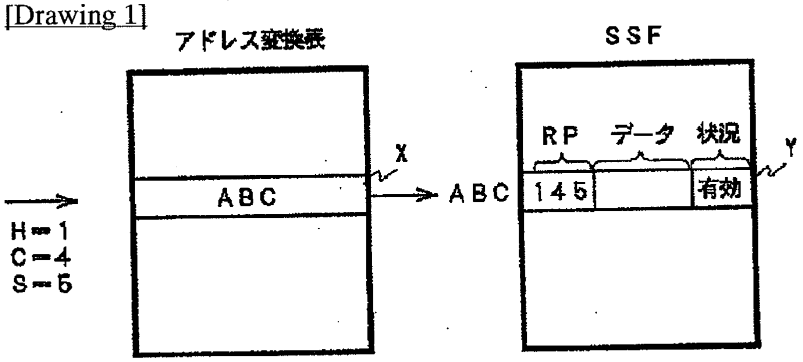
* NOTICES *

JPO and INPIT are not responsible for any damages caused by the use of this translation.

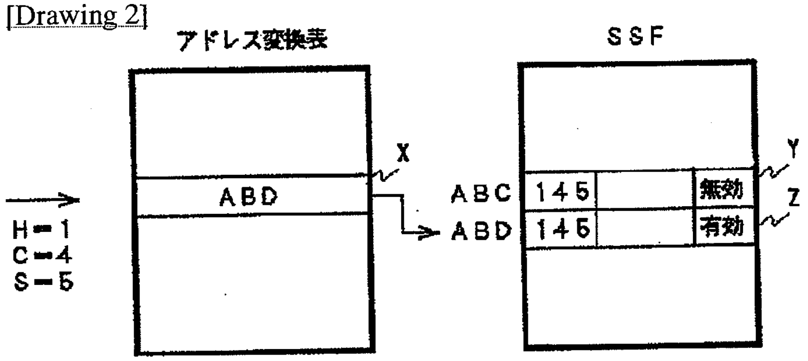
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

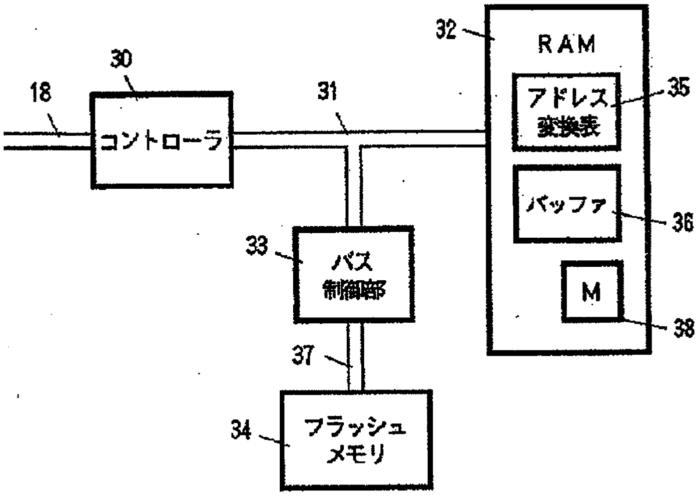
[Drawing 1]


アドレス変換表
SSF
RP
データ
状況
X
Y
ABC
ABC
145
有効
H=1
C=4
S=5

[Drawing 2]


アドレス変換表
SSF
X
Y
Z
ABD
ABC
ABD
145
145
無効
有効
H=1
C=4
S=5

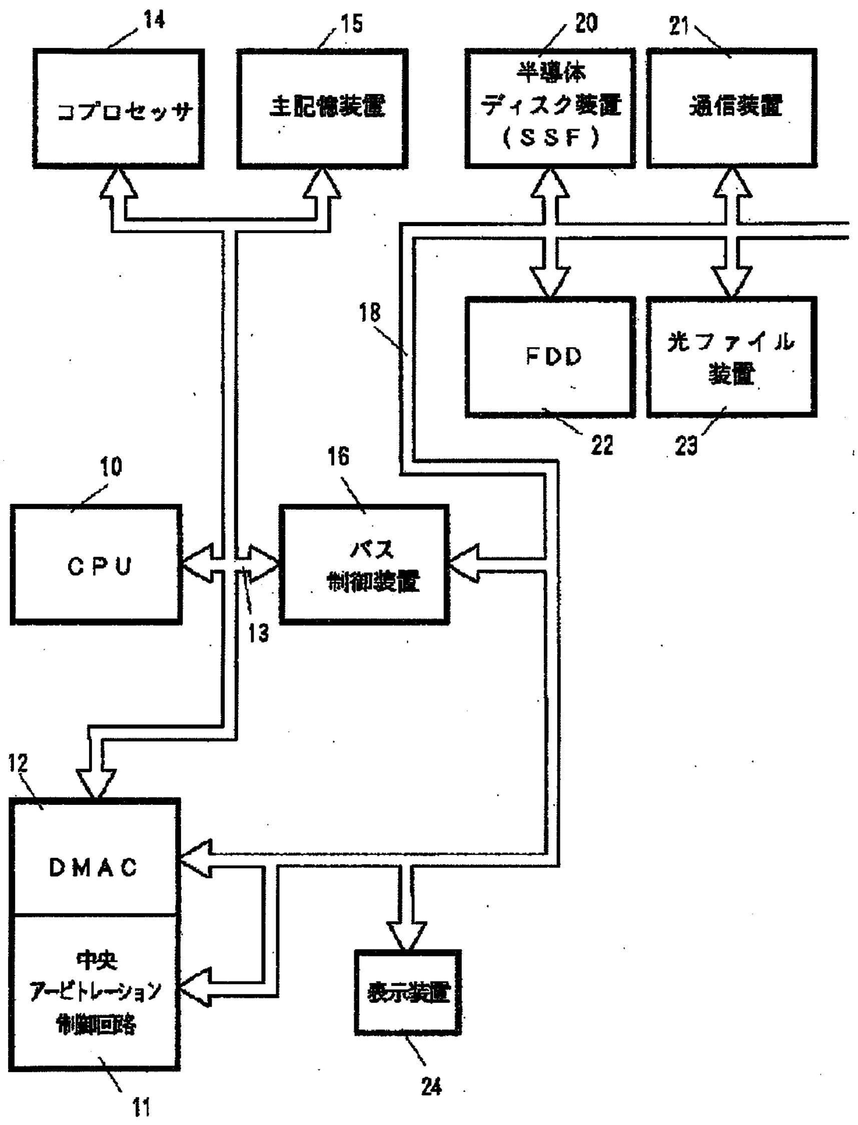
[Drawing 4]


32
RAM
30
18
31
アドレス
変換表
35
コントローラ
バッファ
36
33
バス
制御部
M
38
37
34
フラッシュ
メモリ

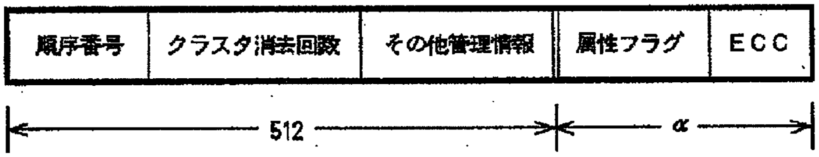
[Drawing 6]

ユーザ・データ	属性フラグ	ECC

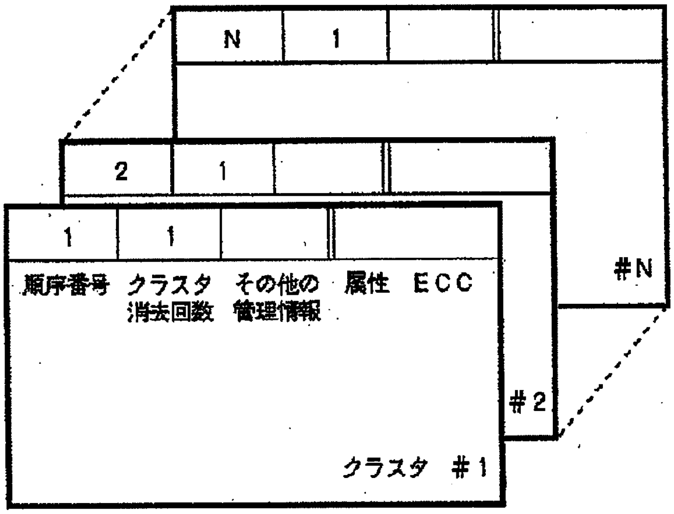
[Drawing 3]


14
コプロセッサ
15
主記憶装置
20
半導体
ディスク装置
（SSF）
21
通信装置
18
FDD
22
光ファイル
装置
23
10
CPU
16
バス
制御装置
13
12
DMAC
中央
アービトレーション
制御回路
11
表示装置
24

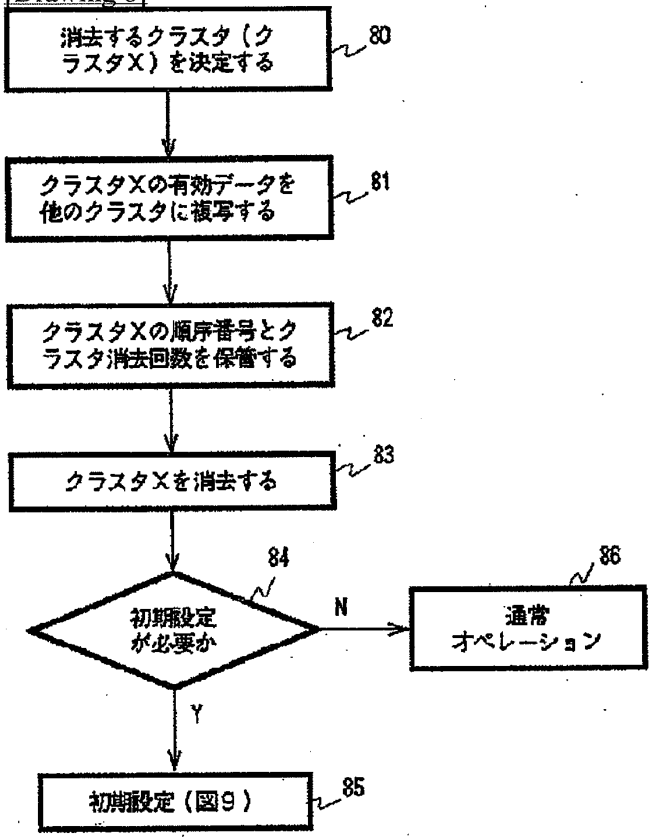
[Drawing 5]


順序番号
クラスタ消去回数
その他管理情報
属性フラグ
ECC
512
α

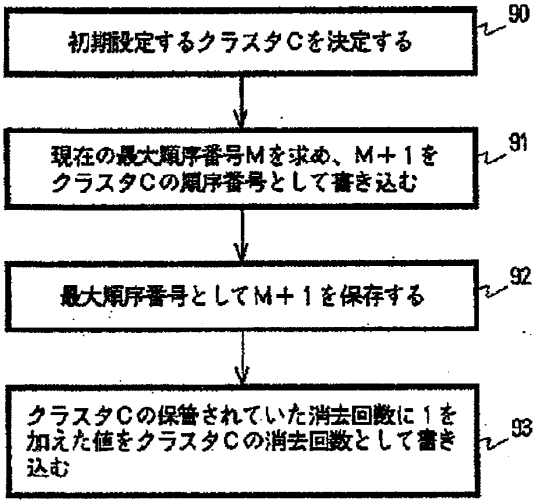
[Drawing 7]


N
1
2
1
1
1
順序番号
クラスタ消去回数
その他の管理情報
属性
ECC
#N
#2
クラスタ #1

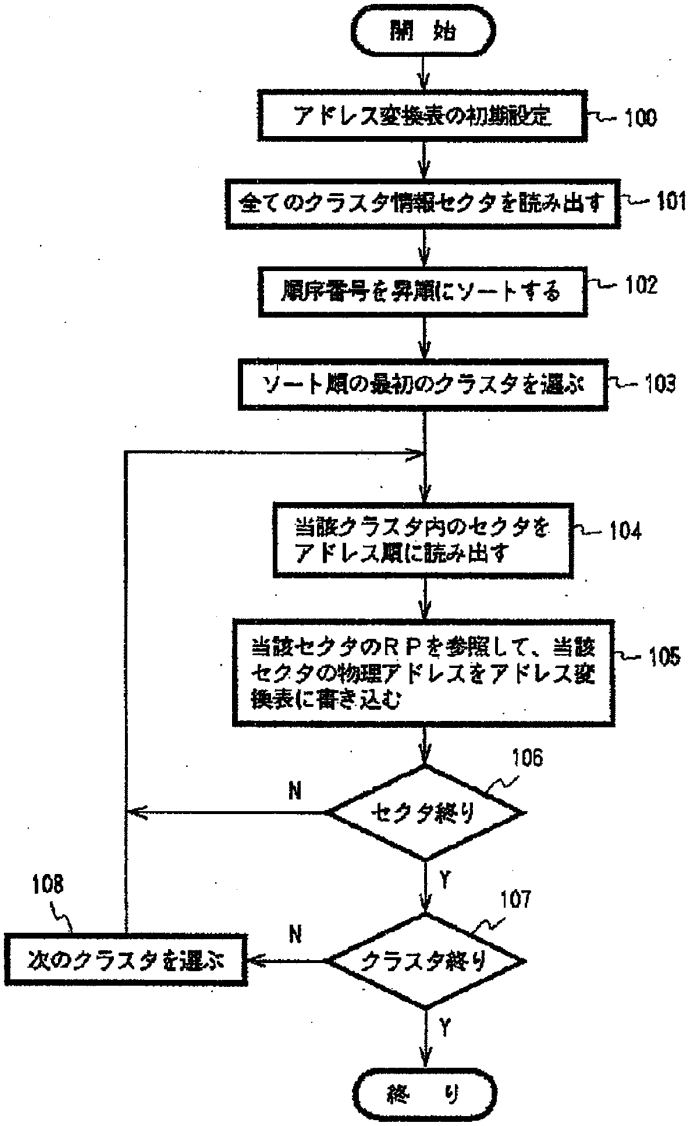
[Drawing 8]


消去するクラスタ（クラスタX）を決定する
80
クラスタXの有効データを他のクラスタに複写する
81
クラスタXの順序番号とクラスタ消去回数を保管する
82
クラスタXを消去する
83
84
初期設定が必要か
N
86
通常オペレーション
Y
初期設定（図9）
85

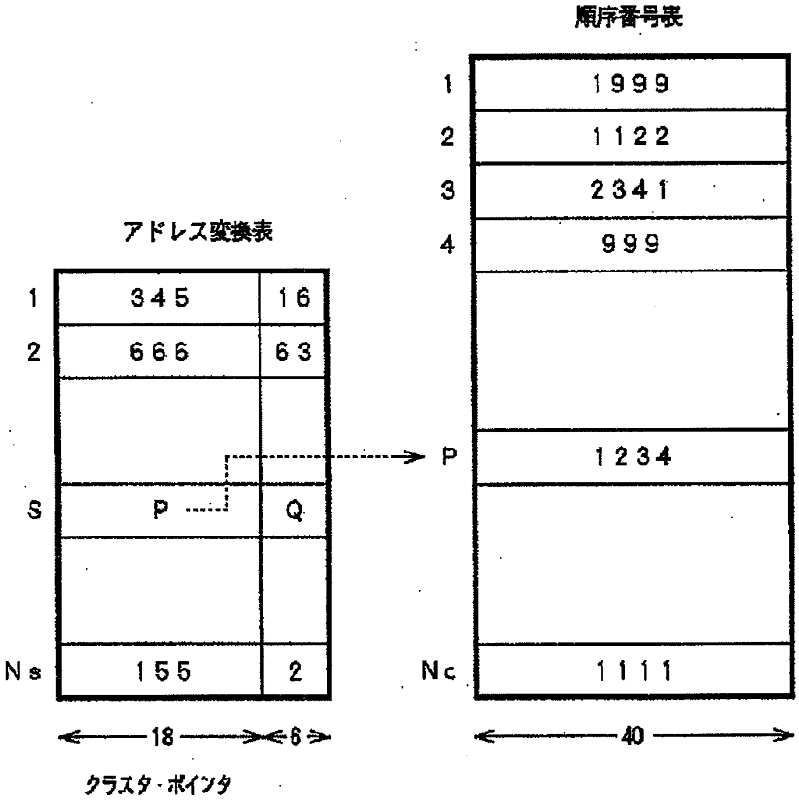
[Drawing 9]


初期設定するクラスタCを決定する
90
現在の最大順序番号Mを求め、M＋１をクラスタCの順序番号として書き込む
91
最大順序番号としてM＋１を保存する
92
クラスタCの保管されていた消去回数に１を加えた値をクラスタCの消去回数として書き込む
93

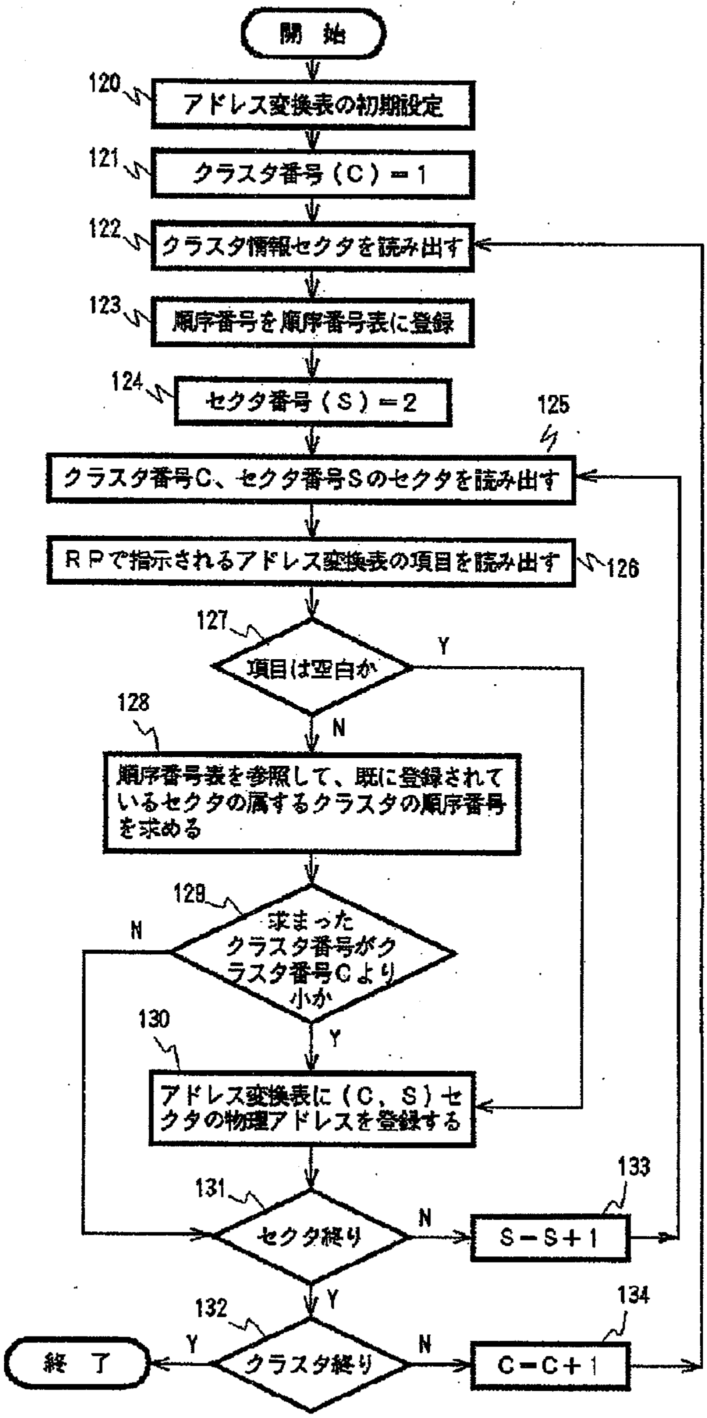
[Drawing 10]


開 始
アドレス変換表の初期設定
100
全てのクラスタ情報セクタを読み出す
101
順序番号を昇順にソートする
102
ソート順の最初のクラスタを選ぶ
103
当該クラスタ内のセクタをアドレス順に読み出す
104
当該セクタのＲＰを参照して、当該セクタの物理アドレスをアドレス変換表に書き込む
105
106
セクタ終り
N
Y
107
クラスタ終り
N
Y
108
次のクラスタを選ぶ
終 り

[Drawing 11]


順序番号表
1999
P
Nc
アドレス変換表
S
P
Q
Ns
クラスタ・ポインタ

[Drawing 12]


開始
120
アドレス変換表の初期設定
121
クラスタ番号（C）＝1
122
クラスタ情報セクタを読み出す
123
順序番号を順序番号表に登録
124
セクタ番号（S）＝2
125
クラスタ番号C、セクタ番号Sのセクタを読み出す
RPで指示されるアドレス変換表の項目を読み出す
126
127
項目は空白か
Y
N
128
順序番号表を参照して、既に登録されているセクタの属するクラスタの順序番号を求める
129
求まったクラスタ番号がクラスタ番号Cより小か
N
Y
130
アドレス変換表に（C，S）セクタの物理アドレスを登録する
131
セクタ終り
N
133
S＝S＋1
Y
132
クラスタ終り
Y
N
134
C＝C＋1
終了

[Translation done.]

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-105661
(43)Date of publication of application : 24.04.1998

---

(51)Int.Cl.

G06K 17/00
G06F 3/06
G06F 3/08
G06F 12/02
G06K 19/07

---

(21)Application number : 09-171430
(22)Date of filing : 27.06.1997

(71)Applicant : SONY CORP
(72)Inventor : KUSAKABE SUSUMU
TAKADA MASAYUKI

---

(30)Priority

Priority number : 08168966 Priority date : 28.06.1996 Priority country : JP

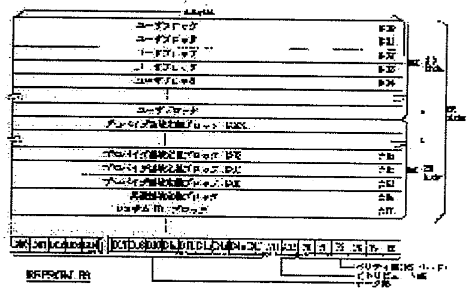
---

(54) INFORMATION PROCESSING METHOD, INFORMATION PROCESSOR AND TRANSMISSION MEDIUM

(57)Abstract:

PROBLEM TO BE SOLVED: To logically suppress the generation of memory collapsion by storing the data of users more than one in a 1st area, setting a 2nd area in the idle area, which is not used as that 1st area, and processing a command while utilizing the storage parts of 1st and 2nd areas.

SOLUTION: When issuing an IC card, an EEPROM 66 registers a provider to provide a system utilizing the IC card, allocates the provider to the common area definition block or provider area definition block of the 1st area to be used by the provider and stores it. Then, the setting of a storage area as the 2nd area to be used by all the providers is stored in the idle area which is not used as that common area definition block or provider area definition block and while utilizing the storage parts of these storage areas, the command is processed by a sequencer. Thus, the generation of memory collapsion can be logically suppressed and the utilization efficiency of a memory can be improved.



LEGAL STATUS

[Date of request for examination] 25.06.2004

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－105661

(43)公開日　平成10年(1998)４月24日

(51)Int.Cl.^{6}	識別記号	ＦＩ	
Ｇ０６Ｋ　17/00		Ｇ０６Ｋ　17/00	Ｆ
			Ｄ
Ｇ０６Ｆ　3/06	３０１	Ｇ０６Ｆ　3/06	３０１Ｊ
3/08		3/08	Ｃ
12/02	５１０	12/02	５１０Ａ

審査請求　未請求　請求項の数33　ＯＬ　（全 32 頁）　最終頁に続く

(21)出願番号	特願平9－171430
(22)出願日	平成９年(1997)６月27日
(31)優先権主張番号	特願平8－168966
(32)優先日	平８(1996)６月28日
(33)優先権主張国	日本（ＪＰ）

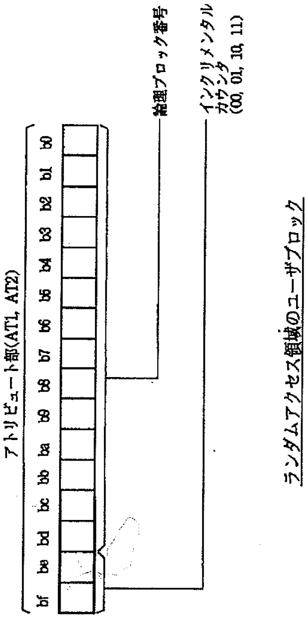
(71)出願人	000002185 ソニー株式会社 東京都品川区北品川６丁目７番35号
(72)発明者	日下部　進 東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者	高田　昌幸 東京都品川区北品川６丁目７番35号　ソニー株式会社内
(74)代理人	弁理士　稲本　義雄

(54)【発明の名称】　**情報処理方法および情報処理装置、並びに伝送媒体**

(57)【要約】

【課題】　メモリコラプションを抑制する。

【解決手段】　ランダムアクセス領域の記憶されているデータを読み出すときにおいては、論理ブロック番号で、読み出すデータ（物理ブロック）を検索し、その論理ブロック番号を有するデータのインクリメンタルカウンタを参照して最も新しいデータを読み出す。ランダムアクセス領域にデータを記憶する場合、既にランダムアクセス領域に記憶されているデータの論理ブロック番号とインクリメンタルカウンタを参照し、不要となった物理ブロックを、ライトバッファとした後、そのライトバッファにデータを書き込む。


論理ブロック番号
インクリメンタルカウンタ
(00, 01, 10, 11)
ランダムアクセス領域のユーザブロック
アトリビュート部(AT1, AT2)
b0
b1
b2
b3
b4
b5
b6
b7
b8
b9
ba
bb
bc
bd
be
bf

【特許請求の範囲】

【請求項１】　所定の利用者からのコマンドを検出するステップと、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、前記第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶部を利用して前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えることを特徴とする情報処理方法。

【請求項２】　前記第２の領域は、１以上のブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項１に記載の情報処理方法。

【請求項３】　所定の利用者からのコマンドを検出するステップと、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、前記第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶部を利用して前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項４】　外部からの入力信号を検出する検出手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、前記第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶手段と、
前記記憶手段を利用して前記入力信号に対応する処理を行う処理手段と、
前記処理手段の処理の結果を外部に出力する出力手段とを備えることを特徴とする情報処理装置。

【請求項５】　前記第２の領域は、１以上のブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項４に記載の情報処理装置。

【請求項６】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさの物理ブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置における情報処理方法において、
前記処理手段が、前記物理ブロックに記憶されるデータに、論理ブロック番号を割り当てるステップと、
前記記憶手段が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶するステップとを備えることを特徴とする情報処理方法。

【請求項７】　前記第２の領域は、１以上の物理ブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項６に記載の情報処理方法。

【請求項８】　前記物理ブロックに記憶されるデータは、同一の前記論理ブロック番号を有するデータの新しさを識別する識別情報をさらに有し、
前記記憶手段は、所定の論理ブロック番号を有する新たなデータを、前記識別情報の値を参照し、同一の前記論理ブロック番号を有するデータのうちの最新のデータが記憶されている物理ブロック以外の物理ブロックに記憶することを特徴とする請求項６に記載の情報処理方法。

【請求項９】　前記識別情報は、前記論理ブロック番号を有するデータの更新数を表すカウンタの値、前記論理ブロック番号を有するデータの記憶時の時刻、または前記論理ブロック番号を有するデータの記憶時のカウンタの値であることを特徴とする請求項８に記載の情報処理方法。

【請求項１０】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさの物理ブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置に用いられるプログラムを伝送する伝送媒体において、
前記処理手段が、前記物理ブロックに記憶されるデータに、論理ブロック番号を割り当てるステップと、
前記記憶手段が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項１１】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、

前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさの物理ブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置において、
前記処理手段は、前記物理ブロックに記憶されるデータに、論理ブロック番号を割り当て、
前記記憶手段は、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶することを特徴とする情報処理装置。

【請求項１２】　前記第２の領域は、１以上の物理ブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項１１に記載の情報処理装置。

【請求項１３】　前記物理ブロックに記憶されるデータは、同一の前記論理ブロック番号を有するデータの新しさを識別する識別情報をさらに有し、
前記記憶手段は、所定の論理ブロック番号を有する新たなデータを、前記識別情報の値を参照し、同一の前記論理ブロック番号を有するデータのうちの最新のデータが記憶されている物理ブロック以外の物理ブロックに記憶することを特徴とする請求項１１に記載の情報処理装置。

【請求項１４】　前記識別情報は、前記論理ブロック番号を有するデータの更新数を表すカウンタの値、前記論理ブロック番号を有するデータの記憶時の時刻、または前記論理ブロック番号を有するデータの記憶時のカウンタの値であることを特徴とする請求項１３に記載の情報処理装置。

【請求項１５】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置における情報処理方法において、
前記第２の領域の所定のブロックのデータは、認識番号を有し、
前記処理手段が、前記利用者より供給された前記コマンドが有する認識番号と、前記データが有する認識番号を比較して、その比較結果に対応して前記コマンドを処理するステップを備えることを特徴とする情報処理方法。

【請求項１６】　前記第２の領域は、１以上のブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項１５に記載の情報処理方法。

【請求項１７】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置に用いられるプログラムを伝送する伝送媒体において、
前記第２の領域の所定のブロックのデータは、認識番号を有し、
前記処理手段は、前記利用者より供給された前記コマンドが有する認識番号と、前記データが有する認識番号を比較して、その比較結果に対応して前記コマンドを処理するステップを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項１８】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域とが形成される記憶手段とを備える情報処理装置において、
前記第２の領域の所定のブロックのデータは、認識番号を有し、
前記処理手段は、前記利用者より供給された前記コマンドが有する認識番号と、前記データが有する認識番号を比較して、その比較結果に対応して前記コマンドを処理することを特徴とする情報処理装置。

【請求項１９】　前記第２の領域は、１以上のブロックを有し、
前記利用者のデータの一部として、前記第２の領域において、その利用者により使用される領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を、前記第１の領域に記憶することを特徴とする請求項１８に記載の情報処理装置。

【請求項２０】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
１以上の利用者のデータを記憶する第１の領域と、前記

第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備える情報処理装置における情報処理方法において、

前記処理手段が、前記ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当てるステップと、

前記第１の領域に、前記利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶した前記記憶手段において、前記最後の番号を有するブロックが、前記最後のブロックである場合、新たなデータを、前記先頭のブロックに記憶し、前記最後の番号を有するブロックが、前記最後のブロックではない場合、前記新たなデータを、前記最後の番号を有するブロックの次のブロックに記憶するステップとを備えることを特徴とする情報処理方法。

【請求項２１】　前記新たなデータと同一のデータを有するブロックがある場合、前記新たなデータは記憶されないことを特徴とする請求項２０に記載の情報処理方法。

【請求項２２】　所定の利用者からのコマンドを検出する検出手段と、

前記コマンドを処理する処理手段と、

前記処理手段の処理の結果を出力する出力手段と、

１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備える情報処理装置に用いられるプログラムを伝送する伝送媒体において、

前記処理手段が、前記ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当てるステップと、

前記第１の領域に、前記利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶した前記記憶手段において、前記最後の番号を有するブロックが、前記最後のブロックである場合、新たなデータを、前記先頭のブロックに記憶し、前記最後の番号を有するブロックが、前記最後のブロックではない場合、前記新たなデータを、前記最後の番号を有するブロックの次のブロックに記憶するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項２３】　所定の利用者からのコマンドを検出する検出手段と、

前記コマンドを処理する処理手段と、

前記処理手段の処理の結果を出力する出力手段と、

１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、

前記処理手段は、前記ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当て、

前記記憶手段は、前記第１の領域に、前記利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶し、前記最後の番号を有するブロックが、前記最後のブロックである場合、新たなデータを、前記先頭のブロックに記憶し、前記最後の番号を有するブロックが、前記最後のブロックではない場合、前記新たなデータを、前記最後の番号を有するブロックの次のブロックに記憶することを特徴とする情報処理装置。

【請求項２４】　前記新たなデータと同一のデータを有するブロックがある場合、前記新たなデータは記憶されないことを特徴とする請求項２３に記載の情報処理装置。

【請求項２５】　所定の利用者からのコマンドを検出するステップと、

１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、

前記処理の結果を出力するステップとを備えることを特徴とする情報処理方法。

【請求項２６】　所定の利用者からのコマンドを検出するステップと、

１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、

前記処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項２７】　所定の利用者からのコマンドを検出する検出手段と、

前記コマンドを処理する処理手段と、

前記処理手段の処理の結果を出力する出力手段と、

１以上の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、

前記記憶手段は、前記第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に

記憶することを特徴とする情報処理装置。

【請求項２８】　所定の利用者からのコマンドを検出するステップと、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域を複数の利用者が共同して使用するデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えることを特徴とする情報処理方法。

【請求項２９】　所定の利用者からのコマンドを検出するステップと、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域を複数の利用者が共同して使用するデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項３０】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、
前記記憶手段は、前記第２の領域における所定の領域を複数の利用者が共同して使用するデータを、前記第１の領域に記憶することを特徴とする情報処理装置。

【請求項３１】　所定の利用者からのコマンドを検出するステップと、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えることを特徴とする情報処理方法。

【請求項３２】　所定の利用者からのコマンドを検出するステップと、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、前記第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に記憶する記憶部を利用して、前記コマンドを処理するステップと、
前記処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする伝送媒体。

【請求項３３】　所定の利用者からのコマンドを検出する検出手段と、
前記コマンドを処理する処理手段と、
前記処理手段の処理の結果を出力する出力手段と、
複数の利用者のデータを記憶する第１の領域と、前記第１の領域に記憶されている前記複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、
前記記憶手段は、前記第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、前記第１の領域に記憶することを特徴とする情報処理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、情報処理方法および情報処理装置、並びに伝送媒体に関し、特に、所定の利用者からのコマンドを受信し、そのコマンドを処理し、処理の結果を送信する情報処理方法および情報処理装置、並びに伝送媒体に関する。

【０００２】

【従来の技術】電子マネーシステムやセキュリティシステムで利用されるＩＣカード（スマートカード）が開発されている。

【０００３】このようなＩＣカードは、各種処理を行うＣＰＵや、処理に必要なデータなどを記憶するメモリを内蔵し、所定のリーダ／ライタ（Ｒ／Ｗ）に接触させた状態で、データの送受信を行っている。

【０００４】また、ＩＣカードの中には、自らはバッテリを有していないバッテリレス型のＩＣカードもある。このようなバッテリレス型のＩＣカードは、Ｒ／Ｗから電力を供給される。

【０００５】

【発明が解決しようとする課題】しかしながら、このようなＩＣカードにおいては、Ｒ／Ｗに接触させた状態で使用することを前提としているので、非接触で使用する場合、電力を取得することが困難であるという問題を有している。

【０００６】また、電磁波を利用して、非接触でＩＣカードとＲ／Ｗとの間でデータの送受信を行うとともに、その電磁波でＩＣカードに必要な電力を供給する方法も考えられるが、このような方法においては、ＩＣカードが内蔵するメモリにアクセスしている途中で、電磁波の

受信状態が不良になった場合、十分な電力が得られなくなり、メモリにおけるデータの整合性に欠陥が生じる（メモリコラプション（Memory Corruption）が生じる）可能性があるという問題を有している。

【０００７】さらに、ＭＳ－ＤＯＳ（Microsoft-Disc Operating System）のＦＡＴ（FileAllocation Table）のように、データが記憶される単位（ＭＳ－ＤＯＳの場合はセクタ）毎に情報を保持すると、データが記憶される領域の大きさに比例した領域が、データ管理のために必要となり、メモリの利用効率が低下するという問題を有している。また、記憶領域を、データが記憶される所定の単位で管理すると、その単位に満たない大きさのデータを記憶するとき、使用されない記憶領域が発生し、さらに、メモリの利用効率が低下するという問題を有している。

【０００８】さらに、上述のＩＣカードにおいては、Ｒ／Ｗに対して一様な処理を行っているため、複数のＲ／Ｗに対応して個別の処理を行うことが困難であるという問題を有している。

【０００９】本発明は、このような状況に鑑みてなされたもので、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用され、所定の大きさの物理ブロック単位で管理される第２の領域とを含む記憶部を利用するとともに、その物理ブロックに記憶されるデータに、論理ブロック番号を割り当て、そのデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶したり、物理ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当て、最後の番号を有する物理ブロックが、最後の物理ブロックである場合、そのデータを、先頭の物理ブロックに記憶し、最後の番号を有する物理ブロックが、最後の物理ブロックではない場合、そのデータを、最後尾の番号を有する物理ブロックの次の物理ブロックに記憶することで、メモリにおけるメモリコラプションの発生を論理的に抑制するものである。

【００１０】また、本発明は、各利用者により使用される領域の先頭の物理ブロックに対応する番号および最後の物理ブロックに対応する番号を保持することで、利用者により使用される領域の大きさではなく、利用者の数に比例した量の情報（先頭の物理ブロックに対応する番号および最後の物理ブロックに対応する番号）で、データを管理することができるようにするものである。

【００１１】さらに、本発明は、上述の記憶部において、第２の領域における所定の領域、および、それぞれ異なるアクセス権を規定する複数のデータを、１利用者に対応して第１の領域に記憶したり、第２の領域における所定の領域を規定するデータを、複数の利用者に対応して、第１の領域に記憶することで、複数の利用者（Ｒ／Ｗ）に対応して個別の処理を行うことができるようにするものである。

【００１２】

【課題を解決するための手段】請求項１に記載の情報処理方法は、所定の利用者からのコマンドを検出するステップと、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶部を利用してコマンドを処理するステップと、処理の結果を出力するステップとを備えることを特徴とする。

【００１３】請求項３に記載の伝送媒体は、所定の利用者からのコマンドを検出するステップと、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶部を利用してコマンドを処理するステップと、処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする。

【００１４】請求項４に記載の情報処理装置は、外部からの入力信号を検出する検出手段と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶手段と、記憶手段を利用して入力信号に対応する処理を行う処理手段と、処理手段の処理の結果を外部に出力する出力手段とを備えることを特徴とする。

【００１５】請求項６に記載の情報処理方法は、処理手段が、物理ブロックに記憶されるデータに、論理ブロック番号を割り当てるステップと、記憶手段が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶するステップとを備えることを特徴とする。

【００１６】請求項１０に記載の伝送媒体は、処理手段が、物理ブロックに記憶されるデータに、論理ブロック番号を割り当てるステップと、記憶手段が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶するステップとを備えるプログラムを伝送することを特徴とする。

【００１７】請求項１１に記載の情報処理装置は、処理手段は、物理ブロックに記憶されるデータに、論理ブロック番号を割り当て、記憶手段は、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶することを特徴とする。

【００１８】請求項１５に記載の情報処理方法は、第２の領域の所定のブロックのデータは、認識番号を有し、処理手段が、利用者より供給されたコマンドが有する認

識番号と、データが有する認識番号を比較して、その比較結果に対応してコマンドを処理するステップを備えることを特徴とする。

【００１９】請求項１７に記載の伝送媒体は、第２の領域の所定のブロックのデータは、認識番号を有し、処理手段は、利用者より供給されたコマンドが有する認識番号と、データが有する認識番号を比較して、その比較結果に対応してコマンドを処理するステップを備えるプログラムを伝送することを特徴とする。

【００２０】請求項１８に記載の情報処理装置は、第２の領域の所定のブロックのデータは、認識番号を有し、処理手段は、利用者より供給されたコマンドが有する認識番号と、データが有する認識番号を比較して、その比較結果に対応してコマンドを処理することを特徴とする。

【００２１】請求項２０に記載の情報処理方法は、処理手段が、ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当てるステップと、第１の領域に、利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶した記憶手段において、最後の番号を有するブロックが、最後のブロックである場合、新たなデータを、先頭のブロックに記憶し、最後の番号を有するブロックが、最後のブロックではない場合、新たなデータを、最後の番号を有するブロックの次のブロックに記憶するステップとを備えることを特徴とする。

【００２２】請求項２２に記載の伝送媒体は、処理手段が、ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当てるステップと、第１の領域に、利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶した記憶手段において、最後の番号を有するブロックが、最後のブロックである場合、新たなデータを、先頭のブロックに記憶し、最後の番号を有するブロックが、最後のブロックではない場合、新たなデータを、最後の番号を有するブロックの次のブロックに記憶するステップとを備えるプログラムを伝送することを特徴とする。

【００２３】請求項２３に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段と、処理手段の処理の結果を出力する出力手段と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、処理手段は、ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当て、記憶手段は、第１の領域に、利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶し、最後の番号を有するブロックが、最後のブロックである場合、新たなデータを、先頭のブロックに記憶し、最後の番号を有するブロックが、最後のブロックではない場合、新たなデータを、最後の番号を有するブロックの次のブロックに記憶することを特徴とする。

【００２４】請求項２５に記載の情報処理方法は、所定の利用者からのコマンドを検出するステップと、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えることを特徴とする。

【００２５】請求項２６に記載の伝送媒体は、所定の利用者からのコマンドを検出するステップと、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする。

【００２６】請求項２７に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段と、処理手段の処理の結果を出力する出力手段と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、記憶手段は、第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶することを特徴とする。

【００２７】請求項２８に記載の情報処理方法は、所定の利用者からのコマンドを検出するステップと、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域を複数の利用者が共同して使用するデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えることを特徴とする。

【００２８】請求項２９に記載の伝送媒体は、所定の利用者からのコマンドを検出するステップと、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域を複数の利用者が共同し

て使用するデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする。

【００２９】請求項３０に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段と、処理手段の処理の結果を出力する出力手段と、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、記憶手段は、第２の領域における所定の領域を複数の利用者が共同して使用するデータを、第１の領域に記憶することを特徴とする。

【００３０】請求項３１に記載の情報処理方法は、所定の利用者からのコマンドを検出するステップと、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えることを特徴とする。

【００３１】請求項３２に記載の伝送媒体は、所定の利用者からのコマンドを検出するステップと、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理するステップと、処理の結果を出力するステップとを備えるプログラムを伝送することを特徴とする。

【００３２】請求項３３に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段と、処理手段の処理の結果を出力する出力手段と、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段とを備え、記憶手段は、第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶することを特徴とする。

【００３３】請求項１に記載の情報処理方法、請求項３に記載の伝送媒体、および請求項４に記載の情報処理装置においては、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用され、第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶部を利用してコマンドが処理される。

【００３４】請求項６に記載の情報処理方法、請求項１０に記載の伝送媒体、および請求項１１に記載の情報処理装置においては、処理手段が、物理ブロックに記憶されるデータに、論理ブロック番号を割り当て、記憶手段が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶する。

【００３５】請求項１５に記載の情報処理方法、請求項１７に記載の伝送媒体、および請求項１８に記載の情報処理装置においては、第２の領域の所定のブロックのデータが、認識番号を有し、処理手段が、利用者により供給されたコマンドが有する認識番号と、データが有する認識番号を比較して、その比較結果に対応してコマンドを処理する。

【００３６】請求項２０に記載の情報処理方法、請求項２２に記載の伝送媒体、および請求項２３に記載の情報処理装置においては、処理手段で、ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当てられ、記憶手段で、最後の番号を有するブロックが、最後のブロックである場合、新たなデータが、先頭のブロックに記憶され、最後の番号を有するブロックが、最後のブロックではない場合、新たなデータが、最後の番号を有するブロックの次のブロックに記憶される。

【００３７】請求項２５に記載の情報処理方法、請求項２６に記載の伝送媒体、および請求項２７に記載の情報処理装置においては、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域に対して、および、各利用者に対してそれぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理する。

【００３８】請求項２８に記載の情報処理方法、請求項２９に記載の伝送媒体、および請求項３０に記載の情報処理装置においては、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用され、所定の大きさのブロック単位で管理される第２の領域が形成され、第１の領域において、第２の領域における所定の領域を複数の利用者が共同して使用するデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理する。

【００３９】請求項３１に記載の情報処理方法、請求項３２に記載の伝送媒体、および請求項３３に記載の情報処理装置においては、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用され、所定の大きさのブロック単位で管理される第２の領域が形成され、第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセセ

ス権を規定する複数のデータを、第１の領域に記憶する記憶部を利用して、コマンドを処理する。

【００４０】

【発明の実施の形態】以下に本発明の実施の形態を説明するが、特許請求の範囲に記載の発明の各手段と以下の実施の形態との対応関係を明らかにするために、各手段の後の括弧内に、対応する実施の形態（但し一例）を付加して本発明の特徴を記述すると、次のようになる。但し勿論この記載は、各手段を記載したものに限定することを意味するものではない。

【００４１】請求項４に記載の情報処理装置は、外部からの入力信号を検出する検出手段（例えば図３のＢＰＳＫ復調回路６２）と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、第１の領域として使用されていない空き領域に設定される第２の領域とが形成される記憶手段（例えば図３のＥＥＰＲＯＭ６６）と、記憶手段を利用して入力信号に対応する処理を行う処理手段（例えば図３のシーケンサ９１）と、処理手段の処理の結果を外部に出力する出力手段（例えば図３のＢＰＳＫ変調回路６８）とを備えることを特徴とする。

【００４２】請求項１１に記載の情報処理装置は、処理手段（例えば図３のシーケンサ９１）が、物理ブロックに記憶されるデータに、論理ブロック番号を割り当て、記憶手段（例えば図３のＥＥＰＲＯＭ６６）が、所定の論理ブロック番号を有する新たなデータを、その論理ブロック番号を有するデータが記憶されている物理ブロック以外の物理ブロックに記憶することを特徴とする。

【００４３】請求項１８に記載の情報処理装置は、第２の領域の所定のブロックのデータは、認識番号を有し、処理手段（例えば図３のシーケンサ９１）が、利用者より供給されたコマンドが有する認識番号と、データが有する認識番号を比較して、その比較結果に対応してコマンドを処理することを特徴とする。

【００４４】請求項２３に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段（例えば図３のシーケンサ９１）と、処理手段の処理の結果を出力する出力手段と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段（例えば図３のＥＥＰＲＯＭ６６）とを備え、処理手段は、ブロックに記憶されるデータに、記憶される順番に対応する番号を割り当て、記憶手段は、第１の領域に、利用者が使用する領域の先頭のブロックに対応する番号と最後のブロックに対応する番号を記憶し、最後の番号を有するブロックが、最後のブロックである場合、新たなデータを、先頭のブロックに記憶し、最後の番号を有するブロックが、最後のブロックではない場合、新たなデータを、最後の番号を有するブロックの次のブロックに記憶することを特徴とする。

【００４５】請求項２７に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段と、コマンドを処理する処理手段（例えば図３のシーケンサ９１）と、処理手段の処理の結果を出力する出力手段と、１以上の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている１以上の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段（例えば図３のＥＥＰＲＯＭ６６）とを備え、記憶手段は、第２の領域における所定の領域に対して、および、各利用者に対して、それぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶することを特徴とする。

【００４６】請求項３０に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段（例えば図３のＢＰＳＫ復調回路６２）と、コマンドを処理する処理手段（例えば図３のシーケンサ９１）と、処理手段の処理の結果を出力する出力手段（例えば図３のＢＰＳＫ変調回路６８）と、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段（例えば図３のＥＥＰＲＯＭ６６）とを備え、記憶手段は、第２の領域における所定の領域を複数の利用者が共同して使用するデータを、第１の領域に記憶することを特徴とする。

【００４７】請求項３３に記載の情報処理装置は、所定の利用者からのコマンドを検出する検出手段（例えば図３のＢＰＳＫ復調回路６２）と、コマンドを処理する処理手段（例えば図３のシーケンサ９１）と、処理手段の処理の結果を出力する出力手段（例えば図３のＢＰＳＫ変調回路６８）と、複数の利用者のデータを記憶する第１の領域と、第１の領域に記憶されている複数の利用者により使用される、所定の大きさのブロック単位で管理される第２の領域が形成される記憶手段（例えば図３のＥＥＰＲＯＭ６６）とを備え、記憶手段は、第２の領域における所定の領域、および、複数の利用者のそれぞれ異なるアクセス権を規定する複数のデータを、第１の領域に記憶することを特徴とする。

【００４８】図１は、Ｒ／Ｗ１およびＩＣカード２を利用した非接触カードシステムの一例を示している。Ｒ／Ｗ１およびＩＣカード２は、電磁波を利用して非接触で、データの送受信を行う。

【００４９】Ｒ／Ｗ１が、所定のコマンドをＩＣカード２に送信すると、ＩＣカード２は、そのコマンドを受信し、そのコマンドに対応する処理を行うようになされている。

【００５０】本発明の情報処理装置の一実施の形態であるＩＣカード２は、Ｒ／Ｗ１がデータをＩＣカード２に送信すると、そのコマンドを受信し、受信したコマンドを処理し、その処理結果に対応する応答データをＲ／Ｗ

１に送信するようになされている。

【００５１】また、Ｒ／Ｗ１は、所定のインタフェース（例えばＲＳ−４８５Ａ）を介してコントローラ３に接続され、コントローラ３より所定の制御信号を供給され、その制御信号に従って、処理を行うようになされている。

【００５２】図２は、Ｒ／Ｗ１の構成を示している。

【００５３】ＩＣ２１においては、データの処理を行うＤＰＵ（Data Processing Unit）３１、ＩＣカード２に送信するデータおよびＩＣカード２から受信したデータの処理を行うＳＰＵ（Signal Processing Unit）３２、コントローラ３との通信を行うＳＣＣ（Serial Communication Controller）３３、および、データの処理に必要な情報を予め記憶しているＲＯＭ部４１と、処理途中のデータを一時的に記憶するＲＡＭ部４２で構成されるメモリ部３４が、バスを介して接続されている。

【００５４】また、このバスには、所定のデータを記憶するフラッシュメモリ２２も接続されている。

【００５５】ＤＰＵ３１は、ＩＣカード２に送信するコマンドをＳＰＵ３２に出力するとともに、ＩＣカード２から受信した応答データをＳＰＵ３２から受け取るようになされている。

【００５６】ＳＰＵ３２は、ＩＣカード２に送信するコマンドに対して所定の処理（例えば、ＢＰＳＫ（BiPhase Shift Keying）変調（後述））を行った後、変調回路２３に出力するとともに、ＩＣカード２により送信されてきた応答データを復調回路２５から受け取り、そのデータに対して所定の処理を行うようになされている。

【００５７】変調回路２３は、発振器２６より供給された所定の周波数（例えば１３．５６ＭＨｚ）の搬送波を、ＳＰＵ３２より供給されたデータで、ＡＳＫ（AmplitudeShift Keying）変調し、生成された変調波をアンテナ２７を介して、電磁波としてＩＣカード２に出力するようになされている。このとき、変調回路２３は、変調度を１未満にして、ＡＳＫ変調を行う。即ち、データがローレベルのときにおいても、変調波の最大振幅がゼロにならないようにする。

【００５８】復調回路２４は、アンテナ２７を介して受信した変調波（ＡＳＫ変調波）を復調し、復調されたデータをＳＰＵ３２に出力するようになされている。

【００５９】図３は、ＩＣカード２の構成例を示している。このＩＣカード２においては、ＩＣ５１は、アンテナ５３を介して、Ｒ／Ｗ１により送信された変調波を受信するようになされている。なお、コンデンサ５２は、アンテナ５３とともにＬＣ回路を構成し、所定の周波数（キャリア周波数）の電磁波に同調するようになされている。

【００６０】ＩＣ５１においては、ＲＦインタフェース部６１は、ＡＳＫ復調部８１で、アンテナ５３を介して受信した変調波（ＡＳＫ変調波）を検波して復調し、復調後のデータをＢＰＳＫ復調回路６２およびＰＬＬ（Phase Locked Loop）部６３に出力するとともに、電圧レギュレータ８２で、ＡＳＫ復調部８１が検波した信号を安定化し、各回路に直流電力として供給するようになされている。

【００６１】また、ＲＦインタフェース部６１は、発振回路８３でデータのクロック周波数と同一の周波数の信号を発振し、その信号をＰＬＬ部６３に出力するようになされている。

【００６２】そして、ＲＦインタフェース部６１のＡＳＫ変調部８４は、演算部６４より供給されたデータに対応して、ＩＣカード２の電源としてのアンテナ５３の負荷を変動させる（例えば、データに対応して所定のスイッチング素子をオン／オフさせ、スイッチング素子がオン状態であるときだけ所定の負荷をアンテナ５３に並列に接続させる）ことにより、アンテナ５３を介して受信している変調波（ＩＣカード２からデータを送信するときは、変調波の最大振幅を一定にしている）をＡＳＫ変調し、その変調成分を、アンテナ５３を介してＲ／Ｗ１に送信する（Ｒ／Ｗ１のアンテナ２７の端子電圧を変動させる）ようになされている。

【００６３】ＰＬＬ部６３は、ＡＳＫ復調部８１より供給されたデータより、そのデータに同期したクロック信号を生成し、そのクロック信号をＢＰＳＫ復調回路６２およびＢＰＳＫ変調回路６８に出力するようになされている。

【００６４】ＢＰＳＫ復調回路６２は、ＡＳＫ復調部８１で復調されたデータが、ＢＰＳＫ変調されている場合、ＰＬＬ部６３より供給されたクロック信号に従って、そのデータの復調を行い、復調したデータを演算部６４に出力するようになされている。

【００６５】演算部６４は、ＢＰＳＫ復調回路６２より供給されたデータが暗号化されている場合、そのデータを暗号／復号部９２で復号化した後、そのデータを、コマンドとして、シーケンサ９１で処理するようになされている。なお、データが暗号化されていない場合、ＢＰＳＫ復調回路６２より供給されたデータは、暗号／復号部９２を介さず、シーケンサ９１に、直接供給される。

【００６６】シーケンサ９１は、供給されたコマンドに対応する処理を行うようになされている。例えば、このとき、シーケンサ９１は、ＥＥＰＲＯＭ６６に記憶されているデータの処理を行う。

【００６７】演算部６４のパリティ演算部９３は、ＥＥＰＲＯＭ６６に記憶されるデータや、ＥＥＰＲＯＭ６６に記憶されているデータから、パリティとして、リードソロモン符号を算出するようになされている。

【００６８】さらに、演算部６４は、シーケンサ９１で所定の処理を行った後、その処理に対応する応答データ（Ｒ／Ｗ１に送信するデータ）をＢＰＳＫ変調回路６８に出力するようになされている。

【００６９】ＢＰＳＫ変調回路６８は、演算部６４より供給されたデータをＢＰＳＫ変調し（後述）、変調後のデータをＲＦインタフェース部６１のＡＳＫ変調部８４に出力するようになされている。

【００７０】ＲＡＭ６７は、シーケンサ９１が処理を行うとき、処理の途中のデータなどを、一時的に記憶するようになされている。

【００７１】ＥＥＰＲＯＭ（Electrically Erasable and Programmable ROM）６６は、不揮発性のメモリであり、ＩＣカード２がＲ／Ｗ１との通信を終了し、電力供給が停止した後も、データを記憶し続けるようになされている。ＲＯＭ６５には、シーケンサ９１がＲ／Ｗ１からのコマンドを処理するのに必要な基礎的なプログラムが記憶されている。

【００７２】図４は、ＥＥＰＲＯＭ６６のメモリの割り当ての一例を示している。

【００７３】ＥＥＰＲＯＭ６６は、４０バイトの物理ブロックを、２５６個有している。各物理ブロックは、３２バイトのデータ部（Ｄ００乃至Ｄ１ｆ）、２バイトのアトリビュート部（ＡＴ１，ＡＴ２）、および、６バイトのパリティ部（Ｐ０乃至Ｐ５）の合計４０バイトで構成されている。

【００７４】ＥＥＰＲＯＭ６６の物理ブロック番号ｆｆＨ（Ｈは１６進数を表している）は、システムＩＤブロックに割り当てられている。システムＩＤブロックは、ＩＣカード２のセキュリティに関する情報を記憶している。

【００７５】次に、物理ブロック番号ｆｄＨから００Ｈに向かって順次、物理ブロックが、共通領域定義ブロック（Common Area Definition Block）（第１の領域）またはプロバイダ領域定義ブロック（Provider Area Definition Block）（第１の領域）に割り当てられている。

【００７６】ＥＥＰＲＯＭ６６には、ＩＣカード２が発行されるとき、所定の装置（発行機）により、このＩＣカード２を利用したシステムを提供する者（プロバイダ）が登録される。発行機は、１プロバイダ当たり１物理ブロックで、プロバイダ領域定義ブロックを、物理ブロック番号ｆｄＨから００Ｈに向かって順次使用し、プロバイダを登録する。

【００７７】共通領域定義ブロックおよびプロバイダ領域定義ブロックは、プロバイダが使用する記憶領域の位置などの情報を記憶している。

【００７８】そして、システムＩＤブロック、共通領域定義ブロック、およびプロバイダ領域定義ブロックとして利用されない物理ブロックが、プロバイダにより使用されるユーザブロック（User Block）に割り当てられる。

【００７９】図５は、システムＩＤブロックに対する各データの割り当ての一例を示している。

【００８０】データ部のＤ００乃至Ｄ０ｆは、ＥＥＰＲＯＭ６６の製造時の製造ＩＤ（Manufacture ID）（ＩＤｍ）が記憶されている。領域Ｄ００乃至Ｄ０３、領域Ｄ０４乃至Ｄ０７、領域Ｄ０８乃至Ｄ０ｂ、および、領域Ｄ０ｃ乃至Ｄ０ｆは、ＥＥＰＲＯＭ６６のＩＣコード、ＥＥＰＲＯＭ６６を作成した製造機のコード（Manufacture Equipment Code）、ＥＥＰＲＯＭ６６の製造日（Manufacture Date）、および、ＥＥＰＲＯＭ６６の製造シリアルナンバ（Manufacture Serial Number）を、それぞれ記憶している。

【００８１】このＩＤｍの情報を利用することにより、すべてのＩＣカード２（ＥＥＰＲＯＭ６６）の識別を行うことができる。なお、製造日は、２０００年1月１日を００００Ｈとして、２０００年1月１日からの日数とする。なお、製造日が１９９０年台である場合、製造日は、２の補数を利用して、２０００年1月１日からの負の日数として表現される。

【００８２】データ部のＤ１０乃至Ｄ１ｆは、このＩＤカード２を発行したときの発行ＩＤ（Issue ID）（ＩＤｉ）が記憶されている。領域Ｄ１０乃至Ｄ１３、領域Ｄ１４乃至Ｄ１７、領域Ｄ１８乃至Ｄ１ｂ、および、領域Ｄ１ｃ乃至１ｆは、ＩＣカード２の属するカテゴリおよびグループを示すカテゴリ／グループナンバ、このＩＣカード２を発行した発行機のコード、ＩＣカード２を発行した日にち、および、ＩＣカード２の有効期限を、それぞれ記憶している。

【００８３】図６は、システムＩＤブロックのアトリビュート部を示している。アトリビュート部は、登録されているプロバイダの数を記憶している。発行機は、１つのプロバイダを登録する際に、１つの物理ブロックを使用し、そのとき、このアトリビュート部の値を更新する。

【００８４】アトリビュート部の値は、製造時に、ゼロに設定されており、その後、発行機が、ＩＣカード２にプロバイダを登録するとき、アトリビュート部の値を、登録されるプロバイダの数で更新する。

【００８５】システムＩＤブロックのパリティ部は、データ部およびアトリビュート部の各ビットの値から、パリティ演算部９３で演算されるリードソロモン符号（ＲＳ符号）を記憶している。従って、パリティ部の値は、データ部またはアトリビュート部が更新される度に、演算し直される。

【００８６】図７は、共通領域定義ブロックおよびプロバイダ領域定義ブロックの一例を示している。なお、これらのブロックは、ＩＣカード２が発行されるとき、予め、発行機により書き込まれている。

【００８７】共通領域定義ブロックは、ＥＥＰＲＯＭ６６の物理ブロック番号ｆｅＨに配置され、全プロバイダにより使用される記憶領域（共通領域（Common Area））（第２の領域）の設定を記憶している。

【００８８】プロバイダ領域定義ブロックは、ＥＥＰＲ

ＯＭ６６の物理ブロック番号ｆｄＨから、００Ｈに向かって配置され、１プロバイダ当たり１物理ブロックで、プロバイダの情報を記憶している。

【００８９】図７に示すように、領域定義ブロック（共通領域定義ブロックおよびプロバイダ領域定義ブロック）のデータ部Ｄ００乃至Ｄ１ｆの領域Ｄ００，Ｄ０１は、プロバイダの種類を示すプロバイダコード（Provider Code）を記憶している。共通領域定義ブロックの場合、領域Ｄ００，Ｄ０１の値は、００００Ｈとされ、プロバイダ領域定義ブロックの場合、領域Ｄ００，Ｄ０１の値は、０００１Ｈ乃至ＦＦＦＦＨのいずれかの値とされている。

【００９０】領域定義ブロックのデータ部の領域Ｄ０２乃至Ｄ０５は、このプロバイダが使用する記憶領域（プロバイダ領域（Provider Area））（第２の領域）の先頭の物理ブロックの番号BN_0（領域Ｄ０２，Ｄ０３）と、終わりの物理ブロックの番号の次の物理ブロックの番号BN_1（領域Ｄ０４，Ｄ０５）（$BN_1>BN_0$）で構成されるアロケーションテーブル（Allocation Table）を記憶している。プロバイダ領域は、図８に示すように、システムブロック（システムＩＤブロック、領域定義ブロック）を除く、ＥＥＰＲＯＭ６６の所定の位置（物理ブロック番号BN_0乃至（BN_1-1））に設定される。

【００９１】このように、BN_0とBN_1でプロバイダ領域を指定しているので、プロバイダ（利用者）により使用される領域の大きさではなく、プロバイダの数に比例した量の情報で、データを管理することができ、メモリの利用効率を高くすることができる。

【００９２】領域定義ブロックのデータ部の領域Ｄ０６乃至Ｄ０９は、プロバイダが使用する記憶領域のうち、ランダムアクセス領域（後述）のブロック数B_{RA}（領域Ｄ０６，Ｄ０７）と、ランダムアクセス領域中のリード／ライトブロックのブロック数B_{RW}（領域Ｄ０８，Ｄ０９）で構成されるパーティションテーブル（Partition Table）を記憶している。このとき、ランダムアクセス領域のブロック数B_{RA}は、式

$$B_{RA}=0$$

または、式

$$2\times n\leqq B_{RA}\leqq BN_1-BN_0$$

（ｎはライトバッファ（後述）の数）を満足する値に設定され、リード／ライトブロックのブロック数B_{RW}は、$B_{RA}=0$である場合においては、$B_{RW}=0$に設定され、$B_{RA}\neq 0$である場合においては、式

$$n\leqq B_{RW}\leqq B_{RA}-n$$

を満足する値に設定される。

【００９３】領域定義ブロックのデータ部の領域Ｄ０ａ，Ｄ０ｂは、ランダムアクセス領域のライトバッファの数ｎを記憶している。ｎ個のライトバッファは、ｎ個のデータを、ランダムアクセス領域の論理ブロック番号００Ｈ乃至（００＋ｎ（１６進数表示））Ｈに、同時に記憶させるときに利用される。なお、ランダムアクセス領域のうち、その他の論理ブロック番号を有する物理ブロックにデータを記憶するときにおいては、ライトバッファは、１個だけ利用される。

【００９４】以上のように、領域定義ブロックに従って、図８に示すように、物理ブロック番号BN_0乃至（BN_1-1）の領域（プロバイダ領域または共通領域）は、プロバイダコードで指定されるプロバイダに割り当てられ、さらに、その領域（プロバイダ領域または共通領域）のうちのB_{RA}個の物理ブロックが、ランダムアクセス領域に割り当てられ、残りの物理ブロックがシーケンシャルアクセス領域（後述）に割り当てられている。

【００９５】さらに、領域定義ブロックに従って、図８に示すように、ランダムアクセス領域は、B_{RW}個のリード／ライトブロック、リードオンリーブロック、および、ｎ個のライトバッファに、論理的に割り当てられている。なお、リード／ライトブロックおよびライトバッファ以外の物理ブロックが、リードオンリーブロックに割り当てられる。

【００９６】領域定義ブロックのデータ部の領域Ｄ０ｃ，Ｄ０ｄは、このプロバイダが利用する記憶領域（ランダムアクセス領域）におけるパースブロック（Purse Block）（後述）に対するアクセス権の情報を有するパースブロックパーミッションを記憶している。

【００９７】図９は、パースブロックパーミッションの一例を示している。

【００９８】パースブロックパーミッション（１６ビット、b_0乃至b_f）は、パースブロックに対する読み出し、加算命令、および、減算命令の許可または不許可を示している。

【００９９】共通領域定義ブロックのパースブロックパーミッションは、共通領域定義ブロックで設定される記憶領域（共通領域）においてパースブロックを使用するか否かを、領域（ビット）b_bに記憶している。即ち、$b_b=0$の場合、パースブロックを使用しない。$b_b=1$の場合、パースブロックを使用する。そして、共通領域定義ブロックのパースブロックパーミッションにおけるその他の領域（ビット）は、特に使用されない。なお、$b_b=1$の場合、論理ブロック番号が００Ｈであるリード／ライトブロックが、パースブロックとして使用される。

【０１００】次に、プロバイダ領域定義ブロックのパースブロックパーミッションにおいては、このプロバイダ領域定義ブロックで設定された記憶領域でパースブロックを使用するか否かを領域b_3に記憶している。即ち、$b_3=0$の場合、パースブロックを使用しない。$b_3=1$の場合、パースブロックを使用する。なお、$b_3=1$の場合、論理ブロック番号が００Ｈであるリード／ライト

ブロックが、パースブロックとして使用される。

【０１０１】そして、そのパースブロックに対する加算命令の可否を領域b_2に記憶し、そのパースブロックに対する減算命令の可否を領域b_1に記憶し、そのパースブロックに対する読み出しの可否を領域b_0に記憶している（b_i＝１（ｉ＝０，１，２）の場合、その命令は許可され、b_i＝０の場合、その命令は許可されない）。また、共通領域定義ブロックで設定された記憶領域でパースブロックを使用するか否かを領域b_bに記憶している。なお、b_bには、共通領域定義ブロックのパースブロックパーミッションのb_bと同じ値が記憶されている。

【０１０２】さらに、そのパースブロックに対する加算命令の可否を領域b_aに記憶し、そのパースブロックに対する減算命令の可否を領域b_9に記憶し、そのパースブロックに対する読み出しの可否を領域b_8に記憶している（b_i＝１（ｉ＝８，９，ａ）の場合、その命令は許可され、b_i＝０の場合、その命令は許可されない）。

【０１０３】図７の領域定義ブロックのデータ部の領域Ｄ０ｅ，Ｄ０ｆは、プロバイダ（Ｒ／Ｗ１）の認証、並びに、暗号化および復号化に利用されるセキュリティキー（共通鍵とプロバイダ鍵）のバージョン番号を記憶し、領域Ｄ１０乃至１ｆは、そのセキュリティキーを記憶している。

【０１０４】なお、Ｒ／Ｗ１がポーリングを行ったときは、ＩＣカード２は、この２つのキー（共通鍵とプロバイダ鍵）のバージョン番号を返送する。従って、Ｒ／Ｗ１とＩＣカード２との間の認証においては、複数のバージョンのセキュリティキーを使い分けることができる。

【０１０５】そして、領域定義ブロックのアトリビュート部ＡＴ１，ＡＴ２は、予備として設けられており、特に情報は記憶されていない。領域定義ブロックのパリティ部は、データ部およびアトリビュート部のすべてのビットの値より演算されるパリティ（ＲＳ符号）を記憶している。

【０１０６】このように、発行機により設定される領域定義ブロックは、プロバイダコード、アロケーションテーブル、パーティションテーブル、パースブロックパーミッション、セキュリティキーバージョン、および、セキュリティキーを記憶している。

【０１０７】図１０は、ユーザブロックの一例を示している。図４を参照して上述したように、ＥＥＰＲＯＭ６６のメモリ空間のうち、システムＩＤブロック、共通領域定義ブロック、プロバイダ領域定義ブロック以外の物理ブロックが、ユーザブロックとして、プロバイダにより使用される。

【０１０８】例えば、図４に示すように、メモリ空間が２５６ブロックで構成されている場合において、８個のプロバイダが登録されると、システムＩＤブロック、共通領域定義ブロック、および、８個のプロバイダ領域定義ブロックの合計１０（＝１＋１＋８）個のシステムブロック以外の、２４６（＝２５６－１０）ブロックがユーザブロックとして使用される。また、４０個のプロバイダを登録すると、システムブロックは合計４２（＝１＋１＋４０）個となり、２１４（＝２５６－４２）個のユーザブロックが確保される。

【０１０９】ユーザブロックは、領域定義ブロックのアロケーションテーブル（図７）に従って、各プロバイダに割り当てられる。なお、プロバイダは、アロケーションテーブルを参照して、予め割り当てられているユーザブロックを使用するので、アロケーションテーブルで割り当てられた領域（プロバイダ領域または共通領域）以外にアクセスすることはない。

【０１１０】アロケーションテーブルで割り当てられた領域（プロバイダ領域または共通領域）のユーザブロックは、上述のパーティションテーブル（図７）に従って、ランダムアクセス領域と、シーケンシャルアクセス領域に割り当てられている。

【０１１１】さらに、ランダムアクセス領域のユーザブロックは、リード／ライトブロック、リードオンリーブロック、および、ライトバッファのいずれかとして利用され、これらのブロックの数は、上述のように、パーティションテーブルおよびライトバッファの数に従って設定されている。

【０１１２】このように割り当てられているユーザブロックのデータ部Ｄ００乃至Ｄ１ｆは、そのユーザブロックが割り当てられているプロバイダによる処理に応じて使用される。

【０１１３】ランダムアクセス領域のユーザブロックのアトリビュート部は、図１１に示すように、インクリメンタルカウンタ（Incremental Counter）（ビットb_f，b_e）および論理ブロック番号（ビットb_d乃至b_0）を記憶している。

【０１１４】論理ブロック番号とインクリメンタルカウンタは、ランダムアクセス領域のユーザブロックにアクセスするときに利用される。

【０１１５】ランダムアクセス領域に記憶されているデータを読み出すときにおいては、論理ブロック番号で、読み出すデータ（物理ブロック）を検索し、その論理ブロック番号を有するデータのインクリメンタルカウンタを参照して最も新しいデータを読み出す。

【０１１６】一方、ランダムアクセス領域にデータを記憶する場合、既にランダムアクセス領域に記憶されているデータの論理ブロック番号とインクリメンタルカウンタを参照し、不要となった物理ブロック（後述）を、ライトバッファとした後、そのライトバッファにデータを書き込む。

【０１１７】なお、上述の領域定義ブロックのパースブロックパーミッションが、パースブロックを使用するよ

うに設定されている場合、論理ブロック番号が００Ｈであるリード／ライトブロックは、パースブロックとして使用される。

【０１１８】パースブロックは、データの加算および減算を頻繁に行うとき、既に記憶している値を読み出したくないとき（情報が漏洩する可能性が増すので）、データに対するアクセス権を細かく設定するときなどに利用される。

【０１１９】図１２は、パースブロックの一例を示している。パースブロックのデータ部Ｄ００乃至Ｄ１ｆの領域Ｄ００乃至Ｄ０７は、パースデータ部として利用される。パースブロックのデータ部Ｄ００乃至Ｄ１ｆの領域Ｄ０８乃至Ｄ０ｆは、エグゼキューションＩＤ（Execution ID）を記憶している。なお、パースブロックのデータ部の領域Ｄ１０乃至Ｄ１ｆは、ユーザデータ部として利用されるが、読み出し専用に設定される。

【０１２０】パースデータ部は、所定のデータを記憶している。エグゼキューションＩＤは、パースブロックに対する加算命令または減算命令が実行されるときに参照され、その加算命令または減算命令に含まれているエグゼキューションＩＤと比較される。

【０１２１】一方、シーケンシャルアクセス領域のユーザブロックのアトリビュート部は、図１３に示すように、ラップラウンド番号（ビットb_f乃至b_0）を記憶している。シーケンシャルアクセス領域には、領域の先頭の物理ブロックから順番に（シーケンシャルに）データが記憶されていき、領域の最後の物理ブロックまでデータが記憶されると、再び、領域の先頭の物理ブロックから順番にデータが記憶されていく（上書きされていく）。ラップラウンド番号は、その順番を記憶している。

【０１２２】従って、ラップラウンド番号は、シーケンシャルアクセス領域のユーザブロックにアクセスするときに利用されるとともに、シーケンシャルアクセス領域にデータを記憶する場合、順次参照される。そして、それまでの最後尾のラップラウンド番号を有する物理ブロックの次の物理ブロックに、データが記憶される。このとき、データが記憶された物理ブロックのラップラウンド番号は、それまでの最後尾のラップラウンド番号に１を加算した数に設定される。

【０１２３】なお、例えば前回の書込のときに書込の途中で障害が発生して、最後尾のラップラウンド番号を有する物理ブロックにパリティエラー（物理的なメモリコラプション）が生じている場合、新たなデータは、その物理ブロックに記憶される。また、最後尾のラップラウンド番号を有する物理ブロックがシーケンシャルアクセス領域の終わりの物理ブロックの場合、新たなデータは、シーケンシャルアクセス領域の先頭の物理ブロックに記憶される。

【０１２４】以上のように、ＥＥＰＲＯＭ６６は、各プロバイダに適宜利用される。

【０１２５】次に、図１４のフローチャートおよび図１５のタイミングチャートを参照して、ＩＣカード２およびＲ／Ｗ１の動作について説明する。

【０１２６】最初にステップＳ１において、ＩＣカード２に登録されているプロバイダに対応しているＲ／Ｗ１は、アンテナ２７から所定の電磁波を放射して、アンテナ２７の負荷状態を監視し、ＩＣカード２が接近し、負荷状態の変化が検出されるまで待機する。なお、ステップＳ１においては、Ｒ／Ｗ１は、所定の短いパターンのデータでＡＳＫ変調した電磁波を放射して、ＩＣカード２への呼びかけを、ＩＣカード２からの応答が一定時間内において得られるまで繰り返すようにしてもよい。

【０１２７】Ｒ／Ｗ１がステップＳ１においてＩＣカード２の接近を検出した場合（図１５の時刻t_0）、ステップＳ２に進み、Ｒ／Ｗ１のＳＰＵ３２は、図１６（ａ）に示すような所定の周波数（例えば、データのクロック周波数の２倍の周波数）の矩形波を搬送波として、ＩＣカード２に送信するデータ（ＩＣカード２に実行させる処理に対応するコマンド）（例えば、図１６（ｂ）に示すデータ）で、ＢＰＳＫ変調を行い、生成した変調波（ＢＰＳＫ変調信号）（図１６（ｃ））を変調回路２３に出力する。

【０１２８】なお、ＢＰＳＫ変調時においては、差動変換を利用して、図１６（ｃ）に示すように、値が０のデータが現れた場合、直前のＢＰＳＫ変調信号（「１」「０」または「０」「１」）と同じものをＢＰＳＫ変調信号とし、値が１のデータが現れた場合、直前のＢＰＳＫ変調信号の位相を反転させたもの（「１」を「０」に反転させ、「０」を「１」に反転させたもの）をＢＰＳＫ変調信号としている。

【０１２９】このように差動変換を利用して、変調波の位相の変化でデータを保持することにより、ＢＰＳＫ変調信号が反転した場合も、元のデータに復調されるので、復調するとき変調波の極性を配慮する必要が無くなる。

【０１３０】そして、変調回路２３は、そのＢＰＳＫ変調信号で、所定の搬送波を１未満（例えば０．１）の変調度（＝データ信号の最大振幅／搬送波の最大振幅）でＡＳＫ変調させ、生成された変調波（ＡＳＫ変調波）を、アンテナ２７を介してＩＣカード２に送信する（図１５の時刻t_0乃至時刻t_1の間）。

【０１３１】なお、送信を行わないとき、変調回路２３は、デジタル信号の２つのレベル（ハイレベルとローレベル）のうちのハイレベルで変調波を生成するようになされている。

【０１３２】次にステップＳ３において、ＩＣカード２は、アンテナ５３およびコンデンサ５２で、Ｒ／Ｗ１のアンテナ２７が放射した電磁波の一部を電気信号に変換し、その電気信号（変調波）を、ＩＣ５１のＲＦインタ

フェース部６１に出力する。そして、ＲＦインタフェース部６１のＡＳＫ復調部８１は、その変調波を整流および平滑し（即ち、包絡線検波し）、生成された信号を電圧レギュレータ８２に供給するとともに、生成された信号の直流成分を抑制してデータ信号を抽出し、そのデータ信号をＢＰＳＫ復調回路６２およびＰＬＬ部６３に出力する。

【０１３３】電圧レギュレータ８２は、ＡＳＫ復調部８１より供給された信号を安定化し、直流電力を生成し、各回路に供給する。

【０１３４】なお、このとき、アンテナ５３の端子電圧V_0は、例えば次のようになる。

$V_0 = V_{10}\,(1 + k \times Vs\,(t))\cos(\omega t)$

【０１３５】ここで、V_{10}は、搬送波成分の振幅を、ｋは変調度を、Ｖｓ（ｔ）は信号成分を、それぞれ示している。

【０１３６】また、ＡＳＫ復調部８１による整流後の電圧V_1におけるローレベルの値V_{LR}は、例えば次のようになる。

$V_{LR} = V_{10}\,(1 + k \times (-1)) - Vf$

【０１３７】ここで、Ｖｆは、整流回路のダイオードＤにおける電圧降下を示している。通常Ｖｆは０．７ボルト程度である。

【０１３８】そして、電圧レギュレータ８２は、ＡＳＫ復調部８１により整流および平滑された信号を安定化し、直流電力として、演算部６４を始めとする各回路に供給する。なお、変調波の変調度ｋは１未満であるので、整流後の電圧変動（ハイレベルとローレベルの差）が小さい。従って、電圧レギュレータ８２は、直流電力を容易に生成することができる。

【０１３９】例えば、変調度ｋが５％の変調波を、V_{10}が３ボルト以上になるように受信した場合、整流後のローレベル電圧V_{LR}は、２．１５（＝３×（１－０．０５）－０．７）ボルト以上となり、電圧レギュレータ８２は、電源として充分な電圧を各回路に供給することができるとともに、整流後の電圧V_1の交流成分（データ成分）の振幅$2 \times k \times V_{10}$（Peak-to-Peak値）は、０．３（＝２×０．０５×３）ボルト以上になり、ＡＳＫ復調部８１は、十分高いＳ／Ｎ比でデータの復調を行うことができる。

【０１４０】このように、変調度ｋが１未満のＡＳＫ変調波を利用することにより、エラーレートの低い（Ｓ／Ｎ比の高い状態で）通信を行うとともに、電源として充分な直流電圧がＩＣカード２に供給される。

【０１４１】そして、ＢＰＳＫ復調回路６２は、ＰＬＬ部６３より供給されるクロック信号に従って、ＡＳＫ復調部８１からのデータ信号（ＢＰＳＫ変調信号）を復調し、復調したデータを演算部６４に出力する。

【０１４２】次に、ステップＳ４において、演算部６４は、ＢＰＳＫ復調回路６２より供給されたデータが暗号化されている場合は、暗号／復号部９２で復号化した後、そのデータ（コマンド）をシーケンサ９１に供給し、そのコマンドに対応する処理を行う（図１５の時刻t_1乃至時刻t_2の間）。なお、この期間、即ちＩＣカード２からの返答を受信するまでの間、Ｒ／Ｗ１は、値が１のデータを送信したまま待機している。従って、この期間においては、ＩＣカード２は、最大振幅が一定である変調波を受信している。

【０１４３】次に、ステップＳ５において、演算部６４のシーケンサ９１は、処理結果などのデータ（Ｒ／Ｗ１に送信するデータ）を、ＢＰＳＫ変調回路６８に出力する。ＢＰＳＫ変調回路６８は、Ｒ／Ｗ１のＳＰＵ３２と同様に、そのデータをＢＰＳＫ変調した後、ＲＦインタフェース部６１のＡＳＫ変調部８４に出力する。

【０１４４】そして、ＡＳＫ変調部８４は、アンテナ５３の両端に接続される負荷を、スイッチング素子を利用してデータに応じて変動させることにより、受信している変調波（ＩＣカード２の送信時においては、変調波の最大振幅は一定になっている）を、送信するデータに応じてＡＳＫ変調させ、それに応じてＲ／Ｗ１のアンテナ２７の端子電圧を変動させて、そのデータをＲ／Ｗ１に送信する（図１５の時刻t_2乃至時刻t_3の間）。

【０１４５】ステップＳ６において、Ｒ／Ｗ１の変調回路２３は、ＩＣカード２からのデータの受信時においても、値が１（ハイレベル）のデータの送信を継続している。そして、復調回路２５は、ＩＣカード２のアンテナ２７と電磁気的に結合しているアンテナ２７の端子電圧の微小な変動（例えば、数十マイクロボルト）から、ＩＣカード２により送信されてきたデータを検出する。

【０１４６】そして、復調回路２５は、検出した信号（ＡＳＫ変調波）を高利得の増幅器で増幅した後、復調し、生成されたデジタルデータをＳＰＵ３２に出力する。

【０１４７】そして、ステップＳ７において、Ｒ／Ｗ１のＳＰＵ３２は、そのデータ（ＢＰＳＫ変調信号）を復調した後、ＤＰＵ３１に出力し、ＤＰＵ３１は、そのデータを処理する（図１５の時刻t_3乃至時刻t_4の間）。

【０１４８】さらに、ステップＳ８において、Ｒ／Ｗ１のＤＰＵ３１は、処理結果に応じて、通信を終了するか否かを判断し、再度、通信を行うと判断した場合、ステップＳ２に戻り、ステップＳ２乃至ステップＳ７で、次のデータ（コマンド）の通信を行う（図１５の時刻t_4乃至時刻t_8）。一方、通信を終了すると判断した場合、Ｒ／Ｗ１は、ＩＣカード２との通信を終了する。

【０１４９】以上のように、Ｒ／Ｗ１は、変調度ｋが１未満であるＡＳＫ変調を利用して、ＩＣカード２に所定のコマンドを送信し、ＩＣカード２は、そのコマンドを受け取り、そのコマンドに対応する処理を行って、その処理の結果に対応するデータを、Ｒ／Ｗ１に返送する。

【０１５０】次に、上述のステップＳ４におけるＩＣカ

ード２による処理の例として、ＥＥＰＲＯＭ６６に対してデータの書込を行うときの動作について、図１７乃至図２１のフローチャートを参照して説明する。

【０１５１】最初に、図１７乃至図１９のフローチャートを参照して、ＥＥＰＲＯＭ６６のランダムアクセス領域にデータを書き込むときの動作について説明する。

【０１５２】ステップＳ２１において、シーケンサ９１は、データを書き込む物理ブロックがリード／ライトブロック（パースブロックは含まない）である（図８に示すように、BN_0から順番に、B_{RW}個までのブロックはリード／ライトブロックとされる）か否かを判断し、リード／ライトブロックであると判断した場合、ステップＳ２２に進む。

【０１５３】シーケンサ９１は、Ｒ／Ｗ１のプロバイダコードを有するプロバイダ領域定義ブロックのパースブロックパーミッション（図９）を参照し、パースブロックを使用している（b_3＝１）か否かを判断し、パースブロックを使用していない場合（b_3＝０の場合）、ステップＳ２３（図１８）に進む。

【０１５４】一方、ステップＳ２２においてパースブロックを使用していると判断した場合、シーケンサ９１は、ステップＳ２４において、記憶する（書き込む）データの論理ブロック番号が００Ｈであるか否か、即ち、データを書き込むリード／ライトブロックがパースブロックと重なっているか否かを判断し、データを書き込むリード／ライトブロックがパースブロックと重なっていないと判断した場合、ステップＳ２３に進む。

【０１５５】データを書き込むリード／ライトブロックがパースブロックと重なっていると判断した場合、シーケンサ９１は、ステップＳ２５において、エラー処理を行った後、処理を終了する。

【０１５６】また、ステップＳ２１においてデータを書き込む物理ブロックがリード／ライトブロックではないと判断した場合、ステップＳ２６に進み、シーケンサ９１は、データを書き込む物理ブロックがパースブロックであるか否かを判断し、パースブロックであると判断した場合、ステップＳ２７に進む。

【０１５７】データを書き込む物理ブロックがパースブロックではないと判断した場合、シーケンサ９１は、ステップＳ２８において、エラー処理を行った後、処理を終了する。

【０１５８】ステップＳ２７において、シーケンサ９１は、ランダムアクセス領域において、パースブロック（論理ブロック番号が００Ｈの物理ブロック）を探し、パースブロックを発見した場合、ステップＳ２９に進む。

【０１５９】ステップＳ２７でパースブロックが発見されなかった場合、パースブロックに対する書込を行うことができないので、シーケンサ９１は、ステップＳ３０において、エラー処理を行った後、処理を終了する。

【０１６０】次に、ステップＳ２９において、シーケンサ９１は、そのパースブロックに対する命令（コマンド）が加算命令であるか否かを判断し、加算命令であると判断した場合、ステップＳ３１に進み、プロバイダ領域定義ブロックのパースブロックパーミッションを参照して、加算命令が許可されているか（b_2＝１）否かを判断する。

【０１６１】そして、ステップＳ３１で、シーケンサ９１が、パースブロックに対する加算命令が許可されていると判断した場合、ステップＳ２３に進む。

【０１６２】一方、ステップＳ３１で、パースブロックに対する加算命令が許可されていないと判断した場合（b_2＝０の場合）、シーケンサ９１は、加算命令を実行せずに、ステップＳ３２において、エラー処理を行った後、処理を終了する。

【０１６３】また、ステップＳ２９において、パースブロックに対する命令が加算命令ではないと判断した場合、ステップＳ３３に進み、シーケンサ９１は、そのパースブロックに対する命令が減算命令であるか否かを判断し、減算命令であると判断した場合、ステップＳ３４に進む。

【０１６４】そして、ステップＳ３４において、シーケンサ９１は、プロバイダ領域定義ブロックのパースブロックパーミッションを参照して、減算命令が許可されている（b_1＝１）か否かを判断し、パースブロックに対する減算命令が許可されていると判断した場合、ステップＳ２３に進む。

【０１６５】一方、ステップＳ３４で、パースブロックに対する減算命令が許可されていないと判断した場合（b_1＝０の場合）、シーケンサ９１は、減算命令を実行せずに、ステップＳ３５において、エラー処理を行った後、処理を終了する。

【０１６６】また、ステップＳ３３において、パースブロックに対する命令が減算命令ではないと判断した場合、シーケンサ９１は、ステップＳ３６において、エラー処理を行った後、処理を終了する。

【０１６７】次に、図１８のステップＳ２３において、シーケンサ９１は、ランダムアクセス領域の物理ブロックを検索して、書込を行うデータの論理ブロック番号と同一の論理ブロック番号を有する物理ブロックを探す。

【０１６８】そして、ステップＳ３７において、シーケンサ９１は、ステップＳ２３で発見した物理ブロックの数が２個であるか否かを判断する。すなわち、このシステムにおいては、各論理ブロックについて、少なくとも、前回のデータと、前々回のデータを記憶するようにする。そして、さらに新たなデータを記憶するときは、前々回のデータの上に新たなデータを記憶する（他の論理ブロック番号の前々回のデータの上に記憶される場合もある）。同一の論理ブロック番号の物理ブロックが２個存在する場合、ステップＳ３８に進み、その２つの物

理ブロックにおけるインクリメンタルカウンタの値（００，０１，１０，１１のいずれか）を読み出し、比較する。

【０１６９】そして、インクリメンタルカウンタの値が大きい物理ブロックを、新しいデータが記憶されている物理ブロック（新しい物理ブロック）とし、インクリメンタルカウンタの値が小さい物理ブロックを、古いデータが記憶されている物理ブロック（古い物理ブロック）とする。

【０１７０】ただし、２つのインクリメンタルカウンタの値が００と１１である場合は、インクリメンタルカウンタの値が００である物理ブロックを、新しい物理ブロックとし、インクリメンタルカウンタの値が１１である物理ブロックを、古い物理ブロックとする。

【０１７１】ステップＳ３９において、シーケンサ９１は、２つの物理ブロックのうち、新しい物理ブロックの番号（物理ブロック番号）を、変数Ｙとして、ＲＡＭ６７に記憶し、古い物理ブロックの番号を、変数Ｗ（ライトブロックとして利用される物理ブロックの番号）として、ＲＡＭ６７に記憶させる。

【０１７２】このように、シーケンサ９１が、変数Ｙと変数Ｗを記憶させた後、ステップＳ４９に進む。

【０１７３】一方、ステップＳ３７において、ステップＳ２３で発見した物理ブロックの数が２個ではないと判断した場合、ステップＳ４０に進み、シーケンサ９１は、ステップＳ２３で発見した物理ブロックの数が１個であるか否かを判断する。そして、１個であると判断した場合、ステップＳ４１に進む。

【０１７４】ステップＳ４０において、シーケンサ９１が、ステップＳ２３で発見した物理ブロックの数が１個ではないと判断した場合、ステップＳ４２において、エラー処理を行った後、処理を終了する。

【０１７５】同一の論理ブロックが１個しか存在しないということは、何等かの理由により、前々回のデータが存在しないことになる。そこで、この場合は、他の論理ブロック番号の物理ブロックで、前回と前々回のデータを有する物理ブロック（すなわち、同一の論理ブロック番号を有する物理ブロックの数が２個である物理ブロック）を検索し、そのうちの前々回の物理ブロックをライトブロックとして利用する。このため、ステップＳ４１において、シーケンサ９１は、発見した物理ブロック（１個）の番号を、変数Ｙとして、ＲＡＭ６７に記憶させた後、ステップＳ４３に進む。

【０１７６】ステップＳ４３において、シーケンサ９１は、ランダムアクセス領域の物理ブロックを検索して、所定の（任意の）同一の論理ブロック番号（いま書込対象としている論理ブロック番号とは無関係の論理ブロック番号）を有する２個の物理ブロックを探す。

【０１７７】なお、物理ブロックを検索するときは、論理ブロック番号００Ｈから順次検索していくので、頻繁に書込処理を行うデータの論理ブロック番号を、より小さい番号すると、検索時間を短くすることができる。

【０１７８】そして、ステップＳ４４において、シーケンサ９１は、論理ブロック番号が同一である２個の物理ブロックがステップＳ４３で発見されたか否かを判断し、発見されたと判断した場合、ステップＳ４５に進み、発見された２個の物理ブロックのインクリメンタルカウンタを参照し、２個の物理ブロックのうち、古い方の物理ブロックの番号を、変数Ｗ（ライトブロックの番号）として、ＲＡＭ６７に記憶させた後、ステップＳ４９（図１９）に進む。

【０１７９】一方、ステップＳ４４において、ステップＳ４３で２個の物理ブロックが発見されなかったと判断した場合、ステップＳ４６に進み、シーケンサ９１は、ランダムアクセス領域の各物理ブロックのパリティを順次計算して、各物理ブロックのパリティ部に記憶されている値と比較し、パリティエラーを起こしている物理ブロックを探す。

【０１８０】そして、パリティエラーを起こしている物理ブロックがあるか否かを判断し、パリティエラーを起こしている物理ブロックがあると判断した場合、ステップＳ４７に進み、シーケンサ９１は、その物理ブロックの番号を、変数Ｗ（ライトブロックの番号）として、ＲＡＭ６７に記憶させた後、ステップＳ４９に進む。

【０１８１】ステップＳ４６において、パリティエラーを起こしている物理ブロックがないと判断した場合、シーケンサ９１は、ステップＳ４８において、エラー処理を行った後、処理を終了する。

【０１８２】次に、図１９のステップＳ４９において、シーケンサ９１は、データを書き込む物理ブロックがパースブロック（論理ブロック番号が００Ｈである物理ブロック）であるか否かを判断し、パースブロックであると判断した場合、ステップＳ５０に進み、パースブロックに対して行われる命令のエグゼキューションＩＤが、ステップＳ３９またはステップＳ４１で変数Ｙとして記憶された番号の物理ブロックのエグゼキューションＩＤ（図１２）と同一であるか否かを判断し、同一であると判断した場合、この命令は既に処理されていると判断し、処理を終了する。

【０１８３】このようにエグゼキューションＩＤを利用することにより、Ｒ／Ｗ１が同じコマンドをリトライした場合において、そのコマンドが既に処理されているときは、ＩＣカード２は、そのコマンドの処理を行わないので、同じコマンドが２度処理されることはない。

【０１８４】ステップＳ５０において、パースブロックに対して行われる命令のエグゼキューションＩＤが、変数Ｙとして記憶された番号の物理ブロックのエグゼキューションＩＤと同一ではないと判断した場合、シーケンサ９１は、ステップＳ５１において、パースブロックに対して行われる命令が加算命令であるか否かを判断し、

加算命令である場合、ステップＳ５２に進む。

【０１８５】ステップＳ５２において、シーケンサ９１は、変数Ｙの番号の物理ブロックのパースデータを読み出し、そのパースデータと、パースブロックに対して行われる命令に含まれているデータの和を計算し、その和を新規ブロックデータにおけるパースデータ（新規パースデータ）とする。このように処理を行った後、ステップＳ５４に進む。なお、このとき、変数Ｙの番号の物理ブロックのエグゼキューションＩＤを新規ブロックデータのエグゼキューションＩＤとする。これにより２重の処理を防止する。

【０１８６】一方、ステップＳ５１において、パースブロックに対して行われる命令が加算命令ではない（即ち、減算命令である）と判断した場合、ステップＳ５３に進み、シーケンサ９１は、変数Ｙの番号の物理ブロックのパースデータを読み出し、そのパースデータと、パースブロックに対して行われる命令に含まれているデータの差を計算し、その差を新規ブロックデータにおけるパースデータ（新規パースデータ）とする。このように処理を行った後、ステップＳ５４に進む。なお、このとき、変数Ｙの番号の物理ブロックのエグゼキューションＩＤを新規ブロックデータのエグゼキューションＩＤとする。これにより２重の処理を防止する。

【０１８７】また、ステップＳ４９において、シーケンサ９１は、データを書き込む物理ブロックがパースブロックではない（即ち、リード／ライトブロックである）と判断した場合、ステップＳ５４に進む。

【０１８８】そして、ステップＳ５４において、シーケンサ９１は、変数Ｙの番号の物理ブロックのインクリメンタルカウンタの値に１を加算した数を、新規ブロックデータのインクリメンタルカウンタの値とする。ただし、変数Ｙの番号の物理ブロックのインクリメンタルカウンタの値が１１である場合、シーケンサ９１は、新規ブロックデータのインクリメンタルカウンタの値を００とする。

【０１８９】次に、ステップＳ５５において、シーケンサ９１は、パリティ演算部９３に、新たに書き込むデータ、インクリメンタルカウンタおよび論理ブロック番号のパリティを計算させ、そのパリティの値を、新規ブロックデータのパリティ部の値とする。

【０１９０】そして、ステップＳ５６において、シーケンサ９１は、ステップＳ３９、ステップＳ４５、または、ステップＳ４７のいずれかで記憶された変数Ｗの番号の物理ブロック（ライトバッファ）に、新規ブロックデータ（新たに記憶するデータ（パースブロックの場合、パースデータとエグゼキューションＩＤ）、その論理ブロック番号、インクリメンタルカウンタ、および、これらのパリティ）を記憶させる。

【０１９１】以上のように、論理ブロック番号と、インクリメンタルカウンタを利用して、データを記憶する物理ブロック（ライトバッファ）を選択することにより、データの書込の最中に、障害が発生した場合においても、そのデータの論理ブロック番号と同一の論理ブロック番号のデータがメモリに残されているので、論理的には、メモリコラプションが発生することはない。

【０１９２】上記実施の形態では、ランダムアクセス領域の同一の論理ブロックのうち、新しいデータが記録されているブロックを判別するためにインクリメンタルカウンタを用いたが、例えば、記録時の絶対時刻（日付と時刻、あるいはカウンタの値）をランダムアクセス領域に、例えば４バイトの領域を確保して、そこに記録させることによって、新しいデータが記録されているブロックを判別することも可能である。

【０１９３】次に、図２０および図２１のフローチャートを参照して、ＥＥＰＲＯＭ６６のシーケンシャルアクセス領域にデータを書き込むときの動作について説明する。

【０１９４】ステップＳ６１において、シーケンサ９１は、シーケンシャルアクセス領域の先頭の物理ブロックの番号を、変数Ｚとして、ＲＡＭ６７に記憶させる。

【０１９５】次に、ステップＳ６２において、シーケンサ９１は、物理ブロック番号がＺである物理ブロックのラップラウンド番号を読み出し、変数Ａとして、ＲＡＭ６７に記憶させるとともに、物理ブロック番号がＺ＋１である物理ブロックのラップラウンド番号を読み出し、変数Ｂとして、ＲＡＭ６７に記憶させる。

【０１９６】そして、ステップＳ６３において、シーケンサ９１は、変数Ａの値と変数Ｂの値の差（Ａ－Ｂ）が１であるか否かを判断し、１ではない場合、物理ブロック番号Ｚの物理ブロックが、最後尾のラップラウンド番号を有するデータを記憶する物理ブロックであると判断し、ステップＳ６６に進む。

【０１９７】変数Ａの値と変数Ｂの値の差（Ａ－Ｂ）が１であると判断した場合、シーケンサ９１は、ステップＳ６４において、物理ブロック番号Ｚが、シーケンシャルアクセス領域の終わりの物理ブロックの番号と同一であるか否かを判断し、同一であると判断した場合、シーケンシャルアクセス領域の終わりの物理ブロックが、最後尾のラップラウンド番号を有するデータを記憶する物理ブロックであると判断し、ステップＳ６６に進む。

【０１９８】ステップＳ６４において、物理ブロック番号Ｚが、シーケンシャルアクセス領域の終わりの物理ブロックの番号と同一ではないと判断した場合、シーケンサ９１は、ステップＳ６５において、ＲＡＭ６７に記憶させた変数Ｚの値を１だけ増加させた後、ステップＳ６２に戻る。そして、ステップＳ６２乃至ステップＳ６５の処理を、変数Ｚの値（検索する物理ブロック番号の値）を変化させながら順次繰り返す。

【０１９９】このようにして、シーケンシャルに記憶されているデータのラップラウンド番号の最後尾を発見す

る。そして、ステップＳ６６において、シーケンサ９１は、変数Ｚの番号（＝ラップラウンド番号の最後尾の物理ブロックの番号）のブロックのパリティチェックを行う。

【０２００】そして、ステップＳ６７において、シーケンサ９１は、その物理ブロックにパリティエラーが生じているか否かを判断し、パリティエラーが生じていると判断した場合、ステップＳ６８に進む。

【０２０１】ステップＳ６８において、シーケンサ９１は、変数Ｚの値が、シーケンシャルアクセス領域の先頭の物理ブロックの番号と同一であるか否かを判断し、同一であると判断した場合、データ（パリティエラーを起こしているものは含まない）の最後尾が、シーケンシャルアクセス領域の終わりの物理ブロックであると判断し、ステップＳ７０において、シーケンシャルアクセス領域の終わりの物理ブロックの番号を、新たな変数Ｙとして、ＲＡＭ６７に記憶させた後、ステップＳ７２（図２１）に進む。

【０２０２】変数Ｚの値が、シーケンシャルアクセス領域の先頭の物理ブロックの番号と同一ではないと判断した場合、ステップＳ７１において、シーケンサ９１は、データの最後尾の物理ブロックの番号を、変数Ｚの値から１を減算して算出し、算出した値（Ｚ－１）を、変数Ｙとして、ＲＡＭ６７に記憶させた後、ステップＳ７２に進む。

【０２０３】一方、ステップＳ６７でパリティエラーが生じていないと判断した場合、ステップＳ６９において、シーケンサ９１は、データの最後尾の物理ブロックの番号（この場合、変数Ｚの値）を、変数Ｙとして、ＲＡＭ６７に記憶させた後、ステップＳ７２に進む。

【０２０４】次に、ステップＳ７２において、シーケンサ９１は、データの最後尾の物理ブロックの番号（変数Ｙの値）と、シーケンシャルアクセス領域の終わりの物理ブロックの番号が同一であるか否かを判断し、同一であると判断した場合、ステップＳ７３に進む。

【０２０５】そして、ステップＳ７３において、シーケンサ９１は、シーケンシャルアクセス領域の先頭の物理ブロックの番号を、新たなデータを書き込む物理ブロックの番号とし、その番号を変数Ｗとして、ＲＡＭ６７に記憶させた後、ステップＳ７５に進む。

【０２０６】ステップＳ７２においてデータの最後尾の物理ブロックの番号（変数Ｙの値）と、シーケンシャルアクセス領域の終わりの物理ブロックの番号が同一ではないと判断した場合、ステップＳ７４において、シーケンサ９１は、変数Ｙの値に１を加算した数を、新たなデータを書き込む物理ブロックの番号とし、その番号を変数Ｗとして、ＲＡＭ６７に記憶させた後、ステップＳ７５に進む。

【０２０７】次にステップＳ７５において、シーケンサ９１は、新たに記憶するデータと、変数Ｙの番号の物理ブロック（最後尾のデータ）が同一であるか否かを判断し、同一である場合、新たに記憶するデータが既に記憶されているので、処理を終了する。

【０２０８】一方、新たに記憶するデータと、変数Ｙの番号の物理ブロック（最後尾のデータ）が同一ではないと判断された場合、ステップＳ７６において、シーケンサ９１は、変数Ｙの番号の物理ブロックのラップラウンド番号を読み出し、その値に１を加算した数を、新たに記憶されるデータ（新規ブロックデータ）のラップラウンド番号とする。

【０２０９】次にステップＳ７７において、シーケンサ９１は、パリティ演算部９３に、記憶するデータおよびラップラウンド番号（新規ブロックデータ）のパリティを演算させ、ステップＳ７８において、番号Ｗの物理ブロックに新規ブロックデータを書き込む。

【０２１０】このように、シーケンシャルに記憶されているデータにおけるラップラウンド番号を順次検索していき、最後尾のデータの次の物理ブロック（または、シーケンシャルアクセス領域の先頭の物理ブロック）に、新たなデータを記憶するので、新たなデータの書込の最中に、障害が発生した場合においても、書き込んでいたデータのラップラウンド番号より小さいラップラウンド番号のデータが残っているので、論理的には、メモリコラプションは発生しない。

【０２１１】以上のように、ＥＥＰＲＯＭ６６は、複数のプロバイダに対して、独立に記憶領域を提供することができるとともに、アトリビュート部の情報を利用して、メモリコラプションの発生を抑制するようになされている。

【０２１２】なお、複数のプロバイダに対して、同一のユーザブロックを割り当てることもできる。その場合、それらのプロバイダ（オーバラッププロバイダ）が登録されているプロバイダ領域定義ブロックのアロケーションテーブルで、同一のユーザブロックを割り当てるようにする。このとき、各プロバイダ毎に、プロバイダ領域定義ブロックのパーティションテーブルを設定することにより、同一のユーザブロックに対して、プロバイダ毎に異なるアクセス権（リード／ライトまたはリードオンリー）を設定することができる。さらに、所定のプロバイダに対してはパースブロックを使用しないように設定し、他のプロバイダに対してはパースブロックを使用するように設定することにより、所定のプロバイダは、他のプロバイダが使用するパースブロックのユーザデータ部（他のプロバイダに対しては読み出し専用）に対して、データの書込を行うことができる。

【０２１３】また、領域定義ブロックの領域Ｄ０ｅ，Ｄ０ｆ（通常、セキュリティキーのバージョン番号が記憶されている領域）の値を、所定の値（例えば、ＦＦＦＦＨ）に設定し、さらに、領域定義ブロックの領域Ｄ１０乃至Ｄ１ｆに、所定のプロバイダのプロバイダコード